

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**C CASE STUDY:**

**AN EXPERIENTIAL LEARNING REPORT ON THE TOPIC:**

**TRAIN TICKET BOOKING**

**UNDER THE SUPERVISION OF: Mr. SANDEEP PADESUR**

**SUBMITTED BY: GROUP: CVR JAVA 05 SEMESTER: 6th BRANCH: C S E**

**STUDENT DETAILS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **SL NO** | **NAME** | **CRANES REGD NO.** | **EMAIL ID & CONTACT NO** |
| **1** | **RAJAT KUMAR SAHU** | **CL2025010601898745** | [**rajatrks15@gmail.com**](mailto:rajatrks15@gmail.com) **9777625154** |
| **2** | **PURNENDU SENAPATI** | **CL2025010601951352** | [**purnendusenapati29@gmail.com**](mailto:purnendusenapati29@gmail.com) **7735963625** |
| **3** | **RAJEET KUMAR** | **CL20250106019056114** | [**2201020947@cgu-odisha.ac.in**](mailto:2201020947@cgu-odisha.ac.in) **7352042610** |
| **4** | **ARUMBHA PAIKRAY** | **CL2025010601926798** | [**arumbhapaikray123@gmail.com**](mailto:arumbhapaikray123@gmail.com) **8260565253** |

**ACKNOWLEDGEMENT:**

I sincerely express my gratitude to everyone who has contributed to the successful completion of my project, **"Train Ticket Booking System."**

I extend my heartfelt thanks to my mentor, [Mentor's Name], for their invaluable guidance, constant support, and insightful suggestions throughout the project. Their expertise and encouragement have been instrumental in shaping this project.

I also appreciate my institution, [Institution Name], for providing the necessary resources and a conducive learning environment. Special thanks to my friends and family for their encouragement and motivation.

Finally, I acknowledge the importance of teamwork, research, and dedication in completing this project successfully.

**CONTENT:**

|  |  |
| --- | --- |
| **S.NO** | **TITLE** |
| **1** | **ABSTRACT** |
| **2** | **OBJECTIVE** |
| **3** | **PROBLEM STATEMENT** |
| **4** | **TECHNOLGY STACK** |
| **5** | **SYSTEM MODULES AND FEATURES** |
| **6** | **SOURCE CODE** |
| **7** | **EXPECTED OUTPUT** |
| **8** | **TIMELINES ANDMILESTONES** |
| **9** | **FUTURE ENHANCEMENT** |
| **10** | **CONCLUSION** |

**ABSTRACT:**

The **Train Ticket Booking System** is a C-based software application designed to automate and streamline the process of booking train tickets. This system allows users to check train availability, book tickets, cancel reservations, and view booking details efficiently.

The project aims to replace the traditional manual booking system with a computerized solution, minimizing errors and enhancing user experience. It features a structured menu-driven interface, user authentication, real-time seat availability management, and secure data handling using file operations. The system ensures efficient data storage and retrieval while maintaining simplicity and ease of use.

By implementing this system, users can experience a seamless and fast ticket booking process, improving accessibility and efficiency in railway ticket management.

**OBJECTIVE:**

The **Train Ticket Booking System** is developed using C programming to automate and enhance the traditional train ticket reservation process. The main objectives of this project include:

**1. Automation of the Ticket Booking Process**

* Replace the manual booking process with a digital system for faster and more efficient reservations.
* Allow users to book, modify, and cancel tickets with minimal effort.

**2. User-Friendly Interface**

* Develop a simple and intuitive menu-driven interface to facilitate smooth navigation.
* Ensure accessibility for both technical and non-technical users.

**3. Train Schedule and Seat Availability Management**

* Maintain an updated database of train schedules, routes, and available seats.
* Enable real-time seat availability checking before booking.

**4. Secure and Structured Data Handling**

* Use file operations in C to securely store user data, booking records, and transaction history.
* Prevent data corruption and unauthorized modifications.

**5. Data Accuracy and Reliability**

* Minimize human errors by automating calculations related to fare, seat availability, and ticket status.
* Provide accurate and consistent booking information.

**6. Authentication and User Management**

* Implement a basic login system to allow registered users to access their bookings.
* Prevent unauthorized users from making changes to the booking database.

**7. Efficient Searching and Filtering**

* Enable users to search for trains based on departure and arrival stations, travel dates, and train types.
* Provide sorting options based on ticket price, travel duration, and seat class.

**8. Ticket Generation and Confirmation**

* Generate a ticket receipt with passenger details, train information, and booking reference number.
* Allow users to print or save their tickets for future use.

**9. Cancellation and Refund Processing**

* Provide an option for users to cancel tickets and process refunds as per predefined policies.
* Update the seat availability dynamically after cancellations.

**10. Cost and Time Efficiency**

* Reduce operational costs associated with paper-based ticketing systems.
* Save passengers' time by eliminating the need for physical ticket booking at railway stations.

**11. Scalability and Future Enhancements**

* Design the system in a modular way, allowing future upgrades such as online payment integration and database connectivity.
* Ensure flexibility to accommodate more features, including seat selection and real-time train tracking.

By achieving these objectives, the **Train Ticket Booking System** enhances efficiency, reduces errors, and provides a convenient and secure ticketing experience for railway passengers.

4o

**PROBLEM STATEMENT:**

The conventional train ticket booking system, which relies on manual processes, presents several challenges for both passengers and railway authorities. Passengers often face difficulties such as long waiting times, limited access to train schedules, and errors in ticket reservations. Similarly, railway authorities struggle with managing passenger records, seat availability, and cancellations due to the lack of an automated system.

Some key issues in the manual booking system include:

1. **Time-Consuming Process** – Passengers must visit railway stations and stand in long queues to book tickets, leading to delays and inconvenience.
2. **Lack of Real-Time Seat Availability** – Manual systems do not provide instant updates on seat availability, leading to overbooking or confusion.
3. **Human Errors** – Mistakes in ticket allocation, cancellations, or schedule management often occur due to manual record-keeping.
4. **Data Mismanagement** – Passenger details, booking history, and train schedules may not be well-organized, leading to inefficiencies.
5. **Security Concerns** – Without proper authentication, unauthorized access or tampering with booking records can occur.
6. **Lack of Accessibility** – Passengers may not have an easy way to check train availability, fares, or book tickets remotely.
7. **Difficulty in Handling Cancellations and Modifications** – Refunds and modifications in manual systems take time and are prone to errors.
8. **Paper-Based System Issues** – Traditional paper-based ticketing is not eco-friendly and can lead to ticket loss or damage.

To overcome these problems, the **Train Ticket Booking System using C** is developed to provide a computerized solution for ticket reservations, cancellations, and train schedule management. This system will:

* **Automate the Booking Process** – Allow passengers to search for trains, book tickets, and manage their reservations through a digital interface.
* **Provide Real-Time Seat Availability Updates** – Ensure users get accurate seat availability information before booking.
* **Ensure Data Security and Accuracy** – Store booking records securely using file handling, reducing errors and unauthorized access.
* **Improve User Convenience** – Provide an easy-to-use interface that enables passengers to book tickets from anywhere.
* **Enhance Efficiency** – Reduce paperwork and speed up ticket booking and cancellation procedures.

By implementing this system, railway ticket management will become more streamlined, efficient, and user-friendly, significantly improving the passenger experience.

**TECHNOLOGY STACK:**

The **Train Ticket Booking System** is developed using the **C programming language**, utilizing various tools and libraries to ensure efficient functionality, data management, and user interaction. Below is the technology stack used in the project:

**1. Programming Language**

* **C Language** – The core programming language used for developing the entire system, offering efficiency, speed, and portability.

**2. Development Environment & Compiler**

* **GCC (GNU Compiler Collection)** – A widely used compiler for compiling and running C programs.
* **Code::Blocks / Dev-C++ / Turbo C++** – IDEs that can be used for writing and debugging C code.

**3. Data Storage & Management**

* **File Handling in C** – Used for storing train schedules, passenger details, booking records, and transaction history in text or binary files.
* **Structures & Arrays** – Used for managing train details, user data, and ticket information within the program.

**4. User Interface & Interaction**

* **Console-Based UI** – The system operates through a command-line interface (CLI), displaying menus and taking user input via the terminal.
* **Standard Input/Output Functions** – Functions like printf(), scanf(), gets(), and puts() are used for displaying and receiving data.

**5. Security & Authentication**

* **User Authentication** – Password protection and basic user verification using string comparison (strcmp()) for secure access.
* **File Encryption (Optional)** – Simple encryption techniques (e.g., XOR encryption) can be implemented for securing stored data.

**6. Algorithms & Logic Implementation**

* **Sorting & Searching Algorithms** – Used for managing train schedules and finding available trains efficiently.
* **Queue & Linked List (Optional Enhancements)** – Can be implemented for handling ticket waitlists or real-time updates.

**7. Error Handling & Optimization**

* **Validation Checks** – Ensures correct input for train selection, ticket booking, and cancellations.
* **Memory Management** – Efficient handling of memory using malloc() and free() if dynamic memory allocation is implemented.

**Future Enhancements (Scalability Options)**

* **Database Integration (MySQL, SQLite)** – For better data management instead of file handling.
* **Graphical User Interface (GUI) with C++/GTK+** – To improve the user experience with a visual booking system.
* **Online Connectivity (Socket Programming in C)** – To enable remote booking in a networked environment.

This technology stack ensures that the **Train Ticket Booking System** is lightweight, efficient, and easy to use while maintaining secure data handling and streamlined operations.

**SYSTEM MODULES & FEATURES:**

The **Train Ticket Booking System** is divided into multiple modules, each handling a specific aspect of the booking process. Below is a breakdown of the key modules and their functionalities:

**1. User Authentication Module**

**Features:**

* User registration and login system.
* Authentication using stored credentials (basic file-based authentication).
* Role-based access (e.g., Admin vs. Passenger).

**2. Train Management Module *(Admin Feature)***

**Features:**

* Add, update, or remove train details (train number, route, time, seats, fare, etc.).
* View all available trains and their schedules.
* Manage seat availability dynamically.

**3. Ticket Booking Module**

**Features:**

* Display available trains based on user input (source, destination, date).
* Book tickets by selecting a train, number of seats, and class type.
* Assign unique booking reference numbers for each ticket.
* Store passenger details (name, age, gender, contact info).
* Update seat availability after successful booking.

**4. Ticket Cancellation Module**

**Features:**

* Allow users to cancel previously booked tickets using the reference number.
* Process refunds based on predefined cancellation rules.
* Update seat availability after cancellation.

**5. Ticket Inquiry & Search Module**

**Features:**

* Search for train schedules based on station names, date, or train number.
* Check seat availability before booking.
* Retrieve ticket details using booking reference number.

**6. Payment and Fare Calculation Module**

**Features:**

* Calculate fare based on distance, class type, and number of seats.
* Display total fare before confirming booking.
* Implement basic payment simulation (without real transactions).

**7. Report & Data Management Module *(Admin Feature)***

**Features:**

* Generate reports for booked tickets, available seats, and revenue collection.
* View and manage all passenger booking records.
* Backup and restore data using file handling.

**8. File Handling and Data Storage Module**

**Features:**

* Use text or binary files to store train details, user information, and booking history.
* Implement file read/write operations for data persistence.
* Ensure data consistency and security.

**Additional Features (Future Enhancements)**

* **Waitlist Management System** – Assign waitlist numbers if seats are full.
* **Online Booking Integration** – Use socket programming for remote ticket booking.
* **Graphical User Interface (GUI)** – Implement with GTK+ or C++ for better user experience.

This modular approach ensures the **Train Ticket Booking System** is efficient, scalable, and easy to maintain.

**SOURCE CODE:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_TICKETS 100

typedef struct {

int ticket\_id;

char passenger\_name[50];

int train\_number;

char seat\_class[10];

} Ticket;

Ticket tickets[MAX\_TICKETS];

int ticket\_count = 0;

void book\_ticket() {

if (ticket\_count >= MAX\_TICKETS) {

printf("Ticket booking full!\n");

return;

}

tickets[ticket\_count].ticket\_id = ticket\_count + 1;

printf("Enter passenger name: ");

getchar();

fgets(tickets[ticket\_count].passenger\_name, sizeof(tickets[ticket\_count].passenger\_name), stdin);

tickets[ticket\_count].passenger\_name[strcspn(tickets[ticket\_count].passenger\_name, "\n")] = 0; // Remove trailing newline

printf("Enter train number: ");

scanf("%d", &tickets[ticket\_count].train\_number);

printf("Enter seat class (Sleeper/AC/General): ");

scanf("%s", tickets[ticket\_count].seat\_class);

printf("Ticket booked successfully! Ticket ID: %d\n", tickets[ticket\_count].ticket\_id);

ticket\_count++;12

}

void display\_tickets() {

if (ticket\_count == 0) {

printf("No tickets booked yet!\n");

return;

}

printf("Booked Tickets:\n");

for (int i = 0; i < ticket\_count; i++) {

printf("Ticket ID: %d, Name: %s, Train No: %d, Class: %s\n",

tickets[i].ticket\_id, tickets[i].passenger\_name,

tickets[i].train\_number, tickets[i].seat\_class);

}

}

void cancel\_ticket() {

int id;

printf("Enter Ticket ID to cancel: ");

scanf("%d", &id);

int found = 0;

for (int i = 0; i < ticket\_count; i++) {

if (tickets[i].ticket\_id == id) {

found = 1;

for (int j = i; j < ticket\_count - 1; j++) {

tickets[j] = tickets[j + 1];

}

ticket\_count--;

printf("Ticket ID %d canceled successfully!\n", id);

break;

}

}

if (!found) {

printf("Ticket ID not found!\n");

}

}

int main() {

int choice;

while (1) {

printf("\nTrain Ticket Booking System\n");

printf("1. Book Ticket\n2. Display Tickets\n3. Cancel Ticket\n4. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

book\_ticket();

break;

case 2:

display\_tickets();

break;

case 3:

cancel\_ticket();

break;

case 4:

exit(0);

default:

printf("Invalid choice! Please try again.\n");

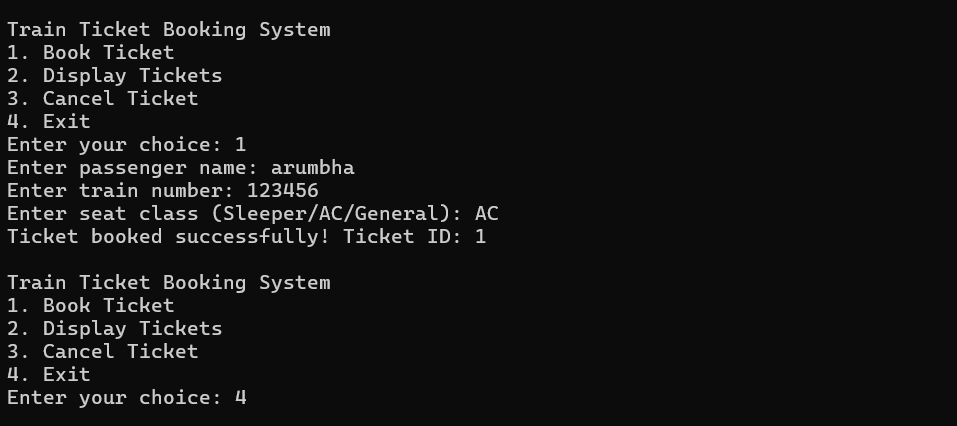
}

}

return 0;

}

**EXPECTED OUTCOME:**



**TIMELINES AND MILESTONES:**

**The Train Ticket Booking System development follows a structured timeline with key milestones for efficient completion.**

**Week 1-2: Planning & System Design**

**✅ Define project scope and requirements.  
✅ Design system architecture and data storage structure.**

**Week 3-6: Core Development**

**✅ Implement User Authentication & Train Management modules.  
✅ Develop Ticket Booking & Cancellation functionality.  
✅ Integrate Fare Calculation & Data Storage using file handling.**

**Week 7-8: Testing & Debugging**

**✅ Test all modules for errors and performance optimization.  
✅ Validate user inputs and ensure data accuracy.**

**Week 9-10: Finalization & Deployment**

**✅ Integrate all modules into a complete system.  
✅ Prepare documentation and conduct a project demo.**

**This streamlined timeline ensures a structured, efficient, and timely completion of the Train Ticket Booking System.**

**FUTURE ENHANCEMENT:**

After the initial development, the **Train Ticket Booking System** can be improved with additional features to enhance functionality and user experience.

**1. Graphical User Interface (GUI)**

🔹 Replace the text-based console system with a user-friendly graphical interface using **GTK+ or C++ with Qt**.

**2. Database Integration**

🔹 Use **MySQL or SQLite** instead of file handling for better data management, faster queries, and improved security.

**3. Online Booking System**

🔹 Implement **Socket Programming in C** to allow remote ticket booking over the internet.

**4. Waitlist Management**

🔹 Introduce a **waitlist system** where users can book if seats are unavailable and get auto-confirmation upon cancellations.

**5. Payment Gateway Integration**

🔹 Implement basic **online payment simulation** and later integrate real payment options (UPI, Credit/Debit Cards).

**6. Seat Selection Feature**

🔹 Allow users to choose their **preferred seat** (window, aisle, sleeper, AC/non-AC class).

**7. Real-Time Train Tracking**

🔹 Integrate APIs to show **live train status** and estimated arrival times.

**8. Mobile App Integration**

🔹 Extend the system to a **mobile app** using C-based frameworks or transition to web development for broader accessibility.

**CONCLUSION:**

The **Train Ticket Booking System** developed using C successfully automates the ticket reservation process, making it more efficient, reliable, and user-friendly. By integrating key features such as **train management, ticket booking, cancellations, fare calculation, and data storage**, the system eliminates the inefficiencies of manual booking methods.

Through **file handling for data storage**, the system ensures data persistence, while **user authentication** enhances security. The structured design of modules allows for easy navigation and future scalability.

This project not only demonstrates proficiency in **C programming, file handling, and structured programming concepts** but also lays the foundation for future enhancements such as **database integration, GUI implementation, online booking, and real-time train tracking**.

Overall, the **Train Ticket Booking System** provides a **fast, secure, and convenient** solution for railway ticket management, significantly improving the booking experience for passengers.

**REFERENCES:**

<https://www.geeksforgeeks.org/railway-reservation-system-in-c/>

<https://www.geeksforgeeks.org/railway-reservation-system-in-c/>

<https://www.slideshare.net/slideshow/online-train-ticket-booking-system-project-pdf/269832878>

<https://code-projects.org/train-ticket-reservation-system-c-programming-source-code/>

**THANK YOU**