➢ How to Download, Install, and Use the IU Nav Bus App from GitHub

Step 1: Access the GitHub Repository

1. **Visit the Repository:**
   - Click on the link provided to the GitHub repository containing the IU Nav Bus App project.
2. **Explore the Repository:**
   - On the repository page, you will see the project files, folders, and a README file with basic information.

Step 2: Download the Project Code

1. **Clone the Repository (Recommended Method):**
   - Open your terminal or command prompt.
   - Navigate to the directory where you want to clone the project.
   - Click on the project.
2. **Download as ZIP (Alternative Method):**
   - On the GitHub repository page, click on the "Code" button.
   - Select "Download ZIP" from the dropdown menu.
   - Save the ZIP file to a convenient location on your computer.
   - Extract the ZIP file to access the project files.

Step 3: Set Up the Development Environment

1. **Install Flutter SDK:**
   - Download and install the Flutter SDK from Flutter's official website.
   - Follow the installation instructions specific to your operating system (Windows, macOS, or Linux).
2. **Install Required Dependencies:**
   - Open your terminal or command prompt.
   - Navigate to the project directory where the `pubspec.yaml` file is located.
   - Run the following command to install all necessary dependencies:

```
bash
Copy code
flutter pub get
```

Step 4: Set Up Firebase

1. **Create a Firebase Project:**
   - Go to the Firebase Console.

- Click on "Add Project" and follow the steps to create a new project.

2. **Add Firebase to Your App:**
   - In the Firebase console, click on "Add App" and choose Android or iOS.
   - Follow the instructions to download the `google-services.json` (for Android) or `GoogleService-Info.plist` (for iOS) file.
   - Place the file in the appropriate directory in your Flutter project:
     - Android: `android/app/`
     - iOS: `ios/Runner/`
3. **Enable Firebase Services:**
   - In the Firebase console, enable the services you need, such as Firestore, Authentication, and Cloud Messaging.
4. **Integrate Firebase with Flutter:**
   - Add the required Firebase dependencies in your `pubspec.yaml` file.
   - Run `flutter pub get` to install them.
   - Initialize Firebase in your Flutter app by following the official Firebase documentation.

Step 5: Set Up Supabase

1. **Create a Supabase Project:**
   - Visit the Supabase website.
   - Sign up or log in and create a new project.
2. **Configure Supabase in Your App:**
   - Copy your Supabase project URL and the public API key from the Supabase dashboard.
   - Add these details to your Flutter app. Typically, you would store them in a `.env` file or directly in your Flutter code, but ensure they are securely managed.
3. **Add Supabase Dependencies:**
   - Add the Supabase Flutter package to your `pubspec.yaml` file:

     ```
     yaml
     Copy code
     dependencies:
       supabase_flutter: ^0.2.0
     ```

   - Run `flutter pub get` to install the package.
4. **Initialize Supabase:**
   - Initialize Supabase in your Flutter app, typically in the `main.dart` file:

     ```
     dart
     Copy code
     import 'package:supabase_flutter/supabase_flutter.dart';

     void main() async {
       await Supabase.initialize(
         url: 'your-supabase-url',
         anonKey: 'your-anon-key',
       );
       runApp(MyApp());
     }
     ```

Step 6: Set Up a Simulator or Connect a Device

1. **Run the App on a Simulator:**
    - Open Android Studio or Visual Studio Code, depending on your preference.
    - Launch an Android or iOS emulator.
    - Ensure the emulator is running before proceeding.
2. **Run the App on a Physical Device:**
    - Connect your Android or iOS device to your computer via USB.
    - Enable USB debugging (for Android) or trust the device (for iOS).

Step 7: Build and Run the App

1. **Build the App:**
    - Open your terminal or command prompt.
    - Ensure you are in the project directory.
    - Run the following command to build and run the app:

      ```
      bash
      Copy code
      flutter run
      ```

    - The app should launch on the connected device or emulator.

Step 8: Access and Use the App

1. **Explore the App Features:**
    - Once the app is running, you can explore its features such as real-time bus tracking, notifications, feedback, and emergency support.
    - For students, log in or sign up to access bus schedules and track buses in real time.
    - For drivers, log in to update route status and communicate with students through the app.
2. **User Guide:**
    - Detailed user instructions can be found within the app's "Help" section, where you can learn more about how to navigate and use all the features.
3. **Provide Feedback:**
    - Use the feedback feature in the app to share your experience and help improve the service.

Step 9: Push Changes (For Developers)

1. **Make Edits to the Code:**
   - If you wish to contribute to the project, make your changes in the project files.
2. **Commit and Push Changes:**
   - After making changes, commit them with a descriptive message:

     ```
     bash
     Copy code
     git commit -m "Your commit message"
     ```

   - Push the changes to the repository:

     ```
     bash
     Copy code
     git push origin main
     ```

   - Replace `main` with the appropriate branch name if you are working on a different branch.

Note:

- Ensure that your device or emulator has an active internet connection for the app to function correctly.
- If you encounter any issues during the installation, please refer to the "Help" section in the app or contact support.