

Implementation Guide

Clickstream Analytics on AWS



Clickstream Analytics on AWS: Implementation Guide

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

Solution Overview	1
Features and benefits	2
Use cases	3
Terms and concepts	3
Architecture overview	6
Architecture diagram	6
Solution end-to-end architecture	6
Ingestion module	7
Data processing module	9
Data modeling module	10
Reporting module	12
Analytics Studio	12
AWS Well-Architected pillars	14
Operational excellence	14
Security	14
Reliability	14
Performance efficiency	15
Cost optimization	15
Sustainability	15
Architecture details	16
Solution components	16
Web console	16
Analytics Studio	16
SDKs	16
Data pipeline	17
AWS services in this solution	18
Plan your deployment	21
Cost	21
Ingestion module	22
Data processing & modeling modules	25
Reporting module	12
Logging and monitoring	27
Additional features	27
Security	29

IAM Roles	29
Amazon VPC	29
Security Groups	29
Amazon CloudFront	29
Supported AWS Regions	30
Deployment	36
Prerequisites	36
Deployment in AWS Regions	36
Deployment in AWS China Regions	36
Deployment within Amazon VPC	36
Launch with Cognito User Pool	37
Deployment Overview	37
Step 1: Launch the Stack	37
Step 2: Launch the web console	39
Launch with OpenID Connect (OIDC)	40
Prerequisites	40
Deployment overview	40
Step 1. Create OIDC client	41
Step 2. Launch the Stack	47
Step 3. Update the callback URL of OIDC client	50
Step 4. Setup DNS Resolver	50
Step 4. Launch the Web Console	51
Launch within VPC	51
Prerequisites	51
Deployment Overview	52
Step 1. Create OIDC client	52
Step 2. Launch the stack	52
Step 3. Update the callback URL of OIDC client	54
Step 4. Launch the web console	54
Getting started	56
Step 1: Create a project	56
Prerequisites	56
Steps	56
Step 2: Configure data pipeline	57
Steps	57
Step 3: Integrate SDK	60

Steps	60
Generate sample data	61
Step 4: Analyze data	62
Steps	62
Pipeline management	63
Prerequisites	63
Ingestion	64
Ingestion endpoint	64
Data sink – Kafka	69
Data sink – Kinesis	70
Data sink - S3	71
Data processing	71
Data schema	72
Execution parameters	82
Processing plugin	84
Data modeling	86
Preset data views	86
Reporting	89
Pipeline maintenance	89
Monitoring and Alarms	90
Pipeline modification	90
Pipeline upgrade	91
SDK manual	92
Key features and benefits	92
Android SDK	93
Introduction	93
Integrate the SDK	93
Swift SDK	116
Introduction	116
Integrate the SDK	116
Data format definition	125
Preset events	129
Change log	142
Web SDK	142
Introduction	142
Integrate the SDK	142

Get started	143
Data format definition	150
Preset events	154
Event attributes	157
Change logs	165
References	165
Flutter SDK	166
Introduction	166
Integrate the SDK	166
Data format definition	172
Preset events	176
Event attributes	176
Change log	176
References	176
HTTP API	177
Request endpoint	177
API Specification	177
Request code example	185
Verification data reported successfully	188
User identifier	188
User ID	188
Device ID	189
User Pseudo ID	190
Migrate from third-party SDKs	191
Introduction	191
Step 1: Integrate Clickstream Web SDK	192
Step 2: Encapsulate common data logger methods	192
Step 3: Migrate to common APIs in minutes	194
Summary	195
Analytics Studio	196
Modules	196
Terms and concepts	196
Dashboard	197
Overview	197
Acquisition report	198
Engagement report	206

Activity report	216
Retention report	223
Device report	234
User report	241
Crash report	247
Custom report	250
Exploration	254
Access Exploration	254
How Exploration works	254
How to use Exploration	255
Event Analysis	257
Funnel Analysis	259
Path Analysis	262
Retention Analysis	264
Analyses	267
Access Analyses	267
How it works	268
Data management	269
Access Data Management	269
How it works	269
Metadata dimensions	270
Update event display name and description	272
Customize data dictionary for Event Parameter values	273
Identity Management	274
User management	274
User roles	274
User roles management	275
Upgrade the solution	278
Planning and Preparation	278
Upgrade Process	278
Upgrade web console stack	278
Upgrade the pipeline of project	279
Post-Upgrade Actions	280
Migrate the existing data after upgrading from 1.0.x	280
Frequently Asked Questions	282
General	282

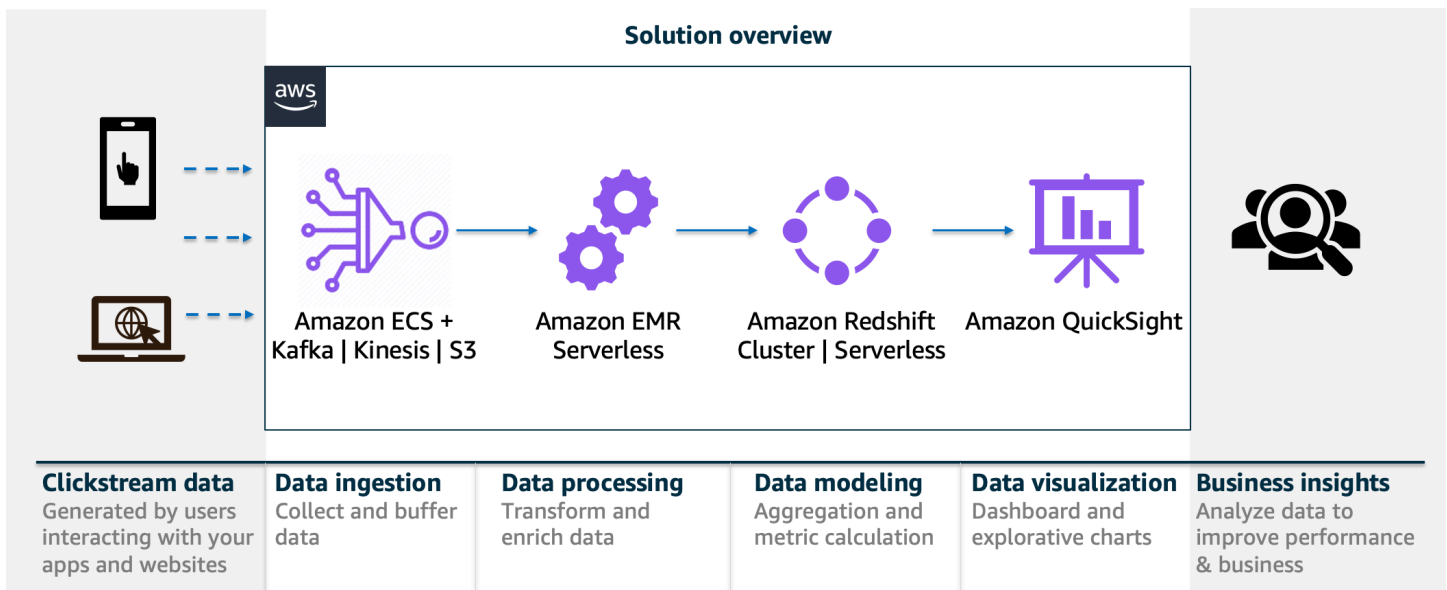
Data pipeline	282
SDK	284
Analytics Studio	284
Pricing	285
Troubleshooting	286
Problem: Deployment failure due to "Invalid Logging Configuration: The CloudWatch Logs Resource Policy size was exceeded"	286
Problem: Cannot delete the CloudFormation stacks created for the Clickstream pipeline	287
Problem: Reporting stack(Clickstream-Reporting-xxx) deployment fail	288
Problem: Clickstream-DataModelingRedshift-xxxxx stack upgrade failed in UPDATE_ROLLBACK_FAILED	288
Problem: Can not sink data to MSK cluster, got "InvalidReplicationFactor (Broker: Invalid replication factor)" log in Ingestion Server	289
Problem: data processing job failure	289
Problem: data loading workflow failure due to meeting the 25,000 events limit in a single execution history	290
Uninstall the solution	291
Step 1. Delete projects	291
Step 2. Delete Clickstream Analytics on AWS stack	291
Additional resources	292
Upload SSL Certificate to IAM	292
Developer guide	293
Source code	293
Contributors	294
Revisions	295
Notices	296

An end-to-end solution to collect, ingest, analyze, and visualize clickstream data inside your web and mobile applications

Publication date: *July 2023* ([last update](#): December 2023)

The Clickstream Analytics on AWS solution allows you to collect, ingest, process and analyze clickstream data from websites and mobile applications to drive your business growth. You can use the solution to spin up an analytics platform that fits your organizational needs, and maintain complete ownership and control over your valuable user behavior data. The solution can be applied to various use cases such as user behavior analysis and marketing analysis to improve website and application's performance.

The solution provides modularized and configurable components of a data pipeline so that you can choose and customize the components to accelerate the building of a Well-Architected data pipeline from weeks to minutes. The purpose-built SDKs and guidance allow you to collect client-side data from different application platforms (for example, Android, iOS, and JavaScript) to AWS. In addition, ready-to-use dashboards and explorative analytics studio enable you to derive actionable business insights easily and quickly.



Use this navigation table to quickly find answers to these questions:

If you want to ...	Read...
Know the cost for running this solution	Cost
Understand the security considerations for this solution	Security
Know which AWS Regions are supported for this solution	Supported AWS Regions
Get started with the solution quickly to build an end-to-end data pipeline, send data into the pipeline, and then view the ready-to-use dashboards	Getting started
Learn the concepts related to pipeline, and how to manage a data pipeline throughout its lifecycle	Pipeline management
Practice how to customize analytics dashboard in an intuitive manner	Analytics studio

This guide is intended for IT architects, developers, DevOps, data analysts, and marketing technology professionals who have practical experience architecting in the AWS Cloud.

Features and benefits

The solution includes the following key features and benefits:

- **Visual data pipeline builder**

You can simply define the data pipeline from a web-based UI console. The solution will take care of the underlying infrastructure creation, required security setup, and data integrations. Each pipeline module is built with various features and designed to be loosely-coupled, making it flexible for you to customize for specific use cases.

- **Purpose-built SDKs**

The SDKs are optimized for collecting data from Android, iOS, and JavaScript platforms, which automatically collect common events (for example, first visit, screen view), support built-in local cache, retry, and verification mechanisms to ensure high completeness of data transmission.

- **Out-of-the-box dashboard**

The solution offers a dozen of built-in visualizations (for example, acquisition, engagement, retention, and user demographic) and exploratory reporting templates (for example, user details, event details), powering various critical business analytics use cases such as user behavior analytics, marketing analytics, and product analytics.

Use cases

Clickstream data play a pivotal role in numerous online business analytics use cases, and the Clickstream Analytics on AWS can be applied to the following:

- **User behavior analytics**

Clickstream data in user behavior analytics provides insights into the sequential and chronological patterns of user interactions on a website or application, helping businesses understand user navigation, preferences, and engagement levels to enhance the overall product experience and drive product innovation.

- **First-party customer data platform (CDP)**

Clickstream data, together with other business data sources (for example, order history, and user profile), allow customers to create a first-party customer data platform that offers a comprehensive view of their users, enabling businesses to personalize customer experiences, optimize customer journeys, and deliver targeted marketing messages.

- **Marketing analytics**

Clickstream data in marketing analytics offers detailed information about users' click paths and interactions with marketing campaigns, enabling businesses to measure campaign effectiveness, optimize marketing strategies, and enhance conversion rates.

Terms and concepts

This section describes key concepts and defines terminology specific to this solution:

Project

A project in this solution is the top-level entity, like a container, that groups your apps and data pipeline for collecting and processing clickstream data. One project contains one data pipeline, and can have one or more apps registered to it.

Data pipeline

A data pipeline is deployed into one AWS region, which means all the underlining resources are created in one AWS region. A data pipeline in this solution contains four modules:

- **Ingestion server:** a web service that provides an endpoint to collect data through HTTP requests, and sink the data in streaming services or S3.
- **Data processing:** a module that transforms raw data to the solution schema and enriches data with additional dimensions.
- **Data modeling:** a module that aggregates data to calculate metrics for business analytics.
- **Reporting:** a module that creates metrics and out-of-the-box visualizations in QuickSight.

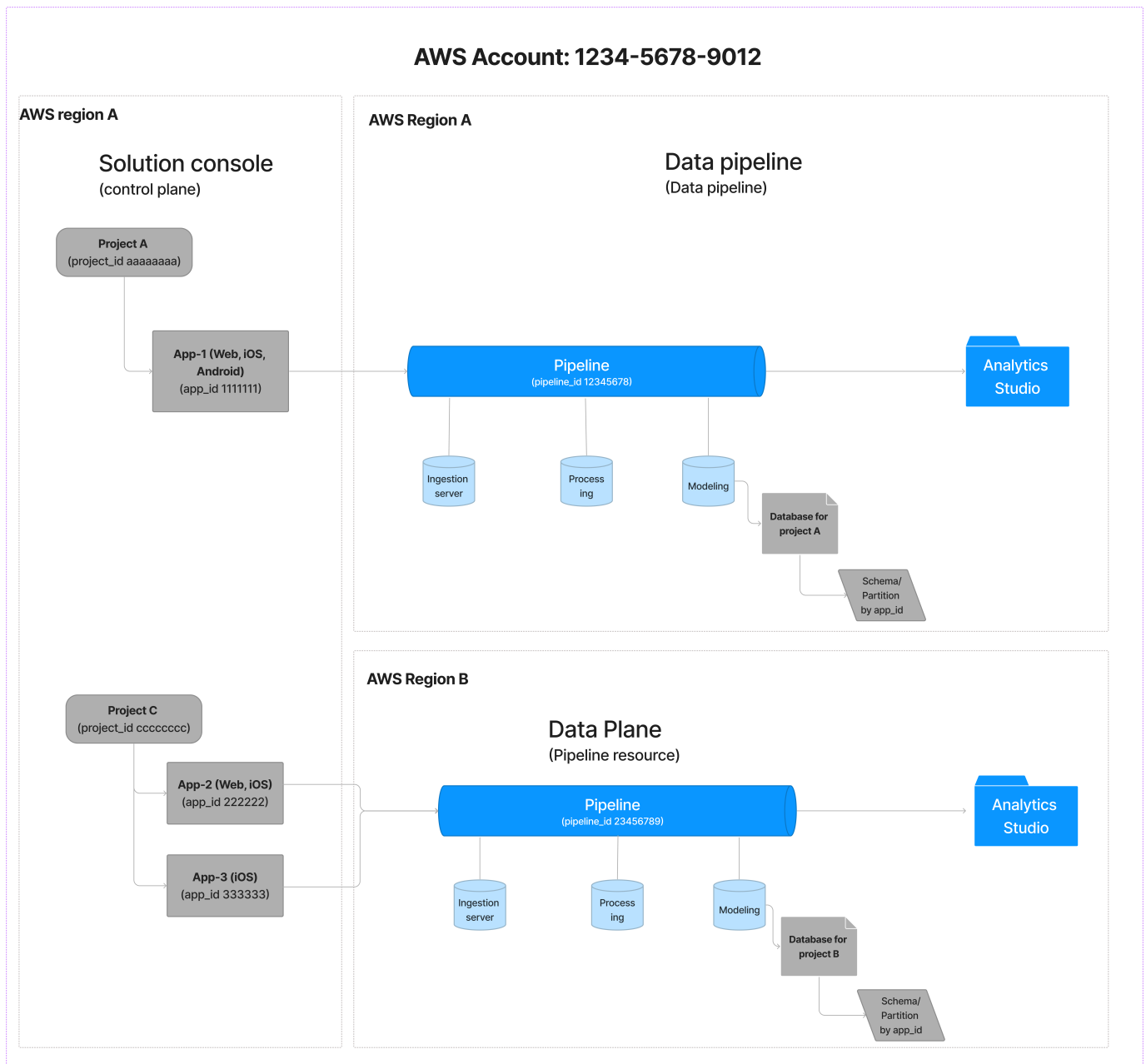
App

An app in this solution can represent an application in your business, which might be built on one or multiple platforms (for example, Android, iOS, and Web).

Analytics Studio

Analytics Studio is a web console for business or data analysts to view dashboards, query, and manage clickstream data.

Below is a diagram to help you better understand those concepts and their relationship with each other in the AWS context.



For a general reference of AWS terms, see the [AWS glossary](#) in the *AWS General Reference*.

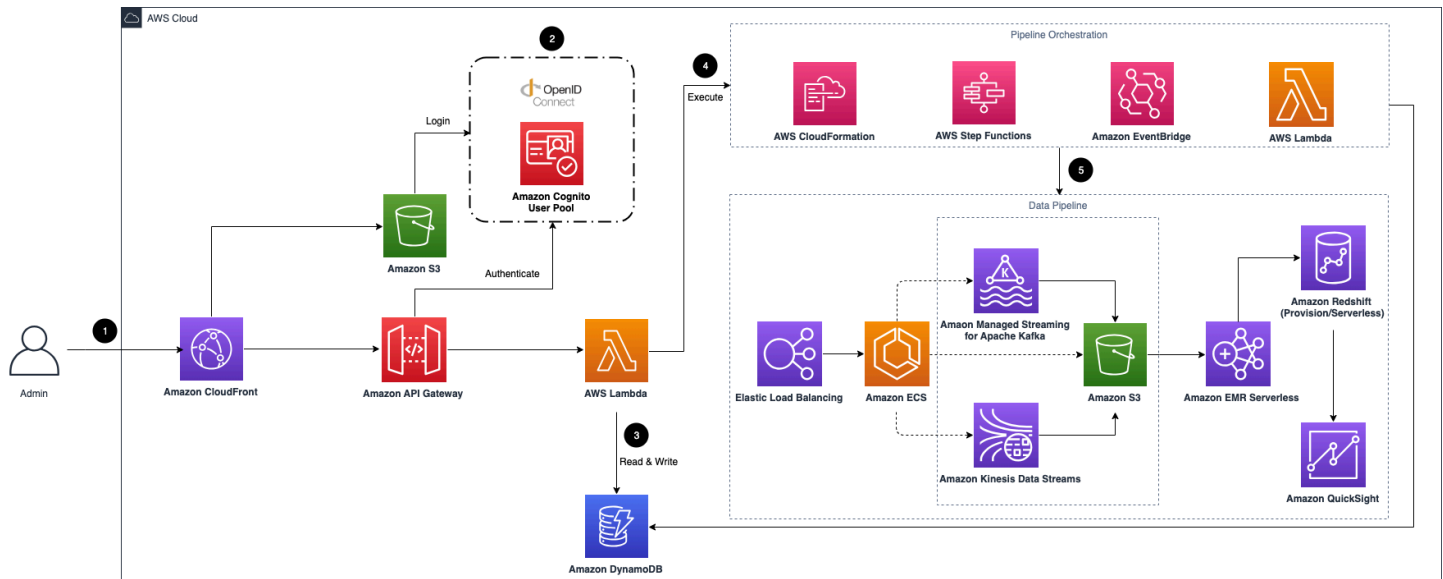
Architecture overview

This section provides a reference implementation architecture diagram for the components deployed with this solution.

Architecture diagram

Solution end-to-end architecture

Deploying this solution with the default parameters builds the following environment in AWS:



Clickstream Analytics on AWS architecture

This solution deploys the AWS CloudFormation template in your AWS Cloud account and completes the following settings.

1. [Amazon CloudFront](#) distributes the frontend web UI assets hosted in the [Amazon S3](#) bucket, and the backend APIs hosted with [Amazon API Gateway](#) and [AWS Lambda](#).
2. The [Amazon Cognito](#) user pool or OpenID Connect (OIDC) is used for authentication.
3. The web UI console uses [Amazon DynamoDB](#) to store persistent data.
4. [AWS Step Functions](#), [AWS CloudFormation](#), AWS Lambda, and [Amazon EventBridge](#) are used for orchestrating the lifecycle management of data pipelines.
5. The data pipeline is provisioned in the region specified by the system operator. It consists of [Application Load Balancer](#), [Amazon ECS](#), [Amazon Managed Streaming for Apache Kafka](#)

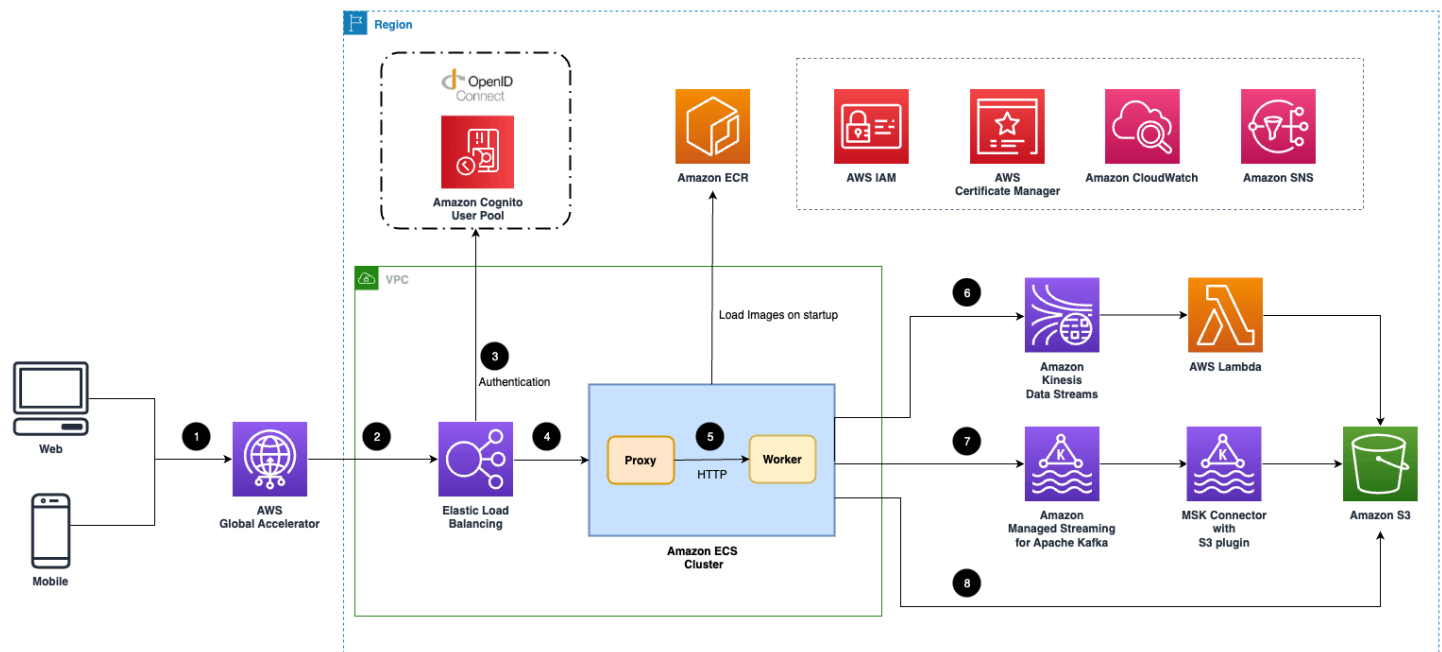
([Amazon MSK](#)), [Amazon Kinesis](#) Data Streams, Amazon S3, [Amazon EMR](#) Serverless, [Amazon Redshift](#), and [Amazon QuickSight](#).

The key functionality of this solution is to build a data pipeline to collect, process, and analyze their clickstream data. The data pipeline consists of four modules:

- Ingestion module
- Data processing module
- Data modeling module
- Reporting module

The following introduces the architecture diagram for each module.

Ingestion module



Ingestion module architecture

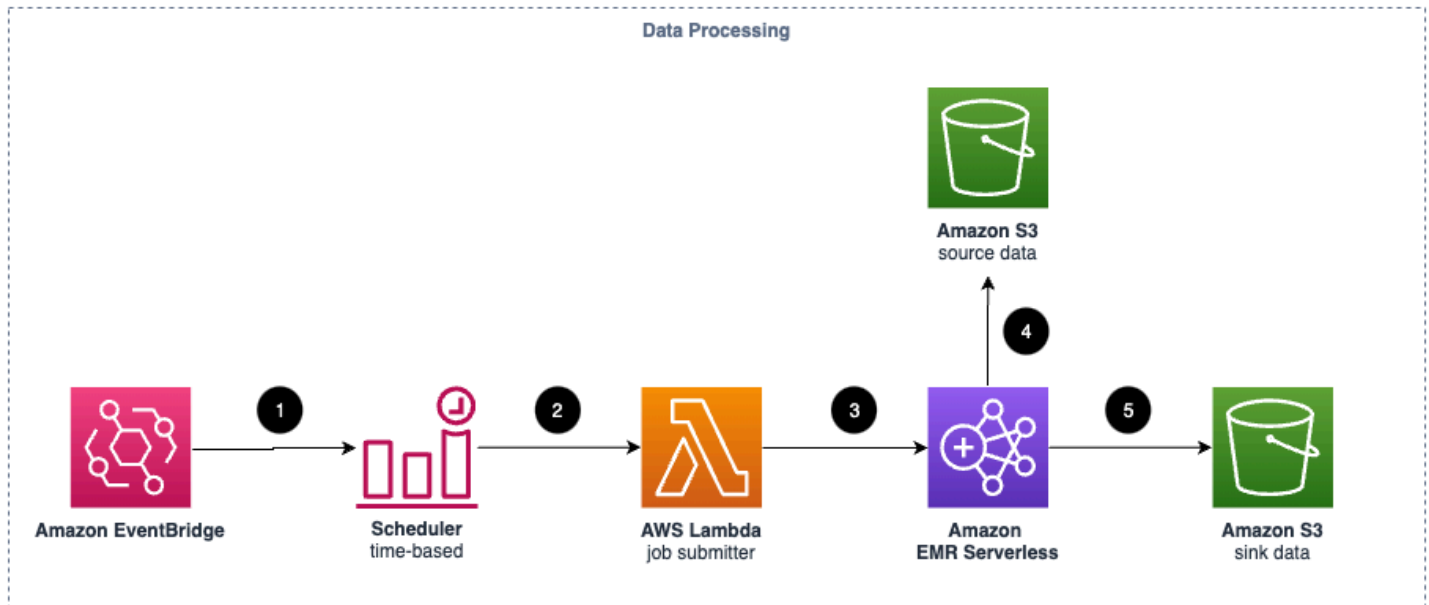
Suppose you create a data pipeline in the solution. This solution deploys the Amazon CloudFormation template in your AWS account and completes the following settings.

Note

The ingestion module supports three types of data sinks. You can only have one type of data sink in a data pipeline.

1. (Optional) The ingestion module creates an AWS global accelerator endpoint to reduce the latency of sending events from your clients (web applications or mobile applications).
2. [Elastic Load Balancing \(ELB\)](#) is used for load balancing ingestion web servers.
3. (Optional) If you enable the authenticating feature, the ALB will communicate with the OIDC provider to authenticate the requests.
4. ALB forwards all authenticated and valid requests to the ingestion servers.
5. Amazon ECS cluster is hosting the ingestion fleet servers. Each server consists of a proxy and a worker service. The proxy is a facade of the HTTP protocol, and the worker will send the events to a data sink based on your choice.
6. If Amazon Kinesis Data Streams is used as a buffer, AWS Lambda consumes the clickstream data in Kinesis Data Streams and then sinks them to Amazon S3 in batches.
7. If Amazon MSK is used as a buffer, MSK Connector is provisioned with an S3 connector plugin that sinks the clickstream data to Amazon S3 in batches.
8. If Amazon S3 is selected as data sink, the ingestion server will buffer a batch of events and sink them to Amazon S3.

Data processing module

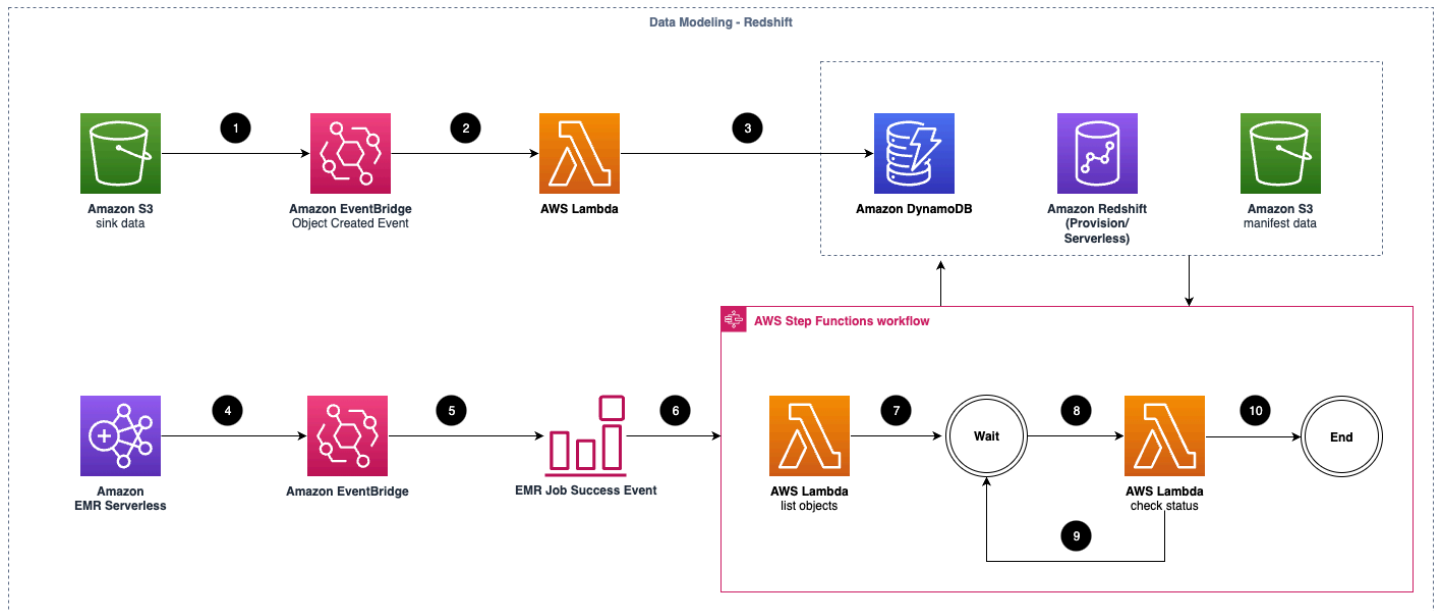


Data processing module architecture

Suppose you create a data pipeline in the solution and enable data processing. This solution deploys the Amazon CloudFormation template in your AWS Cloud account and completes the following settings.

1. Amazon EventBridge is used to trigger the data processing jobs periodically.
2. The configurable time-based scheduler invokes an AWS Lambda function.
3. The Lambda function kicks off an EMR Serverless application based on Spark to process a batch of clickstream events.
4. The EMR Serverless application uses the configurable transformer and enrichment plug-ins to process the clickstream data from the source S3 bucket.
5. After processing the clickstream events, the EMR Serverless application sinks the processed clickstream data to the sink S3 bucket.

Data modeling module

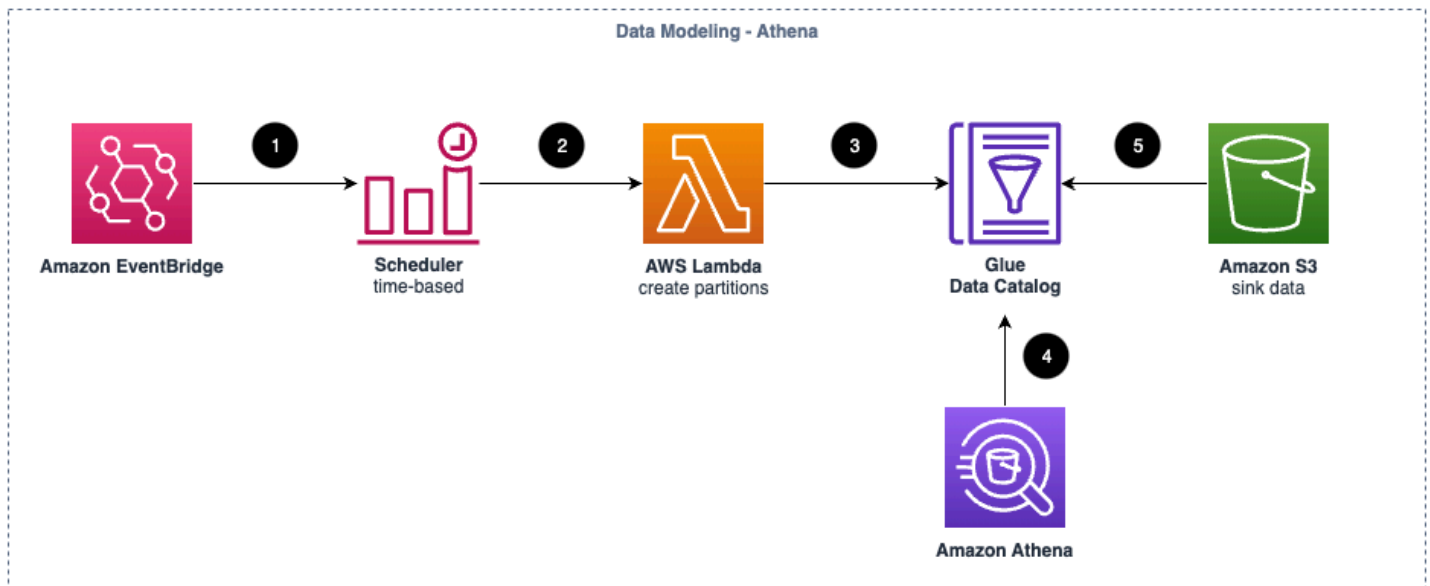


Data modeling in Redshift architecture

Suppose you create a data pipeline in the solution and enable data modeling in Amazon Redshift. This solution deploys the Amazon CloudFormation template in your AWS Cloud account and completes the following settings.

1. After the processed clickstream data is written in the Amazon S3 bucket, the Object Created Event is emitted.
2. An Amazon EventBridge rule is created for the event emitted in step 1, and an AWS Lambda function is invoked when the event happens.
3. The Lambda function persists the source event to be loaded in an Amazon DynamoDB table.
4. When data processing job is done, an event is emitted to Amazon EventBridge.
5. The pre-defined event rule of Amazon EventBridge processes the EMR job success event.
6. The rule invokes the AWS Step Functions workflow.
7. The workflow invokes the `list objects` Lambda function that queries the DynamoDB table to find out the data to be loaded, then creates a manifest file for a batch of event data to optimize the load performance.
8. After a few seconds, the `check status` Lambda function checks the status of the loading job.
9. If the load is still in progress, the `check status` Lambda function waits for a few more seconds.

10 After all objects are loaded, the workflow ends.

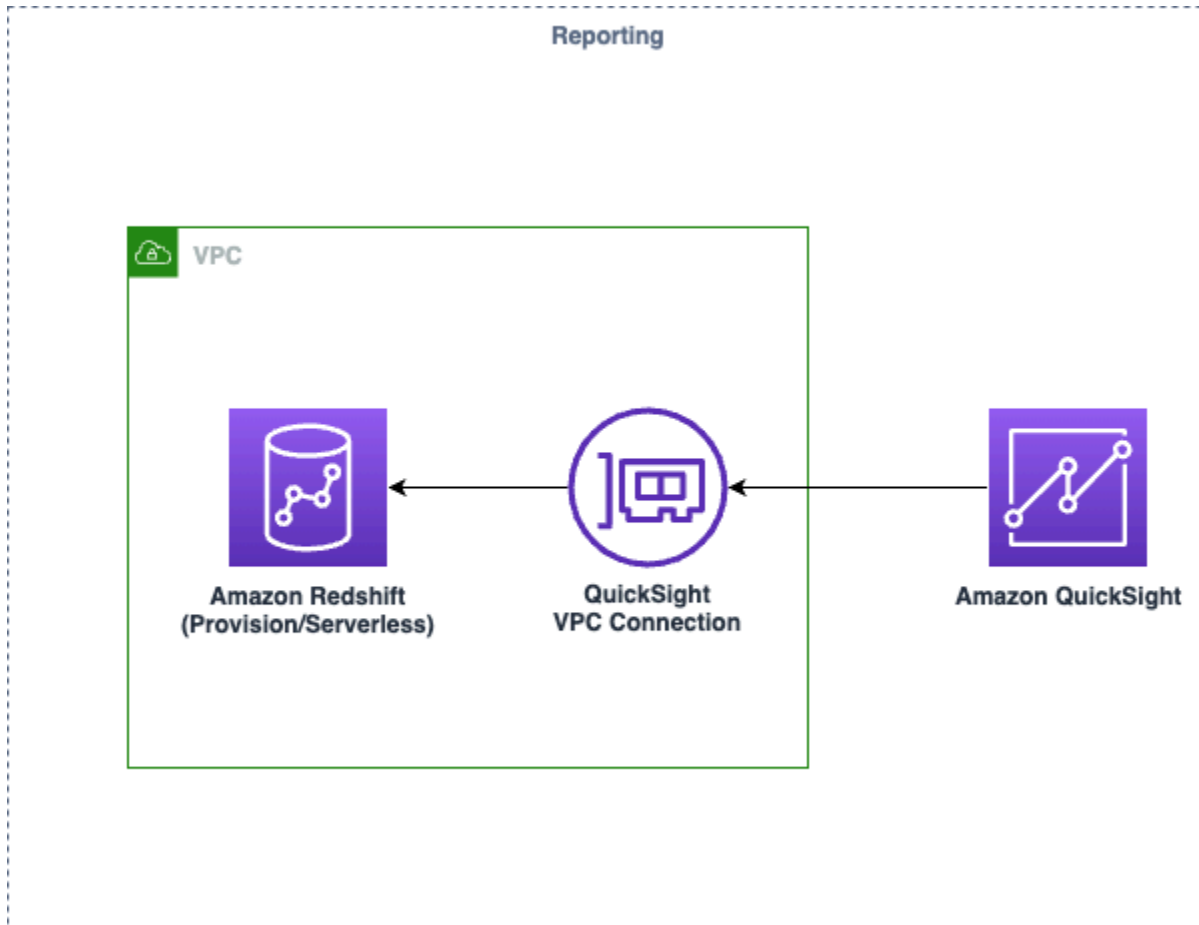


Data modeling in Athena architecture

Suppose you create a data pipeline in the solution and enable data modeling in Amazon Athena. This solution deploys the Amazon CloudFormation template in your AWS Cloud account and completes the following settings.

1. Amazon EventBridge invokes the data load into [Amazon Athena](#) periodically.
2. The configurable time-based scheduler invokes an AWS Lambda function.
3. The AWS Lambda function creates the partitions of the [AWS Glue](#) table for the processed clickstream data.
4. Amazon Athena is used for interactive querying of clickstream events.
5. The processed clickstream data is scanned via the Glue table.

Reporting module



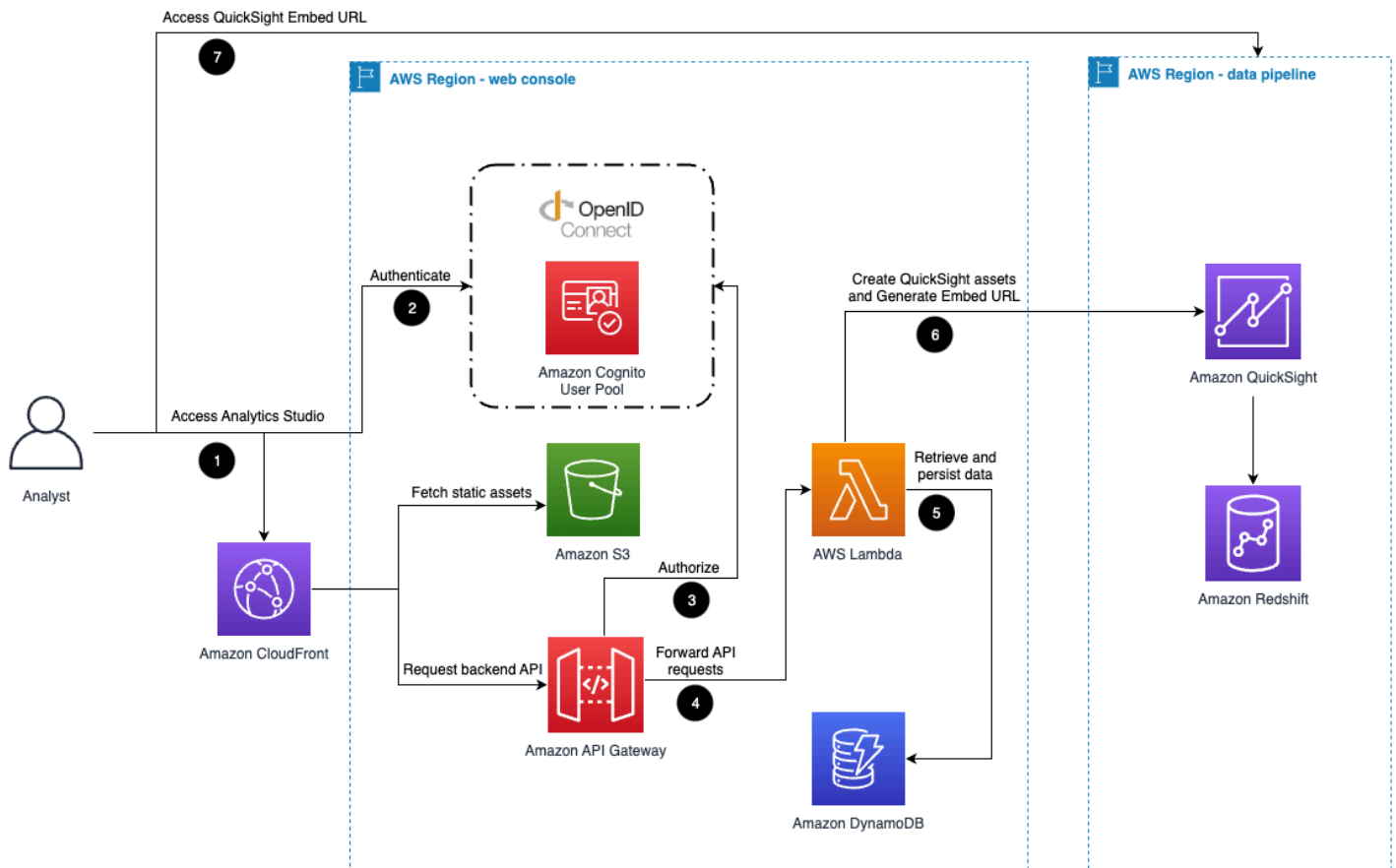
Reporting module architecture

Suppose you create a data pipeline in the solution, enable data modeling in Amazon Redshift, and enable reporting in Amazon QuickSight. This solution deploys the Amazon CloudFormation template in your AWS Cloud account and completes the following settings.

1. VPC connection in Amazon QuickSight is used for securely connecting your Redshift within VPC.
2. The data source, data sets, template, analysis, and dashboard are created in Amazon QuickSight for out-of-the-box analysis and visualization.

Analytics Studio

Analytics Studio is a unified web interface for business analysts or data analysts to view and create dashboards, query and explore clickstream data, and manage metadata.



Analytics Studio architecture

1. When analysts access Analytics Studio, requests are sent to [Amazon CloudFront](#), which distributes the web application.
2. When the analysts log in to Analytics Studio, the requests are redirected to the [Amazon Cognito](#) user pool or OpenID Connect (OIDC) for authentication.
3. [Amazon API Gateway](#) hosts the backend API requests and uses the custom Lambda authorizer to authorize the requests with the public key of OIDC.
4. API Gateway integrates with [AWS Lambda](#) to serve the API requests.
5. The AWS Lambda function uses [Amazon DynamoDB](#) to retrieve and persist the data.
6. When analysts create analyses, the Lambda function requests [Amazon QuickSight](#) to create assets and get the embed URL in the data pipeline Region.
7. The browser of analysts accesses the QuickSight embed URL to view the QuickSight dashboards and visuals.

AWS Well-Architected pillars

This solution was designed with best practices from the [AWS Well-Architected Framework](#) which helps customers design and operate reliable, secure, efficient, and cost-effective workloads in the cloud.

This section describes how the design principles and best practices of the Well-Architected Framework were applied when building this solution.

Operational excellence

This section describes how the principles and best practices of the [operational excellence pillar](#) were applied when designing this solution.

The Clickstream Analytics on AWS solution pushes metrics, logs and traces to Amazon CloudWatch at various stages to provide observability into the infrastructure, Elastic load balancer, Amazon ECS cluster, Lambda functions, EMR serverless application, Step Function workflow and the rest of the solution components. This solution also creates the CloudWatch dashboard for each [data pipeline](#).

Security

This section describes how the principles and best practices of the [security pillar](#) were applied when designing this solution.

- Clickstream Analytics on AWS web console users are authenticated and authorized with Amazon Cognito or OpenID Connect.
- All inter-service communications use AWS IAM roles.
- All roles used by the solution follows least-privilege access. That is, it only contains minimum permissions required so the service can function properly.

Reliability

This section describes how the principles and best practices of the [reliability pillar](#) were applied when designing this solution.

- Using AWS serverless services wherever possible (for example, EMR Serverless, Redshift Serverless, Lambda, Step Functions, Amazon S3, and Amazon SQS) to ensure high availability and recovery from service failure.

- Data ingested by [data pipeline](#) is stored in Amazon S3 and Amazon Redshift, so it persists in multiple Availability Zones (AZs) by default.

Performance efficiency

This section describes how the principles and best practices of the [performance efficiency pillar](#) were applied when designing this solution.

- The ability to launch this solution in any Region that supports AWS services in this solution such as: Amazon S3, Amazon ECS, and Elastic load balancer.
- Using Analytics Serverless architectures removes the need for you to run and maintain physical servers for traditional compute activities.
- Automatically testing and deploying this solution daily. Reviewing this solution by solution architects and subject matter experts for areas to experiment and improve.

Cost optimization

This section describes how the principles and best practices of the [cost optimization pillar](#) were applied when designing this solution.

- The solution uses Autoscaling Group so that the compute costs are only related to how much data is ingested and processed.
- The solution uses serverless services such as Amazon S3, Amazon Kinesis Data Streams, Amazon EMR Serverless and Amazon Redshift Serverless so that customers only get charged for what they use.

Sustainability

This section describes how the principles and best practices of the [sustainability pillar](#) were applied when designing this solution.

- The solution's serverless design (using Amazon Kinesis Data Streams, Amazon EMR Serverless, Amazon Redshift Serverless and Amazon QuickSight) and the use of managed services (such as Amazon ECS, Amazon MSK) are aimed at reducing carbon footprint compared to the footprint of continually operating on-premises servers.

Architecture details

This section describes the components and AWS services that make up this solution and the architecture details on how these components work together.

Solution components

The Clickstream Analytics on AWS solution has four components: a web console, Analytics Studio, SDKs, and the data pipeline.

Web console

This solution provides a simple web console which allows you to create and manage projects with their data pipeline to ingest, process, analyze and visualize the Clickstream data.

Analytics Studio

Analytics Studio is a unified web interface for business analysts or data analysts to view and create dashboards, query and explore clickstream data, and manage metadata. It supports the following features:

- provide a pre-canned user lifecycle dashboard
- provide an explorative analytics model to query and analyze clickstream data
- support creating custom analysis and visualization in a drag-and-drop manner
- auto-generate metadata for clickstream data, and support metadata management

SDKs

This solution provides native SDKs to help you easily collect and report in-app events from your applications to your Clickstream pipelines.

- [Android SDK](#)
- [Swift SDK](#)
- [Web SDK](#)
- [Flutter SDK](#)

Data pipeline

This solution uses the web console to manage the project and its data pipeline. The data pipeline consists of four modules.

Ingestion module

The ingestion module serves as web server for ingesting the Clickstream data. It supports the following features:

- Specify the auto scaling group capability
- Specify warm pool size to scale out faster and save costs
- Support authenticate with OIDC
- Support SSL
- Support enabling AWS Global Accelerator for ELB
- Support different sink buffer, S3, KDS and MSK

Data processing module

The data processing module transforms and enriches the ingested data to solution's data model by the Apache Spark application running in EMR serverless. It supports the following features:

- Specify the batch interval of data processing
- Specify the data refreshness age
- Provider out-of-the-box enrichment plug-ins
 - UA enrichment to parse OS, device, browser information from User Agent string of the HTTP request header
 - IP enrichment to mapping device location information (for example, city, country, region) based on the request source IP
- Support third-party transformer plug-ins
- Support third-party enrichment plug-ins

Data modeling module

The data modeling module loads the processed data into lake house. It supports the following features:

- Support both provisioned Redshift and Redshift Serverless as data warehouse
 - Users can specify the time range for storing data in Redshift
 - Users can specify the interval to update user dimension table
- Support use Athena to query the data in data lake

Reporting module

The reporting module creates a secure connection to the data warehouse and provisions the out-of-the-box dashboards in business intelligence Amazon QuickSight.

AWS services in this solution

The following AWS services are included in this solution:

AWS service	Description
Amazon Elastic Load Balancing	Core. To distribute network traffic to ingestion fleet.
Amazon ECS	Core. To run the ingestion module fleet.
Amazon EC2	Core. To provide the underlying computing resources for ingestion fleet.
Amazon ECR	Core. To host the container images used by ingestion fleet.
Amazon S3	Core. To store the ingested and processed Clickstream data. And it also stores the service logs and static web assets (frontend user interface).
AWS Global Accelerator	Supporting. To improve the availability, performance, and security of the ingestion service in AWS Regions.
AWS CloudWatch	Supporting. To monitor the metrics, logs and trace of data pipeline.

AWS service	Description
Amazon SNS	Supporting. To provide topic and email subscription notifications for the alarms of data pipeline.
Amazon Kinesis Data Streams	Supporting. To provide the ingestion buffer.
AWS Lambda	Supporting. To integrate with kinds of AWS services. For example, sink ingestion data to S3, manage the lifecycle of AWS resources.
Amazon Managed Streaming for Apache Kafka (MSK)	Supporting. To provide the ingestion buffer with Apache Kafka.
Amazon EMR Serverless	Supporting. To process the ingested data.
Amazon Glue	Supporting. To manage the data catalog of ingested data.
Amazon EventBridge	Supporting. To integrate with AWS services with events or schedule.
Amazon Redshift	Supporting. To analyze your Clickstream data in data warehouse.
Amazon Athena	Supporting. To analyze your Clickstream data in data lake.
AWS Step Functions	Supporting. To orchestrate the lifecycle management of project's pipeline. Also it manages the workflow to load data into data warehouse.
AWS Secrets Manager	Supporting. To store the credential for OIDC credentials and BI user in Redshift.
Amazon QuickSight	Supporting. Visual your analysis reporting of your Clickstream data.

AWS service	Description
Amazon CloudFront	Supporting. To made available the static web assets (frontend user interface) and proxy the backend in the same origin.
Amazon Cognito	Supporting. To authenticate users (in AWS Regions).
Amazon API Gateway	Supporting. To provide the backend APIs.
Amazon DynamoDB	Supporting. To store projects data.
AWS CloudFormation	Supporting. To provision the AWS resources for the modules of data pipeline.

Plan your deployment

This section describes the cost, security, and Region considerations for planning your deployment.

Cost

Important

The following cost estimations are examples and may vary depending on your environment.

You are responsible for the cost of AWS services used when running this solution. Deploying this solution will only create a solution web console in your AWS account, which is completely serverless and typically can be covered within free tier.

The cost for this solution is mostly incurred by the data pipeline. As of this revision, the main factors affecting the solution cost include:

- **Ingestion module**, the cost depends on the size of the ingestion server and the type of the data sink you choose.
- **Data processing and modeling module** (optional), the cost depends on whether you choose to enable this module and its relevant configurations
- **Enabled Dashboards** (optional), the cost depends on whether you choose to enable this module and its relevant configurations
- **Additional features**

The following are cost estimations for data volumes of 10/100/1000/10000 RPS (request per second) with different data pipeline configurations. Cost estimation are provided by modules. To get a total cost for your use case, sum the cost by modules based on your actual configuration.

Important

As of this revision, the following cost is calculated with On-Demand prices in the us-east-1 Region and measured in USD.

Ingestion module

Ingestion module includes the following cost components:

- Application load balancer
- EC2 for ECS
- Data sink (Kinesis | Kafka | Direct to S3)
- S3 storage

Key assumptions include:

- Compressed request payload: 2KB (10 events per request)
- MSK configurations (m5.large * 2)
- KDS configuration (on-demand, provision)
- 10/100/1000 request per second (RPS)

RPS	ALB cost	EC2 cost	Buffer type	Buffer cost	S3 cost	Total (USD per month)
10 RPS (49GB per month)	\$18	\$122	Kinesis (On-Demand)	\$38	\$3	\$181
	\$18	\$122	Kinesis (Provisioned 2 shard)	\$22	\$3	\$165
	\$18	\$122	MSK (m5.large * 2, connector MCU * 1)	\$417	\$3	\$560
	\$18	\$122	None		\$3	\$143

RPS	ALB cost	EC2 cost	Buffer type	Buffer cost	S3 cost	Total (USD per month)
100 PRS (490GB per month)	\$43	\$122	Kinesis(O n-demand)	\$115	\$4	\$284
	\$43	\$122	Kinesis (Provisio ned 2 shard)	\$26	\$4	\$195
	\$43	\$122	MSK (m5.large * 2, connector MCU * 1)	\$417	\$4	\$586
1000 RPS (4900GB per month)	\$43	\$122	None		\$4	\$169
	\$252	\$122	Kinesis(O n-demand)	\$1051	\$14	\$1439
	\$252	\$122	Kinesis (Provisio ned 10 shard)	\$180	\$14	\$568
	\$252	\$122	MSK (m5.large * 2, connector MCU * 2~3)	\$590	\$14	\$978
	\$252	\$122	None		\$14	\$388

Data transfer

There are associated costs for data transfer from EC2 to the downstream data sink. Below is an example of data transfer costs based on 1000 RPS and a 1KB request payload.

- EC2 Network In: This does not incur any costs.
- EC2 Network Out: There are three data sink types (Amazon S3, Amazon MSK, and Amazon Kinesis Data Streams).

Data sink type	Way to access data sink	Dimensions	Total (USD)
Amazon S3	Amazon S3 gateway endpoints	The Amazon S3 gateway endpoints do not incur any costs.	\$0
Amazon MSK	N/A	Data processed cost (\$0.010 per GB in/out/between EC2 Availability Zones)	\$210
Amazon Kinesis Data Streams	NAT	NAT fixed cost: \$64 (2 Availability Zones and a NAT per Availability Zone, \$0.045 per NAT Gateway Hour). Data processed cost: \$1201 (\$0.045 per GB Data Processed by NAT gateways).	\$1266
Amazon Kinesis Data Streams	VPC endpoint	VPC endpoint fixed cost: \$14.62 (Availability Zones \$0.01 per AZ Hour).	\$281.62

Data sink type	Way to access data sink	Dimensions	Total (USD)
		Data processed cost: \$267 (\$0.01 per GB Data Processed by Interface endpoints).	

You are recommended to use a VPC endpoint for the Amazon Kinesis Data Streams data sink. For more information, refer to the [VPC endpoint](#) documentation.

Data processing & modeling modules

Data processing & modeling module include the following cost components if you enable:

- EMR Serverless
- Redshift

Key assumptions include:

- 10/100/1000 RPS
- Data processing interval: hourly/6-hourly/daily
- EMR running three built-in plugins to process data

RPS	EMR schedule interval	EMR cost	Redshift type	Redshift load cost	Redshift storage cost	S3 cost	Total (USD/ Month)
10 RPS	Hourly	\$72 (\$1.47/GB)	Serverless (8 based RPU)	\$74	\$3.4	\$0.36	\$149.76
	6-hourly	\$41.5 (\$0.84/GB)	Serverless (8)	\$13	\$3.4	\$0.36	\$58.26

RPS	EMR schedule interval	EMR cost	Redshift type	Redshift load cost	Redshift storage cost	S3 cost	Total (USD/ Month)
			based RPU)				
	Daily	\$26.7 (\$0.54/ GB)	Serverless (8 based RPU)	\$3	\$3.4	\$0.36	\$38.46
100 RPS	Hourly	\$321 (\$0.65/ GB)	Serverless (8 based RPU)	\$96	\$34	\$3.6	\$454.60
	6-hourly	\$202 (\$0.41/ GB)	Serverless (8 based RPU)	\$31	\$34	\$3.6	\$270.60
	Daily	\$281 (\$0.57/ GB)	Serverless (8 based RPU)	\$21	\$34	\$3.6	\$339.60
1000 RPS	40 minutes (recommended)	\$1926 (\$0.39/ GB)	Serverless (8 based RPU)	\$440	\$340	\$36	\$2742

Note

The term **Redshift storage cost** refers to the cost of Redshift storage incurred for one month based on the corresponding RPS. If the data is stored for more than one month, please refer to the [Redshift pricing](#).

Reporting module

Reporting module includes the following cost components if you choose to enable:

- QuickSight

Key assumptions include

- QuickSight Enterprise subscription
- Exclude Q cost
- Analytics Studio
- 10GB SPICE capacity

Daily data volume/ RPS	Cost for authors	Cost for SPICE	Total cost (USD/ Month)
All size	\$24	0	\$24

Note

The QuickSight cost applies to all your data pipelines, including the visualization managed outside the solution.

Logging and monitoring

The solution utilizes CloudWatch Logs, CloudWatch Metrics and CloudWatch Dashboard to implement logging, monitoring and visualizing features. The total cost is around \$14 per month and may vary based on the volume of logs and the number of metrics being monitored.

Additional features

You will be charged with additional cost only if you choose to enable the following features.

Secrets Manager

- If you enable reporting, the solution creates a secret in Secrets Manager to store the Redshift credentials used by QuickSight visualization. **Cost:** 0.40 USD/month.
- If you enable the authentication feature of the ingestion module, you need to create a secret in Secrets Manager to store the information for OIDC. **Cost:** 0.40 USD/month.

AWS Global Accelerator

It incurs a fixed hourly charge and a per-day volume data transfer cost.

Key assumptions: Ingestion deployment in us-east-1

RPS	Fixed hourly cost	Data transfer cost	Total cost (USD/ Month)
10 RPS	\$18	\$0.60	\$18.60
100 RPS	\$18	\$6	\$24
1000 RPS	\$18	\$60	\$78

Application Load Balancer access logs

The charged cost includes the storage cost for Amazon S3, but not for the bandwidth used by Elastic Load Balancing to send log files to Amazon S3. For more information, see [Amazon S3 pricing](#).

RPS	Log size (GB)	S3 cost (USD/Month)
10 RPS	16.50	\$0.38
100 RPS	165	\$3.80
1000 RPS	1650	\$38

Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This [shared responsibility model](#) reduces your operational burden because AWS operates, manages, and controls the components including the host operating system, the virtualization layer, and the physical security of the facilities in which the services operate. For more information about AWS security, see [AWS Cloud Security](#).

IAM Roles

AWS Identity and Access Management (IAM) roles allow customers to assign granular access policies and permissions to services and users on the AWS Cloud. This solution creates IAM roles that grant the solution's AWS Lambda functions, Amazon API Gateway and Amazon Cognito or OpenID connect access to create regional resources.

Amazon VPC

This solution optionally deploys a web console within your VPC. You can isolate access to the web console via Bastion hosts, VPNs, or Direct Connect. You can create [VPC endpoints](#) to let traffic between your Amazon VPC and AWS services not leave the Amazon network to satisfy the compliance requirements.

Security groups

The security groups created in this solution are designed to control and isolate network traffic between the solution components. We recommend that you review the security groups and further restrict access as needed once the deployment is up and running.

Amazon CloudFront

This solution optionally deploys a web console hosted in an Amazon S3 bucket and Amazon API Gateway. To help reduce latency and improve security, this solution includes an Amazon CloudFront distribution with an Origin Access Control (OAC), which is a CloudFront user that provides public access to the solution's website bucket contents. For more information, refer to [Restricting access to an Amazon S3 origin](#) in the Amazon CloudFront Developer Guide.

Supported AWS Regions

This solution uses services which may not be currently available in all AWS Regions. Launch this solution in an AWS Region where required services are available. For the most current availability by Region, refer to the [AWS Regional Services List](#).

Clickstream Analytics on AWS provides two types of authentication for its web console, [Amazon Cognito User Pool](#) and [OpenID Connect \(OIDC\) Provider](#). You must choose to launch the solution with OpenID Connect in case one of the following scenarios:

- Amazon Cognito User Pool is not available in your AWS Region.
- You already have an OpenID Connect Provider and want to authenticate against it.

Supported Regions for web console deployment

Region Name	Launch with Amazon Cognito user pool	Launch with OpenID Connect
US East (N. Virginia)	Yes	Yes
US East (Ohio)	Yes	Yes
US West (N. California)	Yes	Yes
US West (Oregon)	Yes	Yes
Africa (Cape Town)	No	Yes
Asia Pacific (Hong Kong)	No	Yes
Asia Pacific (Jakarta)	No	Yes
Asia Pacific (Mumbai)	Yes	Yes
Asia Pacific (Osaka)	No	Yes
Asia Pacific (Seoul)	Yes	Yes
Asia Pacific (Singapore)	Yes	Yes

Region Name	Launch with Amazon Cognito user pool	Launch with OpenID Connect
Asia Pacific (Sydney)	Yes	Yes
Asia Pacific (Tokyo)	Yes	Yes
Canada (Central)	Yes	Yes
Europe (Frankfurt)	Yes	Yes
Europe (Ireland)	Yes	Yes
Europe (London)	Yes	Yes
Europe (Milan)	No	Yes
Europe (Paris)	Yes	Yes
Europe (Stockholm)	Yes	Yes
Middle East (Bahrain)	No	Yes
South America (Sao Paulo)	Yes	Yes
China (Beijing) Region Operated by Sinnet	No	Yes
China (Ningxia) Region Operated by NWCD	No	Yes

This solution provides [modular components](#) for supporting different data pipeline architecture. The data processing, and reporting modules are optional, that is, you can create a data pipeline without data processing and reporting modules if needed.

Pipeline modules availability

Region Name	Data ingestion with MSK as buffer	Data ingestion with KDS as buffer	Data ingestion with S3 as buffer	Data processing	Data modeling with Redshift Serverless	Data modeling with Provisioned Redshift	Reporting with QuickSight
US East (N. Virginia)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
US East (Ohio)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
US West (N. California)	Yes	Yes	Yes	Yes	Yes	Yes	No
US West (Oregon)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Africa (Cape Town)	Yes	Yes	Yes	No	No	No	No
Asia Pacific (Hong Kong)	Yes	Yes	Yes	Yes	No	Yes	No
Asia Pacific (Mumbai)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Asia Pacific (Osaka)	Yes	Yes	Yes	No	No	No	No

Region Name	Data ingestion with MSK as buffer	Data ingestion with KDS as buffer	Data ingestion with S3 as buffer	Data processing	Data modeling with Redshift Serverless	Data modeling with Provisioned Redshift	Reporting with QuickSight
Asia Pacific (Seoul)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Asia Pacific (Singapore)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Asia Pacific (Sydney)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Asia Pacific (Tokyo)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Canada (Central)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Europe (Frankfurt)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Europe (Ireland)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Europe (London)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Europe (Milan)	Yes	Yes	Yes	No	No	No	No

Region Name	Data ingestion with MSK as buffer	Data ingestion with KDS as buffer	Data ingestion with S3 as buffer	Data processing	Data modeling with Redshift Serverless	Data modeling with Provisioned Redshift	Reporting with QuickSight
Europe (Paris)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Europe (Stockholm)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Middle East (Bahrain)	Yes	Yes	Yes	Yes	No	Yes	No
South America (Sao Paulo)	Yes	Yes	Yes	Yes	No	Yes	Yes
China (Beijing) Region Operated by Sinnet*	Yes	Yes	Yes	Yes	No	Yes	Yes
China (Ningxia) Region Operated by NWCD*	Yes	Yes	Yes	Yes	No	Yes	No

Note

AWS China Regions don't support using AWS Global Accelerator to accelerate the ingestion endpoint.

Deployment

Before you launch the solution, review the architecture, supported Regions, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the solution into your account.

Prerequisites

Review all the [considerations](#) and make sure you have the following in the target Region you want to deploy the solution:

- At least two vacant S3 buckets.

Deployment in AWS Regions

Clickstream Analytics on AWS provides two ways to authenticate and log into the solution web console. For some AWS Regions where Cognito User Pool is unavailable (for example, Hong Kong), you must launch the solution with OpenID Connect.

- [Launch with Cognito User Pool](#) (Recommended)
- [Launch with OpenID Connect](#)

For more information about supported Regions, see [Regional deployments](#).

Deployment in AWS China Regions

AWS China Regions do not have Cognito User Pool. You must launch the solution with OpenID Connect.

- [Launch with OpenID Connect](#)

Deployment within Amazon VPC

Clickstream Analytics on AWS supports being deployed into an Amazon VPC, allowing access to the web console without leaving your VPC network.

- [Launch within VPC](#)

Launch with Cognito User Pool

Time to deploy: Approximately 15 minutes

Deployment overview

Use the following steps to deploy this solution on AWS.



[Step 1. Launch the stack](#)

[Step 2. Launch the web console](#)

Step 1: Launch the Stack

This AWS CloudFormation template automatically deploys the Clickstream Analytics on AWS solution on AWS.

1. Sign in to the [AWS Management Console](#) and select the button to launch the AWS CloudFormation template.

	Launch in AWS Management Console
Launch stack	
Launch stack with custom domain	

2. The template is launched in the default Region after you log in to the console. To launch the Clickstream Analytics on AWS solution in a different AWS Region, use the Region selector in the console navigation bar.
3. On the **Create stack** page, verify that the correct template URL is shown in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack. For information about naming character limitations, refer to IAM and AWS STS quotas in the *AWS Identity and Access Management User Guide*.
5. Under **Parameters**, review the parameters for the template and modify them as necessary.

- This solution uses the following parameters:

Parameter	Default	Description
Email	<i><Requires input></i>	Specify the email of the Administrator. This email address will receive a temporary password to access the Clickstream Analytics on AWS web console. You can create more users directly in the provisioned Cognito User Pool after launching the solution.

Important

By default, this deployment uses TLSv1.0 and TLSv1.1 in CloudFront. However, we recommend that you manually configure CloudFront to use the securer TLSv1.2/ TLSv1.3 and apply for a certificate and custom domain to enable this. We highly recommend that you update your TLS configuration and cipher suite selection according to the following recommendations:

- **Transport Layer Security Protocol:** Upgrade to TLSv1.2 or higher
 - **Key Exchange:** ECDHE
 - **Block Cipher Mode of Operation:** GCM
 - **Authentication:** ECDSA
 - **Encryption Cipher:** AES256
 - **Message Authentication:** SHA(256/384/any hash function except for SHA1)
- For example, TLSv1.2_2021 can be used to meet these recommendations.

- If you are launching the solution with custom domain in AWS regions, this solution uses the additional following parameters:

Parameter	Default	Description
Hosted Zone ID	<i><Requires input></i>	Choose the public hosted zone ID of Amazon Route 53.
Hosted Zone Name	<i><Requires input></i>	The domain name of the public hosted zone, for example, <code>example.com</code> .
Record Name	<i><Requires input></i>	The sub name (as known as record name in R53) of the domain name of console. For example, enter <code>clickstream</code> , if you want to use custom domain <code>clickstream.example.com</code> for the console.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Select the box acknowledging that the template creates AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a **CREATE_COMPLETE** status in approximately 15 minutes.

Step 2: Launch the web console

After the stack is successfully created, this solution generates a CloudFront domain name that gives you access to the Clickstream Analytics on AWS web console. Meanwhile, an auto-generated temporary password will be sent to your email address.

1. Sign in to the [AWS CloudFormation console](#).
2. On the **Stacks** page, select the solution's stack.

3. Choose the **Outputs** tab and record the domain name.
4. Open the **ControlPlaneURL** using a web browser, and navigate to a sign-in page.
5. Enter the **Email** and the temporary password.
 - a. Set a new account password.
 - b. (Optional) Verify your email address for account recovery.

After the verification is complete, the system opens the Clickstream Analytics on AWS web console.

Once you have logged into the Clickstream Analytics on AWS console, you can start to [create a project](#) for your applications.

Launch with OpenID Connect (OIDC)

Time to deploy: Approximately 30 minutes

Prerequisites

Important

The Clickstream Analytics on AWS console is served via CloudFront distribution which is considered as an Internet information service. If you are deploying the solution in **AWS China Regions**, the domain must have a valid [ICP Recordal](#).

- **A domain.** You will use this domain to access the Clickstream Analytics on AWS console. This is required for AWS China Regions, and is optional for AWS Regions.
- **An SSL certificate in AWS IAM.** The SSL must be associated with the given domain. Follow [this guide](#) to upload SSL certificate to IAM. This is required for AWS China Regions, but is not recommended for AWS Regions.
- Make sure to request or import the ACM certificate in the US East (N. Virginia) Region (us-east-1). This is not required for AWS China Regions, and is optional for AWS Regions.

Deployment overview

Use the following steps to deploy this solution on AWS.

[Step 1. Create OIDC client](#)

[Step 2. Launch the stack](#)

[Step 3. Update the callback URL of OIDC client](#)

[Step 4. Set up DNS Resolver](#)

[Step 5. Launch the web console](#)

Step 1. Create OIDC client

You can use different kinds of OpenID Connect (OIDC) providers. This section introduces Option 1 to Option 4.

- (Option 1) Using Amazon Cognito from another Region as OIDC provider.
- (Option 2) [Authing](#), which is an example of a third-party authentication provider.
- (Option 3) [Keycloak](#), which is a solution maintained by AWS and can serve as an authentication identity provider.
- (Option 4) [ADFS](#), which is a service offered by Microsoft.
- (Option 5) Other third-party authentication platforms such as [Auth0](#).

Follow the steps below to create an OIDC client, and obtain the `client_id` and `issuer`.

(Option 1) Using Amazon Cognito User Pool from another Region

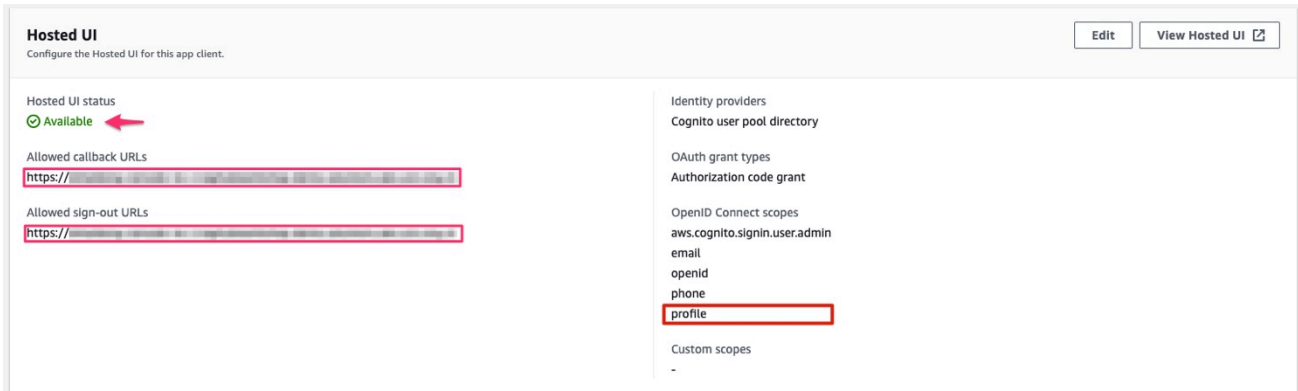
You can leverage the [Amazon Cognito User Pool](#) in a supported AWS Region as the OIDC provider.

1. Go to the [Amazon Cognito console](#) in an AWS Region.
2. Set up the hosted UI with the Amazon Cognito console based on this [guide](#).
 - Choose **Public client** when selecting the **App type**. Make sure NOT to change the selection **Don't generate a client secret** for **Client secret**.
 - Add **Profile** in **OpenID Connect scopes**.
3. Enter the **Callback URL** and **Sign out URL** using your domain name for Clickstream Analytics on AWS console.
 - Callback URL: `http[s]://<domain-name>/signin`
 - Sign out URL: `http[s]://<domain-name>`

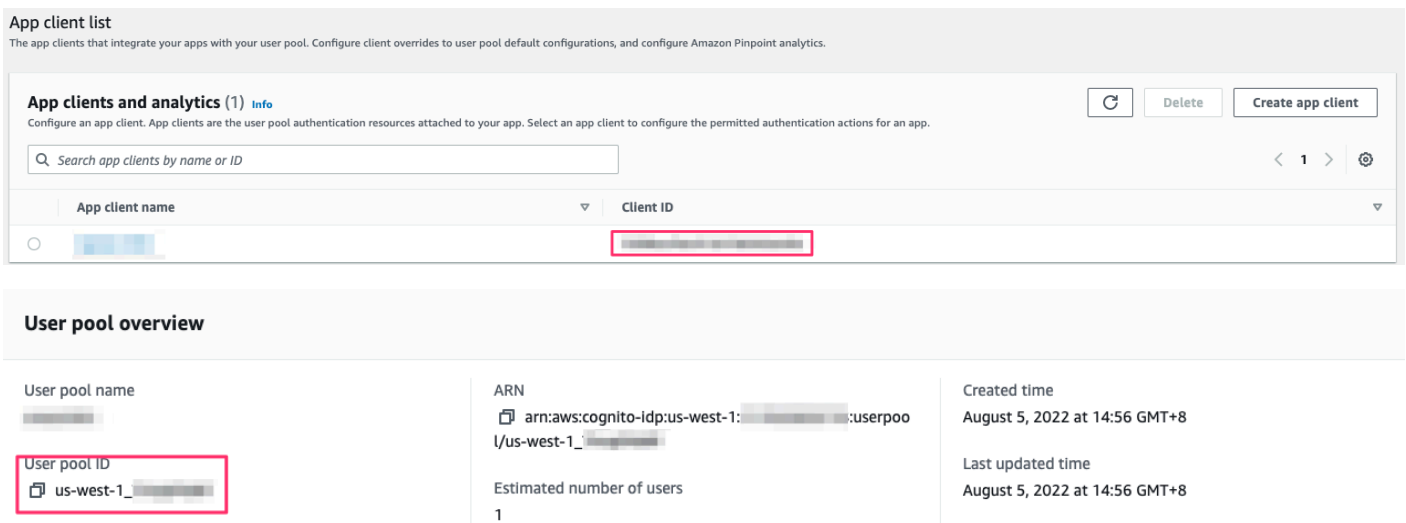
Note

If you don't know the custom domain name of console, you can firstly enter one, for example, `clickstream.example.com`, and then update it following guidelines in [Step 3 Update the callback URL of OIDC client](#).

4. If your hosted UI is set up, you should be able to see something like below:



5. Save the App client ID, User pool ID and the AWS Region to a file, which will be used later.

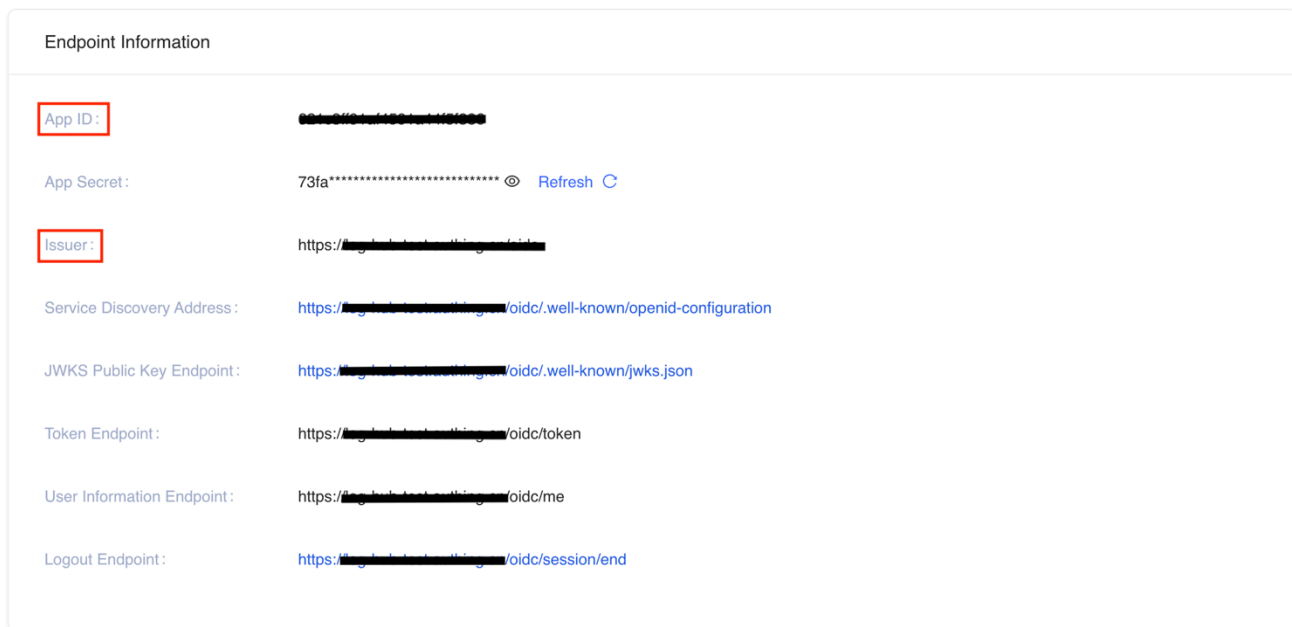


In [Step 2. Launch the stack](#), enter the parameters below from your Cognito User Pool.

- **OIDCClientId:** App client ID
- **OIDCProvider:** `https://cognito-idp.${REGION}.amazonaws.com/${USER_POOL_ID}`

(Option 2) Authing.cn OIDC client

1. Go to the [Authing console](#).
2. Create a user pool if you don't have one.
3. Select the user pool.
4. On the left navigation bar, select **Self-built App** under **Applications**.
5. Choose **Create**.
6. Enter the **Application Name**, and **Subdomain**.
7. Save the App ID (that is, `client_id`) and Issuer to a text file from Endpoint Information, which will be used later.



Endpoint Information

App ID:	██
App Secret:	73fa***** @ Refresh
Issuer:	https://██
Service Discovery Address:	https://██/oidc/well-known/openid-configuration
JWKS Public Key Endpoint:	https://██/oidc/well-known/jwks.json
Token Endpoint:	https://██/oidc/token
User Information Endpoint:	https://██/oidc/me
Logout Endpoint:	https://██/oidc/session/end

8. Update the Login Callback URL and Logout Callback URL to your IPC recorded domain name.
9. Set the Authorization Configuration.

Authorization Configuration Save

Authorization Flow [?]: authorization_code implicit refresh_token password
 client_credentials

Return Type [?]: code id_token token code id_token code token code
 id_token token id_token none

Id_token signature algorithm [?]: HS256 RS256

Don't enforce https for implicit mode callback [?]:

Enable id_token encryption [?]:

Client Verification Method for Fetching Token: client_secret_post client_secret_basic none

Client Verification Method for Validating Token: client_secret_post client_secret_basic none

Client Verification Method for Revoking Token: client_secret_post client_secret_basic none

You have successfully created an Authing self-built application.

In [Step 2. Launch the stack](#), enter the parameters below from your Cognito User Pool.

- **OIDCClientId:** client ID
- **OIDCProvider:** Issuer

(Option 3) Keycloak OIDC client

1. Deploy the Keycloak solution in AWS China Regions following [this guide](#).
2. Sign in to the Keycloak console.
3. On the left navigation bar, select **Add realm**. Skip this step if you already have a realm.
4. Go to the realm setting page. Choose **Endpoints**, and then **OpenID Endpoint Configuration** from the list.

Example 

General Login Keys Email Themes Localization Cache Tokens Client Registration Security Defenses

* Name

Display name

HTML Display name

Frontend URL

Enabled

User-Managed Access

Endpoints

5. In the JSON file that opens up in your browser, record the **issuer** value which will be used later.

```

{"issuer": "https://1.elb.amazonaws.com.cn/auth/realms/example/protocol/openid-connect/auth", "token_endpoint": "https://keycloak-159azf1.elb.amazonaws.com.cn/auth/realms/example/protocol/openid-connect/token", "introspection_endpoint": "https://keycloak-1.elb.amazonaws.com.cn/auth/realms/example/protocol/openid-connect/token/introspect", "userinfo_endpoint": "https://keycloak-1.elb.amazonaws.com.cn/auth/realms/example/protocol/openid-connect/userinfo"}

```

- Go back to Keycloak console and select **Clients** on the left navigation bar, and choose **Create**.
- Enter a Client ID, which must contain 24 letters (case-insensitive) or numbers. Record the **Client ID** which will be used later.
- Change client settings. Enter `http[s]://<Clickstream Analytics on AWS Console domain>/signin` in **Valid Redirect URIs**, and enter `<console domain>` and `+` in **Web Origins**.

 **Note**

If you're not using custom domain for the console, the domain name of console is not available yet. You can enter a fake one, for example, `clickstream.example.com`, and then update it following guidelines in Step 3.

- In the Advanced Settings, set the **Access Token Lifespan** to at least 5 minutes.
- Select **Users** on the left navigation bar.
- Click **Add user** and enter **Username**.
- After the user is created, select **Credentials**, and enter **Password**.

In [Step 2. Launch the stack](#), enter the parameters below from your Keycloak realm.

- **OIDCClientId:** `client id`
- **OIDCProvider:** `https://<KEYCLOAK_DOMAIN_NAME>/auth/realms/<REALM_NAME>`

(Option 4) ADFS OpenID Connect Client

1. Make sure your ADFS is installed. For information about how to install ADFS, refer to [this guide](#).
2. Log in to the ADFS Sign On page. The URL should be `>https://ads.domain.com/ads/ls/idpinitiatedSignOn.aspx`, and you need to replace **ads.domain.com** with your real ADFS domain.
3. Log on your **Domain Controller**, and open **Active Directory Users and Computers**.
4. Create a **Security Group** for Clickstream Analytics on AWS users, and add your planned users to this Security Group.
5. Log on to ADFS server, and open **ADFS Management**.
6. Right click **Application Groups**, choose **Application Group**, and enter the name for the Application Group. Select **Web browser accessing a web application** option under **Client-Server Applications**, and choose **Next**.
7. Record the **Client Identifier** (`client_id`) under **Redirect URI**, enter your Clickstream Analytics on AWS domain (for example, `xx.domain.com`), and choose **Add**, and then choose **Next**.
8. In the **Choose Access Control Policy** window, select **Permit specific group**, select **parameters** under Policy part, add the created Security Group in Step 4, then choose **Next**. You can configure other access control policy based on your requirements.
9. Under Summary window, choose **Next**, and choose **Close**.
10. Open the Windows PowerShell on ADFS Server, and run the following commands to configure ADFS to allow CORS for your planned URL.

```
Set-AdfsResponseHeaders -EnableCORS $true
Set-AdfsResponseHeaders -CORSTrustedOrigins https://<your-domain>
```

11. Under Windows PowerShell on ADFS server, run the following command to get the Issuer (`issuer`) of ADFS, which is similar to `https://ads.domain.com/ads`.

```
Get-ADFSProperties | Select IdTokenIssuer
```

```
PS C:\Users\Administrator.AWS> Get-ADFSProperties | Select IdTokenIssuer
IdTokenIssuer
-----
https://sts.aws.azeroth.zone/adfs
```

In [Step 2. Launch the stack](#), enter the parameters below from your ADFS server.

- **OIDCClientId:** `client id`
- **OIDCProvider:** Get the server of the issuer from above step 11

Step 2. Launch the Stack

1. Sign in to the [AWS Management Console](#) and use the button below to launch the AWS CloudFormation template.

	Launch in AWS Management Console
Launch in AWS Regions	Launch Stack
Launch with custom domain in AWS Regions	Launch Stack
Launch in AWS China Regions	Launch Stack

2. The template is launched in the default region after you log in to the console. To launch the Clickstream Analytics on AWS solution in a different AWS Region, use the Region selector in the console navigation bar.
3. On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack. For information about naming character limitations, refer to [IAM and AWS STS quotas](#) in the *AWS Identity and Access Management User Guide*.
5. Under **Parameters**, review the parameters for the template and modify them as necessary.
 - This solution uses the following parameters:

Parameter	Default	Description
OIDCClientId	<Requires input>	OpenID Connect client Id.
OIDCProvider	<Requires input>	OpenID Connector provider issuer. The issuer must begin with https://
Email	<Requires input>	Specify the email of the Administrator.

Important

By default, this deployment uses TLSv1.0 and TLSv1.1 in CloudFront. However, we recommend that you manually configure CloudFront to use the securer TLSv1.2/ TLSv1.3 and apply for a certificate and custom domain to enable this. We highly recommend that you update your TLS configuration and cipher suite selection according to the following recommendations:

- **Transport Layer Security Protocol:** Upgrade to TLSv1.2 or higher
 - **Key Exchange:** ECDHE
 - **Block Cipher Mode of Operation:** GCM
 - **Authentication:** ECDSA
 - **Encryption Cipher:** AES256
 - **Message Authentication:** SHA(256/384/any hash function except for SHA1)
- For example, TLSv1.2_2021 can be used to meet these recommendations.

- If you are launching the solution with custom domain in AWS Regions, this solution has the following additional parameters:

Parameter	Default	Description
Hosted Zone ID	<Requires input>	Choose the public hosted zone ID of Amazon Route 53.

Parameter	Default	Description
Hosted Zone Name	<Requires input>	The domain name of the public hosted zone, for example, example.com .
Record Name	<Requires input>	The sub name (as known as record name in R53) of the domain name of console. For example, enter clickstream , if you want to use custom domain clickstream.example.com for the console.

- If you are launching the solution in AWS China Regions, this solution has the following additional parameters:

Parameter	Default	Description
Domain	<Requires input>	Custom domain for the web console. Do NOT add http(s) prefix.
IamCertificateID	<Requires input>	The ID of the SSL certificate in IAM. The ID is composed of 21 characters of capital letters and digits. Use the list-server-certificates command to retrieve the ID.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template creates AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a **CREATE_COMPLETE** status in approximately 15 minutes.

Step 3. Update the callback URL of OIDC client

Important

If you don't deploy stack with custom domain, you must complete below steps.

1. Sign in to the [AWS CloudFormation console](#).
2. Select the solution's stack.
3. Choose the **Outputs** tab.
4. Obtain the **ControlPlaneURL** as the endpoint.
5. Update or add the callback URL to your OIDC.
 - For Cognito, add or update the url in **Allowed callback URL** of your client with value `${ControlPlaneURL}/signin`. The url must start with `https://`.
 - For Keycloak, add or update the url in **Valid Redirect URIs** of your client with value `${ControlPlaneURL}/signin`.
 - For Authing.cn, add or update the url in **Login Callback URL** of **Authentication Configuration**.

Step 4. Setup DNS Resolver

Important

If you deploy stack in AWS Regions, you can skip this step.

This solution provisions a CloudFront distribution that gives you access to the Clickstream Analytics on AWS console.

1. Sign in to the [AWS CloudFormation console](#).
2. Select the solution's stack.
3. Choose the **Outputs** tab.
4. Obtain the **ControlPlaneURL** and **CloudFrontDomainName**.

5. Create a CNAME record for **ControlPlaneURL** in DNS resolver, which points to the domain **CloudFrontDomainName** obtained in previous step.

Step 5. Launch the Web Console

Important

Your login credentials are managed by the OIDC provider. Before signing in to the Clickstream Analytics on AWS console, make sure you have created at least one user in the OIDC provider's user pool.

1. Use the previous assigned domain name or generated **ControlPlaneURL** in a web browser.
2. Choose **Sign in**, and navigate to OIDC provider.
3. Enter sign-in credentials. You may be asked to change your default password for first-time login, which depends on your OIDC provider's policy.
4. After the verification is complete, the system opens the Clickstream Analytics on AWS web console.

Once you have logged into the Clickstream Analytics on AWS console, you can start to [create a project](#) for your applications.

Launch within VPC

Time to deploy: Approximately 30 minutes

Prerequisites

Review all the [considerations](#) and make sure you have the following in the target region you want to deploy the solution:

- At least one Amazon VPC.
- At least two private or isolated [subnets](#) across two Availability Zones (AZs).

Deployment Overview

Use the following steps to deploy this solution on AWS.

[Step 1. Create OIDC client](#)

[Step 2. Launch the stack](#)

[Step 3. Update the callback url of OIDC client](#)

[Step 4. Launch the web console](#)

Step 1. Create OIDC client



You can use existing OpenID Connector (OIDC) provider or following [this guide](#) to create an OIDC client.

Note

This solution deploys the console in VPC without requiring SSL certificate by default. You have to use an OIDC client to support callback url with http protocol.

Step 2. Launch the stack

1. Sign in to the AWS Management Console and use the button below to launch the AWS CloudFormation template.

	Launch in AWS Management Console
Launch in AWS Regions	
Launch in AWS China Regions	

2. The template is launched in the default Region after you log in to the console. To launch the Clickstream Analytics on AWS solution in a different AWS Region, use the Region selector in the console navigation bar.

3. On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack. For information about naming character limitations, refer to [IAM and AWS STS quotas](#) in the *AWS Identity and Access Management User Guide*.
5. Under **Parameters**, review the parameters for the template and modify them as necessary.

This solution uses the following parameters:

Parameter	Default	Description
VpcId	<Requires input>	Select the VPC in which the solution will be deployed.
PrivateSubnets	<Requires input>	Select the subnets in which the solution will be deployed. You must choose two subnets across two AZs at least.
OIDCClientId	<Requires input>	OpenID Connect client Id.
OIDCProvider	<Requires input>	OpenID Connector provider issuer. The issuer must begin with https://
Email	<Requires input>	Specify the email of the Administrator.

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template creates AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a **CREATE_COMPLETE** status in approximately 10 minutes.

Step 3. Update the callback URL of OIDC client

1. Sign in to the [AWS CloudFormation console](#).
2. Select the solution's stack.
3. Choose the **Outputs** tab.
4. Obtain the **ControlPlaneURL** as the endpoint.
5. Update or add the callback URL **`\${ControlPlaneURL}/signin** to your OIDC client.
 - For Keycloak, add or update the url in **Valid Redirect URIs** of your client with value **`\${ControlPlaneURL}/signin**.
 - For Authing.cn, add or update the url in **Login Callback URL** of **Authentication Configuration**.

Step 4. Launch the web console

Important

Your login credentials is managed by the OIDC provider. Before signing in to the Clickstream Analytics on AWS console, make sure you have created at least one user in the OIDC provider's user pool.

1. Because you deploy the solution console in your VPC without public access, you have to setup a network connection to the solution console serving by an internal application load balancer. There are some options for your reference.
 - (Option 1) Use bastion host, for example, [Linux Bastion Hosts on AWS](#) solution
 - (Option 2) Use AWS Client VPN or [AWS Site-to-Site VPN](#)
 - (Option 3) Use [AWS Direct Connect](#)
2. The application load balancer only allows the traffic from specified security group, you can find the security group id from the output named **SourceSecurityGroup** from the stack you deployed in step 2. Then attach the security group to your bastion host or other source to access the solution console.
3. Use the previously assigned domain name or the generated **ControlPlaneURL** in a web browser.
4. Choose **Sign In**, and navigate to OIDC provider.
5. Enter sign-in credentials. You may be asked to change your default password for first-time login, which depends on your OIDC provider's policy.

6. After the verification is complete, the system opens the Clickstream Analytics on AWS web console.

Once you have logged into the Clickstream Analytics on AWS console, you can start to [create a project](#) for your applications.

Getting started

After [deploying the solution](#), refer to this section to quickly learn how to leverage the Clickstream Analytics on AWS solution to collect and analyze clickstream data from your applications. This chapter shows you how to create a serverless data pipeline to collect data from an application, and use Analytics Studio to view the out-of-the-box user lifecycle dashboard and query the clickstream data with exploration analytics.

- [Step 1: Create a project.](#)

Create a project.

- [Step 2: Configure a data pipeline.](#)

Configure a data pipeline with serverless infrastructure.

- [Step 3: Integrate SDK.](#)

Integrate SDK into your application to automatically collect data and send data to the pipeline.

- [Step 4: Analyze data.](#)

View the out-of-the-box dashboards based on the data automatically collected from your applications.

Step 1: Create a project

To get started with the Clickstream Analytics on AWS solution, you need to firstly create a project in the solution console. A project is like a container for all the AWS resources provisioned for collecting and analyzing the clickstream data from your apps.

Prerequisites

Make sure you have deployed the Clickstream Analytics on AWS solution. If you haven't, please refer to the [deployment guide](#).

Steps

Following below steps to create a project.

1. Log into **Clickstream Analytics on AWS Management Console**.

2. On the **Home** page, choose **Create Project**.
3. In the window that pops up, enter a project name, for example, `quickstart`.
4. (Optional) Customize the project ID that was automatically created by solution. To do so, click the `edit` icon and update the project ID as per your need.
5. Enter a description for your project, for example, `This is a demo project`.
6. Choose **Next**.
7. Enter an email address to receive notification regarding this project, for example, `email@example.com`, and choose **Next**.
8. Specify an environment type for this project. In this example, select `Dev`.
9. Choose **Create**. Wait until the project creation completed, and you will be directed to the **Projects** page.

We have completed all the steps of creating a project.

Step 2: Configure data pipeline


After you create a project, you need to configure the data pipeline for it. A data pipeline is a set of connected modules that collect and process the clickstream data sent from your applications. A data pipeline contains four modules of ingestion, processing, modeling and reporting. For more information, see [pipeline management](#).

Here we provide an example with steps to create a data pipeline with end-to-end serverless infrastructure.

Steps

1. Sign in to **Clickstream Analytics on AWS Management Console**.
2. In the left navigation pane, choose **Projects**, then select the project you just created in **Step 1**, choose **View Details** in the top right corner to navigate to the project homepage.
3. Choose **Configure pipeline**, and it will bring you to the wizard of creating data pipeline for your project.
4. On the **Basic information** page, fill in the form as follows:
 - AWS Region: **us-east-1**
 - VPC: select a VPC that meets the following requirements
 - At least two public subnets across two different AZs (Availability Zone)

- At least two private subnets across two different AZs
- One NAT Gateway or Instance
- Data collection SDK: **Clickstream SDK**
- Data location: select an S3 bucket. (You can create one bucket, and select it after choosing **Refresh**.)

 **Note**

- Please comply with [Security best practices for Amazon S3](#) to create and configure Amazon S3 buckets. For example, Enable Amazon S3 server access logging, Enable S3 Versioning and so on.
- If you don't have a VPC meet the criteria, you can create a VPC with VPC wizard quickly. For more information, see [Create a VPC](#).

5. Choose **Next**.

6. On the **Configure ingestion** page, fill in the information as follows:

- Fill in the **Ingestion endpoint settings** form.
 - Public Subnets: Select two public subnets in two different AZs
 - Private Subnets: Select two private subnets in the same AZs as public subnets
 - Ingestion capacity: Keep the default values
 - Enable HTTPS: Uncheck and then **Acknowledge** the security warning
 - Additional settings: Keep the default values
- Fill in the **Data sink settings** form.
 - Sink type: **Amazon Kinesis Data Stream(KDS)**
 - Provision mode: **On-demand**
 - In **Additional Settings**, change **Sink Maximum Interval** to 60 and **Batch Size** to 1000
- Choose **Next** to move to step 3.

 **Important**

Using HTTP is not a recommended configuration for production workload. This example configuration is to help you get started quickly.

7. On the **Configure data processing** information, fill in the information as follows:

- In the **Enable data processing** form, turn on **Enable data processing**
- In the **Execution parameters** form,
 - Data processing interval:
 - Select **Fixed Rate**
 - Enter **10**
 - Select **Minutes**
 - Event freshness: **35 Days**

⚠ Important

This example sets Data processing interval to be 10 minutes so that you can view the data faster. You can change the interval to be less frequent later to save cost. Refer to [Pipeline Management](#) to make changes to data pipeline.

- In the **Enrichment plugins** form, make sure the two plugins of **IP lookup** and **UA parser** are selected.
 - In the form of **Analytics engine**, fill in the form as follow:
 - Select the box for **Redshift**
 - Select the **Redshift Serverless**
 - Keep **Base RPU** as **8**
 - VPC: select the default VPC or the same one you selected previously in the last step
 - Security group: select the default security group
 - Subnet: select **three** subnets across three different AZs
 - Keep **Athena** selection as default
 - Choose **Next**.
8. On the **Reporting** page, fill in the form as follows:
- If your AWS account has not subscribed to QuickSight, please follow this [guide](#) to subscribe.
 - Toggle on the option **Enable Analytics Studio**.
 - Choose **Next**.
9. On the **Review and launch** page, review your pipeline configuration details. If everything is configured properly, choose **Create**.

We have completed all the steps of configuring a pipeline for your project. This pipeline will take about 15 minutes to create, and please wait for the pipeline status change to be **Active** in pipeline detail page.

Step 3: Integrate SDK

Once pipeline's status becomes **Active**, it is ready to receive clickstream data. Now you need to register an application to the pipeline, then you can integrate SDK into your application to enable it to send data to the pipeline.

Steps

1. Log into **Clickstream Analytics on AWS Management Console**.
2. In the left navigation pane, choose **Projects**, then select the project (quickstart) you just created in previous steps, click its title, and it will bring you to the project page.
3. Choose **+ Add application** to start adding application to the pipeline.
4. Fill in the form as follows:
 - App name: **test-app**
 - App ID: The system will generate one ID based on the name, and you can customize it if needed.
 - Description: **A test app for Clickstream Analytics on AWS solution**
 - Android package name: leave it blank
 - App Bundle ID: leave it blank
5. Choose **Register App & Generate SDK Instruction**, and wait for the registration to be completed.
6. Select the tab **Android**, and you will see the detailed instruction of adding SDK into your application. You can follow the steps to add SDK.
7. Choose **Download the config json file** to download the config file, and keep this file open, which will be used later.

It will take about 3 ~ 5 minutes to update the pipeline with the application you just add. When you see the pipeline status become **Active** again, it is ready to receive data from your application.

We have completed all the steps of adding an application to a project.

Generate sample data

You might not have immediate access to integrate SDK with your app. In this case, we provide a Python script to generate sample data to the pipeline you just configured, so that you can view and experience the analytics dashboards.

Important

Python 3.8+ is required.

1. Clone the repository to your local environment.

```
git clone https://github.com/aws-labs/clickstream-analytics-on-aws.git
```

2. After you cloned the repository, change directory into the `examples/standalone-data-generator` project folder.
3. Install the dependencies of the project.

```
pip3 install requests
```

4. Put `amplifyconfiguration.json` into the root of `examples/standalone-data-generator` which you downloaded in **register an app** step. See the `examples/standalone-data-generator/README.md` for more information.
5. Open an terminal at this project folder location. For example, if you are using Visual Studio Code IDE, at the top of **Visual Studio Code**, click **Terminal** -> **New Terminal** to open a terminal.
6. Copy the following command and paste it to the terminal:

```
python3 create_event.py
```

Let's enter the `Enter` key in terminal to execute the program. If you see the following output, this means that the program execution is completed.

```
job finished, upload 4360476 events, cost: 95100ms
```

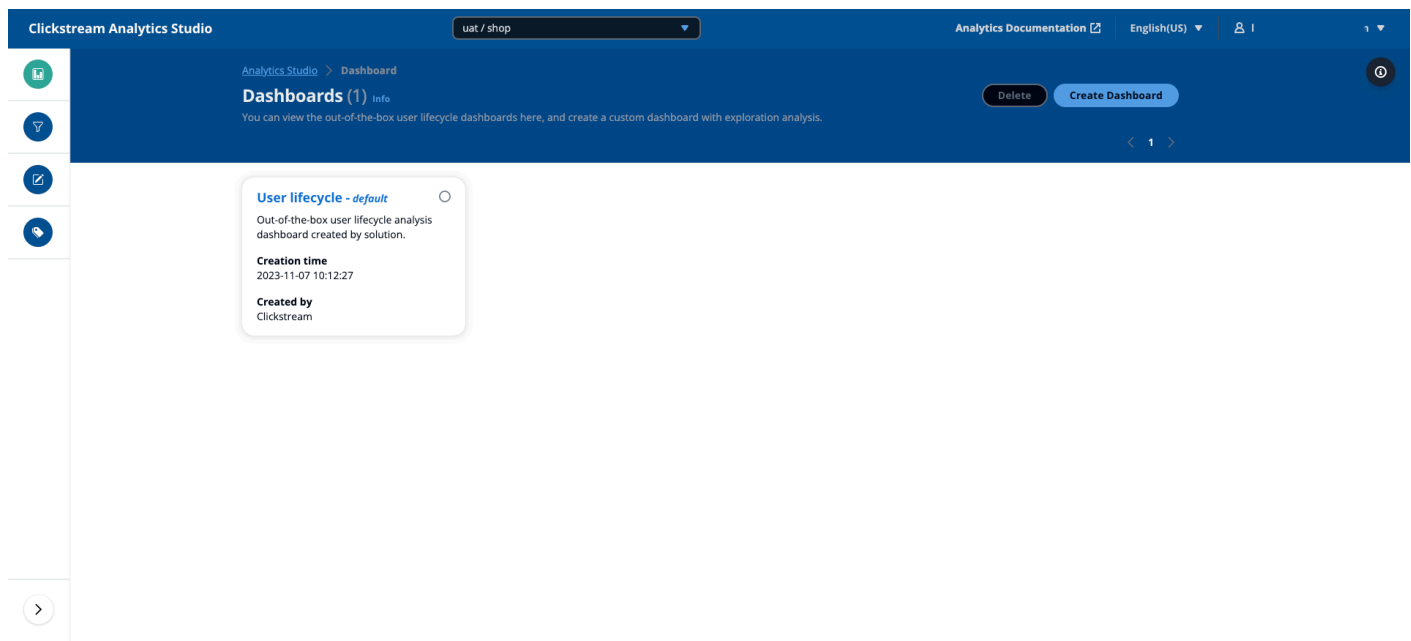
This process will take about 10 minutes with default configuration. After job is finished, you can move to next step.

Step 4: Analyze data

After your application sends data (or the sample data are sent) to the pipeline, you can go into the Analytics Studio to view dashboard and query data.

Steps

1. Log into **Clickstream Analytics on AWS Management Console**.
2. In the left navigation pane, choose **Analytics Studio**, a new tab opens in your browser.
3. In the Analytics Studio page, select the project and app you just created in the drop-down list at the top of the web page.
4. By default, you will be navigated to the **Dashboards** page. If not, choose **Dashboards** from the left navigation pane.



5. Choose **User lifecycle dashboard - default**. You can see the dashboard created by the solution.
6. Choose **Exploration** in the left navigation pane. You can query the clickstream data by using the exploratory analytics models.

Congratulations! You have completed the getting started tutorial. You can explore the Analytic Studio or continue to learn more about this solution later.

Pipeline management

Data pipeline is the core functionality of this solution. In the Clickstream Analytics on AWS solution, we define a data pipeline as a sequence of integrated AWS services that ingest, process, and model the clickstream data into a destination data warehouse for analytics and visualization. It is also designed to efficiently and reliably collect data from your websites and apps to a S3-based data lake, where it can be further processed, analyzed, and utilized for additional use cases (such as real-time monitoring, and recommendation).

To get a basic understanding of the pipeline, you can refer to [terms and concepts](#) for more information.

Prerequisites

You can configure the pipeline in all AWS Regions. For opt-in regions, you need to [enable them](#) first.

Before you start to configure the pipeline in a specific region, make sure you have the following in the target region:

- At least one Amazon VPC.
- At least two public subnets across two AZs in the VPC.
- At least two private (with NAT gateways or instances) subnets across two AZs, or at least two isolated subnets across two AZs in the VPC. If you want to deploy the solution resources in the isolated subnets, you have to create [VPC endpoints](#) for below AWS services,
 - s3, logs, ecr.api, ecr.dkr, ecs, ecs-agent, ecs-telemetry.
 - kinesis-streams if you use KDS as sink buffer in ingestion module.
 - emr-serverless, glue if you enable data processing module.
 - redshift-data, sts, dynamodb, states and lambda if you enable Redshift as analytics engine in data modeling module.
- An Amazon S3 bucket located in the same region.
- If you need to enable Redshift Serverless as analytics engine in data modeling module, you need have subnets across at least three AZs.
- QuickSight Enterprise edition subscription is required if the reporting is enabled.

Ingestion

Ingestion module contains a web service that provides an endpoint to collect data through HTTP/HTTPS requests, which mainly is composed of Amazon Application Load Balancer and Amazon Elastic Container Service. It also supports sinking data into a stream service or S3 directly.

You can create an ingestion module with the following settings:

- [Ingestion endpoint settings](#): Create a web service as an ingestion endpoint to collect data sent from your SDKs.
- Data sink settings: Configure how the solution sinks the data for downstream consumption. Currently, the solution supports three types of data sink:
 - [Apache Kafka](#)
 - [Amazon S3](#)
 - [Amazon Kinesis Data Stream \(KDS\)](#)

Throttle

Currently, there is no built-in throttling feature available with this solution. If needed, you can configure AWS WAF to implement throttling feature. Please refer to [WAF](#) documentation.

Ingestion endpoint

The solution creates a web service as an ingestion endpoint to collect data sent from your SDKs. You can set below configurations for ingestion endpoint.

- **Public subnets**: select at least two existing VPC public subnets, and the Amazon Application Load Balancers (ALBs) will be deployed in these subnets.
- **Private subnets**: select at least two existing VPC private subnets, and the EC2 instances running in ECS will be deployed in these subnets.

Note

The availability zones where the public subnets are located must be consistent with those of the private subnets.

- **Ingestion capacity**: This configuration sets the capacity of the ingestion server, and the ingestion server will automatically scale up or down based on the utilization of the processing CPU.

- **Ingestion Capacity Unit (ICU):** A single Ingestion Compute Unit (ICU) represents billable compute and memory units, approximately 8 gigabytes (GB) of memory and 2 vCPUs. 1 ICU generally can support 4000~6000 requests per second.
- **Minimum capacity:** The minimum capacity to which the ingestion server will scale down.
- **Maximum capacity:** The maximum capacity to which the ingestion server will scale up.
- **Warm pool:** Warm pool gives you the ability to decrease latency for your applications that have exceptionally long boot time. For more information, please refer to [Warm pools for Amazon EC2 Auto Scaling](#).
- **Enable HTTPS:** Users can choose HTTPS/HTTP protocol for the Ingestion endpoint.

⚠ Warning

If you switch between enabling HTTPS and disabling HTTPS, there may be interruptions in the ingestion service.

- **Enable HTTPS:** If users choose to enable HTTPS, the ingestion server will provide HTTPS endpoint.
 - **Domain name:** enter a domain name.

ℹ Note

Once the ingestion server is created, use the custom endpoint to create an alias or CNAME mapping in your Domain Name System (DNS) for the custom endpoint.

- **SSL Certificate:** User need to select an ACM certificate corresponding to the domain name that you input. If there is no ACM certificate, please refer to [create public certificate](#) to create it.
- **Disable HTTPS:** If users choose to disable HTTPS, the ingestion server will provide HTTP endpoint.

⚠ Important

Using HTTP protocol is not secure, because data will be sent without any encryption, and there are high risks of data being leaked or tampered during transmission. Please acknowledge the risk to proceed.

- **Cross-Origin Resource Sharing (CORS):** You can enable CORS to limit requests to data ingestion API from a specific domain. Note that, you need to input a complete internet address, for example, <https://www.example.com>, <http://localhost:8080>. Use comma to separate domain if you have multiple domain for this setting.

⚠ Warning

CORS is a mandatory setting if you are collecting data from a website. If you do not set value for this parameter, the ingestion server to reject all the requests from Web platform.

- **Additional Settings**
 - **Request path:** User can input the path of ingestion endpoint to collect data, the default path is `"/collect"`.
 - **AWS Global Accelerator:** User can choose to create an accelerator to get static IP addresses that act as a global fixed entry point to your ingestion server, which will improves the availability and performance of your ingestion server. Note that additional charges apply.
 - **Authentication:** User can use OIDC provider to authenticate the request sent to your ingestion server. If you plan to enable it, please create an OIDC client in the OIDC provider then create a secret in AWS Secret Manager with information:
 - issuer
 - token endpoint
 - User endpoint
 - Authorization endpoint
 - App client ID
 - App Client Secret

The format is like:

```
{
  "issuer": "xxx",
  "userEndpoint": "xxx",
  "authorizationEndpoint": "xxx",
  "tokenEndpoint": "xxx",
  "appClientId": "xxx",
  "appClientSecret": "xxx"
```

```
}
```

In the OIDC provider, you need to add `https://<ingestion server endpoint>/oauth2/idpresponse` to "Allowed callback URLs".

If you need to obtain the authentication token directly without inputting credential (username/password) manually, you can refer to [alb headless authentication client code](#) to set up your client to obtain the authentication token automatically.

Warning

If you switch between enabling Authentication and disabling Authentication, there may be interruptions in the ingestion service.

- Access logs: ALB supports delivering detailed logs of all requests it receives. If you enable this option, the solution will automatically enable access logs for you and store the logs into the S3 bucket you selected in previous step.

Important

The bucket must have a bucket policy that grants Elastic Load Balancing permission to write the access logs to the bucket. For details, refer to [Step 2: Attach a policy to your S3 bucket](#).

Below is an example policy for the bucket in Regions available before August 2022.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::<elb-account-id>:root"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::<BUCKET>/clickstream/*"
    }
  ]
}
```

You need to replace `e1b-account-id` with the ID of the AWS account for Elastic Load Balancing in your Region:

- US East (N. Virginia) – 127311923021
- US East (Ohio) – 033677994240
- US West (N. California) – 027434742980
- US West (Oregon) – 797873946194
- Africa (Cape Town) – 098369216593
- Asia Pacific (Hong Kong) – 754344448648
- Asia Pacific (Jakarta) – 589379963580
- Asia Pacific (Mumbai) – 718504428378
- Asia Pacific (Osaka) – 383597477331
- Asia Pacific (Seoul) – 600734575887
- Asia Pacific (Singapore) – 114774131450
- Asia Pacific (Sydney) – 783225319266
- Asia Pacific (Tokyo) – 582318560864
- Canada (Central) – 985666609251
- Europe (Frankfurt) – 054676820928
- Europe (Ireland) – 156460612806
- Europe (London) – 652711504416
- Europe (Milan) – 635631232127
- Europe (Paris) – 009996457667
- Europe (Stockholm) – 897822967062
- Middle East (Bahrain) – 076674570225
- South America (São Paulo) – 507241528517
- China (Beijing) – 638102146993
- China (Ningxia) – 037604701340

Data sink – Kafka

This data sink will stream the clickstream data collected by the ingestion endpoint into a topic in a Kafka cluster. Currently, solution support Amazon Managed Streaming for Apache Kafka (Amazon MSK) or a self-hosted Kafka cluster.

Amazon MSK

- **Select an existing Amazon MSK cluster.** Select an MSK cluster from the drop-down list, and the MSK cluster needs to meet the following requirements:
 - MSK cluster and this solution need to be in the same VPC
 - Enable **Unauthenticated access** in Access control methods
 - Enable **Plaintext** in Encryption
 - Set **auto.create.topics.enable** as true in MSK cluster configuration. This configuration sets whether MSK cluster can create topic automatically.
 - The value of **default.replication.factor** cannot be larger than the number of MKS cluster brokers

Note

If there is no MSK cluster, the user needs to create an MSK Cluster following above requirements.

- **Topic:** The user can specify a topic name. By default, the solution will create a topic with “project-id”.

Self-hosted Kafka

Users can also use self-hosted Kafka clusters. To integrate the solution with Kafka clusters, provide the following configurations:

- **Broker link:** Enter the brokers link of Kafka cluster that you wish to connect to. The Kafka cluster needs to meet the following requirements:
 - The Kafka cluster and this solution need to be in the same VPC.
 - At least two Kafka cluster brokers are available.

- **Topic:** User can specify the topic for storing the data
- **Security Group:** This VPC security group defines which subnets and IP ranges can access the Kafka cluster.

Connector

Enable solution to create Kafka connector and a custom plugin for this connector. This connector will sink the data from Kafka cluster to S3 bucket.

Additional Settings

- **Sink maximum interval:** Specifies the maximum length of time (in seconds) that records should be buffered before streaming to the AWS service.
- **Batch size:** The maximum number of records to deliver in a single batch.

Data sink – Kinesis

This data sink will stream the clickstream data collected by the ingestion endpoint into KDS. The solution will create a KDS in your AWS account based on your specifications.

Provision mode

Two modes are available: **On-demand** and **Provisioned**

- **On-demand:** In this mode, KDS shards are provisioned based on the workshop automatically. On-demand mode is suited for workloads with unpredictable and highly-variable traffic patterns.
- **Provisioned:** In this mode, KDS shards are set at creation. The provisioned mode is suited for predictable traffic with capacity requirements that are easy to forecast. You can also use the provisioned mode if you want fine-grained control over how data is distributed across shards.
 - **Shard number:** With the provisioned mode, you must specify the number of shards for the data stream. For more information, please refer to [provisioned mode](#).

Additional settings

- **Sink maximum interval:** You can specify the maximum interval (in seconds) that records should be buffered before streaming to the AWS service.
- **Batch size:** You can specify the maximum number of records to deliver in a single batch.

Data sink - S3

In this option, clickstream data is buffered in the memory of ingestion Server, then sink into a S3 bucket. This option provides the best cost-performance in case real-time data consumption is not required.

Note

Unlike Kafka and KDS data sink, this option buffers data in the ingestion server and responses 200 code to SDK client before sink into S3, so there is chance data could be lost while ingestion server fails and auto-scaled machine is in the process of creation. But it is worth to note that this probability is very low because of the High-availability design of the solution.

- **Buffer size:** Specify the data size to buffer before sending to Amazon S3. The higher buffer size may be lower in cost with higher latency, while the lower buffer size will be faster in delivery with higher cost. Min: 1 MiB, Max: 50 MiB
- **Buffer interval:** Specify the maximum interval (in seconds) for saving buffer to S3. The higher interval allows more time to collect data and the size of data may be bigger. The lower interval sends the data more frequently and may be more advantageous when looking at shorter cycles of data activity. Min: 60 Seconds, Max: 3600 Seconds

Data processing

Clickstream Analytics on AWS provides an inbuilt data schema to parse and model the raw event data sent from your web and mobile apps, which makes it easy for you to analyze the data in analytics engines (such as RedShift and Athena).

Data Processing module includes two functionalities:

- **Transformation:** Extract the data from files sank by ingestion module, then parse each event data and transform them to solution data model.
- **Enrichment:** Add additional dimensions/fields to event data.

This chapter includes:

- [Data schema](#)
- [Configure execution parameters](#)
- [Configure custom plugins](#)

Data schema

This article explains the data schema and format in Clickstream Analytics on AWS. This solution uses an **event-based** data model to store and analyze clickstream data, every activity (e.g., click, view) on the clients is modeled as an event with multiple dimensions. Dimensions are common for all events, but customers have the flexibility to use JSON store value for some dimensions (e.g., event parameters, user attributes) to add information that are specific for their business. Those JSON will be stored in special data types which allow customers to unnest the values in the analytics engines.

Database and table

For each project, the solution creates a database with name of <project-id> in Redshift and Athena. In Redshift, each App will have a schema with name of app_id, within which event data are stored in ods_event tables, user-related attributes are stored in dim_user table. In Athena, data from all apps in the project will be stored in ods_event table with partitions of app_id, year, month, and day.

Columns

Each column in the ods_event table represents an event-specific parameter. Note that some parameters are nested within a Super field in Redshift or Array in Athena, and some fields such as items and event_params are repeatable. Table columns are described below.

Event table fields

Field Name	Data Type - Redshift	Data Type - Athena	Description
event_id	VARCHAR	STRING	Unique ID for the event.
event_date	DATE	DATE	The date when the event was logged

Field Name	Data Type - Redshift	Data Type - Athena	Description
			(YYYYMMDD format in UTC).
event_timestamp	BIGINT	BIGINT	The time (in microseconds, UTC) when the event was logged on the client.
event_previous_timestamp	BIGINT	BIGINT	The time (in microseconds, UTC) when the event was previously logged on the client.
event_name	VARCHAR	STRING	The name of the event.
event_value_in_usd	BIGINT	BIGINT	The currency-converted value (in USD) of the event's "value" parameter.
event_bundle_sequence_id	BIGINT	BIGINT	The sequential ID of the bundle in which these events were uploaded.
ingest_timestamp	BIGINT	BIGINT	Timestamp offset between collection time and upload time in micros.
device.mobile_brand_name	VARCHAR	STRING	The device brand name.
device.mobile_model_name	VARCHAR	STRING	The device model name.

Field Name	Data Type - Redshift	Data Type - Athena	Description
device.manufacturer	VARCHAR	STRING	The device manufacturer name.
device.carrier	VARCHAR	STRING	The device network provider name.
device.network_type	VARCHAR	STRING	The network_type of the device, e.g., WIFI, 5G
device.operating_system	VARCHAR	STRING	The operating system of the device.
device.operating_system_version	VARCHAR	STRING	The OS version.
device.vendor_id	VARCHAR	STRING	IDFV (present only if IDFA is not collected).
device.advertising_id	VARCHAR	STRING	Advertising ID/IDFA.
device.system_language	VARCHAR	STRING	The OS language.
device.time_zone_offset_seconds	BIGINT	BIGINT	The offset from GMT in seconds.
device.ua_browser	VARCHAR	STRING	The browser in which the user viewed content, derived from User Agent string
device.ua_browser_version	VARCHAR	STRING	The version of the browser in which the user viewed content, derive from User Agent

Field Name	Data Type - Redshift	Data Type - Athena	Description
device.ua_device	VARCHAR	STRING	The device in which user viewed content, derive from User Agent.
device.ua_device_category	VARCHAR	STRING	The device category in which user viewed content, derive from User Agent.
device.screen_width	VARCHAR	STRING	The screen width of the device.
device.screen_height	VARCHAR	STRING	The screen height of the device.
geo.continent	VARCHAR	STRING	The continent from which events were reported, based on IP address.
geo.sub_continent	VARCHAR	STRING	The subcontinent from which events were reported, based on IP address.
geo.country	VARCHAR	STRING	The country from which events were reported, based on IP address.
geo.region	VARCHAR	STRING	The region from which events were reported, based on IP address.

Field Name	Data Type - Redshift	Data Type - Athena	Description
geo.metro	VARCHAR	STRING	The metro from which events were reported, based on IP address.
geo.city	VARCHAR	STRING	The city from which events were reported, based on IP address.
geo.locale	VARCHAR	STRING	The locale information obtained from device.
traffic_source.name	VARCHAR	STRING	Name of the marketing campaign that acquired the user when the events were reported.
traffic_source.medium	VARCHAR	STRING	Name of the medium (paid search, organic search, email, etc.) that acquired the user when the events were reported.
traffic_source.source	VARCHAR	STRING	Name of the network source that acquired the user when the event were reported.
app_info.id	VARCHAR	STRING	The package name or bundle ID of the app.

Field Name	Data Type - Redshift	Data Type - Athena	Description
app_info.app_id	VARCHAR	STRING	The App ID (created by this solution) associated with the app.
app_info.install_source	VARCHAR	STRING	The store that installed the app.
app_info.version	VARCHAR	STRING	The app's versionName (Android) or short bundle version.
platform	VARCHAR	STRING	The data stream platform (Web, IOS or Android) from which the event originated.
project_id	VARCHAR	STRING	The project id associated with the app.
items	SUPER	ARRAY	Key-value records contain information about items associated with the event
user_id	VARCHAR	STRING	The unique ID assigned to a user through setUserId () API.
user_pseudo_id	VARCHAR	STRING	The pseudonymous id generated by SDK for the user.

Event parameter table fields

Field Name	Data Type - Redshift	Data Type - Athena	Description
event_timestamp	BIGINT	STRING	The time (in microseconds, UTC) when the event was logged on the client.
event_id	VARCHAR	BIGINT	Unique ID for the event.
event_name	VARCHAR	STRING	The name of the event.
event_params_key	VARCHAR	STRING	The name of the event parameter.
event_params_string_value	VARCHAR	STRING	If the event parameter is represented by a string, such as a URL or campaign name, it is populated in this field.
event_params_int_value	BIGINT	INTEGER	If the event parameter is represented by an integer, it is populated in this field.
event_params_double_value	DOUBLE PRECISION	FLOAT	If the event parameter is represented by a double value, it is

Field Name	Data Type - Redshift	Data Type - Athena	Description
			populated in this field.
event_params_float_value	DOUBLE PRECISION	FLOAT	If the event parameter is represented by a floating point value, it is populated in this field. This field is not currently in use.

User table fields

Field Name	Data Type - Redshift	Data Type - Athena	Description
event_timestamp	BIGINT	STRING	The timestamp of when the user attributes was collected.
user_id	VARCHAR	STRING	The unique ID assigned to a user through setUserId () API.
user_pseudo_id	VARCHAR	STRING	The pseudonymous id generated by SDK for the user.
user_first_touch_timestamp	BIGINT	BIGINT	The time (in microseconds) at which the user first opened the app or visited the site.

Field Name	Data Type - Redshift	Data Type - Athena	Description
user_properties	SUPER	ARRAY	Properties of the user.
user_properties.key	VARCHAR	STRING	The name of the user property.
user_properties.value	SUPER	ARRAY	A record for the user property value.
user_properties.value.string_value	VARCHAR	STRING	The string value of the user property.
user_properties.value.int_value	BIGINT	BIGINT	The integer value of the user property.
user_properties.value.double_value	DOUBLE PRECISION	FLOAT	The double value of the user property.
user_properties.value.float_value	DOUBLE PRECISION	FLOAT	This field is currently unused.
user_ltv	SUPER	ARRAY	The Lifetime Value of the user.
_first_visit_date	Date	Date	Date of the user's first visit
_first_referer	VARCHAR	STRING	The first referer detected for the user
_first_traffic_source_type	VARCHAR	STRING	The the network source that acquired the user that was first detected for the user, e.g., Google, Baidu

Field Name	Data Type - Redshift	Data Type - Athena	Description
_first_traffic_source_medium	VARCHAR	STRING	The medium of the network source that acquired the user that was first detected for the user, e.g., paid search, organic search, email, etc.
_first_traffic_source_name	VARCHAR	STRING	The name of the marketing campaign that acquired the user that was first detected for the user.
device_id_list	SUPER	ARRAY	A record of all device_id associated with the user_pseudo_id
_channel	VARCHAR	STRING	The install channel for the user, e.g., Google Play

Item table fields

Field Name	Data Type - Redshift	Data Type - Athena	Description
event_timestamp	BIGINT	STRING	The timestamp of when the item attributes was collected.
id	VARCHAR	STRING	The id for the item

Field Name	Data Type - Redshift	Data Type - Athena	Description
properties	SUPER	ARRAY	A record for the item property value.
properties.value.string_value	VARCHAR	STRING	The string value of the item property.
properties.value.integer_value	BIGINT	BIGINT	The integer value of the item property.
properties.value.double_value	DOUBLE PRECISION	FLOAT	The double value of the item property.
properties.value.float_value	DOUBLE PRECISION	FLOAT	The float_value of the item property.

Execution parameters

Execution parameters control how the transformation and enrichment jobs are orchestrated.

Parameters

You can configure the following **Execution parameters** after you turn on **Enable data processing**.

Parameter	Description	Values
Data processing interval/ Fixed Rate	Specify the interval to batch the data for data processing by fixed rate	1 hour 12 hours 1 day
Data processing interval/ Cron Expression	Specify the interval to batch the data for data processing by cron expression	cron(0 * * ? *) cron(0 0,12 * ? *) cron(0 0 * ? *)

Parameter	Description	Values
Event freshness	Specify the days after which the solution will ignore the event data. For example, if you specify 3 days for this parameter, the solution will ignore any event which arrived more than 3 days after the events are triggered	3 days 5 days 30 days

Cron expression syntax

Syntax

cron(minutes hours day-of-month month day-of-week year)

For more information, refer to [Cron-based schedules](#).

Config Spark job parameters

By default, the Clickstream pipeline automatically adjusts EMR job parameters based on the dataset volume that requires processing. In most of time, you do not need to adjust the EMR job parameters, but if you want to override the EMR job parameters, you can put spark-config.json file in S3 bucket to set your job parameters.

To add your customized the EMR job parameters, you can add a file s3://{PipelineS3Bucket}/{PipelineS3Prefix}{ProjectId}/config/spark-config.json in the S3 bucket.

Please replace {PipelineS3Bucket}, {PipelineS3Prefix}, and {ProjectId} with the values of your data pipeline. These values are found in the Clickstream-DataProcessing-`<uuid>` stack's Parameters.

Also, you can get these values by running the below commands,

```
stackNames=$(aws cloudformation list-stacks --stack-status-filter CREATE_COMPLETE
UPDATE_COMPLETE --no-paginate | jq -r '.StackSummaries[].StackName' | grep
Clickstream-DataProcessing | grep -v Nested)
```

```
echo -e "$stackNames" | while read stackName; do
  aws cloudformation describe-stacks --stack-name $stackName | jq
  '.Stacks[].Parameters' | jq 'map(select(.ParameterKey == "PipelineS3Bucket"
  or .ParameterKey == "PipelineS3Prefix" or .ParameterKey == "ProjectId"))'
done
```

Here is an example of the file `spark-config.json`:

```
{
  "sparkConfig": [
    "spark.emr-serverless.executor.disk=200g",
    "spark.executor.instances=16",
    "spark.dynamicAllocation.initialExecutors=16",
    "spark.executor.memory=100g",
    "spark.executor.cores=16",
    "spark.network.timeout=10000000",
    "spark.executor.heartbeatInterval=10000000",
    "spark.shuffle.registration.timeout=120000",
    "spark.shuffle.registration.maxAttempts=5",
    "spark.shuffle.file.buffer=2m",
    "spark.shuffle.unsafe.file.output.buffer=1m"
  ],
  "inputRePartitions": 2000
}
```

Please make sure your account has enough `emr-serverless` quotas, you can view the quotas via `emr-serverless-quotas` in the Region `us-east-1`. For more configurations, please refer to [Spark job properties](#) and application [worker config](#).

Processing plugin

There are two types of plugins: **transformer** and **enrichment**. You can choose to have only one **transformer**, and zero or multiple **enrichment**.

Built-in plugins

Below plugins are provided by Clickstream Analytics on AWS.

Plugin name	Type	Description
UAEnrichment	enrichment	User-agent enrichment, use <code>ua_parser</code> Java library

Plugin name	Type	Description
		to enrich User-Agent in the HTTP header to ua_browser,ua_browser_version,ua_os,ua_os_version,ua_device
IpEnrichment	enrichment	IP address enrichment, use GeoLite2 data by MaxMind to enrich IP to city, continent , country

The UAEnrichment uses [UA Parser](#) to parse user-agent in Http header

The IpEnrichment plugin uses [GeoLite2-City data](#) created by MaxMind, available from <https://www.maxmind.com>

Custom plugins

You can add custom plugins to transform raw event data or enrich the data for your need.

Note

To add custom plugins, you must develop your own plugins firstly, see [Develop Custom Plugins](#).

To add your plugins, choose **Add Plugin**, which will open a new window, in which you can upload your plugins.

1. Enter the plugin **Name** and **Description**
2. Choose **Plugin Type**
3. **Enrichment:** Plugin to add fields into event data collected by SDK (both Clickstream SDK or third-party SDK)
4. **Transformation:** A plugin used to transform a third-party SDK's raw data into solution built-in schema
5. Upload plugin java JAR file
6. (Optional) Upload the dependency files if any

7. **Main function class:** fill the full class name of your plugin class name, e.g. `com.company.sol.CustomTransformer`

Develop Custom Plugins

The simplest way to develop custom plugins is making changes based on our example project.

1. Clone/Fork the example project.

```
git clone https://github.com/aws-labs/clickstream-analytics-on-aws.git
cd examples/custom-plugins
```

- For enrichment plugin, please refer to the example: `custom-enrich/`
 - For transformer plugin, please refer to the example: `custom-sdk-transformer/`
2. Change packages and classes name as you desired.
 3. Implement the method `public Dataset<row> transform(Dataset<row> dataset)` to do transformation or enrichment.
 4. (Optional) Write test code.
 5. Run gradle to package code to jar `./gradlew clean build`.
 6. Get the jar file in build output directory `./build/libs/`.

Data modeling

Once the data pipeline processes the event data, you can load the data into an analytics engine for data modeling, such as Redshift or Athena, where data will be aggregated and organized into different views (such as event, device, session), as well as calculated metrics that are commonly used. Below are the preset data views this solution provides if you choose to enable data modeling module.

Preset data views

Data model name	Redshift	Description
<code>clickstream_device_view_v1</code>	Materialized view	A view that contains all device dimensions

Data model name	Redshift	Description
clickstream_event_view_v1	Materialized view	A view that contains all event dimensions
clickstream_event_parameter_view_v1	Materialized view	A view that contains all event dimensions
clickstream_user_dim_view_v1	Materialized view	A view that contains all user dimensions
clickstream_user_attr_view_v1	Materialized view	A view that contains all user custom attributes.
clickstream_session_view_v1	Materialized view	A view that contains all session dimension and relevant metrics, for example, session duration, session views.
clickstream_retention_view_v1	Materialized view	A view that contains metrics of retentions by dates and return days
clickstream_lifecycle_daily_view_v1	Materialized view	A view that contains metrics of user number by lifecycle stages by day, that is, New, Active, Return, Churn.
clickstream_lifecycle_weekly_view_v1	Materialized view	A view that contains metrics of user number by lifecycle stages by week, that is, New, Active, Return, Churn.

You can choose to use Redshift or Athena, or both.

Note

We recommend you select both, that is, using Redshift for hot data modeling and using Athena for all-time data analysis.

You can set below configurations for Redshift.

- **Redshift Mode:** Select Redshift serverless or provisioned mode.
- **Serverless mode**
 - **Base RPU:** RPU stands for Redshift Processing Unit. Amazon Redshift Serverless measures data warehouse capacity in RPUs, which are resources used to handle workloads. The base capacity specifies the base data warehouse capacity Amazon Redshift uses to serve queries and is specified in RPUs. Setting higher base capacity improves query performance, especially for data processing jobs that consume a lot of resources.
 - **VPC:** A virtual private cloud (VPC) based on the Amazon VPC service is your private, logically isolated network in the AWS Cloud.

Note

If you place the cluster within the isolated subnets, the VPC must have VPC endpoints for S3, Logs, Dynamodb, STS, States, Redshift and Redshift-data service.

- **Security Group:** This VPC security group defines which subnets and IP ranges can access the endpoint of Redshift cluster.
- **Subnets:** Select at least three existing VPC subnets.

Note

We recommend using private subnets to deploy in accordance with the security best practices.

- **Provisioned mode**
 - **Redshift Cluster:** With a provisioned Amazon Redshift cluster, you build a cluster with node types that meet your cost and performance specifications. You have to set up, tune, and manage Amazon Redshift provisioned clusters.

- **Database user:** The solution requires permissions to access and create database in Redshift cluster. By default, it grants Redshift Data API with the permissions of the admin user to execute the commands to create DB, tables, and views, as well as loading data.
- **Data range:** Considering the cost performance issue of having Redshift to save all the data, we recommend that Redshift save hot data and that all data are stored in S3. It is necessary to delete expired data in Redshift on a regular basis.
- **Athena:** Choose Athena to query all data on S3 using the table created in the AWS Glue Data Catalog.

Reporting

Once the data are processed and modeled by the data pipeline, you can enable the Analytics Studio for the pipeline, which will allow the solution create out-of-the-box dashboards in QuickSight, provide advanced analytics model for user to query their clickstream data, and data management functionalities.

Note

To enable this module, your AWS account needs to have subscription in QuickSight. If it hasn't, please follow this [sign up for Amazon QuickSight](#) to create a subscription first.

You need to make the following configuration for Reporting.

- **Create sample dashboard in QuickSight:** Enabling this feature allows the solution to create sample dashboards in your QuickSight account.
- **QuickSight user:** Select an admin user for the solution to create QuickSight resources. (Only required for AWS China Regions)

Pipeline maintenance

This solution provides three features to help you manage and operate the data pipeline after it gets created.

Monitoring and Alarms

The solution collects metrics from each resource in the data pipeline and creates monitoring dashboards in CloudWatch, which provides you a comprehensive view into the pipeline status. It also provides a set of alarms that will notify project owner if anything goes abnormal.

Following are steps to view monitoring dashboards and alarms.

Monitoring dashboards

To view monitoring dashboard for a data pipeline, follows below steps:

1. Go to project detail page.
2. Choose project id or **View Details**, which will direct to the pipeline detail page.
3. Select the "**Monitoring**" tab.
4. In the tab, choose **View in CloudWatch**, which will direct you to the monitoring dashboard.

Alarms

To view alarms for a data pipeline, follows below steps:

1. Go to project detail page.
2. Choose project id or **View Details**, which will direct to the pipeline detail page.
3. Select the "**Alarms**" tab.
4. In the tab, you can view all the alarms. You can also choose **View in CloudWatch**, which will direct you to CloudWatch alarm pages to view alarm details.
5. You can also enable or disable an alarm by selecting the alarm then choosing **Enable** or **Disable**.

Pipeline modification

You are able to modify some configuration the data pipeline after it created, follow below steps to update a pipeline.

1. Go to project detail page.
2. Choose project id or **View Details**, which will direct to the pipeline detail page.

3. In the project details page, choose **Edit**, which will bring you to the pipeline creation wizard page. Note that some configuration are in disable mode, which means they cannot be updated after creation.
4. If needed, update those configuration options which are editable.
5. After editing the configuration, choose **Next** until you reach last page, and choose **Save**.

You will see pipeline is in Updating status.

Pipeline upgrade

For detailed procedure, see [upgrade the solution](#).

SDK manual

Clickstream Analytics on AWS provides purpose-built software development kits (SDKs), which can make it easier for you to report events to the data pipeline created in the solution. Currently, the solution supports the following platforms:

- [Android](#)
- [Swift](#)
- [Web](#)
- [Flutter](#)

In addition, you can use HTTP API to collect clickstream data from other platforms through HTTP request.

Key features and benefits

- **Automatic data collection.** Clickstream SDKs provide built-in capabilities to automatically collect common events, such as screen view, session, and user engagement, so that you only need to focus on recording business-specific events.
- **Ease of use.** Clickstream SDKs provide multiple APIs and configuration options to simplify the event reporting and attribute setting process.
- **Cross-platform analytics.** Clickstream SDKs are consistent in event data structure, attribute validation rules, and event sending mechanism, so that data can be normalized in the same structure for cross-platform analytics.

Note

All Clickstream SDKs are open source under Apache 2.0 License in [Github](#). You can customize the SDKs if needed. All contributions are welcome.

Android SDK

Introduction

Clickstream Android SDK can help you easily collect and report in-app events from Android devices to AWS. As part of the solution Clickstream Analytics on AWS, this SDK provisions data pipeline to ingest and process event data into AWS services such as Amazon S3, and Amazon Redshift.

The SDK is based on the Amplify for Android SDK Core Library and developed according to the Amplify Android SDK plug-in specification. In addition, the SDK is equipped with features that automatically collect common user events and attributes (for example, screen view, and first open) to simplify data collection for users.

Platform support

Clickstream Android SDK supports Android 4.1 (API level 16) and later.

Integrate the SDK

1 Include the SDK

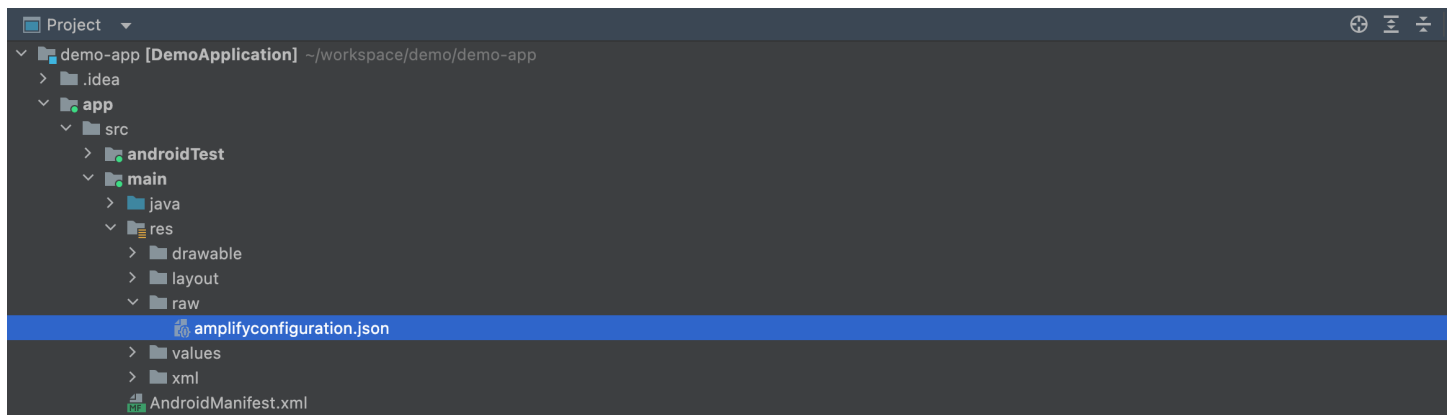
Add the Clickstream SDK dependency to your app module's `build.gradle` file. For example,

```
dependencies {  
    implementation 'software.aws.solution:clickstream:0.10.1'  
}
```

If needed, you can synchronize your project with [the latest version](#).

2 Configure parameters

Find the `res` directory under your `project/app/src/main`, and manually create a `raw` folder in the `res` directory.



Download your `amplifyconfiguration.json` file from your clickstream control plane, and paste it to the raw folder. The JSON file is like:

```
{
  "analytics": {
    "plugins": {
      "awsClickstreamPlugin": {
        "appId": "appId",
        "endpoint": "https://example.com/collect",
        "isCompressEvents": true,
        "autoFlushEventsInterval": 10000,
        "isTrackAppExceptionEvents": false
      }
    }
  }
}
```

In the file, your `appId` and `endpoint` are already configured. The explanation for each property is as follows:

- **appId:** the app id of your project in control plane.
- **endpoint:** the endpoint url you will upload the event to AWS server.
- **isCompressEvents:** whether to compress event content when uploading events, and the default value is true
- **autoFlushEventsInterval:** event sending interval, and the default value is 10s
- **isTrackAppExceptionEvents:** whether auto track exception event in app, and the default value is false

3 Initialize the SDK

Initialize the SDK in the application onCreate() method.

```
import software.aws.solution.clickstream.ClickstreamAnalytics;

public void onCreate() {
    super.onCreate();

    try{
        ClickstreamAnalytics.init(this);
        Log.i("MyApp", "Initialized ClickstreamAnalytics");
    } catch (AmplifyException error){
        Log.e("MyApp", "Could not initialize ClickstreamAnalytics", error);
    }
}
```

4 Start using

Record event

Add the following code where you need to report an event.

```
import software.aws.solution.clickstream.ClickstreamAnalytics;
import software.aws.solution.clickstream.ClickstreamEvent;

// for record an event with custom attributes
ClickstreamEvent event = ClickstreamEvent.builder()
    .name("button_click")
    .add("category", "shoes")
    .add("currency", "CNY")
    .add("value", 279.9)
    .build();
ClickstreamAnalytics.recordEvent(event);

// for record an event directly
ClickstreamAnalytics.recordEvent("button_click");
```

Add global attribute

```
import software.aws.solution.clickstream.ClickstreamAttribute;
```

```
import software.aws.solution.clickstream.ClickstreamAnalytics;

ClickstreamAttribute globalAttribute = ClickstreamAttribute.builder()
    .add("channel", "Play Store")
    .add("level", 5.1)
    .add("class", 6)
    .add("isOpenNotification", true)
    .build();
ClickstreamAnalytics.addGlobalAttributes(globalAttribute);

// for delete an global attribute
ClickstreamAnalytics.deleteGlobalAttributes("level");
```

Please add the global attribute after the SDK initialization is completed, the global attribute will be added to the attribute object in all events.

Login and logout

```
import software.aws.solution.clickstream.ClickstreamAnalytics;

// when user login success
ClickstreamAnalytics.setUserId("UserId");

// when user logout
ClickstreamAnalytics.setUserId(null);
```

Add user attribute

```
import software.aws.solution.clickstream.ClickstreamAnalytics;
import software.aws.solution.clickstream.ClickstreamUserAttribute;

ClickstreamUserAttribute clickstreamUserAttribute = ClickstreamUserAttribute.builder()
    .add("_user_age", 21)
    .add("_user_name", "carl")
    .build();
ClickstreamAnalytics.addUserAttributes(clickstreamUserAttribute);
```

Current login user's attributes will be cached in disk, so the next time app launch you don't need to set all user's attribute again, of course you can use the same api `ClickstreamAnalytics.addUserAttributes()` to update the current user's attribute when it changes.

⚠ Important

If your application is already published and most users have already logged in, please manually set the user attributes once when integrate the Clickstream SDK for the first time to ensure that subsequent events contains user attributes.

Record event with items

You can add the following code to log an event with an item.

```
import software.aws.solution.clickstream.ClickstreamAnalytics;
import software.aws.solution.clickstream.ClickstreamItem;

ClickstreamItem item_book = ClickstreamItem.builder()
    .add(ClickstreamAnalytics.Item.ITEM_ID, "123")
    .add(ClickstreamAnalytics.Item.ITEM_NAME, "Nature")
    .add(ClickstreamAnalytics.Item.ITEM_CATEGORY, "book")
    .add(ClickstreamAnalytics.Item.PRICE, 99)
    .add("book_publisher", "Nature Research")
    .build();

ClickstreamEvent event = ClickstreamEvent.builder()
    .name("view_item")
    .add(ClickstreamAnalytics.Item.ITEM_ID, "123")
    .add(ClickstreamAnalytics.Item.CURRENCY, "USD")
    .add("event_category", "recommended")
    .setItems(new ClickstreamItem[] {item_book})
    .build();

ClickstreamAnalytics.recordEvent(event);
```

For more information about logging more attribute in an item, please refer to Item attributes.

⚠ Important

Only pipelines from version 1.1.0 can handle items with custom attribute.

Send event immediately

```
// for send event immediately.  
ClickstreamAnalytics.flushEvent();
```

Disable SDK

You can disable the SDK if needed. After disabling the SDK, the SDK will not handle the logging and sending of any events. Of course you can enable the SDK when you need to continue logging events.

Please note that the disable and enable code needs to be run in the main thread.

```
import software.aws.solution.clickstream.ClickstreamAnalytics;  
  
// disable SDK  
ClickstreamAnalytics.disable();  
  
// enable SDK  
ClickstreamAnalytics.enable();
```

Configuration update

After initializing the SDK, you can use the following code to customize the configuration of the SDK.

Important

This configuration will override the default configuration in `amplifyconfiguration.json` file.

```
import software.aws.solution.clickstream.ClickstreamAnalytics;  
  
// config the SDK after initialize.  
ClickstreamAnalytics.getClickStreamConfiguration()  
    .withAppId("your appId")  
    .withEndpoint("https://example.com/collect")  
    .withAuthCookie("your authentication cookie")  
    .withSendEventsInterval(10000)  
    .withSessionTimeoutDuration(1800000)  
    .withTrackScreenViewEvents(false)
```

```

.withTrackUserEngagementEvents(false)
.withTrackAppExceptionEvents(false)
.withLogEvents(true)
.withCustomDns(CustomOkhttpDns.getInstance())
.withCompressEvents(true);

```

Here is an explanation of each method.

Method name	Parameter type	Required	Default value	Description
withAppId()	String	true	N/A	the app id of your application in web console
withEndpoint()	String	true	N/A	the endpoint path you will upload the event to Clickstream ingestion server
withAuthCookie()	String	false	N/A	your auth cookie for AWS application load balancer auth cookie
withSendEventsInterval()	long	false	1800000	event sending interval in milliseconds
withSessionTimeoutDuration()	long	false	5000	the duration of the session timeout in milliseconds
withTrackScreenViewEvents()	boolean	false	true	whether to auto-record screen view events

Method name	Parameter type	Required	Default value	Description
withTrackUserEngagementEvents()	boolean	false	true	whether to auto-record user engagement events
withTrackAppExceptionEvents()	boolean	false	true	whether to auto-record app exception events
withLogEvents()	boolean	false	false	whether to automatically print event JSON for debugging events
withCustomDns()	String	false	N/A	the method for setting your custom DNS
withCompressEvents()	boolean	false	true	whether to compress event content by gzip when uploading events

Debug events

You can follow the steps below to view the event raw JSON and debug your events.

1. Use `ClickstreamAnalytics.getClickStreamConfiguration()` API and set the `withLogEvents()` method with `true` in debug mode, for example:

```
import software.aws.solution.clickstream.ClickstreamAnalytics;

// log the event in debug mode.
```



```
ClickstreamAnalytics.getClickStreamConfiguration()
    .withLogEvents(BuildConfig.DEBUG);
```

2. Integrate the SDK and launch your app by Android Studio, then open the **Logcat** window.
3. Input EventRecorder to the filter, and you will see the JSON content of all events recorded by Clickstream Android SDK.

Configure custom DNS

```
import software.aws.solution.clickstream.ClickstreamAnalytics;

// config custom DNS.
ClickstreamAnalytics.getClickStreamConfiguration()
    .withCustomDns(CustomOkhttpDns.getInstance());
```

If you want to use custom DNS for network request, you can create your CustomOkhttpDns which implementation okhttp3.Dns, then configure `.withCustomDns(CustomOkhttpDns.getInstance())` to make it work. Here is an [example code](#).

Data format definition

Data types

Clickstream Android SDK supports the following data types:

Data type	Range	Example
int	-214748364 ~ 2147483647	12
long	-9223372036854775808 ~ 9223372036854775807	26854775808
double	4.9E-324 ~ 1.7976931 348623157E308	3.14
boolean	true, false	true
String	1024 characters maximum	"Clickstream"

Naming rules

1. The event name and attribute name cannot start with a number, and only contain uppercase and lowercase letters, numbers, and underscores. In case of an invalid event name, it will throw `IllegalArgumentException`. In case of an invalid attribute name or user attribute name, it will discard the attribute and record error.
2. Do not use `_` as prefix in an event name or attribute name, because the `_` prefix is reserved for the solution.
3. The event name and attribute name are case sensitive, so `Add_to_cart` and `add_to_cart` will be recognized as two different event names.

Event and attribute limitation

To improve the efficiency of querying and analysis, we apply limitations to event data as follows:

Item	Recommended	Maximum	Strategy	Error code
Event name invalid	N/A	N/A	discard event, print log and record <code>_clickstream_error</code> event	1001
Length of event name	less than 25 characters	50 characters	discard event, print log and record <code>_clickstream_error</code> event	1002
Length of event attribute name	less than 25 characters	50 characters	discard the attribute, print log and record error in event attribute	2001
Attribute name invalid	N/A	N/A	discard the attribute, print log and record	2002

Item	Recommended	Maximum	Strategy	Error code
			error in event attribute	
Length of event attribute value	less than 100 characters	1024 characters	discard the attribute, print log and record error in event attribute	2003
Event attribute per event	less than 50 attributes	500 event attributes	discard the attribute, print log and record error in event attribute	2004
User attribute number	less than 25 attributes	100 user attributes	discard the attribute , print log and record_clickstream_error event	3001
Length of user attribute name	less than 25 characters	50 characters	discard the attribute , print log and record_clickstream_error event	3002
User attribute name invalid	N/A	N/A	discard the attribute , print log and record_clickstream_error event	3003

Item	Recommended	Maximum	Strategy	Error code
Length of user attribute value	less than 50 characters	256 characters	discard the attribute , print log and record <code>_clickstream_error</code> event	3004

Important

- The character limits are the same for single-width character languages (for example, English) and double-width character languages (for example, Chinese).
- The limitation for event attribute per event involves both common attributes and preset attributes.
- If the attribute or user attribute with the same name is added more than twice, the latest value will apply.

Preset events

Automatically collected events

Event name	Triggered	Event attributes
<code>_first_open</code>	when the user launches an app the first time after installation	N/A
<code>_session_start</code>	when a user first open the app or a user returns to the app after 30 minutes of inactivity period	<code>_session_id</code> <code>_session_start_timestamp</code>
<code>_screen_view</code>	when a new screen opens	<code>_screen_name</code>

Event name	Triggered	Event attributes
		_screen_id _screen_unique_id _previous_screen_name _previous_screen_id _previous_screen_unique_id _entrances _previous_timestamp _engagement_time_msec
_user_engagement	when a user navigates away from current screen and the screen is in focus for at least one second	_engagement_time_msec
_app_start	every time the app goes to visible	_is_first_time (when it is the first _app_start event after the application starts, the value is true)
_app_end	every time the app goes to invisible	N/A
_profile_set	when the addUserAttributes() or setUserId() API is called	N/A
_app_exception	when the app crashes	_exception_message _exception_stack

Event name	Triggered	Event attributes
<code>_app_update</code>	when the app is updated to a new version and launched again	<code>_previous_app_version</code>
<code>_os_update</code>	when device operating system is updated to a new version	<code>_previous_os_version</code>
<code>_clickstream_error</code>	event_name is invalid or user attribute is invalid	<code>_error_code</code> <code>_error_message</code>

Session definition

In Clickstream Android SDK, we do not limit the total time of a session. As long as the time between the next entry of the app and the last exit time is within the allowable timeout period, the current session is considered to be continuous.

The `_session_start` event triggered when the app open for the first time, or the app was open to the foreground and the time between the last exit exceeded `session_time_out` period. The following are session-related attributes.

- **`_session_id`:** It is calculated by concatenating the last 8 characters of `uniqueid` and the current millisecond, for example: `dc7a7a18-20230905-131926703`.
- **`_session_duration`:** We calculate the `_session_duration` by minus the current event create timestamp and the session's `_session_start_timestamp`. This attribute will be added in every event during the session.
- **`_session_number`:** The auto increment number of session in current device, and the initial value is 1.
- **Session timeout duration:** By default, it is 30 minutes, which can be customized through the configuraton update API.

Screen view definition

In Clickstream Android SDK, we define the `_screen_view` as an event that records a user's browsing path of screen. When a screen transition started, the `_screen_view` event will be recorded if any of the following conditions is met:

- No screen was previously set.
- The new screen name differs from the previous screen title.
- The new screen id differs from the previous screen id.
- The new screen unique id differs from the previous screen unique id.

This event listens for Activity's `onResume` lifecycle method to judgment the screen transition. In order to track screen browsing path, we use `_previous_screen_name` , `_previous_screen_id` and `_previous_screen_unique_id` to link the previous screen. In addition, there are some other attributes in screen view event.

- `_screen_unique_id`: We calculate the screen unique id by getting the current screen's hashcode, for example: "126861252".
- `_entrances`: The first screen view event in a session is 1, others is 0.
- `_previous_timestamp`: The timestamp of the previous `_screen_view` event.
- `_engagement_time_msec`: The previous page last engagement milliseconds.

User engagement definition

In Clickstream Android SDK, we define the `user_engagement` as an event that records the screen browsing time, and we only send this event when user leave the screen and the screen has focus for at least one second.

We define that users leave the screen in the following situations.

- When the user navigates to another screen.
- The user moves the app screen to the background.
- The user exit the app, or kill the process of app.

`engagement_time_msec`: We calculate the milliseconds from when a screen is visible to when the user leave the screen.

Event attributes

Sample event structure

```
{
  "hashCode": "80452b0",
  "unique_id": "c84ad28d-16a8-4af4-a331-f34cdc7a7a18",
  "event_type": "add_to_cart",
  "event_id": "460daa08-0717-4385-8f2e-acb5bd019ee7",
  "timestamp": 1667877566697,
  "device_id": "f24bec657ea8eff7",
  "platform": "Android",
  "os_version": "10",
  "make": "Samsung",
  "brand": "Samsung",
  "model": "TAS-AN00",
  "locale": "zh_CN_#Hans",
  "carrier": "CDMA",
  "network_type": "Mobile",
  "screen_height": 2259,
  "screen_width": 1080,
  "zone_offset": 28800000,
  "system_language": "zh",
  "country_code": "CN",
  "sdk_version": "0.7.1",
  "sdk_name": "aws-solution-clickstream-sdk",
  "app_version": "1.0",
  "app_package_name": "com.notepad.app",
  "app_title": "Notepad",
  "app_id": "notepad-4a929eb9",
  "user": {
    "_user_id": {
      "value": "312121",
      "set_timestamp": 1667877566697
    },
    "_user_name": {
      "value": "carl",
      "set_timestamp": 1667877566697
    },
    "_user_first_touch_timestamp": {
      "value": 1667877267895,
      "set_timestamp": 1667877566697
    }
  }
},
```



```

    "attributes": {
      "event_category": "recommended",
      "currency": "CNY",
      "_session_id": "dc7a7a18-20221108-031926703",
      "_session_start_timestamp": 1667877566703,
      "_session_duration": 391809,
      "_session_number": 1,
      "_screen_name": "ProductDetailActivity",
      "_screen_unique_id": "126861252"
    }
  }
}

```

All user attributes will be stored in user object, and all custom and global attributes in attributes object.

Common attribute

Attribute name	Description	It is generated...	It is used to or for...
hashCode	the event object's hash code	generated from <code>Integer.toHexString(AnalyticsEvent.hashCode())</code>	distinguish events
app_id	clickstream app id	generated when clickstream app create from solution control plane	identify the events for your apps
unique_id	the unique id for user	generated from <code>UUID.randomUUID().toString()</code> during the SDK first initialization. It will be changed after user relogin to	identity different users and associate the behavior of logging in and not logging in

Attribute name	Description	It is generated...	It is used to or for...
		another user who never login, and when user relogin to the previous user in same device. The unique_id will reset to the previous user's unique_id.	
device_id	the unique id for device	generated from Settings. System.getString(context.getContentResolver(), Settings.Secure.ANDROID_ID) . If Android ID is null or "", we will use UUID instead.	distinguish different devices
event_type	event name	set by developer or SDK	distinguish different event types
event_id	the unique id for event	generated from UUID.randomUUID().toString() when the event is created	distinguish each event

Attribute name	Description	It is generated...	It is used to or for...
timestamp	event create timestamp	generated from <code>System.currentTimeMillis()</code> when the event is created	data analysis needs
platform	the platform name	for Android device, it is always "Android"	data analysis needs
os_version	the platform version code	generated from <code>Build.VERSION.RELEASE</code>	data analysis needs
make	the manufacturer of the device	generated from <code>Build.MANUFACTURER</code>	data analysis needs
brand	the brand of the device	generated from <code>Build.BRAND</code>	data analysis needs
model	the model of the device	generated from <code>Build.MODEL</code>	data analysis needs
carrier	the device network operator name	generated from <code>TelephonyManager.getNetworkOperatorName()</code> , the default is "UNKNOWN"	data analysis needs

Attribute name	Description	It is generated...	It is used to or for...
network_type	the current device network type	generated from <code>android.net.ConnectivityManager</code> , it can be "Mobile", "WIFI" or "UNKNOWN"	data analysis needs
screen_height	the absolute height of the available display size in pixels	generated from <code>applicationContext.getResources().displayMetrics.heightPixels</code>	data analysis needs
screen_width	the absolute width of the available display size in pixels	generated from <code>applicationContext.getResources().displayMetrics.widthPixels</code>	data analysis needs
zone_offset	the device raw offset from GMT in milliseconds	generated from <code>java.util.Calendar.get(Calendar.ZONE_OFFSET)</code>	data analysis needs

Attribute name	Description	It is generated...	It is used to or for...
locale	the default locale (language, country and variant) for this device of the Java Virtual Machine	generated from <code>java.util.Locale.getDefault()</code>	data analysis needs
system_language	the device language code	generated from <code>java.util.Locale.getLanguage()</code> , and its default is value "UNKNOWN"	data analysis needs
country_code	country/region code for this device	generated from <code>java.util.Locale.getCountry()</code> and its default value is "UNKNOWN"	data analysis needs
sdk_version	clickstream SDK version	generated from <code>BuildConfig.VERSION_NAME</code>	data analysis needs
sdk_name	clickstream SDK name	it is always "aws-solution-clickstream-sdk"	data analysis needs

Attribute name	Description	It is generated...	It is used to or for...
app_version	the app version name of user's app	generated from <code>android.content.pm.PackageManager.getVersionName</code> , and its default value is "UNKNOWN"	data analysis needs
app_package_name	the app package name of user's app	generated from <code>android.content.pm.PackageManager.getPackageName</code> , and its default value is "UNKNOWN"	data analysis needs
app_title	the display name of user's app	generated from <code>android.content.pm.ApplicationLabel(appInfo)</code>	data analysis needs

User attributes

Attribute name	Description
_user_id	Reserved for user id that is assigned by app
_user_ltv_revenue	Reserved for user lifetime value
_user_ltv_currency	Reserved for user lifetime value currency

Attribute name	Description
_user_first_touch_timestam	The time (in microseconds) when the user first opened the app or visited the site, and it is included in every event in user object

Event attributes

Attribute name	Data type	Auto track	Description
_traffic_source_medium	String	false	Reserved for traffic medium. Use this attribute to store the medium that acquired user when events were logged.
_traffic_source_name	String	false	Reserved for traffic name. Use this attribute to store the marketing campaign that acquired user when events were logged.
_traffic_source_source	String	false	Reserved for traffic source. Name of the network source that acquired the user when the event were reported.
_channel	String	false	The channel for app was downloaded
_session_id	String	true	Added in all events.

Attribute name	Data type	Auto track	Description
_session_start_timestamp	long	true	Added in all events.
_session_duration	long	true	Added in all events.
_session_number	int	true	Added in all events.
_screen_name	String	true	Added in all events.
_screen_unique_id	String	true	Added in all events.

Swift SDK

Introduction

Clickstream Swift SDK can help you easily collect and report in-app events from iOS devices to AWS. As part of the solution Clickstream Analytics on AWS, this SDK provisions data pipeline to ingest and process event data into AWS services such as Amazon S3, and Amazon Redshift.

The SDK is based on the Amplify for Swift Core Library and developed according to the Amplify Swift SDK plug-in specification. In addition, the SDK is equipped with features that automatically collect common user events and attributes (for example, screen view, first open) to simplify data collection for users.

Platform Support

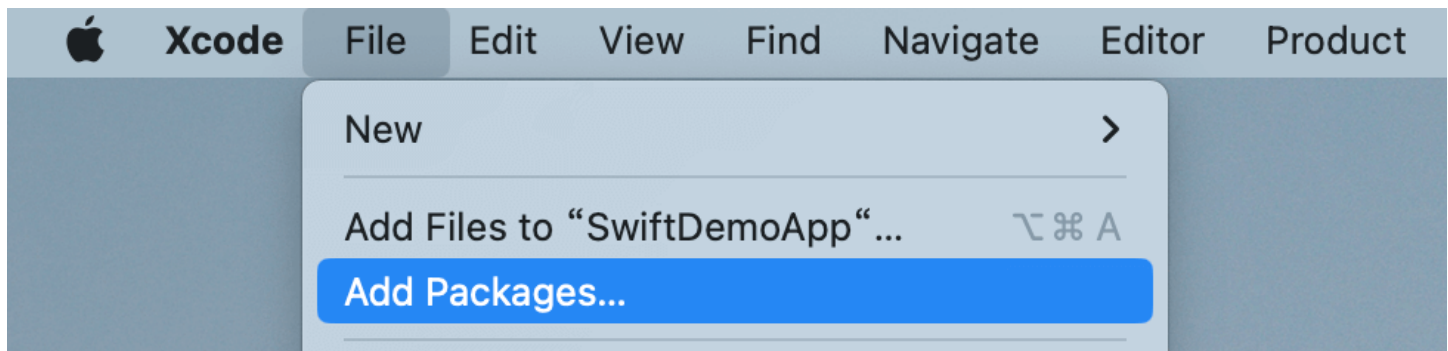
Clickstream Swift SDK supports iOS 13+.

Clickstream Swift SDK requires Xcode 13.4 or higher to build.

Integrate the SDK

1 Add Package

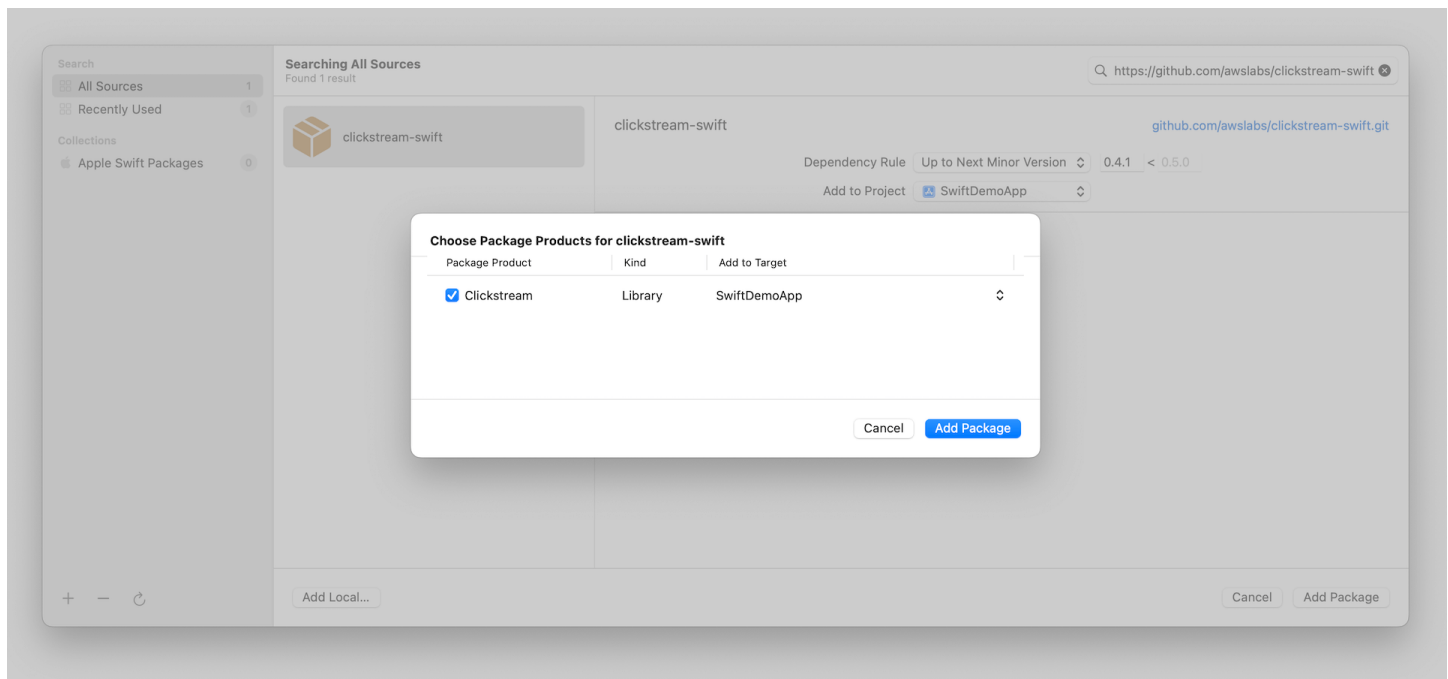
The solution uses Swift Package Manager to distribute Clickstream Swift SDK. Open your project in Xcode and select **File > Add Packages**.



1. Copy the Clickstream Swift SDK GitHub repository URL and paste it into the search bar.

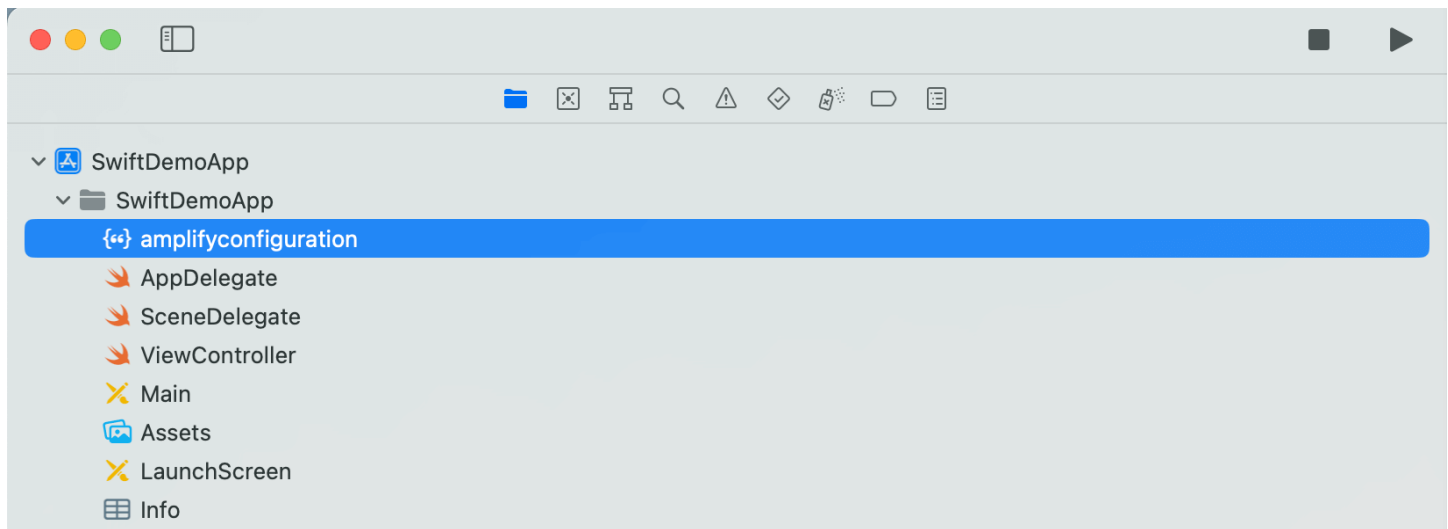
```
https://github.com/awslabs/clickstream-swift
```

2. Check the rules for the version of the SDK that you want Swift Package Manager to install, it is recommended to choose **Up to Next Major Version**, then click **Add Package**.
3. Keep the Clickstream product checked as default.
4. Choose **Add Package** again to finish the package installation.



2 Parameter configuration

Download your `amplifyconfiguration.json` file from your Clickstream solution control plane. Copy and paste it to your project root folder:



The JSON file will be as follows:

```
{
  "analytics": {
    "plugins": {
      "awsClickstreamPlugin ": {
        "appId": "appId",
        "endpoint": "https://example.com/collect",
        "isCompressEvents": true,
        "autoFlushEventsInterval": 10000,
        "isTrackAppExceptionEvents": false
      }
    }
  }
}
```

In the file, your appId and endpoint are already configured. The explanation for each property is as follows:

- **appId:** the app id of your project in control plane
- **endpoint:** the endpoint url you will upload the event to AWS server
- **isCompressEvents:** whether to compress event content when uploading events, and the default value is true
- **autoFlushEventsInterval:** event sending interval, and the default value is 10s
- **isTrackAppExceptionEvents:** whether auto track exception event in app, and the default value is false

3 Initialize the SDK

Once you have configured the parameters, you need to initialize it in AppDelegate's `didFinishLaunchingWithOptions` lifecycle method to use the SDK.

Swift

```
import Clickstream
...
func application(_ application: UIApplication, didFinishLaunchingWithOptions
  launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
  // Override point for customization after application launch.
  do {
    try ClickstreamAnalytics.initSDK()
  } catch {
    assertionFailure("Fail to initialize ClickstreamAnalytics: \(error)")
  }
  return true
}
```

Objective-C

```
@import Clickstream;

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:
(NSDictionary *)launchOptions {
    NSError *error = nil;
    [ClickstreamObjc initSDKAndReturnError:&error];
    if (error) {
        NSLog(@"Fail to initialize ClickstreamAnalytics: %@",
error.localizedDescription);
    }
    return YES;
}
```

If your project is developed with SwiftUI, you need to create an application delegate and attach it to your App through `UIApplicationDelegateAdaptor`.

```
@main
struct YourApp: App {
    @UIApplicationDelegateAdaptor(AppDelegate.self) var appDelegate
    var body: some Scene {
```

```
        WindowGroup {
            YourView()
        }
    }
}
```

For SwiftUI, we do not yet support automatic collection of screen view events. You need to disable screen view event by setting `configuration.isTrackScreenViewEvents = false`.

5 Start using

Record event

Add the following code where you need to report an event.

```
import Clickstream

// for record an event with custom attributes
let attributes: ClickstreamAttribute = [
    "category": "shoes",
    "currency": "CNY",
    "value": 279.9
]
ClickstreamAnalytics.recordEvent("button_click", attributes)

// for record an event directly
ClickstreamAnalytics.recordEvent("button_click")
```

Add global attribute

```
import Clickstream

let globalAttribute: ClickstreamAttribute = [
    "channel": "apple",
    "class": 6,
    "level": 5.1,
    "isOpenNotification": true,
]
ClickstreamAnalytics.addGlobalAttributes(globalAttribute)

// for delete an global attribute
ClickstreamAnalytics.deleteGlobalAttributes("level")
```

Please add the global attribute after the SDK initialization is completed, the global attribute will be added to the attribute object in all events.

Login and logout

```
import Clickstream

// when user login usccess.
ClickstreamAnalytics.setUserId("userId")

// when user logout
ClickstreamAnalytics.setUserId(nil)
```

Add user attribute

```
import Clickstream

let userAttributes : ClickstreamAttribute=[
    "_user_age": 21,
    "_user_name": "carl"
]
ClickstreamAnalytics.addUserAttributes(userAttributes)
```

Current login user's attributes will be cached in disk, so the next time app launch you don't need to set all user's attribute again, of course you can use the same API `ClickstreamAnalytics.addUserAttributes()` to update the current user's attribute when it changes.

Important

If your application is already published and most users have already logged in, please manually set the user attributes once when integrate the Clickstream SDK for the first time to ensure that subsequent events contains user attributes.

Record event with items

You can add the following code to log an event with an item.

```
import Clickstream
```

```
let attributes: ClickstreamAttribute = [  
    ClickstreamAnalytics.Item.ITEM_ID: "123",  
    ClickstreamAnalytics.Item.CURRENCY: "USD",  
    "event_category": "recommended"  
]  
  
let item_book: ClickstreamAttribute = [  
    ClickstreamAnalytics.Item.ITEM_ID: 123,  
    ClickstreamAnalytics.Item.ITEM_NAME: "Nature",  
    ClickstreamAnalytics.Item.ITEM_CATEGORY: "book",  
    ClickstreamAnalytics.Item.PRICE: 99.9,  
    "book_publisher": "Nature Research"  
]  
ClickstreamAnalytics.recordEvent("view_item", attributes, [item_book])
```

For more information about logging more attributes in an item, refer to [item attributes](#).

Important

Only pipelines from version 1.1.0 can handle items with custom attribute.

Send event immediately

```
import Clickstream  
// for send event immediately.  
ClickstreamAnalytics.flushEvents()
```

Disable SDK

You can disable the SDK if needed. After disabling the SDK, the SDK will not handle the logging and sending of any events. Of course you can enable the SDK when you need to continue logging events.

```
import Clickstream  
  
// disable SDK  
ClickstreamAnalytics.disable()  
  
// enable SDK
```

```
ClickstreamAnalytics.enable()
```

Configuration update

After initializing the SDK, you can use the following code to customize the configuration of the SDK.

```
import Clickstream

// config the sdk after initialize.
do {
    var configuration = try ClickstreamAnalytics.getClickstreamConfiguration()
    configuration.appId = "your appId"
    configuration.endpoint = "https://example.com/collect"
    configuration.authCookie = "your authentication cookie"
    configuration.sessionTimeoutDuration = 1800000
    configuration.isTrackScreenViewEvents = false
    configuration.isTrackUserEngagementEvents = false
    configuration.isLogEvents = true
    configuration.isCompressEvents = true
} catch {
    print("Failed to config ClickstreamAnalytics: \$(error)")
}
```

Note

This configuration will override the default configuration in `amplifyconfiguration.json` file.

Here is an explanation of each option.

Method name	Parameter type	Required	Default value	Description
<code>appId</code>	String	true	--	the app id of your application in web console
<code>endpoint</code>	String	true	--	the endpoint path you will

Method name	Parameter type	Required	Default value	Description
				upload the event to Clickstream ingestion server
authCookie	String	false	--	your auth cookie for AWS application load balancer auth cookie
sessionTimeoutDuration	Int64	false	1800000	the duration for session timeout in milliseconds
isTrackScreenViewEvents	Bool	false	true	whether to auto-record screen view events
isTrackUserEngagementEvents	Bool	false	true	whether to auto-record user engagement events
isLogEvents	Bool	false	false	whether to automatically print event JSON for debugging events (Learn more)
isCompressEvents	Bool	false	true	whether to compress event content by gzip when uploading events

Debug events

You can follow the steps below to view the event raw JSON and debug your events.

1. Set the `isLogEvents` option with `true` in debug mode.
2. Integrate the SDK and launch your app by Xcode, then open the log panel.
3. Input `EventRecorder` to the filter, and you will see the JSON content of all events recorded by Clickstream Swift SDK.

Data format definition

Data type

Clickstream Swift SDK supports the following data types:

Data type	Range	Example
Int	-214748364 ~ 2147483647	12
Int64	-9223372036854775808 ~ 9223372036854775807	26854775808
Double	-2.22E-308 ~ 1.79E+308	3.14
Boolean	true, false	true
String	1024 characters maximum	"Clickstream"

Naming rules

1. The event name and attribute name cannot start with a number, and only contain uppercase and lowercase letters, numbers, and underscores. In case of an invalid event name, it will throw `precondition failure`. In case of an invalid attribute name or user attribute name, it will discard the attribute and record error.
2. Do not use `_` as prefix in an event name or attribute name, because the `_` prefix is reserved for the solution.

3. The event name and attribute name are case sensitive, so `Add_to_cart` and `add_to_cart` will be recognized as two different event names.

Event and attribute limitation

To improve the efficiency of querying and analysis, we apply limitations to event data as follows:

Item	Recommended	Maximum (hard limit)	Strategy	Error code
Event name invalid	N/A	N/A	discard event, print log and record <code>_clickstream_error</code> event	1001
Length of event name	less than 25 characters	50 characters	discard event, print log and record <code>_clickstream_error</code> event	1002
Length of event attribute name	less than 25 characters	50 characters	discard the attribute, print log and record error in event attribute	2001
Attribute name invalid	N/A	N/A	discard the attribute, print log and record error in event attribute	2002
Length of event attribute value	less than 100 characters	1024 characters	discard the attribute, print log and record	2003

Item	Recommended	Maximum (hard limit)	Strategy	Error code
			error in event attribute	
Event attribute per event	less than 50 attributes	500 event attributes	discard the attribute, print log and record error in event attribute	2004
User attribute number	less than 25 attributes	100 user attributes	discard event, print log and record_clickstream_error event	3001
Length of user attribute name	less than 25 characters	50 characters	discard event, print log and record_clickstream_error event	3002
User attribute name invalid	N/A	N/A	discard event, print log and record_clickstream_error event	3003
Length of user attribute value	less than 50 characters	256 characters	discard event, print log and record_clickstream_error event	3004

Item	Recommended	Maximum (hard limit)	Strategy	Error code
Item number in one event	less than 50 items	100 items	discard the item, print log and record error in event attribute	4001
Length of item attribute value	less than 100 characters	256 characters	discard the item, print log and record error in event attribute	4002
Custom item attribute number in one item	less than 10 custom attributes	10 custom attributes	discard the item, print log and record error in event attribute	4003
Length of item attribute name	less than 25 characters	50 characters	discard the item, print log and record error in event attribute	4004
Item attribute name invalid	N/A	N/A	discard the item, print log and record error in event attribute	4005

Rules:

- The character limits are the same for single-width character languages (for example, English) and double-width character languages (for example, Chinese).
- The limitation of event attribute per event involves both common attributes and preset attributes.
- If the attribute or user attribute with the same name is added more than twice, the latest value will apply.

- All errors that exceed the limit will be recorded `_error_code` and `_error_message` in the event attributes.

Preset events

Automatically collected events

Event name	Triggered	Event attributes
<code>_first_open</code>	when the user launches an app the first time after installation	N/A
<code>_session_start</code>	when a user first opens the app or a user returns to the app after 30 minutes of inactivity period	<code>_session_id</code> <code>_session_start_timestamp</code>
<code>_screen_view</code>	when a new screen opens	<code>_screen_name</code> <code>_screen_id</code> <code>_screen_unique_id</code> <code>_previous_screen_name</code> <code>_previous_screen_id</code> <code>_previous_screen_unique_id</code> <code>_entrances</code> <code>_previous_timestamp</code> <code>_engagement_time_msec</code>
<code>_user_engagement</code>	when user navigates away from current screen and the	<code>_engagement_time_msece</code>

Event name	Triggered	Event attributes
	screen is in focus for at least one second	
<code>_app_start</code>	every time the app goes to visible	<code>_is_first_time</code> (when it is the first <code>_app_start</code> event after the application starts, the value is true)
<code>_app_end</code>	every time the app goes to invisible	
<code>_profile_set</code>	when the <code>addUserAttributes()</code> or <code>setUserId()</code> API is called	
<code>_app_exception</code>	when the app crashes	<code>_exception_message</code> <code>_exception_stack</code>
<code>_app_update</code>	when the app is updated to a new version and launched again	<code>_previous_app_version</code>
<code>_os_update</code>	when device operating system is updated to a new version	<code>_previous_os_version</code>
<code>_clickstream_error</code>	<code>event_name</code> is invalid or user attribute is invalid	<code>_error_code</code> <code>_error_message</code>

Session definition

In Clickstream Swift SDK, we do not limit the total time of a session. As long as the time between the next entry of the app and the last exit time is within the allowable timeout period, the current session is considered to be continuous.

The **_session_start** event triggered when the app open for the first time, or the app was open to the foreground and the time between the last exit exceeded **session_time_out** period. The following are session-related attributes.

- **_session_id**: It is calculated by concatenating the last 8 characters of `uniqueid` and the current millisecond, for example: `dc7a7a18-20230905-131926703`.
- **_session_duration**: We calculate the `_session_duration` by minus the current event create timestamp and the session's `_session_start_timestamp`. This attribute will be added in every event during the session.
- **_session_number**: The auto increment number of session in current device, and the initial value is 1.
- **Session timeout duration**: By default, it is 30 minutes, which can be customized through the configuraton update API.

Screen view definition

In Clickstream Swift SDK, we define the `_screen_view` as an event that records a user's browsing path of screen. When a screen transition started, the `_screen_view` event will be recorded if any of the following conditions is met:

- No screen was previously set.
- The new screen name differs from the previous screen title.
- The new screen id differs from the previous screen id.
- The new screen unique id differs from the previous screen unique id.

This event listens for `UIViewController`'s `onViewDidAppear` lifecycle method to determine the screen transition.. In order to track screen browsing path, we use `_previous_screen_name` , `_previous_screen_id` and `_previous_screen_unique_id` to link the previous screen. In addition, there are some other attributes in screen view event.

- **_screen_unique_id**: We calculate the screen unique id by getting the current screen's hashcode, for example: `"5260751568"`.
- **_entrances**: The first screen view event in a session is 1, others is 0.
- **_previous_timestamp**: The timestamp of the previous `_screen_view` event.
- **_engagement_time_msec**: The previous page last engagement milliseconds.

User engagement definition

In Clickstream Swift SDK, we define the `user_engagement` as an event that records the screen browsing time, and we only send this event when user leave the screen and the screen has focus for at least one second.

We define that users leave the screen in the following situations.

- When the user navigates to another screen.
- The user moves the app screen to the background.
- The user exit the app, or kill the process of app.

engagement_time_msec: We calculate the milliseconds from when a screen is visible to when the user leave the screen.

Event attributes

Sample event structure

```
{
  "app_id": "Shopping",
  "app_package_name": "com.company.app",
  "app_title": "Shopping",
  "app_version": "1.0",
  "brand": "apple",
  "carrier": "UNKNOWN",
  "country_code": "US",
  "device_id": "A536A563-65BD-49BE-A6EC-6F3CE7AC8FBE",
  "device_unique_id": "",
  "event_id": "91DA4BBE-933F-4DFA-A489-8AEFBC7A06D8",
  "event_type": "add_to_cart",
  "hashCode": "63D7991D",
  "locale": "en_US",
  "make": "apple",
  "model": "iPhone 14 Pro",
  "network_type": "WIFI",
  "os_version": "16.4",
  "platform": "iOS",
  "screen_height": 2556,
  "screen_width": 1179,
  "sdk_name": "aws-solution-clickstream-sdk",
```



```

"sdk_version": "0.4.1",
"system_language": "en",
"timestamp": 1685082174195,
"unique_id": "0E6614B7-2D2C-4774-AB2F-B0A9E6C3BFAC",
"zone_offset": 28800000,
"user": {
  "_user_city": {
    "set_timestamp": 1685006678437,
    "value": "Shanghai"
  },
  "_user_first_touch_timestamp": {
    "set_timestamp": 1685006678434,
    "value": 1685006678432
  },
  "_user_name": {
    "set_timestamp": 1685006678437,
    "value": "carl"
  }
},
"attributes": {
  "event_category": "recommended",
  "currency": "CNY",
  "_session_duration": 15349,
  "_session_id": "0E6614B7-20230526-062238846",
  "_session_number": 3,
  "_session_start_timestamp": 1685082158847,
  "_screen_name": "ProductDetailViewController",
  "_screen_unique_id": "5260751568"
}
}

```

All user attributes will be stored in user object, and all custom and global attributes in attributes object.

Common attribute

Attribute name	Description	It is generated ...	It is used to or for ...
hashCode	the AnalyticsEvent Object's hash code	generated from String(format: "%08X", hasher.co	distinguish different events

Attribute name	Description	It is generated ...	It is used to or for ...
		mbine(eventjson))	
app_id	clickstream app id	generated when clickstream app create from solution control plane	identify the events for your apps
unique_id	the unique id for user	generated from UUID().uuidString during the SDK first initialization. It will be changed after user relogin to another user who never login, and when user relogin to the previous user in same device. The unique_id will reset to the previous user's unique_id .	identity different users and associate the behavior of logging in and not logging in

Attribute name	Description	It is generated ...	It is used to or for ...
device_id	the unique id for device	generated from <code>UIDevice.current.identifierForVendor?.uuidString ?? UUID().uuidString</code> . It will be changed after app is reinstalled.	distinguish different devices
device_unique_id	the device advertising id	generated from <code>ASIdentifierManager.shared().advertisingIdentifier.uuidString ?? ""</code>	distinguish different devices
event_type	event name	set by user or SDK	distinguish different event types
event_id	the unique id for event	generated from <code>UUID().uuidString</code> when the event is created	distinguish each event

Attribute name	Description	It is generated ...	It is used to or for ...
timestamp	event create timestamp	generated from <code>Date().timeIntervalSinceSince1970 * 1000</code> when the event is created	data analysis needs
platform	the platform name	for iOS device, it is always "iOS"	data analysis needs
os_version	the iOS operating system version	generated from <code>UIDevice.current.systemVersion</code>	data analysis needs
make	the manufacturer of the device	for iOS device, it is always "apple"	data analysis needs
brand	the brand of the device	for iOS device, it is always "apple"	data analysis needs
model	the model of the device	generated from mapping of device identifier	data analysis needs
carrier	the device network operator name	generated from <code>CTTelephonyNetworkInfo().serviceSubscriberCellularProviders?.first?.value</code> , and its default value is "UNKNOWN"	data analysis needs

Attribute name	Description	It is generated ...	It is used to or for ...
network_type	the current device network type	generated from <code>NWPathMonitor</code> , it can be "Mobile", "WIFI" or "UNKNOWN"	data analysis needs
screen_height	the absolute height of the available display size in pixels	generated from <code>UIScreen.main.bounds.size.height * UIScreen.main.scale</code>	data analysis needs
screen_width	the absolute width of the available display size in pixels	generated from <code>UIScreen.main.bounds.size.width * UIScreen.main.scale</code>	data analysis needs
zone_offset	the device raw offset from GMT in milliseconds	generated from <code>TimeZone.current.secondsFromGMT() * 1000</code>	data analysis needs
locale	the default locale (language, country and variant) for this device of the Java Virtual Machine	generated from <code>Locale.current</code>	data analysis needs

Attribute name	Description	It is generated ...	It is used to or for ...
system_language	the device language code	generated from <code>Locale.current.languageCode</code> , and its default is value "UNKNOWN"	data analysis needs
country_code	country/region code for this device	generated from <code>Locale.current.regionCode</code> and its default value is "UNKNOWN"	data analysis needs
sdk_version	clickstream SDK version	generated from <code>PackageInfo.version</code>	data analysis needs
sdk_name	clickstream SDK name	it is always "aws-solution-clickstream-sdk"	data analysis needs
app_version	the app version name of user's app	generated from <code>Bundle.main.infoDictionary["CFBundleShortVersionString"] ?? ""</code>	data analysis needs

Attribute name	Description	It is generated ...	It is used to or for ...
app_package_name	the app package name of user's app	generated from Bundle.main.infoDictionary["CFBundleIdentifier"] ?? ""	data analysis needs
app_title	the display name of user's app	generated from Bundle.main.infoDictionary["CFBundleName"] ?? ""	data analysis needs

User attributes

Attribute name	Description
_user_id	Reserved for user id that is assigned by app
_user_ltv_revenue	Reserved for user lifetime value
_user_ltv_currency	Reserved for user lifetime value currency
_user_first_touch_timestamp	The time (in microseconds) when the user first opened the app or visited the site, and it is included in every event in user object

Event attributes

Attribute name	Data type	Auto track	Description
_traffic_source_medium	String	false	Reserved for traffic medium. Use this attribute to store the medium that acquired user when events were logged.
_traffic_source_name	String	false	Reserved for traffic name. Use this attribute to store the marketing campaign that acquired user when events were logged.
_traffic_source_source	String	false	Reserved for traffic source. Name of the network source that acquired the user when the event were reported.
_channel	String	false	The channel for app was downloaded
_session_id	String	true	Added in all events.
_session_start_timestamp	long	true	Added in all events.
_session_duration	long	true	Added in all events.
_session_number	int	true	Added in all events.
_screen_name	String	true	Added in all events.
_screen_unique_id	String	true	Added in all events.

Item attributes

Attribute name	Data type	Required	Description
id	string	False	The id of the item
name	string	False	The name of the item
brand	string	False	The brand of the item
currency	string	False	The currency of the item
price	number	False	The price of the item
quantity	string	False	The quantity of the item
creative_name	string	False	The creative name of the item
creative_slot	string	False	The creative slot of the item
location_id	string	False	The location id of the item
category	string	False	The category of the item
category2	string	False	The category2 of the item
category3	string	False	The category3 of the item
category4	string	False	The category4 of the item

Attribute name	Data type	Required	Description
category5	string	False	The category5 of the item

You can use the listed preset item attributes, or you can add custom attributes to an item. In addition to the preset attributes, an item can add up to 10 custom attributes.

Change log

For more information, see [change logs on GitHub](#).

Web SDK

Introduction

Clickstream Web SDK can help you easily collect click stream data from browser to your AWS environments through the data pipeline provisioned by this solution. This SDK is part of an AWS solution - Clickstream Analytics on AWS, which provisions data pipeline to ingest and process event data into AWS services such as S3, Redshift.

The SDK is based on the amplify-js SDK core library and developed according to the amplify-js SDK plug-in specification. In addition, the SDK is equipped with features that automatically collect common user events and attributes (for example, page view and first open) to simplify data collection for users.

Integrate the SDK

Using NPM

1. Include SDK

```
npm install @aws/clickstream-web
```

2. Initialize the SDK

You need to configure the SDK with default information before using it. To do this, firstly copy your initial code from your solution web console. The initial code is like:

```
import { ClickstreamAnalytics } from '@aws/clickstream-web';

ClickstreamAnalytics.init({
  appId: "your appId",
  endpoint: "https://example.com/collect",
});
```

Then, add the code to your app's root entry point, for example `index.js/app.tsx` in React or `main.ts` in Vue/Angular.

In the code, `appId` and `endpoint` are already set up. Alternatively, you can manually add this code snippet and replace the values of `appId` and `endpoint` after you registered app to a data pipeline in the solution web console.

Using JS File

1. Download the `clickstream-web.min.js` from the assets in [GitHub Release](#) page, and then copy it into your project.
2. Add the following initial code into your `index.html`.

```
<script src="clickstream-web.min.js"></script>
<script>
  window.ClickstreamAnalytics.init({
    appId: 'your appId',
    endpoint: 'https://example.com/collect',
  })
</script>
```

You can find the `appId` and `endpoint` in the application detail page of the solution web console.

To lazy load the SDK, use the `async` attribute and place the `ClickstreamAnalytics.init()` method after `window.onload` or `DOMContentLoaded`.

Get started

Record event

Add the following code where you need to record event.

```
import { ClickstreamAnalytics } from '@aws/clickstream-web';
// record event with attributes
ClickstreamAnalytics.record({
  name: 'button_click',
  attributes: {
    category: 'shoes',
    currency: 'CNY',
    value: 279.9,
  }
});

//record event with name
ClickstreamAnalytics.record({ name: 'buttonClick' });
```

Add global attribute

- Add global attributes when initializing the SDK

```
ClickstreamAnalytics.init({
  appId: "your appId",
  endpoint: "https://example.com/collect",
  globalAttributes:{
    _traffic_source_medium: "Search engine",
    _traffic_source_name: "Summer promotion",
  }
});
```

- Add global attributes after initializing the SDK

```
ClickstreamAnalytics.setGlobalAttributes({
  _traffic_source_medium: "Search engine",
  level: 10,
});
```

It is recommended to set global attributes when initializing the SDK, and global attributes will be included in all events that occur after it is set. You also can remove a global attribute by setting its value to null.

Login and logout

```
import { ClickstreamAnalytics } from '@aws/clickstream-web';

// when user login success.
ClickstreamAnalytics.setUserId("1234");

// when user logout
ClickstreamAnalytics.setUserId(null);
```

Add user attribute

```
ClickstreamAnalytics.setUserAttributes({
  userName: "carl",
  userAge: 22
});
```

Current login user's attributes will be cached in localStorage, so the next time the browser opens you don't need to set up all user's attributes again. You can also use the same api `ClickstreamAnalytics.setUserAttributes()` to update the current user's attributes in case of any changes.

Important

If your application is already published and most users have already logged in, please manually set the user attributes once when integrating the Clickstream SDK for the first time to ensure that subsequent events contain user attributes.

Record event with items

You can add the following code to log an event with an item.

```
import { ClickstreamAnalytics, Item } from '@aws/clickstream-web';

const itemBook: Item = {
  id: '123',
  name: 'Nature',
  category: 'book',
  price: 99,
  book_publisher: 'Nature Research',
```

```
};

ClickstreamAnalytics.record({
  name: 'view_item',
  attributes: {
    currency: 'USD',
    event_category: 'recommended',
  },
  items: [itemBook],
});
```

For more information about logging more attributes in an item, refer to [item attributes](#).

Important

Only pipelines from version 1.1.0 can handle items with custom attribute.

Send event immediate in batch mode

In batch mode, you can still send an event immediately by setting the `isImmediate` attribute to true, as shown in the following code.

```
import { ClickstreamAnalytics } from '@aws/clickstream-web';

ClickstreamAnalytics.record({
  name: 'button_click',
  isImmediate: true,
});
```

Customize configurations

In addition to the required `appId` and `endpoint`, you can configure other information for customization purposes:

```
import { ClickstreamAnalytics, SendMode, PageType } from '@aws/clickstream-web';

ClickstreamAnalytics.init({
  appId: "your appId",
  endpoint: "https://example.com/collect",
  sendMode: SendMode.Batch,
  sendEventsInterval: 5000,
```

```

isTrackPageViewEvents: true,
isTrackUserEngagementEvents: true,
isTrackClickEvents: true,
isTrackSearchEvents: true,
isTrackScrollEvents: true,
pageType: PageType.SPA,
isLogEvents: false,
authCookie: "your auth cookie",
sessionTimeoutDuration: 1800000,
searchKeyWords: ['product', 'class'],
domainList: ['example1.com', 'example2.com'],
});

```

Each option is explained below:

Name	Required	Default value	Description
appId	true	N/A	the app id of your application in the web console
endpoint	true	N/A	the endpoint path where you will upload the event to Clickstream ingestion server
sendMode	false	Immediate	there are two ways to send events: Immediate and Batch
sendEventsInterval	false	5000	event sending interval in milliseconds, only works in Batch mode
isTrackPageViewEvents	false	true	whether to auto record page view events in the browser

Name	Required	Default value	Description
isTrackUserEngagementEvents	false	true	whether to auto record user engagement events in the browser
isTrackClickEvents	false	true	whether to auto record link click events in the browser
isTrackSearchEvents	false	true	whether to auto record search result page events in the browser
isTrackScrollEvents	false	true	whether to auto record page scroll events in the browser
pageType	false	SPA	the website type: SPA for single page application, and multiPageApp for multiple page application. This attribute works only when the value of attribute isTrackPageViewEvents is true.
isLogEvents	false	false	whether to print out event json in the web console for debugging

Name	Required	Default value	Description
authCookie	false	--	your auth cookie for AWS application load balancer auth cookie
sessionTimeoutDuration	false	1800000	the duration for session timeout in milliseconds
searchKeyWords	false	--	the customized keywords to trigger the <code>_search</code> event. By default, it supports <code>q</code> , <code>s</code> , <code>search</code> , <code>query</code> and <code>keyword</code> query parameters.
domainList	false	--	the domain list can be configured if a website crosses multiple domains. The <code>_outbound</code> attribute of the <code>_click</code> event will be true when a link leads to a website that's not a part of your configured domain.

Update configuration

You can update the default configuration after initializing the SDK. Below are the additional configuration options you can customize.

```
import { ClickstreamAnalytics } from '@aws/clickstream-web';

ClickstreamAnalytics.updateConfigure({
  isLogEvents: true,
```

```
authCookie: 'your auth cookie',
isTrackPageViewEvents: false,
isTrackUserEngagementEvents: false,
isTrackClickEvents: false,
isTrackScrollEvents: false,
isTrackSearchEvents: false,
});
```

Debug events

You can follow the steps below to view the event raw json and debug your events.

1. Use `ClickstreamAnalytics.init()` API to set the `isLogEvents` attribute to true in debug mode.
2. Integrate the SDK and launch your web application in a browser, and then open the Inspection page and switch to console tab.
3. Enter `EventRecorder` to Filter, and you will see the JSON content of all events recorded by Clickstream Web SDK.

Data format definition

Data type

Clickstream Web SDK supports the following data types:

Data type	Range	Example
number	5e-324~1.79e+308	12, 26854775808, 3.14
boolean	true false	true
string	max 1024 characters	"Clickstream"

Naming rules

1. The event name and attribute name cannot start with a number, and only contains uppercase and lowercase letters, numbers, and underscores. If the event name is invalid, the SDK will

record `_clickstream_error` event; if the attribute or user attribute name is invalid, the attribute will be discarded and the SDK also records `_clickstream_error` event.

- Do not use `_` as prefix in an event name or attribute name, because the `_` prefix is reserved for the solution.
- The event name and attribute name are case sensitive, so `Add_to_cart` and `add_to_cart` will be recognized as two different event names.

Event and attribute limitation

In order to improve the efficiency of querying and analysis, we apply limits to event data as follows:

Error code	Name	Suggestion	Hard limit	Strategy
1001	Invalid event name	N/A	N/A	discard event, print log and record <code>_clickstream_error</code> event
1002	Length of event name	Less than 25 characters	50 characters	discard event, print log and record <code>_clickstream_error</code> event
2001	Length of event attribute name	Less than 25 characters	50 characters	discard the attribute, print log and record error in event attribute
2002	Attribute name invalid	N/A	N/A	discard the attribute, print log and record error in event attribute

Error code	Name	Suggestion	Hard limit	Strategy
2003	Length of event attribute value	Less than 100 characters	1024 characters	discard the attribute, print log and record error in event attribute
2004	Event attribute per event	Less than 50 attributes	500 event attributes	discard the attribute that exceed, print log and record error in event attribute
3001	User attribute number	Less than 25 attributes	100 user attributes	discard the attribute that exceed, print log and record <code>_clickstream_error</code> event
3002	Length of user attribute name	Less than 25 characters	50 characters	discard the attribute, print log and record <code>_clickstream_error</code> event
3003	User attribute name invalid	N/A	N/A	discard the attribute, print log and record <code>_clickstream_error</code> event

Error code	Name	Suggestion	Hard limit	Strategy
3004	Length of User attribute value	Less than 50 characters	256 characters	discard the attribute, print log and record <code>_clickstream_error</code> event
4001	Item number in one event	Less than 50 items	100 items	discard the item, print log and record error in event attribute
4002	Length of item attribute value	Less than 100 characters	256 characters	discard the item, print log and record error in event attribute
4003	Custom item attribute number in one item	Less than 10 custom attributes	10 custom attributes	discard the item, print log and record error in event attribute
4004	Length of item attribute name	Less than 25 characters	50 characters	discard the item, print log and record error in event attribute
4005	Item attribute name invalid	N/A	N/A	discard the item, print log and record error in event attribute

Important

- The character limits are the same for single-width character languages (for example, English) and double-width character languages (for example, Chinese).

- The limit of event attribute per event includes common attributes and preset attributes.
- If the attribute or user attribute with the same name is added more than twice, the latest value will apply.
- All errors that exceed the limit will be recorded `_error_code` and `_error_message` these two attribute in the event attributes.

Preset events

Automatically collected events

Event name	Triggered	Event Attributes
<code>_first_open</code>	the first time a user launches the site in a browser	
<code>_session_start</code>	when a user first visits the site or a user returns to the website after 30 minutes of inactivity period, Learn more	1. <code>_session_id</code> 2. <code>_session_start_timestamp</code>
<code>_page_view</code>	when new page is opens, Learn more	1. <code>_page_referrer</code> 2. <code>_page_referrer_title</code> 3. <code>_entrances</code> 4. <code>_previous_timestamp</code> 5. <code>_engagement_time_msec</code>
<code>_user_engagement</code>	when user navigates away from current webpage and the page is in focus for at least one second, Learn more	1. <code>_engagement_time_msec</code>
<code>_app_start</code>	every time the browser goes to visible	1. <code>_is_first_time</code> (when it is the first <code>_app_start</code> event after

Event name	Triggered	Event Attributes
		the application starts, the value is true)
_app_end	every time the browser goes to invisible	
_profile_set	when the addUserAttributes() or setUserId() api called	
_scroll	the first time a user reaches the bottom of each page (that is, when a 90% vertical depth becomes visible)	_engagement_time_msec
_search	each time a user performs a site search, indicated by the presence of a URL query parameter, by default we detect q, s, search, query and ke query parameters	_search_key (the keyword name) _search_term (the search content)
_click	each time a user clicks a link that leads away from the current domain (or configured domain list)	1._link_classes(the content of class in tag <a>) 2._link_domain (the domain of herf in tag <a>) 3._link_id (the content of id in tag <a>) 4._link_url (the content of herf in tag <a>) 5._outbound (if the domain is not in configured domain list, the attribute value is true)

Event name	Triggered	Event Attributes
<code>_clickstream_error</code>	event_name is invalid or user attribute is invalid	<ol style="list-style-type: none"> <code>_error_code</code> <code>_error_message</code>

Session definition

In Clickstream Web SDK, there is no limit to the total time of a session. As long as the time between the next entry of the browser and the last exit time is within the allowable timeout period, the current session is considered to be continuous.

The `_session_start` event is initiated when the website opens for the first time, or the browser opens to the foreground and the time between the last exit exceeded `session_time_out` period, and the following are session-related attributes.

- `_session_id`: We calculate the session id by concatenating the last 8 characters of `uniqueid` and the current millisecond, for example, `dc7a7a18-20230905-131926703`.
- `_session_duration` : We calculate the session duration by minus the current event create timestamp and the session's `_session_start_timestamp`, this attribute will be added in every event during the session.
- `_session_number` : It indicates the auto increment number of session in current browser, and has an initial value 1.
- Session timeout duration: By default, it is 30 minutes, which can be customized through the [configuration](#) API.

Page view definition

In Clickstream Web SDK, the `_page_view` refers to an event that records a user's browsing path of page. When a page transition started, the `_page_view` event will be recorded if any of the following conditions is met:

- No page was previously set.
- The new page title differs from the previous page title.
- The new page URL differs from the previous page URL.

This event listens for `pushState`, `popState` in history, and `replaceState` of window to determine the page transition. In order to track page browsing path, we use `_page_referrer` (last page URL) and `page_referrer_title` to link the previous page. There are some other attributes in page view event.

- `_entrances`: It is 1 for the first page view event in a session. Otherwise, it is 0.
- `_previous_timestamp`: The timestamp of the previous `_page_view` event.
- `_engagement_time_msec`: The previous page last engagement in milliseconds.

User engagement definition

In Clickstream Web SDK, the `_user_engagement` refers to an event that records the page browsing time. This event is sent only when a user leaves the page and the page has focus for at least one second.

We define that users leave the page in the following situations.

- When the user navigates to another page under the current domain.
- When the user clicks a link that leads away from the current domain.
- When the user clicks another browser tab or minimizes the current browser window.
- When the user closes the website tab or closes the browser application.

`engagement_time_msec`: We calculate the milliseconds from when a page is visible to when the user leaves the page.

Event attributes

Sample event structure

```
{
  "hashCode": "80452b0",
  "unique_id": "c84ad28d-16a8-4af4-a331-f34cdc7a7a18",
  "event_type": "add_to_cart",
  "event_id": "460daa08-0717-4385-8f2e-acb5bd019ee7",
  "timestamp": 1667877566697,
  "device_id": "f24bec657ea8eff7",
  "platform": "Web",
  "make": "Google Inc.",
```

```
"locale": "zh_CN",
"screen_height": 1080,
"screen_width": 1920,
"viewport_height": 980,
"viewport_width": 1520,
"zone_offset": 28800000,
"system_language": "zh",
"country_code": "CN",
"sdk_version": "0.2.0",
"sdk_name": "aws-solution-clickstream-sdk",
"host_name": "https://example.com",
"app_id": "appId",
"items": [{
  "id": "123",
  "name": "Nike",
  "category": "shoes",
  "price": 279.9
}],
"user": {
  "_user_id": {
    "value": "312121",
    "set_timestamp": 166787756697
  },
  "_user_name": {
    "value": "carl",
    "set_timestamp": 166787756697
  },
  "_user_first_touch_timestamp": {
    "value": 1667877267895,
    "set_timestamp": 166787756697
  }
},
"attributes": {
  "event_category": "recommended",
  "currency": "CNY",
  "_session_id": "dc7a7a18-20221108-031926703",
  "_session_start_timestamp": 1667877566703,
  "_session_duration": 391809,
  "_session_number": 1,
  "_latest_referrer": "https://amazon.com/s?k=nike",
  "_latest_referrer_host": "amazon.com",
  "_page_title": "index",
  "_page_url": "https://example.com/index.html"
}
```

}

All user attributes will be stored in user object, and all custom and global attributes in attributes object.

Common attributes

Attribute name	Data type	Description	How to generate	Usage and purpose
hashCode	string	the event object's hash code	calculated by library <code>@aws-crypto/sha256-js</code>	distinguish different events
app_id	string	the app_id for your app	generated by clickstream solution when you register an app to a data pipeline	identify the events for your apps
unique_id	string	the unique id for user	generated from <code>uuidV4()</code> by the SDK first initialization. It will be changed if user logout and then login to a new user. When user re-login to the previous user in the same browser, the <code>unique_Id</code> will be reset to the	the unique id to identity different users and associating the behavior of logged-in and not logged-in

Attribute name	Data type	Description	How to generate	Usage and purpose
			same previous unique_id.	
device_id	string	the unique id for device	generated from <code>uuidV4()</code> when the website is first open, then the uuid will stored in <code>localStorage</code> and will never be changed	distinguish different devices
event_type	string	event name	set by developer or SDK	distinguish different events type
event_id	string	the unique id for event	generated from <code>uuidV4()</code> when the event create	distinguish different events
timestamp	number	event create timestamp in millisecond	generated from <code>new Date().getTime()</code> when event create	data analysis needs
platform	string	the platform name	for browser is always Web	data analysis needs

Attribute name	Data type	Description	How to generate	Usage and purpose
make	string	the browser make	generated from <code>window.navigator.product</code> or <code>window.navigator.vendor</code>	data analysis needs
screen_height	number	the screen height pixel	generated from <code>window.screen.height</code>	data analysis needs
screen_width	number	the screen width pixel	generated from <code>window.screen.width</code>	data analysis needs
viewport_height	number	the website viewport height pixel	generated from <code>window.innerHeight</code>	data analysis needs
viewport_width	number	the website viewport width pixel	generated from <code>window.innerWidth</code>	data analysis needs
zone_offset	number	the device raw offset from GMT in milliseconds.	generated from <code>currentTime.getTimezoneOffset()*60000</code>	data analysis needs
locale	string	the default locale(language, country and variant) for the browser	generated from <code>window.navigator.language</code>	data analysis needs

Attribute name	Data type	Description	How to generate	Usage and purpose
system_language	string	the browser language code	generated from <code>window.navigator.language</code>	data analysis needs
country_code	string	country/region code for the browser	generated from <code>window.navigator.language</code>	data analysis needs
sdk_version	string	clickstream sdk version	generated from <code>package.json</code>	data analysis needs
sdk_name	string	clickstream sdk name	this will always be <code>aws-solution-clickstream-sdk</code>	data analysis needs
host_name	string	the website hostname	generated from <code>window.location.hostname</code>	data analysis needs

User attributes

Attribute name	Description
<code>_user_id</code>	Reserved for user id that is assigned by app
<code>_user_ltv_revenue</code>	Reserved for user lifetime value
<code>_user_ltv_currency</code>	Reserved for user lifetime value currency

Attribute name	Description
_user_first_touch_timestamp	Added to the user object for all events. The time (in milliseconds) when the user first visited the website.

Event attributes

Attribute name	Data type	Auto track	Description
_traffic_source_medium	string	false	Reserved for traffic medium. Use this attribute to store the medium that acquired user when events were logged. Example: Email, Paid search, Search engine.
_traffic_source_name	string	false	Reserved for traffic name. Use this attribute to store the marketing campaign that acquired user when events were logged. Example: Summer promotion.
_traffic_source_source	string	false	Reserved for traffic source. Name of the network source that acquired the user when the event were reported. Example:

Attribute name	Data type	Auto track	Description
			Google, Facebook, Bing, Baidu.
_session_id	string	true	Added in all events.
_session_start_timestamp	number	true	Added in all events. The value is millisecond.
_session_duration	number	true	Added in all events. The value is millisecond.
_session_number	number	true	Added in all events.
_page_title	string	true	Added in all events.
_page_url	string	true	Added in all events.
_latest_referrer	string	true	Added in all events. The last off-site url.
_latest_referrer_host	string	true	Added in all events. The last off-site domain name.

Item attributes

Attribute name	Data type	Required	Description
id	string	False	The id of the item
name	string	False	The name of the item
brand	string	False	The brand of the item
price	number	False	The price of the item

Attribute name	Data type	Required	Description
quantity	string	False	The quantity of the item
creative_name	string	False	The creative name of the item
creative_slot	string	False	The creative slot of the item
location_id	string	False	The location id of the item
category	string	False	The category of the item
category2	string	False	The category2 of the item
category3	string	False	The category3 of the item
category4	string	False	The category4 of the item
category5	string	False	The category5 of the item

You can use the listed preset item attributes, and you can also add custom attributes to an item. In addition to the preset attributes, an item can add up to 10 custom attributes.

Change logs

For more information, see the [change logs on GitHub](#).

References

[Source code](#)

[Project issue](#)

Flutter SDK

Introduction

Clickstream Flutter SDK can help you easily collect in-app click stream data from mobile devices to your AWS environments through the data pipeline provisioned by this solution.

The SDK relies on the [Clickstream Android SDK](#) and [Clickstream Swift SDK](#). Therefore, Flutter SDK also supports automatically collecting common user events and attributes (for example, session start, first open). In addition, we've added easy-to-use APIs to simplify data collection in Flutter apps.

Platform Support

Android: 4.1 (API level 16) and later

iOS: 13 and later

Integrate the SDK

1. Include SDK

```
flutter pub add clickstream_analytics
```

After completion, rebuild your Flutter application.

```
flutter run
```

2. Initialize the SDK

Copy your configuration code from your clickstream solution web console, and the configuration code should be as follows. You can also manually add this code snippet and replace the values of appId and endpoint after you registered app to a data pipeline in the Clickstream Analytics solution console.

```
import 'package:clickstream_analytics/clickstream_analytics.dart';

final analytics = ClickstreamAnalytics();
```

```
analytics.init(  
  appId: "your appId",  
  endpoint: "https://example.com/collect"  
);
```

Important

- Your appId and endpoint are already set up.
- We only need to initialize the SDK once after the application starts. It is recommended to do it in the main function of your App.
- We can use `bool result = await analytics.init()` to get the boolean value of the initialization result.

3. Start using

Record event

Add the following code where you need to record event.

```
import 'package:clickstream_analytics/clickstream_analytics.dart';  
  
final analytics = ClickstreamAnalytics();  
  
// record event with attributes  
analytics.record(name: 'button_click', attributes: {  
  "event_category": "shoes",  
  "currency": "CNY",  
  "value": 279.9  
});  
  
// record event with name  
analytics.record(name: "button_click");
```

Add global attribute

```
analytics.addGlobalAttributes({  
  "_traffic_source_medium": "Search engine",  
  "_traffic_source_name": "Summer promotion",  
  "level": 10  
});
```

```
// delete global attribute
analytics.deleteGlobalAttributes(["level"]);
```

It is recommended to set global attributes after each SDK initialization, and global attributes will be included in all events that occur after it is set.

Login and logout

```
// when user login success.
analytics.setUserId("userId");

// when user logout
analytics.setUserId(null);
```

Add user attribute

```
analytics.setUserAttributes({
  "userName": "carl",
  "userAge": 22
});
```

Current login user's attributes will be cached in disk, so the next time app launches you don't need to set all user's attribute again, of course you can use the same api `analytics.setUserAttributes()` to update the current user's attribute when it changes.

Record event with items

You can add the following code to log an event with an item, and you can add custom item attribute in the attributes Map. In addition to the preset attributes, an item can add up to 10 custom attributes.

```
var itemBook = ClickstreamItem(
  id: "123",
  name: "Nature",
  category: "book",
  price: 99,
  attributes: {
    "book_publisher": "Nature Research"
  }
}
```

```
);

analytics.record(
  name: "view_item",
  attributes: {
    "currency": 'USD',
    "event_category": 'recommended'
  },
  items: [itemBook]
);
```

For logging more attributes in an item, please refer to [item attributes](#).

Important

Only pipelines from version 1.1.0 can handle items with custom attribute.

Other configurations

In addition to the required appId and endpoint, you can configure other information to get more customized usage:

```
final analytics = ClickstreamAnalytics();
analytics.init(
  appId: "your appId",
  endpoint: "https://example.com/collect",
  isLogEvents: false,
  isCompressEvents: false,
  sendEventsInterval: 10000,
  isTrackScreenViewEvents: true,
  isTrackUserEngagementEvents: true,
  isTrackAppExceptionEvents: false,
  authCookie: "your auth cookie",
  sessionTimeoutDuration: 1800000
);
```

Here is an explanation of each option:

Name	Required	Default value	Description
appId	true	N/A	the app id of your application in control plane
endpoint	true	N/A	the endpoint path you will upload the event to Clickstream ingestion server
isLogEvents	false	false	whether to print out event json in console for debugging events
isCompressEvents	false	true	whether to compress event content by gzip when uploading events
sendEventsInterval	false	10000	event sending interval in milliseconds
isTrackScreenViewEvents	false	true	whether auto record screen view events in app
isTrackUserEngagementEvents	false	true	whether auto record user engagement events in app
isTrackAppExceptionEvents	false	false	whether auto track exception event in app

Name	Required	Default value	Description
authCookie	false	N/A	your auth cookie for AWS application load balancer auth cookie
sessionTimeoutDuration	false	1800000	the duration for session timeout in milliseconds

Configuration update

You can update the default configuration after initializing the SDK, below are additional configuration options you can customize.

```
final analytics = ClickstreamAnalytics();
analytics.updateConfigure(
    appId: "your appId",
    endpoint: "https://example.com/collect",
    isLogEvents: true,
    isCompressEvents: false,
    isTrackScreenViewEvents: false
    isTrackUserEngagementEvents: false,
    isTrackAppExceptionEvents: false,
    sessionTimeoutDuration: 100000,
    authCookie: "test cookie");
```

Send event immediately

```
final analytics = ClickstreamAnalytics();
analytics.flushEvents();
```

Disable SDK

You can disable the SDK in the scenario you need. After disabling the SDK, the SDK will not handle the logging and sending of any events. Of course, you can enable the SDK when you need to continue logging events.

```
final analytics = ClickstreamAnalytics();
```

```
// disable SDK
analytics.disable();

// enable SDK
analytics.enable();
```

Debug events

You can follow the steps below to view the event raw JSON and debug your events.

1. Using `analytics.updateConfigure()` api and set the `isLogEvents` attributes with `true` in debug mode, for example:

```
// log the event in debug mode.
analytics.updateConfigure(isLogEvents: true);
```

2. Integrate the SDK and launch your app.
 - a. For Android application logs, we can see the logs directly in the terminal window. You can also use filters in the Android Studio **Logcat** window to view logs.
 - b. For iOS application logs, we should launch it via Xcode and open the log panel to see it.
3. Input `EventRecorder` to the filter, and you will see the JSON content of all events recorded by Clickstream Flutter SDK.

Data format definition

Data types

Clickstream Flutter SDK supports the following data types.

Data type	Range	Example
int	-9223372036854775808 ~ 9223372036854775807	12
double	5e-324 ~ 1.79e+308	3.14
bool	true, false	true
String	max 1024 characters	"Clickstream"

Naming rules

1. The event name and attribute name cannot start with a number, and must contain only uppercase and lowercase letters, numbers, and underscores. In case of an invalid attribute name or user attribute name, it will discard the attribute and record error.
2. Do not use `_` as prefix in an event name or attribute name, because the `_` prefix is reserved for the solution.
3. The event name and attribute name are case-sensitive, so `Add_to_cart` and `add_to_cart` will be recognized as two different event names.

Event and attribute limitation

In order to improve the efficiency of querying and analysis, we apply limits to event data as follows:

Name	Suggestion	Hard limit	Strategy	Error code
Event name invalid	N/A	N/A	discard event, print log and record <code>_clickstream_error</code> event	1001
Length of event name	under 25 characters	50 characters	discard event, print log and record <code>_clickstream_error</code> event	1002
Length of event attribute name	under 25 characters	50 characters	discard the attribute, print log and record error in event attribute	2001
Attribute name invalid	N/A	N/A	discard the attribute, print log and record	2002

Name	Suggestion	Hard limit	Strategy	Error code
			error in event attribute	
Length of event attribute value	under 100 characters	1024 characters	discard the attribute, print log and record error in event attribute	2003
Event attribute per event	under 50 attributes	500 event attributes	discard the attribute that exceed, print log and record error in event attribute	2004
User attribute number	under 25 attributes	100 user attributes	discard the attribute that exceed, print log and record_clickstream_error event	3001
Length of User attribute name	under 25 characters	50 characters	discard the attribute, print log and record_clickstream_error event	3002
User attribute name invalid	N/A	N/A	discard the attribute, print log and record_clickstream_error event	3003

Name	Suggestion	Hard limit	Strategy	Error code
Length of User attribute value	under 50 characters	256 characters	discard the attribute, print log and record <code>_clickstream_error</code> event	3004
Item number in one event	under 50 items	100 items	discard the item, print log and record error in event attribute	4001
Length of item attribute value	under 100 characters	256 characters	discard the item, print log and record error in event attribute	4002
Custom item attribute number in one item	under 10 custom attributes	10 custom attributes	discard the item, print log and record error in event attribute	4003
Length of item attribute name	under 25 characters	50 characters	discard the item, print log and record error in event attribute	4004
Item attribute name invalid	N/A	N/A	discard the item, print log and record error in event attribute	4005

Important

- The character limits are the same for single-width character languages (e.g., English) and double-width character languages (e.g., Chinese).

- The limit of event attribute per event include preset attributes.
- If the attribute or user attribute with the same name is added more than twice, the latest value will apply.
- All errors that exceed the limit will be recorded `_error_code` and `_error_message` these two attribute in the event attributes.

Preset events

For Android, refer to [Android SDK preset events](#).

For iOS, refer to [Swift SDK preset events](#).

Event attributes

For Android, refer to [Android SDK event attributes](#).

For iOS, refer to [Swift SDK event attributes](#).

Change log

For more information, see [GitHub change log](#).

Native SDK version dependencies

Flutter SDK Version	Android SDK Version	Swift SDK Version
0.2.0	0.10.0	0.9.1
0.1.0	0.9.0	0.8.0

References

[Source code](#)

[Project issue](#)

HTTP API

This section introduces how to send your clickstream data directly to the Clickstream ingestion server via HTTP requests. The Clickstream data processing module will correctly process your data simultaneously by following the guidelines below. Then, you can visually analyze them in the subsequent report module.

Request endpoint

After creating the application in the solution web console, you will get the **Server Endpoint** and **App ID** on the details page. For example:

- **Server Endpoint:** `https://example.com/collect`
- **App ID:** `my_app`

API Specification

- The app ID in the query parameters must be one of the applications you create for the project in the solution web console. Otherwise, the server will respond to HTTP status code 403.
- The request body contains four parts: common attributes, items, user, and attributes. The public attributes require the `event_type`, `event_id`, `timestamp`, and `app_id`; the rest are optional parameters.
- The total size of the body of a single request cannot exceed 1MB. Otherwise, the HTTP status code 413 will return.

Request method

POST

Request headers

Parameter name	Required	Example	Description
Content-Type	YES	application/json; charset=utf-8	Content type

Parameter name	Required	Example	Description
X-Forwarded-For	NO	101.188.67.134	Source IP address, it's required if you forward the client requests to clickstream servers from your servers
User-Agent	NO	Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Mobile Safari/537.36	User-Agent
cookie	NO	your auth cookie	The authentication token for your request. Please refer to server side configuration .

Request query parameters

Parameter name	Required	Example	Description
appld	YES	test_app	The app ID for your application is created in the solution web console
platform	NO	Android/iOS/Web/...	Distinguish between platforms
event_bundle_sequence_id	NO	1	Request sequence number, an

Parameter name	Required	Example	Description
			incrementing integer starting from 1
hashCode	NO	478acd09	The first eight digits of the sha256 calculation result of the request body string
compression	NO	gzip	Request body compression method. Currently, only gzip is supported. Keeping it absent means no compression

Request body

The request body is an array structure that contains the JSON string of one or more events. For example:

```
[
  {
    "event_type": "button_click",
    "event_id": "460daa08-0717-4385-8f2e-acb5bd019ee7",
    "timestamp": 1667877566697,
    "app_id": "your appId",
    "attributes": {
      "productName": "shoes",
      "Price": 99.9
    }
  },
  {
    "event_type": "item_view",
    "event_id": "c6067c1c-fd8d-4fdb-bfaf-cc1212ca0195",
    "timestamp": 1667877565698,
    "app_id": "your appId",
    "attributes": {
```

```

    "productName": "book",
    "Price": 39.9
  }
}
]
```

Event attributes

Attributes name	Required	Data Type	Example	Description
event_type	YES	String	button_click	Event type
event_id	YES	String	460daa08-0717-4385-8f2e-acb5bd019ee7	Event's unique ID, we recommend using UUID to generate
timestamp	YES	Long	1667877566697	The timestamp when the event was generated, in milliseconds
app_id	YES	String	shopping_dev	The corresponding id when creating the application in the Clickstream web console
platform	NO	String	Android/iOS/Web/...	Device platform
os_version	NO	String	10	System version
unique_id	NO	String	c84ad28d-16a8-4af4-a331-f34cdc7a7a18	Unique ID to identify different users and associate

Attributes name	Required	Data Type	Example	Description
				the behavior of logged-in and not logged-in
device_id	NO	String	f24bec657ea8eff7	Distinguish between different devices
make	NO	String	Samsung	Device manufactory
brand	NO	String	Samsung	Device brand
model	NO	String	S23 Ultra	Device model
carrier	NO	String	CDMA	Device network operator name
network_type	NO	String	Mobile	Current device network type
locale	NO	String	zh_CN	Local information
system_language	NO	String	zh	Device language code
country_code	NO	String	CN	Device country code
zone_offset	NO	int	2880000	Device's raw offset from GMT in milliseconds
screen_height	NO	int	1920	Screen height in pixels

Attributes name	Required	Data Type	Example	Description
screen_width	NO	int	1080	Screen width in pixels
viewport_height	NO	int	540	App viewport height
viewport_width	NO	int	360	App viewport width
sdk_version	NO	String	1.2.3	SDK version
sdk_name	NO	String	aws-solution-clickstream-sdk	SDK name
app_package_name	NO	String	com.example.app	User's Application package name
app_version	NO	String	1.1.0	Application version number
app_title	NO	String	shopping	Application name

Attributes name	Required	Data Type	Example	Description
items	NO	Object	<pre>[{ "id": "b011ddc3 -632f-47c b-a68a-ad 83678ecfe d", "name": "Classic coat- rack", "category ": "housewares", "price": 167 }]</pre>	Item list, Supports uploading multiple items at one time. A maximum of 100 items can be uploaded at one time For the item quantity limit, please refer to Event and Attribute Limitation For the supported attributes of the item, please refer to item attribute

Attributes name	Required	Data Type	Example	Description
user	NO	Object	<pre>{ "_user_id": { "value": "0202d0e1", "set_time stamp": 1695006816345 }, "username": { "value": "carl", "set_time stamp": 1695006816345 } }</pre>	<p>User attributes. Each attribute key is the user attribute name. Each attribute contains an object. The object contains two attributes: value: The value of the user attribute. set_timestamp: The timestamp millisecond value when setting the attribute. Up to 100 user attributes can be added to an event. For specific restrictions, please refer to: Event and Attribute Limitations</p>

Attributes name	Required	Data Type	Example	Description
attributes	NO	Object	{ "productName": "book", "Price": 39.9 }	Custom attributes. Up to 500 custom attributes can be added to an event, and the attribute name must meet the naming rules .

Request response

If the HttpStatusCode status code returned by the request is 200, the request is considered successful, other status codes are failures, and the request does not return any other content.

HttpCode

Code	Description
200	Request successful
403	Request failed. Please check if appId and endpoint match, if configured with authentication, please check whether the authentication cookie is correct
413	Request failed. The request body exceeds 1MB

Request code example

cURL:

```
curl --location 'https://example.com/collect?
appId=test_release&platform=Android&event_bundle_sequence_id=1' \
```

```
--header 'Content-Type: application/json; charset=utf-8' \
--header 'X-Forwarded-For: 101.188.67.134' \
--data '[{"event_type":"button_click","event_id":"460daa08-0717-4385-8f2e-
acb5bd019ee7","timestamp":1667877566697,"app_id":"your appId","attributes":
{"productName":"shoes","Price":99.9}},{"event_type":"item_view","event_id":"c6067c1c-
fd8d-4fdb-bfaf-cc1212ca0195","timestamp":1667877565698,"app_id":"your
appId","attributes":{"productName":"book","Price":39.9}}]'
```

C# HttpClient:

```
var client = new HttpClient();
var request = new HttpRequestMessage(HttpMethod.Post, "https://example.com/collect?
appId=test_release&platform=Android&event_bundle_sequence_id=1");
request.Headers.Add("X-Forwarded-For", "101.188.67.134");
var content = new StringContent("[{\"event_type\": \"button_click\", \"event_id\":
\"460daa08-0717-4385-8f2e-acb5bd019ee7\", \"timestamp\": 1667877566697, \"app_id\": \"your
appId\", \"attributes\": {\"productName\": \"shoes\", \"Price\": 99.9}}, {\"event_type
\": \"item_view\", \"event_id\": \"c6067c1c-fd8d-4fdb-bfaf-cc1212ca0195\", \"timestamp
\": 1667877565698, \"app_id\": \"your appId\", \"attributes\": {\"productName\": \"book\",
\"Price\": 39.9}}]", null, "application/json; charset=utf-8");
request.Content = content;
var response = await client.SendAsync(request);
response.EnsureSuccessStatusCode();
Console.WriteLine(await response.Content.ReadAsStringAsync());
```

Java Okhttp:

```
OkHttpClient client=new OkHttpClient().newBuilder()
    .build();
    MediaType mediaType=MediaType.parse("application/json; charset=utf-8");
    RequestBody body=RequestBody.create(mediaType,"[\"event_type\": \"button_click
\", \"event_id\": \"460daa08-0717-4385-8f2e-acb5bd019ee7\", \"timestamp\": 1667877566697,
\", \"app_id\": \"your appId\", \"attributes\": {\"productName\": \"shoes\", \"Price\": 99.9}},
\", {\"event_type\": \"item_view\", \"event_id\": \"c6067c1c-fd8d-4fdb-bfaf-cc1212ca0195\",
\", \"timestamp\": 1667877565698, \"app_id\": \"your appId\", \"attributes\": {\"productName\":
\", \"Price\": 39.9}}]");
    Request request=new Request.Builder()
    .url("https://example.com/collect?
appId=test_release&platform=Android&event_bundle_sequence_id=1")
    .method("POST", body)
    .addHeader("Content-Type", "application/json; charset=utf-8")
    .addHeader("X-Forwarded-For", "101.188.67.134")
    .build();
```

```
Response response=client.newCall(request).execute();
```

JavaScript Fetch:

```
var myHeaders = new Headers();
myHeaders.append("Content-Type", "application/json; charset=utf-8");
myHeaders.append("X-Forwarded-For", "101.188.67.134");

var raw = "[{\"event_type\":\"button_click\",\"event_id\":\"460daa08-0717-4385-8f2e-acb5bd019ee7\",\"timestamp\":1667877566697,\"app_id\":\"your appId\",\"attributes\":{\"productName\":\"shoes\",\"Price\":99.9}},{\"event_type\":\"item_view\",\"event_id\":\"c6067c1c-fd8d-4fdb-bfaf-cc1212ca0195\",\"timestamp\":1667877565698,\"app_id\":\"your appId\",\"attributes\":{\"productName\":\"book\",\"Price\":39.9}}]";

var requestOptions = {
  method: 'POST',
  headers: myHeaders,
  body: raw,
  redirect: 'follow'
};

fetch("https://example.com/collect?
appId=test_release&platform=Android&event_bundle_sequence_id=1", requestOptions)
  .then(response => response.text())
  .then(result => console.log(result))
  .catch(error => console.log('error', error));
```

Python Request:

```
import requests

url = "https://example.com/collect?
appId=test_release&platform=Android&event_bundle_sequence_id=1"

payload = "[{\"event_type\":\"button_click\",\"event_id\":
\"460daa08-0717-4385-8f2e-acb5bd019ee7\",\"timestamp\":1667877566697,\"app_id\":\"your
appId\",\"attributes\":{\"productName\":\"shoes\",\"Price\":99.9}},{\"event_type
\":\"item_view\",\"event_id\":\"c6067c1c-fd8d-4fdb-bfaf-cc1212ca0195\",\"timestamp
\":1667877565698,\"app_id\":\"your appId\",\"attributes\":{\"productName\":\"book\",
\"Price\":39.9}}]"

headers = {
  'Content-Type': 'application/json; charset=utf-8',
  'X-Forwarded-For': '101.188.67.134'
```

```
}  
  
response = requests.request("POST", url, headers=headers, data=payload)  
  
print(response.text)
```

Verification data reported successfully

If you enabled data processing, you can query the **event**, **event_parameter**, **user**, **item** or **ingestion_events** table in Athena directly through SQL.

Moreover, if you enabled Redshift in data modeling, you can query the **event**, **event_parameter**, **user**, or **item** table in Redshift directly through SQL.

User identifier

When you perform data analysis, you usually need to select an appropriate user identifier for analysis based on your business analysis scenario. This will help you improve the accuracy of your analysis, especially in funnel, retention, session and other analysis scenarios.

Clickstream Analytics on AWS solution mainly contains three types of IDs:

- User ID
- Device ID
- User Pseudo ID

This section introduces how to use these three IDs respectively, and you will learn in detail how to use User ID and User Pseudo ID to correlate user behavior.

User ID

User ID is usually a unique identifier that describes the user in your business database, which is relatively more accurate and unique.

- When the user is not registered or logged in, the value of User ID is empty.
- The SDK provides the `ClickstreamAnalytics.setUserId("your user id")` method to set the User ID. When logging out, set null/nil to clear the User ID.
- The User ID is stored in the `user_id` field in the user table.

Device ID

The solution uses Device ID to identify user devices.

- The Device ID will be automatically generated when the app is launched for the first time after integrating the SDK.
- The Device ID may not be the unique identifier of the device. Usually the Device ID may be regenerated after the user uninstalls the app or clears the cache on the web page.
- The Device ID is stored in the `device_id_list` field in the user table.

The following table will introduce how the SDK on each end generates the Device ID.

SDK Types	Generate Rules	Storage Location	Is Unique
Android SDK	By default, <code>AndroidId</code> is used as the Device ID. If the <code>AndroidId</code> cannot be obtained, a random UUID is used instead	Stored in <code>SharedPreferences</code> key-value pair file	Usually, the <code>AndroidId</code> will not change even if the app is uninstalled and reinstalled. If using the UUID as Device ID, it will change after the user uninstalls and reinstall the app
Swift SDK	If your app has been authorized to obtain IDFA, use IDFA as the Device ID. Otherwise, use IDFV as the Device ID. If IDFV cannot be obtained, use a random UUID	<code>UserDefault</code> key-value pair file	Usually, the IDFA does not change even if the app is uninstalled and reinstalled. When using IDFV or UUID, the Device ID will change after the user uninstalls and reinstalls the app
Web SDK	By default, a random UUID is used as the device ID	In the browser's <code>localStorage</code>	Device ID will be regenerated after

SDK Types	Generate Rules	Storage Location	Is Unique
			user clears browser cache

User Pseudo ID

Clickstream Analytics on AWS solution uses User Pseudo ID to correlate logged-in and non-logged-in behavior on the same device.

- User Pseudo IDs are generated from random UUIDs in all SDKs.
- User Pseudo ID will only be reassigned when a new user logs in on the current device. When switching to a user who has already logged in on the current device, it will revert to the User Pseudo ID of the previous user.
- The User Pseudo ID is stored in the `user_pseudo_id` field of the user table.

The following table lists the correspondence between Device ID, User ID, and User Pseudo ID under various scenarios.

Sequence	Events	Device ID	User ID	User Pseudo ID
1	Install App	S	--	1
2	Use the App	S	--	1
3	Logged in user A	S	A	1
4	Use the App	S	A	1
5	Sign out and view	S	--	1
6	Logged in user B	S	B	2
7	Use the App	S	B	2
8	Sign out and view	S	--	2

Sequence	Events	Device ID	User ID	User Pseudo ID
9	Logged in user A	S	A	1
10	Use the App	S	A	1
11	Sign out and view	S	--	1
12	Logged in user C	S	C	3
13	Use the App	S	C	3

As shown in the table, you can count all the behavioral events of user A on device S when user A did not log in and after logging in twice by looking for `user_pseudo_id=1`. Additionally, you can use User ID to join user clickstream data with data from your business systems to build a more complete customer data platform.

Note

When the user uninstalls the app or clears the browser cache, the relationship between the original User Pseudo ID and User ID will be cleared on the device, and a new User Pseudo ID will be generated on the device.

Migrate from third-party SDKs

Introduction

This article provides a best practice for you to migrate from a third-party SDK to Clickstream SDK. If you already have an SDK in your app or website, and you want to replace it with Clickstream SDK, we recommend you adopt this practice, which allow you to achieve a smooth migration with the following benefits:

- Minimum code changes
- Reuse existing data tracking codes
- Quick implementation time

- Dual measurement to ensure data completeness

In summary, we recommend you create one overarching analytic logger function that encapsulates all the event logging methods from both legacy SDK and Clickstream SDK, so that you have one API to log event data to multiple destinations. Once satisfied with the data, you can easily update the function to disable the legacy SDK data logging.

To make it easier to understand, this example uses Clickstream Web SDK to replace Firebase Web SDK (GA4 SDK). Assuming you have integrated Firebase Web SDK into your website, follow the steps below.

Step 1: Integrate Clickstream Web SDK

1. Include SDK

```
npm install @aws/clickstream-web
```

2. Initialize the SDK

Copy your configuration code from your clickstream solution web console. We recommend you add the code to your app's root entry point, for example `index.js/app.tsx` in React or `main.ts` in Vue/Angular. The configuration code should look as follows.

```
import { ClickstreamAnalytics } from '@aws/clickstream-web';

ClickstreamAnalytics.init({
  appId: "your appId",
  endpoint: "https://example.com/collect",
});
```

Step 2: Encapsulate common data logger methods

When integrating multiple data analysis SDKs, it is strongly recommended that you encapsulate all event-logging methods in one function. Processing data logging codes of different SDKs in the same place can make the code concise and easy for you to maintain. Below is an example of our encapsulation that you can copy directly into your project.

```
import { ClickstreamAnalytics } from "@aws/clickstream-web";
```

```
import { getAnalytics, logEvent, setUserProperties, setUserId } from "firebase/
analytics";

export const AnalyticsLogger = {

  log(eventName, attributes, items) {
    attributes = attributes ?? {}
    const [{"items": items, ...mAttributes} = attributes;

    // Clickstream SDK
    ClickstreamAnalytics.record({
      name: eventName,
      attributes: mAttributes,
      items: items
    })

    //Firebase SDK
    const analytics = getAnalytics();
    logEvent(analytics, eventName, attributes);
  },

  setUserAttributes(attributes) {
    // Clickstream SDK
    ClickstreamAnalytics.setUserAttributes(attributes);

    // Firebase SDK
    const analytics = getAnalytics();
    setUserProperties(analytics, attributes);
  },

  setUserId(userId) {
    //Clickstream SDK
    ClickstreamAnalytics.setUserId(userId)

    //Firebase SDK
    const analytics = getAnalytics();
    setUserId(analytics, userId);
  },
}
```

We need to encapsulate three APIs `log()`、`setUserAttributes()` and `setUserId()`. When we invoke the `AnalyticsLogger.log('testEvent')` method, both Clickstream and Firebase SDK will log the event, so we only need to call the `AnalyticsLogger` API when you need to log event data.

Step 3: Migrate to common APIs in minutes

For log events

```
onSignedUp(user) {
  let attributes = {
    _user_id: user.id,
    username: user.username,
    email: user.email,
  };
  -- logEvent(analytics, 'sign_up', attributes);
  ++ AnalyticsLogger.log('sign_up', attributes);
}
```

For the events log API, we need to get the event name and attributes for the events log API and pass them into the new API. Of course, you can also use the "Replace in File" feature to make quick changes, as shown in the image below.

The screenshot shows an IDE interface with a "Replace in Files" dialog box. The dialog box contains the following text:

```
Replace All
Replace 13 occurrences of
'logEvent(analytics, '
across 2 files with 'AnalyticsLogger.log('?
```

Buttons for "Cancel" and "Replace" are visible in the dialog. In the background, a list of files is shown with several instances of `logEvent(analytics, ...)` highlighted in yellow. The file `AnalyticsHandler.js` is selected and highlighted in blue. Below the list, the code in `AnalyticsHandler.js` is visible, showing the replacement of `logEvent` with `AnalyticsLogger.log` at line 244:

```
242     ],
243   }
244   logEvent(analytics, eventName: "add_to_cart", attributes);
245 },
246
```

For log user attributes

```
userSignedIn(user) {
  -- setUserId(analytics, user.id);
```

```
++ AnalyticsLogger.setUserId(user.id);
   let attributes = {
       _user_id: user.id,
       username: user.username,
       email: user.email,
   };
-- setUserProperties(analytics, attributes);
++ AnalyticsLogger.setUserAttributes(attributes);
   }
```

For user id, replace `setUserId()` with `AnalyticsLogger.setUserId()` .

For user attributes,

replace `setUserProperties()` with `AnalyticsLogger.setUserAttributes()` .

Summary

In summary, it is easy to get Clickstream SDK and Firebase SDK to work together. After these three steps, your data will be uploaded to Clickstream Analytics and Firebase, these two SDKs will work well together and will not influence each other. After you are satisfied with the data, you only need to modify the `AnalyticsLogger` file to remove or disable another SDK smoothly.

Analytics Studio

Analytics Studio is a unified web interface for business analysts or data analysts to create and view dashboards, query and explore clickstream data, and manage metadata.

Modules

Below are the modules included in the Analytics Studio.

- **Dashboard.** View the out-of-the-box dashboard and custom dashboards (if any).
- **Exploration.** Use advanced analytics models to query clickstream data beyond the out-of-the-box dashboard.
- **Analyses.** Create and modify dashboards, as well as manage datasets.
- **Data management.** View and manage the metadata for clickstream data.

Terms and concepts

This section describes key concepts and terms used in Analytics Studio.

Event. Clickstream data generated by a user action within an app or website (such as clicking, and visiting a page).

Preset event. Events that are automatically collected by clickstream SDK, usually with a name starting with an underscore '_'.

Custom event. Events that are defined and collected by an app owner. The collection timing and business purpose of the event varies by apps.

Event parameter. Event parameters are used to describe the various dimensions of information at the time an event occurred. They fall into two categories: public parameters and private parameters.

Public parameter. The parameters that all events include, such as user information (`user_id`), and device information (app version, device model, etc.)

Private parameter. The parameters that are unique to certain events, such as user-customized parameters.

User attribute. User attribute is used to record the property of a user. They fall into two categories: preset attributes collected by SDK presets, such as '_first_visit_date'; custom attributes, that is, user attributes reported by the user themselves, such as 'email_address'.

Dashboard

Overview

Clickstream Analytics on AWS collects data from your websites and apps to create dashboards that derive insights. You can use dashboards to monitor traffic, investigate data, and understand your users and their activities.

The data will appear in the QuickSight dashboards after being processed by the data pipeline. Depending on your pipeline configuration, the time it takes varies for data to be available in your dashboard. For example, if you set the data processing interval to be 1 day, the dashboard will show data T+1 day (T as reporting date).

View dashboards

Use this procedure to view dashboards:

1. Go to **Clickstream Analytics on AWS Console**, in the **Navigation Bar**, choose **Analytics Studio**.
2. In the Analytics Studio page that opens, select the project and app you just created in the drop-down list at the top of the web page.
3. Choose the **User lifecycle - default** dashboard.

Reports

The dashboard contains a set of reports throughout the user lifecycle, which aim to help you understand how people use your website or app, from acquisition to retention.

Report name	What it is
Acquisition	Summarizes key metrics about new users, and provides detail view user profile
Engagement	Summarizes key metrics about user engagements and sessions

Report name	What it is
Activity	Summarizes key metrics about events user generates in the app, and provides detail view of event attributes
Retention	Summarizes key metrics about active users and user retentions
Device	Summarizes key metrics about the devices users are using to access your apps and websites, and provides detail view of each device
User	Summarizes an individual user's attributes and the events the user performed
Crash	Summarizes the metrics and other information about the crash events in your app

Custom report

If you want to investigate certain pieces of data further, you can write SQL to create views in Redshift or Athena, then add dataset into QuickSight to create visualization. Refer to [this example](#) to learn how to create a customize report with Redshift.

Acquisition report

You can use the User acquisition report to get insights into how new users find your website or app for the first time. This report also allows you view the detail user profile.

Note

This article describes the default report. You can customize the report by applying filters or comparisons or by changing the dimensions, metrics, or charts in QuickSight. For more information, refer to [Visualizing data in Amazon QuickSight](#).

View the report

1. Access the dashboard for your application. Refer to [Access dashboard](#).
2. In the dashboard, choose the sheet with name of Acquisition.

Where the data comes from

Acquisition report is created based on the QuickSight dataset of User_Dim_View-<app>-<project>, which connects to the `clickstream_user_dim_view` view in analytics engine (that is, Redshift or Athena). Below is the SQL command that generates the view.

Redshift SQL:

```

user_id,
_first_visit_date AS first_visit_date,
_first_referer AS first_referer,
CASE
  WHEN NULLIF(_first_traffic_source, '') IS NULL THEN '(Direct)'
  ELSE _first_traffic_source
END AS first_traffic_source_source,
_first_traffic_medium AS first_traffic_source_medium,
_first_traffic_source_type AS first_traffic_source_name,
CASE
  WHEN user_id IS NOT NULL THEN 'Registered'
  ELSE 'Non-registered'
END AS registration_status
FROM
  {{schema}}.user_m_view
), first_visit_attr AS (
SELECT
  user_pseudo_id,
  app_info.install_source::VARCHAR AS first_visit_install_source,
  device.system_language::VARCHAR AS first_visit_device_language,
  platform AS first_platform,
  geo.country::VARCHAR AS first_visit_country,
  geo.city::VARCHAR AS first_visit_city
FROM
  {{schema}}.event
WHERE
  event_name IN ('_first_open', '_first_visit')
), device_id AS (
SELECT

```

```

    user_pseudo_id,
    LISTAGG(d_id, ' | ') WITHIN GROUP (ORDER BY user_pseudo_id) AS device_id
FROM (
    SELECT
        user_pseudo_id,
        d_id::VARCHAR
    FROM
        {{schema}}.user_m_view u, u.device_id_list d_id
)
GROUP BY
    user_pseudo_id
)
SELECT
    u.*,
    f.first_visit_install_source,
    f.first_visit_device_language,
    f.first_platform,
    f.first_visit_country,
    f.first_visit_city,
    d.device_id
FROM
    user_base u
LEFT JOIN
    first_visit_attr f ON u.user_pseudo_id = f.user_pseudo_id
LEFT JOIN
    device_id d ON u.user_pseudo_id = d.user_pseudo_id
;

```

Athena SQL:

```

with base as (
    select
        *
    from {{database}}.{{eventTable}}
    where partition_app = ?
        and partition_year >= ?
        and partition_month >= ?
        and partition_day >= ?
),
clickstream_user_dim_mv_1 as (
    SELECT
        user_pseudo_id
        , event_date as first_visit_date

```

```

    , app_info.install_source as first_visit_install_source
    , device.system_language as first_visit_device_language
    , platform as first_platform
    , geo.country as first_visit_country
    , geo.city as first_visit_city
    , (case when nullif(traffic_source.source, '') is null then '(direct)' else
traffic_source.source end) as first_traffic_source_source
    , traffic_source.medium as first_traffic_source_medium
    , traffic_source.name as first_traffic_source_name
from base
where event_name in ('_first_open', '_first_visit')
),

clickstream_user_dim_mv_2 AS (
  select user_pseudo_id,
         count
         (
           distinct user_id
         ) as user_id_count
from base ods
where event_name not in
      (
        '_first_open',
        '_first_visit'
      ) group by 1
)

SELECT upid.*,
       (
         case when uid.user_id_count>0 then 'Registered' else 'Non-registered' end
       ) as is_registered
from clickstream_user_dim_mv_1 as upid left outer join
clickstream_user_dim_mv_2 as uid on upid.user_pseudo_id=uid.user_pseudo_id

```

Dimensions and metrics

The report includes the following dimensions and metrics. You can add more dimensions or metrics by creating calculated field in QuickSight dataset. For more information, refer to [Adding calculated fields](#).

Field	Type	Description	Data source
user_pseudo_id	Dimension	An SDK-generated unique ID for the user	Query from analytics engine
user_id	Dimension	The user ID set via the setUserId API in SDK	Query from analytics engine
device_id	Dimension	The unique ID for the device. You can refer to SDK Manual for how the device id was obtained.	Query from analytics engine
first_visit_date	Dimension	The date that the user first visited your website or first opened the app	Query from analytics engine
first_visit_install_source	Dimension	The installation source when user first opened your app. Blank for web	Query from analytics engine
first_traffic_source_source	Dimension	The traffic source for the user when first visit the app or web	Query from analytics engine
first_traffic_source_medium	Dimension	The traffic medium for the user when first visit the app or web	Query from analytics engine
first_traffic_source_name	Dimension	The traffic campaign name for the user when first visit the app or web	Query from analytics engine

Field	Type	Description	Data source
first_visit_device_language	Dimension	The system language of the device user used when they first opened your app or first visited your website.	Query from analytics engine
first_visit_device_language	Dimension	The system language of the device user used when they first opened your app or first visited your website.	Query from analytics engine
first_platform	Dimension	The platform when user first visited your website or first opened your app	Query from analytics engine
first_referer	Dimension	The referer when user first visited your website	Query from analytics engine
first_visit_country	Dimension	The country where user first visited your website or first opened your app.	Query from analytics engine
first_visit_city	Dimension	The city where user first visited your website or first opened your app.	Query from analytics engine
custom_attr_key	Dimension	The name of the custom attribute key of the user.	Query from analytics engine

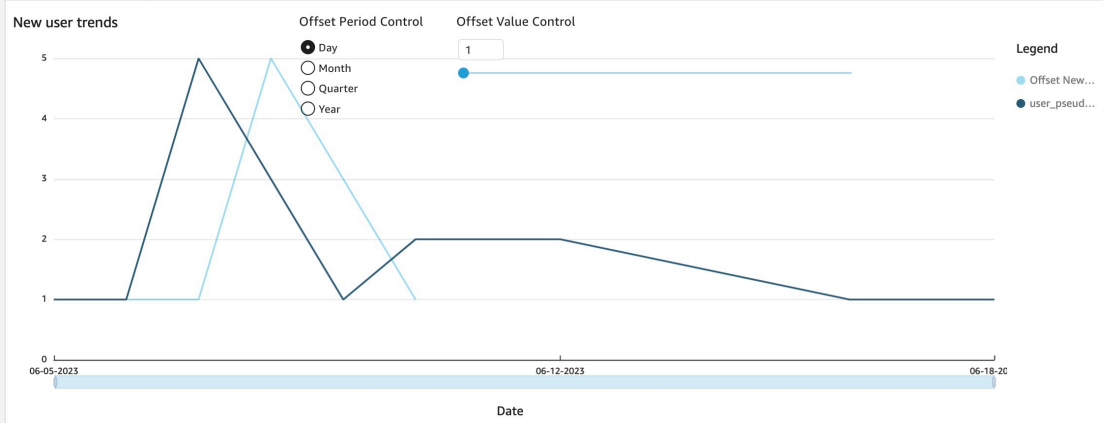
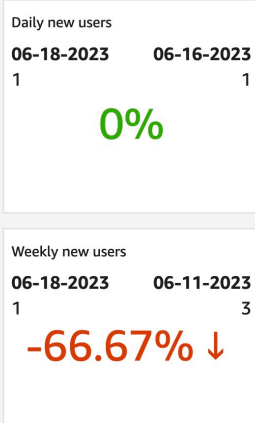
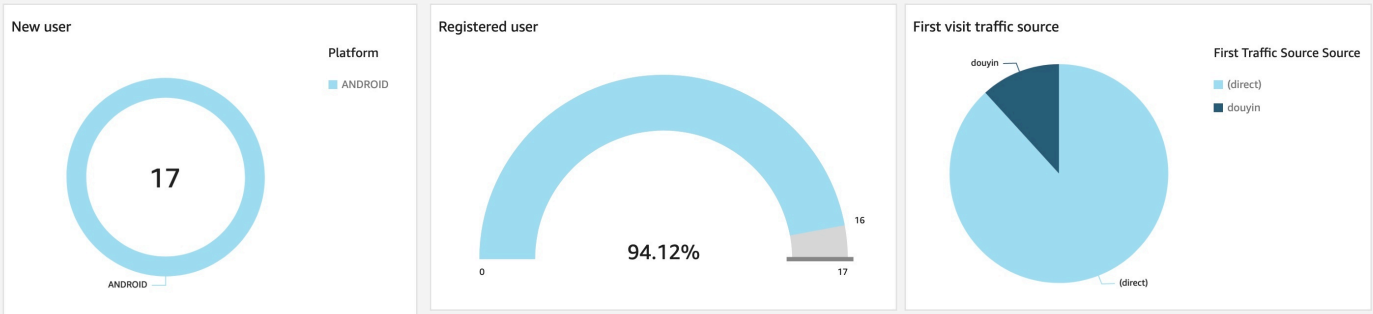
Field	Type	Description	Data source
custom_attr_value	Dimension	The value of the custom attribute key of the user.	Query from analytics engine
registration_status	Dimension	If user had registered or not	Query from analytics engine
Logged-in Rate	Metric	Number of distinct user_id divide by number of distinct user_pseudo_id	Calculated field in QuickSight

Sample dashboard

Below image is a sample dashboard for your reference.

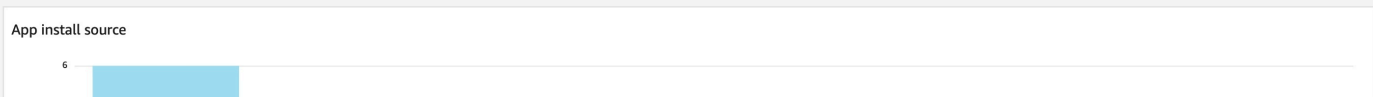
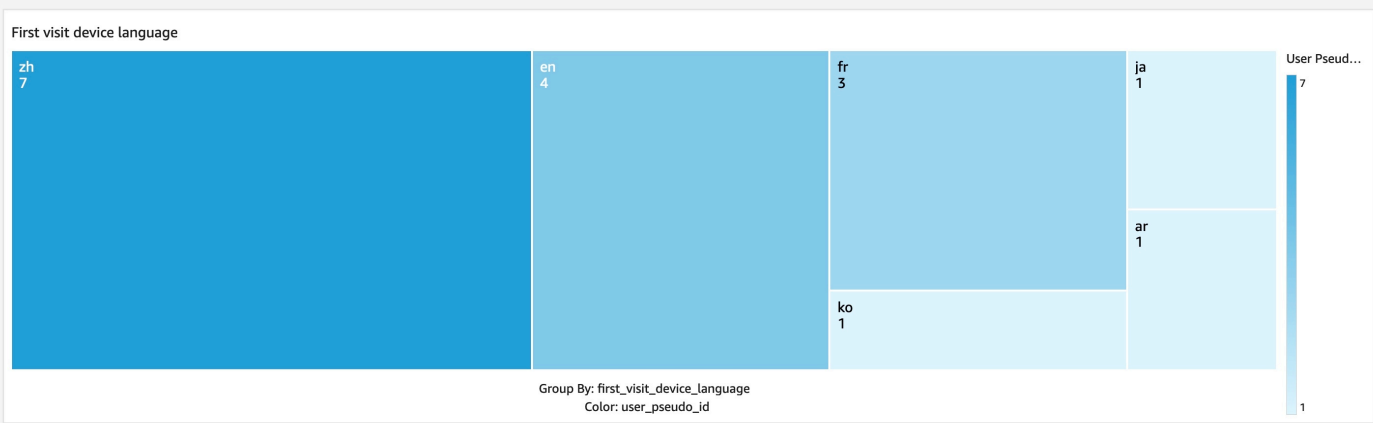
Acquisition report

User acquisition report provides insights into how new users find your website or app for the first time, and help you understand the user profiles and user growth trends. This report mainly based on the user " first visit" event data in view of clickstream user dim.



New user location table

first_visit_country	first_visit_city	user_pseudo_id
Japan	null	1
United States	Ashburn	2
China	Beijing	3
United States	Boardman	1
Brazil	Campinas	1
China	Chengdu	1
Spain	Elche	1
China	Huimin	1
France	Paris	2
China	Shanghai	1
United States	The Bronx	1
China	Yakeshi	2



Engagement report

You can use the Engagement report to get insights into the engagement level of the users when using your websites and apps. This report measures user engagement by the sessions that users trigger and the web pages and app screens that users visit.

Note

This article describes the default report. You can customize the report by applying filters or comparisons or by changing the dimensions, metrics, or charts in QuickSight. For more information, refer to [Visualizing data in Amazon QuickSight](#).

View the report

1. Access the dashboard for your application. Refer to [Access dashboard](#).
2. In the dashboard, choose the sheet with name of Engagement.

Where the data comes from

Engagement report are created based on the QuickSight dataset of Session_View-`<app id>`-`<project id>`, which connects to the `clickstream_session_view_v1` view in analytics engines (that is, Redshift or Athena). Below is the SQL command that generates the view.

Redshift SQL:

```
SELECT
  event.event_id,
  event.event_name,
  event.event_date,
  event.platform,
  event.user_id,
  event.user_pseudo_id,
  event.event_timestamp,
  event_parameter.event_param_key,
  event_parameter.event_param_double_value,
  event_parameter.event_param_float_value,
  event_parameter.event_param_int_value,
  event_parameter.event_param_string_value
FROM {{schema}}.event
```

```

JOIN {{schema}}.event_parameter ON event.event_timestamp =
event_parameter.event_timestamp AND event.event_id = event_parameter.event_id
),

session_part_1 AS (
  SELECT
    es.session_id::VARCHAR,
    user_pseudo_id,
    platform,
    MAX(session_duration) AS session_duration,
    (CASE WHEN (MAX(session_duration) >= 10000 OR SUM(view) >= 1) THEN 1 ELSE 0 END) AS
engaged_session,
    (CASE WHEN (MAX(session_duration) >= 10000 OR SUM(view) >= 1) THEN 0 ELSE 1 END) AS
bounced_session,
    MIN(session_st) AS session_start_timestamp,
    SUM(view) AS session_views,
    SUM(engagement_time) AS session_engagement_time
  FROM
  (
    SELECT
      user_pseudo_id,
      event_id,
      platform,
      MAX(CASE WHEN event_param_key = '_session_id' THEN event_param_string_value ELSE
NULL END) AS session_id,
      MAX(CASE WHEN event_param_key = '_session_duration' THEN event_param_int_value
ELSE NULL END) AS session_duration,
      MAX(CASE WHEN event_param_key = '_session_start_timestamp' THEN
event_param_int_value ELSE NULL END) AS session_st,
      MAX(CASE WHEN (event_param_key = '_engagement_time_msec' AND event_name =
'_user_engagement') THEN event_param_int_value ELSE NULL END) AS engagement_time,
      (CASE WHEN MAX(event_name) IN ('_screen_view', '_page_view') THEN 1 ELSE 0 END)
    AS view
    FROM base_data
    GROUP BY 1,2,3
  ) AS es
  GROUP BY 1,2,3
),

session_part_2 AS (
  SELECT session_id, first_sv_event_id, last_sv_event_id, COUNT(event_id) FROM (
    SELECT
      session_id::VARCHAR,
      event_id,

```

```

    FIRST_VALUE(event_id) OVER(PARTITION BY session_id ORDER BY event_timestamp ASC
ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS first_sv_event_id,
    LAST_VALUE(event_id) OVER(PARTITION BY session_id ORDER BY event_timestamp ASC
ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS last_sv_event_id
FROM (
    SELECT
        event_name,
        event_id,
        event_timestamp,
        MAX(CASE WHEN event_param_key = '_session_id' THEN event_param_string_value
ELSE NULL END) AS session_id
    FROM base_data WHERE event_name IN ('_screen_view','_page_view')
    GROUP BY 1,2,3
)
)
GROUP BY 1,2,3
),

tmp_data AS (
    SELECT event_id, MAX(
        CASE
            WHEN (event_param_key = '_screen_name' OR event_param_key = '_page_title') THEN
event_param_string_value
            ELSE NULL
        END) AS view
    FROM base_data
    GROUP BY 1
),

session_f_sv_view AS (
    SELECT session_f_l_sv.*, t.view AS first_sv_view
    FROM session_part_2 AS session_f_l_sv
    LEFT OUTER JOIN tmp_data t ON session_f_l_sv.first_sv_event_id = t.event_id
),

session_f_l_sv_view AS (
    SELECT session_f_sv_view.*, t.view AS last_sv_view
    FROM session_f_sv_view
    LEFT OUTER JOIN tmp_data t ON session_f_sv_view.last_sv_event_id = t.event_id
)

SELECT
    CASE
        WHEN session.session_id IS NULL THEN CAST('#' AS VARCHAR)

```

```

    WHEN session.session_id = '' THEN CAST('#' AS VARCHAR)
    ELSE session.session_id
END AS session_id,
user_pseudo_id,
platform,
session_duration::BIGINT,
session_views::BIGINT,
engaged_session::BIGINT,
bounced_session,
session_start_timestamp,
CASE
    WHEN session.session_engagement_time IS NULL THEN CAST(0 AS BIGINT)
    ELSE session.session_engagement_time
END::BIGINT AS session_engagement_time,
DATE_TRUNC('day', TIMESTAMP 'epoch' + session_start_timestamp/1000 * INTERVAL '1
second') AS session_date,
DATE_TRUNC('hour', TIMESTAMP 'epoch' + session_start_timestamp/1000 * INTERVAL '1
second') AS session_date_hour,
first_sv_view::VARCHAR AS entry_view,
last_sv_view::VARCHAR AS exit_view
FROM session_part_1 AS session
LEFT OUTER JOIN session_f_l_sv_view ON session.session_id =
    session_f_l_sv_view.session_id
;

```

Athena SQL:

```

with temp_1 as (
    select
        event.event_id,
        event.event_name,
        event.event_date,
        event.platform,
        event.user_id,
        event.user_pseudo_id,
        event.event_timestamp,
        event_parameter.event_param_key,
        event_parameter.event_param_double_value,
        event_parameter.event_param_float_value,
        event_parameter.event_param_int_value,
        event_parameter.event_param_string_value
    from {{database}}.{{eventTable}} as event
    join {{database}}.{{eventParamTable}} as event_parameter

```

```
on event.event_timestamp = event_parameter.event_timestamp and event.event_id =
event_parameter.event_id
where event.partition_app = ?
and event.partition_year >= ?
and event.partition_month >= ?
and event.partition_day >= ?
),
temp_2 as
(
  SELECT
    user_pseudo_id
    ,event_id
    ,platform
    ,max(case when event_param_key = '_session_id' then event_param_string_value else
null end) as session_id
    ,max(case when event_param_key = '_session_duration' then event_param_int_value
else null end) as session_duration
    ,max(case when event_param_key = '_session_start_timestamp' then
event_param_int_value else null end) as session_st
    ,max(case when event_param_key = '_engagement_time_msec' then event_param_int_value
else null end) as engagement_time
    ,(case when max(event_name) in ('_screen_view', '_page_view') then 1 else 0 end) as
view
  FROM temp_1
  group by 1,2,3
),
temp_3 as (
  select
    event_name
    ,event_id
    ,event_timestamp
    ,max(case when event_param_key = '_session_id' then event_param_string_value else
null end) as session_id
    ,(case when max(event_name) in ('_screen_view', '_page_view') then 1 else 0 end) as
view
    from temp_1 where event_name in ('_screen_view','_page_view')
    group by 1,2,3
),
session_part_1 as (
  SELECT
    session_id
    ,user_pseudo_id
    ,platform
    ,max(session_duration) as session_duration
```

```

    ,(case when (max(session_duration)>10000 or sum(view) >1) then 1 else 0 end) as
engaged_session
    ,(case when (max(session_duration)>10000 or sum(view) >1) then 0 else 1 end) as
bounced_session
    ,min(session_st) as session_start_timestamp
    ,sum(view) as session_views
    ,sum(engagement_time) as session_engagement_time
FROM temp_2
GROUP BY 1,2,3
),
session_part_2 as (
    select session_id, first_sv_event_id, last_sv_event_id, count(event_id) from (
        select
            session_id
            ,event_id
            ,first_value(event_id) over(partition by session_id order by event_timestamp asc
rows between unbounded preceding and unbounded following) as first_sv_event_id,
            last_value(event_id) over(partition by session_id order by event_timestamp asc
rows between unbounded preceding and unbounded following) as last_sv_event_id
        from temp_3
    ) group by 1,2,3
),
session_f_sv_view as (
    select
        session_f_l_sv.*,
        t.view as first_sv_view
    from session_part_2 as session_f_l_sv left outer join
    temp_3 as t on session_f_l_sv.first_sv_event_id=t.event_id
),
session_f_l_sv_view as (
    select
        session_f_sv_view.*,
        t.view as last_sv_view
    from session_f_sv_view left outer join
    temp_3 as t on session_f_sv_view.last_sv_event_id=t.event_id
)
select
    CASE
        WHEN session.session_id IS NULL THEN CAST('#' AS VARCHAR)
        WHEN session.session_id = '' THEN CAST('#' AS VARCHAR)
        ELSE session.session_id
    END AS session_id
    ,user_pseudo_id
    ,platform

```

```

,cast(session_duration as bigint) as session_duration
,cast(session_views as bigint) as session_duration
,engaged_session
,bounced_session
,session_start_timestamp
,session_engagement_time
,CASE
  WHEN session.session_engagement_time IS NULL THEN CAST(0 AS BIGINT)
  ELSE session.session_engagement_time
END AS session_engagement_time
,DATE_TRUNC('day', from_unixtime(session_start_timestamp/1000)) as session_date
,DATE_TRUNC('hour', from_unixtime(session_start_timestamp/1000)) as
session_date_hour
,first_sv_view as entry_view
,last_sv_view as exit_view
from session_part_1 as session left outer join
session_f_l_sv_view on session.session_id = session_f_l_sv_view.session_id

```

Dimensions and metrics

The report includes the following dimensions and metrics. You can add more dimensions or metrics by creating calculated field in QuickSight dataset. For more information, refer to [Adding calculated fields](#).

Field	Type	Description	Data source
session_id	Dimension	A SDK-generated unique ID for the session user triggered when using your websites and apps	Query from analytics engine
user_pseudo_id	Dimension	A SDK-generated unique id for the user	Query from analytics engine
platform	Dimension	The platform user used during the session	Query from analytics engine

Field	Type	Description	Data source
session_duration	Dimension	The length of the session in millisecond	Query from analytics engine
session_views	Metric	Number of screen view or page view within the session	Query from analytics engine
engaged_session	Dimension	Whether the session is engaged or not. Engaged session is defined as if the session last more than 10 seconds or have two or more screen views page views	Query from analytics engine
session_start_time stamp	Dimension	The start timestamp of the session	Query from analytics engine
session_engagement_time	Dimension	The total engagement time of the session in millisecond	Query from analytics engine
entry_view	Dimension	The screen name or page title of the first screen or page user viewed in the session	Query from analytics engine
exit_view	Dimension	The screen name or page title of the last screen or page user viewed in the session	Query from analytics engine

Field	Type	Description	Data source
Average engaged session per user	Metric	Average number of session per user in the selected time period	Calculated field in QuickSight
Average engagement time per session	Metric	Average engagement time per session in the selected time period	Calculated field in QuickSight
Average engagement time per user	Metric	Average engagement time per user in the selected time period	Calculated field in QuickSight
Average screen view per user	Metric	Average number of screen views per user in the selected time period	Calculated field in QuickSight

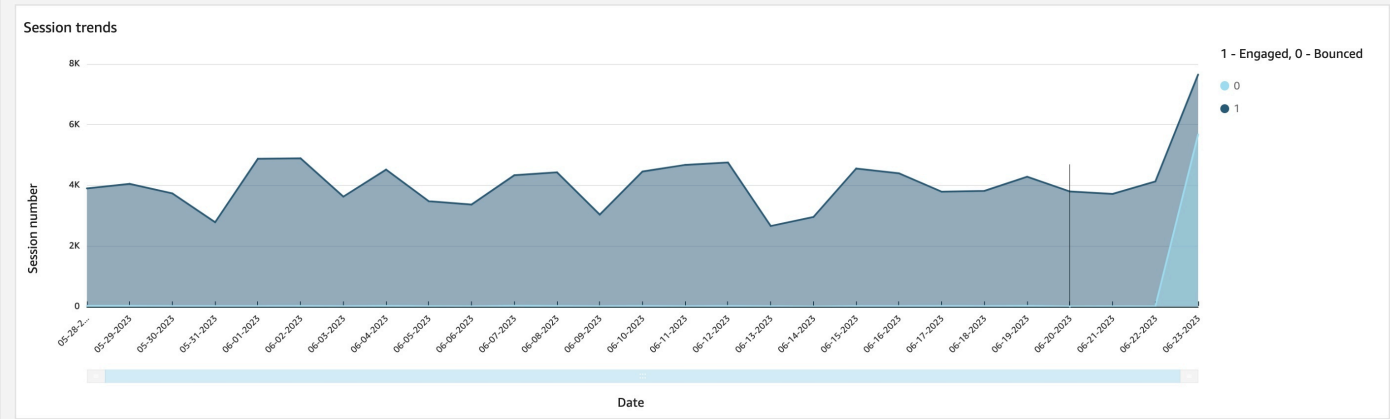
Sample dashboard

Below image is a sample dashboard for your reference.

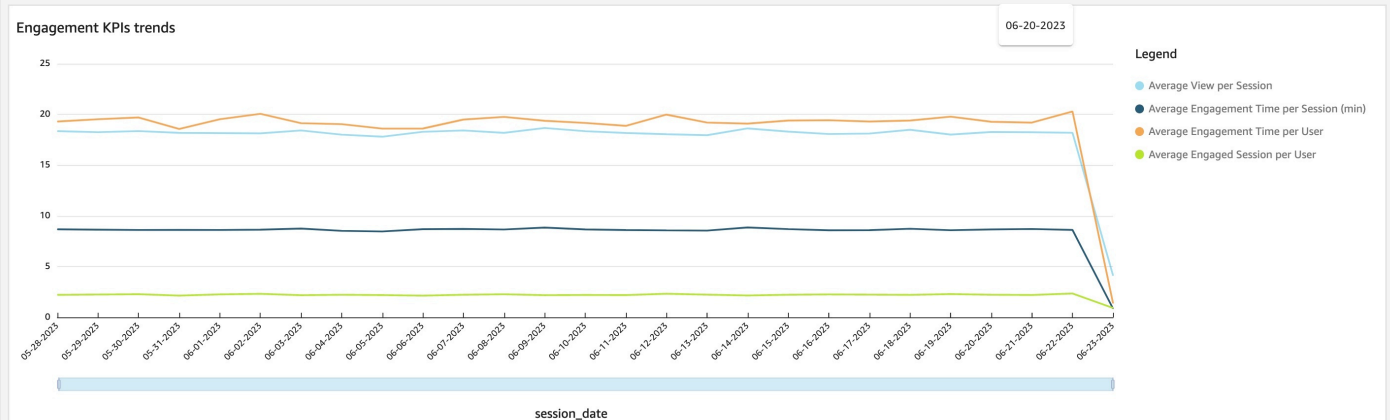
Engagement report

User engagement report provides metrics based on sessions and views that your users trigger when they are using with your apps to help you understand their engagement level. This report mainly based on the the session data that aggregated at view of clickstream session view.

Sessions 06-23-2023 12,102 7,952 ↑	06-22-2023 4,150	Session duration (min) 06-23-2023 6.44 -2.17 ↓	06-22-2023 8.61	Engaged session number 06-23-2023 7,652 3,522 ↑	06-22-2023 4,130	Engagement rate 06-23-2023 61.93% -37.77% ↓	06-22-2023 99.52%
---	---------------------	---	--------------------	--	---------------------	--	----------------------



Average engagement time per user 06-23-2023 1.38 -18.92 ↓	06-22-2023 20.29	Average engaged session per user 06-23-2023 0.91 -1.43 ↓	06-22-2023 2.35	Session engagement time (min) 06-23-2023 0.85 -7.76 ↓	06-22-2023 8.61	Views in session 06-23-2023 4.11 -14.08 ↓	06-22-2023 18.19
--	---------------------	---	--------------------	--	--------------------	--	---------------------



Date	Session number	Engagement rate	Average Engaged Session per User	Average Engagement Time per User	Average Engagement Time per Session (min)	Average View per Session
05-28-2023	3,931	99.46%	2.21	19.31	8.67	18.36
05-29-2023	4,081	99.44%	2.24	19.53	8.64	18.25
05-30-2023	3,760	99.55%	2.28	19.7	8.61	18.36
05-31-2023	2,809	99.32%	2.14	18.57	8.62	18.18
06-01-2023	4,902	99.59%	2.26	19.53	8.61	18.16
06-02-2023	4,919	99.53%	2.31	20.06	8.64	18.13
06-03-2023	3,649	99.53%	2.18	19.14	8.75	18.42
06-04-2023	4,552	99.41%	2.22	19.04	8.52	18.01
06-05-2023	3,496	99.51%	2.19	18.61	8.46	17.81
06-06-2023	3,384	99.59%	2.13	18.61	8.68	18.29
06-07-2023	4,368	99.38%	2.22	19.5	8.71	18.42
06-08-2023	4,458	99.46%	2.27	19.76	8.66	18.19
06-09-2023	3,057	99.41%	2.18	19.37	8.84	18.67
06-10-2023	4,486	99.4%	2.2	19.16	8.66	18.35
06-11-2023	4,694	99.64%	2.19	18.88	8.6	18.17

Maximum

Highest day is Jun 23, 2023 with total count of records of **13,328**

Top screen as session entrance NotepadActivity NoteShareActivity null	Top screen as session exits NotepadActivity null NoteShareActivity
---	--

Engagement report

Activity report

You can use the Activity report to get insights into the activities the users performed when using your websites and apps. This report measures user activity by the events that users triggered, and let you view the detail attributes of the events.

Note

This article describes the default report. You can customize the report by applying filters or comparisons or by changing the dimensions, metrics, or charts in QuickSight. For more information, refer to [Visualizing data in Amazon QuickSight](#).

View the report

1. Access the dashboard for your application. Refer to [Access dashboard](#).
2. In the dashboard, choose the sheet with name of Activity.

Where the data comes from

Activity report are created based on the following QuickSight datasets:

- Events_View-<app id>-<project id> that connects to the `clickstream_event_view_v1` view in analytics engines (that is, Redshift or Athena)
- Events_Parameter_View-<app id>-<project id> that connects to the `clickstream_events_parameter_view_v1` view in analytics engines

Below is the SQL command that generates the related views.

Redshift SQL:

```
SORTKEY(event_date)
AUTO REFRESH YES
AS
select
event_date
, event_name
, event_id
, event_bundle_sequence_id::bigint as event_bundle_sequence_id
```

```
, event_previous_timestamp::bigint as event_previous_timestamp
, event_timestamp
, event_value_in_usd
, app_info.app_id::varchar as app_info_app_id
, app_info.id::varchar as app_info_package_id
, app_info.install_source::varchar as app_info_install_source
, app_info.version::varchar as app_info_version
, app_info.sdk_name::varchar as app_info_sdk_name
, app_info.sdk_version::varchar as app_info_sdk_version
, device.vendor_id::varchar as device_id
, device.mobile_brand_name::varchar as device_mobile_brand_name
, device.mobile_model_name::varchar as device_mobile_model_name
, device.manufacturer::varchar as device_manufacturer
, device.screen_width::bigint as device_screen_width
, device.screen_height::bigint as device_screen_height
, device.carrier::varchar as device_carrier
, device.network_type::varchar as device_network_type
, device.operating_system::varchar as device_operating_system
, device.operating_system_version::varchar as device_operating_system_version
, device.host_name::varchar
, device.ua_browser::varchar
, device.ua_browser_version::varchar
, device.ua_os::varchar
, device.ua_os_version::varchar
, device.ua_device::varchar
, device.ua_device_category::varchar
, device.system_language::varchar as device_system_language
, device.time_zone_offset_seconds::bigint as device_time_zone_offset_seconds
, geo.continent::varchar as geo_continent
, geo.country::varchar as geo_country
, geo.city::varchar as geo_city
, geo.metro::varchar as geo_metro
, geo.region::varchar as geo_region
, geo.sub_continent::varchar as geo_sub_continent
, geo.locale::varchar as geo_locale
, platform
, project_id
, traffic_source.name::varchar as traffic_source_name
, traffic_source.medium::varchar as traffic_source_medium
, traffic_source.source::varchar as traffic_source_source
, event.user_id
, event.user_pseudo_id
, u.user_first_touch_timestamp
from {{schema}}.event
```

```
left join (  
  select  
    *  
  from (  
    select  
      user_pseudo_id,  
      user_first_touch_timestamp  
      ,ROW_NUMBER() over (partition by user_pseudo_id ORDER BY event_timestamp desc)  
    AS et_rank  
    from {{schema}}.user  
  )  
  where et_rank = 1  
) as u on event.user_pseudo_id = u.user_pseudo_id  
;
```

Athena SQL:

```
select  
  event_date  
,event_name  
,event_id  
,event_bundle_sequence_id as event_bundle_sequence_id  
,event_previous_timestamp as event_previous_timestamp  
,event_timestamp  
,event_value_in_usd  
,app_info.app_id as app_info_app_id  
,app_info.id as app_info_package_id  
,app_info.install_source as app_info_install_source  
,app_info.version as app_info_version  
,device.vendor_id as device_id  
,device.mobile_brand_name as device_mobile_brand_name  
,device.mobile_model_name as device_mobile_model_name  
,device.manufacturer as device_manufacturer  
,device.screen_width as device_screen_width  
,device.screen_height as device_screen_height  
,device.carrier as device_carrier  
,device.network_type as device_network_type  
,device.operating_system as device_operating_system  
,device.operating_system_version as device_operating_system_version  
,device.ua_browser  
,device.ua_browser_version  
,device.ua_os  
,device.ua_os_version
```

```

,device.ua_device
,device.ua_device_category
,device.system_language as device_system_language
,device.time_zone_offset_seconds as device_time_zone_offset_seconds
,geo.continent as geo_continent
,geo.country as geo_country
,geo.city as geo_city
,geo.metro as geo_metro
,geo.region as geo_region
,geo.sub_continent as geo_sub_continent
,geo.locale as geo_locale
,platform
,project_id
,traffic_source.name as traffic_source_name
,traffic_source.medium as traffic_source_medium
,traffic_source.source as traffic_source_source
,event.user_id
,event.user_pseudo_id
,u.user_first_touch_timestamp
from {{database}}.{{eventTable}} as event
left join (
  select
    *
  from (
    select
      user_pseudo_id,
      user_first_touch_timestamp
      ,ROW_NUMBER() over (partition by user_pseudo_id ORDER BY event_timestamp desc)
    AS et_rank
    from {{database}}.{{userTable}}
  )
  where et_rank = 1
) as u on event.user_pseudo_id = u.user_pseudo_id
where event.partition_app = ?
and event.partition_year >= ?
and event.partition_month >= ?
and event.partition_day >= ?

```

Dimensions and metrics

The report includes the following dimensions and metrics. You can add more dimensions or metrics by creating calculated field in QuickSight dataset. For more information, refer to [Adding calculated fields](#).

Field	Type	Description	Data source
event_id	Dimension	A SDK-generated unique ID for the event user triggered when using your websites and apps	Query from analytics engine
event_name	Dimension	The name of the event	Query from analytics engine
platform	Dimension	The platform user used during the session	Query from analytics engine
Event User Type	Dimension	The type of user performed the event, that is, new user or existing user	Calculated field in QuickSight
event_date	Metric	The date when the event was logged (YYYYMMDD format in UTC).	Query from analytics engine
event_timestamp	Dimension	The time (in microseconds, UTC) when the event was logged on the client.	Query from analytics engine
app_info_version	Dimension	The version of the app or website when event was logged	Query from analytics engine
event_parameter_key	Dimension	The key of the event parameter	Query from analytics engine

Field	Type	Description	Data source
event_parameter_key	Dimension	The value of the event parameter	Query from analytics engine
User activity number in last 7 days	Metrics	Number of events logged in last 7 days	Calculated field in QuickSight
User activity number in last 30 days	Metrics	Number of events logged in last 30 days	Calculated field in QuickSight
Views	Metrics	Number of events that are _screen_view or _page_view	Calculated field in QuickSight
Screen Time	Metrics	Engagement time (in minutes) on a screen or web page	Calculated field in QuickSight

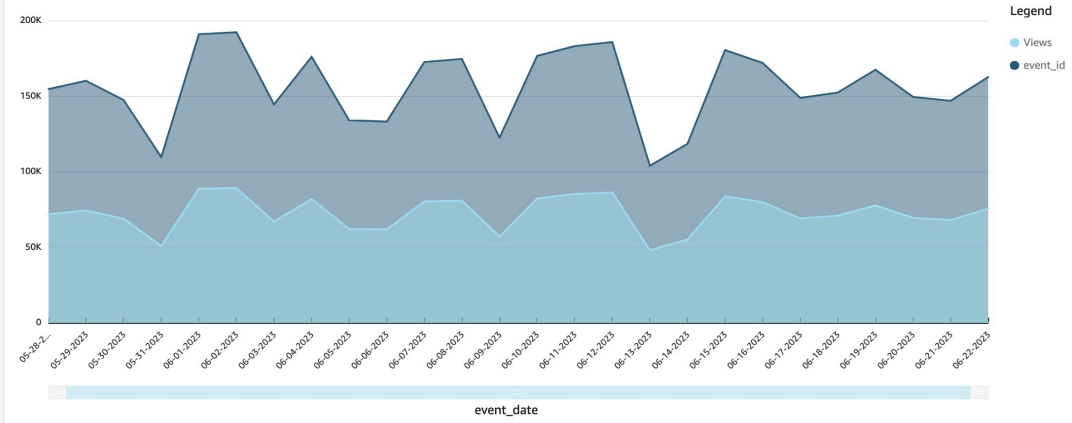
Sample dashboard

Below image is a sample dashboard for your reference.

Activity report

You can use the Activity report to get insights into the activities the users performed when using your websites and apps. This report measures user activity by the events that users triggered, and let you view the detail attributes of the events. This report mainly bases on data in clickstream ods events view and clickstream ods events parameters view.

Activity trends



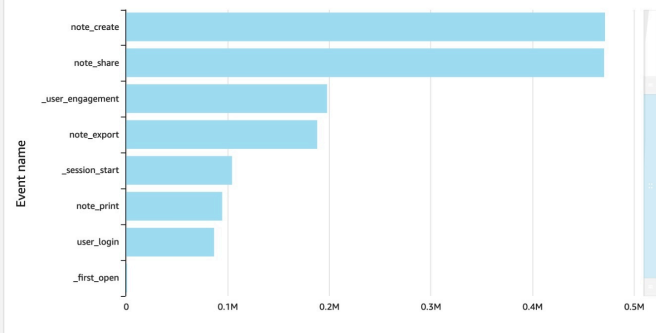
Events
 06-22-2023 162,671
 06-21-2023 146,966

15,705 ↑

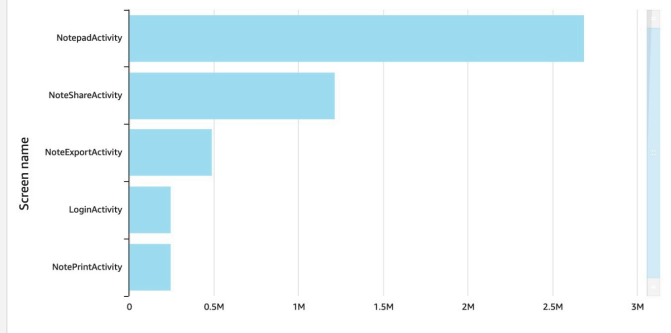
Views
 06-22-2023 75,614
 06-21-2023 68,098

7,516 ↑

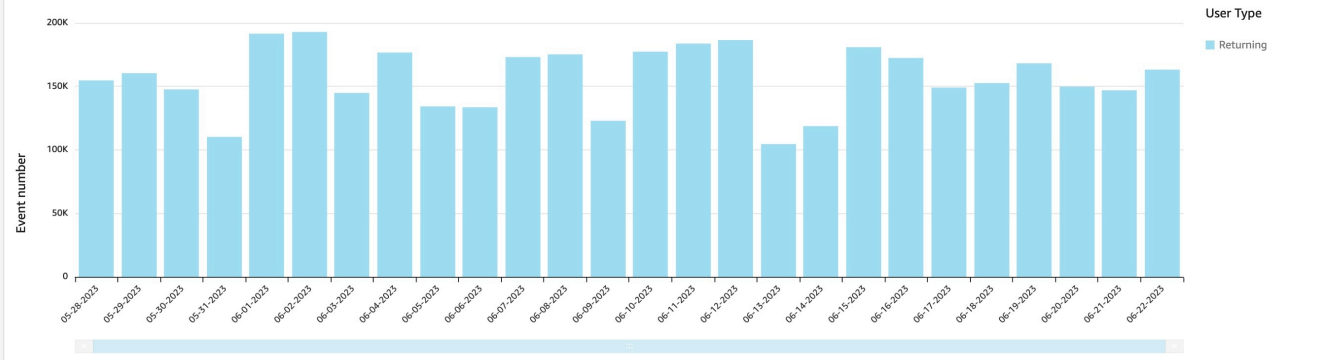
Top events



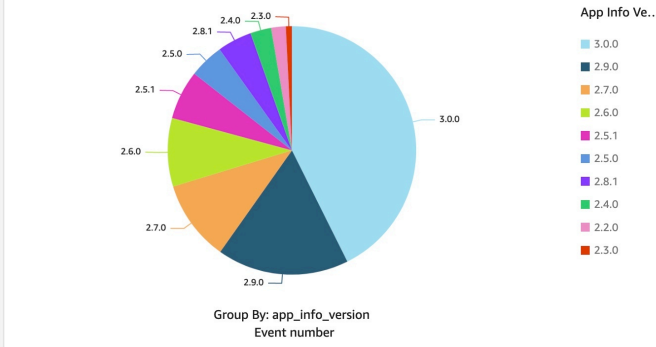
Top screen



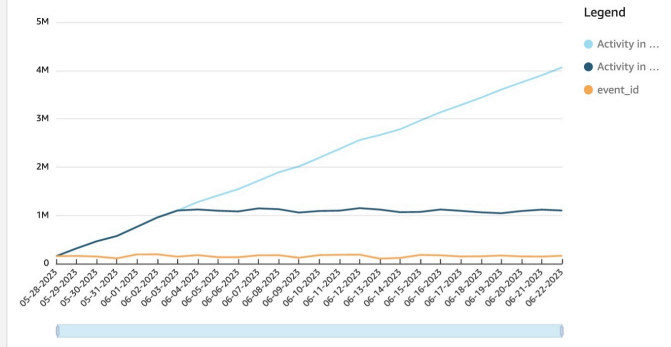
Event number by user type



Events by app version



User activity trends



Activity report

event_id equals: All | event_name equals: All | user_id equals: All | user_pseudo_id equals: All | event_parameter_key equals: All

app_info_package_id equals: All | app_info_version equals: All | app_info_install_source equals: All | User Type equals: All | event_parameter_value equals: All

Event details | Event attributes

Retention report

You can use the Retention report to get insights into how frequently and for how long users engage with your website or mobile app after their first visit. The report helps you understand how well your app is doing in terms of attracting users back after their first visit.

Note

This article describes the default report. You can customize the report by applying filters or comparisons or by changing the dimensions, metrics, or charts in QuickSight. For more information, refer to [Visualizing data in Amazon QuickSight](#).

View the report

1. Access the dashboard for your application. Refer to [Access dashboard](#).
2. In the dashboard, choose the sheet with name of Retention.

Where the data comes from

Retention report are created based on the following QuickSight dataset:

- `lifecycle_weekly_view-<app id>-<project id>`, which connects to the `clickstream_lifecycle_weekly_view_v1` view in analytics engines (that is, Redshift or Athena).
- `lifecycle_daily_view-<app id>-<project id>`, which connects to the `clickstream_lifecycle_daily_view_v1` view in analytics engines (that is, Redshift or Athena).
- `retention_view-<app id>-<project id>` that connects to the `clickstream_retention_view_v1` view in analytics engines

Below is the SQL command that generates the view.

Redshift SQL:

```
clickstream_lifecycle_weekly_view_v1.sql
```

```
select
```

```

    user_pseudo_id,
    DATE_TRUNC('week', dateadd(ms,event_timestamp, '1970-01-01')) as time_period
from {{schema}}.event
where event_name = '_session_start' group by 1,2 order by 1,2),
-- detect if lag and lead exists
lag_lead as (
  select user_pseudo_id, time_period,
         lag(time_period,1) over (partition by user_pseudo_id order by user_pseudo_id,
         time_period),
         lead(time_period,1) over (partition by user_pseudo_id order by user_pseudo_id,
         time_period)
  from weekly_usage),
-- calculate lag and lead size
lag_lead_with_diffs as (
  select user_pseudo_id, time_period, lag, lead,
         datediff(week,lag,time_period) lag_size,
         datediff(week,time_period,lead) lead_size
  from lag_lead),
-- case to lifecycle stage
calculated as (select time_period,
  case when lag is null then '1-NEW'
        when lag_size = 1 then '2-ACTIVE'
        when lag_size > 1 then '3-RETURN'
  end as this_week_value,

  case when (lead_size > 1 OR lead_size IS NULL) then '0-CHURN'
        else NULL
  end as next_week_churn,
  count(distinct user_pseudo_id)
  from lag_lead_with_diffs
  group by 1,2,3)
select time_period, this_week_value, sum(count)
  from calculated group by 1,2
union
select time_period+7, '0-CHURN', -1*sum(count)
  from calculated where next_week_churn is not null group by 1,2;

```

clickstream_lifecycle_daily_view_v1.sql

```

select
  user_pseudo_id,
  DATE_TRUNC('day', dateadd(ms,event_timestamp, '1970-01-01')) as time_period
from {{schema}}.event

```

```

where event_name = '_session_start' group by 1,2 order by 1,2),
-- detect if lag and lead exists
lag_lead as (
  select user_pseudo_id, time_period,
         lag(time_period,1) over (partition by user_pseudo_id order by user_pseudo_id,
time_period),
         lead(time_period,1) over (partition by user_pseudo_id order by user_pseudo_id,
time_period)
  from daily_usage),
-- calculate lag and lead size
lag_lead_with_diffs as (
  select user_pseudo_id, time_period, lag, lead,
         datediff(day,lag,time_period) lag_size,
         datediff(day,time_period,lead) lead_size
  from lag_lead),
-- case to lifecycle stage
calculated as (select time_period,
  case when lag is null then '1-NEW'
        when lag_size = 1 then '2-ACTIVE'
        when lag_size > 1 then '3-RETURN'
  end as this_day_value,

  case when (lead_size > 1 OR lead_size IS NULL) then '0-CHURN'
        else NULL
  end as next_day_churn,
  count(distinct user_pseudo_id)
  from lag_lead_with_diffs
  group by 1,2,3)
select time_period, this_day_value, sum(count)
  from calculated group by 1,2
union
select time_period+1, '0-CHURN', -1*sum(count)
  from calculated where next_day_churn is not null group by 1,2;

```

clickstream_retention_view_v1.sql

```

WITH user_first_date AS (
  SELECT
    user_pseudo_id,
    min(event_date) as first_date
  FROM {{schema}}.event
  GROUP BY user_pseudo_id
),

```

```
retention_data AS (  
SELECT  
    user_pseudo_id,  
    first_date,  
    DATE_DIFF('day', first_date, event_date) AS day_diff  
FROM {{schema}}.event  
JOIN user_first_date USING (user_pseudo_id)  
)  
  
retention_counts AS (  
SELECT  
    first_date,  
    day_diff,  
    COUNT(DISTINCT user_pseudo_id) AS returned_user_count  
FROM retention_data  
WHERE day_diff <= 42 -- Calculate retention rate for the last 42 days  
GROUP BY first_date, day_diff  
)  
  
total_users AS (  
SELECT  
    first_date,  
    COUNT(DISTINCT user_pseudo_id) AS total_users  
FROM user_first_date  
group by 1  
)  
  
retention_rate AS (  
SELECT  
    first_date,  
    day_diff,  
    returned_user_count,  
    total_users  
FROM retention_counts join total_users using(first_date)  
)  
  
SELECT  
*  
FROM retention_rate;
```

Athena SQL:

clickstream-lifecycle-weekly-query.sql

```

with weekly_usage as (
  select
    user_pseudo_id,
    DATE_TRUNC('week', event_date) as time_period
  from {{database}}.{{eventTable}}
  where partition_app = ?
    and partition_year >= ?
    and partition_month >= ?
    and partition_day >= ?
    and event_name = '_session_start' group by 1,2 order by 1,2
),
lag_lead as (
  select user_pseudo_id, time_period,
    lag(time_period,1) over (partition by user_pseudo_id order by user_pseudo_id,
time_period) as lag,
    lead(time_period,1) over (partition by user_pseudo_id order by user_pseudo_id,
time_period) as lead
  from weekly_usage
),
lag_lead_with_diffs as (
  select user_pseudo_id, time_period, lag, lead,
    date_diff('week',lag,time_period) lag_size,
    date_diff('week',time_period,lead) lead_size
  from lag_lead
),
calculated as (
  select time_period,
    case when lag is null then '1-NEW'
      when lag_size = 1 then '2-ACTIVE'
      when lag_size > 1 then '3-RETURN'
    end as this_week_value,
    case when (lead_size > 1 OR lead_size IS NULL) then '0-CHURN'
      else NULL
    end as next_week_churn,
    count(distinct user_pseudo_id) as cnt
  from lag_lead_with_diffs
  group by 1,2,3
)
select time_period, this_week_value, sum(cnt) as cnt from calculated group by 1,2
union
select date_add('day', 7, time_period), '0-CHURN', -1*sum(cnt) as cnt
  from calculated
where next_week_churn is not null

```

```
group by 1,2
```

clickstream-lifecycle-daily-query.sql

```
with daily_usage as (  
  select  
    user_pseudo_id,  
    DATE_TRUNC('day', event_date) as time_period  
  from {{database}}.{{eventTable}}  
  where partition_app = ?  
    and partition_year >= ?  
    and partition_month >= ?  
    and partition_day >= ?  
    and event_name = '_session_start' group by 1,2 order by 1,2  
)  
lag_lead as (  
  select user_pseudo_id, time_period,  
    lag(time_period,1) over (partition by user_pseudo_id order by user_pseudo_id,  
  time_period) as lag,  
    lead(time_period,1) over (partition by user_pseudo_id order by user_pseudo_id,  
  time_period) as lead  
  from daily_usage  
)  
lag_lead_with_diffs as (  
  select user_pseudo_id, time_period, lag, lead,  
    date_diff('day',lag,time_period) lag_size,  
    date_diff('day',time_period,lead) lead_size  
  from lag_lead  
)  
calculated as (  
  select time_period,  
    case when lag is null then '1-NEW'  
      when lag_size = 1 then '2-ACTIVE'  
      when lag_size > 1 then '3-RETURN'  
    end as this_day_value,  
  
    case when (lead_size > 1 OR lead_size IS NULL) then '0-CHURN'  
      else NULL  
    end as next_day_churn,  
    count(distinct user_pseudo_id) as cnt  
  from lag_lead_with_diffs  
  group by 1,2,3  
)
```



```

select time_period, this_day_value, sum(cnt) as cnt
  from calculated group by 1,2
union
select date_add('day', 1, time_period) as time_period, '0-CHURN', -1*sum(cnt) as cnt
  from calculated
  where next_day_churn is not null
  group by 1,2;

```

clickstream-lifecycle-daily-query.sql

```

with base as (
  select
    *
  from {{database}}.{{eventTable}}
  where partition_app = ?
     and partition_year >= ?
     and partition_month >= ?
     and partition_day >= ?
),
user_first_date AS (
  SELECT
    user_pseudo_id,
    min(event_date) as first_date
  FROM base
  GROUP BY user_pseudo_id
),
retention_data AS (
  SELECT
    user_pseudo_id,
    first_date,
    DATE_DIFF('day', first_date, event_date) AS day_diff
  FROM base
  JOIN user_first_date USING (user_pseudo_id)
),
retention_counts AS (
  SELECT
    first_date,
    day_diff,
    COUNT(DISTINCT user_pseudo_id) AS returned_user_count
  FROM retention_data
  WHERE day_diff <= 42 -- Calculate retention rate for the last 42 days

```

```

GROUP BY first_date, day_diff
),

total_users AS (
  SELECT
    first_date,
    COUNT(DISTINCT user_pseudo_id) AS total_users
  FROM user_first_date
  group by 1
),

retention_rate AS (
  SELECT
    first_date,
    day_diff,
    returned_user_count,
    total_users
  FROM retention_counts join total_users using(first_date)
)

SELECT
*
FROM retention_rate

```

Dimensions and metrics

The report includes the following dimensions and metrics. You can add more dimensions or metrics by creating calculated field in QuickSight dataset. For more information, refer to [Adding calculated fields](#).

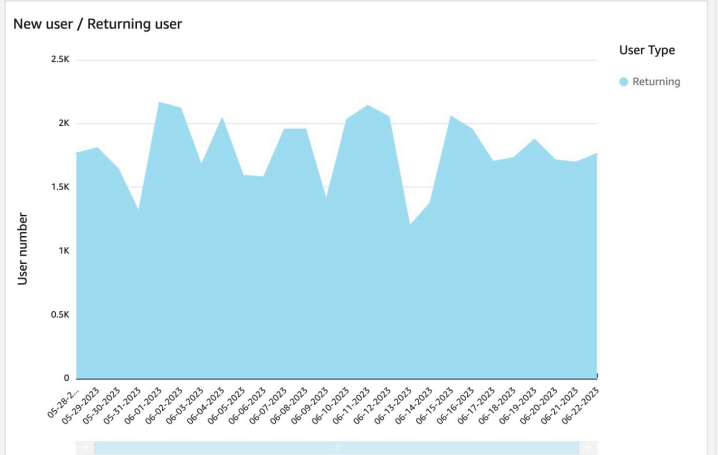
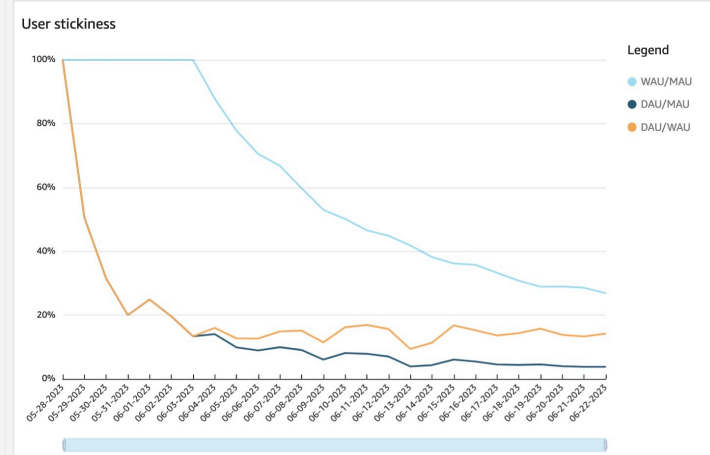
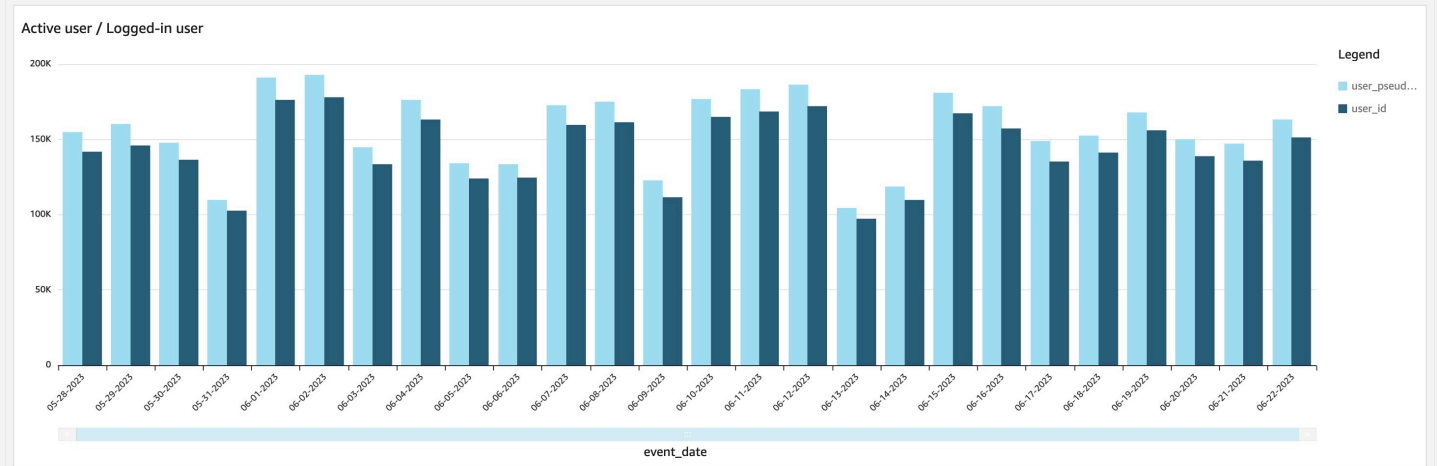
Field	Type	Description	Data source
Daily Active User (DAU)	Metric	Number of active users per date	QuickSight aggregation
Weekly Active User (WAU)	Metric	Number of active users in last 7 days	Calculated field in QuickSight
Monthly Active User (MAU)	Metric	Number of active users in last 30 days	Calculated field in QuickSight

Field	Type	Description	Data source
user_pseudo_id	Dimension	A SDK-generated unique id for the user	Query from analytics engine
user_id	Dimension	The user ID set via the setUserId API in SDK	Query from analytics engine
DAU/WAU	Metric	DAU/WAU % for user stickiness	Calculated field in QuickSight
WAU/MAU	Metric	WAU/MAU % for user stickiness	Calculated field in QuickSight
DAU/MAU	Metric	DAU/MAU % for user stickiness	Calculated field in QuickSight
Event User Type	Dimension	The type of user performed the event, that is, new user or existing user	Calculated field in QuickSight
User first touch date	Metric	The first date that a user use your websites or apps	Calculated field in QuickSight
Retention rate	Metric	Distinct active users number / Distinct active user number by User first touch date	Calculated field in QuickSight
time_period	Dimension	The week or day for the user lifecycle	Query from analytics engine

Field	Type	Description	Data source
this_week_value	Dimension	The user lifecycle stage, that is, New, Active, Return, and Churn	Query from analytics engine
this_day_value	Dimension	The user lifecycle stage, that is, New, Active, Return, and Churn	Query from analytics engine

Sample dashboard

Below image is a sample dashboard for your reference.



User retention
Based on events in the selected period.

First action date	Next action date													Retention R...	
	05-28-2023	05-29-2023	05-30-2023	05-31-2023	06-01-2023	06-02-2023	06-03-2023	06-04-2023	06-05-2023	06-06-2023	06-07-2023	06-08-2023	06-09-2023		06-10-2023
May 28, 2023	100%	26.33%	17.78%	14.21%	21.74%	22.14%	17.33%	19.93%	16.08%	15.01%	18.4%	19.2%	13.36%	21.63%	
May 29, 2023		100%	23.92%	12.67%	22.5%	21.31%	15.72%	18.41%	14.01%	15.28%	20.27%	19.45%	14.61%	21.46%	
May 30, 2023			100%	22.52%	20.83%	20.54%	19.54%	23.71%	16.77%	17.86%	20.44%	18.65%	15.08%	21.33%	
May 31, 2023				100%	37.5%	22.26%	15.55%	21.8%	13.87%	15.09%	21.19%	21.19%	13.87%	23.32%	
Jun 1, 2023					100%	27.65%	14.61%	19.51%	15.69%	17.94%	21.76%	20.2%	13.92%	21.08%	
Jun 2, 2023						100%	22.62%	21.76%	15.7%	16.93%	19.53%	17.06%	14.83%	18.05%	
Jun 3, 2023							100%	33.08%	18.08%	15.38%	21.54%	19.42%	13.65%	17.69%	
Jun 4, 2023								100%	22.75%	13.14%	18.63%	20.59%	11.96%	18.04%	
Jun 5, 2023									100%	24.86%	16.57%	18.51%	14.09%	18.51%	
Jun 6, 2023										100%	30.04%	23.08%	15.75%	19.41%	
Jun 7, 2023											100%	32.27%	8.51%	14.18%	
Jun 8, 2023												100%	23.35%	22.96%	
Jun 9, 2023													100%	31.76%	
Jun 10, 2023														100%	



Device report

You can use the Device report to get insights into the devices that users used when using your apps or websites. The report provides more information for your user profile.

Note

This article describes the default report. You can customize the report by applying filters or comparisons or by changing the dimensions, metrics, or charts in QuickSight. For more information, refer to [Visualizing data in Amazon QuickSight](#).

View the report

1. Access the dashboard for your application. Refer to [Access dashboard](#).
2. In the dashboard, choose the sheet with name of Device.

Where the data comes from

Device reports are created based on the following QuickSight dataset:

- Device_View-<app id>-<project id> , which connects to the `clickstream_device_view_v1` view in analytics engines (that is, Redshift or Athena).

Below is the SQL command that generates the view.

Redshift SQL:

```
select
device.vendor_id::varchar as device_id
, event_date
, device.mobile_brand_name::varchar
, device.mobile_model_name::varchar
, device.manufacturer::varchar
, device.screen_width::int
, device.screen_height::int
, device.carrier::varchar
, device.network_type::varchar
, device.operating_system::varchar
, device.operating_system_version::varchar
```

```
, device.ua_browser::varchar
, device.ua_browser_version::varchar
, device.ua_os::varchar
, device.ua_os_version::varchar
, device.ua_device::varchar
, device.ua_device_category::varchar
, device.system_language::varchar
, device.time_zone_offset_seconds::int
, device.advertising_id::varchar
, device.host_name::varchar
, user_pseudo_id
, user_id
, count(event_id) as usage_num
--please update the following schema name with your schema name
from {{schema}}.event
group by
device_id
, event_date
, device.mobile_brand_name
, device.mobile_model_name
, device.manufacturer
, device.screen_width
, device.screen_height
, device.carrier
, device.network_type
, device.operating_system
, device.operating_system_version
, device.ua_browser
, device.ua_browser_version
, device.ua_os
, device.ua_os_version
, device.ua_device
, device.ua_device_category
, device.system_language
, device.time_zone_offset_seconds
, device.advertising_id
, device.host_name
, user_pseudo_id
, user_id;
```

Athena SQL:

```
select
```

```
    device.vendor_id as device_id
  ,event_date
  ,device.mobile_brand_name
  ,device.mobile_model_name
  ,device.manufacturer
  ,device.screen_width
  ,device.screen_height
  ,device.carrier
  ,device.network_type
  ,device.operating_system
  ,device.operating_system_version
  ,device.ua_browser
  ,device.ua_browser_version
  ,device.ua_os
  ,device.ua_os_version
  ,device.ua_device
  ,device.ua_device_category
  ,device.system_language
  ,device.time_zone_offset_seconds
  ,device.advertising_id
  ,device.host_name
  ,user_pseudo_id
  ,user_id
  ,count(event_id) as usage_num
from {{database}}.{{eventTable}}
where partition_app = ?
    and partition_year >= ?
    and partition_month >= ?
    and partition_day >= ?
group by
  device.vendor_id
  ,event_date
  ,device.mobile_brand_name
  ,device.mobile_model_name
  ,device.manufacturer
  ,device.screen_width
  ,device.screen_height
  ,device.carrier
  ,device.network_type
  ,device.operating_system
  ,device.operating_system_version
  ,device.ua_browser
  ,device.ua_browser_version
  ,device.ua_os
```



```

,device.ua_os_version
,device.ua_device
,device.ua_device_category
,device.system_language
,device.time_zone_offset_seconds
,device.advertising_id
,device.host_name
,user_pseudo_id
,user_id

```

Dimensions and metrics

The report includes the following dimensions and metrics. You can add more dimensions or metrics by creating calculated field in QuickSight dataset. For more information, refer to [Adding calculated fields](#).

Field	Type	Description	Data source
device_id	Dimension	The unique ID for the device, please refer to SDK Manual for how the device id was obtained	QuickSight aggregation
user_pseudo_id	Dimension	A SDK-generated unique id for the user	Query from analytics engine
user_id	Dimension	The user ID set via the setUserId API in SDK	Query from analytics engine
event_date	Dimension	The event data of when the device information was logged	Query from analytics engine
mobile_brand_name	Dimension	The brand name for the device	Query from analytics engine

Field	Type	Description	Data source
mobile_model_name	Dimension	The model name for the device	Query from analytics engine
manufacturer	Dimension	The manufacturer for the device	Query from analytics engine
network_type	Dimension	The network type when user logged the events	Query from analytics engine
operating_system	Dimension	The operating system of the device	Query from analytics engine
operating_system_version	Dimension	The operating system version of the device	Query from analytics engine
screen_height	Dimension	The screen height of the device	Query from analytics engine
screen_width	Dimension	The screen width of the device	Query from analytics engine
Screen Resolution	Dimension	The screen resolution (i.e., screen height x screen width) of the device	Calculated field in QuickSight
system_language	Dimension	The system language of the solution	Query from analytics engine
us_browser	Dimension	The browser derived from user agent	Query from analytics engine
us_browser_version	Dimension	The browser version derived from user agent	Query from analytics engine

Field	Type	Description	Data source
us_os	Dimension	The operating system derived from user agent	Query from analytics engine
us_device	Dimension	The device derived from user agent	Query from analytics engine
us_device_category	Dimension	The device category derived from user agent	Query from analytics engine
usage_num	Metric	Number of event that logged for the device ID	Query from analytics engine

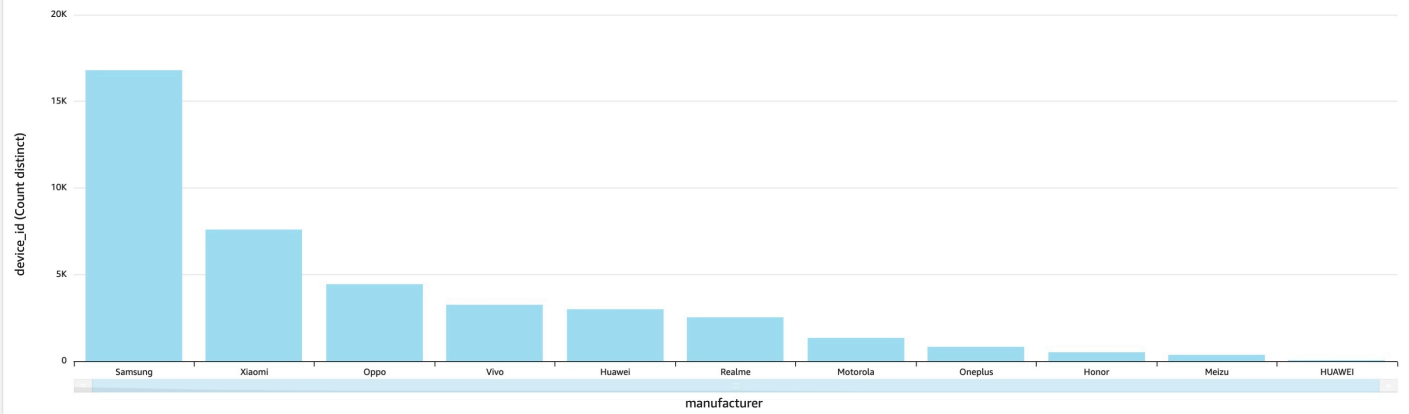
Sample dashboard

Below image is a sample dashboard for your reference.

Device report

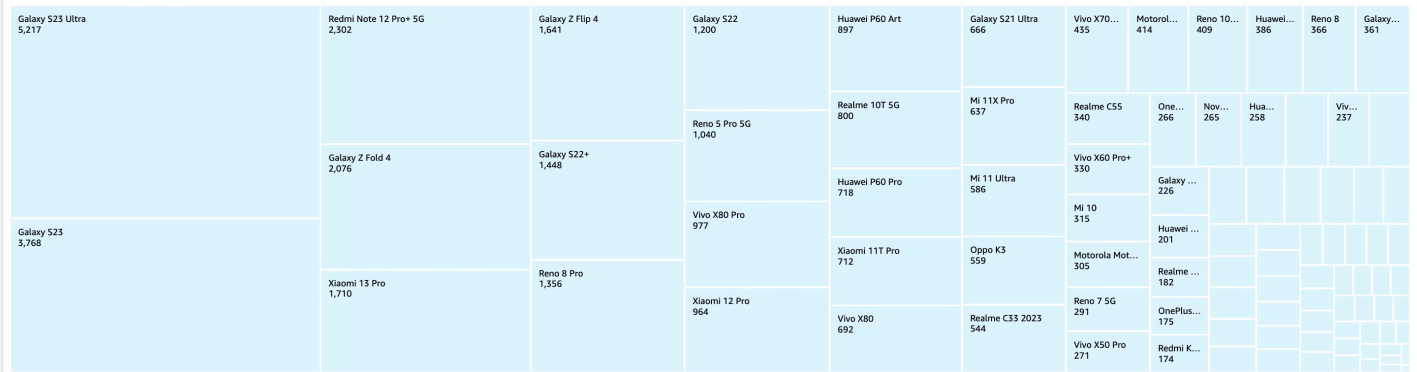
Device report helps you understand what device your users use to access your apps. This report mainly based on data from the view of clickstream_device_view.

Device manufacturer



Device model name

SHOWING TOP 100 IN MOBILE_MODEL_NAME



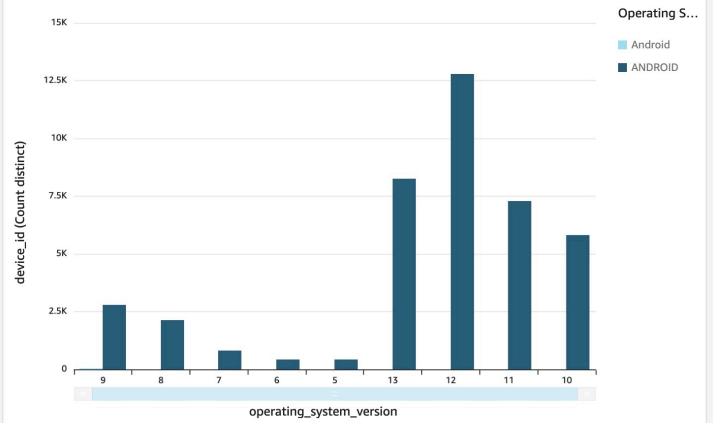
Group By: mobile_model_name
Size: device_id (Count distinct)

Operating System

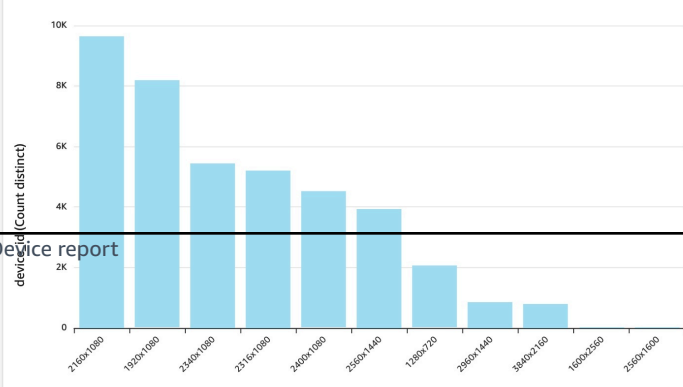


Group By: operating_system
Size: device_id (Count distinct)

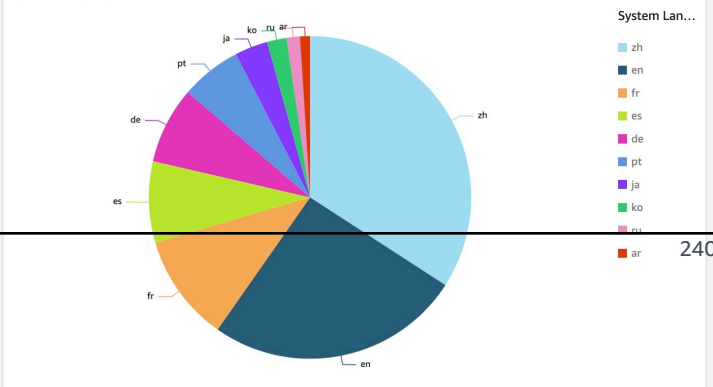
Operating system version



Screen resolution



System language



User report

You can use the User report to query and view an individual user's attributes and the events the user performed.

Note

This article describes the default report. You can customize the report by applying filters or comparisons or by changing the dimensions, metrics, or charts in QuickSight. For more information, refer to [Visualizing data in Amazon QuickSight](#).

View the report

1. Access the dashboard for your application. Refer to [Access dashboard](#).
2. In the dashboard, choose the sheet with name of User.

Where the data comes from

User reports are created based on the following QuickSight dataset:

- User_Dim_View-<app>-<project> which connects to the `clickstream_user_dim_view_v1` view in analytics engine (that is, Redshift or Athena)
- User_Attr_View-<app>-<project> which connects to `clickstream_user_attr_view_v1`.

Below is the SQL command that generates the view.

Redshift SQL:

```
user_id,
_first_visit_date AS first_visit_date,
_first_referer AS first_referer,
CASE
  WHEN NULLIF(_first_traffic_source, '') IS NULL THEN '(Direct)'
  ELSE _first_traffic_source
END AS first_traffic_source_source,
_first_traffic_medium AS first_traffic_source_medium,
_first_traffic_source_type AS first_traffic_source_name,
```

```

CASE
  WHEN user_id IS NOT NULL THEN 'Registered'
  ELSE 'Non-registered'
END AS registration_status
FROM
  {{schema}}.user_m_view
), first_visit_attr AS (
  SELECT
    user_pseudo_id,
    app_info.install_source::VARCHAR AS first_visit_install_source,
    device.system_language::VARCHAR AS first_visit_device_language,
    platform AS first_platform,
    geo.country::VARCHAR AS first_visit_country,
    geo.city::VARCHAR AS first_visit_city
  FROM
    {{schema}}.event
  WHERE
    event_name IN ('_first_open', '_first_visit')
), device_id AS (
  SELECT
    user_pseudo_id,
    LISTAGG(d_id, ' | ') WITHIN GROUP (ORDER BY user_pseudo_id) AS device_id
  FROM (
    SELECT
      user_pseudo_id,
      d_id::VARCHAR
    FROM
      {{schema}}.user_m_view u, u.device_id_list d_id
  )
)
GROUP BY
  user_pseudo_id
)
SELECT
  u.*,
  f.first_visit_install_source,
  f.first_visit_device_language,
  f.first_platform,
  f.first_visit_country,
  f.first_visit_city,
  d.device_id
FROM
  user_base u
LEFT JOIN
  first_visit_attr f ON u.user_pseudo_id = f.user_pseudo_id

```

```
LEFT JOIN
  device_id d ON u.user_pseudo_id = d.user_pseudo_id
;
```

Athena SQL:

```
with base as (
  select
    *
  from {{database}}.{{eventTable}}
  where partition_app = ?
    and partition_year >= ?
    and partition_month >= ?
    and partition_day >= ?
),
clickstream_user_dim_mv_1 as (
  SELECT
    user_pseudo_id
    , event_date as first_visit_date
    , app_info.install_source as first_visit_install_source
    , device.system_language as first_visit_device_language
    , platform as first_platform
    , geo.country as first_visit_country
    , geo.city as first_visit_city
    , (case when nullif(traffic_source.source, '') is null then '(direct)' else
traffic_source.source end) as first_traffic_source_source
    , traffic_source.medium as first_traffic_source_medium
    , traffic_source.name as first_traffic_source_name
  from base
  where event_name in ('_first_open', '_first_visit')
),
clickstream_user_dim_mv_2 AS (
  select user_pseudo_id,
    count
    (
      distinct user_id
    ) as user_id_count
  from base ods
  where event_name not in
    (
      '_first_open',
      '_first_visit'
```

```

    ) group by 1
)

SELECT upid.*,
(
    case when uid.user_id_count>0 then 'Registered' else 'Non-registered' end
) as is_registered
from clickstream_user_dim_mv_1 as upid left outer join
clickstream_user_dim_mv_2 as uid on upid.user_pseudo_id=uid.user_pseudo_id

```

Dimensions and metrics

The report includes the following dimensions and metrics. You can add more dimensions or metrics by creating calculated field in QuickSight dataset. For more information, refer to [Adding calculated fields](#).

Field	Type	What is it	How it's populated
user_pseudo_id	Dimension	An SDK-generated unique id for the user	Query from analytics engine
user_id	Dimension	The user ID set via the setUserId API in SDK	Query from analytics engine
device_id	Dimension	The unique ID for the device. You can refer to SDK Manual for how the device id was obtained	Query from analytics engine
first_visit_date	Dimension	The date that the user first visited your website or first opened the app	Query from analytics engine
first_visit_install_source	Dimension	The installation source when user first opened your app. Blank for web	Query from analytics engine
first_traffic_source_source	Dimension	The traffic source for the user when first visit the app or web	Query from analytics engine

Field	Type	What is it	How it's populated
first_traffic_source_medium	Dimension	The traffic medium for the user when first visit the app or web	Query from analytics engine
first_traffic_source_name	Dimension	The traffic campaign name for the user when first visit the app or web	Query from analytics engine
first_visit_device_language	Dimension	The system language of the device user used when they first opened your app or first visited your website.	Query from analytics engine
first_platform	Dimension	The platform when user first visited your website or first opened your app	Query from analytics engine
first_referrer	Dimension	The referer when user first visited your website	Query from analytics engine
first_visit_country	Dimension	The country where user first visited your website or first opened your app.	Query from analytics engine
first_visit_city	Dimension	The city where user first visited your website or first opened your app.	Query from analytics engine
custom_attr_key	Dimension	The name of the custom attribute key of the user.	Query from analytics engine
custom_attr_value	Dimension	The value of the custom attribute key of the user.	Query from analytics engine
registration_status	Dimension	If user had registered or not	Query from analytics engine

Field	Type	What is it	How it's populated
Event Time (HH:MM:SS)	Dimension	The time in MMDDYYYY HH:MM:SS format when the event was recorded in the client	Calculated field in QuickSight
event_id	Dimension	An SDK-generated unique id for the event user triggered when using your websites and apps	Query from analytics engine
event_name	Dimension	The name of the event	Query from analytics engine
platform	Dimension	The platform user used during the session	Query from analytics engine
event_value_in_usd	Metric	The value in USD associated with the event	Query from analytics engine
app_info_version	Dimension	The app version associated with the event	Query from analytics engine
geo_locale	Dimension	The geo and locale information associated with the event	Query from analytics engine
event_parameter_key	Dimension	The key of the event parameter	Query from analytics engine
event_parameter_value	Dimension	The value of the event parameter	Query from analytics engine
event_date	Metric	The date when the event was logged (YYYYMMDD format in UTC)	Query from analytics engine

Field	Type	What is it	How it's populated
event_timestamp	Dimension	The time (in microseconds, UTC) when the event was logged on the client	Query from analytics engine
app_info_version	Dimension	The version of the app or website when event was logged	Query from analytics engine

Crash report

You can use the Crash report to view the metrics and other information related to the crash events in your app.

This article describes the default report. You can customize the report by applying filters or comparisons or by changing the dimensions, metrics, or charts in QuickSight. For more information, refer to [Visualizing data in Amazon QuickSight](#).

View the report

1. Access the dashboard for your application. Refer to [Access dashboard](#).
2. In the dashboard, choose the sheet with name of Crash.

Where the data comes from

Crash reports are created based on the following QuickSight dataset:

- clickstream_user_dim_view_v1 - Events_View-<app id>-<project id> that connects to the clickstream_event_view_v1 view in analytics engines (that is, Redshift)
- Events_Parameter_View-<app id>-<project id> that connects to the clickstream_events_parameter_view_v1 view in analytics engines

Dimensions and metrics

The report includes the following dimensions and metrics. You can add more dimensions or metrics by creating calculated field in QuickSight dataset. For more information, refer to [Adding calculated fields](#).

Field	Type	What is it	How it's populated
user_pseudo_id	Dimension	An SDK-generated unique id for the user	Query from analytics engine
user_id	Dimension	The user ID set via the setUserId API in SDK	Query from analytics engine
device_id	Dimension	The unique ID for the device, please refer to SDK Manual for how the device id was obtained	Query from analytics engine
Event Time (HH:MM:SS)	Dimension	The time in MMDDYYYY HH:MM:SS format when the event was recorded in the client	Calculated field in QuickSight
event_id	Dimension	An SDK-generated unique id for the event user triggered when using your websites and apps	Query from analytics engine
event_name	Dimension	The name of the event	Query from analytics engine

Field	Type	What is it	How it's populated
platform	Dimension	The platform user used during the session	Query from analytics engine
Crash Rate (by device)	Metric	The percentage of the devices with crash events	Calculated field in QuickSight
app_info_version	Dimension	The app version associated with the event	Query from analytics engine
geo_locale	Dimension	The geo and locale information associated with the event	Query from analytics engine
event_parameter_key	Dimension	The key of the event parameter	Query from analytics engine
event_parameter_value	Dimension	The value of the event parameter	Query from analytics engine
event_date	Metric	The date when the event was logged (YYYYMMDD format in UTC).	Query from analytics engine
event_timestamp	Dimension	The time (in microseconds, UTC) when the event was logged on the client.	Query from analytics engine
app_info_version	Dimension	The version of the app or website when event was logged	Query from analytics engine

Field	Type	What is it	How it's populated
app_info_package_id	Dimension	The package id of the app or website when event was logged	Query from analytics engine
app_info_sdk_name	Dimension	The sdk name when event was logged	Query from analytics engine
app_info_sdk_version	Dimension	The sdk version when event was logged	Query from analytics engine
app_info_package_id	Dimension	The package id of the app or website when event was logged	Query from analytics engine
device_mobile_model_name	Dimension	The model name for the device	Query from analytics engine
device_network_type	Dimension	The network type when user logged the events	Query from analytics engine
device_operating_system	Dimension	The operating system of the device	Query from analytics engine
device_operating_system_version	Dimension	The operating system version of the device	Query from analytics engine

Custom report

One of the key benefits of this solution is that you have complete control over the clickstream data collected from your apps and websites. You have complete flexibility to analyze the data for your specific business needs. This article illustrates the steps of creating a custom report with an example of creating funnel analysis by using Redshift Serverless as analytics engine and QuickSight as reporting tools.

Steps

Creating a custom report mainly consists of two parts: the first part is to prepare the dataset in your analytics engine, and the second part is to create visualization in QuickSight.

Part 1 - Dataset preparation

1. Open **Redshift Serverless dashboard**.
2. Choose the workgroup starting with `clickstream-<project-id>` created by the solution.
3. Choose **Query data**. You will be directed to the Redshift Query Editor.
4. In the **Editor** view on the Redshift Query Editor, right click on the workgroup with name of `clickstream-<project-id>`. In the prompted drop-down, select **Edit connection**, and you will be asked to provide connection parameters. Follow this [guide](#) to use an appropriate method to connect.

Important

Read and write permissions are required for the database (with name as `<project-id>`) to create custom view or table. For example, you can use Admin user to connect to the cluster or workgroup. If you don't know the password for the Admin user, you can reset the admin password in the Redshift Console. For more information, refer to [Security and connections in Amazon Redshift Serverless](#).

5. If it is the first time you access the query editor, you will be prompted to configure the account. Choose **Config account** to open query editor.
6. Add a new SQL editor, and make sure you selected the correct workgroup and schema.
7. Create a new view for funnel analysis. In this example, we used below SQL.

```
CREATE OR REPLACE VIEW notepad.clickstream_funnel_view as
SELECT
platform,
COUNT(DISTINCT step1_id) AS login_users,
COUNT(DISTINCT step2_id) AS add_button_click_users,
COUNT(DISTINCT step3_id) AS note_create_users
FROM (
SELECT
platform,
user_pseudo_id AS step1_id,
```

```
    event_timestamp AS step1_timestamp,
    step2_id,
    step2_timestamp,
    step3_id,
    step3_timestamp
FROM
    notepad.ods_events
LEFT JOIN (
SELECT
    user_pseudo_id AS step2_id,
    event_timestamp AS step2_timestamp
FROM
    notepad.ods_events
WHERE
    event_name = 'add_button_click' )
ON
    user_pseudo_id = step2_id
    AND event_timestamp < step2_timestamp
LEFT JOIN (
SELECT
    user_pseudo_id AS step3_id,
    event_timestamp AS step3_timestamp
FROM
    notepad.ods_events
WHERE
    event_name= 'note_create' )
ON
    step3_id = step2_id
    AND step2_timestamp < step3_timestamp
WHERE
    event_name = 'user_login' )
group by
platform
```

8. Go to QuickSight console, choose **Dataset**, and then choose **New dataset**.
9. In the New Dataset page, choose **Redshift Manual connect** to add dataset, and fill in the prompted form with the following parameters.
 - **Data source name:** clickstream-funnel-view-<project-id>
 - **Connection type:** select VPC connections / VPC Connection for Clickstream pipeline <project-id>
 - **Database server:** input the endpoint url of the serverless workgroup, which you can find on the workgroup console.

- **Port:** 5439
- **Database name:** <project-id>
- **User name:** name of the user you used to created the custom view in previous steps
- **Password:** password of the user you used to created the custom view in previous steps

10. Validate the connection, and then choose **Create data source**.

11. Choose the view from Redshift as data source - "**clickstream_funnel_view**", then

- Schema: select notepad
- Tables: clickstream_funnel_view

Note

When prompted to select Import to SPICE or Directly query your data, select **Directly query your data** for this example.

- Choose **Edit/Preview data** to preview the data. Once you're familiar with the data, choose **PUBLISH & VISUALIZE** at the top-right.

Part 2 - Create visualizations in QuickSight

1. When prompted, select a layout for your visualization.
2. Choose "**+Add**" at the top-left of the screen then choose "**Add visual**".
3. Select a Visual type at the bottom-left of the screen, in this example, select **Vertical bar chart**.
4. In the Field wells, select platform as X axis, login_user, add_button_click_users, and note_create_users as Value.

Now you can publish this analysis as dashboard or continue to format it. For more information, see [Visualizing data in Amazon QuickSight](#).

Part 3 - Add the custom dashboard to Analytics Studio

To enable your custom dashboard to appear in the Dashboards module of the Analytics Studio, you need to add the dashboard into the Shared folder with the name of <project-id>_<app_id>, which was pre-created by the solution.

Exploration

The Exploration module provides a collection of advanced analytics models with flexible and easy-to-use query method, allowing you to derive deeper insights from your clickstream data.

When you want to explore the clickstream data in more details, you can use explorations to:

- perform ad hoc queries
- focus on the most relevant data by using filters on event parameters and user attributes
- group metrics by dimension to make it easy to compare
- easily switch visualization type and drill down or up into data
- view and export summary data into the excel or csv
- save the explorative analytics results into a dashboard for share or regular use

Access Exploration

You can create an ingestion module with the following settings:

To access Explorations, follow below steps:

1. Go to Clickstream Analytics on AWS Console, in the **Navigation Bar**, choose **Analytics Studio**.
2. In the Analytics Studio page that opens, choose **Explorations** in the left navigation panel.

How Exploration works

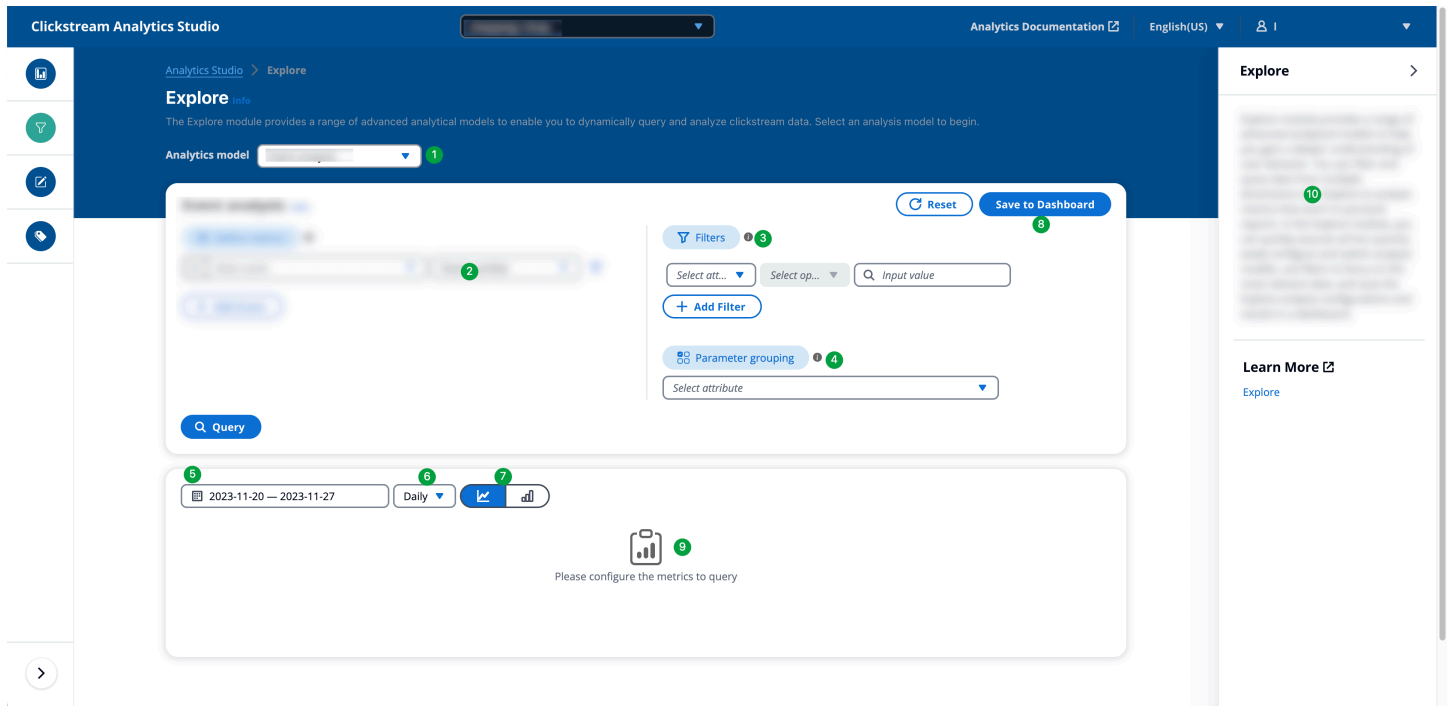
Currently, exploration provides 4 analytics models:

Report name	Description
Event Analysis	Event analysis is used to study the frequency of certain behavioral events. You can conduct multi-dimensional analysis of user behavior through custom metrics, groupings, filters, and various visual charts.
Funnel Analysis	Funnel analysis, or conversion analysis, is mainly used to analyze the conversion status

Report name	Description
	<p>of users in a specified process. The model first breaks down the entire process into steps and then counts the conversion rate from each step to the next. It can be used to measure the performance of each step. Common usage scenarios include analyzing registration conversion rates, purchase conversion rates, etc.</p>
Path Analysis	<p>Path analysis refers to the distribution of the behavior of a group of users after sorting them in sequential order. It is mainly used to analyze and record the distribution ratio of users between pages or events. For example, check how many customers clicked on the product list after opening the app, how many customers viewed the product detail page after visiting landing page, how many customers searched for the product, etc.</p>
Retention Analysis	<p>Retention analysis supports configuring initial event and returning event to calculate the retention or attrition rate of target user groups. It also supports setting associated attributes for initial event and return event.</p>

How to use Exploration

Explorations interface consists of the following components:



- **Analytics Model.** A drop-down list to select or switch analytics model.
- **Model Configuration.** Specifies the configurations for the analytics model, such as select event, adding filters. Each model might have different configuration.
- **Global Filters.** Filters that apply to all the metrics that defined in the Model Configuration. You can add multiple filters and adjust the filter relationship (i.e., 'And' or 'Or').
- **Grouping.** Group the analytics result by specified parameter's value, make it easy for you can compare the metrics at a dimension.
- **Analysis Time Range.** Specifies the time range for the analysis.
- **Aggregation Granularity.** Specifies the level of granularity to display metrics, such by day, week, or month.
- **Visual Type.** Specifies the chart type of the visualization.
- **Save to Dashboard.** Save the exploration analysis
- **Result Display Area.** Show visualization and detail data.
- **Help Panel.** Display additional helpful info when click "Info" icon.

Event Analysis

Event analysis is used to study the frequency of certain behavioral events. You can conduct multi-dimensional analysis of user behavior through custom metrics, groupings, filters, and various visual charts.

Use cases


Event analysis are commonly used when analyzing user behaviors, for example:

- Query on user usage of certain product functions, such as adding favorite, video playback, and view live stream;
- Compare different groups of user behaviors, such as the number of logins per country;
- Compare different channel's effectiveness, such as sign-up rate per traffic source.

Key concept

- **Metric:** perform aggregation on a selected event, such as the number of events, or the number of distinct users generating the event.

How to use event analysis

1. Select an event, and select the aggregation method for the metric.
2. Add filter to the event by clicking the  icon next to the metric.
3. Select an event parameter or user attribute as filter. You can add multiple filters by clicking on the filter icon. You can also configure the filter relationship by choosing AND or OR.
4. Repeat above step to add more metric if needed.
5. If needed, configure global filter by selecting event parameter or user attributes. Similar to event filter, you can add multiple global filters and configure the filter relationship.
6. If needed, configure grouping by selecting an event parameter or user attribute.
7. Choose **Query** to start the analysis.
8. Adjust the data granularity, such as Daily, Weekly, Monthly, if needed.
9. Adjust query time range if needed.
10. Choose **Save to Dashboard** to save the analysis to a Dashboard, enter a name, description, and select a dashboard and sheet.

Example

Calculate the daily page views (PV) and active user count (UV) on the web from different countries over the past month, requiring active users to have a session duration of at least 30,000 milliseconds.

Steps

1. Select the **Event Analysis** model.
2. In the left **Define Metrics** area, choose `_page_view` as the metric for calculating events and select Event number as the metric type.
3. Click the **+ Add Event** button to add another metric. Choose `_app_end` as the metric for calculating events and select User number as the metric type.
4. Click the filter icon to the right of `_app_end` to add a event filter condition:
 - Filter property: `event._session_duration`
 - Operation: `>=`
 - Value: 30000 (the unit of `event._session_duration` is millisecond)
5. Configure a global filter in the right **Filters** area:
 - Choose `other.platform` as the filter property.
 - Operation: `=`
 - Value: Web
6. In the right **Attribute Grouping** area, configure grouping by selecting `geo.country`.
7. In the time selector at the bottom, choose Past Month and click **OK**.
8. Click the **Save to Dashboard** button in the top right corner. In the pop-up dialog, enter:
 - Chart Name: PV and UV
 - Chart Description: PV and UV on the web over the past month (at least 30 seconds)
 - Choose a Dashboard: Select a dashboard. (You need to create a dashboard first. For more information, see [Create dashboard.](#))
 - Choose a Worksheet: Select a worksheet.
 - Click **OK**.

All configurations are shown below:

Funnel Analysis

Funnel analysis, or conversion analysis, is mainly used to analyze the conversion status of users in a specified process. The model first breaks down the entire process into steps and then counts the conversion rate from each step to the next. It can be used to measure the performance of each step.

Use cases

Funnel analysis are commonly used when analyzing the following user behaviors:

- Analysis of the conversion rate of the key process in a product such as order to purchase rate, and registration completion rate;
- Analysis of the conversion rate of promotion such as conversion rate of different in-app promotion spot;
- Analysis of marketing channels's effectiveness such as purchase rate of new users brought by different ad campaigns.

Key concepts

- **Metric:** The entity used for funnel analysis, such as event number or user number.

- **Funnel:** A funnel is a sequence of events that represents a process. It contains at least two events, and each event represents a step in the funnel.
- **Funnel window:** A funnel window refers to the time for the user to complete the entire process. It is considered a successful conversion only when the user completes all the selected steps within the set window period.

How to use funnel analysis

1. Select a metric type.
 - a. User number: calculate the number of distinct users passing through the entire funnel.
 - b. Event number: calculate the number of completion of the entire funnel.
2. Configure the funnel window.
 - a. Custom: you can define any duration as the funnel window.
 - b. The day: complete the funnel within the same date of the first step.
3. Select event for as the step. If needed, choose **+Add Step** to add more steps. You can add up to 10 steps.
4. Choose the filter button to filter the event. Only the events meeting the filter criterial will be considered as passing through the funnel. You can add multiple filters to one event.
5. If needed, configure global filter by selecting event parameter or user attributes. Similar to event filter, you can add multiple global filters and configure the filter relationship.
6. If needed, configure grouping by selecting an event parameter or an user attribute.

Note

The Funnel visualization does not support grouping. If you need to group funnel result, please select bar chart.

1. If you want to only apply the grouping on the first event, toggle on Apply grouping to first step only. If this option is not selected, the grouping will apply to all the steps in the funnel, which means all the events should have parameter or attributes that used to group.
2. Choose **Query** to start the analysis.
3. Adjust the data granularity, such as Daily, Weekly, Monthly, if needed.
4. Adjust query time range if needed.

5. Choose **Save to Dashboard** to save the analysis to a Dashboard. Enter a name, description, and select a dashboard and sheet.

Example

Calculate the conversion rate of users on the web from opening the website -> viewing the product details page -> adding to the shopping cart -> making a payment over the past week.

1. Select the **Funnel Analysis** model.
2. Choose User number as the metric.
3. In the left **Define Funnel** area, choose The Day as the funnel window.
4. Choose `_session_start`, `view_item`, `add_to_cart`, `purchase` as funnel events.
5. Configure a global filter in the right **Filters** area:
 - Choose `other.pl...` as the filter property.
 - Operation: `=`
 - Value: `Web`
6. Click **Query**.

All configurations are as shown in the image below:

The screenshot displays the Clickstream Analytics Studio interface for configuring a Funnel Analysis. The top navigation bar shows the current path 'shopping / shop' and includes links for 'Analytics Documentation' and 'English(US)'. The main configuration area is titled 'Funnel analysis' and includes a 'Reset' button and a 'Save to Dashboard' button. The configuration is divided into several sections:

- Analytics model:** Set to 'Funnel analysis' (highlighted with a red box and '1').
- Metrics:** Set to 'User number' (highlighted with a red box and '2').
- Funnel window:** Set to 'The Day' (highlighted with a red box and '3').
- Funnel steps:** A list of four steps: 1. '_session_start', 2. 'view_item', 3. 'add_to_cart', and 4. 'purchase' (highlighted with a red box and '4').
- Filters:** A filter is configured with the property 'other.pl...', the operation '=', and the value 'Web' (highlighted with a red box and '5').
- Query:** A 'Query' button is visible at the bottom left of the configuration area (highlighted with a red box and '6').

At the bottom of the interface, there is a section for 'Daily' and 'Last 7 day' with a 'Query' button and a message: 'Please configure the metrics to query'.

Path Analysis

Path analysis refers to the distribution of the behavior of a group of users after sorting them in sequential order. It is mainly used to analyze and record the distribution ratio of users between pages or events. For example, you may want to check how many customers clicked on the product list after opening the app, how many customers viewed the product detail page after visiting landing page, how many customers searched for the product.

Use cases

Path analysis are commonly used when analyzing user navigation pattern.

- Analyze the behavioral path distribution of users after entering the product or arriving at landing page
- Analyze the transition steps or screen within a specific processes (such as registration, login, payment) or product modules

Key concepts

- **Metric:** the entity used for path analysis, such as event number or user number.
- **Session:** the length of the path analysis session. Only the events happened within the session can form a path, events happened in different path will not be counted in the same path.
- **Node:** a node in the path can be an event, a screen, or a page.
- **Starting node:** the starting point of the path analysis.

How to use path analysis

1. Select a metric type.
 - a. User number: calculate the number of distinct users passing through the entire path.
 - b. Event number: calculate the number of completion of the entire path.
2. Configure the session.
 - a. Session ID: Use the session generated by the SDK as the session for path analysis. Only the events happened with the same session ID could be counted as a path.
 - b. Custom: you can define any duration as the funnel window.
3. Specify the type of node for the path.

- a. Event name: use event as a node, for example, `_page_view`, `screen_view`.
 - b. Screen name: use the name of a screen in the App as a node.
 - c. Screen ID: use the class ID of a screen in the App as a node.
 - d. Page title: use the page title as a node.
 - e. Page url: use the page url as a node.
4. Set a starting node. The start node will be the first node of the path. Only events happened after the starting node will be counted into the path.
 5. Select nodes to participate in the analysis. Only nodes selected will be highlighted as nodes in the path. You can add up to 10 nodes.
 6. Specify if you want to include other nodes in the path. If this option is toggled on, all the nodes that are not selected will be counted and display as "Other" in the path.
 7. Specify if you want to merge consecutive nodes. If this option is toggled on, nodes that are repeated continuously in the same session will be merged into a single node.
 8. If needed, configure global filter by selecting event parameter or user attributes. Similar to event filter, you can add multiple global filters and configure the filter relationship.
 9. Choose **Query** to start the analysis.
 10. Adjust query time range if needed.
 11. Choose **Save to Dashboard** to save the analysis to a Dashboard. Enter a name, description, and select a dashboard and sheet.

Example

Calculate the distribution of events triggered by customers on the web after the session starts, focusing on login, registration, product exposure, search, viewing products, adding to cart, starting checkout, and placing an order, as well as all other events.

1. Select the **Path Analysis** model.
2. Choose User number as the metric.
3. Choose Session ID as the session definition.
4. Choose Event as the node type.
5. In the left **Select Nodes** area, choose `_session_start` as the starting node.
6. Choose `sign_up`, `login`, `product_exposure`, `search`, `view_item`, `add_to_cart`, `begin_checkout`, `purchase` in sequence as path nodes.

7. Enable **Include Other Events**.
8. Enable **Merge Consecutive Events**.
9. Configure a global filter in the right **Filters** area:
 - Choose other.platform as the filter property.
 - Operation: =
 - Value: Web
10. Choose **Query**.

All configurations are as shown in the image below:

Retention Analysis

Retention rate is a common metric used to assess the user stickiness for an app or website. Retention means that the user returns to your app or website again some time after they used your app. In addition to the standard retention metrics in the default dashboard, Retention Analysis module allows you to select a start event and returning event to customize a retention or attrition rate of a target user group.

Use cases

Retention analysis are commonly used to understand how well your app or website is doing in terms of retaining users.

- Calculate new user retention rate to measure the effectiveness of traffic channel;
- Calculate the active users retention rate to measure the effectiveness for a promotion campaign;
- Compare the repurchase rate for different groups of users to identify the most valuable customers.

Key concept

- **Start:** the event indicates that users start using the app or website.
- **Revisit:** the event indicates that users returning to the app or website.
- **Associated parameter:** Associated parameter are used to keep the value of a parameter consistent between the starting event and the return event. For example, promotion campaign name, page title, or product titles must be the same values for both starting event and return event.

Note


The two associated parameter must both have values, and the value types must be consistent.

- **Retention Rate:** retention rate refers to the relationship between the number of users who perform the specified starting event on the start date (or week, or month depending on the granularity selection) and the number of the same users who perform the specified returning event on the the return date (or week, or month).

How to use retention analysis

1. Select a **Start** event, you can add filter to the event by clicking the filter icon.
2. Select a **Revisit** event, you can add filter to the event by clicking the filter icon.

3. If needed, you can toggle on **Associate parameter**, then select parameters for both starting event and return event.
4. Repeat above step to add more metric if needed.
5. If needed, configure global filter by selecting event parameter or user attributes. Similar to event filter, you can add multiple global filters and configure the filter relationship.
6. If needed, configure grouping by selecting an event parameter or user attribute.
7. Choose **Query** to start the analysis.
8. Adjust the data granularity, such as Daily, Weekly, Monthly, if needed.
9. Specify query time range.

 **Note**

The start time will be the starting point (i.e., day 0) for the retention analysis, and the retention rate % will be calculated against the number of users who performed the specified starting event on the start date (or week, or month depends on the granularity selection).

10. Choose **Save to Dashboard** to save the analysis to a Dashboard. Enter a name, description, and select a dashboard and sheet.

Example

Calculate the retention rate of new customers who downloaded from different app markets on Android one week ago.

1. Select the **Retention Analysis** model.
2. Choose `_first_open` as the start event.
3. Choose `_app_start` as the return event.
4. Configure a global filter in the right **Filters** area:
 - Choose `other.platform` as the filter property.
 - Operation: `=`
 - Value: `Android`
5. In the right **Attribute Grouping** area, configure grouping by selecting `app_info.instal_source`.
6. Click **Query**.

All configurations are as shown in the image below:

Analyses

Analyses module allows you to create and modify dashboards based on the clickstream datasets in a drag-and-drop approach. It provides greater flexibility for users to create business-specific metrics and visualizations. You can use the module to:

- create dashboard that are not provided in preset dashboard or not supported by explorations.
- make changes to the custom dashboard saved from exploration analysis, such as adding calculation fields to calculate custom metrics, adjust visual types etc.
- join clickstream data with external datasets, such as adding item master data to enrich clickstream datasets.

Access Analyses

To access Analyses, follow below steps:

1. Go to **Clickstream Analytics on AWS Console**, in the **Navigation Bar**, click on "**Analytics Studio**", a new tab will be opened in your browser.
2. In the Analytics Studio page, click the **Analyses** icon in the left navigation panel.

How it works

Analyses module is essentially the author interface of QuickSight, in which you have the admin access to all the QuickSight functionalities, for example, create analysis, add or manage datasets, publish and share dashboards.

Note

Only the user with Administrator or Analyst role can access this module.

The solution automatically added the following datasets for each project and app:

Dataset name	Description
Event_View_app_name_project_name	Event data that includes all public event parameters
Event_Parameter_View_app_name_project_name	Events data that includes all private event parameters
User_Dim_View_app_name_project_name	User data that includes all public attributes
User_Attr_View_app_name_project_name	User data that includes all private(custom) attributes
Session_View_app_name_project_name	Data contains measures and dimension about session
Device_View_app_name_project_name	Data contains information about user device
Retention_View_app_name_project_name	Data provides metrics on total users and returned user for each date
Lifecycle_Weekly_View_app_name_project_name	User lifecycle metrics for every week
Lifecycle_Daily_View_app_name_project_name	User lifecycle metrics for every date

To create a custom analysis, you can follow below QuickSight documentation to prepare data and create visualization:

1. [Connecting to data](#)
2. [Preparing data](#)
3. [Visualizing data](#)

Data management

The Data Management module automatically scans the data in your clickstream to generate metadata, making it convenient for you to view and manage all events, event properties, and user properties. You can also modify the display names, descriptions, and data dictionaries of events and properties, making it easy for you to use them in the Exploration module.

Access Data Management

Follow below steps:

1. Go to **Clickstream Analytics on AWS Console**, in the **Navigation Bar**, choose **Analytics Studio**.
2. In the Analytics Studio page that opens, choose **Data Management** in the left navigation panel.

Note

Only the user with Administrator or Analyst role can modify the metadata, such as display name, description.

How it works

The solution automatically scans clickstream data to generate metadata and then stored them in Redshift on a daily basis. There are three type of metadata:

- **Event:** metadata describes clickstream events, which are stored in event_metadata table in Redshift.
- **Event Parameter:** metadata describes clickstream event parameters, which are stored in event_parameter_metadata table in Redshift.

- **User Attribute:** metadata describes user attributes, which are stored in `user_attribute_metadata` table in Redshift.

Metadata dimensions

Below tables list all the dimensions included in each type of the metadata.

Event

Dimension name	Description
Event name	The name of the event reported from SDK
Display name	The display name of the event. By default, it is the same as Event name, user can customize the display name.
Description	The description name of the event reported from SDK. User can customize the display name. For the event automatically collected by the clickstream SDK, the solution has pre-populated description
Source	Describe how the event was collected , Preset indicates the event is automatically collected by SDK, Custom indicates the event is defined and collected by app owner
Platform	Describe which platform the event was collected from, i.e., from Android, Web or iOS
Data volume last day	Describe how much data was collected in last day (in UTC timezone)
SDK version	Describe the version of the SDK that collected the event
Associate preset parameters	The preset event parameters associated with the event

Dimension name	Description
Associate custom parameters	The custom event parameters associated with the event

Event parameters

Dimension name	Description
Parameter name	The name of the event event parameter reported from SDK
Display name	The display name of the event parameter. By default, it is the same as Parameter name, user can customize the display name.
Description	The description name of the event parameter reported from SDK. User can customize the display name. For the event automatically collected by the clickstream SDK, the solution has pre-populated description
Source	Describe how the event parameter was collected, Preset indicates the event parameter is automatically collected by SDK, Custom indicates the event parameter is defined and collected by app owner
Data type	Describe the data type of the event parameter value, e.g., int, string,
Associate event	The event that the event parameters associate d with.
Dictionary	The unique value for the event parameters, user can customize the display value.

User attribute

Dimension name	Description
Attribute name	The name of the user attribute reported from SDK
Display name	The display name of the user attribute. By default, it is the same as Attribute name, user can customize the display name.
Description	The description name of the user attribute reported from SDK. User can customize the display name. For the user attribute automatically collected by the clickstream SDK, the solution has pre-populated description
Source	Describe how the event parameter was collected, Preset indicates the user attribute is automatically collected by SDK, Custom indicates the user attribute is defined and collected by app owner
Data type	Describe the data type of the user attribute value, e.g., int, string,

Update event display name and description

1. Click on the **Events** page, select any event, for example, view_item.
2. Click on the column labeled **Display Name**, enter a name, such as View Product Details, and then click confirm.
3. Go back to Exploration, and in the filter dropdown, you can see view_item now displayed as View Product Details.

Follow the same steps to update display name and description for Event Parameters and User Attributes.

Customize data dictionary for Event Parameter values

1. Click on the **Event Properties** page, and in the search box, select an event parameter, such as `_entrances`.
2. The detailed information for the property should open automatically, choose the **Dictionary** page.
3. Click on the column labeled **Display Value**, enter a new value, such as Non-First Entry for 0, and enter First Entry for 1, then click confirm.
4. Go back to Exploration, and in the filter dropdown, you can see `_entrances` displayed as Non-First Entry and First Entry.

Follow the same steps to customize data dictionary for User Attributes values.

Identity Management

Clickstream Analytics on AWS supports a built-in Cognito user pool or third-party OpenID Connect (OIDC) for user management based on your [deployment type](#).

User management

If you use built-in Cognito for user management, you can find the [Cognito user pool](#) starting with `userPoolIDC9497E0` in your deployment Region. When you deploy the web console of the solution, a user with the required email address will be created as the first user with administrator permission. For more information about user management, refer to [Managing users in your user pool](#). You can also follow [Adding user pool sign-in through a third party](#) to add federated third-party providers, such as SAML and OIDC.

If you are using an OIDC provider, you need to follow the documentation of the OIDC provider to manage users.

User roles

There are four different types of roles that you can assign to users:

Role	Description
Administrator	Have full access to the solution, including identity management
Operator	Manage projects, alarms, and plug-ins
Analyst	View and update in Analytics Studio
Analyst Reader	View in Analytics Studio

The specific features for roles are shown in the following table:

Feature	Administrator	Operator	Analyst	Analyst Reader
Project management	Yes	Yes	No	No
Operation and Alarm	Yes	Yes	No	No
Plugin Management	Yes	Yes	No	No
Identity Management	Yes	No	No	No
Analytics Studio - Dashboards	Yes	No	Yes	Yes
Analytics Studio - Exploration	Yes	No	Yes	Yes
Analytics Studio - Analyses	Yes	No	Yes	No
Analytics Studio - Data Management	Yes	No	Yes	Yes

User role management

By default, the authenticated users do not have a role in the solution. You have two options to manage the user roles in the solution:

Option 1:

Choose **System - Users** in the web console of the solution as Administrator user. Then, add, update, or remove the user roles. This setting has precedence over other settings.

Option 2:

Choose **Setting** in **System - Users** in the web console of the solution as Administrator user. Configure the roles of the solution mapping to the groups or roles in your OIDC provider.

By default, the solution supports mapping [group information from the Cognito user pool](#) to multiple roles in the solution with the following rules:

Group name in Cognito	Solution role
ClickstreamAdmin	Administrator
ClickstreamOperator	Operator
ClickstreamAnalyst	Analyst
ClickstreamAnalystReader	Analyst Reader

For example, you create a group named ClickstreamAnalyst, then add users in the user pool to that group. After those users log in to the solution, the user has an analyst role to access Analyst Studio.

The solution supports mapping multiple groups to a single system role, with various group names separated by commas. For example, by modifying the **Operator Role Name**: Group1,Group2, both user groups can be mapped to the **Operator** role of the system.

If you need to support other OIDC providers, modify **User Role Json Path**.

Example: Modify **User Role Json Path** to \$.payload.realm_access.roles. It can support the mapping of [Keycloak](#) roles to solution roles, where the token format of Keycloak is as follows:

```
{
  "exp": 1701070445,
  "iat": 1701063245,
  "auth_time": 1701062050,
  "jti": "4a892061-56e1-4997-a5f3-84a5d38215f0",
  "iss": "https://keycloak.xxxx.cn/auth/realms/xxx",
  "aud": "P*****Y",
  "sub": "29563a2d-****-43bb-b861-c163da7fe984",
  "typ": "ID",
  "azp": "P*****Y",
  "session_state": "4df36df4-****-4e53-9c1a-43e6d27ffbb9",
  "at_hash": "P*****Y",
  "acr": "0",
```



```
"sid": "4df36df4-****-4e53-9c1a-43e6d27ffbb9",  
"email_verified": false,  
"realm_access": {  
  "roles": [  
    "role1",  
    "role2",  
    "role3",  
  ]  
},  
"preferred_username": "your name",  
"email": "your-name@example.com"  
}
```

Upgrade the solution

Planning and Preparation

1. **Backup of Modified QuickSight Analysis and Dashboard:** The solution upgrade may involve modifying the out-of-the-box analysis and dashboard. In this case, you can back them up following [this documentation](#).
2. **Data Processing Interval** (only applicable to upgrade from v1.0.x): The pipeline upgrade will take about 20 minutes. Make sure no data processing job is running while upgrading the existing pipeline. You can update the existing pipeline to increase the interval and view whether there are running jobs of the EMR Serverless application in the console.

Upgrade Process

Upgrade the web console stack

1. Log in to [AWS CloudFormation console](#), select your existing web console stack, and choose **Update**.
2. Select **Replace current template**.
3. Under **Specify template**:
 - Select Amazon S3 URL.
 - Refer to the table below to find the link for your deployment type.
 - Paste the link in the Amazon S3 URL box.
 - Choose **Next**.

Template	Description
Use Cognito for authentication	Deploy as public service in AWS Regions
Use Cognito for authentication with custom domain	Deploy as public service with custom domain in AWS Regions
Use OIDC for authentication	Deploy as public service in AWS Regions

Template	Description
Use OIDC for authentication with custom domain	Deploy as public service with custom domain in AWS Regions
Use OIDC for authentication within VPC	Deploy as private service within VPC in AWS Regions
Use OIDC for authentication with custom domain in AWS China	Deploy as public service with custom domain in AWS China Regions
Use OIDC for authentication within VPC in AWS China	Deploy as private service within VPC in AWS China Regions

- Under **Parameters**, review the parameters for the template and modify them as necessary. Refer to [Deployment](#) for details about the parameters.
- Choose **Next**.
- On the **Configure stack options** page, choose **Next**.
- On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template might create (IAM) resources.
- Choose **View change set** and verify the changes.
- Choose **Execute change set** to deploy the stack.

You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive an UPDATE_COMPLETE status after a few minutes.

Upgrade the pipeline of project

Important

If you encounter any issues during the upgrade process, refer to [Troubleshooting](#) for more information.

- Log in to the web console of the solution.
- Go to **Projects**, and choose the project to be upgraded.
- Choose project id or **View Details** button.

- In the project details page, choose the **Upgrade** button. You will be prompted to confirm the upgrade action.
- Choose **Confirm**.

You can view the status of the pipeline in the solution console in the **Status** column. After a few minutes, you can receive an Active status.

Post-Upgrade Actions

Migrate the existing data after upgrading from 1.0.x

When you upgraded the pipeline from v1.0.x, you need to perform the following to migrate data from old table `ods_events` to new tables `event`, `event_parameter`, `user`, and `item` in the Redshift:

- Open [Redshift query editor v2](#). For more information, refer to [Working with query editor v2](#) to log in and query data using Redshift query editor v2.

Note

You must use the admin user or the user with schema (known as the app ID) ownership permission.

- Select the Serverless workgroup or provisioned cluster, `<project-id>-><app-id>->Tables`, and make sure tables for the appId are listed there.
- Create a new SQL Editor.
- Execute below SQL in editor.

```
-- please replace `<app-id>` with your actual app id  
CALL "<app-id>".sp_migrate_ods_events_1_0_to_1_1();
```

- Wait for the SQL to complete. The execution time depends on the volume of data in table `ods_events`.
- Execute the below SQL to check the stored procedure execution log, and make sure no errors are found there.

```
-- please replace `<app-id>` with your actual app id
```

```
SELECT * FROM "<app-id>". "clickstream_log" where log_name = 'sp_migrate_ods_events'  
order by log_date desc;
```

7. If your applications no longer use the legacy tables and views, run the SQLs below to clean them to save the storage of Redshift.

```
-- please replace `<app-id>` with your actual app id  
DROP TABLE "<app-id>".dim_users CASCADE;  
DROP TABLE "<app-id>".ods_events CASCADE;  
  
DROP PROCEDURE "<app-id>".sp_clear_expired_events(retention_range_days integer);  
DROP PROCEDURE "<app-id>".sp_upsert_users();  
DROP PROCEDURE "<app-id>".sp_migrate_ods_events_1_0_to_1_1();
```

Frequently Asked Questions

General

Q: What is Clickstream Analytics on AWS?

An AWS Solution that enables customers to build clickstream analytic system on AWS easily. This solution automates the data pipeline creation per customers' configurations with a visual pipeline builder, and provides SDKs for web and mobiles apps (including iOS, and Android) to help customers to collect and ingest client-side data into the data pipeline on AWS. After data ingestion, the solution allows customers to further enrich and model the event data for business users to query, and provides built-in visualizations (for example, acquisition, engagement, retention) to help them generate insights faster.

Data pipeline

Q: When do I choose Amazon Redshift Serverless for data modeling?

If the data pipeline meets the below criteria, Amazon Redshift Serverless is preferred.

- The data processing interval is equal to or larger than one hour.
- The report querying and other usage, such as ETL, is intensive for a few hours at most.
- Redshift is not used for streaming ingestion.
- The estimated cost is lower than the provisioned cluster.

Q: How do I monitor the health of the data pipeline for my project?

You can open the built-in [observability dashboard](#) to view the key metrics of your data pipeline.

The dashboard displays metrics for different components of your data pipeline, including data ingestion, processing, and modeling.

- **Data Ingestion - Server**
 - **Server Request Counts:** The total requests received by the ingestion servers in the given period. You can use it to calculate the request per second (RPS) of the ingestion servers.
 - **Server Response Time:** The average response time in seconds of the ingestion servers in the given period.

- **Server (ECS) Tasks:** The number of tasks/instances running for the ingestion servers.
- **Data Ingestion - Sink - Kinesis Data Stream** (available when enabling KDS as ingestion sink)
 - **Kinesis Throttled and Failed Records:** The total putting records of KDS were throttled or failed in the given period.
 - **Kinesis to S3 Lambda Error count:** The total error count in a given period when sinking records in KDS to S3.
 - **Kinesis to S3 Lambda success rate (%):** The success rate of sinking KDS records to S3.
- **Data Processing** (available when enabling data processing)
 - **Data Processing Job success rate (%):** The success rate of a data processing job in a given period.
 - **Data Processing Row counts:** The chart contains four metrics.
 - **source count:** The raw request count of the ingestion server received in the batch data processing.
 - **flatted source count:** The SDK sends multiple clickstream events in a request in batch. It's the total clickstream events in the processed `source` requests.
 - **sink count:** The total number of valid clickstream events that are transformed and enriched in data processing, and sink to S3 again.
 - **corrupted count:** The total invalid or unprocessable events in the data processing batch. You can check the corrupted file log in the bucket with path `clickstream/<project id>/data/pipeline-temp/<project id>/job-data/etl_corrupted_json_data/jobName=<emr serverless run job id>/` configured for your data pipeline.
- **Data Modeling** (available when enabling data modeling on Redshift)
 - **'Load data to Redshift tables' workflow:** The success or failure count of the workflow loading processed data into Redshift in the given period.
 - **File max-age:** The maximum age of processed files located in S3 is not loaded to Redshift. There is a built-in alarm that will be triggered when the max-age exceeds the data processing interval.
 - **Redshift-Serverless ComputeCapacity** (available when using Redshift serverless): The RPU usage of the Redshift serverless workgroup. If the used RPU count always reaches the maximum RPU number of the Redshift serverless, it means there are insufficient compute resources for the workload in Redshift.

Q: How do I re-run a failed data processing job?

Occasionally, the data processing job fails. You can re-run the failed job to reprocess the data in that given period. The steps are:

1. Locate the failed job in **EMR Studio - Applications - Clickstream-<project id>**.
2. Choose **Clone**.
3. Keep all parameters unchanged, and choose **Submit job run**.

Q: How do I resume a failed data-loading workflow?

This solution uses a workflow named `ClickstreamLoadDataWorkflow`, orchestrated by AWS Step Functions to load the processed data into Amazon Redshift. The workflow uses a DynamoDB table to record the files to be loaded that are processed by the data processing job. Any failure won't lose any data for loading into Redshift. It's safe to execute the workflow again to resume the loading workflow after it fails.

SDK

Q: Could I use other SDK to send data to the pipeline created by this solution?

Yes. The solution supports using third-party SDK to send data to the pipeline. Note that, if you want to enable data processing and modeling module when using a third-party SDK to send data, you need to provide an transformation plugin to map third-party SDK's data structure to solution data schema. Please refer to [Custom plugin](#) for more details.

Analytics Studio

Q: Why is the Analytics Studio is not available?

Possible reasons are:

- The version of the pipeline is not v1.1 or higher. You can try upgrading the pipeline and wait for the upgrade to complete before trying again.
- The reporting module is not enabled on the pipeline.

Q: How can I modify the default dashboard?

You are not allowed to modify the default dashboard directly. However, you can create a new analysis from the default dashboard and then create a new dashboard from the analysis that you copied. Below are the steps to create analysis from the default dashboard:

1. In Analytics Studio, choose the Analyzes module, and choose **Dashboards**.
2. Open the default dashboard with the name of Clickstream Dashboard - - .
3. Choose the **Share** icon and choose **Share Dashboard** in the upper right corner.
4. In the new window, turn on Allow "save as" in the **Save as Analysis** column for "ClickstreamPublishUser" (scroll the window to the right if you don't see the column).
5. Go back to the Dashboard, refresh the webpage, and you should be able to see the **Save as** button in the upper right corner.
6. Choose **Save as**, enter a name for the analysis, and choose **SAVE**. Now you should be able to see a new analysis in the Analyzes, with which now you can edit and publish a new dashboard.

Pricing

Q: How will I be charged and billed for the use of this solution?

The solution is free to use, and you are responsible for the cost of AWS services used while running this solution. You pay only for what you use, and there are no minimum or setup fees. Refer to the Cost section for detailed cost estimation.

Troubleshooting

The following help you to fix errors or problems that you might encounter when using Clickstream Analytics on AWS.

Problem: Deployment failure due to "Invalid Logging Configuration: The CloudWatch Logs Resource Policy size was exceeded"

If you encounter a deployment failure due to creating CloudWatch log group with an error message like the one below,

```
Cannot enable logging. Policy document length breaking Cloudwatch Logs Constraints, either < 1 or > 5120 (Service: AmazonApiGatewayV2; Status Code: 400; Error Code: BadRequestException; Request ID: xxx-yyy-zzz; Proxy: null)
```

Resolution:

[CloudWatch Logs resource policies are limited to 5120 characters](#). The remediation is merging or removing useless policies, then updating the resource policies of CloudWatch logs to reduce the number of policies.

Below is a sample command to reset resource policy of CloudWatch logs:

```
aws logs put-resource-policy --policy-name AWSLogDeliveryWrite20150319 \  
--policy-document '  
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "AWSLogDeliveryWrite2",  
      "Effect": "Allow",  
      "Principal": {  
        "Service": "delivery.logs.amazonaws.com"  
      },  
      "Action": [  

```

```

    "logs:CreateLogStream",
    "logs:PutLogEvents"
  ],
  "Resource": [
  ],
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": "<your AWS account id>"
    },
    "ArnLike": {
      "aws:SourceArn": "arn:aws:logs:<AWS region>:<your AWS account id>:*"
    }
  }
}
]
}
'

```

Problem: Cannot delete the CloudFormation stacks created for the Clickstream pipeline

If you encounter a failure with an error message like the one below when deleting the CloudFormation stacks created for the Clickstream pipeline,

```

Role arn:aws:iam::<your AWS account id>:role/<stack nam>-
ClickStreamApiStackActionSta-<random suffix> is invalid or cannot be
assumed

```

Resolution:

It results from deleting the web console stack for this solution before the CloudFormation stacks are made for the Clickstream pipeline.

Please create a new IAM role with the identical name mentioned in the above error message and trust the CloudFormation service with sufficient permission to delete those stacks.

Note

You can delete the IAM role after successfully removing those CloudFormation stacks.

Problem: Reporting stack(Clickstream-Reporting-xxx) deployment fail

Reporting stack deployment failed with message like

```
Connection attempt timed out
```

And it happened when creating DataSource(AWS::QuickSight::DataSource).

Resolution:

Login solution web console and click "Retry" button in pipeline detail information page.

Problem: Clickstream-DataModelingRedshift-xxxxx stack upgrade failed in UPDATE_ROLLBACK_FAILED

When upgrading from 1.0.x to the latest version, if the CloudFormation stack Clickstream-DataModelingRedshift-xxxxx is in the UPDATE_ROLLBACK_FAILED state, you need to manually fix it by following the steps below.

Resolution:

1. In the Cloudformation **Resource** tab, find the Lambda Function name with logical ID CreateApplicationSchemasCreateSchemaForApplicationsFn.
2. Update the fn_name and aws_region in below script and execute it in a shell terminal. Make sure you have AWS CLI installed and configured.

```
aws_region=<us-east-1> # replace this with your AWS region, and remove '<', '>'
fn_name=<fn_name_to_replace> # replace this with actual function name in step 1 and
remove '<', '>'

cat <<END | > ./index.mjs
export const handler = async (event) => {
  console.log('No ops!')
  const response = {
    Status: 'SUCCESS',
    Data: {
      DatabaseName: '',
```

```
        RedshiftBIUsername: ''
    }
};
return response;
};
END

rm ./noops-lambda.zip > /dev/null 2>&1
zip ./noops-lambda.zip ./index.mjs

aws lambda update-function-code --function-name ${fn_name} \
  --zip-file fileb://./noops-lambda.zip \
  --region ${aws_region} | tee /dev/null
```

3. In the Cloudformation web console, choose **Stack actions** and **Continue update rollback**.
4. Wait until the stack status is UPDATE_ROLLBACK_COMPLETE.
5. Retry the upgrade from the solution web console.

Problem: Can not sink data to MSK cluster, got "InvalidReplicationFactor (Broker: Invalid replication factor)" log in Ingestion Server

If you notice that data can not be sunk into S3 through MSK cluster, and the error message in log of Ingestion Server (ECS) worker task is as below:

```
Message production error: InvalidReplicationFactor (Broker: Invalid replication factor)
```

Resolution:

This is caused by replication factor larger than available brokers, please edit the MSK cluster configuration, set `default.replication.factor` not larger than the total number of brokers.

Problem: data processing job failure

If the data processing job implemented by EMR serverless fails with the below errors:

- `IOException: No space left on device`

```
Job failed, please check complete logs in configured logging destination. ExitCode:
1. Last few exceptions: Caused by: java.io.IOException: No space left on device
Exception in thread "main" org.apache.spark.SparkException:
```

- **ExecutorDeadException**

```
Job failed, please check complete logs in configured logging destination. ExitCode:
1. Last few exceptions: Caused by: org.apache.spark.ExecutorDeadException:
The relative remote executor(Id: 34), which maintains the block data to
fetch is dead. org.apache.spark.shuffle.FetchFailedException Caused by:
org.apache.spark.SparkException: Job aborted due to stage failure: ShuffleMapStage
```

- **Could not find CoarseGrainedScheduler**

```
Job failed, please check complete logs in configured logging destination.
ExitCode: 1. Last few exceptions: org.apache.spark.SparkException: Could not find
CoarseGrainedScheduler.
```

You need to tune the EMR job default configuration. For more information, refer to [configure execution parameters](#).

Problem: data loading workflow failure due to meeting the 25,000 events limit in a single execution history

It's caused by the large volume of data to be loaded or the Redshift load being very high. You could mitigate this error by increasing the compute resources of Redshift (for example, RPU for Redshift serverless) or reducing the [data processing interval](#). Then [restart the data-loading workflow](#).

Uninstall the solution

You will encounter an IAM role missing error if you delete Clickstream Analytics on AWS main stack before you delete the stacks created for Clickstream projects. Clickstream Analytics on AWS console launches additional CloudFormation stacks for the Clickstream pipelines. We recommend you delete projects before uninstalling the solution.

Step 1. Delete projects

1. Go to the Clickstream Analytics on AWS console.
2. In the left sidebar, choose **Projects**.
3. Select the project to be deleted.
4. Choose the **Delete** button in the upper right corner.
5. Repeat steps 3 and 4 to delete all your projects.

Step 2. Delete Clickstream Analytics on AWS stack

1. Go to the [CloudFormation console](#).
2. Find the CloudFormation stack of the solution.
3. Delete the CloudFormation Stack of the solution.
4. (Optional) Delete the S3 bucket created by the solution.
 - a. Choose the CloudFormation stack of the solution, and select the **Resources** tab.
 - b. In the search bar, enter DataBucket. It shows all resources with the name DataBucket created by the solution. You can find the resource type **AWS::S3::Bucket**, and the **Physical ID** field is the S3 bucket name.
 - c. Go to the S3 console, and find the S3 bucket with the bucket name. **Empty** and **Delete** the S3 bucket.

Additional resources

Upload SSL Certificate to IAM

Upload the SSL certificate by running the AWS CLI command `upload-server-certificate` similar to the following:

```
aws iam upload-server-certificate --path /cloudfront/ \  
--server-certificate-name YourCertificate \  
--certificate-body file://Certificate.pem \  
--certificate-chain file://CertificateChain.pem \  
--private-key file://PrivateKey.pem
```

Replace the file names and `Your Certificate` with the names for your uploaded files and certificate. You must specify the `file://` prefix in the `certificate-body`, `certificate-chain` and `private-key` parameters in the API request. Otherwise, the request fails with a `MalformedCertificate: Unknown error message`.

Note

You must specify a path using the `--path` option. The path must begin with `/cloudfront` and must include a trailing slash (for example, `/cloudfront/test/`).

After the certificate is uploaded, the AWS command `upload-server-certificate` returns metadata for the uploaded certificate, including the certificate's Amazon Resource Name (ARN), friendly name, identifier (ID), and expiration date.

To view the uploaded certificate, run the AWS CLI command `list-server-certificates`:

```
aws iam list-server-certificates
```

For more information, see [uploading a server certificate](#) to IAM.

Developer guide

This section provides the source code for the solution.

Source code

Visit our GitHub repository to download the [source code](#) for this solution. The Clickstream Analytics on AWS template is generated using the [AWS Cloud Development Kit \(AWS CDK\) \(CDK\)](#). Refer to the [README.md](#) file for additional information.

Contributors

- Chen, Haiyun
- Deng, Mingtong
- Li, Jingnan
- Li, Min
- Liu, Yong
- Luo, Robin
- Qiao, Wei
- Qian, Yang
- Que, Mingfei
- Zhu, Mengxin
- Zhu, Xiaowei

Revisions

Date	Change
July 2023	Initial release.
December 2023	Released version 1.1.0 <ul style="list-style-type: none">• Added Analytics Studio to provide a unified interface for analysts to view and query clickstream data.• Added Flutter SDK that simplifies collecting data from the Flutter App.

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents AWS current product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. AWS responsibilities and liabilities to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Clickstream Analytics on AWS is licensed under the terms of the Apache License Version 2.0 available at [The Apache Software Foundation](https://www.apache.org/licenses/LICENSE-2.0).