Tour Management System

Session: 2025 – 2029

Submitted by:

Jahanzaib Gull 2025(S)-CS-63

Supervised by:

Mr. Laeeq Khan Niazi

Department of Computer Science

University of Engineering and Technology

Lahore Pakistan

Table of Contents

Contents:

Abstract: Tour Management System

System Overview:

This C++ program implements a comprehensive Tour Management System with dual user roles (admin and customer) and file-based data persistence. The application provides tour management functionalities for administrators and booking/feedback capabilities for customers.

Core Features:

- **Administrative Functions:**
  - Tour Management: Add, view (sorted by price), and delete tours
  - User Management: View and delete registered users
  - Feedback Monitoring: Access all customer feedback submissions
  - Booking Oversight: View all customer bookings with automatic discount calculations
- **Customer Functions:**
  - Account Management: Registration, login, and profile updates
  - Tour Exploration: View tours by category, see detailed descriptions
  - Booking System: Reserve tours with group size pricing
  - Feedback Mechanism: Submit and view ratings/comments

Technical Implementation:

- Data Storage: Uses text files ("tours.txt", "users.txt") for persistent data storage
- Data Structures: Parallel arrays for managing tours and users
- User Interface: Console-based menu system with clear navigation
- Security: Basic authentication for admin and user roles

Key Components:

- Tour Data: Name, destination, price, duration, category, description
- User Data: Credentials, contact info, bookings, feedback
- Business Logic: Dynamic pricing with tiered discounts (5%-20% based on total)

Limitations

- Basic console interface without graphical elements
- Simple authentication without encryption
- The system does not support multiple users accessing or modifying data at the same time.

This system provides a complete solution for small tour operators needing digital management of offerings, customers, and bookings through a straightforward terminal interface.

📌 Short Description of the Project

The **Tour Management System** is a desktop-based application developed for adding and managing tours, searching and viewing tours, tour booking processing, and giving feedbacks. It allows businesses to manage tour details, monitor tours, handle customer bookings, and keep a feedbacks given by customers. The project was developed using C++.

👤 Users of the Application

- **Admin/Manager**:
  - Add, update, or delete tours.
  - View feedbacks given by various customers.
  - View and delete users.
- **Customers:**
  - View tours list.
  - Book tour and view booked tour.
  - Give feedback and view feedback.
  - View and update customer profile

☑ Functional Requirements

1. Add new tours.
2. Update or delete existing tours.
3. View all tours.
4. Remove user.
5. View bookings done by various customers.
6. View feedbacks given by various customers.
7. Check tours before booking.
8. Sorting tours based on price and category.
9. Accounts based login/sign up.
10. Generate user friendly error messages.

Wireframes (along its code)

- **Landing Page:**

```
void main_header()
{
    cout << "*****************************************************************" << endl;
    cout << "*                    TOUR MANAGEMENT SYSTEM                     *" << endl;
    cout << "*****************************************************************" << endl;
    cout << "*                      PRESTIGE  PATHWAYS                       *" << endl;
    cout << "*****************************************************************" << endl;

    cout << endl;
}
void separation()
{
    cout << "-------------------------------------------------------" << endl;
}
string main_menu()
{
    cout << "Main Menu -->" << endl;
    separation();
    cout << "1- Admin Login" << endl;
    cout << "2- Customer Sign Up" << endl;
    cout << "3- Customer Login" << endl;
    cout << "4- Exit" << endl;
    cout << " Your Option---";
    string option;
    cin >> option;
    return option;
}
```

```
D:\New folder (2)\PF\Semeste
*****************************************************************
*                    TOUR MANAGEMENT SYSTEM                     *
*****************************************************************
*                      PRESTIGE  PATHWAYS                       *
*****************************************************************

Main Menu -->
-------------------------------------------------------
1- Admin Login
2- User Sign Up
3- User Login
4- Exit
 Your Option---
```

- **Admin Login:**

```
void admin_login()
{

        cout << "Admin Login -->" << endl;
        separation();
        cout << "Enter Admin Name: ";
        cin >> admin_Name;
        cout << "Enter Admin Password: ";
        cin >> admin_Password;
        if (admin_Name == "admin" && admin_Password == "admin123")
        {
            cout << "Welcome " << admin_Name << endl;
            login_admin=true;
        }
        else
        {
            cout << "Invalid credentials. Please try again." << endl;
        }
        clear_screen();
        return;
}
```

```
D:\New folder (2)\PF\Semeste
**********************************************************************
*                     TOUR MANAGEMENT SYSTEM                         *
**********************************************************************
*                       PRESTIGE  PATHWAYS                           *
**********************************************************************

Admin Login -->
----------------------------------------------------------
Enter Admin Name: admin
Enter Admin Password: admin123
Welcome admin
----------------------------------------------------------
Press any key to continue
```

- **Customer Sign up:**

```
void user_signup()
{
    load_users_from_file();
    if (user_count >= MAX_USERS)
    {
        cout << "User limit reached." << endl;
        return;
    }
    cout << "User Sign Up -->" << endl;
    separation();
    cout << "Enter Username: ";
    cin.ignore();
    getline(cin, user_nameA[user_count]);
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == user_nameA[user_count])
        {
            cout << "User name already exists." << endl;
            cout << "Please choose a different user name." << endl;
            return;
        }
    }
    cout << "Enter User Password: ";
    getline(cin, user_passwordA[user_count]);
    cout << "Enter User Full Name: ";
    getline(cin, user_full_nameA[user_count]);
    cout << "Enter User Email: ";
    getline(cin, user_emailA[user_count]);
    cout << "Enter User Phone Number: ";
    getline(cin, user_phoneA[user_count]);

    user_feedbackA[user_count] = " ";
    user_feedback_starsA[user_count] = " ";
    user_total_priceA[user_count] = 0;
    user_booked_tour_nameA[user_count] = "";
    user_count++;
    cout << "User signed up successfully." << endl;
    save_users_to_file();
}
```

```
D:\New folder (2)\PF\Semeste
************************************************************
*                    TOUR MANAGEMENT SYSTEM                *
************************************************************
*                      PRESTIGE  PATHWAYS                  *
************************************************************

User Sign Up -->
------------------------------------------------------------
Enter UserName: ahmed
Enter User Password: ahmed123
Enter User Full Name: ahmed yaseen
Enter User Email: ahmed123@gmail.com
Enter User Phone Number: 1234567890
User signed up successfully.
------------------------------------------------------------
Press any key to continue
```

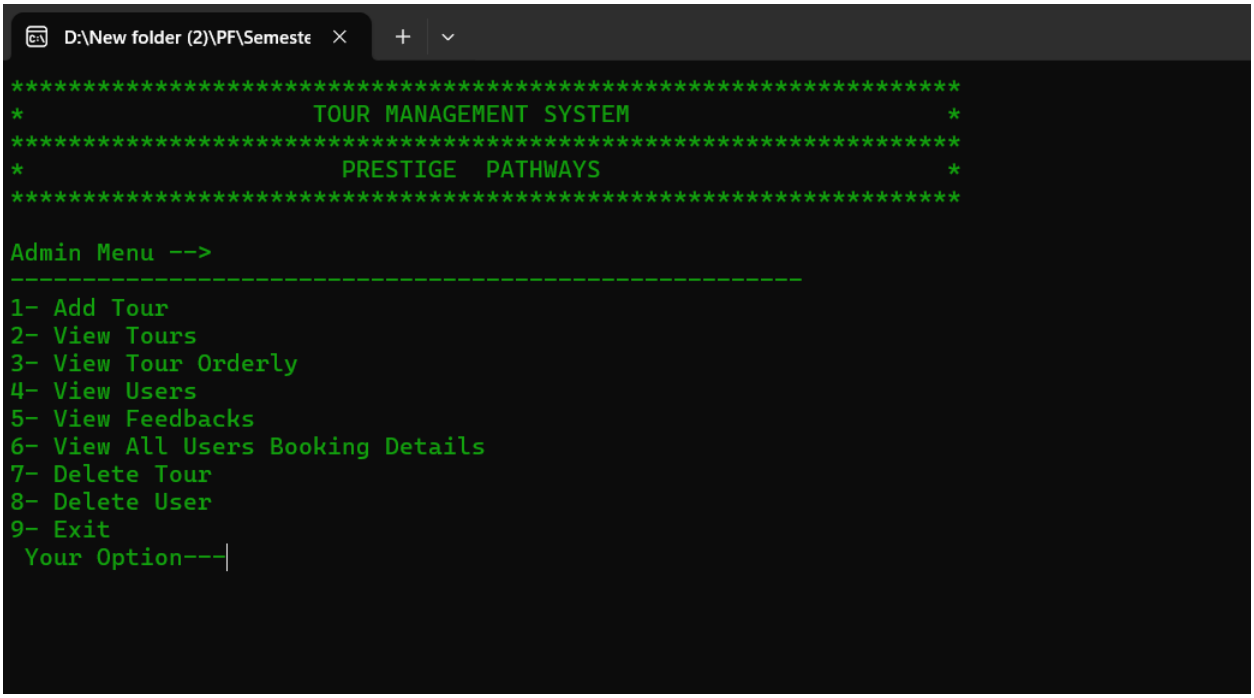
- **Customer Login:**

```
void user_login()
{
    load_users_from_file();
    if (user_count == 0)
    {
        cout << "no user available." << endl;
        return;
    }
    cout << "User Login -->" << endl;
    separation();
    string user_name, user_password;
    cout << "Enter User Name: ";
    cin >> user_name;
    cout << "Enter User Password: ";
    cin >> user_password;
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == user_name && user_passwordA[i] == user_password)
        {
            cout << "Welcome " << user_nameA[i] << endl;
            logined_user_name = user_nameA[i];
            loggedin_user_password = user_passwordA[i];
            login_user = true;
            return;
        }
    }
    cout << "Invalid username or password. Please try again"<<endl;
}
```

```
D:\New folder (2)\PF\Semeste
**********************************************************************
*                      TOUR MANAGEMENT SYSTEM                        *
**********************************************************************
*                        PRESTIGE  PATHWAYS                          *
**********************************************************************

User Login -->
----------------------------------------------------------
Enter User Name: ahmed
Enter User Password: ahmed123
Welcome ahmed
----------------------------------------------------------
Press any key to continue
```

- **Admin Menu:**

```
string admin_menu()
{
    cout << "Admin Menu -->" << endl;
    separation();
    cout << "1- Add Tour" << endl;
    cout << "2- View all Tours" << endl;
    cout << "3- View Tour Orderly" << endl;
    cout << "4- View Users" << endl;
    cout << "5- View Feedbacks" << endl;
    cout << "6- View All Users Booking Details" << endl;
    cout << "7- Delete Tour" << endl;
    cout << "8- Delete User" << endl;
    cout << "9- Exit" << endl;
    cout << " Your Option---";
    string option;
    cin >> option;
    return option;
}
```



- **Admin Menu > Add Tour:**

```
void add_tour()
{
    load_tours_from_file();
    if (tour_count >= MAX_TOURS)
    {
        cout << "Tour limit reached. Cannot add more tours." << endl;
        return;
    }
    cout << "Add Tour -->" << endl;
    separation();
    cout << "Enter Tour Name
    cin.ignore();
    getline(cin, tour_nameA[tour_count]);
    for (int i = 0; i < tour_count; i++)
    {
        if (tour_nameA[i] == tour_nameA[tour_count])
        {
            cout << "Tour name already exists." << endl;
            return;
        }
    }
    cout << "Enter Tour Destination: ";
    getline(cin, tour_destinationA[tour_count]);
    cout << "Enter Tour Pickup Point: ";
    getline(cin, tour_pickupA[tour_count]);
    cout << "Enter Tour Duration: ";
    getline(cin, tour_durationA[tour_count]);
    cout << "Enter Tour Price: ";
    cin >> tour_priceA[tour_count];
    cout << "Enter Tour Description: ";
    cin.ignore();
    getline(cin, tour_discriptionA[tour_count]);
    cout << "Enter Tour Category: ";
    getline(cin, tour_categoryA[tour_count]);
    tour_count++;
    cout << "Tour added successfully." << endl;
    save_tours_to_file();
```

int tour_count

Generate Copilot summary

```
D:\New folder (2)\PF\Semeste

**********************************************************************
*                     TOUR MANAGEMENT SYSTEM                         *
**********************************************************************
*                       PRESTIGE  PATHWAYS                           *
**********************************************************************

Add Tour -->
------------------------------------------------------
Enter Tour Name: tour1
Enter Tour Destination: babusar
Enter Tour Pickup Point: islamabad
Enter Tour Duration: 8 days
Enter Tour Price: 50000
Enter Tour Description: A tour from Islamabd to Babusar.
Enter Tour Category: adventure
Tour added successfully.
------------------------------------------------------
Press any key to continue
```

- **Admin Menu > List all tours or Customer Menu > List all tours:**

```
void view_tours()
{
    load_tours_from_file();
    if (tour_count == 0)
    {
        cout << "No tours available." << endl;
        return;
    }
    cout << "-----------All Tours-----------" << endl;
    cout << "Tour Name\tTour Price\tTour Duration\tTour Pickup\tTour destination\tTour Category" << endl;
    for (int i = 0; i < tour_count; i++)
    {
        cout << tour_nameA[i] << "\t\t" << tour_priceA[i] << "\t\t" << tour_durationA[i] << "\t\t"
        << tour_pickupA[i] << "\t\t" << tour_destinationA[i] << "\t\t" << tour_categoryA[i] << endl;
    }
}
```

```
D:\New folder (2)\PF\Semeste
**********************************************************************
*                    TOUR MANAGEMENT SYSTEM                          *
**********************************************************************
*                     PRESTIGE  PATHWAYS                             *
**********************************************************************

-----------All Tours-----------
Tour Name       Tour Price      Tour Duration   Tour Pickup     Tour destination        Tour Category
tour1           50000           8 days          islamabad               babusar         adventure
tour2           20000           4 days          islamabad               bahawalpur              historical
tour3           25000           4 days          faisalabad              muhenjo daro            historical
tour4           30000           6               islamabad               sakardu         adventure
-------------------------------------------------------
Press any key to continue
```

- **Admin Menu > Sorted tours or Customer Menu > Sorted tours:**

```
void view_tour_orderly()
{
    load_tours_from_file();
    if (tour_count == 0)
    {
        cout << "No tours available." << endl;
        return;
    }
    // higher to lower
    cout << "-----------Tours in order of price----------" << endl;
    cout << "Tour Name\tTour Price\tTour Duration\tTour Pickup\tTour destination\tTour Category" << endl;
    for (int i = 0; i < tour_count; i++)
    {
        for (int j = i + 1; j < tour_count; j++)
        {
            if (tour_priceA[i] > tour_priceA[j])
            {
                string temp_name = tour_nameA[i];
                tour_nameA[i] = tour_nameA[j];
                tour_nameA[j] = temp_name;

                int temp_price = tour_p
                tour_priceA[i] = tour_p
                tour_priceA[j] = temp_p
                std::__cxx11::string tour_durationA[20]
                Generate Copilot summary

                string temp_duration = tour_durationA[i];
                tour_durationA[i] = tour_durationA[j];
                tour_durationA[j] = temp_duration;

                string temp_pickup = tour_pickupA[i];
                tour_pickupA[i] = tour_pickupA[j];
                tour_pickupA[j] = temp_pickup;

                string temp_destination = tour_destinationA[i];
                tour_destinationA[i] = tour_destinationA[j];
                tour_destinationA[j] = temp_destination;

                string temp_category = tour_categoryA[i];
                tour_categoryA[i] = tour_categoryA[j];
                tour_categoryA[j] = temp_category;

                string temp_discription = tour_discriptionA[i];
                tour_discriptionA[i] = tour_discriptionA[j];
                tour_discriptionA[j] = temp_discription;
            }
        }
    }
    for (int i = tour_count - 1; i >= 0; i--)
    {
        cout << tour_nameA[i] << "\t\t" << tour_priceA[i] << "\t\t" << tour_durationA[i] << "\t\t" << tour_pickupA[i] << "\t\t" << tour_destinationA[i] << "\t\t" << tour_categoryA[i] << endl;
    }
    save_tours_to_file();
}
```

```
D:\New folder (2)\PF\Semeste

**********************************************************************
*                    TOUR MANAGEMENT SYSTEM                          *
**********************************************************************
*                      PRESTIGE  PATHWAYS                            *
**********************************************************************

----------Tours in order of price----------
Tour Name       Tour Price      Tour Duration   Tour Pickup     Tour destination        Tour Category
tour1           50000           8 days          islamabad               babusar         adventure
tour4           30000           6               islamabad               sakardu         adventure
tour3           25000           4 days          faisalabad              muhenjo daro            historical
tour2           20000           4 days          islamabad               bahawalpur              historical
------------------------------------------------------
Press any key to continue
```

- **Admin Menu > list all users:**

```
void view_users()
{
    load_users_from_file();
    if (user_count == 0)
    {
        cout << "No users available." << endl;
        return;
    }
    cout << "----------All Users----------" << endl;
    cout << "Username\tPassword\tEmail\t\t\t\tPhone Number\t\tFull Name" << endl;
    for (int i = 0; i < user_count; i++)
    {
        cout << user_nameA[i] << "\t\t" << user_passwordA[i] << "\t\t" << user_emailA[i] << "\t\t" << user_phoneA[i] << "\t\t" << user_full_nameA[i] << endl;
    }
}
```

```
D:\New folder (2)\PF\Semeste
**************************************************************
*                   TOUR MANAGEMENT SYSTEM                   *
**************************************************************
*                     PRESTIGE  PATHWAYS                     *
**************************************************************

----------All Users----------
Username        Password        Email                   Phone Number            Full Name
ahmed           ahmed123                ahmed123@gmail.com              1234567890              ahmed yaseen
ayesha          ayesha123               ayesha123@gmail.com             1234567890              ayesha ali
umer            umer123         umer123@gmail.com               1234567890              M.umer
amna            amna            amna123@gmail.com               1234567890              amna ali
-----------------------------------------------------
Press any key to continue
```

- **Customer Menu > View tours by catrgory:**

```
void view_tour_by_category()
{
    load_tours_from_file();
    if (tour_count == 0)
    {
        cout << "No tours available." << endl;
        return;
    }
    cout << "View Tours by Category -->" << endl;
    separation();
    string category;
    cout << "Enter Tour Category: ";
    cin.ignore();
    getline(cin, category);
    cout << "-----------Tours in " << category << " Category----------" << endl;
    cout << "Tour Name\tTour Price\tTour Duration\tTour Pickup\tTour destination" << endl;
    for (int i = 0; i < tour_count; i++)
    {
        if (tour_categoryA[i] == category)
        {
            cout << tour_nameA[i] << "\t \t" << tour_priceA[i] << "\t \t" << tour_durationA[i] << "\t \t" << tour_pickupA[i] << "\t \t" << tour_destinationA[i] << endl;
        }
    }
}
```

```
D:\New folder (2)\PF\Semeste
**************************************************************
*                    TOUR MANAGEMENT SYSTEM                  *
**************************************************************
*                      PRESTIGE  PATHWAYS                    *
**************************************************************

View Tours by Category -->
-----------------------------------------------------
Enter Tour Category: historical
-----------Tours in historical Category-----------
Tour Name       Tour Price      Tour Duration   Tour Pickup     Tour destination
tour2           20000           4 days          islamabad               bahawalpur
tour3           25000           4 days          faisalabad              muhenjo daro
-----------------------------------------------------
Press any key to continue
```

- **Customer Menu > View tour discription:**

```
void tour_discrription()
{
    load_tours_from_file();
    if (tour_count == 0)
    {
        cout << "No tours available." << endl;
        return;
    }
    string tour_name;
    cout << "Enter Tour Name to view description: ";
    cin.ignore();
    getline(cin, tour_name);
    cout << "--------Tour Discription--------" << endl;
    for (int i = 0; i < tour_count; i++)
    {
        if (tour_nameA[i] == tour_name)
        {
            cout << "Tour Name: " << tour_nameA[i] << endl;
            cout << "Tour Price: " << tour_priceA[i] << endl;
            cout << "Tour Duration: " << tour_durationA[i] << endl;
            cout << "Tour Pickup Point: " << tour_pickupA[i] << endl;
            cout << "Tour Destination: " << tour_destinationA[i] << endl;
            cout << "Tour Category: " << tour_categoryA[i] << endl;
            cout << "Tour Description: " << endl
                 << "\t" << tour_discriptionA[i] << endl;
            return;
        }
    }
    cout << "Tour not found." << endl;
}
```

```
D:\New folder (2)\PF\Semeste
**********************************************************************
*                     TOUR MANAGEMENT SYSTEM                         *
**********************************************************************
*                       PRESTIGE  PATHWAYS                           *
**********************************************************************

Enter Tour Name to view description: tour3
--------Tour Discription--------
Tour Name: tour3
Tour Price: 25000
Tour Duration: 4 days
Tour Pickup Point: faisalabad
Tour Destination: muhenjo daro
Tour Category: historical
Tour Description:
        A tour from faialabad to muhenjo daro
----------------------------------------------------------
Press any key to continue
```

- **Customer Menu > View profile:**

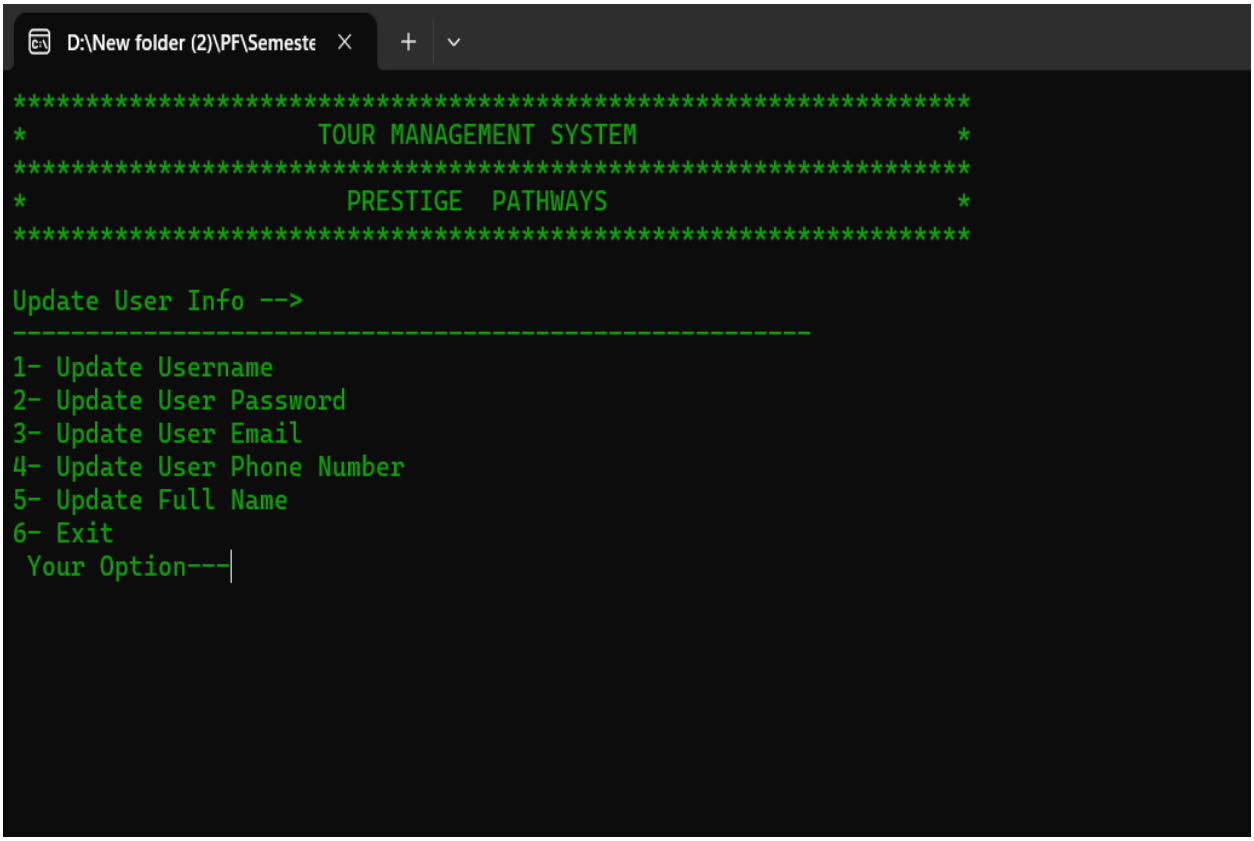
```
void view_profile()
{
    load_users_from_file();
    cout << "User Profile -->" << endl;
    separation();
    cout << "Username: " << logined_user_name << endl;
    cout << "User Password: " << loggedin_user_password << endl;
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == logined_user_name)
        {
            cout << "User Email: " << user_emailA[i] << endl;
            cout << "User Phone Number: " << user_phoneA[i] << endl;
            cout << "User Full Name: " << user_full_nameA[i] << endl;
            break;
        }
    }
}
```

```
D:\New folder (2)\PF\Semeste
************************************************************
*                   TOUR MANAGEMENT SYSTEM                 *
************************************************************
*                     PRESTIGE  PATHWAYS                   *
************************************************************

User Profile -->
-----------------------------------------------------
Username: umer
User Password: umer123
User Email: umer123@gmail.com
User Phone Number: 1234567890
User Full Name: M.umer
-----------------------------------------------------
Press any key to continue
```

- **Customer Menu > Update profile:**

```
void update_user_info()
{
    load_users_from_file();
    cout << "Update User Info -->" << endl;
    separation();
    char option;
    cout << "1- Update Username" << endl;
    cout << "2- Update User Password" << endl;
    cout << "3- Update User Email" << endl;
    cout << "4- Update User Phone Number" << endl;
    cout << "5- Update Full Name" << endl;
    cout << "6- Exit" << endl;
    cout << " Your Option---";
    cin >> option;

    if (option == '1')
    {
        change_user_name();
    }
    else if (option == '2')
    {
        change_password();
    }
    else if (option == '3')
    {
        change_user_email();
    }
    else if (option == '4')
    {
        change_user_phone();
    }
    else if (option == '5')
    {
        change_user_full_name();
    }
    else if (option == '6')
    {
        cout << "Exiting..." << endl;
        return;
    }
    else
    {
        cout << "Invalid option. Please try again." << endl;
    }
    save_users_to_file();
}
```

```
void change_password()
{
    cout << "--------Change User Password--------" << endl;
    string new_user_password;
    cout << "Enter New User Password: ";
    cin.ignore();
    getline(cin, new_user_password);
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == logined_user_name)
        {
            user_passwordA[i] = new_user_password;
            cout << "Password changed successfully." << endl;
            return;
        }
    }
}
void change_user_name()
{
    cout << "--------Change Username--------" << endl;
    string new_user_name;
    cout << "Enter New Username: ";
    cin.ignore();
    getline(cin, new_user_name);
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == logined_user_name)
        {
            user_nameA[i] = new_user_name;
            logined_user_name = user_nameA[i];
            cout << "User name changed successfully." << endl;
            return;
        }
    }
}
void change_user_email()
{
    cout << "--------Change User Email--------" << endl;
    string new_user_email;
    cout << "Enter New User Email: ";
    cin.ignore();
    getline(cin, new_user_email);
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == logined_user_name)
        {
            user_emailA[i] = new_user_email;
            cout << "User email changed successfully." << endl;
            return;
        }
    }
}
void change_user_phone()
{
    cout << "--------Change User Phone Number--------" << endl;
    string new_user_phone_number;
    cout << "Enter New User Phone Number: ";
    cin.ignore();
    getline(cin, new_user_phone_number);
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == logined_user_name)
        {
            user_phoneA[i] = new_user_phone_number;
```



- **Customer Menu > Tour booking:**

```
void book_tour_no_of_people_by_user()
{
    load_tours_from_file();
    if (tour_count == 0)
    {
        cout << "No tours available." << endl;
        return;
    }
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == logined_user_name && user_total_priceA[i] != 0)
        {
            cout << "You have already booked a tour." << endl;
            cout << "if you want to book another tour by changing the previous one,so enter yes or no" << endl;
            string option;
            cin >> option;
            if (option == "yes" || option == "Yes")
            {
                user_total_priceA[i] = 0;
                cout << "You can book a new tour now." << endl;
            }
            else      You, 4 days ago • final project
            {
                cout << "You can continue with your previous booking." << endl;
                return;
            }
        }
    }

    cout << "--------Tour Booking--------" << endl;
    cout << "Enter Tour Name to book: ";
    string user_book_tour_name;
    cin.ignore();
    getline(cin, user_book_tour_name);
    cout << "Enter number of people: ";
    int number_of_people;
    cin >> number_of_people;
    if (number_of_people <= 0)
    {
        cout << "Invalid number of people." << endl;
```

```
    return;
}
for (int i = 0; i < tour_count; i++)
{
    if (tour_nameA[i] == user_book_tour_name)
    {
        system("CLS");
        main_header();
        cout << "--------Tour Booking--------" << endl;
        user_book_tour_name = tour_nameA[i];
        cout << "tour name: " << tour_nameA[i] << endl;
        cout << "Tour Price per person: " << tour_priceA[i] << endl;
        cout << "Tour Duration: " << tour_durationA[i] << endl;
        cout << "Tour Pickup Point: " << tour_pickupA[i] << endl;
        cout << "Tour Destination: " << tour_destinationA[i] << endl;
        cout << "Tour Category: " << tour_categoryA[i] << endl;
        cout << "Tour Description: " << tour_discriptionA[i] << endl;
        cout << "Number of People: " << number_of_people << endl;
        for (int j = 0; j < user_count; j++)
        {
            if (user_nameA[j] == logined_user_name)
            {
                user_nameA[j] = logined_user_name;
                user_total_priceA[j] = tour_priceA[i] * number_of_people;
                cout << "Total Price: " << user_total_priceA[j] << endl;
                user_booked_tour_nameA[j] = user_book_tour_name;
            }
        }
        cout << "Tour booked successfully." << endl;
        cout << "Thank you for booking with us!" << endl;
        save_users_to_file();
        return;
    }
}
cout << "Tour not found." << endl;
cout << "Please check the tour name and try again." << endl;
}
```

```
D:\New folder (2)\PF\Semeste
**********************************************************************
*                      TOUR MANAGEMENT SYSTEM                        *
**********************************************************************
*                        PRESTIGE  PATHWAYS                          *
**********************************************************************

--------Tour Booking--------
tour name: tour1
Tour Price per person: 50000
Tour Duration: 8 days
Tour Pickup Point: islamabad
Tour Destination: babusar
Tour Category: adventure
Tour Description: A tour from Islamabd to Babusar.
Number of People: 5
Total Price: 250000
Tour booked successfully.
Thank you for booking with us!
----------------------------------------------------------
Press any key to continue
```

- **Customer Menu > View booking details:**

```
void user_booking_details()
{
    load_users_from_file();
    cout << "--------User Booking Details--------" << endl;
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == logined_user_name && user_total_priceA[i] != 0)
        {
            cout << "Username: " << user_nameA[i] << endl;
            cout << "Tour Name: " << user_booked_tour_nameA[i] << endl;
            cout << "Total price: " << user_total_priceA[i] << endl;
            float discount_amount;
            if (user_total_priceA[i] > 2000000)
            {
                discount_amount = user_total_priceA[i] * 0.2;
            }
            else if (user_total_priceA[i] > 1000000)
            {
                discount_amount = user_total_priceA[i] * 0.15;
            }
            else if (user_total_priceA[i] > 500000)
            {
                discount_amount = user_total_priceA[i] * 0.1;
            }
            else if (user_total_priceA[i] > 100000)
            {
                discount_amount = user_total_priceA[i] * 0.05;
            }
            else
            {
                discount_amount = 0;
            }
            cout << "amount after discount: " << user_tot                    unt << endl;
            cout << "paid amount: " << user_total_priceA[i] - discount_amount << endl;
            return;
        }
    }
    cout << "No booking details found." << endl;
    cout << "Please book a tour first." << endl;
}
```

int i

Generate Copilot summary

```
D:\New folder (2)\PF\Semeste

**********************************************************************
*                    TOUR MANAGEMENT SYSTEM                          *
**********************************************************************
*                      PRESTIGE  PATHWAYS                            *
**********************************************************************

--------User Booking Details--------
Username: umer
Tour Name: tour1
Total price: 250000
amount after discount: 237500
paid amount: 237500
------------------------------------------------------------
Press any key to continue
```

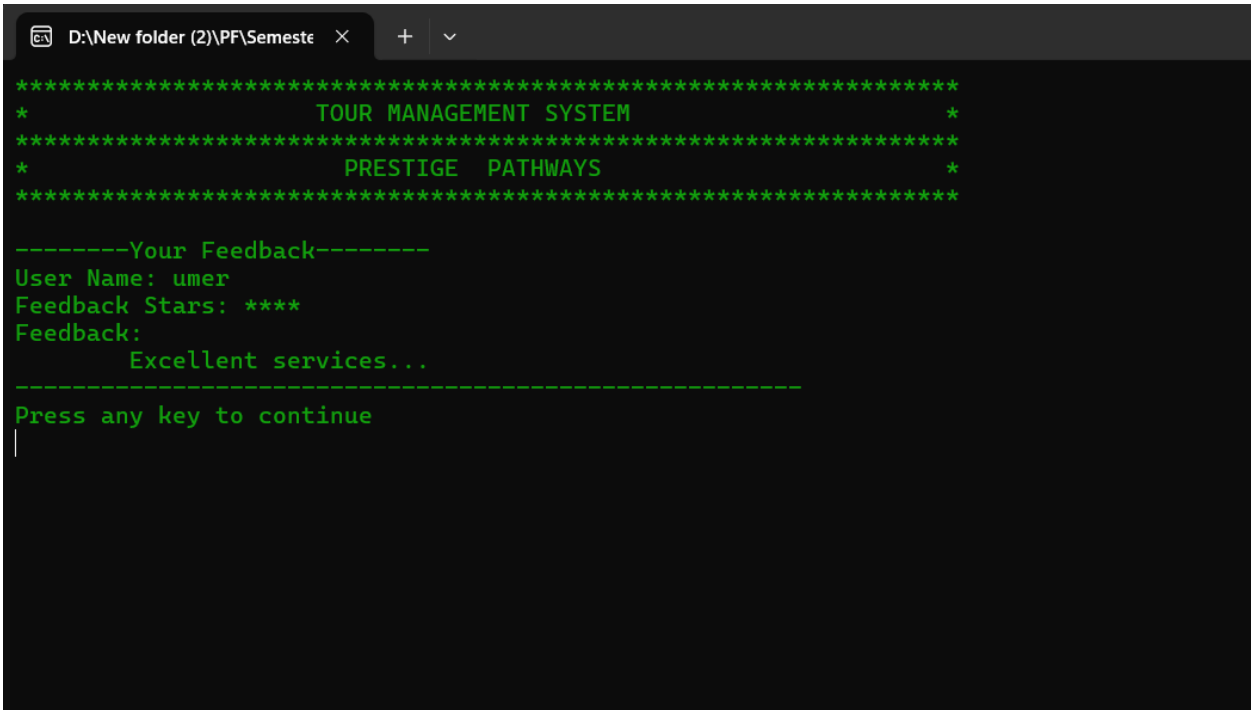
- **Customer Menu > Give feedback:**

```cpp
void give_feedback()
{
    cout << "--------Feedback--------" << endl;
    cout << "Enter your feedback stars (1-5): ";
    cin.ignore();
    string feedback_stars;
    getline(cin, feedback_stars);
    cout << "Enter your feedback: ";
    string feedback;
    getline(cin, feedback);
    if (feedback_stars.empty() || feedback.empty())
    {
        cout << "All fields are required." << endl;
        return;
    }
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == logined_user_name)
        {
            user_feedback_starsA[i] = feedback_stars;
            user_feedbackA[i] = feedback;

            cout << "Feedback added successfully." << endl;
            save_users_to_file();
            return;
        }
    }
}
```

```
******************************************************************
*                     TOUR MANAGEMENT SYSTEM                     *
******************************************************************
*                       PRESTIGE  PATHWAYS                       *
******************************************************************

--------Feedback--------
Enter your feedback stars (1-5): ****
Enter your feedback: Excellent services...
Feedback added successfully.
-------------------------------------------------------
Press any key to continue
```

- **Customer Menu > View your feedback:**

```
void view_your_feedback()
{
    load_users_from_file();
    cout << "--------Your Feedback--------" << endl;
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == logined_user_name && user_feedbackA[i] != " ")
        {
            cout << "User Name: " << user_nameA[i] << endl;
            cout << "Feedback Stars: " << user_feedback_starsA[i] << endl;
            cout << "Feedback: " << endl
                 << "\t" << user_feedbackA[i] << endl;
        }
    }
}
```



- **Customer Menu > View feedbacks or Admin Menu > View feedbacks**

```
void view_feedbacks()
{
    load_users_from_file();
    if (user_count == 0)
    {
        cout << "no user available." << endl;
        return;
    }
    cout << "--------All Feedbacks--------" << endl;
    for (int i = 0; i < user_count; i++)
    {
        if (user_feedbackA[i] == " ")
        {
            continue;
        }
        cout << "Username: " << user_nameA[i] << endl;
        cout << "Feedback Stars: " << user_feedback_starsA[i] << endl;
        cout << "Feedback: " << endl
             << "\t" << user_feedbackA[i] << endl;
        cout << "-------------------------------------" << endl;
    }
}
```

```
D:\New folder (2)\PF\Semeste
**********************************************************************
*                      TOUR MANAGEMENT SYSTEM                        *
**********************************************************************
*                        PRESTIGE  PATHWAYS                          *
**********************************************************************

--------All Feedbacks--------
Username: umer
Feedback Stars: ****
Feedback:
        Excellent services...
-------------------------------------
Username: amna
Feedback Stars: *****
Feedback:
        Good experience
-------------------------------------
------------------------------------------------------
Press any key to continue
```

- **Admin Menu > View all users bookings:**

```
void view_all_users_booking_details()
{
    load_users_from_file();
    if (user_count == 0)
    {
        cout << "no user available." << endl;
        return;
    }
    cout << "--------All Users Booking Details--------" << endl;
    for (int i = 0; i < user_count; i++)
    {
        if (user_total_priceA[i] != 0)
        {
            cout << "User Name: " << user_nameA[i] << endl;
            cout << "Tour Name: " << user_booked_tour_nameA[i] << endl;
            cout << "Total price: " << user_total_priceA[i] << endl;
            float discount_amount;
            if (user_total_priceA[i] > 2000000)
            {
                discount_amount = user_total_priceA[i] * 0.2;
            }
            else if (user_total_priceA[i] > 1000000)
            {
                discount_amount = user_total_priceA[i] * 0.15;
            }
            else if (user_total_priceA[i] > 500000)
            {
                discount_amount = user_total_priceA[i] * 0.1;
            }
            else if (user_total_priceA[i] > 100000)
            {
                discount_amount = user_total_priceA[i] * 0.05;
            }
            else
            {
                discount_amount = 0;
            }
            cout << "amount after discount: " << user_total_priceA[i] - discount_amount << endl;
            cout << "paid amount: " << user_total_priceA[i] - discount_amount << endl;
            cout << "---------------------------------------" << endl;
        }
    }
}
```

```
D:\New folder (2)\PF\Semeste
**************************************************************
*                    TOUR MANAGEMENT SYSTEM                  *
**************************************************************
*                      PRESTIGE  PATHWAYS                    *
**************************************************************

--------All Users Booking Details--------
User Name: umer
Tour Name: tour1
Total price: 250000
amount after discount: 237500
paid amount: 237500
---------------------------------------
User Name: amna
Tour Name: tour3
Total price: 100000
amount after discount: 100000
paid amount: 100000
---------------------------------------
------------------------------------------------------
Press any key to continue
```

- **Admin Menu > Delete tour:**

```
void dlt_tour()
{
    load_users_from_file();
    load_tours_from_file();
    if (tour_count == 0)
    {
        cout << "No tours available to delete." << endl;
        return;
    }
    cout << "Delete Tour -->" << endl;
    separation();
    string tour_name;
    cout << "Enter Tour Name to delete: ";
    cin.ignore();
    getline(cin, tour_name);
    for (int i = 0; i < user_count; i++)
    {
        if (user_booked_tour_nameA[i] == tour_name)
        {
            user_booked_tour_nameA[i] = "";
            user_total_priceA[i] = 0;
        }
    }
    for (int i = 0; i < tour_count; i++)
    {
        if (tour_nameA[i] == tour_name)
        {

            for (int j = i; j < tour_count - 1; j++)
            {
                tour_nameA[j] = tour_nameA[j + 1];
                tour_priceA[j] = tour_priceA[j + 1];
                tour_durationA[j] = tour_durationA[j + 1];
                tour_pickupA[j] = tour_pickupA[j + 1];
                tour_destinationA[j] = tour_destinationA[j + 1];
                tour_categoryA[j] = tour_categoryA[j + 1];
                tour_discriptionA[j] = tour_discriptionA[j + 1];
            }
            tour_count--;
            cout << "Tour deleted successfully." << endl;
            save_tours_to_file();
            save_users_to_file();
            return;
        }
    }
    cout << "Invalid tour name. Please enter correct tour name."<<endl;
}
```

```
D:\New folder (2)\PF\Semeste
**************************************************************
*                    TOUR MANAGEMENT SYSTEM                  *
**************************************************************
*                      PRESTIGE  PATHWAYS                    *
**************************************************************

Delete Tour -->
------------------------------------------------------------
Enter Tour Name to delete: tour4
Tour deleted successfully.
------------------------------------------------------------
Press any key to continue
```

- **Admin Menu > Delete user:**

```
void dlt_user()
{
    load_users_from_file();
    if (user_count == 0)
    {
        cout << "No users available to delete." << endl;
        return;
    }
    cout << "Delete User -->" << endl;
    separation();
    string user_name;
    cout << "Enter Username to delete: ";
    cin >> user_name;
    for (int i = 0; i < user_count; i++)
    {
        if (user_nameA[i] == user_name)
        {
            for (int j = i; j < user_count - 1; j++)
            {
                user_nameA[j] = user_nameA[j + 1];
                user_passwordA[j] = user_passwordA[j + 1];
                user_emailA[j] = user_emailA[j + 1];
                user_phoneA[j] = user_phoneA[j + 1];
                user_full_nameA[j] = user_full_nameA[j + 1];
                user_feedbackA[j] = user_feedbackA[j + 1];
                user_feedback_starsA[j] = user_feedback_starsA[j + 1];
                user_total_priceA[j] = user_total_priceA[j + 1];
                user_booked_tour_nameA[j] = user_booked_tour_nameA[j + 1];
            }
            user_count--;
            cout << "User deleted successfully." << endl;
            save_users_to_file();
            return;
        }
    }
    cout << "Invalid username. Please enter correct username."<<endl;
}
```

```
D:\New folder (2)\PF\Semeste
**********************************************************************
*                      TOUR MANAGEMENT SYSTEM                        *
**********************************************************************
*                         PRESTIGE  PATHWAYS                         *
**********************************************************************

Delete User -->
----------------------------------------------------------
Enter Username to delete: ahmed
User deleted successfully.
----------------------------------------------------------
Press any key to continue
```

🗃 Data Structures

```
// data structures
const int MAX_TOURS = 20;
string tour_nameA[MAX_TOURS];
string tour_destinationA[MAX_TOURS];
string tour_pickupA[MAX_TOURS];
string tour_durationA[MAX_TOURS];
string tour_discriptionA[MAX_TOURS];
string tour_categoryA[MAX_TOURS];
int tour_priceA[MAX_TOURS];
int tour_count = 0;

const int MAX_USERS = 20;
string user_nameA[MAX_USERS];
string user_passwordA[MAX_USERS];
string user_emailA[MAX_USERS];
string user_phoneA[MAX_USERS];
string user_full_nameA[MAX_USERS];
string user_feedbackA[MAX_USERS];
string user_feedback_starsA[MAX_USERS];
int user_total_priceA[MAX_USERS];
string user_booked_tour_nameA[MAX_USERS];      You, 4 days ago • final project …
int user_count = 0;

string admin_Name, admin_Password;
string logined_user_name, loggedin_user_password;
bool login_user = false;
bool login_admin=false;
```

🛠 Function Prototypes

```
// function prototypes
void main_header();
void separation();
string main_menu();
void admin_login();
string admin_menu();
void add_tour();
void save_tours_to_file();
void load_tours_from_file();
void view_tours();
void view_tour_orderly();
void view_users();
void dlt_tour();
void dlt_user();
void user_signup();
void user_login();
void save_users_to_file();
void load_users_from_file();
string user_menu();
void view_profile();
void view_tour_by_category();
void update_user_info();
void change_user_name();
void change_password();
void change_user_email();
void change_user_phone();
void change_user_full_name();
void tour_discrription();
void clear_screen();
void give_feedback();
void view_your_feedback();
void view_feedbacks();
void about_us();
void green();
void book_tour_no_of_people_by_user();
void user_booking_details();
void view_all_users_booking_details();
```

🔁 Functions Working Flow

🧪 Test Cases

1. Admin Login

- **Test Case 1: Valid admin credentials**
  - Input: Username = "admin", Password = "admin123"
  - Expected: Successful login, access to admin menu
- **Test Case 2: Invalid admin credentials**
  - Input: Username = "admin", Password = "wrongpass"
  - Expected: Error message, denied access

2. User Signup

- **Test Case 3: New user registration**
  - Input: Unique username, valid details
  - Expected: Account created, saved to users.txt
- **Test Case 4: Duplicate username**
  - Input: Existing username
  - Expected: Error message, signup rejected

3. User Login

- **Test Case 5: Valid user credentials**
  - Input: Correct username/password
  - Expected: Successful login, user menu access
- **Test Case 6: Invalid credentials**
  - Input: Wrong username/password
  - Expected: Error message, retry prompt

4. Tour Management (Admin)

- **Test Case 7: Add a new tour**
  - Input: Tour details (name, price, etc.)
  - Expected: Tour appears in view_tours, saved to tours.txt
- **Test Case 8: Delete a tour**
  - Input: Existing tour name
  - Expected: Tour removed from list/file

5. Booking System (User)

- **Test Case 9: Book a valid tour**
  - Input: Tour name + number of people
  - Expected: Booking confirmed, price calculated
- **Test Case 10: Book non-existent tour**
  - Input: Invalid tour name
  - Expected: Error message, no booking

6. Feedback Submission

- **Test Case 11: Submit feedback**
  - Input: Rating (1-5) + comments
  - Expected: Feedback saved to user profile
- **Test Case 12: View feedbacks (Admin)**
  - Expected: All feedbacks displayed in admin panel

7. Edge Cases

- **Test Case 13: Max user/tour limit**
  - Action: Add 21st user/tour
  - Expected: Error message "Limit reached"
- **Test Case 14: Empty input fields**
  - Action: Submit blank login/signup form
  - Expected: Validation error

🚀 Future Work

- Replace the console-based system with a modern graphical-user interference for better user experience.
- Replace Text Files with a Database for efficient data management.
- Role-based access control for different admin levels.
- Multi-Language Support to deal with international customers.
- API for Third-Party Integrations that Allow travel agencies to fetch tour data via REST API.
- Replace fixed-size arrays with vectors (C++) or linked lists for Dynamic Memory Allocation.

Conclusions:

This Tour Management System was my first-semester project, developed in C++ using file handling and basic data structures. Through this project, I gained valuable insights into **real-world programming** and **software development principles**. Here's what I learned:

- Understood arrays, loops, functions, and conditional statements in C++.
- Learned file handling (fstream) to store and retrieve data.
- Implemented modular programming by dividing the system into functions for better readability.
- Encountered and fixed logical errors and Improve error handling.
- Learned the importance of testing.

Student Reg. No. : 2025(S)-CS-63 **Student Name.** Jahanzaib Gull

	A-Extensive Evidence	**B-Convincing Evidence**	**C-Limited Evidence**	**D-No Evidence**
Documentation Formatting **Grade:**	All the documentation meets all the criteria.	Documentation is well formatted but some of the criteria is not fulfilled.	Documentation is required a lot of improvement.	Documentation is not Available
Documentation Formatting Criteria: In **Binder**, **Title** Page, **Header**-Footers, Font **Style**, Font **Size** all are all consistence and according to given **guidelines**. Project **Poster** is professionally design and well presented				
Documentation Contents **Grade:**	Documentation includes all of the criteria.	Documentation meet more than 80% of the criteria given.	Documentation meet more than 50% of the criteria.	When the documentation meet less than 50% of the criteria.
Documentation Contents Criteria: Title Page - **Table** of Contents - Project **Abstract** - **Functional** Requirements - **Wire** Frames –**Data Flow** Diagram-**Data** Structure (Arrays)-**Function** Headers and Description - **Algorithms** and Flow Charts of all functions- **Test Cases** are defined -Project **Code.** - **Weakness** in the Project and **Future** Directions. - **Conclusion** and What you **Learn** from the Project and Course and What is your **Future** Planning.				
Project Complexity **Grade:**	Project has at least 2 user's types and each user has at least 5 functionalities.	Project complexity meet 80% criteria given in extensive evidence	Project complexity meet 50% criteria given in extensive evidence	Project complexity meet less than 50% criteria given in extensive evidence
Code Style **Grade:**	All Code style criteria is followed	All code style criteria followed but some improvements required	A lot of improvements required in coding style.	**Did not follow** code style,
Code Style Criteria: Consistent code style. Code is well indented. Variable and Function names are well defined. White Spaces are well used. Comments are added.				
Code Documentation Mapping **Grade:**	Code and documentation is synchronized.	Code and documentation does not synchronized at **some** places	Code and documentation does not synchronized at **many** places	Code and documentation **does not** synchronized.
Data Structure (Arrays) **Grade:**	Data structure is sufficient for the project requirements	Data Structure is sufficient but require improvement to meet project requirements.	Data structure is not sufficient and need a lot of improvement	Data Structure is not properly identified and declared.
Sorting Features **Grade:**	Sort working 100% and generating useful report	Sorting Feature is working but sorted data is not useful for project.	Sorting feature is partial implemented	Project do not contain sorting
Modularity **Grade:**	Meet all Modularity criteria	Meet all Modularity criteria but at some places it is missing	Do not sufficiently meet the modularity criteria.	No modularity or very minimum modularity.
Modularity criteria: Functions are defined for each major feature. Functions are independent (identify from parameter list and return types)- Demo Data Functionality Added-At least Two Unit Tests are defined.				
Validations **Grade:**	Validations on all number type inputs are applied	Validations are applied but at some places it is missing.	Validations are missing at lot of places	No Validations are used
Recommendation Feature	Proper meaning full recommendation is present into system	Partial Recommendation is implemented	Implemented but not meaning full.	Not implemented
Presentation and Demo **Grade:**	Presentation and Demo was 100% working	Presentation and Demo require some improvements	Presentation and Demo require a lot of improvements	Presentation was not ok and Demo was not working
Student Understanding with the Code. **Grade:**	Student has complete understanding how the code is working and knows the concept.	Student has good understand but some place he does not know the concepts	Student has a very little understand and lack the major concepts.	Student does not have any level of understanding of the code.

Checked by: Self Check

Student Reg. No. : 2025(S)-CS-63 **Student Name.** Jahanzaib Gull

	A-Extensive Evidence	**B-Convincing Evidence**	**C-Limited Evidence**	**D-No Evidence**
Documentation Formatting **Grade:**	All the documentation meets all the criteria.	Documentation is well formatted but some of the criteria is not fulfilled.	Documentation is required a lot of improvement.	Documentation is not Available
Documentation Formatting Criteria: In **Binder**, **Title** Page, **Header**-Footers, Font **Style**, Font **Size** all are all consistence and according to given **guidelines**. Project **Poster** is professionally design and well presented				
Documentation Contents **Grade:**	Documentation includes all of the criteria.	Documentation meet more than 80% of the criteria given.	Documentation meet more than 50% of the criteria.	When the documentation meet less than 50% of the criteria.
Documentation Contents Criteria: Title Page - **Table** of Contents - Project **Abstract** - **Functional** Requirements - **Wire** Frames –**Data Flow** Diagram-**Data** Structure (Arrays)-**Function** Headers and Description - **Algorithms** and Flow Charts of all functions- **Test Cases** are defined -Project **Code.** - **Weakness** in the Project and **Future** Directions. - **Conclusion** and What you **Learn** from the Project and Course and What is your **Future** Planning.				
Project Complexity **Grade:**	Project has at least 2 user's types and each user has at least 5 functionalities.	Project complexity meet 80% criteria given in extensive evidence	Project complexity meet 50% criteria given in extensive evidence	Project complexity meet less than 50% criteria given in extensive evidence
Code Style **Grade:**	All Code style criteria is followed	All code style criteria followed but some improvements required	A lot of improvements required in coding style.	**Did not follow** code style,
Code Style Criteria: Consistent code style. Code is well indented. Variable and Function names are well defined. White Spaces are well used. Comments are added.				
Code Documentation Mapping **Grade:**	Code and documentation is synchronized.	Code and documentation does not synchronized at **some** places	Code and documentation does not synchronized at **many** places	Code and documentation **does not** synchronized.
Data Structure (Arrays) **Grade:**	Data structure is sufficient for the project requirements	Data Structure is sufficient but require improvement to meet project requirements.	Data structure is not sufficient and need a lot of improvement	Data Structure is not properly identified and declared.
Sorting Features **Grade:**	Sort working 100% and generating useful report	Sorting Feature is working but sorted data is not useful for project.	Sorting feature is partial implemented	Project do not contain sorting
Modularity **Grade:**	Meet all Modularity criteria	Meet all Modularity criteria but at some places it is missing	Do not sufficiently meet the modularity criteria.	No modularity or very minimum modularity.
Modularity criteria: Functions are defined for each major feature. Functions are independent (identify from parameter list and return types)- Demo Data Functionality Added-At least Two Unit Tests are defined.				
Validations **Grade:**	Validations on all number type inputs are applied	Validations are applied but at some places it is missing.	Validations are missing at lot of places	No Validations are used
Recommendation Feature	Proper meaning full recommendation is present into system	Partial Recommendation is implemented	Implemented but not meaning full.	Not implemented
Presentation and Demo **Grade:**	Presentation and Demo was 100% working	Presentation and Demo require some improvements	Presentation and Demo require a lot of improvements	Presentation was not ok and Demo was not working
Student Understanding with the Code. **Grade:**	Student has complete understanding how the code is working and knows the concept.	Student has good understand but some place he does not know the concepts	Student has a very little understand and lack the major concepts.	Student does not have any level of understanding of the code.

Checked by: Click or tap here to enter text.