

PMH 1 –

TP2

by Yassir Hoossan Buksh

Lecturer: Shiam Beeharry

Date: 19 Octobre 2023

Table of Contents

PMH 1 -.....	1
TP2.....	1
Task 2 -.....	3
Take your Flutter app from boring to beautiful.....	3
Making modifications to the icons on the left to better illustrate navigation.....	5
Changing the typography to better represent the application's feel.....	7
Applying a theme to the application.....	10

Task 2 -

Take your Flutter app from boring to beautiful

Swiftly starting things off I cloned the github repository in the third page of the codelabs guide.

```
Cloning into 'codelabs'...
remote: Enumerating objects: 30366, done.
remote: Counting objects: 100% (1649/1649), done.
remote: Compressing objects: 100% (748/748), done.
remote: Total 30366 (delta 1073), reused 1396 (delta 880), pack-reused 28717
Receiving objects: 100% (30366/30366), 93.28 MiB | 11.10 MiB/s
Receiving objects: 100% (30366/30366), 97.99 MiB | 10.81 MiB/s, done.
Resolving deltas: 100% (20674/20674), done.
Updating files: 100% (19603/19603), done.
```

Once that was done I ran the app as instructed

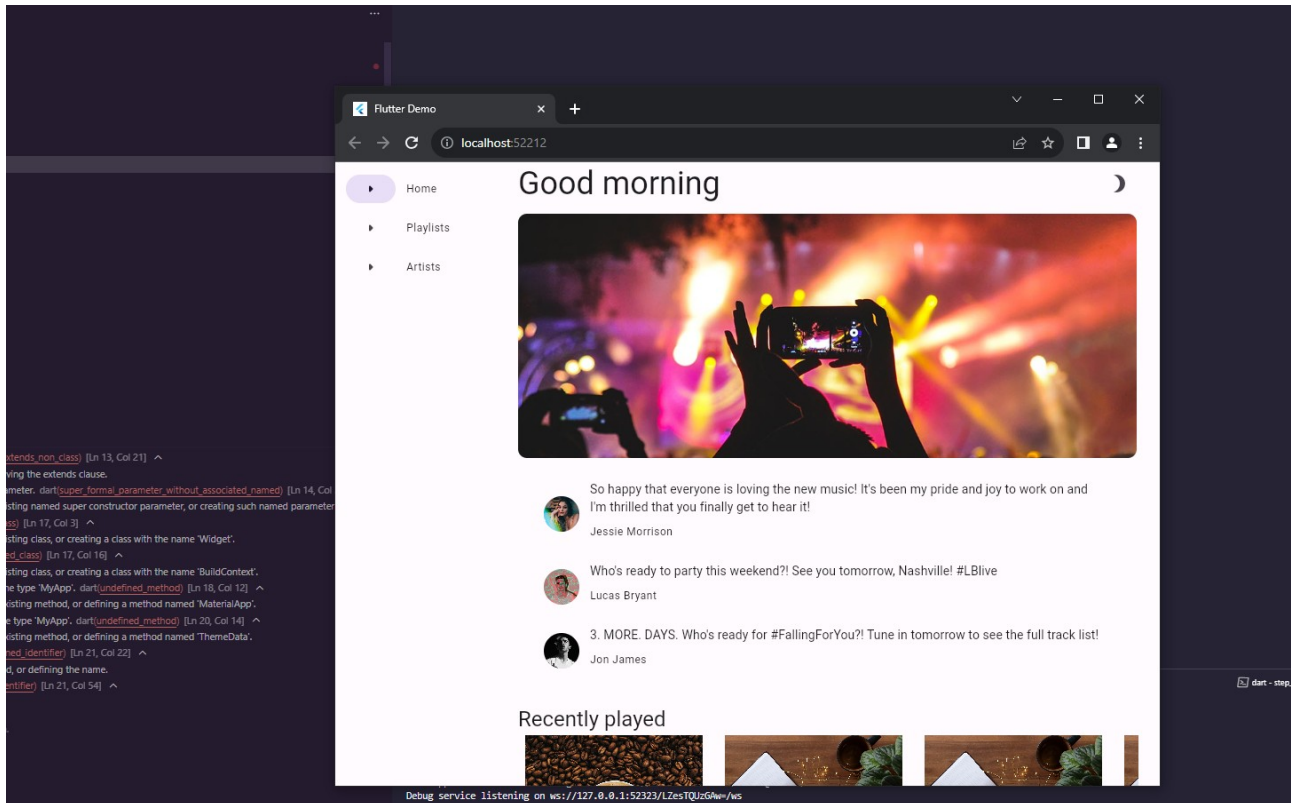
```
PS C:\Users\PCYas\Documents\GitHub\PMH\PMH_ShiamBeehary\TP2\Task1\codelabs\boring_to_beautiful\step_01> flutter run
Connected devices:
Windows (desktop) • windows • windows-x64 • Microsoft Windows [Version 10.0.19045.3803]
Chrome (web) • chrome • web-javascript • Google Chrome 119.0.6045.200
Edge (web) • edge • web-javascript • Microsoft Edge 120.0.2210.91
[1]: Windows (windows)
[2]: Chrome (chrome)
[3]: Edge (edge)
Please choose one (or "q" to quit): 2
Resolving dependencies...
```

I ran into a little problem since my dart sdk was outdated, but this was relatively quickly patched by running the "flutter upgrade" command

```
Resolving dependencies...
The current Dart SDK version is 3.1.4.

Because myartist requires SDK version ^3.2.0, version solving failed.
PS C:\Users\PCYas\Documents\GitHub\PMH\PMH_ShiamBeehary\TP2\Task1\codelabs\boring_to_beautiful\step_01> flutter upgrade
```

after a little bit of time the application boots up as expected and shown in the guide. I went with the chrome installation for no particular reason besides ease of deployment.



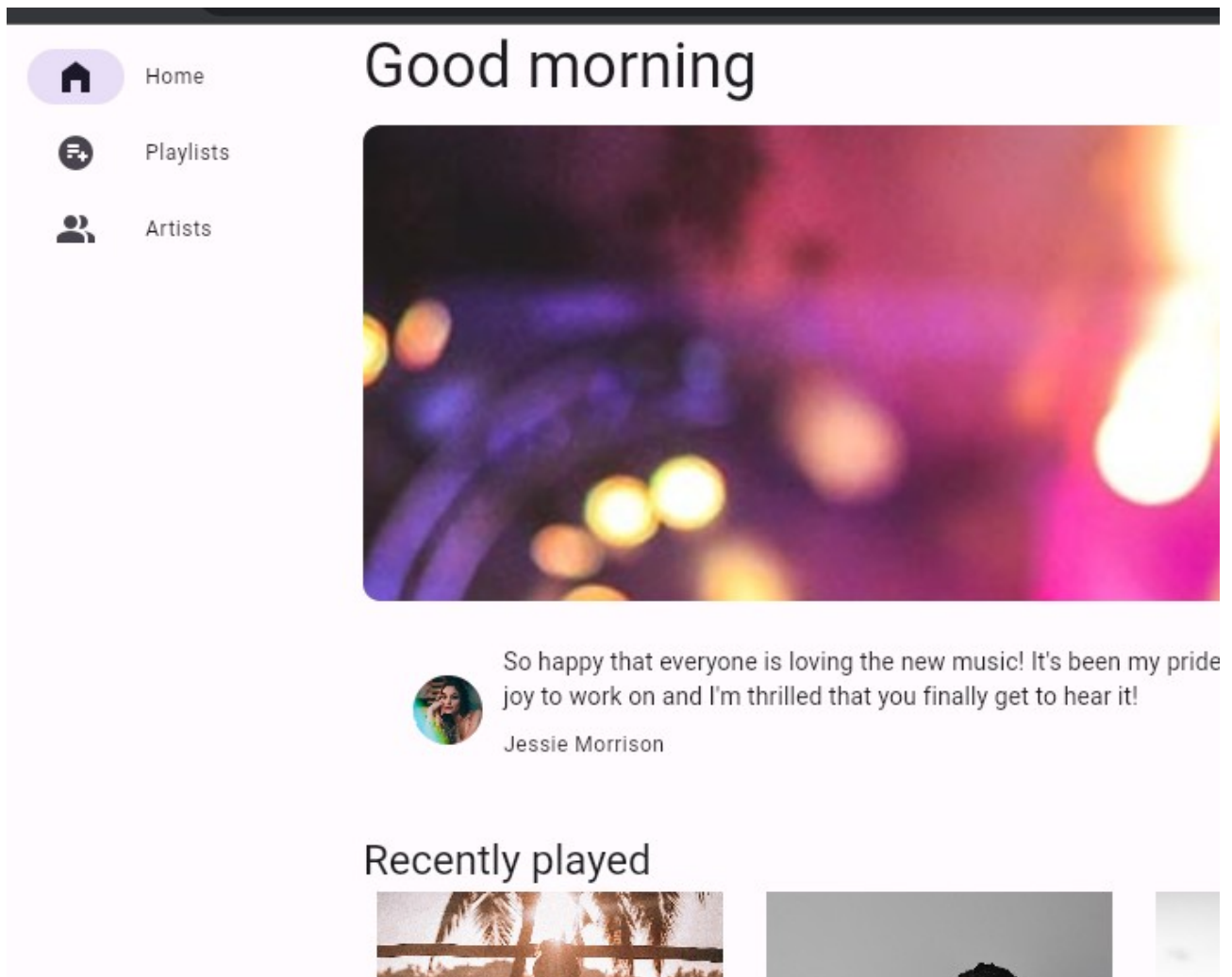
Making modifications to the icons on the left to better illustrate navigation.

```
22 const List<NavigationDestination> destinations = [  
23   NavigationDestination(  
24     Label: 'Home',  
25     icon: Icon(Icons.arrow_right_rounded), // Modify this line  
26     route: '/',  
27   ), // NavigationDestination  
28   NavigationDestination(  
29     Label: 'Playlists',  
30     icon: Icon(Icons.arrow_right_rounded), // Modify this line  
31     route: '/playlists',  
32   ), // NavigationDestination  
33   NavigationDestination(  
34     Label: 'Artists',  
35     icon: Icon(Icons.arrow_right_rounded), // Modify this line  
36     route: '/artists',  
37   ), // NavigationDestination  
38 ];
```

We'll be changing the lines to the appropriate vector icons to make the navigation easier:

```
✓ const List<NavigationDestination> destinations = [  
✓   NavigationDestination(  
     Label: 'Home',  
     icon: Icon(Icons.home_filled),  
     route: '/',  
   ), // NavigationDestination  
✓   NavigationDestination(  
     Label: 'Playlists',  
     icon: Icon(Icons.playlist_add_circle_rounded),  
     route: '/playlists',  
   ), // NavigationDestination  
✓   NavigationDestination(  
     Label: 'Artists',  
     icon: Icon(Icons.people_alt),  
     route: '/artists',  
   ), // NavigationDestination  
   ];
```

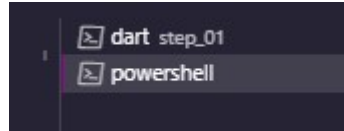
I decided to try out different icons to give the application a more interesting look



Changing the typography to better represent the application's feel.

Next we'll be implementing newer and better typography as recommended by the guide

instead of closing the app to do this I've opened a second powershell terminal.

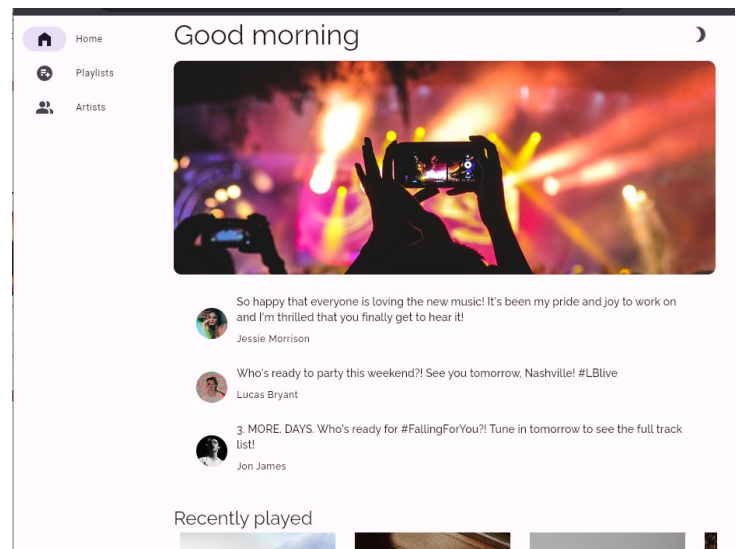
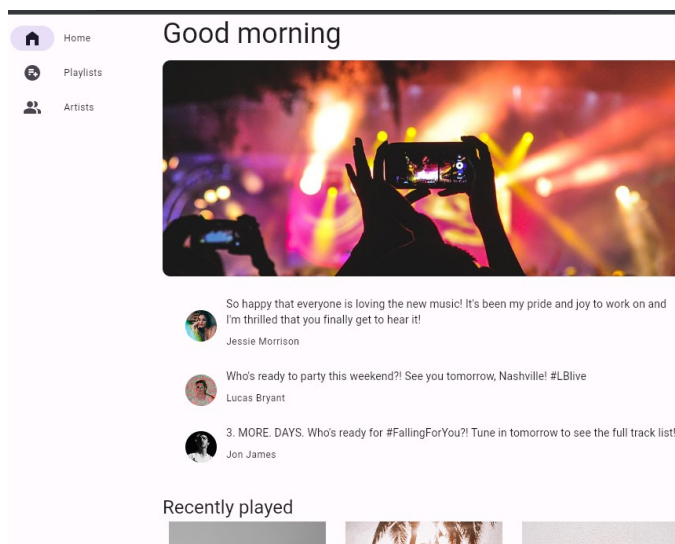


```
5 import 'package:flutter/material.dart';
6 import 'package:google_fonts/google_fonts.dart';
7 // Add Google Fonts Package import
8
9 extension TypographyUtils on BuildContext {
10   ThemeData get theme => Theme.of(this);
11   TextTheme get textTheme => theme.textTheme; // Modify this line
12   ColorScheme get colors => theme.colorScheme;
13   TextStyle get displayLarge => textTheme.displayLarge.copyWith(
```

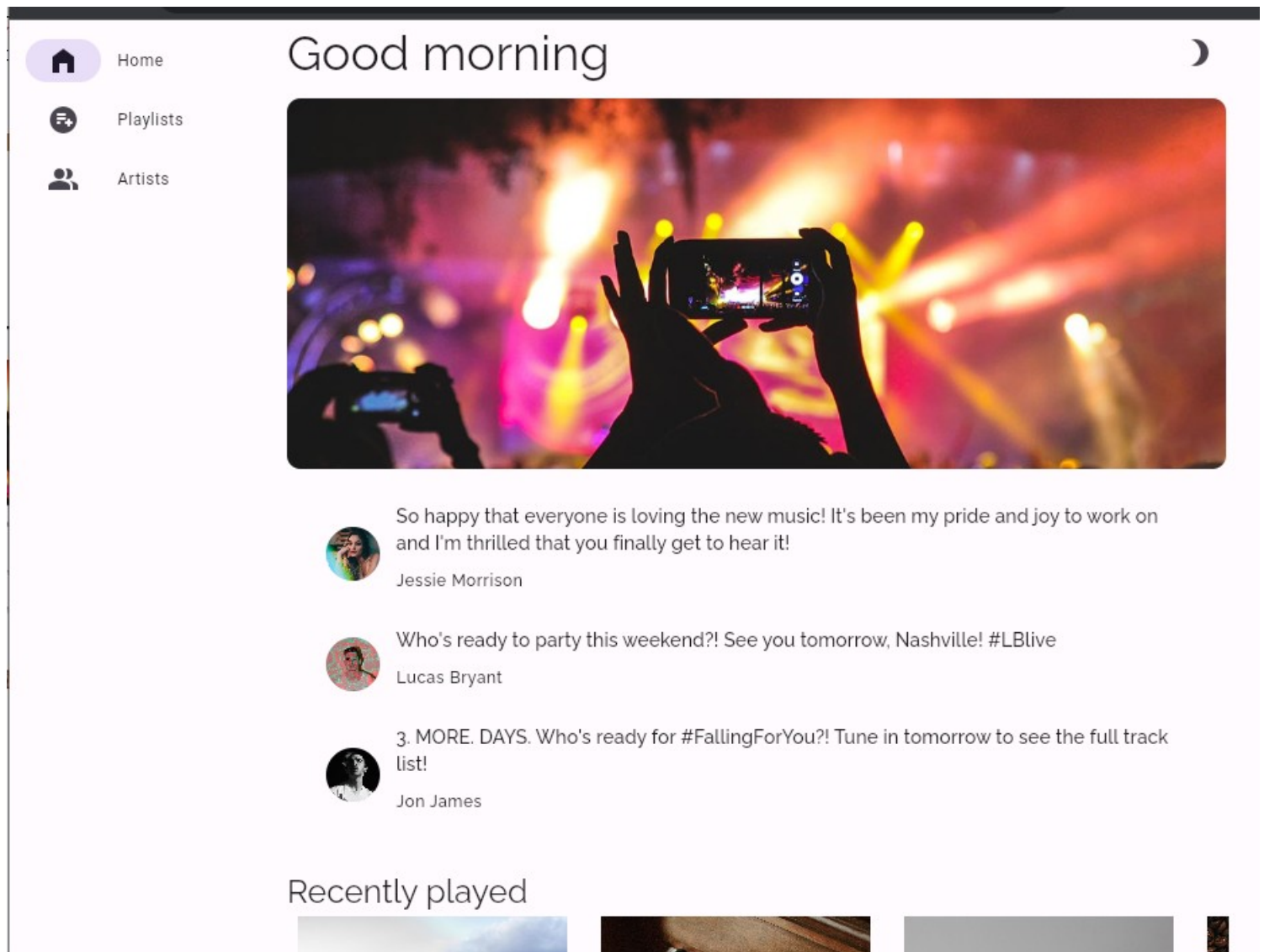
`TextTheme get textTheme => GoogleFonts.ralewayTextTheme(theme.textTheme);`

instead of using the suggested montserrat text theme I went with raleway, the procedure is the same since it proceeds with the same logic.

Now we can hot reload our application to see the changes;



Before & After



After result

Applying a theme to the application

Now we can apply a theme to the application to polish its look and finalize the aesthetic we're going for.

The guide implementation does this neatly by having the entire theme applied by a "Theme provider". This simplifies the implementation for all components.

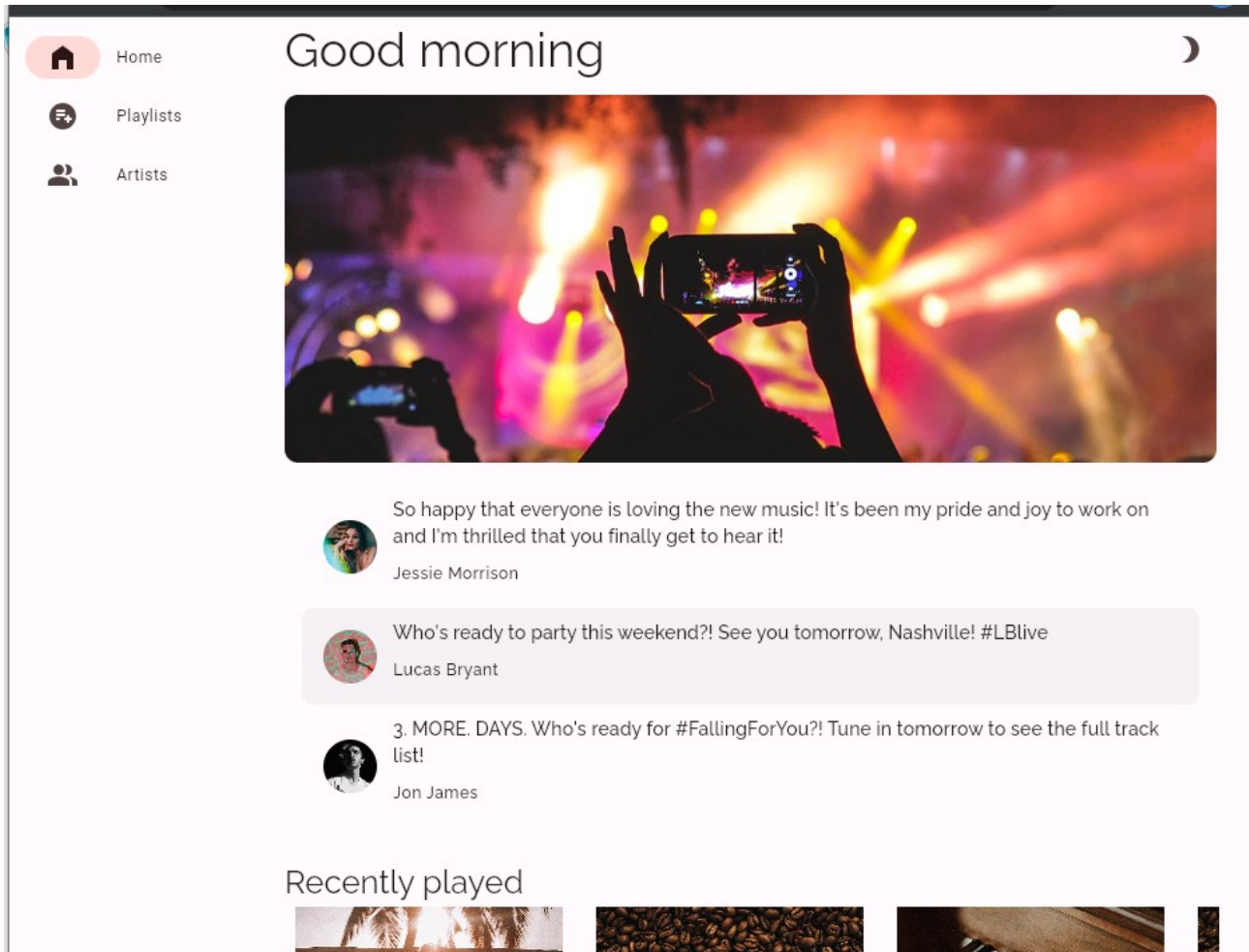
To add a theme we just have to create an instance of ThemeProvider and apply the theme to all of the ui

```
1  valueListenable: settings,
2  builder: (context, value, _) {
3    final theme = ThemeProvider.of(context); // Create theme instance
4    return MaterialApp.router(
5      debugShowCheckedModeBanner: false,
6      title: 'Flutter Demo',
7      theme: theme.light(settings.value.sourceColor), // Add theme
8      // Add dark theme
9      // Add theme mode
10     routeInformationParser: appRouter.routeInformationParser,
11     routeInformationProvider:
12       appRouter.routeInformationProvider,
13     routerDelegate: appRouter.routerDelegate,
14     builder: (context, child) {
15       return PlayPauseListener(child: child!);
16     },
17   );
18 }
```

To choose the colours for the application I've gone and used the masthead banner image of the application to chose the palette on material theme builder



Here's what the application looks like after applying the primary color;

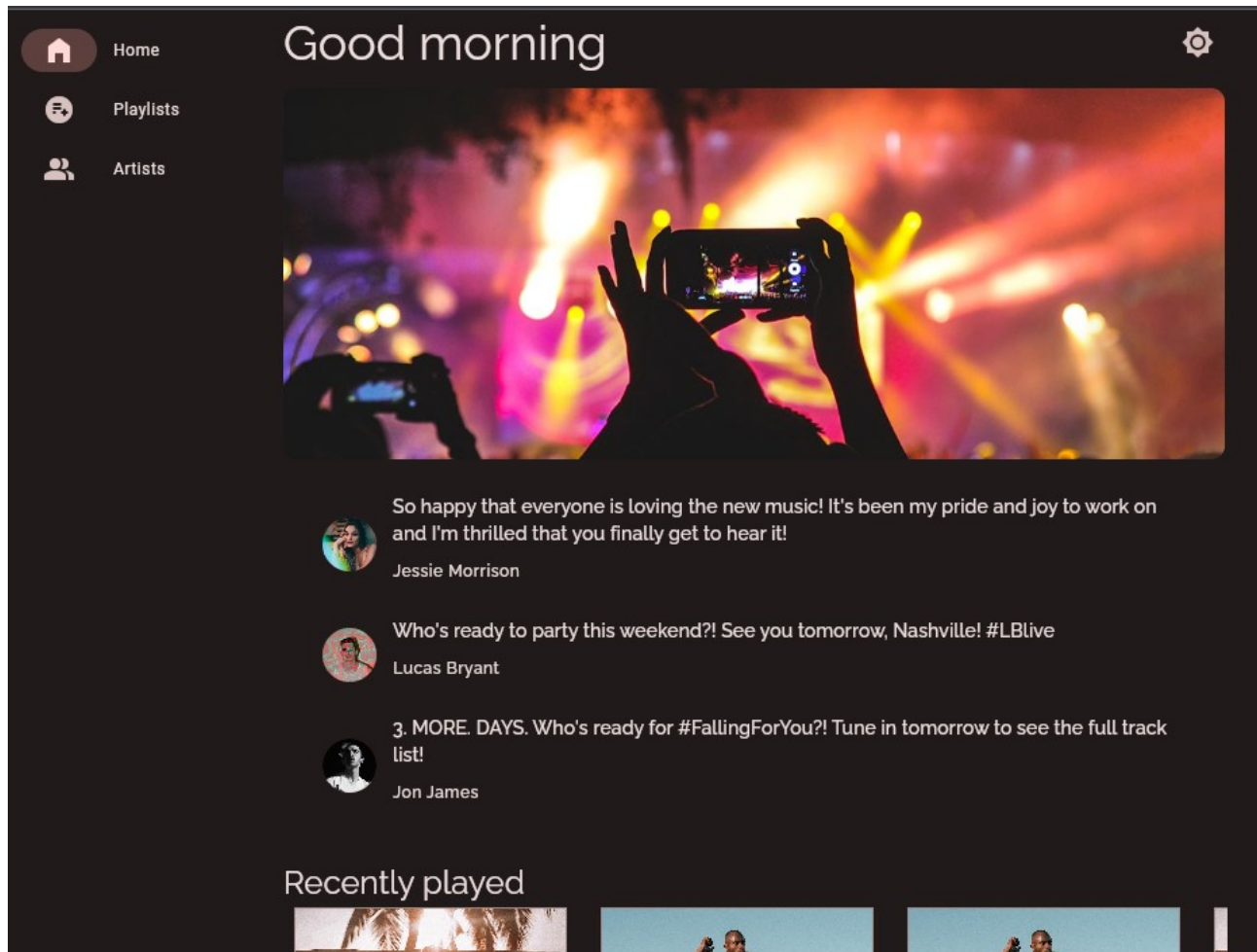


```
builder: (context, value, _) {  
  final colors = Theme.of(context).colorScheme;  
  final theme =  
    ThemeProvider.of(context); // Create theme instance
```

After adding colorScheme we can change outlines and secondary colours:

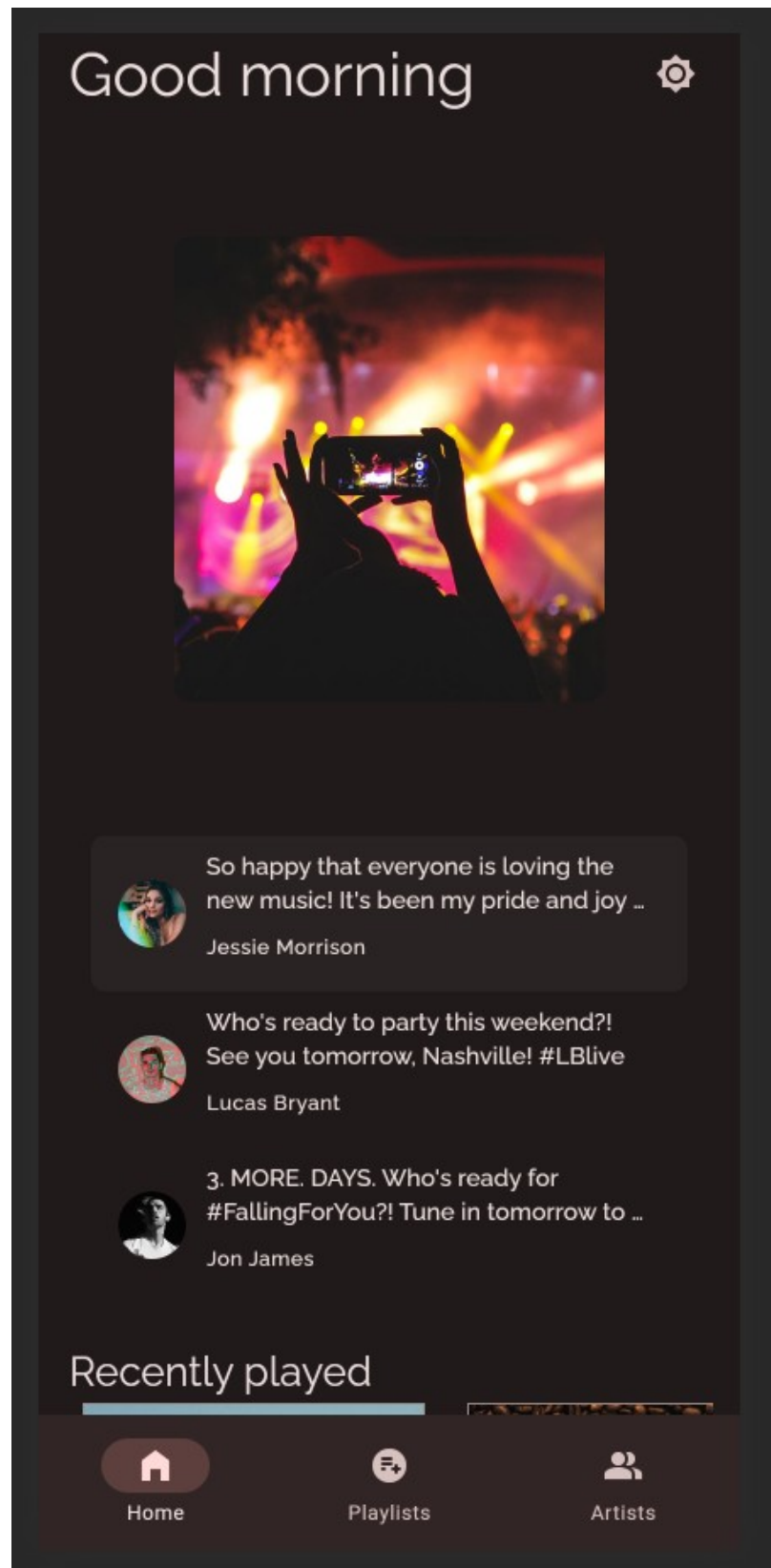
```
o's ready to party this  
as Bryant  
CODE DAYS WITH ME
```

Implemented dark mode and mode switching

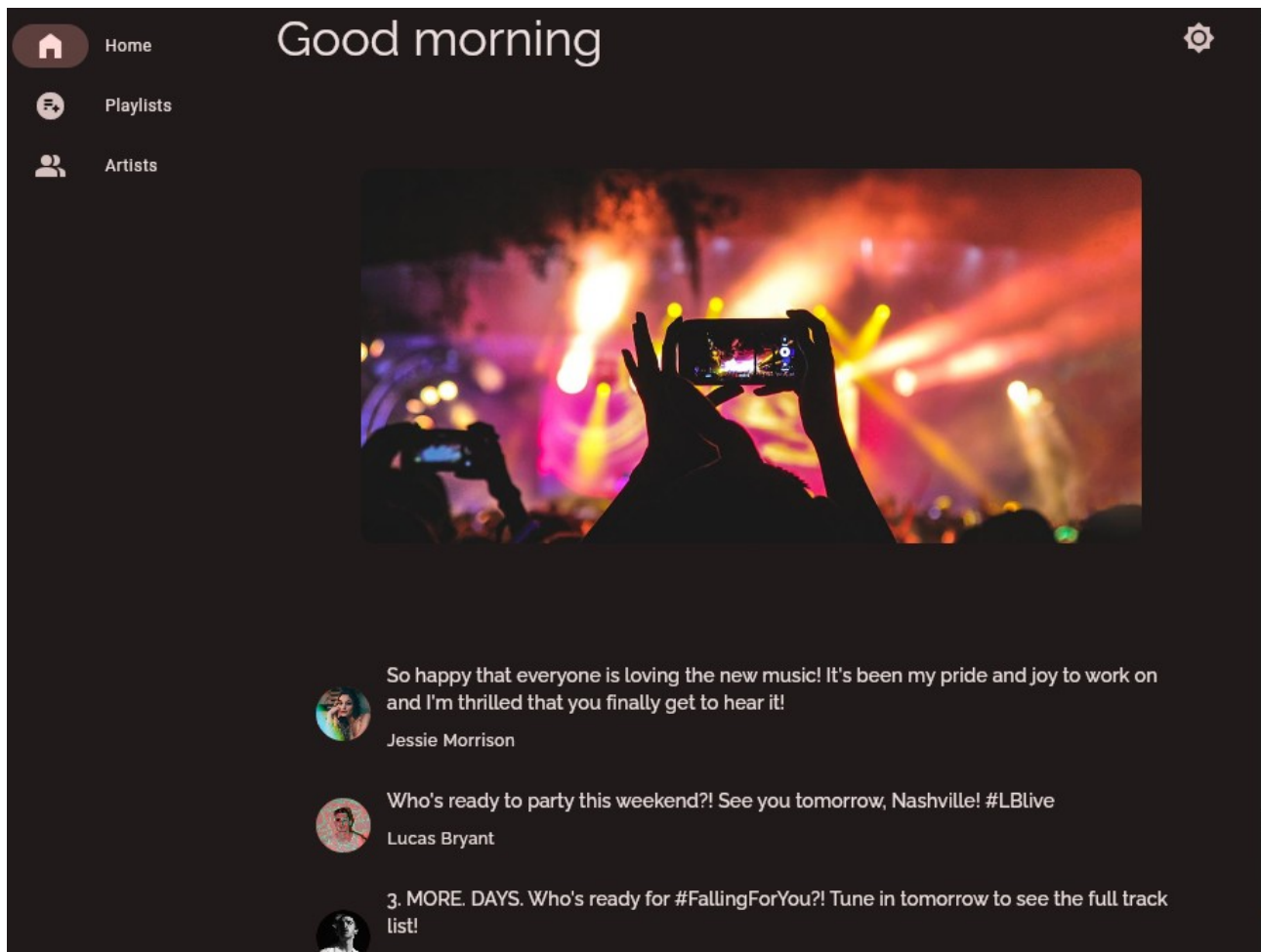


Adding adaptive design

Added padding and adaptive design that changes for mobile devices



On Tablet:



Adding animations and responsive feedback to app (check by running app in repository)

Task 3

Adding Flutter Authentication

In this section I'll be making a very basic application that has an authentication system. This is a simple login form application as instructed by the task.

Auth0

Created and configured an account for the service

Applying dependencies and changing the main.dart file

```
auth0 > lib > main.dart > MyApp
1  /// -----
2  /// External Packages
3  /// -----
4
5  import 'package:flutter/material.dart';
6
7  /// -----
8  /// Profile Widget
9  /// -----
10
11  /// -----
12  /// Login Widget
13  /// -----
14
15  /// -----
16  /// App
17  /// -----
18
19  Run | Debug | Profile
20  void main() => runApp(MyApp());
21
22  class MyApp extends StatelessWidget {
23    @override
24    _MyAppState createState() => _MyAppState();
25  }
26
27  /// -----
28  /// App State
29  /// -----
30  class _MyAppState extends State<MyApp> {
31    bool isBusy = false;
32    late String errorMessage;
33
34    @override
35    Widget build(BuildContext context) {
36      return MaterialApp(
37        title: 'Auth0 Demo',
38        home: Scaffold(
39          appBar: AppBar(
40            title: const Text('Auth0 Demo'),
41          ), // AppBar
42          body: Center(
43            child: const Text('Implement User Authentication'),
44          ), // Center
45        ),
46      );
47    }
48  }
49
50  flutter_lints 2.0.3 (3.0.1 available)
51  + flutter_web_plugins 0.0.0 from sdk flutter
52  + js 0.6.7 (0.7.0 available)
53  + lints 2.1.1 (3.0.0 available)
54  + matcher 0.12.16 (0.12.16+1 available)
55  + material_color_utilities 0.5.0 (0.8.0 available)
56  + meta 1.10.0 (1.11.0 available)
57  + path 1.8.3 (1.9.0 available)
58  + plugin_platform_interface 2.1.8
59  + test_api 0.6.1 (0.7.0 available)
60  + web 0.3.0 (0.4.0 available)
61  flutter pub get --no-examples running...
```

This is our final application

Auth0 Demo



The image shows a screenshot of the Auth0 Demo application interface. It has a light pink background. In the top right corner, there is a red ribbon with the word 'DEBUG' in white. In the center of the screen, there is a single, light purple rounded rectangular button with the word 'Login' in a darker purple font.

Login

Use ChatGPT to create a Flutter Mobile App

write flutter code to make a simple application template




ChatGPT

Sure, I can provide you with a simple Flutter code for a basic application template. This template will create a Flutter app with a simple structure, including a home screen and an app bar. Make sure you have Flutter and Dart installed on your machine before running this code.

1. Create a new Flutter project:

bash

 Copy code

```
flutter create my_flutter_app  
cd my_flutter_app
```

1. Replace the contents of `lib/main.dart` with the following code:

```
dart Copy code

import 'package:flutter/material.dart';


void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'My Flutter App',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: MyHomePage(),
    );
  }
}

class MyHomePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Home'),
      ),
      body: Center(
        child: Text(
          'Welcome to My Flutter App!',
          style: TextStyle(fontSize: 20),
        ),
      ),
    );
  }
}
```

1. Save the changes and run the app:

```
bash
```

 Copy code

```
flutter run
```

This code creates a simple Flutter app with a `MaterialApp` containing a home screen represented by `MyHomePage` and an app bar. The app displays a welcome message in the center of the screen.

Running this code we get;

