Class Diagram For Weather App

Overview of the UML Class Diagram
This UML class diagram provides a detailed representation of the core structure and functionality of the weather app. It highlights the relationships between the key components and defines their attributes and behaviors.

Key Components

1. **MainActivity**
   - **Purpose**: This is the central class and entry point of the app. It handles user interactions and manages the overall flow of the application.
   - **Attributes**:
     - Includes UI components like RelativeLayout, TextView, and RecyclerView to display weather data.
     - Holds important data objects like ArrayList<WeatherRVModal> (for weather details) and WeatherRVAdapter (for managing RecyclerView).
     - Contains the LocationManager to get the user's location and cityName to fetch weather data for a specific city.
   - **Methods**:
     - onCreate: Initializes the app and sets up the UI.
     - onRequestPermissionsResult: Handles user permissions for location access.
     - getCityName: Retrieves the city name based on latitude and longitude.
     - getWeatherInfo: Fetches weather details for a given city.
2. **WeatherRVAdapter**
   - **Purpose**: Acts as a bridge between the weather data (model) and the RecyclerView (view). It binds weather information to UI components dynamically.
   - **Attributes**:
     - Stores the weather data in an ArrayList<WeatherRVModal>.
     - Uses Context for accessing resources and application context.
   - **Methods**:
     - onCreateViewHolder and onBindViewHolder: Create and bind views for each weather item in the RecyclerView.
     - getItemCount: Returns the total number of items.
   - **Nested Class: ViewHolder**:
     - **Purpose**: Represents individual items in the RecyclerView.
     - Attributes like TextView and ImageView are used to display specific weather details such as wind speed, temperature, and icons.
3. **WeatherRVModal**
   - **Purpose**: Represents the data model for weather details.
   - **Attributes**:
     - Holds the properties for a weather item, such as time, temperature, icon, and wind speed.
   - **Methods**:
     - Getters and setters for each attribute ensure data encapsulation and easy manipulation.

Relationships Between Components

1. **Aggregation Relationship**:
   - **MainActivity aggregates WeatherRVAdapter and ArrayList<WeatherRVModal>**:
     - MainActivity uses WeatherRVAdapter to manage the weather data displayed in the RecyclerView.
     - The ArrayList<WeatherRVModal> is the data source for the adapter.
2. **Composition Relationship**:
   - **WeatherRVAdapter has a composition relationship with ViewHolder**:
     - The ViewHolder is tightly bound to the WeatherRVAdapter since it cannot exist independently.
3. **Interaction**:
   - **WeatherRVAdapter interacts with WeatherRVModal**:
     - The adapter accesses the WeatherRVModal objects to display the data in the RecyclerView.

How This Structure Works

1. **MainActivity** acts as the controller, initializing the UI, fetching weather data, and passing it to the adapter.
2. The **WeatherRVAdapter** takes the weather data from ArrayList<WeatherRVModal> and binds it to the UI components in the RecyclerView using **ViewHolder**.
3. The **WeatherRVModal** serves as the data model, encapsulating the weather-related information such as time, temperature, and icons.
4. Together, this structure follows a clean MVC (Model-View-Controller) pattern, ensuring modularity and separation of concerns.