National Public School Koramangala

Computer Science

Project Report

Airline Reservation System (AIRS)



*By: Ayush Agrawal*

*Rajat Agarwal*

*Pavan Bykampadi*

# Index

Acknowledgement Page 3

Introduction Page 4

Algorithm Page 6

Flow Charts Page 8

Source Code Page 10

Screenshots Page 40

# Acknowledgements:

We would like to thank our Computer Science teachers Chandita Ma’am and Kavitha Ma’am for giving us the opportunity to work on this project and showcase our talents. The project would not have been possible without their continued guidance and support.

# Introduction

The Airline Reservation System (AIRS) is an interface for users and airlines to interact and connect to each other. It makes it convenient for both airlines and consumers to discover each other, providing a single platform for users to book flights across airlines. As a result, it is a one-stop program for a consumers who wish to book air travel tickets to and from their favorite destinations. Moreover, it also helps airlines manage their flight logistics and schedules across destinations.

Consumers can search for flights to and from their required destinations, getting instant results. They can then pick and choose which flight they want to book, enter the number of seats, fill in their payment details and complete their booking. If they wish to, they can view or cancel it at a later date.

Similarly, airlines can list their flights on to the platform, making it available for consumers to book. They can also view their old listings, and delete them if they wish. Thus, it is a convenient platform for them to work on.

# Algorithm:

***General Algorithm:***

1. Start
2. Enter login details
3. Go to user or admin menu based on login details
4. Check if user wants to log out
5. If yes, stop
6. If no, go to step 4

***User:***

1. Start
2. Menu displayed: book flights, view current bookings, exit.
3. If user chooses book flight
   1. Input search criteria
   2. Display results
   3. Confirm, book and pay
4. If user chooses view booking status
   1. Displays current bookings
   2. If user wants to cancel booking, booking is cancelled
   3. If user wants to exit cancellation menu, program returns to step 2
5. If user wants to exit program, program stops

***Algorithm if admin:***

1. Start
2. Menu displayed: Add flight, delete flight, display all flights, logout.
3. If admin chooses to add flight
   1. Flight details are input
   2. A resulting flight is created
4. If admin chooses to delete flight
   1. All flights are displayed
   2. Serial number of flight to be deleted is taken as an input
   3. Corresponding flight is deleted
5. If admin wants to logout, program stops

# Flow Charts:

*General Flow:*

**Enter log in credentials**

**User or Admin**

**User menu and functions**

**Admin menu and functions**

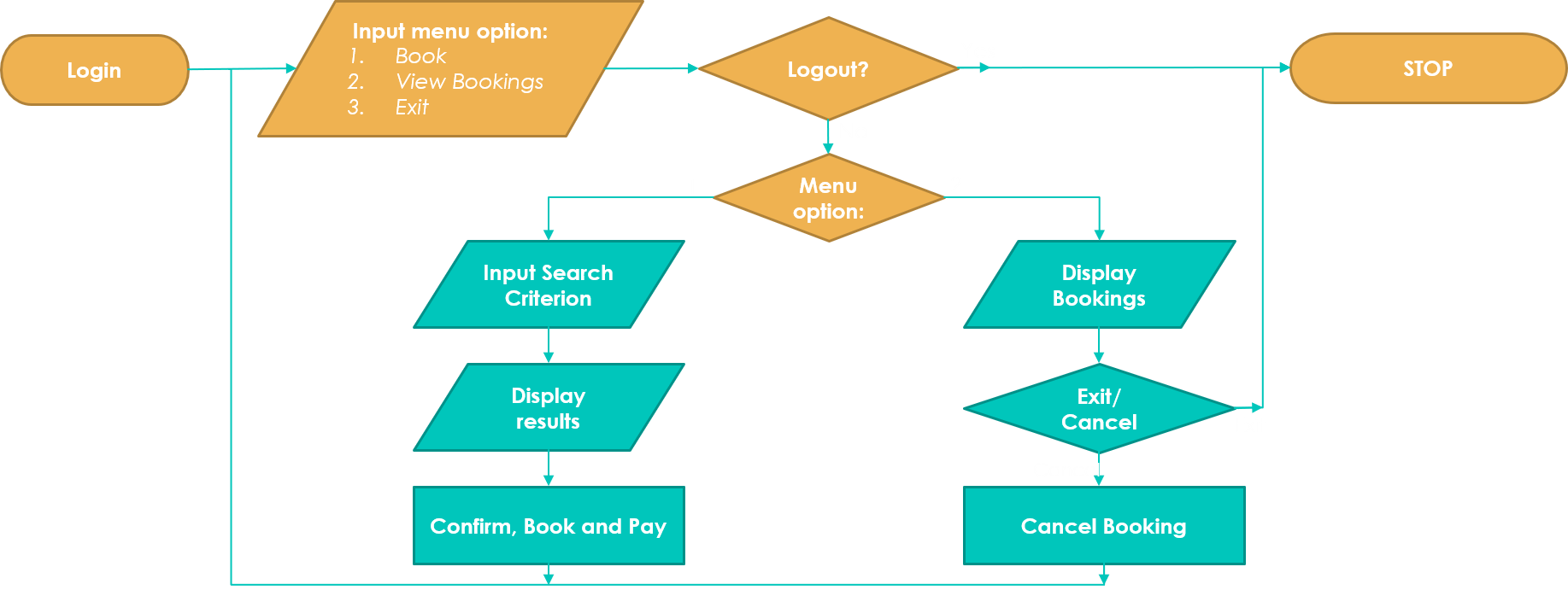
**Log out?**

**START**

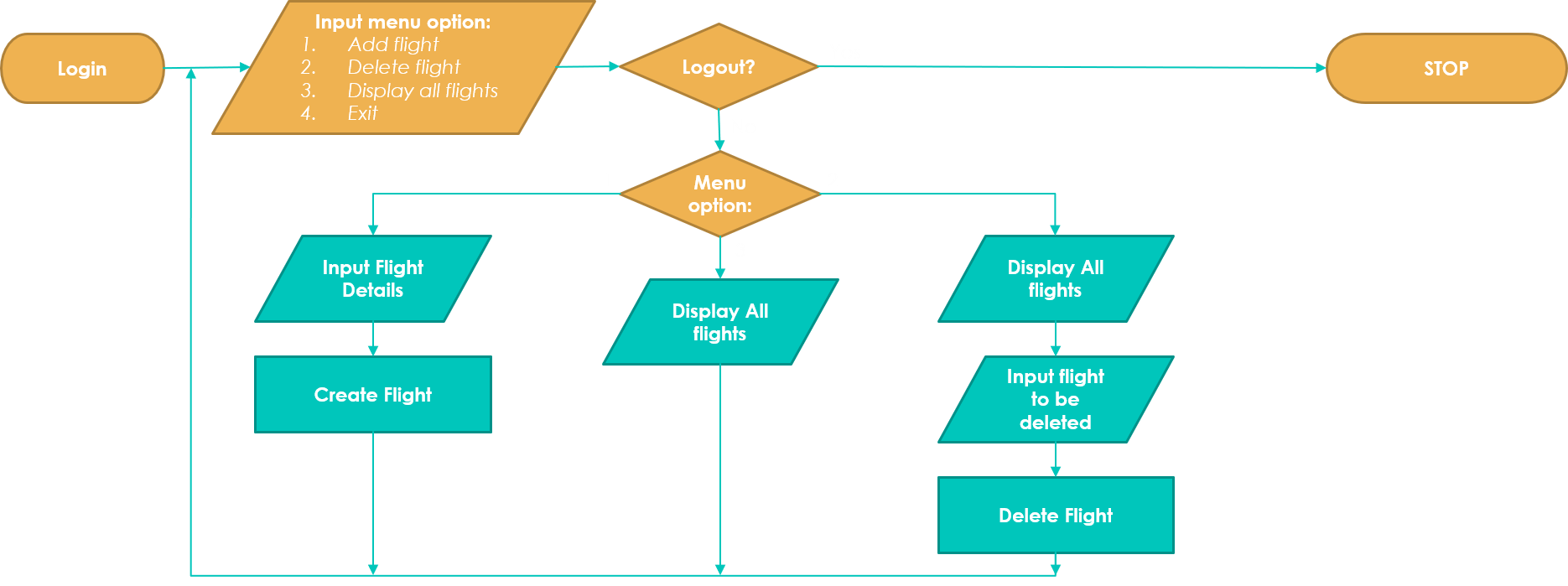
**STOP**

User

Admin

*User Flow*

*Admin Flow*

**

# Source code:

# /\*

# Airline Reservation System

# Date: 08/12/2017

# Author: @Ayush Agrawal, @Rajat Agarwal, @Pavan Bykampadi

# \*/

# #include <iostream>

# #include <stdio.h>

# #include <stdlib.h>

# #include <iomanip>

# #include <conio.h>

# #include <process.h>

# #include <string.h>

# using namespace std;

# /\*

# STRUCTURES

# \*/

# struct Date

# {

# int day,

# month,

# year;

# };

# struct Booking{

# int pos,

# seats;

# };

# struct Flight

# {

# char num[6],

# al[20],

# src[4],

# dst[4];

# int seats,

# i, //index of booking

# etdh, //etd hour

# etah; //eta hour

# Booking booked[20]; //users who have booked

# double price;

# Date etd;

# } flight[20];

# struct User

# {

# char name[20],

# pwd[20];

# int bcount,

# type; //0 - user, 1 - airline

# Booking booked[20]; //users who have booked

# } user[10];

# //GLOBAL VARIABLES:

# int n = 5, upos = 0, bcount; //number of flights, user logged in

# /\*

# Functions for AIRS Flights

# Date: 19/12/2017

# Author: @Ayush Agrawal, @Rajat Agarwal, @Pavan Bykampadi

# \*/

# //Declarations:

# int main();

# void table\_line(int i, int sl, int seats);

# void admin();

# void userf();

# void password();

# void display\_all\_flights();

# void add\_flight();

# void del\_flight();

# void status();

# void pay(int pos, int seats);

# void confirmation(int pos);

# void search();

# void front\_page();

# void bookings();

# //Definitions

# //hardcoded value presets

# void preset\_vals()

# {

# //Users

# strcpy(user[0].name, "ayush");

# strcpy(user[1].name, "rajat");

# strcpy(user[2].name, "pavan");

# strcpy(user[3].name, "airline");

# strcpy(user[0].pwd, "ayush");

# strcpy(user[1].pwd, "rajat");

# strcpy(user[2].pwd, "pavan");

# strcpy(user[3].pwd, "airline");

# user[0].bcount = 0;

# user[1].bcount = 0;

# user[2].bcount = 0;

# user[3].bcount = 0;

# //presets all values storing index of booked flights to -1

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

# {

# for(int j = 0; j < 10 ; j++)

# {

# user[i].booked[j].pos=-1;

# }

# }

# //presets all values storing index of bookings to -1

# for (int i = 0; i < 20; i++){

# for (int j = 0; j < 20; j++){

# flight[i].booked[j].pos = -1;

# }

# }

# user[0].type = 0;

# user[1].type = 0;

# user[2].type = 0;

# user[3].type = 1;

# //Flights

# flight[0].price=20000;

# flight[0].seats=150;

# flight[0].etdh=1200;

# flight[0].etah=1400;

# flight[0].etd.day=1;

# flight[0].etd.month=2;

# flight[0].etd.year=2018;

# flight[1].price=25000;

# flight[1].seats=150;

# flight[1].etdh=1100;

# flight[1].etah=1500;

# flight[1].etd.day=1;

# flight[1].etd.month=2;

# flight[1].etd.year=2018;

# flight[2].price=27000;

# flight[2].seats=150;

# flight[2].etdh=1000;

# flight[2].etah=1400;

# flight[2].etd.day=8;

# flight[2].etd.month=10;

# flight[2].etd.year=2018;

# flight[3].price=30000;

# flight[3].seats=180;

# flight[3].etdh=2000;

# flight[3].etah=2350;

# flight[3].etd.day=1;

# flight[3].etd.month=2;

# flight[3].etd.year=2018;

# flight[4].price=20000;

# flight[4].seats=150;

# flight[4].etdh=1200;

# flight[4].etah=1400;

# flight[4].etd.day=1;

# flight[4].etd.month=2;

# flight[4].etd.year=2018;

# strcpy(flight[0].al,"Emirates");

# strcpy(flight[0].src,"blr");

# strcpy(flight[0].dst,"dxb");

# strcpy(flight[0].num,"em449");

# strcpy(flight[1].al,"Emirates");

# strcpy(flight[1].src,"dxb");

# strcpy(flight[1].dst,"gnv");

# strcpy(flight[1].num,"em364");

# strcpy(flight[2].al,"Air France");

# strcpy(flight[2].src,"blr");

# strcpy(flight[2].dst,"cdg");

# strcpy(flight[2].num,"ai489");

# strcpy(flight[3].al,"Ethihad");

# strcpy(flight[3].src,"blr");

# strcpy(flight[3].dst,"dxb");

# strcpy(flight[3].num,"et420");

# strcpy(flight[4].al,"Lufthansa");

# strcpy(flight[4].src,"blr");

# strcpy(flight[4].dst,"dxb");

# strcpy(flight[4].num,"lu287");

# }

# //display welcome screen

# void front\_page()

# {

# cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_";

# cout<<"\n| |";

# cout<<"\n| |";

# cout<<"\n| \_\_\_ \_\_ .\_\_\_\_\_\_ \_\_\_\_\_\_\_. |";

# cout<<"\n| / \\ | | | \_ \\ / | |";

# cout<<"\n| / ^ \\ | | | |\_) | | (----` |";

# cout<<"\n| / /\_\\ \\ | | | / \\ \\ |";

# cout<<"\n| / \_\_\_\_\_ \\ | | | |\\ \\----..----) | |";

# cout<<"\n| /\_\_/ \\\_\_\\ |\_\_| | \_| `.\_\_\_\_\_||\_\_\_\_\_\_\_/ |";

# cout<<"\n| |";

# cout<<"\n| |";

# cout<<"\n| Press Any key to start! |";

# cout<<"\n| |";

# cout<<"\n|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|";

# getch();

# }

# //password input: to display echoed characters as \*

# void password(char pass[])

# {

# int i;

# cout << "\nPassword: \n";

# //input each character

# for(i = 0; i < 20 ; i++)

# {

# pass[i]=getch();

# if(pass[i] !=8 && pass[i] != '\r') //Echoes \* for each non-backspace/enter character entered

# {

# cout << "\*";

# }

# if(pass[i] == '\r') //Exits loop if user presses enter

# break;

# else if(pass[i] == 8 && i > 0) //Removes two \* and decrements index on entering backspace

# {

# i-=2;

# putch('\b');

# putch(' ');

# putch('\b');

# }

# }

# pass[i]='\0';

# }

# //draws a line extending 120 pixels of screen width

# void drawline()

# {

# cout << "\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

# }

# //draws a common header at the top of each page

# void header()

# {

# system("cls");

# cout << setw(100) << "Logged in as: " << user[upos].name;

# drawline();

# cout << "\t\t\t\t\tAIRLINE RESERVATION SYSTEM";

# drawline();

# }

# //outputs table column headings

# void table\_header()

# {

# cout << "\n\n" << setw(5) << "Sl" << setw(15) << "Flight" << setw(25) << "Airline" << setw(25) << "Departure" << setw(20) << "Arrival" << setw(15) << "Price" << setw(10) << "Seats";

# cout << endl;

# }

# //outputs a single table row entry

# void table\_line(int i, int sl, int seats)

# {

# cout << '\n' << setw(5) << sl << setw(15) << flight[i].num << setw(25) << flight[i].al;

# cout << setw(7) << flight[i].src << setw(4) << flight[i].etd.day << '/' << setw(2) << flight[i].etd.month << '/' << setw(4) << flight[i].etd.year << setw(6) << flight[i].etdh;

# cout << setw(13) << flight[i].dst << setw(7) << flight[i].etah << setw(15) << flight[i].price << setw(10) << seats;

# }

# //displays all flights

# void display\_all\_flights()

# {

# header();

# table\_header();

# for(int i=0; i<n; i++)

# table\_line(i, i+1, flight[i].seats);

# cout << "\n\n";

# }

# //display flight bookings

# void bookings(){

# int sl = 1, fpos, ind, uind;

# header();

# table\_header();

# for (int i = 0; i < n; i++){

# table\_line(i, i+1, flight[i].seats);

# }

# do{

# cout << "\n\nEnter flight to view details: \n";

# cin >> fpos;

# } while (fpos < 1 || fpos > n+1);

# fpos--;

# ind = flight[fpos].i;

# cout << setw(5) << "Sl" << setw(20) << "Booking Name" << setw(10) << "Seats";

# for(int i = 0; i < 20; i++)

# {

# for(int j = 0; j <= ind; j++)

# {

# if(flight[fpos].booked[j].pos == i)

# {

# cout << endl << setw(5) << sl << setw(20) << user[i].name << setw(10) << flight[fpos].booked[j].seats;

# sl++;

# }

# }

# }

# getch();

# }

# //add a flight as an airline admin

# void add\_flight()

# {

# header();

# char fno[6];

# int i;

# //enter airline

# cout << "\nEnter Airline: ";

# cin >> flight[n].al;

# //input flight time

# cout << "\n\nEnter Date of Departure: (DD MM YYYY): " << endl;

# cin >> flight[n].etd.day >> flight[n].etd.month >> flight[n].etd.year;

# cout << "\n\nEnter Estimated time of Departure (24 hour): " << endl;

# cin >> flight[n].etdh;

# cout << "\n\nEnter Estimated time of Arrival (24 hour): " << endl;

# cin >> flight[n].etah;

# //generates flight number (eg AI250 etc)

# for (i = 0; i < 2; i++)

# fno[i] = flight[n].al[i];

# for (; i < 5; i++)

# fno[i] = rand() % 10 + 48;

# fno[6]='\0';

# strcpy(flight[n].num, fno);

# //input destination and source

# cout << "\nEnter Source Airport: " << endl;

# cin >> flight[n].src;

# cout << "\n\nEnter Destination Airport: " << endl;

# cin >> flight[n].dst;

# cout << "\nenter cost per seat : ";

# cin >> flight[n].price;

# n++;

# //continue

# cout << "\nEnter any key to continue";

# getch();

# }

# //delete a flight as an airline admin

# void del\_flight()

# {

# display\_all\_flights();

# cout << "\n\nEnter sl no of flight to be deleted : ";

# int pos;

# cin >> pos;

# for (int i = pos - 1; i <= n; i++) //deletes flight entry from array

# flight[i] = flight[i+1];

# n--;

# }

# //displays all flights booked by user and allows cancellation

# void status()

# {

# int cpos, sl = 1, fpos, ind;

# char ch, conf;

# bcount = user[upos].bcount;

# header();

# table\_header();

# for(int i = 0; i < n; i++)

# {

# for(int j = 0; j <= bcount; j++)

# {

# if(user[upos].booked[j].pos == i)

# {

# table\_line(i, sl, user[upos].booked[j].seats);

# sl++;

# break;

# }

# }

# }

# do

# {

# cout << "\n\nPress R to return to main menu";

# cout << "\nPress C to cancel a booking";

# ch = getch();

# //Cancel flight booking

# if(ch=='C' || ch=='c')

# {

# cout << "\nenter serial number of flight to cancel: ";

# cin >> cpos;

# if(cpos-1>=0)

# {

# fpos = user[upos].booked[cpos-1].pos; //index of flight as stored in the Flights array

# //index of entry in flight bookings

# for (int i = 0; i < 20; i++){

# if (flight[fpos].booked[i].pos == upos){

# ind = i;

# break;

# }

# }

# if(strcmp(flight[fpos].src, "") !=0) //if flight entry exists

# {

# cout << "\nCancel flight from "<<flight[fpos].src<<" to "<<flight[fpos].dst<<"?\nPress C to confirm : ";

# cin >> conf;

# if(conf=='C' || conf=='c')

# {

# user[upos].bcount--;

# bcount--;

# flight[fpos].i--;

# //deleting entry

# for (int i = cpos-1; i < bcount; i++){

# user[upos].booked[i] = user[upos].booked[i+1];

# }

# for (int i = ind; i < flight[fpos].i; i++){

# flight[fpos].booked[i] = flight[fpos].booked[i+1];

# }

# }

# }

# }

# }

# else if (ch != 'r' && ch != 'R') //Error handling for invalid inputs

# {

# cout << "\n\n\nPlease enter valid input";

# }

# } while (ch != 'c' && ch != 'C' && ch != 'r' && ch != 'R');

# }

# //payment page for flights

# void pay(int pos, int seats)

# {

# header();

# int CVV, card\_no;

# cout << "\nenter card no. : ";

# cin >> card\_no;

# cout << "\n\nenter CVV : ";

# cin >> CVV;

# cout << "\nproccessing payment...";

# for (double aa = 0; aa < 100000000; aa++) {} //delay timer

# bcount = user[upos].bcount;

# int ind = flight[pos].i;

# flight[pos].booked[ind].pos = upos;

# flight[pos].booked[ind].seats = seats;

# flight[pos].i++;

# user[upos].booked[bcount].pos = pos;

# user[upos].booked[bcount].seats = seats;

# user[upos].bcount++;

# bcount++;

# }

# //flight confirmation page

# void confirmation(int pos)

# {

# int seats, flag = 1;

# do{

# flag = 1;

# header();

# cout << "\nenter no. of seats : ";

# cin >> seats;

# flight[pos].seats -= seats;

# if (flight[pos].seats < 0){

# flag = 0;

# flight[pos].seats += seats;

# cout << "\nSorry, only " << flight[pos].seats << " seats are available";

# getch();

# }

# } while (!flag);

# cout << "\nPrice payable : " << flight[pos].price \* seats;

# cout << "\n\n" << "press any key to continue to payment page";

# getch();

# pay(pos,seats);

# }

# //searches for flights

# void search()

# {

# header();

# char src[4], des[4];

# int day, month, year;

# cout << "\nenter source airport code : ";

# cin >> src;

# cout << "\nenter destination airport code : ";

# cin >> des;

# cout << "\nenter date of departure : (DD MM YYYY) : ";

# cin >> day >> month >> year;

# int sl = 0, result[10], j = 0, bpos; //result stores the search results with user friendly indices (j)

# //bpos is sl no of flight to be booked

# //display search results in tabular format

# table\_header();

# for(int i=0; i<n; i++)

# {

# if ( strcmpi(src,flight[i].src) == 0 && strcmpi(des,flight[i].dst) == 0 && flight[i].etd.day == day && flight[i].etd.month == month && flight[i].etd.year == year && flight[i].seats > 0) //flight results - origin, destination and date

# {

# result[j] = i;

# j++;

# sl++;

# table\_line(i, sl, flight[i].seats);

# }

# }

# if(sl != 0)

# {

# cout << "\n\n\nEnter serial no of flight to book : ";

# cin >> bpos;

# confirmation(result[bpos-1]); //result[bpos-1] gives corresponding index of flight as stored in Flights array

# }

# else

# {

# cout << "\n\n\nSorry!No flights match your request";

# getch();

# }

# }

# //main home page for a user logged in

# void userf()

# {

# int flag = 1, ch = 1;

# bcount = user[upos].bcount;

# do

# {

# if(ch > 3 || ch < 0)

# cout << "Please enter value between 1 and 3";

# header();

# cout << "\n1: Book flights\n2: View current booking\n3: Exit";

# cout << "\n\noption: ";

# cin >> ch;

# switch(ch)

# {

# case 1:

# search();

# break;

# case 2:

# status();

# break;

# case 3:

# cout << "\nThank you for using our services!!";

# flag=0;

# break;

# }

# ch = 0;

# } while(flag && (ch > 3 || ch < 1)); //checks for correct input and exit

# }

# //airline main home page

# void admin()

# {

# int flag = 1,ch = 1;

# do

# {

# if (ch > 5 || ch < 1) //error handling

# cout << "\nEnter valid option";

# ch = 1;

# header();

# cout << "\n1. Add flight\n2. Delete flight\n3. Display Flights\n4. View Bookings\n5. Logout";

# cout << "\n\nOption : ";

# cin >> ch;

# switch(ch)

# {

# case 1:

# add\_flight();

# break;

# case 2:

# del\_flight();

# break;

# case 3:

# display\_all\_flights();

# getch();

# break;

# case 4:

# bookings();

# break;

# case 5:

# flag = 0;

# break;

# }

# ch = 0;

# } while (flag && (ch > 5 || ch < 1));

# }

# //main where execution starts - login page

# int main()

# {

# preset\_vals();

# front\_page();

# char userName[20];

# char pass[20];

# int loginAttempt = 0, i, flag = 1;

# while (loginAttempt < 5 && flag == 1)

# {

# system("cls");

# drawline(); //output formatting

# cout << "\t\t\t\t\tAIRLINE RESERVATION SYSTEM";

# drawline();

# cout<<"\nPlease enter your user name: ";

# cin >> userName;

# password(pass); //input password

# loginAttempt++; //keeps track of number of login attempts

# //checks username and password

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

# {

# if(strcmp(userName, user[i].name)==0 && strcmp(pass, user[i].pwd)==0)

# {

# upos = i;

# if (user[i].type == 0)

# userf();

# else

# admin();

# loginAttempt=0;

# flag = 1; //login successful

# break;

# }

# }

# if(i==10)

# {

# cout << "\n\nInvalid username or password";

# getch();

# }

# }

# if (loginAttempt == 5)

# {

# cout<<"\nToo many login attempts! The program will now terminate.";

# }

# cout<<"\nHave a good day!!\n";

# return 0;

# }

# Screen-shots:

# 