🎟️ Ticket Booking System – Project Documentation

1. Project Title

Ticket Booking System
A simple and efficient ticket reservation platform for managing bookings, cancellations, and user data.

2. Objective

The main goal of this project is to design and implement a ticket booking system that allows users to:
- View available tickets
- Book tickets
- Cancel tickets
- View booking history

This project simulates real-world booking systems like those used in theaters, buses, railways, and events.

3. Tech Stack Used

- Language: C
- Database: MySQL / File System
- IDE: VS Code / IntelliJ / Eclipse
- Other Tools: Git, GitHub

4. Modules

1. User Module
- Register/Login
- View account details
- View booking history

2. Booking Module
- List available tickets
- Book a ticket
- Show booking confirmation

3. Cancellation Module
- Cancel existing bookings

- Refund handling (if applicable)

4. Admin Module (optional)
- Add or remove events/trips
- Manage ticket limits
- View all bookings

5. Features

✅ User-friendly interface
✅ Validations on inputs
✅ Booking limit enforcement
✅ Unique ticket ID generation
✅ Option to cancel with reason
✅ Simple menu-based navigation (if CLI)

6. Sample Test Cases

Action	Input	Expected Output
Book Ticket	Event ID: 102	Ticket booked successfully
Cancel Ticket	Ticket ID: T1001	Ticket cancelled
Book Ticket	No tickets left	Sorry, sold out!

7. System Architecture

- Main Class – Starts the application
- Booking Class – Handles ticket booking
- User Class – Manages user data
- Ticket Class – Stores ticket details

8. Code Snippet Example

```
typedef struct {
    int ticketId;
    char event[50];
    int isBooked;
} Ticket;

void initTicket(Ticket* t, int id, const char* eventName) {
    t->ticketId = id;
```

```
    strcpy(t->event, eventName);
    t->isBooked = 0;
}
```

9. Limitations

- No online payment integration
- Not scalable for real-world use
- CLI-based, not GUI/web-based

10. Future Scope

- Add GUI or web frontend (JavaFX or React)
- Integrate with a real payment gateway
- Role-based access (User/Admin)
- Dynamic seat allocation

11. Problem Statement

In today's fast-paced world, people rely heavily on online services for convenience. Traditional ticket booking methods, such as manual booking counters or phone reservations, are time-consuming, prone to human error, and often result in issues like double booking, long queues, lack of real-time availability, and mismanagement of records.

Customers face challenges in:

- Knowing the real-time availability of tickets.
- Booking or canceling tickets without physically visiting counters.
- Making secure and instant payments.
- Receiving quick confirmation of bookings.

On the other hand, administrators struggle with:

- Managing large volumes of ticket records manually.
- Generating timely reports and tracking revenue.
- Preventing booking conflicts or fraudulent activities

12.System Requirements

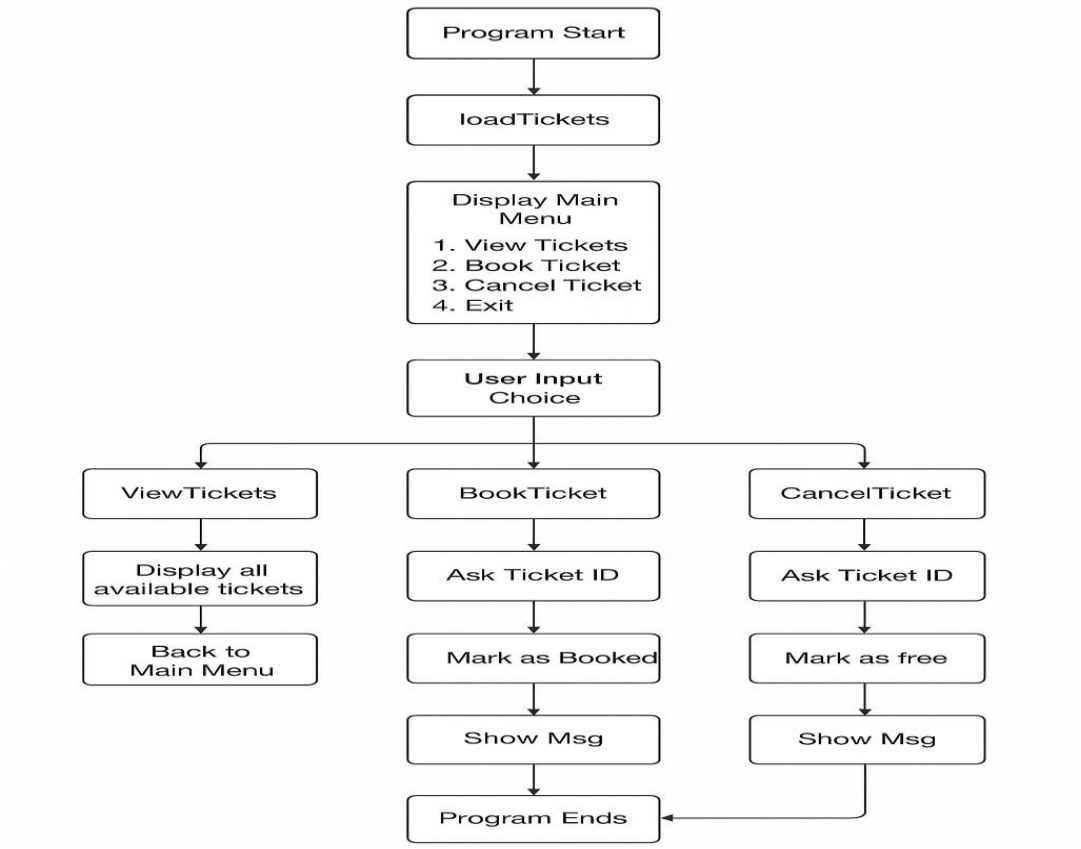
Hardware Requirements

- Processor: Intel i3 or higher
- RAM: Minimum 4 GB
- Storage: 250 GB HDD or above

Software Requirements

- Operating System: Windows/Linux
- Programming Language: Java / Python / PHP / .NET
- Database: MySQL / PostgreSQL / SQL Server
- Web Server: Apache / Tomcat / IIS
- Frontend: HTML, CSS, JavaScript, React/Angular (optional)

13.Diagram



14.Team member

1.Devyani Gade	2403031461193
2.Srushti Chaudhari	2403031460155
3.Aastha Gupta	2403031460143
4.Amogh Khadse	2403031460431
5.Pranav Chaudhari	2403031460422
6.Sarwan Choudhary	2403031461231

15. References

- Let Us C by Yashavant Kanetkar
- www.geeksforgeeks.org
- C Programming Tutorials – TutorialsPoint
- Array of Strings in C – Visual Diagram