National University of Computer & Emerging Sciences
Karachi Campus

Project Report

CyberNet/eSports Café Management System:

Café Azula

Course Teacher:
Sir Zulfiqar Ali Memon
Ma'am Erum Shaheen

Group Members:
Zaid Bin Shahab 19K-1512
Umer Ahmed 19K-0181

Table of Contents

Introduction 4
Description 4
Functionalities of the system 5
Customer Perspective 5
Staff Perspective 5
System Perspective 5
Utilizing Technologies and Software's 6
Normalization: 7
Bookings 7
Functional Dependencies: 7
2NF: 8
3NF: 8
Events: 9
Functional Dependencies: 9
2NF: 9
Leaderboard: 10
Functional Dependencies: 10
2NF: 10
3NF: 10
Staff: 11
Functional Dependencies: 11
2NF: 11
ER Diagram: 12
Database Tables 13
Implementation: 14
Classes: 14
Joins: 14
Triggers: 14

Views: 15
PL/SQL and Procedures: 16
Built-In Functions: 16
Insert: 17
Update: 17
Delete: 17
Visuals: 19
Login Page: 19
Customer Page: 20
Book Seats: 20
Profile Page: 21
View Events: 22
View Leaderboard: 23
Cancel Seats: 23
Booking History: 24
Staff Page: 25
Career Profile: 26
Account Management: 26
Leaderboard Management: 27
Game Management: 28
Computer Management: 29
Event Management: 30
Seat Management: 31

Introduction

Café Azula has been around since 1977 and has recently started to revamp its structure from the ground up. Formerly a small traditional café, it now functions as an Internet café and Esports center, letting the customer enjoy the best of virtual world with an arsenal of one of the high-class desktop systems and an unmatched internet connection, but still retains its eastern influences, which is, chai.

The café has garnered so much popularity that now it requires an online booking in advance, even allowing the customers, you, to occupy specific seats with the specific timings. There is no limit to what you can achieve in this café by experiencing our luxury/premium seats with next-Gen computer systems for smooth gaming and designed infrastructure to motivate teamwork collaborations.

Description

The Cybernet/eSports Café Management System is a multifaceted system that encompasses entirely around the ease of our customers whilst keeping each of their public records and personal achievements intact and safe in our system. The availability of games with weekly, monthly and special events that increases the competition among our top and rising players. The system also manages the records of every staff with their supervisors. The system is interactive and robust enough to identify each account separately and without the need of individual user portals for customer and staffs.

Special vouchers in game presented by Café Azula makes the gaming environment more fun as players can experience the game at discounted prices but with the same excitement and amazing experiences. What keeps our gaming café more competitive, interactive and exciting is the top 10 players of every game in our leaderboard system, top 3 players of each game at the end of month in the leaderboard system will get a chance to avail vouchers for themselves as a reward of their hard work and ultimate skills. The Esports event organized by café gets more exciting as there are special prizes waiting for the winners.

Functionalities of the system

Final product of the project will be able to perform these functionalities that are given below:

Customer Perspective

- Customer can either log on to their already existing account in our database or can create a new one.
- Customer can book their timings and specific seats with seats type (Premium or Standard) in advance through our portal application. Customers that have account can also avail vouchers to play games in discounted prices.
- Customers can also register themselves and their team in the on-going events by paying the prices of events.

Staff Perspective

- Staff can log on their already existing account or a new one for a newly hired staff member can be created by the Supervisors. Staff members with the selected job nature can have specific supervisors that they report to.
- Staff Members with the job title as “Event Manager” can create and set new events with its required details. Staff Members with the job title as “Manager” or higher posts can delete the records of customers as well as staff members. Staff Members with their designation as “Accounts” can create and set vouchers. Staff Members with job nature as “Technical Support” can create new records of new computer systems in the café with their specific id’s.

System Perspective

- System can calculate fees by itself looking at the type of customer, type of game and type of seats specified by the customer.
- System also identifies and separates customer account with Staff accounts and redirect them to the next screen accordingly.
- System will show the leaderboard of each game accordingly.

Utilizing Technologies and Software's

We are using these following technologies and software in our project:

1. **Microsoft SQL Server** for creating, keeping and managing the database of the project.
2. **C#** as our language of implantation in the project.
3. **Microsoft Windows Forms** for creating the front-end of the Project.
4. **Microsoft Visual Studio** as our IDE for the project.

Normalization:

Bookings

Seat No	Customer ID	Account No	Date of Booking	Start Time	End time	Cust Name	Phone No	Username	Password
Seat Current Status	Premium YesNo	Computer ID	CPU	GPU	Ram	Net speed	Amount Paid		

Events

Event ID	Event Name	Start Time	End Time	Game ID	Game Name	Genre	Description	Popularity	Max Participants
Poster Link									

Leaderboard

Game ID	Game Name	Genre	Description	Popularity	Customer ID	Cust Name	Phone No	Account No	Username
Password	Rank								

Staff

Staff ID	Staff Name	Phone No	Salary	Account No	Position	Supervisor ID	Username	Password

Bookings:

Bookings relation is already in 1NF. But there are insert, update and delete anomalies as Customer can't exist without booking and seat and so on.

Functional Dependencies:

1. Seat No, Customer ID, Start Time, End Time → Whole Table **(Candidate Key)**
2. Seat No → Seat CurrentStatus, Premium YesNo, Computer ID, CPU, GPU, Ram, Net Speed
3. Computer ID → CPU, GPU, RAM, Net Speed
4. Customer ID → Cust Name, Phone No, Account No, Username, Password
5. Account No → Username, Password
6. Username → Account No, Password

2NF:

1. Using Partial Dependency FD 2:

Seats

Seat No	Current Status	Premium YesNo	Computer ID	CPU	GPU	Ram	Net Speed

2. Using Partial Dependency FD 4:

Customers

Customer ID	Cust Name	Phone No	Account No	Username	Password

Bookings:

Seat No	Date of Booking	Start Time	End Time	Amount Paid	Customer ID

3NF:

1. Using Transitive Dependency FD 3:

Computers

Computer ID	CPU	GPU	RAM	Net Speed

2. Using Partial Dependency FD 2:

Seats

Seat No	Current Status	Premium YesNo	Computer ID

3. Using Partial Dependency FD 5:

Accounts:

Account No	Username	Password

4. Using Partial Dependency FD 4:

Customers

Customer ID	Cust Name	Phone No	Account No

Bookings:

Seat No	Date of Booking	Start Time	End Time	Amount Paid	Customer ID

This table is also in BCNF since no NPK determines a PK.

Events:

Events table is already in 1NF. There are insert, delete and update anomalies as a game cannot exist without there being an event held for it.

Functional Dependencies:

1. Event Name, Start Time, Game ID → Whole Table **(Candidate Key)**
2. Game ID → Game Name, Genre, Description, Popularity

2NF:

1. Using Partial Dependencies FD 2:

Games:

Game ID	Name	Genre	Description	Popularity

Events

Event ID	Event Name	Start Time	End Time	Max Participants	Poster Link

This table is already in 3NF and BCNF since there are no Transitive dependencies and no NPK determines a PK.

Leaderboard:

Leaderboard is already in 1NF. Customers table has already been made due to FDs in Bookings relation. Games table has already been made due to FDs in Events table.

Functional Dependencies:

1. Game ID, Customer ID, Rank → Whole table **(Candidate Key)**
2. Game ID → Game Name, Genre, Description, Popularity
3. Customer ID → Cust Name, Phone No, Account No, Username, Password
4. Account No → Username, Password
5. Username → Account No, Password

2NF:

1. Using Partial dependency FD 2. There is already a Games relation due to Events relation normalization. Therefore, Game ID will stay in Leaderboard relation as FK.

Leaderboard

Game ID	Customer ID	Cust Name	Phone No	Account No	Username	Password	Rank

2. Using Partial dependency FD 3. There is already a Customers relation due to Bookings relation normalization. Therefore, Customer ID will stay in Leaderboard relation as FK.

Leaderboard

Game ID	Customer ID	Account No	Username	Password	Rank

3NF:

1. Using Transitive dependency FD 4. There is already an Accounts relation that is referenced in Customers, so we will remove these attributes from leaderboard table.

Leaderboard

Game ID	Customer ID	Rank

Staff:

Staff table is already in 1NF. Accounts table has already been made in Bookings relation.

Functional Dependencies:

1. Phone No, Account No → Whole table **(Candidate Key)**

Assumption: Phone No is unique, and only one staff can be registered per phone number.

2. Account No → Username, Password
3. Username → Account No, Password

2NF:

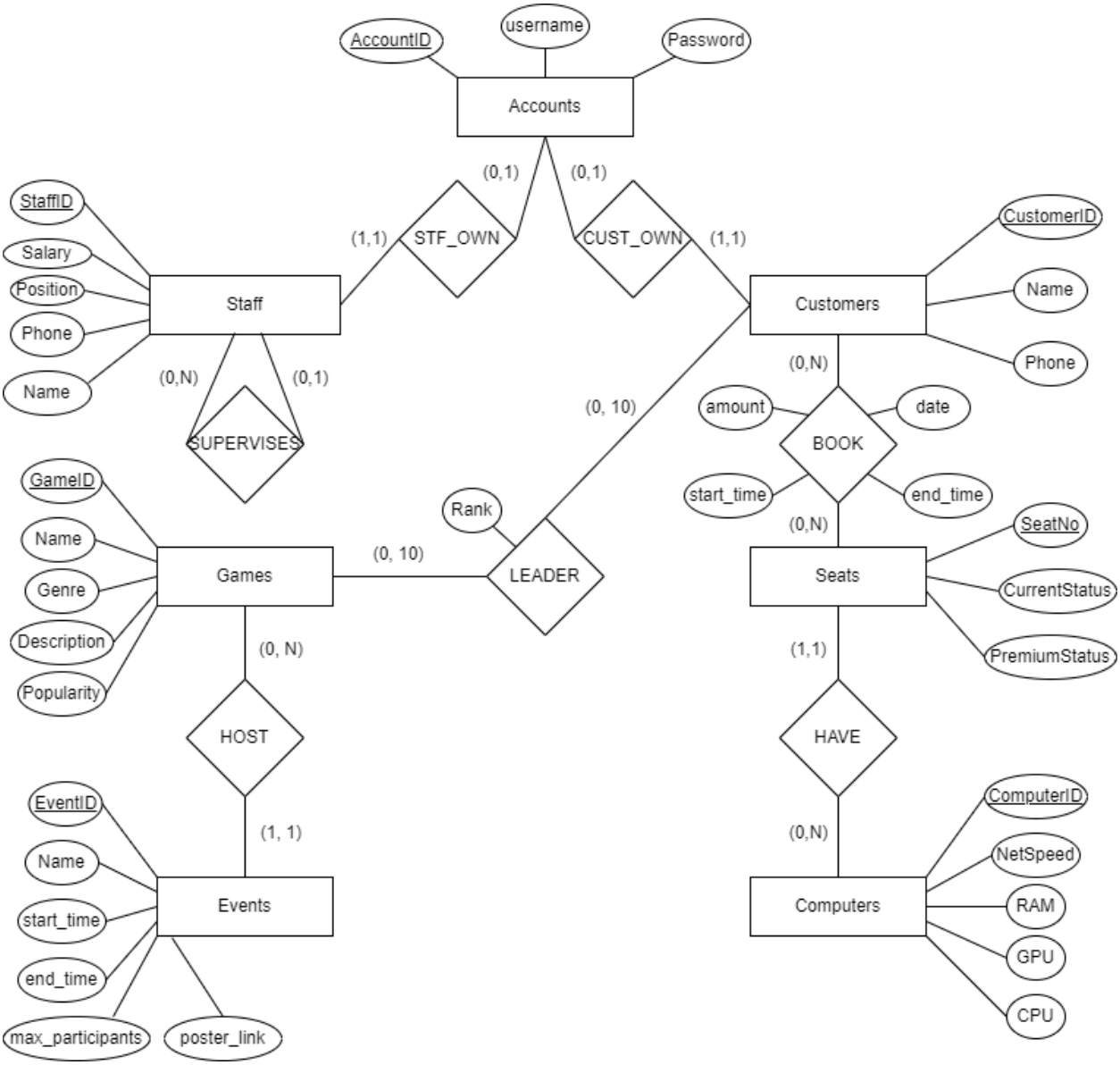
1. Using Partial dependency FD 2. There is already an accounts table, so its PK will be in Staff table as an FK.

Staff

Staff ID	Staff Name	Phone No	Salary	Account No	Position	Supervisor ID

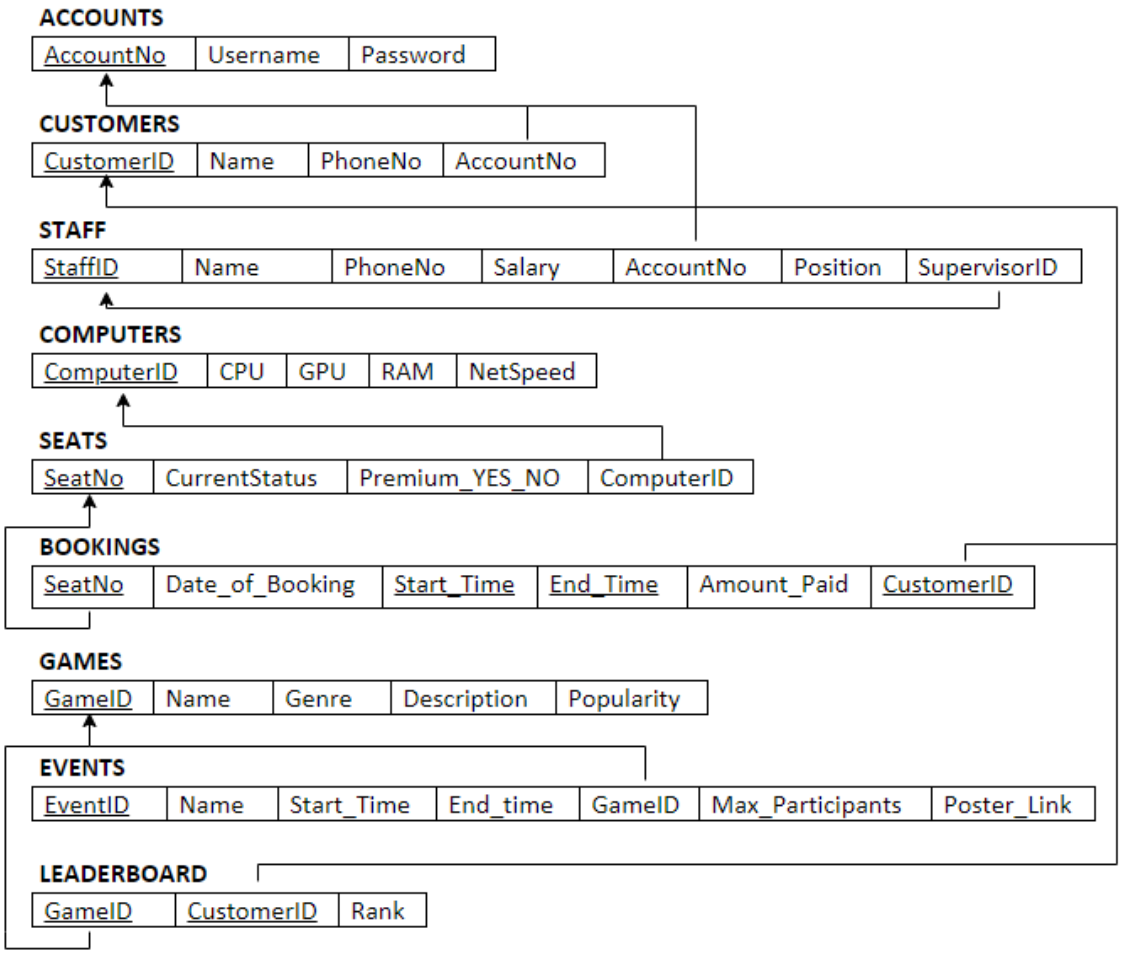
ER Diagram:

The database system consists of 8 entities and 8 relationships, amounting to 10 tables.



Database Tables

Following are the 10 tables in the database system, along with all the Primary and Foreign Keys marked by underline and arrows.



Implementation:

Classes:

- AzulaForm
- Login
- CafeFeatures
- TicketingSystem
- ChangePasswordForm
- ChangeNameForm
- ReceiptWindow

Joins:

```
1.  string newQuery = "SELECT * FROM Events AS E LEFT OUTER JOIN Games AS G ON E.GameID =
    G.GameID WHERE Start_Time > '" + DateTime.Now.ToString("yyyy-MM-dd HH:mm:ss") + "'";
2.  SqlCommand cmd = new SqlCommand(newQuery, cnn);
3.  SqlDataReader reader = cmd.ExecuteReader();
```

```
1.  string sql = "select Accounts.Username, Staff.StaffName, Staff.Salary, Staff.Position from
    Accounts, Staff where Accounts.AccountNo = Staff.AccountNo";
2.  SqlDataAdapter adapt = new SqlDataAdapter(sql, cnn);
3.  DataTable data = new DataTable();
```

```
1.  string sql = "Select Username from Accounts, Customers where Accounts.AccountNo =
    Customers.AccountNo and Customers.CustomerID = " +
    trackcusid[CustNameDropDown.SelectedIndex];
2.  SqlCommand cmd = new SqlCommand(sql, cnn);
3.  ShowCustGamerTag.Text = "Gamer Tag : " + cmd.ExecuteScalar().ToString();
```

And many more…

Triggers:

1. Trigger to delete customer or staff when account is deleted. Which could not be done with simple cascade ON DELETE due to Staff Relation contained an FK referencing PK of itself.

```
2.  CREATE OR ALTER TRIGGER Account_delete_cascade_trigger ON Accounts
3.  INSTEAD OF DELETE
4.  AS
5.  BEGIN
6.    DELETE FROM Customers
```

```
7.    WHERE AccountNo IN (SELECT AccountNo FROM Deleted);
8.    PRINT('Customer with accountNo deleted');
9.
10.   DELETE FROM Staff
11.   WHERE AccountNo IN (SELECT AccountNo FROM Deleted);
12.   PRINT('Staff with accountNo deleted');
13.
14.   DELETE FROM Accounts
15.   WHERE AccountNo IN (SELECT AccountNo FROM Deleted);
16.   PRINT('Deleted the row');
17. END;
```

2. Trigger to set the supervisorID to NULL if that the record of that supervisor is deleted. This was also prevented with ON DELETE due to the FK referencing PK of the relation itself.

```
1.  CREATE OR ALTER TRIGGER Supervisor_delete_trigger ON Staff
2.  INSTEAD OF DELETE
3.  AS
4.  DECLARE
5.    @deletedStaffID INT;
6.  BEGIN
7.    UPDATE Staff
8.    Set Supervisor_ID = NULL
9.    WHERE Supervisor_ID IN (SELECT StaffID FROM Deleted);
10.   PRINT('Referenced SupervisorIDs set NULL');
11.
12.   DELETE FROM Staff
13.   WHERE StaffID IN (SELECT StaffID FROM Deleted);
14.   PRINT('Deleted the row');
15. END;
```

Views:

1. Leaderboard_Details view to combine the Accounts, Customers, Games and Leaderboard information into a single table.

```
1.  CREATE VIEW Leaderboard_Details AS
2.  SELECT L.GameRank AS "Rank", A.Username AS "Gamer Tag", C.CustName AS "Name", G.GameName AS
    "Game"
3.  FROM Leaderboard AS L JOIN Customers AS C
4.                                                          ON L.CustomerID = C.CustomerID
5.                                                  JOIN Accounts AS A
6.                                                          ON C.AccountNo = A.AccountNo
7.                                                  JOIN Games AS G
8.                                                          ON L.GameID = G.GameID;
```

2. Customer_Profile view to display all profile information of a customer.

```
1.  CREATE VIEW Customer_Profile AS
2.  SELECT CustomerID, CustName, PhoneNo, Username, AccPassword
3.  FROM Accounts AS A JOIN Customers AS C
4.  ON A.AccountNo = C.AccountNo;
```

PL/SQL and Procedures:

The following procedure is designed to set the current status of the seat to be “Occupied” or “Free”. It sets it according to whether the current time is in the interval between the Start_Time and End_Time of a booking on that particular seat number.

```
CREATE OR REPLACE PROCEDURE Update_Seat_Status()
IS
BEGIN
	FOR c IN (SELECT SeatNo, Start_Time, End_Time FROM Bookings) LOOP
		IF (c.Start_Time <= SYSDATETIME AND c.End_Time >= SYSDATETIME) THEN
			UPDATE Seats SET CurrentStatus = 'Occupied' WHERE SeatNo = c.SeatNo;
		ELSE
			UPDATE Seats SET CurrentStatus = 'Free' WHERE SeatNo = c.SeatNo;
		END IF;
	END LOOP;
END;
```

Built-In Functions:

Max function:

```
1.  string sql1 = "Select gameid from Games where GameName = '" + gamename + "'", sql2 = "select
    max(EventID) from Events",
2.  sql3 = "select * from Events where EventName = '" + EventName + "' and Start_Time = '" +
    start_time.ToString("yyyy-MM-dd HH:mm:ss") + "'";
```

Count function:

```
1.  string newQuery = "SELECT count(*) as cnt FROM Seats";
2.  SqlCommand cmd = new SqlCommand(newQuery, cnn);
3.  SqlDataReader reader = cmd.ExecuteReader();
```

Insert:

```
1.  queryString = "INSERT INTO Bookings VALUES(" + SeatNo.ToString() + ", '" +
    DateTime.Now.ToString("yyyy-MM-dd") + "', '" + Start_Time.ToString("yyyy-MM-dd HH:mm:ss") +
    "', '" + End_Time.ToString("yyyy-MM-dd HH:mm:ss") + "', " + Amount_Paid.ToString() + ", " +
    CustomerID.ToString() + ")";
2.  cmd = new SqlCommand(queryString, cnn);
```

```
1.  sql = "Insert into Games values (" + (Maxid + 1) + ", '" + gamename + "', '" + genre + "',
    '" + gamedesc + "', " + popular + ")";
```

```
1.  sql1 = "Insert into Accounts (AccountNo, Username, AccPassword) values (" + (MaxAcc + 1) +
    ", '" + username + "', '" + password + "')";
2.                  sql2 = "Insert into Customers (CustomerID, CustName, PhoneNo, AccountNo)
    values (" + (MaxCum + 1) + ", '" + Name + "', '" + phone + "', " + (MaxAcc + 1) + ")";
```

And many more…

Update:

```
1.  string newQuery = "UPDATE Accounts SET AccPassword = '" + NewPasswordTextBox.Text.ToString()
    + "' WHERE AccountNo = (SELECT AccountNo FROM Customers WHERE CustomerID = " +
    customerID_PassForm.ToString() + ")";
2.  SqlCommand cmd = new SqlCommand(newQuery, cnn);
3.  cmd.ExecuteNonQuery();
```

```
1.  string newQuery = "UPDATE Customers SET CustName = '" + NewNameTextBox.Text.ToString() + "'
    WHERE CustomerID = " + customerID_NameForm.ToString();
2.  SqlCommand cmd = new SqlCommand(newQuery, cnn);
3.  cmd.ExecuteNonQuery();
```

```
1.  string newQuery = "UPDATE Customers SET PhoneNo = '" + NewNameTextBox.Text.ToString() + "'
    WHERE CustomerID = " + customerID_NameForm.ToString();
2.  SqlCommand cmd = new SqlCommand(newQuery, cnn);
3.  cmd.ExecuteNonQuery();
```

Delete:

```
1.  string sql = "Delete from Leaderboard where GameID = " + gameid + " and CustomerID = " +
    customerid;
2.  cmd = new SqlCommand(sql, cnn);
3.  cmd.ExecuteNonQuery();
```

```
1.  string sql = "Delete from Games where GameID = " + gameid;
2.  cmd = new SqlCommand(sql, cnn);
3.  cmd.ExecuteNonQuery();
```

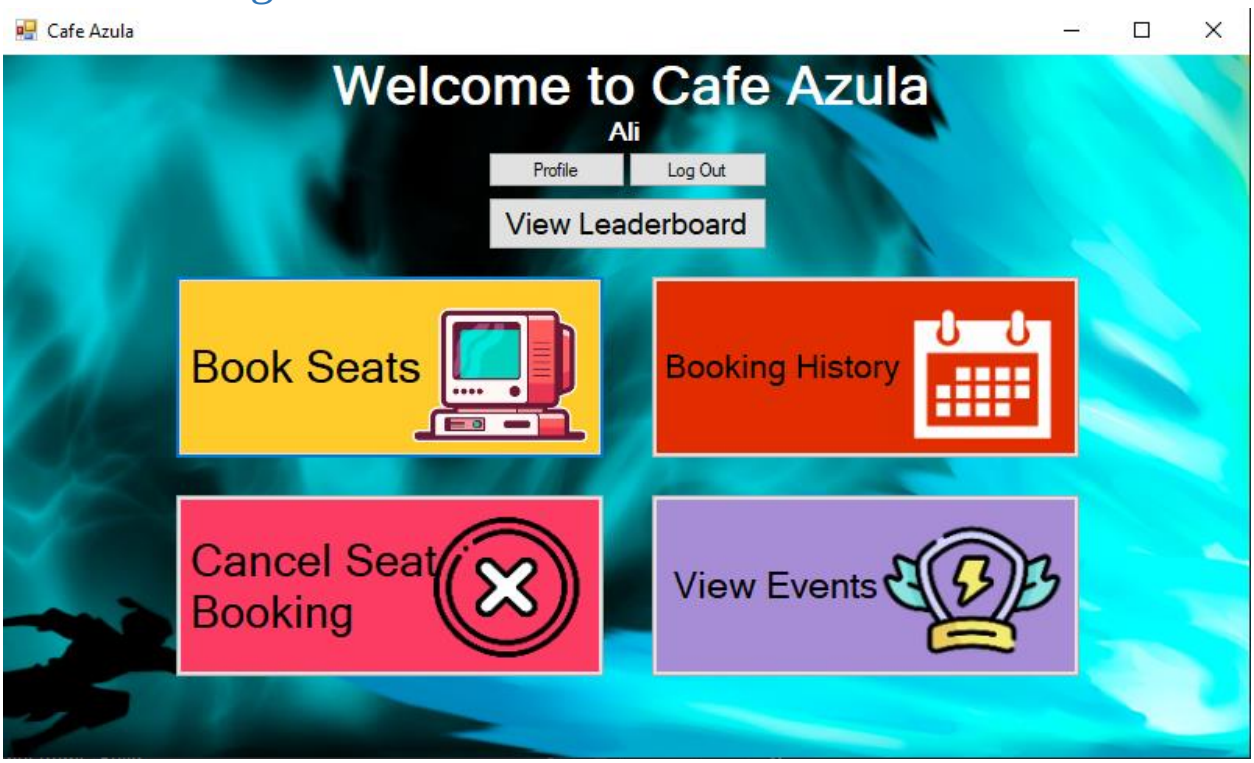
```
1.  string sql = "Delete from Computers where ComputerID = " + computerid;
2.  cmd = new SqlCommand(sql, cnn);
3.  cmd.ExecuteNonQuery();
```

And many more…

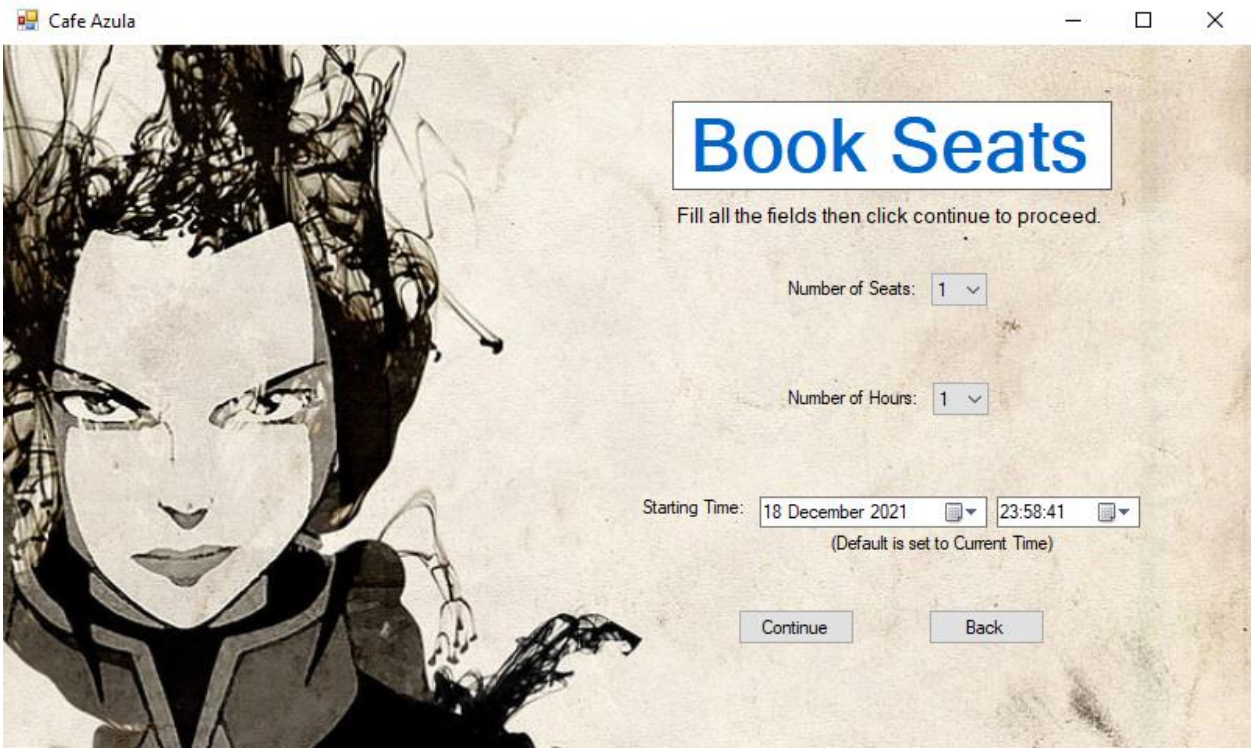
Visuals:

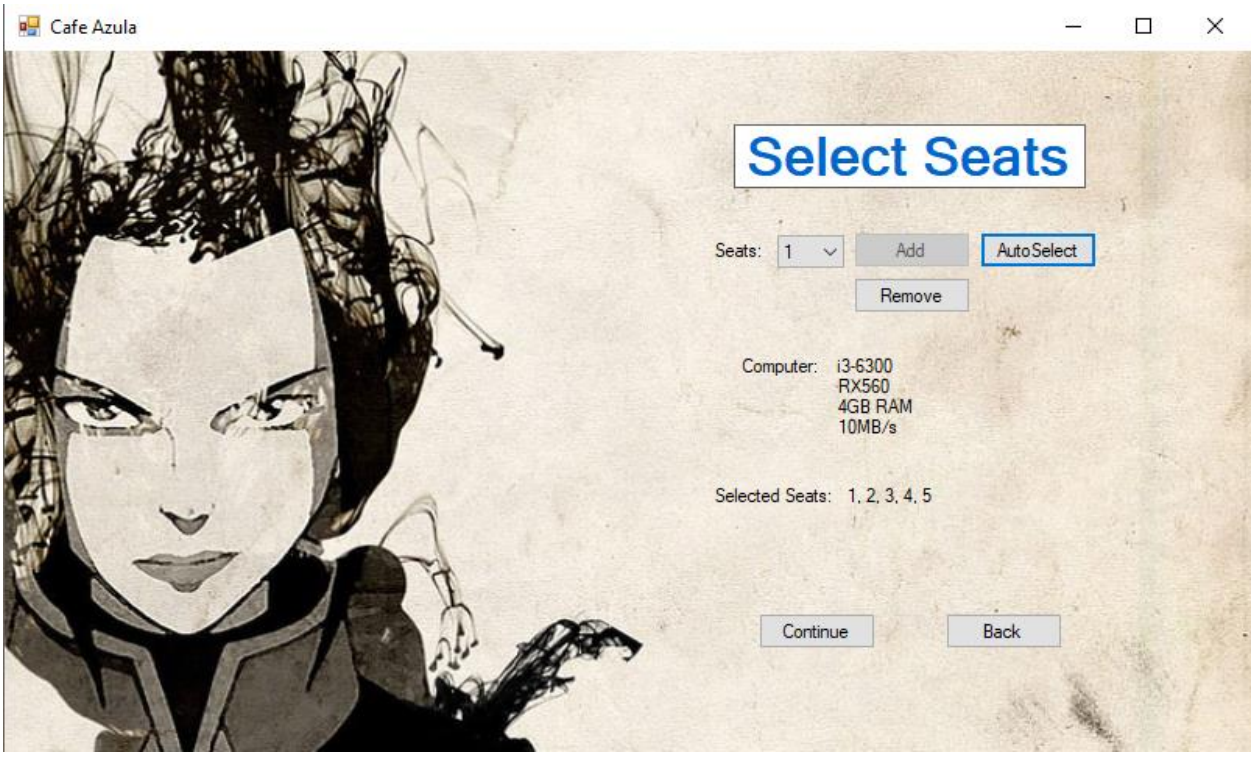
Login Page:

Customer Page:

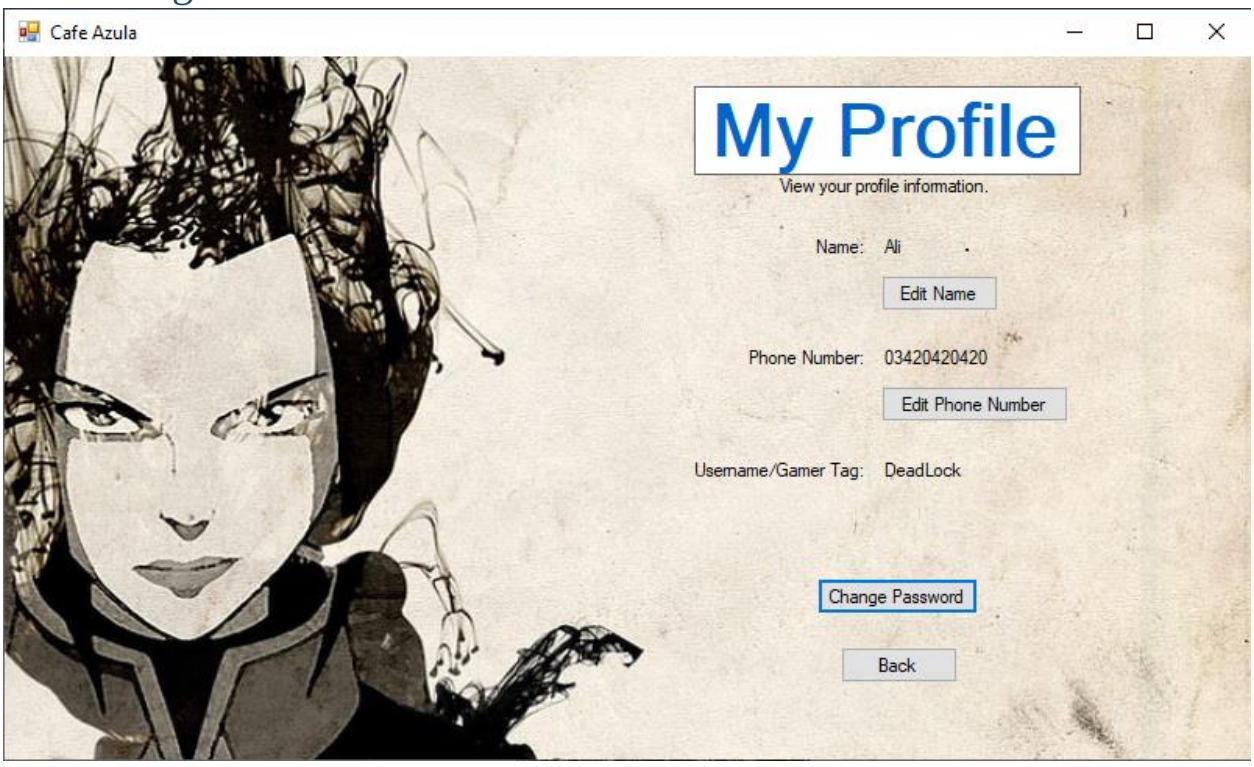


Book Seats:

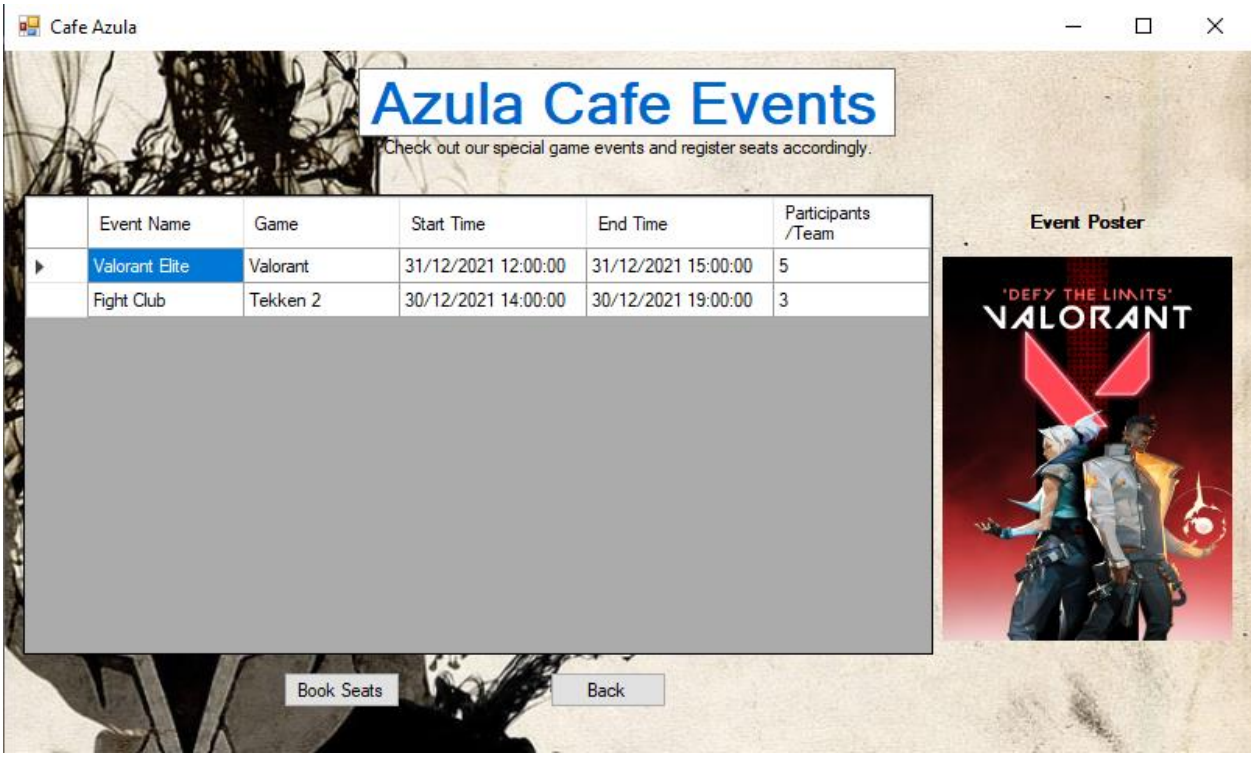


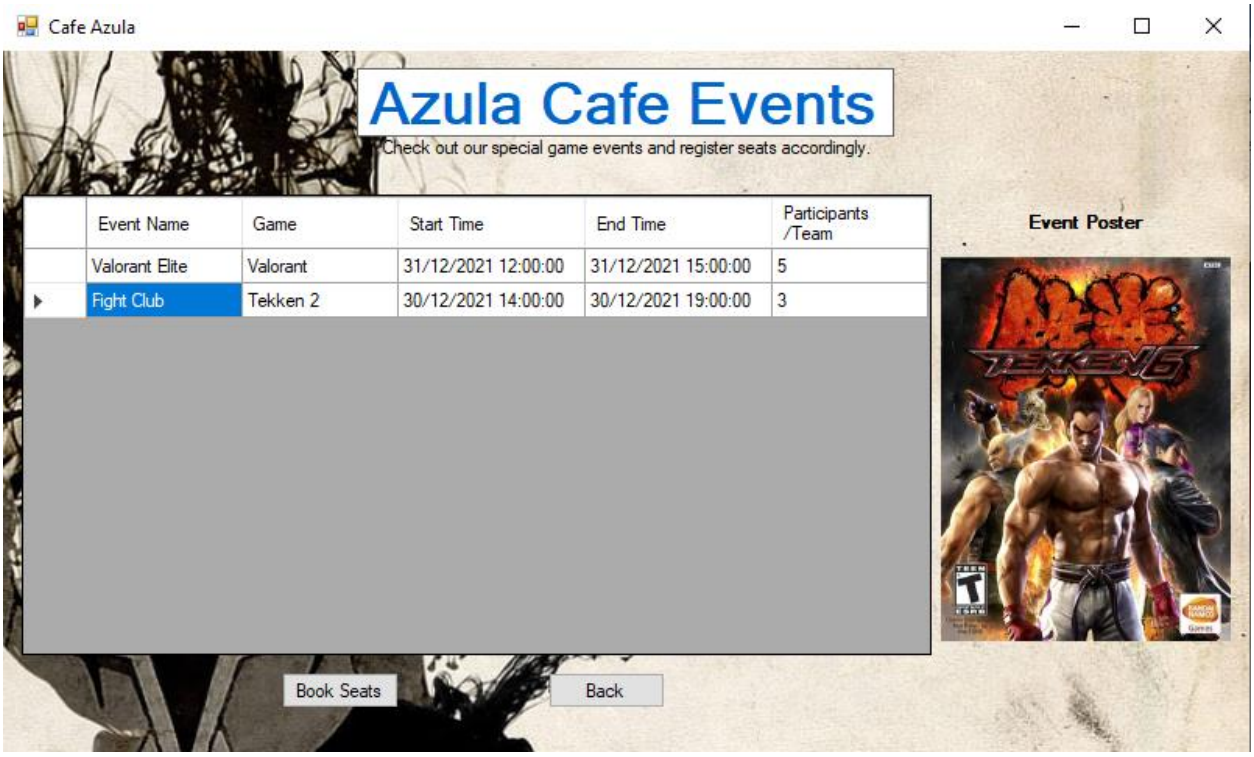


Profile Page:

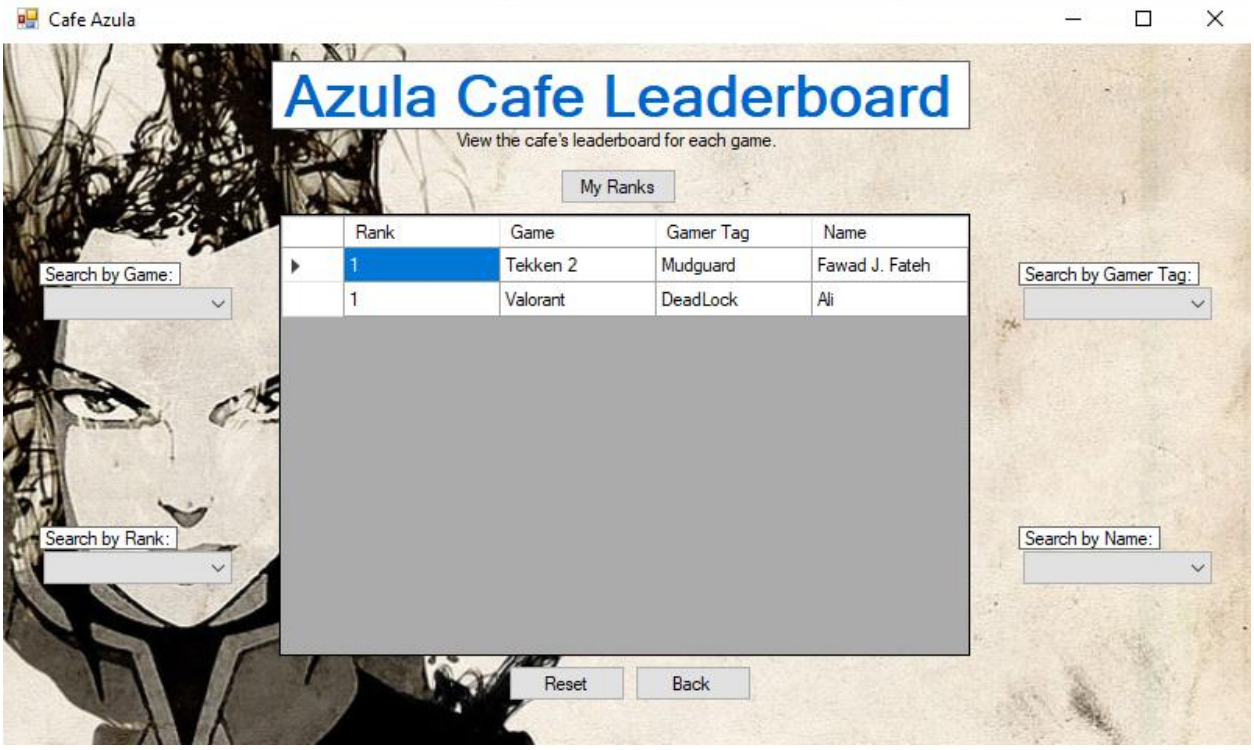


View Events:

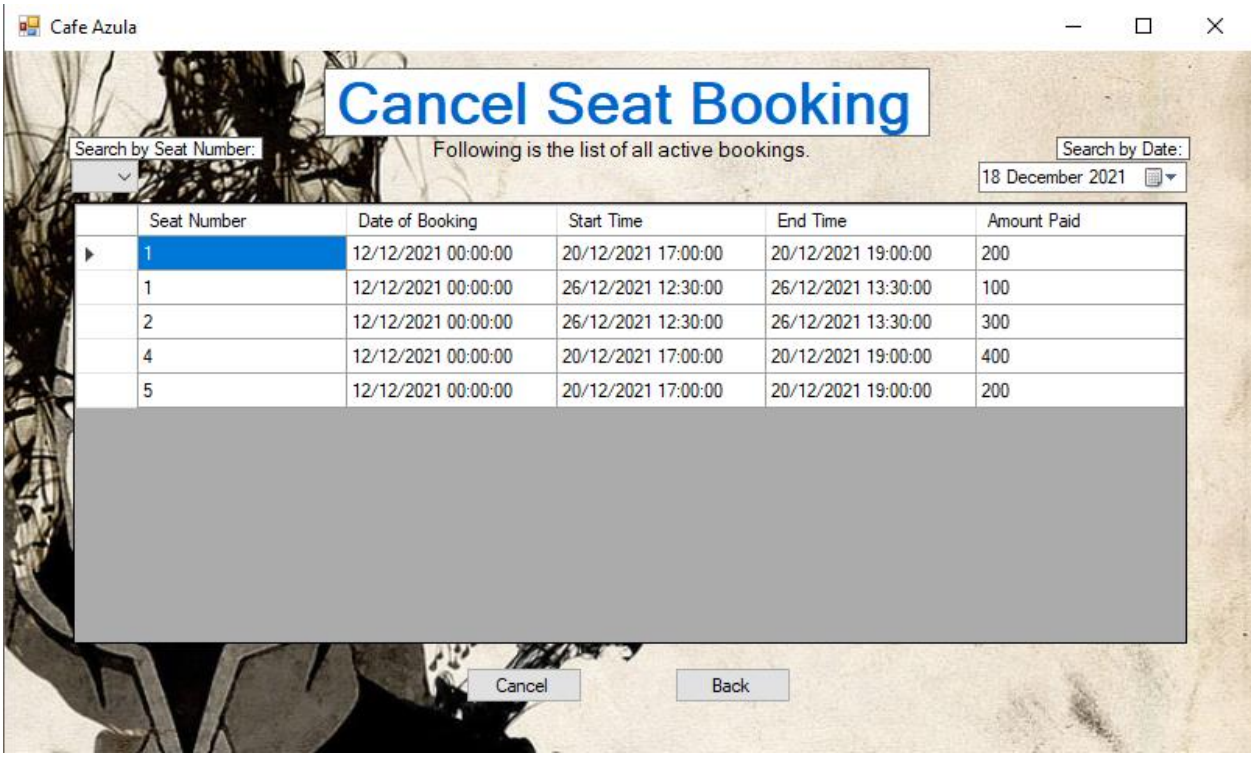

Cafe Azula
Azula Cafe Events
Check out our special game events and register seats accordingly.
Event Name
Game
Start Time
End Time
Participants /Team
Valorant Elite
Valorant
31/12/2021 12:00:00
31/12/2021 15:00:00
5
Fight Club
Tekken 2
30/12/2021 14:00:00
30/12/2021 19:00:00
3
Event Poster
'DEFY THE LIMITS'
VALORANT
Book Seats
Back


Cafe Azula
Azula Cafe Events
Check out our special game events and register seats accordingly.
Event Name
Game
Start Time
End Time
Participants /Team
Valorant Elite
Valorant
31/12/2021 12:00:00
31/12/2021 15:00:00
5
Fight Club
Tekken 2
30/12/2021 14:00:00
30/12/2021 19:00:00
3
Event Poster
TEKKEN 8
Book Seats
Back

View Leaderboard:

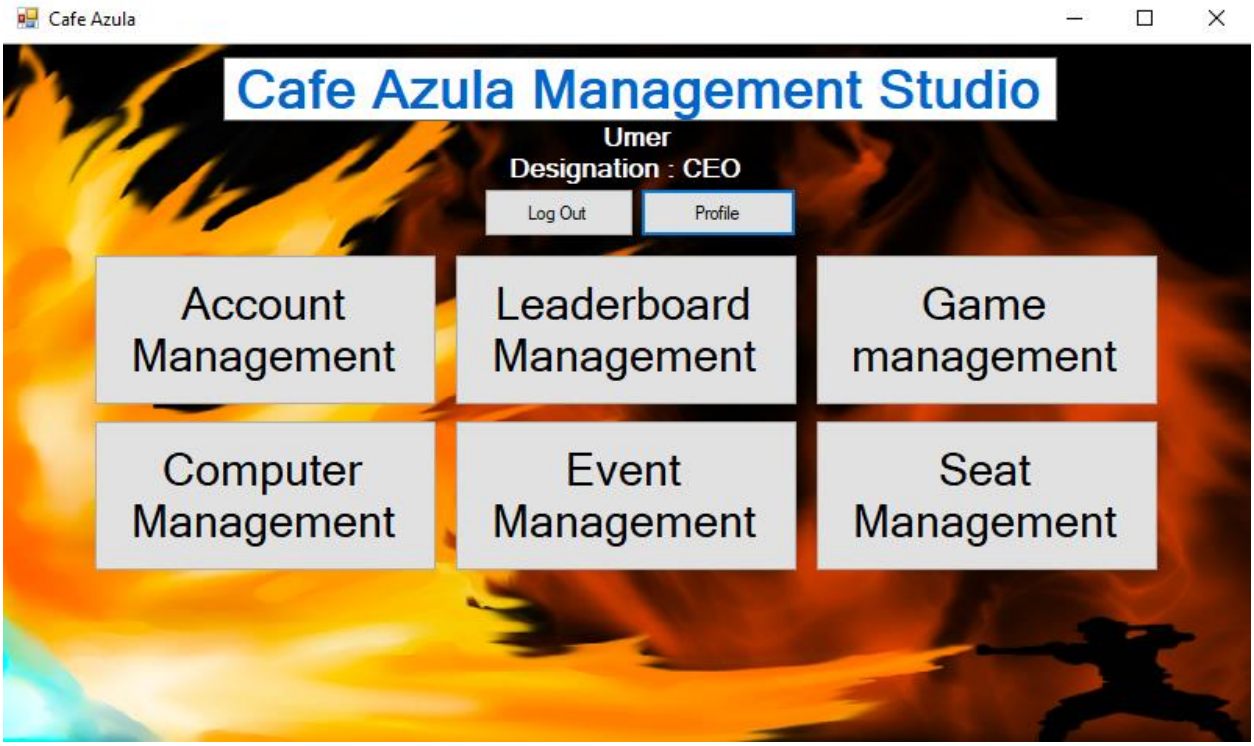


Cancel Seats:



Booking History:

Staff Page:


Cafe Azula
Cafe Azula Management Studio
Umer
Designation : CEO
Log Out
Profile
Account Management
Leaderboard Management
Game management
Computer Management
Event Management
Seat Management

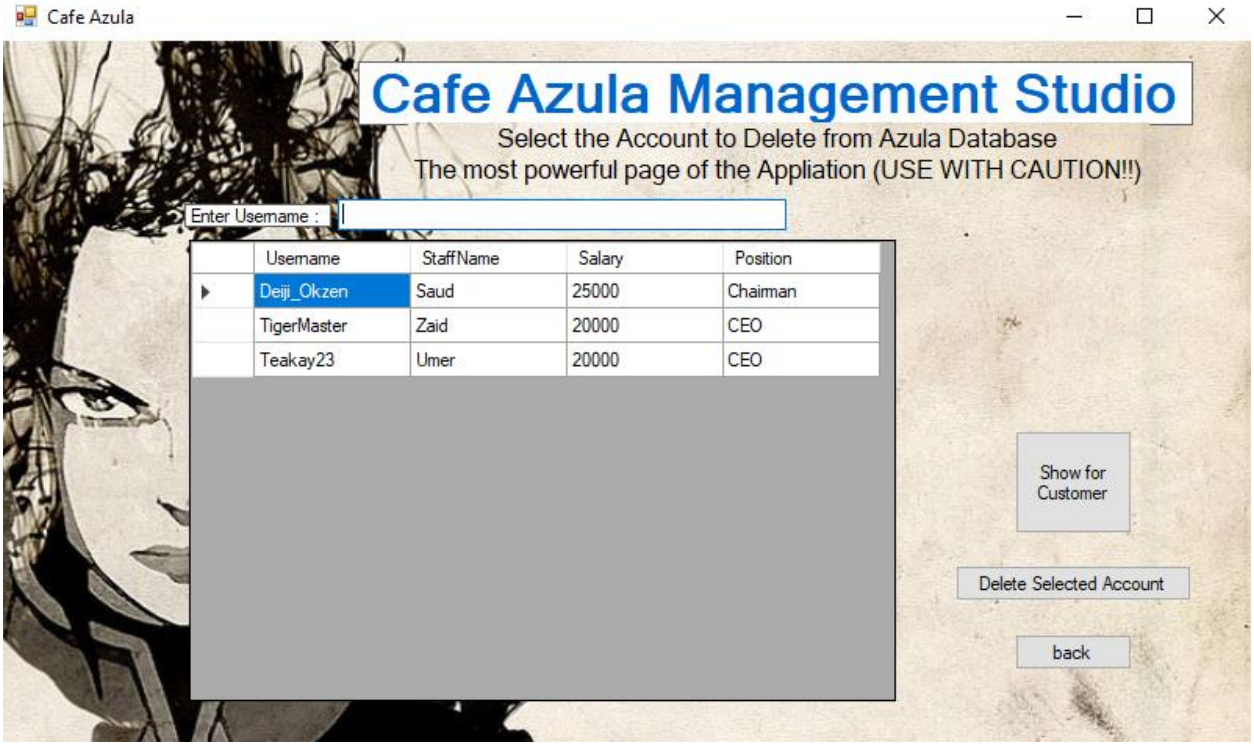

Cafe Azula
Cafe Azula Management Studio
Umer
Designation : CEO
Log Out
Profile
Account Management
Add
Delete
Leaderboard Management
Game management
Computer Management
Event Management
Seat Management
Add
Delete

Career Profile:

Account Management:

Add:

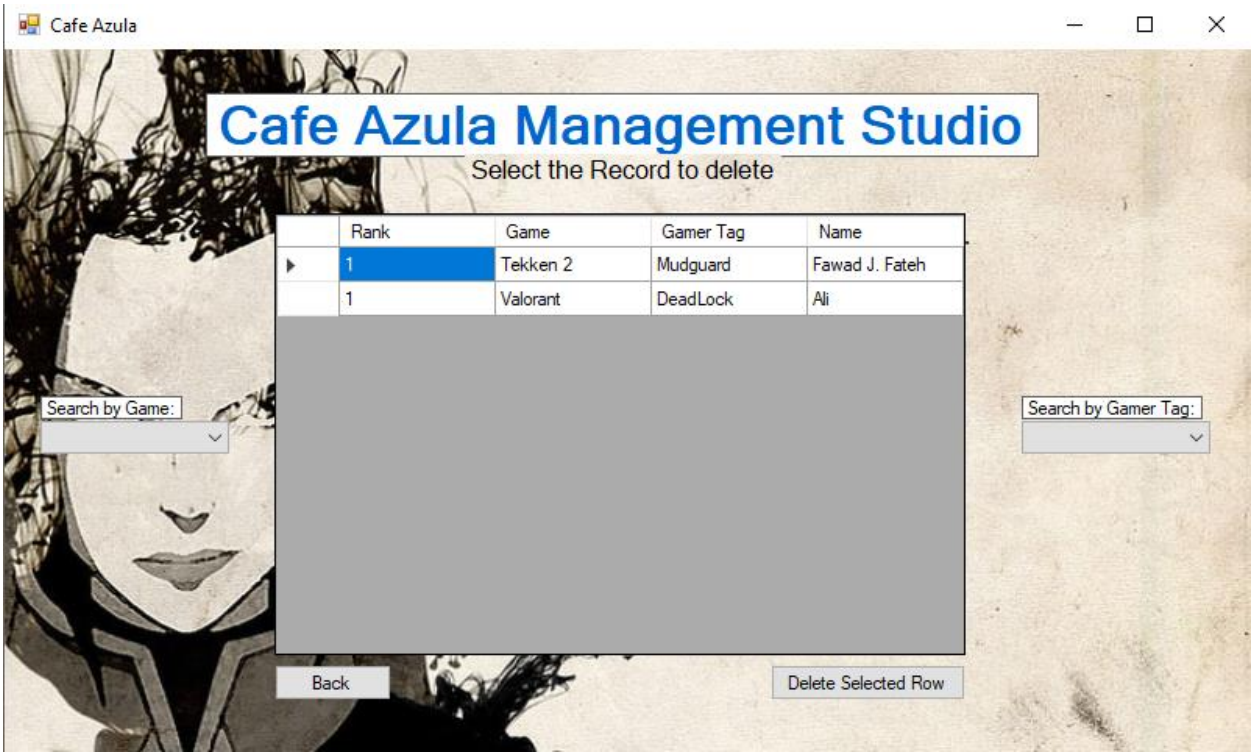
Delete:



Leaderboard Management:

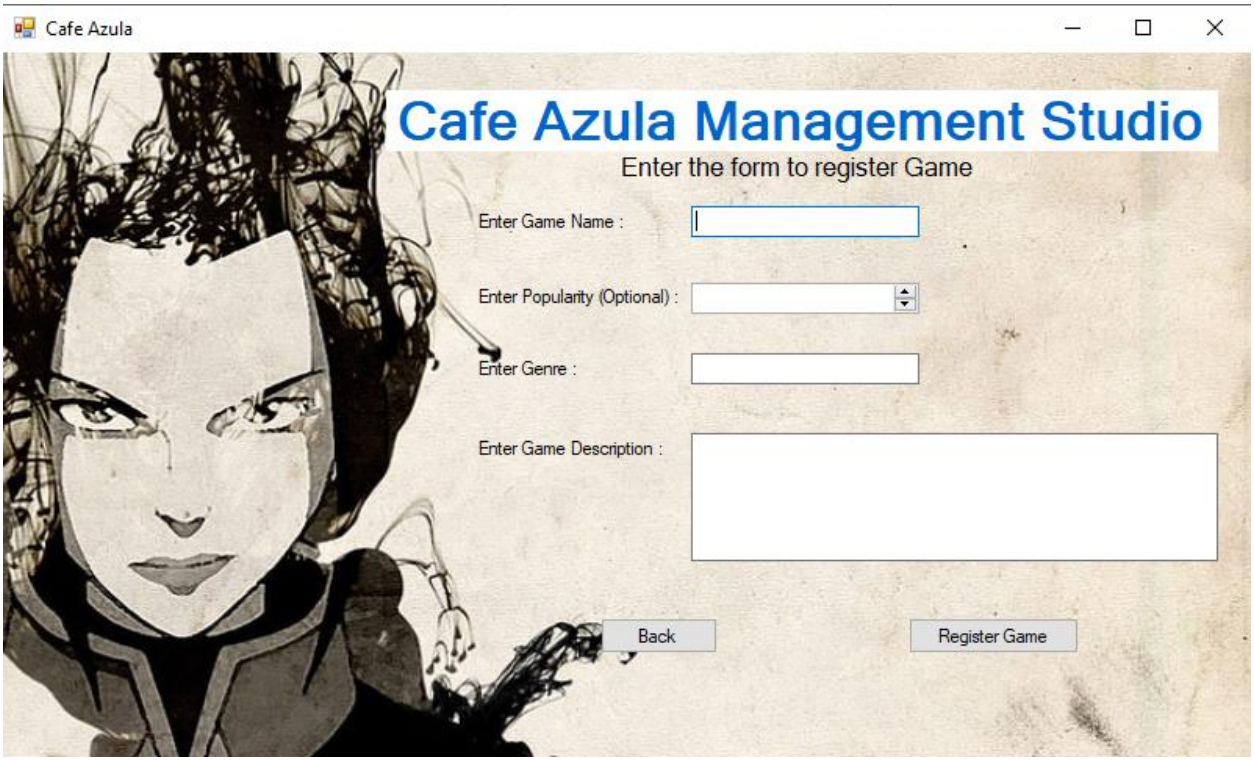
Add:

Delete:

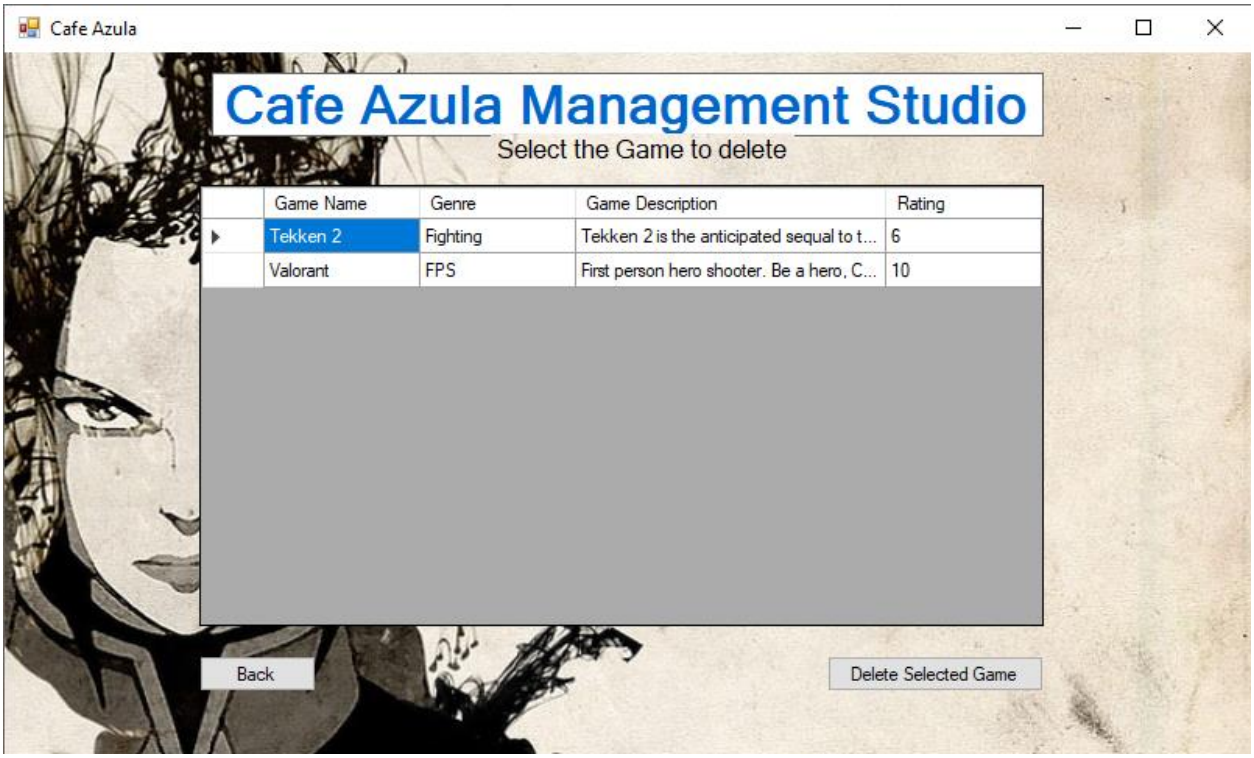


Game Management:

Add:

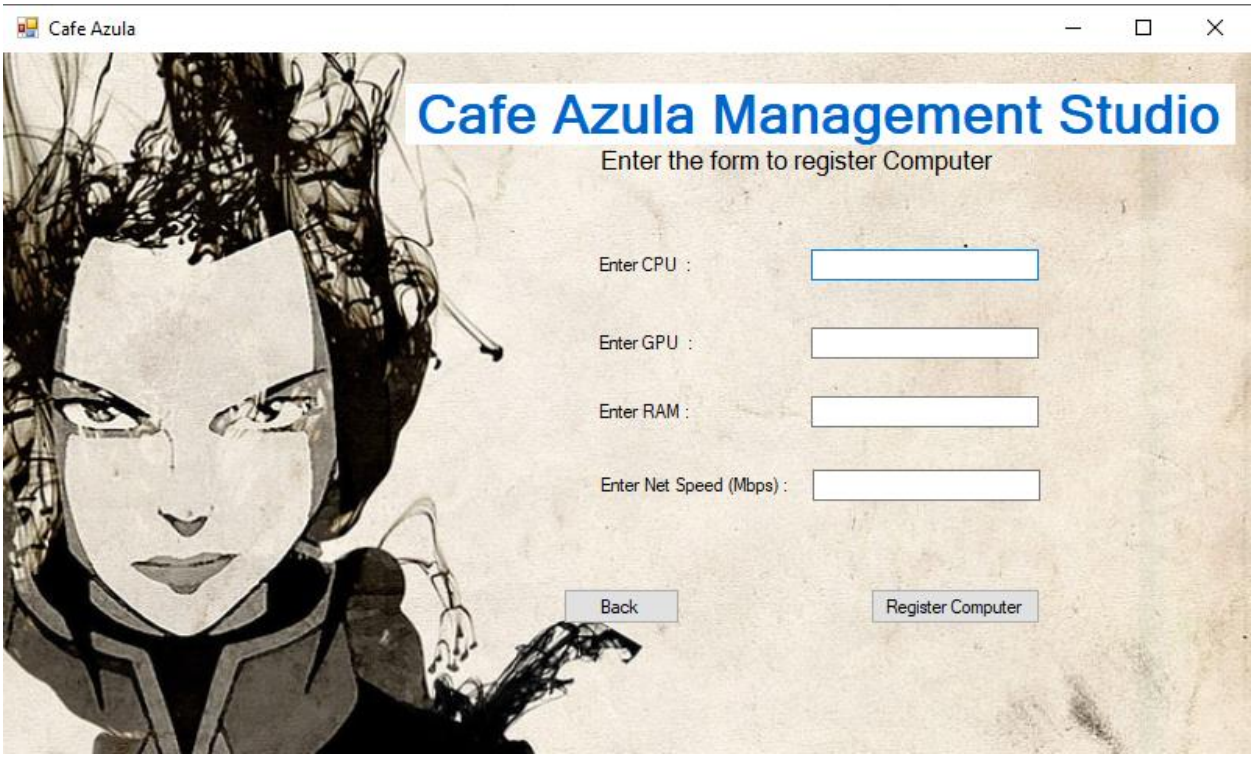


Delete:

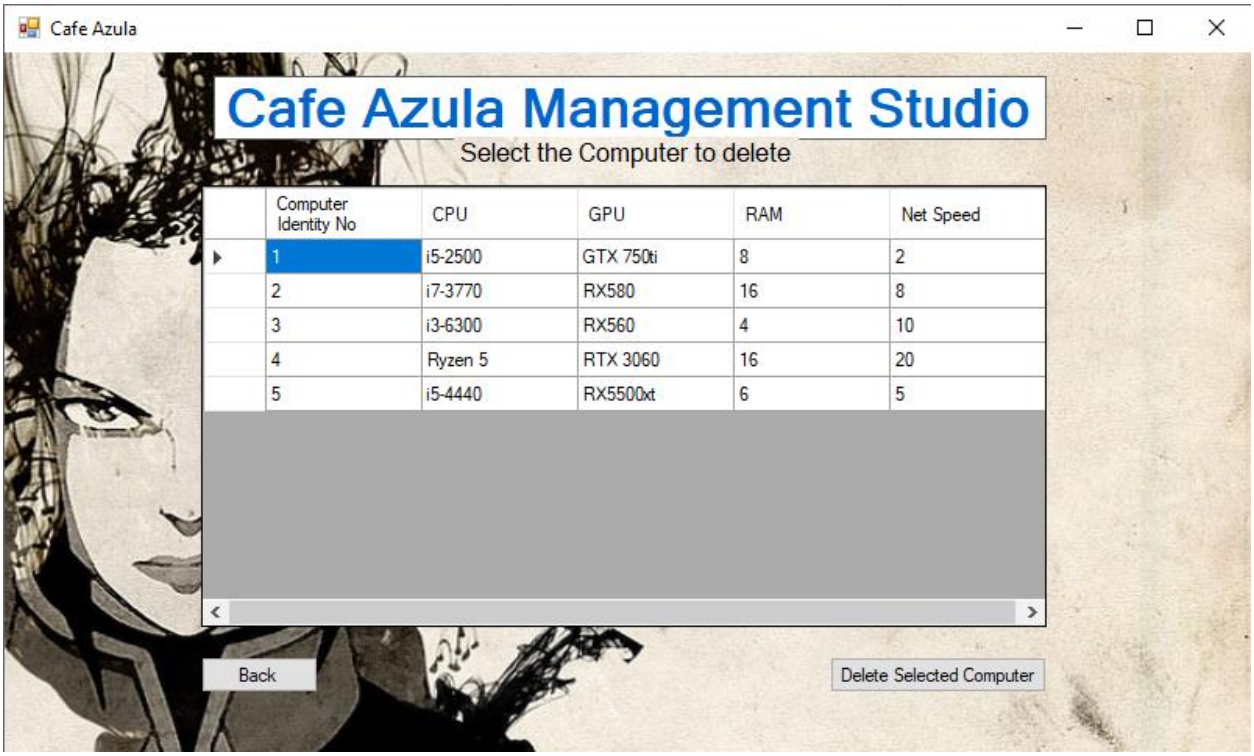


Computer Management:

Add:



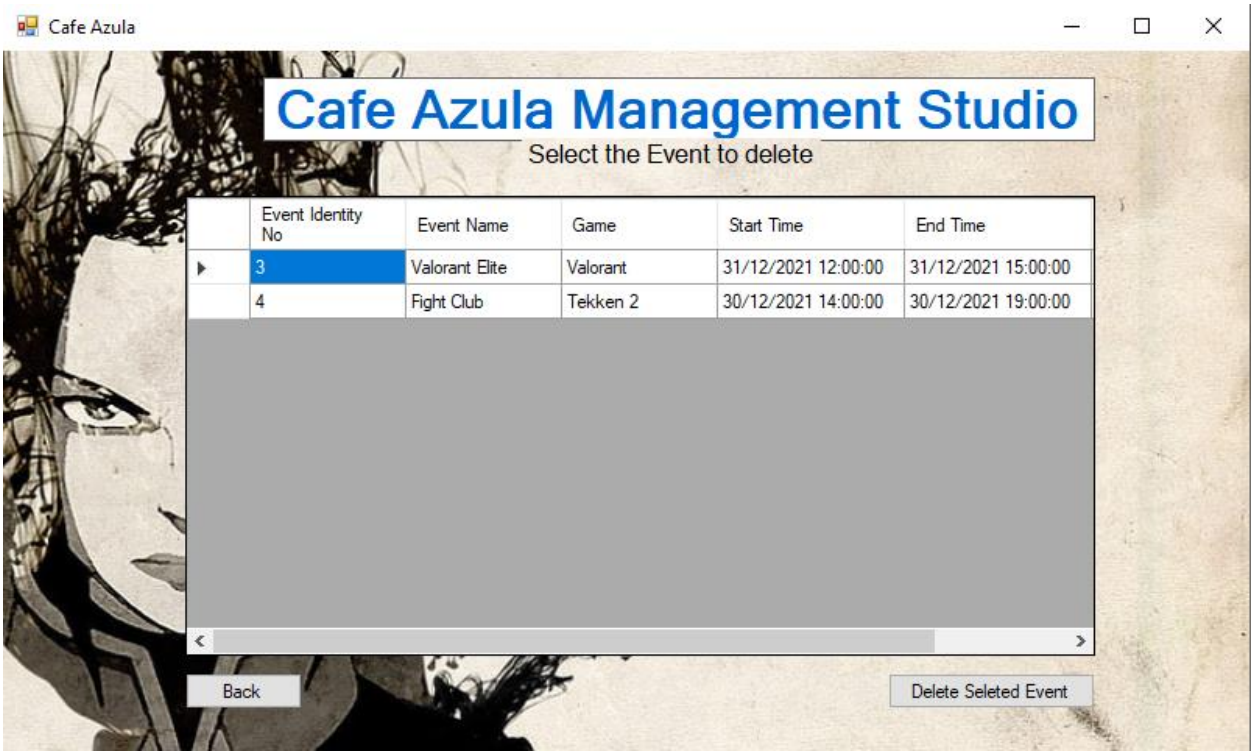
Delete:



Event Management:

Add:

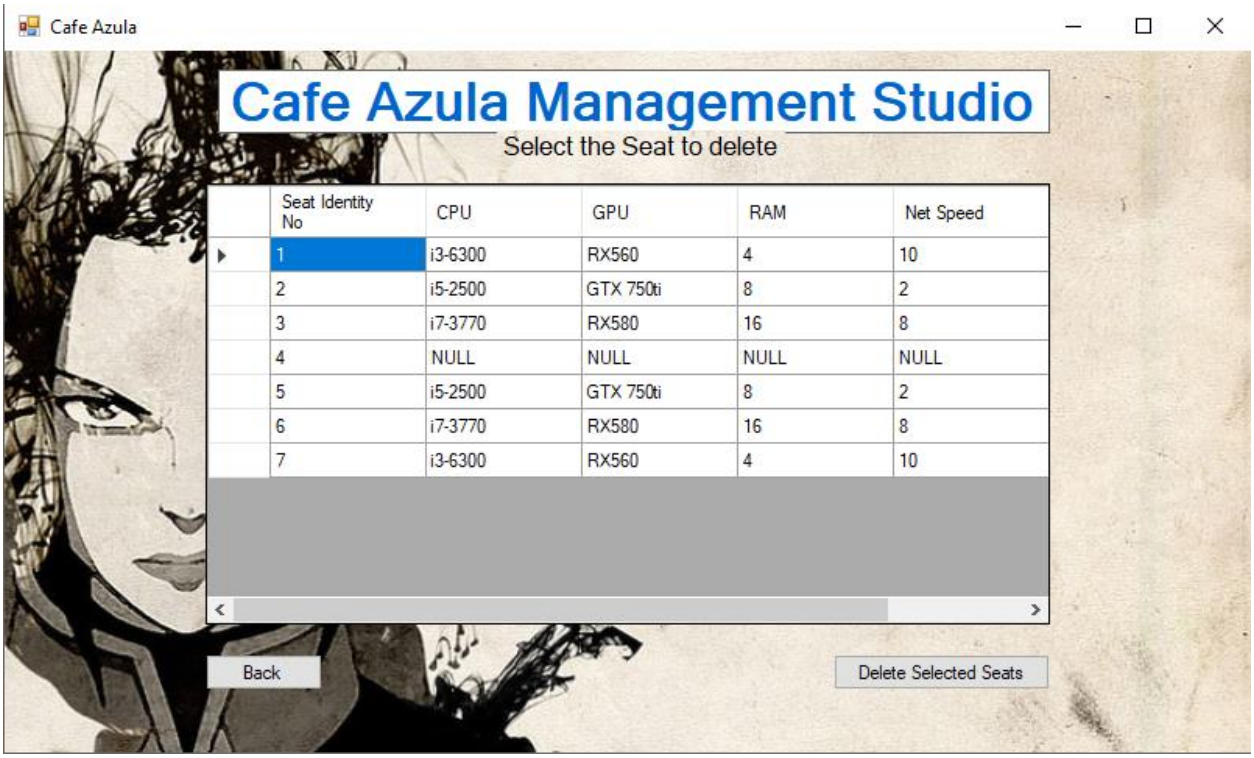
Delete:



Seat Management:

Add:

Delete:


Cafe Azula
Cafe Azula Management Studio
Select the Seat to delete
Seat Identity No
CPU
GPU
RAM
Net Speed
1 i3-6300 RX560 4 10
2 i5-2500 GTX 750ti 8 2
3 i7-3770 RX580 16 8
4 NULL NULL NULL NULL
5 i5-2500 GTX 750ti 8 2
6 i7-3770 RX580 16 8
7 i3-6300 RX560 4 10
Back
Delete Selected Seats