Hands-On Tutorial

Build Flutter Mobile App Part Two

Build Flutter Mobile App Part Two: Hands-On Tutorial

Table of Contents

Introduction 1
Overview 1
Prerequisites 1
What you will accomplish 1
Implementation 2
Modules 2
Module 1: Clone the Flutter app 3
Overview 1
What you will accomplish 1
Implementation 2
Conclusion 10
Module 2: Add the Past Trips feature 11
Overview 1
What you will accomplish 1
Implementation 2
Conclusion 10
Module 3: Add the Activity feature 36
Overview 1
What you will accomplish 1
Implementation 2
Conclusion 10
Module 4: Add the Profile feature 93
Overview 1
What you will accomplish 1
Implementation 2
Conclusion 10
Congratulations! 127
Clean up resources 128

Build a Flutter Mobile App Using AWS Amplify - Part 2

Use nested data and Amplify functions to create a trip planner app for iOS and Android

Overview

In this how-to guide, you will continue building the cross-platform Flutter mobile app we started in the first tutorial in this series. The app is a trip planner where users can create a trip and set its name, destination, and dates. Additionally, they can upload a banner image for the trip.

You will introduce multiple new features to the App, such as allowing users to add activities for their trips, set their categories and dates, and upload a file and image for the activity. You will create an Amplify function to create the user's profile data and you will allow them to use the App to update their profile, change their name, and set their home city.

You will clone the app from GitHub using the terminal in the first module. Then you will update the app to introduce additional features like displaying past trips, adding activities to a trip, and editing the profile of the app's user.

Prerequisites

- An AWS account: If you don't already have an account, follow the Setting Up Your AWS Environment tutorial for a quick overview.
- Install and configure the Amplify CLI.
- Install and configure Flutter.
- Install and configure Git.
- A text editor combined with Flutter's command-line tools. For this guide we will use VSCode, but you can use your preferred IDE.

What you will accomplish

This how-to guide will walk you through creating an app to help users plan trips. You will:

- Clone the app we built in an earlier how-to guide from GitHub
- Use the Amplify CLI to create an Amplify backend for this app

- Update the app to display past trips
- Create a data model for the trip's activities and the user's profile, and use the GraphQL API to synchronize to the Amplify backend

Implementation

AWS Experience	Beginner
Time to Complete	160 minutes
Cost to Complete	Free Tier eligible
Last updated	September 12, 2023

Modules

This how-to guide is divided into the following modules. You must complete each module before moving to the next one.

1. Module 1: Clone the Flutter app (30 minutes): Clone the Flutter app from Github, update its dependencies, and create an Amplify backend.
2. Module 2: Add the Past Trips feature (40 minutes): Implement logic and UI to display past trips, and introduce a navigation drawer to the app to allow the users to navigate the different pages you will introduce in this guide.
3. Module 3: Add the Activity feature (45 minutes): Add and display the activities of a trip using an Amplify GraphQL API.
4. Module 4: Add the Profile feature (45 minutes): Create a profile for the user using an Amplify function and implement the logic and UI of creating, updating, and displaying the profile in the app.

Module 1: Clone the Flutter app

Overview

In this module, we will start by cloning the Flutter app we built in the first tutorial of this series from GitHub. Then we will add additional dependencies to the app and use the Amplify CLI to provision a cloud backend for the app. And finally, we will improve the readability and maintainability of the app by refactoring some of its code.

What you will accomplish

In this module, you will:

- Clone the Flutter app
- Add additional dependencies to the app
- Create an Amplify backend for the app

Implementation

Minimum time to complete	30 minutes

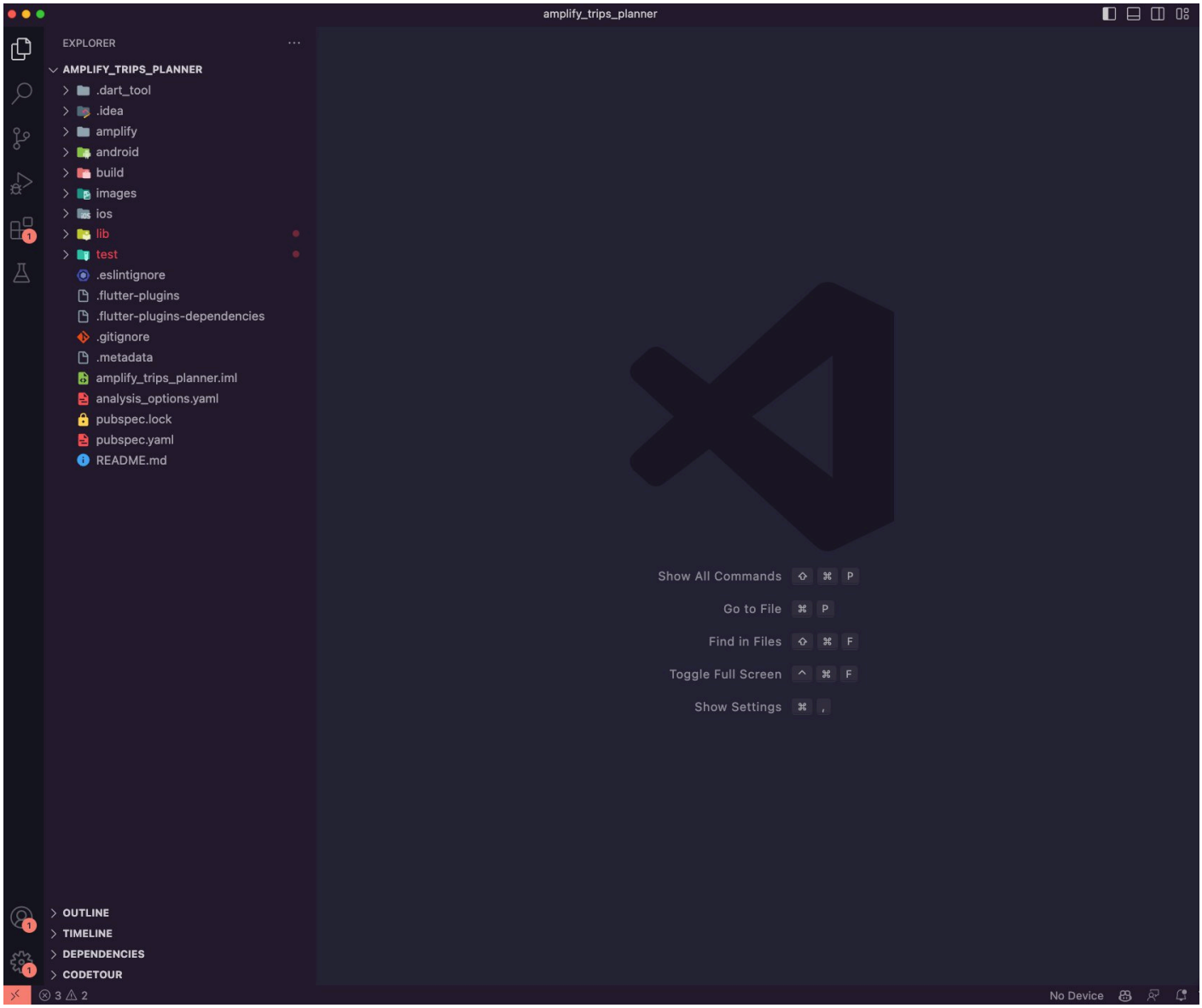
Clone the amplify_trips_planner app

1. Clone the Flutter app by running the command below in your terminal.

```
git clone https://github.com/aws-samples/amplify-trips-planner.git
```

2. Open the newly cloned Flutter app using VSCode. You can do that by running the commands below in your terminal.

```
cd amplify_trips_planner
code . -r
```



Add the app dependencies

1. The app will use additional dependencies for its new functionalities, such as uploading and displaying files and creating a timeline for the activities. Update the **pubspec.yaml** file in the application root directory to add the following packages under dependencies.

```
file_picker: ^5.0.0
timelines: ^0.1.0
url_launcher: ^6.1.5
```

The file **pubspec.yaml** should look like this:

```
name: amplify_trips_planner
description: A new Flutter project.
version: 1.0.0+1

environment:
 sdk: ">=3.0.2 <4.0.0"

dependencies:
 amplify_api: ^1.0.0
 amplify_auth_cognito: ^1.0.0
 amplify_authenticator: ^1.0.0
 amplify_flutter: ^1.0.0
 amplify_storage_s3: ^1.0.0
 cached_network_image: ^3.2.3
 cupertino_icons: ^1.0.5
 flutter:
   sdk: flutter
 flutter_riverpod: ^2.1.3
 go_router: ^7.0.0
 image_picker: ^0.8.0
 intl: ^0.18.0
 path: ^1.8.3
 riverpod_annotation: ^2.0.1
 uuid: ^3.0.7
 file_picker: ^5.0.0
 timelines: ^0.1.0
 url_launcher: ^6.1.5

dev_dependencies:
 amplify_lints: ^3.0.0
 build_runner:
 custom_lint:
 flutter_test:
   sdk: flutter
 riverpod_generator: ^2.1.3
 riverpod_lint: ^1.1.5

flutter:
 uses-material-design: true
 assets:
```

```
    - images/amplify.png
```

Step 2: Run the following command in your terminal to install the dependencies you added to the **pubspec.yaml** file.

```
flutter pub get
```

Create an Amplify backend for the app

1. Navigate to the app's root folder, and provision an Amplify backend for the app by running the following command in your terminal.

```
amplify init
```

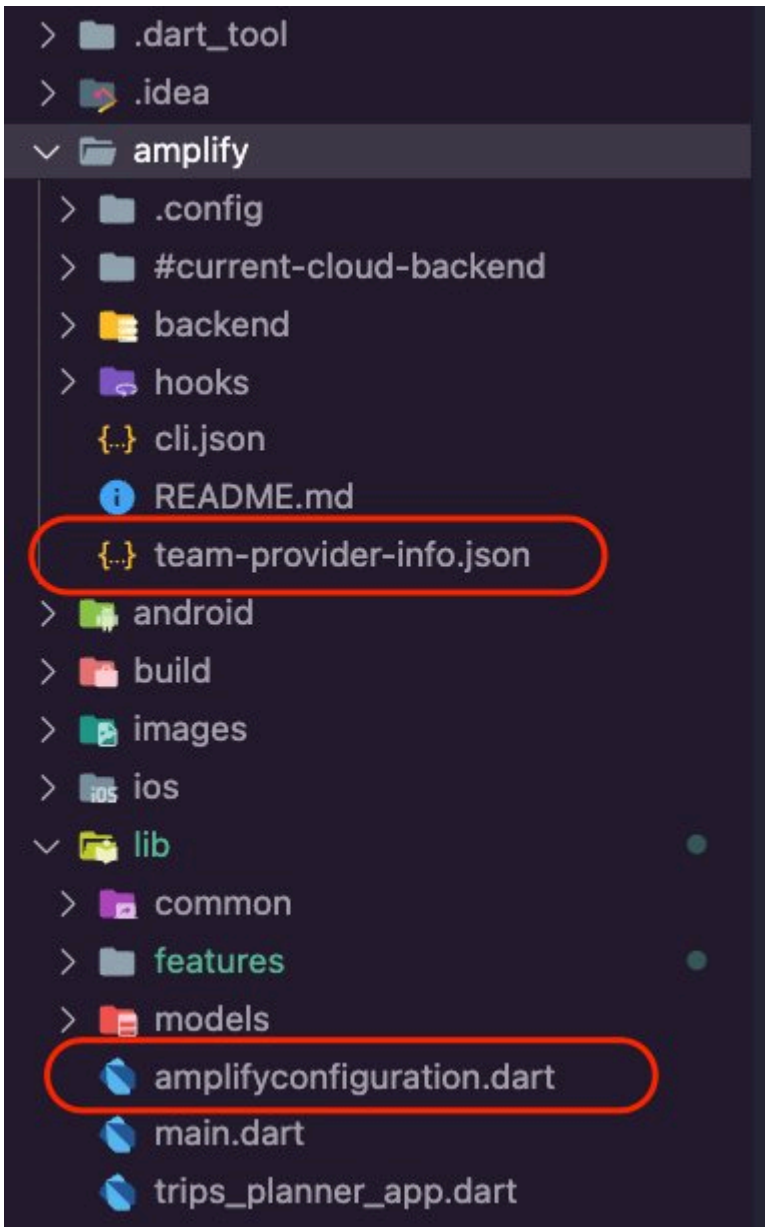
2. Accept the auto-generated option for the environment name and choose the default editor. In this guide, we are using VSCode. Then select the AWS authentication method; we are using an AWS profile. Finally, choose the profile you want to use.

```
Note: It is recommended to run this command from the root of your app directory
? Enter a name for the environment dev
? Choose your default editor: Visual Studio Code
Using default provider  awscloudformation
? Select the authentication method you want to use: AWS profile

For more information on AWS Profiles, see:
https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-profiles.html

? Please choose the profile you want to use default
```

Press **Enter.** The Amplify CLI will initialize the backend and connect the project to the cloud. Once complete, you will get a confirmation, as shown in the screenshot.

```
Deployment state saved successfully.
⠿ Initializing your environment: dev⚠ WARNING: owners may reassign ownership for the
following model(s) and role(s): Trip: [owner]. If this is not intentional, you may wa
nt to apply field-level authorization rules to these fields. To read more: https://do
cs.amplify.aws/cli/graphql/authorization-rules/#per-user--owner-based-data-access.
✅ GraphQL schema compiled successfully.

Edit your schema at /Users/malakamm/development/dev_update/part2/amplify_trips_planne
r/amplify/backend/api/amplifytripsplanner/schema.graphql or place .graphql files in a
 directory at /Users/malakamm/development/dev_update/part2/amplify_trips_planner/ampl
ify/backend/api/amplifytripsplanner/schema
✔ Initialized provider successfully.
✅ Initialized your environment successfully.

Your project has been successfully initialized and connected to the cloud!
```

The Amplify CLI will add a new file **team-provider-info.json** to the **amplify** folder, which contains the Amplify backend details. It will also add a new dart file **amplifyconfiguration.dart** to the **lib/** folder. The app will use this file to know how to reach your provisioned backend resources at runtime.



3. In the previous how-to guide, you added the following categories to the app:

   - **Amplify Auth:** allows users to sign up, sign in, and manage their account
   - **Amplify API:** allows users to create, read, update, and delete trips
   - **Amplify Storage:** allows users to upload and view images in their app

   Run the command **amplify push** to create the resources of the above categories in the cloud.

```
   Current Environment: dev

| Category | Resource name                | Operation | Provider plugin     |
| Api      | amplifytripsplanner          | Create    | awscloudformation   |
| Auth     | amplifytripsplanner1ec293ff  | Create    | awscloudformation   |
| Storage  | s3b101c2c7                   | Create    | awscloudformation   |

? Are you sure you want to continue? (Y/n) ›
```

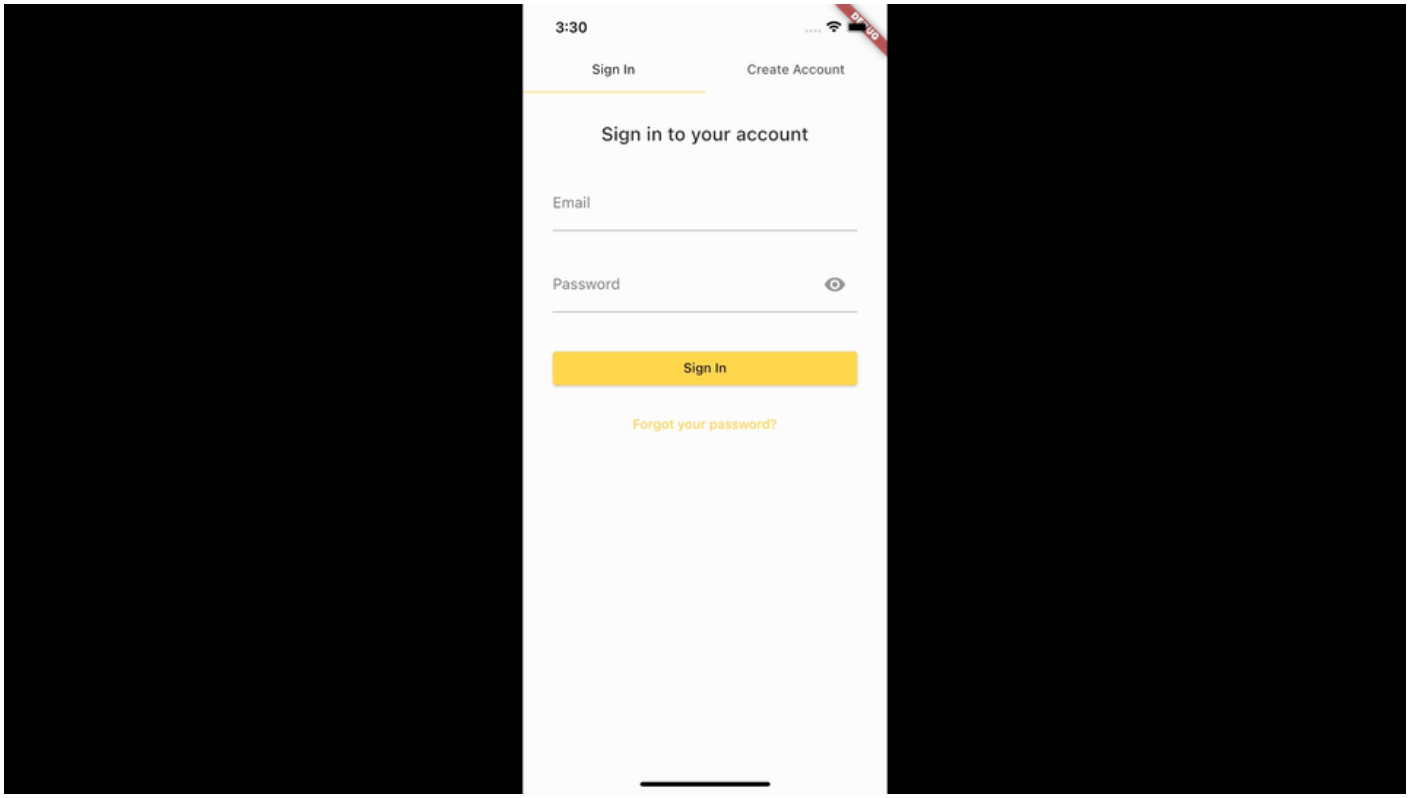
4. Press **Enter**. The Amplify CLI will deploy the resources and display a confirmation, as shown in the screenshot.

```
        S3AuthPrivatePolicy             AWS::IAM::Policy                CREATE_IN_PR
        S3GuestPublicPolicy             AWS::IAM::Policy                CREATE_IN_PR
        S3AuthUploadPolicy              AWS::IAM::Policy                CREATE_IN_PR
        S3GuestReadPolicy               AWS::IAM::Policy                CREATE_IN_PR
        S3AuthPublicPolicy              AWS::IAM::Policy                CREATE_IN_PR
        S3AuthProtectedPolicy           AWS::IAM::Policy                CREATE_IN_PR
        S3AuthReadPolicy                AWS::IAM::Policy                CREATE_IN_PR

Deployment state saved successfully.

GraphQL endpoint: https://qmrokxalkng4xdaegffe4g2nsq.appsync-api.us-west-1.amazonaws.
com/graphql

GraphQL transformer version: 2
```

5. Run the app in an emulator or simulator and try the following:

   - Create a new account
   - Create a new trip
   - Edit the newly created trip
   - Upload an image for the trip
   - Delete the trip

   The following is an example using an iPhone simulator.



Conclusion

You learned how to clone a Flutter app from GitHub in this module. You also learned how to use the Amplify CLI to provision a cloud backend for the app.

Module 2: Add the Past Trips feature

Overview

In this module, you will implement logic and UI to display past trips. You will also add a navigation drawer to the app allowing the users to navigate the different pages you will introduce in this tutorial.

What you will accomplish

In this module, you will:

- Add a navigation drawer to the app
- Add the trip **TripGridViewItem** widget
- Implement the past trips list page
- Implement the selected past trip details page

Implementation

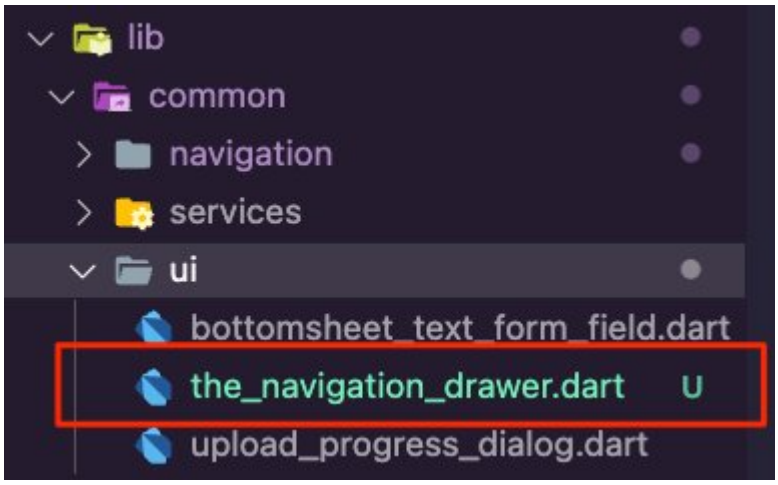
Minimum time to complete	40 minutes

Add the navigation drawer to the app

1. Open the file **lib/common/navigation/router/routes.dart.** Update it to add **pastTrips** and **pastTrip** enum values. The file **routes.dart** should look like the following:

```
enum AppRoute {
 home,
 trip,
 editTrip,
 pastTrips,
 pastTrip,
}
```

2. Create a new dart file inside the folder **lib/common/ui** and name it the **the_navigation_drawer.dart**.



3. Open **the_navigation_drawer.dart** file and update it with the following code to create the options to navigate to the trip's route and the past trip's route.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class TheNavigationDrawer extends ConsumerWidget {
 const TheNavigationDrawer({
   super.key,
 });

 @override
 Widget build(BuildContext context, WidgetRef ref) {
   return Drawer(
     child: ListView(
       padding: EdgeInsets.zero,
       children: [
         const DrawerHeader(
           decoration: BoxDecoration(
             color: Color(constants.primaryColorDark),
           ),
           padding: EdgeInsets.all(16),
           child: Column(
             children: [
               SizedBox(height: 10),
```

```
              Text(
                'Amplify Trips Planner',
                style: TextStyle(fontSize: 22, color: Colors.white),
              ),
            ],
          ),
        ),
        ListTile(
          leading: const Icon(Icons.home),
          title: const Text('Trips'),
          onTap: () {
            context.goNamed(
              AppRoute.home.name,
            );
          },
        ),
        ListTile(
          leading: const Icon(Icons.category),
          title: const Text('Past Trips'),
          onTap: () {
            context.goNamed(
              AppRoute.pastTrips.name,
            );
          },
        ),
      ],
    ),
  );
 }
}
```

4. Open the **lib/features/trip/ui/trips_list/trips_list_page.dart** file and update it to add the navigation drawer.

```
drawer: const TheNavigationDrawer(),
```

The **trips_list_page.dart** should look like the following code snippet.

```
import 'package:amplify_trips_planner/common/ui/the_navigation_drawer.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/trip/controller/
trips_list_controller.dart';
```

```
import 'package:amplify_trips_planner/features/trip/ui/trips_gridview/
trips_list_gridview.dart';
import 'package:amplify_trips_planner/features/trip/ui/trips_list/
add_trip_bottomsheet.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

class TripsListPage extends ConsumerWidget {
const TripsListPage({
super.key,
});

Future<void> showAddTripDialog(BuildContext context) =>
  showModalBottomSheet<void>(
    isScrollControlled: true,
    elevation: 5,
    context: context,
    builder: (sheetContext) {
      return const AddTripBottomSheet();
    },
  );

@override
Widget build(BuildContext context, WidgetRef ref) {
final tripsListValue = ref.watch(tripsListControllerProvider);
return Scaffold(
  appBar: AppBar(
    centerTitle: true,
    title: const Text(
      'Amplify Trips Planner',
    ),
    backgroundColor: const Color(constants.primaryColorDark),
  ),
  drawer: const TheNavigationDrawer(),
  floatingActionButton: FloatingActionButton(
    onPressed: () {
      showAddTripDialog(context);
    },
    backgroundColor: const Color(constants.primaryColorDark),
    child: const Icon(Icons.add),
  ),
  body: TripsListGridView(
    tripsList: tripsListValue,
  ),
```

```
);
}
}
```

5. Open the **lib/features/trip/ui/trip_page/trip_page.dart** file and update it to add the navigation drawer.

```
drawer: const TheNavigationDrawer(),
```

The **trip_page.dart** should look like the following code snippet.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/ui/the_navigation_drawer.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/trip/controller/trip_controller.dart';
import 'package:amplify_trips_planner/features/trip/ui/trip_page/trip_details.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class TripPage extends ConsumerWidget {
const TripPage({
required this.tripId,
super.key,
});
final String tripId;

@override
Widget build(BuildContext context, WidgetRef ref) {
final tripValue = ref.watch(tripControllerProvider(tripId));

return Scaffold(
  appBar: AppBar(
    centerTitle: true,
    title: const Text(
      'Amplify Trips Planner',
    ),
    actions: [
      IconButton(
        onPressed: () {
          context.goNamed(
            AppRoute.home.name,
          );
```

```
        },
        icon: const Icon(Icons.home),
      ),
    ],
    backgroundColor: const Color(constants.primaryColorDark),
  ),
  drawer: const TheNavigationDrawer(),
  body: TripDetails(
    tripId: tripId,
    trip: tripValue,
  ),
);
}
}
```

Update the trip TripGridViewItem widget

1. The app will highlight past trips by greying out their card widget. To achieve this, you will use a filter to transform the color. Open the **lib/common/utils/colors.dart** file and update it using the following to define a constant of type **List** for the **colorFilter**.

```
const List<double> greyoutMatrix = [
 0.2126,
 0.7152,
 0.0722,
 0,
 0,
 0.2126,
 0.7152,
 0.0722,
 0,
 0,
 0.2126,
 0.7152,
 0.0722,
 0,
 0,
 0,
 0,
 0,
 1,
 0,
```

```
];
```

The **colors.dart** should look like the following code snippet.

```
import 'package:flutter/material.dart';
const Map<int, Color> primarySwatch = {
 50: Color.fromRGBO(255, 207, 68, .1),
 100: Color.fromRGBO(255, 207, 68, .2),
 200: Color.fromRGBO(255, 207, 68, .3),
 300: Color.fromRGBO(255, 207, 68, .4),
 400: Color.fromRGBO(255, 207, 68, .5),
 500: Color.fromRGBO(255, 207, 68, .6),
 600: Color.fromRGBO(255, 207, 68, .7),
 700: Color.fromRGBO(255, 207, 68, .8),
 800: Color.fromRGBO(255, 207, 68, .9),
 900: Color.fromRGBO(255, 207, 68, 1),
};
const MaterialColor primaryColor = MaterialColor(0xFFFFCF44, primarySwatch);
const int primaryColorDark = 0xFFFD9725;
const List<double> greyoutMatrix = [
 0.2126,
 0.7152,
 0.0722,
 0,
 0,
 0.2126,
 0.7152,
 0.0722,
 0,
 0,
 0.2126,
 0.7152,
 0.0722,
 0,
 0,
 0,
 0,
 0,
 1,
 0,
];
```

2. Open the **lib/features/trip/ui/trips_gridview/trip_gridview_item.dart** file and update it with the following code. The app will use a widget to display the trip card in a **GridView**. If the trip is in the past, it will be greyed out using a color filter and the constant you created above.

> **Note**
>
> VSCode will show an error in the **trips_list_gridview.dart** file about a missing **isPast** parameter for the **TripGridViewItem** widget. You will fix that in the next steps.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/trip/ui/trips_gridview/
trip_gridview_item_card.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';

class TripGridViewItem extends StatelessWidget {
 const TripGridViewItem({
   required this.trip,
   required this.isPast,
   super.key,
 });

 final Trip trip;
 final bool isPast;

 @override
 Widget build(BuildContext context) {
   return InkWell(
     splashColor: Theme.of(context).primaryColor,
     borderRadius: BorderRadius.circular(15),
     onTap: () {
       context.goNamed(
         isPast ? AppRoute.pastTrip.name : AppRoute.trip.name,
         pathParameters: {'id': trip.id},
         extra: trip,
       );
     },
     child: isPast
         ? ColorFiltered(
```

```
              colorFilter: const ColorFilter.matrix(constants.greyoutMatrix),
              child: TripGridViewItemCard(
                trip: trip,
              ),
            )
          : TripGridViewItemCard(
              trip: trip,
            ),
    );
  }
}
```

3. Open the **lib/features/trip/ui/trips_gridview/trips_list_gridview.dart** file and update it with the following code.

```
import 'package:amplify_trips_planner/features/trip/ui/trips_gridview/
trip_gridview_item.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

class TripsListGridView extends StatelessWidget {
 const TripsListGridView({
   required this.tripsList,
   required this.isPast,
   super.key,
 });

 final AsyncValue<List<Trip>> tripsList;
 final bool isPast;

 @override
 Widget build(BuildContext context) {
   switch (tripsList) {
     case AsyncData(:final value):
       return value.isEmpty
           ? const Center(
               child: Text('No Trips'),
             )
           : OrientationBuilder(
               builder: (context, orientation) {
                 return GridView.count(
                   crossAxisCount:
```

```
                    (orientation == Orientation.portrait) ? 2 : 3,
                mainAxisSpacing: 4,
                crossAxisSpacing: 4,
                padding: const EdgeInsets.all(4),
                childAspectRatio:
                    (orientation == Orientation.portrait) ? 0.9 : 1.4,
                children: value.map((tripData) {
                  return TripGridViewItem(
                    trip: tripData,
                    isPast: isPast,
                  );
                }).toList(growable: false),
              );
            },
          );

    case AsyncError():
      return const Center(
        child: Text('Error'),
      );
    case AsyncLoading():
      return const Center(
        child: CircularProgressIndicator(),
      );

    case _:
      return const Center(
        child: Text('Error'),
      );
  }
 }
}
```

4. Open the **lib/features/trip/ui/trips_list/trips_list_page.dart** file and update it to set the **isPast** parameter.

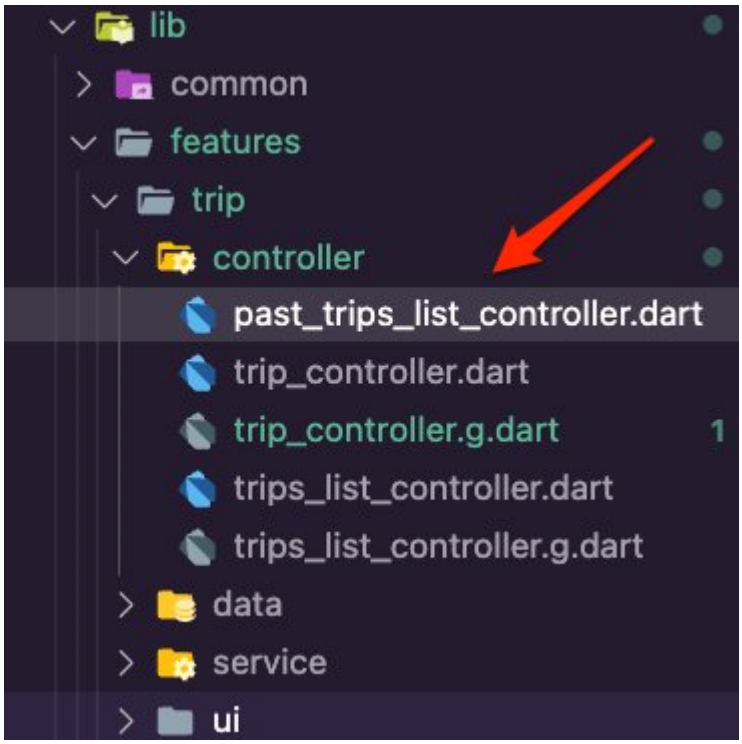
```
import 'package:amplify_trips_planner/common/ui/the_navigation_drawer.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/trip/controller/
trips_list_controller.dart';
import 'package:amplify_trips_planner/features/trip/ui/trips_gridview/
trips_list_gridview.dart';
```

```
import 'package:amplify_trips_planner/features/trip/ui/trips_list/
add_trip_bottomsheet.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

class TripsListPage extends ConsumerWidget {
 const TripsListPage({
   super.key,
 });

 Future<void> showAddTripDialog(BuildContext context) =>
     showModalBottomSheet<void>(
       isScrollControlled: true,
       elevation: 5,
       context: context,
       builder: (sheetContext) {
         return const AddTripBottomSheet();
       },
     );

 @override
 Widget build(BuildContext context, WidgetRef ref) {
   final tripsListValue = ref.watch(tripsListControllerProvider);
   return Scaffold(
     appBar: AppBar(
       centerTitle: true,
       title: const Text(
         'Amplify Trips Planner',
       ),
       backgroundColor: const Color(constants.primaryColorDark),
     ),
     drawer: const TheNavigationDrawer(),
     floatingActionButton: FloatingActionButton(
       onPressed: () {
         showAddTripDialog(context);
       },
       backgroundColor: const Color(constants.primaryColorDark),
       child: const Icon(Icons.add),
     ),
     body: TripsListGridView(
       tripsList: tripsListValue,
       isPast: false,
     ),
   );
```

```
 }
}
```

Add the PastTripsList page to the app

1. Create a new dart file inside the **lib/feature/trip/controller** folder and name it **past_trips_list_controller.dart**.



2. Open the **past_trips_list_controller.dart** file and update it with the following code. The UI will use the controller to get the user's past trips using the **tripsRepository.getPastTrips()** function.

> **Note**
>
> VSCode will show errors due to the missing **past_trips_list_controller.g.dart** file. You will fix that in the next step.

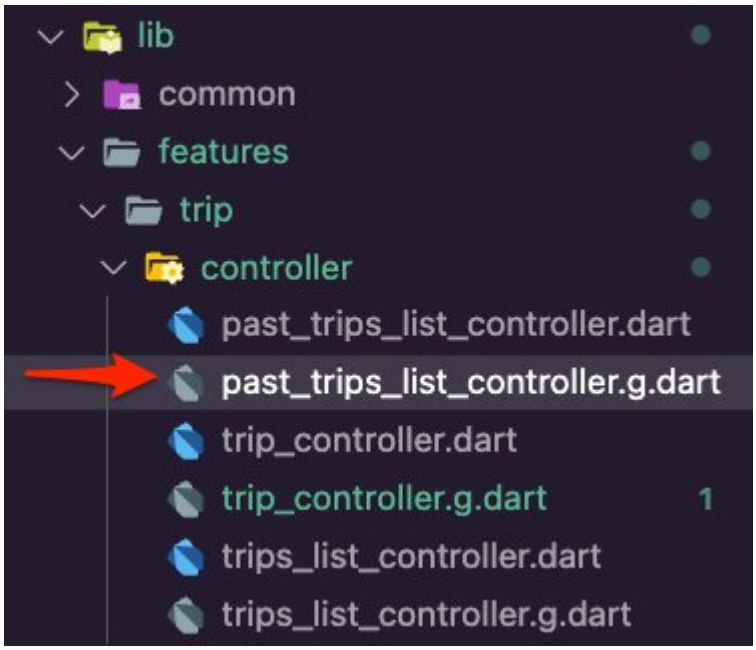
```
import 'dart:async';

import 'package:amplify_trips_planner/features/trip/data/trips_repository.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:riverpod_annotation/riverpod_annotation.dart';
```

```
part 'past_trips_list_controller.g.dart';

@riverpod
class PastTripsListController extends _$PastTripsListController {
Future<List<Trip>> _fetchTrips() async {
final tripsRepository = ref.read(tripsRepositoryProvider);
final trips = await tripsRepository.getPastTrips();
return trips;
}

@override
FutureOr<List<Trip>> build() async {
return _fetchTrips();
}
}
```

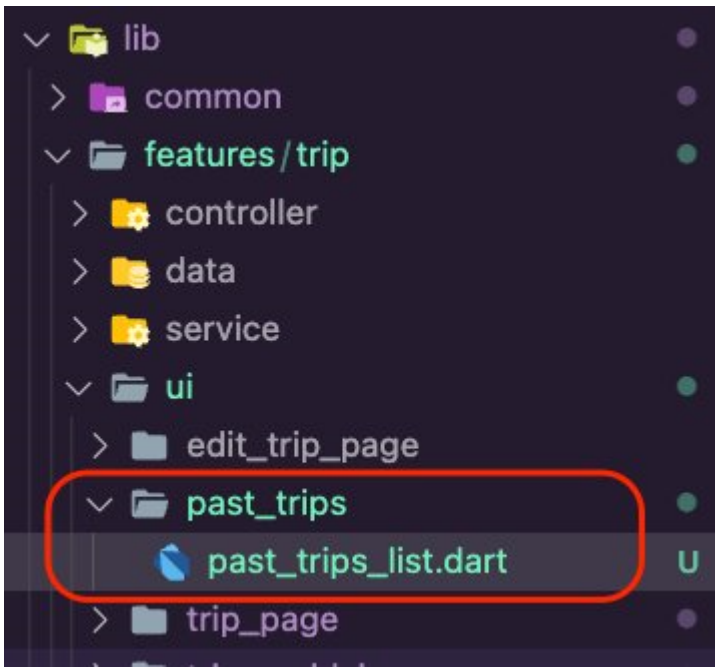
3. Navigate to the app's root folder and run the following command in your terminal.

```
dart run build_runner build -d
```

This will generate the **past_trips_list_controller.g.dart** file in the **lib/feature/trip/controller** folder.



4. Create a new folder in the **lib/features/trip/ui** folder, name it **past_trips**, and then create the file **past_trips_list.dart** inside it.



5. Open the **past_trips_list.dart** file and update it with the following code to create a **GridView** for displaying the past trips.

```
import 'package:amplify_trips_planner/common/ui/the_navigation_drawer.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/trip/controller/
past_trips_list_controller.dart';
import 'package:amplify_trips_planner/features/trip/ui/trips_gridview/
trips_list_gridview.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

class PastTripsList extends ConsumerWidget {
const PastTripsList({
super.key,
});

@override
Widget build(BuildContext context, WidgetRef ref) {
final tripsListValue = ref.watch(pastTripsListControllerProvider);

return Scaffold(
  appBar: AppBar(
    centerTitle: true,
    title: const Text(
      'Amplify Trips Planner',
```

```
    ),
    backgroundColor: const Color(constants.primaryColorDark),
  ),
  drawer: const TheNavigationDrawer(),
  body: TripsListGridView(
    tripsList: tripsListValue,
    isPast: true,
  ),
);
}
}
```

6. Open the **lib/common/navigation/router/router.dart** file and update it to add the **PastTripsList** route.

```
    GoRoute(
      path: '/pasttrips',
      name: AppRoute.pastTrips.name,
      builder: (context, state) => const PastTripsList(),
    ),
```

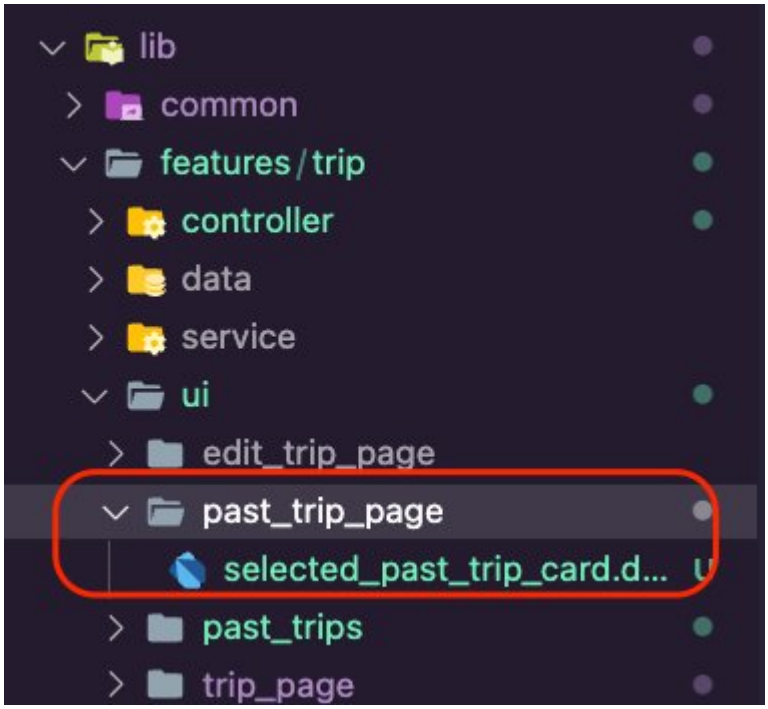
The **router.dart** file should look like the following code snippet.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/features/trip/ui/edit_trip_page/
edit_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trips/
past_trips_list.dart';
import 'package:amplify_trips_planner/features/trip/ui/trip_page/trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/trips_list/
trips_list_page.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';

final router = GoRouter(
  routes: [
    GoRoute(
      path: '/',
      name: AppRoute.home.name,
      builder: (context, state) => const TripsListPage(),
    ),
    GoRoute(
```

```
      path: '/trip/:id',
      name: AppRoute.trip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return TripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/edittrip/:id',
      name: AppRoute.editTrip.name,
      builder: (context, state) {
        return EditTripPage(
          trip: state.extra! as Trip,
        );
      },
    ),
    GoRoute(
      path: '/pasttrips',
      name: AppRoute.pastTrips.name,
      builder: (context, state) => const PastTripsList(),
    ),
  ],
  errorBuilder: (context, state) => Scaffold(
    body: Center(
      child: Text(state.error.toString()),
    ),
  ),
);
```

Add the PastTrips details page to the app

1. Create a new folder inside the **lib/features/trip/ui** folder, name it **past_trip_page**, and then create the file **selected_past_trip_card.dart** inside it.

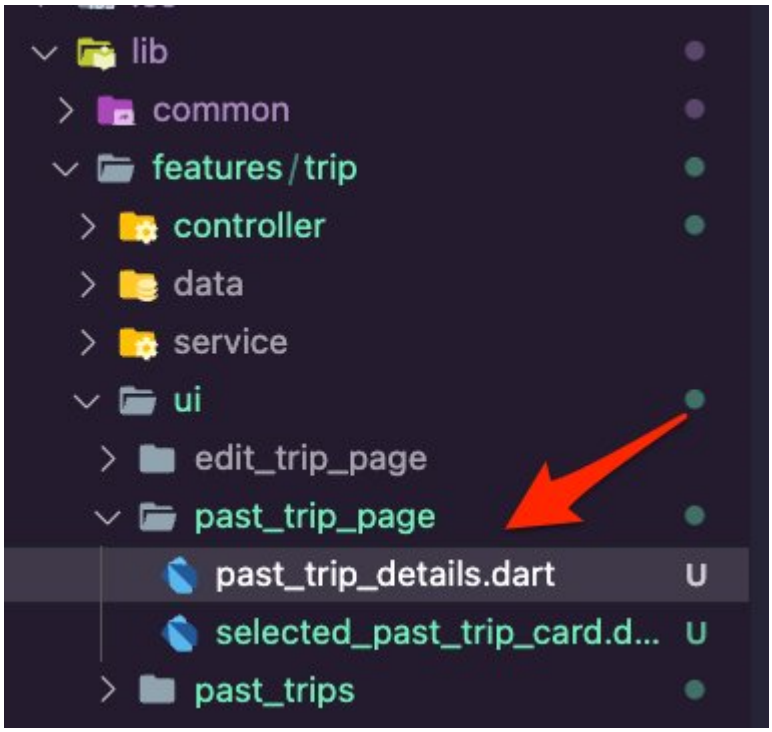


2. Open the **selected_past_trip_card.dart** file and update it with the following code. Here we check if there is an image for the past trip and display it in a card widget. We use the placeholder image from the app assets if there is no image.

```
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:cached_network_image/cached_network_image.dart';
import 'package:flutter/material.dart';

class SelectedPastTripCard extends StatelessWidget {
 const SelectedPastTripCard({
   required this.trip,
   super.key,
 });

 final Trip trip;

 @override
 Widget build(BuildContext context) {
   return Card(
     clipBehavior: Clip.antiAlias,
     shape: RoundedRectangleBorder(
       borderRadius: BorderRadius.circular(15),
     ),
     elevation: 5,
```

```
      child: Column(
        mainAxisSize: MainAxisSize.min,
        children: [
          Text(
            trip.tripName,
            textAlign: TextAlign.center,
            style: const TextStyle(
              fontSize: 20,
              fontWeight: FontWeight.bold,
            ),
          ),
          const SizedBox(
            height: 8,
          ),
          Container(
            alignment: Alignment.center,
            color: const Color(constants.primaryColorDark),
            height: 150,
            child: trip.tripImageUrl != null
                ? Stack(
                    children: [
                      const Center(child: CircularProgressIndicator()),
                      CachedNetworkImage(
                        imageUrl: trip.tripImageUrl!,
                        cacheKey: trip.tripImageKey,
                        width: double.maxFinite,
                        height: 500,
                        alignment: Alignment.topCenter,
                        fit: BoxFit.fill,
                      ),
                    ],
                  )
                : Image.asset(
                    'images/amplify.png',
                    fit: BoxFit.contain,
                  ),
          ),
        ],
      ),
    );
  }
}
```

3. Create a new dart file inside the **lib/features/trip/ui/past_trip_page** folder and call it **past_trip_details.dart**.



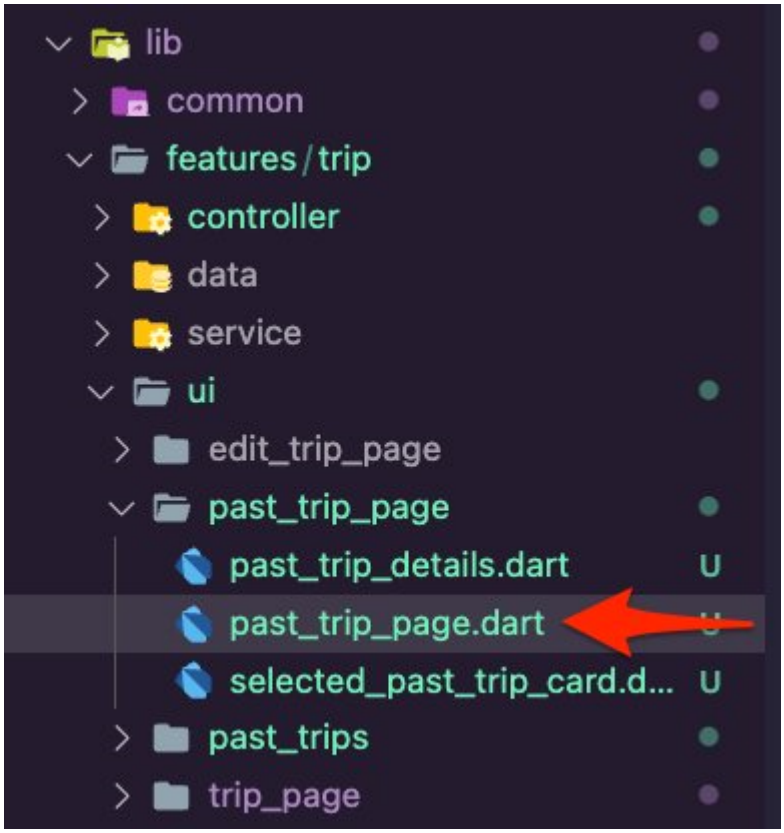
4. Open the **past_trip_details.dart** file and update it with the following code to create the **PastTripDetails**, which will use the **SelectedPastTripCard** you created above to display the data.

```
import 'package:amplify_trips_planner/features/trip/ui/past_trip_page/
selected_past_trip_card.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

class PastTripDetails extends ConsumerWidget {
 const PastTripDetails({
   required this.trip,
   super.key,
 });

 final AsyncValue<Trip> trip;

 @override
 Widget build(BuildContext context, WidgetRef ref) {
   switch (trip) {
     case AsyncData(:final value):
```

```
        return Column(
          crossAxisAlignment: CrossAxisAlignment.stretch,
          children: [
            const SizedBox(
              height: 8,
            ),
            SelectedPastTripCard(trip: value),
            const SizedBox(
              height: 20,
            ),
            const Divider(
              height: 20,
              thickness: 5,
              indent: 20,
              endIndent: 20,
            ),
            const Text(
              'Your Activities',
              textAlign: TextAlign.center,
              style: TextStyle(
                fontSize: 20,
                fontWeight: FontWeight.bold,
              ),
            ),
            const SizedBox(
              height: 8,
            ),
          ],
        );

      case AsyncError():
        return const Center(
          child: Text('Error'),
        );
      case AsyncLoading():
        return const Center(
          child: CircularProgressIndicator(),
        );

      case _:
        return const Center(
          child: Text('Error'),
        );
    }
```

```
 }
}
```

5. Create a new dart file inside the **lib/features/trip/ui/past_trip_page** folder and name it **past_trip_page.dart**.



6. Open the **past_trip_page.dart** file and update it with the following code to create the **PastTripPage**, which will get the past trip details using the **tripId**. The **PastTripPage** will grey out and use the **PastTripDetails** you created above to display the data.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/ui/the_navigation_drawer.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/trip/controller/trip_controller.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trip_page/
past_trip_details.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class PastTripPage extends ConsumerWidget {
  const PastTripPage({
```

```
    required this.tripId,
    super.key,
  });
  final String tripId;

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    final tripValue = ref.watch(tripControllerProvider(tripId));
    return Scaffold(
      appBar: AppBar(
        centerTitle: true,
        title: const Text(
          'Amplify Trips Planner',
        ),
        actions: [
          IconButton(
            onPressed: () {
              context.goNamed(
                AppRoute.home.name,
              );
            },
            icon: const Icon(Icons.home),
          ),
        ],
        backgroundColor: const Color(constants.primaryColorDark),
      ),
      drawer: const TheNavigationDrawer(),
      body: ColorFiltered(
        colorFilter: const ColorFilter.matrix(constants.greyoutMatrix),
        child: PastTripDetails(
          trip: tripValue,
        ),
      ),
    );
  }
}
```

7. Open the **/lib/common/navigation/router/router.dart** file and update it to add the **PastTripPage** route.

```
GoRoute(
  path: '/pasttrip/:id',
  name: AppRoute.pastTrip.name,
```

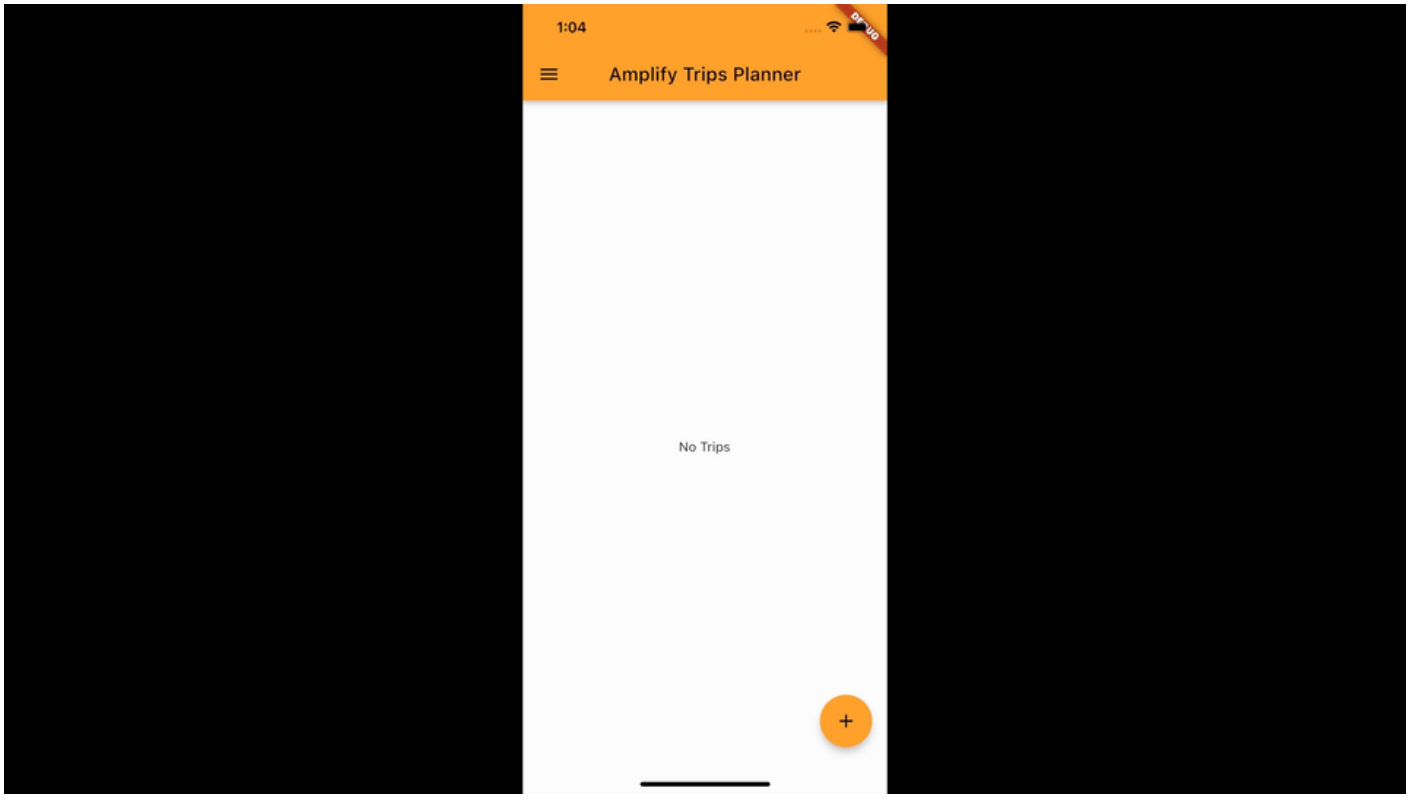
```
  builder: (context, state) {
    final tripId = state.pathParameters['id']!;
    return PastTripPage(tripId: tripId);
  },
),
```

The **router.dart** should look like the following code snippet.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/features/trip/ui/edit_trip_page/
edit_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trip_page/
past_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trips/
past_trips_list.dart';
import 'package:amplify_trips_planner/features/trip/ui/trip_page/trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/trips_list/
trips_list_page.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';

final router = GoRouter(
  routes: [
    GoRoute(
      path: '/',
      name: AppRoute.home.name,
      builder: (context, state) => const TripsListPage(),
    ),
    GoRoute(
      path: '/trip/:id',
      name: AppRoute.trip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return TripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/edittrip/:id',
      name: AppRoute.editTrip.name,
      builder: (context, state) {
        return EditTripPage(
          trip: state.extra! as Trip,
```

```
        );
      },
    ),
    GoRoute(
      path: '/pasttrip/:id',
      name: AppRoute.pastTrip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return PastTripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/pasttrips',
      name: AppRoute.pastTrips.name,
      builder: (context, state) => const PastTripsList(),
    ),
  ],
  errorBuilder: (context, state) => Scaffold(
    body: Center(
      child: Text(state.error.toString()),
    ),
  ),
);
```

8. Run the app in an emulator or simulator and create a trip. Set its start and end date in the past. The following is an example using an iPhone simulator.



Conclusion

In this module, you implemented the UI to display the user's past trips. In the next module, we will add the trip's Activity feature to your app.

Module 3: Add the Activity feature

Overview

In this module, you will update the Amplify API to retrieve and persist your trip's activities data. The API is a GraphQL API that uses AWS AppSync (a managed GraphQL service) backed by DynamoDB (a NoSQL database).

What you will accomplish

In this module, you will:

- Add the activity data model to the app
- Implement the create, read, update, and delete (CRUD) operations and flow for the activity feature
- Implement the activities listing UI
- Add the Activity Details page to the app

Implementation

Minimum time to complete	45 minutes

Add activity data model to the app

1. Open the **amplify/backend/api/amplifytripsplanner/schema.graphql** file and update it as follows:
   - Create a data model for the activity
   - Introduce an enum for the activity's categories
   - Update the Trip model to set up a 1:n relation with the Activity

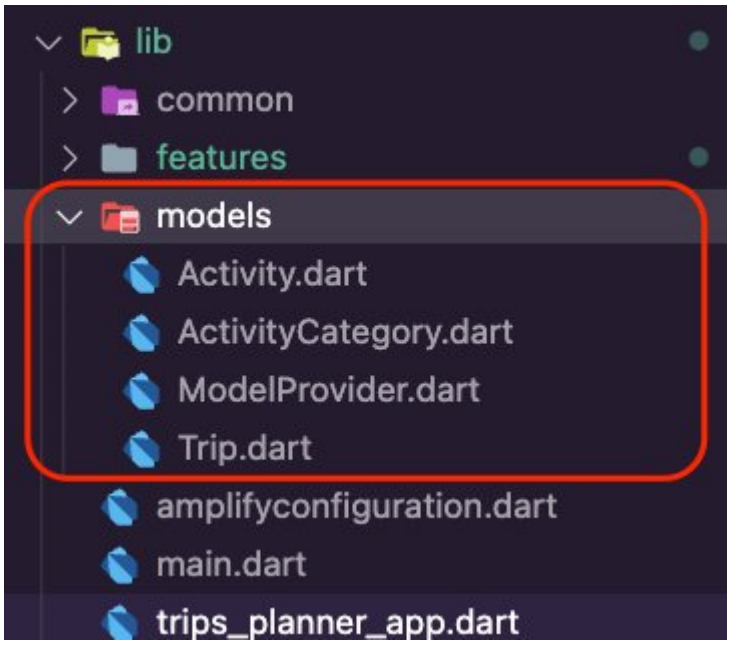
```
type Trip @model @auth(rules: [{ allow: owner }]) {
  id: ID!
  tripName: String!
```

```
  destination: String!
  startDate: AWSDate!
  endDate: AWSDate!
  tripImageUrl: String
  tripImageKey: String
  Activities: [Activity] @hasMany(indexName: "byTrip", fields: ["id"])
}

type Activity @model @auth(rules: [{allow: owner}]) {
  id: ID!
  activityName: String!
  tripID: ID! @index(name: "byTrip", sortKeyFields: ["activityName"])
  trip: Trip! @belongsTo(fields: ["tripID"])
  activityImageUrl: String
  activityImageKey: String
  activityDate: AWSDate!
  activityTime: AWSTime
  category: ActivityCategory!
}

enum ActivityCategory { Flight, Lodging, Meeting, Restaurant }
```

2. Run the following command in the root folder of the app to generate the models files.

```
amplify codegen models
```

The Amplify CLI will generate the dart files in the **lib/models** folder.



3. Run the command **amplify push** to create the resources in the cloud.

```
    Current Environment: dev

| Category | Resource name                | Operation | Provider plugin     |
| Api      | amplifytripsplanner          | Update    | awscloudformation   |
| Auth     | amplifytripsplanner1ec293ff  | No Change | awscloudformation   |
| Storage  | s3b101c2c7                   | No Change | awscloudformation   |

? Are you sure you want to continue? (Y/n) ›
```

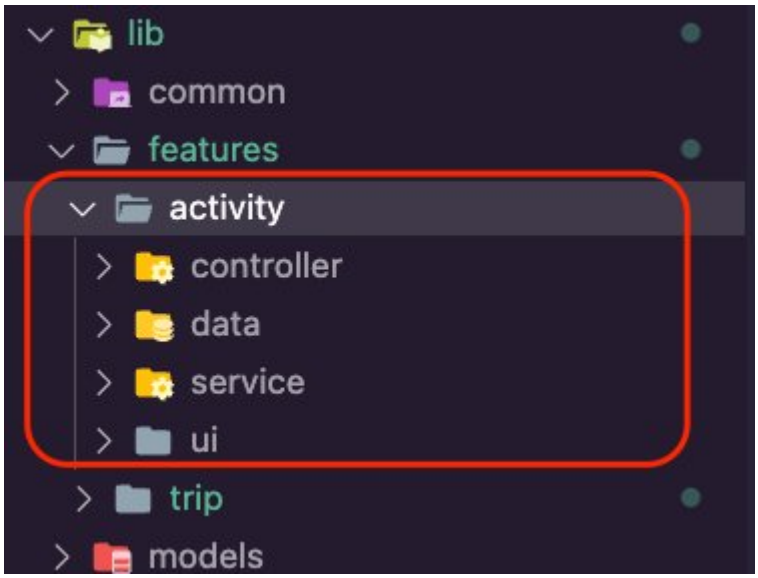
4. Press **Enter**. The Amplify CLI will deploy the resources and display a confirmation, as shown in the screenshot.

```
        apiamplifytripsplanner          AWS::CloudFormation::Stack     UPDATE_COMPL
Deployed api amplifytripsplanner [ ======================================
        GraphQLAPITransformerSchema3C… AWS::AppSync::GraphQLSchema    UPDATE_COMPL
        Trip                           AWS::CloudFormation::Stack     UPDATE_COMPL
        Activity                       AWS::CloudFormation::Stack     CREATE_COMPL
        ConnectionStack                AWS::CloudFormation::Stack     CREATE_COMPL
        CustomResourcesjson            AWS::CloudFormation::Stack     UPDATE_COMPL

Deployment state saved successfully.

GraphQL endpoint: https://qmrokxalkng4xdaegffe4g2nsq.appsync-api.us-west-1.amazonaws.
com/graphql

GraphQL transformer version: 2
```

Create the folders for the Activity

1. Create a new folder inside **lib/features** and name it **activity**.

2. Create the following new folders inside the **activity** folder:
   - **service:** The layer to connect with the Amplify backend.
   - **data:** This will be the repository layer that abstracts away the networking code, specifically **service** .
   - **controller:** This is the domain layer to connect the UI with the repository.
   - **ui:** Here, we will create the widgets and the pages that the app will present to the user.

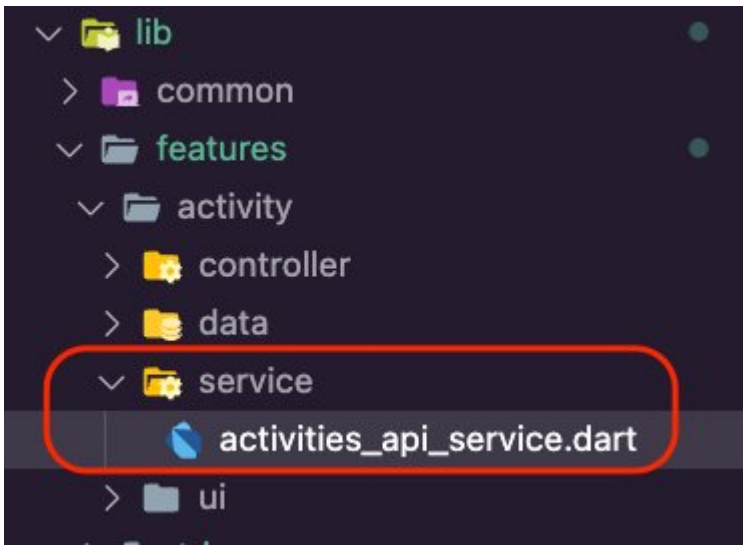


3. Open the **lib/common/navigation/router/routes.dart** file. Update it to add the enum values for the activity feature. The **routes.dart** file should look like this:

```
enum AppRoute {
 home,
 trip,
 editTrip,
 pastTrips,
 pastTrip,
 activity,
 addActivity,
 editActivity,
}
```

Implement the CRUD operations and flow for the Activity feature

1. Create a new dart file inside the **lib/features/activity/service** folder and call it **activities_api_service.dart**.



2. Open the **activities_api_service.dart** file and update it with the following code snippet to create the **ActivitiesAPIService**, which contains the following functions:

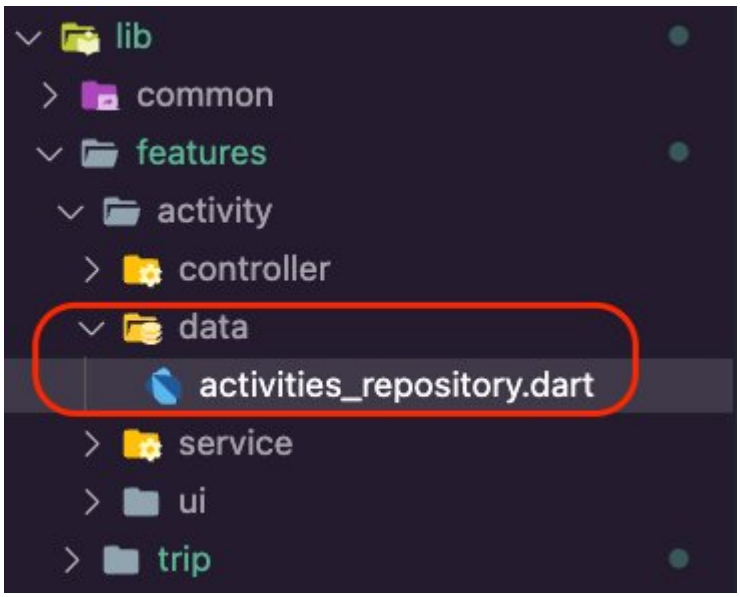
   - **getActivitiesForTrip:** Queries the Amplify API for the activities of a specific trip and returns a list of the activities.
   - **getActivity:** Queries the Amplify API for a specific activity and returns its details.
   - **getActivity, deleteActivity,** and **updateActivity:** Add, delete, or update the activities in the Amplify API.

```
import 'dart:async';

import 'package:amplify_api/amplify_api.dart';
import 'package:amplify_flutter/amplify_flutter.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

final activitiesAPIServiceProvider = Provider<ActivitiesAPIService>((ref) {
  final service = ActivitiesAPIService();
  return service;
});

class ActivitiesAPIService {
  ActivitiesAPIService();

  Future<List<Activity>> getActivitiesForTrip(String tripId) async {
    try {
      final request = ModelQueries.list(
        Activity.classType,
```

```
      where: Activity.TRIP.eq(tripId),
    );

    final response = await Amplify.API.query(request: request).response;

    final activites = response.data?.items;
    if (activites == null) {
      safePrint('errors: ${response.errors}');
      return const [];
    }
    activites.sort(
      (a, b) => a!.activityDate
          .getDateTime()
          .compareTo(b!.activityDate.getDateTime()),
    );
    return activites.map((e) => e as Activity).toList();
  } on Exception catch (error) {
    safePrint('getActivitiesForTrip failed: $error');
    return const [];
  }
}

Future<void> addActivity(Activity activity) async {
  try {
    final request = ModelMutations.create(activity);
    final response = await Amplify.API.mutate(request: request).response;

    final createdActivity = response.data;
    if (createdActivity == null) {
      safePrint('errors: ${response.errors}');
      return;
    }
  } on Exception catch (error) {
    safePrint('addActivity failed: $error');
  }
}

Future<void> deleteActivity(Activity activity) async {
  try {
    await Amplify.API
        .mutate(
          request: ModelMutations.delete(activity),
        )
        .response;
```

```
    } on Exception catch (error) {
      safePrint('deleteActivity failed: $error');
    }
  }

  Future<void> updateActivity(Activity updatedActivity) async {
    try {
      await Amplify.API
          .mutate(
            request: ModelMutations.update(updatedActivity),
          )
          .response;
    } on Exception catch (error) {
      safePrint('updateActivity failed: $error');
    }
  }

  Future<Activity> getActivity(String activityId) async {
    try {
      final request = ModelQueries.get(
        Activity.classType,
        ActivityModelIdentifier(id: activityId),
      );
      final response = await Amplify.API.query(request: request).response;

      final activity = response.data!;
      return activity;
    } on Exception catch (error) {
      safePrint('getActivity failed: $error');
      rethrow;
    }
  }
}
```

3. Create a new dart file in the **lib/features/activity/data** folder and name it **activities_repository.dart**.



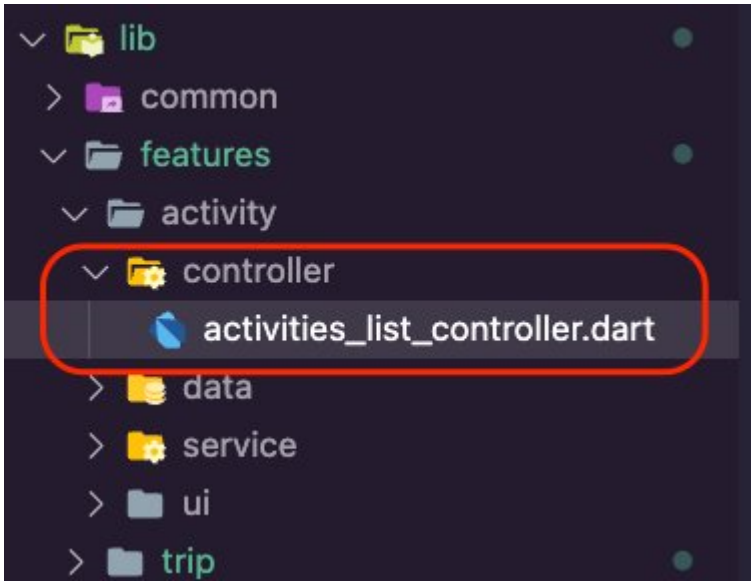
4. Open the **activities_repository.dart** file and update it with the following code:

```
import 'package:amplify_trips_planner/features/activity/service/
activities_api_service.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

final activitiesRepositoryProvider = Provider<ActivitiesRepository>((ref) {
 final activitiesAPIService = ref.read(activitiesAPIServiceProvider);
 return ActivitiesRepository(activitiesAPIService);
});

class ActivitiesRepository {
 ActivitiesRepository(
   this.activitiesAPIService,
 );

 final ActivitiesAPIService activitiesAPIService;

 Future<List<Activity>> getActivitiesForTrip(String tripId) {
   return activitiesAPIService.getActivitiesForTrip(tripId);
 }

 Future<Activity> getActivity(String activityId) {
   return activitiesAPIService.getActivity(activityId);
 }

 Future<void> add(Activity activity) async {
```

```
    return activitiesAPIService.addActivity(activity);
  }

  Future<void> delete(Activity activity) async {
    return activitiesAPIService.deleteActivity(activity);
  }

  Future<void> update(Activity activity) async {
    return activitiesAPIService.updateActivity(activity);
  }
}
```

Implement the activities listing UI

1. Create a new dart file inside the **lib/features/activity/controller** folder and name it **activities_list_controller.dart.**



2. Open the **activities_list_controller.dart** file and update it with the following code. The UI will use the controller to get the activities for a trip, add a new activity, and delete an activity.

> **Note**
>
> VSCode will show errors due to the missing **activities_list_controller.g.dart** file. You will fix that in the next step.

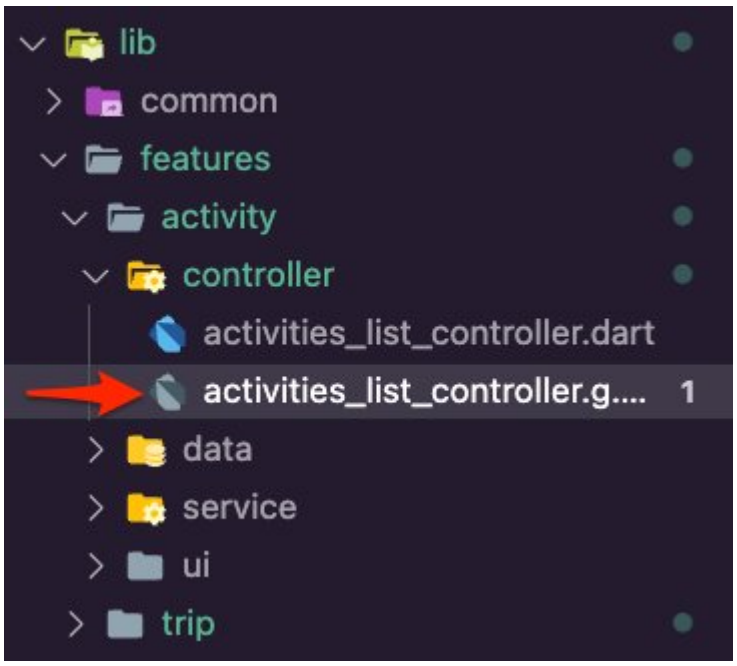
```
import 'dart:async';

import 'package:amplify_flutter/amplify_flutter.dart';
import 'package:amplify_trips_planner/common/utils/date_time_formatter.dart';
import 'package:amplify_trips_planner/features/activity/data/
activities_repository.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:riverpod_annotation/riverpod_annotation.dart';

part 'activities_list_controller.g.dart';

@riverpod
class ActivitiesListController extends _$ActivitiesListController {
 Future<List<Activity>> _fetchActivities(String tripId) async {
   final activitiesRepository = ref.read(activitiesRepositoryProvider);
   final activities = await activitiesRepository.getActivitiesForTrip(tripId);
   return activities;
 }

 @override
 FutureOr<List<Activity>> build(String tripId) async {
   return _fetchActivities(tripId);
 }

 Future<void> removeActivity(Activity activity) async {
   state = const AsyncValue.loading();
   state = await AsyncValue.guard(() async {
     final activitiesRepository = ref.read(activitiesRepositoryProvider);
     await activitiesRepository.delete(activity);

     return _fetchActivities(activity.trip.id);
   });
 }

 Future<void> add({
   required String name,
   required String activityDate,
   required TimeOfDay activityTime,
   required ActivityCategory category,
   required Trip trip,
 }) async {
```

```
    final now = DateTime.now();
    final time = DateTime(
      now.year,
      now.month,
      now.day,
      activityTime.hour,
      activityTime.minute,
    );

    final activity = Activity(
      activityName: name,
      activityDate: TemporalDate(DateTime.parse(activityDate)),
      activityTime: TemporalTime.fromString(time.format('HH:mm:ss.sss')),
      trip: trip,
      category: category,
    );
    state = const AsyncValue.loading();
    state = await AsyncValue.guard(() async {
      final activitiesRepository = ref.read(activitiesRepositoryProvider);

      await activitiesRepository.add(activity);

      return _fetchActivities(trip.id);
    });
  }
}
```

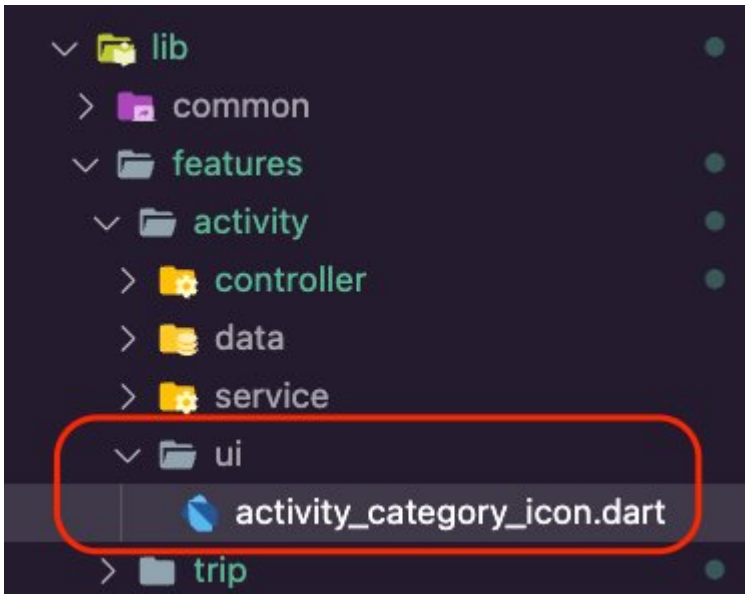
3. Navigate to the app's root folder and run the following command in your terminal.

```
dart run build_runner build -d
```

This will generate the **activities_list.g.dart** file inside the **lib/feature/activity/controller** folder.



4. Create a new dart file in the **lib/features/activity/ui** folder and name it **activity_category_icon.dart**.



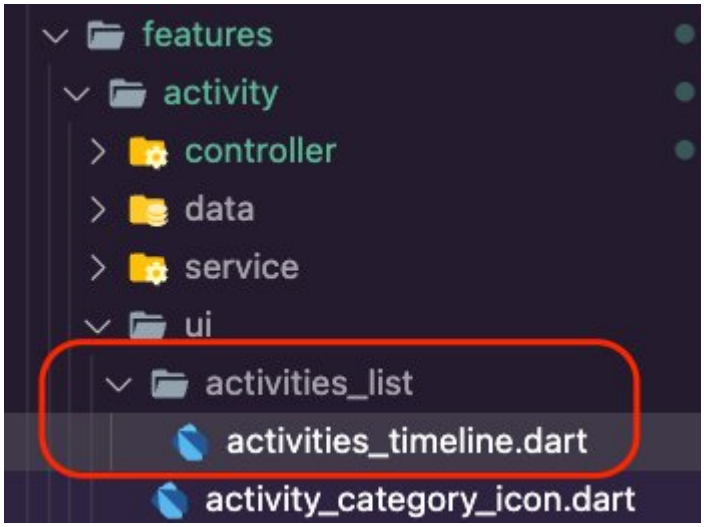
5. Open the **activity_category_icon.dart** file and update it with the following code. This will allow the app to display an icon representing the activity's category.

```
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';

class ActivityCategoryIcon extends StatelessWidget {
```

```
const ActivityCategoryIcon({
  required this.activityCategory,
  super.key,
});
final ActivityCategory activityCategory;

@override
Widget build(BuildContext context) {
  switch (activityCategory) {
    case ActivityCategory.Flight:
      return const Icon(
        Icons.flight,
        size: 50,
        color: Color(constants.primaryColorDark),
      );

    case ActivityCategory.Lodging:
      return const Icon(
        Icons.hotel,
        size: 50,
        color: Color(constants.primaryColorDark),
      );
    case ActivityCategory.Meeting:
      return const Icon(
        Icons.computer,
        size: 50,
        color: Color(constants.primaryColorDark),
      );
    case ActivityCategory.Restaurant:
      return const Icon(
        Icons.restaurant,
        size: 50,
        color: Color(constants.primaryColorDark),
      );
    default:
      ActivityCategory.Flight;
  }
  return const Icon(
    Icons.flight,
    size: 50,
    color: Color(constants.primaryColorDark),
  );
}
```

```
}
```

6. Create a new folder inside the **lib/features/activity/ui folder**, name it **activities_list**, and then create the file **activities_timeline.dart** inside it.



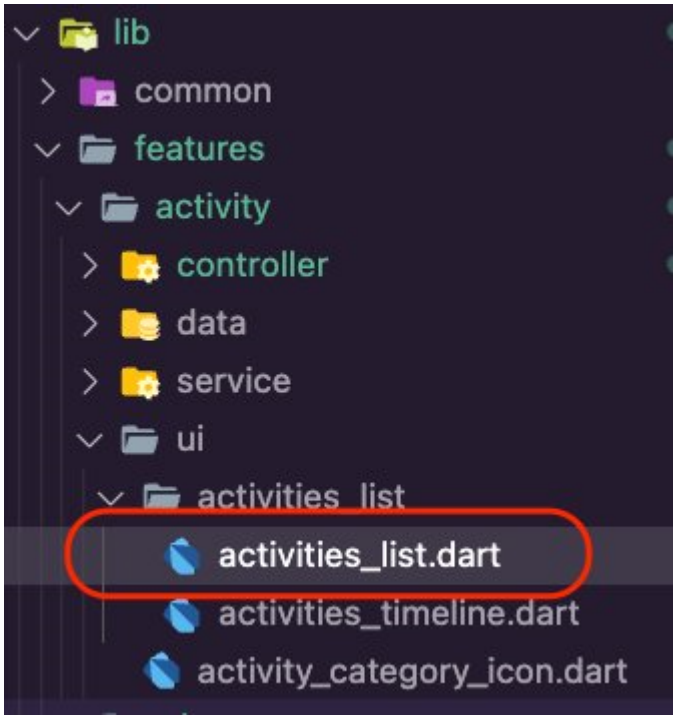
7. Open the **activities_timeline.dart** file and update it with the following code. This will display a timeline of the trip's activities.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/utils/date_time_formatter.dart';
import 'package:amplify_trips_planner/features/activity/ui/
activity_category_icon.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';
import 'package:timelines/timelines.dart';

class ActivitiesTimeline extends StatelessWidget {
  const ActivitiesTimeline({
    super.key,
    required this.activities,
  });

  final List<Activity> activities;

  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        Flexible(
```

```
          child: Timeline.tileBuilder(
            builder: TimelineTileBuilder.fromStyle(
              oppositeContentsBuilder: (context, index) {
                return InkWell(
                  onTap: () => context.goNamed(
                    AppRoute.activity.name,
                    pathParameters: {'id': activities[index].id},
                  ),
                  child: Padding(
                    padding: const EdgeInsets.only(bottom: 15),
                    child: ActivityCategoryIcon(
                      activityCategory: activities[index].category,
                    ),
                  ),
                );
              },
              contentsAlign: ContentsAlign.alternating,
              contentsBuilder: (context, index) => InkWell(
                onTap: () => context.goNamed(
                  AppRoute.activity.name,
                  pathParameters: {'id': activities[index].id},
                ),
                child: Padding(
                  padding: const EdgeInsets.all(24),
                  child: Column(
                    children: [
                      Text(
                        activities[index].activityName,
                        style: Theme.of(context).textTheme.titleMedium,
                        textAlign: TextAlign.center,
                      ),
                      const SizedBox(
                        height: 5,
                      ),
                      Text(
                        activities[index]
                            .activityDate
                            .getDateTime()
                            .format('yyyy-MM-dd'),
                        style: Theme.of(context).textTheme.bodySmall,
                      ),
                      Text(
                        activities[index]
                            .activityTime!
```

```
                            .getDateTime()
                            .format('hh:mm a'),
                        style: Theme.of(context).textTheme.bodySmall,
                      ),
                    ],
                  ),
                ),
              ),
              itemCount: activities.length,
            ),
          ),
        ),
      ],
    );
  }
}
```

8. Create the file **activities_list.dart** inside the **lib/features/activity/ui/activities_list** folder.



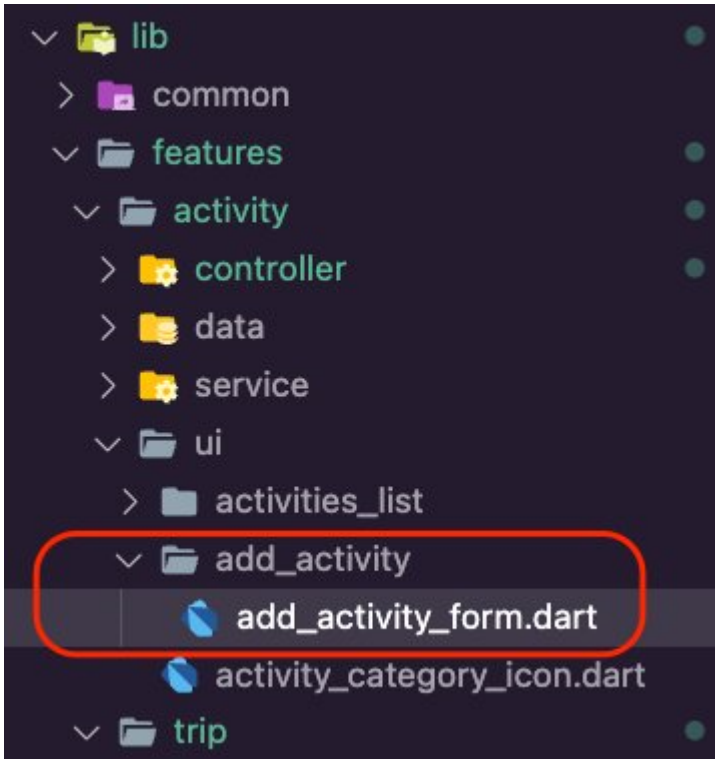
9. Open the **activities_list.dart** file and update it with the following code to use the **ActivitiesTimeline** widget you created previously to display a timeline of the trip's activities.

```
import 'package:amplify_trips_planner/features/activity/controller/
activities_list_controller.dart';
import 'package:amplify_trips_planner/features/activity/ui/activities_list/
activities_timeline.dart';
```

```
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

class ActivitiesList extends ConsumerWidget {
  const ActivitiesList({
    required this.trip,
    super.key,
  });
  final Trip trip;

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    final activitiesListValue = ref.watch(activitiesListControllerProvider(trip.id));
    switch (activitiesListValue) {
      case AsyncData(:final value):
        return value.isEmpty
            ? const Center(
                child: Text('No Activities'),
              )
            : ActivitiesTimeline(activities: value);

      case AsyncError():
        return const Center(
          child: Text('Error'),
        );
      case AsyncLoading():
        return const Center(
          child: CircularProgressIndicator(),
        );

      case _:
        return const Center(
          child: Text('Error'),
        );
    }
  }
}
```

Implement the UI for adding activities

1. Create a new folder inside the **lib/features/activity/ui** folder, name it **add_activity**, and then create the file **add_activity_form.dart** inside it.



2. Open the **add_activity_form.dart** file and update it with the following code. This will allow us to present a form to the user to submit the required details to create a new activity for the selected trip.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/ui/bottomsheet_text_form_field.dart';
import 'package:amplify_trips_planner/common/utils/date_time_formatter.dart';
import 'package:amplify_trips_planner/features/activity/controller/
activities_list_controller.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class AddActivityForm extends ConsumerStatefulWidget {
const AddActivityForm({
required this.trip,
super.key,
});
```

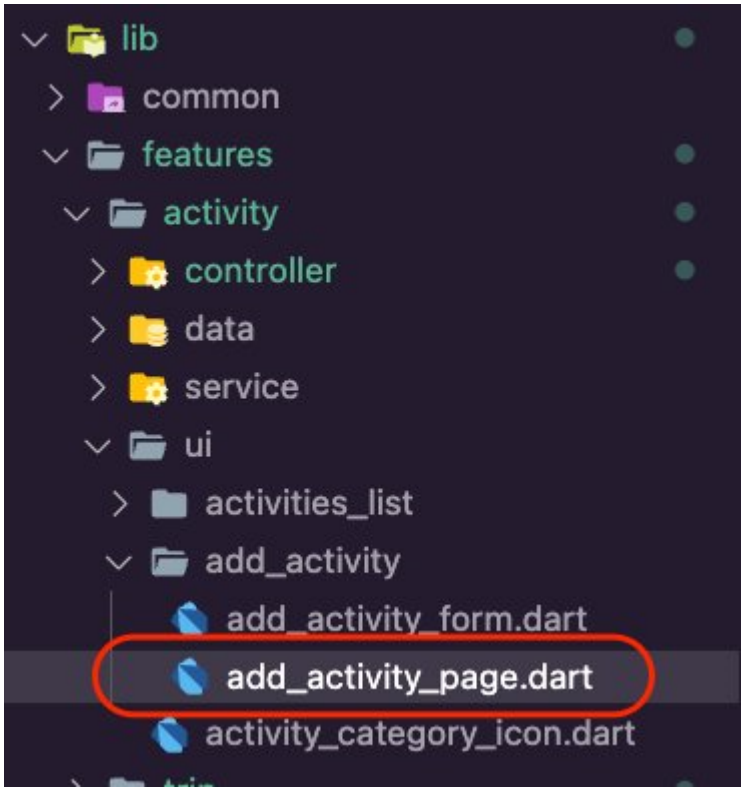
```
final AsyncValue<Trip> trip;

@override
AddActivityFormState createState() => AddActivityFormState();
}

class AddActivityFormState extends ConsumerState<AddActivityForm> {
final formGlobalKey = GlobalKey<FormState>();

final activityNameController = TextEditingController();
final activityDateController = TextEditingController();
final activityTimeController = TextEditingController();
var activityCategory = ActivityCategory.Flight;
var activityTime = TimeOfDay.now();

@override
Widget build(BuildContext context) {
final activityNameController = TextEditingController();
final activityDateController = TextEditingController();
final activityTimeController = TextEditingController();
var activityCategory = ActivityCategory.Flight;
var activityTime = TimeOfDay.now();
switch (widget.trip) {
  case AsyncData(:final value):
    return SingleChildScrollView(
      child: Form(
        key: formGlobalKey,
        child: Container(
          padding: EdgeInsets.only(
            top: 15,
            left: 15,
            right: 15,
            bottom: MediaQuery.of(context).viewInsets.bottom + 15,
          ),
          width: double.infinity,
          child: Column(
            mainAxisSize: MainAxisSize.min,
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              BottomSheetTextFormField(
                labelText: 'Activity Name',
                controller: activityNameController,
                keyboardType: TextInputType.name,
```

```
                ),
                const SizedBox(
                  height: 20,
                ),
                DropdownButtonFormField<ActivityCategory>(
                  onChanged: (value) {
                    activityCategory = value!;
                  },
                  value: activityCategory,
                  decoration: const InputDecoration(
                    labelText: 'Category',
                  ),
                  items: [
                    for (var category in ActivityCategory.values)
                      DropdownMenuItem(
                        value: category,
                        child: Text(category.name),
                      ),
                  ],
                ),
                const SizedBox(
                  height: 20,
                ),
                BottomSheetTextFormField(
                  labelText: 'Activity Date',
                  controller: activityDateController,
                  keyboardType: TextInputType.datetime,
                  onTap: () async {
                    final pickedDate = await showDatePicker(
                      context: context,
                      initialDate: DateTime.parse(value.startDate.toString()),
                      firstDate: DateTime.parse(value.startDate.toString()),
                      lastDate: DateTime.parse(value.endDate.toString()),
                    );

                    if (pickedDate != null) {
                      activityDateController.text =
                          pickedDate.format('yyyy-MM-dd');
                    } else {}
                  },
                ),
                const SizedBox(
                  height: 20,
                ),
```

```
                BottomSheetTextFormField(
                  labelText: 'Activity Time',
                  controller: activityTimeController,
                  keyboardType: TextInputType.datetime,
                  onTap: () async {
                    await showTimePicker(
                      context: context,
                      initialTime: activityTime,
                      initialEntryMode: TimePickerEntryMode.dial,
                    ).then((timeOfDay) {
                      if (timeOfDay != null) {
                        final localizations =
                            MaterialLocalizations.of(context);
                        final formattedTimeOfDay =
                            localizations.formatTimeOfDay(timeOfDay);

                        activityTimeController.text = formattedTimeOfDay;
                        activityTime = timeOfDay;
                      }
                    });
                  },
                ),
                const SizedBox(
                  height: 20,
                ),
                TextButton(
                  child: const Text('OK'),
                  onPressed: () async {
                    final currentState = formGlobalKey.currentState;
                    if (currentState == null) {
                      return;
                    }
                    if (currentState.validate()) {
                      await ref
                          .watch(activitiesListControllerProvider(value.id)
                              .notifier)
                          .add(
                            name: activityNameController.text,
                            activityDate: activityDateController.text,
                            activityTime: activityTime,
                            category: activityCategory,
                            trip: value,
                          );
                      if (context.mounted) {
```

```
                        context.goNamed(
                          AppRoute.trip.name,
                          pathParameters: {'id': value.id},
                        );
                      }
                    }
                  }, //,
                ),
              ],
            ),
          ),
        ),
      );

  case AsyncError():
    return const Center(
      child: Text('Error'),
    );
  case AsyncLoading():
    return const Center(
      child: CircularProgressIndicator(),
    );

  case _:
    return const Center(
      child: Text('Error'),
      );
    }
  }
}
```

3. Create a new file inside the **lib/features/activity/ui/add_activity** folder and name it **add_activity_page.dart**.

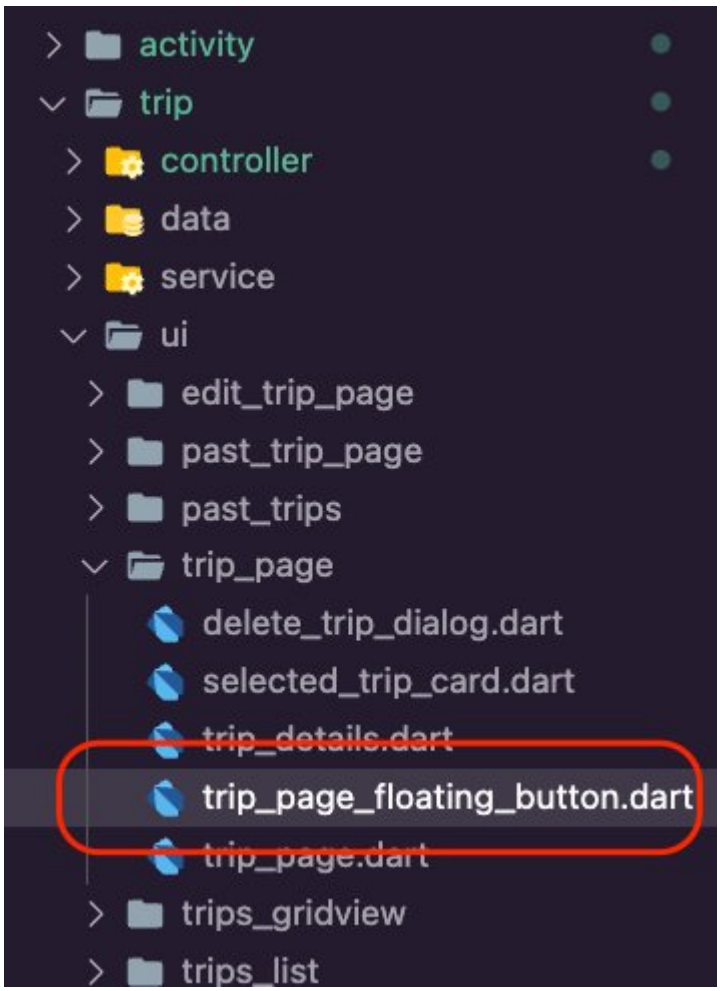


4. Open the **add_activity_page.dart** file and update it with the following code to use the **AddActivityForm** you created above to create a new activity for the selected trip.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/activity/ui/add_activity/
add_activity_form.dart';
import 'package:amplify_trips_planner/features/trip/controller/trip_controller.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class AddActivityPage extends ConsumerWidget {
  AddActivityPage({
    required this.tripId,
    super.key,
  });

  final String tripId;

  final formGlobalKey = GlobalKey<FormState>();

  @override
```

```
  Widget build(BuildContext context, WidgetRef ref) {
    final tripValue = ref.watch(tripControllerProvider(tripId));

    return Scaffold(
      appBar: AppBar(
        centerTitle: true,
        title: const Text(
          'Amplify Trips Planner',
        ),
        leading: IconButton(
          onPressed: () {
            context.goNamed(
              AppRoute.trip.name,
              pathParameters: {'id': tripId},
            );
          },
          icon: const Icon(Icons.arrow_back),
        ),
        backgroundColor: const Color(constants.primaryColorDark),
      ),
      body: AddActivityForm(
        trip: tripValue,
      ),
    );
  }
}
```

5. Create a new file in the **lib/features/trip/ui/trip_page** folder and name it **trip_page_floating_button.dart**.



6. Open the **lib/features/trip/ui/trip_page/trip_page_floating_button.dart** file and update it to a **floatingActionButton** to open the **AddActivityForm**.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class TripPageFloatingButton extends StatelessWidget {
const TripPageFloatingButton({
required this.trip,
super.key,
});

final AsyncValue<Trip> trip;
```

```
@override
Widget build(BuildContext context) {
switch (trip) {
  case AsyncData(:final value):
    return FloatingActionButton(
      onPressed: () {
        context.goNamed(
          AppRoute.addActivity.name,
          pathParameters: {'id': value.id},
        );
      },
      backgroundColor: const Color(constants.primaryColorDark),
      child: const Icon(Icons.add),
    );

  case AsyncError():
    return const Placeholder();
  case AsyncLoading():
    return const SizedBox();

  case _:
    return const SizedBox();
    }
  }
}
```

7. Open the **lib/features/trip/ui/trip_page/trip_page.dart** file and update it to use the **TripPageFloatingButton** to add an activity to the trip.

```
floatingActionButton: TripPageFloatingButton(
  trip: tripValue,
),
```

The **trip_page.dart** should look like the following code snippet.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/ui/the_navigation_drawer.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/trip/controller/trip_controller.dart';
import 'package:amplify_trips_planner/features/trip/ui/trip_page/trip_details.dart';
import 'package:amplify_trips_planner/features/trip/ui/trip_page/
trip_page_floating_button.dart';
import 'package:flutter/material.dart';
```

```
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class TripPage extends ConsumerWidget {
  const TripPage({
    required this.tripId,
    super.key,
  });
  final String tripId;

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    final tripValue = ref.watch(tripControllerProvider(tripId));

    return Scaffold(
      appBar: AppBar(
        centerTitle: true,
        title: const Text(
          'Amplify Trips Planner',
        ),
        actions: [
          IconButton(
            onPressed: () {
              context.goNamed(
                AppRoute.home.name,
              );
            },
            icon: const Icon(Icons.home),
          ),
        ],
        backgroundColor: const Color(constants.primaryColorDark),
      ),
      drawer: const TheNavigationDrawer(),
      floatingActionButton: TripPageFloatingButton(
        trip: tripValue,
      ),
      body: TripDetails(
        tripId: tripId,
        trip: tripValue,
      ),
    );
  }
}
```

8. Open the **lib/features/trip/ui/trip_page/trip_details.dart** file and update it as shown in the following to display the list of activities for the trip.

```
    Expanded(
    child: ActivitiesList(
    trip: value,
    ),
)
```

The **trip_details.dart** should look like the following code snippet.

```
import 'package:amplify_trips_planner/features/activity/ui/activities_list/
activities_list.dart';
import 'package:amplify_trips_planner/features/trip/controller/trip_controller.dart';
import 'package:amplify_trips_planner/features/trip/ui/trip_page/
selected_trip_card.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

class TripDetails extends ConsumerWidget {
const TripDetails({
required this.trip,
required this.tripId,
super.key,
});

final AsyncValue<Trip> trip;
final String tripId;

@override
Widget build(BuildContext context, WidgetRef ref) {
switch (trip) {
  case AsyncData(:final value):
    return Column(
      crossAxisAlignment: CrossAxisAlignment.stretch,
      children: [
        const SizedBox(
          height: 8,
        ),
        SelectedTripCard(trip: value),
        const SizedBox(
```

```
          height: 20,
        ),
        const Divider(
          height: 20,
          thickness: 5,
          indent: 20,
          endIndent: 20,
        ),
        const Text(
          'Your Activities',
          textAlign: TextAlign.center,
          style: TextStyle(
            fontSize: 20,
            fontWeight: FontWeight.bold,
          ),
        ),
        const SizedBox(
          height: 8,
        ),
        Expanded(
          child: ActivitiesList(
            trip: value,
          ),
        )
      ],
    );

  case AsyncError():
    return Center(
      child: Column(
        children: [
          const Text('Error'),
          TextButton(
            onPressed: () async {
              ref.invalidate(tripControllerProvider(tripId));
            },
            child: const Text('Try again'),
          ),
        ],
      ),
    );
  case AsyncLoading():
    return const Center(
      child: CircularProgressIndicator(),
```

```
    );

  case _:
    return const Center(
      child: Text('Error'),
      );
    }
  }
}
```

9. Open the **lib/common/navigation/router/router.dart** file and update it to add the **AddActivityPage** route.

```
GoRoute(
  path: '/addActivity/:id',
  name: AppRoute.addActivity.name,
  builder: (context, state) {
    final tripId = state.pathParameters['id']!;
    return AddActivityPage(tripId: tripId);
  },
),
```

The **router.dart** should look like the following code snippet.

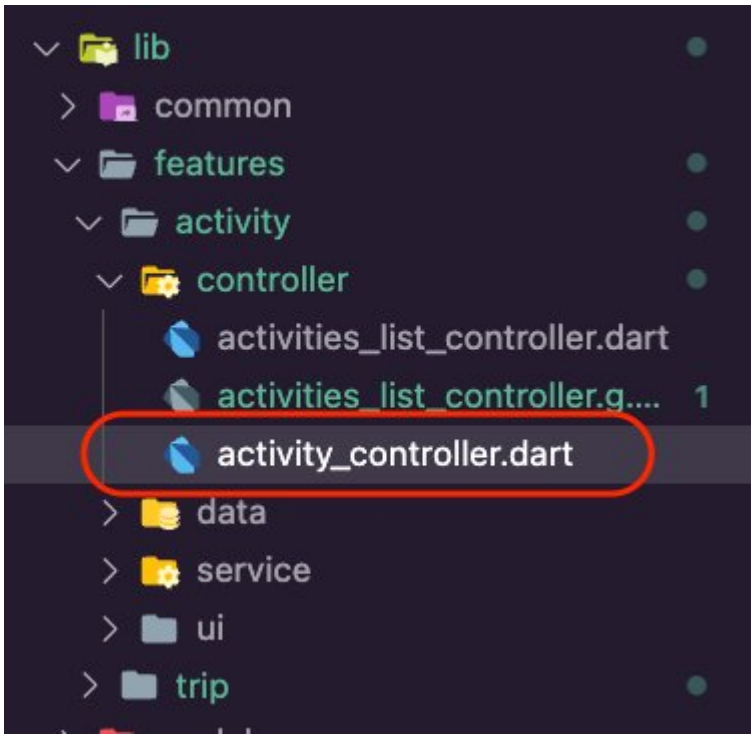
```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/features/activity/ui/add_activity/
add_activity_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/edit_trip_page/
edit_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trip_page/
past_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trips/
past_trips_list.dart';
import 'package:amplify_trips_planner/features/trip/ui/trip_page/trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/trips_list/
trips_list_page.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';

final router = GoRouter(
  routes: [
    GoRoute(
```

```
      path: '/',
      name: AppRoute.home.name,
      builder: (context, state) => const TripsListPage(),
    ),
    GoRoute(
      path: '/trip/:id',
      name: AppRoute.trip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return TripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/edittrip/:id',
      name: AppRoute.editTrip.name,
      builder: (context, state) {
        return EditTripPage(
          trip: state.extra! as Trip,
        );
      },
    ),
    GoRoute(
      path: '/pasttrip/:id',
      name: AppRoute.pastTrip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return PastTripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/pasttrips',
      name: AppRoute.pastTrips.name,
      builder: (context, state) => const PastTripsList(),
    ),
    GoRoute(
      path: '/addActivity/:id',
      name: AppRoute.addActivity.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return AddActivityPage(tripId: tripId);
      },
    ),
  ],
  errorBuilder: (context, state) => Scaffold(
```

```
    body: Center(
      child: Text(state.error.toString()),
    ),
  ),
);
```

Implement the UI for the Activity Details page

1. Create a new dart file inside the folder **lib/features/activity/controller** and name it **activity_controller.dart.**



2. Open the **activity_controller.dart** file and update it with the following code. The UI will use this controller for editing and deleting an activity using its ID. The UI will also use the controller for uploading a file for the activity.

> **Note**
>
> VSCode will show errors due to the missing **activity_controller.g.dart** file. You will fix that in the next step.

```
import 'dart:io';
```

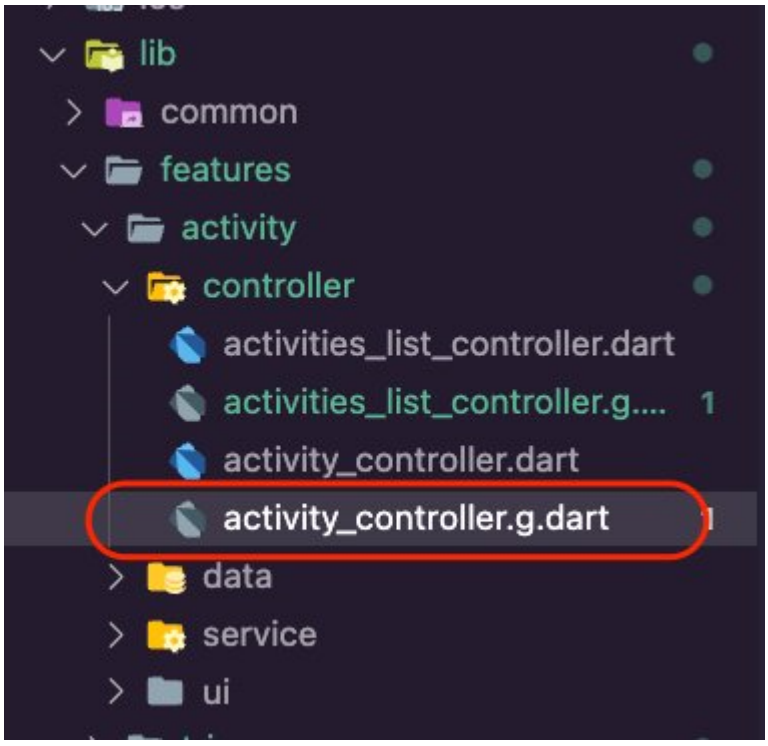
```
import 'package:amplify_trips_planner/common/services/storage_service.dart';
import 'package:amplify_trips_planner/features/activity/data/
activities_repository.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:riverpod_annotation/riverpod_annotation.dart';

part 'activity_controller.g.dart';

@riverpod
class ActivityController extends _$ActivityController {
  Future<Activity> _fetchActivity(String activityId) async {
    final activitiesRepository = ref.read(activitiesRepositoryProvider);
    return activitiesRepository.getActivity(activityId);
  }

  @override
  FutureOr<Activity> build(String activityId) async {
    return _fetchActivity(activityId);
  }

  Future<void> uploadFile(File file, Activity activity) async {
    final fileKey = await ref.read(storageServiceProvider).uploadFile(file);
    if (fileKey != null) {
      final imageUrl =
          await ref.read(storageServiceProvider).getImageUrl(fileKey);
      final updatedActivity = activity.copyWith(
        activityImageKey: fileKey,
        activityImageUrl: imageUrl,
      );
      await updateActivity(updatedActivity);
      ref.read(storageServiceProvider).resetUploadProgress();
    }
  }

  Future<String> getFileUrl(Activity activity) async {
    final fileKey = activity.activityImageKey;

    return ref.read(storageServiceProvider).getImageUrl(fileKey!);
  }

  ValueNotifier<double> uploadProgress() {
    return ref.read(storageServiceProvider).getUploadProgress();
```

```
  }

  Future<void> updateActivity(Activity activity) async {
    state = const AsyncValue.loading();
    state = await AsyncValue.guard(() async {
      final activitiesRepository = ref.read(activitiesRepositoryProvider);
      await activitiesRepository.update(activity);
      return _fetchActivity(activity.id);
    });
  }
}
```

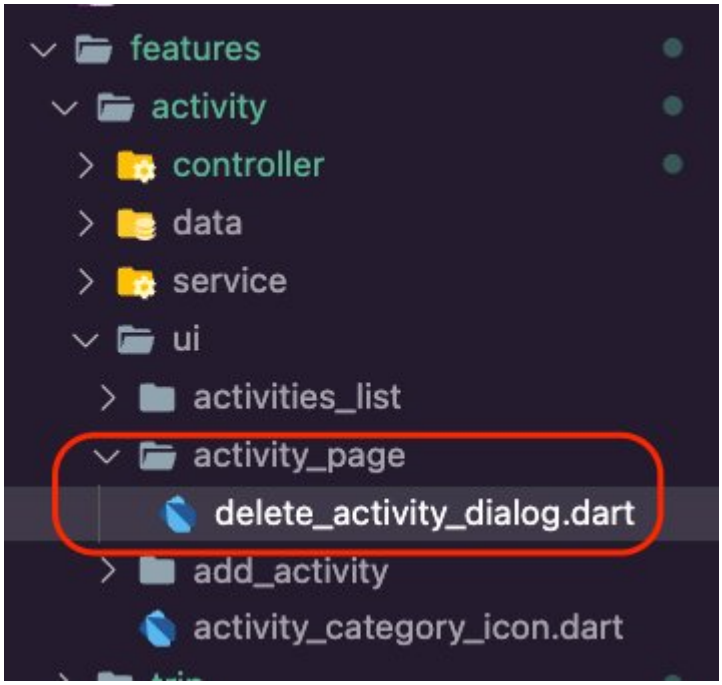
3. Navigate to the app's root folder and run the following command in your terminal.

```
dart run build_runner build -d
```

   This will generate the **activity_controller.g.dart** file inside the **lib/feature/activity/controller** folder.



4. Create a new folder inside the **lib/features/activity/ui** folder, name it **activity_page**, and then create the file **delete_activity_dialog.dart** inside it.

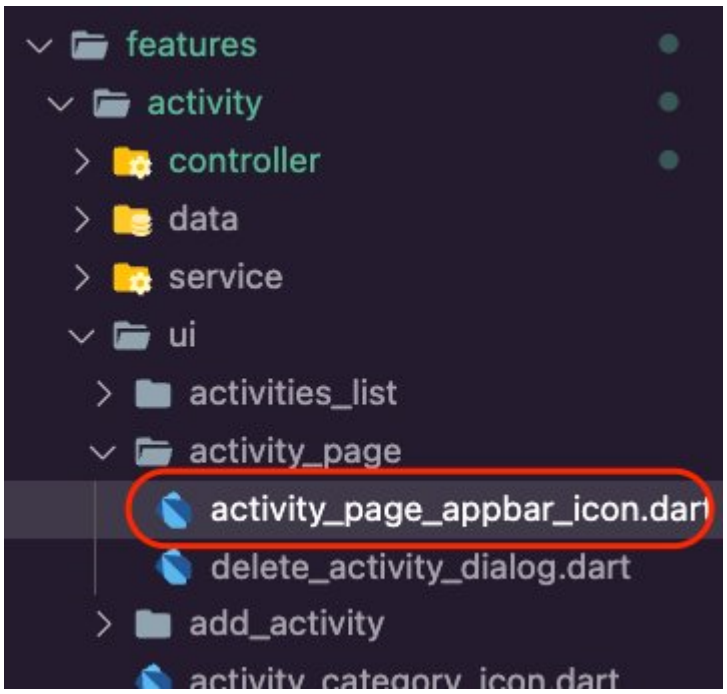


5. Open the **delete_activity_dialog.dart** file and update it with the following code. This will display a dialog for the user to confirm deleting the selected activity.

```
import 'package:flutter/material.dart';

class DeleteActivityDialog extends StatelessWidget {
  const DeleteActivityDialog({
    super.key,
  });

  @override
  Widget build(BuildContext context) {
    return AlertDialog(
      title: const Text('Please Confirm'),
      content: const Text('Delete this activity?'),
      actions: [
        TextButton(
          onPressed: () async {
            Navigator.of(context).pop(true);
          },
          child: const Text('Yes'),
        ),
        TextButton(
          onPressed: () {
            Navigator.of(context).pop(false);
```

```
          },
          child: const Text('No'),
        )
      ],
    );
  }
}
```

6. Create a new dart file inside the folder **lib/features/activity/ui/activity_page** and name it **activity_page_appbar_icon.dart**.

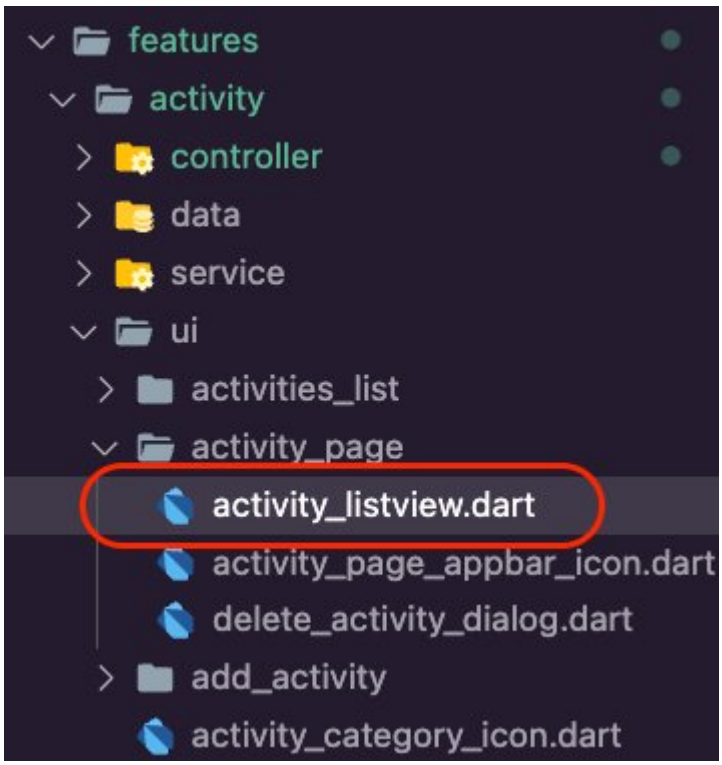


7. Open the **activity_page_appbar_icon.dart** file and update it with the following code to create an **IconButton** to navigate back to the activity's trip page.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class ActivityPageAppBarIcon extends StatelessWidget {
  const ActivityPageAppBarIcon({
    super.key,
    required this.activity,
  });
```

```
  final AsyncValue<Activity> activity;

  @override
  Widget build(BuildContext context) {
    switch (activity) {
      case AsyncData(:final value):
        return IconButton(
          onPressed: () {
            context.goNamed(
              AppRoute.trip.name,
              pathParameters: {'id': value.trip.id},
            );
          },
          icon: const Icon(Icons.arrow_back),
        );

      case AsyncError():
        return const Placeholder();
      case AsyncLoading():
        return const SizedBox();

      case _:
        return const Center(
          child: Text('Error'),
        );
    }
  }
}
```

8. Create a new dart file inside the folder **lib/features/activity/ui/activity_page** and name it **activity_listview.dart**.



9. Open the **activity_listview.dart** file and update it with the following code to display the activity details and enable the user to upload and open a file for the activity.

```
import 'dart:io';

import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/ui/upload_progress_dialog.dart';
import 'package:amplify_trips_planner/common/utils/date_time_formatter.dart';
import 'package:amplify_trips_planner/features/activity/controller/
activities_list_controller.dart';

import 'package:amplify_trips_planner/features/activity/controller/
activity_controller.dart';
import 'package:amplify_trips_planner/features/activity/ui/
activity_category_icon.dart';
import 'package:amplify_trips_planner/features/activity/ui/activity_page/
delete_activity_dialog.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:file_picker/file_picker.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';
import 'package:url_launcher/url_launcher.dart';
```

```
class ActivityListView extends ConsumerWidget {
  const ActivityListView({
    required this.activity,
    super.key,
  });

  final AsyncValue<Activity> activity;

  Future<bool> deleteActivity(
    BuildContext context,
    WidgetRef ref,
    Activity activity,
  ) async {
    var value = await showDialog<bool>(
      context: context,
      builder: (BuildContext context) {
        return const DeleteActivityDialog();
      },
    );
    value ??= false;

    if (value) {
      await ref
          .watch(activitiesListControllerProvider(activity.trip.id).notifier)
          .removeActivity(activity);
    }

    return value;
  }

  Future<void> openFile({
    required BuildContext context,
    required WidgetRef ref,
    required Activity activity,
  }) async {
    final fileUrl = await ref
        .watch(activityControllerProvider(activity.id).notifier)
        .getFileUrl(activity);

    final url = Uri.parse(fileUrl);
    await launchUrl(url);
  }

  Future<bool> uploadFile({
```

```
    required BuildContext context,
    required WidgetRef ref,
    required Activity activity,
  }) async {
    final result = await FilePicker.platform.pickFiles(
      type: FileType.custom,
      allowedExtensions: ['jpg', 'pdf', 'png'],
    );

    if (result == null) {
      return false;
    }

    final platformFile = result.files.first;

    final file = File(platformFile.path!);
    if (context.mounted) {
      await showDialog<String>(
        context: context,
        barrierDismissible: false,
        builder: (BuildContext context) {
          return const UploadProgressDialog();
        },
      );
      await ref
          .watch(activityControllerProvider(activity.id).notifier)
          .uploadFile(file, activity);
    }
    return true;
  }

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    switch (activity) {
      case AsyncData(:final value):
        return ListView(
          children: [
            Card(
              child: ListTile(
                leading: ActivityCategoryIcon(activityCategory: value.category),
                title: Text(
                  value.activityName,
                  style: Theme.of(context).textTheme.titleLarge,
                ),
```

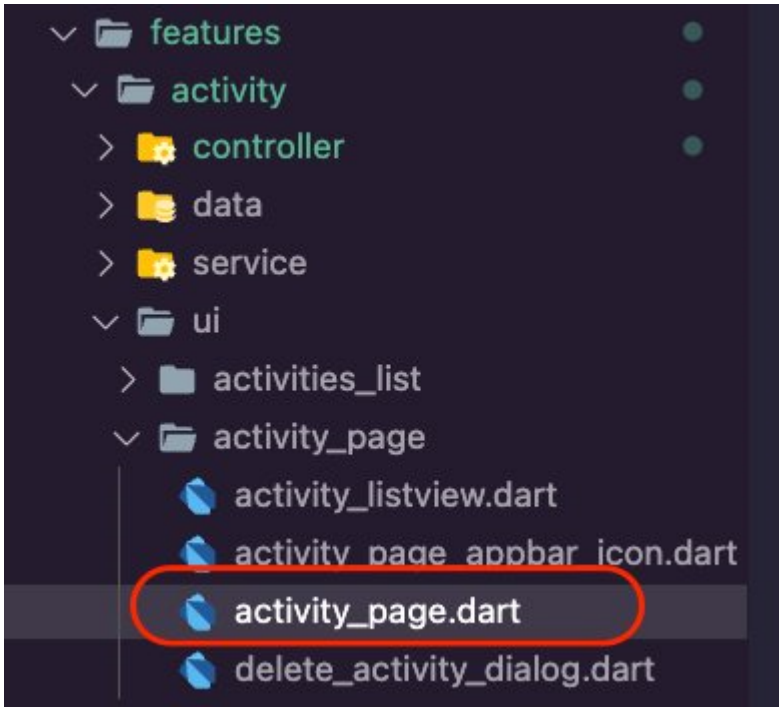
```
              subtitle: Text(value.category.name),
            ),
          ),
          ListTile(
            dense: true,
            title: Text(
              'Activity Date',
              style: Theme.of(context)
                  .textTheme
                  .titleSmall!
                  .copyWith(color: Colors.white),
            ),
            tileColor: Colors.grey,
          ),
          Card(
            child: ListTile(
              title: Text(
                value.activityDate.getDateTime().format('EE MMMM dd'),
                style: Theme.of(context).textTheme.titleLarge,
              ),
              subtitle: Text(
                value.activityTime!.getDateTime().format('hh:mm a'),
              ),
            ),
          ),
          ListTile(
            dense: true,
            title: Text(
              'Documents',
              style: Theme.of(context)
                  .textTheme
                  .titleSmall!
                  .copyWith(color: Colors.white),
            ),
            tileColor: Colors.grey,
          ),
          Card(
            child: value.activityImageUrl != null
                ? Row(
                    mainAxisAlignment: MainAxisAlignment.spaceAround,
                    children: [
                      TextButton(
                        style: TextButton.styleFrom(
                          textStyle: const TextStyle(fontSize: 20),
```

```
                        ),
                        onPressed: () {
                          openFile(
                            context: context,
                            ref: ref,
                            activity: value,
                          );
                        },
                        child: const Text('Open'),
                      ),
                      TextButton(
                        style: TextButton.styleFrom(
                          textStyle: const TextStyle(fontSize: 20),
                        ),
                        onPressed: () {
                          uploadFile(
                            context: context,
                            activity: value,
                            ref: ref,
                          ).then(
                            (isUploaded) => isUploaded ? context.pop() : null,
                          );
                        },
                        child: const Text('Replace'),
                      ),
                    ],
                  )
                : ListTile(
                    title: TextButton(
                      style: TextButton.styleFrom(
                        textStyle: const TextStyle(fontSize: 20),
                      ),
                      onPressed: () {
                        uploadFile(
                          context: context,
                          activity: value,
                          ref: ref,
                        ).then(
                          (isUploaded) => isUploaded ? context.pop() : null,
                        );
                        // Navigator.of(context, rootNavigator: true)
                        //     .pop());
                      },
                      child: const Text('Attach a PDF or photo'),
```

```
                    ),
                  ),
              ),
              const ListTile(
                dense: true,
                tileColor: Colors.grey,
                visualDensity: VisualDensity(vertical: -4),
              ),
              Card(
                child: Row(
                  mainAxisAlignment: MainAxisAlignment.spaceAround,
                  children: [
                    TextButton(
                      style: TextButton.styleFrom(
                        textStyle: const TextStyle(fontSize: 20),
                      ),
                      onPressed: () {
                        context.goNamed(
                          AppRoute.editActivity.name,
                          pathParameters: {'id': value.id},
                          extra: value,
                        );
                      },
                      child: const Text('Edit'),
                    ),
                    TextButton(
                      style: TextButton.styleFrom(
                        textStyle: const TextStyle(fontSize: 20),
                      ),
                      onPressed: () {
                        deleteActivity(context, ref, value).then(
                          (isDeleted) {
                            if (isDeleted) {
                              context.goNamed(
                                AppRoute.trip.name,
                                pathParameters: {'id': value.trip.id},
                              );
                            }
                          },
                        );
                      },
                      child: const Text('Delete'),
                    ),
                  ],
```

```
              ),
            )
          ],
        );

      case AsyncError():
        return const Center(
          child: Text('Error'),
        );
      case AsyncLoading():
        return const Center(
          child: CircularProgressIndicator(),
        );

      case _:
        return const Center(
          child: Text('Error'),
        );
    }
  }
}
```

10. Create a new dart file inside the folder **lib/features/activity/ui/activity_page** and name it **activity_page.dart**.



11. Open the **activity_page.dart** file and update it with the following code to create the **ActivityPage** which will use the **ActivityListView** you created above to display the activity's details.

```
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/activity/controller/
activity_controller.dart';
import 'package:amplify_trips_planner/features/activity/ui/activity_page/
activity_listview.dart';
import 'package:amplify_trips_planner/features/activity/ui/activity_page/
activity_page_appbar_icon.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

class ActivityPage extends ConsumerWidget {
  const ActivityPage({
    required this.activityId,
    super.key,
  });

  final String activityId;

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    final activityValue = ref.watch(activityControllerProvider(activityId));

    return Scaffold(
      appBar: AppBar(
        centerTitle: true,
        title: const Text(
          'Amplify Trips Planner',
        ),
        leading: ActivityPageAppBarIcon(
          activity: activityValue,
        ),
        backgroundColor: const Color(constants.primaryColorDark),
      ),
      body: ActivityListView(
        activity: activityValue,
      ),
    );
  }
```

```
}
```

12. Open the **/lib/common/navigation/router/router.dart** file and update it to add the **AddActivityPage** route.

```
    GoRoute(
      path: '/activity/:id',
      name: AppRoute.activity.name,
      builder: (context, state) {
        final activityId = state.pathParameters['id']!;
        return ActivityPage(activityId: activityId);
      },
    ),
```

The **router.dart** file should look like the following code snippet.

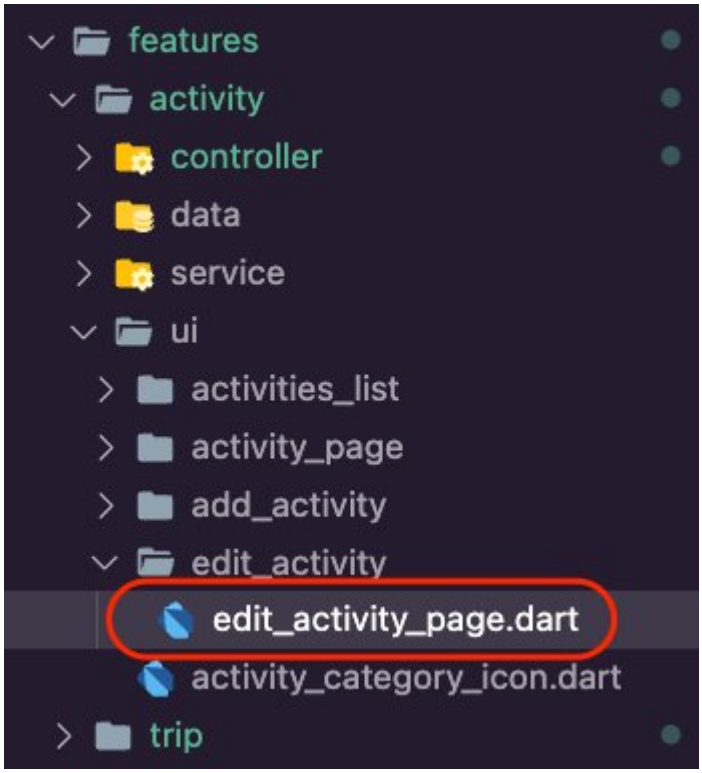
```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/features/activity/ui/activity_page/
activity_page.dart';
import 'package:amplify_trips_planner/features/activity/ui/add_activity/
add_activity_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/edit_trip_page/
edit_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trip_page/
past_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trips/
past_trips_list.dart';
import 'package:amplify_trips_planner/features/trip/ui/trip_page/trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/trips_list/
trips_list_page.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';

final router = GoRouter(
  routes: [
    GoRoute(
      path: '/',
      name: AppRoute.home.name,
      builder: (context, state) => const TripsListPage(),
    ),
    GoRoute(
      path: '/trip/:id',
```

```
      name: AppRoute.trip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return TripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/edittrip/:id',
      name: AppRoute.editTrip.name,
      builder: (context, state) {
        return EditTripPage(
          trip: state.extra! as Trip,
        );
      },
    ),
    GoRoute(
      path: '/pasttrip/:id',
      name: AppRoute.pastTrip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return PastTripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/pasttrips',
      name: AppRoute.pastTrips.name,
      builder: (context, state) => const PastTripsList(),
    ),
    GoRoute(
      path: '/addActivity/:id',
      name: AppRoute.addActivity.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return AddActivityPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/activity/:id',
      name: AppRoute.activity.name,
      builder: (context, state) {
        final activityId = state.pathParameters['id']!;
        return ActivityPage(activityId: activityId);
      },
    ),
```

```
  ],
  errorBuilder: (context, state) => Scaffold(
    body: Center(
      child: Text(state.error.toString()),
    ),
  ),
);
```

Implement the UI for editing an activity

1. Create a new folder inside the **lib/features/activity/ui** folder, name it **edit_activity**, and then create the file **edit_activity_page.dart** inside it.



2. Open the **edit_activity_page.dart** file and update it with the following code. This will allow us to present a form to the user to update the details of the selected activity.

```
import 'package:amplify_flutter/amplify_flutter.dart';
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/ui/bottomsheet_text_form_field.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/common/utils/date_time_formatter.dart';
```

```
import 'package:amplify_trips_planner/features/activity/controller/
activity_controller.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';
import 'package:intl/intl.dart';

class EditActivityPage extends ConsumerStatefulWidget {
 const EditActivityPage({
   required this.activity,
   super.key,
 });

 final Activity activity;

 @override
 EditActivityPageState createState() => EditActivityPageState();
}

class EditActivityPageState extends ConsumerState<EditActivityPage> {
 @override
 void initState() {
   activityNameController.text = widget.activity.activityName;
   activityDateController.text =
       widget.activity.activityDate.getDateTime().format('yyyy-MM-dd');

   activityTime =
       TimeOfDay.fromDateTime(widget.activity.activityTime!.getDateTime());
   activityTimeController.text =
       widget.activity.activityTime!.getDateTime().format('hh:mm a');

   activityCategoryController.text = widget.activity.category.name;

   activityCategory = widget.activity.category;

   super.initState();
 }

 final formGlobalKey = GlobalKey<FormState>();
 final activityNameController = TextEditingController();
 final activityDateController = TextEditingController();

 var activityTime = TimeOfDay.now();
```

```
final activityTimeController = TextEditingController();

final activityCategoryController = TextEditingController();

var activityCategory = ActivityCategory.Flight;

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      centerTitle: true,
      title: const Text(
        'Amplify Trips Planner',
      ),
      leading: IconButton(
        onPressed: () {
          context.goNamed(
            AppRoute.activity.name,
            pathParameters: {'id': widget.activity.id},
          );
        },
        icon: const Icon(Icons.arrow_back),
      ),
      backgroundColor: const Color(constants.primaryColorDark),
    ),
    body: SingleChildScrollView(
      child: Form(
        key: formGlobalKey,
        child: Container(
          padding: EdgeInsets.only(
            top: 15,
            left: 15,
            right: 15,
            bottom: MediaQuery.of(context).viewInsets.bottom + 15,
          ),
          width: double.infinity,
          child: Column(
            mainAxisSize: MainAxisSize.min,
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              BottomSheetTextFormField(
                labelText: 'Activity Name',
                controller: activityNameController,
                keyboardType: TextInputType.name,
```

```
            ),
            const SizedBox(
              height: 20,
            ),
            DropdownButtonFormField<ActivityCategory>(
              onChanged: (value) {
                activityCategoryController.text = value!.name;
                activityCategory = value;
              },
              value: activityCategory,
              decoration: const InputDecoration(
                labelText: 'Category',
              ),
              items: [
                for (var category in ActivityCategory.values)
                  DropdownMenuItem(
                    value: category,
                    child: Text(category.name),
                  ),
              ],
            ),
            const SizedBox(
              height: 20,
            ),
            BottomSheetTextFormField(
              labelText: 'Activity Date',
              controller: activityDateController,
              keyboardType: TextInputType.datetime,
              onTap: () async {
                final pickedDate = await showDatePicker(
                  context: context,
                  initialDate: DateTime.parse(
                      widget.activity.activityDate.toString()),
                  firstDate: DateTime.parse(
                      widget.activity.trip.startDate.toString()),
                  lastDate: DateTime.parse(
                      widget.activity.trip.endDate.toString()),
                );

                if (pickedDate != null) {
                  activityDateController.text =
                      pickedDate.format('yyyy-MM-dd');
                } else {}
              },
```

```
                ),
                const SizedBox(
                  height: 20,
                ),
                BottomSheetTextFormField(
                  labelText: 'Activity Time',
                  controller: activityTimeController,
                  keyboardType: TextInputType.datetime,
                  onTap: () async {
                    await showTimePicker(
                      context: context,
                      initialTime: activityTime,
                      initialEntryMode: TimePickerEntryMode.dial,
                    ).then((timeOfDay) {
                      if (timeOfDay != null) {
                        final localizations = MaterialLocalizations.of(context);
                        final formattedTimeOfDay =
                            localizations.formatTimeOfDay(timeOfDay);

                        activityTimeController.text = formattedTimeOfDay;
                        activityTime = timeOfDay;
                      }
                    });
                  },
                ),
                const SizedBox(
                  height: 20,
                ),
                TextButton(
                  child: const Text('OK'),
                  onPressed: () async {
                    final currentState = formGlobalKey.currentState;
                    if (currentState == null) {
                      return;
                    }
                    if (currentState.validate()) {
                      final format = DateFormat.jm();

                      activityTime = TimeOfDay.fromDateTime(
                        format.parse(activityTimeController.text),
                      );

                      final now = DateTime.now();
                      final time = DateTime(
```

```
                        now.year,
                        now.month,
                        now.day,
                        activityTime.hour,
                        activityTime.minute,
                      );

                      final updatedActivity = widget.activity.copyWith(
                        category: ActivityCategory.values
                            .byName(activityCategoryController.text),
                        activityName: activityNameController.text,
                        activityDate: TemporalDate(
                          DateTime.parse(activityDateController.text),
                        ),
                        activityTime: TemporalTime.fromString(
                          time.format('HH:mm:ss.sss'),
                        ),
                      );

                      await ref
                          .watch(
                            activityControllerProvider(widget.activity.id)
                                .notifier,
                          )
                          .updateActivity(updatedActivity);
                      if (context.mounted) {
                        context.goNamed(
                          AppRoute.activity.name,
                          pathParameters: {'id': widget.activity.id},
                        );
                      }
                    }
                  },
                ),
              ],
            ),
          ),
        ),
      ),
    );
  }
}
```

3. Open the **/lib/common/navigation/router/router.dart** file and update it to add the **EditActivityPage** route.

```
GoRoute(
 path: '/editactivity/:id',
 name: AppRoute.editActivity.name,
 builder: (context, state) {
   return EditActivityPage(
     activity: state.extra! as Activity,
   );
 },
),
```

The **router.dart** should look like the following code snippet.

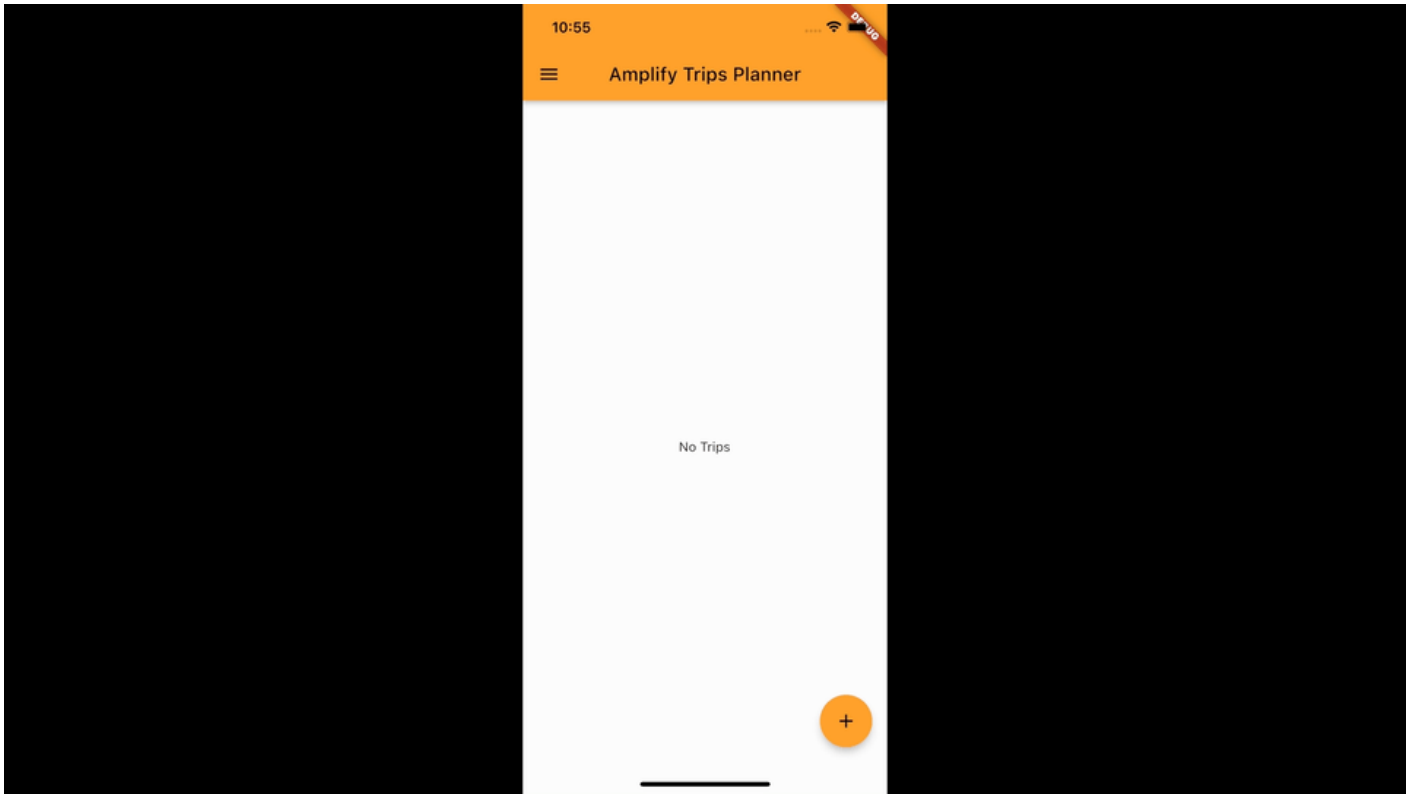
```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/features/activity/ui/activity_page/
activity_page.dart';
import 'package:amplify_trips_planner/features/activity/ui/add_activity/
add_activity_page.dart';
import 'package:amplify_trips_planner/features/activity/ui/edit_activity/
edit_activity_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/edit_trip_page/
edit_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trip_page/
past_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trips/
past_trips_list.dart';
import 'package:amplify_trips_planner/features/trip/ui/trip_page/trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/trips_list/
trips_list_page.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';

final router = GoRouter(
 routes: [
   GoRoute(
     path: '/',
     name: AppRoute.home.name,
     builder: (context, state) => const TripsListPage(),
   ),
```

```
    GoRoute(
      path: '/trip/:id',
      name: AppRoute.trip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return TripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/edittrip/:id',
      name: AppRoute.editTrip.name,
      builder: (context, state) {
        return EditTripPage(
          trip: state.extra! as Trip,
        );
      },
    ),
    GoRoute(
      path: '/pasttrip/:id',
      name: AppRoute.pastTrip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return PastTripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/pasttrips',
      name: AppRoute.pastTrips.name,
      builder: (context, state) => const PastTripsList(),
    ),
    GoRoute(
      path: '/addActivity/:id',
      name: AppRoute.addActivity.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return AddActivityPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/activity/:id',
      name: AppRoute.activity.name,
      builder: (context, state) {
        final activityId = state.pathParameters['id']!;
        return ActivityPage(activityId: activityId);
```

```
      },
    ),
    GoRoute(
      path: '/editactivity/:id',
      name: AppRoute.editActivity.name,
      builder: (context, state) {
        return EditActivityPage(
          activity: state.extra! as Activity,
        );
      },
    ),
  ],
  errorBuilder: (context, state) => Scaffold(
    body: Center(
      child: Text(state.error.toString()),
    ),
  ),
);
```

4. Run the app in an emulator or simulator and create a trip, then add a few activities to it. The following is an example using an iPhone simulator.

> **Note**
>
> Due to the changes in the data schema, you need to erase the app and its contents from the emulator or simulator.



Conclusion

In this module, you introduced create, read, update, and delete (CRUD) functionality for trip activities in your app. You also updated the Amplify API to retrieve and persist your trip's activities data.

Module 4: Add the Profile feature

Overview

In this module, you will add the ability to display and edit the user's profile in your app. You will add an Amplify function to create the profile when the user completes sign-up.

The Amplify function will interact with a GraphQL API that uses AWS AppSync (a managed GraphQL service) backed by DynamoDB (a NoSQL database).

What you will accomplish

In this module, you will:

- Add the profile data model to the app
- Add an Amplify function to create the user profile
- Implement the UI for displaying and editing the profile

Implementation

Minimum time to complete	45 minutes

Add profile data model to the app

1. You need to configure the API for AWS Identity and Access Management (IAM) authentication to grant the Lambda function access to the GraphQL API. Run the following command in the root folder of the app to update the authorization modes of the API.

   ```
   amplify update api
   ```

2. Select the **GraphQL** option, then select **Authorization modes** to edit it. Accept the **Amazon Cognito User Pool** as the default authorization type, select **Y** in response to the **Configure additional auth types** prompt. Finally, choose **IAM** from the additional authorization types.

```
? Select from one of the below mentioned services: GraphQL

General information

- Name: amplifytripsplanner

- API endpoint: https://qmrokxalkng4xdaegffe4g2nsq.appsync-api.us-
west-1.amazonaws.com/graphql

Authorization modes

- Default: Amazon Cognito User Pool

Conflict detection (required for DataStore)

- Disabled

? Select a setting to edit Authorization modes
? Choose the default authorization type for the API Amazon Cognito User Pool
Use a Cognito user pool configured as a part of this project.
? Configure additional auth types? Yes
? Choose the additional authorization types you want to configure for the API IAM
```

3. Open the **amplify/backend/api/amplifytripsplanner/schema.graphql** file and add a data model for the profile

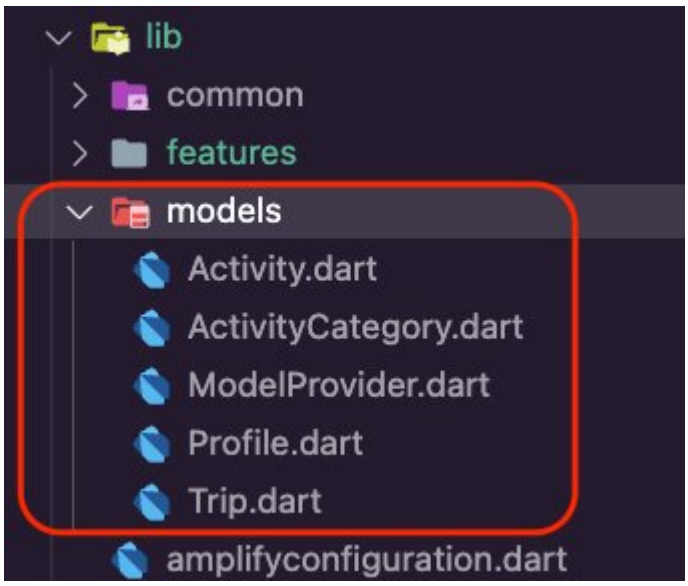
```
type Trip @model @auth(rules: [{ allow: owner }]) {
 id: ID!
 tripName: String!
 destination: String!
 startDate: AWSDate!
 endDate: AWSDate!
 tripImageUrl: String
 tripImageKey: String
 Activities: [Activity] @hasMany(indexName: "byTrip", fields: ["id"])
}

type Activity @model @auth(rules: [{allow: owner}]) {
 id: ID!
 activityName: String!
 tripID: ID! @index(name: "byTrip", sortKeyFields: ["activityName"])
 trip: Trip! @belongsTo(fields: ["tripID"])
```

```
 activityImageUrl: String
 activityImageKey: String
 activityDate: AWSDate!
 activityTime: AWSTime
 category: ActivityCategory!
}

type Profile
 @model
 @auth(
   rules: [
     { allow: private, provider: iam }
     { allow: owner, operations: [read, update, create] }
   ]
 ) {
 id: ID!
 email: String!
 firstName: String
 lastName: String
 homeCity: String
 owner: String!
}

enum ActivityCategory { Flight, Lodging, Meeting, Restaurant }
```

4. Run the following command in the root folder of the app to generate the models files.

```
amplify codegen models
```

The Amplify CLI will generate the dart files in the **lib/models** folder.



5. Run the command **amplify push** to create the resources in the cloud.

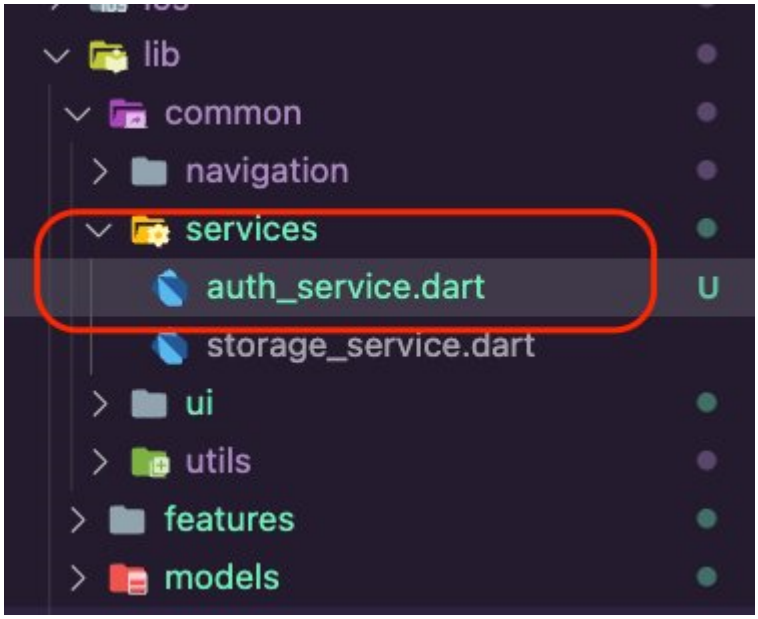
```
    Current Environment: dev

| Category | Resource name                | Operation | Provider plugin     |
| Api      | amplifytripsplanner          | Update    | awscloudformation   |
| Auth     | amplifytripsplanner1ec293ff  | No Change | awscloudformation   |
| Storage  | s3b101c2c7                   | No Change | awscloudformation   |

? Are you sure you want to continue? (Y/n) ›
```

6. Press **Enter**. The Amplify CLI will deploy the resources and display a confirmation, as shown in the screenshot.

```
        GraphQLAPI                      AWS::AppSync::GraphQLApi       UPDATE_COMPL
        AuthRolePolicy01921FC820        AWS::IAM::ManagedPolicy        CREATE_COMPL
        GraphQLAPITransformerSchema3C… AWS::AppSync::GraphQLSchema     UPDATE_COMPL
        Trip                            AWS::CloudFormation::Stack     UPDATE_COMPL
        Activity                        AWS::CloudFormation::Stack     UPDATE_COMPL
        Profile                         AWS::CloudFormation::Stack     CREATE_IN_PR
        ConnectionStack                 AWS::CloudFormation::Stack     UPDATE_IN_PR

Deployment state saved successfully.

GraphQL endpoint: https://qmrokxalkng4xdaegffe4g2nsq.appsync-api.us-west-1.amazonaws.
com/graphql

GraphQL transformer version: 2
```

Create the Auth service

1. Create a new dart file inside the folder **lib/common/services** and name it **auth_service.dart**.



2. Open **auth_service.dart** file and update it with the following code to create the **AuthService**. The service will use Amplify to sign the user out.

```
import 'package:amplify_flutter/amplify_flutter.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

final authServiceProvider = Provider<AuthService>((ref) {
  return AuthService();
});

class AuthService {
  AuthService();

  Future<void> signOut() async {
    try {
      await Amplify.Auth.signOut();
    } on Exception catch (e) {
      debugPrint(e.toString());
    }
  }
}
```

3. Open the file **lib/common/navigation/ui/the_navigation_drawer.dart**. Update it with the following code to add the option to sign out from the app.

```
          ListTile(
            leading: const Icon(Icons.exit_to_app),
            title: const Text('Logout'),
            onTap: () => ref.read(authServiceProvider).signOut(),
          ),
```

The **the_navigation_drawer.dart** should look like the following code snippet.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/services/auth_service.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class TheNavigationDrawer extends ConsumerWidget {
  const TheNavigationDrawer({
    super.key,
  });

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    return Drawer(
      child: ListView(
        padding: EdgeInsets.zero,
        children: [
          const DrawerHeader(
            decoration: BoxDecoration(
              color: Color(constants.primaryColorDark),
            ),
            padding: EdgeInsets.all(16),
            child: Column(
              children: [
                SizedBox(height: 10),
                Text(
                  'Amplify Trips Planner',
                  style: TextStyle(fontSize: 22, color: Colors.white),
                ),
              ],
```

```
            ),
          ),
          ListTile(
            leading: const Icon(Icons.home),
            title: const Text('Trips'),
            onTap: () {
              context.goNamed(
                AppRoute.home.name,
              );
            },
          ),
          ListTile(
            leading: const Icon(Icons.category),
            title: const Text('Past Trips'),
            onTap: () {
              context.goNamed(
                AppRoute.pastTrips.name,
              );
            },
          ),
          ListTile(
            leading: const Icon(Icons.exit_to_app),
            title: const Text('Logout'),
            onTap: () => ref.read(authServiceProvider).signOut(),
          ),
        ],
      ),
    );
  }
}
```

Add the Amplify function to create the profile

1. Navigate to the root folder of the app and run the following command in your terminal to update the authentication configurations.

```
amplify update auth
```

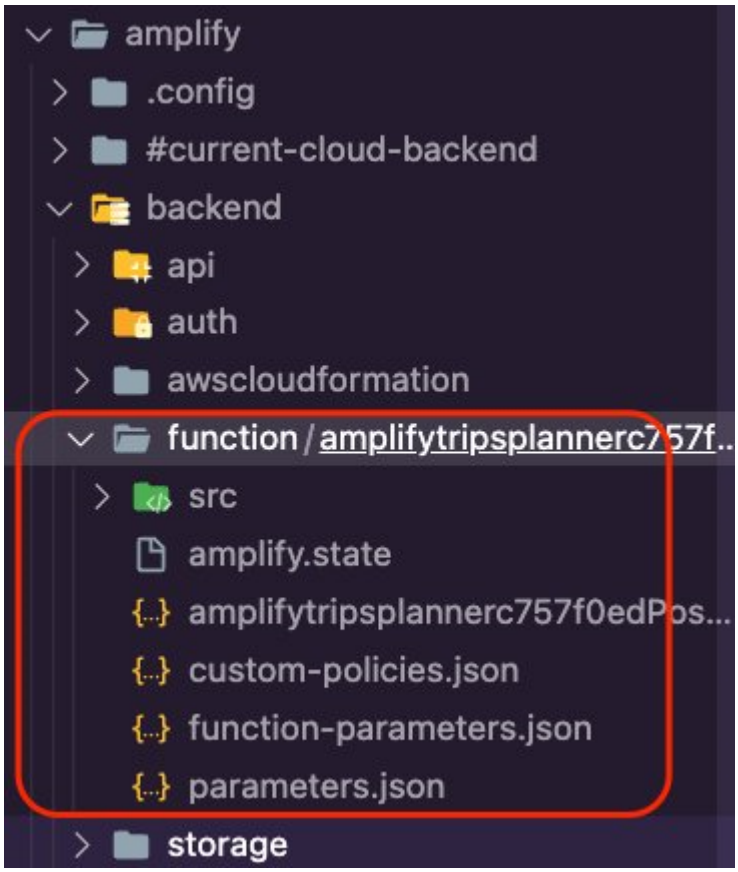
2. Go through the configuration and select **Yes** for configuring Lambda triggers and then enable a **Post Confirmation** trigger:

```
Please note that certain attributes may not be overwritten if you choose to use defaults setting
s.

You have configured resources that might depend on this Cognito resource.  Updating this Cognito
 resource could have unintended side effects.

Using service: Cognito, provided by: awscloudformation
 What do you want to do? Walkthrough all the auth configurations
 Select the authentication/authorization services that you want to use: User Sign-Up,
 Sign-In, connected with AWS IAM controls (Enables per-user Storage features for imag
es or other content, Analytics, and more)
 Allow unauthenticated logins? (Provides scoped down permissions that you can control
 via AWS IAM) No
 Do you want to enable 3rd party authentication providers in your identity pool? No
 Do you want to add User Pool Groups? No
 Do you want to add an admin queries API? No
 Multifactor authentication (MFA) user login options: OFF
 Email based user registration/forgot password: Enabled (Requires per-user email entr
y at registration)
 Specify an email verification subject: Your verification code
 Specify an email verification message: Your verification code is {####}
 Do you want to override the default password policy for this User Pool? No
 Specify the app's refresh token expiration period (in days): 30
 Do you want to specify the user attributes this app can read and write? No
 Do you want to enable any of the following capabilities?
 Do you want to use an OAuth flow? No
? Do you want to configure Lambda Triggers for Cognito? Yes
? Which triggers do you want to enable for Cognito Post Confirmation
? What functionality do you want to use for Post Confirmation Create your own module
Successfully added resource amplifytripsplanner1ec293ffPostConfirmation locally.
```

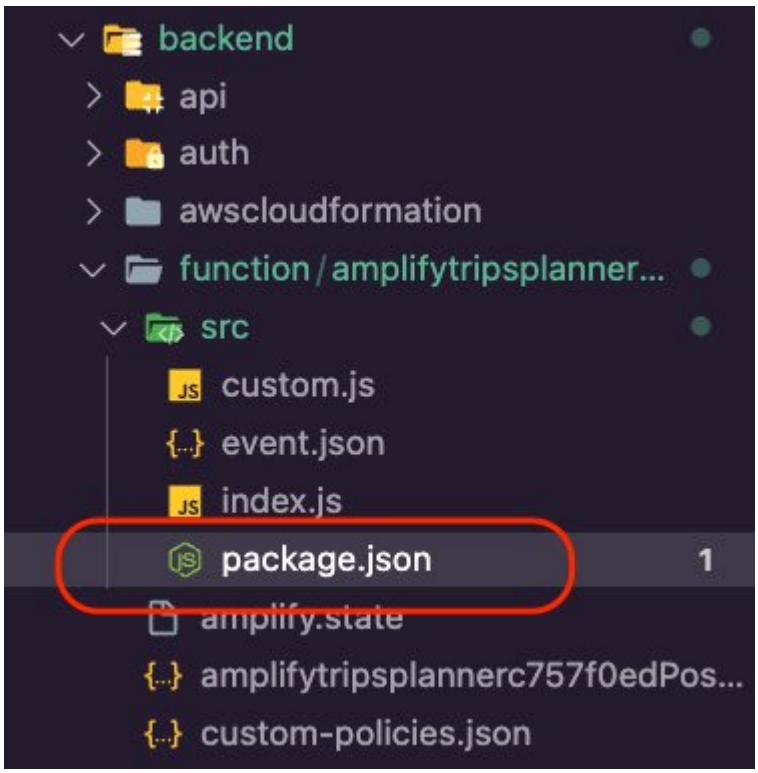
```
Please note that certain attributes may not be overwritten if you choose to use
 defaults settings.

You have configured resources that might depend on this Cognito resource.  Updating
 this Cognito resource could have unintended side effects.

Using service: Cognito, provided by: awscloudformation
What do you want to do? Walkthrough all the auth configurations
Select the authentication/authorization services that you want to use: User Sign-Up,
 Sign-In, connected with IAM controls (Enables per-user Storage features for images
 or other content, Analytics, and more)
Allow unauthenticated logins? (Provides scoped down permissions that you can control
 via IAM) No
Do you want to enable 3rd party authentication providers in your identity pool? No
Do you want to add User Pool Groups? No
Do you want to add an admin queries API? No
Multifactor authentication (MFA) user login options: OFF
Email based user registration/forgot password: Enabled (Requires per-user email entry
 at registration)
Specify an email verification subject: Your verification code
```

```
Specify an email verification message: Your verification code is {####}

Do you want to override the default password policy for this User Pool? No
Specify the app's refresh token expiration period (in days): 30
Do you want to specify the user attributes this app can read and write? No
Do you want to enable any of the following capabilities?
Do you want to use an OAuth flow? No
? Do you want to configure Lambda Triggers for Cognito? Yes
? Which triggers do you want to enable for Cognito Post Confirmation
? What functionality do you want to use for Post Confirmation Create your own module
Successfully added resource amplifytripsplannere57d0b5cPostConfirmation locally.
```

The Amplify CLI will add a new folder for the function which will include the **Post Confirmation** trigger where you will write the function to create the user's profile



3. **Open the** package.json **file in the** src** folder, which is inside the function's folder.



4. Update the **package.json** file as shown in the following to install the required dependencies.

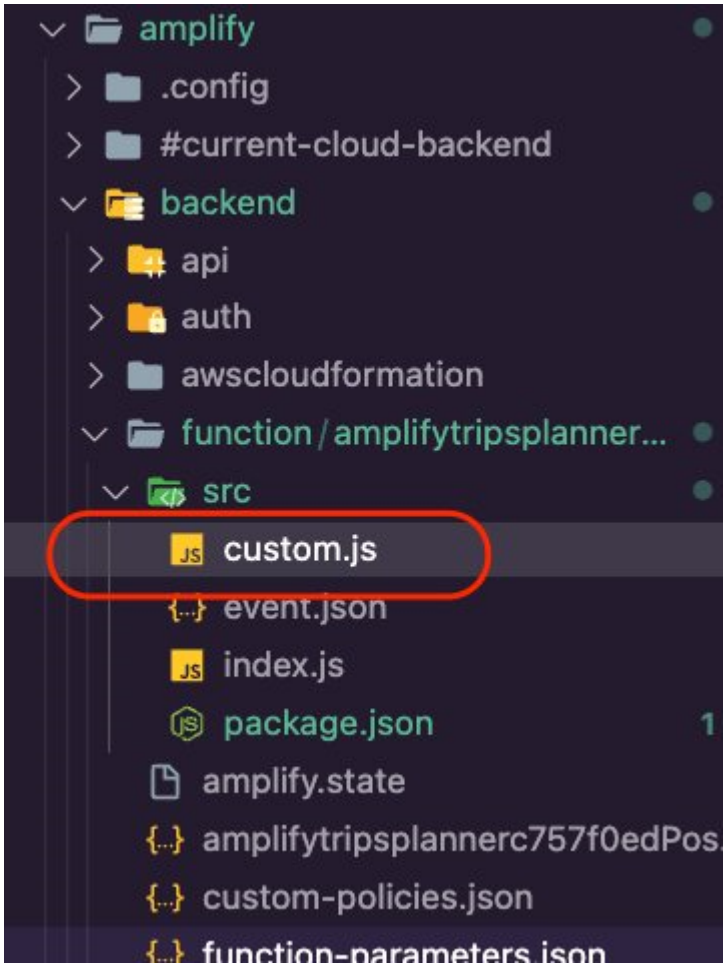
```
"dependencies": {
 "axios": "latest",
 "@aws-crypto/sha256-js": "^2.0.1",
 "@aws-sdk/credential-provider-node": "^3.76.0",
 "@aws-sdk/protocol-http": "^3.58.0",
 "@aws-sdk/signature-v4": "^3.58.0",
 "node-fetch": "2"
},
```

The **package.json** file should now look like the following.

```
{
 "name": "amplifytripsplannere57d0b5cPostConfirmation",
 "version": "2.0.0",
 "description": "Lambda function generated by Amplify",
 "main": "index.js",
 "license": "Apache-2.0",
 "dependencies": {
   "axios": "latest",
   "@aws-crypto/sha256-js": "^2.0.1",
   "@aws-sdk/credential-provider-node": "^3.76.0",
```

```
    "@aws-sdk/protocol-http": "^3.58.0",
    "@aws-sdk/signature-v4": "^3.58.0",
    "node-fetch": "2"
  },
  "devDependencies": {
    "@types/aws-lambda": "^8.10.92"
  }
}
```

5. Open the **custom.js** file inside the function **src** folder.



6. Update the **custom.js** file as shown in the following to run a GraphQL mutation and pass the required variables as arguments to create a profile record.

```
const { Sha256 } = require("@aws-crypto/sha256-js");
const { defaultProvider } = require("@aws-sdk/credential-provider-node");
const { SignatureV4 } = require("@aws-sdk/signature-v4");
const { HttpRequest } = require("@aws-sdk/protocol-http");
const { default: fetch, Request } = require("node-fetch");
```

```
const GRAPHQL_ENDPOINT =
 process.env.API_AMPLIFYTRIPSPLANNER_GRAPHQLAPIENDPOINTOUTPUT;
const AWS_REGION = process.env.AWS_REGION || 'us-east-1';

const query = /* GraphQL */ `
 mutation createProfile($email: String!,$owner: String!) {
   createProfile(input: {
     email: $email,
     owner: $owner,

   }) {
     email
   }
 }
`;

/**
* @type {import('@types/aws-lambda').PostConfirmationTriggerHandler}
*/
exports.handler = async (event) => {
 console.log(`EVENT: ${JSON.stringify(event)}`);

 const variables = {

     email: event.request.userAttributes.email,
     owner: `${event.request.userAttributes.sub}::${event.userName}`

 };

 const endpoint = new URL(GRAPHQL_ENDPOINT);

 const signer = new SignatureV4({
   credentials: defaultProvider(),
   region: AWS_REGION,
   service: 'appsync',
   sha256: Sha256
 });

 const requestToBeSigned = new HttpRequest({
   method: 'POST',
   headers: {
     'Content-Type': 'application/json',
     host: endpoint.host
```

```
    },
    hostname: endpoint.host,
    body: JSON.stringify({ query, variables }),
    path: endpoint.pathname
  });

  const signed = await signer.sign(requestToBeSigned);
  const request = new Request(endpoint, signed);

  let statusCode = 200;
  let body;
  let response;

  try {
    response = await fetch(request);
    body = await response.json();
    if (body.errors) statusCode = 400;
  } catch (error) {
    statusCode = 500;
    body = {
      errors: [
        {
          message: error.message
        }
      ]
    };
  }

  console.log(`statusCode: ${statusCode}`);
  console.log(`body: ${JSON.stringify(body)}`);

  return {
    statusCode,
    body: JSON.stringify(body)
  };
};
```

7. Navigate to the root folder of the app and run the following command in your terminal to update the configurations of the function you created above.

```
amplify update function
```

8. Select the function you created above and update the Resource access permissions to allow the function to access the API for Query, Mutation, and Subscription

```
? Select the Lambda function you want to update amplifytripsplannere57d0b5cPostConfi
rmation
General information
- Name: amplifytripsplannere57d0b5cPostConfirmation
- Runtime: nodejs

Resource access permission
- Not configured

Scheduled recurring invocation
- Not configured

Lambda layers
- Not configured

Environment variables:
- Not configured

Secrets configuration
- Not configured

? Which setting do you want to update? Resource access permissions
? Select the categories you want this function to have access to. api
? Select the operations you want to permit on amplifytripsplanner Query, Mutation, S
ubscription

You can access the following resource attributes as environment variables from your Lambda function
        API_AMPLIFYTRIPSPLANNER_GRAPHQLAPIENDPOINTOUTPUT
        API_AMPLIFYTRIPSPLANNER_GRAPHQLAPIIDOUTPUT
? Do you want to edit the local lambda function now? No
```

```
? Which setting do you want to update? Resource access permissions
? Select the categories you want this function to have access to. api
? Select the operations you want to permit on amplifytripsplanner Query, Mutation, S
ubscription

You can access the following resource attributes as environment variables from your
 Lambda function
        API_AMPLIFYTRIPSPLANNER_GRAPHQLAPIENDPOINTOUTPUT
        API_AMPLIFYTRIPSPLANNER_GRAPHQLAPIIDOUTPUT
? Do you want to edit the local lambda function now? No
```

9. Run the command **amplify push** to create the resources in the cloud.

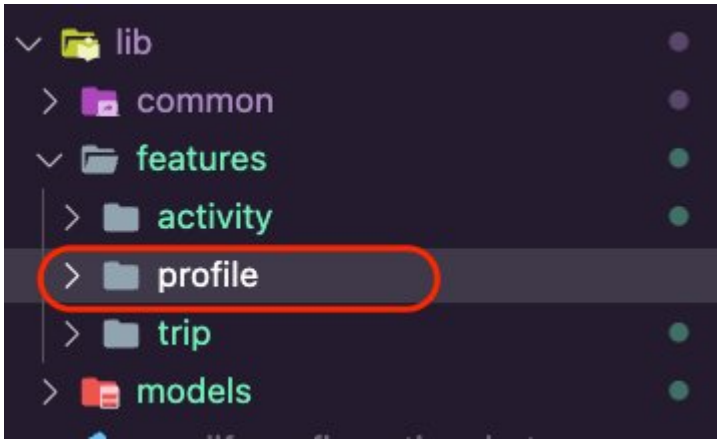
```
   Current Environment: dev

| Category | Resource name                                | Operation | Provider plugin     |
| Function | amplifytripsplanner1ec293ffPostConfirmation | Create    | awscloudformation   |
| Auth     | amplifytripsplanner1ec293ff                  | Update    | awscloudformation   |
| Api      | amplifytripsplanner                          | No Change | awscloudformation   |
| Storage  | s3b101c2c7                                   | No Change | awscloudformation   |

? Are you sure you want to continue? (Y/n) ›
```

10. Press **Enter**. The Amplify CLI will deploy the resources and display a confirmation, as shown in the screenshot.

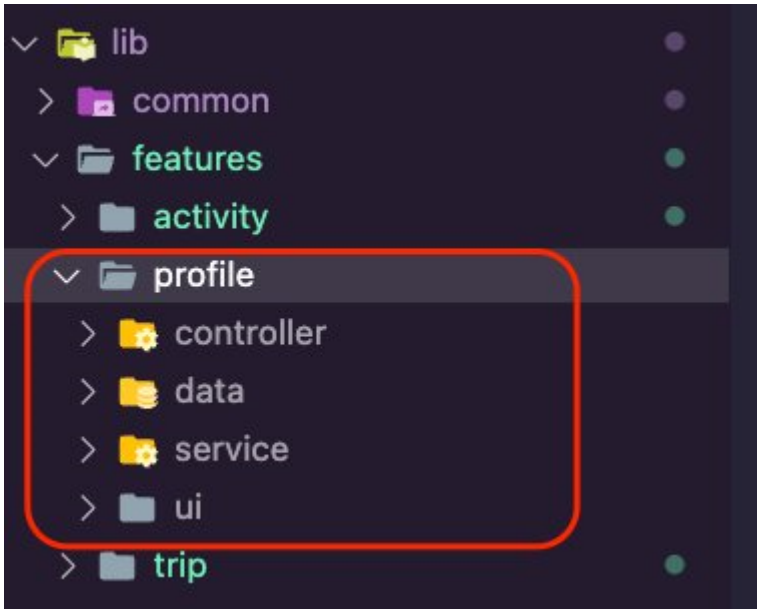
```
        amplify-amplifytripsplanner-d… AWS::CloudFormation::Stack     UPDATE_COMPLETE_CLEANUP_IN_PR
        storages3b101c2c7              AWS::CloudFormation::Stack     UPDATE_COMPLETE
        authamplifytripsplanner1ec293… AWS::CloudFormation::Stack     UPDATE_COMPLETE
        apiamplifytripsplanner         AWS::CloudFormation::Stack     UPDATE_COMPLETE
        functionamplifytripsplanner1e… AWS::CloudFormation::Stack     CREATE_COMPLETE
Deployed auth amplifytripsplanner1ec293ff [ ======================================== ] 10/10
        IdentityPool                   AWS::Cognito::IdentityPool     UPDATE_COMPLETE
Deployed function amplifytripsplanner1ec293ffPostConfirmation [ ==============================
        LambdaExecutionRole            AWS::IAM::Role                 CREATE_COMPLETE
        AmplifyResourcesPolicy         AWS::IAM::Policy               CREATE_IN_PROGRESS

Deployment state saved successfully.

GraphQL transformer version: 2
```

Create the folders for the Profile

1. Create a new folder inside **lib/features**, and name it **profile**.



2. Create the following new folders inside the **profile** folder:
   - **service:** The layer to connect with the Amplify backend.
   - **data:** This will be the repository layer that abstracts away the networking code, specifically **service** .
   - **controller:** This is an abstract layer to connect the UI with the repository.
   - **ui:** Here, we will create the widgets and the pages that the app will present to the user.

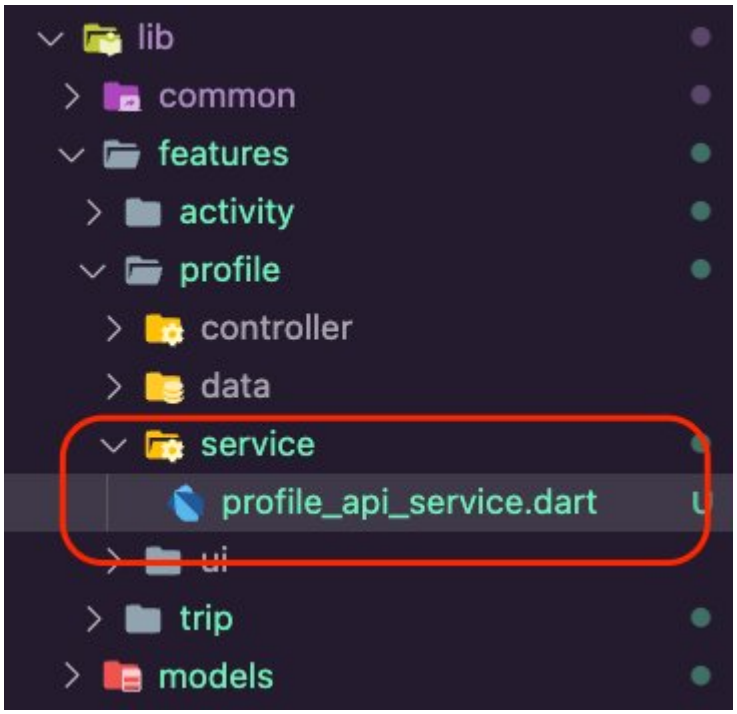


3. Open the file **lib/common/navigation/router/routes.dart**. Update it to add the enum values for the profile feature. The file **routes.dart** should look like the following:

```
enum AppRoute {
```

```
 home,
 trip,
 editTrip,
 pastTrips,
 pastTrip,
 activity,
 addActivity,
 editActivity,
 profile,
}
```

4. Create a new dart file inside the **lib/features/profile/service** folder and call it **profile_api_service.dart**.

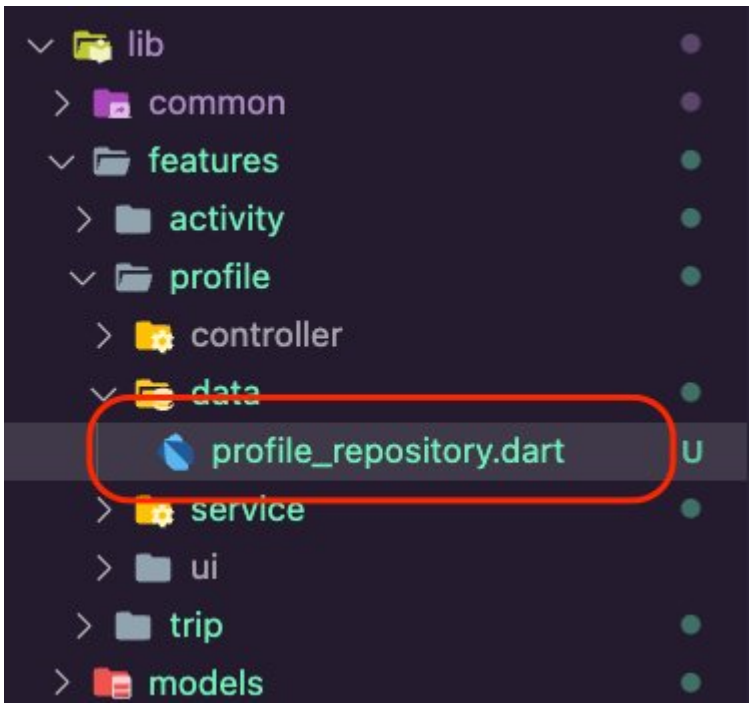


5. Open the **profile_api_service.dart** file and update it with the following code snippet to create **ProfileAPIService**, which contains the following functions:
   - **getProfile:** Queries the Amplify API for the user's profile and returns its details.
   - **updateProfile:** Update the user's profile using the Amplify API.

```
import 'package:amplify_api/amplify_api.dart';
import 'package:amplify_flutter/amplify_flutter.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

final profileAPIServiceProvider = Provider<ProfileAPIService>((ref) {
```

```
  return ProfileAPIService();
});

class ProfileAPIService {
  ProfileAPIService();

  Future<Profile> getProfile() async {
    try {
      final request = ModelQueries.list(Profile.classType);
      final response = await Amplify.API.query(request: request).response;

      final profile = response.data!.items.first;

      return profile!;
    } on Exception catch (error) {
      safePrint('getProfile failed: $error');
      rethrow;
    }
  }

  Future<void> updateProfile(Profile updatedProfile) async {
    try {
      await Amplify.API
          .mutate(
            request: ModelMutations.update(updatedProfile),
          )
          .response;
    } on Exception catch (error) {
      safePrint('updateProfile failed: $error');
    }
  }
}
```

6. Create a new dart file inside the **lib/features/profile/data** folder and call it **profile_repository.dart**.

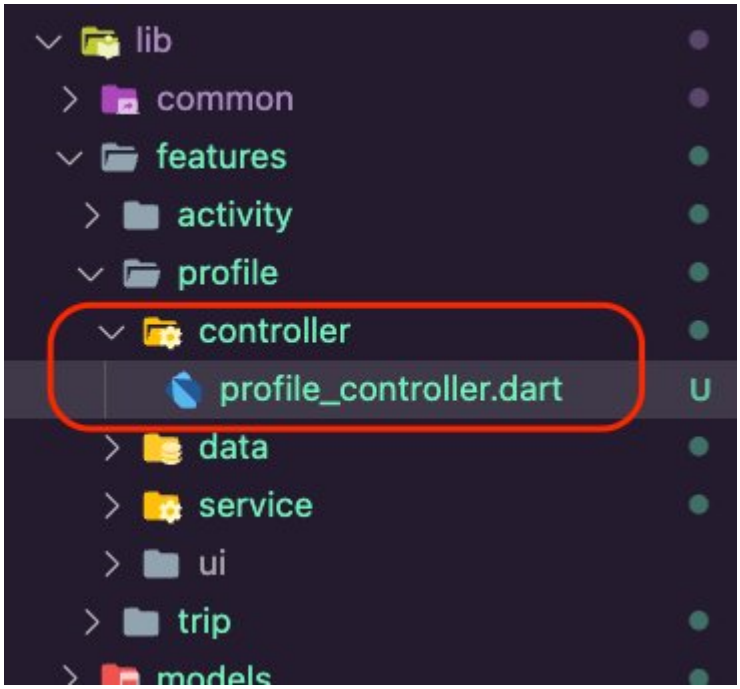


7. Open the **profile_repository.dart** file and update it with the following code:

```
import 'package:amplify_trips_planner/features/profile/service/
profile_api_service.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

final profileRepositoryProvider = Provider<ProfileRepository>((ref) {
 final profileAPIService = ref.read(profileAPIServiceProvider);
 return ProfileRepository(profileAPIService);
});

class ProfileRepository {
 ProfileRepository(this.profileAPIService);
 final ProfileAPIService profileAPIService;

 Future<Profile> getProfile() {
   return profileAPIService.getProfile();
 }

 Future<void> update(Profile updatedProfile) async {
   return profileAPIService.updateProfile(updatedProfile);
 }
}
```

Implement the UI for displaying and updating the profile

1. Create a new dart file inside the folder **lib/features/profile/controller** and name it **profile_controller.dart/**



2. Open the **profile_controller.dart** file and update it with the following code. The UI will use the controller to edit the details of the profile.

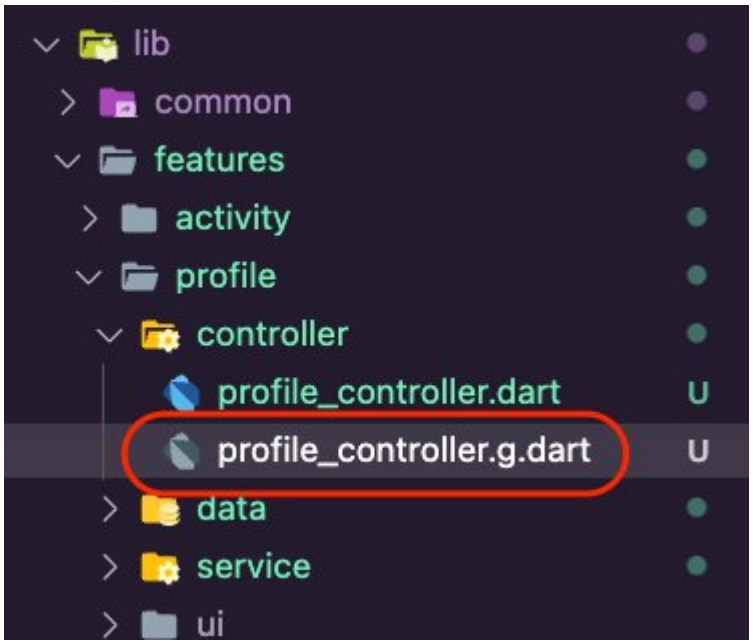
> **Note**
>
> VSCode will show errors due to the missing **profile_controller.g.dart** file. You will fix that in the next step.

```
import 'package:amplify_trips_planner/features/profile/data/profile_repository.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:riverpod_annotation/riverpod_annotation.dart';

part 'profile_controller.g.dart';

@riverpod
class ProfileController extends _$ProfileController {
 Future<Profile> _fetchProfile() async {
   final profileRepository = ref.read(profileRepositoryProvider);
```

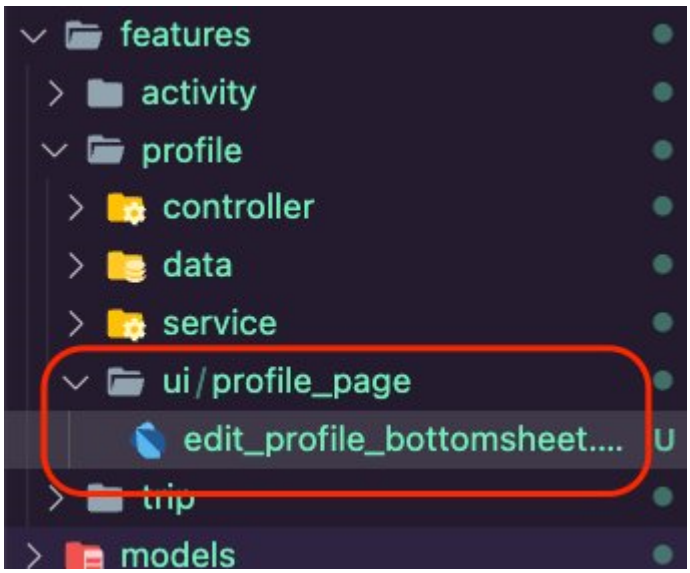
```
    return profileRepository.getProfile();
  }

  @override
  FutureOr<Profile> build() async {
    return _fetchProfile();
  }

  Future<void> updateProfile(Profile profile) async {
    state = const AsyncValue.loading();
    state = await AsyncValue.guard(() async {
      final profileRepository = ref.read(profileRepositoryProvider);
      await profileRepository.update(profile);
      return _fetchProfile();
    });
  }
}
```

3. Create a new folder inside the **lib/features/profile/ui folder**, name it **profile_page**, and then create the file **edit_profile_bottomsheet.dart** inside it.

```
dart run build_runner build -d
```



4. Create a new folder inside the **lib/features/profile/ui** folder, name it **profile_page**, and then create the file **edit_profile_bottomsheet.dart** inside it.



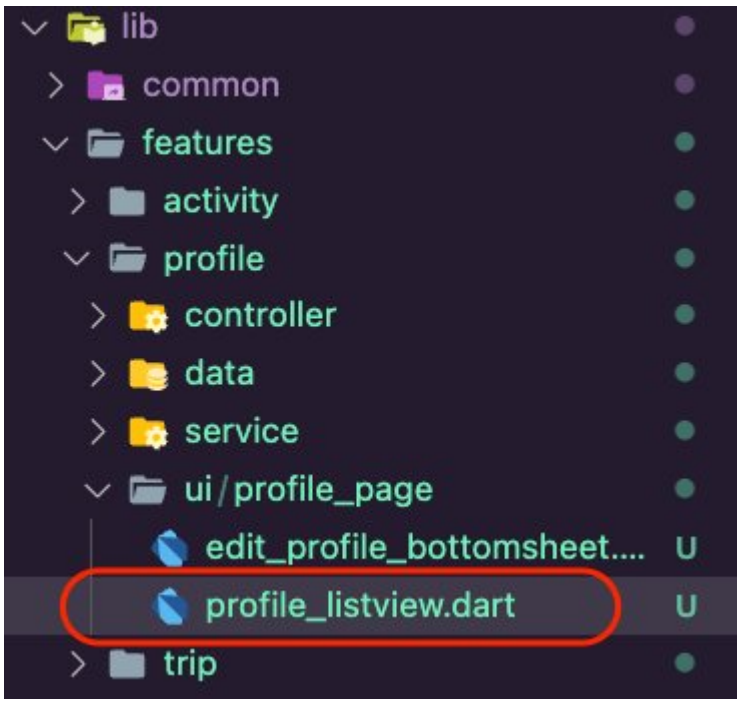
5. Open the **edit_profile_bottomsheet.dart** file and update it with the following code. This will allow us to present a form to the user to submit the required details to update the user's profile.

```
import 'package:amplify_trips_planner/common/ui/bottomsheet_text_form_field.dart';
import 'package:amplify_trips_planner/features/profile/controller/
profile_controller.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class EditProfileBottomSheet extends ConsumerWidget {
  EditProfileBottomSheet({
    required this.profile,
    super.key,
  });

  final Profile profile;

  final formGlobalKey = GlobalKey<FormState>();

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    final firstNameController = TextEditingController(
      text: profile.firstName != null ? profile.firstName! : '',
    );
    final lastNameController = TextEditingController(
      text: profile.lastName != null ? profile.lastName! : '',
```

```
    );
    final homeCityController = TextEditingController(
      text: profile.homeCity != null ? profile.homeCity! : '',
    );

    return Form(
      key: formGlobalKey,
      child: Container(
        padding: EdgeInsets.only(
          top: 15,
          left: 15,
          right: 15,
          bottom: MediaQuery.of(context).viewInsets.bottom + 15,
        ),
        width: double.infinity,
        child: Column(
          mainAxisSize: MainAxisSize.min,
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            BottomSheetTextFormField(
              labelText: 'First Name',
              controller: firstNameController,
              keyboardType: TextInputType.name,
            ),
            const SizedBox(
              height: 20,
            ),
            BottomSheetTextFormField(
              labelText: 'Last Name',
              controller: lastNameController,
              keyboardType: TextInputType.name,
            ),
            const SizedBox(
              height: 20,
            ),
            BottomSheetTextFormField(
              labelText: 'Home City',
              controller: homeCityController,
              keyboardType: TextInputType.name,
            ),
            const SizedBox(
              height: 20,
            ),
            TextButton(
```

```
              child: const Text('OK'),
              onPressed: () async {
                final currentState = formGlobalKey.currentState;
                if (currentState == null) {
                  return;
                }
                if (currentState.validate()) {
                  final updatedProfile = profile.copyWith(
                    firstName: firstNameController.text,
                    lastName: lastNameController.text,
                    homeCity: homeCityController.text,
                  );
                  await ref
                      .watch(profileControllerProvider.notifier)
                      .updateProfile(updatedProfile);

                  if (context.mounted) {
                    context.pop();
                  }
                }
              }, //,
            ),
          ],
        ),
      ),
    );
  }
}
```

6. Create a new dart file inside the folder **lib/features/profile/ui/profile_page** and name it **profile_listview.dart**.



7. Open the **profile_listview.dart** file and update it with the following code. This will display the user's profile details.

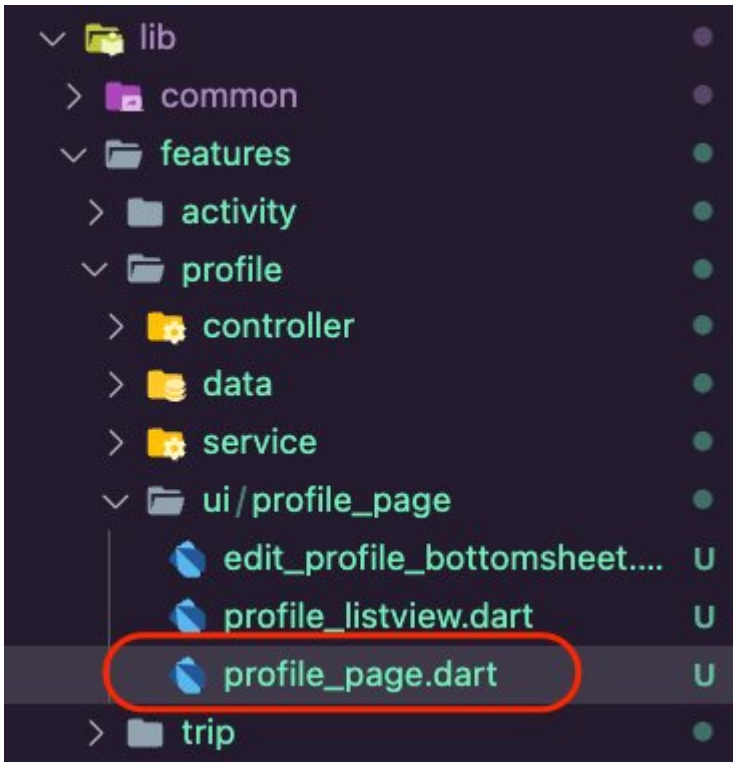
```
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/profile/controller/
profile_controller.dart';
import 'package:amplify_trips_planner/features/profile/ui/profile_page/
edit_profile_bottomsheet.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

class ProfileListView extends ConsumerWidget {
  const ProfileListView({
    required this.profile,
    super.key,
  });

  final AsyncValue<Profile> profile;

  void editProfile(BuildContext context, Profile profile) async {
    await showModalBottomSheet<void>(
      isScrollControlled: true,
      elevation: 5,
      context: context,
```

```
      builder: (BuildContext context) {
        return EditProfileBottomSheet(
          profile: profile,
        );
      },
    );
  }

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    switch (profile) {
      case AsyncData(:final value):
        return ListView(
          children: [
            Card(
              child: ListTile(
                leading: const Icon(
                  Icons.verified_user,
                  size: 50,
                  color: Color(constants.primaryColorDark),
                ),
                title: Text(
                  value.firstName != null ? value.firstName! : 'Add your name',
                  style: Theme.of(context).textTheme.titleLarge,
                ),
                subtitle: Text(value.email),
              ),
            ),
            ListTile(
              dense: true,
              title: Text(
                'Home',
                style: Theme.of(context)
                    .textTheme
                    .titleSmall!
                    .copyWith(color: Colors.white),
              ),
              tileColor: Colors.grey,
            ),
            Card(
              child: ListTile(
                title: Text(
                  value.firstName != null ? value.homeCity! : 'Add your city',
                  style: Theme.of(context).textTheme.titleLarge,
```

```
              ),
            ),
          ),
          const ListTile(
            dense: true,
            tileColor: Colors.grey,
            visualDensity: VisualDensity(vertical: -4),
          ),
          Card(
            child: Row(
              mainAxisAlignment: MainAxisAlignment.spaceAround,
              children: [
                TextButton(
                  style: TextButton.styleFrom(
                    textStyle: const TextStyle(fontSize: 20),
                  ),
                  onPressed: () {
                    editProfile(context, value);
                  },
                  child: const Text('Edit'),
                ),
              ],
            ),
          )
        ],
      );

    case AsyncError():
      return Column(
        mainAxisSize: MainAxisSize.min,
        crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          Padding(
            padding: const EdgeInsets.all(8),
            child: Text(
              'Error',
              style: Theme.of(context).textTheme.titleMedium,
              textAlign: TextAlign.center,
            ),
          ),
          TextButton(
            style: TextButton.styleFrom(
              textStyle: const TextStyle(fontSize: 20),
            ),
```

```
              onPressed: () async {
                ref.invalidate(profileControllerProvider);
              },
              child: const Text('Try again'),
            ),
          ],
        );
      case AsyncLoading():
        return const Center(
          child: CircularProgressIndicator(),
        );

      case _:
        return const Center(
          child: Text('Error'),
        );
    }
  }
}
```

8. Create a new dart file inside the folder **lib/features/profile/ui/profile_page** and name it **profile_page.dart**.



9. Open the **profile_page.dart** file and update it with the following code to use the**ProfileListView** you created above to display the user's profile details.

```
import 'package:amplify_trips_planner/common/ui/the_navigation_drawer.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:amplify_trips_planner/features/profile/controller/
profile_controller.dart';
import 'package:amplify_trips_planner/features/profile/ui/profile_page/
profile_listview.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';

class ProfilePage extends ConsumerWidget {
  const ProfilePage({
    super.key,
  });

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    final profileValue = ref.watch(profileControllerProvider);
    return Scaffold(
      appBar: AppBar(
        centerTitle: true,
        title: const Text(
          'Amplify Trips Planner',
        ),
        backgroundColor: const Color(constants.primaryColorDark),
      ),
      drawer: const TheNavigationDrawer(),
      body: ProfileListView(
        profile: profileValue,
      ),
    );
  }
}
```

10: Open the **lib/common/navigation/router/router.dart** file and update it to add the **ProfilePage** route.

```
    GoRoute(
      path: '/profile',
      name: AppRoute.profile.name,
      builder: (context, state) => const ProfilePage(),
    ),
```

The **router.dart** should look like the following code snippet.

```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/features/activity/ui/activity_page/
activity_page.dart';
import 'package:amplify_trips_planner/features/activity/ui/add_activity/
add_activity_page.dart';
import 'package:amplify_trips_planner/features/activity/ui/edit_activity/
edit_activity_page.dart';
import 'package:amplify_trips_planner/features/profile/ui/profile_page/
profile_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/edit_trip_page/
edit_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trip_page/
past_trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/past_trips/
past_trips_list.dart';
import 'package:amplify_trips_planner/features/trip/ui/trip_page/trip_page.dart';
import 'package:amplify_trips_planner/features/trip/ui/trips_list/
trips_list_page.dart';
import 'package:amplify_trips_planner/models/ModelProvider.dart';
import 'package:flutter/material.dart';
import 'package:go_router/go_router.dart';

final router = GoRouter(
  routes: [
    GoRoute(
      path: '/',
      name: AppRoute.home.name,
      builder: (context, state) => const TripsListPage(),
    ),
    GoRoute(
      path: '/trip/:id',
      name: AppRoute.trip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return TripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/edittrip/:id',
      name: AppRoute.editTrip.name,
      builder: (context, state) {
```

```
        return EditTripPage(
          trip: state.extra! as Trip,
        );
      },
    ),
    GoRoute(
      path: '/pasttrip/:id',
      name: AppRoute.pastTrip.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return PastTripPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/pasttrips',
      name: AppRoute.pastTrips.name,
      builder: (context, state) => const PastTripsList(),
    ),
    GoRoute(
      path: '/addActivity/:id',
      name: AppRoute.addActivity.name,
      builder: (context, state) {
        final tripId = state.pathParameters['id']!;
        return AddActivityPage(tripId: tripId);
      },
    ),
    GoRoute(
      path: '/activity/:id',
      name: AppRoute.activity.name,
      builder: (context, state) {
        final activityId = state.pathParameters['id']!;
        return ActivityPage(activityId: activityId);
      },
    ),
    GoRoute(
      path: '/editactivity/:id',
      name: AppRoute.editActivity.name,
      builder: (context, state) {
        return EditActivityPage(
          activity: state.extra! as Activity,
        );
      },
    ),
    GoRoute(
```

```
      path: '/profile',
      name: AppRoute.profile.name,
      builder: (context, state) => const ProfilePage(),
    ),
  ],
  errorBuilder: (context, state) => Scaffold(
    body: Center(
      child: Text(state.error.toString()),
    ),
  ),
);
```

11. Open the file **lib/common/navigation/ui/the_navigation_drawer.dart**. Update it with the following code to add the option to navigate to the profile route.

```
          ListTile(
            leading: const Icon(Icons.settings),
            title: const Text('Settings'),
            onTap: () {
              context.goNamed(
                AppRoute.profile.name,
              );
            },
          ),
```

The **the_navigation_drawer.dart** should look like the following code snippet.

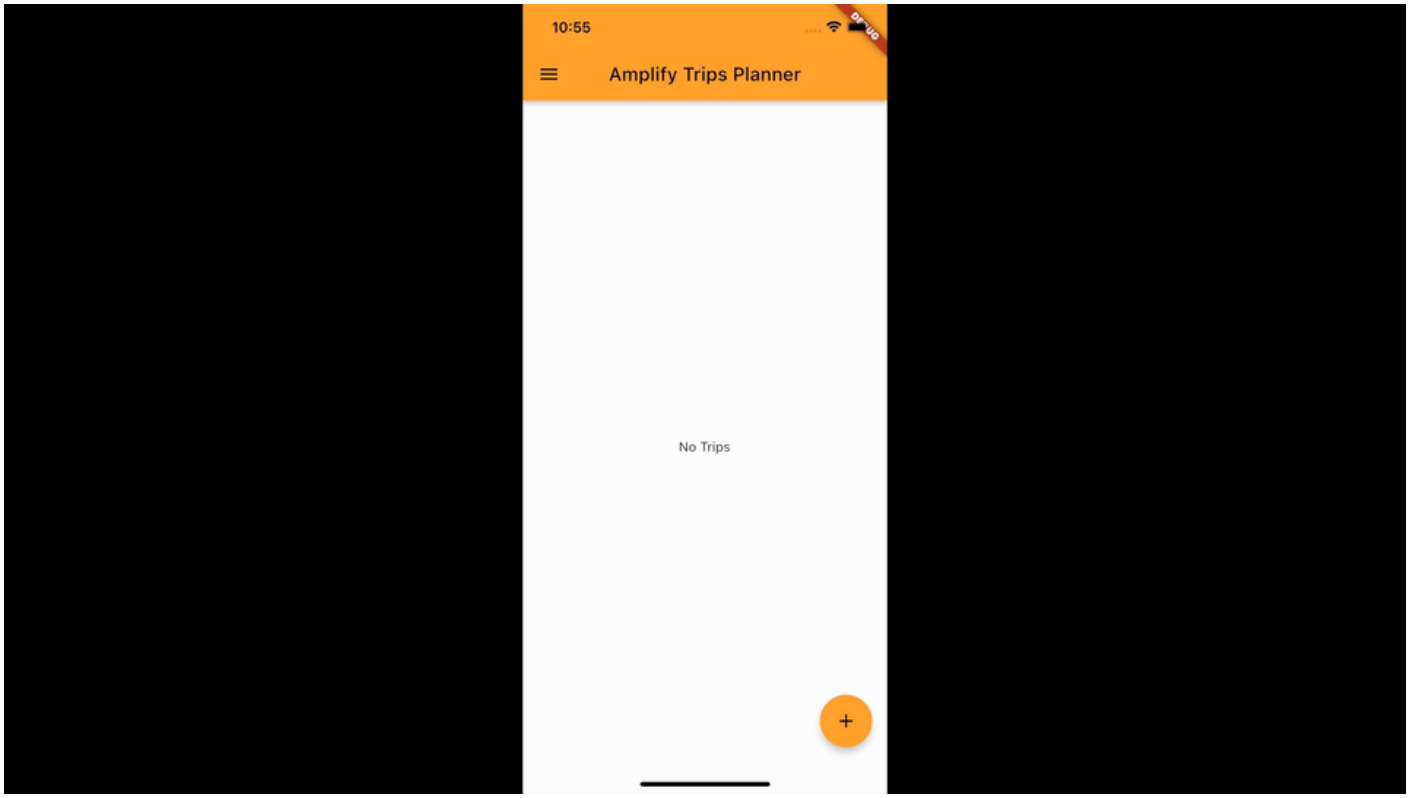
```
import 'package:amplify_trips_planner/common/navigation/router/routes.dart';
import 'package:amplify_trips_planner/common/services/auth_service.dart';
import 'package:amplify_trips_planner/common/utils/colors.dart' as constants;
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:go_router/go_router.dart';

class TheNavigationDrawer extends ConsumerWidget {
  const TheNavigationDrawer({
    super.key,
  });

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    return Drawer(
      child: ListView(
```

```
        padding: EdgeInsets.zero,
        children: [
          const DrawerHeader(
            decoration: BoxDecoration(
              color: Color(constants.primaryColorDark),
            ),
            padding: EdgeInsets.all(16),
            child: Column(
              children: [
                SizedBox(height: 10),
                Text(
                  'Amplify Trips Planner',
                  style: TextStyle(fontSize: 22, color: Colors.white),
                ),
              ],
            ),
          ),
          ListTile(
            leading: const Icon(Icons.home),
            title: const Text('Trips'),
            onTap: () {
              context.goNamed(
                AppRoute.home.name,
              );
            },
          ),
          ListTile(
            leading: const Icon(Icons.category),
            title: const Text('Past Trips'),
            onTap: () {
              context.goNamed(
                AppRoute.pastTrips.name,
              );
            },
          ),
          ListTile(
            leading: const Icon(Icons.settings),
            title: const Text('Settings'),
            onTap: () {
              context.goNamed(
                AppRoute.profile.name,
              );
            },
          ),
```

```
          ListTile(
            leading: const Icon(Icons.exit_to_app),
            title: const Text('Logout'),
            onTap: () => ref.read(authServiceProvider).signOut(),
          ),
        ],
      ),
    );
  }
}
```

12 Run the app in an emulator or simulator and create a new user, navigate to the settings screen and update the user profile. The following is an example using an iPhone simulator.

> **Note**
>
> - Due to the changes in the data schema, you need to erase the app and its contents from the emulator or simulator.
> - If you encounter an error on the settings page, it could be because the VTL resolvers were not updated properly. To resolve the issue, add an empty line to the **schema.graphql** file and run the **amplify push** command.



Conclusion

In this module, you introduced the feature of displaying and editing the user's profile. You used an Amplify function to create the user's profile. You also implemented the UI for displaying the profile details and updating it.

Congratulations!

Congratulations! You have created a cross-platform Flutter mobile app using AWS Amplify! You have cloned the app you created in the first tutorial in this series. You then introduced the ability to display the user's past trips. Additionally, you updated the Amplify API, allowing users to create, read, update, and delete their trip's activities. You have also added an Amplify function to create a user profile, then you implemented the UI to allow the user to update their profile details.

Clean up resources

Now that you've finished this walkthrough, you can delete the backend resources to avoid incurring unexpected costs by running the command below in the root folder of the app.

```
amplify delete
```