VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY

THE INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

SOFTWARE ENGINEERING

IT076IU

FINAL REPORT

Topic: Railway System

By Group 07 – Member List

1. Nguyễn Văn Ngọc Hải	ITCSIU23007	Project Manager
2. Trần Đức Hải Triều	ITWEIU20027	Team Member
3.Lê Hoàng Thái Tuấn	ITITIU20340	Team Member
4. Nguyễn Đức Trí	ITITIU19223	Team Member
5. Trần Quốc Nam	ITITIU21250	Team Member
6. Trần Nguyễn Minh Trân	ITITIU20323	Team Member
7. Lê Nhật Duy	ITITWE22143	Team Member
8. Phạm Thiện Nhân	ITITUN23004	Team Member
9. Nguyễn Hữu Hoàng	ITITIU19014	Team Member
10 .Nguyễn Khải (member out)	ITITIU20222	Team Member

Instructor: Assoc. Prof. Nguyen Thi Thuy Loan

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Background

The railway system can be considered as a modern and effective approach that controls and operates trains for commercial transportation or even for exploitation purposes. Nowadays, complex data systems like railways require extremely high accuracy and high precision during operation. Therefore, this system is also needed for guaranteed quality, safety, and speed during function work. As a result, an explicit database of a railway system is the crucial element in data management. Our final project is willing to conduct the simple version of this database and more specifically, our database will facsimile the railway system based on information collected on the Internet sources.

Problem Statement

Current railway systems involve manual processes that are prone to errors, particularly in tracking train journeys, booking tickets, and managing customer data. These inefficiencies can lead to delayed services, incorrect bookings, and poor customer experiences. A robust database is needed to automate these processes, ensuring smoother operations and higher customer satisfaction.

Scope and Objectives

The primary goal of this project is to develop a simplified database that tracks train journeys, including start and finish times and the stations passed during the journey, then display it in an interface. Additionally, the project seeks to enhance the railway reservation system by reducing errors in ticket booking and cancellations, making it more convenient for customers, and storing vital information like customer details and seat availability.

Customer Requirements

- A system that tracks and displays train journeys and provides real-time information on the status of trains.
- An error-free booking and cancellation system for railway tickets.
- A user-friendly interface for booking tickets and viewing train schedules.
- Secure storage of customer and seat availability data.

Structure of report

This report is organized into eleven chapters. Chapter 2 elaborates further on the expected timeline of the project's tasks. Chapter 3 details the methodology employed for the development of the proposed system. Chapter 4 presents the means and technology that were used throughout the development process in order to implement the system's variety of features. Chapter 5 discusses the findings, compares the algorithm's performance with existing methods, and evaluates its effectiveness. Chapter 6 then evaluates the problems and challenges that the group had to face throughout the process of development and covers the solutions that were devised to remedy them. Finally, Chapter 7 concludes the report and puts out the vision that this project was committed to and how it has affected the whole group.

CHAPTER 2

PROJECT TIMELINE & IN-CHARGE TABLE

1. Project Timeline

Phase	Task	Time Start	Time End	Duration

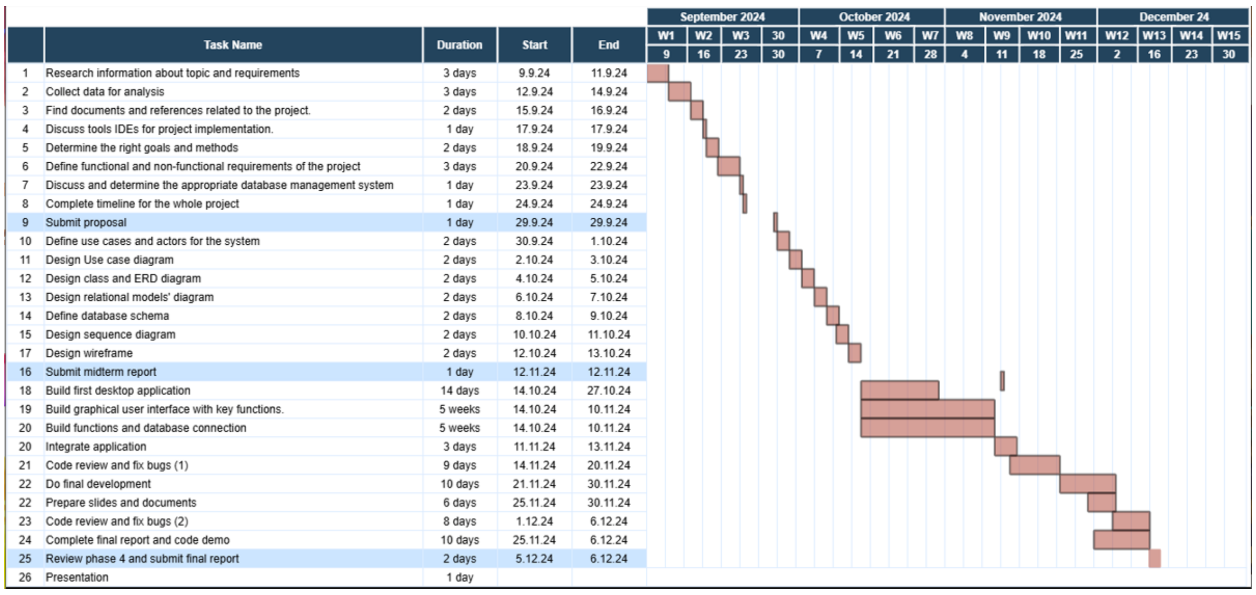
1.Project analysis and planning	Research information about topic and requirements	9/9/2024	11/9/2024	3 days
	Collect data for analysis	12/9/2024	14/9/2024	3 days
	Find documents and references related to the project.	15/9/2024	16/9/2024	2 days
	Discuss tools IDEs for project implementation.	17/9/2024	17/9/2024	1 day
	Determine the right goals and methods	18/9/2024	19/9/2024	2 days
	Define functional and non-functional requirements of the project	20/9/2024	22/9/2024	3 days
	Discuss and determine the appropriate database management system	23/9/2024	23/9/2024	1 day
	Complete timeline for the whole project	24/9/2024	24/9/2024	1 day
	Submit proposal	29/9/2024	29/9/2024	1 day
2. Design diagram	Define use cases and actors for the system	30/9/2024	1/10/2024	2 days
	Design Use case diagram	2/10/2024	3/10/2024	2 days
	Design class and ERD diagram	4/10/2024	5/10/2024	2 days
	Design relational models' diagram	6/10/2024	7/10/2024	2 days
	Define database schema	8/10/2024	9/10/2024	2 days
	Design sequence diagram	10/10/2024	11/10/2024	2 days
	Design wireframe	12/10/2024	13/10/2024	2 days
	Submit midterm report	12/11/2024	12/11/2024	1days

3.Build desktop application	**Build first desktop application**	14/10/2024	27/10/2024	14 days

	Build graphical user interface with key functions.	14/10/2024	10/11/2024	5 weeks
	Build functions and database connection	14/10/2024	10/11/2024	5 weeks
	Integrate application	11/11/2024	13/11/2024	3 days
	Code review and fix bugs (1)	14/11/2024	20/11/2024	9 days
4. Review project and presentation	Do final development	21/11/2024	30/11/2024	10 days
	Prepare slides and documents	25/11/2024	30/11/2024	6 days
	Code review and fix bugs (2)	1/12/2024	6/12/2024	6 days
	Complete final report and code demo	25/11/2024	4/12/2024	10 days
	Review phase 4 and submit final report	5/12/2024	6/12/2024	2 days
	Presentation			1 day

Table 1: Project Timeline

2. Gantt Chart [1]



	Task Name	Duration	Start	End
1	Research information about topic and requirements	3 days	9.9.24	11.9.24
2	Collect data for analysis	3 days	12.9.24	14.9.24
3	Find documents and references related to the project.	2 days	15.9.24	16.9.24
4	Discuss tools IDEs for project implementation.	1 day	17.9.24	17.9.24
5	Determine the right goals and methods	2 days	18.9.24	19.9.24
6	Define functional and non-functional requirements of the project	3 days	20.9.24	22.9.24
7	Discuss and determine the appropriate database management system	1 day	23.9.24	23.9.24
8	Complete timeline for the whole project	1 day	24.9.24	24.9.24
9	Submit proposal	1 day	29.9.24	29.9.24
10	Define use cases and actors for the system	2 days	30.9.24	1.10.24
11	Design Use case diagram	2 days	2.10.24	3.10.24
12	Design class and ERD diagram	2 days	4.10.24	5.10.24
13	Design relational models' diagram	2 days	6.10.24	7.10.24
14	Define database schema	2 days	8.10.24	9.10.24
15	Design sequence diagram	2 days	10.10.24	11.10.24
17	Design wireframe	2 days	12.10.24	13.10.24
16	Submit midterm report	1 day	12.11.24	12.11.24
18	Build first desktop application	14 days	14.10.24	27.10.24
19	Build graphical user interface with key functions.	5 weeks	14.10.24	10.11.24
20	Build functions and database connection	5 weeks	14.10.24	10.11.24
20	Integrate application	3 days	11.11.24	13.11.24
21	Code review and fix bugs (1)	9 days	14.11.24	20.11.24
22	Do final development	10 days	21.11.24	30.11.24
22	Prepare slides and documents	6 days	25.11.24	30.11.24
23	Code review and fix bugs (2)	8 days	1.12.24	6.12.24
24	Complete final report and code demo	10 days	25.11.24	6.12.24
25	Review phase 4 and submit final report	2 days	5.12.24	6.12.24
26	Presentation	1 day		

Figure 1: Gantt Chart

3. Work-breakdown Structure

From the work breakdown structure, the railway system project can be divided into six phases: **analysis**, **design**, **implementation**, **testing**, **deployment**, and **maintenance**.

- **Analysis:**

This phase includes the determination of the goals and scope, identification of stakeholders like passengers, railway operators, and administrators, and their requirements. The focus of the analysis is on functionalities expected from the system, such as ticket booking, train schedules, and real-time updates. This forms the basis necessary to ensure the right project direction meets its stakeholders' expectations.

- **Design:**

In this stage, the system architecture's design is built with tools such as a use case diagram, sequence diagram, and entity-relationship diagram. These artifacts provide a visual interaction of users with the system. User interface mockups are designed, examples being ticket booking pages and train tracking dashboards. Designs for the database are made. This ensures the designs meet the functional requirements of the railway system.

- **Implementation:**

In the Implementation phase, the development environment and necessary frameworks for the railway system are set up. The core tasks involve the development of modules for ticket booking, train scheduling, passenger management, and real-time notifications. The database will be integrated with the system, including server configurations to ensure that everything goes smoothly. This stage now turns the conceptual design into a working railway system application.

- **Testing:**

Testing ensures the system functions as intended. Test scenarios are created for a series of functionalities, from book tickets up to schedule display and delays. Fully tested-ensuring reliability of the system and fixing bugs or inconsistencies before releasing the system. This stage is the most important part to ensure the system's performance and user satisfaction.

- **Deployment:**

Deployment involves hosting the railway system on a server, configuration of network settings, and migration of databases to the production environment. This phase is where one ensures the functioning of the system as expected in a live environment-that is, accessible both to passengers and operators. During this phase, some final adjustments could be done to optimize the performance of the system.

- **Maintenance:**

Performance monitoring of the system, updating with new features, and maintenance are continuously performed post-deployment. Activities include troubleshooting issues reported by users, integrating new functionalities such as dynamic pricing or train delay predictions, and adapting to emerging technologies. In this way, the railway system is kept efficient and up to date with user needs.

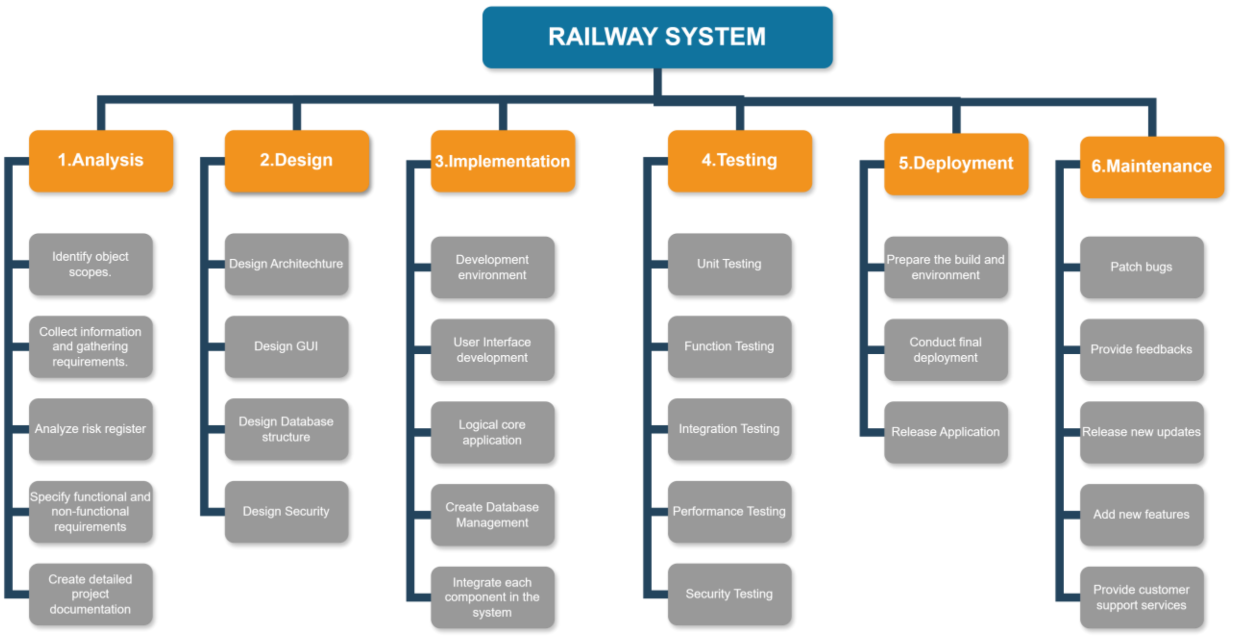


Figure 2: Work-Breakdown Structure

4. Product backlog

Story Name and Description	PRIORITY	COMMENT
User Login and Role System: As a user, I want to log in to the system, so that I can securely manage the railway management system.	HIGH	Includes UI and backend logic for secure authentication.
Train Schedule Management: As a user, I want to add, edit, and delete train schedules, so that I can update and maintain the system effectively.	HIGH	CRUD operations for train schedules.
Ticket Management System: As a user, I want to view, create, and cancel tickets, so that I can manage passenger bookings.	HIGH	Booking and ticket details stored locally.
Offline Train Tracking: As a user, I want to display train routes and predefined statuses, so that I can provide accurate tracking information.	MEDIUM	Simulate train statuses with static data.
Admin Panel: As a user, I want an interface to manage schedules, tickets, and feedback, so that I can perform administrative tasks efficiently.	HIGH	Integrated management interface for users.
Data Security: As a user, I want all data to be securely stored, so that I can protect sensitive information from unauthorized access.	HIGH	Data encryption and secure file storage implementation.
User Interface Enhancement: As a user, I want an intuitive and accessible UI, so that I can manage the system without difficulty.	MEDIUM	Focus on usability and accessibility.

Table 2: Product Backlog

5. Sprint backlog

Task	Responsibility	COMMENT
Design login screen UI	Nam	Design a desktop login interface for user authentication.
Implement function of login	Nam	Develop secure authentication logic for desktop access.
Set up role-based access control	Nam	Ensure only authenticated users can access system features.
Build railway management screen	Trieu, Tri, Nhan	Simulate the status of trains using static data.
Develop railway management logic	Duy, Tri, Tran	Implement backend logic for train schedule operations.
Create and connect database to the application	Tran	To create data of railway information and implement integration to the system
Create use case diagram	Hai, Hoang	Focus on usability and accessibility.
Write use case description	Hoang	Depict core system functionalities like login and scheduling.
Build ERD and database schema	Tuan	Detailed user interactions with key system features.
Build class diagram	Tuan	Design the database structure for whole system
Build sequence diagram	Duy, Hoang	Illustrate system workflows for each action in the system.
Test desktop application	Hai, Hoang, Tuan	Implement testing to find bugs and GUI issues
Do the test case report	Hai, Hoang, Tuan	Write the report for the whole testing to ensure everything works well.

Table 3: Sprint Backlog

CHAPTER 3

METHODOLOGY

The railway system would aim to enhance the overall experience and speed up various operations. In such a context, events within the railway system, along with customer service, could easily be handled by the staff manager. The database system would simplify the process of managing and accessing information regarding the passenger, employee, trains, or even the station when in need. This software also enables the staff to create reservations and present the passengers with their respective ticket information as requested. This chapter will provide details

1. Functional Requirements

Customer Module:

- **Create/Login Account**
  - Users should be able to create a new account or log in with existing accounts.
- **Booking Tickets to Cart**
  - Users can search for train routes and add tickets to their cart.
- **View/Add/Remove Ticket to the Cart**
  - Users can view their current cart, add new tickets, or remove existing tickets.
- **View Tickets List**
  - Users should be able to view their purchased or reserved tickets.
- **Tracking Train**
  - Users can track the live status of their train.
- **CRUD Number of Tickets on Each Train Route**
  - The system should allow staff to create, read, update, and delete the number of available tickets for each train route.
- **Update Customer Information on Ticket**
  - User can update customer details associated with a ticket.
- **CRUD Train/Train Route Information**
  - The system should support creating, reading, updating, and deleting train and route information.
- **CRUD Data Storage**
  - Users can create, read, update, and delete data storage entries.
- **Manage User**
  - Users can manage accounts, roles, and access permissions.

2. Non-functional Requirements

Performance:

- The system must be able to support hundreds of concurrent users in peak hours.
- Response time to user interaction should be optimized for use.

Scalability:

- The system should be designed to accommodate future scaling to handle additional train routes and more users with minimal reduction in system performance.

Reliability:

- The system must have 100% uptime to always ensure availability.
- In case of any system failure, automatic recovery procedures should be applied.

Usability:

- The user interface should be intuitive and easy to use. It shall be evident that all categories of users, regardless of the level of their computer skills, can work with the system productively.

Security:

- All user's data, including login/password and private information, must be protected and properly guarded against unauthorized access.
- The system should have secure authentication mechanisms

Maintainability:

- The system should be developed with a modular architecture so that updates and changes do not affect other parts.
- Code is commented on, if relevant, for easier maintenance.

Compatibility:

- The application should ensure responsiveness on mobile to give users the best experience across all devices.

Compliance:

- Personal data protection regulations should be implemented in developing the system to ensure the privacy of users.

Availability:

- It should perform smooth support for operations by the creation of maintenance windows during off-peak hours.

Efficiency:

- Database queries should be optimized to reduce loading times.
- The system is expected to use minimum resources to avoid extremely high costs of operation.

1. Use Case Diagram [2]

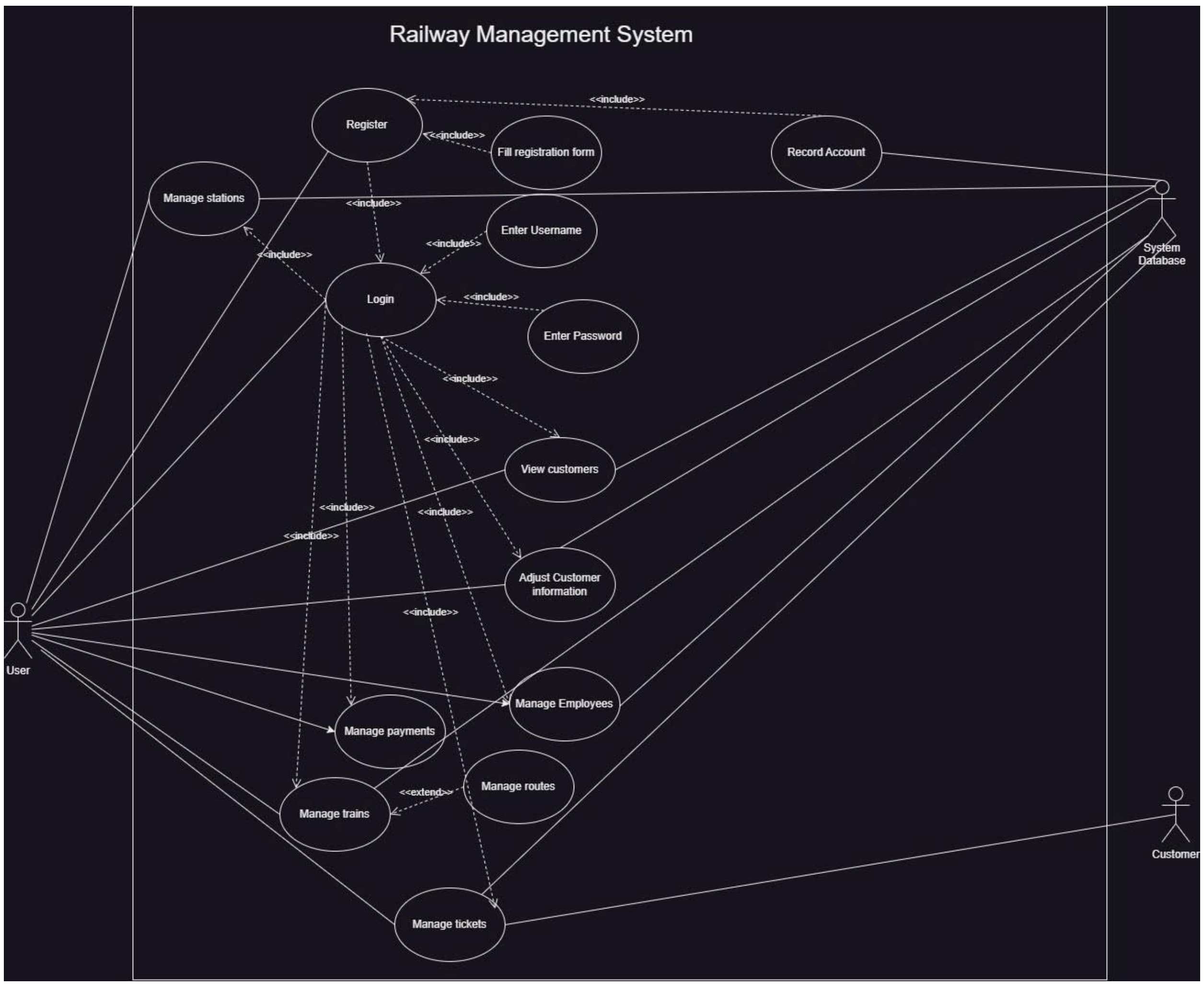


Figure 3: Use Case Diagram

1. Use Case Description

UC Name	Register
UC ID	UC1
Description	For new users to register an account used for logging into the system to use it
Actors	User, System Database
Priority	Must have
Inputs	User enters
Outputs	New account within database

Pre-conditions	The User has pressed the Register button	
Post conditions	The UserID is unique and doesn't already exist in the database	
Basic Course	User	System Database
	1. User enters information for their account's name and password	1.1. Verify account name and password in database
		1.2. Create new Account with entered information if userID doesn't already exists
		1.1. Send a message saying the account has been created and send the User back to the Login page
		1.2. System stores the new Account within the database
Exception Course	If the user enters false information, then they will receive a message saying they entered the wrong information instead	

UC Name	Login	
UC ID	UC2	
Description	For registered users to enter their account information and use the system	
Actors	User, System Database	
Priority	Must have	
Inputs	User enters account information	
Outputs	The User accesses the system management screen	
Pre-conditions	None	
Post conditions	The User has accessed the system management screen	
Basic Course	User	System Database
	1. User enters Username and Password.	1.1 System finds Username and Password in database
		1.2. If the entered information is correct, send the user to the system management screen
Exception Course	If the entered information is false, ask the user to enter their username and password again	

UC Name	Adjust Customer Information	
UC ID	UC3	
Description	For adjusting a passenger's information	
Actors	User, System Database	
Priority	Must have	
Inputs	The User select "Passenger Information"	
Outputs	Changed Passenger Information	
Pre-conditions	The User has logged into their account	
Post conditions	The passenger information has been changed	
Basic Course	User	System Database
	1.User presses "Passenger Information"	1.1 Opens Passenger Information screen
	2. User enters the Passenger's ID as well as their information	2.1 System checks the existence of Passenger's ID
		2.2. If the ID exists, change the information associated with the ID in the database to be the same as that which the User entered
Exception Course	If the ID does not exist, notify the user	

UC Name	View Customers	
UC ID	UC4	
Description	View a list containing	
Actors	User, System Database	
Priority	Must have	
Inputs	The User select "Passenger Information"	
Outputs	List of Customers	
Pre-conditions	The User has logged into their account	
Post conditions	The customer list is shown	
Basic Course	User	System Database
	1.User presses "All Passengers"	1.1.Opens passenger list

Exception Course		

UC Name	Manage Employees	
UC ID	UC5	
Description	A collection of functions for managing employees	
Actors	User, System Database	
Priority	Must have	
Inputs	The User selects “Employee”	
Outputs	List of Employee Commands	
Pre-conditions	The User has logged into their account	
Post conditions		
Basic Course	User	System Database
	1.User presses “All Employees”	1.1.Opens employee list
	2.User presses “Search And Update Employee”	2.1.Opens Employee tab
	2.2 User enters Employee’s ID and new Information	2.3. Updates Employee within the database based on new information entered
	3.User selects “Add Employee”	3.1. Opens New Employee tab
	3.2.User enters the information of the new employee	3.3. Saves new employee into the database
Exception Course	If Employee ID already exists while adding a new employee, send error message. If Employee ID doesn’t already exist which updating employee, send error message.	

UC Name	Manage Trains
UC ID	UC6
Description	A collection of functions for managing trains
Actors	User, System Database

Priority	Must have	
Inputs	The User selects “Train”	
Outputs	List of Train Commands	
Pre-conditions	The User has logged into their account	
Post conditions		
Basic Course	User	System Database
	1.User presses “All Trains”	1.1.Opens train list
	2.User presses “Search And Update Train”	2.1.Opens Train tab
	2.2 User enters Train’s ID and new Information	2.3. Updates Train within the database based on new information entered
	3.User selects “Add Train”	3.1. Opens New Train tab
	3.2.User enters the information of the new train	3.3. Saves new train into the database
Exception Course	If Train ID already exists while adding a new employee, send error message. If Train ID doesn’t already exist which updating employee, send error message.	

UC Name	Manage Stations	
UC ID	UC7	
Description	A collection of functions for managing stations	
Actors	User, System Database	
Priority	Must have	
Inputs	The User selects “Station”	
Outputs	List of Station Commands	
Pre-conditions	The User has logged into their account	
Post conditions		
Basic Course	User	System Database

	1.User presses “All Stations”	1.1.Opens station list
	2.User presses “Search And Update station”	2.1.Opens Station tab
	2.2 User enters Station’s ID and new Information	2.3. Updates Station within the database based on new information entered
Exception Course	If Station ID doesn’t already exist which updating employee, send error message.	

UC Name	Manage Tickets		
UC ID	UC8		
Descripti on	A collection of functions for managing tickets		
Actors	User, System Database, Customer		
Priority	Must have		
Inputs	The User selects “Tickets”		
Outputs	List of Ticket Commands		
Pre-condition s	The User has logged into their account		
Post condition s			
Basic Course	User	System Database	Customer
	1.User presses “Ticket Report”	1.1.Opens ticket list	
	2.User presses “Book Ticket”	2.1.Opens Ticket tab	2.2, Customer asks for the specifics of the ticket they request
	2.3 User enters TicketID, Passenger’s ID and ticket’s infromation	2.4. Create a new ticket with the entered information into the database	
Exceptio n Course	If TicketID already exists within the database, send an error message instead		

Table 5

3. Class Diagram [3]

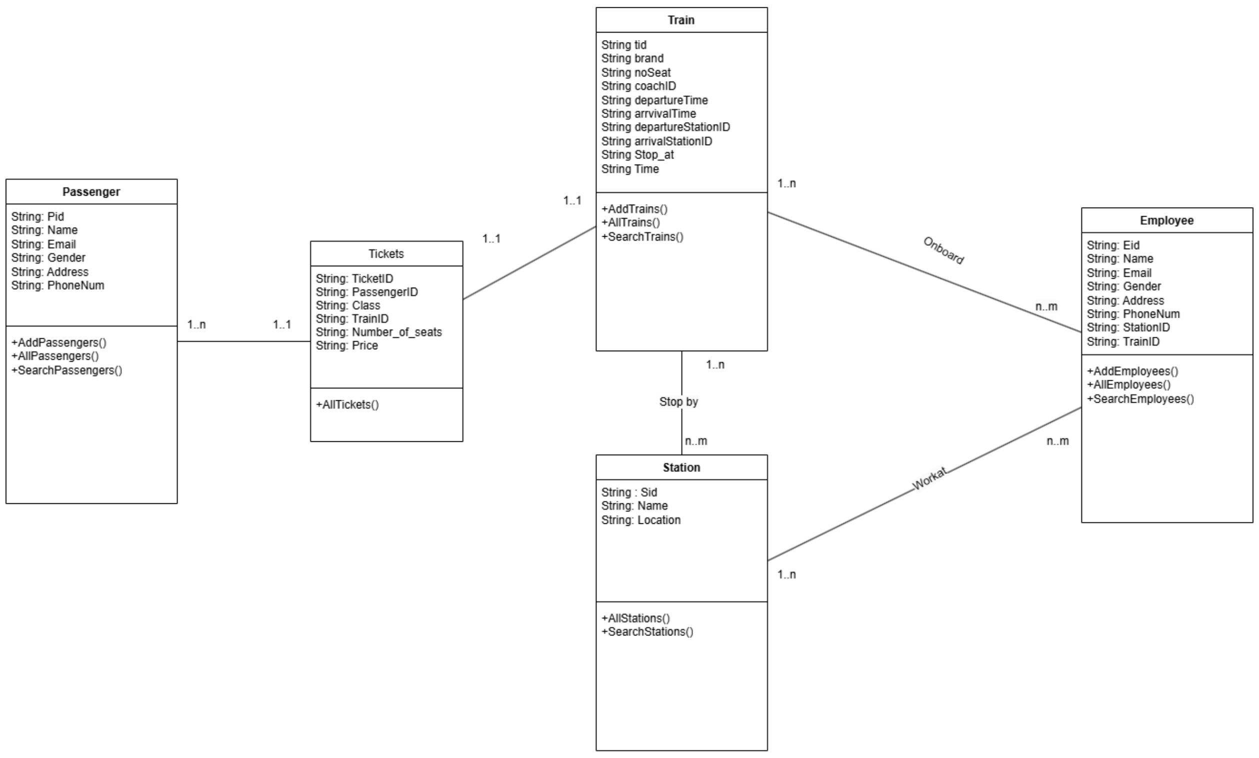


Figure 4: Class Diagram

4. Entity Relationship Diagram

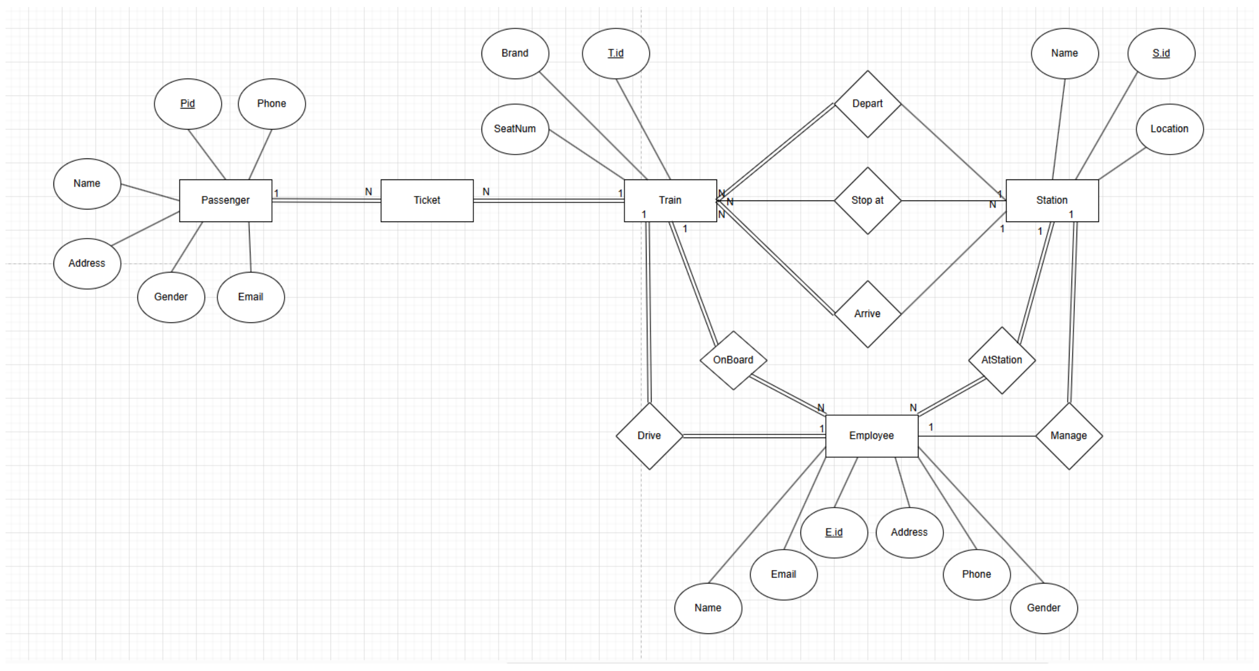


Figure 5: Entity Relationship Diagram

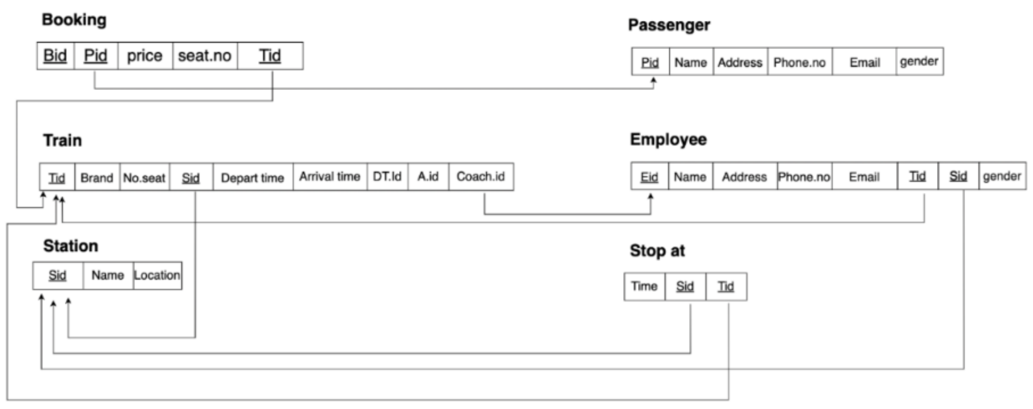


Figure 6: Entity Relationship Diagram (2)

5. Database Schema

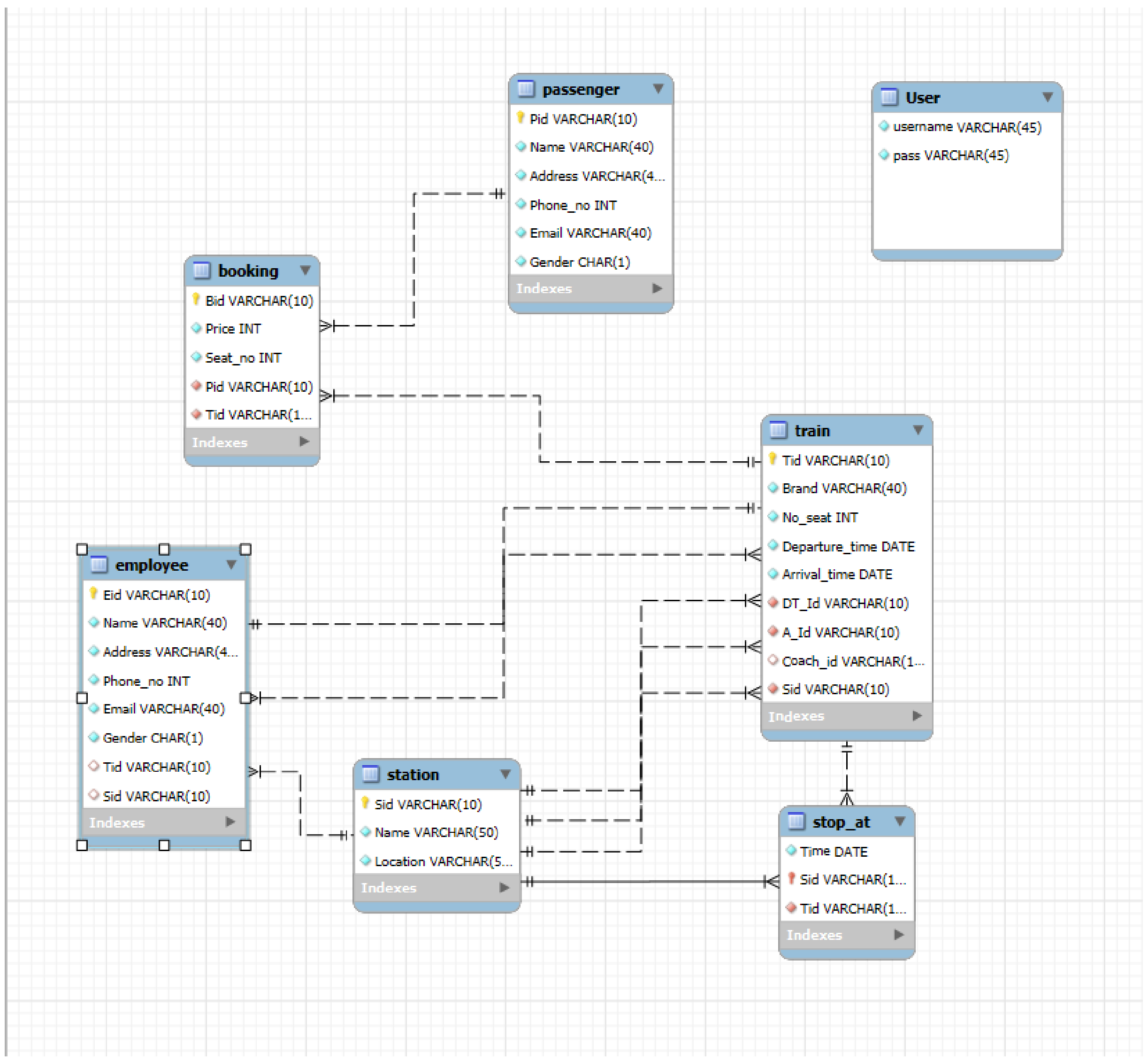


Figure 7: Database Schema

6. Database Description

1. Table: User

- **Purpose:** Stores login credentials for system access.
- **Columns:**
    - username (VARCHAR(45)): The username for the user (primary key).
    - pass (VARCHAR(45)): The password for the user.

2. Table: Passenger

- **Purpose:** Stores personal information about passengers.
- **Columns:**
    - Pid (VARCHAR(10)): Passenger ID (primary key).
    - Name (VARCHAR(40)): Name of the passenger.
    - Address (VARCHAR(40)): Address of the passenger.
    - Phone_no (INT): Passenger's phone number.
    - Email (VARCHAR(40)): Passenger's email address.
    - Gender (CHAR(1)): Gender of the passenger (M, F, or O).

3. Table: Booking

- **Purpose:** Tracks ticket booking details.
- **Columns:**
    - Bid (VARCHAR(10)): Booking ID (primary key).
    - Price (INT): Price of the ticket.
    - Seat_no (INT): Seat number assigned.
    - Pid (VARCHAR(10)): Passenger ID (foreign key, references Passenger.Pid).
    - Tid (VARCHAR(10)): Train ID (foreign key, references Train.Tid).

4. Table: Train

- **Purpose:** Stores details of trains.
- **Columns:**
    - Tid (VARCHAR(10)): Train ID (primary key).
    - Brand (VARCHAR(40)): Brand or name of the train.
    - No_seat (INT): Total number of seats in the train.
    - Departure_time (DATE): Departure time of the train.
    - Arrival_time (DATE): Arrival time of the train.
    - DT_Id (VARCHAR(10)): Departure station ID (foreign key, references Station.Sid).
    - A_Id (VARCHAR(10)): Arrival station ID (foreign key, references Station.Sid).
    - Coach_id (VARCHAR(10)): ID of the train's coach (foreign key, references Employee.Eid).
    - Sid (VARCHAR(10)): Current station ID (foreign key, references Station.Sid).

5. Table: Station

- **Purpose:** Manages details about train stations.
- **Columns:**
    - Sid (VARCHAR(10)): Station ID (primary key).
    - Name (VARCHAR(50)): Name of the station.
    - Location (VARCHAR(50)): Location of the station.

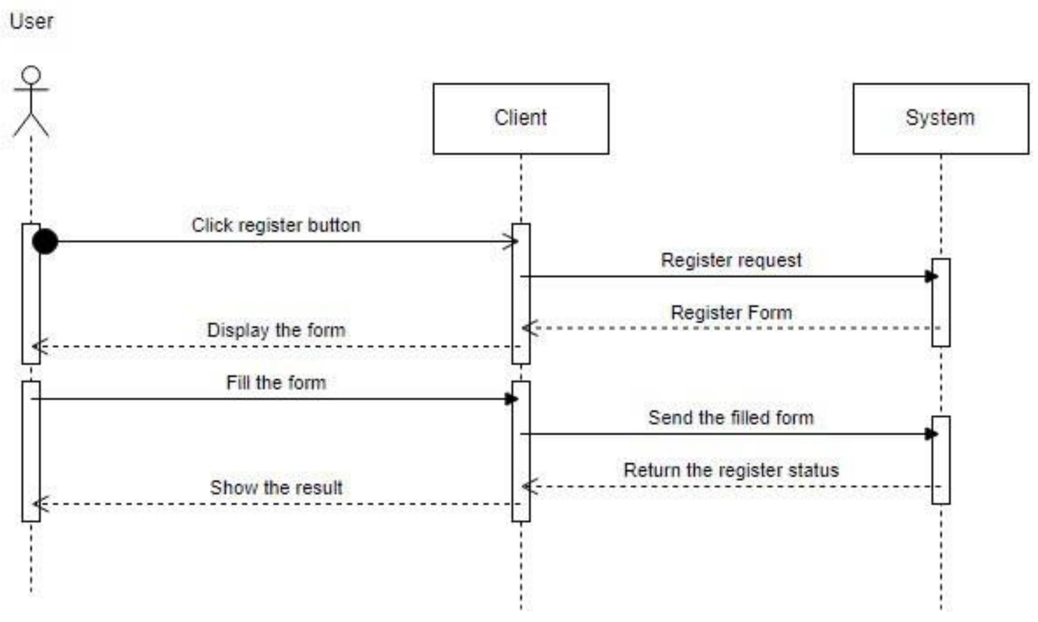
6. Table: Stop_at

- **Purpose:** Tracks the schedule of stops for trains at various stations.
- **Columns:**
    - Time (DATE): The time the train stops at the station.
    - Sid (VARCHAR(10)): Station ID (foreign key, references Station.Sid).
    - Tid (VARCHAR(10)): Train ID (foreign key, references Train.Tid).

7. Table: Employee

- **Purpose:** Stores information about employees working in the railway system.
- **Columns:**
    - Eid (VARCHAR(10)): Employee ID (primary key).
    - Name (VARCHAR(40)): Name of the employee.
    - Address (VARCHAR(40)): Address of the employee.
    - Phone_no (INT): Employee's phone number.
    - Email (VARCHAR(40)): Employee's email address.
    - Gender (CHAR(1)): Gender of the employee (M, F, or O).
    - Tid (VARCHAR(10)): Train ID (foreign key, references Train.Tid).
    - Sid (VARCHAR(10)): Station ID (foreign key, references Station.Sid).

7. Sequence Diagram



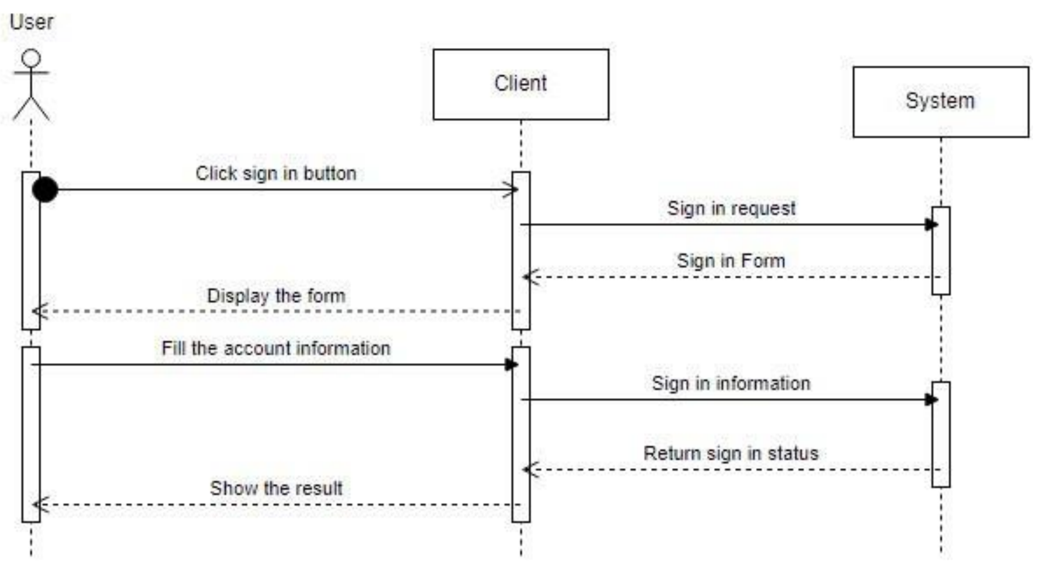


Figure 8: Sign in and register sequence diagram

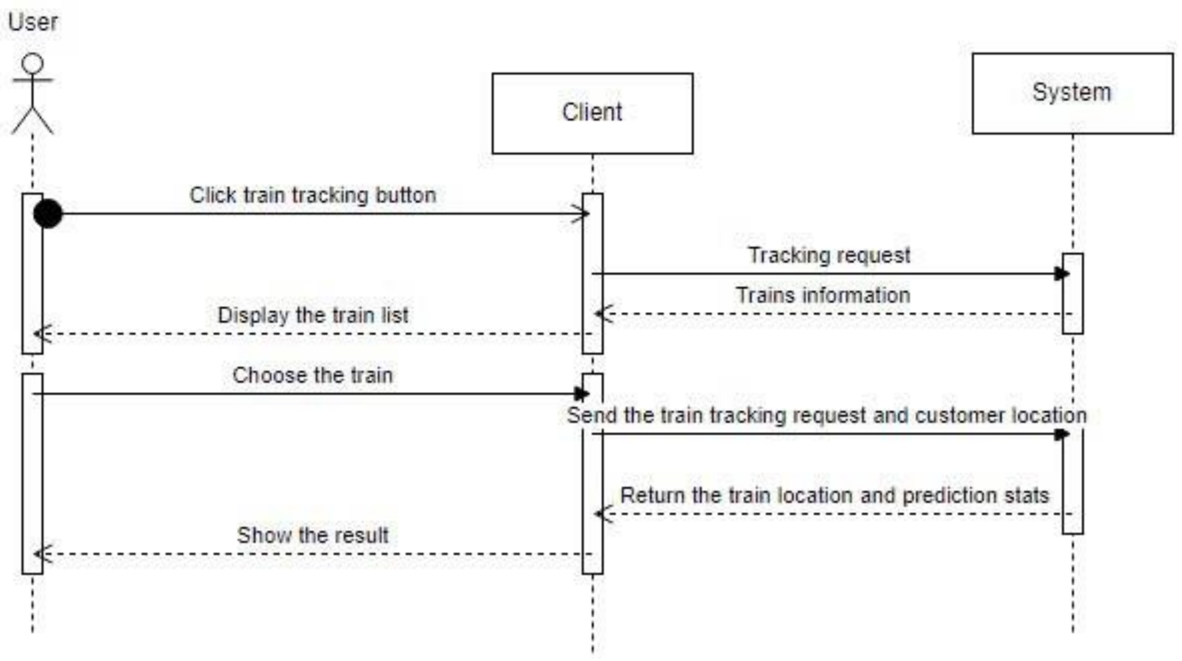
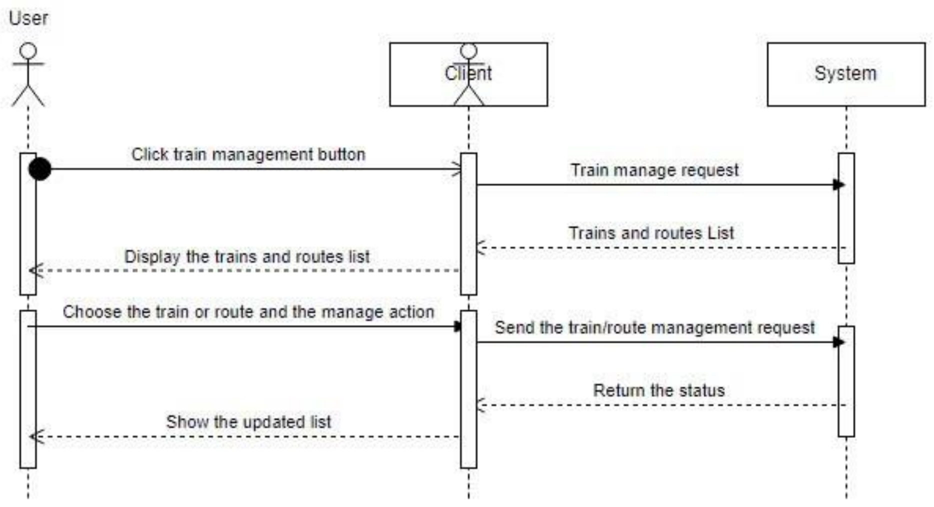

User
Client
System
Click train tracking button
Tracking request
Trains information
Display the train list
Choose the train
Send the train tracking request and customer location
Return the train location and prediction stats
Show the result

Figure 9: Train tracking sequence diagram


User
Client
System
Click train management button
Train manage request
Trains and routes List
Display the trains and routes list
Choose the train or route and the manage action
Send the train/route management request
Return the status
Show the updated list

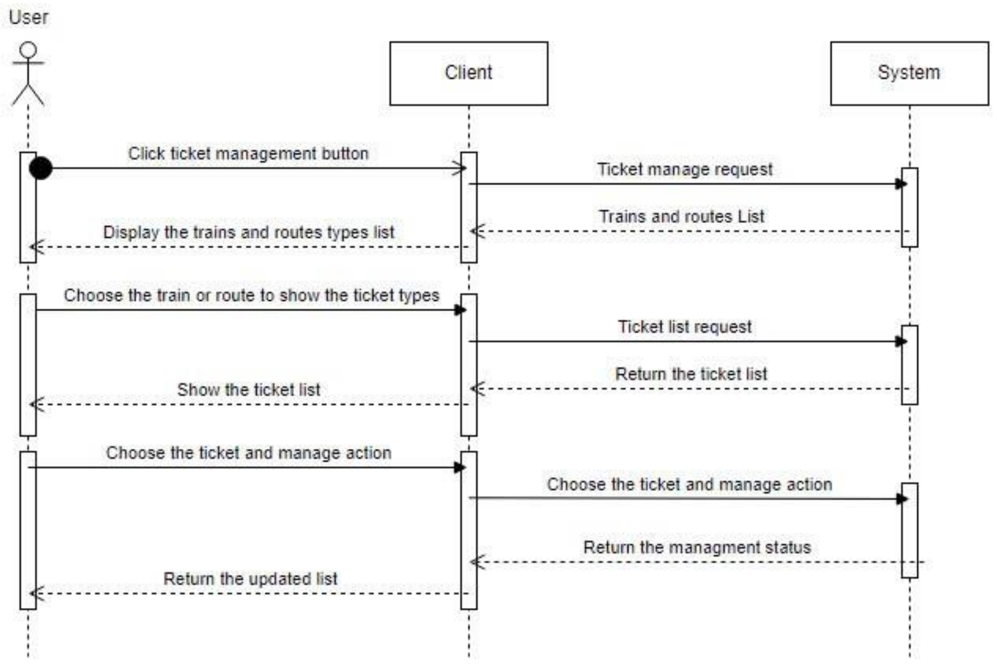
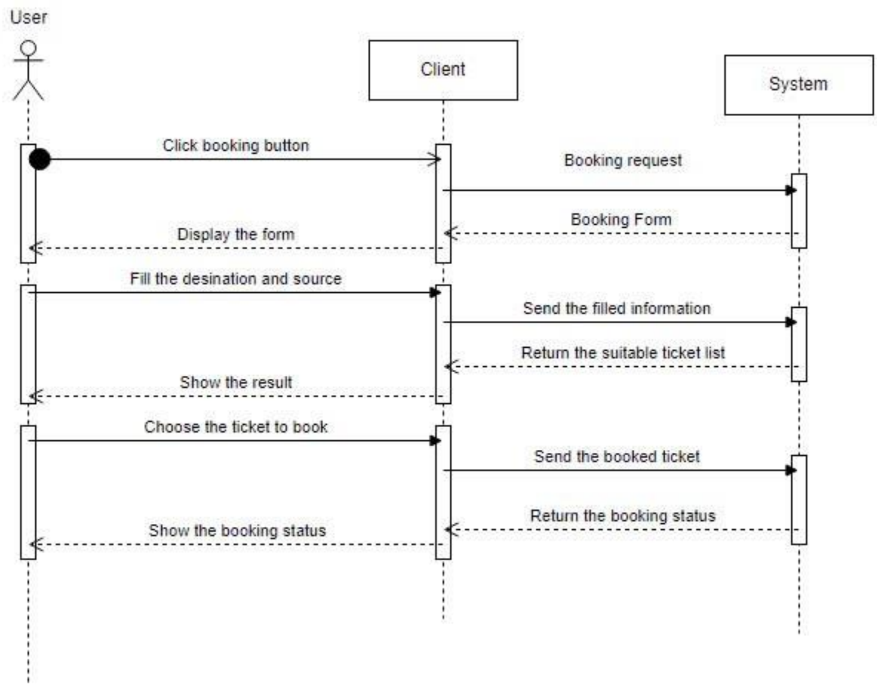

User
Client
System
Click ticket management button
Ticket manage request
Trains and routes List
Display the trains and routes types list
Choose the train or route to show the ticket types
Ticket list request
Return the ticket list
Show the ticket list
Choose the ticket and manage action
Choose the ticket and manage action
Return the managment status
Return the updated list

Figure 10: Train and ticket managment sequence diagram



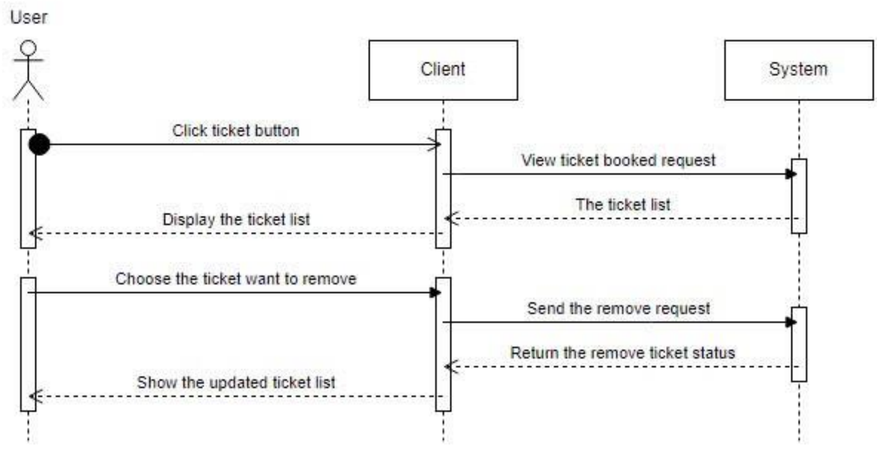

User
Client
System
Click ticket button
View ticket booked request
The ticket list
Display the ticket list
Choose the ticket want to remove
Send the remove request
Return the remove ticket status
Show the updated ticket list

Figure 11: Database Schema

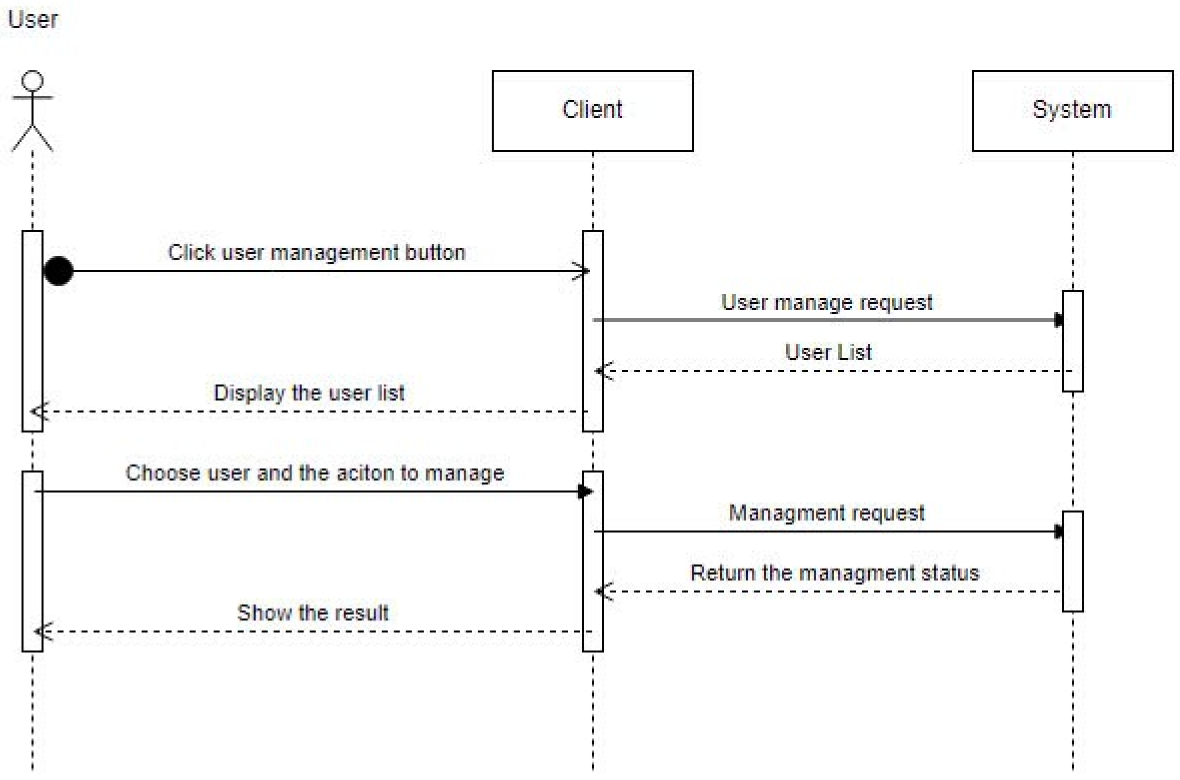


Figure 12: User management sequence diagram

CHAPTER 4

TECHNOLOGIES

In this case, we show all the technologies and implementations that are used for creating railway system projects:

- **Java:**
  Java is a high-level, object-oriented language that is extensively used to implement web-based applications, desktops, and mobiles. It grants platform independence because of its principle known as "Write Once, Run Anywhere", made possible through the use of the JVM. Java hosts rich libraries and frameworks, thus being suitable for enterprise-level applications, including the development of railway systems.
  Java is the core and only programming language in our project based on the knowledge that we used to apply in other courses such as OOP, DSA and PDM
- **MySQL:**
  MySQL Workbench is a software application that is used for configuring, managing, administering, and integrating SQL development. The instrument includes both scripting editors and graphical tools that work with objects and server features.
  In this project, the core of MySQL is to create a database that contains all the information and to implement the entities and the relationships between each of them, which creates efficiency of our building system.

- **NetBeans:**
  NetBeans IDE is a free and open-source integrated development environment for application development on Windows, Mac, Linux, and Solaris operating systems. The IDE simplifies the development of web, enterprise, desktop, and mobile applications that use the Java and HTML5 platforms. The IDE also offers support for the development of PHP and C/C++ applications.
  NetBeans IDE offers first-class tools for Java web, enterprise, desktop, and mobile application development. It is consistently the first IDE to support the latest versions of the JDK, Java EE, and JavaFX. It provides smart overviews to help you understand and manage your applications, including out-of-the-box support for popular technologies such as Maven. With its end-to-end application development features, constantly improving Java Editor, and continual speed and performance enhancements, NetBeans IDE sets the standard for application development with cutting edge technologies out of the box.
  With the support of NetBeans for providing Java development kit, we can easily handle all the system without building environment and libraries as the beginning. Once we can use it to write the function of the system, we also have the support of “. form” that we can build the GUI as well.

CHAPTER 5
IMPLEMENTATION & RESULT

1. Technology

- **Java: [5]**
  Java is a high-level, object-oriented language that is extensively used to implement web-based applications, desktops, and mobiles. It grants platform independence because of its principle known as "Write Once, Run Anywhere", made possible through the use of the JVM. Java hosts rich libraries and frameworks, thus being suitable for enterprise-level applications, including the development of railway systems.
  Java is the core and only programming language in our project based on the knowledge that we used to apply in other courses such as OOP, DSA and PDM
- **MySQL: [6]**
  MySQL Workbench is a software application that is used for configuring, managing, administering, and integrating SQL development. The instrument includes both scripting editors and graphical tools that work with objects and server features.
  In this project, the core of MySQL is to create a database that contains all the information and to implement the entities and the relationships between each of them, which creates efficiency of our building system.

- **NetBeans:**
  NetBeans IDE is a free and open-source integrated development environment for application development on Windows, Mac, Linux, and Solaris operating systems. The IDE simplifies the development of web, enterprise, desktop, and mobile applications that use the Java and HTML5 platforms. The IDE also offers support for the development of PHP and C/C++ applications.
  NetBeans IDE offers first-class tools for Java web, enterprise, desktop, and mobile application development. It is consistently the first IDE to support the latest versions of the JDK, Java EE, and JavaFX. It provides smart overviews to help you understand and manage your applications, including out-of-the-box support for popular technologies such as Maven. With its end-to-end application development features, constantly improving Java Editor, and continual speed and performance enhancements, NetBeans IDE sets the standard for application development with cutting edge technologies out of the box.
  With the support of NetBeans for providing Java development kit, we can easily handle all the system without building environment and libraries as the beginning. Once we can use it to write the function of the system, we also have the support of ". form" that we can build the GUI as well.

2. Testing

1. Registration

Test Case #: 1.1	**Test Case Name:** Registration Test Case
System: Railway Management System	**Subsystem:** Login
Designed by: Trần Quốc Nam	**Design Date:** 12/11/2024
Executed by: Nguyen Van Ngoc Hai	**Execution Date:** 18/11/2024
Short Description: Test registering a new account	

Step	Action	Expected System Response	Pass/Fail	Comments
1	Open the application	The system displays the login page.	Pass	Ensure the application loads.
2	Select "Register"	Registration Screen appears	Pass	Ensure the registration screen is functional.
3	Enter UserID and information and press "Create"	The system processes the request.	Pass	Verify no errors occur
4	Check post-condition 1		Pass	
5	Repeat step 3 with the same UserID	System sends error message	Pass	Verify that no accounts with the same IDs can exist
6	Check post-condition 2		Pass	
7	Select "Back"	Login Screen appears	Pass	

Table 6

Post-Conditions
- New User account is stored in database - There is no duplicate account in database - Can return to login page

2. Login

Test Case #: 2.1	**Test Case Name:** Login

System: Railway Management System	**Subsystem:** Login
Designed by: Trần Quốc Nam	**Design Date:** 12/11/2024
Executed by: Nguyen Van Ngoc Hai	**Execution Date:** 18/11/2024
Short Description: Test the login process	

Step	Action	Expected System Response	Pass/Fail	Comments
1	Open the application.	The system displays the login page.	Pass	Verify the login page loads.
2	Enter a Username and Password that already exists in database and press “Log In”	The system opens the railway management page	Pass	Ensure the Login function operates properly
3	Check post-condition 1		Pass	
4	Repeat step 1 and 2 but with wrong password	Receives a message saying user entered wrong username/password	Pass	

Table 7

Post-Conditions
- User enters management page

3. <u>Check Passengers</u>

Test Case #: 3.1	**Test Case Name:** PassList
System: Railway Management System	**Subsystem:** Railway
Designed by: Trần Đức Hải Triều	**Design Date:** 12/11/2024
Executed by: Nguyen Van Ngoc Hai	**Execution Date:** 18/11/2024
Short Description: Test the passenger list	

Step	Action	Expected System Response	Pass/Fail	Comments
1	Select "Passenger"	Dropdown with "All Passengers" appears	Pass	Verify the button works
2	Select "All Passengers " from dropdown	The system opens passenger list	Pass	Ensure that you can open passenger list
3	Check post-condition 1		Pass	
4	Select "Cancel	Return to Management screen	Pass	

Table 8

Post-Conditions
- Passenger list contains all passengers from the database

4. Check and Edit Employees

Test Case #: 4.1	**Test Case Name:** EmployeeTest
System: Railway Management System	**Subsystem:** Employee System
Designed by: Nguyễn Đức Trí	**Design Date:** 14/11/2024
Executed by: Nguyen Van Ngoc Hai	**Execution Date:** 18/11/2024
Short Description: Test the Employee System	

Step	Action	Expected System Response	Pass/Fail	Comments
1	Select "Employee"	Dropdown with "All Passengers","Add Employee" and "Search and Update Employees" appears	Pass	Verify the button works
2	Select "Add Employee" from dropdown	The system opens Add Employee page	Pass	
3	Enter Employee information	Employee added to database	Pass	
4	Check post-condition 1		Pass	
5	Repeat step 3 but with the same Employee ID	Receive error message	Pass	
6	Check post-condition 2		Pass	
7	Select "Cancel"	Closes page	Pass	
8	Repeat step 1 and select "Search and update Employee"	Opens Employee Update page	Pass	
9	Enter an Employee's ID and a new name and press "Update"	Employee's name is changed within database	Pass	
10	Check post-condition 3		Pass	
11	Enter an employeeID that does not exist in the database	Error message appears	Pass	
12	Enter EmployeeID and press "Remove"	Employee is deleted from database	Pass	
13	Check post-condition 4		Pass	
14	Press "Cancel"	Return to Management screen	Pass	

15	Repeat step 1 and select “All Employees”	Opens page containing all employees	Pass	
16	Select “Cancel”	Return to Management screen		

Table 9

Post-Conditions
- Check if new employee exists within database - Check if there are no duplicates - Check if employee has different name - Check if employee does not exist in database anymore - Employee list displays all employee within the database

5. Check and Edit Trains

Test Case #: 5.1	**Test Case Name:** TrainsTest
System: Railway Management System	**Subsystem:** Train System
Designed by: Trần Quốc Nam	**Design Date:** 16/11/2024
Executed by: Nguyen Van Ngoc Hai	**Execution Date:** 20/11/2024
Short Description: Test the Employee System	

Step	**Action**	**Expected System Response**	**Pass/Fail**	**Comments**
1	Select “Train”	Dropdown with “All Trains”, “Add Trains” and “Search and Update Train” appears	Pass	Verify the button works
2	Select “Add Train”	The system opens Add Train page	Pass	

	from dropdown			
3	Enter Train information and press “Add”	Train added to database	Pass	
4	Check post-condition 1		Pass	
5	Repeat step 3 but with the same Train ID	Receive error message	Pass	
6	Check post-condition 2		Pass	
7	Enter Route information and press “Add route”	Route added to database	Pass	
8	Check post-condition 3		Pass	
9	Select “Cancel”	Closes page	Pass	
10	Repeat step 1 and select “Search and update Train”	Opens Train Update page	Pass	
11	Enter an Train’s ID and a new brand and press “Update”	Train’s brand is changed within database	Pass	
12	Check post-condition 4		Pass	
13	Enter station ID, location and time	Station has new time	Pass	
14	Check post-condition 5		Pass	
11	Enter an TrainID that does not exist in the database	Error message appears	Pass	
12	Enter TrainID and press “Remove”	Train is deleted from database	Pass	
13	Check post-condition 6		Pass	
14	Press “Cancel”	Return to Management screen	Pass	
15	Repeat step 1 and select “All Trains”	Opens page containing all trains	Pass	
16	Select “Cancel”	Return to Management screen		

Table 10

Post-Conditions
- Check if new train exists within database
- Check if there are no duplicates
- Check if route is added to database
- Check if station has changed time in database
- Check if train has different brand
- Check if train does not exist in database anymore

- Train list displays all employee within the database

6. Manage Stations

Test Case #: 6.1	**Test Case Name:** StationTest
System: Railway Management System	**Subsystem:** Railway
Designed by: Trần Quốc Nam	**Design Date:** 12/11/2024
Executed by: Nguyen Van Ngoc Hai	**Execution Date:** 18/11/2024
Short Description: Test the Station system	

Step	Action	Expected System Response	Pass/Fail	Comments
1	Select “Stations”	Dropdown with “All Stations” and “Search and Update stations” appears	Pass	Verify the button works
2	Select “All Stations” from dropdown	The system opens station list	Pass	Ensure that you can open passenger list
3	Check post-condition 1		Pass	
4	Select “Cancel”	Return to Management screen	Pass	
5	Repeat step 1 and select “Search and Update Station”	Opens station management screen	Pass	
6	Enter StationID and a new Name	Update the Station with the StationID with the	Pass	

	and press "Update"	new name within the database		
7	Check post-condition 2		Pass	
8	Enter an TrainID that does not exist in the database and press "Update"	Error Message	Pass	
9	Press "Cancel"	Return to main menu	Pass	

Table 11

Post-Conditions
- Station list contains all stations from the database - Station has new name

7. Manage Tickets

Test Case #: 7.1	**Test Case Name:**TicketTest
System: Railway Management System	**Subsystem:** Railway
Designed by: Trần Nguyễn Minh Trân	**Design Date:** 20/11/2024
Executed by: Nguyen Van Ngoc Hai	**Execution Date:** 27/11/2024
Short Description: Test the Station system	

Step	**Action**	**Expected System Response**	**Pass/Fail**	**Comments**
1	Select "Tickets"	Dropdown with "Ticket Report" and "Book Tickets" appears	Pass	Verify the button works

2	Select "Ticket Report" from dropdown	The system opens ticket list	Pass	Ensure that you can open passenger list
3	Check post-condition 1		Pass	
4	Select "Cancel"	Return to Management screen	Pass	
5	Repeat step 1 and select "Book Ticket"	Opens ticket booking screen	Pass	
6	Enter TicketID that does not exist and fill out information and press "Book"	System creates a new ticket in the database	Pass	
7	Check post-condition 2		Pass	
8	Enter an TicketID that exist in the database and press "Book"	Error Message	Pass	
9	Press "Cancel"	Return to main menu	Pass	

Table 12

Post-Conditions
- Ticket list contains all tickets from the database - New ticket exists in database

8. <u>Passenger Information</u>

Test Case #: 8.1	**Test Case Name:**PassInfo
System: Railway Management System	**Subsystem:** Railway
Designed by: Trần Nguyễn Minh Trân	**Design Date:** 20/11/2024
Executed by: Nguyen Van Ngoc Hai	**Execution Date:** 27/11/2024
Short Description: Test the Station system	

Step	Action	Expected System Response	Pass/Fail	Comments
1	Select “User”	Dropdown with “User information” appears	Pass	Verify the button works
2	Select “User Information” from dropdown	The system opens Passenger screen	Pass	Ensure that you can open passenger list
3	Enter PassengerID and fill out information with a different name and press “Update”	System updates the passenger with the PassengerID with the new name	Pass	
4	Check post-condition 1		Pass	
5	Repeat step 3 but with a passengerID that does not exist within the database	Error Message	Pass	
6	Press “Cancel”	Return to main menu	Pass	

Table 13

Post-Conditions
- Passenger has a different name within database

3. Coding demo

Login page: This is the place where users will input their credentials to connect to the railway management system.

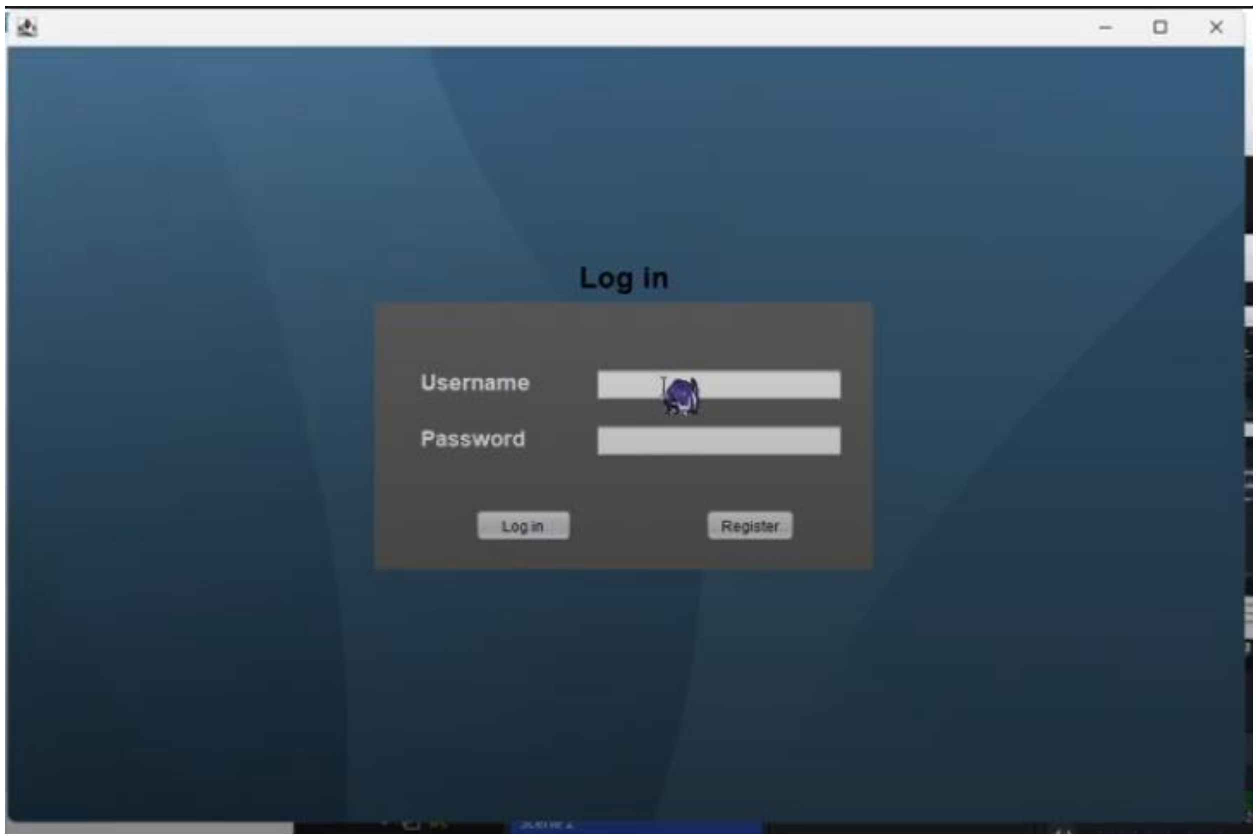
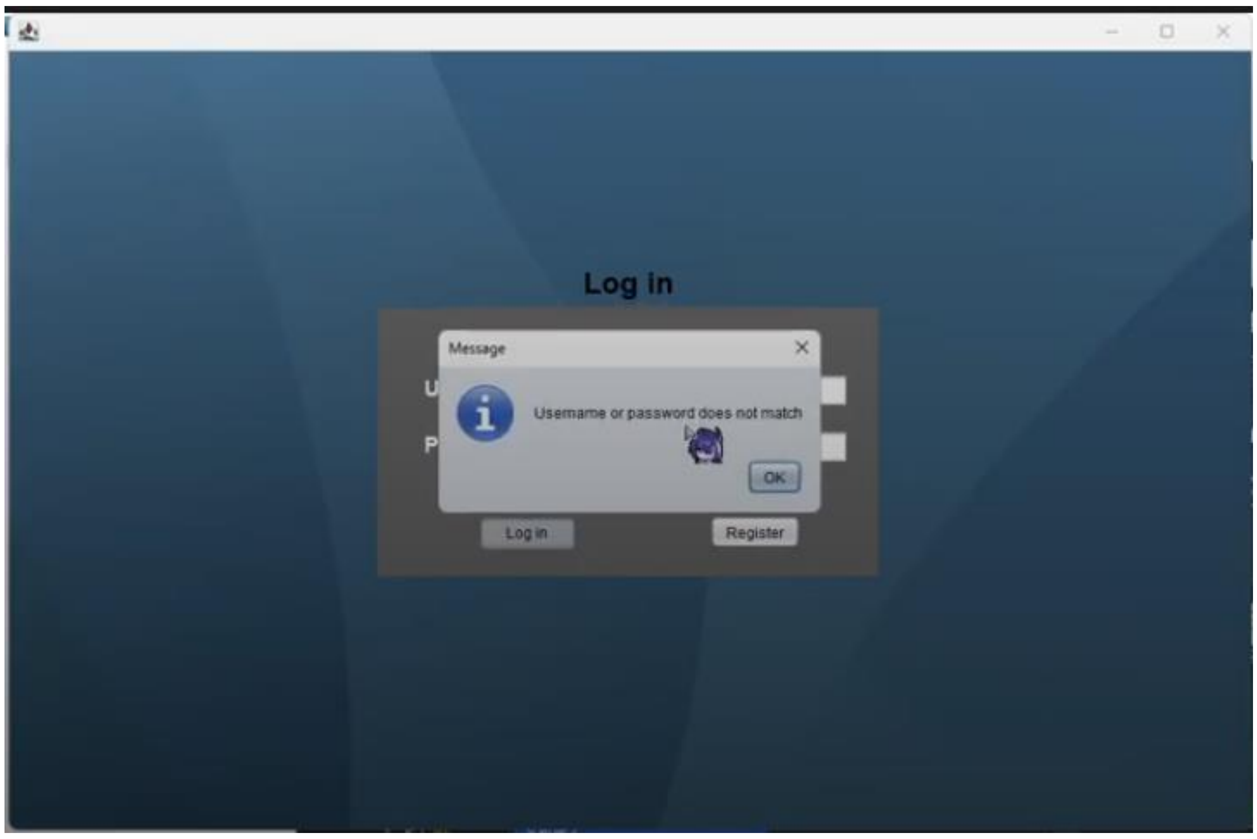


Figure 13: Login Page



Username or password does not match

Figure 14: Login Page notification when input failed

Registration page: This is the place where users will create new accounts to login to the railway management system

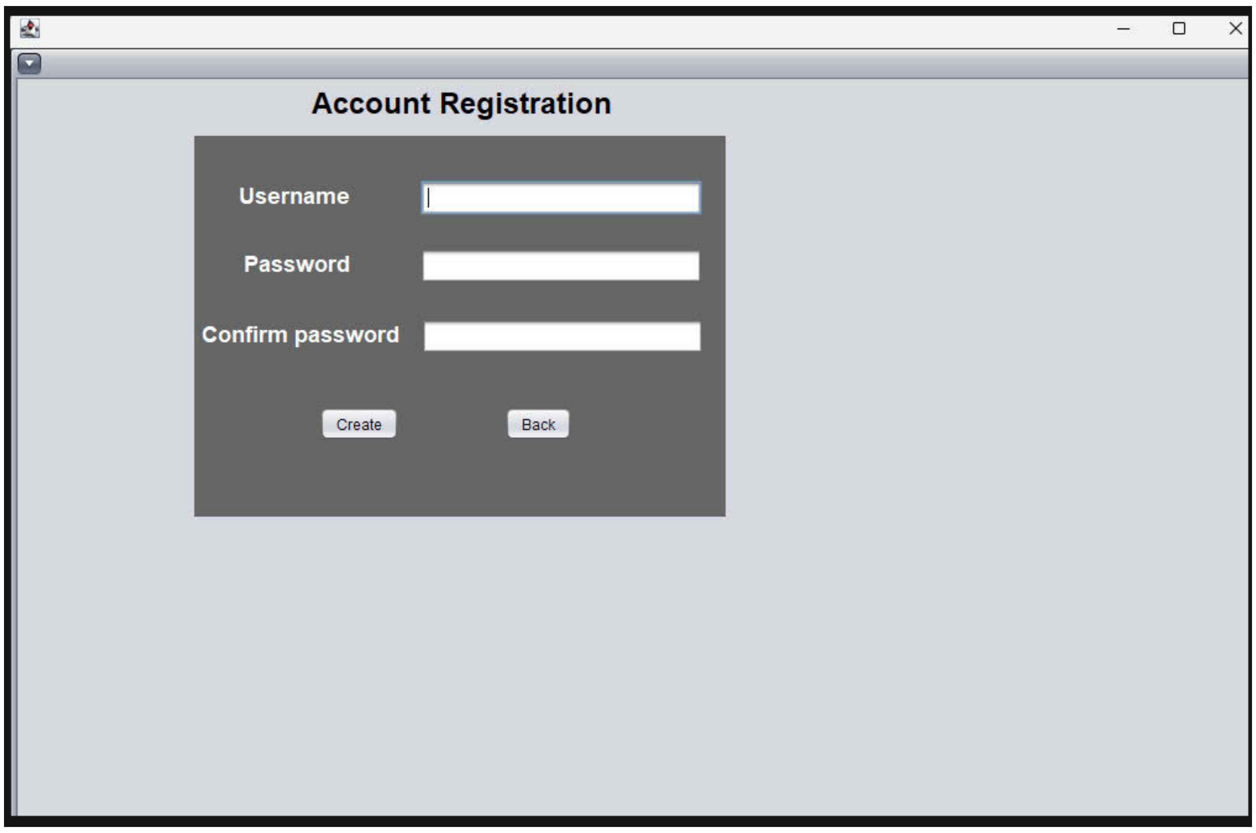


Figure 15: Register new accounts

Homepage: After login and can access to the application, the homepage will show with some menu button on the top left of the screen

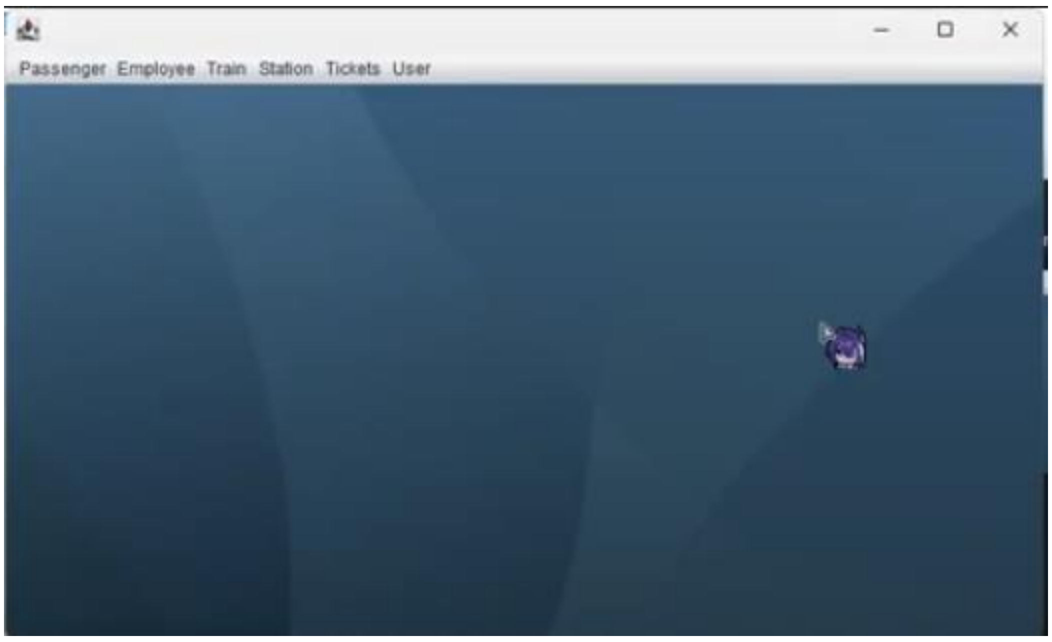


Figure 16: Homepage after login successfully

Passenger menu: The menu shows all kinds of methods related to Passengers

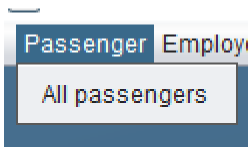
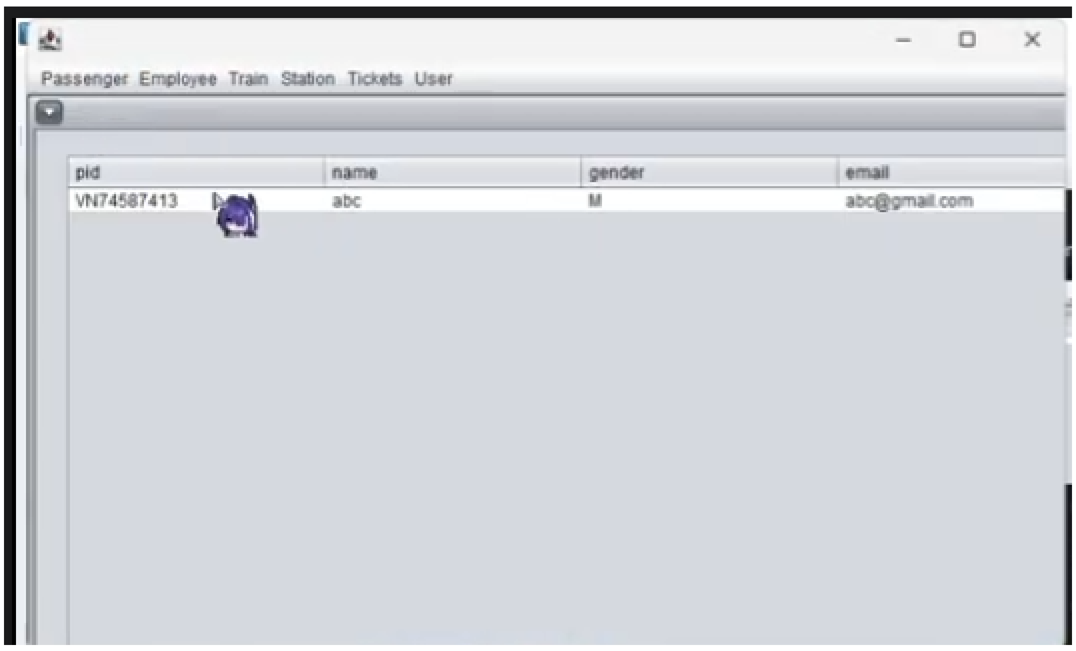


Figure 17: Passenger Menu

All Passenger: The place to show all valid information of passengers in railway system who has registered (including pid, name, gender, email, phone)



pid	name	gender	email
VN74587413	abc	M	abc@gmail.com

Figure 18: View all passengers

Employee menu: The menu shows all kinds of methods related to Employees (including Add employee, search and update employee, all employees)

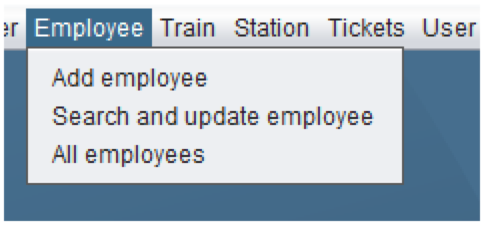


Figure 19: Employee Menu

Add Employees: The place to create new employees' information and store it in the railway system

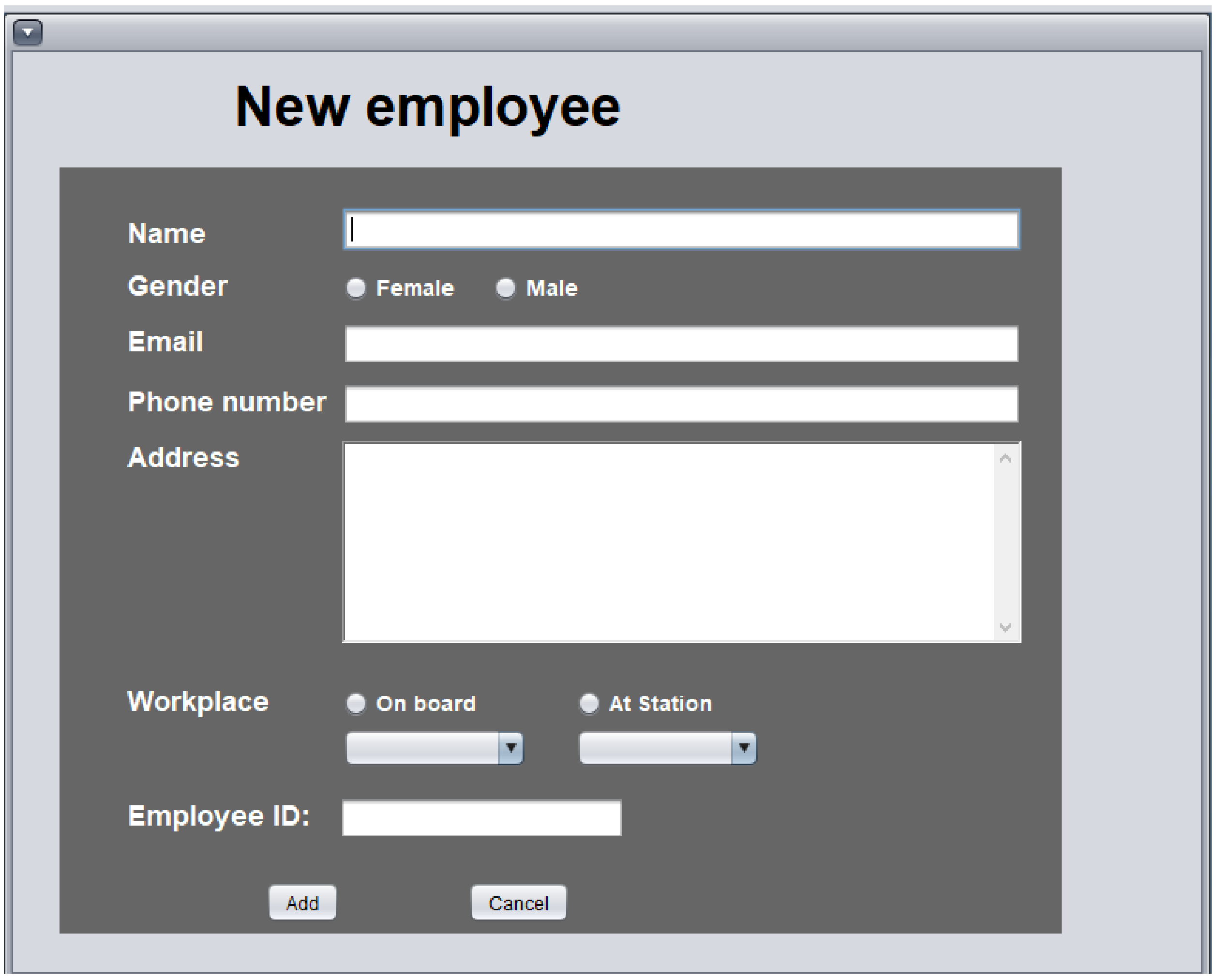


Figure 20: View Add employee

Search and update employees: The place to search information of employees and change information if needed

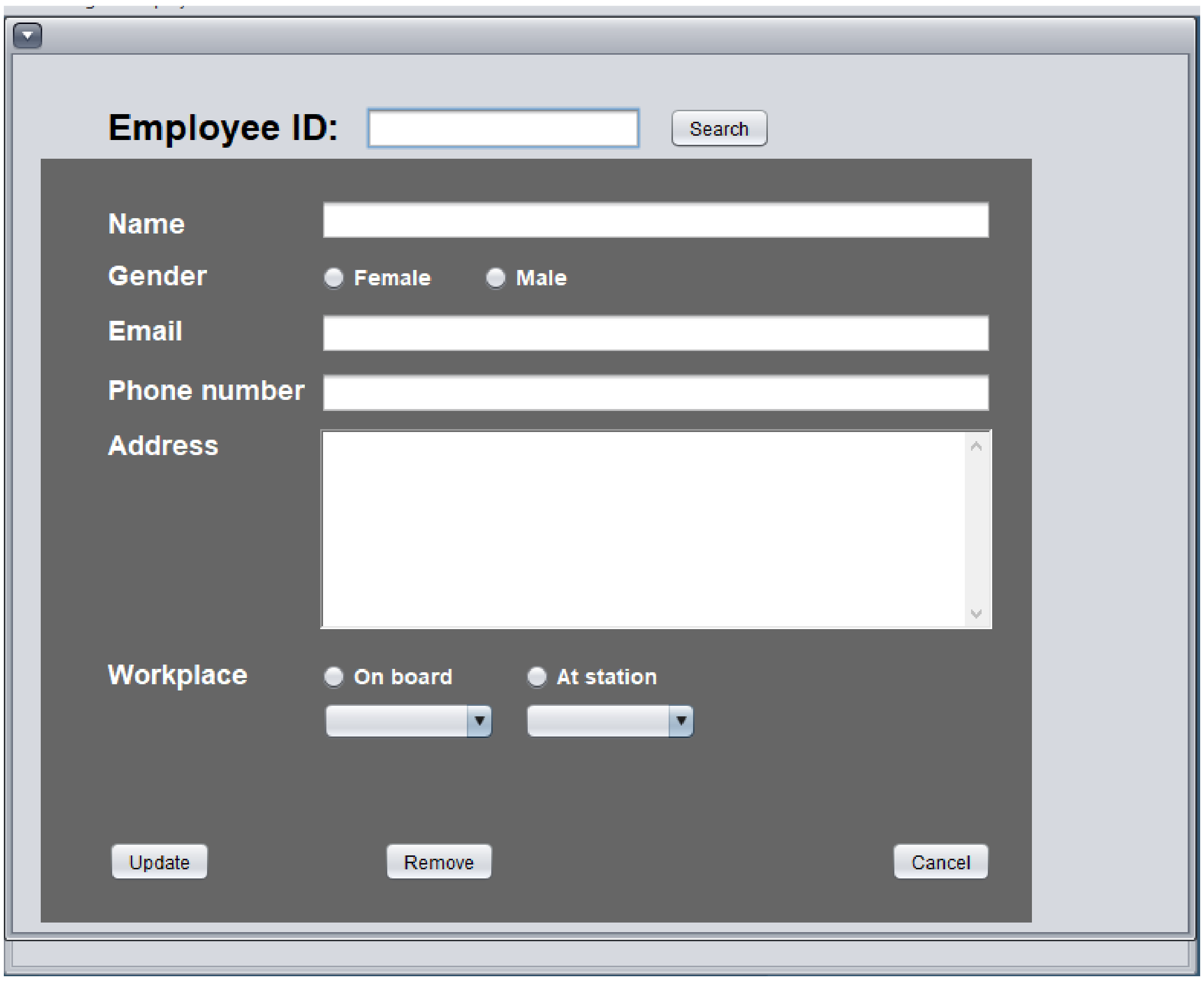


Figure 21: View Search and update employee

All employees: This is the place show all information of all employees who has been registered (including Eid, Name, Email, Phone, Address, Gender, TrainID or StationID where they work at)

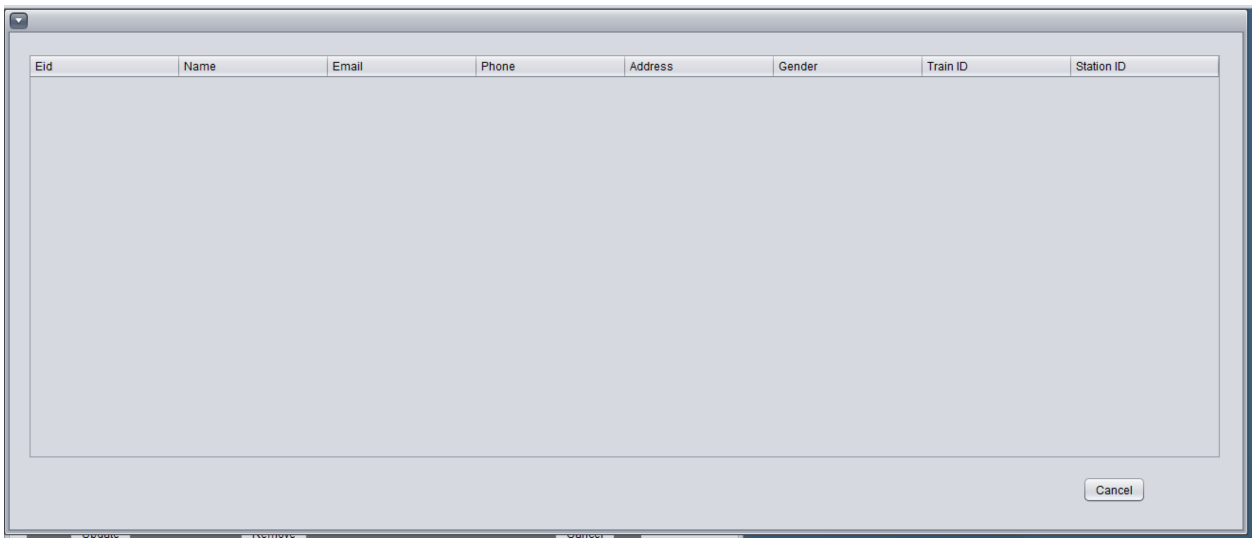


Figure 22: View All employees

Train menu: This is the place where users will create new accounts to login to the railway management system

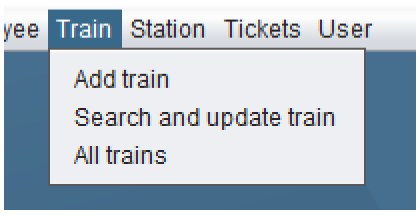
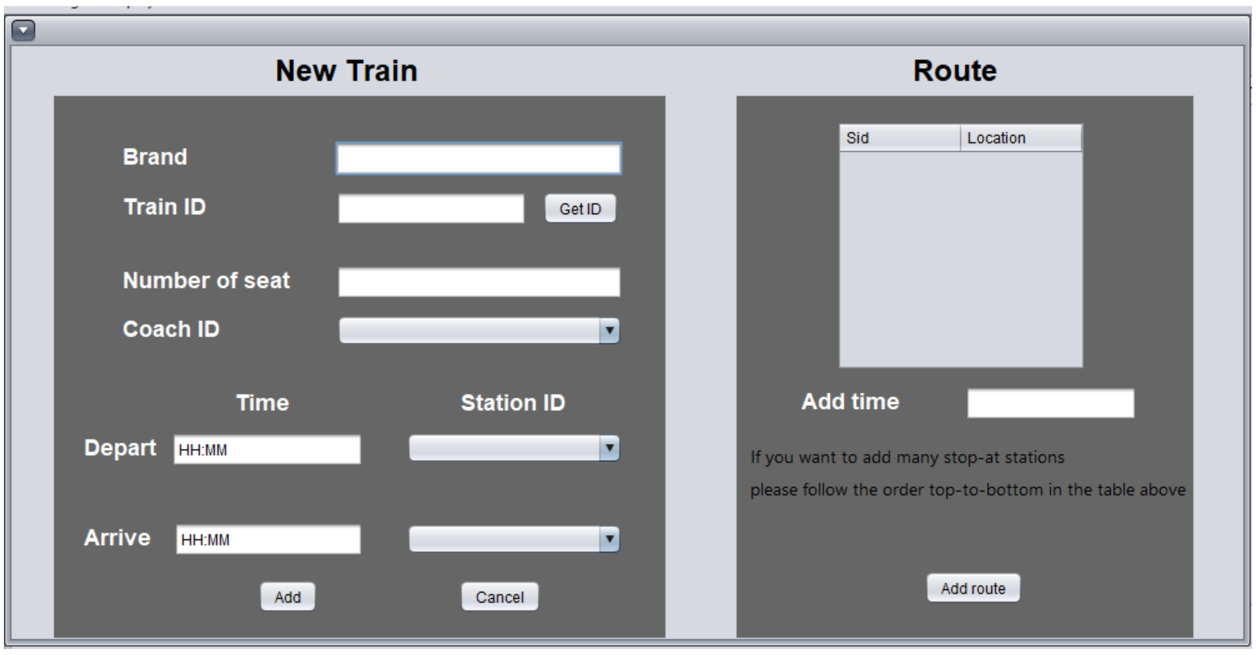


Figure 23: Train Menu

Add train: The page provides information about when a new train is created with some information about the train, the place from beginning to the destination and their routes when they run via some station.



If you want to add many stop-at stations
please follow the order top-to-bottom in the table above

Figure 24: View Add train

View search and update train: This place will take the information of the train based on the ID to find the information, then user can change the information of the train to make it suitable for the schedule.

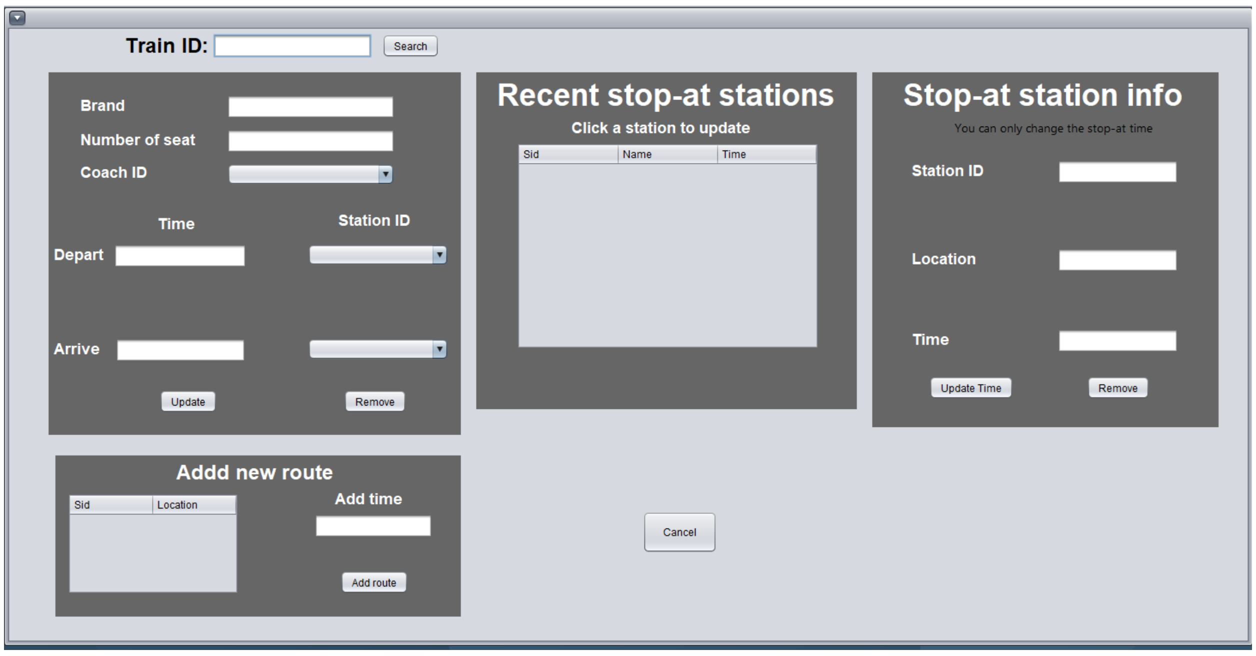


Figure 25: View search and update train

View all trains: Basically, the page shows the information of the trains that have been registered and show all the information of them

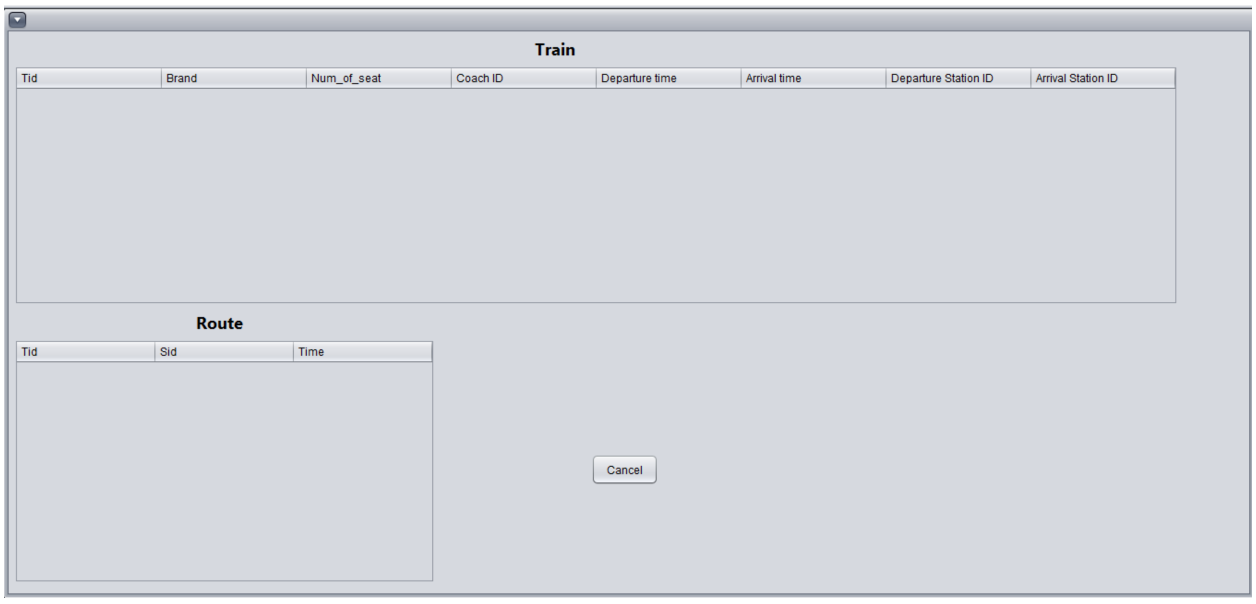


Figure 26: View All trains

Station menu: This is the place to manage the station information of railway system

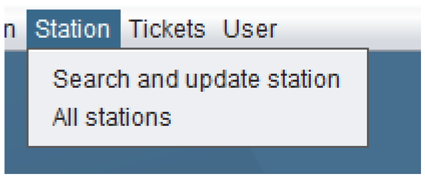


Figure 27: Station Menu

View Search and update station: Using StationID to find out the information of the station and then can change or update the new name or location of the station if needed.

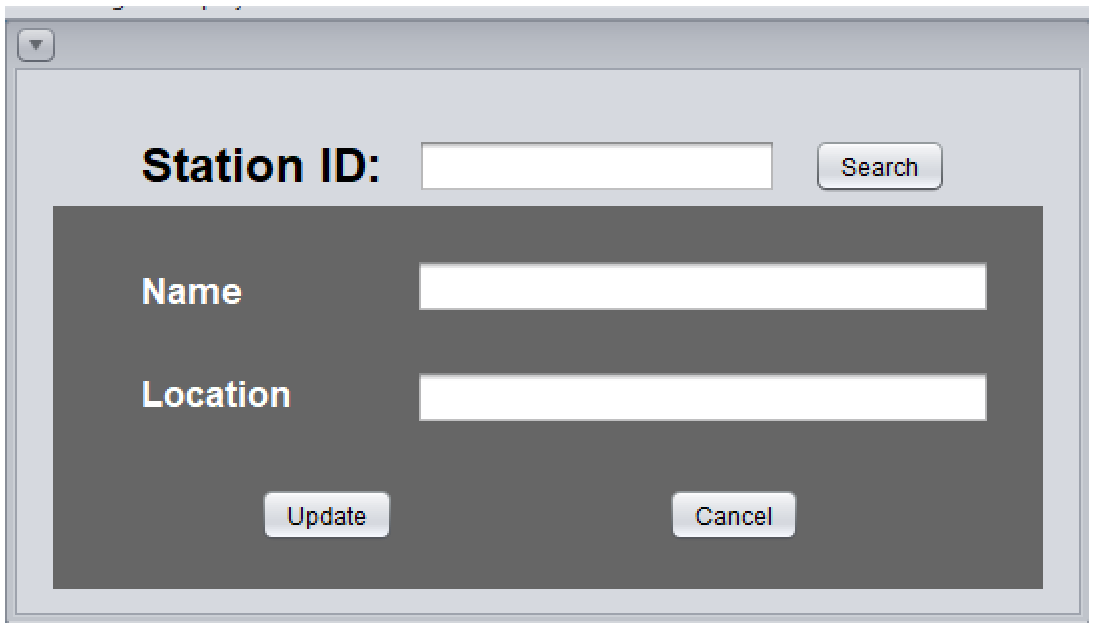


Figure 28: View Search and update station

View all stations: This is the place shows all information of the stations in the railway system

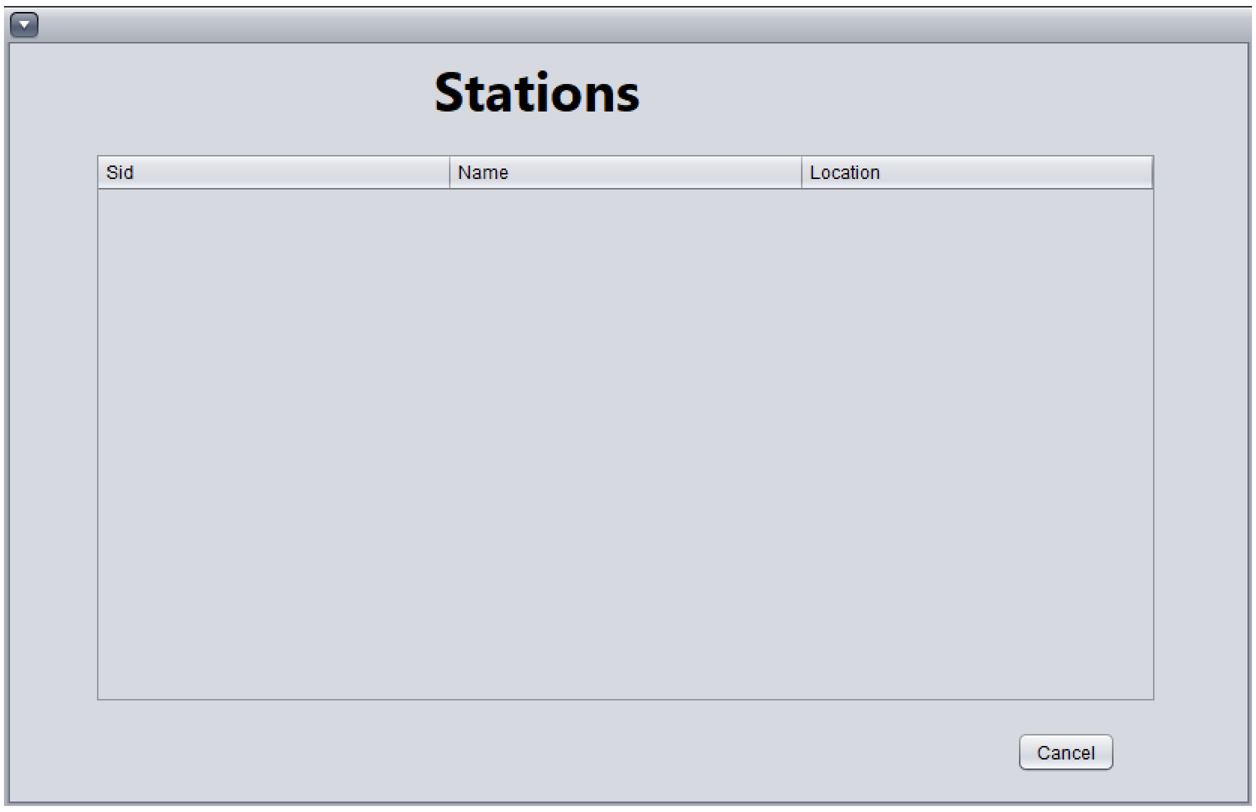


Figure 29: View All stations

Ticket menu: Contains ticket report and book ticket, this menu will show all the ways when passengers begin to register their journey via tickets

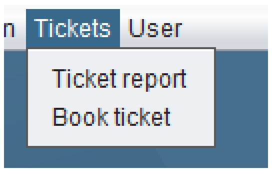
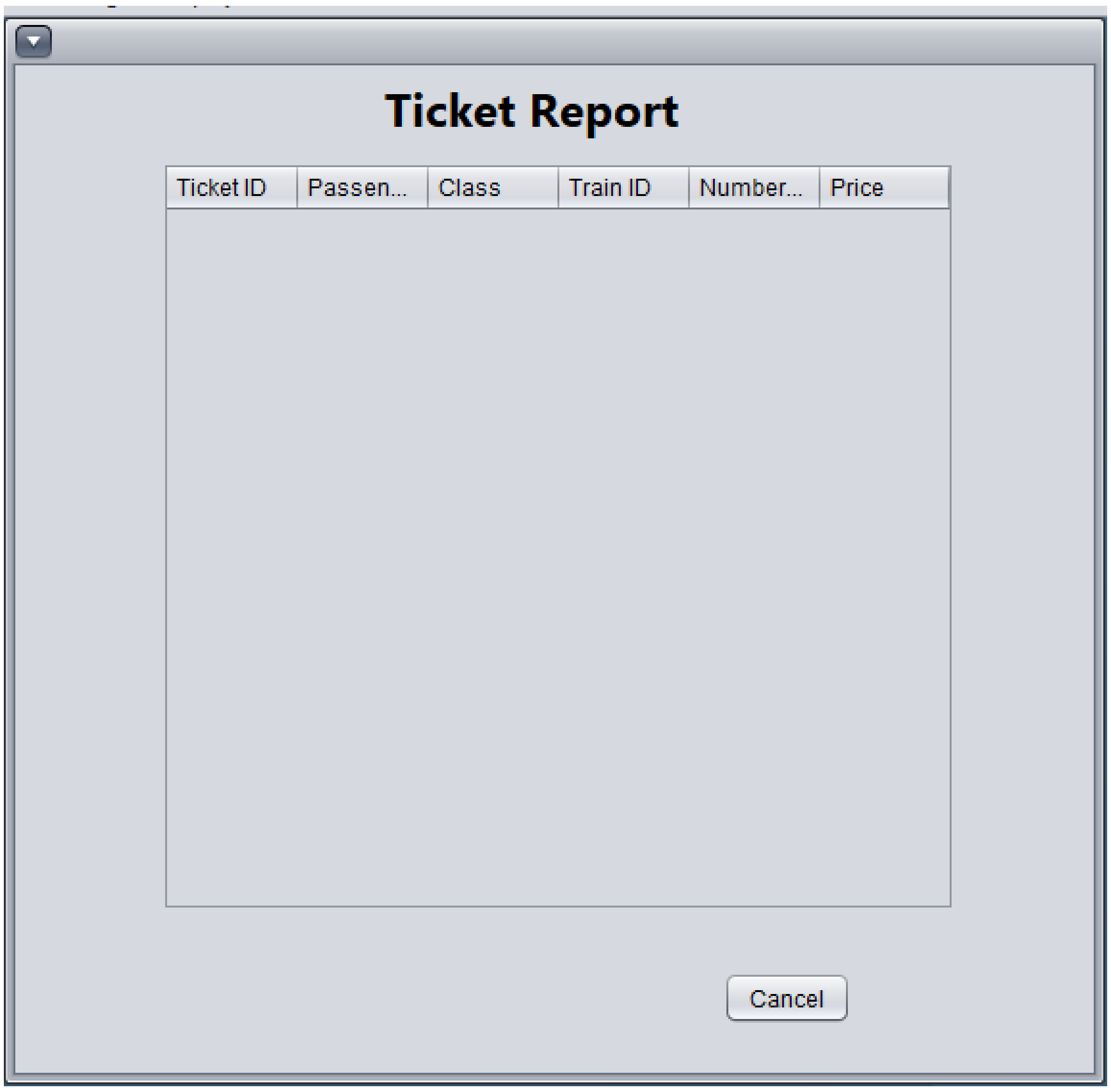


Figure 30: Tickets Menu

View ticket report: This one stores the information of the ticket that has been created by the book ticket, the information of passengers from where they want to go will be stored and show all in here

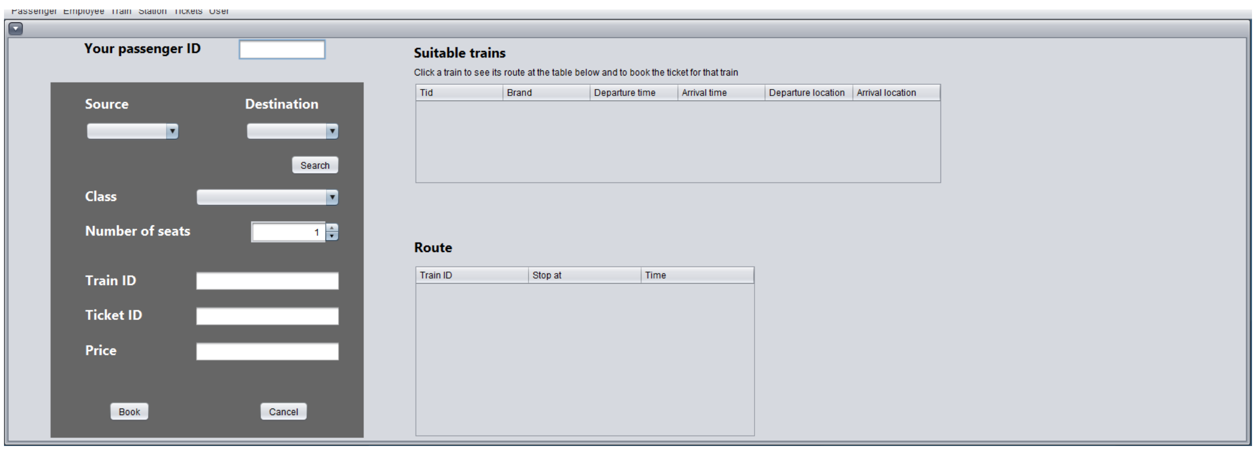


Ticket Report

Ticket ID	Passen...	Class	Train ID	Number...	Price

Figure 31: View Ticket report

Book ticket: The booking place for passengers when they want to use the railway service, the booking will provide some information that the train has about the journey and kind of services inside the train (Number of seats, class)



Your passenger ID

Source | Destination

Class

Number of seats: 1

Train ID

Ticket ID

Price

Suitable trains

Click a train to see its route at the table below and to book the ticket for that train

Tid	Brand	Departure time	Arrival time	Departure location	Arrival location

Route

Train ID	Stop at	Time

Figure 32: Book ticket

User Menu: This the same with passenger menu with contain passenger information

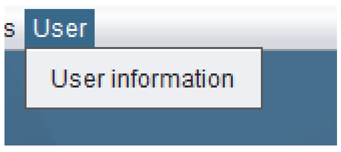


Figure 33: User Menu

View user information: This works as passenger search and update when the information of passengers will be checked and then manager can adjust this information in the update button if needed

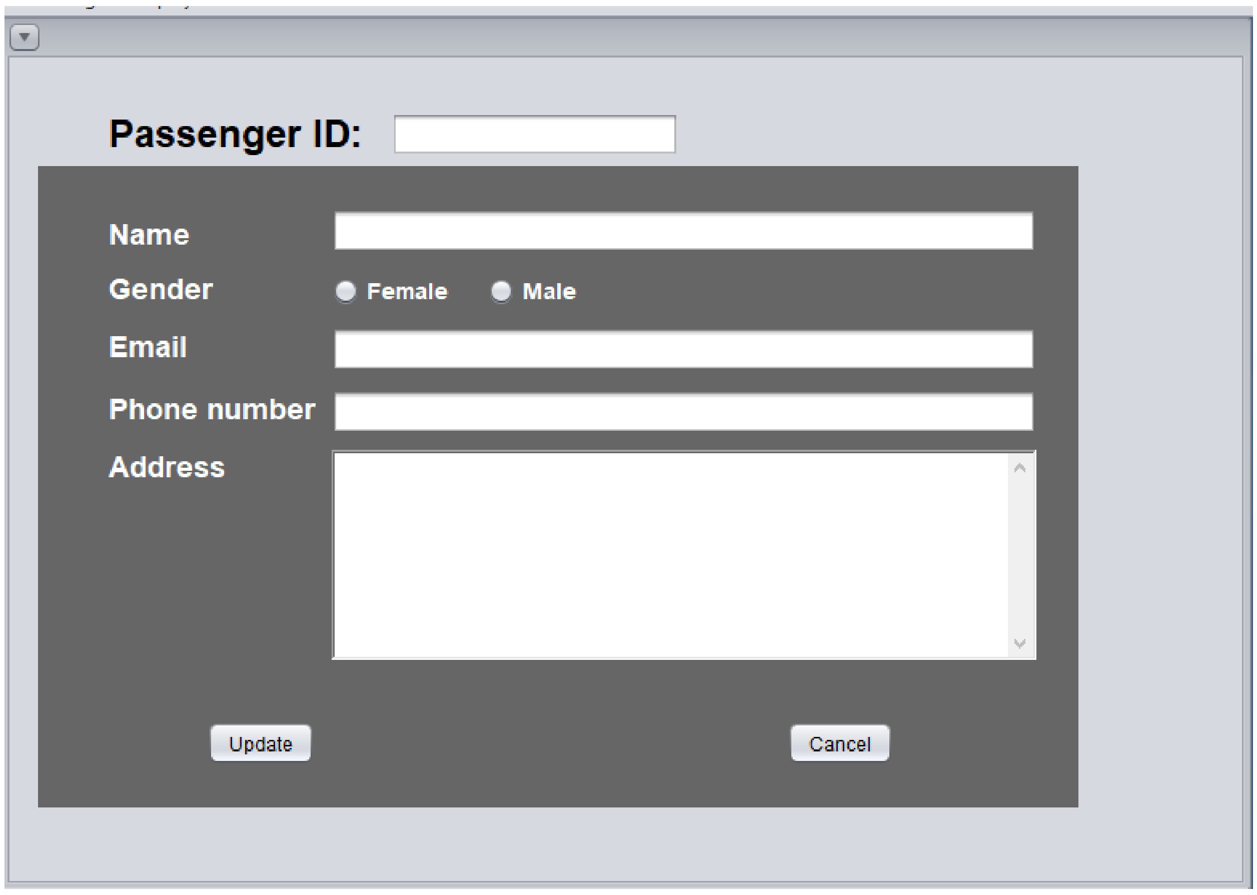


Figure 34: View User Information

CHAPTER 6

DISCUSSION & EVALUATION

Other challenges for the railway system project include disagreements among members, unexpected turnover, data security concern, problems with the coding technique, and overruns of current price estimates. The group concerned the promotion of openness and communications, rescheduling work to balance labor, putting in place strong encryption and security of data. The coding hurdles were cleared through extensive research and thorough testing, while monetary reserve plus financial appraisal helped to clear off most of the budgetary worries. In addressing these, the project has been kept right on target, and it's still going ahead with a decent outcome expected.

Challenges	Effect	Solutions
Lack of consensus among members	When members cannot collaborate and harmonize. The process of the project will be slow or may interrupt. Consequently, lack of agreement can easily cause delays, misunderstandings, and inefficiencies that may disrupt the whole timeline and affect the success of the project.	Create connections among all the team members to keep the environment positive, which will have a positive impact on the organization. Be transparent, reliable, and promise-keeping to align.
Members leaving unexpectedly	Members leaving during the project of building will delay the project timing and create an imbalance in the workload distribution that could have been distributed among them. This creates an impact on the results	Reorganize the workload, giving most priority to the most important, and prepare the remaining members to take charge of the major elements of the project at hand. Consider temporary replacement or resource reallocation to

	with delays in the project process.	supplement for keeping the momentum of the work.
Data security issues	Managing data securely is essential, and the risk of attacks or breaches can harm the project’s progress and the organization’s reputation. Security vulnerabilities could lead to data leaks, system downtime, or legal complications, all of which could undermine the project's success and damage stakeholder trust.	Implement robust data encryption and privacy measures, conduct regular security audits, and follow industry standards to safeguard user and operational data. This will help build trust, avoid financial losses, and ensure compliance with legal regulations.
Coding technique problem	Members may lack the necessary knowledge or skills required for efficient project development, leading to delays and difficulties with system integration. Technical capability can be low, reducing the ability to meet deadlines, build workable systems, and ensure seamless integration among different components of the projects.	Best practices research, additional training when necessary, and heavy testing of the components for compatibility with existing technologies will optimize results and further advance efficiency in development.
Over-budget cost	The project must be stopped or delayed because the budget is higher than the funds of the plans.	Financially analyze the costs of the project and review them with the inclusion of unexpected costs. Adjust the scope if needed, perform only high-priority tasks, and anywhere possible, include strategies of cost saving to stay within the budget of the plan.

CHAPTER 7

CONCLUSION

Technology is constantly growing and plays a vital role in the development of many sectors, including transportation. The development of the railway management system in our course of Software Engineering not only gave us great insight into technical solutions but also taught us the role of efficient system design to ease operations. While digital systems changed the way data was managed and processed, the human element in system design assured us that solutions were user-friendly, effective, and suitable for real-world applications.

Throughout this project, our team gained valuable experience in teamwork, problem-solving, and project management. We logically divided the tasks, managed time effectively, and ensured steady progress that improved our collaboration and understanding of the software development lifecycle. From the conceptualization of the system to the implementation as a functional desktop application, we navigated through various challenges, including designing a structured database and integrating it seamlessly with the programming logic.

The project allowed us to take a critical look at aspects such as analyzing user requirements, translating them into system features, and optimizing code for performance. Besides that, working on a railway management system helped us understand the intricacies of designing systems with functionality and reliability in mind—qualities that any real-world application should possess.

This experience has deepened our understanding of database structures and their importance in system efficiency while honing our programming skills. The knowledge and practical expertise gained through this project will certainly be useful in tackling challenges in the future of software engineering.

REFERENCES

1.https://www.pace.edu.vn/tin-kho-tri-thuc/so-do-gantt

2.https://www.lucidchart.com/pages/uml-use-case-diagram

3.https://www.lucidchart.com/pages/uml-sequence-diagram#:~:text=A%20sequence%20diagram%20is%20a,to%20document%20an%20existing%20process.

4.https://viblo.asia/p/tim-hieu-ve-cach-thiet-ke-class-diagram-L4x5xLyY5BM

5. https://www.geeksforgeeks.org/java/

6. https://www.geeksforgeeks.org/mysql-tutorial/

7.Draw Diagram: https://draw.io/

Note: we mark [number] in some parts of the project for reference purposes.