**Railway Reservation System**

**EXPERIMENT 8**

**Prepare prototype of the Railway Reservation System**

**Definition of Prototype**

A **prototype** is an early version of a system used for testing and improvement before the final version is built. It helps developers and stakeholders understand what’s needed, find mistakes, and make the system better before full development. Prototyping reduces risks and makes the system more user-friendly by allowing early feedback.

**1. Introduction**

The **Railway Reservation System** is an online platform that makes it easy for passengers to book train tickets, check seat availability, and manage their reservations. It improves railway operations while ensuring a smooth experience for travelers.

**2. Key Features**

1. **User Registration and Login** – Passengers can sign up, log in, and manage their reservations securely.
2. **Train Search & Seat Availability** – Users can search for trains based on routes, travel dates, and available seats.
3. **Ticket Booking System** – Passengers can choose their trains, select seats, and make payments online.
4. **PNR Status Tracking** – Users can check their reservation status using a unique Passenger Name Record (PNR) number.
5. **Ticket Cancellation & Refunds** – If needed, users can cancel their tickets and get refunds based on railway policies.
6. **Admin Control Panel** – Railway staff can manage train schedules, reservations, and passenger queries to ensure smooth operations.

**3. Benefits of the Prototype**

* **Early Testing & Feedback** – Helps find and fix issues before the final system is developed.
* **Better User Experience** – Ensures the system is easy to use and navigate.
* **Efficient Railway Operations** – Automates reservations, reducing human errors.
* **Scalable & Secure** – Ensures the system can grow and remain secure as more users join.

**Prototype of the system:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct Train {

    int id;

    char name[50];

    char from[50];

    char to[50];

    int seats;

};

struct Booking {

    int bookingId;

    int trainId;

    char passenger[50];

};

void addDummyTrains() {

    FILE \*fp = fopen("trains.txt", "wb");

    struct Train trains[2] = {

        {1, "Rajdhani Express", "Delhi", "Mumbai", 100},

        {2, "Shatabdi Express", "Delhi", "Bangalore", 80}

    };

    fwrite(trains, sizeof(struct Train), 2, fp);

    fclose(fp);

}

void viewTrains() {

    FILE \*fp = fopen("trains.txt", "rb");

    struct Train t;

    printf("ID\tName\t\t\tFrom\tTo\tSeats\n");

    while (fread(&t, sizeof(t), 1, fp)) {

        printf("%d\t%s\t%s\t%s\t%d\n", t.id, t.name, t.from, t.to, t.seats);

    }

    fclose(fp);

}

void bookTicket() {

    int id;

    char name[50];

    printf("Enter Train ID: ");

    scanf("%d", &id);

    printf("Enter Passenger Name: ");

    scanf("%s", name);

    struct Booking b;

    b.trainId = id;

    strcpy(b.passenger, name);

    FILE \*fp = fopen("bookings.txt", "ab+");

    fseek(fp, 0, SEEK\_END);

    int size = ftell(fp) / sizeof(struct Booking);

    b.bookingId = size + 1;

    fwrite(&b, sizeof(b), 1, fp);

    fclose(fp);

    printf("\nTicket Booked! \nBooking ID: %d\n", b.bookingId);

}

void cancelTicket() {

    int cancelId;

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

    scanf("%d", &cancelId);

    FILE \*fp = fopen("bookings.txt", "rb");

    FILE \*temp = fopen("temp.txt", "wb");

    struct Booking b;

    int found = 0;

    while (fread(&b, sizeof(b), 1, fp)) {

        if (b.bookingId != cancelId) {

            fwrite(&b, sizeof(b), 1, temp);

        } else {

            found = 1;

        }

    }

    fclose(fp);

    fclose(temp);

    remove("bookings.txt");

    rename("temp.txt", "bookings.txt");

    if (found) {

        printf("Booking ID %d cancelled.\n", cancelId);

    } else {

        printf("Booking ID not found.\n");

    }

}

void viewBookings() {

    FILE \*fp = fopen("bookings.txt", "rb");

    struct Booking b;

    printf("Booking ID\tTrain ID\tPassenger\n");

    while (fread(&b, sizeof(b), 1, fp)) {

        printf("%d\t\t%d\t\t%s\n", b.bookingId, b.trainId, b.passenger);

    }

    fclose(fp);

}

int main() {

    int choice;

    addDummyTrains(); // Add demo trains once

    do {

        printf("\nRailway Reservation System\n");

        printf("1. View Trains\n2. Book Ticket\n3. Cancel Ticket\n4. View Bookings\n5. Exit\n");

        printf("Enter choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1: viewTrains(); break;

            case 2: bookTicket(); break;

            case 3: cancelTicket(); break;

            case 4: viewBookings(); break;

            case 5: printf("Goodbye!\n"); break;

            default: printf("Invalid option.\n");

        }

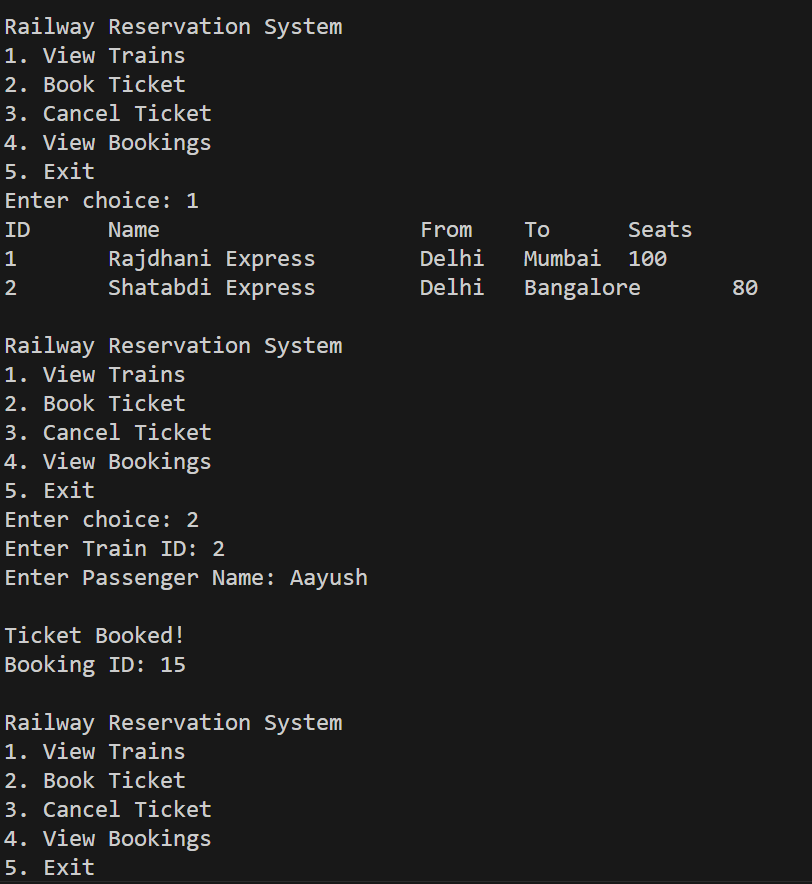
    } while (choice != 5);

    return 0;

}

**Terminal output of the system:**

**For Booking**



**For Cancellation**

