Weather app

🌤️ Project: Weather App (Jetpack Compose Edition)

🌐 Core Features

- **Real-Time Weather Fetching**: Get current weather data by calling the **OpenWeatherMap API**.
- **City Selection**: Users can set or change the city via a **Settings screen**.
- **Weather Display**: Shows current temperature, conditions, and other relevant info on the **Home screen**.

🧱 Tech Stack & Architecture

- **Retrofit (Network Call)**:
  - Makes HTTP requests to the **OpenWeatherMap API**.
  - Parses JSON responses into data classes using **Gson** or **Moshi**.
  - Simple and efficient setup for RESTful APIs.
- **Jetpack DataStore (Caching & Preferences)**:
  - Replaces `SharedPreferences` for a modern, type-safe way to persist key-value data.
  - Used to store the **selected city** and **last fetched weather data**.
  - Supports **asynchronous data access** with Kotlin Coroutines and **Flow**.
- **Jetpack Compose UI**:
  - Built with **Composable functions** for the Home and Settings screens.
  - Displays weather info using intuitive layouts (cards, icons, formatted text).
- **Navigation Compose**:
  - Manages screen transitions between **Home** and **Settings**.
  - City settings are read and passed seamlessly using state and ViewModels.

✅ Why It's a Great Project

- A practical introduction to **Retrofit**, **Jetpack DataStore**, and handling **network responses**.
- Shows how to build reactive UIs using **Jetpack Compose** and **Kotlin Flow**.

- Demonstrates clean **navigation** and **state management** in a real-world app.
- Easy to extend—add features like hourly/daily forecasts, dark mode, location-based search, or offline caching.