CPA Assignment Submission

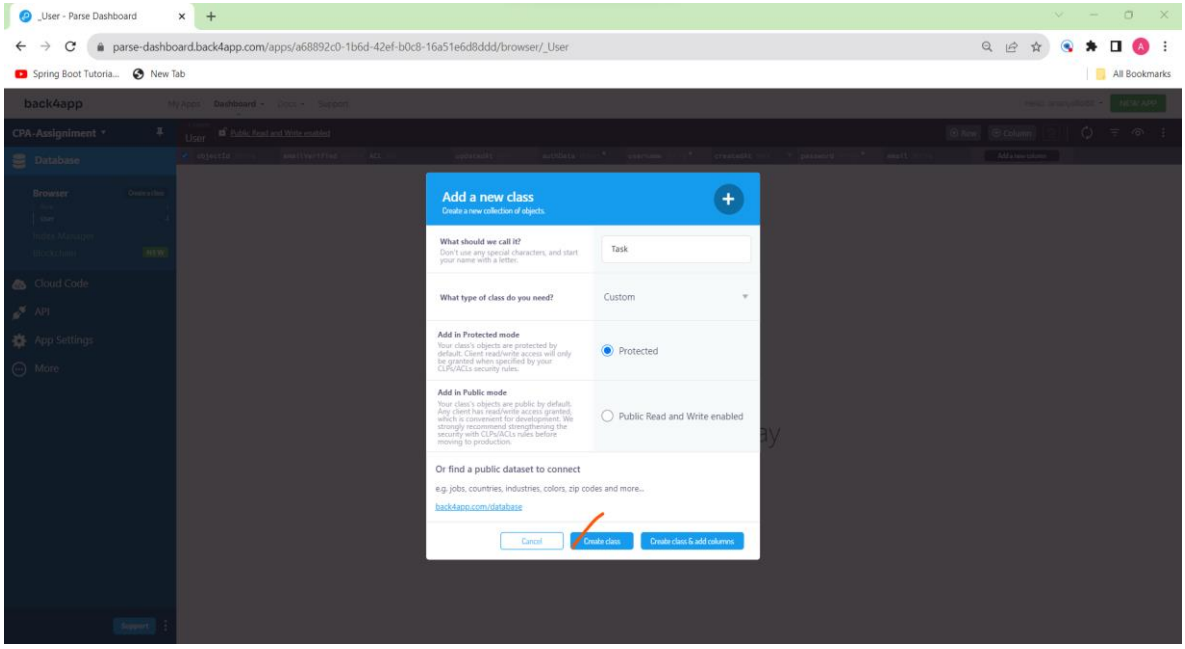
Submit the link to your GitHub repository where you've hosted your Flutter app code.

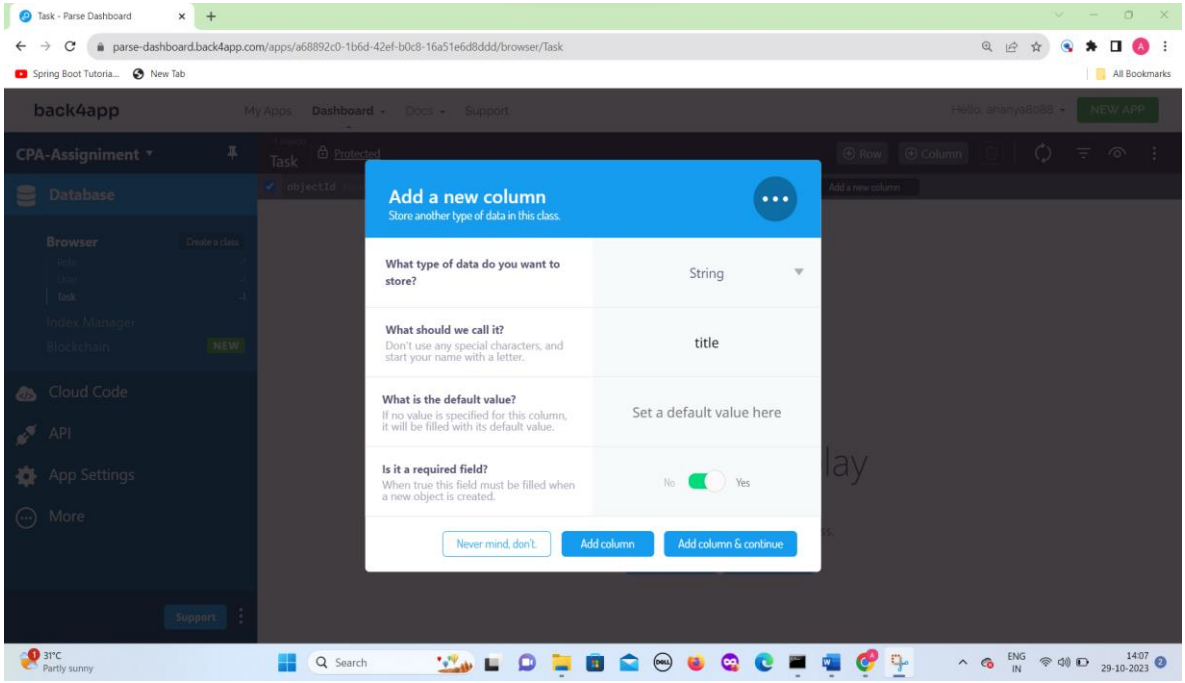
Github URL : https://github.com/sowmyasantoshi4/cpa-assign.git

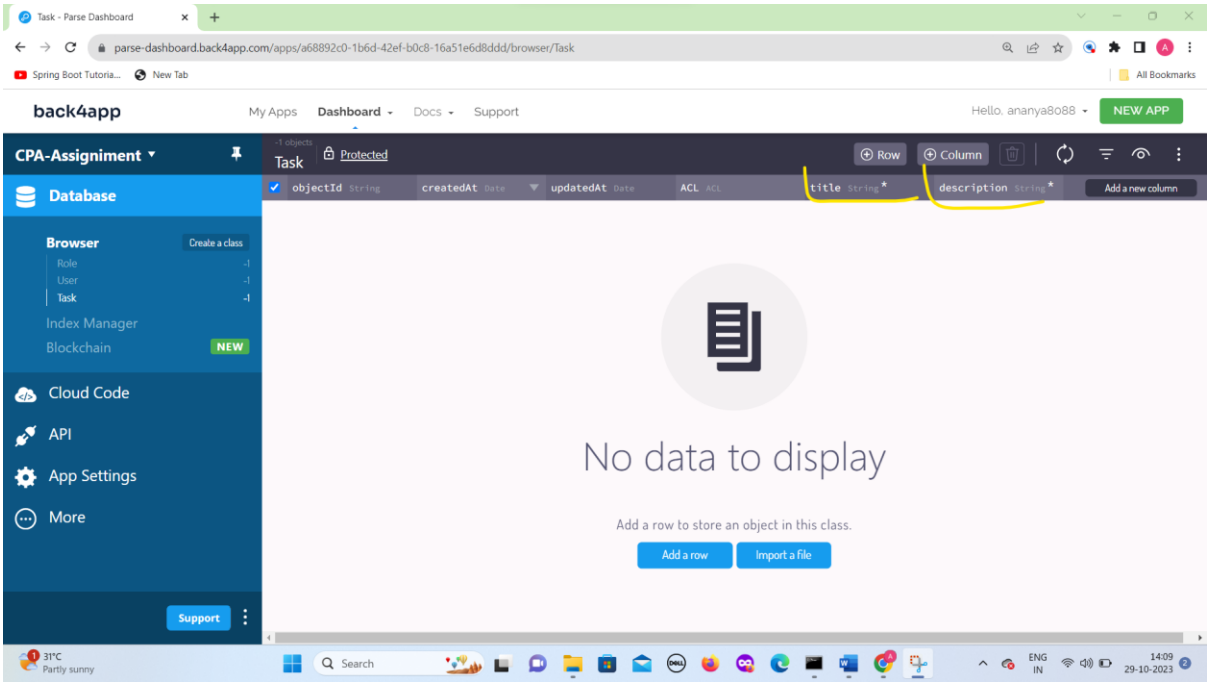
Include a short README explaining how to set up and run your app.

SetUp :

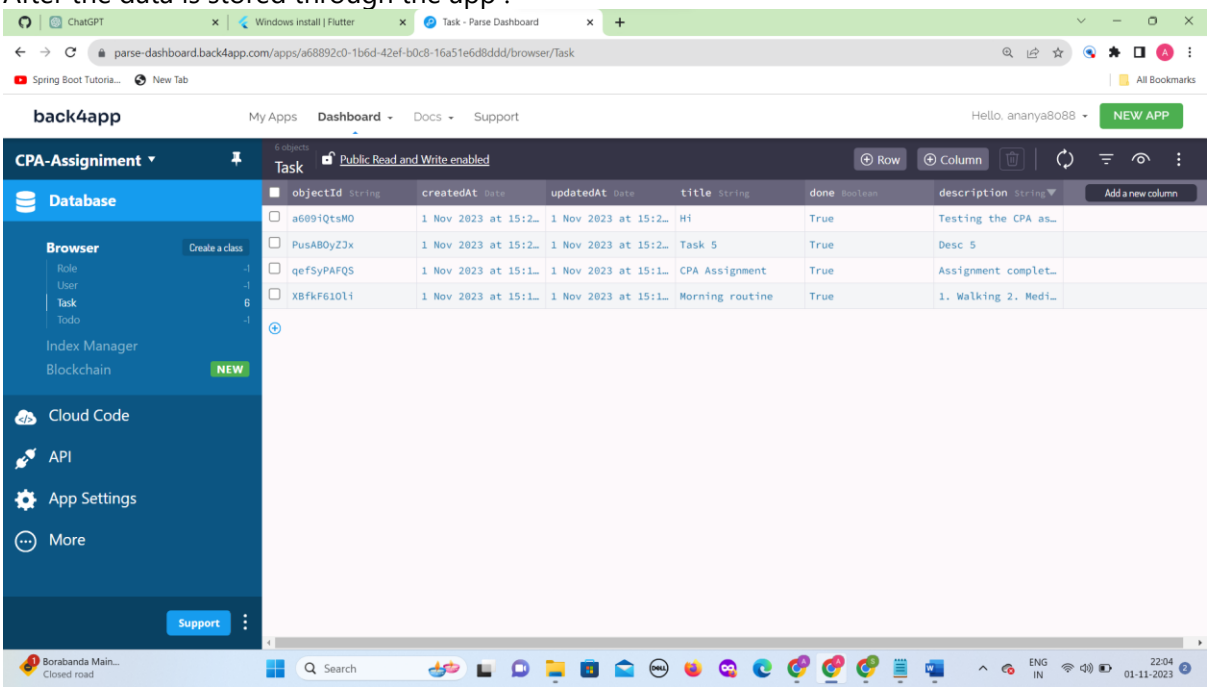
1. Create account in Back4app : https://www.back4app.com/
2. Create a class and columns :







3. After the data is stored through the app :



4. Download and install Flutter in **windows** using below steps :
   - Goto : https://docs.flutter.dev/get-started/install/windows and download the .zip
   - Extract into a folder
   - Set path in environment variables like : D:flutter_path
   - Goto command prompt and give the following commands :
     - where flutter dart // shows where the flutter path
     - flutter doctor // check everything is OK or not
     - flutter doctor --android-licenses
5. After successful installation, create flutter project by using the following command in command prompt :
   flutter create app_name
6. In command prompt, move to the newly created app folder.

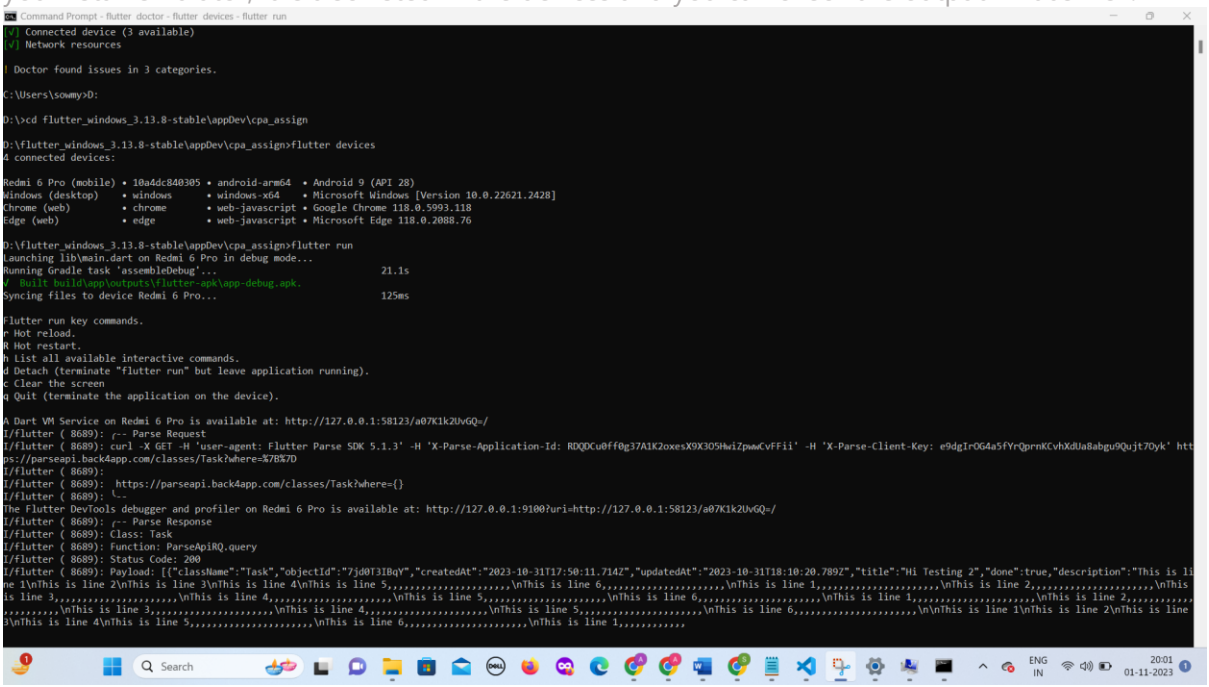
7. Now, open the **pubspec.yaml** (inside the new folder created) using any editor and add the required dependencies :

```
dependencies:
  flutter:
    sdk: flutter


  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.2
  parse_server_sdk: any
```

8. In the command prompt, run the following command to get the dependencies into the project : flutter pub get
9. After the dependencies, add the code to your **main.dart** file (this is the entry point); present inside the lib folder.
10. After done adding the code, goto command prompt and run the following command to check the output :
    flutter devices // gives all the possible ways to check output like browsers, mobile if connected with developer options enabled

```
D:\flutter_windows_3.13.8-stable\appDev\cpa_assign>flutter devices
4 connected devices:

Redmi 6 Pro (mobile) • 10a4dc840305 • android-arm64  • Android 9 (API 28)
Windows (desktop)    • windows      • windows-x64    • Microsoft Windows [Version 10.0.22621.2428]
Chrome (web)         • chrome       • web-javascript • Google Chrome 118.0.5993.118
Edge (web)           • edge         • web-javascript • Microsoft Edge 118.0.2088.76
```

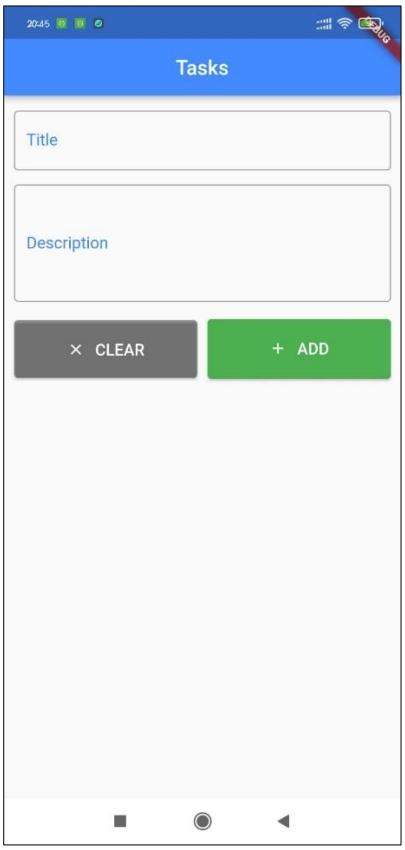
11. Run the application to check the output using the following command :
    flutter run // when you type this and enter, you are provided with the list of options asking to choose the device to run.. I chose edge browser once, and mobile once.. when you choose mobile, it generates .apk file and installs in your mobile. You can run and check it. If you install emulator, it is also listed in the devices and you can check the output in it as well.



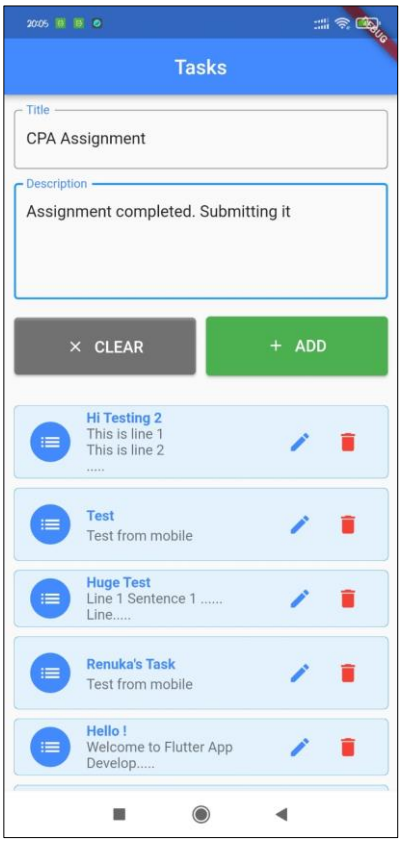
12. Enter the option you wish to.

13. Check the mobile for app installed and view it.

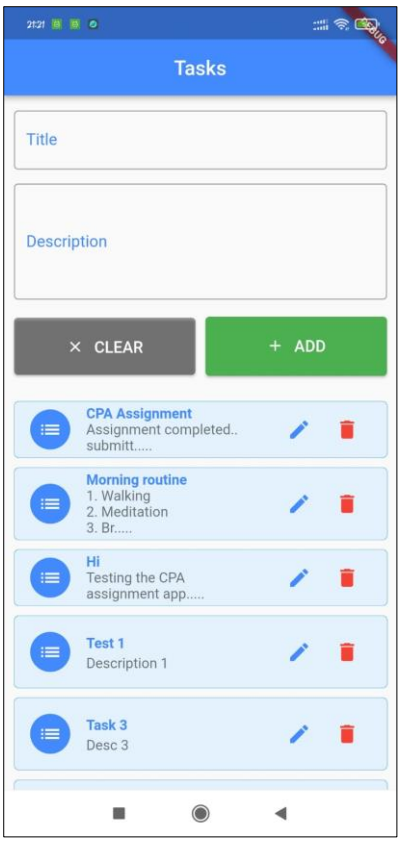
Landing page :



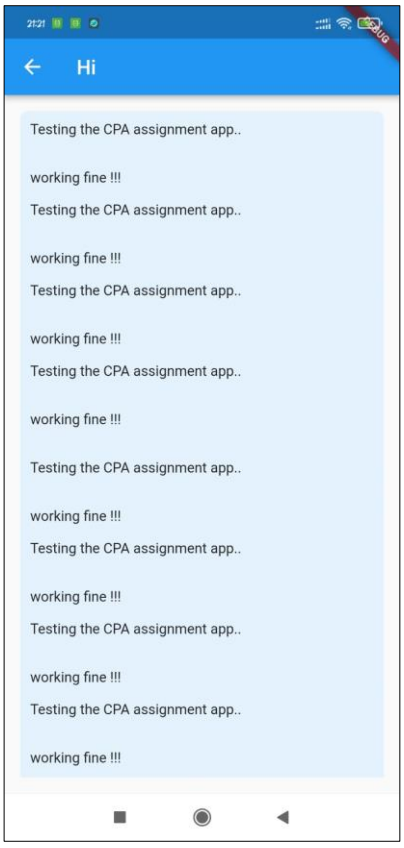
Adding Task :



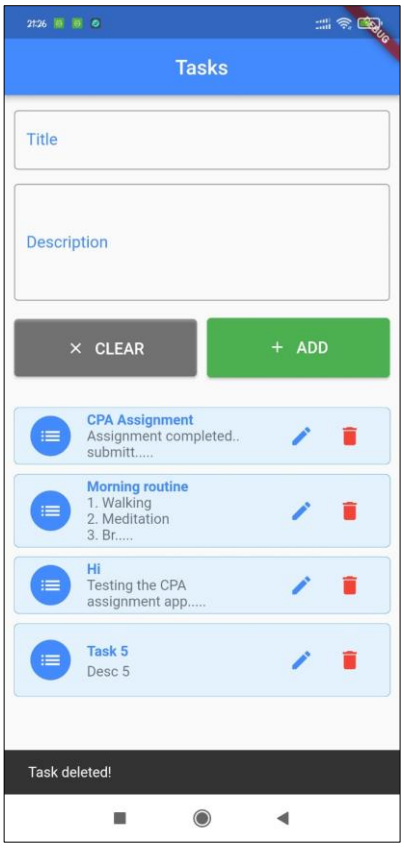
List of Tasks :



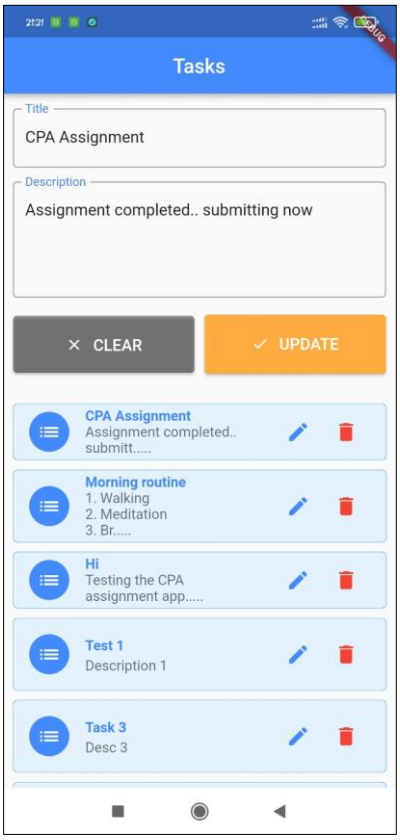
Detailed View :

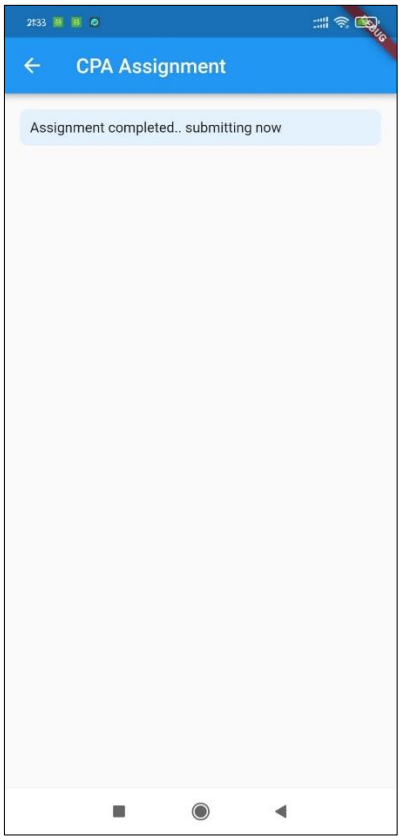


Deleting a task :

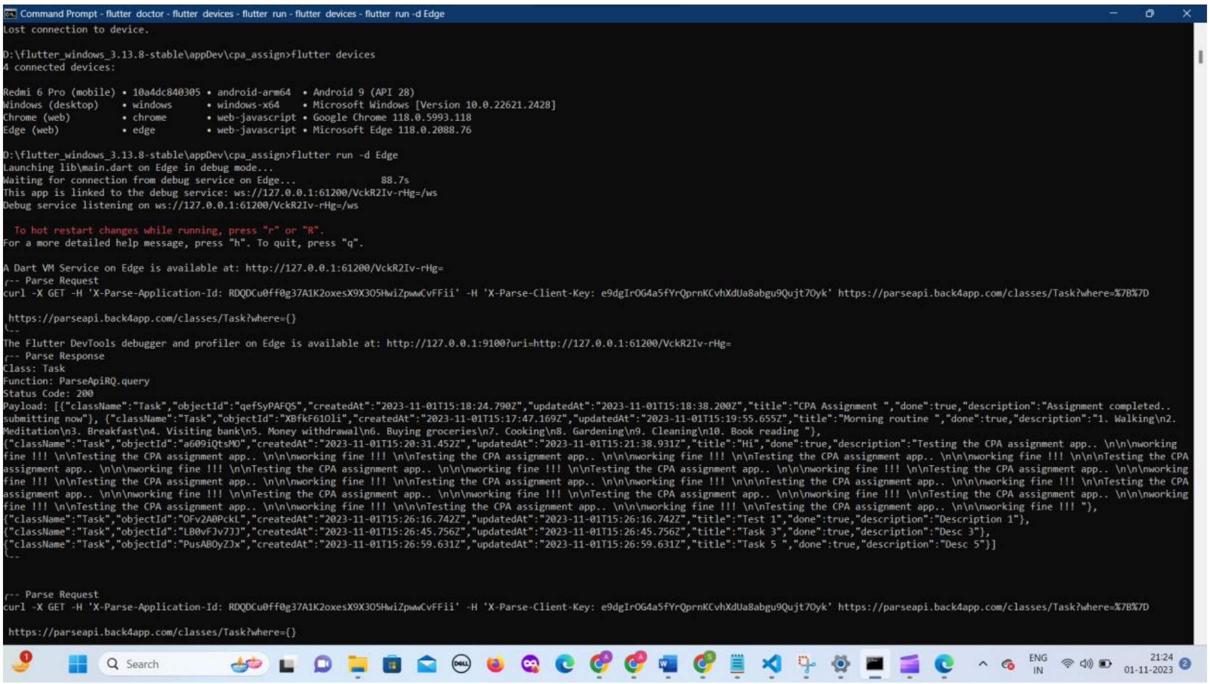


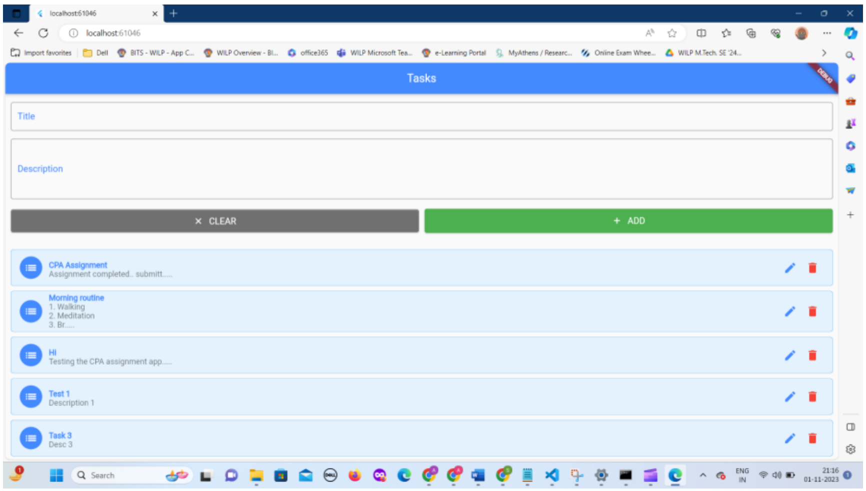
Edit & Updating Task :

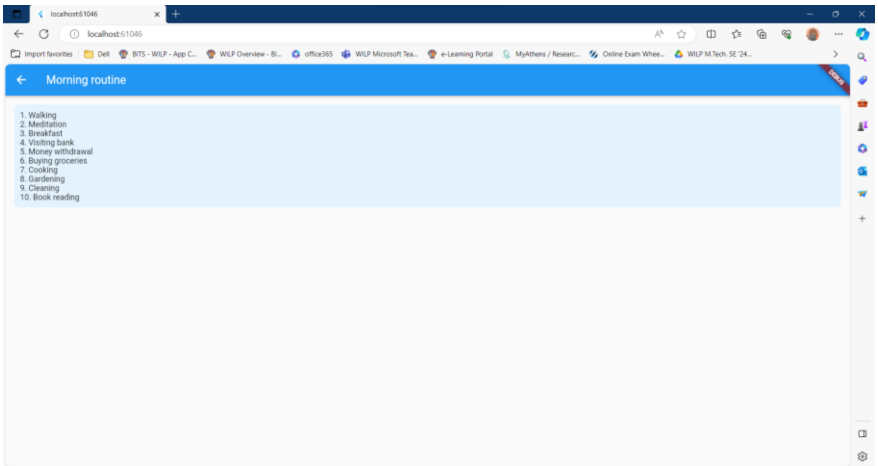


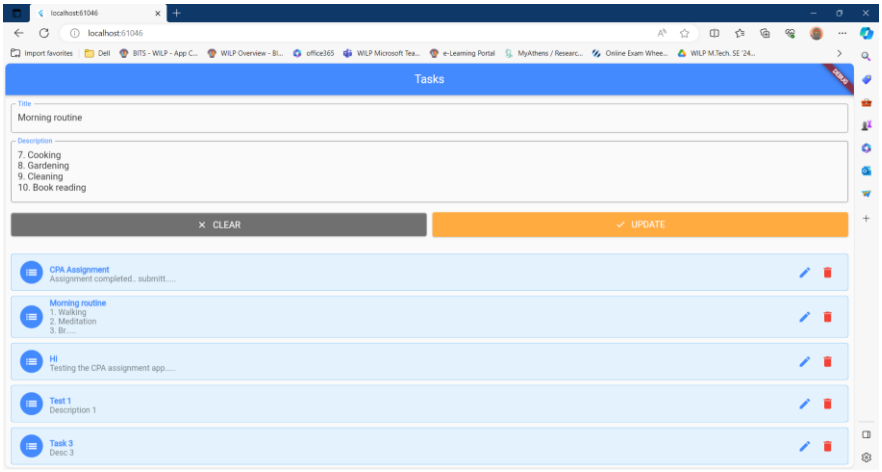


14. Check the browser opened automatically showing the output.









15. Connecting to back4app :

```dart
import 'dart:async';

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

void main() async {
  WidgetsFlutterBinding.ensureInitialized();

  final keyApplicationId = 'RDQDCu0ff0g37AlK2oxesX9X3O5HwiZpwwCvFFii';
  final keyClientKey = 'e9dgIrOG4a5fYrQprnKCvhXdUa8abgu9Qujt7Oyk';
  final keyParseServerUrl = 'https://parseapi.back4app.com';

  await Parse().initialize(keyApplicationId, keyParseServerUrl,
      clientKey: keyClientKey, debug: true);

  runApp(MaterialApp(
    home: Home(),
  ));
}

class Home extends StatefulWidget {
  @override
  _HomeState createState() => _HomeState();
}

class _HomeState extends State<Home> {
  final taskController = TextEditingController();
  final descController = TextEditingController();
  final objIdController = TextEditingController();
  bool isUpdate = false;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Tasks"),
        backgroundColor: Colors.blueAccent,
        centerTitle: true,
```

16. Save() , update(), delete(), get() for accessing the data :

```dart
Future<void> saveTask(String title, String description) async {
  final task = ParseObject('Task')..set('title', title)..set('description', description)..set('done', true);
  await task.save();
}

Future<List<ParseObject>> getTasks() async {
  QueryBuilder<ParseObject> queryTask = QueryBuilder<ParseObject>(ParseObject('Task'));
  final ParseResponse apiResponse = await queryTask.query();

  if (apiResponse.success && apiResponse.results != null) {
    return apiResponse.results as List<ParseObject>;
  } else {
    return [];
  }
}

void editTask(String id, String title, String description)  {
  taskController.text=title;
  descController.text=description;
  objIdController.text=id;
    setState(() => isUpdate=true);
}

Future<void> updateTask(String id, String title, String description) async {
  var task = ParseObject('Task')
    ..set('objectId', id)
    ..set('title', title)
    ..set('description', description)
    ..set('done', true);
  await task.update();
}

Future<void> deleteTask(String id) async {
  var task = ParseObject('Task')..objectId = id;
  await task.delete();
}

void clearTask(){
```

Brief overview of bonus features implemented:

- Added Clear button to clear the data
- Added validation before submitting the empty data.
- Added multiline for description
- Added icons for the Add and Clear buttons
- Added CSS for the task list items and details of the task
- Showed limited text for the description in the task list items when exceeded more than 30 chars and displayed whole text in details
- Dynamic text change on button from Add to Update and Update to Add, according to the mode of operation along with background colour.
- Made containers scrollable if the height is exceeding.