**Event Registration System**

**(Assignment # ……………………02)**

**Submitted To:**

Dr. Muhammad Usman Ali

**Submitted By:**

Ali Hassan

**(Roll #......)**

22021519-013

**Semester:**

2nd (BS\_CS\_23)(A)

**Course Title:**

Object Oriented Programming

**Course Code:**

CS-103

**Department of Computer Science**

A picture containing text, logo, graphics, graphic design

Description automatically generated

|  |  |  |
| --- | --- | --- |
| No. | Content | PgNo. |
| 1 | Description | 3 |
| 2 | Selection Table | 5 |
| 3 | Class Diagram | 6 |
| 4 | Code with functionalities |  |
|  |  |  |
|  |  |  |

**Topic:**

**Event Registration System**

Project Description

What is an Event Registration System?

Event Registration System is a system that will facilitate user to conduct different events held in our surrounding. As marriage event, Seminars, Workshops, and some other related gatherings. So, we require such a software system that will help us to manage these events. In this system people can book and manage their events very quickly. It will allow users to set the event type, Attendee list and schedule. And they will be provided with the choice to update the data from time to time. This system will have the ability to manage and store a large amount of data.

Problems Faced by Manual System:

Basically, manual system contains paperwork which.

pose many limitations, such as:

**1.**Time Consuming

**2.**Errors and inaccuracies

**3.**Communication challenges

**4.**Limited data

**5.**Difficult to update.

So, by noticing all these challenges it is the to develop such a system which have ability to provide user an efficient way for managing and controlling their events.

**Features:**

This system contains a user-friendly menu that will entertain user by giving such options as:

**1.**Event Management

**2.**Attendee Management

**3.**Event Organizer Management

**4.**Ticket Management

**5.**Schedule Management

The payment process will be added to the Ticket Management system.

**Event Management:** This system will allow the user to set the event as of him choice. And anything about the event.

**Attendee Management:** In this section the user will be able to add, view, search, update information of Attendee and removing the attendee. This section of the system also provides an option for special requests to get entry in the event.

**Event Organizer**: This section of the system allows user to set the data of the event organizer his phone contact, and all the things by which the attendee of the event can approach him.

**Ticket System:** Basically, at this stage it should be remembered that some events do not have the ticket system, so they have an invitation to call attendees. So, if such an event having a ticket system will be performed then this class will be used otherwise this class will also be used as communication for contacting people and for inviting people.

**Schedule Management:** In this class of the system the user can set and control the event holding time. As starting time, duration of the event, ceremony time, venue time, and the event ending time and user can update time information. As by requirement.

**How this system will be helpful…?**

The Event Registration System is helpful in the following ways:

Streamlines event management.

**1.**Simplifies attendee registration.

**2.**Enhances communication between organizers and attendees.

**3.**Provides efficient ticket management and payment processing.

**4.**Facilitates schedule management.

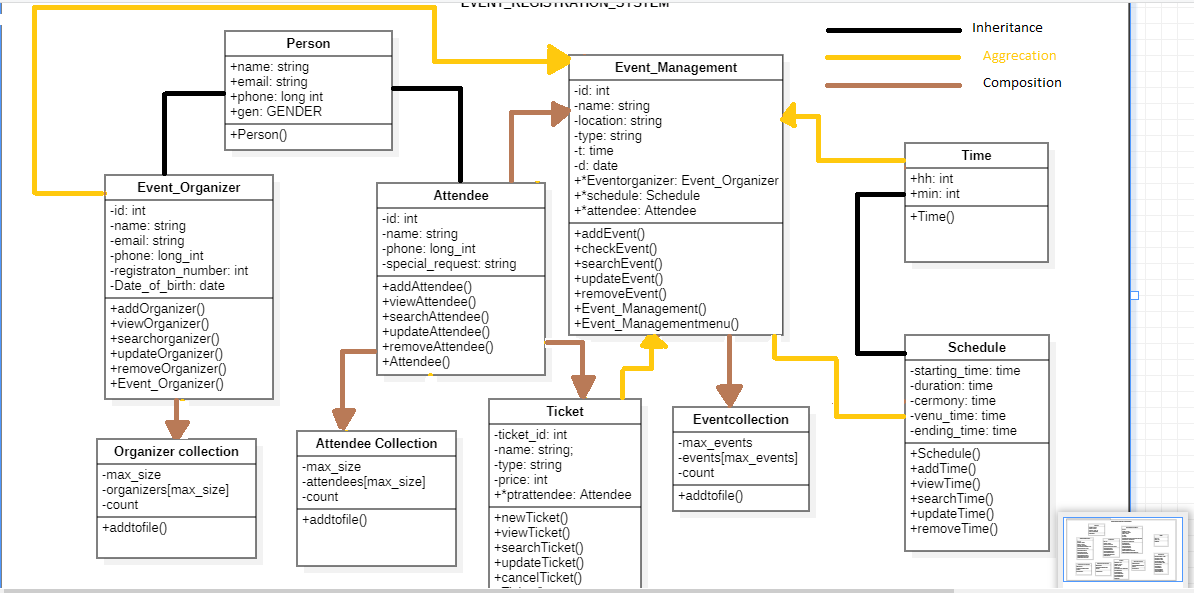
**5.**Enables data storage and analysis for insights.

**6.**Improves overall event organization and attendee experience.

**Selection Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Class Name** | **Selected** | **Reason** |
|  |  |  |  |
| **1** | Event Management | Yes | It is compulsory to add the details of the event. |
|  |  |  |  |
| **2** | Event Organizer | Yes | Because it is important to know about the |
|  |  |  | organizer. And people can approach him easily. |
|  |  |  |  |
| **3** | Attendee | Yes | Because the event is not an event without the |
|  |  |  | attendees all the data about the attendees. |
|  |  |  |  |
| **4** | Ticket | Optional | Because there are some events which do not |
|  |  |  | involve a ticket system. As marriage event. |
|  |  |  |  |
| **5** | Schedule | Yes | Because an event cannot be conduct without a |
|  |  |  | proper schedule. |
|  |  |  |  |
| **6** | Time | Yes | All the time management of the event controlled |
|  |  |  | by this class. |
|  |  |  |  |
| **7** | Date | Yes | Because the date of event occurring is important |
|  |  |  | in this system |
|  |  |  |  |

**CLASS DIAGRAM**

****

**CODE**

**#include <iostream>**

**#include <string>**

**#include <windows.h>**

**#include<cstdlib>**

**#include<fstream>**

**#include <iostream>**

**#include <windows.h>**

**#include<conio.h>**

**using namespace std;**

**#define maxsize 100**

**void showData()**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**CONSOLE\_FONT\_INFOEX fontInfo;**

**fontInfo.cbSize = sizeof(CONSOLE\_FONT\_INFOEX);**

**GetCurrentConsoleFontEx(hConsole, FALSE, &fontInfo);**

**fontInfo.dwFontSize.X = 24; // Set the X font size to 16**

**fontInfo.dwFontSize.Y = 24; // Set the Y font size to 24**

**SetCurrentConsoleFontEx(hConsole, FALSE, &fontInfo);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED);**

**char arrShow[] = "EVENT REGISTRATION SYSTEM";**

**cout << "\n\t\t\t\t\t\t";**

**for (int i = 0; i < 24; i++)**

**{**

**std::cout << arrShow[i];**

**Sleep(300);**

**}**

**system("CLS");**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**}**

**enum GENDER {**

**male, female //0 , 1**

**};**

**class person**

**{**

**public:**

**string name;**

**string email;**

**long int phone;**

**GENDER gen;**

**public:**

**person()**

**{**

**name = '\0';**

**email = '\0';**

**phone = 0;**

**}**

**};**

**class Date**

**{**

**public:**

**int dd;**

**int mm;**

**int yyyy;**

**Date()**

**{**

**dd = 0;**

**mm = 0;**

**yyyy = 0;**

**}**

**~Date()**

**{**

**}**

**};**

**class Time**

**{**

**public:**

**int hh;**

**int min;**

**Time()**

**{**

**hh = 0;**

**min = 0;**

**}**

**~Time()**

**{**

**}**

**};**

**class Event\_Management**

**{**

**public:**

**static int idcount;**

**int id;**

**string name;**

**Time t;**

**Date d;**

**string location;**

**string type;**

**string description;**

**public:**

**Event\_Management()**

**{**

**id = ++idcount;**

**name = '\0';**

**location = '\0';**

**type = '\0';**

**description = '\0';**

**}**

**~Event\_Management() {**

**}**

**static void event\_managementmenu();**

**void addEvent();**

**void checkEvent();**

**void searchEvent();**

**void removeEvent();**

**void updateEvent();**

**};**

**int Event\_Management::idcount = 0;**

**class Event\_Organizer : public person**

**{**

**public:**

**static int idcount;**

**int id;**

**int registration\_number;**

**Date date\_of\_birth;**

**public:**

**Event\_Organizer()**

**{**

**id = ++idcount;**

**registration\_number = 0;**

**}**

**~Event\_Organizer() {**

**}**

**static void eventorganizermenu();**

**void addOrganizer();**

**void viewOrganizer();**

**static void searchOrganizer();**

**static void updateOrganizer();**

**static void removeOrganizer();**

**};**

**int Event\_Organizer::idcount = 0;**

**//-----------------**

**class Attendee :public person**

**{**

**public:**

**static int idcount;**

**int id;**

**char special\_request;**

**public:**

**Attendee()**

**{**

**id = ++idcount;**

**special\_request = '\0';**

**}**

**~Attendee() {**

**}**

**static void attendeemenu();**

**void addAttendee();**

**void viewAttendee();**

**void searchAttendee();**

**void removeAttendee();**

**void updateAttendee();**

**};**

**int Attendee::idcount = 0;**

**Attendee\* fetchAttendeeByID(int id);**

**class Ticket**

**{**

**public:**

**int ticket\_id;**

**string name;**

**string type;**

**int price;**

**Attendee\* ptrattendee;**

**public:**

**Ticket()**

**{**

**ticket\_id = 0;**

**price = 0;**

**name = '\0';**

**type = '\0';**

**ptrattendee = NULL;**

**}**

**~Ticket() {**

**}**

**int getID() const {**

**return ticket\_id;**

**}**

**static void ticketmenu();**

**void newTicket();**

**void viewTicket();**

**void searchTicket();**

**void updateTicket();**

**void cancelTicket();**

**};**

**class Schedule : public Time**

**{**

**private:**

**int starting\_time;**

**int duration;**

**int ceremony;**

**int venue\_time;**

**int ending\_time;**

**public:**

**Schedule()**

**{**

**starting\_time = 0;**

**duration = 0;**

**ceremony = 0;**

**ending\_time = 0;**

**}**

**~Schedule() {**

**}**

**static void schedule();**

**void addTime();**

**void viewTime();**

**void updateTime();**

**void searchTime();**

**void removeTime();**

**};**

**class Utility**

**{**

**public:**

**static int GetNumberFromUser(int len)**

**{**

**char strNum[11];**

**if (len > 10) len = 10;**

**char ch;**

**int i = 0;**

**while (i < len)**

**{**

**ch = \_getch();**

**if (ch >= '0' && ch <= '9')**

**{**

**cout << ch;**

**strNum[i] = ch;**

**i++;**

**}**

**if (ch == '\r')**

**{**

**break;**

**}**

**if (ch == '\b')**

**{**

**cout << "\b \b";**

**i--;**

**}**

**}**

**cout << "\n";**

**strNum[i] = '\0';**

**return atoi(strNum);**

**}**

**static string GetStringFromUser(int len)**

**{**

**char strNum[256];**

**if (len > 10) len = 10;**

**char ch;**

**int i = 0;**

**while (i < len)**

**{**

**ch = \_getch();**

**if (ch >= 'a' && ch <= 'z')**

**{**

**cout << ch;**

**strNum[i] = ch;**

**i++;**

**}**

**if (ch == '\r')**

**{**

**break;**

**}**

**if (ch == '\b')**

**{**

**cout << "\b \b";**

**i--;**

**}**

**}**

**cout << "\n";**

**strNum[i] = '\0';**

**string str = strNum;**

**return str;**

**}**

**};**

**class login {**

**public:**

**string adminusername;**

**long int adminpassword;**

**public:**

**void inputdataadmin()**

**{**

**cout << "\n\n\nEnter the new user name : " << endl;**

**adminusername = Utility::GetStringFromUser(15);**

**cout << "\n\n\nEnter the new password : " << endl;**

**cin >> adminpassword;**

**ofstream write;**

**write.open("E:\\login1.txt");**

**if (write.is\_open())**

**{**

**write << "Admin username : " << adminusername << endl;**

**write << "Admin password: " << adminpassword << endl;**

**write.close();**

**cout << "=====================================================" << endl;**

**cout << "Your account has been creating please wait";**

**for (int i = 0; i < 6; i++)**

**{**

**cout << ".";**

**Sleep(500);**

**}**

**}**

**else**

**{**

**cout << "Error in opening file..." << endl;**

**}**

**}**

**/\*Login\*/**

**bool loginadmin(void)**

**{**

**string au, ap;**

**cout << "\n\n\nEnter the admin username : " << endl;**

**au = Utility::GetStringFromUser(15);**

**cout << "\n\n\nEnter the admin password : " << endl;**

**cin >> ap;**

**ifstream read;**

**read.open("E:\\login1.txt");**

**if (read.is\_open())**

**{**

**string line;**

**while (getline(read, line))**

**{**

**if (line.find("Admin username : ") != string::npos)**

**{**

**string storeuser = line.substr(line.find(": ") + 2);**

**getline(read, line);**

**string storepass = line.substr(line.find(": ") + 2);**

**if (au == storeuser && ap == storepass)**

**{**

**cout << "Admin login successfully..." << endl;**

**read.close();**

**return true;**

**}**

**}**

**}**

**read.close();**

**}**

**else**

**{**

**cout << "Error in opening...." << endl;**

**}**

**cout << "Invalid username or password..." << endl;**

**return false;**

**}**

**static int menu(int op)**

**{**

**cout << "\t\t\tLogin Page...!!!\n\n" << endl;**

**cout << "\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**

**cout << "\t\t1. For Registration..." << endl;**

**cout << "\t\t2. For Login...." << endl;**

**cout << "\t\t3. Exit Program..." << endl;**

**cout << "\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**

**cout << "\t\t Enter the choice : ";**

**op = Utility::GetNumberFromUser(1);**

**return op;**

**}**

**};**

**// event collection class**

**class EventCollection {**

**private:**

**static const int MAX\_EVENTS = 100;**

**Event\_Management events[MAX\_EVENTS];**

**int count;**

**public:**

**EventCollection() {**

**count = 0;**

**}**

**void addEvent(const Event\_Management& event) {**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**if (count < MAX\_EVENTS) {**

**events[count++] = event;**

**ofstream outputFile("E:\\events.txt", ios::app);**

**if (outputFile.is\_open()) {**

**for (int i = 0; i < count; i++) {**

**outputFile << "Event ID: " << events[i].id << endl;**

**outputFile << "Name: " << events[i].name << endl;**

**outputFile << "Date: " << events[i].d.dd << "/" << events[i].d.mm << "/" << events[i].d.yyyy << endl;**

**outputFile << "Time: " << events[i].t.hh << ":" << events[i].t.min << endl;**

**outputFile << "Location: " << events[i].location << endl;**

**outputFile << "Type: " << events[i].type << endl;**

**outputFile << "Description: " << events[i].description << endl;**

**outputFile << "-----------------------------------------" << endl;**

**}**

**outputFile.close();**

**cout << "Events saved to file successfully!" << endl;**

**}**

**else {**

**cout << "Error in opening the file." << endl;**

**}**

**cout << "Event added successfully!" << endl;**

**}**

**else {**

**cout << "Event collection is full. Cannot add more events." << endl;**

**}**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**}**

**void saveEventsToFile() {**

**}**

**};**

**void Event\_Management::addEvent() {**

**EventCollection eventCollection;**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << "\t\t-----------\* Enter event details \*------------" << endl;**

**cout << endl << endl;**

**// Prompt the user to enter event details**

**cout << "Event Name: ";**

**cin.ignore();**

**getline(cin, name);**

**cout << "Event Date (dd mm yyyy): ";**

**cin >> d.dd >> d.mm >> d.yyyy;**

**cout << "Event Time (hh mm): ";**

**cin >> t.hh >> t.min;**

**cout << "Event Location: ";**

**cin.ignore();**

**getline(cin, location);**

**cout << "Event Type: ";**

**getline(cin, type);**

**cout << "Event Description: ";**

**getline(cin, description);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**Event\_Management event;**

**event.id = ++id;**

**event.name = name;**

**event.d = d;**

**event.t = t;**

**event.location = location;**

**event.type = type;**

**event.description = description;**

**eventCollection.addEvent(event);**

**}**

**void Event\_Management::checkEvent()**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**ifstream inputFile("E:\\events.txt");**

**if (!inputFile.is\_open()) {**

**cout << "Error in opening the file." << endl;**

**return;**

**}**

**cout << "\t\t-------\* Event List \*----------" << endl;**

**cout << endl << endl;**

**string line;**

**while (getline(inputFile, line)) {**

**if (line.find("Event ID: ") != string::npos) {**

**cout << line << "| ";**

**getline(inputFile, line); // Read the next line**

**cout << line << "| ";**

**getline(inputFile, line);**

**cout << line << "| ";**

**getline(inputFile, line);**

**cout << line << "| ";**

**getline(inputFile, line);**

**cout << line << "| ";**

**getline(inputFile, line);**

**cout << line << "| ";**

**getline(inputFile, line);**

**cout << line << "| ";**

**cout << endl << endl;**

**}**

**}**

**inputFile.close();**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**}**

**void Event\_Management::searchEvent()**

**{**

**int id;**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << "\t\t------\* Search \*-------" << endl;**

**cout << endl << endl;**

**cout << "Enter the ID of the Event...";**

**cin >> id;**

**ifstream read;**

**read.open("E:\\events.txt");**

**if (read.is\_open())**

**{**

**string data;**

**while (getline(read, data))**

**{**

**if (data.find("Event ID: " + id) != string::npos)**

**{**

**cout << "\t\t-----\* Event Details \*-------" << endl;**

**cout << endl << endl;**

**for (int i = 0; i < 6; i++)**

**{**

**// data is in 6 lines**

**getline(read, data);**

**cout << data << "| ";**

**}**

**cout << "\n---------------------------" << endl;**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**read.close();**

**return;**

**}**

**}**

**read.close();**

**cout << "Event with id " << id << " not found...!!!" << endl;**

**}**

**}**

**void Event\_Management::updateEvent()**

**{**

**//implementation**

**}**

**void Event\_Management::removeEvent()**

**{**

**//implementation**

**}**

**// Organizer collection class**

**class organizercollection**

**{**

**private:**

**static const int max\_org = 100;**

**int count;**

**public:**

**Event\_Organizer orgs[max\_org];**

**organizercollection()**

**{**

**count = 0;**

**}**

**void addorganizer(Event\_Organizer& org)**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**if (count < max\_org)**

**{**

**orgs[count++] = org;**

**ofstream write;**

**write.open("E:\\orgs.txt", ios::app);**

**if (write.is\_open())**

**{**

**for (int i = 0; i < count; i++)**

**{**

**write << "ID: " << orgs[i].id << endl;**

**write << "Name: " << orgs[i].name << endl;**

**write << "Phone: " << orgs[i].phone << endl;**

**write << "Email: " << orgs[i].email << endl;**

**write << "Registration\_number: " << orgs[i].registration\_number << endl;**

**write << "DOB: " << orgs[i].date\_of\_birth.dd << " / " << orgs[i].date\_of\_birth.mm << " / " << orgs[i].date\_of\_birth.yyyy << endl;**

**write << "GENDER" << orgs[i].gen << endl;**

**}**

**write.close();**

**cout << "Data saved successfully..." << endl;**

**}**

**else**

**{**

**cout << "Error in opening the file..." << endl;**

**}**

**cout << "Data saved..." << endl;**

**}**

**else**

**{**

**cout << "There is no space to accomodate..." << endl;**

**}**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**}**

**};**

**void Event\_Organizer::addOrganizer()**

**{**

**organizercollection organizer;**

**pass:**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << "\t\t-------\* Enter Organizer Details \*---------\n" << endl;**

**cout << endl << endl;**

**cout << "Enter the Name of Organizer : ";**

**cin >> name;**

**cout << "Enter the Phone Number : ";**

**cin >> phone;**

**if (phone < 0)**

**{**

**cout << "Wrong Input" << endl;**

**goto pass;**

**}**

**cout << "Enter the Email..";**

**cin >> email;**

**cout << "Enter the Registration number : ";**

**cin >> registration\_number;**

**if (registration\_number < 0)**

**{**

**cout << "Wrong input" << endl;**

**goto pass;**

**}**

**cout << "Enter the date of birth : " << endl;**

**cin >> date\_of\_birth.dd >> date\_of\_birth.mm >> date\_of\_birth.yyyy;**

**if (date\_of\_birth.dd < 0 || date\_of\_birth.mm < 0 || date\_of\_birth.yyyy < 0)**

**{**

**cout << "Wrong Input" << endl;**

**goto pass;**

**}**

**cout << "Enter the Gender (M/F): " << endl;**

**char g;**

**cin >> g;**

**if (!g == 'M' || !g == 'F')**

**{**

**cout << "Wrong input" << endl;**

**goto pass;**

**}**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**Event\_Organizer eor;**

**eor.id = ++id;**

**eor.name = name;**

**eor.phone = phone;**

**eor.email = email;**

**eor.registration\_number = registration\_number;**

**eor.date\_of\_birth = date\_of\_birth;**

**organizer.addorganizer(eor);**

**}**

**void Event\_Organizer::viewOrganizer()**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**ifstream read;**

**read.open("E:\\orgs.txt");**

**if (!read.is\_open())**

**{**

**cout << "Error in opening the file..." << endl;**

**return;**

**}**

**else**

**{**

**cout << "\t\t------\* Organizer List \*-------" << endl;**

**cout << endl << endl;**

**string line;**

**while (getline(read, line))**

**{**

**if (line.find("ID: ") != string::npos)**

**{**

**cout << line << "| "; //1**

**getline(read, line);**

**cout << line << "| "; // 2**

**getline(read, line);**

**cout << line << "| ";//3**

**getline(read, line);**

**cout << line << "| ";//4**

**getline(read, line);**

**cout << line << "| "; // 5**

**getline(read, line);**

**cout << line << "| ";//6**

**getline(read, line);**

**cout << endl;**

**}**

**}**

**read.close();**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**}**

**}**

**void Event\_Organizer::searchOrganizer()**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**int id;**

**cout << "\t\t------\* Search \*-------" << endl;**

**cout << endl << endl;**

**cout << "Enter the ID: " << endl;**

**cin >> id;**

**ifstream read;**

**read.open("E:\\orgs.txt");**

**if (!read.is\_open())**

**{**

**cout << "Error in opening file..." << endl;**

**return;**

**}**

**else**

**{**

**string data;**

**while (getline(read, data))**

**{**

**if (data.find("ID: " + id) != string::npos)**

**{**

**cout << "\t\t------\* Organizer Details \*------" << endl;**

**cout << endl << endl;**

**for (int i = 0; i < 6; i++)**

**{**

**getline(read, data);**

**cout << data << "| ";**

**}**

**}**

**}**

**}**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**}**

**void Event\_Organizer::updateOrganizer()**

**{**

**//implementation**

**}**

**void Event\_Organizer::removeOrganizer()**

**{**

**//implementation**

**}**

**// Attendee collection class**

**class attendeecollection {**

**private:**

**static const int max\_size = 100;**

**Attendee attendee[max\_size];**

**int count;**

**public:**

**attendeecollection()**

**{**

**count = 0;**

**}**

**void addattendee(Attendee& att)**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**if (count < max\_size)**

**{**

**attendee[count++] = att;**

**ofstream write;**

**write.open("E:\\attendee.txt", ios::app);**

**if (!write.is\_open())**

**{**

**cout << "Error in openning...." << endl;**

**return;**

**}**

**else**

**{**

**for (int i = 0; i < count; i++)**

**{**

**write << "ID: " << attendee[i].id << endl;**

**write << "Name: " << attendee[i].name << endl;**

**write << "Phone: " << attendee[i].phone << endl;**

**write << "Email: " << attendee[i].email << endl;**

**write << "Special Request: " << attendee[i].special\_request << endl;**

**}**

**cout << "Data saved in file successfully..." << endl;**

**write.close();**

**}**

**cout << "Data Saved..." << endl;**

**}**

**else**

**{**

**cout << "Cannot Accomadate.." << endl;**

**}**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**}**

**};**

**void Attendee::addAttendee()**

**{**

**attendeecollection attendeecol;**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << "\t\t------\* Attendee Details \*------" << endl;**

**cout << endl << endl;**

**cout << "------------------\n" << endl;**

**cout << "Enter the Name : ";**

**cin >> name;**

**cout << "Enter the Phone : ";**

**cin >> phone;**

**cout << "Enter the email : ";**

**cin >> email;**

**cout << "Special request(y/n) :";**

**cin >> special\_request;**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**Attendee att;**

**att.id = ++id;**

**att.name = name;**

**att.phone = phone;**

**att.email = email;**

**att.special\_request = special\_request;**

**attendeecol.addattendee(att);**

**}**

**void Attendee::viewAttendee()**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**ifstream read;**

**read.open("E:\\attendee.txt");**

**if (!read.is\_open())**

**{**

**cout << "Error..." << endl;**

**return;**

**}**

**else**

**{**

**string line;**

**while (getline(read, line))**

**{**

**if (line.find("ID: " + id) != string::npos)**

**{**

**cout << "\t\t--------\* Attendee list \*-------" << endl;**

**cout << endl << endl;**

**cout << line << endl; //1**

**getline(read, line);**

**cout << line << endl; // 2**

**getline(read, line);**

**cout << line << endl;//3**

**getline(read, line);**

**cout << line << endl;//4**

**getline(read, line);**

**cout << line << endl; // 5**

**getline(read, line);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**}**

**}**

**read.close();**

**}**

**}**

**void Attendee::searchAttendee()**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << "\t\t-----\* Search \*-------" << endl;**

**cout << endl << endl;**

**int id;**

**cout << "Enter the ID : ";**

**cin >> id;**

**ifstream read;**

**read.open("E:\\attendee.txt");**

**if (!read.is\_open())**

**{**

**cout << "Error.." << endl;**

**return;**

**}**

**else**

**{**

**string data;**

**while (getline(read, data))**

**{**

**if (data.find("ID: " + id) != string::npos)**

**{**

**cout << "Attendee Details..." << endl;**

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

**{**

**getline(read, data);**

**cout << data << endl;**

**}**

**}**

**}**

**read.close();**

**}**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**}**

**void Attendee::updateAttendee()**

**{**

**//implementation**

**}**

**void Attendee::removeAttendee()**

**{**

**//implementation**

**}**

**void Ticket::newTicket() {**

**}**

**void Ticket::viewTicket()**

**{**

**}**

**void Ticket::searchTicket()**

**{**

**//implementation**

**}**

**void Ticket::updateTicket()**

**{**

**//implementation**

**}**

**void Ticket::cancelTicket()**

**{**

**//implementation**

**}**

**void Schedule::addTime()**

**{**

**//implementation**

**}**

**void Schedule::viewTime()**

**{**

**//implementation**

**}**

**void Schedule::searchTime()**

**{**

**//implementation**

**}**

**void Schedule::updateTime()**

**{**

**//implementation**

**}**

**void Schedule::removeTime()**

**{**

**//implementation**

**}**

**// Sub Menus in static functions**

**void Event\_Management::event\_managementmenu() {**

**int eventChoice;**

**Event\_Management eventObj;**

**do**

**{**

**system("CLS");**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << "\n===================================" << endl;**

**cout << "\tEVENT MANAGEMENT MENU";**

**cout << "\n===================================" << endl;**

**cout << "1. Add Event" << endl;**

**cout << "2. Check Event" << endl;**

**cout << "3. Search Event" << endl;**

**cout << "4. Remove Event" << endl;**

**cout << "5. Update Event" << endl;**

**cout << "0. Go Back" << endl;**

**cout << "\n====================" << endl;**

**cout << "Enter your choice: ";**

**eventChoice = Utility::GetNumberFromUser(1);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**switch (eventChoice)**

**{**

**case 1:**

**system("cls");**

**eventObj.addEvent();**

**system("pause");**

**system("cls");**

**break;**

**case 2:**

**system("cls");**

**eventObj.checkEvent();**

**system("pause");**

**system("cls");**

**break;**

**case 3:**

**system("cls");**

**eventObj.searchEvent();**

**system("pause");**

**system("cls");**

**break;**

**case 4:**

**eventObj.removeEvent();**

**system("pause");**

**system("cls");**

**break;**

**case 5:**

**eventObj.updateEvent();**

**system("pause");**

**system("cls");**

**break;**

**case 0:**

**break;**

**default:**

**cout << "Invalid choice!" << endl;**

**system("pause");**

**system("cls");**

**break;**

**}**

**} while (eventChoice != 0);**

**}**

**void Event\_Organizer::eventorganizermenu() {**

**int organizerChoice;**

**Event\_Organizer organizerObj;**

**do**

**{**

**system("CLS");**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << "\n===================================" << endl;**

**cout << "\tEVENT ORGANIZER MENU";**

**cout << "\n===================================" << endl;**

**cout << "1. Add Organizer" << endl;**

**cout << "2. View Organizer" << endl;**

**cout << "3. Search Organizer" << endl;**

**cout << "4. Update Organizer" << endl;**

**cout << "5. Remove Organizer" << endl;**

**cout << "0. Go Back" << endl;**

**cout << "\n====================" << endl;**

**cout << "Enter your choice: ";**

**organizerChoice = Utility::GetNumberFromUser(1);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**switch (organizerChoice)**

**{**

**case 1:**

**system("cls");**

**organizerObj.addOrganizer();**

**system("pause");**

**system("cls");**

**break;**

**case 2:**

**system("cls");**

**organizerObj.viewOrganizer();**

**system("pause");**

**system("cls");**

**break;**

**case 3:**

**system("cls");**

**organizerObj.searchOrganizer();**

**system("pause");**

**system("cls");**

**break;**

**case 4:**

**organizerObj.updateOrganizer();**

**system("pause");**

**system("cls");**

**break;**

**case 5:**

**organizerObj.removeOrganizer();**

**system("pause");**

**system("cls");**

**break;**

**case 0:**

**break;**

**default:**

**cout << "Invalid choice!" << endl;**

**system("pause");**

**system("cls");**

**break;**

**}**

**} while (organizerChoice != 0);**

**}**

**void Attendee::attendeemenu() {**

**system("CLS");**

**int attendeeChoice;**

**Attendee attendeeObj;**

**do**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << "\n===================================" << endl;**

**cout << "\tATTENDEE MENU";**

**cout << "\n===================================" << endl;**

**cout << "1. Add Attendee" << endl;**

**cout << "2. View Attendee" << endl;**

**cout << "3. Search Attendee" << endl;**

**cout << "4. Remove Attendee" << endl;**

**cout << "5. Update Attendee" << endl;**

**cout << "0. Go Back" << endl;**

**cout << "\n====================" << endl;**

**cout << "Enter your choice: ";**

**attendeeChoice = Utility::GetNumberFromUser(1);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**switch (attendeeChoice)**

**{**

**case 1:**

**attendeeObj.addAttendee();**

**system("pause");**

**system("cls");**

**break;**

**case 2:**

**attendeeObj.viewAttendee();**

**system("pause");**

**system("cls");**

**break;**

**case 3:**

**attendeeObj.searchAttendee();**

**system("pause");**

**system("cls");**

**break;**

**case 4:**

**attendeeObj.removeAttendee();**

**system("pause");**

**system("cls");**

**break;**

**case 5:**

**attendeeObj.updateAttendee();**

**system("pause");**

**system("cls");**

**break;**

**case 0:**

**break;**

**default:**

**cout << "Invalid choice!" << endl;**

**system("pause");**

**system("cls");**

**break;**

**}**

**} while (attendeeChoice != 0);**

**}**

**void Ticket::ticketmenu() {**

**system("CLS");**

**int ticketChoice;**

**Ticket ticketObj;**

**do**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << "\n===================================" << endl;**

**cout << "\tTICKET MENU";**

**cout << "\n===================================" << endl;**

**cout << "1. New Ticket" << endl;**

**cout << "2. View Ticket" << endl;**

**cout << "3. Search Ticket" << endl;**

**cout << "4. Update Ticket" << endl;**

**cout << "5. Cancel Ticket" << endl;**

**cout << "0. Go Back" << endl;**

**cout << "\n====================" << endl;**

**cout << "Enter your choice: ";**

**ticketChoice = Utility::GetNumberFromUser(1);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**switch (ticketChoice)**

**{**

**case 1:**

**ticketObj.newTicket();**

**system("pause");**

**system("cls");**

**break;**

**case 2:**

**ticketObj.viewTicket();**

**system("pause");**

**system("cls");**

**break;**

**case 3:**

**ticketObj.searchTicket();**

**system("pause");**

**system("cls");**

**break;**

**case 4:**

**ticketObj.updateTicket();**

**system("pause");**

**system("cls");**

**break;**

**case 5:**

**ticketObj.cancelTicket();**

**system("pause");**

**system("cls");**

**break;**

**case 0:**

**break;**

**default:**

**cout << "Invalid choice!" << endl;**

**system("pause");**

**system("cls");**

**break;**

**}**

**} while (ticketChoice != 0);**

**}**

**void Schedule::schedule() {**

**system("CLS");**

**int scheduleChoice;**

**Schedule scheduleObj;**

**do**

**{**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << "\n===================================" << endl;**

**cout << "\tSCHEDULE MENU";**

**cout << "\n===================================" << endl;**

**cout << "1. Add Time" << endl;**

**cout << "2. View Time" << endl;**

**cout << "3. Update Time" << endl;**

**cout << "4. Search Time" << endl;**

**cout << "5. Remove Time" << endl;**

**cout << "0. Go Back" << endl;**

**cout << "\n====================" << endl;**

**cout << "Enter your choice: ";**

**scheduleChoice = Utility::GetNumberFromUser(1);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**switch (scheduleChoice)**

**{**

**case 1:**

**scheduleObj.addTime();**

**system("pause");**

**system("cls");**

**break;**

**case 2:**

**scheduleObj.viewTime();**

**system("pause");**

**system("cls");**

**break;**

**case 3:**

**scheduleObj.updateTime();**

**system("pause");**

**system("cls");**

**break;**

**case 4:**

**scheduleObj.searchTime();**

**system("pause");**

**system("cls");**

**break;**

**case 5:**

**scheduleObj.removeTime();**

**system("pause");**

**system("cls");**

**break;**

**case 0:**

**break;**

**default:**

**cout << "Invalid choice!" << endl;**

**system("pause");**

**system("cls");**

**break;**

**}**

**} while (scheduleChoice != 0);**

**}**

**int main()**

**{**

**//**

**int opp = 0;**

**login l;**

**do {**

**system("cls");**

**opp = login::menu(opp);**

**switch (opp)**

**{**

**case 1:**

**system("cls");**

**l.inputdataadmin();**

**break;**

**case 2:**

**{**

**system("cls");**

**if (l.loginadmin())**

**{**

**system("CLS");**

**showData();**

**Event\_Management e;**

**Event\_Organizer E;**

**Attendee a;**

**Ticket t;**

**Schedule s;**

**int choice;**

**string title = "Event Registration by Ali Hassan";**

**string menu = "MAIN MENU";**

**cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**

**cout << "\* \*" << endl;**

**cout << "\* " << title << " \*" << endl;**

**cout << "\* \*" << endl;**

**cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;**

**cout << endl << endl;**

**system("pause");**

**system("CLS");**

**do**

**{**

**system("CLS");**

**HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN);**

**cout << endl;**

**cout << " ==========================================" << endl;**

**cout << "|\t\t" << menu << "\t\t |" << endl;**

**cout << " ==========================================" << endl;**

**cout << "| 1. Event Management" << endl;**

**cout << "| 2. Event Organizer" << endl;**

**cout << "| 3. Attendee" << endl;**

**cout << "| 4. Ticket" << endl;**

**cout << "| 5. Schedule" << endl;**

**cout << "| 0. Exit" << endl;**

**cout << "==========================================" << endl;**

**cout << "Enter your choice: ";**

**choice = Utility::GetNumberFromUser(1);**

**SetConsoleTextAttribute(hConsole, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);**

**switch (choice)**

**{**

**case 1:**

**{**

**Event\_Management::event\_managementmenu();**

**break;**

**}**

**case 2:**

**{**

**Event\_Organizer::eventorganizermenu();**

**break;**

**}**

**case 3:**

**{**

**Attendee::attendeemenu();**

**break;**

**}**

**case 4:**

**{**

**Ticket::ticketmenu();**

**break;**

**}**

**case 5:**

**{**

**Schedule::schedule();**

**break;**

**}**

**case 0:**

**break;**

**}**

**} while (choice != 0);**

**cout << "System is running..." << endl;**

**system("pause");**

**}**

**break;**

**}**

**case 3:**

**cout << "Exiting program please wait";**

**for (int i = 0; i < 6; i++)**

**{**

**cout << ".";**

**Sleep(500);**

**}**

**exit(0);**

**default:**

**cout << "Invalid Choice......" << endl;**

**}**

**} while (opp != 3);**

**return 0;**

**}**