



GRIFFITH COLLEGE DUBLIN

Assignment Cover Sheet

Student name:	Toluwani Aderibigbe		
Student number:	3050667		
Faculty:	Computing Science		
Course:	BSCH	Stage/year:	4
Subject:	Mobile Development		
Study Mode:	Full time <u>yes</u>	Part-time	_____
Lecturer Name:	Tracey Cassells		
Assignment Title:	MD Milestone 2 Travel Journal App		
No. of pages:	8		
Disk included?	Yes _____	No	<u>no</u>
Additional Information:	(ie. number of pieces submitted, size of assignment, A2, A3 etc)		

Date due:	19/12/2025		
Date submitted:	10/12/2025		

Plagiarism disclaimer:

I understand that plagiarism is a serious offence and have read and understood the college policy on plagiarism. I also understand that I may receive a mark of zero if I have not identified and properly attributed sources which have been used, referred to, or have in any way influenced the preparation of this assignment, or if I have knowingly allowed others to plagiarise my work in this way.

I hereby certify that this assignment is my own work, based on my personal study and/or research, and that I have acknowledged all material and sources used in its preparation. I also certify that the assignment has not previously been submitted for assessment and that I have not copied in part or whole or otherwise plagiarised the work of anyone else, including other students.

Signed: Ade

Date: 10/12/2025

Please note: Students **MUST** retain a hard / soft copy of **ALL** assignments as well as a receipt issued and signed by a member of Faculty as proof of submission

Milestone 3

1. Introduction

My Travel Journal App Milestone3 is a mobile application I developed using Kotlin, Jetpack Compose, and Room Database.

its purpose is to allow users to record, organize, and review trips conveniently on their mobile device.

MY project demonstrates essential Android development concepts such as: Declarative UI with Jetpack Compose, Local data persistence with Room, MVVM architectural pattern, State management using Kotlin Flows, Navigation between composable screens

My final application is lightweight, user-friendly, and designed with clean UI practices.

2. App Overview

My Travel Journal App provides two main functions.

Add a New Trip : Users can fill in details such as the destination, description, travel date, and personal notes.

View Saved Trips : All saved trips are displayed in a scrollable list, organized using Compose Cards.

Although simple, my app is fully functional and follows modern Android development standards.

3. Technologies Used

Component	Explanation
Kotlin	Primary programming language
Jetpack Compose	Used to build the entire UI declaratively
Room Database	Handles local data storage using DAO and Entity
MVVM Architecture	Separates logic, data, and UI for scalability
Navigation Compose	Moves between Add Trip and Trip List screens

This combination ensures a clean, maintainable, and testable application.

4. **Architecture Summary**

The project is structured into clearly separated layers.

Model Layer (Data) : Trip data class annotated with @Entity, Room table trip_table

Data Access Layer : TripDao includes getAllTrips() returns all trips as a Flow, and insertTrip() inserts a new entry

Repository Layer : TripRepository supports between DAO and ViewModel.

ViewModel Layer : TripViewModel it contains a flow list of trips, addTrip() function is to insert new trips

UI Layer (Compose Screens) : TripListScreen it displays all saved trips, AddTripScreen it allows adding a trip, TravelJournalApp it handles navigation

5. **Screenshots**

First Screenshot Add Trip Screen : This screen allows the user to input a new travel entry. It includes four neatly arranged text fields for: Destination, Description, Date, Notes and there is also Save Trip button.

Second Screenshot Trip List Screen : This screen displays all saved trips in visually appealing cards. Each card includes. Destination title, Trip description, Travel date with an icon, Notes with an icon.

Add a New Trip

Destination

Cape Town, South Africa

Description

A coastal adventure that includes hiking Table Mountain, visiting the V&A Waterfront, and taking a day trip to Cape Point.

Date

August 22, 2026

Notes

Confirm cable car hours. Keep sunscreen handy. Try local dishes like bobotie and Cape Malay curry.

Save Trip







Travel Journal

Cape Town, South Africa


A coastal adventure that includes hiking Table Mountain, visiting the V&A Waterfront, and taking a day trip to Cape Point.


 August 22, 2026

 Confirm cable car hours. Keep sunscreen handy. Try local dishes like bobotie and Cape Malay curry.

Buenos Aires, Argentina


A vibrant city trip focusing on tango culture, classic architecture, and delicious South American cuisine. Areas to visit include Palermo, La Boca, and San Telmo.


 March 9, 2026

 Catch a live tango show. Try empanadas from a local bakery. Bring cash for street markets.

Lisbon, Portugal

A relaxed city escape filled with ocean views, colorful streets, and historic trams. Plan to explore Alfama, Belém Tower, and local cafés.

 June 11, 2025

 Wear comfortable walking shoes—lots of hills. Taste the original Pastéis de Belém. Take tram 28 early to avoid crowds.



6. Conclusion

My Travel Journal App is a complete, modern Android application demonstrating. Proficiency in Kotlin, Ability to build responsive UI with Jetpack Compose, Correct implementation of Room Database, Understanding of MVVM and state management

By integrating these technologies, my app delivers a smooth and user-friendly experience for recording travel memories.

Milestone 2 Travel Journal – GPS & Motion Tracking

Travel Journal – Sensor-Based Location & Motion Tracking Application

1. Introduction

My milestone extends my earlier work by integrating two core hardware sensors available on modern Android devices: the GPS location sensor and the accelerometer. My aim was to build a simple, practical mobile application that can read real-time sensor data and present it clearly to the user using Jetpack Compose.

To achieve this, I developed a small application titled Travel Journal Sensors. My app provides two major capabilities

GPS Tracking - Retrieves the device's geographic location (latitude, longitude, accuracy) at frequent intervals.

Motion Tracking - Reads the phone's X, Y, and Z acceleration values from the accelerometer and calculates movement strength.

2. Application Structure Overview

My application is divided into three main screens, each responsible for a separate task:

Welcome Screen (Jetpack Compose) – It serves as the landing page, displays simple navigation options, allowing user to choose between GPS Tracking and Motion Tracking. Implemented entirely using Jetpack Compose layouts and Material 3 components.

LocationActivity (GPS Tracking) - This activity manages all location-related operations. Its main responsibilities include: Requesting runtime permission for ACCESS_FINE_LOCATION, Initializing the FusedLocationProviderClient for high-accuracy positioning. Receiving continuous location updates; displaying Latitude, Longitude, Accuracy (meters). Stopping updates when the activity is no longer active, preventing unnecessary battery consumption.

AccelerometerActivity (Motion Tracking) - This activity handles accelerometer input. Its tasks include: Connecting to the SensorManager, Listening for accelerometer changes (X, Y, Z values), Calculating the overall movement magnitude using $\text{magnitude} = \sqrt{x^2 + y^2 + z^2}$. Presenting live motion values in the UI. Registering and unregistering the sensor listener appropriately inside onResume() and onPause().

All UI components for movement and location tracking are created using Jetpack Compose, making the screens reactive and clean.

3. Features Implemented

GPS Tracking – It Provides live location readings every 1.5 seconds, Uses Google's Fused Location Provider for accuracy and efficiency. Includes runtime permission handling using `ActivityResultContracts` and safeguards to prevent crashes when permission is denied.

Accelerometer (Motion Sensor) Tracking – It Reads all three axes of motion: X, Y, Z. Computes a movement magnitude to indicate total acceleration and displays sensor data instantly using `mutableStateOf`, allowing automatic UI updates.

Modern Jetpack Compose Interface - All screens follow a consistent layout and styling approach, UI responds automatically to state changes and Clean layout with centered elements and clear spacing

Multi-Activity Navigation - Users can navigate from the welcome page to GPS Tracking Screen, Motion Tracking Screen. Intent-based navigation is used for simplicity and clarity.

Correct Use of Activity Lifecycle - Accelerometer listener runs only when the activity is visible, location updates start only after pressing the "Start GPS Tracking" button and both sensors stop listening when the user leaves the screen.

4. Screenshots

My Screenshots Welcome, GPS Tracking, Motion Tracking

10:43 9 9

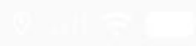
5G 50%

Travel Journal Sensors

Open GPS Tracking

Open Motion Tracking

10:49



GPS Tracking

Latitude: 37.4219983

Longitude: -122.084

Accuracy: 5.0 m

Start GPS Tracking

10:50 9 9



Motion Tracking

X: 0.0
Y: 9.776321
Z: 0.812345

Movement Strength: 9.81

5. Implementation Challenges & Solutions

Challenge 1 - Handling Runtime Permissions Android requires explicit user approval for fine-location access.

Solution - I implemented a permission launcher using `ActivityResultContracts.RequestPermission()`, which improves safety and avoids deprecated APIs.

Challenge 2 - Preventing App Crashes When Permission Was Missing Initially, attempting to start location updates without permission caused warnings.

Solution - A permission check runs before requesting updates, and the update function exits early if permission is not granted.

Challenge 3: Keeping Sensor Reading Live in UI Sensor values were updating internally but not reflecting on-screen.

Solution - I stored sensor values inside `observable mutableStateOf`, allowing Compose to automatically recompose UI whenever the values changed.

Challenge 4: Managing Activity Lifecycle Sensors can drain the battery if left running.

Solution - I used `onResume()` to start accelerometer listening, `onPause()` to stop listening. This ensures efficient behavior and good practice

6. Reflection

My milestone gave me practical experience with hardware sensors, permissions, and real-time UI updates. I gained confidence working with Android's sensor APIs and improved my understanding of Compose state management.

I also learned the importance of lifecycle management and permission handling, which are important skills for building real Android applications.

Overall, My milestone helped me produce a working, interactive app that can track both movement and location accurately

7. Conclusion

My Travel Journal Sensors app fully meets the requirements of Milestone 2. It successfully integrates GPS and accelerometer sensors, displays their readings in real time with Jetpack Compose, and manages permissions and lifecycle events correctly. By separating features into different activities and updating the UI reactively, my app demonstrates a strong understanding of sensor integration and modern Android development.

Student Name: Toluwani Aderibigbe

Student ID: 3050667

Course: BSc (Hons) in Computer Science

Module: Mobile Development Milestone 1

1. Project Overview

My Personal Travel Journal App is an Android application that I developed in Kotlin using Jetpack Compose. It allows users to record and manage travel destinations in an interactive and organized way. my app provides a modern and simple interface where users can view, add, and explore destinations. It will later integrate GPS and sensors for location tracking and activity logging.

My app was chosen for its balance between creativity and functionality, allowing me to demonstrate key Android development skills while designing a userfriendly experience.

2. Objectives

- To design and implement a clean user interface with Jetpack Compose.
- To create a navigation system connecting multiple screens: Welcome, Destination List, and Add Destination
- To ensure smooth navigation using Navigation Compose.
 - To prepare the app structure for future milestones (sensor and database integration)

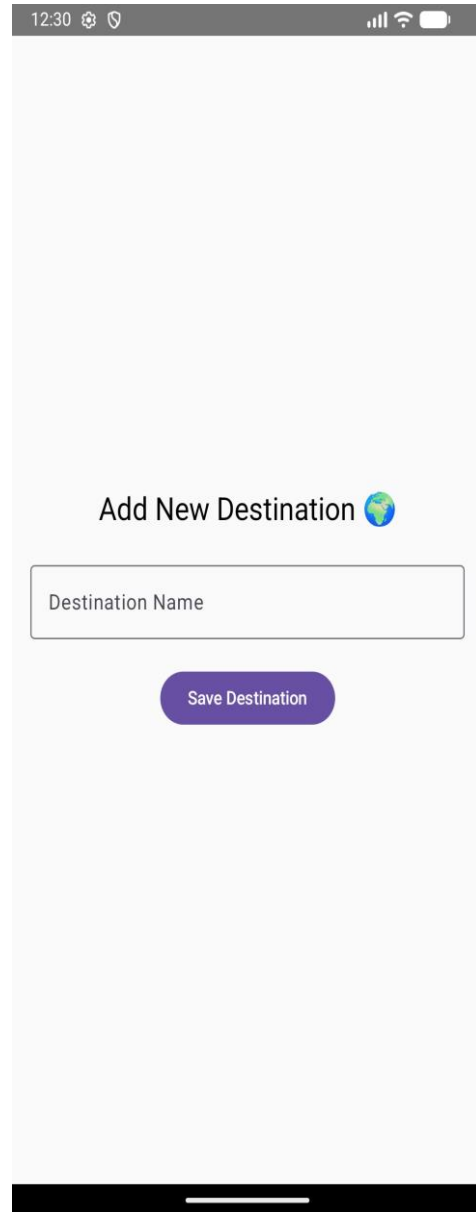
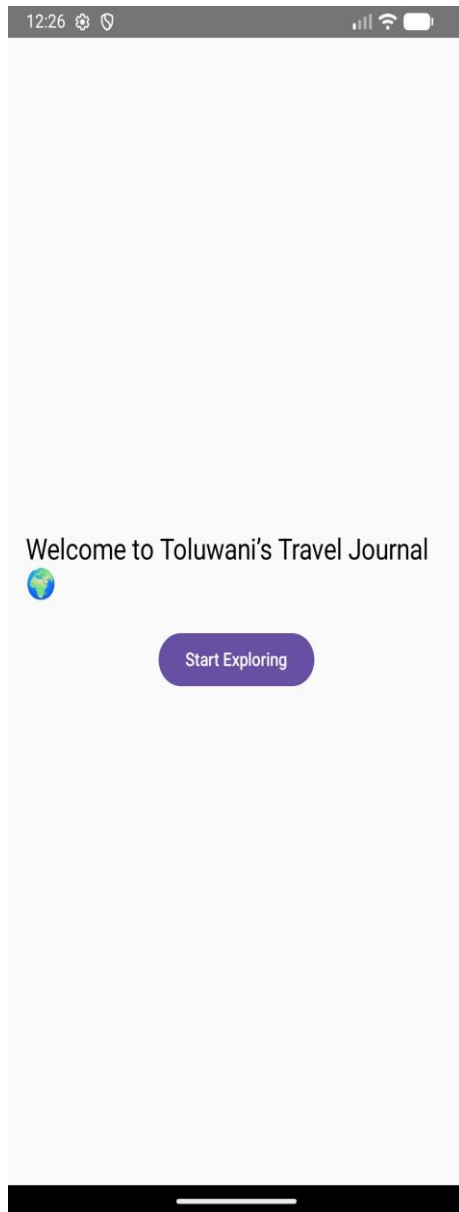
3. System Components

Component	Description
Welcome Screen	Introduces the app and includes a “Start Exploring” button to navigate to the main destinations page.
Destinations Screen	Displays saved travel destinations with an option to add new entries.
Add Destination Screen	Allows the user to input and save new destinations.
Navigation	Manages transitions between screens using Jetpack Navigation Compose

4. Sensor and Feature Plan

- GPS Sensor: Will be added in Milestone 2 to log travel locations.
- Accelerometer: Planned for detecting motion or step count during travel.
- Local Storage: SQLite or Room Database will be used in Milestone 3 for o\line persistent

5. Screenshots



6. Reflection

This milestone successfully established the base app structure, connected screens with Compose Navigation, and implemented placeholder UIs for interaction. The process improved my understanding of modern Android development with Kotlin and Compose