Project Structure and File Explanation

This document provides a detailed overview of the project's structure and the purpose of each file and directory.

Project Architecture

The application is built using the **MVVM (Model-View-ViewModel)** architecture pattern, which separates the UI from the business logic. This makes the code more modular, testable, and easier to maintain.

Directory and File Breakdown

`app/src/main/java/com/example/geomarker/`

This is the root package of the application.

- **`MainActivity.kt`**: The main and only activity in the application. It hosts the NavHostFragment, which manages the navigation between the different fragments.
- **`AppDatabase.kt`**: This file defines the Room database for the application. It specifies the entities that are part of the database and provides access to the DAOs (Data Access Objects).

`api` Package

- **`ApiService.kt`**: This file contains the Retrofit API interface. It defines the HTTP methods (GET, POST, PUT, DELETE) and the endpoints for interacting with the REST API.

`model` Package

- **`Entity.kt`**: This is the data class that represents a single geographic entity. It's used by both Room (as a database entity) and Retrofit (for parsing JSON responses).
- **`EntityDao.kt`**: This is the Data Access Object (DAO) for the `Entity` class. It defines the methods for interacting with the `entities` table in the Room database (e.g., insert, update, delete, get all).

`repository` Package

- **`EntityRepository.kt`**: This class acts as a single source of truth for the application's data. It's responsible for fetching data from the `ApiService` (remote data source) and the `EntityDao` (local data source). It also handles the logic for caching data in the local database.

`viewmodel` Package

- `MapViewModel.kt`: This ViewModel provides data to the `MapFragment` and `EntityListFragment`. It holds the list of all entities as `LiveData`, which the UI observes for changes.
- `EntityFormViewModel.kt`: This ViewModel is responsible for the business logic of the `EntityFormFragment`. It handles the creation and updating of entities by calling the appropriate methods in the `EntityRepository`.

`ui` Package (Fragments and Adapters)

The `ui` package contains the fragments that make up the user interface of the application.

- `MapFragment.kt`: This fragment displays the OpenStreetMap and the markers for each entity. It also handles user interactions with the map, such as tapping on markers.
- `EntityListFragment.kt`: This fragment displays a list of all entities in a `RecyclerView`. It allows users to edit or delete entities from the list.
- `EntityFormFragment.kt`: This fragment provides a form for users to create new entities or edit existing ones. It includes fields for the title, latitude, longitude, and an image.
- `adapter/EntityListAdapter.kt`: This is the `RecyclerView.Adapter` for the list of entities in `EntityListFragment`. It's responsible for binding the entity data to the views in each list item.

`res` Directory

This directory contains all the resources for the application.

- `layout`: Contains the XML layout files for the fragments and list items.
- `drawable`: Contains image resources.
- `navigation`: Contains the navigation graph (`nav_graph.xml`), which defines the navigation paths between the fragments.