#6 Create the Home Page

Introduction

The home page is the first thing users see when they visit an app. Therefore, it is essential to make a good first impression and provide a clear indication of what the app is all about. This section will guide you through creating the home page for the educative.io clone with Flutter.

Let's get started! 🚀

Let’s Get Started

Since we are using Riverpod as the state management for this project, we need to install the **Flutter Riverpod Snippets** extension in VS Code to facilitate the development process using Riverpod state management. You can follow the video below for the installation process!

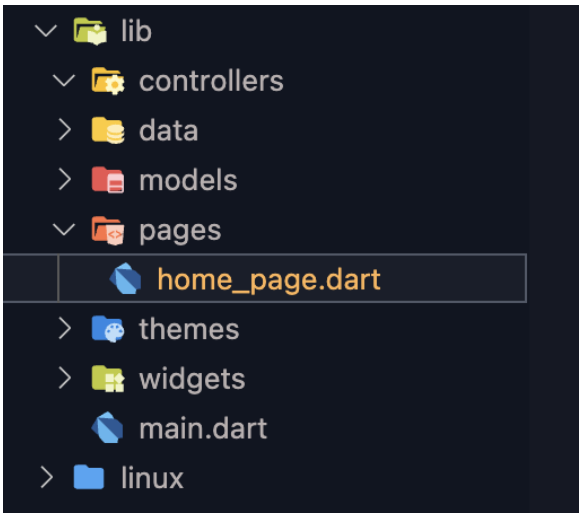
https://www.loom.com/share/44bff4693cf043c698366e08c141a8d3

Step-by-step guide

Okay, if already installed the Flutter Riverpod Snippets extension. Now, let’s get started to create a Home Page step by step.

Step 1 - Create home_page.dart file

Inside `lib/pages` create a new file called `home_page.dart`.



Step 2 - Create a ConsumerWidget

`ConsumerWidget` is a widget provided by the Flutter Riverpod package to help manage the state of an application. It is similar to `StatefulWidget` but has the added benefit of being able to rebuild only when the state it is watching has changed, increasing the efficiency of the application. It is used to connect a widget to the state management system and render the UI based on the current state of the application.

After creating the `home_page.dart` file, the next step is to create a `ConsumerWidget` that will represent our home page. We will name it `HomePage`.

The `HomePage` widget will return a `Scaffold` widget, which provides us with a framework for implementing the basic material design visual layout structure of the app.

Inside the `Scaffold` widget, we will call two parameters: `appBar` and `body`.

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

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

  @override
```

```
  Widget build(BuildContext context, WidgetRef ref) {
    return Scaffold(
      appBar: ,
      body: ,
    );
  }
}
```

Step 3 - Add the AppBar widget

The `appBar` of the `Scaffold` contains a logo, app name, and icons for search and user accounts.

Add the following code snippet into the appBar parameters:

```
      AppBar(
        backgroundColor: Colors.white,
        automaticallyImplyLeading: false,
        title: Row(
          children: [
            SvgPicture.asset(
              'assets/logo.svg',
              width: 24,
              color: MyColors.primary,
            ),
            const SizedBox(width: 5),
            Text(
              'educative',
              style: GoogleFonts.nunitoSans(
                color: MyColors.black,
                fontWeight: FontWeight.w500,
              ),
            )
          ],
        ),
        actions: [
          IconButton(
            icon: const Icon(Icons.search),
            color: Colors.grey,
            onPressed: () {},
          ),
          IconButton(
            padding: const EdgeInsets.only(right: 10),
            icon: const Icon(Icons.account_circle),
            color: Colors.grey,
            iconSize: 30,
            onPressed: () {},
          ),
        ],
      ),
```

Populate the `home_page.dart` file so far should look something like this:

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

import '../themes/colors.dart';

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

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Colors.white,
        automaticallyImplyLeading: false,
        title: Row(
          children: [
            SvgPicture.asset(
              'assets/logo.svg',
              width: 24,
              color: MyColors.primary,
            ),
            const SizedBox(width: 5),
            Text(
              'educative',
              style: GoogleFonts.nunitoSans(
                color: MyColors.black,
                fontWeight: FontWeight.w500,
              ),
            )
          ],
        ),
        actions: [
          IconButton(
            icon: const Icon(Icons.search),
            color: Colors.grey,
            onPressed: () {},
          ),
          IconButton(
            padding: const EdgeInsets.only(right: 10),
            icon: const Icon(Icons.account_circle),
            color: Colors.grey,
            iconSize: 30,
            onPressed: () {},
          ),
```

```
        ],
      ),
      body: ,
    );
  }
}
```

Step 4 - Add the Body

The `body` of the `Scaffold` contains the main content of the home page.

We will use a `ListView.separated` widget to display a list of courses. This widget allows us to create a list that can be scrolled vertically and separated by a divider between each item.

Add the following code snippet into the `body` parameters:

```
ListView.separated(
  itemCount: courses.length,
  padding: const EdgeInsets.only(
    top: 20,
    left: 20,
    right: 20,
    bottom: 30,
  ),
  itemBuilder: (context, index) {
    return CourseCard(
      courses[index],
      onPressed: () {
        // TODO: Navigate to CourseDetailPage
      },
    );
  },
  separatorBuilder: (context, index) {
    return const SizedBox(height: 10);
  },
);
```

`courses` is a list of course objects that we will create later in the tutorial. `CourseCard` is a custom widget that we will define in a later step.

Step 5 - Create the CourseCard widget

To create the `CourseCard` widget, we need to create a new file called `course_card.dart` inside the `lib/widgets` directory.

After that, we can copy and paste the code snippet provided below to create the `CourseCard` widget. This widget is a custom widget that displays the course title, author info, course thumbnail image, and course level.

Add the following code snippet to create the `CourseCard` widget in the `lib/widgets/course_card.dart` file:

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

import '../models/course.dart';
import '../themes/colors.dart';
import '../themes/typography.dart';
import 'level_indicator.dart';

class CourseCard extends ConsumerStatefulWidget {
  const CourseCard(
    this.course, {
    super.key,
    required this.onPressed,
  });
  final Course course;
  final VoidCallback onPressed;

  @override
  ConsumerState<ConsumerStatefulWidget> createState() => _CourseCardState();
}

class _CourseCardState extends ConsumerState<CourseCard> {
  @override
  Widget build(BuildContext context) {
    return GestureDetector(
      onTap: widget.onPressed,
      child: Card(
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(10),
        ),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            // Course Thumbnail
            ClipRRect(
              borderRadius:
                  const BorderRadius.vertical(top: Radius.circular(10)),
              child: Image.network(
                widget.course.imageUrl,
                height: MediaQuery.of(context).size.height * 0.2,
                width: double.infinity,
                fit: BoxFit.cover,
              ),
```

```
            ),
            Padding(
              padding: const EdgeInsets.symmetric(horizontal: 16, vertical: 16),
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  // Course Title
                  Text(
                    widget.course.title,
                    style: MyTypography.titleMedium,
                  ),
                  const SizedBox(height: 10),
                  // Course Author
                  Row(
                    children: [
                      ClipRRect(
                        borderRadius: BorderRadius.circular(10),
                        child: Image.network(
                          widget.course.authorAvatarUrl,
                          height: 30,
                          width: 30,
                        ),
                      ),
                      const SizedBox(width: 10),
                      Text(
                        'by ',
                        style: MyTypography.body,
                      ),
                      Text(
                        widget.course.authorName,
                        style: MyTypography.body,
                      ),
                    ],
                  ),
                  const SizedBox(height: 40),
                  Row(
                    children: [
                      // Course Level
                      Expanded(
                        child: Column(
                          crossAxisAlignment: CrossAxisAlignment.start,
                          children: [
                            LevelIndicator(level: widget.course.level),
                            const SizedBox(height: 10),
                            Text(
                              widget.course.level,
                              style: MyTypography.bodySmall,
                            ),
                          ],
                        ),
                      ),
                      // CTA Button - Get Started
                      OutlinedButton(
                        onPressed: widget.onPressed,
```

```
                        style: OutlinedButton.styleFrom(
                          foregroundColor: MyColors.primary,
                          side: const BorderSide(
                            color: Colors.grey,
                            width: 1,
                          ),
                          shape: RoundedRectangleBorder(
                            borderRadius: BorderRadius.circular(5),
                          ),
                        ),
                        child: Row(
                          mainAxisSize: MainAxisSize.min,
                          children: [
                            Text(
                              'Get Started',
                              style: MyTypography.body,
                            ),
                            const SizedBox(width: 5),
                            Icon(
                              Icons.arrow_forward_rounded,
                              size: 20,
                              color: MyColors.primary,
                            ),
                          ],
                        ),
                      ),
                    ],
                  )
                ],
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

The `CourseCard` widget requires the `Course` object and a callback function that is called when the user taps on the card. We can use `GestureDetector` to handle the tap event.

The `CourseCard` widget is built using a `Card` widget that contains a `Column` widget. The `Column` widget contains a `ClipRRect` widget to display the course thumbnail, `Text` widgets to display the course title and author info, `LevelIndicator` widget to display the course level, and an `OutlinedButton` widget that is used as the CTA button.

After creating the `CourseCard` widget, we can use it on the `home_page.dart` file inside the `ListView.separated` widget to display a list of courses.

Step 6 - Create the LevelIndicator widget

To create the `LevelIndicator` widget, we need to create a new file called `level_indicator.dart` inside the `lib/widgets` directory.

After that, we can copy and paste the code snippet provided below to create the `LevelIndicator` widget. This widget is a custom widget that displays the course level as a colored dot.

Add the following code snippet to create the `LevelIndicator` widget in the `lib/widgets/level_indicator.dart` file:

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

import '../themes/colors.dart';

class LevelIndicator extends StatelessWidget {
  const LevelIndicator({super.key, required this.level});
  final String level;

  @override
  Widget build(BuildContext context) {
    int step = 0;
    switch (level) {
      case 'Beginner':
        step = 1;
        break;
      case 'Intermediate':
        step = 2;
        break;
      case 'Advanced':
        step = 3;
        break;
    }

    return SizedBox(
      height: 8,
      child: ListView.separated(
        physics: const NeverScrollableScrollPhysics(),
        itemCount: 3,
        scrollDirection: Axis.horizontal,
        itemBuilder: (context, index) {
          Color? color = MyColors.primary;
          if (step < index + 1) {
            color = Colors.grey[300];
          }
          return Container(
            height: 8,
            width: 8,
            decoration: BoxDecoration(
```

```
                color: color,
                borderRadius: BorderRadius.circular(5),
              ),
            );
          },
          separatorBuilder: (context, index) {
            Color? color = MyColors.primary;
            if (step < index + 1 || step == index + 1) {
              color = Colors.grey[300];
            }

            return UnconstrainedBox(
              child: Container(
                width: 5,
                height: 2,
                color: color,
              ),
            );
          },
        ),
      );
    }
  }
```

The `LevelIndicator` widget requires a `level` string that is used to determine the color of the dot. We can use a `Container` widget to create a circular shape and use `BoxDecoration` to set the color based on the `level` string.

After creating the `LevelIndicator` widget, make sure to import the `LevelIndicator` widget inside the `lib/widgets/course_card.dart` file.

So, be sure to add the line `import 'level_indicator.dart';` inside the `course_card.dart` file. This way, we can use the `LevelIndicator` widget and display the difficulty level of a course in the `CourseCard`.

Step 6 - Define the Course class

To define the `Course` class, we need to create a new file called `course.dart` inside the `lib/models` directory.

Course Class is a representation of a course table in the database that contains course-related information.

Add the following code snippet to define the `Course` class in the `lib/models/course.dart` file:

```
class Course {
  final String id;
  final String title;
  final String authorName;
  final String authorAvatarUrl;
  final String authorJob;
  final String imageUrl;
  final String description;
  final String level;
  final String duration;
  final String price;
  final String? status;

  Course({
    required this.id,
    required this.title,
    required this.authorName,
    required this.authorJob,
    required this.imageUrl,
    required this.description,
    required this.level,
    required this.duration,
    required this.price,
    required this.authorAvatarUrl,
    required this.status,
  });

  factory Course.fromJson(Map<String, dynamic> json) {
    return Course(
      id: json['id'],
      title: json['title'],
      authorName: json['author'],
      authorJob: json['author_job'],
      authorAvatarUrl: json['author_avatar_url'],
      imageUrl: json['image_url'],
      description: json['description'],
      level: json['level'],
      duration: json['duration'],
      price: json['price'],
      status: json['status'],
    );
  }
}
```

Step 7 - Define the courses list (dummy data)

To define the `courses` list, we need to create a new file called `dummy_data.dart` inside the `lib/data` directory. Inside this file, we can define a list of `Course` objects that will be used to populate the home page. Each `Course` object should have a unique `id`, `title`,

`authorName`, `authorAvatarUrl`, `authorJob`, `imageUrl`, `description`, `level`, `duration`, `price`, and `status` property. For now, we can use placeholder values for these properties.

Add the following code snippet to define the `courses` list in the `lib/data/dummy_data.dart` file:

```
import '../models/course.dart';

final List<Course> courses = [
  Course(
    id: 'c1',
    title: 'Build educative.io clone with Flutter and Supabase',
    authorName: 'John Doe',
    authorAvatarUrl: 'https://i.pravatar.cc/150?img',
    authorJob: 'Senior Software Engineer',
    imageUrl: 'https://www.educative.io/cdn-cgi/image/format=auto,width=950,quality=75/v2a
pi/collection/10370001/6069685319630848/image/6492564120141824',
    description: 'Learn how to build educative.io clone with Flutter and Supabase. This co
urse is free and open source. You can find the source code on GitHub.',
    level: 'Beginner',
    duration: '2 weeks',
    price: 'Free',
    status: null,
  ),
  Course(
    id: 'c2',
    title: 'Django Web Development',
    authorName: 'Jane Smith',
    authorAvatarUrl: 'https://i.pravatar.cc/150?img',
    authorJob: 'Software Engineer',
    imageUrl:
        'https://www.educative.io/v2api/collection/10370001/5551624074297344/image/5668113
229021184',
    description:
        'Learn how to build modern and responsive websites using HTML, CSS, and JavaScrip
t, and popular web development frameworks like React and Angular.',
    level: 'Intermediate',
    duration: '4 weeks',
    price: 'Paid',
    status: null,
  ),
  Course(
    id: 'c3',
    title: 'Machine Learning with Python',
    authorName: 'Bob Johnson',
    authorAvatarUrl: 'https://i.pravatar.cc/150?img',
    authorJob: 'Data Scientist',
    imageUrl:
        'https://www.educative.io/v2api/collection/10370001/6205771374133248/image/4589857
022672896',
    description:
```

```
        'Learn how to design beautiful and user-friendly mobile apps using Sketch, Figma,
 and other popular design tools.',
    level: 'Advanced',
    duration: '6 weeks',
    price: 'Paid',
    status: 'New',
  ),
];
```

After defining the `courses` list, we can use it on the `home_page.dart` file to display a list of courses using the `CourseCard` widget and `ListView.separated` widget.

To use the `courses` list on the `home_page.dart` file, we need to do the following:

1. Import the `Course` class and the `courses` list from their respective files.
2. Use the `courses` list as the data source for the `ListView.separated` widget.

Here's the updated code for the `home_page.dart` file:

```
...
import '../data/dummy_data.dart';   // Import courses list
...

class HomePage extends ConsumerWidget {
  @override
  Widget build(BuildContext context, WidgetRef ref) {
    return Scaffold(
      appBar: AppBar(
        ...
      ),
      body: ListView.separated(
        ...
        itemCount: courses.length,    // Use the courses list as the data source
        ...
        itemBuilder: (BuildContext context, int index) {
          return CourseCard(
            course: courses[index],   // Pass the course object to the CourseCard widget
            onPressed: () {
              // TODO: Navigate to CourseDetailPage
            },
          );
        },
      ),
    );
  }
}
```

In this updated code, we imported the `Course` class and the `courses` list from their respective files. We then used the `courses` list as the data source for the `ListView.separated` widget. For each course in the `courses` list, we created a `CourseCard` widget and passed the course object to it.

Step 8 - Apply HomePage widget on main.dart

Finally, apply the `HomePage` widget on `main.dart`, we need to set the `HomePage` widget as the value of the `home` property in the `MaterialApp` widget.

Here's the updated code for the `main.dart` file:

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

import 'pages/home_page.dart';

void main() async {
  await dotenv.load(fileName: ".env");

  final supabaseUrl = dotenv.get('SUPABASE_URL');
  final supabaseAnonKey = dotenv.get('SUPABASE_ANON_KEY');

  WidgetsFlutterBinding.ensureInitialized();

  await Supabase.initialize(
    url: supabaseUrl,
    anonKey: supabaseAnonKey,
  );

  runApp(
    const ProviderScope(
      child: MyApp(),
    ),
  );
}

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

  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Educative',
      home: HomePage(),      // <-- set this line
    );
```

```
  }
}
```

In this updated code, we set the `HomePage` widget as the value of the `home` property in the `MaterialApp` widget. When we run the app, the `HomePage` widget will be displayed first.

Full home_page.dart file

Here is the full code for the `HomePage` widget in the `home_page.dart` file:

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

import '../data/dummy_data.dart';
import '../themes/colors.dart';
import '../widgets/course_card.dart';

class HomePage extends ConsumerWidget {
  @override
  Widget build(BuildContext context, WidgetRef ref) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Colors.white,
        automaticallyImplyLeading: false,
        title: Row(
          children: [
            SvgPicture.asset(
              'assets/logo.svg',
              width: 24,
              color: MyColors.primary,
            ),
            const SizedBox(width: 5),
            Text(
              'educative',
              style: GoogleFonts.nunitoSans(
                color: MyColors.black,
                fontWeight: FontWeight.w500,
              ),
            )
          ],
        ),
        actions: [
          IconButton(
            icon: const Icon(Icons.search),
            color: Colors.grey,
            onPressed: () {},
          ),
```

```
          IconButton(
            padding: const EdgeInsets.only(right: 10),
            icon: const Icon(Icons.account_circle),
            color: Colors.grey,
            iconSize: 30,
            onPressed: () {},
          ),
        ],
      ),
      body: ListView.separated(
        padding: const EdgeInsets.only(
          top: 20,
          left: 20,
          right: 20,
          bottom: 30,
        ),
        itemCount: courses.length,   // Use the courses list as the data source
        separatorBuilder: (BuildContext context, int index) {
          return const SizedBox(height: 10);   // Add spacing between the CourseCards
        },
        itemBuilder: (BuildContext context, int index) {
          return CourseCard(
            course: courses[index],   // Pass the course object to the CourseCard widget
            onPressed: () {
              // TODO: Navigate to CourseDetailPage
            },
          );
        },
      ),
    );
  }
}
```

Testing the App

Let's test our app. Run the application on an emulator or actual device using the command in a terminal. The command `flutter run` will build the app and install it on your device.

https://www.loom.com/share/84578aefaf0d46969f15c58a6ec172fa

Conclusion

Congratulations! You have successfully created the home page for the educative.io Clone with Flutter. We have created a custom `CourseCard` widget, and a `LevelIndicator` widget, and defined the `Course` class and `courses` list (dummy data). We have also used the `ListView.separated` widget to display a list of courses on the home page.

In the next section, we will create the course details page. This page will provide all the relevant information about a selected course, including its details, information about the author, and the course curriculum. Let's get started! 🚀