Mobile Application Development

Assignment 2

Meet Patel

100785841

**Database Initialization and Structure**: (explaining how databases are used)

Upon initial launch, the software creates an SQLite database named LocationFinder.db. This database's creation and maintenance are managed by a DatabaseHelper class that extends SQLiteOpenHelper. The database structure contains a single table, “locations”, which has four columns:

* id (Primary Key): It automatically sets an identifier for each record and increments on every new record addition.
* Address: This is the city name in our case stored in “TEXT” data type.
* Latitude: This stores the latitude of the place in “REAL” data type.
* Longitude: This will store the longitude of the location in “REAL” data type.

Preloading data involves filling up the database with some initial entries and locations. A json file with 100 predefined location entries stored in the assets folder of the application is available. This JSON would be read once the application launches for the first time; it then parses it and stores that information into the locations database. SharedPreferences settings is used ensure that this initialization is performed only once.

This application also uses an application of CRUD on data in locations; operations are defined in DatabaseHelper and accessed via UI in MainActivity and UpdateDatabaseActivity.

Below are the defined methods:

1. Create: Add Location

addLocation in DatabaseHelper: This method inserts a new location into the table. It takes user Latitude and Longitude. The structuring is made through ContentValues, then it is added into the table through SQLiteDatabase.insert.

1. Read: Query Location

The queryLocation: method allows users to look up a location's latitude and longitude by name. The implementation uses SQL SELECT queries to fetch and display the relevant data when provided the location in question efficiently, even in larger data sets.

1. Update: Move Location

The updateLocation: method allows the user to update a created place by editing its latitude and longitude information. This method uses SQL UPDATE commands, first checking if a record exists before allowing editing.

1. Delete: Remove Location

The function deleteLocation gives users the option to delete an entry based on the address (City name). This use of SQL DELETE commands ensures that the database will not become overwhelming or irrelevant.

**Data Processing and User Interaction**

1. Data Entry and Validation:

Following that, the respective activity UpdateDatabaseActivity of this application has EditTexts for user input of name, latitude, and longitude of the location. It contains field validation checks for assuring the proper filling of the fields; inputType for latitude and longitude fields allows decimal and negative number for full geographical representation.

1. Feedback and Error Handling:

Each manipulation reflects instant status through toast messages on successful insertion and updates, delete, and errors that might have occurred. Log entries are helpful in debugging, hence tracing steps right from initialization to create, read, update, and delete.

GitHub Repository: <https://github.com/patelmeet1372/Assignment1_EMI_calculator.git>