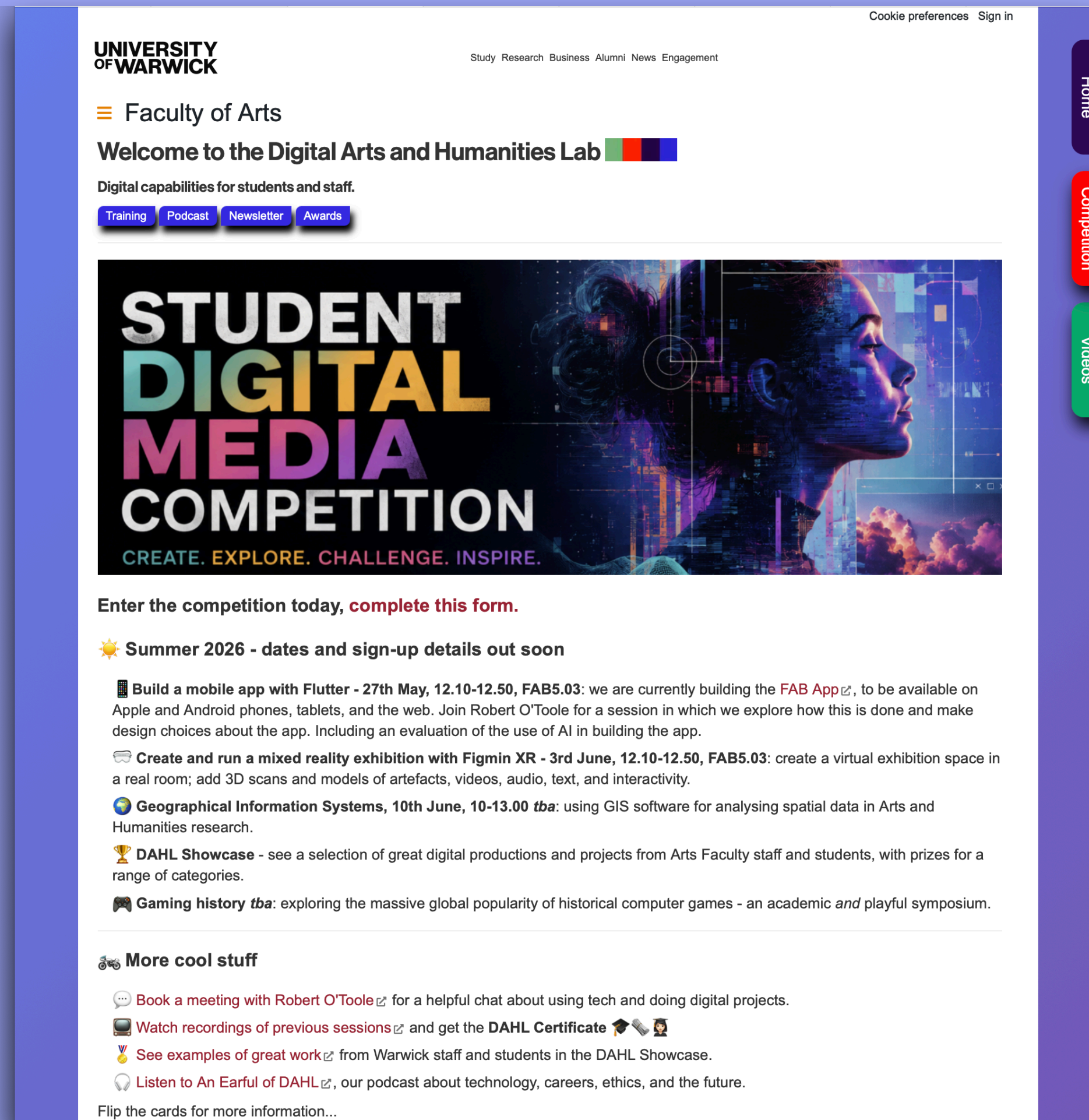


DAHL beginners guide to app development with Flutter & AI

Can AI make an incredibly complex process manageable, without losing too much control?



The screenshot shows the DAHL home page with the following content:

- UNIVERSITY OF WARWICK logo and navigation links (Study, Research, Business, Alumni, News, Engagement).
- Faculty of Arts header.
- Welcome to the Digital Arts and Humanities Lab with a small flag icon.
- Links for Training, Podcast, Newsletter, and Awards.
- STUDENT DIGITAL MEDIA COMPETITION banner with the tagline "CREATE. EXPLORE. CHALLENGE. INSPIRE."
- Text: "Enter the competition today, complete this form."
- Section: "Summer 2026 - dates and sign-up details out soon"
- Activity list:
 - Build a mobile app with Flutter - 27th May, 12.10-12.50, FAB5.03: we are currently building the FAB App, to be available on Apple and Android phones, tablets, and the web. Join Robert O'Toole for a session in which we explore how this is done and make design choices about the app. Including an evaluation of the use of AI in building the app.
 - Create and run a mixed reality exhibition with Figmin XR - 3rd June, 12.10-12.50, FAB5.03: create a virtual exhibition space in a real room; add 3D scans and models of artefacts, videos, audio, text, and interactivity.
 - Geographical Information Systems, 10th June, 10-13.00 tba: using GIS software for analysing spatial data in Arts and Humanities research.
 - DAHL Showcase - see a selection of great digital productions and projects from Arts Faculty staff and students, with prizes for a range of categories.
 - Gaming history tba: exploring the massive global popularity of historical computer games - an academic and playful symposium.
- Section: "More cool stuff"
- Links for:
 - Book a meeting with Robert O'Toole for a helpful chat about using tech and doing digital projects.
 - Watch recordings of previous sessions and get the DAHL Certificate.
 - See examples of great work from Warwick staff and students in the DAHL Showcase.
 - Listen to An Earful of DAHL, our podcast about technology, careers, ethics, and the future.
- Footer: "Flip the cards for more information..."



Dr Robert O'Toole

AI plus good design and coding practices might

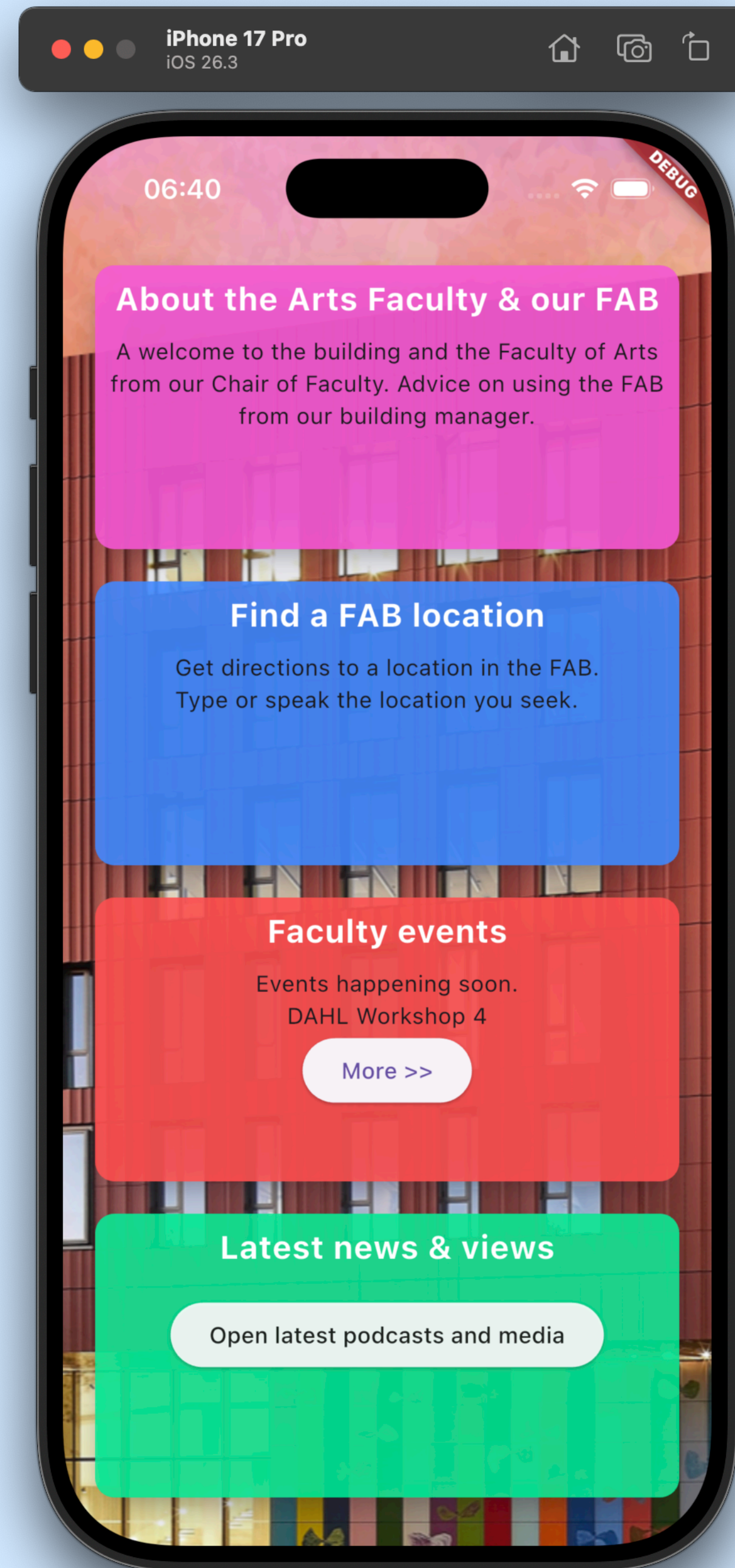
DAHL home page

The aim of this session is to give you enough of an understanding of app development using modern tools, so that you can either:

- 1.start learning how to build your own apps;**
- 2.design an app, working with app developers;**
- 3.decide that app development is too hard.**

**If you want to take it further,
I am happy to give you as
much help as you need**

**Example project used
for learning Flutter:
the FAB App**

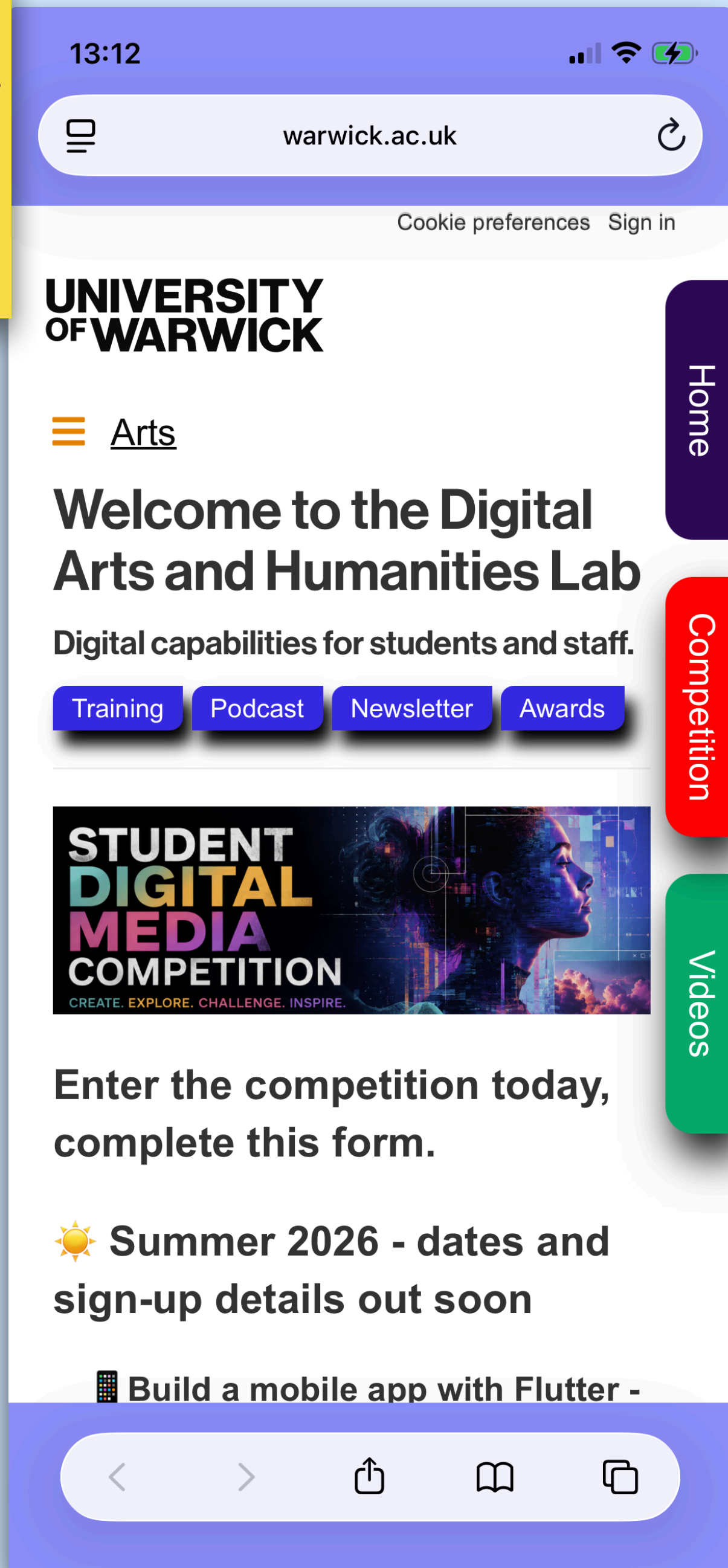


What are apps, why is developing them so difficult, and are they worth it?

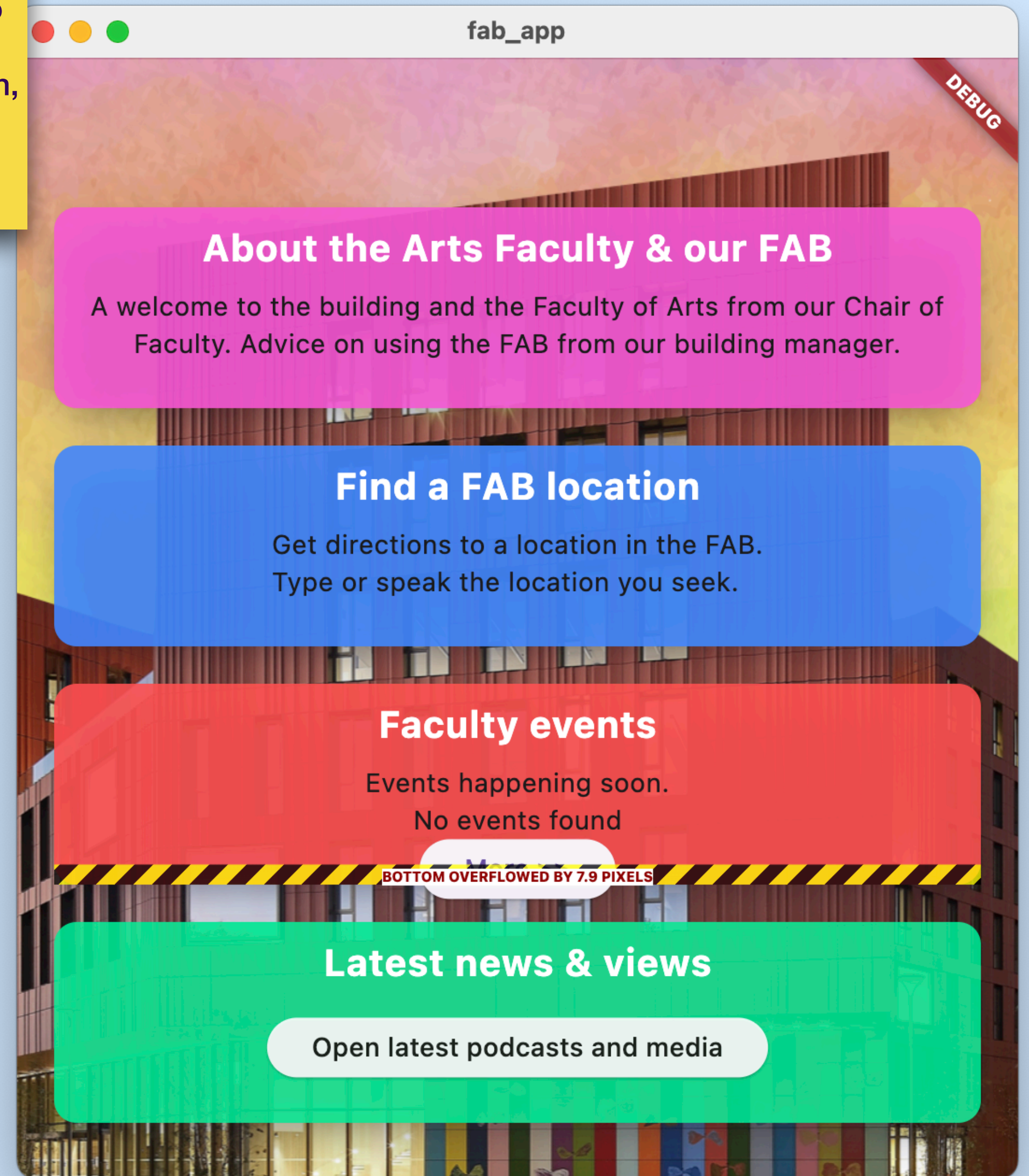
Apps are different to web pages...

- Installed on the device, not continually downloaded from the web (no hosting cost);
- Processing on the device, not in the cloud;
- AI on the device not in the cloud;
- Run without wireless connection;
- Integrated with payment systems;
- Can run in the background;
- Close integration with operating system and device features:
 - Voice assistants;
 - Location services;
 - Cameras, mic, accelerometer;
 - Near Field Communication (NFC) sensor;
 - Contacts list;
 - Notifications;
 - Other apps.

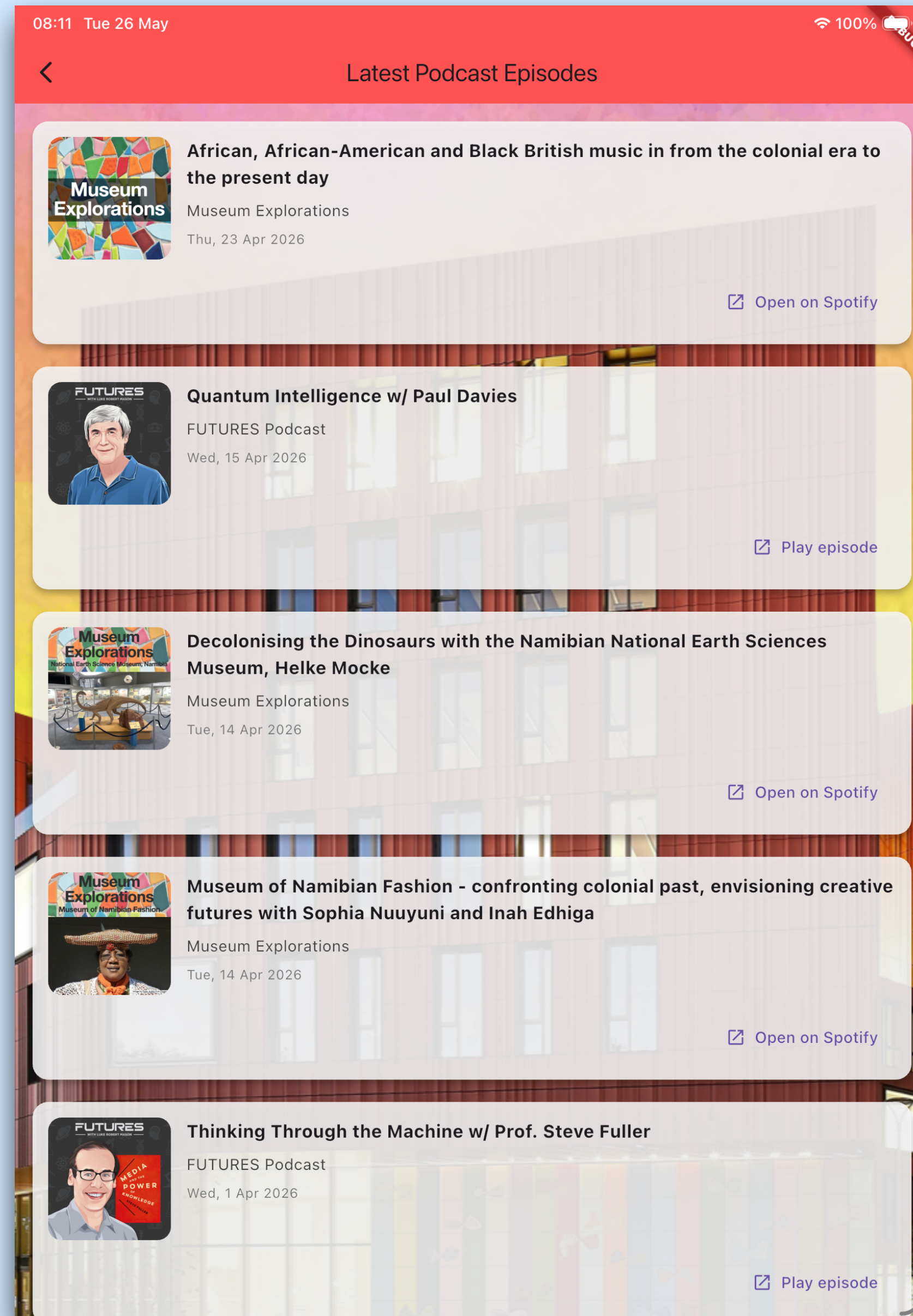
By default, web pages expand and scroll if the contents are too big for the page.



Apps have to fit into the available screen space and orientation, resizing contents where possible.



Unless we add a container that allows for scrolling, such as the CustomScrollView in which the list of podcasts is contained.



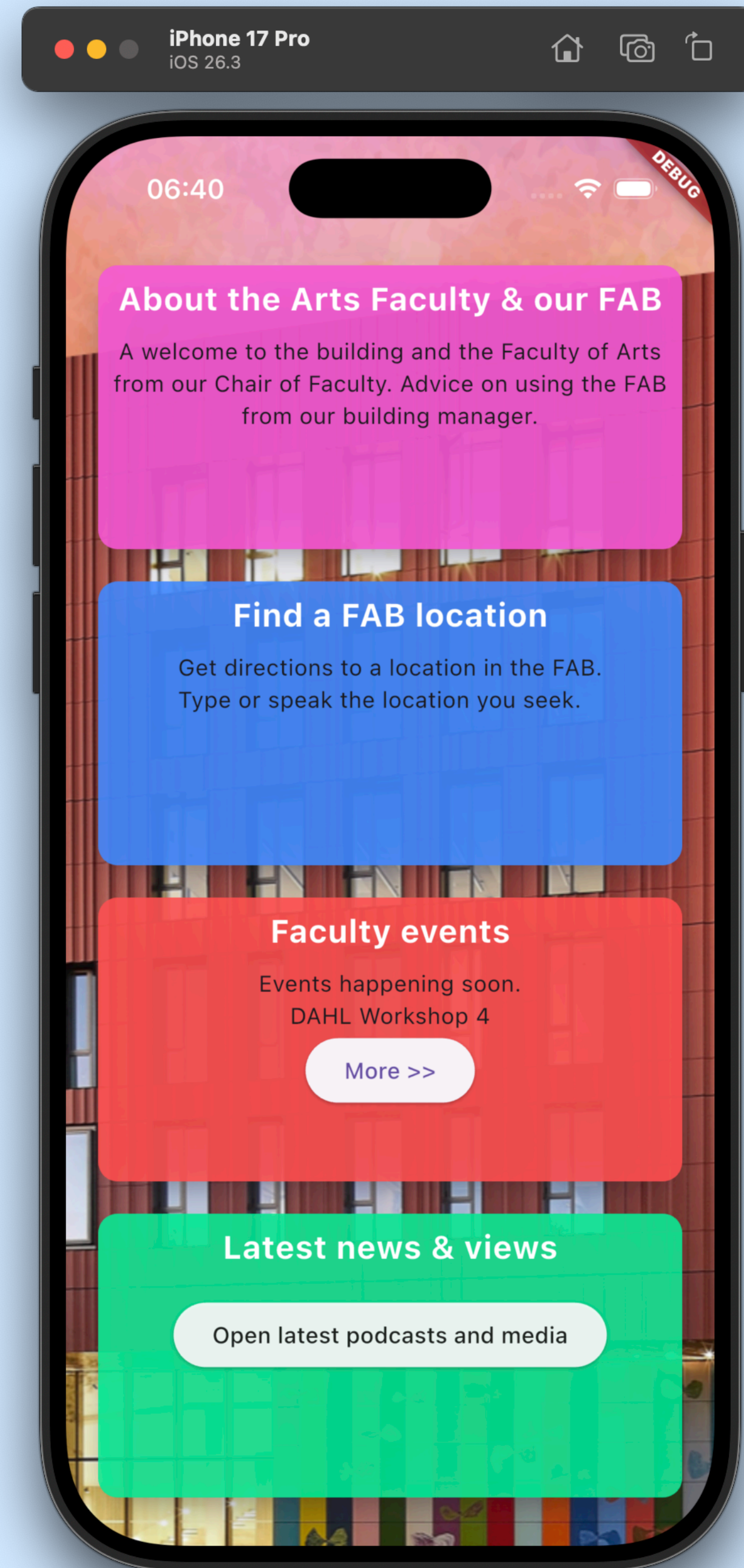
Apple app distribution is controlled through the App Store and Apple's quality control (money making) rules.

Android has app stores, but apps can also be distributed as files.

Apple rule: an app cannot be just repackaged web pages, there must be a strong reason for the app's existence as an app.

Achieving stickiness: there has to be a compelling reason for users to download and keep an app on their device.

Worth it?
Does it fit?
Will it stick?
Will it spread?
Can we grow it
further?



**Different code base,
different design
languages:**

Material (Google)

Cupertino (Apple)

And the other
platforms are also
completely
different.

Flutter can use either or both to
build universal apps.

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

class HybridAppExample extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Platform.isIOS
      ? CupertinoPageScaffold(
        navigationBar: CupertinoNavigationBar(
          middle: Text("Hybrid App"),
        ),
        child: Center(
          child: CupertinoButton(
            color: CupertinoColors.activeGreen,
            onPressed: () {},
            child: Text("Cupertino Style"),
          ),
        ),
      )
      : Scaffold(
        appBar: AppBar(title: Text("Hybrid App")),
        body: Center(
          child: ElevatedButton(
            onPressed: () {},
            child: Text("Material Style"),
            style: ElevatedButton.styleFrom(primary: Colors.green),
          ),
        ),
      );
  }
}
```

<https://blog.stackademic.com/flutter-material-vs-cupertino-understanding-the-differences-and-choosing-the-right-framework-6e1a827e0eac>

Flutter is a solution (from Google)

A software development kit (SDK) for building apps using the Dart language and common components (widgets), and a simple automatic method for translating, running, and testing on Apple, Android, Windows and web.

Rather than developing apps separately in Apple Xcode and Android Studio we can...

“Build once, run anywhere.”

Why not use FlutterFlow?

Fully automated online AI app builder.

Cost.

Dependency.

Environmental impact.

<https://docs.flutter.dev>

The screenshot shows the Flutter documentation website. The top navigation bar includes the Flutter logo, 'Docs' (highlighted), 'Guides', 'Learn', 'AI', and 'Reference'. On the right, there are search, theme, and grid icons, and a 'Get started' button. A purple banner at the top right contains the text: 'Flutter 3.44 is here! Everywhere, everyday, built by everyone, for everyone. Watch the live announcement and read the blog post.'

The left sidebar navigation menu is organized as follows:

- Get started
 - Quick start
 - Custom setup
 - Learn Flutter
- User interface
 - Introduction
 - Widget catalog
 - Layout
 - Adaptive & responsive design
 - Design & theming
 - Interactivity
 - Assets & media
 - Navigation & routing
 - Animations & transitions
 - Accessibility
 - Internationalization
- Beyond UI
 - Data & backend
 - App architecture
 - Platform integration
 - Packages & plugins
 - Testing & debugging
 - Performance & optimization
 - Deployment
 - Add to an existing app
- Resources

The main content area features a large heading 'Flutter documentation' with a subtext: 'Get started with Flutter. Widgets, examples, updates, and API docs to help you write your first Flutter app.' Below this are six cards:

- Install Flutter**: Set up your environment and start building.
- Widget catalog**: Dip into the rich set of Flutter widgets available in the SDK.
- API docs**: Bookmark the API reference docs for the Flutter framework.
- Learning resources**: Browse tutorials, sample code, workshops, and recipes.
- Videos**: View the many videos on the Flutter YouTube channel.
- Create with AI**: Learn how to build with and integrate powerful AI tools.

Below the cards, a link states: 'To see changes to the site since our last release, see [What's new](#).'

The 'New to Flutter?' section asks: 'Ready to build beautiful, multiplatform apps from a single codebase? First, [set up your Flutter environment](#), then follow the [Flutter learning pathway](#).'

The 'Docs' section includes: 'Coming from another platform? Check out Flutter for: [Android](#), [SwiftUI](#), [UIKit](#), [React Native](#), and [Xamarin.Forms](#) developers.'

Under 'Building layouts', it says: 'Learn how to create layouts in Flutter, where everything is a [widget](#).'

Under 'Understanding constraints', it says: 'Once you understand that "Constraints flow down. Sizes flow up. Parents set positions", then you are well on your way to understanding Flutter's layout model.'

At the bottom, a link reads: [Adding interactivity to your Flutter app](#)

<https://docs.flutter.dev/ui/widgets/material>

The screenshot shows the Flutter documentation website. At the top, there's a navigation bar with 'Flutter Docs' and links for 'Guides', 'Learn', 'AI', and 'Reference'. A search icon, a settings icon, and a 'Get started' button are also present. Below the navigation bar is a blue banner announcing 'Flutter 3.44 is here! Everywhere, everyday, built by everyone, for everyone. Watch the live announcement and read the blog post.'

The main content area is titled 'Material component widgets' and is part of the 'Widget catalog' under 'Design systems' > 'Cupertino'. It includes a breadcrumb 'UI > Widgets > Material components' and a description: 'A catalog of Flutter's material component widgets. Visual, behavioral, and motion-rich widgets implementing the Material 3 design specification.'

The text explains that Flutter provides a variety of visual, behavioral, and motion-rich widgets that implement the Material 3 design specification. It notes that Material 3 is the default design language of Flutter, enabling you to design and build beautiful, usable apps that can adapt to any platform.

A callout box states: 'The transition to Material 3 as the default was completed in Flutter 3.16. To learn more about this transition, how to complete it for your own widgets, or how to temporarily opt-out, check out the [Migrate to Material 3](#) migration guide.'

It also mentions: 'To catch these and other widgets in action, check out the [Material 3 demo](#) web app.'

The 'Actions' section features three cards:

- Common buttons**: Clickable blocks that start an action, such as sending an email, sharing a document, or liking a comment. (Example: 'Make payment' button)
- FloatingActionButton**: Clickable block containing an icon that keeps a key action always in reach. (Example: Edit icon button)
- Extended FloatingActionButton**: Clickable block that triggers an action. These wider blocks can fit a text label and provide a larger target area. (Example: '+ New task' button)

The left sidebar contains a 'Get started' section with 'Quick start', 'Custom setup', and 'Learn Flutter'. Below that is the 'User interface' section with 'Introduction', 'Widget catalog' (expanded), 'Overview', 'Design systems' (expanded), 'Cupertino', 'Material components' (selected), 'Base widgets', 'Layout', 'Adaptive & responsive design', 'Design & theming', 'Interactivity', 'Assets & media', 'Navigation & routing', 'Animations & transitions', 'Accessibility', and 'Internationalization'. At the bottom is the 'Beyond UI' section with 'Data & backend', 'App architecture', 'Platform integration', 'Packages & plugins', and 'Testing & debugging'.

Net Ninja free course

YouTube GB

Search

Base 44 | Just Start Building

Bring your ideas to life
Sponsored · base44.com

Start now

Flutter Crash Course

Net Ninja - 1 / 18

- 1 Flutter Crash Course #1 - What is Flutter? (6:56)
- 2 Flutter Crash Course #2 - Installing on Windows (10:49)
- 3 Flutter Crash Course #3 - Installing on a Mac (13:07)
- 4 Flutter Crash Course #4 - Making a New Flutter Project (5:00)
- 5 Flutter Crash Course #5 - Quick Project Overview (4:07)
- 6 Flutter Crash Course #6 - Widgets (8:10)
- 7 Flutter Crash Course #7 - MaterialApp & Scaffold (11:43)
- 8 Flutter Crash Course #8 - Stateless Widgets (6:27)

Flutter Crash Course #1 - What is Flutter?

Net Ninja

318K views 2 years ago Flutter Crash Course

In this Flutter Crash Course tutorial series, you'll learn how to make Flutter applications from scratch.

Get the Flutter Masterclass Course: ...more

Net Ninja 1.89M subscribers

Join

2.7K

Share

Ask

Save

All From the series From Net Ninja Flutte >

Dart programming language

In the same linguistic group as C, C++, C#, Java, Javascript etc. not Python.

Lots of shorthand syntax that can be confusing.

```
if ( age >= 18) { status = "Adult" } else { status = "Minor" }
```

can be written as

```
String status = (age >= 18) ? "Adult" : "Minor";
```

Advanced object orientation is conceptually challenging.

Flutter works in:

Microsoft Visual Studio Code

Integrated Development Environment (IDE)

With a Flutter plugin

(MSVC also used for many other forms of programming, such as Python based Jupyter Notebooks)

fab_app

Update

main.dart x startupscreen.dart eventsfullscreen.dart 1 scrollistfromjsonfeed.dart

CHAT

PODCAST EPISODES DISPLAY WIDGET

Where is the background color of the podcast episode cards specified?

Inspecting background color

Reviewing `mediafullscreen.dart` card definitions to locate the episode card background color.

Reviewed `mediafullscreen.dart` and clarified background color

The episode cards don't have an explicit background color set in `mediafullscreen.dart`.

- The `Card(...)` widget at line ~70 is created without a `color:` property.
- That means it uses the default Material card background from the current theme.

So the card background is effectively coming from the app's default `Card` theme, not from a hardcoded color in `mediafullscreen.dart`.

PROBLEMS 18 OUTPUT DEBUG CONSOLE ... Filter (e.g. text, !ex... flutter (package:fab_...)

```
lib > main.dart > main
1 import 'package:fab_app/model/fabmap.dart';
2 import 'package:fab_app/model/location.dart';
3 import 'package:fab_app/view/startupscreen.dart';
4 import 'package:flutter/material.dart';
5 import 'package:flutter/services.dart';
6
7 void main() {
8   Location location = Location("FAB225");
9   FABMap().addToMap(location);
10  runApp(MaterialApp(
11    title: 'FAB App',
12    home: StartupScreen(),
13  )); // MaterialApp
14  SystemChrome.setPreferredOrientations([ DeviceOrientation.portraitUp ]);
15 }
```

exit code 0

Describe what to build

Local Default Approvals

Ln 15, Col 2 Spaces: 2 UTF-8 LF {} Dart iPhone 17 Pro (ios simulator)

Code hinting + code completion

```
class _EventsFullScreen extends State<EventsFullScreen>
```

```
int selectedIndex = 0;
```

```
void returnToMenu(){
```

```
  if(context.mounted == true)
```

```
{
```

```
  Navigator.pop(context);
```

```
  Nav
```

NavigationBar

NavigationBarTheme

NavigationBarThemeData

NavigationDestination

NavigationDrawer

NavigationDrawerDestination

NavigationDrawerTheme

NavigationDrawerThemeData

NavigationIndicator

Notification

NavigationRail

NavigationRailDestination

```
@override
```

```
void initState() {
```

```
  super.initState();
```

Uses Github Copilot AI if configured

But can be annoying, and gets in the way

Code fixing and QA

```
class _EventsFullScreen extends State<EventsFullScreen>
```

```
{
```



```
int _selectedIndex = 0;
```

Quick Fix

- Remove unused field
- Ignore 'unused_field' for this line
- Ignore 'unused_field' for the whole file
- Ignore 'unused_field' in `analysis_options.yaml`
- Fix
- Explain
- More Actions...
- Remove type annotation
- Add a debug reference to this property
- Add 'late' modifier
- Convert '_selectedIndex' to a getter

```
@override
```

```
void initState() {  
  | super.initState();  
}
```

```
), // Padding
Padding(padding: const EdgeInsets.symmetric(horizontal: 0, vertical: 4)),
Row(mainAxisAlignment: MainAxisAlignment.spaceEvenly, children: [

  ElevatedButton(style: ButtonStyle(fixedSize: WidgetStateProperty.all(Size(100, 30))), onPressed: filterList(0), child: const Text("All", style: TextStyle(fontSize: 11),), ),
  ElevatedButton(style: ButtonStyle(fixedSize: MaterialStateProperty.all(Size(100, 30))), onPressed: filterList(1), child: const Text("Public", style: TextStyle(fontSize: 11),), ),
  ElevatedButton(style: ButtonStyle(fixedSize: MaterialStateProperty.all(Size(100, 30))), onPressed: filterList(2), child: const Text("Students", style: TextStyle(fontSize: 11),), ),
  ElevatedButton(style: ButtonStyle(fixedSize: MaterialStateProperty.all(Size(100, 30))), onPressed: filterList(3), child: const Text("Staff", style: TextStyle(fontSize: 11),), ),
  ElevatedButton(style: ButtonStyle(fixedSize: WidgetStateProperty.all(Size(100, 30))), onPressed: filterList(1), child: const Text("Public", style: TextStyle(fontSize: 11),), ),
  ElevatedButton(style: ButtonStyle(fixedSize: WidgetStateProperty.all(Size(100, 30))), onPressed: filterList(2), child: const Text("Students", style: TextStyle(fontSize: 11),), ),
  ElevatedButton(style: ButtonStyle(fixedSize: WidgetStateProperty.all(Size(100, 30))), onPressed: filterList(3), child: const Text("Staff", style: TextStyle(fontSize: 11),), ),

]), // Row
```

I started to type “class” and Copilot AI autosuggested the code I needed, and guessed some of the other things I might add

```
lib > view > aboutfabfullscreen.dart > ...
1  import 'package:flutter/material.dart';
2
3  class Aboutfabfullscreen extends StatefulWidget {
4    const Aboutfabfullscreen({super.key});
5
6    @override
7    State<Aboutfabfullscreen> createState() => Aboutfabfullscreen();
8  }
9
10 class Aboutfabfullscreen extends State<Aboutfabfullscreen>
    {
11   @override
12   Widget build(BuildContext context) {
13     return Scaffold(
14       body: Container(
15         decoration: BoxDecoration(
16           image: DecorationImage(
17             image: AssetImage("assets/images/FABbackground.png"),
18             opacity: 1,
19             fit: BoxFit.cover,
20           ),
21         ),
22       child:
23         Column(
24           crossAxisAlignment: CrossAxisAlignment.stretch,
25           mainAxisAlignment: MainAxisAlignment.center,
26           children: [
27             Container(child: Padding(padding: EdgeInsets.only(top: 60))),
28             Text("The FAB App is a project developed by the FAB Team at the University of Manchester. It is designed to provide students, staff, and visitors with easy access to informa
29             Container(child: Padding(padding: EdgeInsets.only(top: 20))),
30             Text("For more information about the FAB building and its activities, please visit our website or contact us directly.", style: TextStyle(fontSize: 16, color: Colors.white),
31
32
```

It knew I would want the FAB image in the background!

GitHub Copilot AI integration for generating code interactively

CHAT

← PODCAST EPISODES DISPLAY WIDGET

Where is the background color of the podcast episode cards specified?

Inspecting background color

Reviewing `mediafullscreen.dart` card definitions to locate the episode card background color.

Reviewed `mediafullscreen.dart` and clarified background color

The episode cards don't have an explicit background color set in `mediafullscreen.dart`.

- The `Card(...)` widget at line ~70 is created without a `color:` property.
- That means it uses the default Material card background from the current theme.

So the card background is effectively coming from the app's default `Card` theme, not from a hardcoded color in `mediafullscreen.dart`.

Describe what to build

Local Default Approvals

Dart iPhone 17 Pro (ios simulator)

**Build, run and debug in
Microsoft Visual Studio Code**

Choose a device to run, test and debug on.

The screenshot shows the Flutter DevTools interface. At the top, a dropdown menu titled "Select a device to use" is open, listing various devices under "Available Devices" and "Offline Emulators". The "Available Devices" list includes: iPhone 17 Pro (selected), iPad (A16), iPhone 17 Pro Max, macOS macos - desktop, Robert iPhone 12, and Medium Phone emulator-5554. The "Offline Emulators" section has a "+ Create Android emulator" button.

The code editor in the center shows the following Dart code:

```
1 import 'package:fab_app/mo
2 import 'package:fab_app/mo
3 import 'package:fab_app/vi
4 import 'package:flutter/ma
5 import 'package:flutter/se
6
7 void main() {
8   Location location = Loca
9   FABMap().addToMap(location);
10  runApp(MaterialApp(
11    title: 'FAB App',
12    home: StartupScreen(),
13  )); // MaterialApp
14  SystemChrome.setPreferredOrientations([ DeviceOrientation.portraitUp ]);
15 }
16
17
```

Below the code editor, the status bar shows "Disconnected" and "Run a new debug session to reconnect."

At the bottom, the "DEBUG CONSOLE" tab is active, showing the following log output:

```
-[WFIsolatedShortcutRunner init]_block_invoke Sandbox extensions acquired
Indexing for request: <WFToolkitIndexingRequest: 0x6000017008c0>, changeset: .none, priority: 31
Finished in 0.021377s
-[WFIsolatedShortcutRunner unaliveProcess] Releasing sandbox extensions
-[WFIsolatedShortcutRunner init] Taking sandbox extensions for execution
-[WFIsolatedShortcutRunner init]_block_invoke Sandbox extensions acquired
Indexing for request: <WFToolkitIndexingRequest: 0x6000017081c0>, changeset: .none, priority: 31
Finished in 0.023317s
-[WFIsolatedShortcutRunner unaliveProcess] Releasing sandbox extensions
```

On the right side, a "CHAT" window is open, displaying a question: "Where is the background color of the podcast episode cards specified?". The chat content includes:

- Inspecting background color
- Reviewing `mediafullscreen.dart` card definitions to locate the episode card background color.
- Reviewed `mediafullscreen.dart` and clarified background color
- The episode cards don't have an explicit background color set in `mediafullscreen.dart`.
- The `Card(...)` widget at line ~70 is created without a `color:` property.
- That means it uses the default Material card background from the current theme.
- So the card background is effectively coming from the app's default `Card` theme, not from a hardcoded color in `mediafullscreen.dart`.

At the bottom right, there is a "Describe what to build" input field with a search icon and a "Local" dropdown menu.

fab_app

main.dart eventsfullscreen.dart 1 x scrolllistfromjsonfeed.dart aboutfabfullscreen.dart podcast_feed.dart mediafullscreen.dart

VARIABLES

- > Locals
- > Globals

```
lib > view > eventsfullscreen.dart > _EventsFullScreen > returnToMenu
1 import 'package:fab_app/utilities/scrolllistfromjsonfeed.dart';
2 import 'package:flutter/material.dart';
3
4 class EventsFullScreen extends StatefulWidget {
5   const EventsFullScreen({super.key});
6
7   @override
8   State<EventsFullScreen> createState() => _EventsFullScreen();
9 }
10
11 class _EventsFullScreen extends State<EventsFullScreen>
12 {
13
14   int _selectedIndex = 0;
15
16   void returnToMenu(){
17     if(context.mounted == true)
18     {
19       Navigator.pop(context);
20     }
21   }
22
23   VoidCallback filterList(int index)
24   {
25     return () {
26       setState(() {
27         _selectedIndex = index;
28       });
29     };
30   }
31
32   @override
33   void initState() {
34     super.initState();
35   }
36
37   @override
38   Widget build(BuildContext context) {
39     return Scaffold(
40       body: Container(
41         decoration: BoxDecoration(
42           image: DecorationImage(
43             image: AssetImage("assets/images/FABbackgr
44             opacity: 1,
45             fit: BoxFit.cover,
```

PROBLEMS 18 OUTPUT DEBUG CONSOLE TERMINAL PORTS

CALL STACK

BREAKPOINTS

- All Exceptions
- Uncaught Exceptions
- eventsfullscreen.dart lib/view

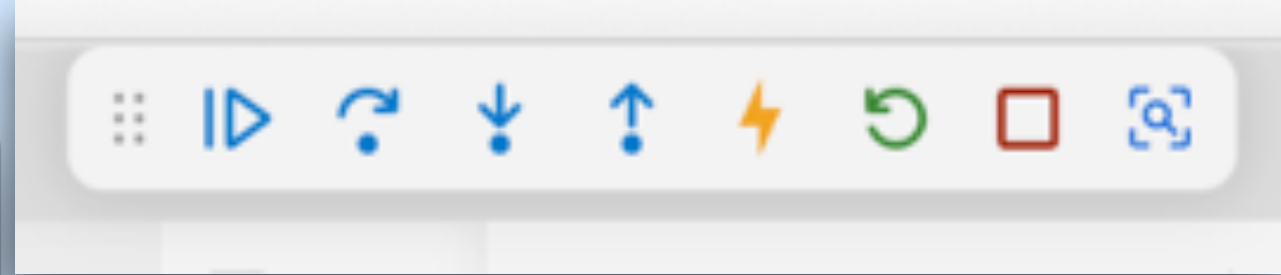
VARIABLES

- > Locals
- > Globals

WATCH

lib > view > eventsfullscreen.dart > _EventsFullScreen > returnToMenu

```
1 import 'package:fab_app/utilities/scrolllistfromjsonfeed.da
2 import 'package:flutter/material.dart';
3
4 class EventsFullScreen extends StatefulWidget {
5   const EventsFullScreen({super.key});
6
7   @override
8   State<EventsFullScreen> createState() => _EventsFullScre
9 }
10
11 class _EventsFullScreen extends State<EventsFullScreen>
12 {
13
14   int _selectedIndex = 0;
15
16   void returnToMenu(){
17     if(context.mounted == true)
18     {
19       Navigator.pop(context);
20     }
21   }
22
23   VoidCallback filterList(int index)
```



Where is the background color of the podcast episode cards specified?

Inspecting background color

Reviewing mediafullscreen.dart card definitions to locate the episode card background color.

Reviewed mediafullscreen.dart and clarified background color

The episode cards don't have an explicit background color set in mediafullscreen.dart

Run on multiple devices (real or simulated) at the same time, but can only debug on one at a time.

In terminal window, navigate to the folder containing the app, and enter “flutter run”, choose from the list of available devices.

Running on Android medium size phone

```
fab_app --zsh -- 80x26
V/MediaPlayer( 5402): resetDrmState: mDrmInfo=null mDrmProvisioningThread=null
mPrepareDrmInProgress=false mActiveDrmScheme=false
V/MediaPlayer( 5402): cleanDrmObj: mDrmObj=null mDrmSessionId=null
V/MediaPlayer( 5402): resetDrmState: mDrmInfo=null mDrmProvisioningThread=null
mPrepareDrmInProgress=false mActiveDrmScheme=false
V/MediaPlayer( 5402): cleanDrmObj: mDrmObj=null mDrmSessionId=null
D/WindowOnBackDispatcher( 5402): setTopOnBackInvokedCallback (unwrapped): io.flu
tter.embedding.android.FlutterActivity$1@5bd92ea
V/MediaPlayer( 5402): resetDrmState: mDrmInfo=null mDrmProvisioningThread=null
mPrepareDrmInProgress=false mActiveDrmScheme=false
V/MediaPlayer( 5402): cleanDrmObj: mDrmObj=null mDrmSessionId=null
V/MediaPlayer( 5402): resetDrmState: mDrmInfo=null mDrmProvisioningThread=null
mPrepareDrmInProgress=false mActiveDrmScheme=false
V/MediaPlayer( 5402): cleanDrmObj: mDrmObj=null mDrmSessionId=null
V/MediaPlayer( 5402): resetDrmState: mDrmInfo=null mDrmProvisioningThread=null
mPrepareDrmInProgress=false mActiveDrmScheme=false
V/MediaPlayer( 5402): cleanDrmObj: mDrmObj=null mDrmSessionId=null
W/libbinder.Binder( 5402): Binder transaction to android.view.IGraphicsStatsCall
back, function: UNKNOWN_FUNCTION_NAME, code: 1, took 2354ms. Data bytes: 88 Repl
y bytes: 0 Flags: 17
D/VRI[MainActivity]( 5402): visibilityChanged oldVisibility=true newVisibility=f
alse
W/example.fab_app( 5402): Reducing the number of considered missed Gc histogram
windows from 3488 to 100
Lost connection to device.
robertotoole@Robert-0Toolle-M3Pro fab_app %
```

Running as a web app

```
fab_app --dartaotruntime - dartvm --resolved_executable_name=/Users...
The specific RenderFlex in question is: RenderFlex#4e95b OVERFLOWING:
creator: Column < Padding < DecoratedBox < Container < Padding < Listener <
_GestureSemantics <
_RawGestureDetector < GestureDetector < Expanded < FABCard < Column < --
parentData: offset=Offset(0.0, 0.0) (can use size)
constraints: BoxConstraints(w=460.0, h=110.0)
size: Size(460.0, 110.0)
direction: vertical
mainAxisAlignment: start
mainAxisSize: max
crossAxisAlignment: center
verticalDirection: down
spacing: 0.0
/////////////////////////////////////////////////////////////////
Another exception was thrown: A RenderFlex overflowed by 4.4 pixels on the
bottom.
Another exception was thrown: A RenderFlex overflowed by 2.7 pixels on the
bottom.
Another exception was thrown: A RenderFlex overflowed by 2.7 pixels on the
bottom.
```

Running on my real iPhone 12

```
fab_app --dart - dartvm --resolved_executable_name=/Users/rober...
creator: Row < Column < Padding < DecoratedBox < Container <
KeyedSubtree-[GlobalKey#6ea14] <
_BodyBuilder < MediaQuery < LayoutId-[<_ScaffoldSlot.body>] <
_CustomMultiChildLayout <
_ActionsScope < Actions < --
parentData: offset=Offset(0.0, 126.0); flex=null; fit=null (can use
size)
constraints: BoxConstraints(w=390.0, 0.0<h<=Infinity)
size: Size(390.0, 48.0)
direction: horizontal
mainAxisAlignment: spaceEvenly
mainAxisSize: max
crossAxisAlignment: center
textDirection: ltr
verticalDirection: down
spacing: 0.0
/////////////////////////////////////////////////////////////////
```

Running as a MacOS app

```
fab_app --dart - dartvm --resolved_executable_name=/Users/robertotoo...
seen. If the content is legitimately bigger than the available space, consider
clipping it with a
ClipRect widget before putting it in the flex, or using a scrollable container
rather than a Flex,
like a ListView.
The specific RenderFlex in question is: RenderFlex#45f7e OVERFLOWING:
creator: Column < Padding < DecoratedBox < Container < Padding < Listener <
_GestureSemantics <
_RawGestureDetector < GestureDetector < Expanded < FABCard < Column < --
parentData: offset=Offset(0.0, 0.0) (can use size)
constraints: BoxConstraints(w=760.0, h=107.5)
size: Size(760.0, 107.5)
direction: vertical
mainAxisAlignment: start
mainAxisSize: max
crossAxisAlignment: center
verticalDirection: down
spacing: 0.0
/////////////////////////////////////////////////////////////////
```

Running on iPad

```
fab_app --dart - dartvm --resolved_executable_name=/Users/robertotoo...
Indexed: 0
Errored: 0
Skipped: [:]
Finished in 0.442397s
-[WFIsoaltedShortcutRunner unaliveProcess] Releasing sandbox extensions
-[WFIsoaltedShortcutRunner init] Taking sandbox extensions for execution
-[WFIsoaltedShortcutRunner init]_block_invoke Sandbox extensions acquired
Indexing for request: <WFToolKitIndexingRequest: 0x600001700380>, changeset: .no
ne, priority: 31
Finished in 0.024378s
-[WFIsoaltedShortcutRunner unaliveProcess] Releasing sandbox extensions
-[WFIsoaltedShortcutRunner init] Taking sandbox extensions for execution
-[WFIsoaltedShortcutRunner init]_block_invoke Sandbox extensions acquired
Indexing for request: <WFToolKitIndexingRequest: 0x600001714c40>, changeset: .no
ne, priority: 31
Finished in 0.025356s
-[WFIsoaltedShortcutRunner unaliveProcess] Releasing sandbox extensions
-[WFIsoaltedShortcutRunner init] Taking sandbox extensions for execution
-[WFIsoaltedShortcutRunner init]_block_invoke Sandbox extensions acquired
Indexing for request: <WFToolKitIndexingRequest: 0x600001714240>, changeset: .no
ne, priority: 31
Finished in 0.020833s
-[WFIsoaltedShortcutRunner unaliveProcess] Releasing sandbox extensions
```

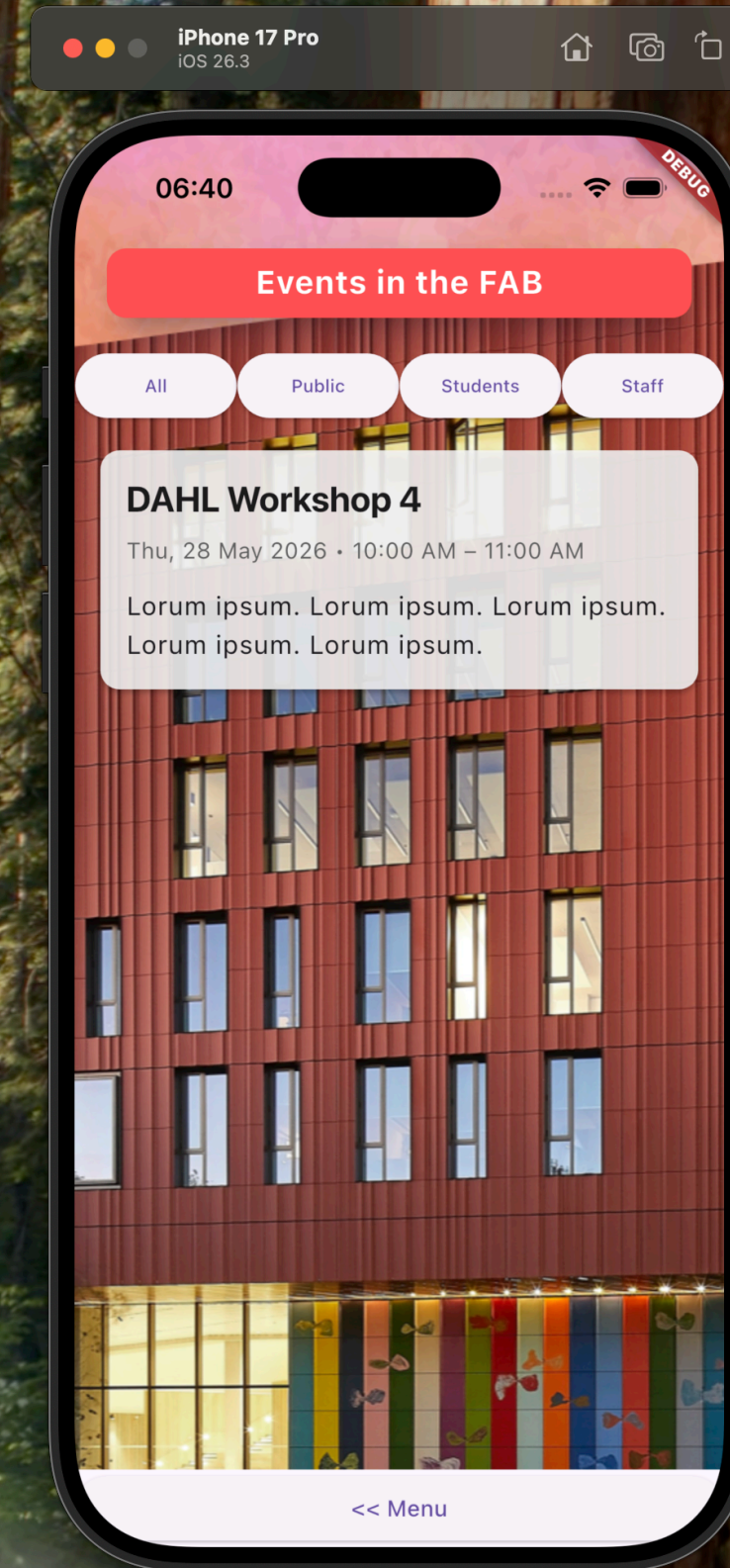
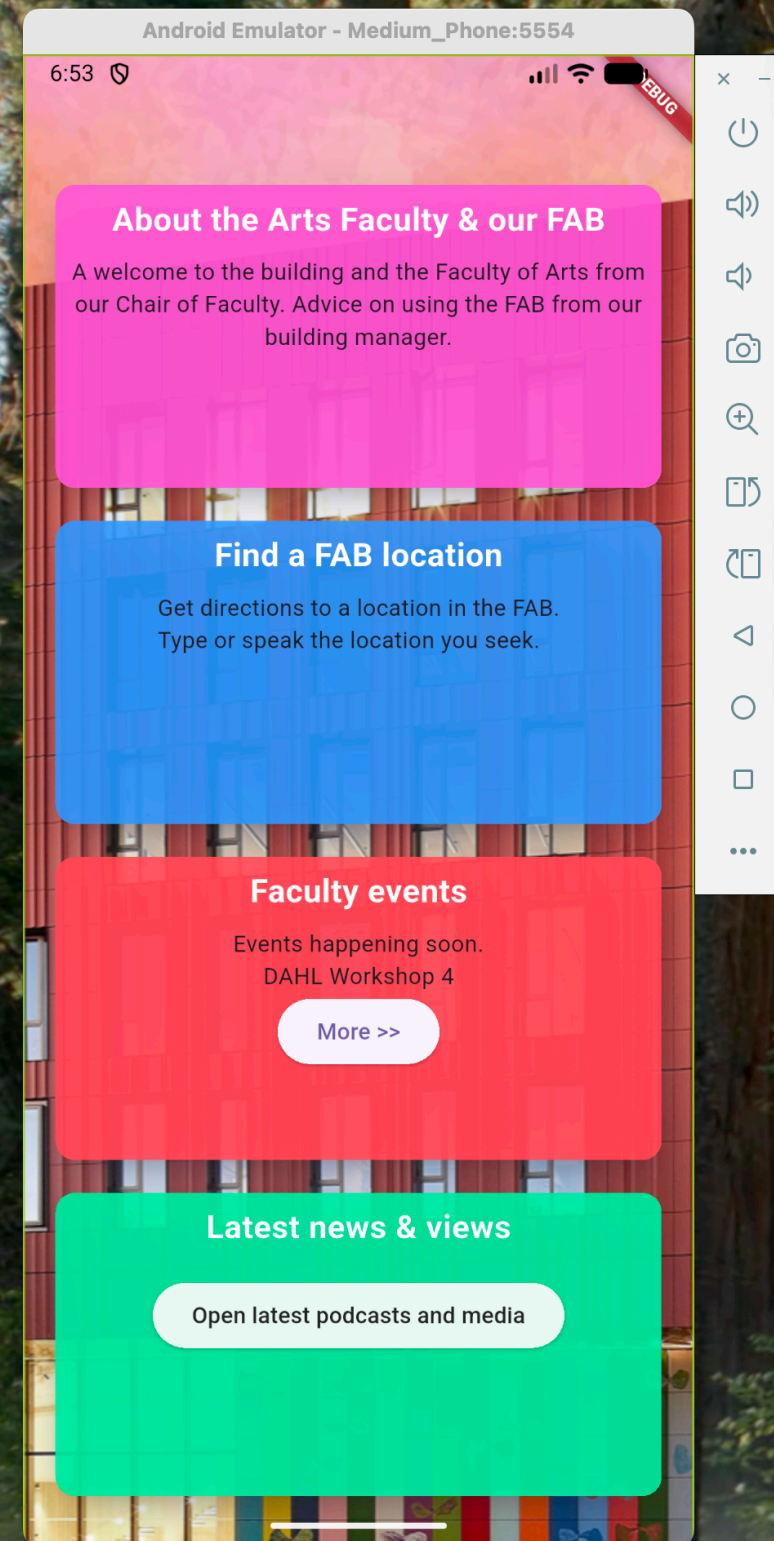
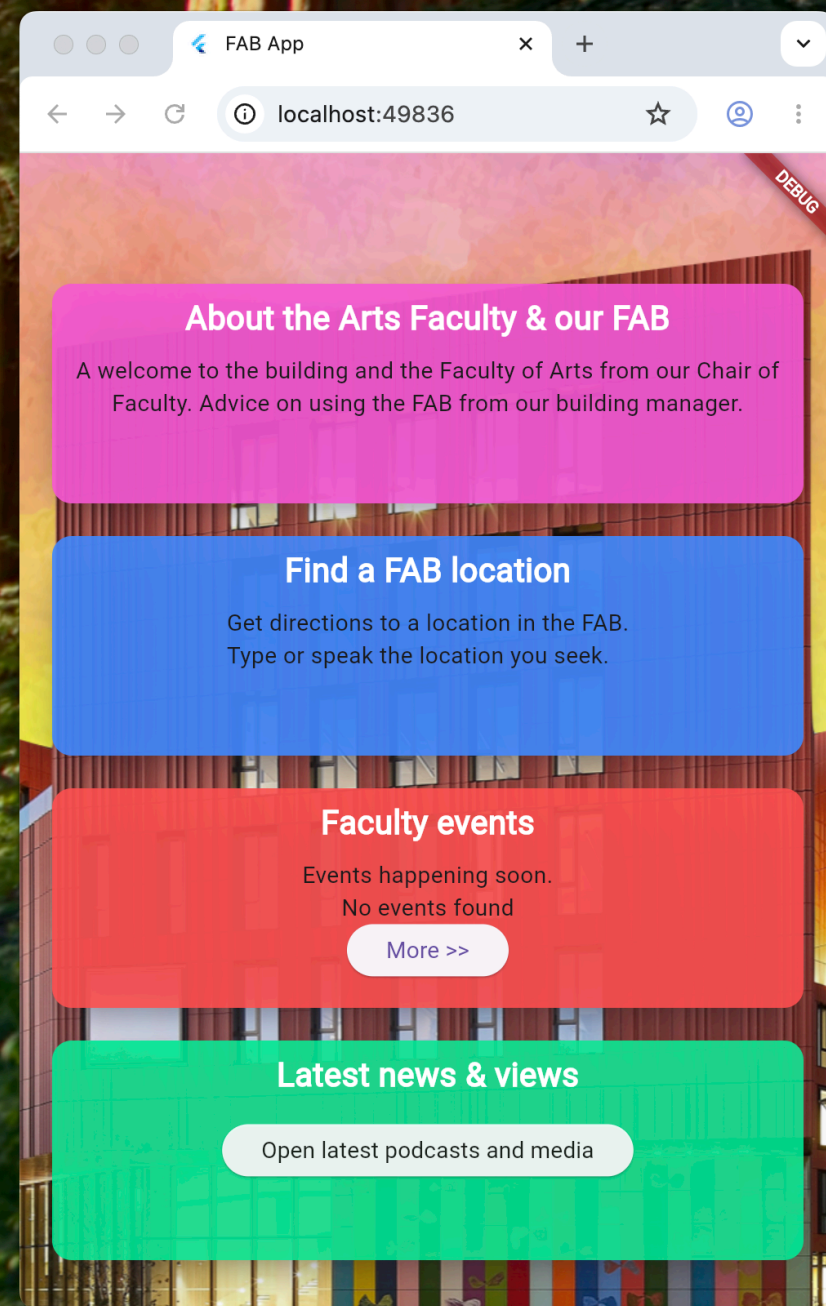
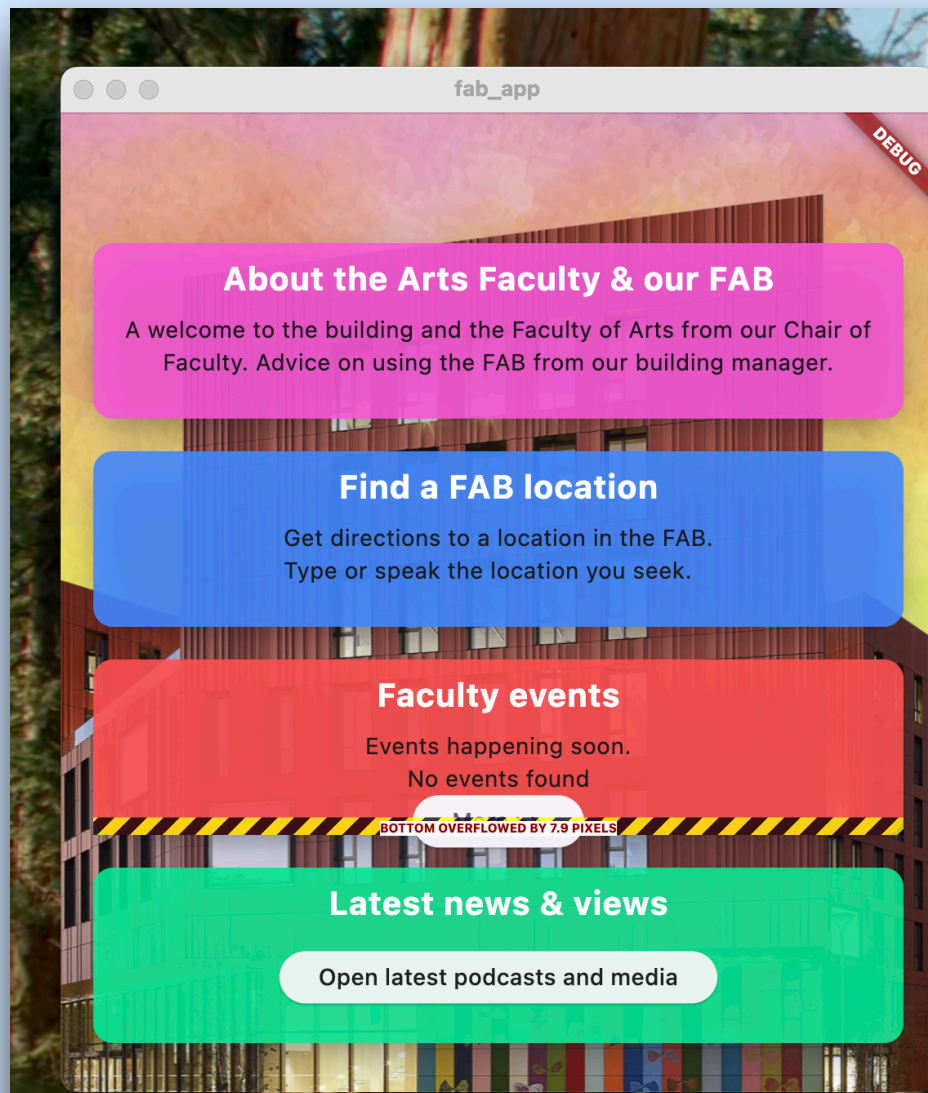
GitHub source control integration

The screenshot displays an IDE interface with a source control provider (GitHub) integrated. The left sidebar shows a file explorer with the following files and their status:

- IDEWorkspaceChecks.plist ios/Runne... U
- WorkspaceSettings.xcsettings ios/R... U
- RunnerTests.swift ios/RunnerTests U
- main.dart lib U
- dbservice.dart lib/model U
- event.dart lib/model U
- fabmap.dart lib/model U
- location.dart lib/model U
- publication.dart lib/model U
- audioplayer.dart lib/utilities U
- getjson.dart lib/utilities U
- getrss.dart lib/utilities U
- podcast_feed.dart lib/utilities U
- scrolllistfromjsonfeed.dart lib/utilities U
- scrolllistfromrssfeed.dart lib/utilities U
- aboutfabfullscreen.dart lib/view U
- eventsfullscreen.dart lib/view 1, U
- mediafullscreen.dart lib/view U
- startupscreen.dart lib/view U
- wayfindingfullscreen.dart lib/view U
- aboutfabcardcontents.dart lib/view/... U
- eventscardcontents.dart li... U
- fabcard.dart lib/view/mainmenuseen 1, U
- mainmenuseen.dart lib/view/mainm... U
- mediacardcontents.dart lib/view/main... U
- wayfindingcardcontents.dart lib/view... U
- .gitignore linux U
- CMakeLists.txt linux U
- CMakeLists.txt linux/flutter U
- generated_plugin_registrant.cc linux... U
- generated_plugin_registrant.h linux/... U
- generated_plugins.cmake linux/flutter U
- CMakeLists.txt linux/runner U
- main.cc linux/runner U
- mv_application.cc linux/runner U

The main editor shows the following Dart code:

```
1 import 'dart:async';
2 import 'dart:convert';
3 import 'package:fab_app/view/eventsfullscreen.dart';
4 import 'package:flutter/material.dart';
5 import 'package:http/http.dart' as http;
6
7 Future<List> fetchList() async {
8   final response = await http.get(
9     Uri.parse('https://sitebuilder.warwick.ac.uk/sitebuilder2/api/rss/news.json?page=/fac/arts/dal/projects/fabapp/fabappcalendar/'),
10    headers: {'Accept': 'application/json'},
11  );
12
13  if (response.statusCode == 200) {
14    List rssDataList = jsonDecode(response.body)['items'];
15    //format list items
16    return rssDataList;
17  } else {
18    throw Exception('Failed to load data');
19  }
20 }
21
22 class EventsCardContents extends StatefulWidget {
23   const EventsCardContents({super.key});
24   @override
25   State<EventsCardContents> createState() => _EventsCardContentsState();
26 }
27
28 class _EventsCardContentsState extends State<EventsCardContents> {
29
30   void goToFullScreen() {
31     if(context.mounted == true)
32     {
33       Navigator.push(
34         context,
35         MaterialPageRoute<void>(
36           builder: (context) => const EventsFullScreen(),
37         ), // MaterialPageRoute
38       );
39     }
40   }
41
42   late Future<List> futureList;
43
44   int x = 0;
45   int itemCount = 0;
46
47   @override
48   void initState() {
49     super.initState();
50     futureList = fetchList();
```



Using Github Copilot AI to generate code

1. First structure your app in a sensible & manageable way.
2. Know what you want created, which widgets to use, how you want it to look.
3. Design prompts in manageable steps, building up the app in a logical way, so that you can understand what has been done, verify that it has been done correctly, and step back if not.
4. Create reusable code to do common tasks, in ways that you can manually customise.

Copilot use example 1

Most of the design coded manually

AI used to write the code to fetch the data

Quicker to use AI then learn from it, than trawling through docs

UNIVERSITY OF WARWICK

Study Research Business Alumni News Engagement Search Warwick

Faculty of Arts

Departments and Research Centres | Study | Outreach and Schools' Resources | Research and Impact | People | News and Events | Digital Arts & Humanities

Staff Resources

Projects FAB App project NOT REAL EVENTS! JUST FOR DEVELOPING THE APP

NOT REAL EVENTS! JUST FOR DEVELOPING THE APP

Monday 25 May - Sunday 21 Jun 2026

Show selected
Select all / none
Edit tags

Describe your event Add

Mon	Tue	Wed	Thu	Fri	Sat	Sun
25 May	26	27	28	29	30	31
			10am - 11am DAHL Workshop 4	9am - 11:59pm Massive party in the FAB FAB		
1 Jun	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21

Upcoming events Latest added events iCalendar

Tel: +44 (0)24 7652 4159
Email: Artsfaculty@warwick.ac.uk
The Faculty of Arts office is located in room 2.42,
Faculty of Arts Building

Staff Intranet

Page contact: Robert O'Toole
Last revised: Wed 27 May 2026

Calendar data is on a web page.

App used the live data from the web page.

Add an event online, appears in app.

iPhone 17 Pro
iOS 26.3

06:40

Events in the FAB

All Public Students Staff

DAHL Workshop 4
Tomorrow • 10:00 AM – 11:00 AM
Lorem ipsum. Lorem ipsum. Lorem ipsum.
Lorem ipsum. Lorem ipsum.

Massive party in the FAB
Fri, 29 May 2026 • 09:00 AM – 11:59 PM
FAB
Every space will be used. No studying allowed.

<< Menu

Copilot use example 2

feeds.csv

```
rss_url
https://rss.libsyn.com/shows/230621/destinations/1700315.xml
https://anchor.fm/s/102cac47c/podcast/rss
https://anchor.fm/s/ffea270c/podcast/rss
https://anchor.fm/s/ec49a6c8/podcast/rss
https://anchor.fm/s/fd53fdc4/podcast/rss
https://anchor.fm/s/f1213044/podcast/rss
https://anchor.fm/s/102ad2160/podcast/rss
```

This file is now online, I can easily edit the list of feeds

Read the list of podcast feeds from the file feeds.csv in the assets folder. For each url in the list in feeds.csv, retrieve data about the latest five podcast episodes. Build a single list combining all of the podcasts. Order the list by publication date of the episodes, so that it is sorted with the most recent episode first. Build a scrolling widget that shows each of the episodes in a card with the cover art of the episode, the title, the podcast channel, published date, and a link to the podcast on spotify. Display that widget on mediafullscreen.dart

Slightly incorrect. The code is mistaking the "generated_at" field for the published date of the episode. Correct this so that it uses the "published" field in the rss data about each episode.

```
class _MediaFullscreenState extends State<MediaFullscreen> {
  Widget build(BuildContext context) {
    // ...
    final episodes = snapshot.data ?? <PodcastEpisode>[];
    if (episodes.isEmpty) {
      return const Center(
        child: Text('No podcast episodes are available right now.'),
      ); // Center
    }

    return Scrollbar(
      thumbVisibility: true,
      child: ListView.separated(
        padding: const EdgeInsets.symmetric(horizontal: 12, vertical: 12),
        itemCount: episodes.length,
        separatorBuilder: (context, index) => const SizedBox(height: 12),
        itemBuilder: (context, index) {
          final episode = episodes[index];
          final formattedDate = DateFormat('EEE, d MMM yyyy').format(episode.pubDate.toLocal());
          final buttonLabel = episode.isSpotifyLink ? 'Open on Spotify' : 'Play episode';

          return Card(
            color: Colors.grey.shade100.withAlpha(480),
            shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(14)),
            elevation: 4,
            child: InkWell(
              onTap: () => _openLink(episode.linkUrl),
              borderRadius: BorderRadius.circular(14),
              child: Padding(
                padding: const EdgeInsets.all(14.0),
                child: Column(
                  crossAxisAlignment: CrossAxisAlignment.start,
                  children: [
                    Row(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        ClipRRect(
                          borderRadius: BorderRadius.circular(12),
                          child: episode.imageUrl.isNotEmpty
                            ? FutureBuilder<Uint8List?>({
                                future: _imageFuture(episode.imageUrl),
                                builder: (context, snapshot) {
                                  if (snapshot.hasData && snapshot.data != null) {
                                    return Image.memory(
                                      snapshot.data!,
                                      width: 110,
                                      height: 110,
                                      fit: BoxFit.cover,
                                    ); // Image.memory
                                  }

                                  if (snapshot.connectionState == ConnectionState.waiting) {
                                    return Container(
                                      width: 110,
                                      height: 110,
                                      color: Colors.grey.shade200,
                                      child: const Center(
                                        child: SizedBox(
                                          width: 24,
                                          height: 24,
                                          child: CircularProgressIndicator(strokeWidth: 2),
                                        ), // SizedBox
                                      ), // Center
                                    ); // Container
                                  }
                                }
                              ), // Container
                        ), // Container
                      ],
                    ), // Row
                    Text(episode.title, style: TextStyle(fontWeight: FontWeight.bold)),
                    Text(episode.channel),
                    Text(formattedDate),
                    Text(buttonLabel, style: TextStyle(color: Colors.black45)),
                  ],
                ), // Column
              ), // InkWell
            ), // Card
          ); // return Card
        },
      ), // ListView.separated
    ); // Scrollbar
  }
}
```

Where is the background color of the podcast episode cards specified?

Inspecting background color

Reviewing mediafullscreen.dart card definitions to locate the episode card background color.

Reviewed mediafullscreen.dart and clarified background color

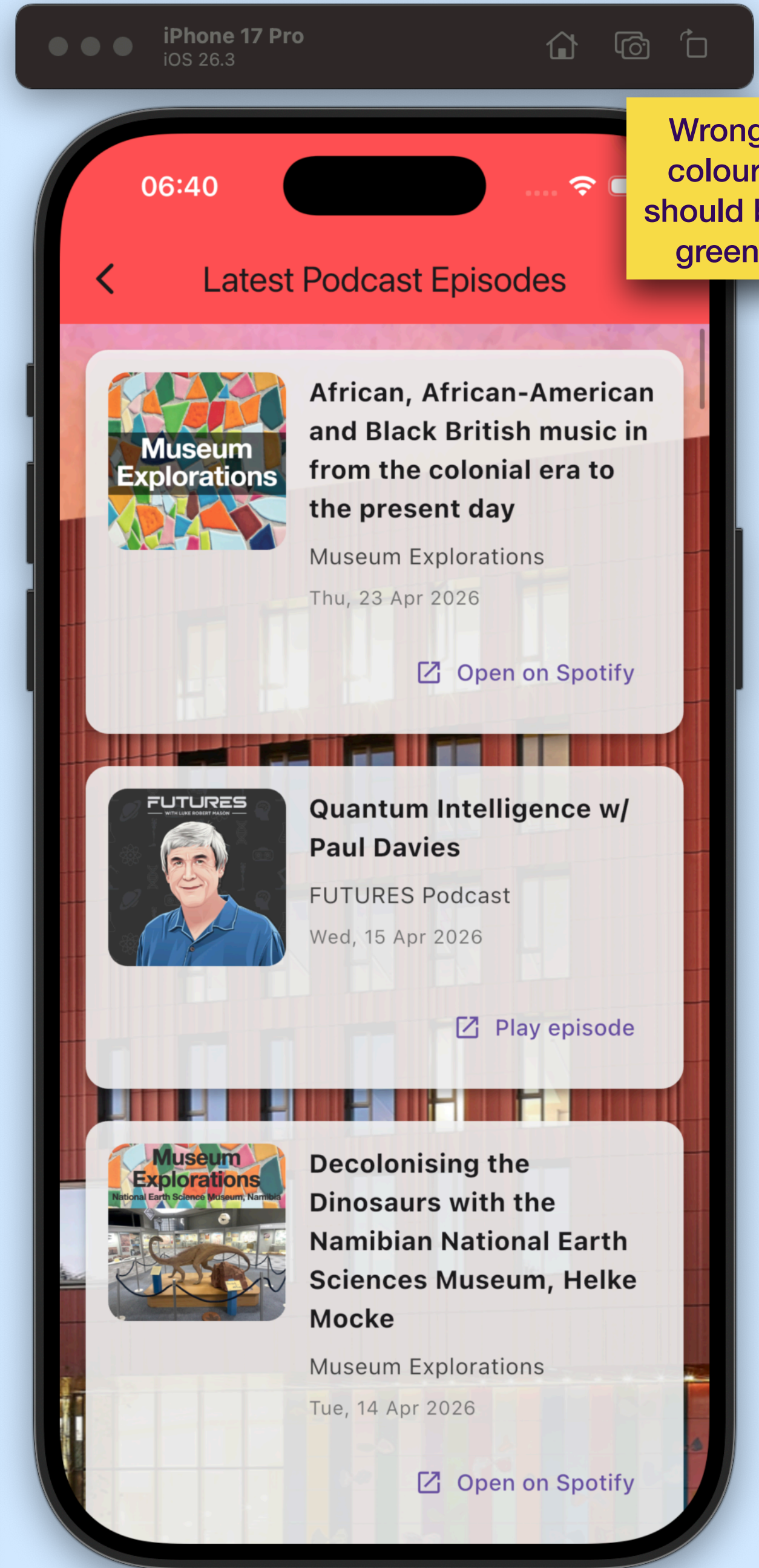
The episode cards don't have an explicit background color set in mediafullscreen.dart .

- The Card(...) widget at line ~70 is created without a color: property.
- That means it uses the default Material card background from the current theme.

So the card background is effectively coming from the app's default Card theme, not from a hardcoded color in mediafullscreen.dart .

Copilot based its design on the Events view I had manually crafted

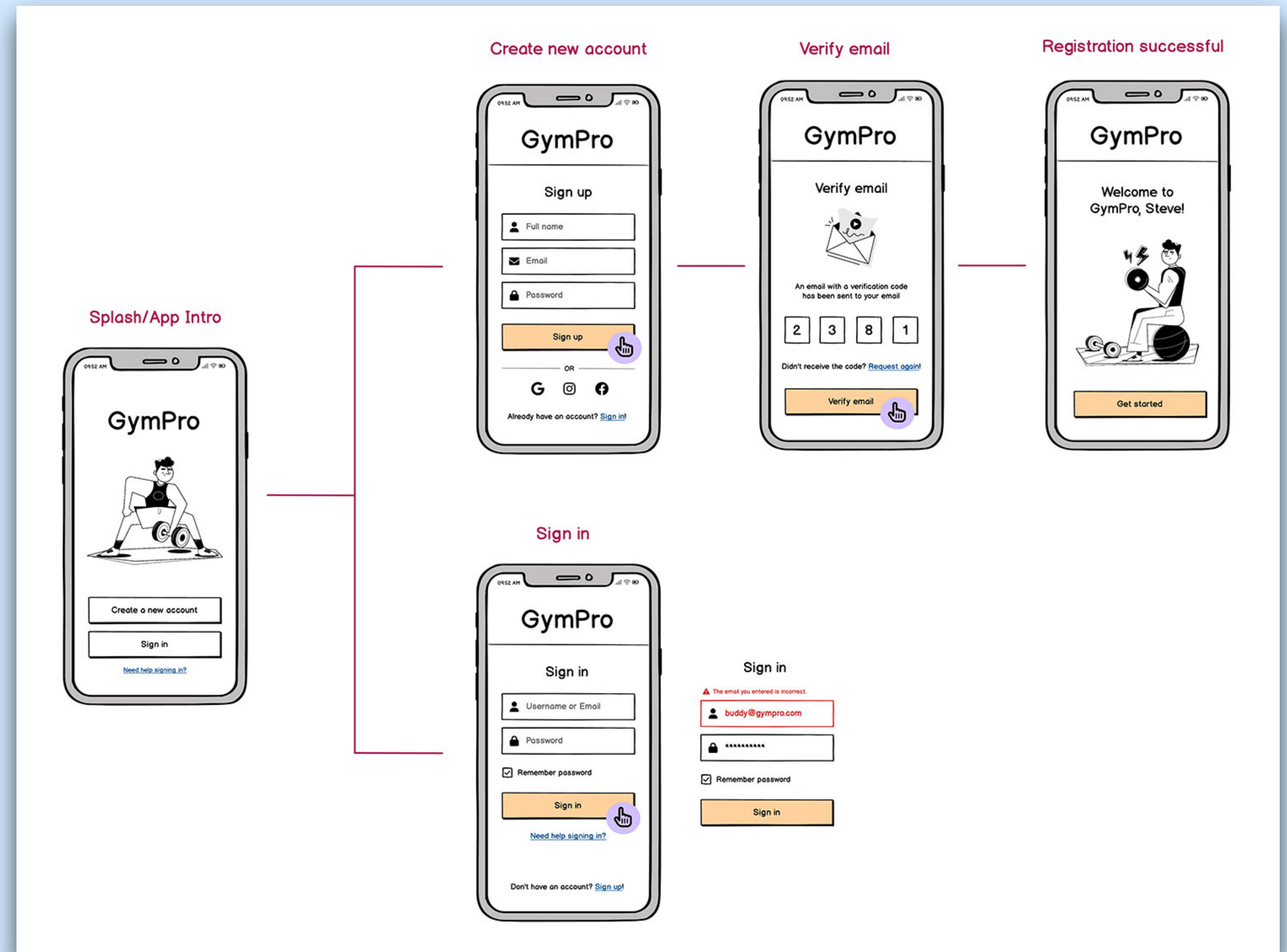
But I still had to tweak some things



Copilot can build widgets and apps based on wireframe sketches.

Not tried it yet.

Apparently it's fairly good.



Example from Balsamic wireframe app.

Many other design tools do this, including Miro

Structure and workflow of an app



Apps always start with main.dart

void main() is always the first code to run

```
main.dart • eventsfullscreen.dart 1 • scrollistfromjsonfeed.dart • aboutfabfullsc  
> main.dart > main  
1 import 'package:fab_app/model/fabmap.dart';  
2 import 'package:fab_app/model/location.dart';  
3 import 'package:fab_app/view/startupscreen.dart';  
4 import 'package:flutter/material.dart';  
5 import 'package:flutter/services.dart';  
6  
Run | Debug | Profile  
7 void main() {  
8   Location location = Location("FAB225");  
9   FABMap().addToMap(location);  
10  runApp(MaterialApp(  
11    title: 'FAB App',  
12    home: StartupScreen(),  
13  )); // MaterialApp  
14  SystemChrome.setPreferredOrientations([ DeviceOrientation.portraitUp ] );  
15 }
```

Initialising the data map of the FAB for the wayfinding feature

Tell it to load StartupScreen first as the apps home screen

This locks the screen orientation for the app to portrait, as adjusting layout to landscape is just too much hassle.

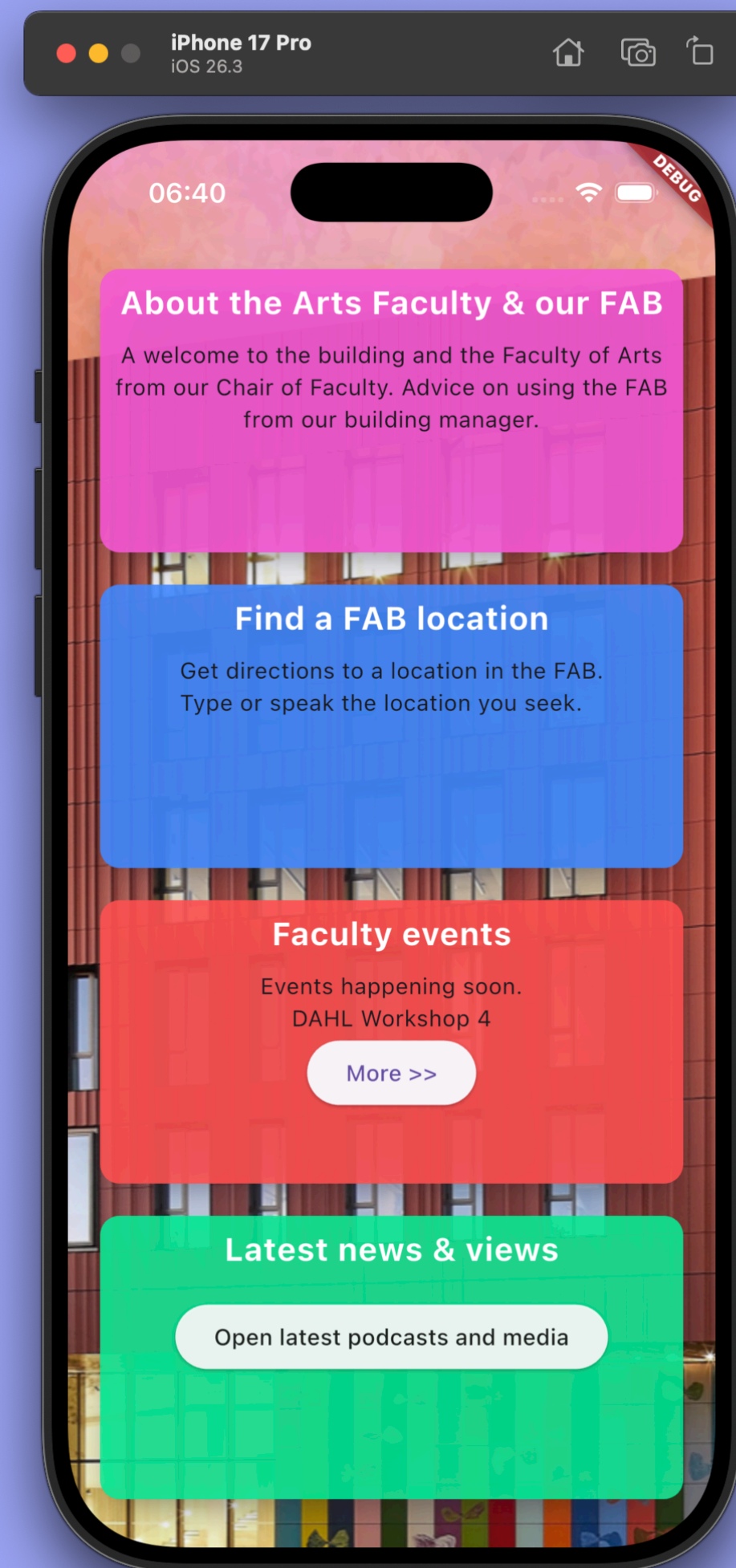
Within the code on a page, we can trigger the app to move to a different page.

```
if(context.mounted == true)
{
  Navigator.push(
    context,
    MaterialPageRoute<void>(
      builder: (context) => const EventsFullScreen(),
    ), // MaterialPageRoute
  );
}
```

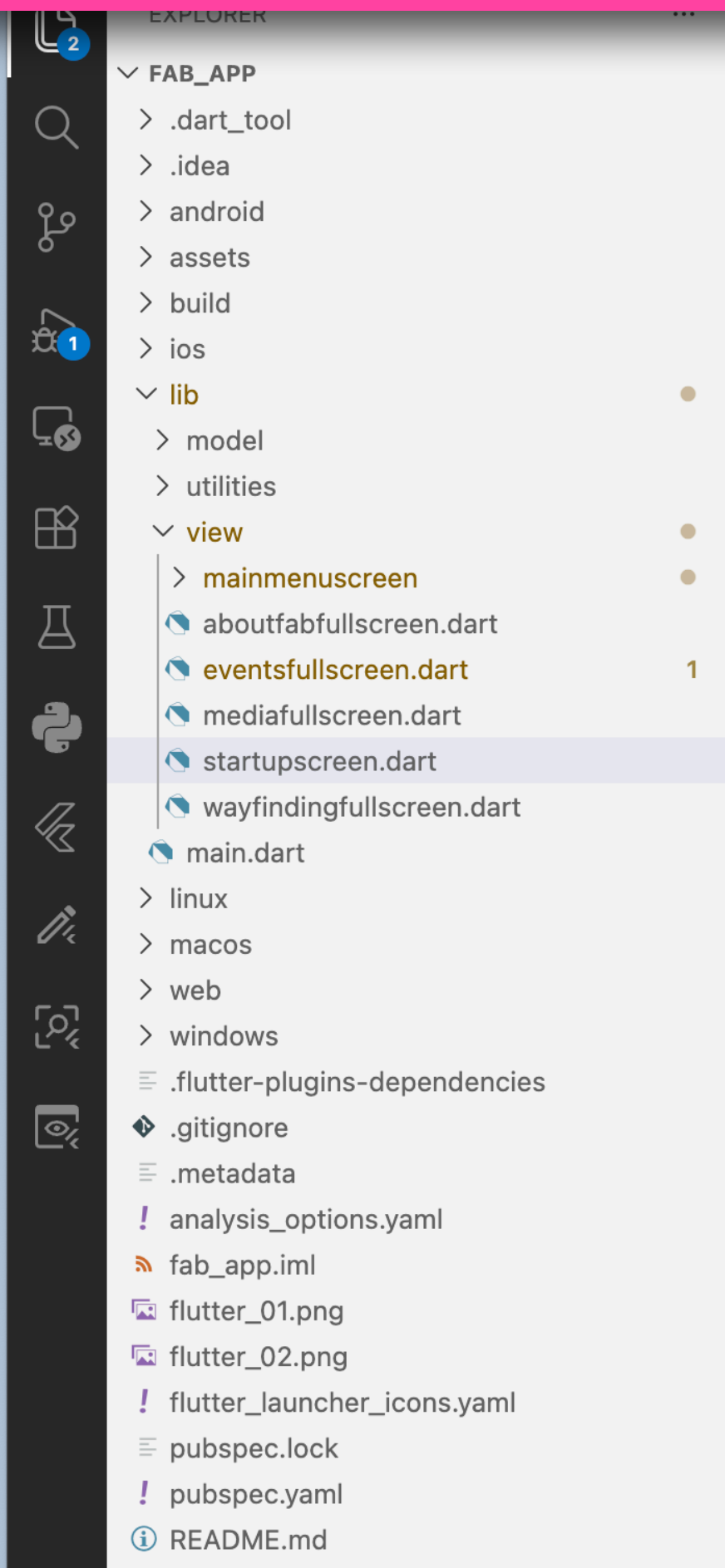
Opens a page called EventsFullScreen, showing a list of events from the FAB Calendar.

Page isn't the correct terminology, we are supposed to call them "routes", which are full-screen "widgets" that contain other widgets.

Structure and workflow of a page



We define classes, what they do, and how they should be used



```
lib > view > startupscreen.dart > _StartupScreenState
1  import 'package:fab_app/view/mainmenuscreen/mainmenuscreen.dart';
2  import 'package:flutter/material.dart';
3  import 'package:fab_app/utilities/audioplayer.dart';
4
5  class StartupScreen extends StatefulWidget {
6    const StartupScreen({super.key});
7    @override
8    State<StartupScreen> createState() => _StartupScreenState();
9  }
10
11 class _StartupScreenState extends State<StartupScreen> {
12   @override
13   void initState() {
14     super.initState();
15     onLoad(context);
16   }
17
18   void onLoad(BuildContext context) {
19     Future.delayed(const Duration(seconds:3), () {
20       if(context.mounted == true)
21       {
22         Navigator.push(
23           context,
24           MaterialPageRoute<void>(
25             builder: (context) => const MainMenuScreen(),
26           ), // MaterialPageRoute
27         );
28       }
29     }); // Future.delayed
30   }
31 }
32
33 @override
34 Widget build(BuildContext context) {
35   playAudio("FABStart.mp3");
36   return Scaffold(
37     //appBar: AppBar(leading: null, title: const Text('FAB App Loading')),
38     body: Container(
39       decoration: BoxDecoration(
40         image: DecorationImage(
41           image: AssetImage("assets/images/FABbackground.png"),
42           opacity: 1,
43           fit: BoxFit.cover,
44         ), // DecorationImage
45       ), // BoxDecoration
46       child: Stack(children: [
47         Center(
48           child: Opacity(opacity: 0.8,
49             child: Container(
50               color: Colors.white,
51               height: 200,
52             ), // Container
53           ), // Opacity
54         ), // Center
55       ]),
56     ),
57   );
```

StartupScreen extends StatefulWidget

StatefulWidget extends Widget (but with state, meaning that any changes made are persisted so they appear each time the user opens the page).

Default code runs when the page first completes building.

I've extended it with a custom function called OnLoad

Extending a class is a key feature of the Object Oriented Programming approach

```
lib > view > startupscreen.dart > _StartupScreenState
1  import 'package:fab_app/view/mainmenuescreen/mainmenuescreen.dart';
2  import 'package:flutter/material.dart';
3  import 'package:fab_app/utilities/audioplayer.dart';
4
5  class StartupScreen extends StatefulWidget {
6    const StartupScreen({super.key});
7    @override
8    State<StartupScreen> createState() => _StartupScreenState();
9  }
10
11 class _StartupScreenState extends State<StartupScreen> {
12   @override
13   void initState() {
14     super.initState();
15     onLoad(context);
16   }
17
18   void onLoad(BuildContext context) {
19     Future.delayed(const Duration(seconds:3), () {
20       if(context.mounted == true)
21       {
22         Navigator.push(
23           context,
24           MaterialPageRoute<void>(
25             builder: (context) => const MainMenuScreen(),
26           ), // MaterialPageRoute
27         );
28       }
29     }); // Future.delayed
30   }
31 }
32
33 @override
34 Widget build(BuildContext context) {
35   playAudio("FABStart.mp3");
36   return Scaffold(
37     //appBar: AppBar(leading: null, title: const Text('FAB App Loading')),
38     body: Container(
39       decoration: BoxDecoration(
40         image: DecorationImage(
41           image: AssetImage("assets/images/FABbackground.png"),
42           opacity: 1,
43           fit: BoxFit.cover,
44         ), // DecorationImage
45       ), // BoxDecoration
46       child: Stack(children: [
47         Center(
48           child: Opacity(opacity: 0.8,
49             child: Container(
50               color: Colors.white,
51               height: 200,
52             ), // Container
53           ), // Opacity
54         ), // Center
55         Center(child:Text("Welcome to the FAB App", style: TextStyle(fontSize: 60, color: Colors.black),), // Stack
56       ],) // Stack
57     ), // Container
58   ); // Scaffold
59 }
60
61
```

This is the structure of the page that is built and displayed.

Let's look at in in detail...

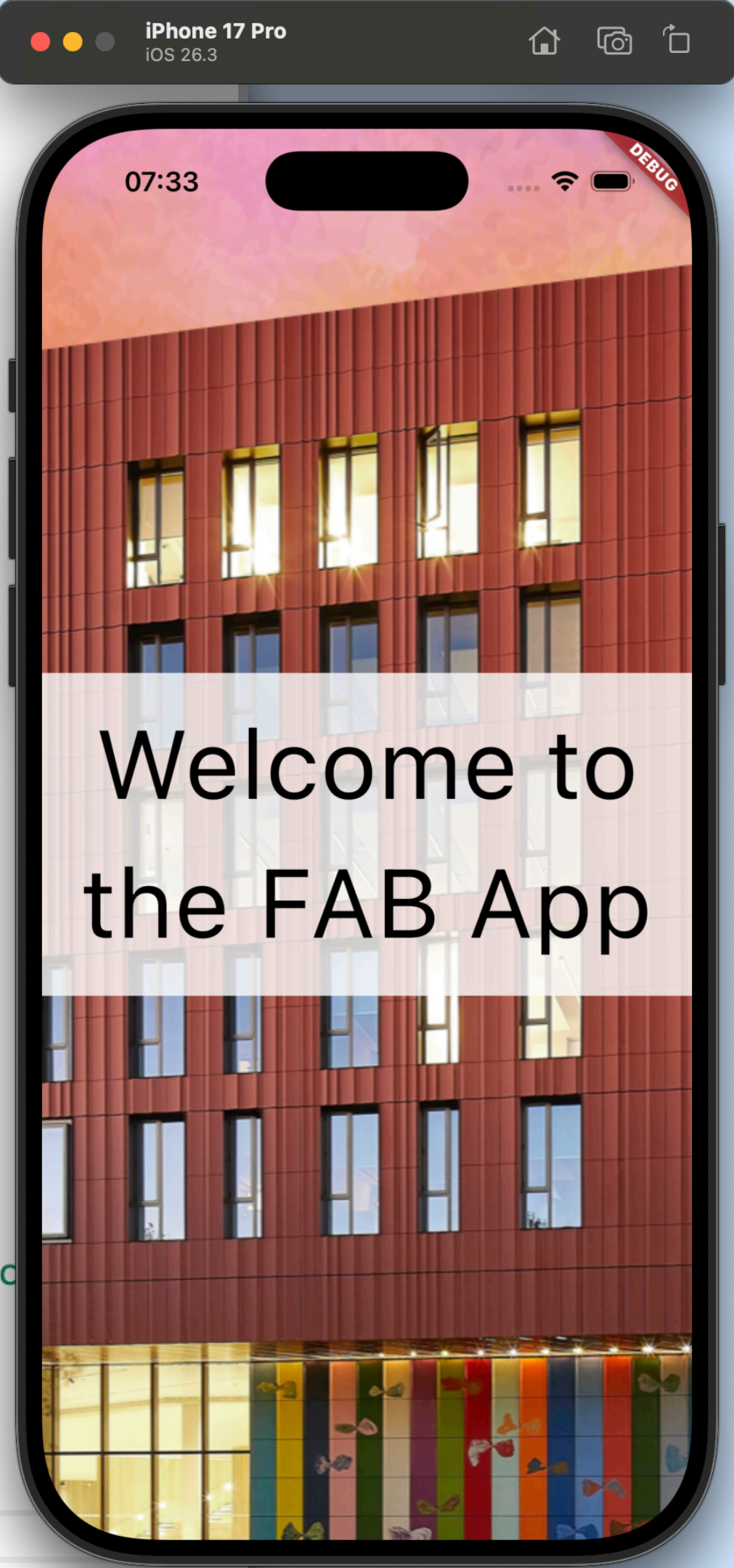
```
32
33 @override
34 Widget build(BuildContext context) {
35   playAudio("FABStart.mp3");
36   return Scaffold(
37     //appBar: AppBar(leading: null, title: const Text('FAB App Loading')),
38     body: Container(
39       decoration: BoxDecoration(
40         image: DecorationImage(
41           image: AssetImage("assets/images/FABbackground.png"),
42           opacity: 1,
43           fit: BoxFit.cover,
44         ), // DecorationImage
45       ), // BoxDecoration
46     child: Stack(children: [
47       Center(
48         child: Opacity(opacity: 0.8,
49           child: Container(
50             color: Colors.white,
51             height: 200,
52           ), // Container
53         ), // Opacity
54       ), // Center
55       Center(child: Text("Welcome to the FAB App", style: TextStyle(fontSize: 60, color: Colors.black), textAlign: TextAlign.c
56     ],) // Stack
57   ), // Container
58 ); // Scaffold
59 }
60
61 }
```

Plays an mp3 file as it is building saying "Welcome to the FAB"

Sets the background image (photo of the FAB)

The opaque white box across the centre

The text within the opaque white box



```
lib > view > mainmenuescreen > mainmenuescreen.dart > _MainMenuScreen
1 import 'package:fab_app/view/mainmenuescreen/aboutfabcardcontents.dart';
2 import 'package:flutter/material.dart';
3 import 'package:fab_app/utilities/audioplayer.dart';
4 import 'package:fab_app/view/mainmenuescreen/fabcard.dart';
5 import 'package:fab_app/view/mainmenuescreen/eventscardcontents.dart';
6 import 'package:fab_app/view/mainmenuescreen/mediacardcontents.dart';
7 import 'package:fab_app/view/mainmenuescreen/wayfindingcardcontents.dart';
8
9
10 class MainMenuScreen extends StatefulWidget {
11
12   const MainMenuScreen({super.key});
13
14   @override
15   State<MainMenuScreen> createState() => _MainMenuScreen();
16 }
17
18 class _MainMenuScreen extends State<MainMenuScreen> {
19   @override
20   void initState() {
21     super.initState();
22     playAudio("FABIntro.mp3");
23   }
24
25   Widget build(BuildContext context) {
26     return Scaffold(
27       //appBar: AppBar(toolbarHeight: 0, title: const Text('FAB App'), automaticallyImplyLeading: false),
28       body:
29         Container(
30           decoration: BoxDecoration(
31             image: DecorationImage(
32               image: AssetImage("assets/images/FABbackground.png"),
33               opacity: 1,
34               fit: BoxFit.cover,
35             ), // DecorationImage
36           ), // BoxDecoration
37           child:
38             Column(
39               crossAxisAlignment: CrossAxisAlignment.stretch,
40               children: [
41                 Container(child: Padding(padding: EdgeInsets.only(top: 70))),
42                 FABCard(color: const Color.fromARGB(255, 247, 90, 213), title: "About the Arts Faculty & our FAB", cardContents: AboutFABCardContents()),
43                 FABCard(color: Colors.blueAccent, title: "Find a FAB location", cardContents: WayfindingCardContents()),
44                 FABCard(color: Colors.redAccent, title: "Faculty events", cardContents: EventsCardContents()),
45                 FABCard(color: const Color.fromARGB(255, 18, 232, 150), title: "Latest news & views", cardContents: MediaCardContents()),
46                 Container(child: Padding(padding: EdgeInsets.only(top: 20))),
47               ],
48             ), // Column
49           ), // Container
50         ); // Scaffold
51   }
52 }
```

Import all of the classes we are using in this class

Plays an mp3 file as it is building saying "Where do you want to go today?"

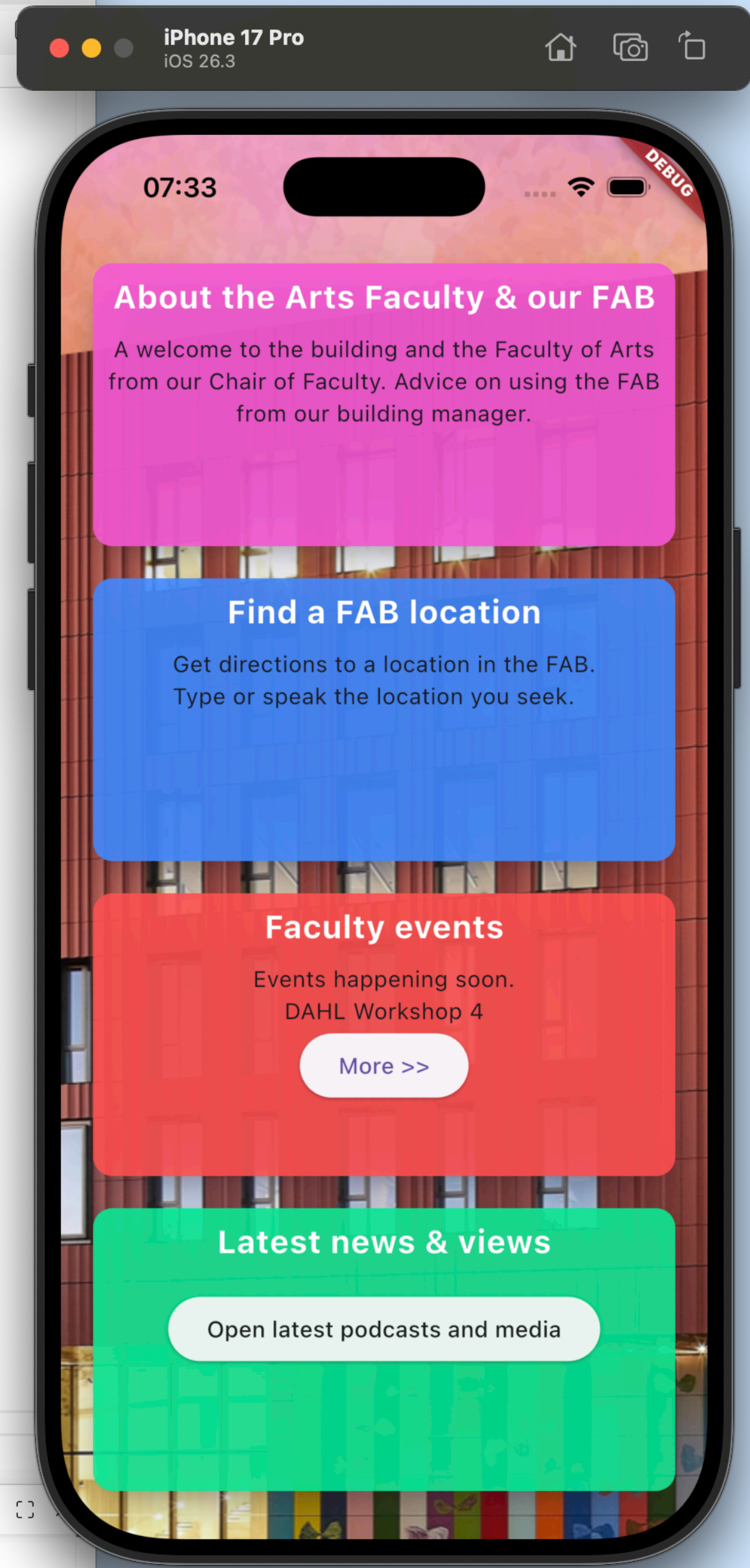
Not showing the AppBar as it covers the top of the background image too much

Loading the background again

Column layout, with convents stretched across the screen

Padding above and below

The four FABCard widgets each with a different colour, title, and contents



```
FABCard(color: Colors.redAccent, title: "Faculty events", cardContents: EventsCardContents(),),
```

To make a card, we must give it these properties

```
main.dart fabcard.dart 2 ●  
lib > view > mainmenuscreen > fabcard.dart > _FABCardState > build
```

```
14 class _FABCardState extends State<FABCard> {  
16   Widget build(BuildContext context) {  
22     child: Padding(  
23       padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 10),  
24       child: Container(  
25         decoration: BoxDecoration(  
26           boxShadow: [BoxShadow(  
27             color: const Color.fromRGBO(50, 50, 93, 0.25),  
28             blurRadius: 27,  
29             spreadRadius: -5,  
30             offset: Offset(0, 13)), // BoxShadow  
31           BoxShadow(  
32             color: Color.fromRGBO(0, 0, 0, 0.3),  
33             blurRadius: 16,  
34             spreadRadius: -8,  
35             offset: Offset(0, 8),  
36           ) // BoxShadow  
37         ],  
38         color: widget.color.withOpacity(0.9),  
39         borderRadius: BorderRadius.circular(12),  
40       ), // BoxDecoration  
41       child: Column(  
42         children: [  
43           Padding(  
44             padding: const EdgeInsets.symmetric(horizontal: 0, vertical: 7),  
45             child: Align(  
46               alignment: Alignment.topCenter,  
47               child: Text(widget.title, style: TextStyle(fontSize: 20, color: Colors.white, fontWeight: FontWeight.bold)),  
48             ), // Align  
49           ), // Padding  
50           Padding(padding: EdgeInsets.all(0.2),  
51             child: widget.cardContents,  
52           ), // Padding  
53         ],), // Column  
54       ), // Container  
55     ), // Padding  
56   ), // GestureDetector  
57 ); // Expanded  
58  
59
```

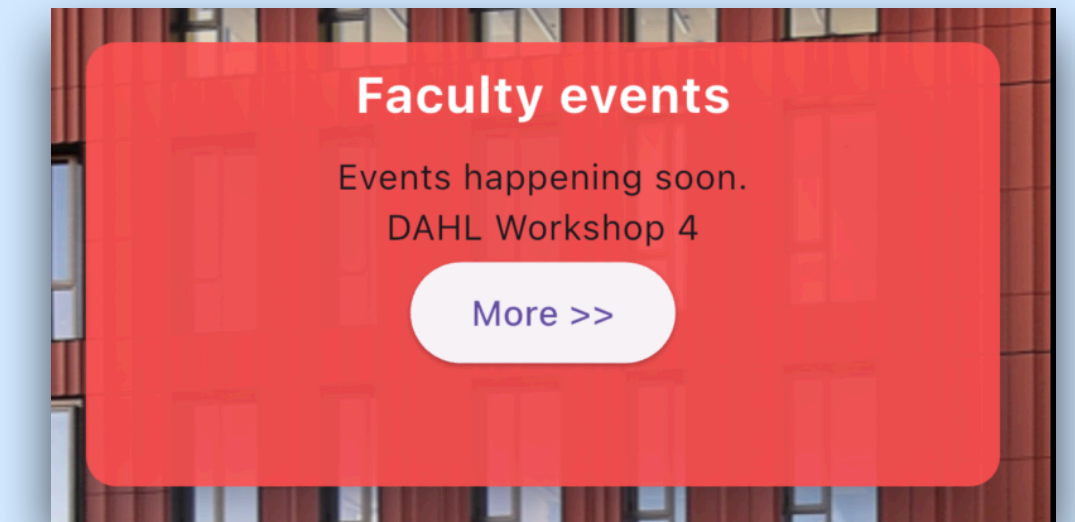
Padding around each card

The drop shadow effect

The color property we gave the widget, with opacity added

Card has rounded corners

Now it needs to make an instance of the class we specified as cardContents, and add that into the card. In this case, it is going to use EventsCardContents



Adding the text property we passed in as text on the card

EventsCardContents

The screenshot shows an IDE with the following structure:

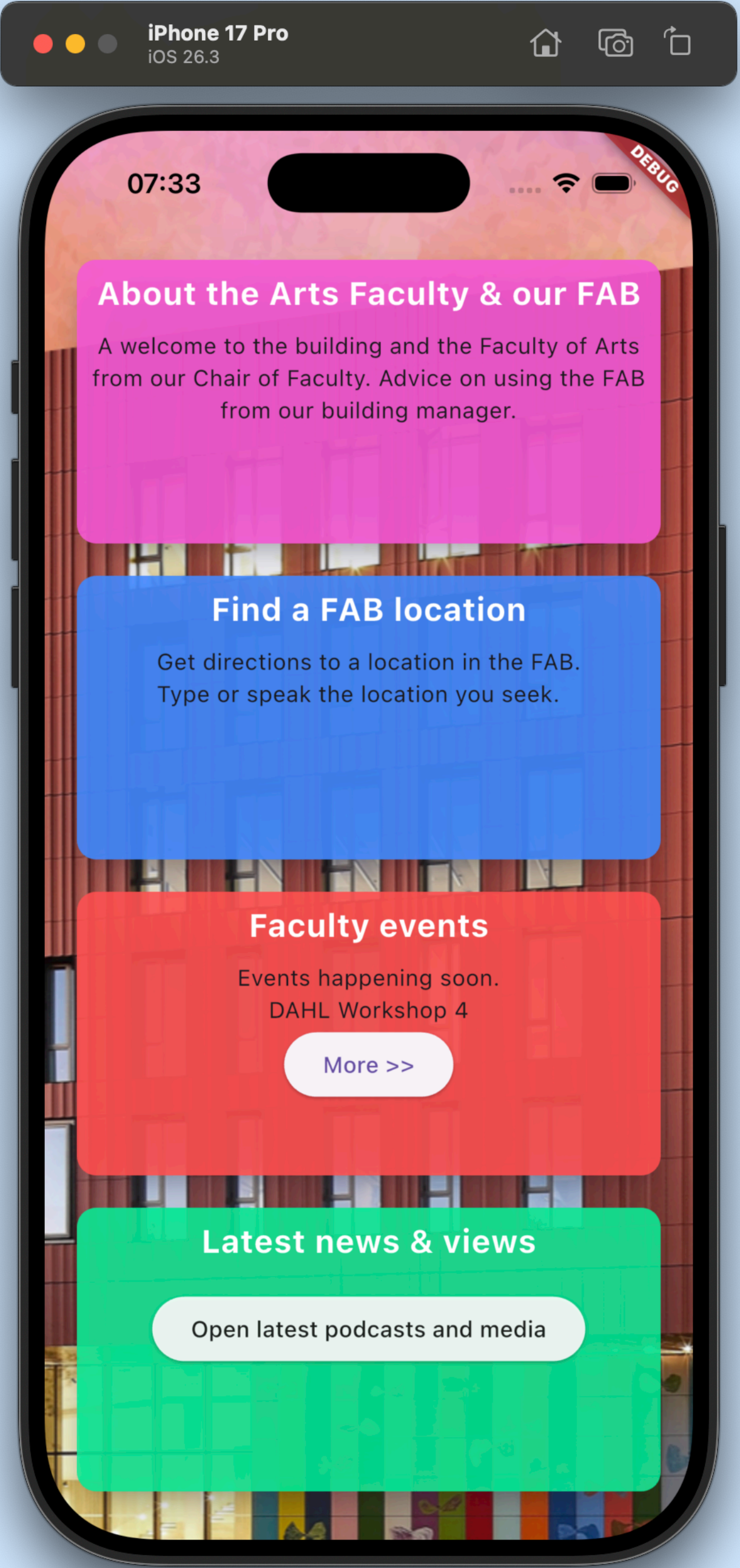
- EXPLORER:** A file tree for the 'FAB_APP' project, including folders like 'lib', 'view', and 'mainmenuscreen'. The file 'eventscardcontents.dart' is selected under 'mainmenuscreen'.
- LIBRARY:** A list of imported packages: 'dart:async', 'dart:convert', 'package:fab_app/view/eventsfullscreen.dart', 'package:flutter/material.dart', and 'package:http/http.dart' as http.
- fetchList:** A Future<List> function that fetches data from an RSS feed. It uses http.get() with a URL and headers, then parses the response and returns a list of items.
- EventsCardContents:** A StatefulWidget class that extends StatefulWidget. It has a constructor, an @override createState() method, and a build() method.
- _EventsCardContentsState:** A State class that extends State<EventsCardContents>. It has a goToFullScreen() method, a late Future<List> futureList, and an initState() method.
- Widget build:** The build method returns a Container widget containing a Column widget. The Column contains a Text widget and a FutureBuilder widget. The FutureBuilder widget is configured with future: futureList and a builder function that checks if data is available and returns a Text widget with the title or 'No events found'.

```
1 import 'dart:async';
2 import 'dart:convert';
3 import 'package:fab_app/view/eventsfullscreen.dart';
4 import 'package:flutter/material.dart';
5 import 'package:http/http.dart' as http;
6
7 Future<List> fetchList() async {
8   final response = await http.get(
9     Uri.parse('https://sitebuilder.warwick.ac.uk/sitebuilder2/api/rss/news.json?page=/fac/arts/dal/projects/fabapp/fabappcalendar/'),
10    headers: {'Accept': 'application/json'},
11  );
12
13  if (response.statusCode == 200) {
14    List rssDataList = jsonDecode(response.body)['items'];
15    //format list items
16    return rssDataList;
17  } else {
18    throw Exception('Failed to load data');
19  }
20 }
21
22 class EventsCardContents extends StatefulWidget {
23   const EventsCardContents({super.key});
24   @override
25   State<EventsCardContents> createState() => _EventsCardContentsState();
26 }
27
28 class _EventsCardContentsState extends State<EventsCardContents> {
29
30   void goToFullScreen() {
31     if(context.mounted == true)
32     {
33       Navigator.push(
34         context,
35         MaterialPageRoute<void>(
36           builder: (context) => const EventsFullScreen(),
37         ), // MaterialPageRoute
38       );
39     }
40   }
41
42   late Future<List> futureList;
43
44   int x = 0;
45   int itemCount = 0;
46
47   @override
48   void initState() {
49     super.initState();
50     futureList = fetchList();
51   }
52
53   @override
54   Widget build(BuildContext context) {
55     return Container(
56       child: Column(
57         children: [
58           Text("Events happening soon.", textAlign: TextAlign.center,),
59           FutureBuilder<List>({
60             future: futureList,
61             builder: (context, snapshot)
62             {
63               if (snapshot.hasData == true) {
64                 try
65                 {
66                   itemCount = snapshot.data!.length;
67                   return(Text(snapshot.data![x]['title']));
68                 }
69                 catch (e)
70                 {
71                   return(Text("No events found"));
72                 }
73               }
74               else
75               {
76                 return(Text("No events found"));
77               }
78             }, // FutureBuilder
79             ElevatedButton(onPressed: goToFullScreen, child: Text("More >>")),
80           ]), // Column
81     ); // Container
82   }
83 }
84
85
```

PROBLEMS: 1 Unnecessary instance of 'Container'. dart(avoid_unnecessary_containers) [Ln 55, Col 12]

Widget Tree tool

The screenshot shows the Flutter DevTools Inspector interface. The top part displays the Widget Tree, which is a hierarchical view of the widget hierarchy. The tree starts with a root widget, followed by MaterialApp, StartupScreen, Scaffold, Container, Stack, Center, Opacity, Container, Center, and Text. Below this, the MainMenuScreen widget is expanded, showing Scaffold, Container, Column, Container, Padding, FABCard, Expanded, GestureDetector, Padding, Container, Column, Padding, Align, Text, Padding, AboutFABCardConte..., Align, Text, FABCard, Expanded, GestureDetector, Padding, Container, and Column. The bottom part of the Inspector shows the Widget Properties for the selected Center widget, with a visual representation of the widget and its dimensions: h=874.0 and w=402.0.



iPhone & iPad simulator using Apple Xcode

test main

Running test on iPad (A16)

PROJECT

- test

TARGETS

- test

Info Build

Config

Debug

Release

Use

- Parallel

Localiz

Localiza

Base

English

Touch Bar

Bring All to Front

test — test.xcodeproj

Based on Configuration File

No Configurations Set

No Configurations Set

When using schemes)

alized

alized

No Filter Results

Filter

Filter

Filter

Filter

Simulators


- iOS
 - iPad (A16)
 - iPad Air 11-inch (M3)
 - iPad Air 13-inch (M3)
 - iPad Pro 11-inch (M5)
 - iPad Pro 13-inch (M5)
 - iPad mini (A17 Pro)
 - iPhone 17
 - iPhone 16e
 - iPhone 17
 - iPhone 17 Pro**
 - iPhone 17 Pro Max**
 - iPhone Air

No Filter Results

iPhone 17 Pro Max

iOS 26.3.1 (23D8133) Show run destination: Automatic

Model: iPhone 17 Pro Max
Identifier: 3968638F-B8E9-425E-B98A-5F709C26D4CD



Development tools: Run, Stop, Attach, Download, Upload, List, Refresh, Share, Connect, Test (test)

Filter: Auto | Filter

iPhone 17 Pro Max

iOS 26.3.1 (23D8133)
 Model: iPhone 17 Pro Max
 Identifier: 3968638F-B8E9-425E-B98A-5F709C26D4CD

Show run destination: Automatic

Devices: **Simulators**

iOS

- iPad (A16)
- iPad Air 11-inch (M3)

test main

test iPad (A16) Running test on iPad (A16)

Filter

Recent

- ✓ iPad (A16) 26.3.1
- iPhone 17 26.3.1

Mac

- My Mac

visionOS

- visionOS 26.2 Not Installed **Get**

Build

- Any iOS Device (arm64)
- ✓ Any iOS Simulator Device (arm64, x86_64)
- Any Mac (arm64, x86_64)

iOS Simulators

- ✓ iPad (A16) 26.3.1
- iPad Air 11-inch (M3) 26.3.1
- iPad Air 13-inch (M3) 26.3.1
- iPad Pro 11-inch (M5) 26.3.1
- iPad Pro 13-inch (M5) 26.3.1
- ✓ iPad mini (A17 Pro) 26.3.1
- iPhone 17 26.3.1
- iPhone 16e 26.3.1
- iPhone 17 26.3.1
- iPhone 17 Pro 26.3.1
- iPhone 17 Pro Max 26.3.1
- iPhone Air 26.3.1

Manage Run Destinations...

Clear Recent

No Filter Results

PROJECT: test

TARGETS: test

Info

Running test on iPad (A16)

on Configuration File

Configurations Set

Configurations Set

Using schemes)

test

Filter

Auto

Filter

Filter

FLUTTER DEVTOOLS: INSPECTOR

main.dart × eventsfullscreen

lib > main.dart > ...

```
1 import 'package:fab_app/mc
2 import 'package:fab_app/mc
3 import 'package:fab_app/vi
4 import 'package:flutter/ma
5 import 'package:flutter/se
6
7 void main() {
8   Location location = Loca
9   FABMap().addToMap(location);
10  runApp(MaterialApp(
11    title: 'FAB App',
12    home: StartupScreen(),
13  )); // MaterialApp
14  SystemChrome.setPreferredOrientations([ DeviceOrientation.portraitUp ]);
15 }
16
17
```

Run | Debug | Profile

Disconnected

Run a new debug session to reconnect.

Select a device to use

- iPhone 17 Pro CB11FE82-BE8D-4DCE-B167-821D0DC52FAF - mobile Available Devices
- Current Device
- Chrome chrome - web
- iPad (A16) 09DB55D8-DA32-435F-9673-9E42EE33150C - mobile
- iPhone 17 Pro Max 3968638F-B8E9-425E-B98A-5F709C26D4CD - mobile
- macOS macos - desktop
- Robert iPhone 12 00008101-000C58192209001E - mobile
- Medium Phone emulator-5554 - mobile
- Create Android emulator Offline Emulators

Update

mediafullscreen.c

CHAT

PODCAST EPISODES DISPLAY WIDGET

Where is the background color of the podcast episode cards specified?

Inspecting background color

Reviewing mediafullscreen.dart card definitions to locate the episode card background color.

Reviewed mediafullscreen.dart and clarified background color

The episode cards don't have an explicit background color set in mediafullscreen.dart .

- The Card(...) widget at line ~70 is created without a color: property.
- That means it uses the default Material card background from the current theme.

So the card background is effectively coming from the app's default Card theme, not from a hardcoded color in mediafullscreen.dart .

PROBLEMS 18 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Filter (e.g. text, !exclude, \e...)

```
-[WFIsoledShortcutRunner init]_block_invoke Sandbox extensions acquired
Indexing for request: <WFToolKitIndexingRequest: 0x6000017008c0>, changeset: .none, priority: 31
Finished in 0.021377s
-[WFIsoledShortcutRunner unaliveProcess] Releasing sandbox extensions
-[WFIsoledShortcutRunner init] Taking sandbox extensions for execution
-[WFIsoledShortcutRunner init]_block_invoke Sandbox extensions acquired
Indexing for request: <WFToolKitIndexingRequest: 0x6000017081c0>, changeset: .none, priority: 31
Finished in 0.023317s
-[WFIsoledShortcutRunner unaliveProcess] Releasing sandbox extensions
```

main.dart

Describe what to build

Auto

Local Default Approvals

Ln 17, Col 1 Spaces: 2 UTF-8 LF Dart iPhone 17 Pro (ios simulator)

fab_app

main.dart eventsfullscreen.dart 1 x scrolllistfromjsonfeed.dart aboutfabfullscreen.dart podcast_feed.dart mediafullscreen.dart

VARIABLES

- > Locals
- > Globals

```
lib > view > eventsfullscreen.dart > _EventsFullScreen > returnToMenu
1 import 'package:fab_app/utilities/scrolllistfromjsonfeed.dart';
2 import 'package:flutter/material.dart';
3
4 class EventsFullScreen extends StatefulWidget {
5   const EventsFullScreen({super.key});
6
7   @override
8   State<EventsFullScreen> createState() => _EventsFullScreen();
9 }
10
11 class _EventsFullScreen extends State<EventsFullScreen>
12 {
13
14   int _selectedIndex = 0;
15
16   void returnToMenu(){
17     if(!context.mounted == true)
18     {
19       Navigator.pop(context);
20     }
21   }
22
23   VoidCallback filterList(int index)
24   {
25     return () {
26       setState(() {
27         _selectedIndex = index;
28       });
29     };
30   }
31
32   @override
33   void initState() {
34     super.initState();
35   }
36
37   @override
38   Widget build(BuildContext context) {
39     return Scaffold(
40       body: Container(
41         decoration: BoxDecoration(
42           image: DecorationImage(
43             image: AssetImage("assets/images/FABbackgr
44             opacity: 1,
45             fit: BoxFit.cover,
```

PROBLEMS 18 OUTPUT DEBUG CONSOLE TERMINAL PORTS

CALL STACK

BREAKPOINTS

- All Exceptions
- Uncaught Exceptions
- eventsfullscreen.dart lib/view

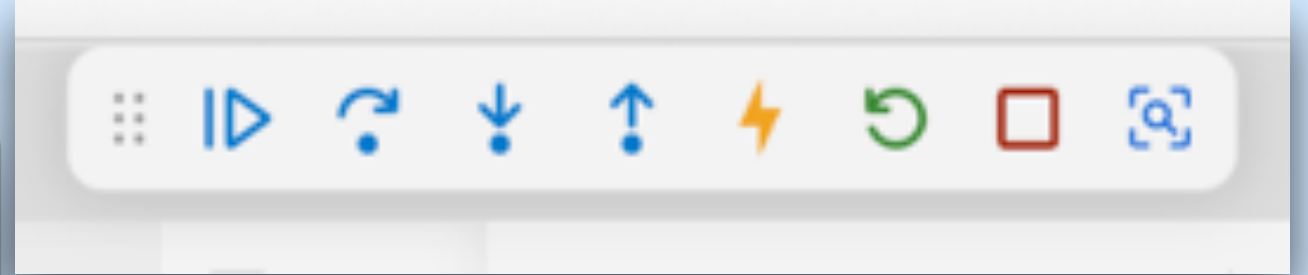
VARIABLES

- > Locals
- > Globals

WATCH

lib > view > eventsfullscreen.dart > _EventsFullScreen > returnToMenu

```
1 import 'package:fab_app/utilities/scrolllistfromjsonfeed.da
2 import 'package:flutter/material.dart';
3
4 class EventsFullScreen extends StatefulWidget {
5   const EventsFullScreen({super.key});
6
7   @override
8   State<EventsFullScreen> createState() => _EventsFullScre
9 }
10
11 class _EventsFullScreen extends State<EventsFullScreen>
12 {
13
14   int _selectedIndex = 0;
15
16   void returnToMenu(){
17     if(!context.mounted == true)
18     {
19       Navigator.pop(context);
20     }
21   }
22
23   VoidCallback filterList(int index)
```



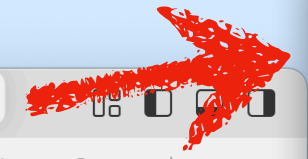
Where is the background color of the podcast episode cards specified?

Inspecting background color

Reviewing mediafullscreen.dart card definitions to locate the episode card background color.

Reviewed mediafullscreen.dart and clarified background color

The episode cards don't have an explicit background color set in mediafullscreen.dart



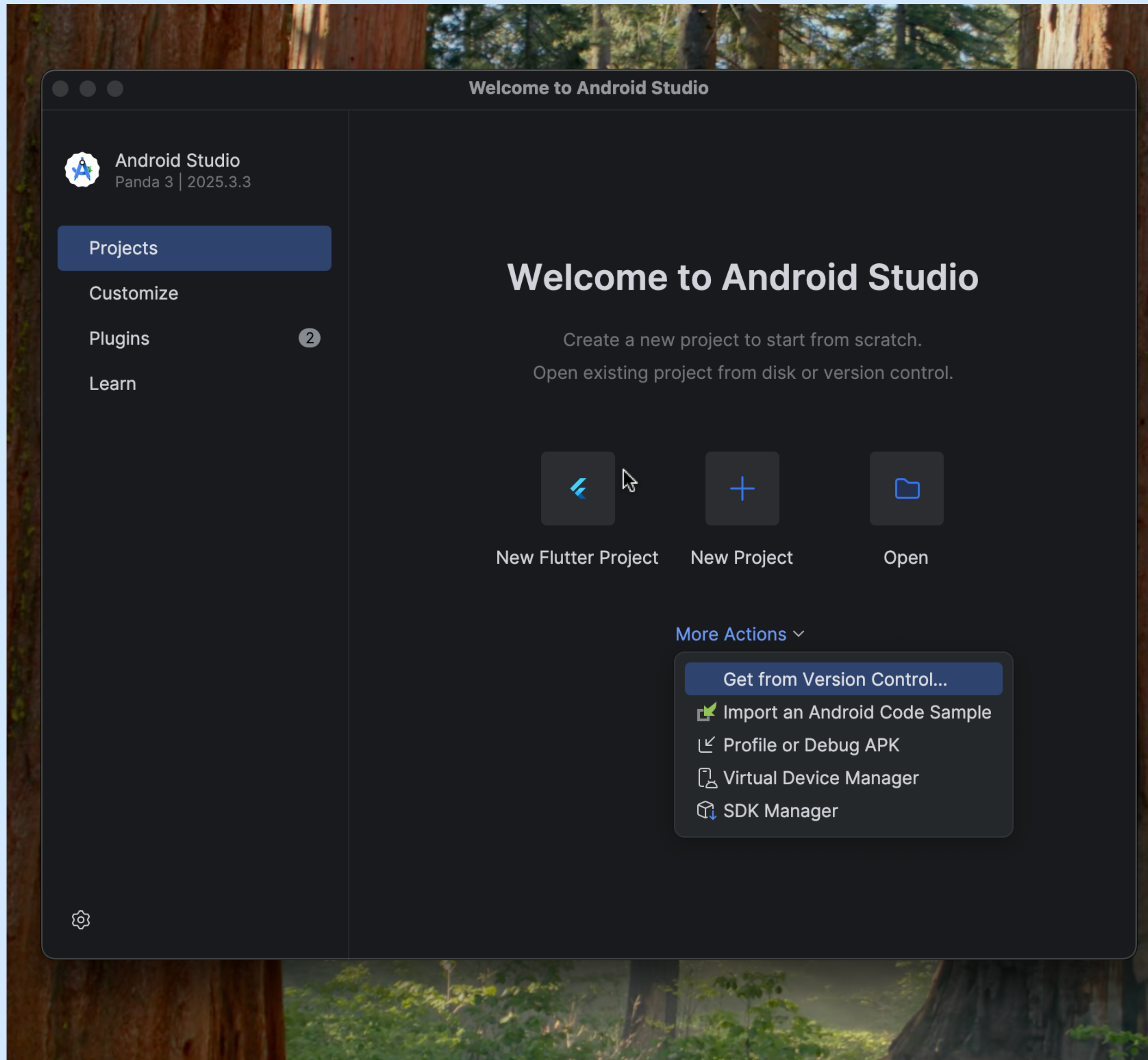
fab_app — dartvm --resolved_executable_name=/Users/robertotoole/flut...

```
[robertotoole@Robert-OTOole-M3Pro fab_app % flutter run
Connected devices:
sdk gphone16k arm64 (mobile) • emulator-5554 •
android-arm64 • Android 17 (API 37) (emulator)
Robert iPhone 12 (mobile) • 00008101-000C58192209001E • ios
• iOS 26.4.2 23E261
iPhone 17 Pro (mobile) • CB11FE82-BE8D-4DCE-B167-821D0DC52FAF • ios
• com.apple.CoreSimulator.SimRuntime.iOS-26-3 (simulator)
iPhone 17 Pro Max (mobile) • 3968638F-B8E9-425E-B98A-5F709C26D4CD • ios
• com.apple.CoreSimulator.SimRuntime.iOS-26-3 (simulator)
iPad (A16) (mobile) • 09DB55D8-DA32-435F-9673-9E42EE33150C • ios
• com.apple.CoreSimulator.SimRuntime.iOS-26-3 (simulator)
macOS (desktop) • macos •
darwin-arm64 • macOS 15.7.3 24G419 darwin-arm64
Chrome (web) • chrome •
web-javascript • Google Chrome 148.0.7778.169

No wireless devices were found.

[1]: sdk gphone16k arm64 (emulator-5554)
[2]: Robert iPhone 12 (00008101-000C58192209001E)
[3]: iPhone 17 Pro (CB11FE82-BE8D-4DCE-B167-821D0DC52FAF)
[4]: iPhone 17 Pro Max (3968638F-B8E9-425E-B98A-5F709C26D4CD)
[5]: iPad (A16) (09DB55D8-DA32-435F-9673-9E42EE33150C)
[6]: macOS (macos)
[7]: Chrome (chrome)
Please choose one (or "q" to quit): █
```

Android simulator using Android Studio



Welcome to Android Studio



Android Studio
Panda 3 | 2025.3.3

Projects

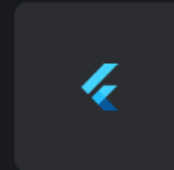
Customize

Plugins 2

Learn

Welcome to Android Studio

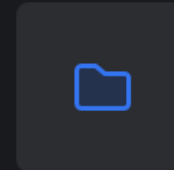
Create a new project to start from scratch.
Open existing project from disk or version control.



New Flutter Project



New Project



Open

More Actions ▾

Get from Version Control...

Import an Android Code Sample

Profile or Debug APK

Virtual Device Manager

SDK Manager



Device Manager

Medium Phone
Android API 37.0 | arm64

Name	API	Type
Medium Phone	37.0	Virtual

Add Device

Form Factor

- Phone
- Tablet
- Wear OS
- Desktop
- TV
- Automotive
- XR

Show obsolete device profiles

Search for a device by name

Name	Play	API	Width	Height	Density
Small Phone	▶	24+	720	1280	320 dpi
Medium Phone	▶	24+	1080	2400	420 dpi
Resizable (Experimental)		34+	1080	2400	420 dpi
Pixel 10 Pro XL	▶	36+	1344	2992	480 dpi
Pixel 10 Pro Fold	▶	36+	2076	2152	390 dpi
Pixel 10 Pro	▶	36+	1280	2856	480 dpi
Pixel 10	▶	36+	1080	2424	420 dpi
Pixel 9a	▶	35+	1080	2424	420 dpi
Pixel 9 Pro XL	▶	35+	1344	2992	480 dpi
Pixel 9 Pro Fold	▶	35+	2076	2152	390 dpi
Pixel 9 Pro	▶	35+	1280	2856	480 dpi
Pixel 9	▶	35+	1080	2424	420 dpi
Pixel 8a	▶	34+	1080	2400	420 dpi
Pixel 8 Pro	▶	34+	1344	2992	480 dpi
Pixel 8	▶	34+	1080	2400	420 dpi
Pixel Fold	▶	34+	2208	1840	420 dpi
Pixel 7a	▶	34+	1080	2400	420 dpi
Pixel 7 Pro	▶	33+	1440	3120	560 dpi
Pixel 7	▶	33+	1080	2400	420 dpi
Pixel 6a	▶	33+	1080	2400	420 dpi
Pixel 6 Pro		31+	1440	3120	560 dpi
Pixel 6		31+	1080	2400	420 dpi

New hardware profile... Import hardware profile... Cancel Previous Next Finish

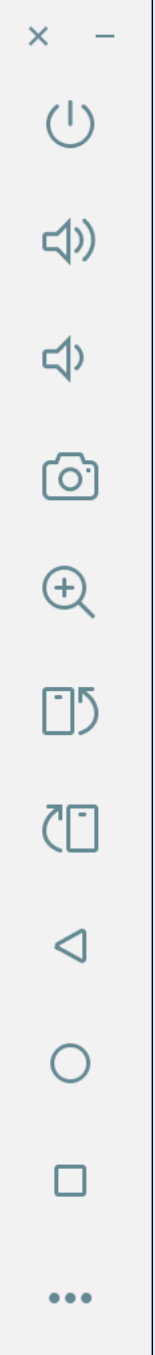
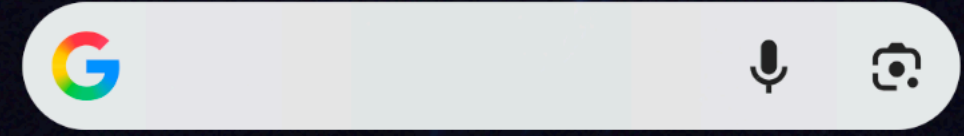
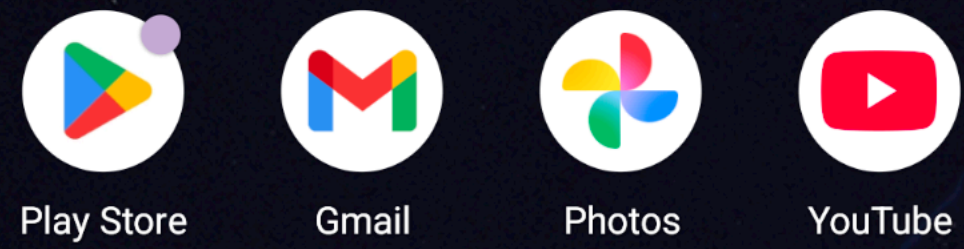
Welcome to Android Studio

Android Studio
Panda 3 | 2025.3.3

- Projects
- Customize
- Plugins 2
- Learn

Create a new project to start from scratch.
Open existing project from disk or version control.

Tue, May 26



fab_app — dartvm --resolved_executable_name=/Users/robertotoole/flut...

```
[robertotoole@Robert-0Toole-M3Pro fab_app % flutter run ]
Connected devices:
sdk gphone16k arm64 (mobile) • emulator-5554 •
android-arm64 • Android 17 (API 37) (emulator)
Robert iPhone 12 (mobile) • 00008101-000C58192209001E • ios
• iOS 26.4.2 23E261
iPhone 17 Pro (mobile) • CB11FE82-BE8D-4DCE-B167-821D0DC52FAF • ios
• com.apple.CoreSimulator.SimRuntime.iOS-26-3 (simulator)
iPhone 17 Pro Max (mobile) • 3968638F-B8E9-425E-B98A-5F709C26D4CD • ios
• com.apple.CoreSimulator.SimRuntime.iOS-26-3 (simulator)
iPad (A16) (mobile) • 09DB55D8-DA32-435F-9673-9E42EE33150C • ios
• com.apple.CoreSimulator.SimRuntime.iOS-26-3 (simulator)
macOS (desktop) • macos •
darwin-arm64 • macOS 15.7.3 24G419 darwin-arm64
Chrome (web) • chrome •
web-javascript • Google Chrome 148.0.7778.169

No wireless devices were found.

[1]: sdk gphone16k arm64 (emulator-5554)
[2]: Robert iPhone 12 (00008101-000C58192209001E)
[3]: iPhone 17 Pro (CB11FE82-BE8D-4DCE-B167-821D0DC52FAF)
[4]: iPhone 17 Pro Max (3968638F-B8E9-425E-B98A-5F709C26D4CD)
[5]: iPad (A16) (09DB55D8-DA32-435F-9673-9E42EE33150C)
[6]: macOS (macos)
[7]: Chrome (chrome)
Please choose one (or "q" to quit): █
```

About the Arts Faculty & our FAB

A welcome to the building and the Faculty of Arts from our Chair of Faculty. Advice on using the FAB from our building manager.

Find a FAB location

Get directions to a location in the FAB.
Type or speak the location you seek.

Faculty events

Events happening soon.
DAHL Workshop 4

[More >>](#)

Latest news & views

[Open latest podcasts and media](#)

