

|  |
| --- |
| Name: SHAIK SADIK |
| Reg. No: CL202409050182653 |
| Batche NO: EMBVLRSR 01 |
| Project Title: Flight Ticket Booking (S D A Aviation) |
| S/w Used: Online GDB Compiler |
| Data Structures USED: C Programming, Double Linked List |
|  |
|  |
| Trainer Name: S. Basavaraj |

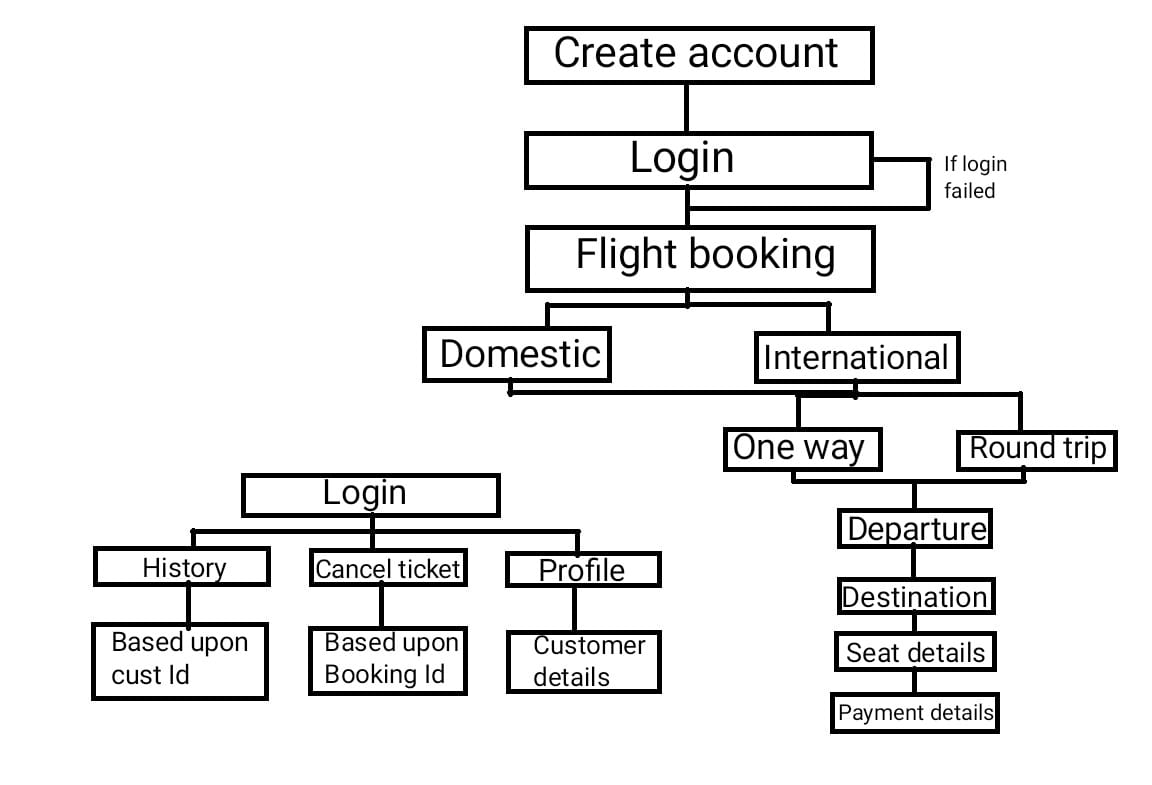
**Abstract:**

S D A Aviation helps the passenger’s to look up flights between two points which can be domestic or international. The passengers can find and book tickets for flights through this software. Developed in C, it is fairly easy to use software having a user-friendly interface. The users can access flight information and book tickets as per their convenience. The main objective is to reduce the mistakes which creep up in manual systems.it provides good level security so it takes care of the user’s safety concerns as well. passengers can access the whole list of all flights available on different routes with their timings and fare both for economy and business classes. One can compare the best deals for them and book a flight ticket accordingly. When the passengers enter all the details the software helps them to find all available flights and also information if there are seats available on that particular flight. The manual work is thus reduced and the chances of errors are reduced to minimum.

**Introduction to Project:**

SDA Aviation is a reputed airline offering a seamless and customer-friendly experience when it comes to booking flight tickets. Whether you are planning a domestic getaway or an international adventure, SDA Aviation provides a range of options designed to suit different budgets and preferences. With a user-centric approach, the airline’s flight booking system is built to make the process as simple, efficient, and transparent as possible.

Customers can book tickets through multiple channels, including their official website, mobile apps, travel agencies, or directly via customer service. The online booking portal offers an intuitive interface where travelers can quickly search for available flights, compare prices, choose seats, and complete the payment process.



/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*FLIGHT TICKET BOOKING APPLICATION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Creating Flight ticket booking application using c with functions to perform operations like register ,login,booking view ticket,cancel ticket ,view history ,profile .

using double liked lit ,switch statements,if else conditions,loops etc..\*/

//header files

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#include <time.h>

// structure for Customer details

typedef struct

{

char cust\_name[100];

long int customer\_id;

char cust\_mailid[200];

char password[200];

char address[100];

char city[100];

long int pin\_code;

char country[100];

long int mno;

char saved\_card[20];

} Customer;

// structure for Booking details details

typedef struct

{

int bookingID;

char flight\_name[100];

char flight\_code[10];

char date[11];

char dep\_country[100];

char dest\_country[100];

float cost;

int adult;

int child;

int ages[20];

char adultname[10][100];

char childname[10][100];

char classes\_of\_service[30];

int seat\_numbers[20];

int seat\_count;

int is\_round\_trip;

int num\_adultss;

int num\_childrens;

float ccost;

float acost;

int cc;

int ac;

} Booking;

//Customer Node defining

typedef struct CustomerNode

{

Customer customer;

struct CustomerNode \*next;

} CustomerNode;

//Booking Node defining

typedef struct BookingNode

{

Booking booking;

struct BookingNode \*next;

} BookingNode;

CustomerNode \*customer\_head = NULL;

BookingNode \*booking\_head = NULL;

CustomerNode \*current\_customer = NULL;

int is\_seat\_available[141] = {0};

// Functions Declaration

void display\_seat\_selection(int seat\_class, int \*seats, int \*seat\_count);

void collect\_payment\_info(char \*saved\_card);

void book\_flight\_ticket();

void display\_ticket();

void create\_account();

void login();

void cancel\_ticket();

void travel\_history();

void display\_customer\_details();

int validate\_pin(char \*pin);

int validate\_password(char \*password);

// Array for destinations, airlines, codes, and costs

const char \*domestic\_locations[] = {"Delhi", "Mumbai", "Bengaluru", "Kolkata", "Chennai", "Hyderabad", "Jaipur", "Goa", "Pune", "Lucknow"};

const char \*international\_locations[] = {"India","New York", "London", "Dubai", "Singapore", "Paris", "Tokyo", "Sydney", "Toronto", "Bangkok", "Hong Kong"};

const char \*airlines[] = {"Air India", "Indigo", "SpiceJet", "Emirates", "Lufthansa", "Qatar Airways", "British Airways", "Air France", "Singapore Airlines", "Turkish Airlines"};

const char \*airline\_codes[] = {"AI", "IG", "SJ", "EK", "LH", "QR", "BA", "AF", "SQ", "TK"};

// Separate costs for domestic and international flights

float domestic\_adult\_costs[] = {5000, 4500, 4000, 6000, 5500, 5000, 4500, 4800, 4200, 4600};

float domestic\_child\_costs[] = {2500, 2250, 2000, 3000, 2750, 2500, 2250, 2400, 2100, 2300};

float international\_adult\_costs[] = {10000, 9000, 8000, 15000, 14000, 13000, 12000, 12500, 11000, 11500};

float international\_child\_costs[] = {5000, 4500, 4000, 7500, 7000, 6500, 6000, 6250, 5500, 5750};

int fn=-1;

//main function that contain main interface of the application

int main()

{

int mainchoice;

srand(time(0)); // Seed random number generation only once

//printing the logo and title

printf("----------------------------------------------------------------------------------------------------\n\n\n");

printf("\t \* \t\t");

printf(" \*\*\*\*\*\* \*\*\*\*\*\* \*\n");

printf("\t \* \t\t");

printf(" \*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\* \*\*\*\n");

printf("\t\* \* \t\t");

printf(" \*\* \*\* \*\* \*\* \*\* \*\*\n");

printf("\t \* \* \t\t");

printf(" \*\* \*\* \*\* \*\* \*\* \n");

printf("\t \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \t\t");

printf(" \*\*\*\*\*\* \*\* \*\* \*\*\*\*\*\*\*\*\*\n");

printf("\t \* \* \t\t");

printf(" \*\* \*\* \*\* \*\*\*\*\*\*\*\*\*\*\*\n");

printf("\t\* \* \t\t");

printf(" \*\* \*\* \*\* \*\* \*\* \*\*\n");

printf("\t \* \t\t");

printf(" \*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\* \*\* \*\*\n");

printf("\t \* \t\t");

printf(" \*\*\*\*\*\* \*\*\*\*\*\* \*\* \*\*\n\n\n");

printf("\tM A K E Y O U R T R I P E A S Y\t\t\t\n");

printf("---------------------------------------------------------------------------------------------------\n\n");

//switch cases for Different operations

while (1)

{

printf("\n1. Create Account\t 2. Login\t \t \t 3. Book Flight Ticket\t 4. Display Ticket\n \n\

5. Cancel Ticket\t 6. View Booking History\t 7. Profile \t \t 8. Exit \n\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

printf("Enter your option: ");

scanf("%d", &mainchoice);

getchar();

switch (mainchoice)

{

case 1:

create\_account();

break;

case 2:

login();

break;

case 3:

if (current\_customer)

{

book\_flight\_ticket();

}

else

{

printf("Please log in before booking a flight ticket.\n");

}

break;

case 4:

display\_ticket();

break;

case 5:

cancel\_ticket();

break;

case 6:

travel\_history();

break;

case 7:

display\_customer\_details();

break;

case 8:

exit(0);

default:

printf("Enter a valid option.\n");

}

}

}

// Function to create an account

void create\_account()

{

system("clear");

//Allocation of memory

CustomerNode \*new\_customer = (CustomerNode \*)malloc(sizeof(CustomerNode));

if (!new\_customer)

{

printf("Memory allocation failed\n");

return;

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CREATE NEW ACCOUNT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

new\_customer->customer.customer\_id = rand() % 100000;

printf("Enter your full name: \n");

getchar();

scanf("%[^\n]s", new\_customer->customer.cust\_name);

printf("Enter your email ID: \n");

scanf("%s", new\_customer->customer.cust\_mailid);

char password[200], confirm\_password[200];

while (1)

{

printf("Enter your password (at least 8 characters,with one uppercase,one lowercase,and one special character): \n");

scanf("%s", password);

if (validate\_password(password))

{

printf("Confirm your password: \n");

scanf("%s", confirm\_password);

if (strcmp(password, confirm\_password) == 0)

{

strcpy(new\_customer->customer.password, password);

break;

}

else

{

printf("Passwords do not match. Please try again.\n");

}

}

else

{

printf("Password does not meet the requirements. Please try again.\n");

}

}

printf("Enter your address: \n");

getchar();

scanf("%[^\n]s", new\_customer->customer.address);

printf("Enter your city: \n");

getchar();

scanf("%[^\n]s", new\_customer->customer.city);

printf("Enter your PIN code: \n");

scanf("%ld", &new\_customer->customer.pin\_code);

printf("Enter your mobile number: \n");

scanf("%ld", &new\_customer->customer.mno);

printf("Enter your country: \n");

getchar();

scanf("%[^\n]s", new\_customer->customer.country);

new\_customer->next = customer\_head;

customer\_head = new\_customer;

printf("Account created successfully!...Your Customer ID is: %ld\n\n\n", new\_customer->customer.customer\_id);

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

}

// Function definition for login page

void login()

{ system("clear");

char email[200];

char password[200];

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*LOGIN\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n\n");

printf("Enter your email ID: \n");

scanf("%s", email);

printf("Enter your password: \n");

scanf("%s", password);

CustomerNode \*temp = customer\_head;

while (temp != NULL)

{

if (strcmp(temp->customer.cust\_mailid, email) == 0 && strcmp(temp->customer.password, password) == 0)

{

current\_customer = temp;

printf("Login successful. Welcome....");

return;

}

temp = temp->next;

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

}

printf("Login failed. Please check your email and password.\n");

}

//function definition for booking operation

void book\_flight\_ticket()

{

system("clear");

//Allocation of memory

BookingNode \*new\_booking = (BookingNode \*)malloc(sizeof(BookingNode));

if (!new\_booking)

{

printf("Memory allocation failed\n");

return;

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TICKET BOOKING\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

printf("Select Flight Type:\n\n1. Domestic\t\t\t2. International\n\n");

int flight\_type;

printf("Enter your choice:\n");

scanf("%d", &flight\_type);

getchar();

printf("Select Trip Type:\n\n1. One Way\t\t\t 2. Round Trip\n\n");

int trip\_type;

printf("Enter your choice:\n");

scanf("%d", &trip\_type);

getchar();

new\_booking->booking.is\_round\_trip = (trip\_type == 2);

const char \*\*locations = flight\_type == 1 ? domestic\_locations : international\_locations;

int location\_count = 10;

// selection of departure details

printf("\n\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Departure\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

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

{

if(i==4||i==8||i==12)

{

printf("\n\n");

}

printf("%d. %s\t\t", i + 1, locations[i]);

}

int dep\_code, dest\_code;

printf("\n\nEnter your choice:\n");

scanf("%d", &dep\_code);

getchar();

// selection of destination details

printf("\n\n\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Destination\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

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

{

if (i + 1 != dep\_code)

{

if(i==4||i==8||i==12)

{

printf("\n\n");

}

printf("%d. %s\t\t", i + 1, locations[i]);

}

}

printf("\n\nEnter your choice:\n");

scanf("%d", &dest\_code);

getchar();

strcpy(new\_booking->booking.dep\_country, locations[dep\_code - 1]);

strcpy(new\_booking->booking.dest\_country, locations[dest\_code - 1]);

printf("\nSelected Depatrure: %s\tAnd\t",new\_booking->booking.dep\_country);

printf("Destination:%s\n\n",new\_booking->booking.dest\_country);

// selection of Flights

printf("\n\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Available Airlines\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

for (int i = 0; i < 10; i++)

{

float adult\_cost = flight\_type == 1 ? domestic\_adult\_costs[i] : international\_adult\_costs[i];

float child\_cost = flight\_type == 1 ? domestic\_child\_costs[i] : international\_child\_costs[i];

printf("%d. %s - %s (Adult: %.2f, Child: %.2f)\n", i + 1, airlines[i], airline\_codes[i], adult\_cost, child\_cost);

}

int airline\_choice;

printf("\n\nEnter your choice:\n");

scanf("%d", &airline\_choice);

getchar();

strcpy(new\_booking->booking.flight\_name, airlines[airline\_choice - 1]);

strcpy(new\_booking->booking.flight\_code, airline\_codes[airline\_choice - 1]);

fn=airline\_choice;

printf("\n\nSelect Seat Class:\n\n1. First Class (50%% premium)\t\t\t2. Premium Economy (20%% premium)\t\t\t3. Economy\n\n");

int seat\_class;

printf("Enter your choice:\n");

scanf("%d", &seat\_class);

getchar();

float class\_multiplier = (seat\_class == 1) ? 1.5 : (seat\_class == 2) ? 1.2 : 1.0;

strcpy(new\_booking->booking.classes\_of\_service, seat\_class == 1 ? "First Class" : seat\_class == 2 ? "Premium Economy" : "Economy");

printf("\nSelected class is: %s\n\n",new\_booking->booking.classes\_of\_service);

// Generating the dates of flights

srand(time(NULL));

struct tm start\_date = {0};

start\_date.tm\_year = 2024-1900;

start\_date.tm\_mon = 10;

start\_date.tm\_mday = 14;

time\_t current\_time = mktime(&start\_date);

printf("\n\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DEPARTURE DATE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf("The Flight you chooseare Avaliable in these Dates\n\n");

for (int i = 0; i < 5; ++i)

{

int random\_gap = 1 + rand() % 3;

current\_time += random\_gap \* 24 \* 60 \* 60;

struct tm \*new\_date = localtime(&current\_time);

printf("%02d-%02d-%d\t",new\_date->tm\_mday,new\_date->tm\_mon + 1,new\_date->tm\_year + 1900 );

}

// operatin to take Departure date

printf("\n\n\nEnter Departure Date (DD-MM-YYYY): ");

fgets(new\_booking->booking.date, sizeof(new\_booking->booking.date), stdin);

new\_booking->booking.date[strcspn(new\_booking->booking.date, "\n")] = 0;

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

// Seat selection

int seat\_count;

int seats[20];

display\_seat\_selection(seat\_class, seats, &seat\_count);

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

{

new\_booking->booking.seat\_numbers[i] = seats[i];

}

new\_booking->booking.seat\_count = seat\_count;

int num\_adults,num\_children;

printf("Enter number of adults: ");

scanf("%d", &num\_adults);

printf("Enter number of children: ");

scanf("%d", &num\_children);

new\_booking->booking.num\_adultss =num\_adults;

new\_booking->booking.num\_childrens =num\_children;

//for loop for taking the details of adults and children

for (int i = 0; i < num\_adults; i++)

{

printf("Enter adult name %d: ", i + 1);

scanf("%s", new\_booking->booking.adultname[i]);

printf("Enter age for %s: ", new\_booking->booking.adultname[i]);

scanf("%d", &new\_booking->booking.ages[i]);

}

for (int i = 0; i < num\_children; i++)

{

printf("Enter child name %d: ", i + 1);

scanf("%s", new\_booking->booking.childname[i]);

printf("Enter age for %s: ", new\_booking->booking.childname[i]);

scanf("%d", &new\_booking->booking.ages[num\_adults + i]);

}

// calculation of the cost

float base\_adult\_cost = flight\_type == 1 ? domestic\_adult\_costs[airline\_choice - 1] : international\_adult\_costs[airline\_choice - 1];

float base\_child\_cost = flight\_type == 1 ? domestic\_child\_costs[airline\_choice - 1] : international\_child\_costs[airline\_choice - 1];

float adult\_cost\_total = base\_adult\_cost \* num\_adults \* class\_multiplier;

float child\_cost\_total = base\_child\_cost \* num\_children \* class\_multiplier;

float round\_trip\_multiplier = new\_booking->booking.is\_round\_trip ? 2.0 : 1.0;

float total\_cost = (adult\_cost\_total + child\_cost\_total) \* round\_trip\_multiplier;

new\_booking->booking.ac=num\_adults;

new\_booking->booking.cc=num\_children;

new\_booking->booking.acost=adult\_cost\_total;

new\_booking->booking.ccost=child\_cost\_total;

new\_booking->booking.cost = total\_cost;

//printing the cost breakdown

printf("\n\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Cost Breakdown\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("Base Adult Cost per person: %.2f\n", base\_adult\_cost);

printf("Base Child Cost per person: %.2f\n", base\_child\_cost);

printf("Seat Class Multiplier: %.2f\n", class\_multiplier);

printf("Adult Total Cost: %.2f\n", adult\_cost\_total);

printf("Child Total Cost: %.2f\n", child\_cost\_total);

if (new\_booking->booking.is\_round\_trip)

{

printf("Round Trip Multiplier: %.2f\n", round\_trip\_multiplier);

}

printf("Total Cost: %.2f\n", total\_cost);

// Generate random booking ID

new\_booking->booking.bookingID = rand() % 100000;

printf("Booking ID: %d\n", new\_booking->booking.bookingID);

collect\_payment\_info(current\_customer->customer.saved\_card);

printf("Ticket booked successfully. Details sent to registered mobile number.\n");

new\_booking->next = booking\_head;

booking\_head = new\_booking;

}

//function definition for the seat displaying selection

void display\_seat\_selection(int seat\_class, int \*seats, int \*seat\_count )

{

//switch cases and conditions to make the seats quantity for different flight

int start\_seat, end\_seat;

switch(seat\_class)

{

case 1:

if(fn==1)

{

start\_seat = 1;

end\_seat = 8;

}

else if(fn==2)

{

start\_seat = 1;

end\_seat = 8;

}

else if(fn==3)

{

start\_seat = 1;

end\_seat = 20;

}

else if(fn==4)

{

start\_seat = 1;

end\_seat = 30;

}

else if(fn==5)

{

start\_seat = 1;

end\_seat = 40;

}

else if(fn==6)

{

start\_seat = 1;

end\_seat = 25;

}

else if(fn==7)

{

start\_seat = 1;

end\_seat = 35;

}

else if(fn==8)

{

start\_seat = 1;

end\_seat = 25;

}

else

{

start\_seat = 1;

end\_seat = 20;

}

break;

case 2:

if(fn==1)

{

start\_seat = 11;

end\_seat = 50;

}

else if(fn==2)

{

start\_seat = 11;

end\_seat = 40;

}

else if(fn==3)

{

start\_seat = 21;

end\_seat = 60;

}

else if(fn==4)

{

start\_seat = 31;

end\_seat = 70;

}

else if(fn==5)

{

start\_seat = 41;

end\_seat = 70;

}

else if(fn==6)

{

start\_seat = 26;

end\_seat = 70;

}

else if(fn==7)

{

start\_seat = 36;

end\_seat = 70;

}

else if(fn==8)

{

start\_seat = 26;

end\_seat = 60;

}

else

{

start\_seat = 21;

end\_seat = 50;

}

break;

case 3:

if(fn==1)

{

start\_seat = 61;

end\_seat = 120;

}

else if(fn==2)

{

start\_seat = 41;

end\_seat = 100;

}

else if(fn==3)

{

start\_seat = 61;

end\_seat = 120;

}

else if(fn==4)

{

start\_seat = 71;

end\_seat = 140;

}

else if(fn==5)

{

start\_seat = 71;

end\_seat = 140;

}

else if(fn==6)

{

start\_seat = 71;

end\_seat = 140;

}

else if(fn==7)

{

start\_seat = 71;

end\_seat = 140;

}

else if(fn==8)

{

start\_seat = 61;

end\_seat = 110;

}

else

{

start\_seat = 51;

end\_seat = 110;

}

break;

default:

return;

}

system("clear");

// desplaying the avaliable seats

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Available Seats\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf("A\t\tB\t\tD\t\tE\t\tF\n\n");

for (int i = start\_seat; i <= end\_seat; i++)

{

if (!is\_seat\_available[i])

{

printf("S-%d\t\t", i);

if(i==5||i==10||i==15||i==20||i==25||i==30||i==35||i==40||i==45||i==50||i==55||i==60||i==65||i==70||i==75||i==80||i==85||i==90||i==95||i==100||i==105||i==110||i==115||i==120||i==125||i==130||i==135)

{

printf("\n\n");

}

}

}

printf("\nEnter the number of seats to book: ");

scanf("%d", seat\_count);

for (int i = 0; i < \*seat\_count; i++)

{

int seat\_no;

printf("Select seat number %d: ", i + 1);

scanf("%d", &seat\_no);

if (seat\_no >= start\_seat && seat\_no <= end\_seat && !is\_seat\_available[seat\_no])

{

is\_seat\_available[seat\_no] = 1;

seats[i] = seat\_no;

}

else

{

printf("Invalid or already booked seat.\n");

i--; // Prompt again for a valid seat number

}

}

}

//function definition for the payment process

void collect\_payment\_info(char \*saved\_card)

{

printf("\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*P A Y M E N T S\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

char card\_number[20], card\_holder[50], pin[5];

printf("Enter Card Number (16 digits): ");

scanf("%s", card\_number);

printf("Enter Card Holder Name: ");

getchar();

fgets(card\_holder, 50, stdin);

card\_holder[strcspn(card\_holder, "\n")] = 0;

while (1)

{

printf("Enter 4-digit Card PIN: ");

scanf("%s", pin);

if (validate\_pin(pin))

{

printf("Payment confirmed.\n");

break;

}

else

{

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

}

}

}

//function definition for displaying the ticket

void display\_ticket()

{

int booking\_id;

printf("Enter Booking ID to display ticket: ");

scanf("%d", &booking\_id);

getchar();

system("clear");

BookingNode \*temp = booking\_head;

int gate=temp->booking.bookingID;

int acount=temp->booking.ac;

int ccount=temp->booking.cc;

if (gate < 0) {

gate = -gate;

}

while (gate >= 10) {

gate /= 10;

}

while (temp != NULL)

{

if (temp->booking.bookingID == booking\_id)

{

if(temp->booking.num\_adultss!=0)

{

for(int i=0; i<temp->booking.num\_adultss; i++)

{

printf("ADULT TICKET NO:%d\n\n",i+1);

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

printf("\t \* \t\t");

printf(" \*\*\*\* \*\*\* \*\n");

printf("\t\* \* \t\t");

printf("\* \* \* \* \*\n");

printf("\t \*\*\*\*\*\*\*\*\*\*\*\t\t");

printf(" \*\*\*\* \* \* \*\*\*\*\*\*\n");

printf("\t\* \* \t\t");

printf(" \* \* \* \* \* \n");

printf("\t \* \t\t");

printf(" \*\*\*\* \*\*\* \* \*\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

printf("\t\t\t\t B O A R D I N G P A S S \n\n\n");

printf("\tBOARDING PASS ID:%d\t\t\t\t", temp->booking.bookingID);

printf("|| ||||| |||||||| ||| |||||||\n\n");

printf("\tName of the passenger\n\n\t%s\t\t\t\t\t\t",temp->booking.adultname[i]);

printf("ADULT PASS\n\n\n");

printf("\tFrom\t\t Flight\t\t\tDate\n");

printf("\t%s\t\t", temp->booking.dep\_country);

printf(" %s\t\t\t",temp->booking.flight\_code);

printf(" %s\n\n", temp->booking.date);

printf("\tTo\n\t%s\n\n", temp->booking.dest\_country);

printf("\tClass: %s\t\t\t\t", temp->booking.classes\_of\_service);

printf("\tTrip Type: %s\n\n", temp->booking.is\_round\_trip ? "Round Trip" : "One Way");

printf("\tSeat\t\tGATE\t");

printf("\t\t\tTotal Cost: RS%.2f\n", temp->booking.acost/acount);

printf("\tS-%d\t\t ", temp->booking.seat\_numbers[i]);

printf("%d\n\n",gate);

printf("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n");

printf("\t\t\tPLEASE BE AT THE GATE AT BOARDING TIME\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

}

}

printf("\n\n");

if(temp->booking.num\_childrens!=0)

{

for(int i=0; i<temp->booking.num\_childrens; i++)

{

printf("CHILD TICKET NO:%d\n\n",i+1);

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

printf("\t \* \t\t");

printf(" \*\*\*\* \*\*\* \*\n");

printf("\t\* \* \t\t");

printf("\* \* \* \* \*\n");

printf("\t \*\*\*\*\*\*\*\*\*\*\*\t\t");

printf(" \*\*\*\* \* \* \*\*\*\*\*\*\n");

printf("\t\* \* \t\t");

printf(" \* \* \* \* \* \n");

printf("\t \* \t\t");

printf(" \*\*\*\* \*\*\* \* \*\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

printf("\t\t\t\t B O A R D I N G P A S S \n\n\n");

printf("\tBOARDING PASS ID:%d\t\t\t\t", temp->booking.bookingID);

printf("|| ||||| |||||||| ||| |||||||\n\n");

printf("\tName of the passenger\n\n\t%s\t\t\t\t\t\t",temp->booking.childname[i]);

printf("CHILD PASS\n\n\n");

printf("\tFrom\t\t Flight\t\t\tDate\n");

printf("\t%s\t\t", temp->booking.dep\_country);

printf(" %s\t\t\t",temp->booking.flight\_code);

printf(" %s\n\n", temp->booking.date);

printf("\tTo\n\t%s\n\n", temp->booking.dest\_country);

printf("\tClass: %s\t\t\t\t", temp->booking.classes\_of\_service);

printf("\tTrip Type: %s\n\n\n", temp->booking.is\_round\_trip ? "Round Trip" : "One Way");

printf("\tSeat\t\tGATE\t");

printf("\t\t\tTotal Cost: RS%.2f\n", temp->booking.ccost/ccount);

printf("\tS-%d\t\t ", temp->booking.seat\_numbers[i]);

printf("%d\n\n",gate);

printf("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n");

printf("\t\t\tPLEASE BE AT THE GATE AT BOARDING TIME\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

}

}

printf("\n\n");

return;

}

temp = temp->next;

}

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

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

}

// function definition for cancellation of ticket

void cancel\_ticket()

{

int booking\_id;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TICKET CANCELLATION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

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

scanf("%d", &booking\_id);

getchar();

system("clear");

BookingNode \*temp = booking\_head, \*prev = NULL;

while (temp != NULL && temp->booking.bookingID != booking\_id)

{

prev = temp;

temp = temp->next;

}

if (temp == NULL)

{

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

return;

}

printf("Confirm cancellation (1 for Yes, 0 for No): ");

int confirm;

scanf("%d", &confirm);

getchar();

if (confirm == 1)

{

if (prev == NULL)

booking\_head = temp->next;

else

prev->next = temp->next;

for (int i = 0; i < temp->booking.seat\_count; i++)

{

is\_seat\_available[temp->booking.seat\_numbers[i]] = 0;

}

free(temp);

printf("Ticket with Booking ID %d cancelled successfully.\n", booking\_id);

}

else

{

printf("Cancellation aborted.\n");

}

}

// function definition for displaying the Customer details

void display\_customer\_details()

{

CustomerNode \*temp = customer\_head;

system("clear");

while (temp != NULL)

{

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PROFILE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n\n");

printf("Customer ID is: %ld\n", temp->customer.customer\_id);

printf("Name: %s\n", temp->customer.cust\_name);

printf("Email ID: %s\n", temp->customer.cust\_mailid);

printf("Address: %s\n", temp->customer.address);

printf("City: %s\n", temp->customer.city);

printf("PIN Code: %ld\n", temp->customer.pin\_code);

printf("Country: %s\n", temp->customer.country);

printf("Mobile Number: %ld\n", temp->customer.mno);

return;

temp = temp->next;

}

}

//function definition for viewin the travel history

void travel\_history()

{

if (!current\_customer)

{

printf("Please log in to view travel history.\n");

return;

}

long int entered\_customer\_id;

printf("Enter Customer ID to view travel history: ");

scanf("%ld", &entered\_customer\_id);

getchar();

system("clear");

// Check if the entered customer ID matches the current logged-in customer's ID

if (entered\_customer\_id != current\_customer->customer.customer\_id)

{

printf("Customer ID does not match the logged-in account.\n");

return;

}

BookingNode \*temp = booking\_head;

int found = 0;

printf("Travel History for Customer ID: %ld\n", entered\_customer\_id);

while (temp != NULL)

{

if (current\_customer->customer.customer\_id == entered\_customer\_id)

{

found = 1;

printf("Booking ID: %d\n", temp->booking.bookingID);

printf("Flight Name: %s (%s)\n", temp->booking.flight\_name, temp->booking.flight\_code);

printf("Departure Date: %s\n", temp->booking.date);

printf("Class of Service: %s\n", temp->booking.classes\_of\_service);

printf("Total Cost: %.2f\n", temp->booking.cost);

printf("-------------------------------------\n");

}

temp = temp->next;

}

if (!found)

{

printf("No travel history found for Customer ID %ld.\n", entered\_customer\_id);

}

}

// function definition for validating the password

int validate\_password(char \*password)

{

int has\_upper = 0, has\_lower = 0, has\_special = 0, length = strlen(password);

if (length < 8) return 0;

for (int i = 0; i < length; i++)

{

if (isupper(password[i])) has\_upper = 1;

else if (islower(password[i])) has\_lower = 1;

else if (ispunct(password[i])) has\_special = 1;

}

return has\_upper && has\_lower && has\_special;

}

// function definition for validating the PIN

int validate\_pin(char \*pin)

{

if (strlen(pin) != 4) return 0;

for (int i = 0; i < 4; i++)

{

if (pin[i] < '0' || pin[i] > '9')

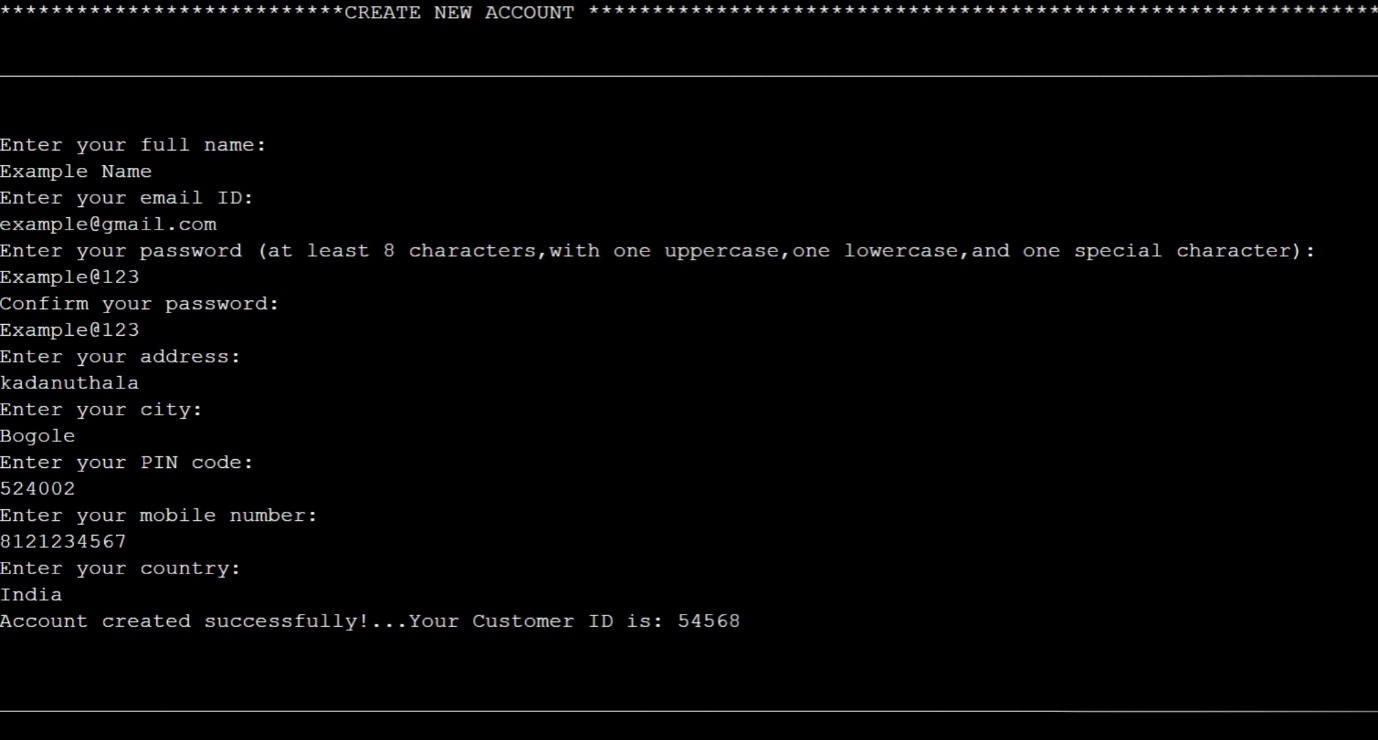
return 0;

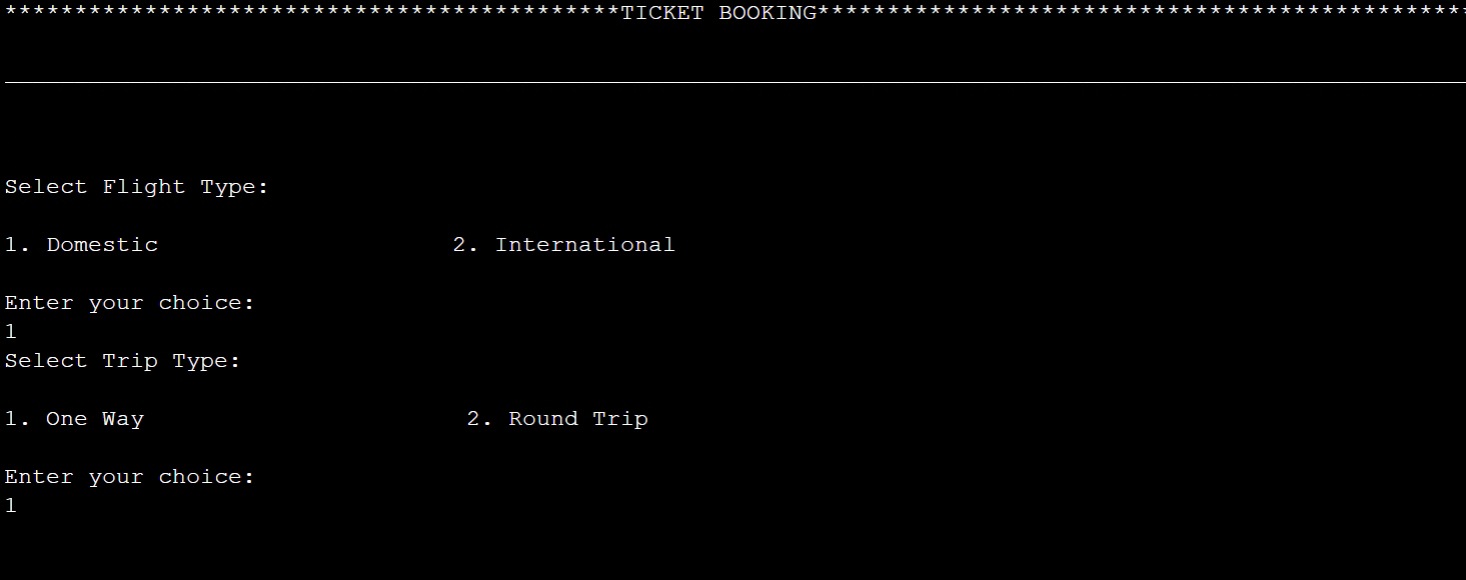
}

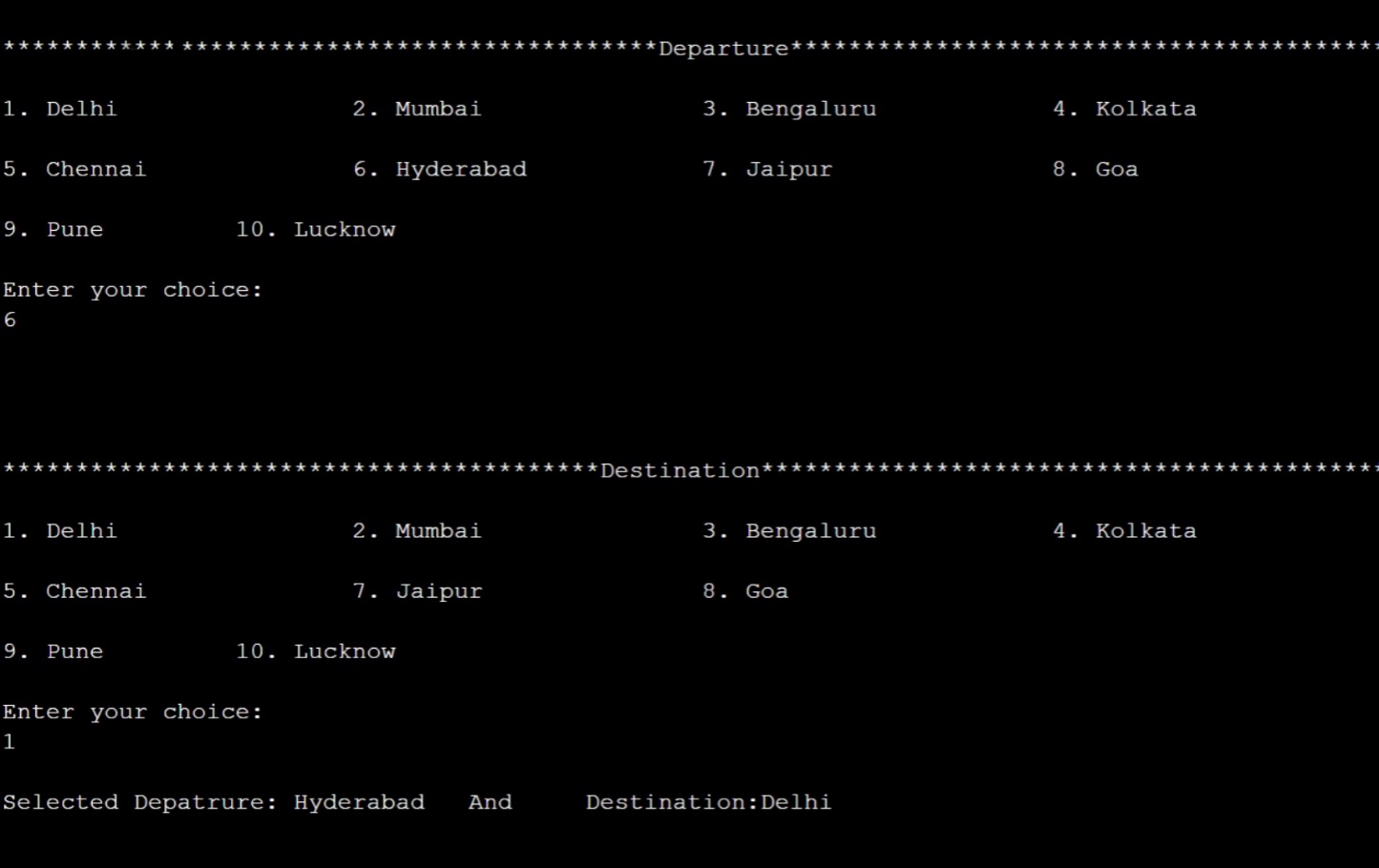
return 1;

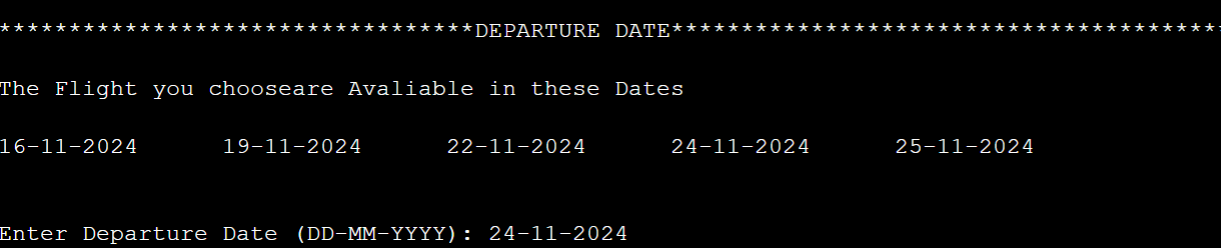
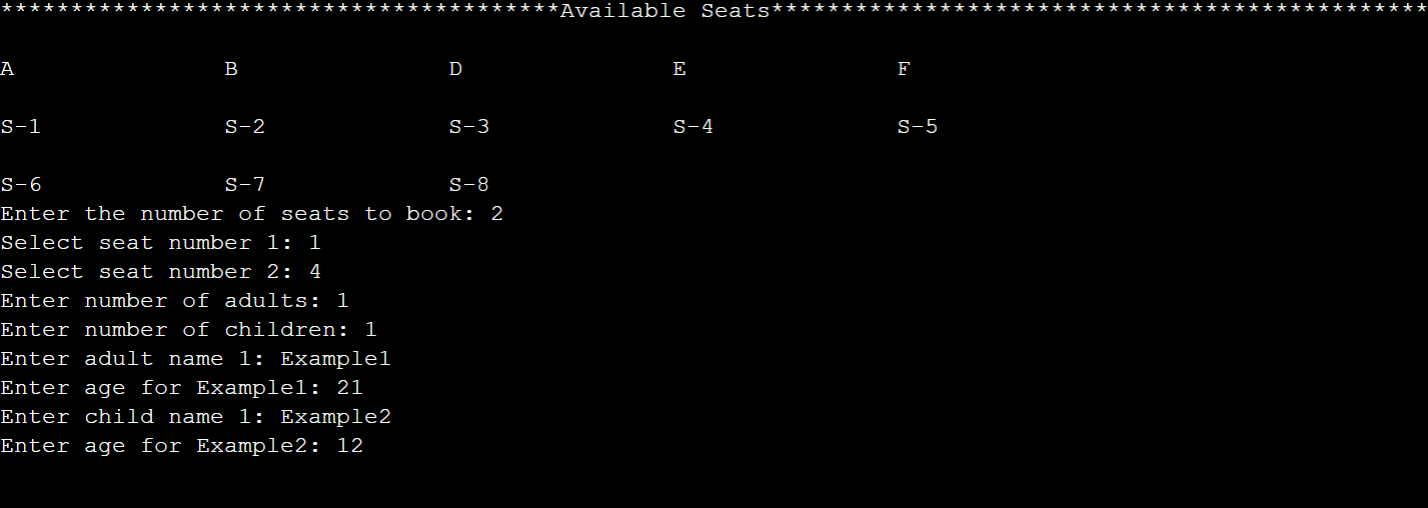
}

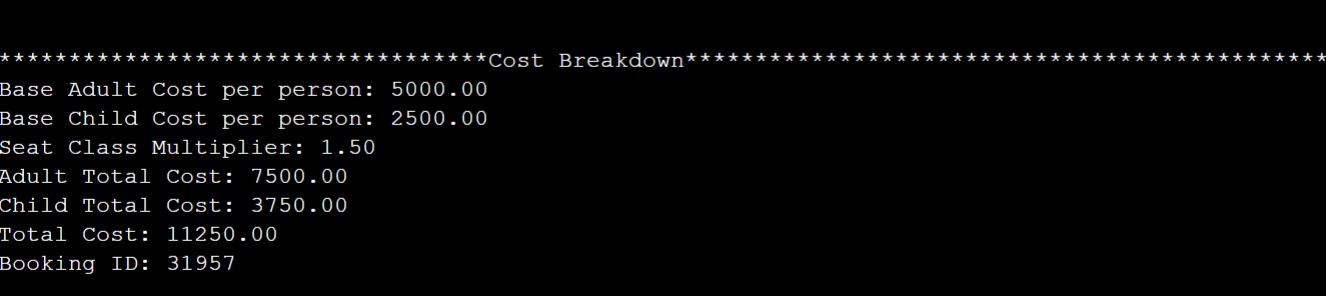
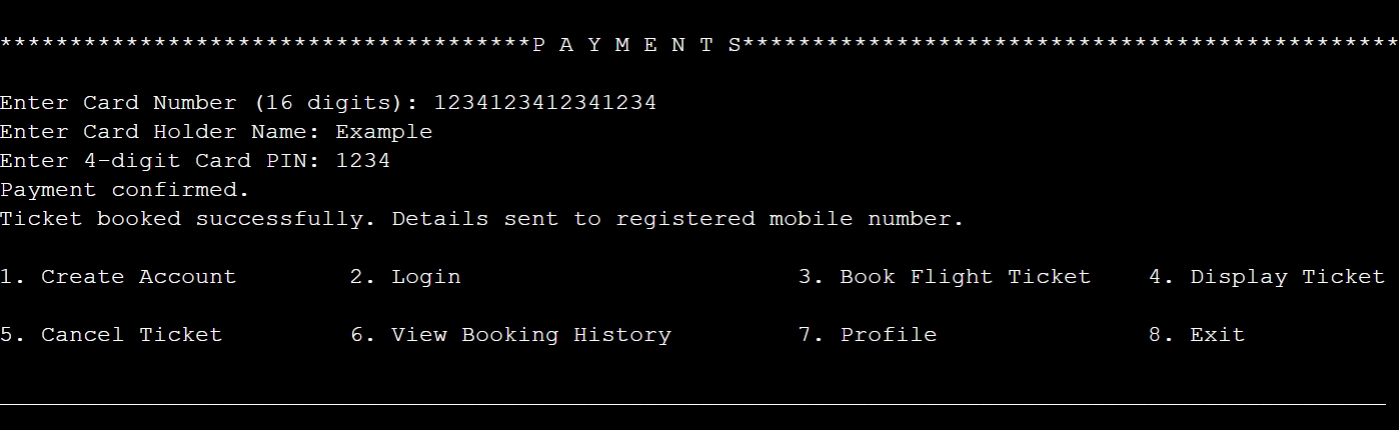
Outputs:





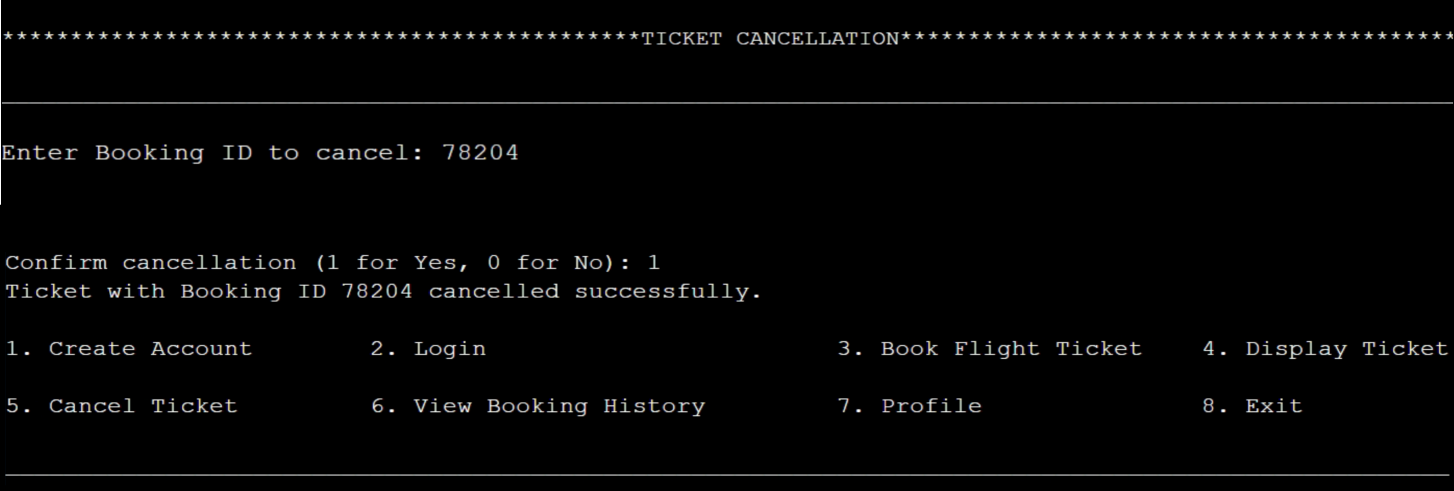


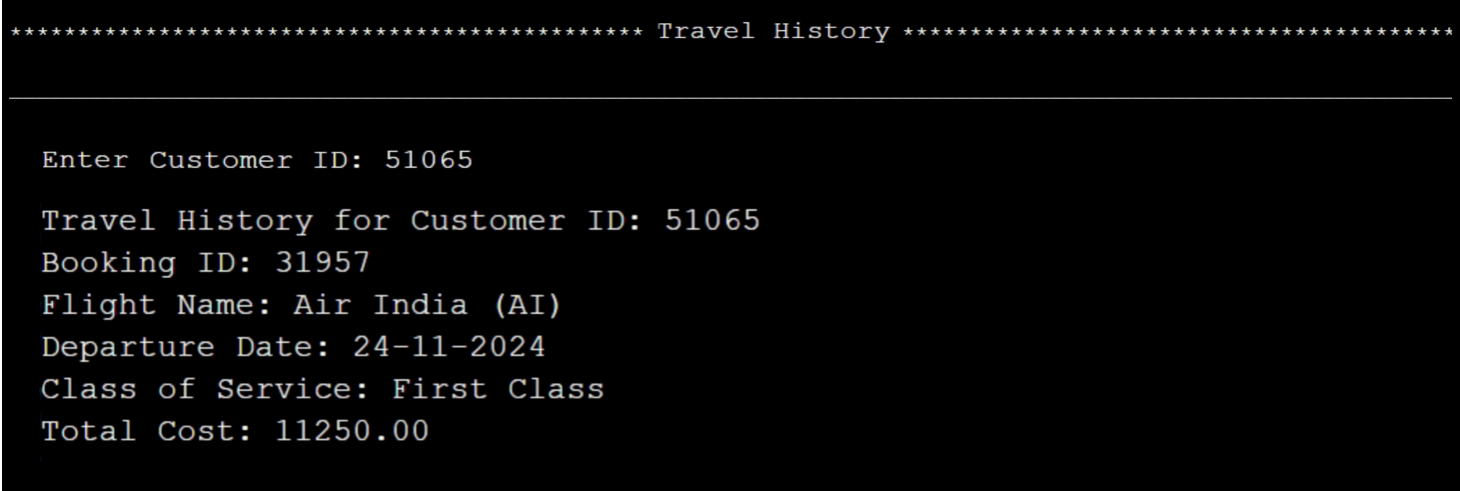


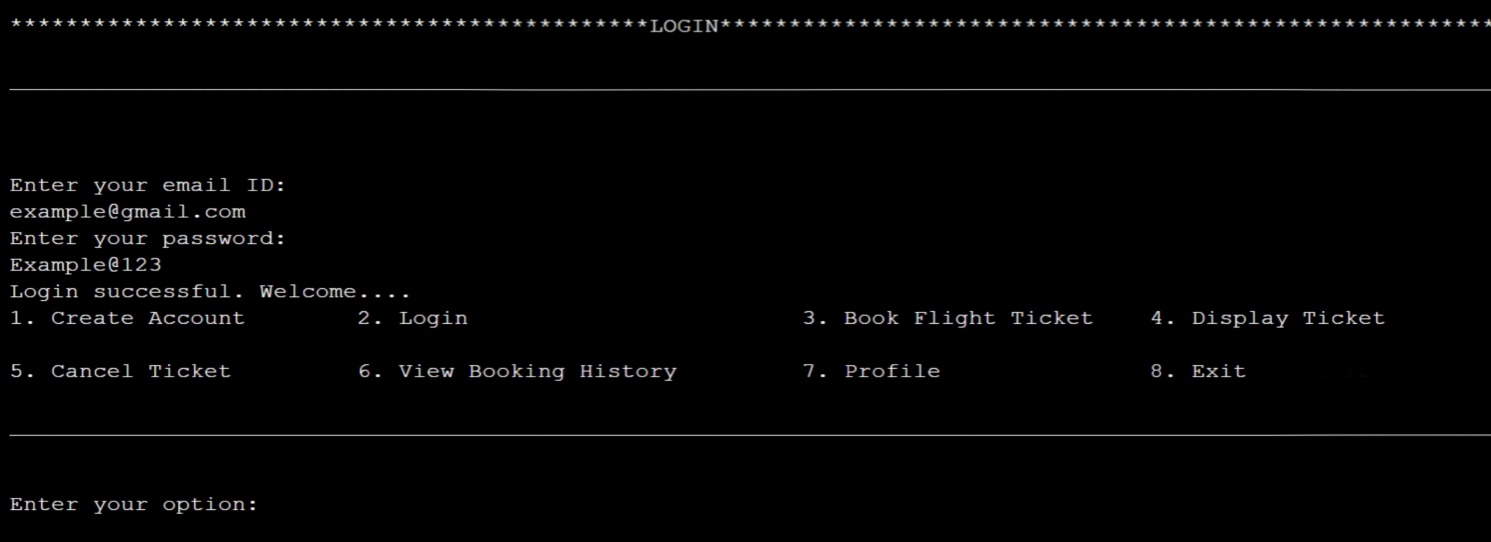












Conclusion:

The flight ticket booking application is one of the greatest improvements ever made to make the traveling planning process easy. Using user-friendly interfaces, secure payment systems, and real-time flight information, the application helps in making a strong customer convenience experience. This digital form of air travel not only clears up the confusion surrounding the journey but also creates trust among its users based on transparency and efficiency. With the strength of its functionalities and ease of accessibility, the app has the potential to transform the way travel is contemplated and carried out to be more personalized, efficient, and hassle-free.

Future Upgrades:

1. AI-driven personalization:

Analysis of user preferences through AI algorithms for flight recommendations, offers, and travel opportunities

2. Voice Command Capability:

Introduce voice command operation that lets users search, book, and manage their tickets with voice assistants, ensuring a hands-free experience

3. Multimodal Integration of Travel:

Extend the platform for use to include other modes of transport, such as trains, buses, and car rentals, to give users end-to-end travel planning capabilities.

4. Sustainability Options:

Give features that allow users to filter flights based on carbon emissions, enabling contributions towards environmental initiatives through their booking.

5. Augmented Reality Navigation:

Implement AR for airport navigation in real-time to guide passengers to gates, lounges, and services.

6. Blockchain for Enhanced Security:

Adoptions of Blockchain to ensure safe ticketing and transaction management to maintain transparency with fraud prevention.

7. Loyalty and Reward Schemes:

Gamify loyalty schemes by giving rewards and a discount to users on frequent booking.

8. Multi-Currency and Crypto Payments:

Accept cryptocurrencies and multi-currency transactions for international customers.

9. Real-time Updates through Push:

Allow real-time updates such as flight status changes, gate assignments, weather conditions as push notifications.

10. Offline Capability:

To enable users to access their bookings and itineraries even without an internet connection, develop offline features.

The solutions mentioned above will not only enhance the user experience but also position the application as the cutting-edge solution in the competitive travel industry.