```markdown
## Project name
Weatherly

## Introduction
Weatherly is a personalized weather application that helps users customize their weather update based on the activity they tend to perform in the future.

## Setting Up Your Development Environment
To set up a Flutter development environment in Visual Studio Code, you need to follow several key steps.

1. **Install Flutter SDK**
  - Download the Flutter SDK from the [Flutter website](https://flutter.dev/docs/get-started/install).
  - Extract the zip file to a desired location on your machine.
  - Add the Flutter tool to your path.

2. **Install Android Studio (Optional but Recommended)**
  - Although not strictly necessary for Flutter, Android Studio provides the Android SDK and device emulators.
  - Download and install [Android Studio](https://developer.android.com/studio).
  - During installation, ensure that the Android SDK, Android SDK Platform-Tools, and Android SDK Build-Tools are selected.

3. **Configure the Android Emulator**
  - In Android Studio, open the AVD Manager and create a new Android Virtual Device.
  - Follow the prompts to choose a device and download the necessary system images.
4. **Install Visual Studio Code**
  - If not already installed, download and install [Visual Studio Code (VS Code)](https://code.visualstudio.com/).

5. **Install the Flutter and Dart Extensions in VS Code**
  - Open VS Code.
  - Go to Extensions (View -> Extensions).
  - Search for "Flutter" and install the Flutter extension (this should automatically install the Dart extension).

6. **Validate the Installation**
  - Open a terminal in VS Code (View -> Terminal).
  - Run `flutter doctor`. This command checks your environment and displays a report to the terminal window.
  - Address any issues identified by `flutter doctor`.
```

```markdown
7. **Create and Run a Flutter App**
  - In VS Code, open the Command Palette (View -> Command Palette or `Ctrl+Shift+P` on
Windows/Linux, `Cmd+Shift+P` on macOS).
  - Type "Flutter", and select the `Flutter: New Project` option.
  - Follow the prompts to create a new Flutter application.
  - Once created, navigate to the project's directory in VS Code.
  - Start your emulator through Android Studio.
  - Run the app by pressing `F5` or selecting “Start Debugging” from the Run menu.

These steps will set up a basic Flutter development environment in VS Code, allowing
you to develop, test, and run Flutter applications.
Also you need to set up Xcode for iOS development on a Mac.

## Project Structure
In the development phase, we created four different scripts: main.dart,
weather_model.dart, weather_service.dart, weather_page.dart.

**Main.dart**
This class is a Flutter application entry point that sets up a Material Design app
with a single screen, WeatherPage, for displaying weather information.
- Main.dart contains imported libraries and dart files necessary for the app's
functionality.
- Main function which calls runApp() to start the app with the MyApp widget.
- MyApp Widget which is a stateless widget that defines the overall structure of the
app, usually including theming, navigation, and home screen.



**weather_model**
The weather_model class defines a data model for weather information, including
properties like city name, temperature, main condition, humidity,
and wind speed. It includes a constructor for initializing these properties and a
factory constructor for creating Weather instances from JSON data.



**weather_service**
This class provides functionalities to fetch weather information for a specified city
and to determine the user's current city based on their geographical location.



**weather_page**
```

This class fetches and displays weather information, with interactive elements like a search field and notifications icon. It also includes a NotificationsPage for navigation.

Getting Started
Before you begin, make sure you have a GitHub account and that you're familiar with basic GitHub workflows. If you're new to Git or GitHub, check out this [Git Handbook](https://guides.github.com/introduction/git-handbook/).

Setting Up Development Environment
1. Fork the repository on GitHub.
2. Clone your fork locally:
   ```bash
   git clone git@github.com:your-username/Weatherly.git
   ```
3. Install the necessary dependencies:
   ```bash
   cd Weatherly
   npm install
   ```
4. Create a new branch for your feature or bugfix:
   ```bash
   git checkout -b feature/your-feature-name
   ```

Making Changes
1. Make your changes in the newly created feature branch.
2. Write tests for your changes and ensure all tests pass:
   ```bash
   npm run test
   ```
3. Add or update documentation as needed.## Committing Your Changes
1. Commit your changes using a clear commit message.
2. Push your changes to your fork:
   ```bash
   git push origin feature/your-feature-name
   ```

Submitting a Pull Request
1. Navigate to the original `Weatherly` repository you created your fork from.
2. Press the "New pull request" button.

```
3. Choose your fork and the feature branch you've created.
4. Fill in the pull request description and submit it.
```