Task Wave Application

How to Run App

Prerequisites:

- Flutter SDK installed and configured
- Git installed
- A code editor or IDE (like Visual Studio Code)

Steps:

1. Clone the Repository:
    - Open a terminal or command prompt.
    - Navigate to the desired directory where you want to clone the project.

Use the `git clone` command followed by the repository URL :

git clone https://github.com/username/repo-name.git

    - Replace `https://github.com/username/repo-name.git` with the actual URL of the repository.

2. Navigate to the Project Directory:

Open a terminal or command prompt and navigate to the cloned project directory

cd repo-name

3. Get Dependencies:

Install the required dependencies by running
flutter pub get

4. Run the App:
    - Connect a physical device or start an emulator.
    - Run the app using one of the following commands:

flutter run

Using an IDE:

- Most IDEs like Visual Studio Code have built-in Git integration. You can clone the repository directly from the IDE.
- Once cloned, the IDE will usually detect the Flutter project and provide options to run, debug, and manage dependencies.

App Features

1. **Task Management:**
    - **Create Tasks:** Users can create new tasks with details such as title and description.
    - **Update Tasks:** Users can update existing tasks.
    - **Delete Tasks:** Users can remove tasks that are no longer needed.
    - **Comment on Tasks:** Users can add comments to tasks for better collaboration.
2. **Theming:**
    - The app supports dynamic theming controlled by a ThemeCubit.
3. **Responsive Design:**
    - The app utilizes the SizeConfig package to ensure proper scaling on different screen sizes.
4. **State Management:** The app uses the BloC architecture pattern for state management by which state of app can be managed easily to manage state and write tests

Packages Used

Flutter Packages:

`get_it` (version ^7.7.0) is a service locator for dependency injection in Dart and Flutter apps.

`dartz` (version ^0.10.1) is a functional programming library for Dart, including types like `Option` and `Either`.

`flutter_bloc` (version ^8.1.6) implements the BLoC pattern in Flutter, helping manage state and business logic.

`equatable` (version ^2.0.5) provides value equality for Dart objects to simplify equality checks.

`dio` (version ^5.5.0+1) is a powerful HTTP client for Dart with features such as interceptors and global configuration.

`internet_connection_checker` (version ^1.0.0+1) checks for internet connectivity status in Dart and Flutter apps.

`google_fonts` (version ^6.2.1) allows easy use of Google Fonts in Flutter apps.

`flutter_colorpicker` (version ^1.1.0) provides a color picker widget for Flutter applications.

`size_config` (version ^2.0.3) helps manage different screen sizes and dimensions in Flutter apps.

`flutter_svg` (version ^2.0.10+1) adds SVG support for Flutter.

`flutter_launcher_icons` (version ^0.13.1) is used to generate app launcher icons for Flutter applications.

`flutter_datetime_picker_plus` (version ^2.2.0) offers a DateTime picker widget with additional features for Flutter.

`intl` (version ^0.19.0) provides internationalization and localization support for Dart and Flutter.

`pie_chart` (version ^5.4.0) is a pie chart widget for Flutter applications.

`uuid` (version ^4.4.2) generates universally unique identifiers (UUIDs).

`go_router` (version ^14.2.1) is used for declarative routing in Flutter applications.

`flutter_boardview` (version ^0.2.1) provides a board view widget for managing and interacting with cards.

`shared_preferences` (version ^2.2.3) is a key-value store for storing simple data persistently in Flutter apps.

`hive` (version ^2.2.3) is a lightweight and fast key-value database for Flutter and Dart.

`hive_flutter` (version ^1.1.0) integrates the Hive database with Flutter.

`mockito` (version ^5.4.4) is a mocking framework for unit testing in Dart.

`build_runner` (version ^2.4.11) is a build system for Dart used for code generation.

`json_serializable` (version ^6.8.0) is used for code generation of JSON serialization and deserialization in Dart.

`hive_generator` (version ^2.0.1) is a code generator for Hive type adapters.

`bloc_test` (version ^9.1.7) provides testing utilities for the BLoC pattern in Flutter apps.

App Video : recording.webm