Api Integration

Workshop #01

Flutter Developer Bootcamp

# **Purpose**

This code demonstrates how to integrate API calls into a Flutter application and handle asynchronous operations to fetch and display data.

# **Problem**

In this workshop, we're diving into Flutter application that seamlessly integrates with a REST API. The code revolves around a StatefulWidget named HomePage, complemented by its State class \_HomePageState. It orchestrates the fetching and display of data retrieved from a specified endpoint. Key components include \_data for storing fetched content and \_isLoading for managing loading states. Through the fetchData() method, the app communicates with the API, updating the UI accordingly. Finally, the UI presents a Scaffold widget featuring an AppBar and a dynamic body that switches between a loading indicator and a ListView populated with fetched data.

# **How to Solve**

1. Checkout the workshop from Git Repo:

git clone -b <user-branch> <repo-URL>

2. Open the root folder inside VS Code

3. To build the app click the run option in the main method { }

4. You want to integrate the api into the code .

5. <https://jsonplaceholder.typicode.com/posts>

# **You Will Achieve**

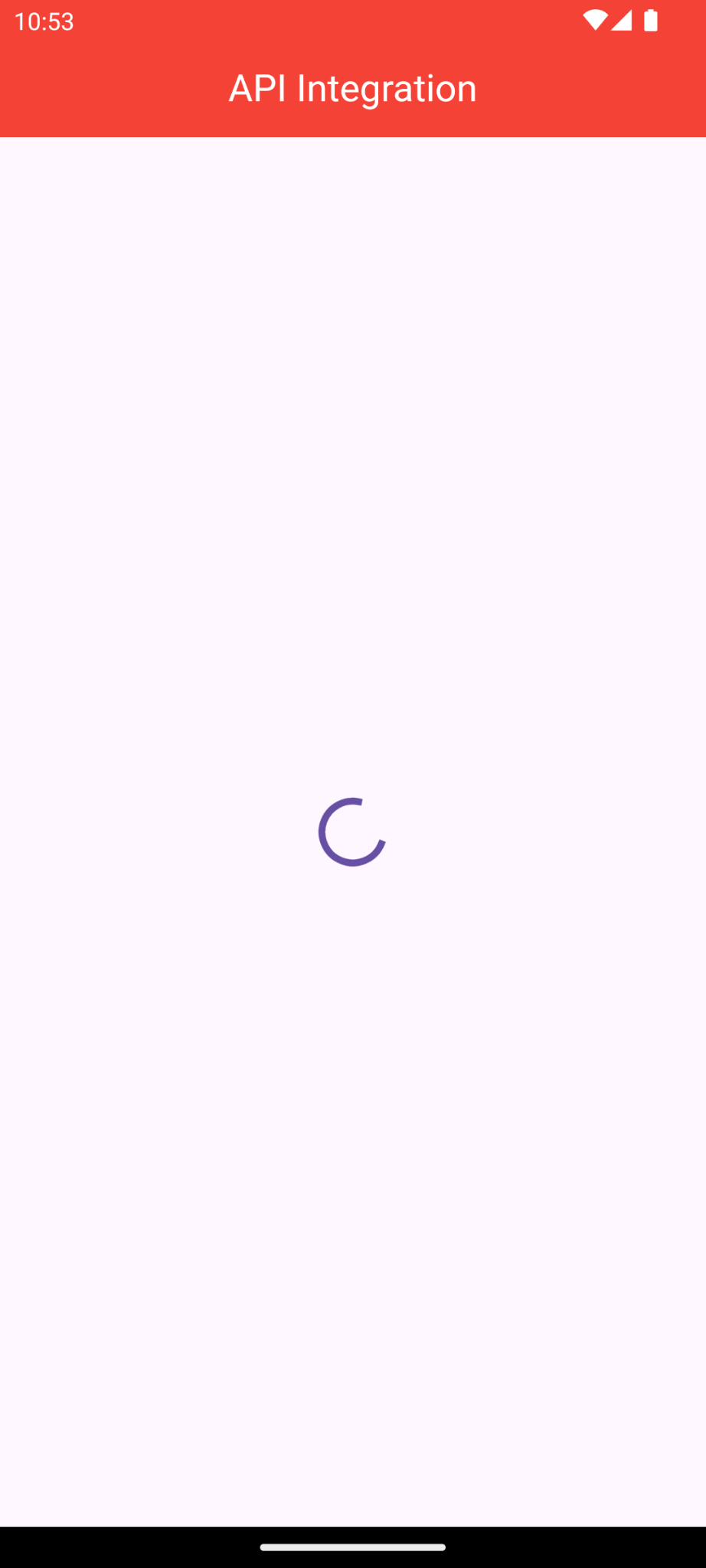
This Flutter code integrates with an API to fetch data and display it in a ListView. Let's break down what it achieves:

* Fetch Data: It makes an HTTP GET request to https://jsonplaceholder.typicode.com/posts to retrieve a list of posts.
* Display Data: Once the data is fetched successfully, it populates a ListView with ListTile widgets, displaying the title and body of each post.
* Loading Indicator: While the data is being fetched, a CircularProgressIndicator is displayed at the centre of the screen.
* Error Handling: If the HTTP request fails (i.e., if the status code is not 200), it throws an exception with the message 'Failed to load data'.

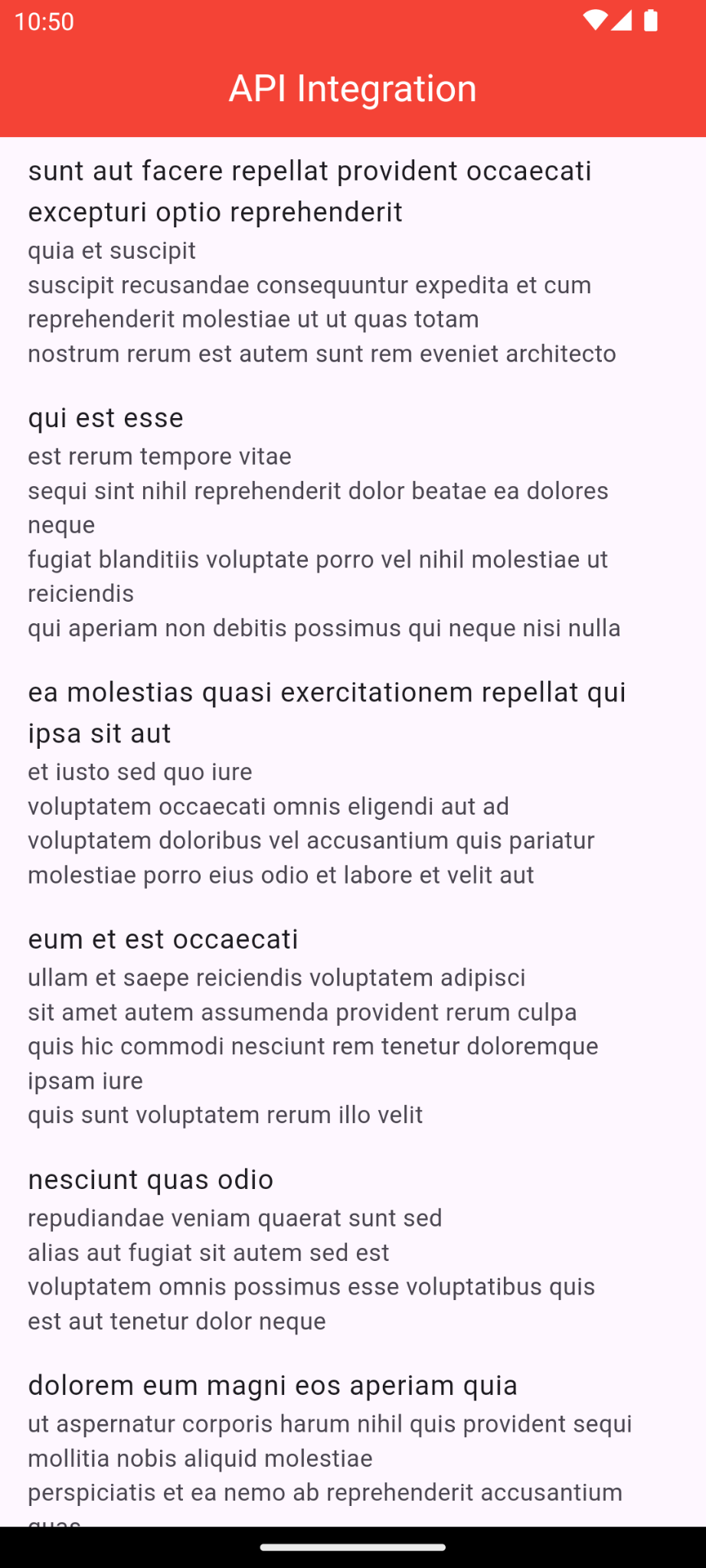
Overall, this code provides a basic implementation of fetching and displaying data from an API in a Flutter app. It demonstrates important concepts such as asynchronous programming with Futures, state management with StatefulWidget, and UI updates with setState().

# **Screenshots**

## **Before implementation (without API)**



## **After implementation (With API)**



# **How to submit your workshop**

Push your project back to the same git branch using command:

<command name>

# **Happy Coding!**