

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

SC2006- SOFTWARE ENGINEERING

Lab 1 deliverables

Lab Group- SCSA

Team- Apollo 11

Members:

Agarwal Radhika	U2323183J
Ang Ming Yang	U2322946G
Gunda Sai Venkata Aaditya	U2320456C
Mohamed Yaseen Aboobucker	U2323243C

Siddhique	
Titus Lua Jie Qing	U2222033A

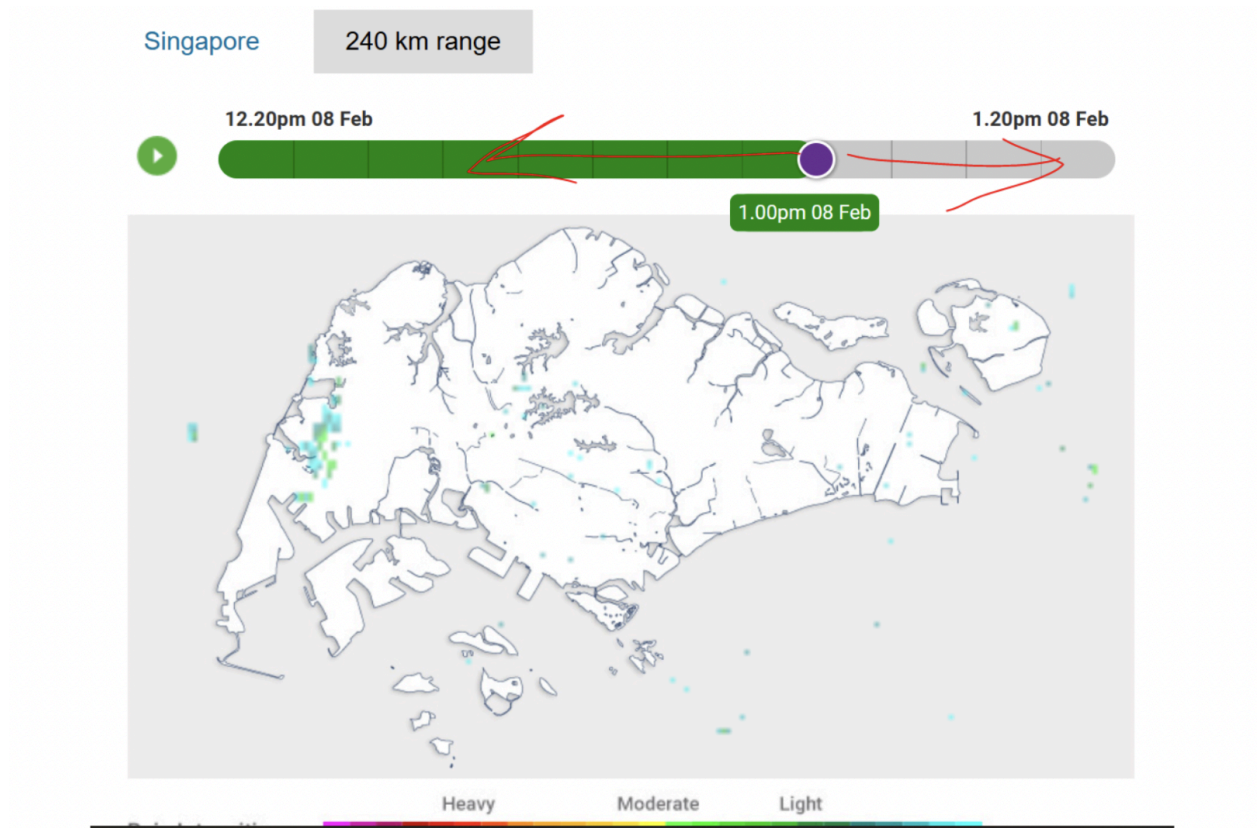
1. Introduction
 - 1.1 Project Overview
 - 1.2 Project Mission Statement
 - 1.3 Target Users & Stakeholders
2. Functional and Non-Functional Requirements
 - 2.1 Functional Requirements
 - 2.2 Non-Functional Requirements
3. Data Dictionary
 - 3.1 Key Terms and Definitions
 - 3.2 Train Data Attributes and Computation Methods
4. Use Case Model
 - 4.1 Use Case Diagram
 - 4.2 Use Case Descriptions
5. UI Mockups
 - 5.1 Overview of UI Design
 - 5.2 Screens and Features
6. Data Processing & System Logic
 - 6.1 MRT Train Load Calculation
 - 6.2 Real-Time Data Integration (LTA DataMall API)
 - 6.3 Historical Data Storage and Analysis
 - 6.4 Alternative Route Computation
 - 6.5 Predictive Congestion Alerts
 - 6.6 System Workflow Summary
7. Testing & Validation
 - 7.1 Test Cases for Functional Requirements
 - 7.2 Performance Testing (Response Time & Data Accuracy)
 - 7.3 Edge Cases & Fail-Safes
- 8 Classes
 - 8.1 Class diagram
 - 8.2 Sequence Diagrams

1.Introduction

1.1 Project Overview

The Train Overcrowding Analysis and Mitigation system is a web-based application designed to help commuters plan their MRT travel by providing real-time and historical train crowdedness data. The application integrates passenger volume data from the Land Transport Authority (LTA) DataMall API, allowing users to make informed decisions about their travel routes based on congestion levels.

This system uses historical trends, live data, and predictive analytics to reduce peak-hour congestion and recommend alternative routes or travel timings. It also serves as a tool for transport authorities like LTA and SMRT to optimise train scheduling and capacity distribution.



1.2 Project Mission Statement

The Apollo 11 team is developing a web/app platform that enables daily commuters to check train crowdedness levels on MRT lines—starting with the East-West Line (EWL). This tool will help passengers avoid highly congested trains by:

- Providing expected train congestion data.
- Suggest alternative routes and the latest time to leave (LTL) travel times.
- Helping LTA and SMRT meet commuters' needs at the lowest possible cost.

This project aligns with Singapore's Smart Mobility Initiative, supporting efforts to enhance public transport efficiency.

1.3 Target Users & Stakeholders

User Type	Needs & Benefits
Daily Commuters	Reduce time wasted on travel, and more time spent on work and personal activities.
Office Workers	Accurate efficient travel routes to maximise time on the job increasing production output.
Students	Identify the latest time to leave (LTL) for school without being late to get the maximum amount of sleep.
LTA & SMRT	Data insights are required to improve scheduling and modify train frequency for optimal cost-effectiveness.

This application primarily serves commuters and helps LTA and SMRT optimise train operations, leading to better public transport management.

2. Functional and Non-Functional Requirements

2.1 Functional Requirements

The system is designed to help MRT commuters plan their travel by providing real-time and historical train congestion data. The key functional requirements are as follows:

1. User journey and features

- Users should be able to input their starting and ending MRT stations.
- The system should display real-time MRT train capacity percentages at each station along the selected route.
- Users should be able to view historical train congestion data to compare peak and non-peak hours.
- The system should suggest alternative travel times and routes to reduce congestion.
- The system should indicate stations with high fluctuations in commuters or stations with low occupancy, where seating availability is likely.
- Users should be able to set alerts for high congestion levels, notifying them when a train or station is overcrowded.

2. Data processing and updates

- The system should fetch real-time train occupancy data from the LTA DataMall API and update it regularly.
- The congestion data should be updated every 24 hours for accuracy.
- The system should store historical data for at least two months for trend analysis.

- The system should distinguish between travel directions, such as Tuas to Pasir Ris versus Pasir Ris to Tuas, to provide more precise crowd estimates.
- The system should compute average congestion levels per station per hour and display them visually using graphs, heatmaps, or colour-coded indicators.
- The system should convert station names and common nicknames to station codes for processing.

3. Overcrowding prediction and alternative route suggestions

- The system should analyse historical patterns to predict future congestion trends.
- The system should recommend arrival times for users to avoid peak-hour crowding.
- The system should dynamically distribute users into different time slots based on probabilistic allocation to maintain optimal train loads.
- Users should be able to filter alternative route suggestions based on time flexibility and travel preferences.

2.2 Non-Functional Requirements

1. Performance and responsiveness

- The system should respond to user queries within 2 seconds.
- The data retrieval process should be optimised to prevent UI delays, ensuring a smooth user experience.

2. Scalability and reliability

- The system should be designed to allow the integration of new MRT lines in the future without significant rework.

- The application should maintain 99.9% uptime, ensuring availability at all times.
- In case of a system failure, it should recover within 5 minutes.

3. UI and UX considerations

- The user interface should be intuitive and minimal, allowing users to access congestion data with as few clicks as possible.
- The application should be mobile-friendly, ensuring a seamless experience on both smartphones and desktops.
- The system should support multiple languages, with an essential Google Translate integration to cater to diverse users.
- To improve clarity, congestion levels should be displayed using visual elements such as heat maps, graphs, or colour-coded indicators.

Examples of Non-Functional Requirements

Usability	Help messages must be displayed in the local language according to the user's locale.
Reliability	After a system reboot, the full system functionality must be restored within 5 minutes.
Performance	When a book is placed in the checkout pad, the system must detect it within 2 seconds.
Supportability	The database must be replaceable with any commercial product supporting standard SQL queries.

3. Data Dictionary

The data dictionary defines key terms and attributes used in the system, ensuring consistency in data processing and interpretation. It also describes relationships between different data entities. Based on the system's functional requirements and the data structure outlined in the project documentation, the following terms have been identified.

3.1 Key Terms and Definitions

Term	Definition
EWL	East-West Line is one of the MRT lines in Singapore. Stations are labelled from EW1 (Pasir Ris) to EW33 (Tuas Link).
CG	Alternate train line from Tanah Merah (CG) to Expo (CG1) and Changi Airport (CG2).
LTA	The Land Transport Authority is the governing body responsible for public transport in Singapore.
SMRT	Singapore Mass Rapid Transit is the operator responsible for managing train services.
start station, end station	The station where the user's journey begins and where it ends.
MRT train capacity %	The percentage of a train's total capacity occupied at a specific time between any two stations (e.g., EW5 → EW6).
Train load data	The number of people in an average train during a specific hour is derived from real-time and historical occupancy

	data.
target load	The ideal maximum number of passengers a train should carry to maintain an optimal commuting experience.
uptime	The duration for which the application is fully functional and accessible.
directional congestion	The number of passengers travelling in each direction (e.g., Tuas to Pasir Ris vs. Pasir Ris to Tuas).
historical congestion trends	Past data on train occupancy levels help predict peak hours and congestion patterns.
probabilistic allocation	A system method that dynamically suggests alternative travel times for users based on statistical models to reduce peak congestion.
alternative route suggestion	The system recommends that users rent MRT lines, transfer at less crowded stations, or travel at non-peak times.
LTL	Latest time to leave
Congestion Threshold	Notification range for congestion if actual above set value, range 0 to 100 , in percent, default: 101

3.2 Train Data Attributes and Computation Methods

The following attributes are used to compute train congestion levels and suggest optimal travel times:

1. Train occupancy data

- Captured from LTA DataMall API in real-time.

- Represents the number of passengers currently on a train between two stations.
- Check for updates every 24 hours for accuracy.

2. Historical congestion trends

- Data is stored for at least two months.
- Used to predict future congestion and suggest optimal travel times.
- Displayed using graphs and heatmaps.

3. Directional congestion

- Data is segmented based on the direction of travel (e.g., Tuas to Pasir Ris vs. Pasir Ris to Tuas).
- Allows more precise alternative route suggestions.

4. Capacity-based rerouting

- Uses train occupancy and congestion trends to recommend travel time shifts for users.
- Probabilistic allocation ensures that commuters are evenly distributed to avoid overloading specific trains.

5. Alternative route computation

- Evaluate congestion levels across different MRT lines.
- Suggests changes in travel routes based on available capacity and historical trends.

4. Use Case Model

4.1 Use Case Diagram

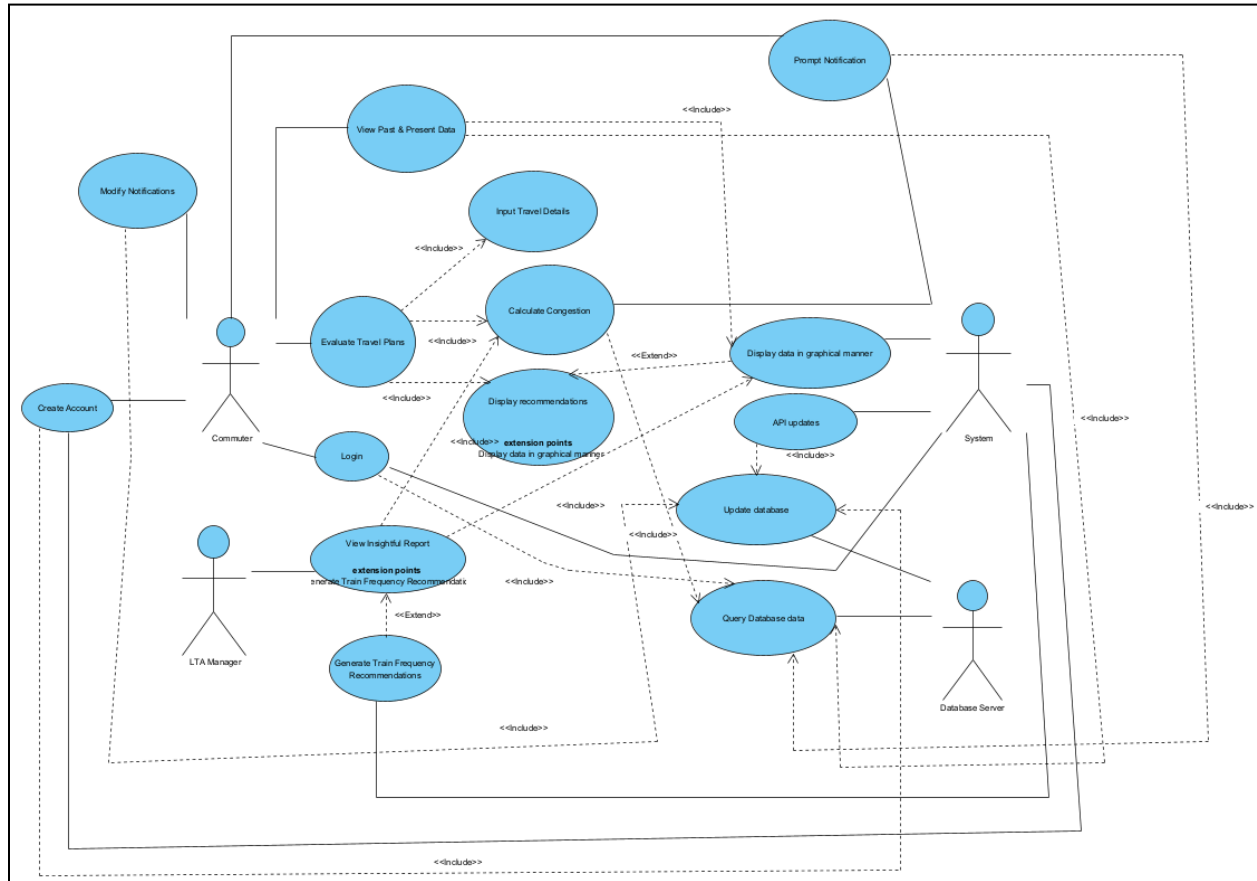


Figure 4.x Use case model

Source: Ang Ming Yang

4.2 Use Case Descriptions

- **User Interactions with the System**
- **System Responses to Input**

UC-001: Check Real-Time Congestion

Field	Details
Use Case ID	UC-001
Use Case Name	Check Real-Time Congestion
Created By	TITUS LUA JIE QING
Last Updated By	TITUS LUA JIE QING
Date Created	25th January 2025
Date Last Updated	10th February 2025
Actor	Commuter
Description	The user inputs MRT stations to check real-time congestion levels using LTA DataMall API.
Flow of Events	<ol style="list-style-type: none">1. Users enter the start and end MRT stations.2. The system retrieves real time data from the LTA API.3. The system displays a color-coded heatmap.4. The user decides whether to continue or select alternative routes.

Alternativ Flows	<ul style="list-style-type: none"> - If the user enters an invalid station, the system shows an error message and prompts for re-entry. - If real-time data is unavailable, the system offers historical congestion trends.
Exceptions	<ul style="list-style-type: none"> - If the API fails, the system displays cached past congestion data. - If no data exists for a station, an error message appears: "No congestion data available for this station."
Includes	Alternative route suggestions
Special Requirements	API should return data within 2 seconds.
Assumptions	The user knows the MRT station names.
Notes & Issues	Improve UI by adding a predictive search dropdown for station selection.

UC-002: View Congestion Data

Field	Details
Use Case ID	UC-002
Use Case Name	View Historical Congestion Data
Created By	MOHAMED YASEEN ABOOBUCKER SIDDHIQUE
Last Updated By	MOHAMED YASEEN ABOOBUCKER SIDDHIQUE
Date Created	25th January 2025

Date Last Updated	10th February 2025
Actor	Commuter
Description	The user selects a past date range to analyse congestion trends.
Flow of Events	<ol style="list-style-type: none"> 1. The user selects a past date range. 2. System fetches API congestion data. 3. System displays trends using graphs and heatmaps.
Alternative Flows	<ul style="list-style-type: none"> - If no API data exists for a date, the system checks the Historical data and display it. -if no Historical data, show: "No congestion data available."
Exceptions	<ul style="list-style-type: none"> - If database retrieval fails, the system prompts retry. - If the data input is incorrect, the system will display "Invalid date format."
Includes	Comparison of congestion across different MRT lines.
Special Requirements	Data must be stored for at least 2 months.
Assumptions	The user understands how to interpret congestion graphs.
Notes & Issues	Add a comparison between weekdays and weekends.

UC-003: Receive Congestion Notifications

Field	Details
Use Case ID	UC-003
Use Case Name	Receive Congestion Notifications
Created By	GUNDA SAI VENKATA
Last Updated By	GUNDA SAI VENKATA
Date Created	25th January 2025
Date Last Updated	10th February 2025
Actor	Commuter
Description	The user enables notifications for congestion alerts based on predefined thresholds.
Flow of Events	<ol style="list-style-type: none"> 1. Users enable congestion notifications. 2. User sets a congestion threshold (e.g., 80% occupancy). 3. The system monitors real-time congestion levels. 4. The system sends push notifications when the threshold is exceeded.
Alternative Flows	- If the user disables notifications, alerts are not sent.
Exceptions	- If the network is down, the system retries sending alerts for 10 minutes before failing.

Includes	Customisable alert preferences.
Special Requirements	Push notifications must be delivered instantly.
Assumptions	The user grants notification permissions to the app.
Notes & Issues	Introduce time-based alerts (e.g., "Alert me only during peak hours").

UC-004: Alternative Route Suggestions with Train Congestion Awareness

Field	Details
Use Case ID	UC-004
Use Case Name	Find Alternative Route Suggestions
Created By	AGARWAL RADHIKA
Last Updated By	AGARWAL RADHIKA
Date Created	25th January 2025
Date Last Updated	10th February 2025
Actor	Commuter
Description	The system provides alternative routes to users when congestion is high.
Flow of Events	<ol style="list-style-type: none"> 1. User selects "Find Alternative Route." 2. The system checks other MRT lines and lowers congestion paths. 3. The system suggests routes with travel time estimates.
Alternative Flows	- If no alternative exists, the system suggests: "Try travelling during non-peak hours."
Exceptions	- If the API is slow, the system displays estimated congestion levels instead of real-time data.
Includes	Map view of suggested routes.

Special Requirements	Congestion data must be refreshed every 5 minutes.
Assumptions	Users are willing to take alternative MRT routes.
Notes & Issues	Allow bus route integration in future updates.

UC-005: Generate Travel Report (LTA/SMRT)

Field	Details
Use Case ID	UC-005
Use Case Name	Analyse Travel Trends
Created By	Ang Ming Yang
Last Updated By	Ang Ming Yang
Date Created	25th January 2025
Date Last Updated	10th February 2025
Actor	LTA/SMRT
Description	Transport authorities analyse travel congestion patterns over time.
Flow of Events	<ol style="list-style-type: none">1. LTA/SMRT requests congestion trends.2. System generates peak-hour reports and travel insights.3. Authorities use data for schedule optimisation.
Alternative Flows	<ul style="list-style-type: none">- The system provides related congestion reports if a specific dataset is unavailable.
Exceptions	<ul style="list-style-type: none">- If the database is down, the system retries for 30 seconds before failing.
Includes	Comparison of train load across different times of the day.

Special Requirements	Data must be exportable as CSV/PDF for reporting.
Assumptions	Users are transport planners analysing trends.
Notes & Issues	Improve real-time reporting features.

UC-006: Generate Recommended Train Scheduling (LTA/SMRT)

Field	Details
Use Case ID	UC-006
Use Case Name	Optimise Train Scheduling
Created By	AGARWAL RADHIKA
Last Updated By	AGARWAL RADHIKA
Date Created	25th January 2025
Date Last Updated	10th February 2025
Actor	LTA/SMRT
Description	The system provides train schedule optimisation based on congestion trends.
Flow of Events	<ol style="list-style-type: none"> 1. LTA/SMRT accesses train load reports. 2. System suggests schedule adjustments based on congestion trends. 3. Authorities modify train frequency during peak hours.
Alternative Flows	- If suggested changes are not feasible, the system recommends the following best adjustments.
Exceptions	- The system triggers manual verification before applying changes if scheduling data is outdated.
Includes	AI-powered crowd distribution analysis.

Special Requirements	The system should predict congestion 24 hours in advance.
Assumptions	LTA/SMRT officials will regularly monitor congestion trends.
Notes & Issues	Future updates should allow dynamic scheduling changes based on live crowd levels.

5. UI Mockups

5.1 Overview of UI Design

The UI design focuses on delivering a minimalist, mobile-friendly, and visually intuitive experience for MRT commuters. It provides real-time and historical congestion data, allowing users to make informed travel decisions. Key design principles include:

- **Simplicity** – A clean interface with minimal input fields and quick access to data.
- **Mobile-first approach** – Optimized for smartphone users who need on-the-go congestion updates.
- **Visual Data Representation** – Heatmaps, colour-coded congestion indicators, and graphs for more straightforward interpretation.
- **Efficiency** – Users can access real-time congestion updates and alternative route suggestions with minimal clicks.

5.2 Screens and Features

1. Home Screen (Real-Time Congestion Data)

- **Functionality:**
 - Users enter their starting and ending MRT stations to retrieve congestion levels.
 - A colour-coded congestion heatmap (red = high, yellow = moderate, green = low) is displayed.
 - Buttons for alternative routes and historical data analysis are available.
- **Improvements from the original design:**
 - Your teammates' rough sketch only included text-based congestion levels, whereas this version introduces visual heatmaps for better understanding.
 - Quick-action buttons allow users to switch between different congestion insights seamlessly.



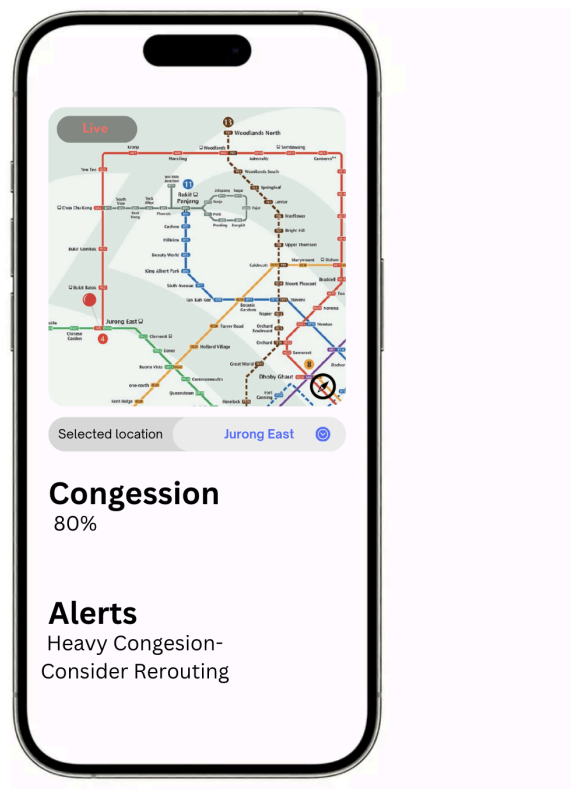
2. Alternative Route Suggestion Screen

- **Functionality:**

- Displays an interactive MRT map with different train lines.
- The original congested route is highlighted in red, while alternative routes with lower congestion are green.
- Below the map, users see estimated travel times and congestion levels for each alternative route.
- Users can select an alternative route using a "Choose Route" button.

- **Improvements from the original design:**

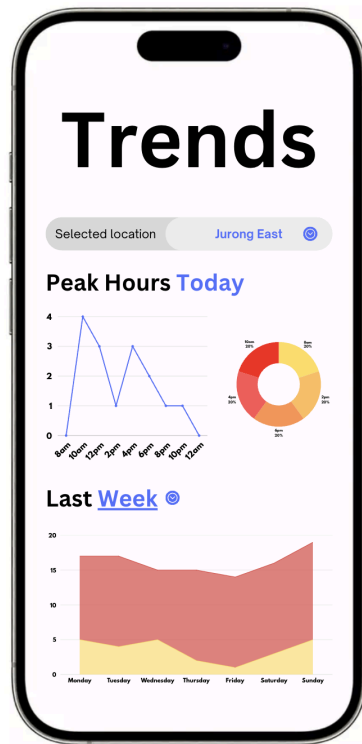
- The previous version did not effectively visualise alternative routes; this version presents them directly on an MRT network map.
- Travel times and congestion percentages help commuters make informed choices.



3. Historical Data Analysis Screen

- **Functionality:**
 - Allows users to select a date and time range to analyse past congestion levels.
 - Displays a line graph showing peak congestion times over different days.
 - Includes heatmaps for MRT stations, indicating past crowdedness levels.
 - Users can apply filters to compare congestion patterns across different timeframes.

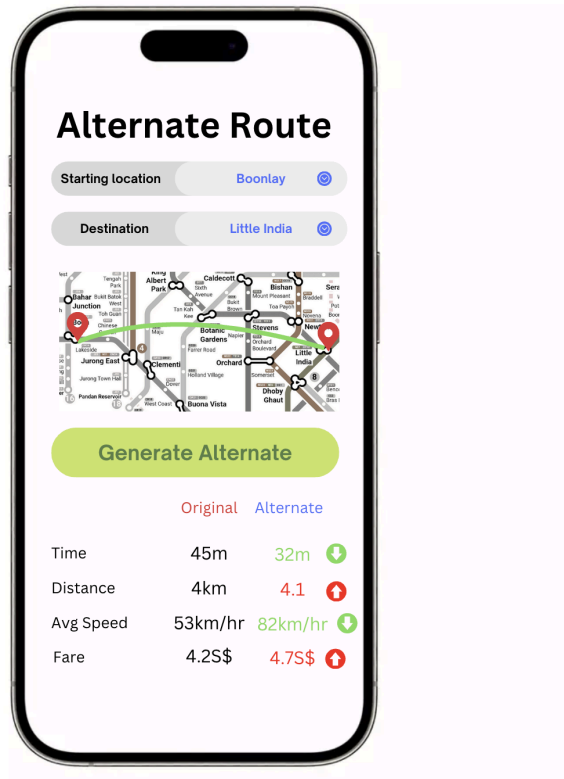
- **Improvements from the original design:**
 - The rough sketch only mentioned storing historical data but did not include a graphical representation.
 - The line graph and heatmap allow users to see congestion patterns clearly, making it easier to plan future trips.

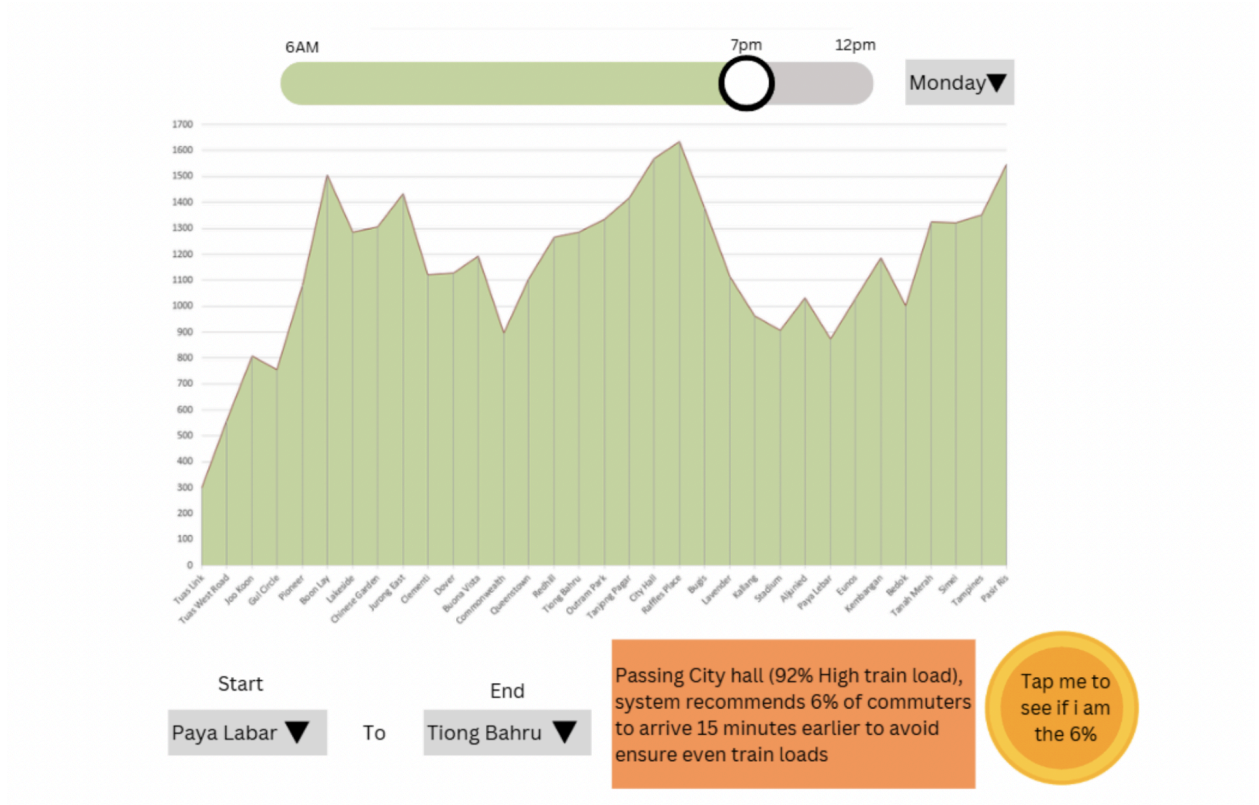


4. Congestion Notification Settings

- **Functionality:**
 - Users can set congestion alerts when train occupancy exceeds a specific threshold (e.g., 80%).
 - Includes toggle buttons to enable or disable alerts.
 - Users can limit notifications to peak hours to avoid excessive alerts.
- **Improvements from the original design:**
 - The rough sketch did not include a notification system—this version adds user-configurable congestion alerts.

- Customisable thresholds allow users to receive only the notifications they find relevant.





6. Data Processing & System Logic

This section explains how the system collects, processes, and analyses congestion data, ensuring commuters receive accurate real-time updates and valuable travel suggestions. The logic behind data handling, congestion prediction, and alternative route computation is outlined below.

6.1 MRT Train Load Calculation

The system calculates train congestion levels by fetching real-time passenger load data from the LTA DataMall API. The congestion percentage for each segment of the train line is determined using the formula:

$$\text{Congestion Level(\%)} = \left(\frac{\text{Current Passengers}}{\text{Max Train Capacity}} \right) \times 100$$

Processing Steps:

1. **Fetch live data** – The system queries the LTA DataMall API for train occupancy information.
2. **Data cleaning** – Removes anomalies such as incomplete or missing values.
3. **Compute train load per station** – The system maps passenger volumes to each MRT segment (e.g., EW5 → EW6)
Assumption: Number of trains in that hour is estimated based on LTA's train frequency estimates rounded
4. **Apply congestion thresholds** – Uses predefined levels:
 - Green (Low): 0 - 50%
 - Yellow (Moderate): 50 - 80%
 - Red (High): 80 - 100%
5. **Update heatmaps** – The congestion levels are displayed visually on the UI.

6.2 Real-Time Data Integration (LTA DataMall API)

The system connects to the LTA DataMall API every few minutes to refresh congestion data.

API Integration Flow:

1. Request real-time train occupancy → System sends an API call to LTA DataMall.
2. Receive JSON response → API returns live congestion data for each train.
3. Parse and store data → System extracts relevant fields, such as:
 - Train ID
 - Line & Station Code
 - Passenger Load %

- Timestamp

4. Display updated congestion levels → The system updates the UI every refresh cycle.

Error Handling:

- If the API fails, the system returns to historical trends instead of real-time data.
- If the API returns incomplete data, the system extrapolates congestion trends based on previous updates.

6.3 Historical Data Storage and Analysis

The system stores and analyses past train occupancy records to improve congestion predictions.

Storage Strategy

- Database Type: PostgreSQL (Relational Database)
- Data Retention: Stores 2 months of historical congestion data
- Data Structure:

Field name	Data type	Description
train_id	string	Unique train identifier
station_code	string	MRT station code
Timestamp	DateTime	The time when data was collected
passenger_load	Integer	Train occupancy percentage
Direction	String	Tuas-pasir ris

Congestion Trend Computation

1. Fetch past congestion records from the database.
2. Calculate the average train load per station over different periods.
3. Identify peak-hour trends based on historical patterns.
4. Display congestion trends using line graphs and heat maps

6.4 Alternative Route Computation

The system suggests alternative MRT routes using Dijkstra's Algorithm, ensuring the user finds a less crowded path.

Steps to Determine the Best Route

1. **Retrieve congestion data** → System checks real-time train loads for all possible routes.
2. **Build a congestion-based graph** → Each MRT station is treated as a node, with edges representing congestion levels.
3. **Apply Dijkstra's Algorithm** → Find the path with the lowest congestion weight.
4. **Suggest alternative routes** → Display travel time comparisons for different options.

Example Scenario:

The East-West Line may be overcrowded for commuters travelling from Jurong East (EW24) to Raffles Place (EW14).

Instead of waiting for a packed train, the system suggests:

- Route 1: Take Downtown Line (DTL) from Bukit Panjang (DT1) → Bugis (DT14) → Raffles Place
- Route 2: Travel earlier or later based on past congestion trends

6.5 Predictive Congestion Alerts

- **Historical congestion data** (previous train load patterns).
- **Real-time fluctuations** (live occupancy changes).
- **Day & time factors** (e.g., Monday 8 AM = High Congestion).

How Predictions Work:

1. **Train a predictive model** → Uses Linear Regression or Random Forest to forecast congestion.

2. **Apply the model to future time slots** → Determines expected train occupancy.
 3. **Alert users ahead of time** → Sends push notifications if a train is predicted to be over 80% full.
-

6.6 System Workflow Summary

Data Processing Flow:

- 1 **Real-time data collection** → Fetches LTA API data every few minutes
- 2 **Data cleaning & validation** → Removes missing or incorrect entries
- 3 **Compute congestion percentages** → Applies thresholds & color coding
- 4 **Store past congestion records** → Saves train load history for trend analysis
- 5 **Predict future congestion** → Uses past trends & machine learning
- 6 **Suggest alternative routes** → Runs shortest path algorithm based on congestion levels
- 7 **Notify users of high congestion** → Sends alerts for peak-hour predictions

7. Testing & Validation

The Testing & Validation phase ensures that the Train Congestion Analysis System functions accurately, efficiently, and reliably under different conditions. This section covers functional testing, performance testing, and edge case handling to verify system robustness.

7.1 Test Cases for Functional Requirements

Test cases are structured test scenarios designed to validate whether a specific feature functions as expected. Each test case has a unique ID, a description of the feature being tested, input conditions, expected output, and pass/fail criteria.

Test Cases Table

Test Case ID	Feature Tested	Test Description	Expected Result	Pass/Fail Criteria
TC-01	Real-Time Congestion Display	The user enters the start and end MRT stations and clicks "Check Congestion."	The system displays a congestion heatmap with colour-coded train occupancy levels	Pass if the correct congestion data loads within 2 seconds
TC-02	Invalid Station Handling	The user enters an invalid station name and clicks "Check Congestion."	The system displays an error message: <i>"Station not found. Please enter a valid MRT station."</i>	Pass if the system rejects invalid input and prompts the user to re-enter

TC-03	Alternative Route Suggestion	The user selects "Find Alternative Route" for a crowded station	The system suggests at least one alternative route with travel time and congestion levels	Pass if at least one valid alternative is displayed within 2 seconds
TC-04	Historical Data Analysis	The user selects a past date range for congestion trends	The system displays congestion trends using graphs and heatmaps	Pass if the system correctly retrieves and visualises historical data
TC-05	Congestion Notifications	The user enables notifications and sets an alert for 80% congestion	The system sends a push notification when congestion exceeds 80%	Pass if the user receives a notification when the congestion threshold is met
TC-06	Database Retrieval	The system queries past congestion data for multiple stations	Historical data is retrieved and displayed in under 3 seconds	Pass if query execution time is below 100ms and data visualisation is accurate
TC-07	API Integration	The system fetches data from the LTA DataMall API	The API response is processed correctly, and congestion data is updated in real time.	Pass if the latest congestion levels are shown correctly.

7.2 Performance Testing (Response Time & Data Accuracy)

Performance testing ensures the system remains fast and responsive, even under high user load and peak-hour congestion.

Performance Testing Metrics

1. Response Time

- The system should retrieve congestion data within 2 seconds.
- Historical congestion trends should load in under 3 seconds.
- Alternative route suggestions should be generated within 2 seconds.
- Congestion notifications should be sent within 5 seconds after exceeding the congestion threshold.

2. Scalability Test

- Simulate 1,000 concurrent users accessing congestion data.
- Verify that server response times remain under 5 seconds.

3. Stress Testing

- Simulate extreme conditions, such as a sudden spike in train congestion (e.g., train breakdowns or special events).
- Ensure the system does not crash or fail under high API request loads.

4. Database Query Optimization

- Ensure historical congestion data retrieval occurs within 100ms.
- Optimise database indexing for faster searches.

5. API Latency Check

- The system must handle API call failures gracefully and retry if necessary.
- The LTA DataMall API will fall back to historical congestion predictions if it is unavailable.

Performance Testing Scenarios

Scenario	Expected Outcome
10 concurrent users check congestion levels	No delay, response time < 2 seconds
1,000 concurrent users check congestion levels	The server handles requests, and response time < 5 seconds
API fails temporarily	The system retries fall back to stored data if unavailable
User searches historical congestion trends for 30 days	Data loads within 3 seconds

7.3 Edge Cases & Fail-Safes

This section ensures that the system gracefully handles unexpected user inputs, API failures, and extreme congestion spikes.

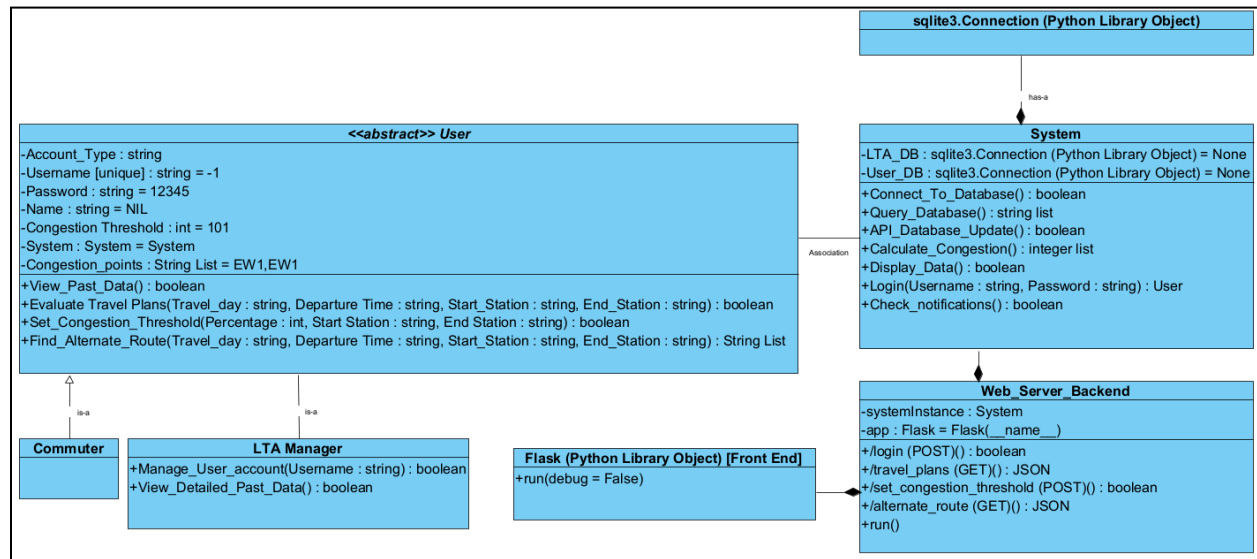
Edge Cases Table

Edge Case ID	Scenario	Expected System Behavior
EC-01	LTA API is down	The system switches to historical congestion trends instead of real-time data
EC-02	The user enters an empty station field	The system displays, "Please enter a valid station name."
EC-03	The user selects a date range with no data available	The system displays "No data available for the selected range."
EC-04	Congestion alert is set at 10,0%, but no train reaches full capacity system	tem does not send unnecessary notifications
EC-05	The user enters a station name in the wrong format (e.g., numbers instead of letters)	The system detects and rejects incorrect input
EC-06	The user has a poor internet connection	The system loads a lightweight version without heatmaps
EC-07	The user spams congestion checks too quickly	The system implements rate limiting to prevent excessive API requests
EC-08	API returns broken data	The system ignores bad data and retries

EC-09	Congestion spikes suddenly (e.g., train breakdown)	The system updates the congestion status dynamically and alerts users
-------	---	---

8 Classes

8.1 Class diagram



8.1.1.0 <<Abstract>> User

Attributes:

- Account_Type: String
- Username [Unqie]: String = -1
- Password: String = 12345
- Name: String = NIL
- Congestion Threshold: int = 101

Methods:

- +View_Past_Data() : boolean
- +Evaluate Travel Plans(Travel_day : string, Departure Time : string, Start_Station : string, End_Station : string) : boolean
- +Set_Congestion_Threshold(Percentage : int, Start Station : string, End Station : string) : boolean
- +Find_Alternate_Route(Travel_day : string, Departure Time : string, Start_Station : string, End_Station : string) : String List

8.1.1.1 Commuter Extends User

Attributes: NIL

Methods: NIL

8.1.1.1 LTA Manager Extends User

Attribute(s): NIL

Methods:

+Manage_User_account(Username : string) : boolean

+View_Detailed_Past_Data(): boolean **overrides** View_Past_Data()

8.1.2 System class

Attributes:

-LTA_DB : sqlite3.Connection (Python Library Object) = None

-User_DB : sqlite3.Connection (Python Library Object) = None

Methods:

+Connect_To_Database() : boolean

+Query_Database() : string list

+API_Database_Update() : boolean

+Calculate_Congestion() : integer list

+Display_Data() : boolean

+Login(Username : string, Password : string) : User

+Check_notifications() : boolean

8.1.3 Database entity class (sqlite3.Connection : Python Library Object)

Reusable Software

View documentation at <https://docs.python.org/3/library/sqlite3.html>

8.1.4 Web_Server_Backend

Attributes:

-systemInstance : System

-app : Flask = Flask(__name__)

Methods:

+/login (POST)() : boolean

+/travel_plans (GET)() : JSON

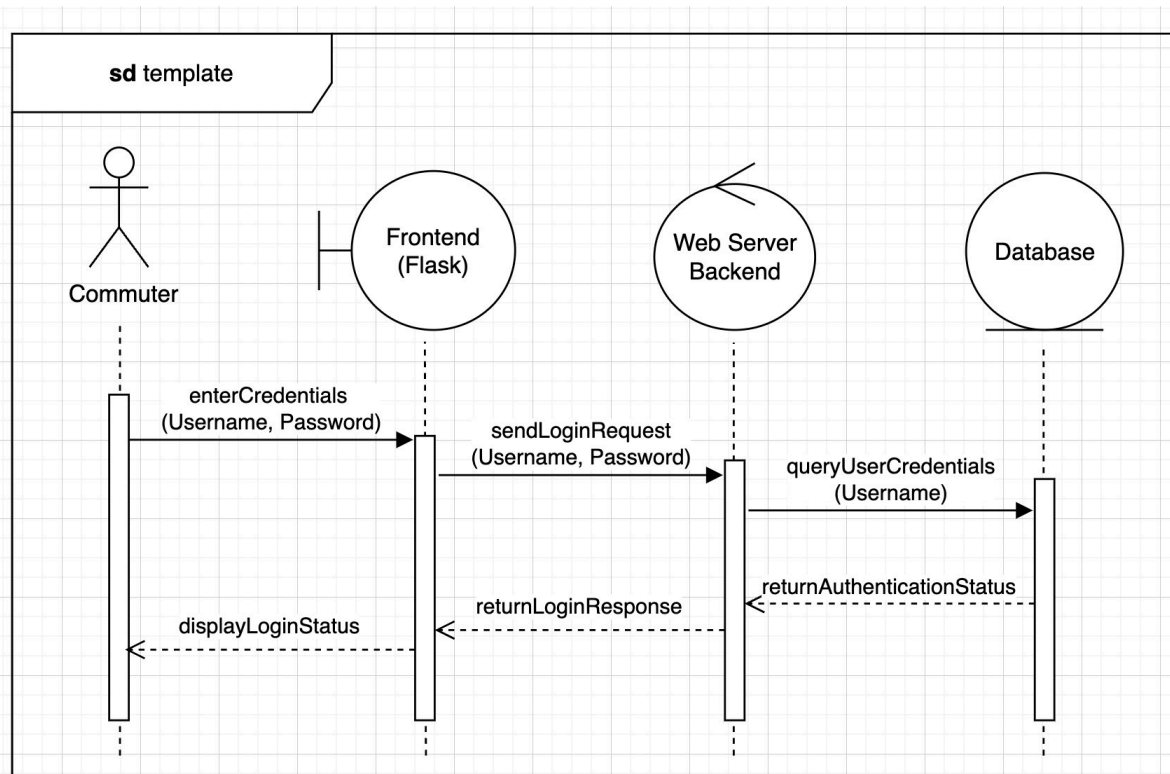
+/set_congestion_threshold (POST)() : boolean

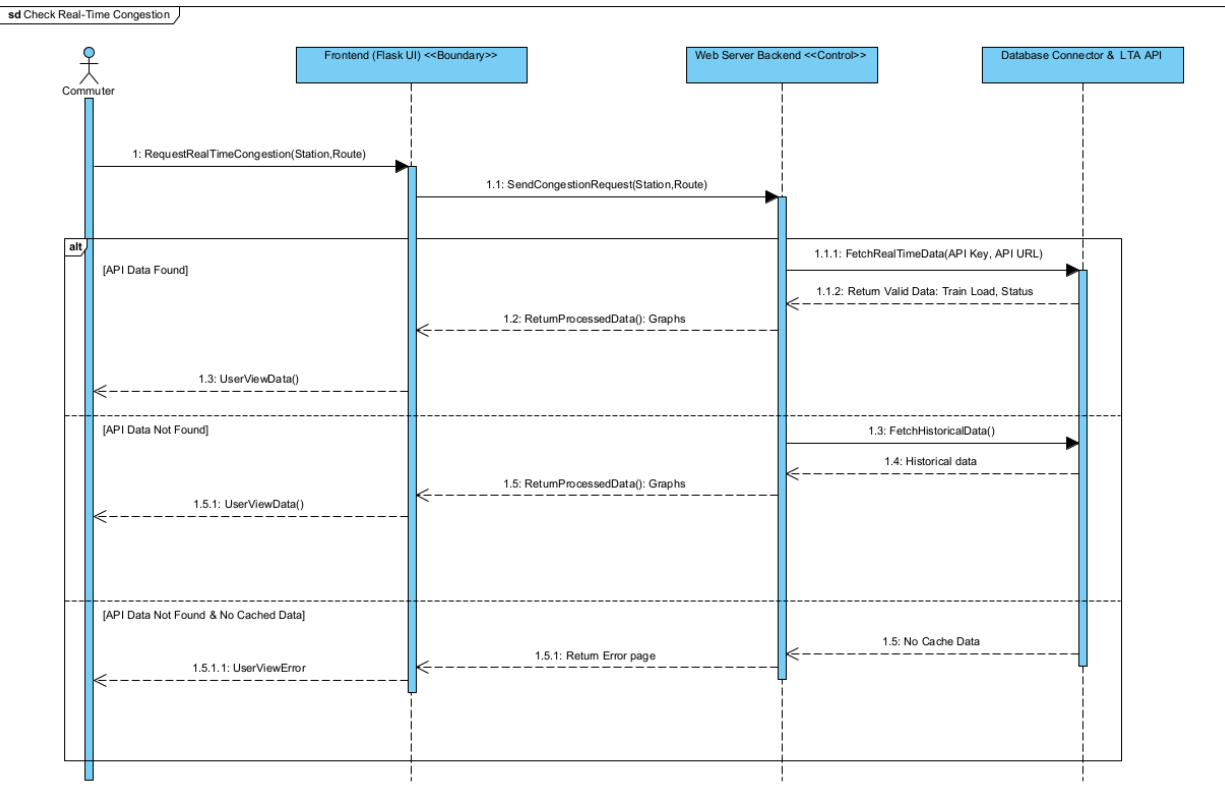
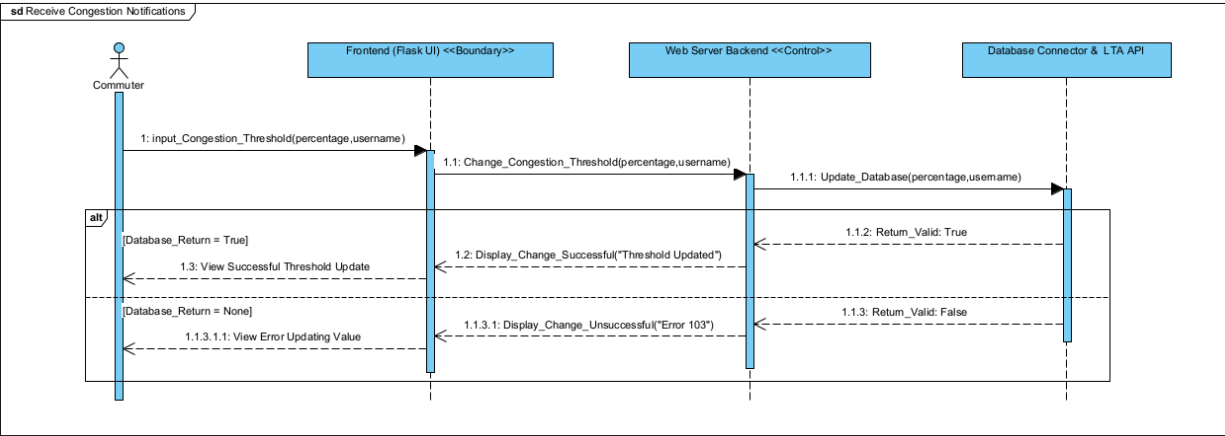
+/alternate_route (GET)() : JSON

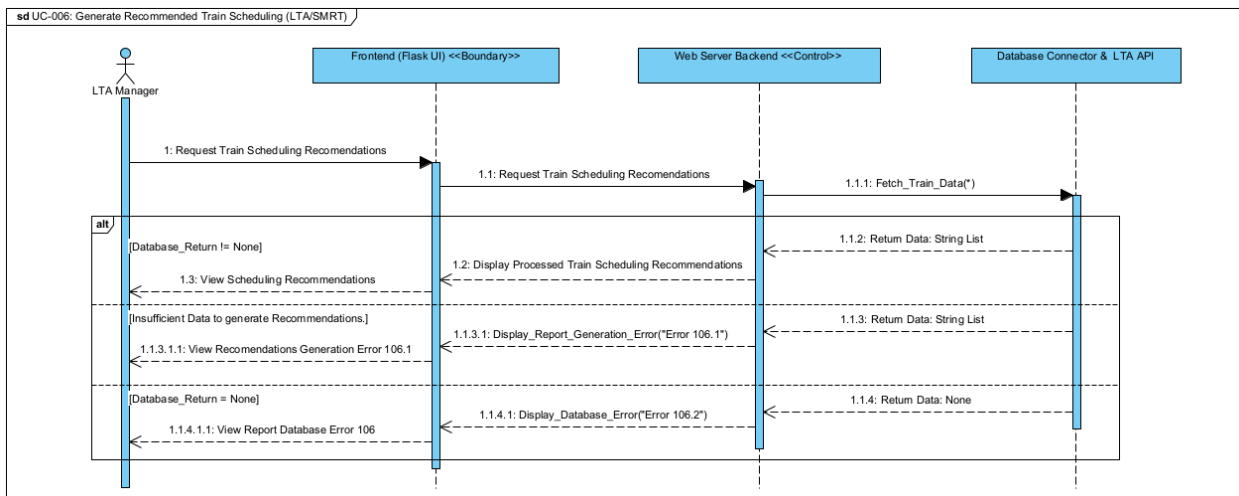
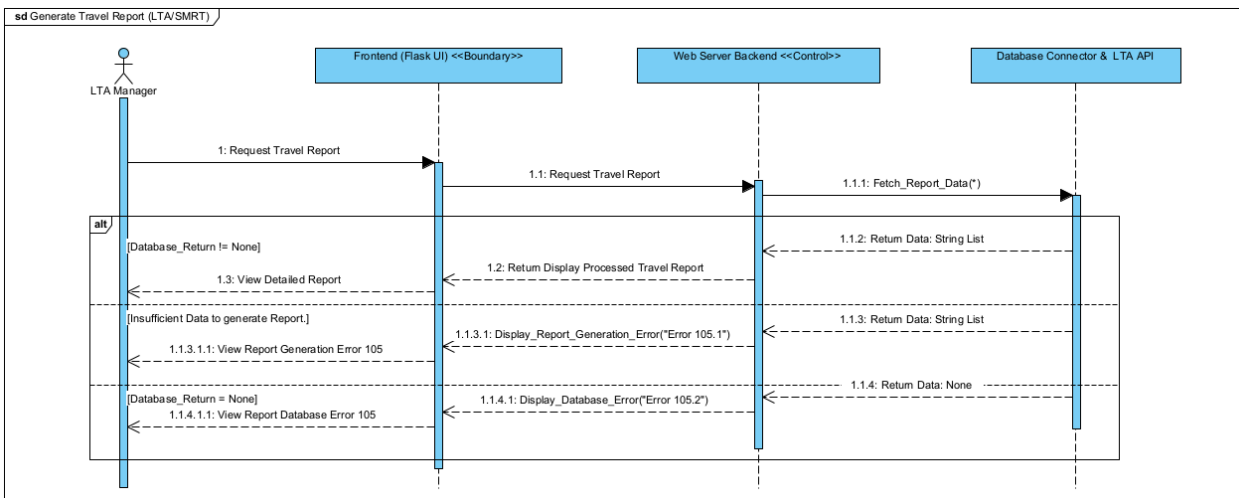
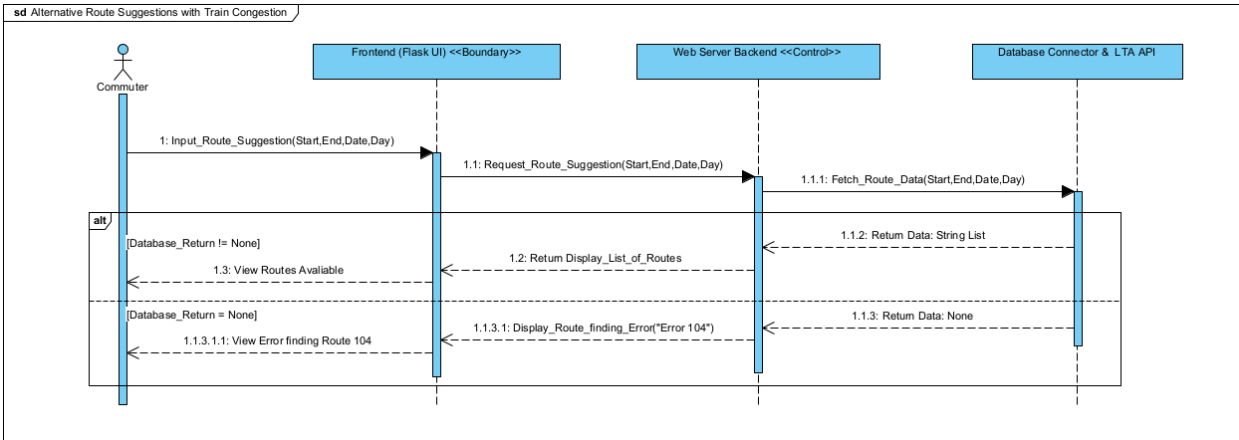
+run()

8.1.6 Class diagram Stereotype

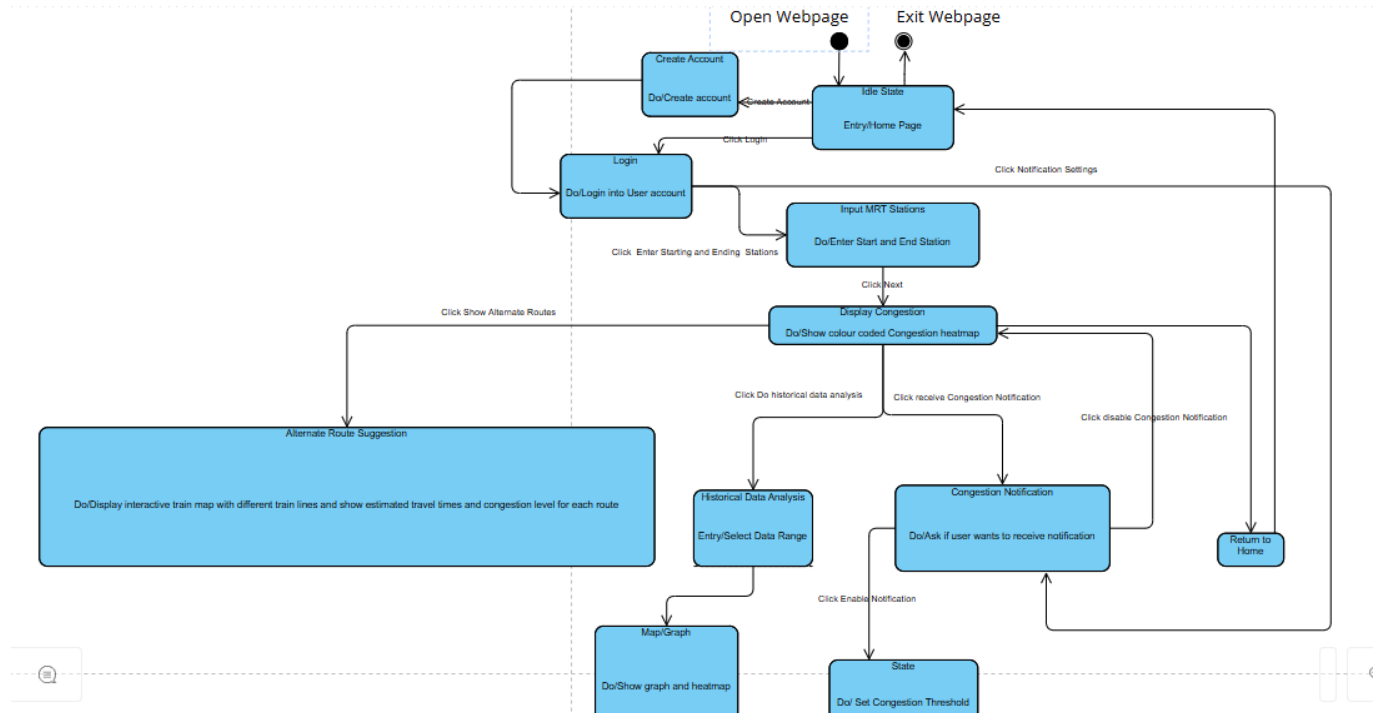
8.2 Sequence Diagrams







8.3 Dialog Map (state machine diagram)



<https://online.visual-paradigm.com/w/tgccurvb/diagrams/#diagram:workspace=tgccurvb&proj=0&id=3&type=StateMachineDiagram&width=11&height=8.5&unit=inch>

9. Future Enhancements & Expansion

The Train Congestion Analysis System is designed for scalability, allowing future enhancements to improve efficiency, user experience, and predictive accuracy. This section outlines upgrades, including additional train line integration, multimodal transport planning, and AI-driven congestion prediction.

9.1 Additional Train Lines Integration

Current Limitation

The initial version of the system focuses on the East-West Line (EWL) due to its high congestion levels. However, other MRT lines, such as the Downtown Line (DTL), North-South Line (NSL), Circle Line (CCL), and Thomson-East Coast Line (TEL), also experience peak-hour crowding.

Enhancement Plan:


Additional MRT lines will be integrated using a modular API structure to scale the system. The system will:

- Fetch real-time congestion data from LTA DataMall for multiple lines.
- Expand heatmap coverage to visualise congestion across all train lines.
- Enable multi-line congestion analysis, helping users decide between different MRT networks.

Technical Implementation:

- Extend database schema to accommodate data from multiple MRT lines.
- Implement dynamic route calculation, allowing users to see congestion comparisons between lines.
- Introduce network-wide congestion alerts, notifying users when congestion shifts between lines.

Example Future Use Case

 A commuter travelling from Jurong East (EWL) to Paya Lebar (CCL) could receive real-time updates about whether switching to Downtown Line (DTL) at Bugis would be faster and less crowded.

9.2 Integration with Bus Services for Seamless Transport

Planning

Current Limitation

The current system only focuses on MRT congestion and does not consider bus availability. Some commuters may switch from MRT to buses when congestion is high.

Enhancement Plan:


A future upgrade will integrate bus congestion and arrival timings to provide a holistic travel experience. The system will:

- Include bus route data from LTA DataMall's Bus Arrival API.
- Suggest alternative bus routes when MRT congestion is too high.
- Calculate combined MRT + Bus travel times, ensuring the fastest multimodal journey.

Technical Implementation

- Fetch real-time bus occupancy data to estimate bus crowd levels.
- Implement MRT-to-Bus transition mapping, allowing users to see alternative bus routes at crowded stations.
- Develop a route comparison engine that evaluates travel time + congestion for MRT and bus options.

Example Future Use Case

 A commuter at Boon Lay MRT (EWL) may be advised to take Bus 179 to NTU instead of MRT if train congestion is at 90%, but bus availability is high.

9.3 Machine Learning for Crowdedness Prediction

Current Limitation

The system currently suggests alternative routes based on historical congestion trends. However, it does not predict future congestion based on live travel patterns.

Enhancement Plan


A machine learning (ML) model will be implemented to predict congestion before it happens. This will allow:

- Proactive congestion alerts, warning users before a station becomes overcrowded.
- Personalised travel recommendations, adjusting based on a commuter’s usual travel time.
- Live congestion forecasting uses weather, events, and peak-hour behaviour.

Technical Implementation

- Train a Time-Series Prediction Model (LSTM or Random Forest) using:
 - Historical congestion data (past 6 months).
 - Live API data (real-time congestion).
 - External factors (public holidays, weather, train disruptions).
- Implement AI-driven congestion alerts, notifying users if their chosen train will likely be crowded 15 minutes in advance.
- Enhance alternative route recommendations by using real-time congestion forecasts.

Example Future Use Case

 A commuter travelling at 8:30 AM from Pasir Ris (EWL) to City Hall (NSL) may receive a prediction alert saying:

"Expected congestion at City Hall MRT (90% full) in 15 minutes. Consider taking the Circle Line instead for a faster journey."

Enhancement Area	Original Plan	Future Expansion
MRT Coverage	Focus on East-West Line (EWL)	Expand to all MRT lines (DTL, CCL, NSL, TEL, etc.)

Alternative Transport	Only train congestion data	Add bus route integration to suggest MRT+Bus combinations
Prediction Capabilities	Historical trends only	ML-based congestion prediction with proactive alerts

10. Conclusion

The Train Congestion Analysis System was developed to address the persistent issue of MRT overcrowding in Singapore. By leveraging real-time congestion data, historical trends, and alternative route suggestions, the system helps commuters make smarter travel decisions while assisting transport authorities in optimising train operations.

Key Achievements of the Project:

1. **Real-time congestion tracking** – Uses LTA DataMall API to provide up-to-date train load data.
2. **Alternative route recommendations** – Implements Dijkstra's Algorithm to suggest less crowded MRT routes.
3. **Historical congestion analysis** – Displays past congestion trends using graphs and heat maps.
4. **Congestion alerts & notifications** – Users can set personalised alerts when congestion exceeds a threshold.
5. **Robust testing & validation** – Ensures high system reliability, fast response times, and error handling.

Future Scalability & Impact:

While the initial implementation focuses on the East-West Line (EWL), the system is designed for scalability. Future expansions will introduce:

1. Integration with all MRT lines (Downtown, North-South, Circle, Thomson-East Coast)
2. Multimodal transport planning (bus integration for seamless MRT + bus journeys)

3. Machine learning-based congestion forecasting (predict train overcrowding before it happens)