

System Design Document

Navi **Multi-Platform Travel Planner**

An EECS3311 F25 Project by Group AK:

Klodiana Kamberi
Tjioe Andrew Elvio Febrian
Akash Deep
Trung Hieu Nguyen
Trong Duc Nguyen

Table of Contents

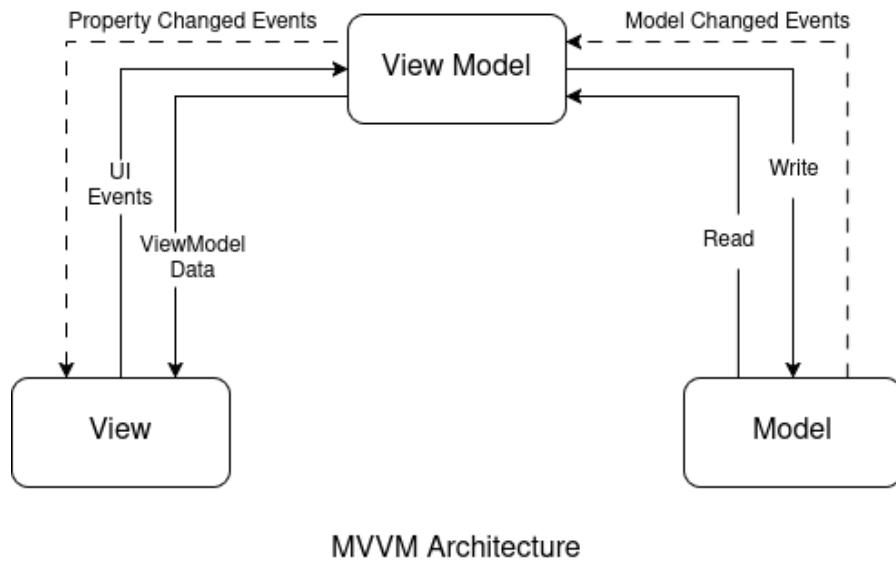
Overview	3
Software Architecture Diagram	4
CRC	5
Class Diagram	7
Use-Case Diagram	8
UserFlow	9

Overview

Navi is a travel planning application designed to make trip organization collaborative and enjoyable. With Navi, users can seamlessly coordinate their travel experiences by managing events within trips, organizing schedules through its itinerary features, and visualizing event locations interactively on a map. The intuitive interface ensures users can easily add, modify, and review trip details, making group planning both efficient and engaging.

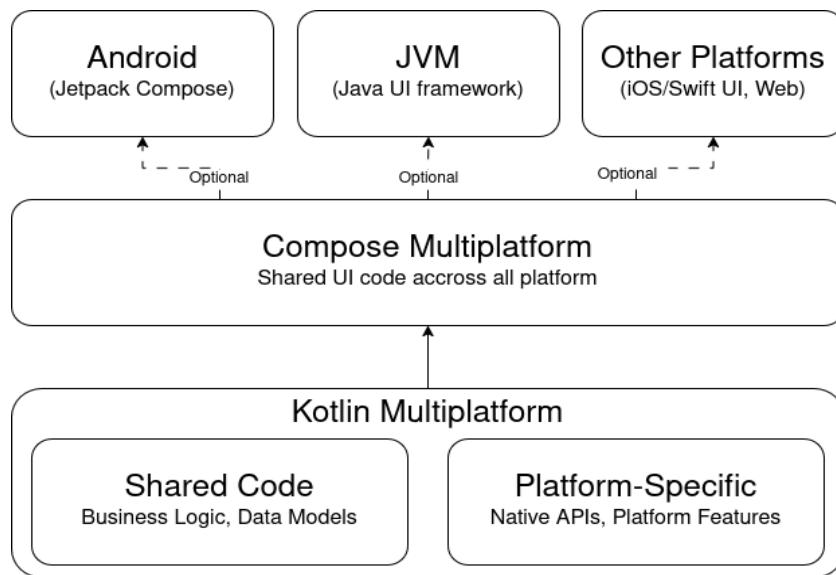
Navi is built with a robust technical foundation focused on delivering performance and flexibility across multiple platforms. Our development leverages Kotlin Multiplatform paired with Compose Multiplatform to simultaneously target iOS, Android, web and JVM environments. The application is backed by a Ktor-based server and a PostgreSQL database for reliable storage and API services. Additionally, a separate Python server is integrated to deliver advanced AI-driven features, ensuring a comprehensive travel planning solution.

Software Architecture Diagram



Reference:

<https://www.geeksforgeeks.org/android/mvvm-model-view-viewmodel-architecture-pattern-in-android/>



Kotlin Multiplatform Architecture

Reference:

<https://student.cs.uwaterloo.ca/~cs346/1251/slides/CS346-11-01-Kotlin-Multiplatform.pdf>

CRC Cards

<u>Class Name:</u> User	
<u>Parent Class:</u> - <u>Subclasses:</u> -	
<u>Responsibilities:</u> <ul style="list-style-type: none">- Knows name- Knows userID- Knows email- Knows list of trips- Makes trips- Edits trips- Manage account info	<u>Collaborators:</u> <ul style="list-style-type: none">• Trips

<u>Class Name:</u> Trip	
<u>Parent Class:</u> - <u>Subclasses:</u> -	
<u>Responsibilities:</u> <ul style="list-style-type: none">- Knows trip title- Knows list of events- Knows list of users- Knows trip duration- Knows header image URL- Make and edit events	<u>Collaborators:</u> <ul style="list-style-type: none">- Events- Duration

<u>Class Name:</u> Event	
<u>Parent Class:</u> - <u>Subclasses:</u> -	
<u>Responsibilities:</u> <ul style="list-style-type: none">- Knows event title- Knows duration- Knows event description- Knows event location	<u>Collaborators:</u> <ul style="list-style-type: none">- Location- Duration

<u>Class Name:</u> Duration	
<u>Parent Class:</u> <u>Subclasses:</u>	

Vars:

- localDate startDate
- localTime startTime
- localDate endDate
- localTime endTime

Responsibilities:

- Knows start and end time and date
- Provides conflict check between durations

Collaborators:

Class Name: **Location**

Parent Class: -

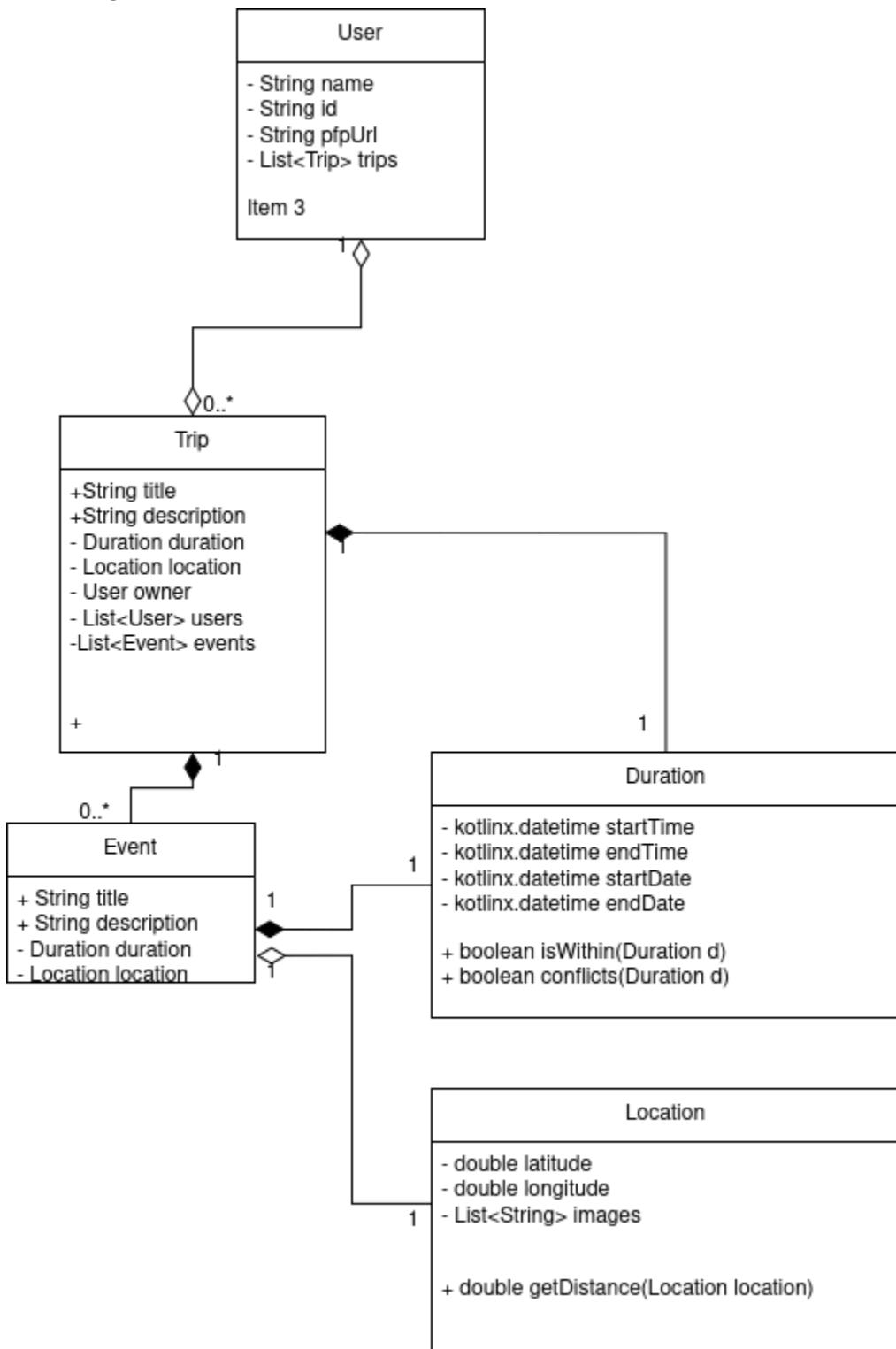
Subclasses: -

Responsibilities:

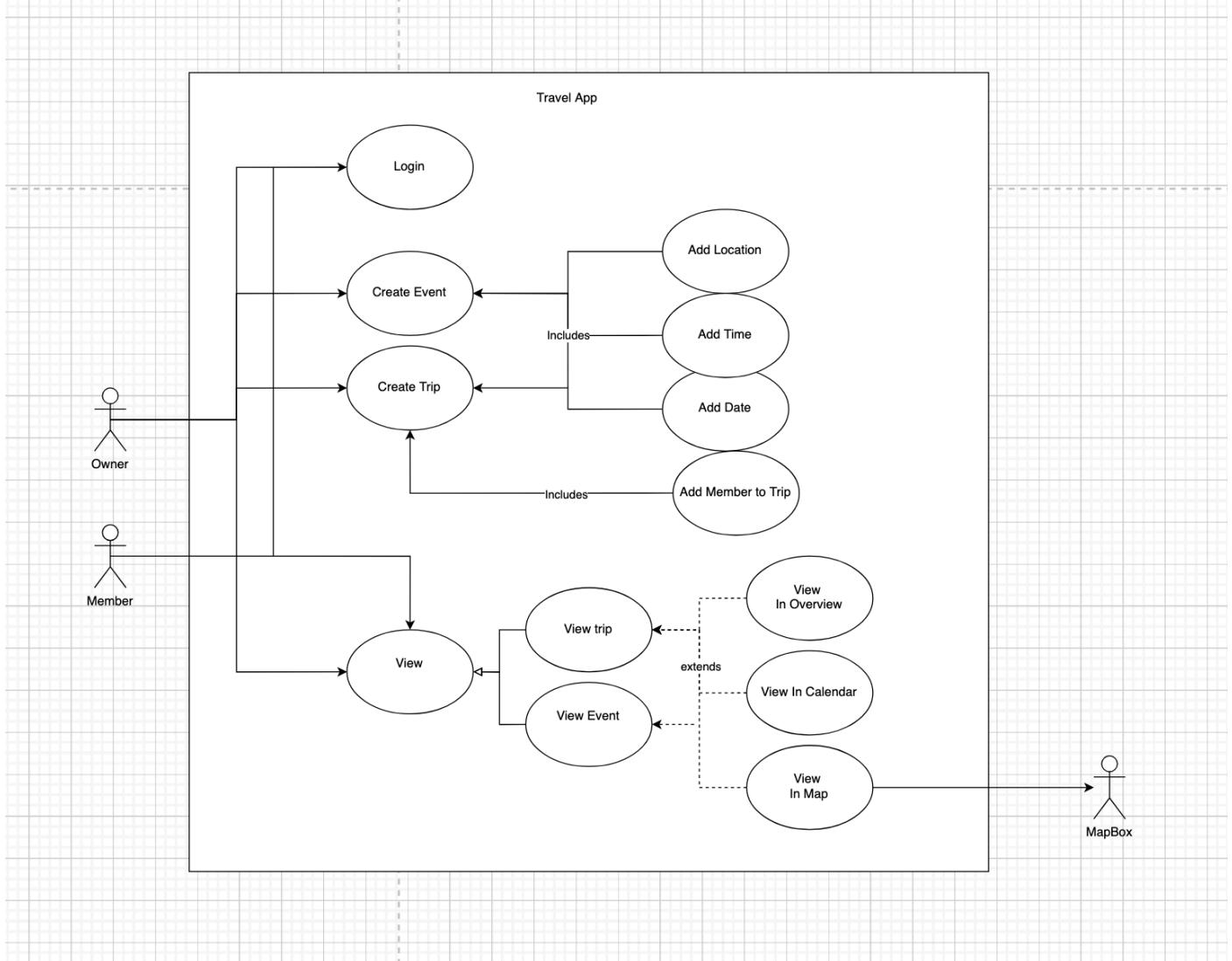
- Knows title
- Knows GPS coordinate
- Knows image of location if any

Collaborators:

Class Diagram



Use-Case Diagram



User-Flow

