

# Mobile Application Development Assignment 2 Report

**Name:** Kunal Pandya

**Student Number:** 100792272

**GitHub Repository:** [https://github.com/kunalpanda/Mobile\\_Dev\\_Assign\\_2](https://github.com/kunalpanda/Mobile_Dev_Assign_2)

## Introduction

This report outlines the design and development of the Mobile Application Development Assignment 2 project. The purpose of this assignment was to build a mobile application that integrates a local database using the Room Persistence Library and displays geographic information using the Google Maps API. The application allows users to manage a collection of GTA locations with full Create, Read, Update, and Delete (CRUD) functionality and visualize searched locations directly on an interactive map. This project demonstrates the practical use of persistent data storage and external API integration within a structured Android environment developed entirely in Kotlin.

## Activities Overview

The application consists of multiple activities that perform specific operations in a modular way, enhancing readability and maintainability.

- **MainActivity:** Serves as the central hub that initializes the Room database and Google Map fragment. It also links the user interface buttons to corresponding activities for adding, updating, deleting, and viewing locations.
- **AddLocationActivity:** Allows users to add a new location by providing an address and its latitude and longitude. Built-in validation ensures that all fields are completed while ensuring coordinate values are correctly formatted before insertion into the database.
- **UpdateLocationActivity:** Enables users to modify an existing record's address or coordinates. It first verifies the location's existence using the shared search feature before applying updates.
- **DeleteLocationActivity:** Provides a dialog to remove an existing location from the database permanently. A secondary confirmation dialog prevents accidental deletions.
- **SearchActivity:** Implements a smart search feature that supports both address-based and coordinate-based queries. The activity dynamically adjusts its input fields based on the dropdown input field and communicates search results to other activities. The dialog automatically dismisses after a successful query for smoother interaction.

Each activity uses consistent design principles and follows the same layout style, maintaining a clean, user-friendly interface. All CRUD operations are responsive and give immediate visual feedback through toast messages.

## Database Integration

The database layer is implemented using the Room Persistence Library, which provides a structured approach to data management. The system consists of three key components:

- Location (Entity Class) – Defines the table schema with columns for id, address, latitude, and longitude.
- LocationDOA (Data Access Object) – Contains methods for inserting, updating, deleting, and querying records. It includes both exact and partial address searches, as well as coordinate lookups with tolerance to handle approximate matches.
- AppDatabase Class – Serves as the main database holder and is configured as a singleton to ensure a single shared instance across the entire application.

## **Map Integration**

The mapping functionality is implemented using the Google Maps API with a SupportMapFragment. When a user searches for a location, the corresponding latitude and longitude are retrieved from the database, and a marker is added to the map. The frame automatically centers on the selected marker, providing immediate visual feedback.

The map initializes with a default view centred on Toronto and zooms in to the location when found. This integration demonstrates practical use of asynchronous map loading through OnMapReadyCallback, combined with dynamic data retrieval from the local database.

## **Conclusion**

This project successfully demonstrates the integration of persistent data storage and interactive mapping within a mobile application. The use of the Room Persistence Library ensures reliable and efficient local data management, while the Google Maps API provides a clear visual representation of geographic information. The modular, activity-based structure, combined with improved database handling and responsive UI design, results in a cohesive, functional application that meets all assignment objectives.