NORTHERN UNIVERSITY

B A N G L A D E S H

Knowledge for Innovation and Change

Department of CSE

Software Development - I

Course Code: CSE 1290

Final Project Report

Online Ticket Reservation System (Swift Book)

Submitted to

Jannat Rosul Nisha

Lecturer

Department of CSE

Northern University Bangladesh

Submission Date: April26, 2025

Project Details

Project Title	Online Ticket Reservation System (Swift Book)
Project Application	C++ Terminal-Based Application

Group Name: Tech Hunters

Group Members:

Student ID	Name	Signature
41230301367	Sanzid Zaman	Sanzid
41230301525	Md. Rakibul Islam	Rakib
41230301887	Fahim Ibney Hafiz	Fahim
41230301564	Joy Biswas	Joy
41230201274	Md. Atikur Rahman	Atikur

Table of Contents

1.1 Introduction

The Online Ticket Booking System is a C++ command-line application skillfully designed to revolutionize the process of bus, train, and flight ticketing to meet the rising demand for easy, accessible, and convenient travel solutions. With digital transformation reshaping industries in a world where everything is now digital, traditional ticketing systems based on time-consuming physical mechanisms such as paper records, on-location reservations, and phone bookings are still haunting users and operators with inefficiencies. Not only do these issues inconvenience customers but also burden transport operators with operating expenses and reduced scalability.

The Online Ticket Reservation System solves these problems by providing a streamlined, electronic system that allows users to manage their travel needs with ease and accuracy. Based on object-oriented programming (OOP) principles in C++, the system is designed with modularity, code reusability, and maintainability in mind, hence being robust and flexible. It accommodates two primary user roles: end-users (passengers) who can register, login, see routes, book or cancel tickets, and see booking details, and administrators who oversee route management and system performance. By using text files to hold information, the system gains data persistence without the cost or complexity of a typical database, offering the ideal solution for small- and medium-sized travel agencies, regional transport companies, or educational institutions that need real-world software solutions.

This effort was fueled by the need to bridge the divide between older manual ticketing processes and newer digital expectations. It aligns with global tendencies toward automation and user empowerment, with a scalable platform that can keep pace with evolving technology. This report provides a comprehensive review of the motivation behind the system, its purpose, its attributes, its architecture, its deployment, its advantages, its limitations, and its future potential, giving insight into its path of evolution and its paradigm-shifting impact on travel booking process.

1.1.1 Objectives

The system aims to:

- Provide secure user registration and login functionality.
- Enable users to view available routes and seats for bus, train, and airline travel.
- Facilitate ticket booking and cancellation with real-time seat updates.
- Allow users to view their ticket details for reference.
- Ensure data persistence using text file storage, suitable for small-scale deployments.

1.2 Motivation

Inefficiencies in manual ticket reservation systems, prevalent in bus terminals, train terminals, and travel agencies, impose significant challenges on customers as well as staff. Customers are frustrated by wait times due to lengthy queues, limited counter hours, and slow manual processing, often exacerbated by peak travel seasons. Staff are exposed to time-consuming, error-prone tasks such as handwritten records or manual seat reservations, leading to mistakes in fare calculations or booking conflicts. These issues increase operational costs and diminish customer trust. Inspired by the success of digital solutions in e-commerce, online banking, and ride-sharing apps, this project aims to transform travel booking. The Online Ticket Reservation System streamlines reservations, ensuring faster, more accurate transactions and 24/7 accessibility. By integrating bus, rail, and flight travel into a single C++ terminal-based system, it caters to diverse travel needs, reduces administrative overhead, and enhances customer satisfaction. The project aligns with global digitalization trends, allowing small to medium-sized operators to digitalize operations and offer seamless, reliable services.

1.3Project Features Description

The Online Ticket Reservation System offers a robust set of features for users and administrators, implemented through a menu-driven terminal interface:

- **User Registration and Login:** Users create accounts with a username and password, stored in users.txt. Login validates credentials to ensure authorized access.
- **Route Viewing:** Displays available routes for bus, train, and airline travel, including origin, destination, time, available seats, and price, retrieved from routes.txt.
- **Ticket Booking:** Users select a route and seat number. The system verifies seat availability, updates routes.txt (decrementing available seats), and records the booking in bookings.txt.
- **Ticket Cancellation:** Users cancel bookings by entering a booking ID. The system updates routes.txt (incrementing available seats) and removes the booking from bookings.txt.
- **Ticket Details:** Users can view their bookings, showing route details, seat number, and price.

● **Admin Route Management:** Administrators (flagged in users.txt) can add routes (bus, train, or airline) with details like origin, destination, time, seats, and price, stored in routes.txt. They can also view all routes.

● **File-Based Storage:** Uses text files (users.txt, routes.txt, bookings.txt) in a CSV-like format for persistent, lightweight storage. These features make the system intuitive and efficient for managing travel reservations across multiple transport modes.

1.3.1 System Design

The system is designed with a modular architecture, comprising:

● User Management: Handles registration and login, storing credentials in users.txt.

● Route Management: Allows admins to add routes, stored in routes.txt.

● Booking Management: Manages bookings and cancellations, stored in bookings.txt.

● Interface: A menu-driven terminal interface for user and admin interactions.

1.3.2 Use Case Diagram

User Flow:

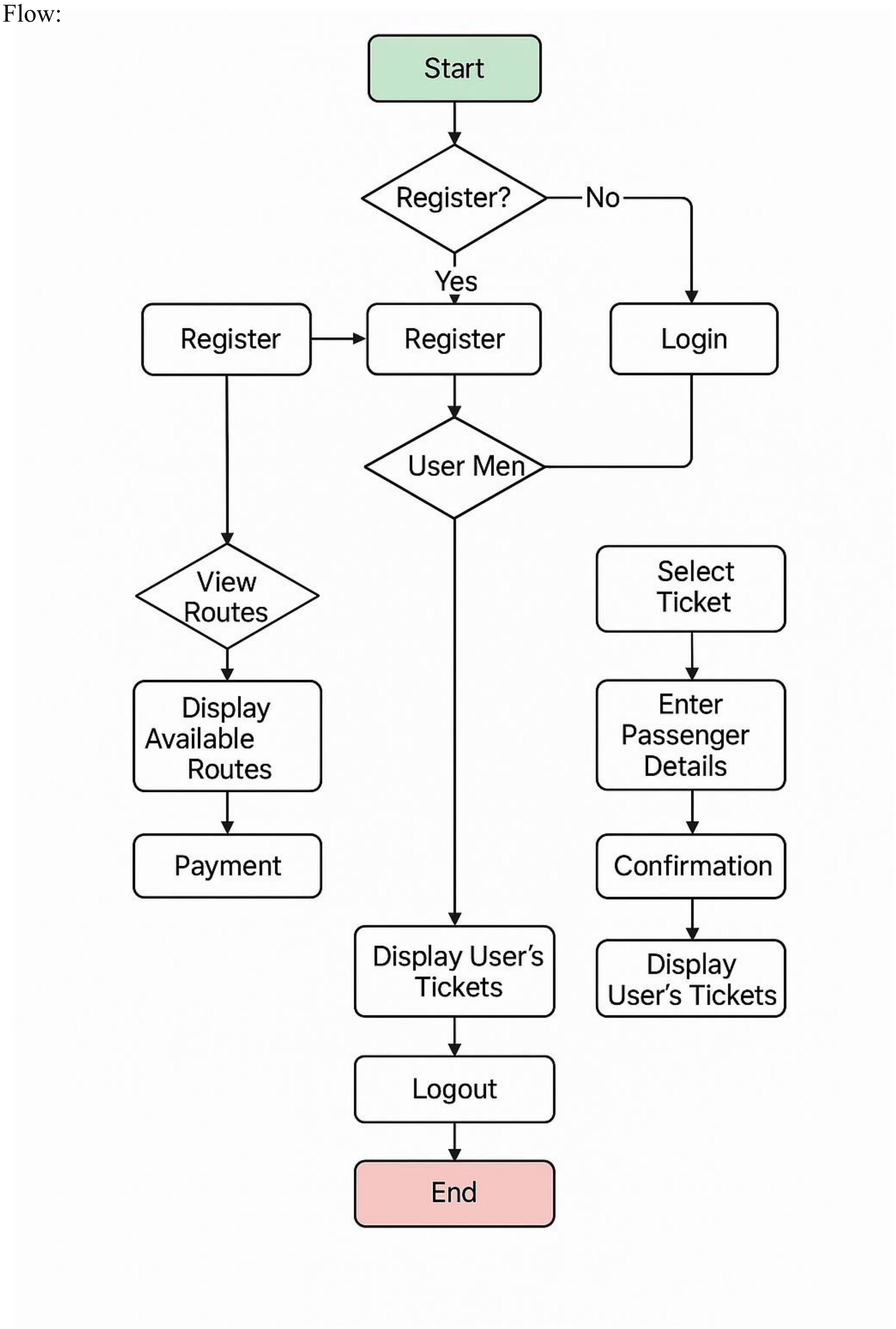


Fig 1.1 User Flow diagram

Admin Flow:

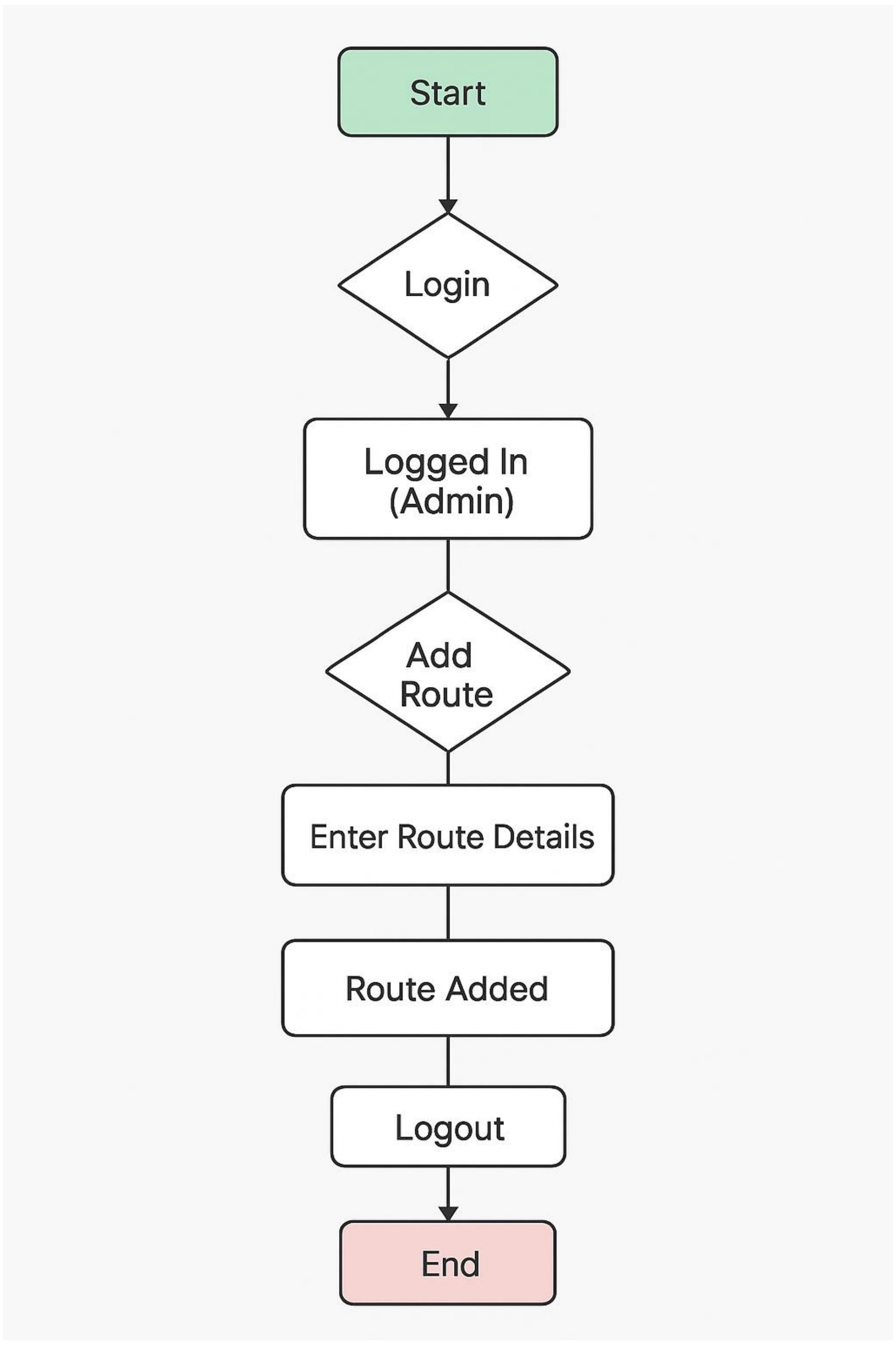


Fig 1.2 Admin Flow Diagram

1.3.3 File Structure

● users.txt: Stores username, password, admin status (e.g., Sanzid, pass123,0).

● routes.txt: Stores route details (e.g., 1, Bus, Dhaka, Chittagong, 14:30, 50, 45, 1000.0).

● bookings.txt: Stores bookings (e.g., 1, Sanzid, 1, 5).

1.3.4 Code Expiations

The C++ program is structured around three entities and modular functions:

1.3.4.1 Structures

● User: Stores username, password, and isAdmin (boolean).

● Route: Stores id, type (Bus/Train/Airline), origin, destination, time, total_seats, available_seats, and price.

● Booking: Stores bookingId, username, routeId, and seatNumber.

1.3.4.2 Key Functions

● registerUser(): Prompts for credentials and saves to users.txt.

● loginUser(): Validates credentials by reading users.txt.

● addRoute(): Admin inputs route details, saved to routes.txt.

● viewRoutes(): Displays routes from routes.txt.

● bookTicket(): Books a seat, updates routes.txt and bookings.txt.

● cancelTicket(): Cancels a booking, updates both files.

● viewTicket(): Shows user’s bookings with route details.

1.3.4.3 File Handling

● Uses fstream for reading/writing text files in CSV-like format.

● Functions like loadRoutes() and loadBookings() parse files into vectors.

1.3.4.4 Main Loop

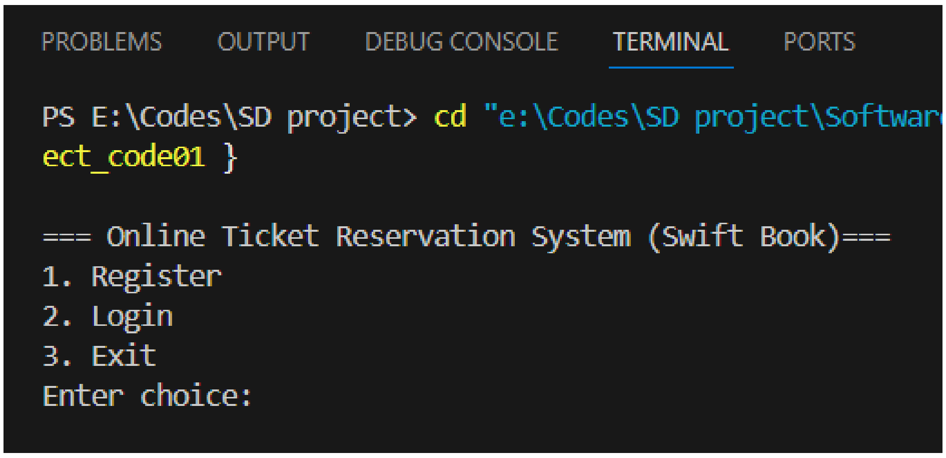
● Offers a menu for registration, login, or exit.

● Post-login, users access options to view routes, book, cancel, or view tickets; admins can add or view routes.

1.3.5 Screenshots

The terminal-based system's interfaces are shown below as sample console outputs:

1.3.5.1 Main Menu

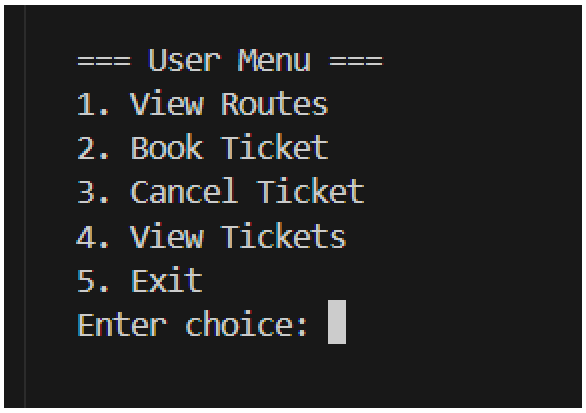


```
PS E:\Codes\SD project> cd "e:\Codes\SD project\Softwar
ect_code01 }

=== Online Ticket Reservation System (Swift Book)===
1. Register
2. Login
3. Exit
Enter choice:
```

Fig 1.3 Main Menu

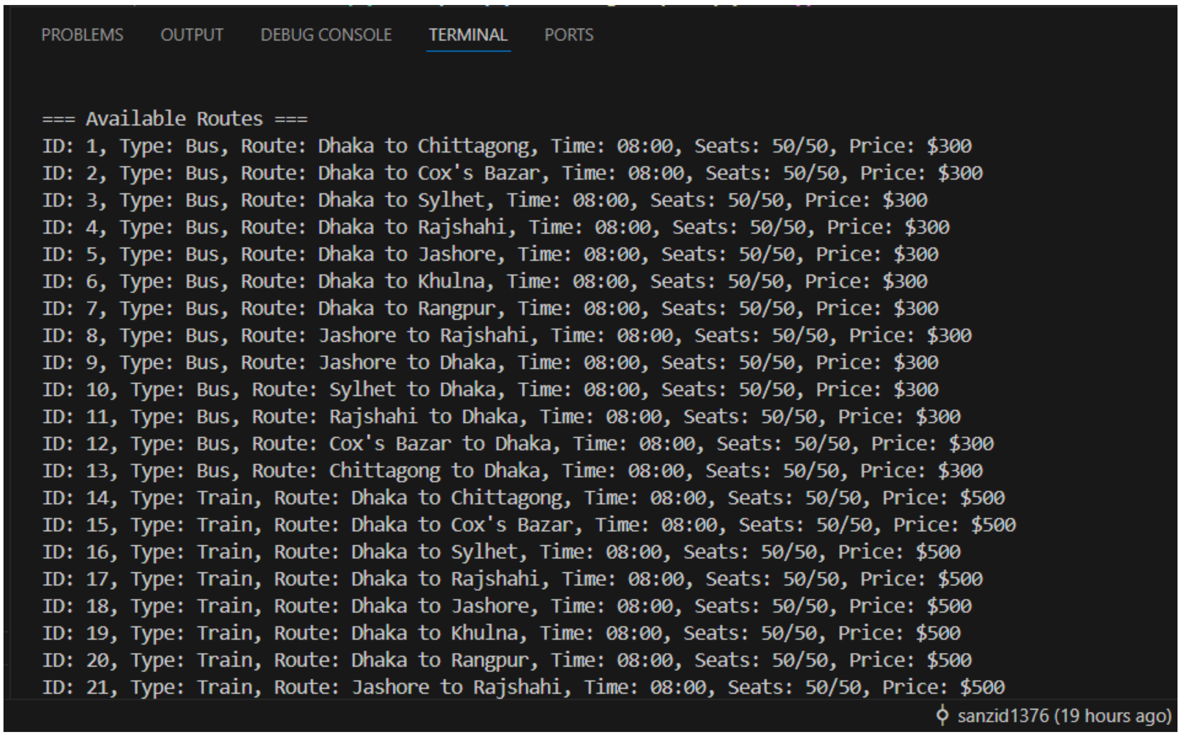
1.3.5.2 User Menu



```
=== User Menu ===
1. View Routes
2. Book Ticket
3. Cancel Ticket
4. View Tickets
5. Exit
Enter choice:
```

Fig 1.4 User Menu

1.3.5.3 Sample Route Display



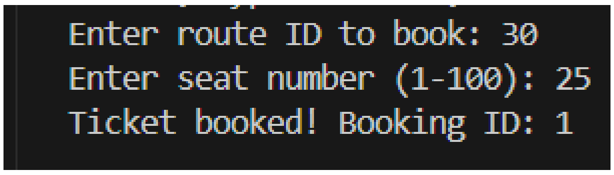
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
=== Available Routes ===
ID: 1, Type: Bus, Route: Dhaka to Chittagong, Time: 08:00, Seats: 50/50, Price: $300
ID: 2, Type: Bus, Route: Dhaka to Cox's Bazar, Time: 08:00, Seats: 50/50, Price: $300
ID: 3, Type: Bus, Route: Dhaka to Sylhet, Time: 08:00, Seats: 50/50, Price: $300
ID: 4, Type: Bus, Route: Dhaka to Rajshahi, Time: 08:00, Seats: 50/50, Price: $300
ID: 5, Type: Bus, Route: Dhaka to Jashore, Time: 08:00, Seats: 50/50, Price: $300
ID: 6, Type: Bus, Route: Dhaka to Khulna, Time: 08:00, Seats: 50/50, Price: $300
ID: 7, Type: Bus, Route: Dhaka to Rangpur, Time: 08:00, Seats: 50/50, Price: $300
ID: 8, Type: Bus, Route: Jashore to Rajshahi, Time: 08:00, Seats: 50/50, Price: $300
ID: 9, Type: Bus, Route: Jashore to Dhaka, Time: 08:00, Seats: 50/50, Price: $300
ID: 10, Type: Bus, Route: Sylhet to Dhaka, Time: 08:00, Seats: 50/50, Price: $300
ID: 11, Type: Bus, Route: Rajshahi to Dhaka, Time: 08:00, Seats: 50/50, Price: $300
ID: 12, Type: Bus, Route: Cox's Bazar to Dhaka, Time: 08:00, Seats: 50/50, Price: $300
ID: 13, Type: Bus, Route: Chittagong to Dhaka, Time: 08:00, Seats: 50/50, Price: $300
ID: 14, Type: Train, Route: Dhaka to Chittagong, Time: 08:00, Seats: 50/50, Price: $500
ID: 15, Type: Train, Route: Dhaka to Cox's Bazar, Time: 08:00, Seats: 50/50, Price: $500
ID: 16, Type: Train, Route: Dhaka to Sylhet, Time: 08:00, Seats: 50/50, Price: $500
ID: 17, Type: Train, Route: Dhaka to Rajshahi, Time: 08:00, Seats: 50/50, Price: $500
ID: 18, Type: Train, Route: Dhaka to Jashore, Time: 08:00, Seats: 50/50, Price: $500
ID: 19, Type: Train, Route: Dhaka to Khulna, Time: 08:00, Seats: 50/50, Price: $500
ID: 20, Type: Train, Route: Dhaka to Rangpur, Time: 08:00, Seats: 50/50, Price: $500
ID: 21, Type: Train, Route: Jashore to Rajshahi, Time: 08:00, Seats: 50/50, Price: $500
```

sanzid1376 (19 hours ago)

Fig 1.5 Sample Route Display

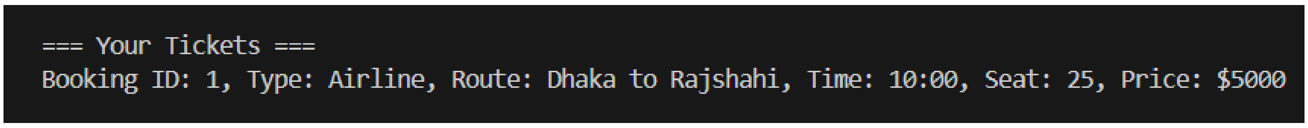
1.3.5.4 Sample of Booking



```
Enter route ID to book: 30
Enter seat number (1-100): 25
Ticket booked! Booking ID: 1
```

Fig 1.6 Sample of Booking

1.3.5.5 Sample Ticket View



```
=== Your Tickets ===
Booking ID: 1, Type: Airline, Route: Dhaka to Rajshahi, Time: 10:00, Seat: 25, Price: $5000
```

Fig 1.7 Sample Ticket View

1.4 Advantage

● Efficiency: Automates booking and cancellation, reducing wait times and manual effort compared to traditional systems.

● Multi-Transport Support: Integrates bus, train, and airline reservations in one platform, catering to diverse needs.

● Accessibility: Runs on any device with a C++ compiler, requiring no internet since data is stored locally.

● Data Persistence: File handling ensures data is saved between sessions without a database.

● Cost-Effective: Minimal infrastructure (no server or database) makes it ideal for small transport companies.

● Error Reduction: Input validation (e.g., seat availability, route IDs) minimizes booking errors.

1.5 Drawbacks

● Terminal-Based Interface: Lacks a GUI, which may be less intuitive for non-technical users compared to web/mobile apps.

● Limited Scalability: File-based storage is slow for large datasets, limiting use in high-volume scenarios.

● No Payment Integration: Requires offline payments, reducing convenience.

● Basic Security: Plain-text passwords in users.txt pose a security risk if files are accessed.

● No Notifications: Lacks email/SMS confirmations, limiting user engagement.

● Single-Instance Limitation: Designed for local use, not supporting concurrent multi-user access.

1.6 Conclusion

The Online Ticket Reservation System effectively automates ticket booking for bus, train, and airline travel, showcasing the capabilities of a lightweight C++ terminal-based application. By replacing manual processes, it significantly improves efficiency, minimizes errors, and supports multiple transport modes. Its modular design and file-based storage ensure suitability for small-scale deployments, while its comprehensive implementation—covering user management, route handling, and booking processes—demonstrates robust functionality.

1.6.1 Assessment of the Project

The project successfully meets its objective of providing a functional and reliable ticket reservation system. Its strengths lie in its simplicity, modularity, and ability to handle diverse travel modes within a single platform. The use of C++ ensures high performance, and the file-based storage system is adequate for small-scale operations. However, the absence of a graphical user interface (GUI) limits user accessibility, and the system's scalability is constrained by its reliance on local file storage. Despite these challenges, the project serves as a proof of concept for automating travel reservations, with clear potential for expansion.

1.6.2 Lessons Learned

Developing this system highlighted the importance of modular design in ensuring maintainability and extensibility. The project underscored the need for careful data management to prevent errors in booking and user records. Additionally, the lack of a GUI revealed the trade-off between development speed and user experience, prompting a deeper appreciation for user-centric design. Working with file-based storage also emphasized the limitations of non-relational data management for larger systems, encouraging exploration of database integration in future iterations.

1.6.3 Future Work

To enhance the system, several improvements can be prioritized. First, integrating a GUI using frameworks like Qt or SDL would improve user accessibility and experience. Second, transitioning from file-based storage to a relational database (e.g., SQLite or MySQL) would enhance scalability and data integrity. Third, adding network capabilities to support online bookings and real-time updates would align the system with modern travel platforms. Finally, implementing advanced features like payment gateways, multi-language support, and mobile app compatibility could further elevate its utility and reach.