Software Requirements Specification

for

TripTally

Version 1.0 approved

Prepared by Tan Wei Song Russell Bryan, Justin Woon Thean Woon, Ethan Jared Chong Rui Zhi, Evelyn Theresia Cuaca, Parvez Kurniawan Wijaya

Nanyang Technological University

9/11/2025

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) document provides a comprehensive description of the TripTally mobile application. It defines the system's functional and non-functional requirements, which serve as a blueprint for the design, development, and validation phases of the project.

TripTally is designed to deliver intelligent route planning for users in Singapore by incorporating real-time traffic information, multi-modal transportation options, and environmental impact metrics to support sustainable travel choices.

1.2 Document Conventions

This document follows specific conventions to ensure clarity and consistency in describing system requirements:

- Shall / Must - Denotes *mandatory* requirements that the system must fulfill.
- Should - Denotes *recommended* requirements that are desirable but not strictly necessary.
- May - Denotes *optional* requirements that can be implemented at the developer's discretion.
- FR - Refers to a *Functional Requirement*, defining specific system behavior or functionality.
- NFR - Refers to a *Non-Functional Requirement*, specifying quality attributes such as performance or security.

- Priority Levels:
    - High (Critical): Essential for core functionality.
    - Medium (Important): Enhances usability or performance but not critical for operation.
    - Low (Nice-to-have): Optional features that improve user experience or add value.

1.3 Intended Audience

This document is intended for the following audiences and their respective purposes:

- Developers - To serve as a reference for system implementation, integration, and maintenance.
- Testers - To guide the creation of test cases and verification procedures that ensure system compliance with requirements.
- Project Managers - To support project planning, progress tracking, and resource allocation.
- End Users - To provide an overview of the system's features, capabilities, and intended benefits.
- Stakeholders - To validate that the documented requirements align with business objectives and user needs.

1.4 Project Scope

TripTally is a mobile application designed to help users efficiently plan optimal and economical routes across Singapore. The system provides a seamless travel experience by integrating multiple transportation modes and real-time data sources. Key features include:

- Multi-modal route planning - supports driving, public transport, walking, and cycling.
- Live traffic monitoring - displays current traffic conditions, congestion levels, and incident alerts.
- AI-powered travel time forecaster based on patterns from traffic cameras.
- Comprehensive travel metrics - calculates cost, duration, distance, and estimated carbon emissions for each route.
- Personalization features - allows users to save favorite locations and frequently used routes.
- Community reporting - enables users to report road incidents or hazards in real time.

- Smart recommendations - provides personalized route suggestions based on user behavior and preferences.

TripTally integrates with external APIs, including Google Maps, TomTom, and LTA DataMall, to deliver accurate, up-to-date, and context-aware routing services throughout Singapore.

2. Overall Description

2.1 Product Perspective

TripTally is a standalone mobile application designed to deliver intelligent route planning and travel insights for users in Singapore. Current map applications such as Google Maps and Waze lack the local information of routes, leaving them less effective for locals who know these kinds of shortcuts. To solve this, one of TripTally's features enables users to share their custom routes. TripTally also advances Singapore's plans to become a smart nation by advocating on sustainability and technology through its eco-friendly and complex features.

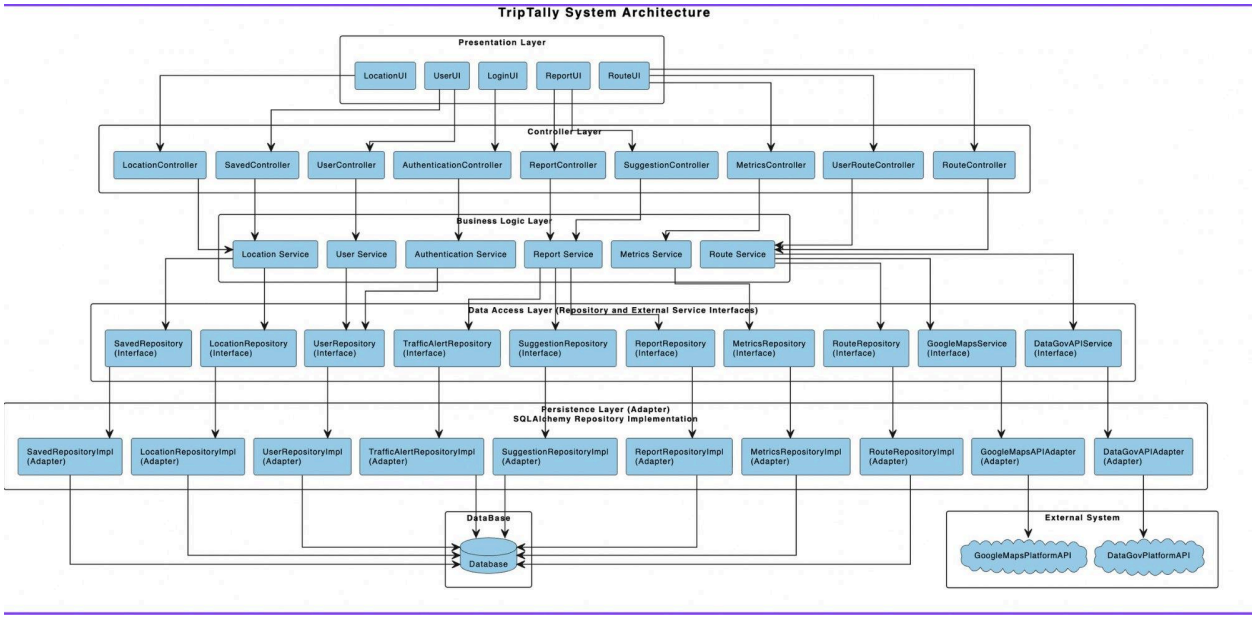
The system operates as an independent product but relies on multiple external APIs and services to provide real-time transportation, traffic, and environmental data.

System Context

TripTally's architecture comprises several interconnected components:

- Frontend: Developed using React Native and Expo, supporting both iOS and Android platforms to provide a responsive and user-friendly interface.
- Backend: HTTPX and FastAPI for RESTful API integration, Pydantic for validation, OpenCV and Ultralytics for processing traffic camera information, Pandas for data processing, and Docker for containerization.
- Database: Utilizes PostgreSQL for persistent data storage, managing user profiles, saved routes, and historical data, and Redis for server-side caching of traffic camera information and congestion forecast.
- External APIs:
    - Google Maps API: Directions, Geocoding, Places, and Photos.
    - TomTom API: Real-time Traffic Flow and Traffic Incident data.
    - LTA DataMall: Bus services, MRT fares, and carpark availability information.
    - data.gov.sg: Traffic camera information.

Architecture Pattern



TripTally follows a modular and maintainable architecture that separates concerns across layers and ensures scalability and testability:

- Hexagonal Architecture (Ports and Adapters): Enables loose coupling between the core business logic and external systems.
- Distributed MVC Pattern: Provides clear separation between data models, user interfaces, and control logic.
- Repository Pattern: Manages data persistence and retrieval operations efficiently.
- Service Layer: Encapsulates business logic and supports modular feature development

2.2 Product Functions

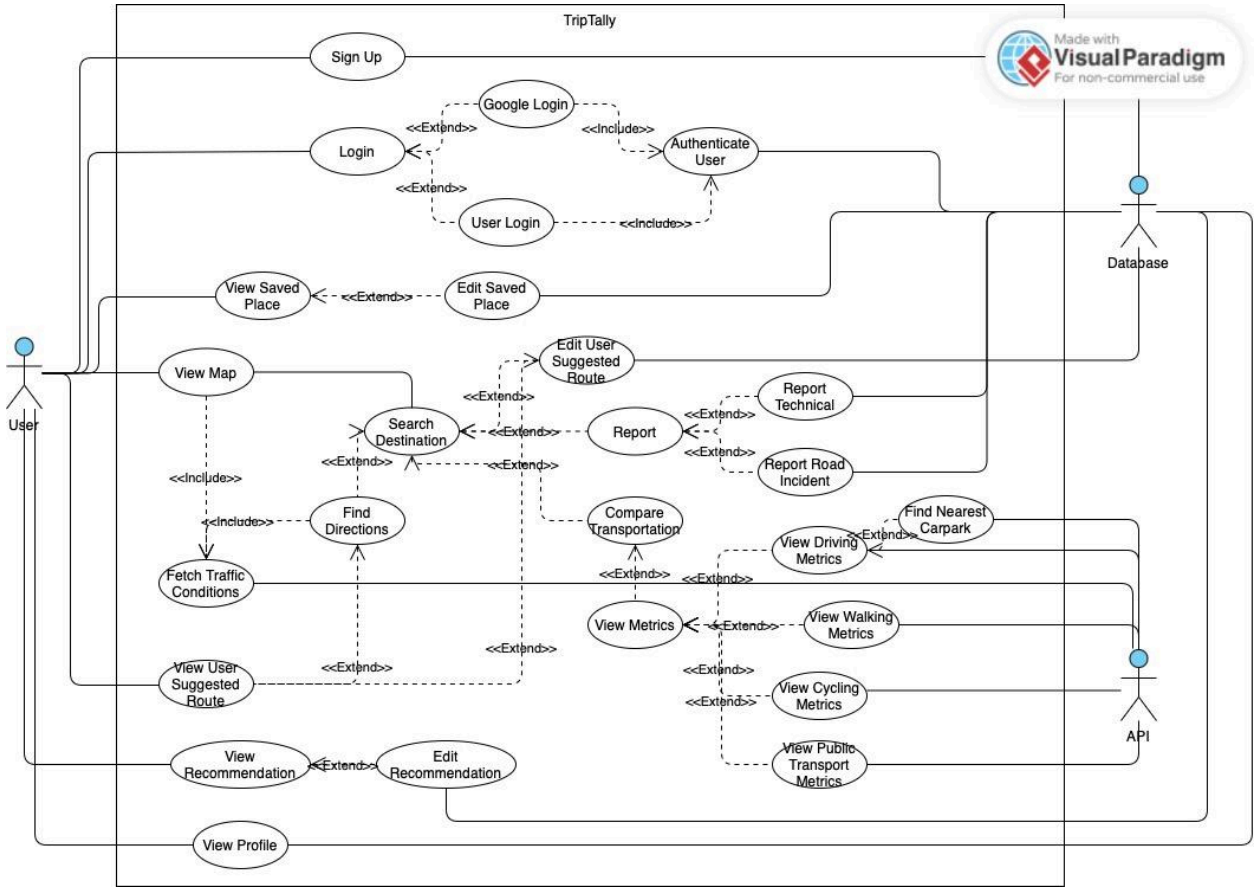
The TripTally system provides a comprehensive set of core functionalities designed to enhance route planning, travel convenience, and user experience. The following summarizes its key functions:

Core Functions

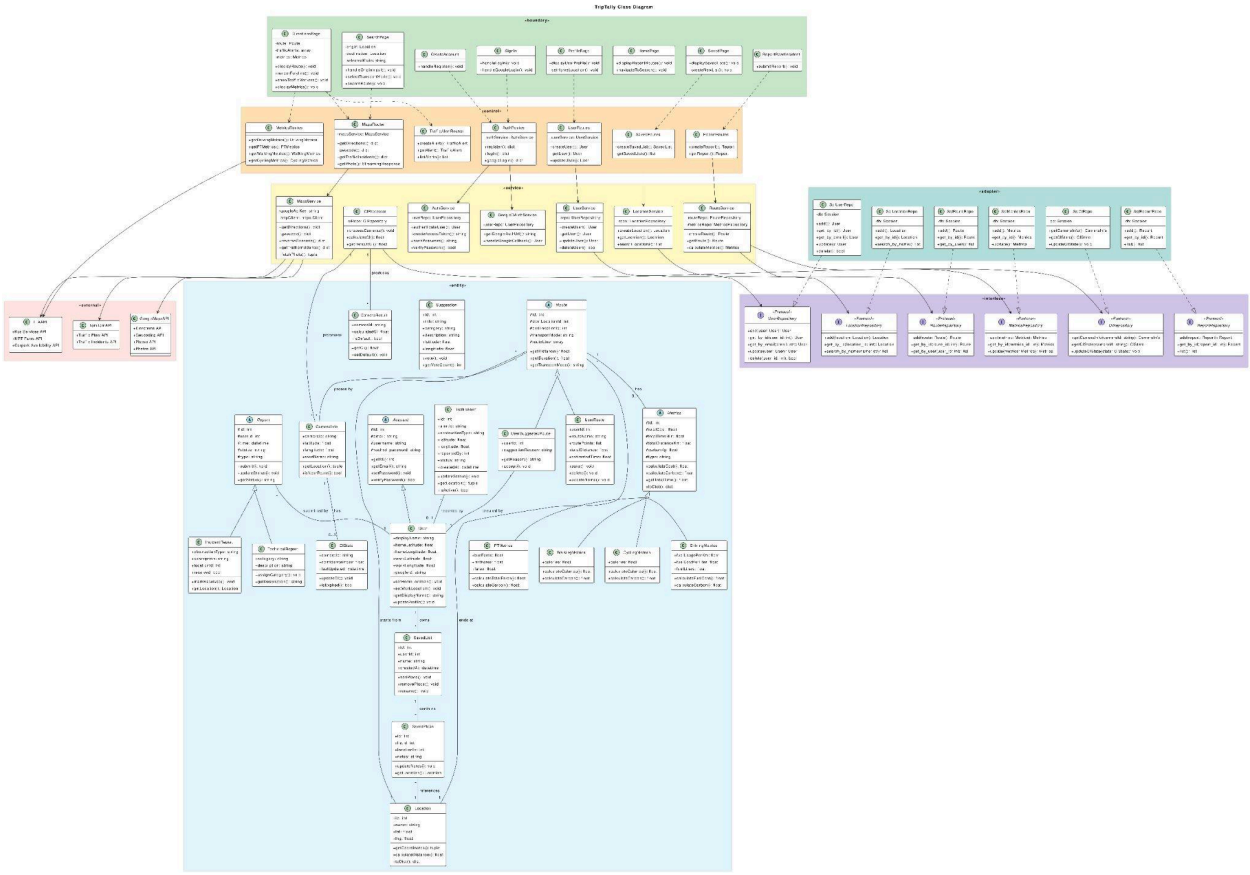
1. Route Planning
   Calculates the most optimal routes based on user preferences such as travel mode, time, and cost.
2. Multi-Modal Transportation
   Supports multiple transport modes, including driving, public transport, walking, and cycling, allowing users to compare and select the most suitable option.

3. Real-Time Traffic Monitoring
   Displays up-to-date traffic conditions, congestion levels, and reported incidents along selected routes.
4. Real-Time Travel Time Forecasting
   Based on monitored traffic congestion levels, forecasts the best time to leave.
5. Travel Metrics Calculation
   Provides detailed route metrics such as estimated travel time, total distance, travel cost, and carbon emission estimates.
6. User Management
   Enables account creation, secure authentication, and personalized profile management.
7. Location Management
   Allows users to save frequently used locations such as home, work, and other favorite destinations.
8. Route Saving and Management
   Lets users bookmark, categorize, and manage previously searched or frequently used routes.
9. Incident Reporting
   Empowers users to report road incidents or hazards, contributing to community-driven traffic updates.
10. Route Visualization
    Displays calculated routes on an interactive map interface with visual indicators such as colored polylines, markers, and traffic overlays.
11. User Route Suggestions
    Users give suggestions on walking and cycling routes which could be faster compared to the recommended routes which are done by tracking users' location.
12. User Recommendations
    Display user Recommendations for different places which are filtered by likes.

2.2.1 Use Case Diagram



2.2.2 Class Diagram



2.3 User Classes and Characteristics

The TripTally application is designed to serve two primary categories of users with distinct usage behaviors and needs.

Primary User Class – Commuters

- Characteristics: Individuals who travel daily within Singapore for work, school, or personal activities.
- Technical Expertise: Basic to moderate familiarity with mobile applications and navigation tools.
- Frequency of Use: Daily or multiple times per day.
- Key Needs: Quick route generation, accurate real-time traffic updates, cost and time comparisons, and reliable multi-modal transport options.

Secondary User Class – Occasional Travelers

- Characteristics: Tourists or infrequent travelers exploring routes or attractions across Singapore.
- Technical Expertise: Basic smartphone proficiency.
- Frequency of Use: Occasional or short-term usage during trips.
- Key Needs: Intuitive user interface, simple location search, easy access to saved or suggested destinations.

2.4 Operating Environment

TripTally operates in a distributed client–server environment comprising mobile clients and a cloud-hosted backend infrastructure.

Client-Side Environment

- Platform: iOS 13.0+ and Android 8.0+
- Display: Optimized for mobile screens ranging from 4.7″ to 6.7″
- Network: Requires an active internet connection (3G / 4G / 5G / Wi-Fi)
- Location Services: GPS or device location services must be enabled for real-time tracking
- Storage: Minimum of 100 MB of free storage space required

Server-Side Environment

- Operating System: Linux-based cloud server
- Runtime Environment: Python 3.12 or higher
- Memory: At least 16 GB of RAM, 32 GB recommended
- Storage: At least 10 GB of free storage space required
- Database: PostgreSQL 14 or higher, Redis 7 or higher
- Web Server: Uvicorn ASGI server for handling concurrent requests
- Network: Stable high-speed internet connection required for API communication

2.5 Design and Implementation Constraints

The development and deployment of TripTally are subject to the following constraints:

1. Technology Constraints

- The mobile application must be developed using React Native to ensure cross-platform compatibility.

- The backend must be implemented in Python using the FastAPI framework.
- Integration is required with specified external APIs, including Google Maps, TomTom, and LTA DataMall.

2. Regulatory Constraints

- The system must comply with Singapore's Personal Data Protection Act (PDPA) to ensure user data privacy.
- Google Maps Platform Terms of Service must be followed for API usage.
- LTA DataMall API policies must be adhered to for accessing public transport and traffic data.

3. Business Constraints

- Limited by API rate quotas from external data providers.
- Budget restrictions for paid API usage and server hosting.
- Development timeline must align with project milestones and academic deliverables.

4. Security Constraints

- User passwords must be hashed using bcrypt before storage.
- API keys must be securely stored in environment variables or configuration vaults.
- All communication between client and server must use HTTPS for data encryption and protection

2.6 Assumptions and Dependencies

Assumptions

1. Users possess smartphones equipped with GPS functionality.
2. Users have a stable internet connection during application use.
3. External APIs remain available and maintain their current specifications.
4. Singapore's public transport data continues to be accessible via the LTA DataMall API.

Dependencies

1. Availability and uptime of the Google Maps Platform.
2. Functionality of the TomTom Traffic API.

3. Accessibility of the LTA DataMall API for transport and parking data.
4. Accessibility of the Data.gov API for traffic camera images.
5. Continued compatibility and updates of third-party libraries (React Native, FastAPI, SQLAlchemy, etc.).
6. Google OAuth service availability for user authentication and social login.

3. System Features (Functional Requirement)

This section describes the functional requirements of the TripTally system through detailed use case specifications.
Each use case represents one or more functional requirements that define the system's expected behavior and interactions between users and the system.

3.1 SignUp

Use Case ID:	#1-1		
Use Case Name:	SignUp		
Created By:	Russell Tan	Last Updated By:	Russell Tan
Date Created:	01/09/2025	Date Last Updated:	01/09/2025

Actor:	User
Description:	Creation of a new TripTally account of a user
Preconditions:	1. App must be installed and the user's email/ contact number has not been registered. 2. Internet connection must be available.
Postconditions:	1. New account created and verified. 2. Confirmation message sent to the users email.
Priority:	Low
Frequency of Use:	High

Flow of Events:	1. User selects Sign Up 2. User enters name, email, password and contact number. User can also sign up through Apple ID or Google. 3. System validates that the email exists and is valid and checks the password strength. 4. User agrees to the Terms and Services of the application. 5. Account is then created and confirmation is sent to the email that has created the account.
Alternative Flows:	AF-1: Sign up available with Apple/Google.
Exceptions:	EX-1: If there is a duplicate email account, system will prompt that account already exists and prompts user login EX-2: A weak password would result in the system prompting to change the password to have at least 8 characters with a mix of alphabets, numbers and special characters. EX-3: No internet would result in a retry option.
Includes:	1. Login 2. AuthenticateUser
Special Requirements:	Passwords must be a minimum of length 8, containing both alphabets, numbers and at least one special character.
Assumptions:	None
Notes and Issues:	None

3.2 Login

Use Case ID:	#1-2		
Use Case Name:	Login		
Created By:	Russell Tan	Last Updated By:	Russell Tan
Date Created:	01/09/2025	Date Last Updated:	01/09/2025
Actor:	User		

Description:	Start a user session with email and the corresponding password.
Preconditions:	1. User has a registered account and clicks the Login button. 2. Internet connection must be available
Postconditions:	1. User has been authenticated and secure session active
Priority:	Low
Frequency of Use:	Low
Flow of Events:	1. User clicks Login button. 2. User enters email/contact number with the correct password. 3. AuthenticateUser is called. 4. User is redirected to the Home page of the application if AuthenticateUser is successful.
Alternative Flows:	AF-1: Wrong password/ username ● If User enters their password incorrectly for 5 tries, their account would be locked for an hour. ● Whenever the User gets their credentials wrong before exceeding 5 tries, a message saying “invalid login/password” will be displayed instead. ● The system will prompt user to re-enter their username and password AF-2: Network issues would bring back the user to the sign in page.
Exceptions:	None
Includes:	AuthenticateUser
Special Requirements:	None
Assumptions:	Assume user enters the right password on the first try
Notes and Issues:	None

3.3 AuthenticateUser

Use Case ID:	#1-3		
Use Case Name:	AuthenticateUser		
Created By:	Russell Tan	Last Updated By:	Russell Tan

Date Created:	01/09/2025	Date Last Updated:	01/09/2025

Actor:	ExternalSystem
Description:	Process of verifying that the user's account exists and matches the password before logging them in.
Preconditions:	1. Login has been initiated. 2. Internet connection must be available.
Postconditions:	1. User securely authenticated and login successful.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. System receives the login credentials from the User. 2. System fetches data matching the account. 3. System checks the status of the account and ensures that it is not locked. 4. System then verifies the user's password. 5. System resets the retry fail count. 6. System returns a true and user is authenticated
Alternative Flows:	AF-1: If the account is locked, system displays "Your account has been locked" to the User.
Exceptions:	EX-1: If user cancels at the account selection, display a retry option
Includes:	1. SignUp 2. Login
Special Requirements:	None
Assumptions:	Account that the user is logging in exists.
Notes and Issues:	None

3.4 ManageSavedPlaces

Use Case ID:	#2-1		
Use Case Name:	ManageSavedPlaces		
Created By:	Russell Tan	Last Updated By:	Russell Tan

Date Created:	01/09/2025	Date Last Updated:	01/09/2025

Actor:	User
Description:	User have the ability to manage their saved locations. (Home, Work, Favourites)
Preconditions:	1. User must be logged in. 2. Internet connection must be available
Postconditions:	1. User has successfully added/edited/deleted the saved location.
Priority:	Low
Frequency of Use:	Low
Flow of Events:	1. User selects the “Saved Places” button. 2. User is able to add, edit or delete the location selected by the user. 3. System updates database and syncs profile. 4. Newly added, edited or deleted location is displayed for the user.
Alternative Flows:	AF-1: Duplicate places shall prompt a rename of the location to the user.
Exceptions:	EX-1: If invalid location is selected, an error will be displayed and bring the user back to the home page.
Includes:	None
Special Requirements:	None
Assumptions:	User logged in.
Notes and Issues:	None

3.5 ViewMap

Use Case ID:	#2-2		
Use Case Name:	ViewMap		
Created By:	Russell Tan	Last Updated By:	Russell Tan
Date Created:	01/09/2025	Date Last	01/09/2025

		Updated:	

Actor:	User, ExternalSystem
Description:	Displays interactive map that is centered on the user's location.
Preconditions:	1. User is logged in. 2. Internet connection must be available.
Postconditions:	1. Map is displayed with the current location displayed to the user.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User opens the home page/ clicks on the explore button. 2. System fetches current location and GoogleMaps API. 3. System displays an interactive map with context.
Alternative Flows:	AF-1: User can manually input their address when GPS is off.
Exceptions:	None
Includes:	None
Special Requirements:	None
Assumptions:	User mobile devices have granted permission to view the user's current location.
Notes and Issues:	None

3.6 SearchDestination

Use Case ID:	#2-3		
Use Case Name:	SearchDestination		
Created By:	Russell Tan	Last Updated By:	Russell Tan
Date Created:	01/09/2025	Date Last Updated:	01/09/2025

Actor:	User, ExternalSystem
Description:	Displays the selected location to the user.
Preconditions:	1. Origin and Destination must be known. 2. Internet connection must be available.
Postconditions:	1. The selected location is displayed on the map for the user with the information regarding the location.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. System displays cached recent searches. 2. User keys in their destination in the search bar. 3. System queries geolocation API to search for the destination. 4. Location is displayed with its information and directions for the user. 5. Users can then select mode of transportation for the directions to their destination. 6. Route calculation is triggered. 7. Multiple routes to the users destination is displayed
Alternative Flows:	AF-1: Destination picked is from recent or Saved Places

Exceptions:	EX-1: Geocoding API is down. EX-2: Invalid location is inputted.
Includes:	1. ViewMap
Special Requirements:	None
Assumptions:	Interactive map is loaded.
Notes and Issues:	None

3.7 SearchSuggestion

Use Case ID:	#2-4		
Use Case Name:	SearchSuggestion		
Created By:	Russell Tan	Last Updated By:	Russell Tan
Date Created:	01/09/2025	Date Last Updated:	01/09/2025

Actor:	User, ExternalSystems
Description:	Display live search suggestions as the user types into the search bar.
Preconditions:	1. User has keyed in a location in the search bar. 2. Internet connection must be available.
Postconditions:	1. Search options displayed for users to select.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User keys in a destination in the search bar. 2. System queries an external search API. 3. Suggestions based on the user's search are displayed. 4. User is able to select one of the suggestions that is displayed.
Alternative Flows:	None
Exceptions:	EX-1: Location/API unavailable would display a “No match/ Try again” text to the user.
Includes:	1. ViewMap 2. SearchDestination
Special Requirements:	None
Assumptions:	User has keyed in a valid location in the search bar.
Notes and Issues:	None

3.8 CompareTransportation

Use Case ID:	#2-5		
Use Case Name:	CompareTransportation		
Created By:	Russell Tan	Last Updated By:	Russell Tan
Date Created:	01/09/2025	Date Last Updated:	01/09/2025

Actor:	User, ExternalSystem
Description:	Side by side comparison of two modes of transportation. (Time, Cost, Carbon Footprint)
Preconditions:	1. Origin and destination must be known. 2. Internet connection must be available
Postconditions:	1. User able to see side by side comparison of the 2 modes of transportation selected.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User select the 2 modes of transportation that they would like to compare. 2. System fetches data of the different modes of transportation from their various APIs. 3. System compares the selected modes of transportation (Driving, Public Transport, Walking, Cycling) 4. System calculates the ETA, distance, cost and carbon footprint. 5. This data is then displayed to the user for comparison.
Alternative Flows:	1. AF-1: User is able to view a brief comparison of all 4 metrics if the compare button is clicked without any selection of the 2 modes.
Exceptions:	EX-1: If one mode is down due to the API being down, display historical data and inform users that data is based on past information.
Includes:	1. ViewMap 2. SearchDestination
Special Requirements:	None
Assumptions:	Origin and destination known.
Notes and Issues:	None

3.9 Report

Use Case ID:	#2-6		
Use Case Name:	Report		
Created By:	Russell Tan	Last Updated By:	Russell Tan
Date Created:	01/09/2025	Date Last Updated:	01/09/2025

Actor:	User
Description:	User can submit report about incident on road or system error faced while using the app
Preconditions:	1. The user is logged in. 2. Network is available.
Postconditions:	1. User is saved in the system with “Submitted” status 2. Admin are automatically notified and the report is placed in the review queue
Priority:	High
Frequency of Use:	Medium
Flow of Events:	1. User open submit report option 2. Systems prompt user to choose type of report (road incident/system error) 3. User select reporting road incident or system error 4. The system displays the form where users can describe the issue and upload relevant supporting material such as picture/video. 5. System validate, store the report and confirm submission. 6. System notify admin queue for review.
Alternative Flows:	AF-1:Road incident report 1. Users input the location of the incident. 2. System prompt user to select incident categories. 3. Users can optionally add description and photos/videos. 4. Once submitted, report is saved with location, time, and marked as submitted 5. Admins are notified and incident is placed into the review queue AF-2:System error report 1. The User selects the category of error (fare calculation error, route error, data error etc.)

	2. The User is prompted to provide a description of the error. 3. User can optionally provide photo/video
Exceptions:	EX-1: Missing required information- System will highlight the required fields and prevent submission. EX-2: File upload error- System provide option to submit report without
Includes:	1. ManageObstructionReport 2. ManageSystemReport
Special Requirements:	Status of Report includes: Submitted, Pending, Resolved
Assumptions:	None
Notes and Issues:	Verified incidents will affect the routing.

3.10 SuggestRoute

Use Case ID:	#2-7		
Use Case Name:	SuggestRoute		
Created By:	Russell Tan	Last Updated By:	Russell Tan
Date Created:	01/09/2025	Date Last Updated:	01/09/2025
Actor:	User		
Description:	User submits different walking /cycling routes from one location to another than the ones generated by the system.		
Preconditions:	1. User is logged in to a valid account 2. Internet connection must be available		
Postconditions:	1. User gets confirmation that their suggested route has been successfully uploaded.		
Priority:	Low		
Frequency of Use:	Low		

Flow of Events:	1. The User selects the option to upload a new route. 2. The User selects from a range of tags that accurately describe the type of route they are uploading. 3. The User is then prompted to search for their start and end points of their destination using the search bar. 4. Once both points are confirmed, the User will tap on the screen to place “checkpoint” pins on the map to show exactly where the route will lead. 5. At each checkpoint the User will have the option to indicate whether the route will continue indoors or outdoors, and also upload helping images. 6. Once all the checkpoints and details are confirmed, the User is prompted to give the route a name and short written description. 7. The User will then tap on the “upload” button to upload the route to the forum.
Alternative Flows:	None
Exceptions:	EX-1: If the forum’s servers are down, return a message stating “The servers are currently down, please come back later”.
Includes:	1. ViewMap 2. SearchDestination
Special Requirements:	None
Assumptions:	The User is not banned from posting routes.
Notes and Issues:	None

3.11 View Driving Metrics

Use Case ID:	#3-1		
Use Case Name:	ViewDrivingMetrics		
Created By:	Ethan Jared Chong Rui Zhi	Last Updated By:	Ethan Jared Chong Rui Zhi
Date Created:	1/8/2025	Date Last Updated:	1/8/2025
Actor:	User		
Description:	User is shown metrics for driving to their destination.		
Preconditions:	1. User has searched for a destination and selected the “driving” mode of transportation.		

Postconditions:	1. The User gets shown the corresponding metrics for driving.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	1. User searches for destination 2. User confirms destination 3. User selects “driving” as mode of transportation 4. Retrieve ERP pricing data, destination parking lot data (call FindNearestCarpark), fuel consumption data, fuel price data, vehicle CO2 emissions data, travel duration. 5. Calculate total cost. 6. Display drop-down table that shows CO2 emission, travel time and total cost.
Alternative Flows:	None
Exceptions:	EX-1: If driving cannot reach the destination, the option to select “driving” will be disabled. EX-2: If some required information cannot be fetched, “note: error retrieving live data” message will be displayed in the corresponding table row and display historical data.
Includes:	FindNearestCarpark
Special Requirements:	User’s phone location services are enabled for TripTally.
Assumptions:	None
Notes and Issues:	None

3.12 View Public Transport Metrics

Use Case ID:	#3-2		
Use Case Name:	ViewPublicTransportMetrics		
Created By:	Ethan Jared Chong Rui Zhi	Last Updated By:	Ethan Jared Chong Rui Zhi
Date Created:	1/8/2025	Date Last Updated:	1/8/2025

Actor:	User
Description:	User is shown metrics for taking public transport to their destination.
Preconditions:	1. User has searched for a destination and selected the “public transport” mode of transportation.
Postconditions:	1. The User gets shown the corresponding metrics for taking public transportation.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User searches for destination 2. User confirms destination 3. User selects "public transport” as mode of transportation 4. Retrieve total trip fares, CO2 emissions data, travel duration. 5. Calculate total cost. 6. Display drop-down table that shows CO2 emission, travel time and cost breakdown of the fares for each leg of the trip.
Alternative Flows:	None
Exceptions:	EX-1: If any required information is missing or invalid, an error message is displayed. EX-2: If public transportation cannot reach the destination, the option to select “public transportation” will be disabled. EX-3: If public transportation is out of operating hours, display “Public Transport currently out of operating hours.”
Includes:	None
Special Requirements:	User’s phone location services are enabled for TripTally.
Assumptions:	None

Notes and Issues:	None

3.13 View Walking Metrics

Use Case ID:	#3-3		
Use Case Name:	ViewWalkingMetrics		
Created By:	Ethan Jared Chong Rui Zhi	Last Updated By:	Ethan Jared Chong Rui Zhi
Date Created:	1/8/2025	Date Last Updated:	9/11/2025

Actor:	User
Description:	User is shown metrics for walking to their destination.
Preconditions:	1. User has searched for a destination and selected the “walking” mode of transportation.
Postconditions:	1. The User gets shown the corresponding metrics for walking to their destination.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	1. User searches for destination 2. User confirms destination 3. User selects “walking” as mode of transportation 4. Retrieve time taken and distance 5. Display drop-down table that shows CO2 emission (zero), travel time and total cost (zero).

Alternative Flows:	None
Exceptions:	None
Includes:	None

Special Requirements:	User’s phone location services are enabled for TripTally.
Assumptions:	None
Notes and Issues:	None

3.14 View Cycling Metrics

Use Case ID:	#3-4		
Use Case Name:	ViewCyclingMetrics		
Created By:	Ethan Jared Chong Rui Zhi	Last Updated By:	Ethan Jared Chong Rui Zhi
Date Created:	1/8/2025	Date Last Updated:	1/8/2025
Actor:	User		
Description:	User is shown metrics for walking to their destination.		
Preconditions:	1. User has searched for a destination and selected the “cycling” mode of transportation.		
Postconditions:	1. The User gets shown the corresponding metrics for cycling to their destination.		
Priority:	High		
Frequency of Use:	Medium		
Flow of Events:	1. User searches for destination 2. User confirms destination 3. User selects “cycling” as mode of transportation 4. Retrieve calories burnt and time taken 5. Display drop-down table that shows CO2 emission (zero), travel time and total cost (zero).		
Alternative Flows:	None		
Exceptions:	None		
Includes:	None		
Special Requirements:	User’s phone location services are enabled for TripTally.		
Assumptions:	None		
Notes and Issues:	None		

Alternative Flows:	None
Exceptions:	None
Includes:	None
Special Requirements:	User’s phone location services are enabled for TripTally.
Assumptions:	None
Notes and Issues:	None

3.15 Fetch Traffic Data

Use Case ID:	#4-1		
Use Case Name:	Fetch Traffic Data		
Created By:	Evelyn Theresia Cuaca	Last Updated By:	Evelyn Theresia Cuaca
Date Created:	01/09/2025	Date Last Updated:	01/09/2025

Actor:	External System
Description:	The system is able to receive and process real-time traffic incidents such as accidents, closures, or roadworks from an external feed.
Preconditions:	1. Connection to traffic alert API is active. 2. The user is viewing a map, searching, or navigating.
Postconditions:	1. Incident data ingested and displayed in the app. 2. Routes and ETAs updated accordingly. 3. Logs and metrics recorded for monitoring.
Priority:	High
Frequency of Use:	High

Flow of Events:	1. The Traffic Alert System (External system) alerts the system about traffic conditions (accident, road closure). 2. The system normalizes and shows the data. 3. The incident is stored in the TripTally database including the timestamp and validity period 4. Routing engine checks whether the incident affects: ○ Active trips in progress. ○ New trip requests. 5. If an active route is affected, drivers receive a soft reroute suggestion. 6. Map view automatically overlays icons for incidents (warning symbols, red markers, etc.).
Alternative Flows:	AF-1: API Failure: If the traffic feed is unavailable, the system falls back to the last known incidents .User is warned that traffic data may be outdated. AF-2: High Latency: If data fetching takes more than 60s, the routing service uses default routes and applies the update once available.
Exceptions:	None
Includes:	None
Special Requirements:	Access to traffic API
Assumptions:	The System has access to traffic alert API
Notes and Issues:	None

3.16 External System Integration

Use Case ID:	#5-1		
Use Case Name:	ExternalSystemIntegration		
Created By:	Evelyn Theresia Cuaca	Last Updated By:	Parvez Kurniawan Wijaya
Date Created:	01/09/2025	Date Last Updated:	08/09/2025

Actor:	External System
Description:	Integrate with external systems (data provider) to get real-time data like routes, traffic incidents, ERP rates, carpark availability, and fuel prices,etc.
Preconditions:	1. Valid API keys/credentials for each external provider 2. Network access to provider endpoints. 3. Data structure and mappings are defined
Postconditions:	1. Data retrieved from external systems is made available to TripTally service
Priority:	High
Frequency of Use:	High
Flow of Events:	1. Mapping ● System calls Google Map API to identify user position ● The API fetch available driving, walking, cycling, and public transportation routes ● Map APIs will render preview of the routes 2. Traffic and Incidents ● TripTally get real-time traffic conditions and road closures from external traffic APIs ● Incident data is applied to update travelling time and rerouting logic. 3. ERP integration ● The system retrieves ERP gantry locations and rate tables by time/day from LTA ERP providers. ● ERP costs calculated per driving route. 4. Parking Integration ● Carpark APIs provide real-time availability and rates

	● The system will update parking charges and lot availability. 5. Fuel Prices ● TripTally retrieves current fuel price data from providers. ● Driving cost calculations is updated based on user's vehicle profile 6. Carbon Footprint ● The average emission factors per transport mode is retrieved from carbon services. ● The system calculates estimated CO_2 footprint per trip. 7. Public Transport Data ● Bus and train schedules retrieved from LTA DataMall. ● Fare tables pulled from fare providers, travelling times and costs are computed. 8. Calorie API ● Retrieve calorie expenditure estimates for active travel modes (walking, cycling). ● Use external health/fitness APIs to compute calories burnt. 9. Traffic Camera Data ● Open Government Data APIs provide information on traffic cameras and their image. ● TripTally analyzes the data, then forecasts the CI. ● TripTally calculates ETA according to the CI. ● The user is presented with the best time to leave, along with the ETA.
Alternative Flows:	AF-1: Data not available
Exceptions:	EX-1: Timeout - Return fallback values after threshold exceeded.
Includes:	1. ViewMap 2. ViewMetrics 3. CompareTransportation
Special Requirements:	● Critical feeds must be updated within ≤ 60s.

	• API responses must integrate into the app within ≤ 2.5s
Assumptions:	1. External providers maintain reliable APIs and SLAs. 2. TripTally has access to provider data. 3. Users may need to install third-party ride-hailing apps for deep linking.
Notes and Issues:	System must support multi-provider fallback Some private carparks may lack reliable data

3.17 Traffic Forecasting

Use Case ID:	#5-1		
Use Case Name:	TrafficForecasting		
Created By:	Parvez Kurniawan Wijaya	Last Updated By:	Parvez Kurniawan Wijaya
Date Created:	01/09/2025	Date Last Updated:	08/09/2025

Actor:	User, External System
Description:	The system analyzes real-time traffic camera images to assess current congestion levels and predicts future traffic conditions along a route. Using computer vision and machine learning, the system forecasts traffic congestion up to 2 hours ahead, enabling users to choose optimal departure times for their journey.
Preconditions:	1. User has selected an origin and destination for route planning 2. Traffic camera monitoring service is operational 3. Traffic cameras are streaming live images 4. Route has been calculated with identified waypoints
Postconditions:	1. Current congestion levels calculated for cameras along the route 2. Traffic forecasts generated for multiple future time intervals 3. Optimal departure time recommendations provided to user 4. Estimated travel times adjusted based on predicted traffic conditions
Priority:	High

Frequency of Use:	High
Flow of Events:	1. TripTally retrieves live traffic camera images from government data sources for cameras along the user's planned route 2. Computer vision system processes each camera image: ○ Detects and counts vehicles in the frame ○ Analyzes vehicle movement and density ○ Calculates congestion index for current conditions 3. System stores current traffic state for each camera location 4. Forecasting engine predicts future congestion: ○ Analyzes historical traffic patterns for each camera ○ Considers time of day, day of week, and trends ○ Generates congestion predictions from 2 to 120 minutes ahead 5. Departure optimizer evaluates all route cameras and determines best departure windows 6. User receives recommendations including: ○ Multiple departure time options (immediate and delayed) ○ Predicted traffic levels for each option ○ Adjusted travel time estimates ○ Overall confidence in predictions
Alternative Flows:	AF-3.1 Camera image unavailable: ● AF-3.1.1. System logs unavailable camera ● AF-3.1.2. Analysis continues with remaining cameras ● AF-3.1.3. User informed of reduced coverage if significant gaps exist AF-4.1 Limited historical data: ● AF-4.1.1. System uses current conditions as baseline prediction ● AF-4.1.2. Confidence level adjusted accordingly ● AF-4.1.3. Historical data accumulates over time for improved forecasts AF-6.1 Traffic predictions outdated: ● AF-6.1.1. System alerts user that data may be stale ● AF-6.1.2. Refresh is triggered automatically ● AF-6.1.3. User can proceed with current data or wait for update AF-8.1 No camera coverage for route: ● AF-8.1.1. User notified that traffic forecasting unavailable ● AF-8.1.2. Route planning proceeds without traffic optimization ● AF-8.1.3. Alternative routes with coverage suggested

Exceptions:	E1: Computer vision system failure ● System alerts administrators ● Service automatically restarts ● Cached predictions used temporarily E2: Data source connection loss ● System attempts automatic reconnection ● Cached data served to users ● Degraded service notification displayed E3: Traffic data API rate limits reached ● System waits for quota refresh ● Existing predictions continue to be served ● Users notified of potential delays E4: Invalid camera location data ● Problematic cameras excluded from analysis ● Error logged for administrator review ● Route analysis continues with valid cameras
Includes:	None
Special Requirements:	1. Traffic predictions delivered to user in less than 5 seconds 2. Individual camera failures don't crash entire system
Assumptions:	1. Government traffic camera data is available and updated regularly 2. Camera locations remain stable over time 3. Computer vision provides sufficient accuracy for congestion assessment 4. Historical pattern analysis provides reasonable future predictions for typical traffic 5. Users primarily plan trips within the next 2 hours 6. Camera coverage adequately represents major route conditions
Notes and Issues:	1. Current forecasting uses pattern-based prediction. Future versions may incorporate advanced machine learning for improved accuracy

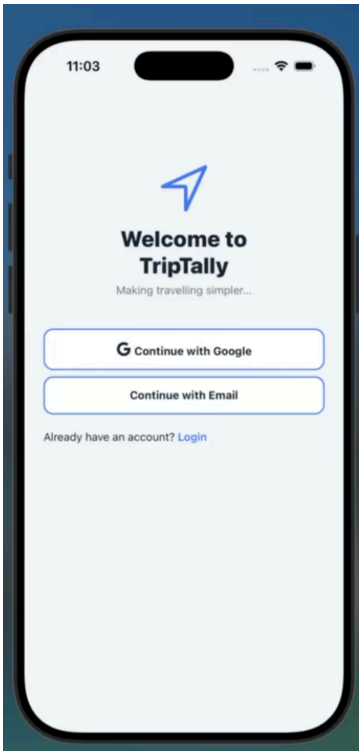
4. External Interface Requirements

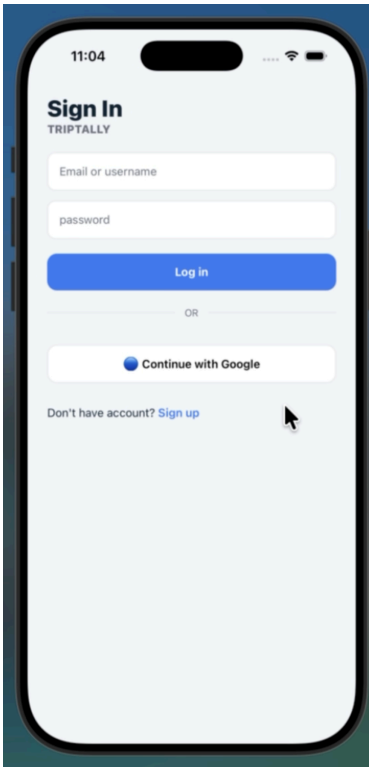
4.1 User Interfaces

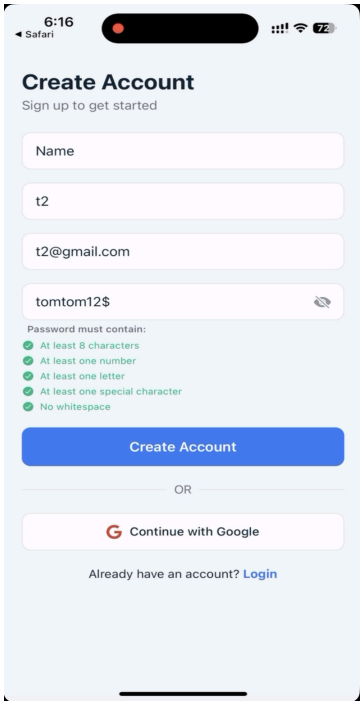
4.1.1 General UI Requirements

ID	Requirement	Description
UI-1	Responsiveness	The interface shall adapt to various screen sizes (4.7″ – 6.7″).
UI-2	Orientation	The interface shall support both portrait and landscape orientations.
UI-3	Platform Guidelines	The interface shall follow Material Design (Android) and iOS Human Interface Guidelines.
UI-4	Feedback	The interface shall provide visual feedback for all user interactions.
UI-5	Loading States	The interface shall display loading indicators during asynchronous operations.
UI-6	Error Handling	The interface shall display clear error messages for failed operations.

4.1.2 Screen Requirements

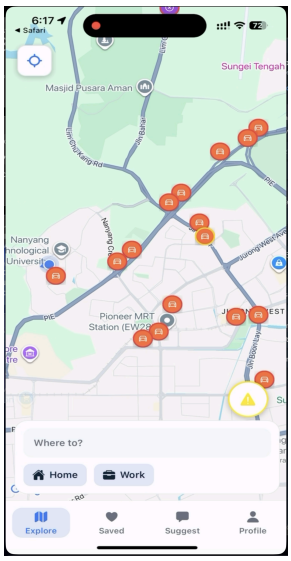


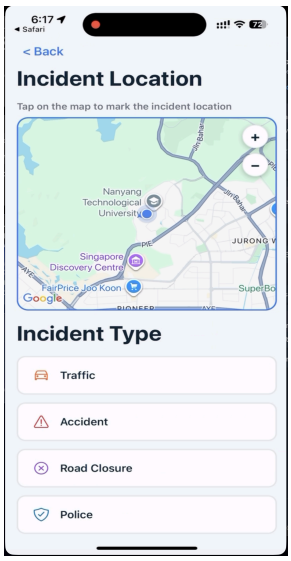


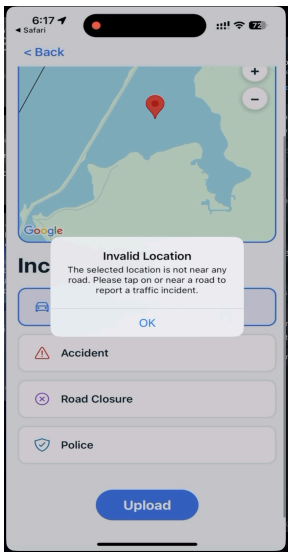


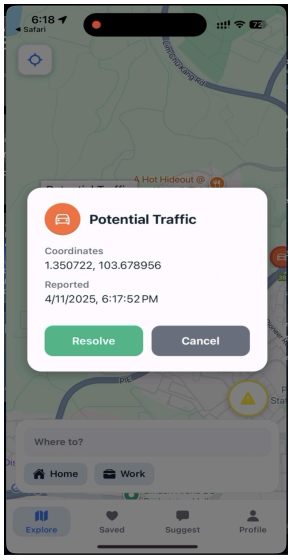
Login / Signup Screen

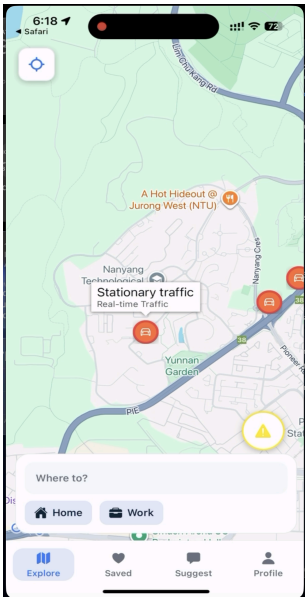
- Email and password input fields
- “Sign In” and “Create Account” buttons
- “Sign in with Google” button
- Password visibility toggle

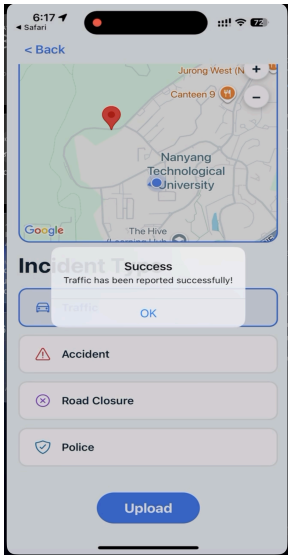






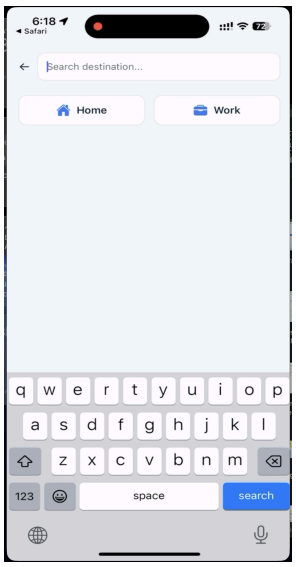


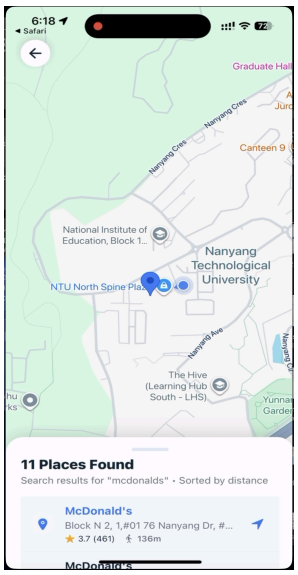


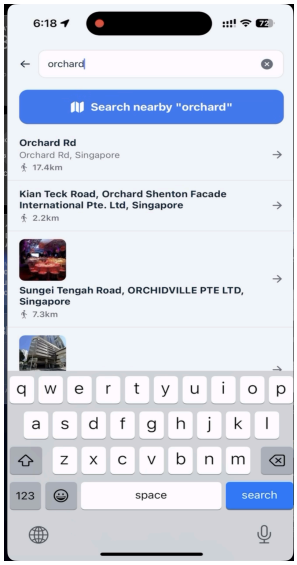


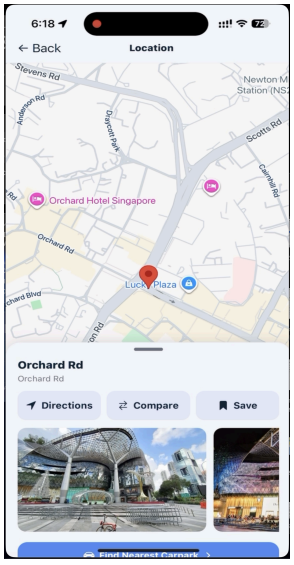
Home Screen

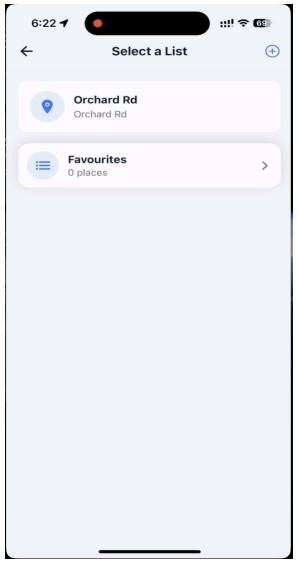
- Top navigation bar with app logo
- Markers on map indicating incidents, yellow border (User), red border (API)
- Reporting Road Incident button
- Quick access to Home and Work.
- Bottom tab navigation





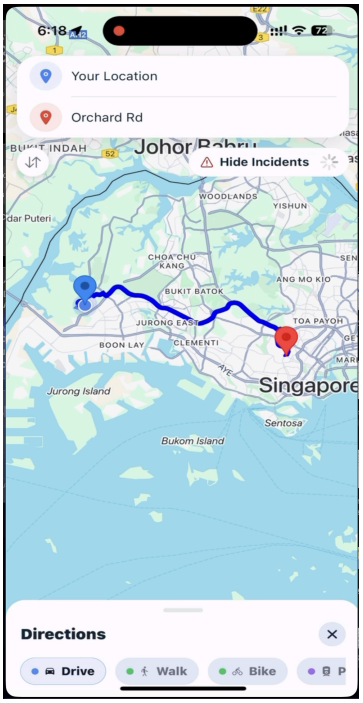


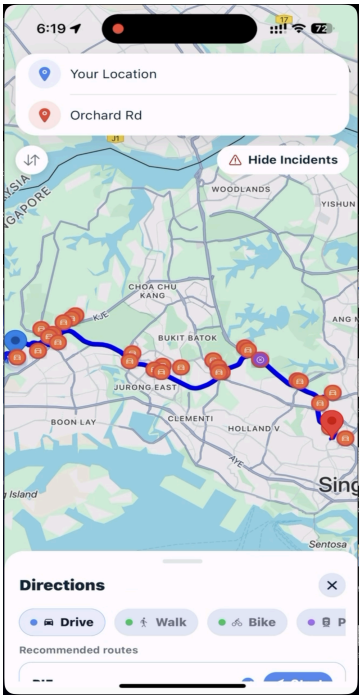


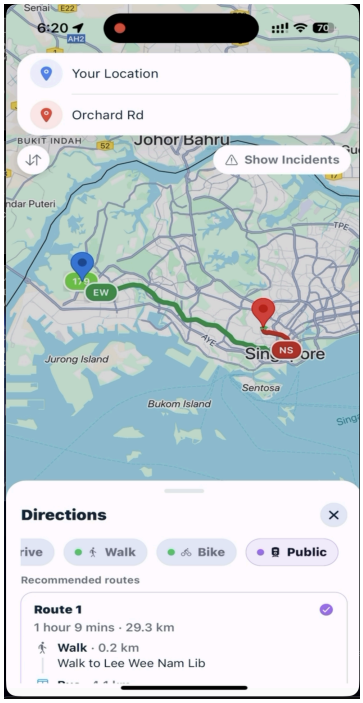


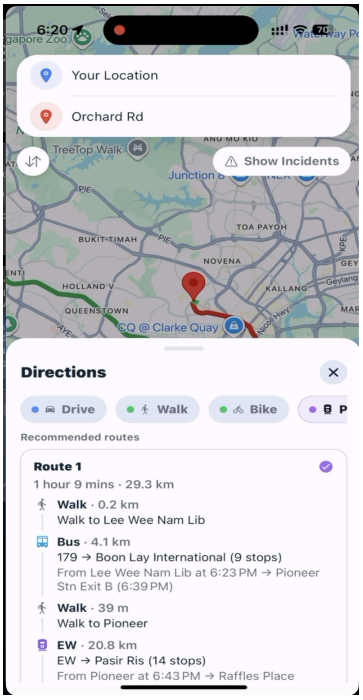
Search and Location Screen

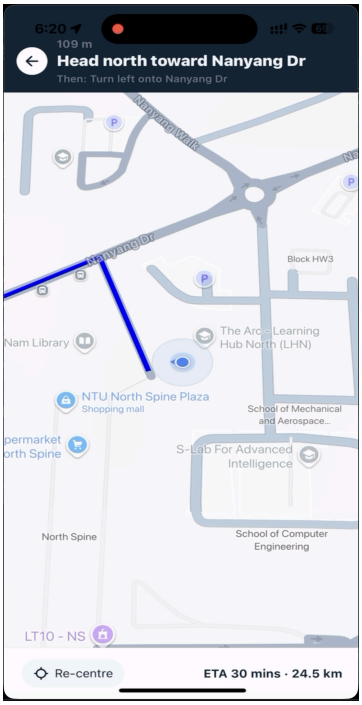
- Home and Work Shortcut buttons
- Keep recent searches
- Autocomplete dropdown for locations
- Photo thumbnails in search results
- “Find Route” button
- Search Based on Nearby Location
- Directions, Save Place, Compare and Nearest Carpark Functionality

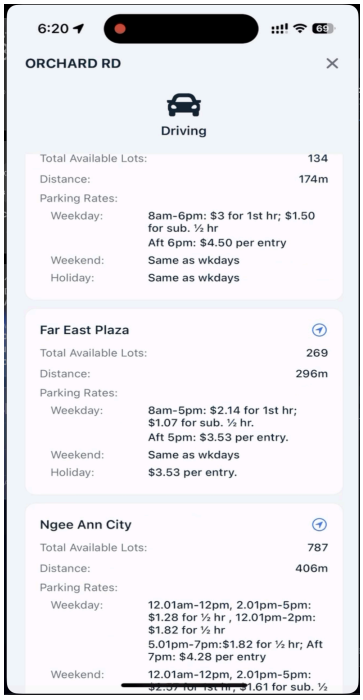

6:18
Your Location
Orchard Rd
Hide Incidents
Singapore
Directions
Drive
Walk
Bike

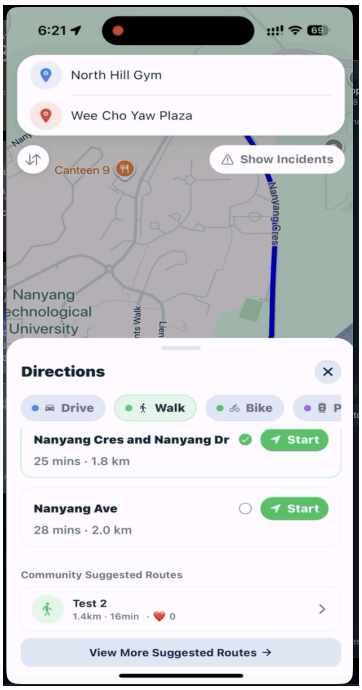

6:19
Your Location
Orchard Rd
Hide Incidents
Directions
Drive
Walk
Bike
Recommended routes

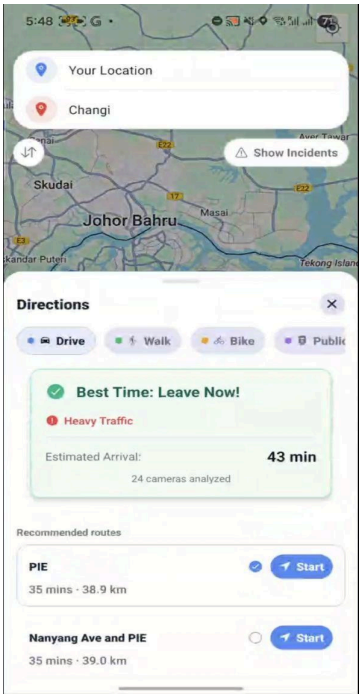

6:20
Your Location
Orchard Rd
Show Incidents
Directions
Walk
Bike
Public
Recommended routes
Route 1
1 hour 9 mins · 29.3 km
Walk · 0.2 km
Walk to Lee Wee Nam Lib

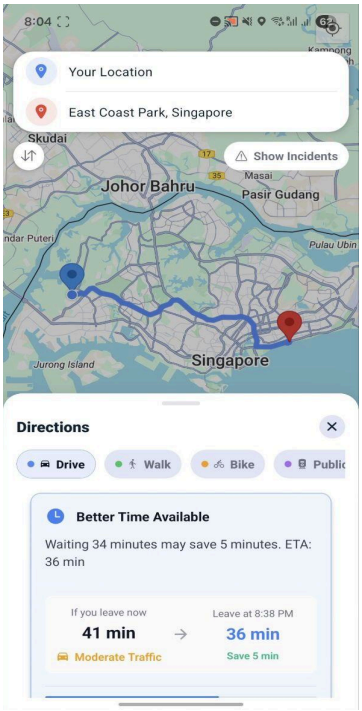

6:20
Your Location
Orchard Rd
Show Incidents
Directions
Drive
Walk
Bike
Recommended routes
Route 1
1 hour 9 mins · 29.3 km
Walk · 0.2 km
Walk to Lee Wee Nam Lib
Bus · 4.1 km
179 → Boon Lay International (9 stops)
From Lee Wee Nam Lib at 6:23 PM → Pioneer Stn Exit B (6:39 PM)
Walk · 39 m
Walk to Pioneer
EW · 20.8 km
EW → Pasir Ris (14 stops)
From Pioneer at 6:43 PM → Raffles Place


6:20
109 m
Head north toward Nanyang Dr
Then: Turn left onto Nanyang Dr
Re-centre
ETA 30 mins · 24.5 km


6:20
ORCHARD RD
Driving
Total Available Lots: 134
Distance: 174m
Parking Rates:
Weekday: 8am-6pm: $3 for 1st hr; $1.50 for sub. ½ hr
Aft 6pm: $4.50 per entry
Weekend: Same as wkdays
Holiday: Same as wkdays
Far East Plaza
Total Available Lots: 269
Distance: 296m
Parking Rates:
Weekday: 8am-5pm: $2.14 for 1st hr; $1.07 for sub. ½ hr.
Aft 5pm: $3.53 per entry.
Weekend: Same as wkdays
Holiday: $3.53 per entry.
Ngee Ann City
Total Available Lots: 787
Distance: 406m
Parking Rates:
Weekday: 12.01am-12pm, 2.01pm-5pm: $1.28 for ½ hr , 12.01pm-2pm: $1.82 for ½ hr
5.01pm-7pm:$1.82 for ½ hr; Aft 7pm: $4.28 per entry
Weekend: 12.01am-12pm, 2.01pm-5pm:

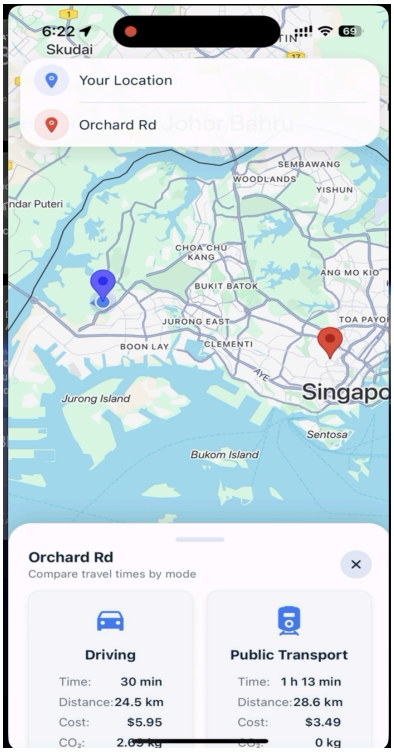


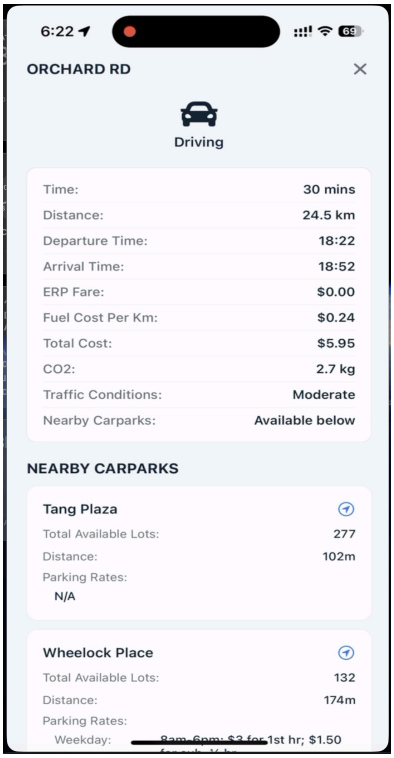


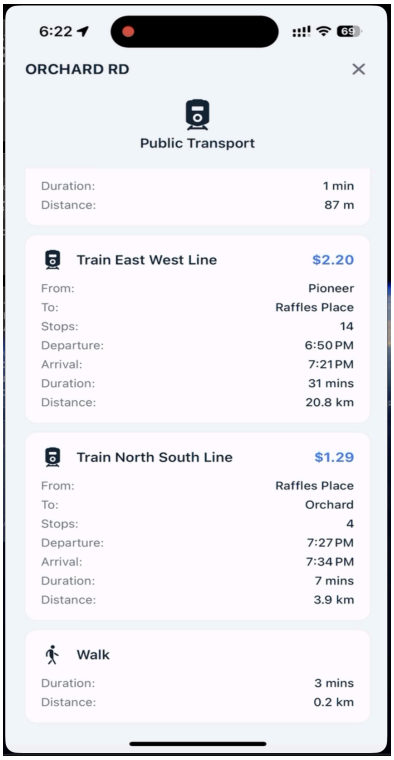


Directions Screen

- Full-screen interactive map view
- Route polylines and traffic incident markers
- Start / End location markers
- Collapsible metrics panel showing travel time, distance, cost, and carbon emissions
- Tabs for mode switching
- Step-by-step directions list
- “Save Route” button
- Forecasting of best time to leave

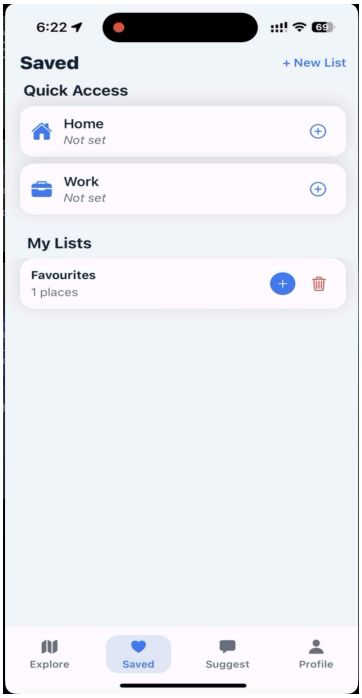


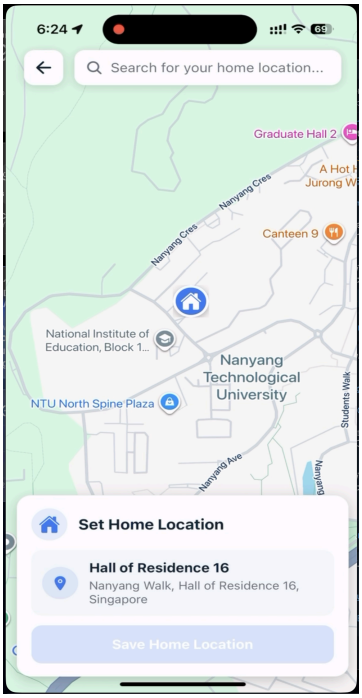




Compare Screen

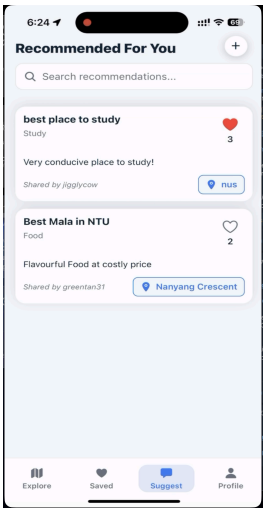
- Display metrics of the different transportation
- Public Transport displays cost breakdowns of MRT/Buses
- Driving gives nearest carparks, cost of ERPs and estimated fuel costs.

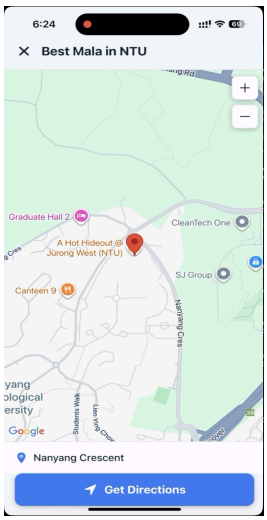


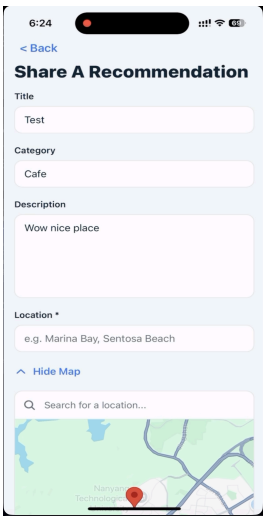


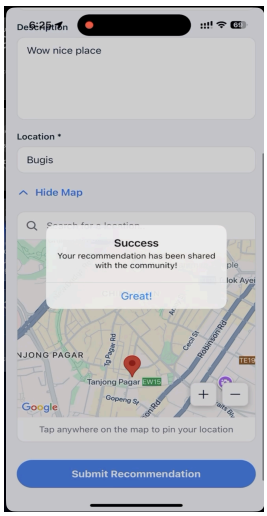
Saved Places Screen

- List of saved lists and places
- “Create New List” button
- Each item shows list name, place count, and thumbnail preview
- Swipe-to-delete actions



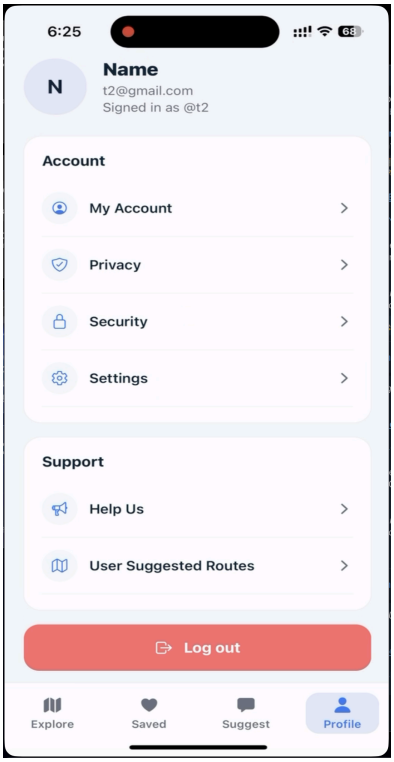


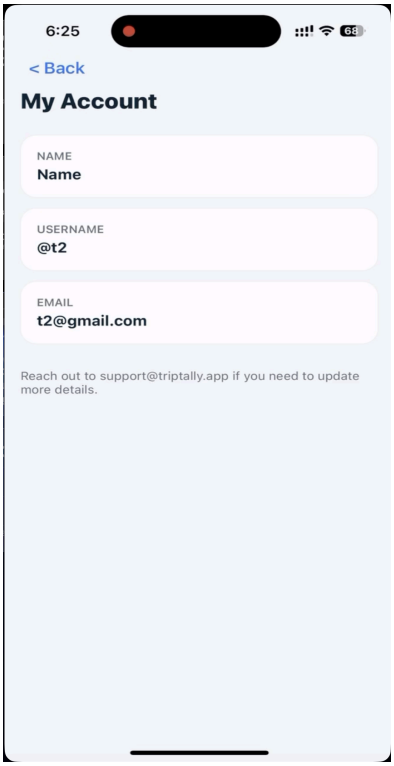


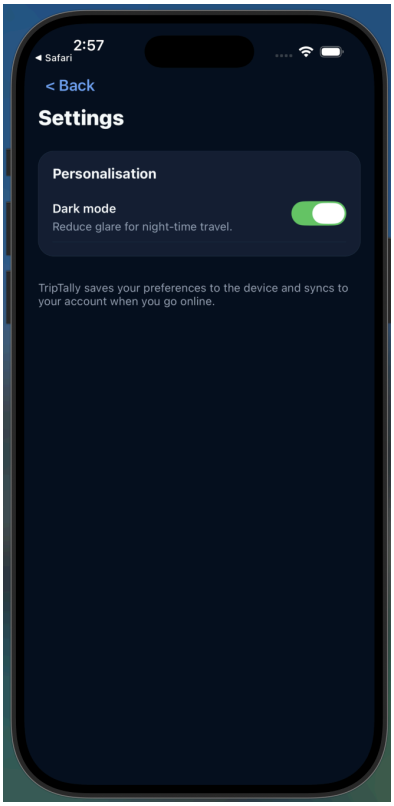


Recommendation Screen

- List of user recommendations
- Filtered by likes
- Add Recommendation Button

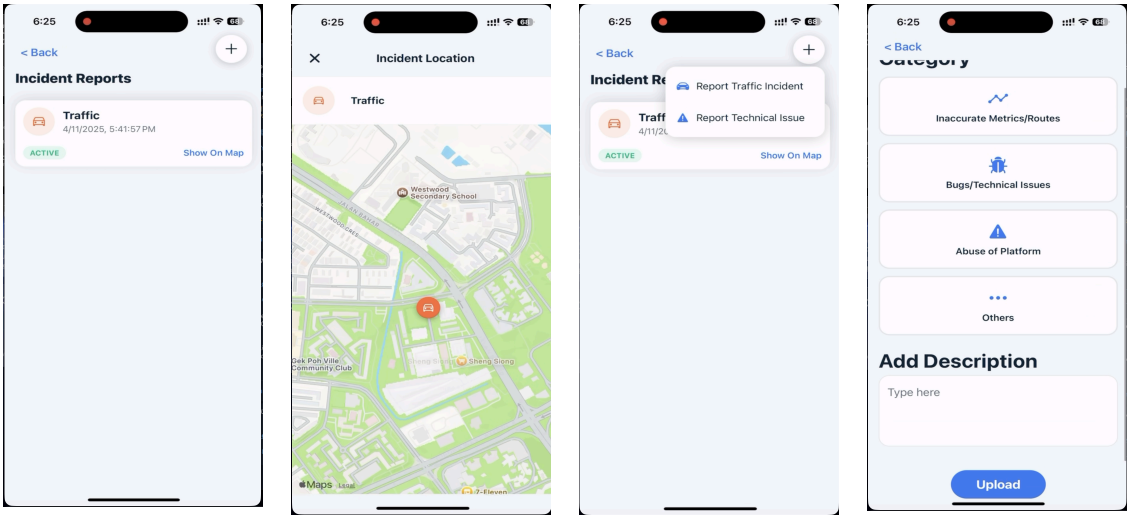






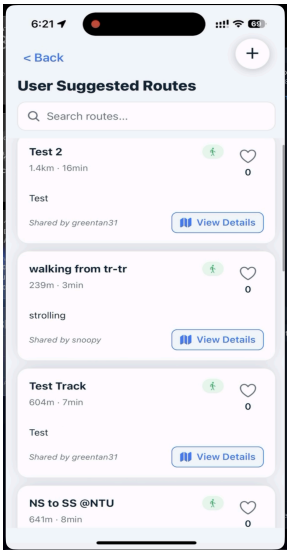
Profile Screen

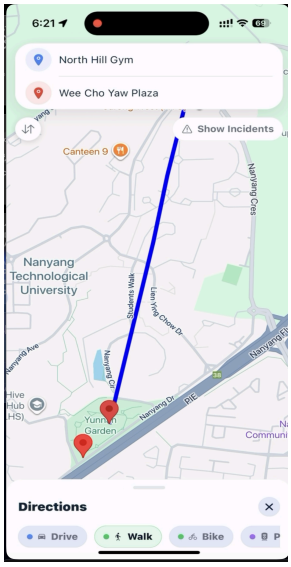
- Display name and email address
- Help Us brings to reported road incidents and technical issues
- Logout button
- Settings allow to change to dark/light mode
- User Suggested Route Button

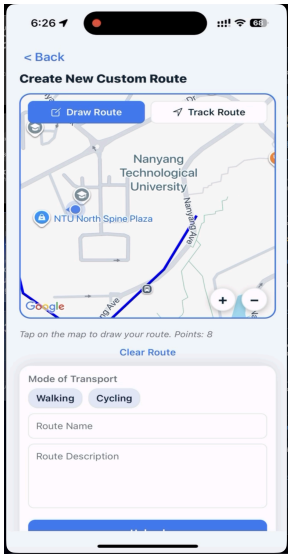


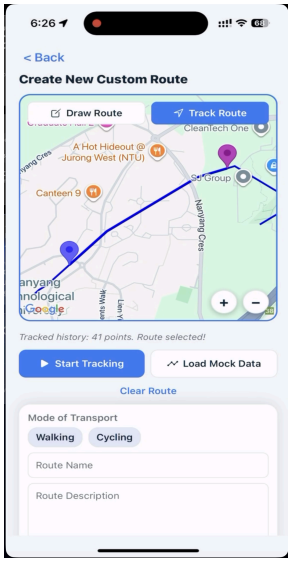
Help Us Screen

- List downs Road Incident Reports by Users
- Add Technical Reports and Road Incident Reports









User Suggested Route Screen

- List of User Suggested Routes
- View Details of the Route
- Filtered by Likes
- Add Route Button
- Able to Draw on the Map by pinning points
- Able to start tracking current location or load up previously tracked location

4.2 Hardware Interfaces

4.2.1 GPS/Location Services

- HW-1: System shall access device GPS for current location.
- HW-2: System shall request location permissions on first use.
- HW-3: System shall gracefully handle disabled location services.
- HW-4: System shall update user location in real time during navigation.

4.2.2 Network Interface

- HW-5: System shall detect network connectivity status.
- HW-6: System shall notify users when offline.
- HW-7: System shall cache essential data for offline access.
- HW-8: System shall retry failed requests upon network restoration.

4.2.3 Storage

- HW-9: System shall store user credentials securely in the device keychain.
- HW-10: System shall cache map tiles to improve performance.
- HW-11: System shall store user preferences locally.
- HW-12: System shall manage storage space efficiently.

4.3 Software Interfaces

4.3.1 Google Maps Platform APIs

ID	API	Purpose	I/O Format	Inputs	Outputs
API-1	Directions API	Calculate routes between locations	JSON via HTTPS (v1)	Origin, destination, mode, alternatives	Polyline, distance, duration, steps

API-2	Geocoding API	Convert addresses to coordinates	JSON via HTTPS (v1)	Address / coordinates	Formatted address, lat/lng, place details
API-3	Places API	Search for locations	JSON via HTTPS (v1)	Search query, location bias	Predictions, details, photos
API-4	Places Photos API	Retrieve location photos	Image via HTTPS (v1)	Photo reference, max size	Image data

4.3.2 TomTom Traffic APIs

ID	API	Purpose	I/O Format	Inputs	Outputs
API-5	Traffic Flow API	Real-time traffic flow data	JSON via HTTPS (v1)	Bounding box coordinates	Speed, congestion levels
API-6	Traffic Incidents API	Real-time traffic incidents	JSON via HTTPS (v4)	Bounding box coordinates	Type, location, description, severity

4.3.3 LTA DataMall APIs

ID	API	Purpose	I/O Format	Inputs	Outputs

API-7	Bus Services API	Bus routes and timings	JSON via HTTPS (v1)	Service no., stop code	Routes, schedules, arrivals
API-8	Train Service Alerts API	MRT alerts and delays	JSON via HTTPS (v1)	None	Active service alerts
API-9	Carpark Availability API	Carpark lot availability	JSON via HTTPS (v1)	None	Carpark locations and available lots

4.3.4 Data.gov APIs

ID	API	Purpose	I/O Format	Inputs	Outputs
API-10	Traffic Images API	Live traffic camera images	JSON via HTTPS	Date and time (optional)	Traffic camera metadata all around Singapore, link to live image at current time

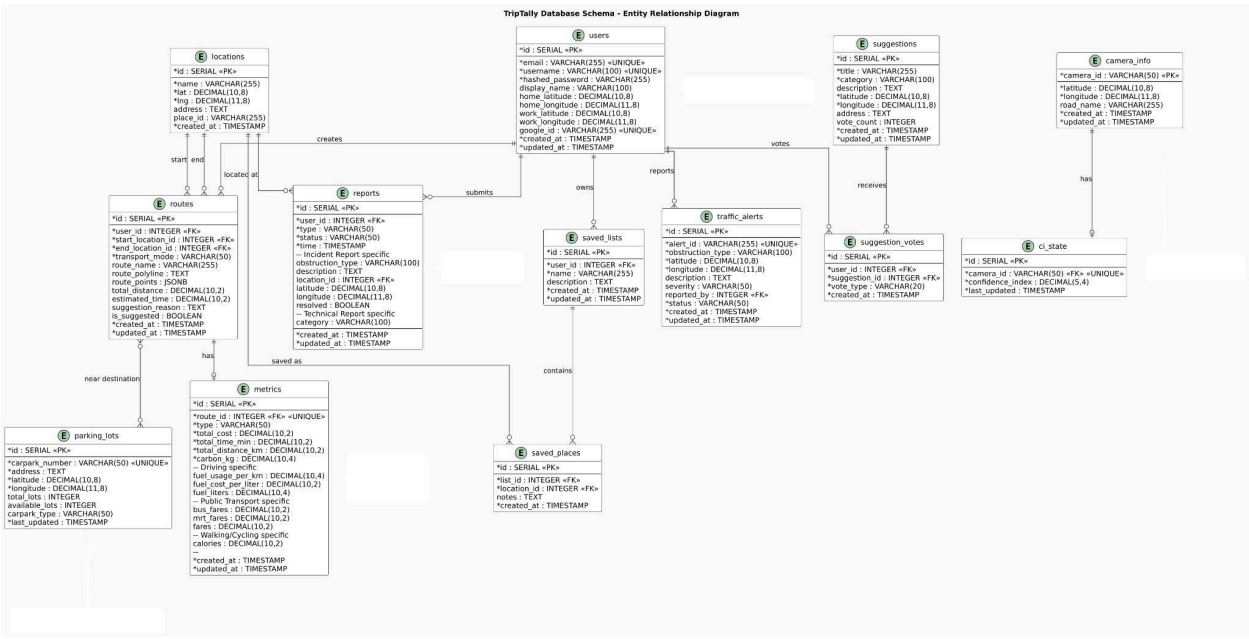
4.3.5 Database Interface

ID	Interface	Purpose	Access Method	Stored Data
DB-1	PostgreSQL Database (14+)	Persistent data storage	SQL via SQLAlchemy ORM	User profiles, saved routes, traffic reports, metrics, camera data

DB-2	Redis Cache Database (7+)	Caching traffic camera data	Redis's Python library or Docker	Traffic camera metadata, current and forecasted congestion index

Our system connects to a PostgreSQL database that stores user, route, traffic, and report data. The database schema follows a relational model to ensure data consistency and efficient querying.

Entity-Relationship Diagram (ERD):
The ERD below illustrates the logical structure of the database and how key entities relate to one another.



4.4 Communications Interfaces

This section describes how TripTally's mobile application communicates with its backend server and with external services such as the Google Maps API. It outlines the protocols, data formats, and authentication mechanisms that ensure secure, reliable, and consistent data exchange across all system components.

4.4.1 HTTP/HTTPS Protocol

TripTally uses standard web protocols for all client–server communication.

- COM-1: All data transmitted between the client and server must use HTTPS with TLS 1.2 or higher to ensure data security.

- COM-2: Every API request includes a valid authentication token in the request headers.
- COM-3: The backend follows RESTful API design principles to maintain clear, predictable, and well-structured endpoints.
- COM-4: The system must handle connection timeouts gracefully, notifying users if requests fail.
- COM-5: A retry mechanism with exponential backoff is implemented to manage temporary network issues.

4.4.2 Data Formats

- COM-6: All request and response bodies shall use JSON format.
- COM-7: Date/time values shall comply with ISO 8601.
- COM-8: Geographic coordinates shall use decimal degrees.
- COM-9: Distance values shall use metric units (km, m).
- COM-10: Currency values shall be expressed in Singapore Dollars (SGD)

4.4.3 Authentication

- COM-11: System shall use JWT tokens for session management
- COM-12: Tokens shall expire after 24 hours.
- COM-13: A token refresh mechanism shall be implemented.
- COM-14: OAuth 2.0 shall be used for Google authentication.

5. Non-Functional Requirements

5.1 Performance Requirements

5.1.1 Response Time

- NFR-1: Route calculation shall complete within 5 seconds under normal network conditions.
- NFR-2: Location search results shall be returned within 2 seconds.
- NFR-3: Map rendering shall initiate within 1 second of request.
- NFR-4: User login shall complete within 3 seconds.
- NFR-5: Traffic incident data shall automatically refresh every 60 seconds.
- NFR-6: Best time to leave forecast shall be returned within 5 seconds.

5.1.2 Throughput

- NFR-7: The system shall support at least 100 concurrent users.
- NFR-8: The backend shall process at least 1,000 API requests per minute.
- NFR-9: The database shall handle 500 queries per second under peak load.

5.1.3 Capacity

- NFR-10: The system shall support at least 10,000 registered users.
- NFR-11: The system shall store up to 100,000 saved routes.
- NFR-12: The system shall process routes up to 500 km in length.
- NFR-13: Each user shall be able to save up to 50 favorite places

5.2 Safety Requirements

- NFR-14: The system shall avoid distracting users while driving.
- NFR-15: The system shall display appropriate warnings for route hazards.
- NFR-16: All user inputs shall be validated to prevent injection or malicious data.
- NFR-17: Rate-limiting shall be implemented to prevent service abuse.
- NFR-18: All security-related events shall be logged for audit and analysis.

5.3 Security Requirements

5.3.1 Authentication and Authorization

- NFR-19: All passwords shall be hashed using bcrypt with unique salts.
- NFR-20: Strong password policies shall be enforced (minimum 8 characters, mixed case).
- NFR-21: Authentication shall be based on JWT tokens.
- NFR-22: All authentication tokens shall be validated on each request.
- NFR-23: Unauthorized access to user data shall be strictly prevented.

5.3.2 Data Protection

- NFR-24: Sensitive data shall be encrypted at rest.
- NFR-25: All data transmissions shall use HTTPS (TLS 1.2+).
- NFR-26: API keys shall not be stored or exposed in client-side code.
- NFR-27: Secure session management shall be implemented.
- NFR-28: The system shall comply with the Personal Data Protection Act (PDPA) of Singapore.

5.3.3 Privacy

- NFR-29: User location data shall not be shared with third parties without consent.
- NFR-30: Users shall be able to delete their accounts and associated data.
- NFR-31: Analytics data shall be anonymized before storage or use.
- NFR-32: The system shall provide a visible Privacy Policy and Terms of Service.

5.4 Software Quality Attributes

5.4.1 Reliability

- NFR-33: The system shall maintain at least 99.9% uptime during business hours.
- NFR-34: API failures shall be handled gracefully with fallback responses.
- NFR-35: Automatic error recovery shall be implemented for transient faults.
- NFR-36: Database backups shall be performed daily.

5.4.2 Availability

- NFR-37: The system shall be available 24/7, excluding scheduled maintenance.
- NFR-38: Maintenance windows shall be scheduled during low-usage periods.
- NFR-39: Users shall be notified in advance of any planned downtime.

5.4.3 Maintainability

- NFR-40: All Python code should follow the PEP 8 style guidelines to maintain readability and consistency.
- NFR-41: JavaScript code should comply with ESLint standards to ensure code quality and prevent common errors.
- NFR-42: The system should include complete and up-to-date API documentation for developers.
- NFR-43: Unit tests should achieve at least 80% code coverage to ensure reliability of core features.
- NFR-44: All source code should be maintained using Git version control which allows for collaboration and change tracking.

- NFR-45: The development process should include Continuous Integration and Continuous Deployment (CI/CD) pipelines to automate testing and deployment.

5.4.4 Portability

- NFR-46: The mobile app shall run on iOS 13.0+ devices.
- NFR-47: The mobile app shall run on Android 8.0+ devices.
- NFR-48: The backend shall be deployable on any Linux server.
- NFR-49: The system shall use Docker for containerization and portability.

5.4.5 Usability

- NFR-50: New users should be able to plan a route within two minutes of their first use.
- NFR-51: The interface should be simple and intuitive enough to use without a user manual.

- NFR-52: Error messages should clearly explain the issue and suggest what the user can do next.
- NFR-53: The system should operate in English.
- NFR-54: The app should comply with WCAG 2.1 Level A accessibility standards to ensure inclusivity.

5.4.6 Scalability

- NFR-55: The system's architecture should allow for horizontal scaling to handle growing demand.
- NFR-56: The database should be able to use read replicas to improve performance.
- NFR-57: Frequently accessed data should be cached to reduce response time.
- NFR-58: Load balancing should be applied across backend servers to distribute traffic evenly.

5.4.7 Testability

- NFR-59: All business logic components shall be unit-testable.
- NFR-60: The system shall support integration testing.
- NFR-61: The mobile interface shall support automated UI testing.
- NFR-62: The system shall automatically generate test coverage reports.

6. Appendix

6.1 Glossary

This section defines key technical terms used throughout the TripTally SRS document.

Term	Definition
API	Application Programming Interface used for system communication between software components.
BCrypt	A secure password hashing algorithm used for storing credentials safely.

CI	Refers to Congestion Index, a measurement of how congested a road is at a given time.
CI/CD	Continuous Integration / Continuous Deployment - automated processes for testing and deploying code.
ETA	Estimated Time of Arrival. Used to estimate the time it will take to go from one point to another.
FastAPI	A high-performance Python web framework used to build TripTally's backend API.
GPS	Global Positioning System for determining geographic location.
HTTPS	Hypertext Transfer Protocol Secure - ensures encrypted communication over the internet.
JWT	JSON Web Token used for securely transmitting authentication information.
LTA	Land Transport Authority of Singapore, provider of DataMall APIs.
MRT	Mass Rapid Transit, Singapore's urban rail transit system.
OAuth	Open standard for secure authorization and third-party authentication.
ORM	Object-Relational Mapping - technique for managing database data using object-oriented models.

PDPA	Personal Data Protection Act - Singapore’s data privacy regulation.
Polyline	Encoded series of geographical coordinates representing a route path.
React Native	Cross-platform mobile development framework for Android and iOS.
REST	Representational State Transfer - architectural style for designing networked APIs.
SQLAlchemy	Python SQL toolkit and ORM used for database interactions.
TLS	Transport Layer Security - cryptographic protocol ensuring secure communications.
WCAG	Web Content Accessibility Guidelines - international standards for digital accessibility.

6.2 Acronyms

Acronym	Expansion
API	Application Programming Interface
CI	Congestion Index

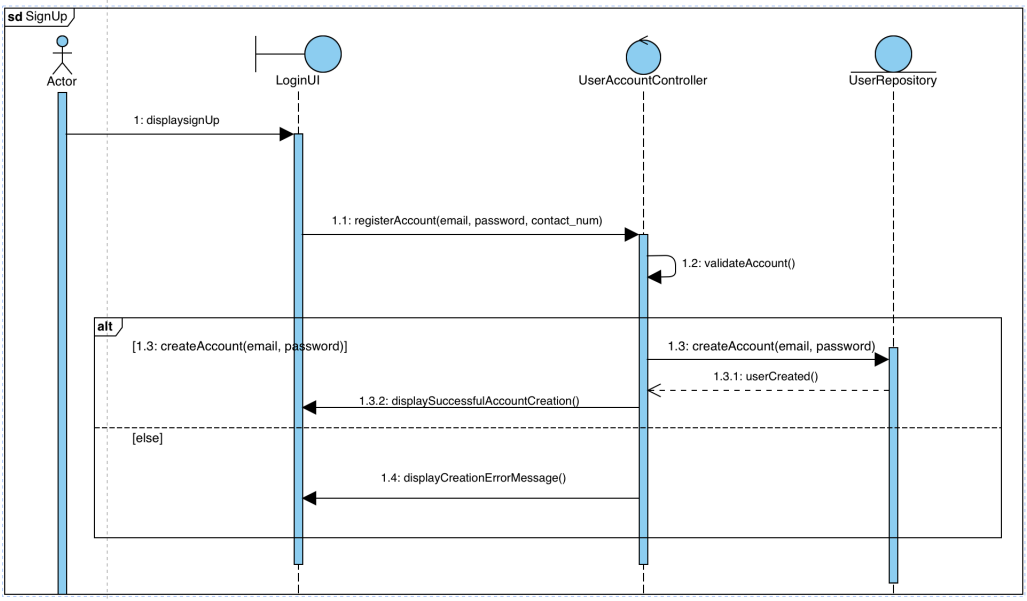
DB	Database
FR	Functional Requirement
GPS	Global Positioning System
HTTP	Hypertext Transfer Protocol
JWT	JSON Web Token
LTA	Land Transport Authority
MVC	Model–View–Controller
NFR	Non-Functional Requirement
ORM	Object–Relational Mapping
REST	Representational State Transfer
SRS	Software Requirements Specification
TTL	Time To Live

UI	User Interface
UX	User Experience

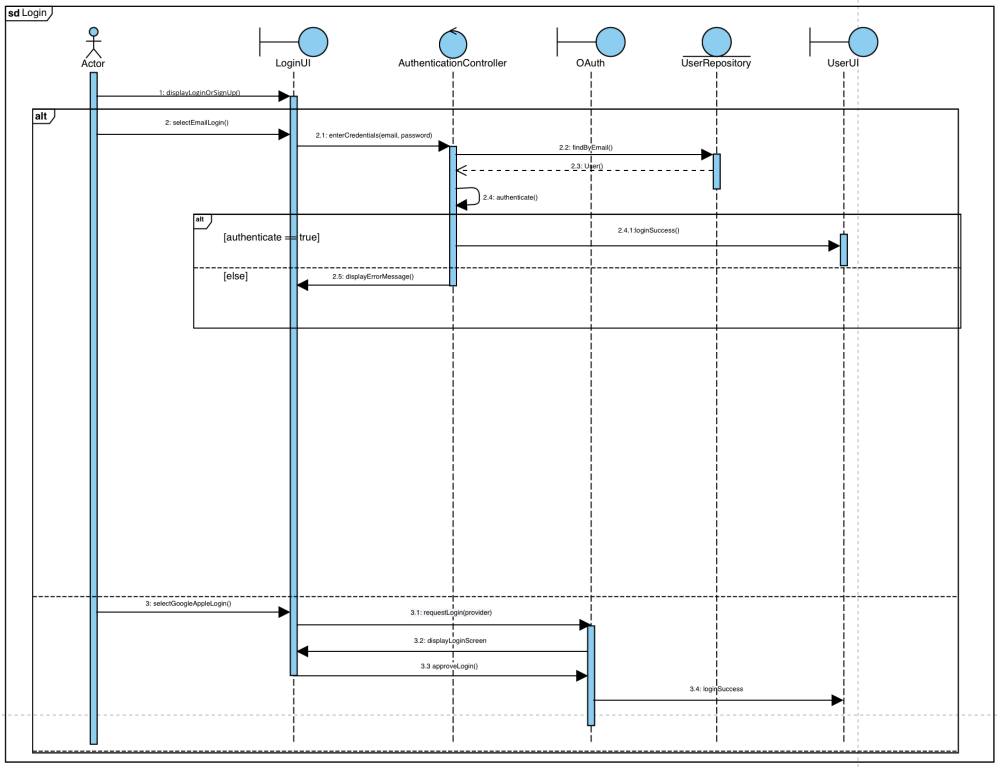
6.3 Analysis Models

6.3.1 Sequence Diagrams

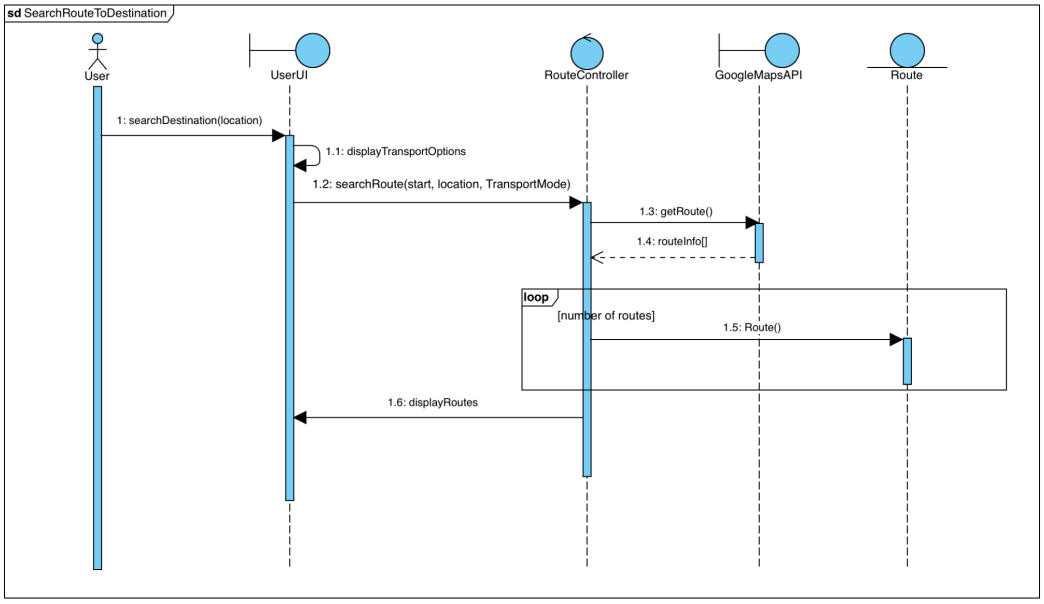
6.3.1.1 SignUp



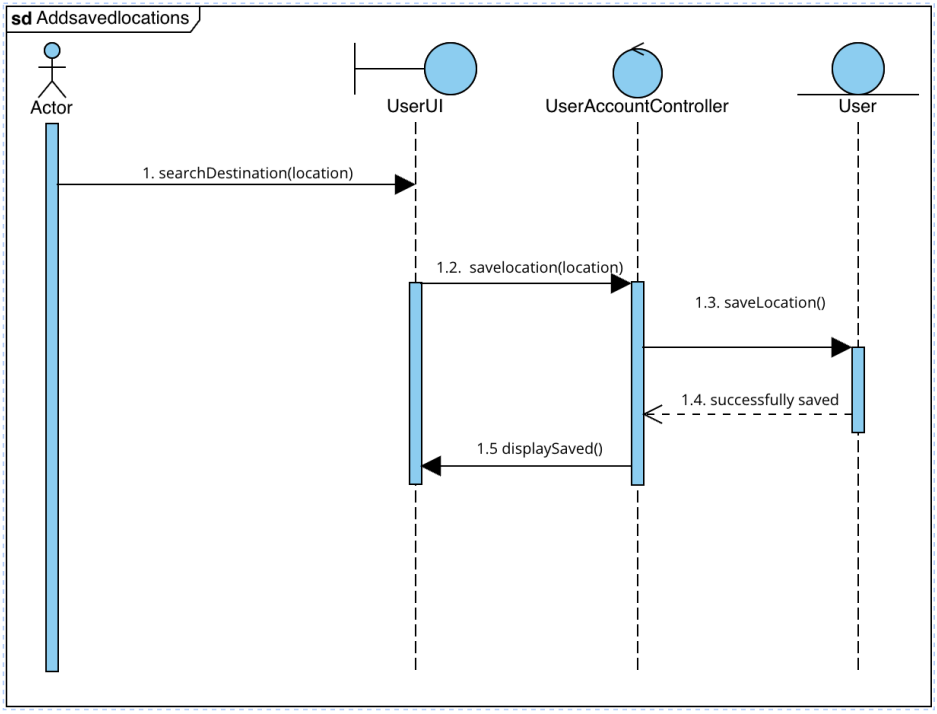
6.3.1.2 Login



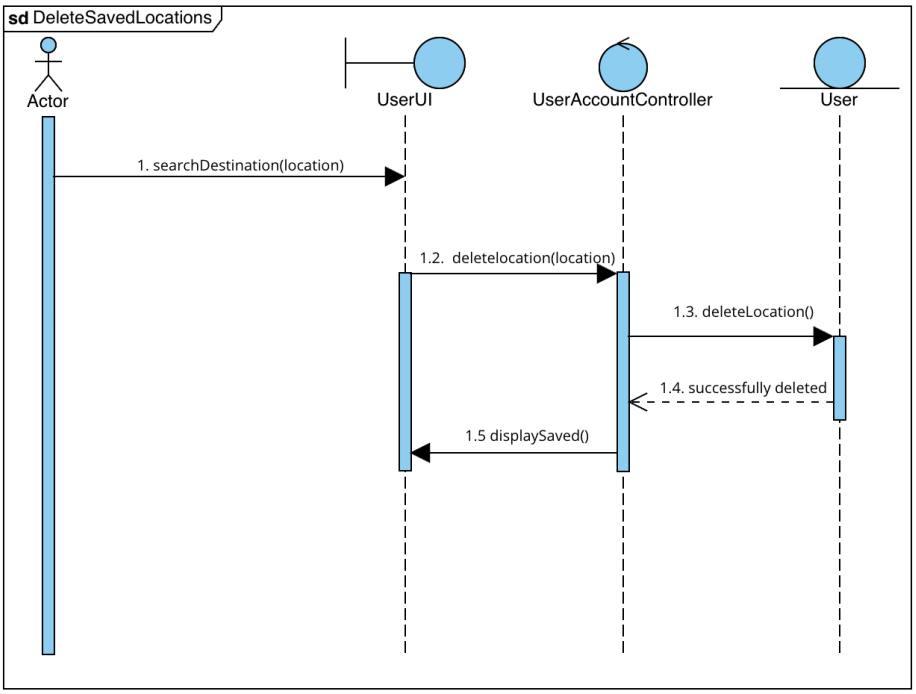
6.3.1.3 SearchRouteToDestination



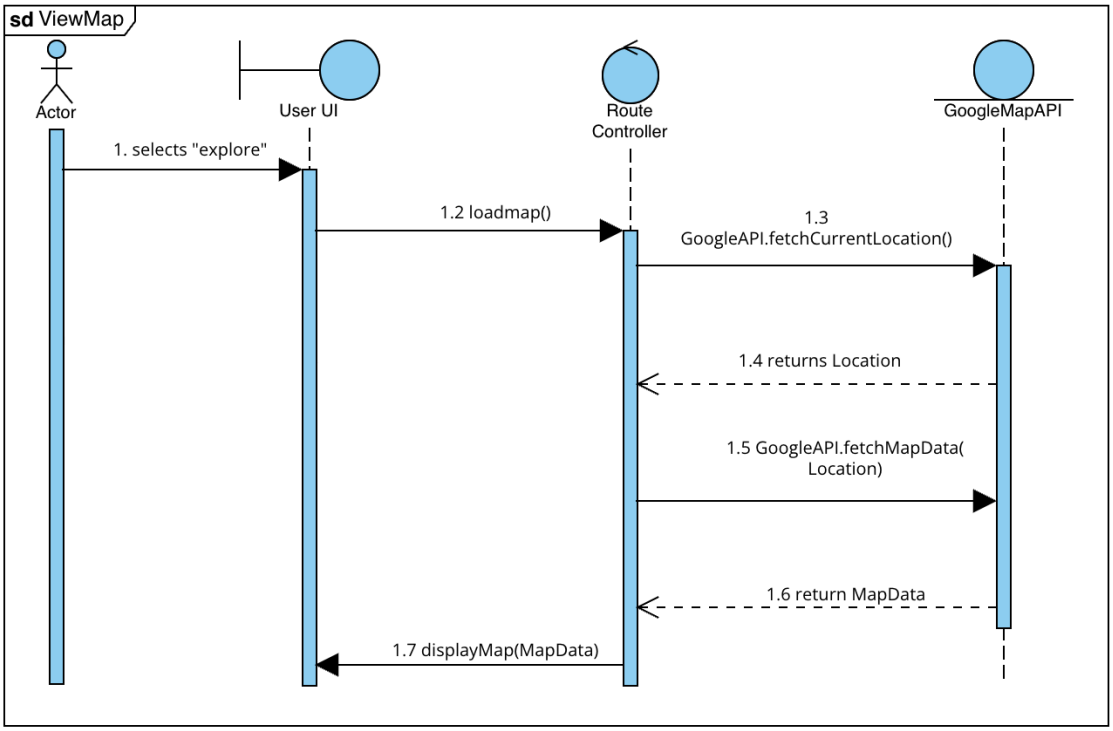
6.3.1.4 ManageSavedPlaces



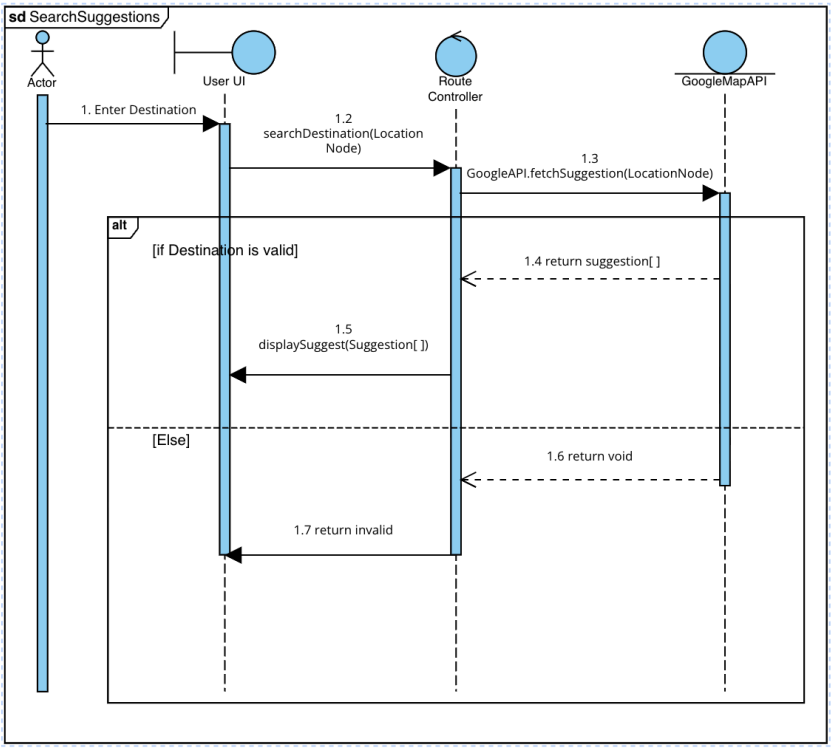
6.3.1.5 DeleteSavedLocations



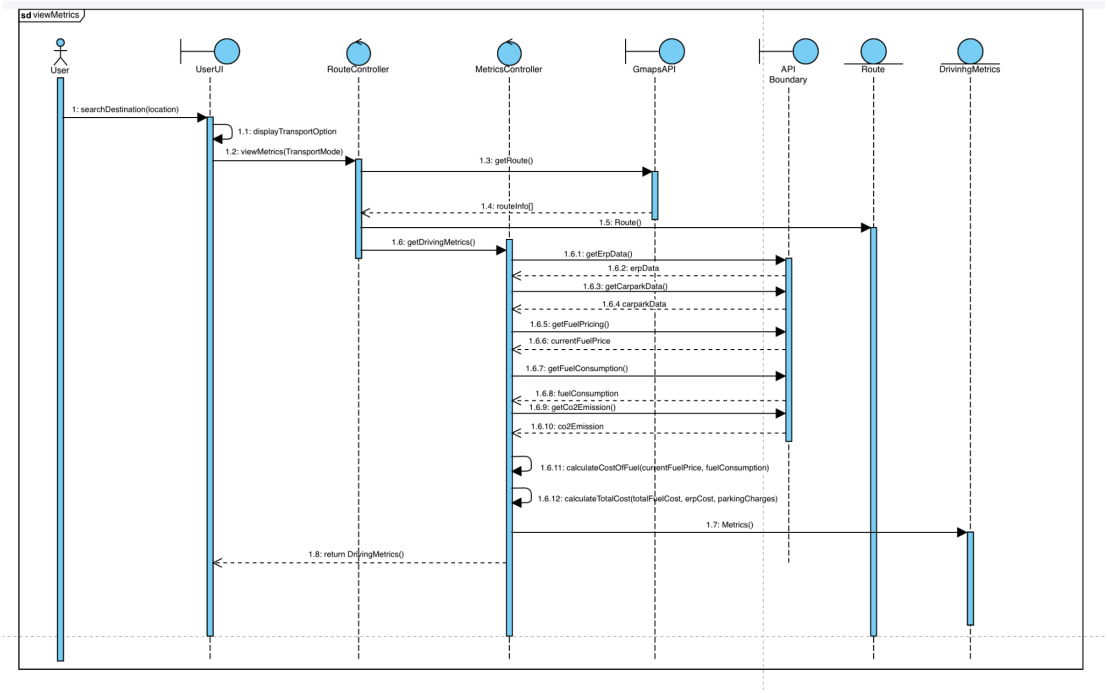
6.3.1.6 View map



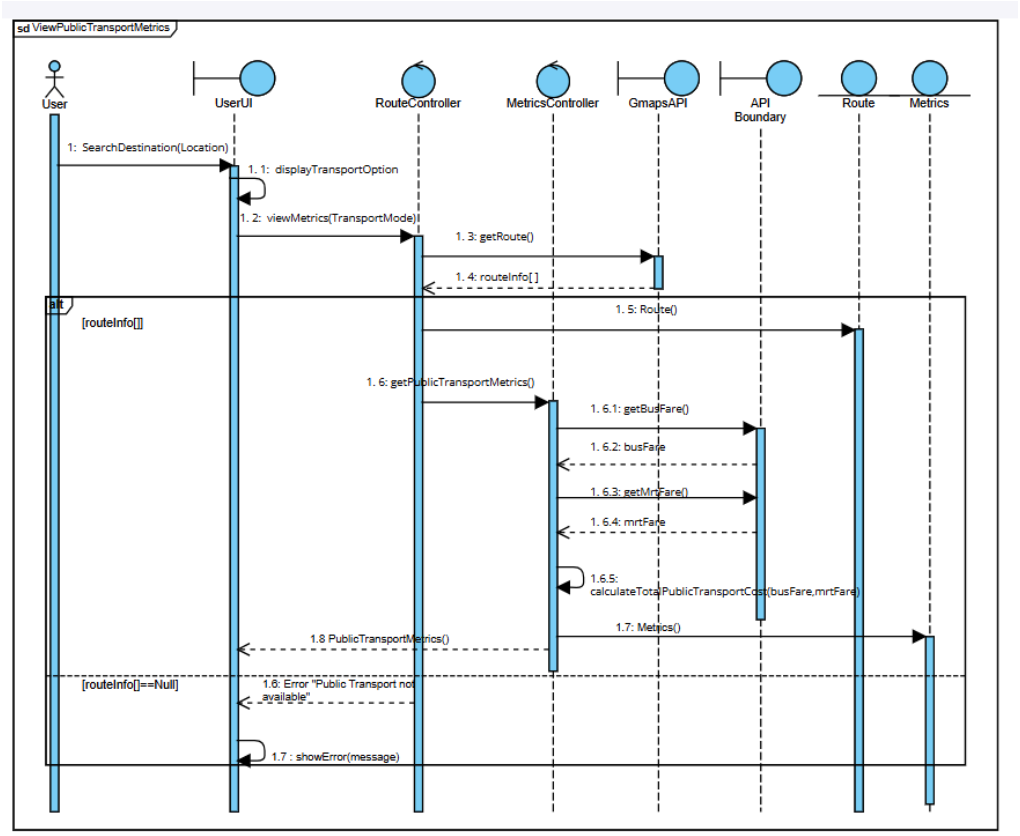
6.3.1.7 SearchSuggestion



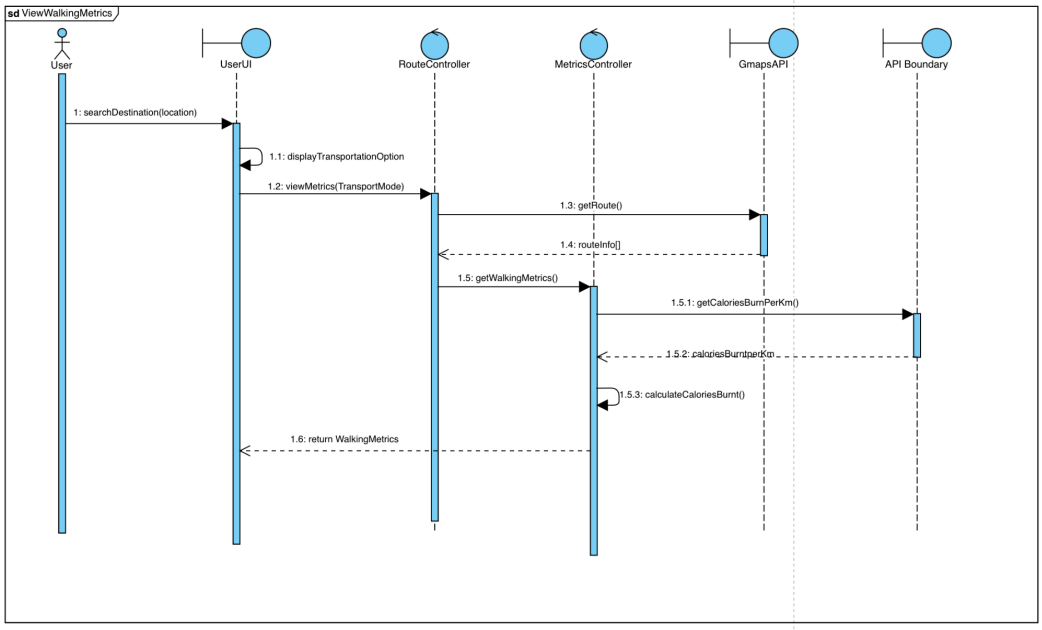
6.3.1.8 ViewDrivingMetrics



6.3.1.9 ViewPublicTransportMetrics



6.3.1.10 ViewWalkingMetrics



6.3.2 Dialog Diagram