CAPSTONE PROJECT

RAILWAY TICKETING SYSTEM

PROPOSED BY:
M JAYANTH NAYAK (23ME01034)
RUDRAPRIYA R CHOUDHARY (23ME02011)
SHAVYA SINGH (23ME02012)
YASHIKA AGARWAL (23ME02041)
YUVRAJ SINGH (23ME02014)

2023

CS1P001
Introduction to Programming & Data Structures

INTRODUCTION

Introducing the Railway Ticket Booking System, a project brought to you by our team of five dedicated (and slightly sleep-deprived) individuals. While we may not promise you a laugh riot, we can assure you of a reliable and efficient ticket booking system that aims to simplify the train ticket reservation process. Let's embark on this journey together – destination: smooth and hassle-free ticket booking. All aboard!

MOTIVATION

The motivation behind the Railway Ticket Booking System is to make train ticket booking convenient, save time for passengers, enhance user experience, modernize transportation, increase accessibility, generate revenue, improve customer satisfaction, and support government initiatives. It can also create job opportunities and scale to serve more routes and passengers.

IMPORTANT HIGHLIGHTS OF OUR PROJECT

1. User Sign-up and Sign-in

Our code begins by offering a user-friendly sign-up and sign-in system. Users can create an account with their name and a unique PIN. This feature adds a layer of personalization and security to the system.

2. Train Information

Our project maintains a database of available trains, including information like train number, name, source, destination, departure, arrival times, and fare for different classes (SL, 3AC, 2AC, 1AC). This database is crucial for users to make informed decisions when booking tickets.

3. Ticket Booking

Users can book tickets by providing essential details such as their name, gender, age, mobile number, train number, class, source, destination, and departure date. The system ensures data integrity by verifying train numbers and destination details.

4. Electronic Reservation Slip

Our code generates an electronic reservation slip for each ticket booked, displaying all relevant information, including the passenger's details, ticket number, class, source, destination, date, departure, arrival times, and fare. This slip serves as a confirmation for passengers.

5. Ticket Cancellation

Our project allows users to cancel their tickets. Users can specify the ticket number to be cancelled, and the system marks that ticket as cancelled. This feature provides flexibility to users who might change their travel plans.

6. Viewing and Managing Tickets

Users can view their booked tickets, including details like passenger information, source, destination, class, and fare. This feature enables users to keep track of their reservations.

7. Robust Input Validation

Our code includes input validation for critical user inputs, such as gender, age, train numbers, classes, mobile numbers, source, and destination. It helps prevent incorrect or malicious data entry.

CONCEPTS LEARNED DURING PROJECT

1. Data Structures

Our project introduced the concept of data structures, including the use of structs to organize and store data effectively. In this system, three essential data structures are employed: `struct username` for user account information, `struct passenger` for storing passenger details, and `struct train` for managing train-related data. These structures play a critical role in the project, enabling data organization and manipulation.

2. Arrays and Looping

Arrays are employed for storing multiple instances of data, such as passenger information and train details. Looping constructs (for and while) are used to iterate through arrays and perform repetitive tasks efficiently.

3. User Authentication

One of the key concepts explored in this project is user authentication. Our code allows users to create accounts, including their names and PINs. Users can then sign in with their credentials. This concept demonstrates the importance of user identity management and access control.

4. Database Management

Although not fully implemented in this project, the concept of database management is introduced. Maintaining a database of available trains and user accounts is essential for managing reservations and ensuring data integrity. The project demonstrates the need for organized data storage.

5. Data Persistence

Although not fully integrated into this project, the idea of data persistence is introduced. It involves storing user accounts, reservations, and train information in files or databases, enabling data to persist beyond the execution of the program. This concept plays a vital role in ensuring effective long-term data management.

6. Error Handling

Error handling is a critical concept emphasized by the project. Handling errors and notifying users of incorrect inputs or system issues is essential for user satisfaction. The project highlights the importance of providing meaningful error messages.

7. Team coding

We needed to find platforms that allow us to code in real-time and verify if the modifications become immediately visible to our fellow team members. We defined roles within the development team, including developers, testers, project managers, and documentation specialists

8. Software Development Process

The project underscores the iterative nature of software development. It involves initial planning, coding, testing, and potential future enhancements or bug fixes.

AREAS OF IMPORVEMENT

1. Data Persistence

Issue: The current implementation does not provide data persistence between program runs. User accounts and reservations are lost when the program terminates.

Improvement: Implement a data storage mechanism, such as file I/O or a database, to persist user accounts and reservation information. This would allow users to retrieve their bookings across multiple sessions.

2. Enhanced User Authentication

Issue: The current authentication relies solely on usernames and PINs, which may not provide the desired level of security.

Improvement: Implement more secure authentication methods, such as password hashing and salting, to protect user account information. Additionally, consider multi-factor authentication for added security.

3. Database Integration

Issue: The code uses arrays to store train and passenger information. This may become inefficient for managing a large volume of data.

Improvement: Integrate a database management system to efficiently store and retrieve train schedules and passenger details. This would improve scalability and data organization.

4. User Interface Enhancement

Issue: The text-based user interface is functional but lacks modern design and user-friendliness.

Improvement: Consider developing a graphical user interface (UI) to provide a more intuitive and visually appealing experience for users. UI libraries like Qt or JavaFX can be used for this purpose.

5. Error Handling and User Feedback

Issue: While the code includes some error handling, there is room for improvement in providing informative error messages to users.

Improvement: Enhance error handling by providing descriptive error messages that guide users on how to correct their inputs. Clear and helpful feedback can improve the user experience.

6. Modular Code Structure

Issue: The current code is structured into a single main function, which can make it challenging to maintain and expand.

Improvement: Refactor the code into modular functions and classes, adhering to software engineering principles. This will improve code readability and maintainability.

7. Code Documentation

Issue: The code lacks comprehensive documentation, making it difficult for other developers to understand and contribute to the project.

Improvement: Provide clear and detailed documentation for functions, classes, and the project as a whole. This will help future developers work on the code more effectively.

FUTURE SCOPE

1. Learning Platform

The project can serve as an educational tool for those learning programming and software development. It can be used to teach programming and system design principles.

2. Customization and Expansion

The project can be extended and customized to cater to specific requirements. By enhancing its functionality, it can be adapted for small-scale railway or bus services, or for local events and bookings. This customization can be valuable for local businesses and event organizers.

3. Integration with Existing Systems

It can be integrated with existing systems of smaller railway operators or travel agencies that may not have the resources to implement a complex booking system. This can serve as a cost-effective solution for those seeking a basic reservation platform.

CONTRIBUTIONS

1. Login and Password Verification- Rudra:

Rudra implemented the login and password verification system using C programming. This likely involved incorporating secure practices for handling user credentials, such as using functions like `strcmp` for string comparison and ensuring data integrity.

2. Insert Train- Jayanth:

Jayanth implemented the code for inserting train details into the system. This involved designing and implementing the logic to add new train details to the system.

3. Booking of Trains, Viewing Tickets and Debugging- Shavya:

Shavya played a key role in developing the booking system in C. This likely involved creating functions for passengers to book tickets, providing functionality to view booked tickets and meticulously debugging the code to identify and resolve any errors that arose during the development process.

4. Struct Defining, Viewing Trains and adding Colour to Text- Yashika:

Yashika contributed to the project by defining essential data structures in C. This includes creating structures like `username`, `passenger`, and `train` to organize and store relevant information. Additionally, Yashika enhanced the user interface and visual experience using C. She implemented features for users to view available trains and added colour to text output for improved readability, likely using ANSI escape codes for colour.

5. Cancel Ticket- Yuvraj:

Yuvraj focused on implementing the ticket cancellation system using C. This would include creating functions to allow passengers to cancel their booked tickets, ensuring the proper updating of data structures, and handling any potential edge cases.

GITHUB LINK

https://github.com/Shavya28/train.git