POLICE FIR MANAGEMENT SYSTEM

BY Muhammad Wasif

SUBMITTED TO :
SIR NAIMAT ULLAH

Description

Description of each function

1. Admin Class:

Description: This class represents admin accounts within the Police FIR system. It encapsulates attributes for username and password.

Constructor: Initializes the admin object with provided username and password.

Attributes:

Username: A string storing the admin's username.

Password: A string storing the admin's password.

2. Criminalrecord Class:

Description: Represents individual criminal records within the Police FIR system. It stores details such as the criminal's name, age, and the crime they committed.

Constructor: Initializes the criminalrecord object with the provided name, age, and crime details.

Attributes:

Name: A string representing the name of the criminal.

Age: An integer representing the age of the criminal.

Crime: A string describing the details of the crime committed.

3. FIR Class:

Description: Represents FIR (First Information Report) records within the Police FIR system. It contains information such as the FIR number, crime details, suspect information, and status.

Constructor: Initializes the FIR object with the provided FIR number, crime details, suspect information, and status.

Attributes:

Firnumber: An integer representing the FIR number.

Crimedetails: A string describing the details of the crime reported in the FIR.

Suspectinformation: A string providing information about the suspect(s) involved.

Status: A string indicating the status of the FIR (e.g., Open, Closed, Under Investigation).

4. Policefirsystem Class:

Description: Serves as the core component managing the entire Police FIR system. It provides functionalities for admin account management, adding/deleting/viewing criminal and FIR records, and generating FIR reports.

Private Attributes:

Admins: A vector storing admin accounts.

Criminalrecords: A vector storing criminal records.

Firrecords: A vector storing FIR records.

Currentadmin: A pointer to store the currently logged-in admin.

Public Member Functions:

Createadminaccount(): Allows the creation of a new admin account.

Adminlogin(): Handles admin login authentication.

Addcriminalrecord(): Adds a new criminal record to the system.

Addfir(): Adds a new FIR record to the system.

Deleterecord(): Deletes either a criminal record or an FIR record based on user choice.

Displayallrecords(): Displays all existing criminal and FIR records.

Generatefirreport(): Generates a report of all FIR records.

Run(): Executes the main menu and handles user interaction.

5. Main Function (main()):

Description: Serves as the entry point of the program. It initializes a policefirsystem object and calls its run() method to start the program execution.

```
#include <iostream>
#include <vector>
#include <string>
using namespace std;
// Class to represent Admin accounts
class Admin {
public:
string username;
string password;
Admin(string uname, string pword) : username(uname), password(pword) {}
};
class CriminalRecord {
public:
string name;
int age;
string crime;
CriminalRecord(string n, int a, string c) : name(n), age(a), crime(c) {}
};
class FIR {
public:
int firNumber;
string crimeDetails;
string suspectInformation;
string status;
FIR(int number, string details, string suspectInfo, string stat) : firNumber(number),
crimeDetails(details), suspectInformation(suspectInfo), status(stat) {}
};
```

```
class PoliceFIRSystem {
private:
vector<Admin> admins;
vector<CriminalRecord> criminalRecords;
vector<FIR> firRecords;
Admin* currentAdmin; // To store the currently logged-in admin
public:
void createAdminAccount() {
string username, password;
cout << "Enter new admin username: ";
cin >> username;
cout << "Enter new admin password: ";
cin >> password;
admins.push_back(Admin(username, password));
cout << "Admin account created successfully!\n";
}
bool adminLogin() {
string username, password;
cout << "Enter admin username: ";
cin >> username;
cout << "Enter admin password: ";
cin >> password;
for (Admin& admin : admins) {
if (admin.username == username && admin.password == password) {
currentAdmin = &admin;
cout << "Login successful!\n";
return true;}}
```

```
cout << "Invalid username or password. Login failed.\n";
return false;
}
void addCriminalRecord() {
string name, crime;
int age;
cout << "Enter criminal's name: ";
cin.ignore(); // Clear the input buffer
getline(cin, name);
cout << "Enter criminal's age: ";
cin >> age;
cout << "Enter details of the crime: ";
cin.ignore(); // Clear the input buffer
getline(cin, crime);
criminalRecords.push_back(CriminalRecord(name, age, crime));
cout << "Criminal record added successfully!\n";
}
void addFIR() {
int firNumber;
string crimeDetails, suspectInformation, status;
cout << "Enter FIR number: ";
cin >> firNumber;
cout << "Enter crime details: ";
cin.ignore();
getline(cin, crimeDetails);
cout << "Enter suspect information: ";
```

```
getline(cin, suspectInformation);
cout << "Enter FIR status (Open/Closed/Under Investigation): ";
getline(cin, status);
firRecords.push_back(FIR(firNumber, crimeDetails, suspectInformation, status));
cout << "FIR added successfully!\n";
}
void deleteRecord() {
int choice;
cout << "1. Delete Criminal Record\n";
cout << "2. Delete FIR Record\n";
cout << "Enter choice: ";
cin >> choice;
switch (choice) {
case 1:
if (!criminalRecords.empty()) {
cout << "Enter the index of the criminal record to delete: ";
int index;
cin >> index;
if (index >= 0 && index < criminalRecords.size()) {
criminalRecords.erase(criminalRecords.begin() + index);
cout << "Criminal record deleted successfully!\n";
} else {
cout << "Invalid index.\n";
}
} else {
cout << "No criminal records available to delete.\n";
}
```

```
break;
case 2:
if (!firRecords.empty()) {
cout << "Enter the index of the FIR record to delete: ";
int index;
cin >> index;
if (index >= 0 && index < firRecords.size()) {
firRecords.erase(firRecords.begin() + index);
cout << "FIR record deleted successfully!\n";
} else {
cout << "Invalid index.\n";
}
} else {
cout << "No FIR records available to delete.\n";
}
break;
default:
cout << "Invalid choice.\n";
}
}
void displayAllRecords() {
cout << "=== Criminal Records ===\n";
for (const CriminalRecord& record : criminalRecords) {
cout << "Name: " << record.name << "\tAge: " << record.age << "\tCrime: " << record.crime <<
"\n";
}
cout << "\n=== FIR Records ===\n";
```

```
for (const FIR& fir : firRecords) {
cout << "FIR Number: " << fir.firNumber << "\tCrime Details: " << fir.crimeDetails << "\tSuspect Information: " << fir.suspectInformation << "\tStatus: " << fir.status << "\n";
}
}
void generateFIRReport() {
cout << "=== FIR Report ===" << endl;
for (const auto& fir : firRecords) {
cout << "FIR Number: " << fir.firNumber << endl;
cout << "Crime Details: " << fir.crimeDetails << endl;
cout << "Suspect Information: " << fir.suspectInformation << endl;
cout << "Status: " << fir.status << endl << endl;
}
}
void run() {
int choice;
do {
cout << "\nMenu:\n";
cout << "1: Create Admin Account\n";
cout << "2: Admin Login\n";
cout << "0: Exit\n";
cout << "Enter choice: ";
cin >> choice;
switch (choice) {
case 1:
createAdminAccount();
break;
```

```
case 2:
if (adminLogin()) {
// If login is successful, show the main menu
do {
cout << "\n=== Main Menu ===\n";
cout << "1. Criminal Record\n";
cout << "2. FIR\n";
cout << "3. Delete Record\n";
cout << "4. Display All Records\n";
cout << "5. FIR Module\n";
cout << "6. Reports Module\n";
cout << "0. Exit\n";
cout << "Enter choice: ";
cin >> choice;
switch (choice) {
case 1:
addCriminalRecord();
break;
case 2:
addFIR();
break;
case 3:
deleteRecord();
break;
case 4:
displayAllRecords();
break;
```

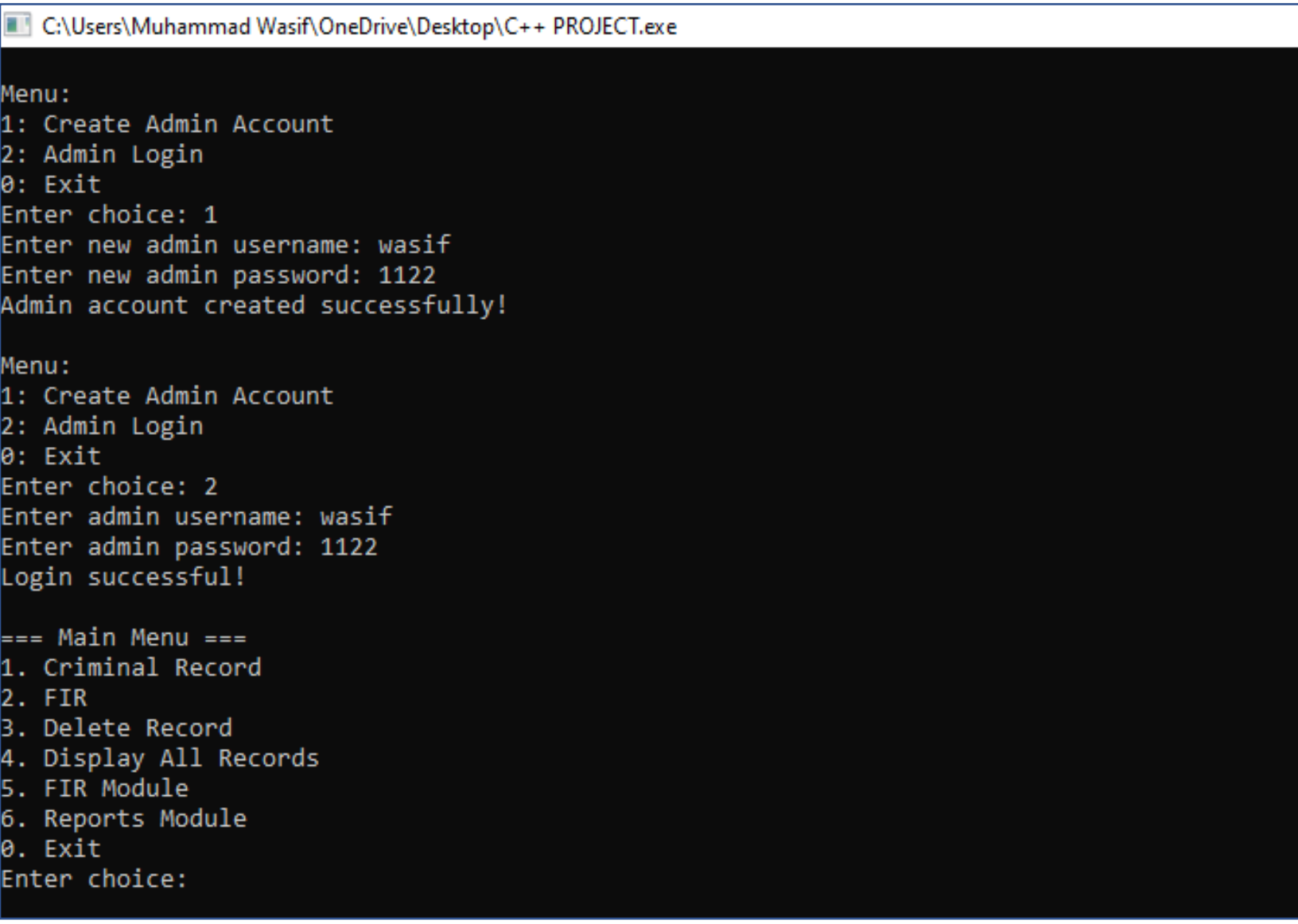
```
case 5:
// FIR Module
do {
cout << "== FIR Module ==" << endl;
cout << "1. Issue FIR" << endl;
cout << "2. Update FIR" << endl;
cout << "3. Change FIR Status" << endl;
cout << "4. Back to Admin Menu" << endl;
cout << "Enter your choice: ";
cin >> choice;
switch(choice) {
case 1:
addFIR();
break;
case 2:
// Update FIR
break;
case 3:
// Change FIR Status
break;
case 4:
// Go back to Admin Menu
break;
default:
cout << "Invalid choice. Please try again." << endl;
}
} while (choice != 4);
```

```
break;
case 6:
// Reports Module
do {
cout << "== Reports Module ==" << endl;
cout << "1. Generate FIR Report" << endl;
cout << "2. Generate Criminal Report" << endl;
cout << "3. Customize Report Parameters" << endl;
cout << "4. Export Reports" << endl;
cout << "5. Back to Admin Menu" << endl;
cout << "Enter your choice: ";
cin >> choice;
switch(choice) {
case 1:
generateFIRReport();
break;
case 2:
// Generate Criminal Report
break;
case 3:
// Customize Report Parameters
break;
case 4:
// Export Reports
break;
case 5:
// Go back to Admin Menu
```

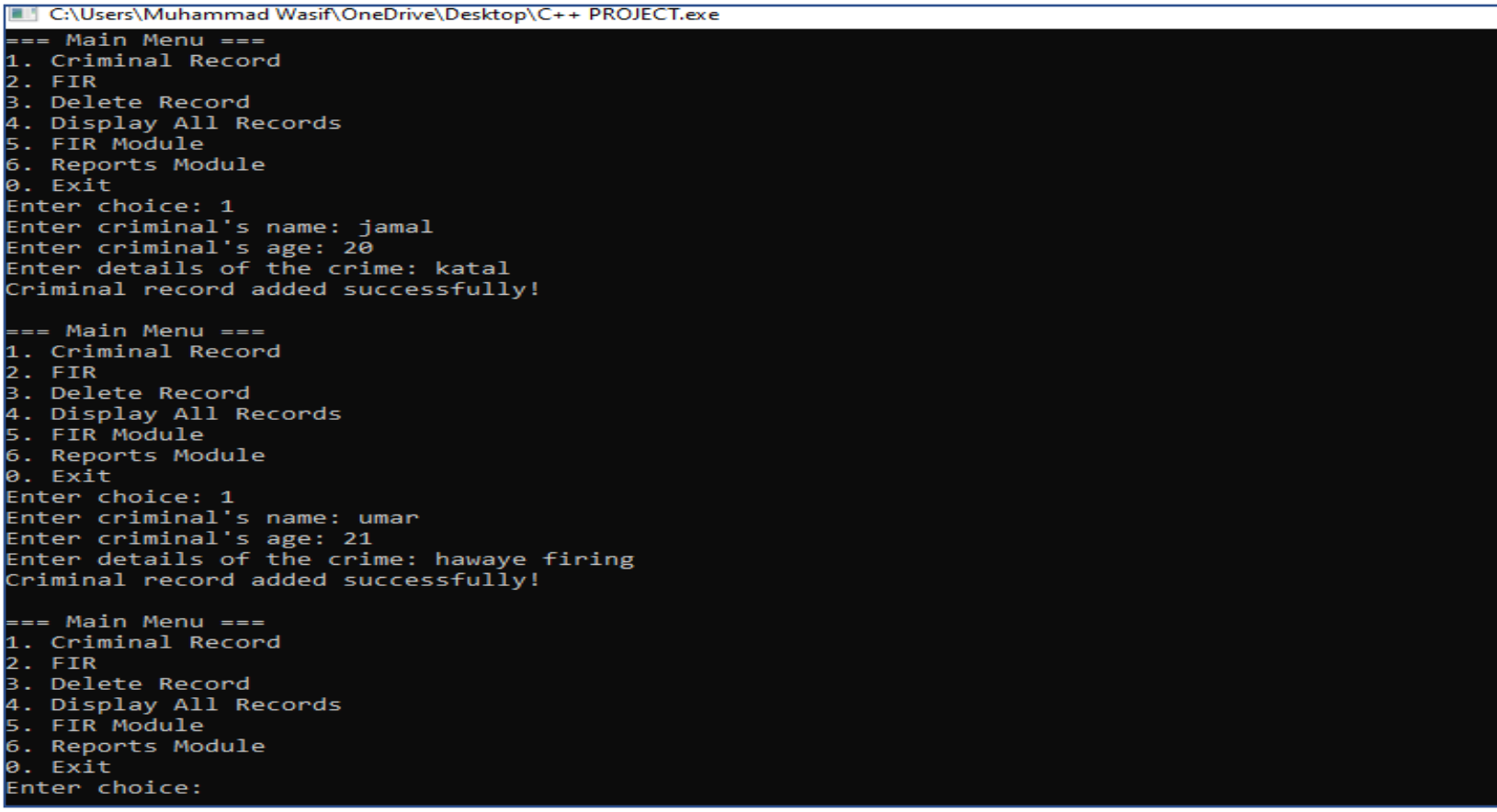
```
break;
default:
cout << "Invalid choice. Please try again." << endl;
}
} while (choice != 5);
break;
case 0:
cout << "Exiting...\n";
break;
default:
cout << "Invalid choice. Please try again.\n";
}
} while (choice != 0);
}
break;
case 0:
cout << "Exiting...\n";
break;
default:
cout << "Invalid choice. Please try again.\n";
}
} while (choice != 0);
}
};
// Main function
int main() {
PoliceFIRSystem policeSystem;
```

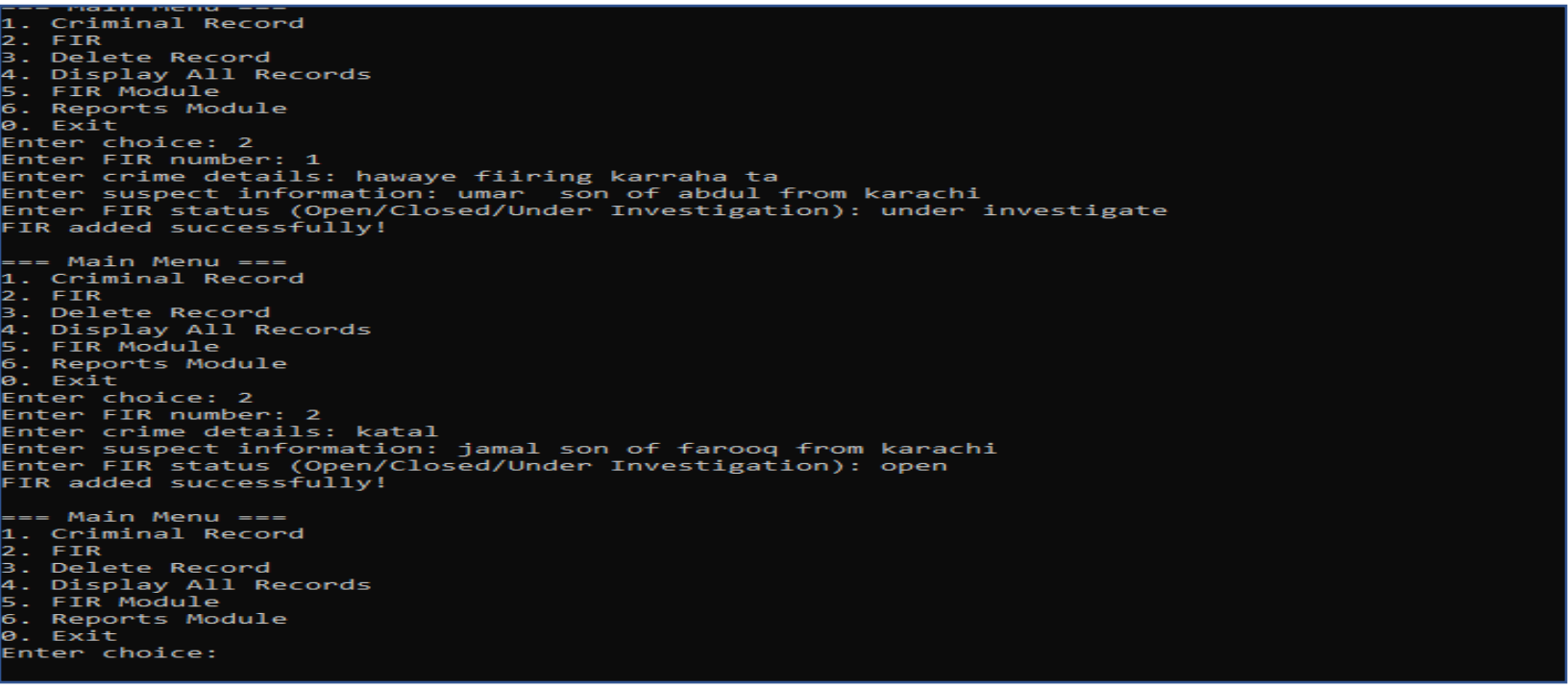
```
policeSystem.run();

return 0;

}
```

Admin Menu :

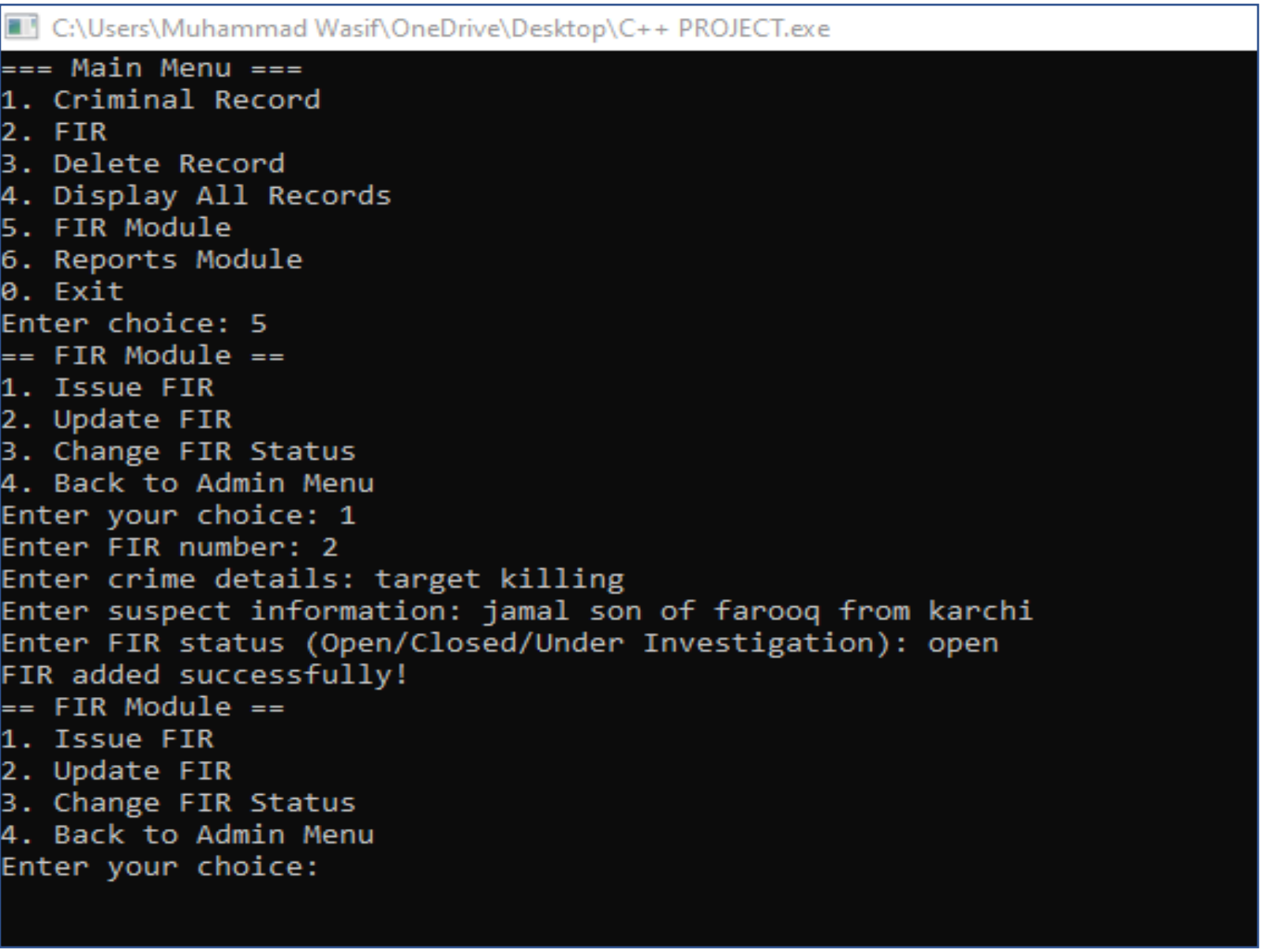


ADD Criminal Record And Fir





Fir Module


C:\Users\Muhammad Wasif\OneDrive\Desktop\C++ PROJECT.exe
=== Main Menu ===
1. Criminal Record
2. FIR
3. Delete Record
4. Display All Records
5. FIR Module
6. Reports Module
0. Exit
Enter choice: 5
== FIR Module ==
1. Issue FIR
2. Update FIR
3. Change FIR Status
4. Back to Admin Menu
Enter your choice: 1
Enter FIR number: 2
Enter crime details: target killing
Enter suspect information: jamal son of farooq from karchi
Enter FIR status (Open/Closed/Under Investigation): open
FIR added successfully!
== FIR Module ==
1. Issue FIR
2. Update FIR
3. Change FIR Status
4. Back to Admin Menu
Enter your choice: