```
/*
Group ID= 07

Members:
   1)Arunabh Aditya          Roll Number: 114026   PRN:1032230178
   2)M Aryane               Roll Number: 114029   PRN:1032232583
   3)Akshay Anurag           Roll Number: 114028   PRN:1032231965
   4)Fardeen Ali            Roll Number: 114027   PRN:1032232038
   5)Ayush Gaikwad           Roll Number: 114039   PRN:1032232473
   6)Shriharsh Deshmukh       Roll Number: 114038   PRN:1032231397

Problem Statement:
write a C program to reserve train tickets with following functionality:
   1)User System:
      i)Login
      ii)signup
      iii)Reset Password
   2)Admin Controls
      i)add new trains
      ii)add delayed trains
      iii)see delayed trains
   3)Tickets reservation:
             i)With option to select preferred class, compartment,seat type (seater or sleeper)      and number of seats
               ii)With payment system.(with payment failure handling case).
      iii)Once reserved, get PNR.
   4)Get Information of reservations (By entering PNR):
      i)To know in which compartment, class seat is reserved.
      ii)To know if ticket is confirmed or in waiting list.
      iii)Get reservation id or referrence number of the reservation.
      iv)To know all about train (max speed, total dist to be travelled )
      V)Option to print tickets in PDF format
   5)Checking Train Status:
      i)To check delayed trains
   6)Option to cancel tickets:
      i)option to cancel ticket.
      ii)with proper refund messages.
   ***ADDITIONAL FEATURES***
   1)QR code payment format
   2)User profile update
   3)Admin panel
   4)ability to cancel tickets or go back to main menu at any step
   5)Proper error handling.
   6)proper user navigation allowing a smoother program experience.
   7)Proper PDF generation.
   8)Proper user interface allowing user to perform all actions without having to restart    the program.
   9)Menu Based program.
   *****CAUTIONS*****
   1)Input should be of the type that is asked.
   2)All input should be in lowercase (unless stated otherwise and as per choice for name and other user details).
```

Input Required:
as per the program flow and user wishes.

Algorithms used (modules used):
1)External library:
i)qrcodegen.h
ii)pdfgen.h
2)Internal header files:
i)stdio.h
ii)stdlib.h
iii)conio.h
iv)string.h
v)time.h
vi)windows.h
vii)ctype.h

Conclusion:
Thus implemented a complete error safe train ticket booking application with TUI (terminal user interface) allowing a menu based program.
*/

//Header Files and #define

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <time.h>
#include <windows.h>
#include "header/qrcodegen.h"
#include <ctype.h>
#include "header/pdfgen.h"

#define MAX_PASSENGERS 100
```

//structure definations total 6 structures:

```
typedef struct {
    char userId[10];
    char email[100];
    char password[100];
    char name[100];
    int age;
    char gender[10];
    long long phone;
} User;

typedef struct {
    char trainId[10];
    char startingPoint[100];
    char destination[100];
    char departureTime[100];
    int cost;
    int compartments;
    char seatType[100];
    char name[100];
    int maxSpeed;
    int totalDist;
} Train;

typedef struct {
    long long pnr;
    char userId[10];
    char name[100];
    char trainId[10];
    int compartment;
    char seatType[10];
    char status[10];
    float cost;
    int seats;
    int noOfSeats;
    char date[20];
    int age;
    char gender[10];
    long long phone;
} Reservation;

typedef struct {
    char trainId[20];
    char startingPoint[20];
    char destination[20];
    char departureTime[15];
    int cost;
    int compartments;
    char seatType[20];
    int isDelayed;
    int delayTime;
} DelayedTrain;

typedef struct PassengerTicket PassengerTicket;

typedef struct {
    char StartingPoint[20];
    char Destination[20];
    char PNR[20];
    char TrainName[20];
    char TrainId[20];
    char class[20];
    char Date[20];
    char dist[20];
    char Cost[20];
    char Compartment[20];
    int totalSeats;
    PassengerTicket* passengerTickets[10];
} Ticket;

struct PassengerTicket {
    char passName[20];
    char age[20];
    char gender[20];
    char seatNum[20];
};
```

//Global Veriables:

```
int delayMilliseconds = 5000;
int logged = 0;
int *loggedPtr = &logged;
char userId[10];
char *userIdPtr = userId;
```

//All function declarations: total 48 functions

```
void sleepProgram(float seconds);
void PrintSleep(float seconds);
void clearTerminal();
void greenColor();
void redColor();
void resetColor();
void yellowColor();
void signup();
int userExists(const char* email);
void login();
void copyFile(const char *source, const char *destination);
void resetPass();
void LogOrSign();
void adminLogin();
void addTrain();
void displayAllTrains();
int compareByCostAsc(const void* a, const void* b);
int compareByCostDesc(const void* a, const void* b);
void sortByCost(int sortOrder);
void findTrainsByDestination(const char* destination);
void findTrainsByStartingPoint(const char* startingPoint);
void adminControls();
void addDelayedTrain();
void displayDelayedTrains(int isAdmin);
void trainListandBook();
void findTrain(const char* trainId);
void showCompartment( int compartment, const char* trainId);
void randomlyBookSeats(int* seats, int numSeats);
int isValidDate(const char *dateStr);
void payNow(const char* trainId, int ticketsToBuy, int choosenCompartment, int ticketsNums[MAX_PASSENGERS], const char* date);
void showTickets(int choosenCompartment, const char* trainId);
int isAllDigits(const char *str);
void paymentGateway(const char* trainId, int ticketsToBuy, int choosenCompartment, float totalPrice, int ticketsNums[MAX_PASSENGERS], const char* date);
long long generate10DigitRandomNumber();
long long generate15DigitRandomNumber();
void writeReservationToFile(const Reservation* passenger);
void confirmTickets(const char* trainId, int ticketsToBuy, int choosenCompartment, int* ticketsNums, float totalPrice, const char* date);
void PrintToBeDeletedReservation(long long pnr);
void deleteReservationByPnr(long long pnr);
void cancelBooking();
void seeReservationInfo(long long pnrInfo);
void reservationInfo();
void getTicketFormat();
void getTicketFormatAfterBooking(long long pnrInfo);
int writeTicketPDF(const Ticket* ticket);
void checkAllReservations();
void showUserInfo();
void changeUserInfo();
void mainMenu();
```

//PROGRAM STRUCTURE:

1)Login Signup processes

```
void LogOrSign()
void login()
void signup()
void showUserInfo();
```

2)Admin Controls

```
void adminControls()
void adminLogin()
void addTrain();
void addDelayedTrain()
```

3) display trains and booking process

```
void displayAllTrains();
void findTrain(const char* trainId);
void showCompartment( int compartment, const char* trainId);
void showTickets(int choosenCompartment, const char* trainId);
```

4)Payment Process

```
void payNow(const char* trainId, int ticketsToBuy, int choosenCompartment, int ticketsNums[MAX_PASSENGERS],
const char* date);

void paymentGateway(const char* trainId, int ticketsToBuy, int choosenCompartment, float totalPrice, int
ticketsNums[MAX_PASSENGERS], const char* date);

void confirmTickets(const char* trainId, int ticketsToBuy, int choosenCompartment, int* ticketsNums, float totalPrice,
const char* date);

void writeReservationToFile(const Reservation* passenger);
```

5)Reservation Information and cancellation

```
void reservationInfo();
void deleteReservationByPnr(long long pnr);
void cancelBooking();
void getTicketFormat();
int writeTicketPDF(const Ticket* ticket);
```

// Function to handle login, signup, reset password, or admin login

```
void LogOrSign() {
   int LogOrSignOpt;

   // Clear the terminal screen
   clearTerminal();

   // Display menu options
   printf("Please Login or Signup to continue booking tickets");
   printf("  Select from the following options:\n\n");
   printf("   \033[1;31m[1]\033[0m Login \n\n");
   printf("   \033[1;31m[2]\033[0m Signup\n\n");
   printf("   \033[1;31m[3]\033[0m Reset Password\n\n\n\n");
   printf("   \033[1;31m[4]\033[0m ADMIN LOGIN\n\n");

   // Get user input
   scanf("%d", &LogOrSignOpt);

   // Process user choice using a switch statement
   switch (LogOrSignOpt) {
      case 1:
         login();
         break;
      case 2:
         signup();
         break;
      case 3:
         resetPass();
         break;
      case 4:
         adminLogin();
         break;
      default:
         // Handle invalid input
         printf("Please Choose Correct Option");
         sleepProgram(1);
         LogOrSign(); // Recursive call to the function to retry
         break;
   }
}
```

//Login Function

```
void login() {
    int wrongPass, wrongUser;
    char email[100], password[100], line[500];
    printf("\nPlease enter your email:");
    scanf("%s", email);
    getchar();
    FILE *file = fopen("files/users.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }
    int userFound = 0;
    while (fgets(line, sizeof(line), file) != NULL) {
        User user;
        sscanf(read user file);
        if (strcmp(user.email, email) == 0) {
            userFound = 1;
            printf("\nUser Found\n");
            printf("Please enter your password: ");
            scanf("%s", password);
            getchar();
            if (strcmp(user.password, password) == 0) {
                printf("\nLogin successful!\n");
                *loggedPtr = 1;
                strcpy(userIdPtr, user.userId);
                mainMenu();
                break;
            } else {
                printf("\nIncorrect password. Please try again.\n");
                printf("\nPress 1 to try again or press 2 to reset password!\n");
                scanf("%d", &wrongPass);
                if (wrongPass == 1) {
                    login();
                } else if (wrongPass == 2) {
                    resetPass();
                }
            }
            break;
        }
    }
    fclose(file);
    if (!userFound) {
        printf("\nUser not found. Please create an account.\n");
        printf("\nPress 1 to create an account or press 2 to try again!\n");
        scanf("%d", &wrongUser);
        if (wrongUser == 1) {
            signup();
        } else if (wrongUser == 2) {
```

```
                        login();
      }
   }
}
```

//Signup Function

```
void signup() {

    clearTerminal();
    int SignedUp, AlrExists;

    FILE *file = fopen("files/users.txt", "a");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }
    User newUser; //initiate structure
    printf("Please enter your name: ");
    scanf("%s", newUser.name);
    printf("Please enter your Age: ");
    scanf("%d", &newUser.age);
    printf("Please enter your Gender: ");
    scanf("%s", newUser.gender);
    printf("Please enter your Phone: ");
    scanf("%lld", &newUser.phone);
    printf("Please enter your email: ");
    scanf("%s", newUser.email);
    if (userExists(newUser.email)) {
        printf("\nUser with the same email already exists. Press 1 to log in. Press 2 to try again\n");
        scanf("%d", &AlrExists);
        if (AlrExists == 1) {
            login();
        } else if (AlrExists == 2) {
            signup();
        }
        return;
    }
    printf("Please enter your password: ");
    scanf("%s", newUser.password);
    sprintf(newUser.userId, "%08d", rand() % 100000000);
    fprintf(file, "%s %s %s %s %d %s %lld\n", newUser.userId, newUser.email, newUser.password, newUser.name,
newUser.age, newUser.gender, newUser.phone);
    printf("\nNew user created successfully!\n");
    printf("\nHere are user Details:\n");
    printf("User ID: %s\n", newUser.userId);
    printf("Email: %s\n", newUser.email);
    printf("Name: %s\n", newUser.name);
    printf("Age: %d\n", newUser.age);
    printf("Gender: %s\n", newUser.gender);
    printf("Phone Number: +91 %lld\n", newUser.phone);
    fclose(file);

    printf("\nAccount created successfully! Please log in.\n");
    printf("Press 1 to login or press 2 to create another account: ");
    scanf("%d", &SignedUp);
    if (SignedUp == 1) {
        login();
    } else if (SignedUp == 2) {
        signup();
    }
}
```

// Function for displaying user information

```
void showUserInfo(const char* userId) {
    int userInfoChoice;
    FILE *file = fopen("files/users.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }
    int foundUser = 0;

    char line[500];
    while (fgets(line, sizeof(line), file) != NULL) {
        User user;
        sscanf(line, "%s %s %s %s %d %s %lld", user.userId, user.email, user.password, user.name, &user.age, user.gender,
&user.phone);

        if (strcmp(user.userId, userId) == 0) {
            foundUser = 1;
            sleepProgram(0.18);
            clearTerminal();
            printf("\t\t\t\t\t\033[1;31mYour Profile\033[0m\t\t");
            printf("\tUser ID: %s\n", user.userId);
            printf("\tName: %s\n", user.name);
            printf("\tAge: %d\n", user.age);
            printf("\tGender: %s\n", user.gender);
            printf("\tPhone: +91 %lld\n", user.phone);
            printf("\tEmail: %s\n", user.email);
            printf("\tPassword: %s\n", user.password);
            break;
        }


    fclose(file);
    if (foundUser == 1) {
        printf("\nPress 1 to change password or Press 2 to update profile details or Press any key to go back to Main
Menu\n");
        scanf("%d", &userInfoChoice);
        if (userInfoChoice == 1) {
            sleepProgram(2);
            resetPass();
        } else if (userInfoChoice == 2) {
            sleepProgram(2);
            changeUserInfo();
        } else {
            mainMenu();
        }
    }
}
```

//Function to show admin controls

```
void adminControls() {
   clearTerminal();
   greenColor();

   printf("\n\t\t\tWelcome To Admin Controls\n");
   printf("  *Please Select Appropriate Option:*\n");

   printf("   \033[1;31m[1]\033[0m VIEW TRAIN LIST \n\n");
   printf("   \033[1;31m[2]\033[0m ADD NEW TRAIN\n\n");
   printf("   \033[1;31m[3]\033[0m Add Delayed Trains\n\n");
   printf("   \033[1;31m[4]\033[0m Display Delayed Trains\n\n");
   printf("   \033[1;31m[5]\033[0m GO BACK TO LOGIN SIGNUP WINDOW\n\n");
   int adminChoice;
   scanf("%d", &adminChoice);
   switch (adminChoice) {
   case 1:
      sleepProgram(1);
      displayAllTrains();
      sleepProgram(10);
      adminControls();
      break;
   case 2:
      sleepProgram(1);
      addTrain();
      break;
   case 3:
      sleepProgram(1);
      addDelayedTrain();
      break;
   case 4:
      sleepProgram(1);
      displayDelayedTrains(0);
      break;
   case 5:
      sleepProgram(1);
      LogOrSign();
      break;
   default:
      printf("Oops Wrong Choice");
      adminControls();
      break;
   }
}
```

//Function For Admin Login

```
void adminLogin() {
   int AdminLoginOpt;
   clearTerminal();
   redColor();
   char enteredAdminPass[100];
   char correctAdminPass[] = "G7";

   printf("Please Login With ADMIN PASS to continue to ADMIN CONTROLS");
   printf("Print Admin PASS (case-sensitive)[ It is G7 :)] :");
   scanf("%s", enteredAdminPass);
   PrintSleep(0.18);
   getchar();

   if (strcmp(enteredAdminPass, correctAdminPass) == 0) {
      printf("Welcome to ADMIN CONTROLS");
      printf("You will be redirected to ADMIN CONTROLS in 3 seconds!");
      sleepProgram(3);
      adminControls();
   } else {
      printf("Incorrect Password");
      printf("Do you want to try again? press 1 to try again, press any other key to return to login");
      scanf("%d", &AdminLoginOpt);
      if (AdminLoginOpt == 1) {
         sleepProgram(1);
         adminLogin();
      } else {
         sleepProgram(1);
         LogOrSign();
      }
   }
}
```

//Function to add train

```
void addTrain() {
  FILE *file = fopen("files/train.txt", "r+");
  if (file == NULL) {
    printf("Error opening file.\n");
    exit(1);
  }
  int numTrains;
  fscanf(file, "%d", &numTrains);
  numTrains++;
  fseek(file, 0, SEEK_SET);
  fprintf(file, "%d\n", numTrains);
  fseek(file, 0, SEEK_END);
  Train newTrain;
  printf("\nPlease enter the Train ID: ");
  scanf("%s", newTrain.trainId)
  printf("Please enter the Starting Point: ");
  scanf("%s", newTrain.startingPoint);
  printf("Please enter the Destination: ");
  scanf("%s", newTrain.destination);
  printf("Please enter the Departure Time: ");
  scanf("%s", newTrain.departureTime);
  printf("Please enter the Cost: ");
  scanf("%d", &newTrain.cost);
  printf("Please enter number of Compartments: ");
  scanf("%d", &newTrain.compartments);
  printf("Please enter Seat type:- sleeper or seater: ");
  scanf("%s", newTrain.seatType);
  printf("Please enter Name of Train: ");
  scanf("%s", newTrain.name);
  printf("Please enter max speed of train: ");
  scanf("%d", &newTrain.maxSpeed);
  printf("Please enter total distance to be travelled by train: ");
  scanf("%d", &newTrain.totalDist);
  fprintf(file, print train data to file);
  fclose(file);
  printf("\nTrain added successfully!\n");
  printf("\nPress 1 to add another train or press any other key to go to admin controls.\n");
  int addAnotherTrain;
  scanf("%d", &addAnotherTrain);
  if (addAnotherTrain == 1) {
    sleepProgram(1);
    addTrain();
  } else {
    sleepProgram(1);
    adminControls();
  }

}
```

//function to add delayed train

```
void addDelayedTrain() {
    FILE *file = fopen("files/delayed.txt", "a");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }
    DelayedTrain delayedTrain; //create structure
    Get delayed train details

    fprintf(store delayed train data in file);

    fclose(file);

    printf("\nDelayed train added successfully!\n");
    sleepProgram(3);
    adminControls();
}
```

//Function to display Trains

```
void displayAllTrains() {
    FILE *file = fopen("files/train.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    int numTrains;
    fscanf(file, "%d", &numTrains);

    printf("| %-10s | %-20s | %-20s | %-15s | %-5s | %-20s | %-20s | %-20s | %-5s | %-5s | ", "Train ID", "Starting Point",
"Destination", "Departure Time", "Cost", "Compartments", "Seat Type", "Train Name", "Max Speed", "Total Distance");

    for (int i = 0; i < numTrains; i++) {
        Train train;
        fscanf(file, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint, train.destination,
train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed, &train.totalDist);

        printf("| %-10s | %-20s | %-20s | %-15s | %-5d | %-20d | %-20s | %-20s | %-5dKM/H | %-12dKM |\n", train.trainId,
train.startingPoint, train.destination, train.departureTime, train.cost, train.compartments, train.seatType, train.name,
train.maxSpeed, train.totalDist);
    }

    fclose(file);
}
```

//Function to find train

```
void findTrain(const char* trainId) {
   int trainNotFound;
   FILE* file = fopen("files/train.txt", "r");
   if (file == NULL) {
      printf("Error opening file.\n");
      exit(1);
   }

   int numTrains;
   fscanf(file, "%d", &numTrains);

   Train train;
   int trainFound = 0;
   PrintSleep(0.18);

   while (fscanf(file, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint, train.destination,
train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed, &train.totalDist)
!= EOF) {
      if (strcmp(train.trainId, trainId) == 0) {
         trainFound = 1;
         clearTerminal();
         printf("Train found!\n");
         printf("Train ID: %s\n", train.trainId);
         printf("Starting Point: %s\n", train.startingPoint);
         printf("Destination: %s\n", train.destination);
         printf("Departure Time: %s\n", train.departureTime);
         printf("Cost: %d (cost for general class ticket!!)\n", train.cost);
         printf("Compartments: %d\n", train.compartments);
         printf("Seat Type: %s\n", train.seatType);
         printf("Train Name: %s\n", train.name);
         printf("Max Speed: %d km/h\n", train.maxSpeed);
         printf("Total Distance to be travelled: %d\n", train.totalDist);
         sleepProgram(1);
         showCompartment(train.compartments, train.trainId);
         break;
      }
   }

   fclose(file);

   if (!trainFound) {
      printf("\nTrain with ID %s not found. Press 1 to view train list or any other key to go back to main menu.\n", trainId);
      scanf("%d", &trainNotFound);
      if (trainNotFound == 1) {
         displayAllTrains();
      } else {
         mainMenu();
      }
   }
```

//Function to show compartment

```
void showCompartment(int compartment, const char* trainId) {
    int chosenCompartment;
    printf("Train Compartments:\n");
    printf("    +----------------");
    for (int i = 0; i < compartment; ++i) {
        printf("----------------");
    }
    printf("+\n");
    printf("  <(     Engine      |");
    for (int i = 0; i < compartment; ++i) {
        printf(" Compartment %2d|", i + 1);
    }
    printf("\n");
    printf("    +----------------");
    for (int i = 0; i < compartment; ++i) {
        printf("----------------");
    }
    printf("+\n");
    sleepProgram(1);
    printf("\n\n\nPlease Enter Your Preferred Compartment:\n");
    yellowColor();
    printf("\n\t1) First compartment is First AC\n");
    sleepProgram(0.5);
    printf("\t2) Second compartment is Executive Class\n");
    sleepProgram(0.5);
    printf("\t3) Third compartment is Third AC\n");
    sleepProgram(0.5);
    printf("\t4) Fourth compartment is Sleeper\n");
    sleepProgram(0.5);
    printf("\t5) Rest Are Unreserved General Class\n");
    scanf("%d", &chosenCompartment);
    resetColor();
    sleepProgram(1);
    showTickets(chosenCompartment, trainId);
}
```

//Function to show tickets

```
void showTickets(int choosenCompartment, const char* trainId) {
    int numSeats = 0;

    char date[20];
    int isValid = 0;

    // Get Departure Date
    do {
        printf("Enter Your Departure Date in dd/mm/yyyy format: ");
        scanf("%99s", date);

        isValid = isValidDate(date);

        if (!isValid) {
            printf("Invalid date or date is in the past. Please enter a valid future date.\n");
        }

    } while (!isValid);

    // Determine Number of Seats
    if (choosenCompartment == 1 || choosenCompartment == 2 || choosenCompartment == 3) {
        numSeats = 54;
    } else if (choosenCompartment == 4) {
        numSeats = 18;
    } else {
        numSeats = 72;
    }

    // Initialize Seat Arrays
    int seats[MAX_PASSENGERS];
    int bookedSeats[MAX_PASSENGERS];

    for (int i = 0; i < numSeats; ++i) {
        seats[i] = 0;
    }

    randomlyBookSeats(seats, numSeats);

    for (int i = 0; i < numSeats; ++i) {
        bookedSeats[i] = seats[i];
    }

    // Display Compartment and Seat Information
    printf("\nYour chosen compartment is: %d", choosenCompartment);
    sleepProgram(0.5);

    if (choosenCompartment == 1 || choosenCompartment == 2 || choosenCompartment == 3) {
        printf("\nThe seat type is: %s", "AC Seater");
    } else if (choosenCompartment == 4) {
        printf("\nThe seat type is: %s", "Sleeper");
    } else {
```

```
    printf("\nThe seat type is: %s", "Seater");
}

printf("\nYour travel date is: %s", date);
sleepProgram(1);

printf("\nThe Tickets are:\n");
printf("\t1) Red Are Booked Tickets\n");
printf("\t2) Green Are Available Tickets\n");
sleepProgram(1.5);

// Display Seat Availability
for (int i = 0; i < numSeats; ++i) {
    if (i % 6 == 0) {
        printf("\n");
    }
    if (i % 18 == 0) {
        printf("\n");
    }

    if (seats[i] == 1) {
        printf("\033[1;31m");
    } else {
        printf("\033[1;32m");
    }

    printf("\t| Seat %2d |", i + 1);
    printf("\033[0m");
}

printf("\n");
sleepProgram(1);

// User Input for Ticket Purchase
printf("\n\n\nEnter how many tickets you want to buy: ");
int ticketsToBuy;
scanf("%d", &ticketsToBuy);
printf("\n");

int TicketsNums[MAX_PASSENGERS];

for (size_t i = 0; i < ticketsToBuy; i++) {
    do {
        printf("Enter your %d ticket number: ", i + 1);
        scanf("%d", &TicketsNums[i]);
        printf("\n");

        if (TicketsNums[i] < 1 || TicketsNums[i] > numSeats) {
            printf("Invalid ticket number. Please enter a valid ticket number.\n");
        } else if (bookedSeats[TicketsNums[i] - 1] == 1) {
            printf("Ticket %d is already booked. Please choose another ticket.\n", TicketsNums[i]);
        }
    } while (TicketsNums[i] < 1 || TicketsNums[i] > numSeats || bookedSeats[TicketsNums[i] - 1] == 1);
```

```
        bookedSeats[TicketsNums[i] - 1] = 1;
    }

    sleepProgram(1);

    // Payment Confirmation
    printf("Please Complete the payment to Confirm your bookings!!!\n");
    payNow(trainId, ticketsToBuy, choosenCompartment, TicketsNums, date);
}
```

//PAYNOW Function

```
void payNow(const char* trainId, int ticketsToBuy, int choosenCompartment, int ticketsNums[MAX_PASSENGERS], const
char* date) {
   float totalPrice = 0;
   int wantToPay;
   FILE* file = fopen("files/train.txt", "r");
   if (file == NULL) {
      printf("Error opening file.\n");
      exit(1);
   }
   int numTrains;
   fscanf(file, "%d", &numTrains);
   Train train;

   while (fscanf(file, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint, train.destination,
train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed, &train.totalDist)
!= EOF) {
      if (strcmp(train.trainId, trainId) == 0) {
         if (choosenCompartment == 1 || choosenCompartment == 2) {
            totalPrice = ticketsToBuy * train.cost * 2.5;
         } else if (choosenCompartment == 3) {
            totalPrice = ticketsToBuy * train.cost * 1.8;
         } else if (choosenCompartment == 4) {
            totalPrice = ticketsToBuy * train.cost * 1.4;
         } else {
            totalPrice = ticketsToBuy * train.cost;
         }
         break;
      }
   }

   fclose(file);
   sleepProgram(1);
   printf("\nYour Total Cost Is: %2f\n", totalPrice);
   sleepProgram(1);
   printf("\nPress 1 to continue to pay or press 2 to go back to ticket selection!!\n");
   scanf("%d", &wantToPay);

   if (wantToPay == 1) {
      sleepProgram(1);
      paymentGateway(trainId, ticketsToBuy, choosenCompartment, totalPrice, ticketsNums, date);
   } else if (wantToPay == 2) {
      sleepProgram(1);
      findTrain(trainId);
   }
}
```

//function for payment gateway

```
void paymentGateway(const char* trainId, int ticketsToBuy, int choosenCompartment, float totalPrice, int
ticketsNums[MAX_PASSENGERS], const char* date)
 {
   int gatewayChoice;
   int wantToPay;
   char cardNumber[100];
   char expiryDate[100];
   char cvv[10];
   clearTerminal();
   printf("\n===================");
   redColor();
   printf("  PAYMENT GATEWAY  ");
   resetColor();
   printf("====================\n\n");
   printf("   \033[1;31m[1]\033[0m CREDIT or DEBIT CARD\n\n");
   sleepProgram(0.5);
   printf("   \033[1;31m[2]\033[0m UPI (QR CODE)\n\n");
   sleepProgram(0.5);
   printf("   \033[1;31m[3]\033[0m CANCEL BOOKING\n\n");
   sleepProgram(0.5);
   printf("   \033[1;31m[4]\033[0m PAYMENT INFO\n\n");
   sleepProgram(0.5);
   printf("   \033[1;31m[5]\033[0m GO BACK TO TICKET BOOKING\n\n");
   sleepProgram(0.5);
   printf("   \033[1;31m[6]\033[0m EXIT TO MAIN MENU\n\n");
   printf("Please enter an option\n");
   scanf("%d", &gatewayChoice);
   getchar();
   switch (gatewayChoice)
   {
   case 1:
     printf("\nTo cancel payment please enter '7887' in card number field\n");

     do {
        printf("Enter Card Number (16 digits): ");
        scanf("%s", cardNumber);
        if (cardNumber[0] == '7' && cardNumber[1] == '8' && cardNumber[2] == '8' && cardNumber[3] == '7')
        {
           findTrain(trainId);
           break;
        }
     printf("\nEnter Expiry Date: in format (mm/yyyy) ");
     scanf("%s", expiryDate);
    printf("Enter CVV  (3 digits): ");
     printf("Payment Successful!!");
     printf("\nYou will be redirected to Tickets Section in 3 seconds\n");
     sleepProgram(3);
     confirmTickets(trainId, ticketsToBuy, choosenCompartment, ticketsNums, totalPrice, date);
```

```
  break;

case 2:

  printf("\nYou have chosen QR payment. Please scan following QR code to make payment\n");
  printf("\nEnter the code obtained by scanning the QR code. ");
  redColor();
  printf("\nPlease Confirm Amount To Be Paid Before Scanning QR Code");
  printf("\nTo cancel payment please enter '7887'\n");
  resetColor();
  printf("\n\n");
  char* paymentInfo = calloc(1000, sizeof(char));
  sprintf(paymentInfo + strlen(paymentInfo), "\nTotal Cost: %.2f\n", totalPrice);
  int i = verifyByQrWithText("Your OTP Is:", paymentInfo);
  free(paymentInfo);
  if (i == 1) {
     PrintSleep(0.18);
     printf("Payment cancelled");
     printf("\nYou will be redirected to train section in 3 seconds\n");
     sleepProgram(3);
     trainListandBook();
  } else if(i == 0) {
     PrintSleep(0.18);
     printf("Payment successful");
     printf("\nYou will be redirected to confirm ticket details in 2 seconds\n");
     sleepProgram(1.5);
     confirmTickets(trainId, ticketsToBuy, choosenCompartment, ticketsNums, totalPrice, date);
  } else{
     PrintSleep(0.18);
     printf("Payment failed");
     sleepProgram(1);
     printf("\nYou will be redirected to tickets section of this train in 3 seconds\n");
     sleepProgram(3);
     findTrain(trainId);
  }
  break;

case 3:
  printf("\nYou have cancelled ticket booking.\n");
  printf("\nYou will be redirected to train section in 5 seconds\n");
  sleepProgram(3);
  findTrain(trainId);
  break;

case 4:
  PrintSleep(0.18);
  printf("\nYour Payment Information Is As Following:\n");
  printf("Total Cost: %f\n", totalPrice);
  printf("Total Tickets: %d\n", ticketsToBuy);
  printf("Train ID: %s\n", trainId);
```

```
      printf("Compartment: %d\n", choosenCompartment);
      if (choosenCompartment == 1 || choosenCompartment == 2 || choosenCompartment == 3)
      {
      printf("\nThe seat type is: %s", "AC Seater");
      } else if (choosenCompartment == 4)
      {
      printf("\nThe seat type is: %s", "Sleeper");
      } else
      {
      printf("\nThe seat type is: %s", "Seater");
      }
      printf("\nYour chosen date of travel is: %s\n", date);
      printf("\nPress 1 to continue to payment or press any other key to go back to main menu.\n");
      scanf("%d", &wantToPay);
      if (wantToPay == 1)
      {
        paymentGateway(trainId, ticketsToBuy, choosenCompartment, totalPrice, ticketsNums, date);
      } else
      {
        mainMenu();
      }
      break;

    case 5:
      printf("You will be redirected to Train Section in 3 seconds.\n");
      sleepProgram(3);
      findTrain(trainId);
      break;

    case 6:
      mainMenu();
      break;

    default:
      mainMenu();
      break;
    }
}
```

//Function to confirm tickets after payment

```
void confirmTickets(const char* trainId, int ticketsToBuy, int choosenCompartment, int* ticketsNums, float totalPrice,
const char* date) {
   int confirmedNowGoBack;
   clearTerminal();
   sleepProgram(2);
   printf("\t\t\t\033[1;31mTrain RESERVATION SUCCESSFUL\033[0m\t\t");
   sleepProgram(1);
   printf("\nPlease Enter following details to complete the booking:\n");

   // Generate a 10-digit random number as PNR
   long long pnr = generate10DigitRandomNumber();

   // Array to store passenger information
   Reservation passengers[MAX_PASSENGERS];

   // Collect passenger details for each ticket
   for (int i = 0; i < ticketsToBuy; ++i) {
      printf("\nEnter passenger name for Seat %2d: ", i + 1);
      scanf("%s", passengers[i].name);
      printf("Enter passenger gender for Seat %2d: ", i + 1);
      scanf("%s", passengers[i].gender);
      printf("Enter passenger age for Seat %2d: ", i + 1);
      scanf("%d", &passengers[i].age);
      printf("Enter passenger Mobile Number for Seat %2d: ", i + 1);
      scanf("%lld", &passengers[i].phone);

      // Set common information for each passenger
      strcpy(passengers[i].userId, userIdPtr);
      strcpy(passengers[i].trainId, trainId);
      passengers[i].compartment = choosenCompartment;
      passengers[i].cost = totalPrice;
      passengers[i].noOfSeats = 1;
      passengers[i].seats = ticketsNums[i];

      // Determine and set seat type based on compartment
      if (choosenCompartment == 1 || choosenCompartment == 2 || choosenCompartment == 3) {
         strcpy(passengers[i].seatType, "(AC)Seater ");
      } else if (choosenCompartment == 4) {
         strcpy(passengers[i].seatType, "Sleeper ");
      } else {
         strcpy(passengers[i].seatType, "Seater ");
      }

      // Set additional information
      strcpy(passengers[i].status, "CONFIRMED");
      passengers[i].pnr = pnr;
      strcpy(passengers[i].date, date);
```

```
        // Write reservation information to a file
        writeReservationToFile(&passengers[i]);
    }

    // Print confirmation message and details
    PrintSleep(0.18);
    printf("\nThank You For Your Details. Your PNR is: %lld\n", pnr);
    redColor();
    printf("  TICKET BOOKING DETAILS:  ");
    resetColor();
    printf("\nTrain ID: %s\n", trainId);
    printf("Compartment: %d\n", choosenCompartment);
    printf("Total Tickets: %d\n", ticketsToBuy);
    printf("Total Cost: %.2f\n", totalPrice);
    printf("Your chosen date of travel is: %s\n", date);
    printf("Seat Numbers: ");
    for (int i = 0; i < ticketsToBuy; i++) {
        printf("%d, \t", ticketsNums[i]);
    }
    printf("\nYour PNR is %lld\n", pnr);
    printf("\n");
    sleepProgram(1);
    printf("\nYou will be redirected to reservation section in 5 seconds!!\n");
    sleepProgram(5);

    // View reservation information
    seeReservationInfo(pnr);
}
```

//Function to write reservation to file after confirming tickets

```
void writeReservationToFile(const Reservation* passenger) {
   FILE* file = fopen("files/reservations.txt", "a");
   if (file == NULL) {
      printf("Error opening reservations file.\n");
      exit(1);
   }
      fprintf(file, "%s %s %s %d %.2f %d %d %s %s %lld %s %s %d %lld\n", passenger->name, passenger->userId, 
passenger->trainId,
          passenger->compartment, passenger->cost, passenger->noOfSeats, passenger->seats, passenger->seatType, 
passenger->status, passenger->pnr, passenger->date, passenger->gender, passenger->age, passenger->phone);

   fclose(file);
}
```

//function to check reservation info using PNR

```
void reservationInfo() {
   long long pnrInfo;
   clearTerminal();
   printf("\nEnter PNR: ");
   scanf("%lld", &pnrInfo);
   PrintSleep(0.18);

   FILE* reserFile = fopen("files/reservations.txt", "r");
   if (reserFile == NULL) {
      printf("Error opening reservations file for reading.\n");
      exit(1);
   }
   FILE* trainDetails = fopen("files/train.txt", "r");
   if (trainDetails == NULL) {
      printf("Error opening trains file for reading.\n");
      exit(1);
   }

   int numTrains;
   fscanf(trainDetails, "%d", &numTrains);

   Reservation currentReservation;
   int reservationFound = 0;
   int l = 0;
   Train train;

   while (fscanf(reserFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld", currentReservation.name, 
currentReservation.userId,
          currentReservation.trainId, &currentReservation.compartment, &currentReservation.cost,
          &currentReservation.noOfSeats, &currentReservation.seats , currentReservation.seatType, 
currentReservation.status, &currentReservation.pnr, currentReservation.date, currentReservation.gender, 
&currentReservation.age, &currentReservation.phone) != EOF) {

      if (currentReservation.pnr == pnrInfo) {
            while (fscanf(trainDetails, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint, 
train.destination, train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed, 
&train.totalDist) != EOF) {
            if (strcmp(currentReservation.trainId, train.trainId) == 0) {
         reservationFound = 1;
         l++;
         sleepProgram(1);

         printf("\t\t||\t Passenger Name: %-25s        ||\n", currentReservation.name);
         printf("\t\t||\t Passenger Age: %-25d         ||\n", currentReservation.age);
         printf("\t\t||\t Passenger Gender: %-25s      ||\n", currentReservation.gender);
         printf("\t\t||\t Passenger Phone: %-27lld     ||\n", currentReservation.phone);
         printf("\t\t||\t User ID: %-25s               ||\n", currentReservation.userId);
         printf("\t\t||\t Train ID: %-25s              ||\n", currentReservation.trainId);
```

```
            printf("\t\t||\t Train Name: %-25s            ||\n", train.name);
            printf("\t\t||\t Train Starting Point: %-25s  ||\n", train.startingPoint);
            printf("\t\t||\t Train Destination: %-25s     ||\n", train.destination);
            printf("\t\t||\t Compartment: %-25d           ||\n", currentReservation.compartment);
            printf("\t\t||\t Cost: %-27.2f                ||\n", currentReservation.cost);
            printf("\t\t||\t Number of Seats: %-25d       ||\n", currentReservation.noOfSeats);
            printf("\t\t||\t Seat Numbers: %-25d          ||\n", currentReservation.seats);
            printf("\t\t||\t Seat Type: %-25s             ||\n", currentReservation.seatType);
            printf("\t\t||\t Status: %-25s                ||\n", currentReservation.status);
            printf("\t\t||\t Date Of Departure: %-25s     ||\n", currentReservation.date);
            printf("\t\t||\t PNR: %-27lld                 ||\n", currentReservation.pnr);
            fseek(trainDetails, 0, SEEK_SET);
            fscanf(trainDetails, "%d", &numTrains);
            break;
              }
            }
          }
        }
        fclose(trainDetails);

        fclose(reserFile);

        if (!reservationFound) {
          printf("\nReservation with PNR %lld not found.\n", pnrInfo);
          printf("\nPress 1 to try again or any other key to go back to the main menu.\n");
          int reservationPrintChoice;
          scanf("%d", &reservationPrintChoice);
          if (reservationPrintChoice == 1) {
            reservationInfo();
            while (getchar() != '\n');
          } else {
            mainMenu();
          }
        }

        printf("\nPress 1 to check another reservation or Press any key to go back to the main menu.\n");
        int reservationFoundPrintChoice;
        scanf("%d", &reservationFoundPrintChoice);
        if (reservationFoundPrintChoice == 1) {
          reservationInfo();
          while (getchar() != '\n');
          printf("\nYou will be redirected to main menu in 5 seconds!\n");
          sleepProgram(5);
          mainMenu();
        } else {
          mainMenu();
        }
      }
```

//function to cancel booking

```
void cancelBooking() {
   long long pnrCancel;

   // Get PNR from user input
   printf("\nEnter PNR: ");
   scanf("%lld", &pnrCancel);

   // Display reservation details before cancellation
   PrintToBeDeletedReservation(pnrCancel);

   // Clear terminal screen
   clearTerminal();

   // Display cancellation message
   printf("\t\t\t\033[1;31mTrain RESERVATION CANCELLED\033[0m\t\t");
   printf("\nYou have cancelled ticket booking.\n");
   printf("\nYou will be redirected to the main menu in 3 seconds\n");

   // Introduce a delay before redirecting to the main menu
   sleepProgram(3);

   // Redirect to the main menu
   mainMenu();
}
```

//function to delete reservation by pnr

```
void deleteReservationByPnr(long long pnr) {
  // Open the original reservations file for reading
  FILE* reserFile = fopen("files/reservations.txt", "r");
  if (reserFile == NULL) {
    printf("Error opening reservations file for reading.\n");
    exit(1);
  }
  // Open a temporary file for writing
  FILE* tempFile = fopen("files/temp.txt", "w");
  if (tempFile == NULL) {
    printf("Error opening temporary file for writing.\n");
    exit(1);
  }
  // Initialize a variable to store the current reservation
  Reservation currentReservation;
  // Loop through the original reservations file
  while (fscanf(reserFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld",
         currentReservation.name, currentReservation.userId,
         currentReservation.trainId, &currentReservation.compartment,
         &currentReservation.cost, &currentReservation.noOfSeats,
         &currentReservation.seats, currentReservation.seatType,
         currentReservation.status, &currentReservation.pnr,
         currentReservation.date, currentReservation.gender,
         &currentReservation.age, &currentReservation.phone) != EOF) {
    // Check if the PNR matches the reservation to be deleted
    if (currentReservation.pnr == pnr) {
      // Skip this reservation, effectively deleting it
      continue;
    }
    // Write the non-deleted reservation to the temporary file
    fprintf(tempFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld\n",
        currentReservation.name, currentReservation.userId,
        currentReservation.trainId, currentReservation.compartment,
        currentReservation.cost, currentReservation.noOfSeats,
        currentReservation.seats, currentReservation.seatType,
        currentReservation.status, currentReservation.pnr,
        currentReservation.date, currentReservation.gender,
        currentReservation.age, currentReservation.phone);
  }
  // Close both files
  fclose(reserFile);
  fclose(tempFile);

  // Remove the original reservations file
  remove("files/reservations.txt");

  // Rename the temporary file to the original reservations file
  rename("files/temp.txt", "files/reservations.txt");
}
```

//Function to get ticket details which then will be used to print pdf

```
void getTicketFormat()  {
   long long pnrInfo;
   clearTerminal();
   printf("\nEnter PNR: ");
   scanf("%lld", &pnrInfo);
   PrintSleep(0.18);
   printf("\nGetting Ticket Format!!!!\n\n");
   PrintSleep(0.18);

   FILE* reserFile = fopen("files/reservations.txt", "r");
   if (reserFile == NULL) {
      printf("Error opening reservations file for reading.\n");
      exit(1);
   }
   FILE* trainDetails = fopen("files/train.txt", "r");
   if (trainDetails == NULL) {
      printf("Error opening trains file for reading.\n");
      exit(1);
   }

   int numTrains;
   fscanf(trainDetails, "%d", &numTrains);

   Reservation currentReservation;
   int reservationFound = 0;
   int l = 0;
   Train train;

   Ticket ticket;

   while (fscanf(reserFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld", currentReservation.name,
currentReservation.userId,
           currentReservation.trainId, &currentReservation.compartment, &currentReservation.cost,
           &currentReservation.noOfSeats, &currentReservation.seats , currentReservation.seatType, currentReservation.status,
&currentReservation.pnr, currentReservation.date, currentReservation.gender, &currentReservation.age,
&currentReservation.phone) != EOF) {

      if (currentReservation.pnr == pnrInfo) {
            while (fscanf(trainDetails, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint, train.destination,
train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed, &train.totalDist) !=
EOF) {
            if (strcmp(currentReservation.trainId, train.trainId) == 0) {
               reservationFound = 1;
               l++;
               ticket.passengerTickets[l] = malloc(sizeof(PassengerTicket));
               if (ticket.passengerTickets[l] == NULL) {
               printf("Memory allocation failed.\n");
               return;
```

```
                }
                strcpy(ticket.StartingPoint, train.startingPoint);
                strcpy(ticket.Destination, train.destination);
                snprintf(ticket.PNR, sizeof(ticket.PNR), "%lld", pnrInfo);
                strcpy(ticket.TrainName, train.name);
                strcpy(ticket.TrainId, train.trainId);

                if (currentReservation.compartment == 1 || currentReservation.compartment  == 2 ||
currentReservation.compartment  == 3)
                {
                strcpy(ticket.class, "AC SEATER");
                } else if (currentReservation.compartment  == 4)
                {
                strcpy(ticket.class, "Sleeper");
                } else
                {
                strcpy(ticket.class, "Seater");
                }

                strcpy(ticket.Date, currentReservation.date);
                snprintf(ticket.dist, sizeof(ticket.dist), "%d", train.totalDist);
                snprintf(ticket.Cost, sizeof(ticket.Cost), "%.2f", currentReservation.cost);
                snprintf(ticket.Compartment, sizeof(ticket.Compartment), "%d", currentReservation.compartment);

                strcpy(ticket.passengerTickets[l]->passName, currentReservation.name);
                snprintf(ticket.passengerTickets[l]->age, sizeof(ticket.passengerTickets[l]->age), "%d", currentReservation.age);
                strcpy(ticket.passengerTickets[l]->gender, currentReservation.gender);
                snprintf(ticket.passengerTickets[l]->seatNum, sizeof(ticket.passengerTickets[l]->seatNum), "%d",
currentReservation.seats);

                ticket.totalSeats = l;

                fseek(trainDetails, 0, SEEK_SET);
                fscanf(trainDetails, "%d", &numTrains);
                break;
            }
        }
    }
}

fclose(trainDetails);

fclose(reserFile);

if (!reservationFound) {
    printf("\nReservation with PNR %lld not found.\n", pnrInfo);
    printf("\nPress 1 to try again or any other key to go back to the main menu.\n");
    int reservationPrintChoice;
    scanf("%d", &reservationPrintChoice);
    if (reservationPrintChoice == 1) {
        reservationInfo();
    } else {
        mainMenu();
    }
}

printf("\nPress 1 to print reservation or Press any key to go back to the main menu.\n");
int reservationFoundPrintChoice;
scanf("%d", &reservationFoundPrintChoice);
if (reservationFoundPrintChoice == 1) {
    writeTicketPDF(&ticket);
} else {
    mainMenu();
}
}
```

//Function to write pdf

```
int writeTicketPDF(const Ticket* ticket){
    struct pdf_doc *pdf = pdf_create(PDF_A4_WIDTH, PDF_A4_HEIGHT, &info);

    if (!pdf created) {
        fprintf(stderr, "Unable to create PDF\n");
        return -1;
    }
    pdf_append_page(pdf);
    print all page 1 data using proper co-ordinates;
    pdf_append_page(pdf);
    print all page 2 data using proper co-ordinates;


   example: pdf_add_text(pdf, NULL, "DATA TO BE ENTERED", font-size, x co-ordinates,, y co-ordinates, PDF_RGB(30, 40,
50));



    char extension[] = ".pdf";
    char ticketName[100];
    sprintf(ticketName, "tickets/%s%s%s%s", ticket->PNR, "_", ticket->Destination, extension);
    pdf_save(pdf, ticketName);
    printf("\n\nTicket Generated \n");
    printf("\nTicket saved to folder \"Tickets\" with name: %s\n", ticketName);
    printf("\nYou will be redirected to main menu in 5 seconds!\n");
    mainMenu();
}return;tf("\nPress 1 to check anoth{
 nd to pays, date);

}
```

///CODE BELOW//

```
/*
Group ID= 07

Members:
   1)Arunabh Aditya          Roll Number: 114026   PRN:1032230178
   2)M Aryane                Roll Number: 114029   PRN:1032232583
   3)Akshay Anurag           Roll Number: 114028   PRN:1032231965
   4)Fardeen Ali             Roll Number: 114027   PRN:1032232038
   5)Ayush Gaikwad           Roll Number: 114039   PRN:1032232473
   6)Shriharsh Deshmukh       Roll Number: 114038   PRN:1032231397

Problem Statement:
write a C program to reserve train tickets with following functionality:
   1)User System:
      i)Login
      ii)signup
      iii)Reset Password
   2)Admin Controls
      i)add new trains
      ii)add delayed trains
      iii)see delayed trains
   3)Tickets reservation:
           i)With option to select preferred class, compartment,seat type (seater or sleeper) and number of seats
           ii)With payment system.(with payment failure handling case).

           iii)Once reserved, get PNR.
   4)Get Information of reservations (By entering PNR):
           i)To know in which compartment, class seat is reserved.
           ii)To know if ticket is confirmed or in waiting list.
           iii)Get reservation id or referrence number of the reservation.
           iv)To know all about train (max speed, total dist to be travelled )
      V)Option to print tickets in PDF format
   5)Checking Train Status:
           i)To check delayed trains
   6)Option to cancel tickets:
           i)option to cancel ticket.
           ii)with proper refund messages.
   ***ADDITIONAL FEATURES***
   1)QR code payment format
   2)User profile update
   3)Admin panel
   4)ability to cancel tickets or go back to main menu at any step
   5)Proper error handling.
   6)proper user navigation allowing a smoother program experience.
   7)Proper PDF generation.
   8)Proper user interface allowing user to perform all actions without having to restart the program.
   9)Menu Based program.
   *****CAUTIONS*****
   1)Input should be of the type that is asked.
   2)All input should be in lowercase (unless stated otherwise and as per choice for name and other user details).

Input Required:
   as per the program flow and user wishes.
```

```
Algorithms used (module use):
   1)External library:
      i)qrcodegen.h
      ii)pdfgen.h
   2)Internal header files:
      i)stdio.h
      ii)stdlib.h
      iii)conio.h
      iv)string.h
      v)time.h
      vi)windows.h
      vii)ctype.h

Conclusion:
   Thus implemented a complete error safe train ticket booking application with TUI (terminal user interface) allowing a
menu based program.
*/


#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <time.h>
#include <windows.h>
#include <ctype.h>
#include "header/qrcodegen.h"
#include "header/pdfgen.h"

#define MAX_PASSENGERS 100


typedef struct {
   char userId[10];
   char email[100];
   char password[100];
   char name[100];
   int age;
   char gender[10];
   long long phone;
} User;

typedef struct {
   char trainId[10];
   char startingPoint[100];
   char destination[100];
   char departureTime[100];
   int cost;
   int compartments;
   char seatType[100];
   char name[100];
   int maxSpeed;
```

```
    int totalDist;
} Train;

typedef struct {
    long long pnr;
    char userId[10];
    char name[100];
    char trainId[10];
    int compartment;
    char seatType[10];
    char status[10];
    float cost;
    int seats;
    int noOfSeats;
    char date[20];
    int age;
    char gender[10];
    long long phone;
} Reservation;

typedef struct {
    char trainId[20];
    char startingPoint[20];
    char destination[20];
    char departureTime[15];
    int cost;
    int compartments;
    char seatType[20];
    int isDelayed;
    int delayTime;
} DelayedTrain;

typedef struct PassengerTicket PassengerTicket;


typedef struct {
    char StartingPoint[20];
    char Destination[20];
    char PNR[20];
    char TrainName[20];
    char TrainId[20];
    char class[20];
    char Date[20];
    char dist[20];
    char Cost[20];
    char Compartment[20];
    int totalSeats;
    PassengerTicket* passengerTickets[10];
} Ticket;

struct PassengerTicket {
    char passName[20];
    char age[20];
```

```c
    char gender[20];
    char seatNum[20];
};



int delayMilliseconds = 5000;
int logged = 0;
int *loggedPtr = &logged;
char userId[10];
char *userIdPtr = userId;

void sleepProgram(float seconds);
void PrintSleep(float seconds);
void clearTerminal();
void greenColor();
void redColor();
void resetColor();
void yellowColor();
void signup();
int userExists(const char* email);
void login();
void copyFile(const char *source, const char *destination);
void resetPass();
void LogOrSign();
void adminLogin();
void addTrain();
void displayAllTrains();
int compareByCostAsc(const void* a, const void* b);
int compareByCostDesc(const void* a, const void* b);
void sortByCost(int sortOrder);
void findTrainsByDestination(const char* destination);
void findTrainsByStartingPoint(const char* startingPoint);
void adminControls();
void addDelayedTrain();
void displayDelayedTrains(int isAdmin);
void trainListandBook();
void findTrain(const char* trainId);
void showCompartment( int compartment, const char* trainId);
void randomlyBookSeats(int* seats, int numSeats);
int isValidDate(const char *dateStr);
void payNow(const char* trainId, int ticketsToBuy, int choosenCompartment, int ticketsNums[MAX_PASSENGERS], const
char* date);
void showTickets(int choosenCompartment, const char* trainId);
int isAllDigits(const char *str);
void paymentGateway(const char* trainId, int ticketsToBuy, int choosenCompartment, float totalPrice, int
ticketsNums[MAX_PASSENGERS], const char* date);
long long generate10DigitRandomNumber();
long long generate15DigitRandomNumber();
void writeReservationToFile(const Reservation* passenger);
void confirmTickets(const char* trainId, int ticketsToBuy, int choosenCompartment, int* ticketsNums, float totalPrice,
const char* date);
void PrintToBeDeletedReservation(long long pnr);
void deleteReservationByPnr(long long pnr);
void cancelBooking();
void seeReservationInfo(long long pnrInfo);
void reservationInfo();
void getTicketFormat();
void getTicketFormatAfterBooking(long long pnrInfo);
int writeTicketPDF(const Ticket* ticket);
void checkAllReservations();
void showUserInfo();
void changeUserInfo();
void mainMenu();




void sleepProgram(float seconds)
{
  Sleep(seconds * 1000);
}

void PrintSleep(float seconds){
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("========");
    sleepProgram(seconds);
    printf("======\n");
}

void clearTerminal(){
    system("cls");
}

void greenColor()
{
  printf("\033[1;32m");
}

void redColor()
{
  printf("\033[1;31m");
}

void resetColor()
{
  printf("\033[0m");
}

void yellowColor() {
  printf("\033[1;33m");
}

void signup() {
    clearTerminal();
    int SignedUp, AlrExists;

    FILE *file = fopen("files/users.txt", "a");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    User newUser;
    printf("Please enter your name: ");
    scanf("%s", newUser.name);

    printf("Please enter your Age: ");
    scanf("%d", &newUser.age);

    printf("Please enter your Gender: ");
    scanf("%s", newUser.gender);

    printf("Please enter your Phone: ");
    scanf("%lld", &newUser.phone);

    printf("Please enter your email: ");
    scanf("%s", newUser.email);

    if (userExists(newUser.email)) {
        printf("\nUser with the same email already exists. Press 1 to log in. Press 2 to try again\n");
          scanf("%d", &AlrExists);
          if (AlrExists == 1) {
            login();
          } else if (AlrExists == 2) {
            signup();
          }
        return;
    }
    printf("Please enter your password: ");
    scanf("%s", newUser.password);

    sprintf(newUser.userId, "%08d", rand() % 100000000);

    fprintf(file, "%s %s %s %s %d %s %lld\n", newUser.userId, newUser.email, newUser.password, newUser.name,
newUser.age, newUser.gender, newUser.phone);
    printf("\nNew user created successfully!\n");
    printf("\nHere are user Details:\n");
    printf("User ID: %s\n", newUser.userId);
    printf("Email: %s\n", newUser.email);
    printf("Name: %s\n", newUser.name);
    printf("Age: %d\n", newUser.age);
    printf("Gender: %s\n", newUser.gender);
    printf("Phone Number: +91 %lld\n", newUser.phone);
    fclose(file);

    printf("\nAccount created successfully! Please log in.\n");
    printf("Press 1 to login or press 2 to create another account:  ");
    scanf("%d", &SignedUp);
    if (SignedUp == 1) {
      login();
    } else if (SignedUp == 2) {
      signup();
    }
}

int userExists(const char* email) {
    FILE *file = fopen("files/users.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    char line[500];
    while (fgets(line, sizeof(line), file) != NULL) {
        User user;
        sscanf(line, "%s %s %s %s %d %s %lld", user.userId, user.email, user.password, user.name, &user.age, user.gender,
&user.phone);

        if (strcmp(user.email, email) == 0) {
            fclose(file);
            return 1;
        }
    }

    fclose(file);
    return 0;
}

void login() {
    int wrongPass, wrongUser;
    char email[100];
    char password[100];
    char line[500];

    clearTerminal();

printf("\n\n\n\n\n===================================================================================
===========\n\n");
    printf("\t\t\t\033[1;31mPlease Login:\033[0m\t\t");

printf("\n\n==========================================================================================
=====\n");
    printf("\nPlease enter your email:");
    scanf("%s", email);
    getchar();

    FILE *file = fopen("files/users.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    int userFound = 0;
    while (fgets(line, sizeof(line), file) != NULL) {
        User user;
        sscanf(line, "%s %s %s %s %d %s %lld", user.userId, user.email, user.password, user.name, &user.age, user.gender,
&user.phone);

        if (strcmp(user.email, email) == 0) {
            userFound = 1;
            printf("\nUser Found\n");
            printf("Please enter your password: ");
            scanf("%s", password);
            getchar();
            if (strcmp(user.password, password) == 0) {
                printf("\nLogin successful!\n");
                *loggedPtr = 1;
                strcpy(userIdPtr, user.userId);
                yellowColor();
                printf("\n\t\t\tYou are logged in and can proceed to book tickets!!");
                resetColor();
                sleepProgram(3);
                mainMenu();
                break;
            } else {
                printf("\nIncorrect password. Please try again.\n");
                printf("\nPress 1 to try again or press 2 to reset password!\n");
                scanf("%d", &wrongPass);
                if (wrongPass == 1) {
                    login();
                } else if (wrongPass == 2) {
                    resetPass();
                }
            }
            break;
        }
    }
    fclose(file);
    if (!userFound) {
        printf("\nUser not found. Please create an account.\n");
        printf("\nPress 1 to create an account or press 2 to try again!\n");
        scanf("%d", &wrongUser);
        if (wrongUser == 1) {
            signup();
        } else if (wrongUser == 2) {
            login();
        }
    }
}

void copyFile(const char *source, const char *destination) {
    FILE *src = fopen(source, "rb");
    FILE *dest = fopen(destination, "wb");

    if (src == NULL || dest == NULL) {
        printf("Error opening files for copying.\n");
        return;
    }

    char ch;
    while ((ch = fgetc(src)) != EOF) {
        fputc(ch, dest);
    }

    fclose(src);
    fclose(dest);
}

void resetPass() {
    int resetNoUser, resetDoneUser;
    char email[100];
    char newPassword[100];
    char line[500];
    clearTerminal();
    printf("\nPlease enter your email: ");
    scanf("%s", email);
    getchar();
    FILE *file = fopen("files/users.txt", "r");

    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    FILE *tempFile = fopen("files/temp.txt", "w");
    if (tempFile == NULL) {
        printf("Error creating temporary file.\n");
        fclose(file);
        exit(1);
    }

    int userFound = 0;
    while (fgets(line, sizeof(line), file) != NULL) {
        User user;
        sscanf(line, "%s %s %s %s %d %s %lld", user.userId, user.email, user.password, user.name, &user.age, user.gender,
&user.phone);

        if (strcmp(user.email, email) == 0) {
            userFound = 1;
            printf("\nUser Found\n");
            printf("Please enter your new password: ");
            scanf("%s", newPassword);
            getchar();

            strcpy(user.password, newPassword);
        }

        fprintf(tempFile, "%s %s %s %s %d %s %lld\n", user.userId, user.email, user.password, user.name, user.age,
user.gender, user.phone);
    }

    fclose(file);
    fclose(tempFile);
    remove("files/users.txt");
    fclose(fopen("files/users.txt", "w"));
    copyFile("files/temp.txt", "files/users.txt");
    remove("files/temp.txt");

    if (!userFound) {
        printf("\nUser not found. Please create an account.\n");
        printf("\nPress 1 to create an account or press 2 to try again!\n");
        scanf("%d", &resetNoUser);

        if (resetNoUser == 1) {
            signup();
        } else if (resetNoUser == 2) {
            resetPass();
        }
    } else {
        printf("\nPassword Reset Successful!!");

printf("\n\n===============================================================================
=====\n");
        printf("\n Do you want to continue to login? or signup? or exit?");
        printf("\nPress 1 to login, Press 2 to signup, Press any other key to exit!\n");
        scanf("%d", &resetDoneUser);

        if (resetDoneUser == 1) {
            login();
        } else if (resetDoneUser == 2) {
            signup();
        } else {
            exit(0);
        }
    }
}

void LogOrSign(){
    int LogOrSignOpt;
    clearTerminal();

printf("\n\n\n\n\n=======================================================================================
===========\n\n");
    printf("Please Login or Signup to continue booking tickets");

printf("\n\n=============================================================================================
=====\n\n");
    printf("  Select from the following options:\n\n");
    printf("   \033[1;31m[1]\033[0m Login \n\n");
    printf("   \033[1;31m[2]\033[0m Signup\n\n");
    printf("   \033[1;31m[3]\033[0m Reset Password\n\n\n\n");
    printf("   \033[1;31m[4]\033[0m ADMIN LOGIN\n\n");
    scanf("%d", &LogOrSignOpt);

    switch (LogOrSignOpt)
    {
    case 1:
        login();
        break;

    case 2:
        signup();
        break;
    case 3:
        resetPass();
        break;
    case 4:
        adminLogin();
        break;
    default:
        printf("Please Choose Correct Option");
        sleepProgram(1);
        LogOrSign();
        break;
    }
}

void adminLogin(){
    int AdminLoginOpt;
    clearTerminal();
    redColor();
    char adminPass[100];
    strcpy(adminPass, "G7");

printf("\n\n\n\n\n===================================================================================
===========\n\n");
    printf("Please Login With ADMIN PASS to continue to ADMIN CONTROLS");

printf("\n\n=========================================================================================
=====\n\n");
    printf("Print Admin PASS (case-sensitive)[ \033[1;32m It is G7 :) \033[1;31m ] :");
    scanf("%s", adminPass);
    PrintSleep(0.18);
    getchar();
    if (strcmp(adminPass, "G7") == 0) {

printf("\n\n\n\n\n===================================================================================
===========\n\n");
        printf("Welcome to ADMIN CONTROLS");

printf("\n\n=========================================================================================
=====\n\n");
        printf("You will be redirected to ADMIN CONTROLS in 3 seconds!");
        sleepProgram(3);
        adminControls();
    } else {

printf("\n\n\n\n\n===================================================================================
===========\n\n");
        printf("Incorrect Password");

printf("\n\n=========================================================================================
=====\n\n");
        printf("Do you want to try again? press 1 to try again, press any other key to return to login");
        scanf("%d", &AdminLoginOpt);
        if (AdminLoginOpt == 1) {
            sleepProgram(1);
            adminLogin();
        } else {
            sleepProgram(1);
            LogOrSign();
        }
    }

}

void addTrain() {
    FILE *file = fopen("files/train.txt", "r+");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    int numTrains;
    fscanf(file, "%d", &numTrains);
    numTrains++;

    fseek(file, 0, SEEK_SET);

    fprintf(file, "%d\n", numTrains);

    fseek(file, 0, SEEK_END);

    Train newTrain;
    printf("\nPlease enter the Train ID: ");
    scanf("%s", newTrain.trainId);

    printf("Please enter the Starting Point: ");
    scanf("%s", newTrain.startingPoint);

    printf("Please enter the Destination: ");
    scanf("%s", newTrain.destination);

    printf("Please enter the Departure Time: ");
    scanf("%s", newTrain.departureTime);

    printf("Please enter the Cost: ");
    scanf("%d", &newTrain.cost);

    printf("Please enter number of Compartments: ");
    scanf("%d", &newTrain.compartments);

    printf("Please enter Seat type:- sleeper or seater: ");
    scanf("%s", newTrain.seatType);

    printf("Please enter Name of Train: ");
    scanf("%s", newTrain.name);

    printf("Please enter max speed of train: ");
    scanf("%d", &newTrain.maxSpeed);

    printf("Please enter total distance to be travelled by train: ");
    scanf("%d", &newTrain.totalDist);

    fprintf(file, "%s %s %s %s %d %d %s %s %d %d\n", newTrain.trainId, newTrain.startingPoint, newTrain.destination,
newTrain.departureTime, newTrain.cost, newTrain.compartments, newTrain.seatType, newTrain.name,
newTrain.maxSpeed, newTrain.totalDist);

    fclose(file);

    printf("\nTrain added successfully!\n");


    printf("\nPress 1 to add another train or press any other key to go to admin controls.\n");

    int addAnotherTrain;
    scanf("%d", &addAnotherTrain);
    if (addAnotherTrain == 1) {
        sleepProgram(1);
        addTrain();
    } else {
        sleepProgram(1);
        adminControls();
    }

}

void displayAllTrains() {
    clearTerminal();
    FILE *file = fopen("files/train.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    int numTrains;
    fscanf(file, "%d", &numTrains);


printf("\n=============================================================================================================
=================================================================================================================\n");
    printf("| %-10s | %-20s | %-20s | %-15s | %-5s | %-20s | %-20s | %-20s | %-5s | %-5s | ", "Train ID", "Starting Point",
"Destination", "Departure Time", "Cost", "Compartments", "Seat Type", "Train Name", "Max Speed", "Total Distance");

printf("\n=============================================================================================================
=================================================================================================================\n");

    for (int i = 0; i < numTrains; i++) {
        Train train;
        fscanf(file, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint, train.destination,
train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed, &train.totalDist);

        printf("| %-10s | %-20s | %-20s | %-15s | %-5d | %-20d | %-20s | %-20s | %-5dKM/H | %-12dKM |\n", train.trainId,
train.startingPoint, train.destination, train.departureTime, train.cost, train.compartments, train.seatType, train.name,
train.maxSpeed, train.totalDist);
    }


printf("===============================================================================================================
===============================================================================================================\n");
    fclose(file);
}

int compareByCostAsc(const void* a, const void* b) {
    return ((Train*)a)->cost - ((Train*)b)->cost;
}

int compareByCostDesc(const void* a, const void* b) {
    return ((Train*)b)->cost - ((Train*)a)->cost;
}
```

```
void sortByCost(int sortOrder) {
    FILE* file = fopen("files/train.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    int numTrains;
    fscanf(file, "%d", &numTrains);

    Train* trains = (Train*)malloc(numTrains * sizeof(Train));
    if (trains == NULL) {
        printf("Memory allocation failed.\n");
        fclose(file);
        exit(1);
    }

    for (int i = 0; i < numTrains; i++) {
        fscanf(file, "%s %s %s %s %d %d %s %s %d %d", trains[i].trainId, trains[i].startingPoint, trains[i].destination,
trains[i].departureTime, &trains[i].cost, &trains[i].compartments, trains[i].seatType, trains[i].name, &trains[i].maxSpeed,
&trains[i].totalDist);
    }

    fclose(file);

    if (sortOrder == 2) {
        qsort(trains, numTrains, sizeof(Train), compareByCostDesc);
    } else {
        qsort(trains, numTrains, sizeof(Train), compareByCostAsc);
    }


printf("\n==========================================================================================================
================================================================================================================\n");
    printf("| %-10s | %-20s | %-20s | %-15s | %-5s | %-20s | %-20s | %-20s | %-5s | %-5s | ", "Train ID", "Starting Point",
"Destination", "Departure Time", "Cost", "Compartments", "Seat Type", "Train Name", "Max Speed", "Total Distance");

printf("\n==========================================================================================================
================================================================================================================\n");

    for (int i = 0; i < numTrains; i++) {
        printf("| %-10s | %-20s | %-20s | %-15s | %-5d | %-20d | %-20s | %-20s | %-5dKM/H | %-12dKM |\n",
trains[i].trainId, trains[i].startingPoint, trains[i].destination, trains[i].departureTime, trains[i].cost, trains[i].compartments,
trains[i].seatType, trains[i].name, trains[i].maxSpeed, trains[i].totalDist);
    }

printf("\n==========================================================================================================
================================================================================================================\n");

    printf("\nPress 1 to continue to booking or press 2 to go back to main menu: ");
    int afterSortChoice;
    scanf("%d", &afterSortChoice);
    if (afterSortChoice == 1) {
        printf("Please enter the train id: ");
        getchar();
        char trainId[10];
        scanf("%s", trainId);
        findTrain(trainId);
    } else{
        sleepProgram(1);
        mainMenu();
    }
    free(trains);
}

void findTrainsByDestination(const char* destination) {
    FILE* file = fopen("files/train.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    int numTrains;
    fscanf(file, "%d", &numTrains);


printf("\n=====================================================================================================
==========================================================================================================\n");
    printf("| %-10s | %-20s | %-20s | %-15s | %-5s | %-20s | %-20s | %-20s | %-5s | %-5s | ", "Train ID", "Starting Point",
"Destination", "Departure Time", "Cost", "Compartments", "Seat Type", "Train Name", "Max Speed", "Total Distance");

printf("\n=====================================================================================================
==========================================================================================================\n");

    Train train;
    int found = 0;

    while (fscanf(file, "%s %s %s %s %d %d %s", train.trainId, train.startingPoint, train.destination, train.departureTime,
&train.cost, &train.compartments, train.seatType) != EOF) {
        if (strcmp(train.destination, destination) == 0) {
            found = 1;
            printf("| %-10s | %-20s | %-20s | %-15s | %-5d | %-20d | %-20s | %-20s | %-5dKM/H | %-12dKM |\n",
train.trainId, train.startingPoint, train.destination, train.departureTime, train.cost, train.compartments, train.seatType,
train.name, train.maxSpeed, train.totalDist);
        }
    }

printf("\n=====================================================================================================
==========================================================================================================\n");

    fclose(file);

    if(found){
        printf("\nPress 1 to continue to booking or press 2 to go back to main menu: ");
        int afterSortChoice;
        scanf("%d", &afterSortChoice);
        if (afterSortChoice == 1) {
            printf("Please enter the train id: ");
            getchar();
            char trainId[10];
            scanf("%s", trainId);
            findTrain(trainId);
        } else{
            sleepProgram(1);
            mainMenu();
        }
    }

    if (!found){
        printf("No trains found for the destination: %s\n", destination);
        printf("You will be redirected to main menu in 3 seconds.\n");
        sleepProgram(3);
        mainMenu();
    }
}

void findTrainsByStartingPoint(const char* startingPoint) {
    FILE* file = fopen("files/train.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    int numTrains;
    fscanf(file, "%d", &numTrains);


printf("\n============================================================================================================
==============================================================================================================\n");
    printf("| %-10s | %-20s | %-20s | %-15s | %-5s | %-20s | %-20s | %-20s | %-5s | %-5s | ", "Train ID", "Starting Point",
"Destination", "Departure Time", "Cost", "Compartments", "Seat Type", "Train Name", "Max Speed", "Total Distance");

printf("============================================================================================================
============================================================================================================\n");

    Train train;
    int found = 0;

    while (fscanf(file, "%s %s %s %s %d %d %s", train.trainId, train.startingPoint, train.destination, train.departureTime,
&train.cost, &train.compartments, train.seatType) != EOF) {
        if (strcmp(train.startingPoint, startingPoint) == 0) {
            found = 1;
            printf("| %-10s | %-20s | %-20s | %-15s | %-5d | %-20d | %-20s | %-20s | %-5dKM/H | %-12dKM |\n",
train.trainId, train.startingPoint, train.destination, train.departureTime, train.cost, train.compartments, train.seatType,
train.name, train.maxSpeed, train.totalDist);
        }
    }
```

```
printf("\n==============================================================================================================================
====================================================================================================================================\n");

    fclose(file);
    if(found){
        printf("\nPress 1 to continue to booking or press 2 to go back to main menu: ");
        int afterSortChoice;
        scanf("%d", &afterSortChoice);
        if (afterSortChoice == 1) {
            printf("Please enter the train id: ");
            getchar();
            char trainId[10];
            scanf("%s", trainId);
            findTrain(trainId);
        } else{
            sleepProgram(1);
            mainMenu();
        }
    }

    if (!found){
        printf("No trains found for the start point: %s\n", startingPoint);
        printf("You will be redirected to main menu in 3 seconds.\n");
        sleepProgram(3);
        mainMenu();
    }

}

void adminControls() {
    clearTerminal();
    greenColor();

printf("\n==============================================================================================================================
===========================================================\n");
    printf("\n\t\t\tWelcome To Admin Controls\n");
    printf("  *Please Select Appropriate Option:*\n");

printf("\n==============================================================================================================================
===========================================================\n");
    printf("   \033[1;31m[1]\033[0m VIEW TRAIN LIST \n\n");
    printf("   \033[1;31m[2]\033[0m ADD NEW TRAIN\n\n");
    printf("   \033[1;31m[3]\033[0m Add Delayed Trains\n\n");
    printf("   \033[1;31m[4]\033[0m Display Delayed Trains\n\n");
    printf("   \033[1;31m[5]\033[0m GO BACK TO LOGIN SIGNUP WINDOW\n\n");
    int adminChoice;
    scanf("%d", &adminChoice);
    switch (adminChoice) {
    case 1:
        sleepProgram(1);
        displayAllTrains();
        sleepProgram(10);
        adminControls();
        break;
    case 2:
        sleepProgram(1);
        addTrain();
        break;
    case 3:
        sleepProgram(1);
        addDelayedTrain();
        break;
    case 4:
        sleepProgram(1);
        displayDelayedTrains(0);
        break;
    case 5:
        sleepProgram(1);
        LogOrSign();
        break;
    default:
        printf("Oops Wrong Choice");
        adminControls();
        break;
    }

}

void addDelayedTrain() {
    FILE *file = fopen("files/delayed.txt", "a");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    DelayedTrain delayedTrain;
    printf("\nPlease enter the Train ID: ");
    scanf("%s", delayedTrain.trainId);

    printf("Please enter the Starting Point: ");
    scanf("%s", delayedTrain.startingPoint);

    printf("Please enter the Destination: ");
    scanf("%s", delayedTrain.destination);

    printf("Please enter the Departure Time: ");
    scanf("%s", delayedTrain.departureTime);

    printf("Please enter the Cost: ");
    scanf("%d", &delayedTrain.cost);

    printf("Please enter number of Compartments: ");
    scanf("%d", &delayedTrain.compartments);

    printf("Please enter Seat type: sleeper or seater: ");
    scanf("%s", delayedTrain.seatType);

    printf("Is the train delayed? (1 for Yes, 0 for No): ");
    scanf("%d", &delayedTrain.isDelayed);

    if (delayedTrain.isDelayed) {
        printf("Enter delay time (in minutes): ");
        scanf("%d", &delayedTrain.delayTime);
    } else {
        delayedTrain.delayTime = 0;
    }

    fprintf(file, "%s %s %s %s %d %d %s %d %d\n", delayedTrain.trainId, delayedTrain.startingPoint,
delayedTrain.destination, delayedTrain.departureTime, delayedTrain.cost, delayedTrain.compartments,
delayedTrain.seatType, delayedTrain.isDelayed, delayedTrain.delayTime);

    fclose(file);

    printf("\nDelayed train added successfully!\n");
    sleepProgram(3);
    adminControls();
}

void displayDelayedTrains(int isAdmin) {
    FILE *file = fopen("files/delayed.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }
    char delayedTrainID[10];
    printf("Enter Delayed Train ID: ");
    scanf("%s", delayedTrainID);
    int delayedFoundTrain = 0;
    DelayedTrain delayedTrain;
    while (fscanf(file, "%s %s %s %s %d %d %s %d %d", delayedTrain.trainId, delayedTrain.startingPoint,
delayedTrain.destination, delayedTrain.departureTime, &delayedTrain.cost, &delayedTrain.compartments,
delayedTrain.seatType, &delayedTrain.isDelayed, &delayedTrain.delayTime) != EOF) {
        if(strcmp(delayedTrain.isDelayed, delayedTrainID) == 0){
            delayedFoundTrain = 1;

printf("\n=============================================================================================
===================================================================\n");
    printf("| %-10s | %-20s | %-20s | %-15s | %-5s | %-20s | %-20s | %-10s | %-15s |\n", "Train ID", "Starting Point",
"Destination", "Departure Time", "Cost", "Compartments", "Seat Type", "Status", "Delay (min)");

printf("=============================================================================================
=================================================================\n");


        printf("| %-10s | %-20s | %-20s | %-15s | %-5d | %-20d | %-20s | %-10s | %-15d |\n", delayedTrain.trainId,
delayedTrain.startingPoint, delayedTrain.destination, delayedTrain.departureTime, delayedTrain.cost,
delayedTrain.compartments, delayedTrain.seatType, delayedTrain.isDelayed ? "Delayed" : "On Time",
delayedTrain.delayTime);
```

```
  }


printf("=========================================================================================================
===================================================================\n");
  }
  fclose(file);

  if (delayedFoundTrain == 0) {
    printf("Train with id %s is on time!!\n", delayedTrainID);
  }

  printf("\n\nPress 1 to check another train status or press 2 to go back to main menu: ");
  int afterDelayed;
  scanf("%d", &afterDelayed);
  if (afterDelayed == 1) {
    displayDelayedTrains(isAdmin);
  } else {
    if(isAdmin == 1){
      mainMenu();
    }else{
      adminControls();
    }
  }

}

void trainListandBook(){
 displayAllTrains();
    int ifSort;
    printf("\nThese trains are daily available trains!! Enter your preferred date of travel on ticket selection section!! \n");
    printf("Do you want to sort trains? (1 for yes) Or Press 2 for ticket booking Or Press anu other key to go back to main
menu: ");
    scanf("%d", &ifSort);
    if (ifSort == 1) {
      yellowColor();
      printf("\n\nDo you want to sort by cost? (1 for ascending, 2 for descinding) Or Do you want to sort based on
destination or starting point? (3 for destination, 4 for starting point) Or press any other key to go back to train list: ");
      int sortChoice;
      scanf("%d", &sortChoice);
      if (sortChoice == 1) {
        resetColor();
        sortByCost(1);
      }
      else if (sortChoice == 2)
      {
        resetColor();
       sortByCost(2);
      }
      else if(sortChoice == 3)
      {
        char destination[100];
        printf("Enter the destination (first letter capital): ");
```

```
            scanf("%s", destination);
            resetColor();
            findTrainsByDestination(destination);
          }
          else if(sortChoice == 4)
          {
            char startingPoint[100];
            printf("Enter the starting point (first letter capital): ");
            scanf("%s", startingPoint);
            resetColor();
            findTrainsByStartingPoint(startingPoint);
          }
          else{
            resetColor();
            trainListandBook();
          }
        }
        else if (ifSort == 2)
        {
         printf("Please enter the train id: ");
         getchar();
         char trainId[10];
         scanf("%s", trainId);
         findTrain(trainId);
        }
        else{
         sleepProgram(1);
         mainMenu();
        }
}

void findTrain(const char* trainId) {
  int trainNotFound;
    FILE* file = fopen("files/train.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }



    int numTrains;
    fscanf(file, "%d", &numTrains);

    Train train;
    int trainFound = 0;
    PrintSleep(0.18);
    while (fscanf(file, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint, train.destination,
train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed, &train.totalDist)
!= EOF) {
        if (strcmp(train.trainId, trainId) == 0) {
            trainFound = 1;
            clearTerminal();
```

```
printf("\n\n============================================================================================
=====\n");
            printf("Train found!\n");
            printf("Train ID: %s\n", train.trainId);
            printf("Starting Point: %s\n", train.startingPoint);
            printf("Destination: %s\n", train.destination);
            printf("Departure Time: %s\n", train.departureTime);
            printf("Cost: %d (cost for general class ticket!!)\n", train.cost);
            printf("Compartments: %d\n", train.compartments);
            printf("Seat Type: %s\n", train.seatType);
            printf("Train Name: %s\n", train.name);
            printf("Max Speed: %d km/h\n", train.maxSpeed);
            printf("Total Distance to be travelled: %d\n", train.totalDist);

printf("================================================================================================
==\n");
            sleepProgram(1);
            showCompartment(train.compartments, train.trainId);
            break;
        }
    }

    fclose(file);

    if (!trainFound) {
        printf("\nTrain with ID %s not found. Press 1 to view train list or any other key to go back to main menu.\n", trainId);
        scanf("%d", &trainNotFound);
        if (trainNotFound == 1) {
            displayAllTrains();
        }
        else{
          mainMenu();
        }
    }
}

void showCompartment( int compartment, const char* trainId){
    int choosenCompartment;
    printf("Train Compartments:\n");
    printf("    +----------------");
    for (int i = 0; i < compartment; ++i) {
        printf("----------------");
    }
    printf("+\n");
    printf("  <(     Engine      |");
    for (int i = 0; i < compartment; ++i) {
        printf(" Compartment %2d|", i + 1);
    }
    printf("\n");
    printf("    +----------------");
    for (int i = 0; i < compartment; ++i) {
        printf("----------------");
    }
    printf("+\n");
    sleepProgram(1);
    printf("\n\n\nPlease Enter Your Preferred Compartment:");
    yellowColor();
    printf("\n\t1)First compartment is First AC");
    sleepProgram(0.5);
    printf("\n\t2)Second compartment is Exectuive Class");
    sleepProgram(0.5);
    printf("\n\t3)Third compartment is Third AC");
    sleepProgram(0.5);
    printf("\n\t4)Fourth compartment is Sleeper");
    sleepProgram(0.5);
    printf("\n\t5)Rest Are  Unreserved General Class\n");
    scanf("%d", &choosenCompartment);
    resetColor();
    sleepProgram(1);
    showTickets(choosenCompartment, trainId);
}

void randomlyBookSeats(int* seats, int numSeats) {
    srand(time(NULL));
    int seatsToBook = numSeats / 2;

    for (int i = 0; i < seatsToBook; ++i) {
        int randomSeat;
        do {
            randomSeat = rand() % numSeats;
        } while (seats[randomSeat] == 1);

        seats[randomSeat] = 1;
    }
}

int isValidDate(const char *dateStr) {
    struct tm userDate = {0};

    if (sscanf(dateStr, "%d/%d/%d", &userDate.tm_mday, &userDate.tm_mon, &userDate.tm_year) != 3) {

        return 0;
    }

    userDate.tm_mon -= 1;
    userDate.tm_year -= 1900;

    time_t currentTime = time(NULL);
    struct tm *currentDate = localtime(&currentTime);

    time_t userTime = mktime(&userDate);

    if (userTime == -1 || userTime < currentTime) {

        return 0;
    }

    return 1;
}

void showTickets(int choosenCompartment, const char* trainId) {
  int numSeats = 0;

    char date[20];
    int isValid = 0;

    do {
        printf("Enter Your Departure Date in dd/mm/yyyy format: ");
        scanf("%99s", date);

        isValid = isValidDate(date);

        if (!isValid) {
            printf("Invalid date or date is in the past. Please enter a valid future date.\n");
        }

    } while (!isValid);

    if (choosenCompartment == 1 || choosenCompartment == 2 || choosenCompartment == 3)
    {
      numSeats = 54;
    } else if (choosenCompartment == 4)
    {
      numSeats = 18;
    } else
    {
      numSeats = 72;
    }
    int seats[MAX_PASSENGERS];
    int bookedSeats[MAX_PASSENGERS];

    for (int i = 0; i < numSeats; ++i) {
        seats[i] = 0;
    }

    randomlyBookSeats(seats, numSeats);

    for (int i = 0; i < numSeats; ++i) {
        bookedSeats[i] = seats[i];
    }

    printf("\nYour chosen compartment is: %d", choosenCompartment);
    sleepProgram(0.5);
    if (choosenCompartment == 1 || choosenCompartment == 2 || choosenCompartment == 3)
    {
    printf("\nThe seat type is: %s", "AC Seater");
    } else if (choosenCompartment == 4)
    {
```

```
printf("\nThe seat type is: %s", "Sleeper");
} else
{
printf("\nThe seat type is: %s", "Seater");
}
printf("\nYour travel date is: %s", date);
sleepProgram(1);
printf("\nThe Tickets are:\n");
printf("\t1) Red Are Booked Tickets\n");
printf("\t2) Green Are Available Tickets\n");
sleepProgram(1.5);

for (int i = 0; i < numSeats; ++i) {
  if (i % 6 == 0) {
    printf("\n");
  }
  if (i % 18 == 0) {
    printf("\n");
  }

  if (seats[i] == 1) {
    printf("\033[1;31m");
  } else {
    printf("\033[1;32m");
  }

  printf("\t| Seat %2d |", i + 1);
  printf("\033[0m");
}

printf("\n");
sleepProgram(1);
printf("\n\n\nEnter how many tickets you want to buy: ");
int ticketsToBuy;
scanf("%d", &ticketsToBuy);
printf("\n");

int TicketsNums[MAX_PASSENGERS];

for (size_t i = 0; i < ticketsToBuy; i++) {
  do {
    printf("Enter your %d ticket number: ", i + 1);
    scanf("%d", &TicketsNums[i]);
    printf("\n");

    if (TicketsNums[i] < 1 || TicketsNums[i] > numSeats) {
      printf("Invalid ticket number. Please enter a valid ticket number.\n");
    } else if (bookedSeats[TicketsNums[i] - 1] == 1) {
      printf("Ticket %d is already booked. Please choose another ticket.\n", TicketsNums[i]);
    }
  } while (TicketsNums[i] < 1 || TicketsNums[i] > numSeats || bookedSeats[TicketsNums[i] - 1] == 1);

  bookedSeats[TicketsNums[i] - 1] = 1;
```

```
    }

    sleepProgram(1);
    printf("Please Complete the payment to Confirm your bookings!!!\n");
    payNow(trainId, ticketsToBuy, choosenCompartment, TicketsNums, date);

}

void payNow(const char* trainId, int ticketsToBuy, int choosenCompartment, int ticketsNums[MAX_PASSENGERS], const
char* date) {
    float totalPrice = 0;
    int wantToPay;
    FILE* file = fopen("files/train.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }


    int numTrains;
    fscanf(file, "%d", &numTrains);

    Train train;

    while (fscanf(file, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint, train.destination,
train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed, &train.totalDist)
!= EOF) {
        if (strcmp(train.trainId, trainId) == 0) {
            if (choosenCompartment == 1 || choosenCompartment == 2)
            {
             totalPrice = ticketsToBuy * train.cost * 2.2;
            }
            else if(choosenCompartment == 3){
             totalPrice = ticketsToBuy * train.cost * 1.7;
            }
            else if (choosenCompartment == 4)
            {
             totalPrice = ticketsToBuy * train.cost;
            } else
            {
             totalPrice = ticketsToBuy * train.cost * 1.3;
            }
            break;
        }
    }

    fclose(file);
    sleepProgram(1);
    printf("\nYour Total Cost Is: %2f\n", totalPrice);
    sleepProgram(1);
    printf("\nPress 1 to continue to pay or press 2 to go back to ticket selection!!\n");
    scanf("%d", &wantToPay);
    if (wantToPay == 1)
```

```
   {
   sleepProgram(1);
     paymentGateway(trainId, ticketsToBuy, choosenCompartment, totalPrice, ticketsNums, date);
   }
   else if (wantToPay == 2)
   {
   sleepProgram(1);
     findTrain(trainId);
   }

}

int isAllDigits(const char *str) {
   for (int i = 0; str[i] != '\0'; i++) {
       if (!isdigit(str[i])) {
           return 0;
       }
   }
   return 1;
}

void paymentGateway(const char* trainId, int ticketsToBuy, int choosenCompartment, float totalPrice, int
ticketsNums[MAX_PASSENGERS], const char* date)
  {
   int gatewayChoice;
   int wantToPay;
   char cardNumber[100];
   char expiryDate[100];
   char cvv[10];
   clearTerminal();
   sleepProgram(1);

printf("\n\n\n\n\n=================================================================================
===========\n\n");
   printf("\t\t\t\033[1;31mBOOKING TICKETS!!\033[0m\t\t");

printf("\n\n===================================================================================
=====\n");
   sleepProgram(1);
   printf("\n====================");
   redColor();
   printf("  PAYMENT GATEWAY  ");
   resetColor();
   printf("====================\n\n");
   printf("   \033[1;31m[1]\033[0m CREDIT or DEBIT CARD\n\n");
   sleepProgram(0.5);
   printf("   \033[1;31m[2]\033[0m UPI (QR CODE)\n\n");
   sleepProgram(0.5);
   printf("   \033[1;31m[3]\033[0m CANCEL BOOKING\n\n");
   sleepProgram(0.5);
   printf("   \033[1;31m[4]\033[0m PAYMENT INFO\n\n");
   sleepProgram(0.5);
   printf("   \033[1;31m[5]\033[0m GO BACK TO TICKET BOOKING\n\n");
```

```
sleepProgram(0.5);
printf("   \033[1;31m[6]\033[0m EXIT TO MAIN MENU\n\n");
printf("Please enter an option\n");
scanf("%d", &gatewayChoice);
getchar();
switch (gatewayChoice)
{
case 1:
  clearTerminal();
  yellowColor();
  sleepProgram(1);
  printf("\nYou have chosen Card payment. Please enter card details to complete the payment:\n");
  sleepProgram(1);
  printf("\nTo cancel payment please enter '7887' in card number field\n");
  sleepProgram(1);
  resetColor();
  printf("\n\n");

  do {
     printf("Enter Card Number (16 digits): ");
     scanf("%s", cardNumber);
     if (cardNumber[0] == '7' && cardNumber[1] == '8' && cardNumber[2] == '8' && cardNumber[3] == '7')
     {
        findTrain(trainId);
        break;
     }
     if (strlen(cardNumber) != 16 || !isAllDigits(cardNumber)) {
        printf("Invalid card number. Please enter a valid 16-digit number.\n");
     }
  } while (strlen(cardNumber) != 16 || !isAllDigits(cardNumber));

  sleepProgram(1);
  printf("\nEnter Expiry Date: in format (mm/yyyy) ");
  scanf("%s", expiryDate);
  do {
     sleepProgram(1);
     printf("Enter CVV  (3 digits): ");
     scanf("%s", cvv);
     if (strlen(cvv) != 3 || !isAllDigits(cvv)) {
        printf("Invalid card CVV. Please enter a valid 3-digit number.\n");
     }
  } while (strlen(cvv) != 3 || !isAllDigits(cvv));
  printf("\n\n");PrintSleep(0.18);
  printf("Payment Successful!!");
  sleepProgram(1);
  printf("\nYou will be redirected to Tickets Section in 3 seconds\n");
  sleepProgram(3);
  confirmTickets(trainId, ticketsToBuy, choosenCompartment, ticketsNums, totalPrice, date);
  break;

case 2:

  clearTerminal();
  yellowColor();
  printf("\nYou have chosen QR payment. Please scan following QR code to make payment\n");
  printf("\nEnter the code obtained by scanning the QR code. ");
  redColor();
  printf("\nPlease Confirm Amount To Be Paid Before Scanning QR Code");
  printf("\nTo cancel payment please enter '7887'\n");
  resetColor();
  printf("\n\n");
  char* paymentInfo = calloc(1000, sizeof(char));
  sprintf(paymentInfo + strlen(paymentInfo), "\nTotal Cost: %.2f\n", totalPrice);
  int i = verifyByQrWithText("Your OTP Is:", paymentInfo);
  free(paymentInfo);
  if (i == 1) {
    PrintSleep(0.18);
    printf("Payment cancelled");
    printf("\nYou will be redirected to train section in 3 seconds\n");
    sleepProgram(3);
    trainListandBook();
  } else if(i == 0) {
    PrintSleep(0.18);
    printf("Payment successful");
    printf("\nYou will be redirected to confirm ticket details in 2 seconds\n");
    sleepProgram(1.5);
    confirmTickets(trainId, ticketsToBuy, choosenCompartment, ticketsNums, totalPrice, date);
  } else{
    PrintSleep(0.18);
    printf("Payment failed");
    sleepProgram(1);
    printf("\nYou will be redirected to tickets section of this train in 3 seconds\n");
    sleepProgram(3);
    findTrain(trainId);
  }
  break;

case 3:
  printf("\nYou have cancelled ticket booking.\n");
  printf("\nYou will be redirected to train section in 5 seconds\n");
  sleepProgram(3);
  findTrain(trainId);
  break;

case 4:
  PrintSleep(0.18);
  printf("\nYour Payment Information Is As Following:\n");
  printf("Total Cost: %f\n", totalPrice);
  printf("Total Tickets: %d\n", ticketsToBuy);
  printf("Train ID: %s\n", trainId);
  printf("Compartment: %d\n", choosenCompartment);
  if (choosenCompartment == 1 || choosenCompartment == 2 || choosenCompartment == 3)
  {
  printf("\nThe seat type is: %s", "AC Seater");
  } else if (choosenCompartment == 4)
  {
      printf("\nThe seat type is: %s", "Sleeper");
      } else
      {
      printf("\nThe seat type is: %s", "Seater");
      }
      printf("\nYour chosen date of travel is: %s\n", date);
      printf("\nPress 1 to continue to payment or press any other key to go back to main menu.\n");
      scanf("%d", &wantToPay);
      if (wantToPay == 1)
      {
        paymentGateway(trainId, ticketsToBuy, choosenCompartment, totalPrice, ticketsNums, date);
      } else
      {
        mainMenu();
      }
      break;

    case 5:
      printf("You will be redirected to Train Section in 3 seconds.\n");
      sleepProgram(3);
      findTrain(trainId);
      break;

    case 6:
      printf("You will be redirected to main menu in 3 seconds.\n");
      sleepProgram(3);
      mainMenu();
      break;

    default:
      printf("Invalid option. You will be redirected to main menu in 3 seconds.\n");
      sleepProgram(3);
      mainMenu();
      break;
    }
}

long long generate10DigitRandomNumber() {
    long long randomNumber = 0;


    srand((unsigned int)time(NULL));


    for (int i = 0; i < 10; ++i) {
        randomNumber = randomNumber * 10 + rand() % 10;
    }

    return randomNumber;
}

long long generate15DigitRandomNumber() {
    long long randomNumber = 0;
    srand((unsigned int)time(NULL));


    for (int i = 0; i < 10; ++i) {
        randomNumber = randomNumber * 15 + rand() % 10;
    }

    return randomNumber;
}

void writeReservationToFile(const Reservation* passenger) {
    FILE* file = fopen("files/reservations.txt", "a");
    if (file == NULL) {
        printf("Error opening reservations file.\n");
        exit(1);
    }
        fprintf(file, "%s %s %s %d %.2f %d %d %s %s %lld %s %s %d %lld\n", passenger->name, passenger->userId,
passenger->trainId,
            passenger->compartment, passenger->cost, passenger->noOfSeats, passenger->seats, passenger->seatType,
passenger->status, passenger->pnr, passenger->date, passenger->gender, passenger->age, passenger->phone);

    fclose(file);
}

void confirmTickets(const char* trainId, int ticketsToBuy, int choosenCompartment, int* ticketsNums, float totalPrice,
const char* date) {
    int confirmedNowGoBack;
    clearTerminal();
    sleepProgram(2);

printf("\n\n\n\n\n==========================================================================================
===========\n\n");
    printf("\t\t\t\033[1;31mTrain RESERVATION SUCCESSFUL\033[0m\t\t");

printf("\n\n============================================================================================
=====\n");
    sleepProgram(1);
    printf("\nPlease Enter following details to complete the booking:\n");
    long long pnr = generate10DigitRandomNumber();

    Reservation passengers[MAX_PASSENGERS];
    for (int i = 0; i < ticketsToBuy; ++i) {
        printf("\nEnter passenger name for Seat %2d: ", i + 1);
        scanf("%s", passengers[i].name);
        printf("Enter passenger gender for Seat %2d: ", i + 1);
        scanf("%s", passengers[i].gender);
        printf("Enter passenger age for Seat %2d: ", i + 1);
        scanf("%d", &passengers[i].age);
        printf("Enter passenger Mobile Number for Seat %2d: ", i + 1);
        scanf("%lld", &passengers[i].phone);
        strcpy(passengers[i].userId, userIdPtr);
        strcpy(passengers[i].trainId, trainId);
        passengers[i].compartment = choosenCompartment;
        passengers[i].cost = totalPrice;
        passengers[i].noOfSeats = 1;
        passengers[i].seats = ticketsNums[i];
        if (choosenCompartment == 1 || choosenCompartment == 2 || choosenCompartment == 3) {
            strcpy(passengers[i].seatType, "(AC)Seater ");
        } else if (choosenCompartment == 4) {
            strcpy(passengers[i].seatType, "Sleeper ");
        } else {
            strcpy(passengers[i].seatType, "Seater ");
        }
        strcpy(passengers[i].status, "CONFIRMED");
        passengers[i].pnr = pnr;
        strcpy(passengers[i].date, date);
        writeReservationToFile(&passengers[i]);
    }
    PrintSleep(0.18);
    printf("\nThank You For Your Details. Your PNR is: %lld\n", pnr);
    printf("\n====================");
    redColor();
    printf("  TICKET BOOKING DETAILS:  ");
    resetColor();
    printf("=====================\n\n");
    printf("\nTrain ID: %s\n", trainId);
    printf("Compartment: %d\n", choosenCompartment);
    printf("Total Tickets: %d\n", ticketsToBuy);
    printf("Total Cost: %.2f\n", totalPrice);
    printf("Your chosen date of travel is: %s\n", date);
    printf("Seat Numbers: ");
    for (int i = 0; i < ticketsToBuy; i++) {
        printf("%d, \t", ticketsNums[i]);
    }
    printf("\nYour PNR is %lld\n", pnr);
    printf("\n");
    sleepProgram(1);
    printf("\nYou will be redirected to reservation section in 5 seconds!!\n");
    sleepProgram(5);
    seeReservationInfo(pnr);
}

void PrintToBeDeletedReservation(long long pnr) {
    clearTerminal();
    FILE* reserFile = fopen("files/reservations.txt", "r");
    if (reserFile == NULL) {
        printf("Error opening reservations file for reading.\n");
        exit(1);
    }

    Reservation currentReservation;
    int reservationFound = 0;
    while (fscanf(reserFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld", currentReservation.name,
currentReservation.userId,
            currentReservation.trainId, &currentReservation.compartment, &currentReservation.cost,
            &currentReservation.noOfSeats, &currentReservation.seats , currentReservation.seatType,
currentReservation.status, &currentReservation.pnr, currentReservation.date, currentReservation.gender,
&currentReservation.age, &currentReservation.phone) != EOF) {

        if (currentReservation.pnr == pnr) {
            reservationFound = 1;

            printf("\nReservation Details:\n");
            printf("Name: %s\n", currentReservation.name);
            printf("Gender: %s\n", currentReservation.gender);
            printf("Age: %d\n", currentReservation.age);
            printf("Phone: +91 %lld\n", currentReservation.phone);
            printf("Train ID: %s\n", currentReservation.trainId);
            printf("Compartment: %d\n", currentReservation.compartment);
            printf("Cost: %.2f\n", currentReservation.cost);
            printf("Number of Seats: %d\n", currentReservation.noOfSeats);
            printf("Seat Numbers: %d \n", currentReservation.seats);
            printf("Seat Type: %s\n", currentReservation.seatType);
            printf("Status: %s\n", currentReservation.status);
            printf("Date: %s\n", currentReservation.date);
            printf("PNR: %lld\n", currentReservation.pnr);
            printf("\n");
        }
    }

    fclose(reserFile);

    if (reservationFound) {
        printf("Do you want to cancel this reservation? (1 for yes, 0 for no): ");
        int confirmation;
        scanf("%d", &confirmation);

        if (confirmation == 1) {
            deleteReservationByPnr(pnr);
            printf("Reservation cancelled successfully.\n");
            printf("Money will be refunded to original payment source within 3-4 business days.\n");
        } else {
            printf("Reservation not cancelled.\n");
        }
    } else {
        printf("Reservation not found for the given PNR.\n");
    }
    printf("You will be redirected to the main menu in 5 seconds\n");
    sleepProgram(5);
    mainMenu();
}

void deleteReservationByPnr(long long pnr) {
    FILE* reserFile = fopen("files/reservations.txt", "r");
    FILE* tempFile = fopen("files/temp.txt", "w");
    if (reserFile == NULL || tempFile == NULL) {
        printf("Error opening files for reading/writing.\n");
        exit(1);
    }

    Reservation currentReservation;


    while (fscanf(reserFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld", currentReservation.name,
currentReservation.userId,
              currentReservation.trainId, &currentReservation.compartment, &currentReservation.cost,
              &currentReservation.noOfSeats, &currentReservation.seats , currentReservation.seatType,
currentReservation.status, &currentReservation.pnr, currentReservation.date, currentReservation.gender,
&currentReservation.age, &currentReservation.phone) != EOF) {

        if (currentReservation.pnr == pnr) {
            continue;
        }


    while (fscanf(tempFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld", currentReservation.name,
currentReservation.userId,
              currentReservation.trainId, &currentReservation.compartment, &currentReservation.cost,
              &currentReservation.noOfSeats, &currentReservation.seats , currentReservation.seatType,
currentReservation.status, &currentReservation.pnr, currentReservation.date, currentReservation.gender,
&currentReservation.age, &currentReservation.phone) != EOF) {

            }
    }
    fclose(reserFile);
    fclose(tempFile);

    remove("files/reservations.txt");
    rename("files/temp.txt", "files/reservations.txt");
}

void cancelBooking(){
    long long pnrCancel;
    printf("\nEnter PNR: ");
    scanf("%lld", &pnrCancel);
    PrintToBeDeletedReservation(pnrCancel);
    clearTerminal();

printf("\n\n\n\n\n=============================================================================
===========\n\n");
    printf("\t\t\t\033[1;31mTrain RESERVATION CANCELLED\033[0m\t\t");

printf("\n\n==============================================================================
=====\n");
    printf("\nYou have cancelled ticket booking.\n");
    printf("\nYou will be redirected to main menu in 3 seconds\n");
    sleepProgram(3);
    mainMenu();
```

```
}

void seeReservationInfo(long long pnrInfo) {
  clearTerminal();
  PrintSleep(0.18);

  FILE* reserFile = fopen("files/reservations.txt", "r");
  if (reserFile == NULL) {
    printf("Error opening reservations file for reading.\n");
    exit(1);
  }
  FILE* trainDetails = fopen("files/train.txt", "r");
  if (trainDetails == NULL) {
    printf("Error opening trains file for reading.\n");
    exit(1);
  }

  int numTrains;
  fscanf(trainDetails, "%d", &numTrains);

  Reservation currentReservation;
  int reservationFound = 0;
  int l = 0;
  Train train;

  while (fscanf(reserFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld", currentReservation.name,
currentReservation.userId,
          currentReservation.trainId, &currentReservation.compartment, &currentReservation.cost,
          &currentReservation.noOfSeats, &currentReservation.seats , currentReservation.seatType,
currentReservation.status, &currentReservation.pnr, currentReservation.date, currentReservation.gender,
&currentReservation.age, &currentReservation.phone) != EOF) {


    if (currentReservation.pnr == pnrInfo) {
          while (fscanf(trainDetails, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint,
train.destination, train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed,
&train.totalDist) != EOF) {
          if (strcmp(currentReservation.trainId, train.trainId) == 0) {
        reservationFound = 1;
        l++;

printf("\n=====================================================================================
====\n");
        printf("\t\t\t\033[1;31mRESERVATION INFORMATION {TICKET %d}\033[0m\t\t", l);

printf("\n=====================================================================================
====\n");
        sleepProgram(1);

        printf("\t\t||\t Passenger Name: %-25s        ||\n", currentReservation.name);
        printf("\t\t||\t Passenger Age: %-25d        ||\n", currentReservation.age);
        printf("\t\t||\t Passenger Gender: %-25s      ||\n", currentReservation.gender);
        printf("\t\t||\t Passenger Phone: %-27lld     ||\n", currentReservation.phone);
```

```
            printf("\t\t||\t User ID: %-25s              ||\n", currentReservation.userId);
            printf("\t\t||\t Train ID: %-25s             ||\n", currentReservation.trainId);
            printf("\t\t||\t Train Name: %-25s           ||\n", train.name);
            printf("\t\t||\t Train Starting Point: %-25s  ||\n", train.startingPoint);
            printf("\t\t||\t Train Destination: %-25s     ||\n", train.destination);
            printf("\t\t||\t Compartment: %-25d           ||\n", currentReservation.compartment);
            printf("\t\t||\t Cost: %-27.2f                ||\n", currentReservation.cost);
            printf("\t\t||\t Number of Seats: %-25d       ||\n", currentReservation.noOfSeats);
            printf("\t\t||\t Seat Numbers: %-25d          ||\n", currentReservation.seats);
            printf("\t\t||\t Seat Type: %-25s             ||\n", currentReservation.seatType);
            printf("\t\t||\t Status: %-25s                ||\n", currentReservation.status);
            printf("\t\t||\t Date Of Departure: %-25s     ||\n", currentReservation.date);
            printf("\t\t||\t PNR: %-27lld                 ||\n", currentReservation.pnr);

printf("\n==========================================================================================
====\n\n");
            fseek(trainDetails, 0, SEEK_SET);
            fscanf(trainDetails, "%d", &numTrains);
            break;
              }
            }
        }
    }
    fclose(trainDetails);

    fclose(reserFile);

    printf("\nPress 1 to print tikcet or Press any key to go back to the main menu.\n");
    int reservationFoundPrintChoice;
    scanf("%d", &reservationFoundPrintChoice);
    if (reservationFoundPrintChoice == 1) {
        getTicketFormatAfterBooking(pnrInfo);
    } else {
        mainMenu();
    }
}

void reservationInfo() {
    long long pnrInfo;
    clearTerminal();
    printf("\nEnter PNR: ");
    scanf("%lld", &pnrInfo);
    PrintSleep(0.18);

    FILE* reserFile = fopen("files/reservations.txt", "r");
    if (reserFile == NULL) {
        printf("Error opening reservations file for reading.\n");
        exit(1);
    }
    FILE* trainDetails = fopen("files/train.txt", "r");
    if (trainDetails == NULL) {
        printf("Error opening trains file for reading.\n");
        exit(1);
    }

    int numTrains;
    fscanf(trainDetails, "%d", &numTrains);

    Reservation currentReservation;
    int reservationFound = 0;
    int l = 0;
    Train train;

    while (fscanf(reserFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld", currentReservation.name,
currentReservation.userId,
             currentReservation.trainId, &currentReservation.compartment, &currentReservation.cost,
             &currentReservation.noOfSeats, &currentReservation.seats , currentReservation.seatType,
currentReservation.status, &currentReservation.pnr, currentReservation.date, currentReservation.gender,
&currentReservation.age, &currentReservation.phone) != EOF) {


        if (currentReservation.pnr == pnrInfo) {
              while (fscanf(trainDetails, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint,
train.destination, train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed,
&train.totalDist) != EOF) {
              if (strcmp(currentReservation.trainId, train.trainId) == 0) {
          reservationFound = 1;
          l++;

printf("\n=========================================================================================
====\n");
          printf("\t\t\t\033[1;31mRESERVATION INFORMATION {TICKET %d}\033[0m\t\t", l);

printf("\n=========================================================================================
====\n");
          sleepProgram(1);

          printf("\t\t||\t Passenger Name: %-25s        ||\n", currentReservation.name);
          printf("\t\t||\t Passenger Age: %-25d         ||\n", currentReservation.age);
          printf("\t\t||\t Passenger Gender: %-25s      ||\n", currentReservation.gender);
          printf("\t\t||\t Passenger Phone: %-27lld     ||\n", currentReservation.phone);
          printf("\t\t||\t User ID: %-25s              ||\n", currentReservation.userId);
          printf("\t\t||\t Train ID: %-25s             ||\n", currentReservation.trainId);
          printf("\t\t||\t Train Name: %-25s           ||\n", train.name);
          printf("\t\t||\t Train Starting Point: %-25s  ||\n", train.startingPoint);
          printf("\t\t||\t Train Destination: %-25s     ||\n", train.destination);
          printf("\t\t||\t Compartment: %-25d           ||\n", currentReservation.compartment);
          printf("\t\t||\t Cost: %-27.2f                ||\n", currentReservation.cost);
          printf("\t\t||\t Number of Seats: %-25d       ||\n", currentReservation.noOfSeats);
          printf("\t\t||\t Seat Numbers: %-25d          ||\n", currentReservation.seats);
          printf("\t\t||\t Seat Type: %-25s             ||\n", currentReservation.seatType);
          printf("\t\t||\t Status: %-25s                ||\n", currentReservation.status);
          printf("\t\t||\t Date Of Departure: %-25s     ||\n", currentReservation.date);
          printf("\t\t||\t PNR: %-27lld                 ||\n", currentReservation.pnr);
printf("\n==============================================================================================
====\n\n");
            fseek(trainDetails, 0, SEEK_SET);
            fscanf(trainDetails, "%d", &numTrains);
            break;
                }
            }
        }
    }
    fclose(trainDetails);

    fclose(reserFile);

    if (!reservationFound) {
        printf("\nReservation with PNR %lld not found.\n", pnrInfo);
        printf("\nPress 1 to try again or any other key to go back to the main menu.\n");
        int reservationPrintChoice;
        scanf("%d", &reservationPrintChoice);
        if (reservationPrintChoice == 1) {
            reservationInfo();
            while (getchar() != '\n');
        } else {
            mainMenu();
        }
    }

    printf("\nPress 1 to check another reservation or Press any key to go back to the main menu.\n");
    int reservationFoundPrintChoice;
    scanf("%d", &reservationFoundPrintChoice);
    if (reservationFoundPrintChoice == 1) {
        reservationInfo();
        while (getchar() != '\n');
        printf("\nYou will be redirected to main menu in 5 seconds!\n");
        sleepProgram(5);
        mainMenu();
    } else {
        mainMenu();
    }
}

void getTicketFormat() {
    long long pnrInfo;
    clearTerminal();
    printf("\nEnter PNR: ");
    scanf("%lld", &pnrInfo);
    PrintSleep(0.18);
    printf("\nGetting Ticket Format!!!!\n\n");
    PrintSleep(0.18);

    FILE* reserFile = fopen("files/reservations.txt", "r");
    if (reserFile == NULL) {
        printf("Error opening reservations file for reading.\n");
        exit(1);
    }
    FILE* trainDetails = fopen("files/train.txt", "r");
    if (trainDetails == NULL) {
        printf("Error opening trains file for reading.\n");
        exit(1);
    }

    int numTrains;
    fscanf(trainDetails, "%d", &numTrains);

    Reservation currentReservation;
    int reservationFound = 0;
    int l = 0;
    Train train;

    Ticket ticket;


    while (fscanf(reserFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld", currentReservation.name,
currentReservation.userId,
             currentReservation.trainId, &currentReservation.compartment, &currentReservation.cost,
             &currentReservation.noOfSeats, &currentReservation.seats , currentReservation.seatType,
currentReservation.status, &currentReservation.pnr, currentReservation.date, currentReservation.gender,
&currentReservation.age, &currentReservation.phone) != EOF) {


        if (currentReservation.pnr == pnrInfo) {
            while (fscanf(trainDetails, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint,
train.destination, train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed,
&train.totalDist) != EOF) {
            if (strcmp(currentReservation.trainId, train.trainId) == 0) {
                reservationFound = 1;
                l++;
                ticket.passengerTickets[l] = malloc(sizeof(PassengerTicket));
                if (ticket.passengerTickets[l] == NULL) {
                printf("Memory allocation failed.\n");
                return;
                }
                strcpy(ticket.StartingPoint, train.startingPoint);
                strcpy(ticket.Destination, train.destination);
                snprintf(ticket.PNR, sizeof(ticket.PNR), "%lld", pnrInfo);
                strcpy(ticket.TrainName, train.name);
                strcpy(ticket.TrainId, train.trainId);

                if (currentReservation.compartment == 1 || currentReservation.compartment  == 2 ||
currentReservation.compartment  == 3)
                {
                strcpy(ticket.class, "AC SEATER");
                } else if (currentReservation.compartment  == 4)
                {
                strcpy(ticket.class, "Sleeper");
                } else
                {
                strcpy(ticket.class, "Seater");
                }

                strcpy(ticket.Date, currentReservation.date);
                snprintf(ticket.dist, sizeof(ticket.dist), "%d", train.totalDist);
                snprintf(ticket.Cost, sizeof(ticket.Cost), "%.2f", currentReservation.cost);
                snprintf(ticket.Compartment, sizeof(ticket.Compartment), "%d", currentReservation.compartment);

                strcpy(ticket.passengerTickets[l]->passName, currentReservation.name);
                snprintf(ticket.passengerTickets[l]->age, sizeof(ticket.passengerTickets[l]->age), "%d",
currentReservation.age);
                strcpy(ticket.passengerTickets[l]->gender, currentReservation.gender);
                snprintf(ticket.passengerTickets[l]->seatNum, sizeof(ticket.passengerTickets[l]->seatNum), "%d",
currentReservation.seats);

                ticket.totalSeats = l;

                fseek(trainDetails, 0, SEEK_SET);
                fscanf(trainDetails, "%d", &numTrains);
                break;
            }
        }
    }
  }

  fclose(trainDetails);

  fclose(reserFile);

  if (!reservationFound) {
    printf("\nReservation with PNR %lld not found.\n", pnrInfo);
    printf("\nPress 1 to try again or any other key to go back to the main menu.\n");
    int reservationPrintChoice;
    scanf("%d", &reservationPrintChoice);
    if (reservationPrintChoice == 1) {
      reservationInfo();
    } else {
      mainMenu();
    }
  }

  printf("\nPress 1 to print reservation or Press any key to go back to the main menu.\n");
  int reservationFoundPrintChoice;
  scanf("%d", &reservationFoundPrintChoice);
  if (reservationFoundPrintChoice == 1) {
    writeTicketPDF(&ticket);
  } else {
    mainMenu();
  }
}

void getTicketFormatAfterBooking(long long pnrInfo) {
  clearTerminal();
  PrintSleep(0.18);
  printf("\nGetting Ticket Format!!!!\n\n");
  PrintSleep(0.18);

  FILE* reserFile = fopen("files/reservations.txt", "r");
  if (reserFile == NULL) {
    printf("Error opening reservations file for reading.\n");
    exit(1);
  }
  FILE* trainDetails = fopen("files/train.txt", "r");
  if (trainDetails == NULL) {
    printf("Error opening trains file for reading.\n");
    exit(1);
  }

  int numTrains;
  fscanf(trainDetails, "%d", &numTrains);

  Reservation currentReservation;
  int reservationFound = 0;
  int l = 0;
  Train train;

  Ticket ticket;


  while (fscanf(reserFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld", currentReservation.name,
currentReservation.userId,
            currentReservation.trainId, &currentReservation.compartment, &currentReservation.cost,
            &currentReservation.noOfSeats, &currentReservation.seats , currentReservation.seatType,
currentReservation.status, &currentReservation.pnr, currentReservation.date, currentReservation.gender,
&currentReservation.age, &currentReservation.phone) != EOF) {


    if (currentReservation.pnr == pnrInfo) {
            while (fscanf(trainDetails, "%s %s %s %s %d %d %s %s %d %d", train.trainId, train.startingPoint,
train.destination, train.departureTime, &train.cost, &train.compartments, train.seatType, train.name, &train.maxSpeed,
&train.totalDist) != EOF) {
            if (strcmp(currentReservation.trainId, train.trainId) == 0) {
                reservationFound = 1;
                l++;
                ticket.passengerTickets[l] = malloc(sizeof(PassengerTicket));
                if (ticket.passengerTickets[l] == NULL) {
                printf("Memory allocation failed.\n");
                return;
                }
                strcpy(ticket.StartingPoint, train.startingPoint);
                strcpy(ticket.Destination, train.destination);
                snprintf(ticket.PNR, sizeof(ticket.PNR), "%lld", pnrInfo);
                strcpy(ticket.TrainName, train.name);
                strcpy(ticket.TrainId, train.trainId);
```

```
                if (currentReservation.compartment == 1 || currentReservation.compartment  == 2 ||
currentReservation.compartment  == 3)
                {
                strcpy(ticket.class, "AC SEATER");
                } else if (currentReservation.compartment  == 4)
                {
                strcpy(ticket.class, "Sleeper");
                } else
                {
                strcpy(ticket.class, "Seater");
                }

                strcpy(ticket.Date, currentReservation.date);
                snprintf(ticket.dist, sizeof(ticket.dist), "%d", train.totalDist);
                snprintf(ticket.Cost, sizeof(ticket.Cost), "%.2f", currentReservation.cost);
                snprintf(ticket.Compartment, sizeof(ticket.Compartment), "%d", currentReservation.compartment);

                strcpy(ticket.passengerTickets[l]->passName, currentReservation.name);
                snprintf(ticket.passengerTickets[l]->age, sizeof(ticket.passengerTickets[l]->age), "%d",
currentReservation.age);
                strcpy(ticket.passengerTickets[l]->gender, currentReservation.gender);
                snprintf(ticket.passengerTickets[l]->seatNum, sizeof(ticket.passengerTickets[l]->seatNum), "%d",
currentReservation.seats);

                ticket.totalSeats = l;

                fseek(trainDetails, 0, SEEK_SET);
                fscanf(trainDetails, "%d", &numTrains);
                break;
              }
            }
        }
    }

    fclose(trainDetails);

    fclose(reserFile);

    if (!reservationFound) {
        printf("\nReservation with PNR %lld not found.\n", pnrInfo);
        printf("\nPress 1 to try again or any other key to go back to the main menu.\n");
        int reservationPrintChoice;
        scanf("%d", &reservationPrintChoice);
        if (reservationPrintChoice == 1) {
            reservationInfo();
        } else {
            mainMenu();
        }
    }

    printf("\nPress 1 to print reservation or Press any key to go back to the main menu.\n");
    int reservationFoundPrintChoice;
    scanf("%d", &reservationFoundPrintChoice);
    if (reservationFoundPrintChoice == 1) {
        writeTicketPDF(&ticket);
    } else {
        mainMenu();
    }
}

int writeTicketPDF(const Ticket* ticket) {

    struct pdf_info info = {.creator = "Group 7",
                            .producer = "Group 7",
                            .title = "Rail Tickets",
                            .author = "Group 7",
                            .subject = "Reservation Tickets!"};
    struct pdf_doc *pdf = pdf_create(PDF_A4_WIDTH, PDF_A4_HEIGHT, &info);

    if (!pdf) {
        fprintf(stderr, "Unable to create PDF\n");
        return -1;
    }
    pdf_append_page(pdf);
    pdf_add_image_file(pdf, NULL, 10, 730, 569, 100, "asset/header.jpg");

    pdf_set_font(pdf, "Helvetica-Bold");
    pdf_add_text(pdf, NULL, ticket->StartingPoint, 9, 65, 712, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, ticket->StartingPoint, 9, 270, 712, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, ticket->Destination, 9, 480, 712, PDF_RGB(30, 40, 50));

    char date[100];
    strcpy(date, "Start Date*:");
    strcat(date, ticket->Date);
    pdf_set_font(pdf, "Helvetica-Bold");
    pdf_add_text(pdf, NULL, ticket->Date, 12, 50, 692, PDF_RGB(0, 128, 255));
    pdf_add_text(pdf, NULL, "Departure* N.A.", 12, 240, 692, PDF_RGB(0, 128, 255));
    pdf_add_text(pdf, NULL, "Arrival* N.A.", 12, 470, 692, PDF_RGB(0, 128, 255));

    pdf_set_font(pdf, "Helvetica-Bold");
    pdf_add_text(pdf, NULL, "PLEASE CHECK TIMINGS BEFORE BOARDING", 9, 20, 669, PDF_RGB(30, 40, 50));

    pdf_add_line(pdf, NULL, 15, 662, 575, 662, 2 ,PDF_RGB(30, 40, 50));

    char TrainName[100];
    strcpy(TrainName, ticket->TrainId);
    strcat(TrainName, "/");
    strcat(TrainName, ticket->TrainName);
    pdf_add_text(pdf, NULL, "PNR", 9, 65, 647, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Train No./Name", 9, 265, 647, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Class", 9, 485, 647, PDF_RGB(30, 40, 50));
    pdf_set_font(pdf, "Helvetica");
    pdf_add_text(pdf, NULL, ticket->PNR, 12, 40, 632, PDF_RGB(0, 128, 255));
    pdf_add_text(pdf, NULL, TrainName, 12, 230, 632, PDF_RGB(0, 128, 255));

if (strcmp(ticket->Compartment, "1") == 0) {
    pdf_add_text(pdf, NULL, "FIRST AC(1A)", 12, 470, 632, PDF_RGB(0, 128, 255));
} else if (strcmp(ticket->Compartment, "2") == 0) {
    pdf_add_text(pdf, NULL, "SECOND AC(2A)", 12, 470, 632, PDF_RGB(0, 128, 255));
} else if (strcmp(ticket->Compartment, "3") == 0) {
    pdf_add_text(pdf, NULL, "THIRD AC(3A)", 12, 470, 632, PDF_RGB(0, 128, 255));
} else if (strcmp(ticket->Compartment, "4") == 0) {
    pdf_add_text(pdf, NULL, "Sleeper", 12, 480, 632, PDF_RGB(0, 128, 255));
} else {
    pdf_add_text(pdf, NULL, "General", 12, 480, 632, PDF_RGB(0, 128, 255));
}

    pdf_set_font(pdf, "Helvetica-Bold");
    pdf_add_text(pdf, NULL, "Quota", 9, 60, 612, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Distance", 9, 275, 612, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Departure Date", 9, 475, 612, PDF_RGB(30, 40, 50));
    pdf_set_font(pdf, "Helvetica");
    char dist[100];
    strcpy(dist, ticket->dist);
    strcat(dist, " kms");
    pdf_add_text(pdf, NULL, "GENERAL", 12, 44, 597, PDF_RGB(0, 128, 255));
    pdf_add_text(pdf, NULL, dist, 12, 270, 597, PDF_RGB(0, 128, 255));
    pdf_add_text(pdf, NULL, ticket->Date, 12, 473, 597, PDF_RGB(0, 128, 255));

    pdf_add_line(pdf, NULL, 15, 587, 575, 587, 2 ,PDF_RGB(30, 40, 50));

    pdf_set_font(pdf, "Helvetica-Bold");
    pdf_add_text(pdf, NULL, "Passenger Details:", 11, 20, 572, PDF_RGB(30, 40, 50));
    pdf_add_line(pdf, NULL, 20, 568, 125, 568, 2 ,PDF_RGB(30, 40, 50));

    pdf_add_text(pdf, NULL, "#", 10, 20, 555, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Name", 10, 35, 555, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Age", 10, 175, 555, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Gender", 10, 225, 555, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Ticket Status", 10, 305, 555, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Compartment", 10, 395, 555, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Seat Num", 10, 485, 555, PDF_RGB(30, 40, 50));
    char currentPassenger[10];
    int p1nextHeight, p1oldHeight;
    p1oldHeight = 555;
    p1nextHeight = p1oldHeight - 15;
    pdf_set_font(pdf, "Helvetica");
    //passenger details below::
    for (size_t i = 1; i <= ticket->totalSeats; i++)
    {
        pdf_set_font(pdf, "Helvetica");
        snprintf(currentPassenger, sizeof(currentPassenger), "%d", i);
        pdf_add_text(pdf, NULL, currentPassenger, 10, 20, p1nextHeight, PDF_RGB(30, 40, 50));
        pdf_add_text(pdf, NULL, ticket->passengerTickets[i]->passName, 10, 35, p1nextHeight, PDF_RGB(30, 40, 50));
        pdf_add_text(pdf, NULL, ticket->passengerTickets[i]->age, 10, 180, p1nextHeight, PDF_RGB(30, 40, 50));
        pdf_add_text(pdf, NULL, ticket->passengerTickets[i]->gender, 10, 230, p1nextHeight, PDF_RGB(30, 40, 50));
        pdf_add_text(pdf, NULL, "RAC/7", 10, 315, p1nextHeight, PDF_RGB(30, 40, 50));
        pdf_add_text(pdf, NULL, ticket->Compartment, 10, 425, p1nextHeight, PDF_RGB(30, 40, 50));
        pdf_add_text(pdf, NULL, ticket->passengerTickets[i]->seatNum, 10, 505, p1nextHeight, PDF_RGB(30, 40, 50));
        p1oldHeight = p1nextHeight;
        p1nextHeight = p1oldHeight - 10;
    }

    p1nextHeight = p1oldHeight - 10;

    pdf_add_line(pdf, NULL, 15, p1nextHeight, 575, p1nextHeight, 2 ,PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 20;

    char transactionIDtxt[50] = "Transaction ID: ";
    char transactionIDnum[20];
    long long transactionID = generate15DigitRandomNumber();
    sprintf(transactionIDnum, "%lld", transactionID);
    strcat(transactionIDtxt, transactionIDnum);

    pdf_add_text(pdf, NULL, transactionIDtxt, 12, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 18;
    pdf_add_text(pdf, NULL, "IR recovers only 57% of cost of travel on an average.", 10, 20, p1nextHeight, PDF_RGB(30,
40, 50));
    pdf_set_font(pdf, "Helvetica-Bold");
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 19;
    pdf_add_text(pdf, NULL, "Payment Details:", 12, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 5;
    pdf_add_line(pdf, NULL, 20, p1nextHeight, 115, p1nextHeight, 2 ,PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 23;

    pdf_set_font(pdf, "Helvetica");
    pdf_add_text(pdf, NULL, "Ticket Fare", 10, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, ticket->Cost, 10, 300, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 15;
    pdf_add_text(pdf, NULL, "IRCTC Convenience Fee (Incl. of GST)", 10, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Rs. 0", 10, 300, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 15;
    pdf_add_text(pdf, NULL, "Group 7 Convenience Fee (Incl. of GST)", 10, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "Rs. 0", 10, 300, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 15;
    pdf_add_text(pdf, NULL, "Total Fare (all inclusive)", 10, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, ticket->Cost, 10, 300, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 5;
    pdf_add_barcode(pdf, NULL, 0 , 380, p1nextHeight, 170, 80, "Ticket Reservation Successful" ,PDF_RGB(30, 40, 50));

    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 25;
    pdf_set_font(pdf, "Helvetica-Bold");
    pdf_add_text(pdf, NULL, "IRCTC Convenience Fee is charged per e-ticket irrespective of number of passengers on the
ticket.", 9, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 20;
    pdf_add_text(pdf, NULL, "*The printed Departure and Arrival Times are liable to change. Please Check correct
departure, arrival from Railway Station Enquiry", 9, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 10;
    pdf_add_text(pdf, NULL, "or Dial 139 or SMS RAIL to 139.", 9, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 10;

    pdf_add_line(pdf, NULL, 15, p1nextHeight, 575, p1nextHeight, 1 ,PDF_RGB(30, 40, 50));

    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 20;
    pdf_set_font(pdf, "Helvetica");
    pdf_add_text(pdf, NULL, "1)This ticket is booked on a personal User ID, its sale/purchase is an offence u/s 143 of the
Railways Act,1989.", 9, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 15;
    pdf_add_text(pdf, NULL, "2)Prescribed original ID proof is required while travelling along with SMS/ VRM/ ERS
otherwise will be treated as without ticket and", 9, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 10;
    pdf_add_text(pdf, NULL, "  penalized as per Railway Rules.", 9, 20, p1nextHeight, PDF_RGB(30, 40, 50));

    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 20;
    pdf_set_font(pdf, "Helvetica-Bold");
    pdf_add_text(pdf, NULL, "Indian Railways GST Details:", 9, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 14;
    pdf_set_font(pdf, "Helvetica");
    pdf_add_text(pdf, NULL, "Invoice Number: PS23870962017511                    Address: Indian Railways New Delhi", 9,
20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 18;
    pdf_set_font(pdf, "Helvetica-Bold");
    pdf_add_text(pdf, NULL, "Supplier Information:", 9, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 14;
    pdf_set_font(pdf, "Helvetica");
    pdf_add_text(pdf, NULL, "SAC Code: 996421                                   GSTIN: 07AAAGM0289C1ZL", 9, 20,
p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 18;
    pdf_set_font(pdf, "Helvetica-Bold");
    pdf_add_text(pdf, NULL, "Recipient Information:", 9, 20, p1nextHeight, PDF_RGB(30, 40, 50));
    p1oldHeight = p1nextHeight;
    p1nextHeight = p1oldHeight - 14;
    pdf_set_font(pdf, "Helvetica");
    pdf_add_text(pdf, NULL, "GSTIN: NA        Name: NA          Address: NA", 9, 20, p1nextHeight, PDF_RGB(30, 40, 50));

    //bottom mark
    pdf_set_font(pdf, "Helvetica-Bold");
    pdf_add_line(pdf, NULL, 10, 22, 585, 22, 1 ,PDF_RGB(30, 40, 50));
    pdf_add_text(pdf, NULL, "GROUP 7 E-RAIL SERVICES", 11, 220, 12, PDF_RGB(255, 0 , 0));

    //border

    pdf_add_line(pdf, NULL, 10, 830, 585, 830, 1 ,PDF_RGB(30, 40, 50)); //horizontal
    pdf_add_line(pdf, NULL, 10, 10, 585, 10, 1 ,PDF_RGB(30, 40, 50));
    pdf_add_line(pdf, NULL, 585, 10, 585, 830, 1 ,PDF_RGB(30, 40, 50));  //vertical
    pdf_add_line(pdf, NULL, 10, 10, 10, 830, 1 ,PDF_RGB(30, 40, 50));
    float height, width;
    int nextHeight, oldHeight ;
    pdf_append_page(pdf);

    pdf_add_text(pdf, NULL, "INSTRUCTIONS:", 11, 20, 810, PDF_RGB(30, 40, 50));
    pdf_add_line(pdf, NULL, 20, 808, 105, 808, 1 ,PDF_RGB(30, 40, 50));
    pdf_set_font(pdf, "Helvetica");
    oldHeight = 790;
    pdf_add_text_wrap(
       pdf, NULL,
       "1. Prescribed Original ID proofs are:- Voter Identity Card / Passport / PAN Card / Driving License / Photo ID card
issued by Central / State Govt. / Public Sector Undertakings of State / Central Government ,District Administrations ,
Municipal bodies and Panchayat Administrations which are having serial number / Student Identity Card with
photograph issued by recognized School or College for their students / Nationalized Bank Passbook with photograph
/Credit Cards issued by Banks with laminated photograph/Unique Identification Card \"Aadhaar\", m-Aadhaar, e-
Aadhaar. /Passenger showing the Aadhaar/Driving Licence from the \"Issued Document\" section by logging into his/her
DigiLocker account considered as valid proof of identity. (Documents uploaded by the user i.e. the document in
\"Uploaded Document\" section will not be considered as a valid proof of identity).",
       8, 20, oldHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       nextHeight = oldHeight - height - 5;
    pdf_add_text_wrap(
       pdf, NULL,
       "2. PNRs having fully waitlisted status will be dropped and automatic refund of the ticket amount after deducting the
applicable CLERKAGE by Railway shall be credited to the account used for payment for booking of the ticket. Passengers
having fully waitlisted e-ticket are not allowed to board the train. However, the names of PARTIALLY waitlisted/confirmed
and RAC ticket passenger will appear in the chart and will be allowed to board the train.",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "3. Passengers travelling on a fully waitlisted e-ticket will be treated as Ticketless",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "4. Obtain certificate from the TTE /Conductor in case of (a) PARTIALLY waitlisted e-ticket when LESS NO. OF
PASSENGERS travel, (b)A.C FAILURE, (c)TRAVEL IN LOWER CLASS. This original certificate must be sent to GGM (IT), IRCTC,
Internet Ticketing Centre, IRCA Building, State Entry Road, New Delhi-110055 after filing TDR online within prescribed
time for claiming refund.",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "5. In case, on a party e-ticket or a family e-ticket issued for travel of more than one passenger, some passengers
have confirmed reservation and others are on RAC or waiting list, full refund of fare, less clerkage, shall be admissible for
confirmed passengers also subject to the condition that the ticket shall be cancelled online or online TDR shall be filed
for all the passengers upto thirty minutes before the scheduled departure of the train.",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "6. In case train is late more than 3 hours, refund is admissible as per railway refund rules only when TDR is filed by
the user before the actual departure of the train at boarding station and passenger has not travelled.",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "7. In case of train cancellation on its entire run, full refund is granted automatically by the system. However, if the
train is cancelled partially on its run or diverted and not touching boarding/destination station, passengers are required
to file online TDR within 72 hours of scheduled departure of the train from passengers boarding station.",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "8. Never purchase e-ticket from unauthorized agents or persons using their personal IDs for commercial purposes.
Such tickets are liable to be cancelled and forfeited without any refund of money, under section (143) of the Indian
Railway Act 1989. List of authorized agents are available on www.irctc.co.in under 'Find NGet Agents' option.",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "9. For detail, Rules, Refund rules, Terms & Conditions of E-Ticketing services, Travel Insurance facility etc. Please
visit www.irctc.co.in",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "10. While booking this ticket, you have agreed of having read the Health Protocol of Destination State of your
travel. You are again advised to clearly read the Health Protocol advisory of destination state before start of your travel
and follow them properly.",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "11. The FIR forms are available with on board ticket checking staff, train guard and train escorting RPF/GRP staff.",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "12. Variety of meals available in more than 1500 trains. For delivery of meal of your choice on your seat log on to
www.ecatering.irctc.co.in or call 1323 Toll Free. For any suggestions/complaints related to Catering services, contact Toll
Free No. 1800-111-321 (07.00 hrs to 22.00 hrs)",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "13. National Consumer Helpline (NCH) Toll Free Number: 1800-11-400 or 14404",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;
    pdf_add_text_wrap(
       pdf, NULL,
       "14. You can book unreserved ticket from UTS APP or ATVMs (Automatic Ticket Vending Machines) located in
Railway Stations.",
       8, 20, nextHeight, 0, PDF_RGB(0, 0, 0), 560, PDF_ALIGN_JUSTIFY, &height);
       oldHeight = nextHeight - 5;
       nextHeight = oldHeight - height;

    pdf_add_text(pdf, NULL, "Contact us on: - care@irctc.co.in OR 24*7 Hrs Customer Support at 14646 OR 0755-6610661,
0755-4090600", 8, 100, nextHeight-10, PDF_RGB(0, 0, 0));

    pdf_add_line(pdf, NULL, 12, nextHeight - 20, 580, nextHeight - 20, 2 ,PDF_RGB(30, 40, 50));

    pdf_add_image_file(pdf, NULL, 160, 240 , 256, 190, "asset/logo.jpeg");
    pdf_add_text(pdf, NULL, "G7 E-Rail Services PVT LIMITED.", 9, 220, nextHeight - 30, PDF_RGB(255, 0, 0));


    pdf_add_text(pdf, NULL, "G7 is registered trademark of G7 E-Rail Services PVT LIMITED.", 8, 180, 226, PDF_RGB(255, 0,
0));
    pdf_add_text(pdf, NULL, "Any unauthorized use of this trademark and/or misuse of this document is strictly prohibited
and may result into strict action inlicating civil and criminal penalties", 8, 14, 216, PDF_RGB(255, 0, 0));

    pdf_add_line(pdf, NULL, 12, 212, 580, 212, 2 ,PDF_RGB(30, 40, 50));
    pdf_add_image_file(pdf, NULL, 12, 10, 569, 200, "asset/help.jpg");


    //border

    pdf_add_line(pdf, NULL, 10, 830, 585, 830, 1 ,PDF_RGB(30, 40, 50)); //horizontal
    pdf_add_line(pdf, NULL, 10, 10, 585, 10, 1 ,PDF_RGB(30, 40, 50));
    pdf_add_line(pdf, NULL, 585, 10, 585, 830, 1 ,PDF_RGB(30, 40, 50));  //vertical
    pdf_add_line(pdf, NULL, 10, 10, 10, 830, 1 ,PDF_RGB(30, 40, 50));
    // Save the PDF to a file
    char extension[] = ".pdf";
    char ticketName[100];
    sprintf(ticketName, "tickets/%s%s%s%s", ticket->PNR, "_", ticket->Destination, extension);
    pdf_save(pdf, ticketName);
    sleepProgram(1);
    printf("\n\nTicket Generated \n");
    PrintSleep(0.18);
    greenColor();
    printf("\nTicket saved to folder \"Tickets\" with name: %s\n", ticketName);
    resetColor();
    sleepProgram(1);
    pdf_destroy(pdf);
    printf("\nYou will be redirected to main menu in 5 seconds!\n");
    sleepProgram(5);
    mainMenu();
}

void checkAllReservations() {
    clearTerminal();
    PrintSleep(0.18);
    int currentReservationFound = 0;

printf("\n=======================================================================================
====\n");
    printf("\t\t\t\033[1;31mALL YOUR RESERVATIONS \033[0m\t\t");

printf("\n=======================================================================================
====\n");

    FILE* reserFile = fopen("files/reservations.txt", "r");
    if (reserFile == NULL) {
        printf("Error opening reservations file for reading.\n");
        exit(1);
    }
    sleepProgram(1);
    Reservation currentReservation;

    printf("\nYour Reservations:\n");

    while (fscanf(reserFile, "%s %s %s %d %f %d %d %s %s %lld %s %s %d %lld", currentReservation.name,
currentReservation.userId,
            currentReservation.trainId, &currentReservation.compartment, &currentReservation.cost,
            &currentReservation.noOfSeats, &currentReservation.seats , currentReservation.seatType,
currentReservation.status, &currentReservation.pnr, currentReservation.date, currentReservation.gender,
&currentReservation.age, &currentReservation.phone) != EOF) {

        if (strcmp(currentReservation.userId, userIdPtr) == 0) {
            currentReservationFound = 1;
            printf("\nReservation Details:\n");
            printf("Name: %s\n", currentReservation.name);
            printf("Age: %d\n", currentReservation.age);
            printf("Gender: %s\n", currentReservation.gender);
            printf("Phone: %lld\n", currentReservation.phone);
            printf("Train ID: %s\n", currentReservation.trainId);
            printf("Compartment: %d\n", currentReservation.compartment);
            printf("Cost: %.2f\n", currentReservation.cost);
            printf("Number of Seats: %d\n", currentReservation.noOfSeats);
            printf("Seat Numbers: %d ", currentReservation.seats);
            printf("\n");
            printf("Seat Type: %s\n", currentReservation.seatType);
            printf("Status: %s\n", currentReservation.status);
            printf("Date: %s\n", currentReservation.date);
            printf("PNR: %lld\n", currentReservation.pnr);
            printf("\n");
        }
    }

    fclose(reserFile);
    if (currentReservationFound == 0) {
        printf("\nYou Don't Have Any Reservations!!\n");
    }

    printf("\nPress any key to go back to Main Menu\n");
    int allResChoice;
    scanf("%d", &allResChoice);
    if (allResChoice == 1) {
        mainMenu();
    }
    else{
        mainMenu();
    }
}

void showUserInfo(){
    int userInfoChoice;
    FILE *file = fopen("files/users.txt", "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }
    int foundUser=0;

    char line[500];
    while (fgets(line, sizeof(line), file) != NULL) {
        User user;
        sscanf(line, "%s %s %s %s %d %s %lld", user.userId, user.email, user.password, user.name, &user.age, user.gender,
&user.phone);

        if (strcmp(user.userId, userId) == 0) {
            foundUser = 1;
            PrintSleep(0.18);
            clearTerminal();

printf("\n\n\n\n\n==========================================================================================
===========\n\n");
            printf("\t\t\t\t\t\033[1;31mYour Profile\033[0m\t\t");
printf("\n\n============================================================================================
=====\n");
            printf("\tUser ID: %s\n", user.userId);
            printf("\tName: %s\n", user.name);
            printf("\tAge: %d\n", user.age);
            printf("\tGender: %s\n", user.gender);
            printf("\tPhone: +91 %lld\n", user.phone);
            printf("\tEmail: %s\n", user.email);
            printf("\tPassword: %s\n", user.password);

printf("\n\n============================================================================================
=====\n");
            break;
            }
        }

    fclose(file);
    if(foundUser == 1){
    printf("\nPress 1 to change password or Press 2 to update profile details or Press any key to go back to Main Menu\n");
    scanf("%d", &userInfoChoice);
    if (userInfoChoice == 1) {
        sleepProgram(2);
        resetPass();
    }
    else if(userInfoChoice == 2){
        sleepProgram(2);
        changeUserInfo();
    }
    else{
        mainMenu();
    }
    }
}

void changeUserInfo(){
    int WantToChange, resetDoneUser;
    char email[100];
    char newPassword[100];
    char line[500];
    getchar();
    FILE *file = fopen("files/users.txt", "r");

    if (file == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    FILE *tempFile = fopen("files/temp.txt", "w");
    if (tempFile == NULL) {
        printf("Error creating temporary file.\n");
        fclose(file);
        exit(1);
```

```
   }

   int userFound = 0;
   while (fgets(line, sizeof(line), file) != NULL) {
      User user;
      sscanf(line, "%s %s %s %s %d %s %lld", user.userId, user.email, user.password, user.name, &user.age, user.gender,
&user.phone);
         if (strcmp(user.userId, userId) == 0) {
         userFound = 1;

printf("\n\n\n\n\n============================================================================================================
===========\n\n");
         printf("\t\t\t\t\t\033[1;31mYour Current Details:\033[0m\t\t");

printf("\n\n====================================================================================================================
=====\n");
         printf("\tUser ID: %s\n", user.userId);
         printf("\tName: %s\n", user.name);
         printf("\tAge: %d\n", user.age);
         printf("\tGender: %s\n", user.gender);
         printf("\tPhone: +91 %lld\n", user.phone);
         printf("\tEmail: %s\n", user.email);
         printf("\tPassword: %s\n", user.password);

printf("\n\n====================================================================================================================
=====\n");
         printf("\n\nTo update ur details press 1 or press any other key to exit to main menu\n");
         scanf("%d", &WantToChange);
         if (WantToChange == 1) {
            printf("\nEnter new details:\n");
            printf("Please enter your new name: ");
            scanf("%s", user.name);
            getchar();

            printf("Please enter your Age: ");
            scanf("%d", &user.age);
            getchar();

            printf("Please enter your Gender: ");
            scanf("%s", user.gender);
            getchar();

            printf("Please enter your Phone: ");
            scanf("%lld", &user.phone);
            getchar();

            printf("Please enter your new password: ");
            scanf("%s", user.password);
            getchar();
            sleepProgram(2);
            printf("\n\nProfile  Update Successful!!");
            printf("\nYou will be redirected to Main Menu in 3 seconds\n");
            sleepProgram(3);
            mainMenu();
          }
          else{
            mainMenu();
          }
          }

          fprintf(tempFile, "%s %s %s %s %d %s %lld\n", user.userId, user.email, user.password, user.name, user.age,
user.gender, user.phone);
        }
    fclose(file);
    fclose(tempFile);
    remove("files/users.txt");
    fclose(fopen("files/users.txt", "w"));
    copyFile("files/temp.txt", "files/users.txt");
    remove("files/temp.txt");
    fclose(file);
}

void mainMenu(){
    resetColor();
    clearTerminal();
    int choice;

printf("\n\n\n\n\n============================================================================================
===========\n\n");
    printf("\t\t\t\033[1;31mTrain RESERVATION\033[0m\t\t");


printf("\n\n====================================================================================================
=====\n");
    printf("\n====================");
    redColor();
    printf("  MAIN MENU  ");
    resetColor();
    printf("=====================\n\n");
    printf("   \033[1;31m[1]\033[0m VIEW TRAIN LIST \n\n");
    printf("   \033[1;31m[2]\033[0m BOOK TICKETS\n\n");
    printf("   \033[1;31m[3]\033[0m CANCEL BOOKING\n\n");
    printf("   \033[1;31m[4]\033[0m RESERVATION INFO\n\n");
    printf("   \033[1;31m[5]\033[0m SEE ALL YOUR RESERVATIONS\n\n");
    printf("   \033[1;31m[6]\033[0m PRINT TICKET\n\n");
    printf("   \033[1;31m[7]\033[0m LOGOUT\n\n");
    printf("   \033[1;31m[8]\033[0m DELAYED TRAIN STATUS\n\n");
    printf("   \033[1;31m[9]\033[0m Profile\n\n\n");
    printf("   \033[1;31m[11]\033[0m EXIT\n\n");
    printf("\n=================================================================");

printf("\n=====================================================================================================
====\n");
    scanf("%d", &choice);
    switch (choice)
    {
```

```
    case 1:
        trainListandBook();
        break;
    case 2:
        trainListandBook();
        break;
    case 3:
        cancelBooking();
        break;
    case 4:
        reservationInfo();
        break;
    case 5:
        checkAllReservations();
        break;
    case 6:
        getTicketFormat();
        break;
    case 7:
        *loggedPtr = 0;
        memset(userIdPtr, '0', strlen(userId));
        LogOrSign();
        break;
    case 8:
        displayDelayedTrains(1);
        break;
    case 9:
        showUserInfo();
        break;
    case 11:
        exit(0);
        break;
    default:
        printf("\nInvalid Choice!!\n");
        mainMenu();
        break;
    }
}

int main(){
    clearTerminal();
    LogOrSign();
    clearTerminal();
    mainMenu();
}
```

//CODE-END//
Github link: https://github.com/MrEthical07/TrainLCA