



AMERICAN INTERNATIONAL UNIVERSITY–BANGLADESH (AIUB)

Dept. of Computer Science
Faculty of Science and Technology

CSC2210: OBJECT ORIENTED PROGRAMMING 2

Summer 2024-2025

Section: [J]

Group:08

Project Report On

Project Name [RENTAL MANAGEMENT SYSTEM]

Supervised By

TOFAYET SULTAN

Submitted By:

Name	ID
Rakibul Islam Akash	23-50363-1
Md. Rakib Hossain	23-50379-1
Riana Ramzan	23-50383-1
Md. Golam Shahria Shakil	23-55475-3

Contents:

Chapter:1

Introduction (Chapter:1.1).....	3
---------------------------------	---

Chapter:2

ER Diagram.....	4
Features/User Details	5
Activity diagram.....	6
Sequence diagram	7

Chapter:3

From Images & Database Query.....	8-16
-----------------------------------	------

Chapter:4

Table of Database	17-21
-------------------------	-------

Chapter :5

Conclusion & Refrence & GitHub Link	22
---	----

Introduction

The Rental Management System is a .NET desktop app that helps tenants view and book vacant flats or rooms. It supports both family flat rent (full flat) and bachelor rent (room by room). Owners are categorized as Normal or Premium, with premium posts shown first. An Admin Panel allows efficient management of users and properties, making the rental process simple, transparent, and organized.

Problem Statement

Finding and managing rental flats/rooms is painful for both sides:

- Tenants can't see real-time availability (full flat vs. single room).
- Owners struggle to publish, prioritize, and maintain listings.
- Admins have no single place to enforce policy and resolve disputes.
-

Gap Covered

Most small apps treat a flat as a single unit. In reality, bachelor availability is room-level while family availability is flat-level. Prioritization for Premium Owners is also usually missing.

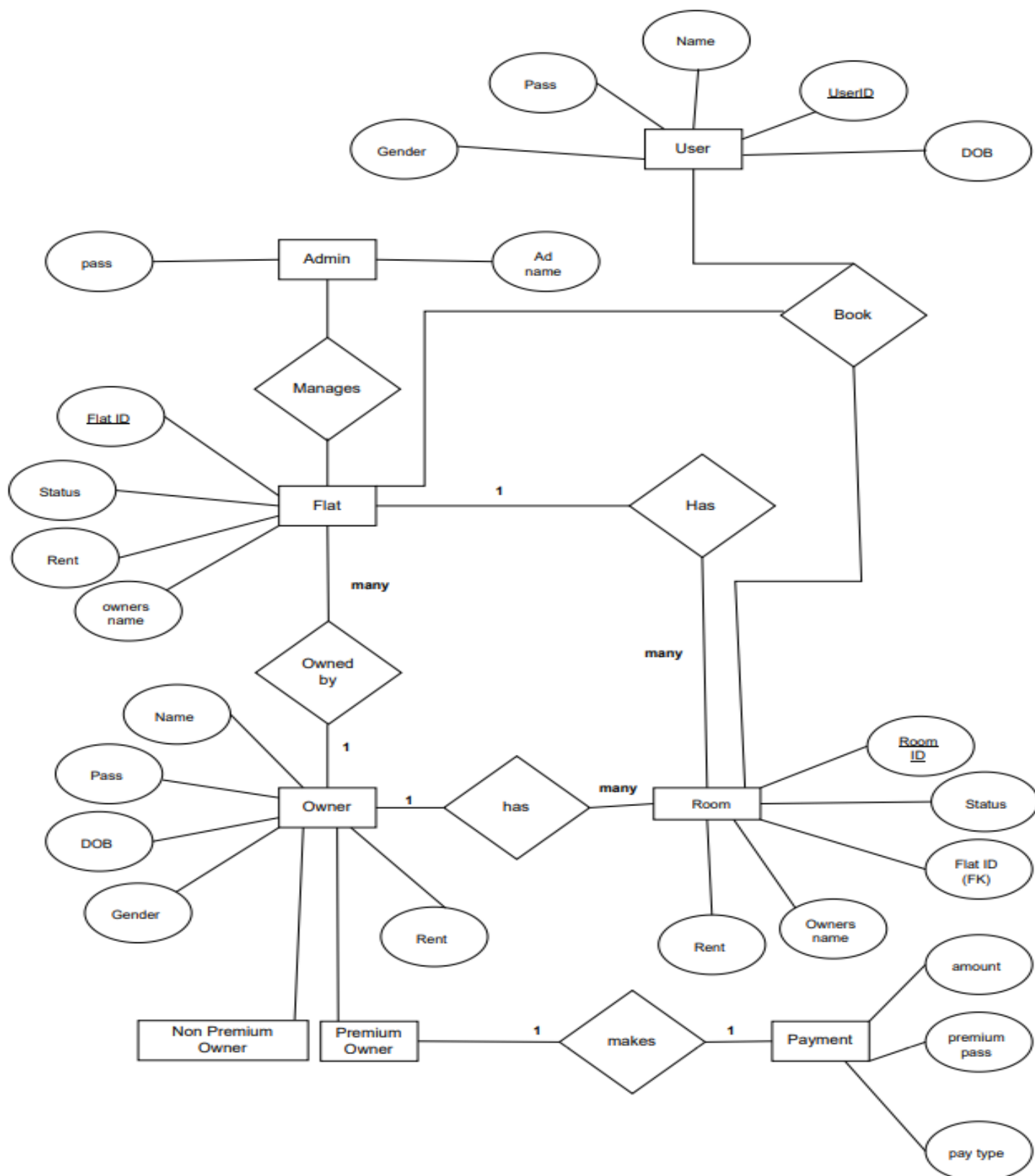
Proposed Solution

A desktop app built with .NET Framework (C#) Rental Management System serves that:

- Shows live availability of flats and rooms in each flat.
- Let's users book either a whole flat (family) or a single room (bachelor).
- Let's Owners (Normal/Premium) post and manage listings; Premium posts get priority.
- Provides an Admin panel to approve owners, moderate content, and audit bookings.
-

Objectives

- Realtime availability (flat vs. per-room).
- Fair booking with conflict prevention.
- Premium-first listing order.
- Simple, auditable data model.

ER Diagram:

Features / User Details:

Tenant

- Search by building, floor, budget.
- Two tabs: Flats (family) and Rooms (bachelor).
- Real-time badges: Available / Booked.
- Book; see history; cancel pending bookings.

Owner (Normal)

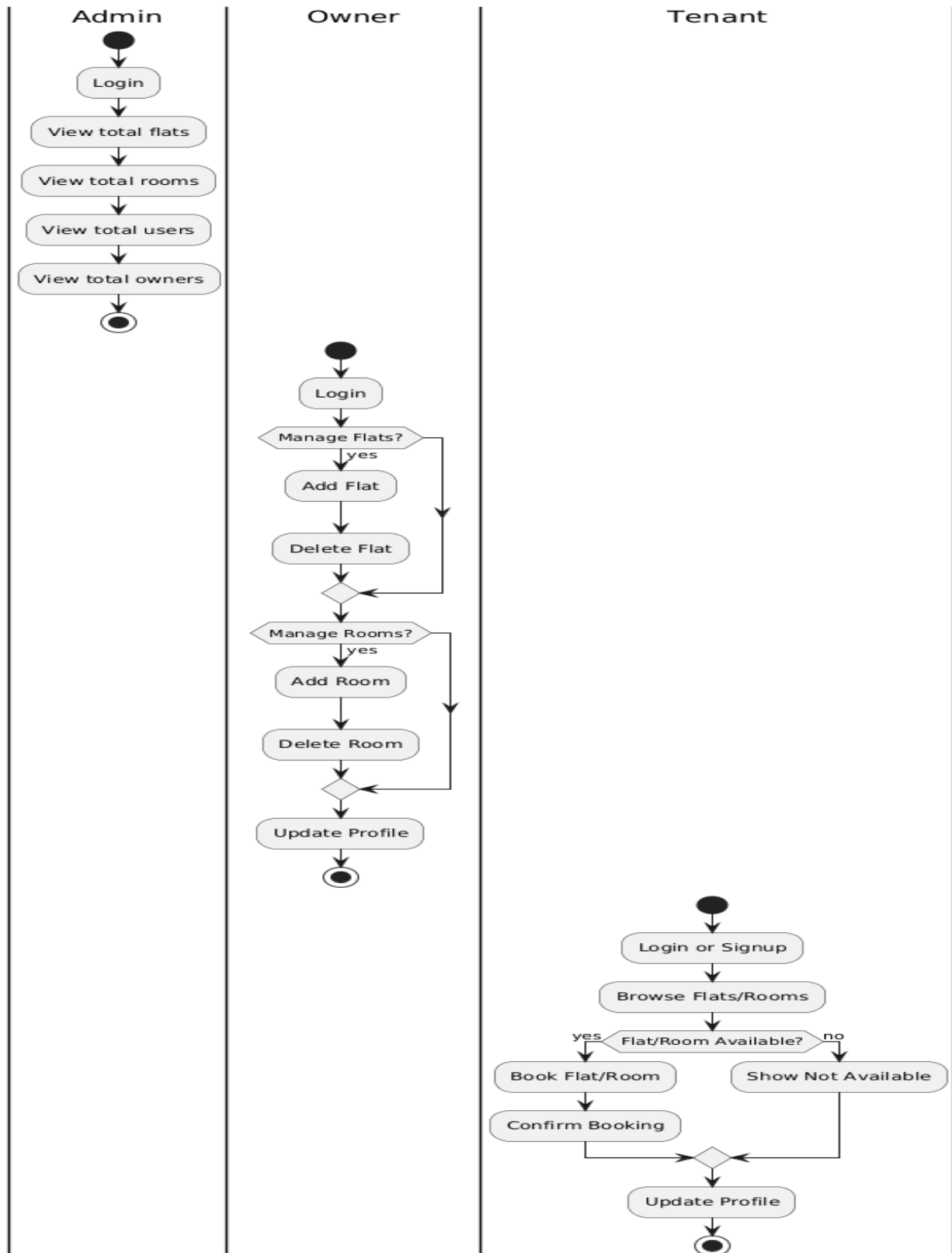
- Register flats and (optionally) rooms per flat.
- Create one active listing per flat (family or bachelor).
- See inquiries & booking requests.

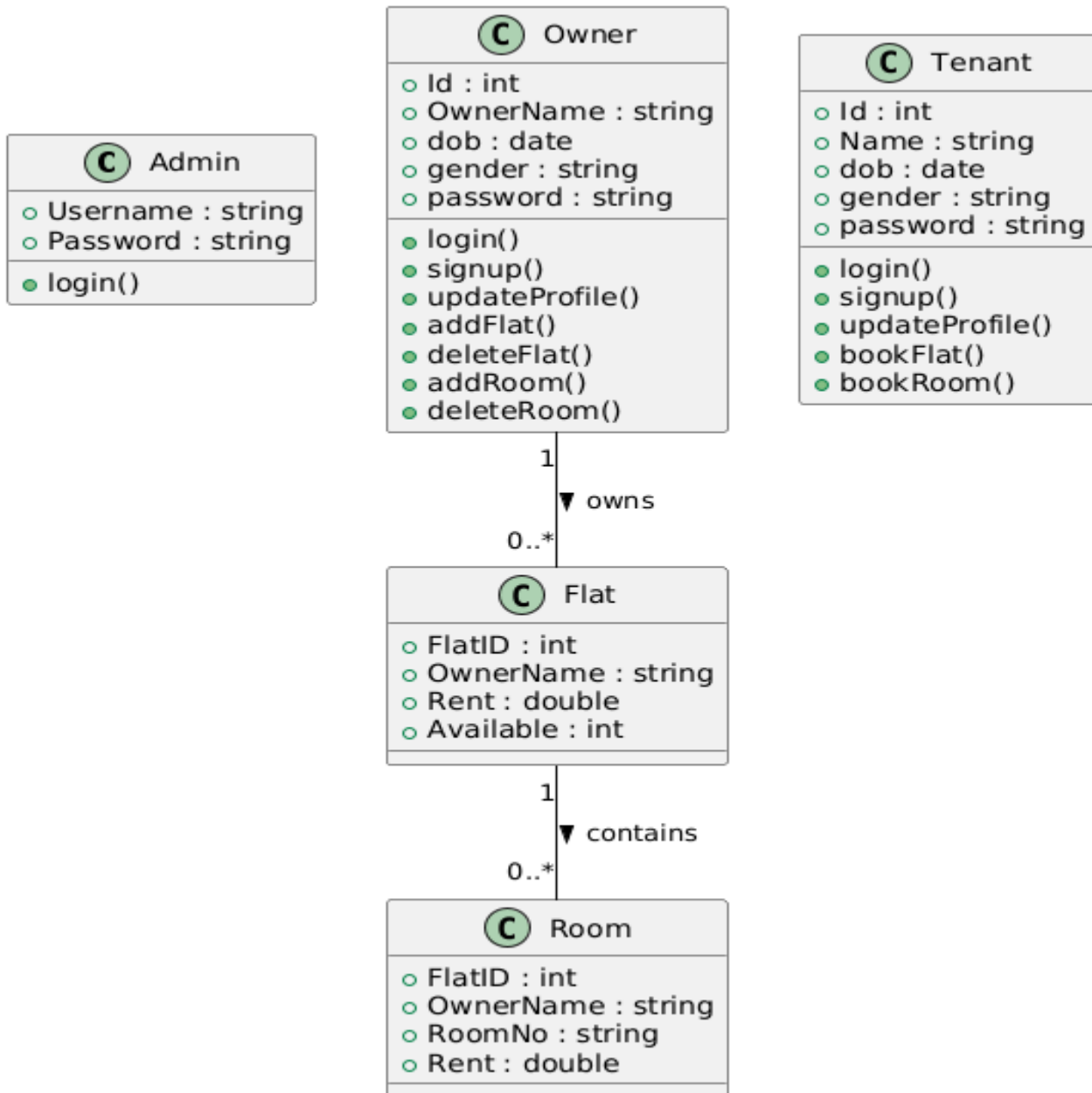
Owner (Premium)

- All Normal features + priority placement.
- Highlighted listing style (icon/badge) in results.

Admin

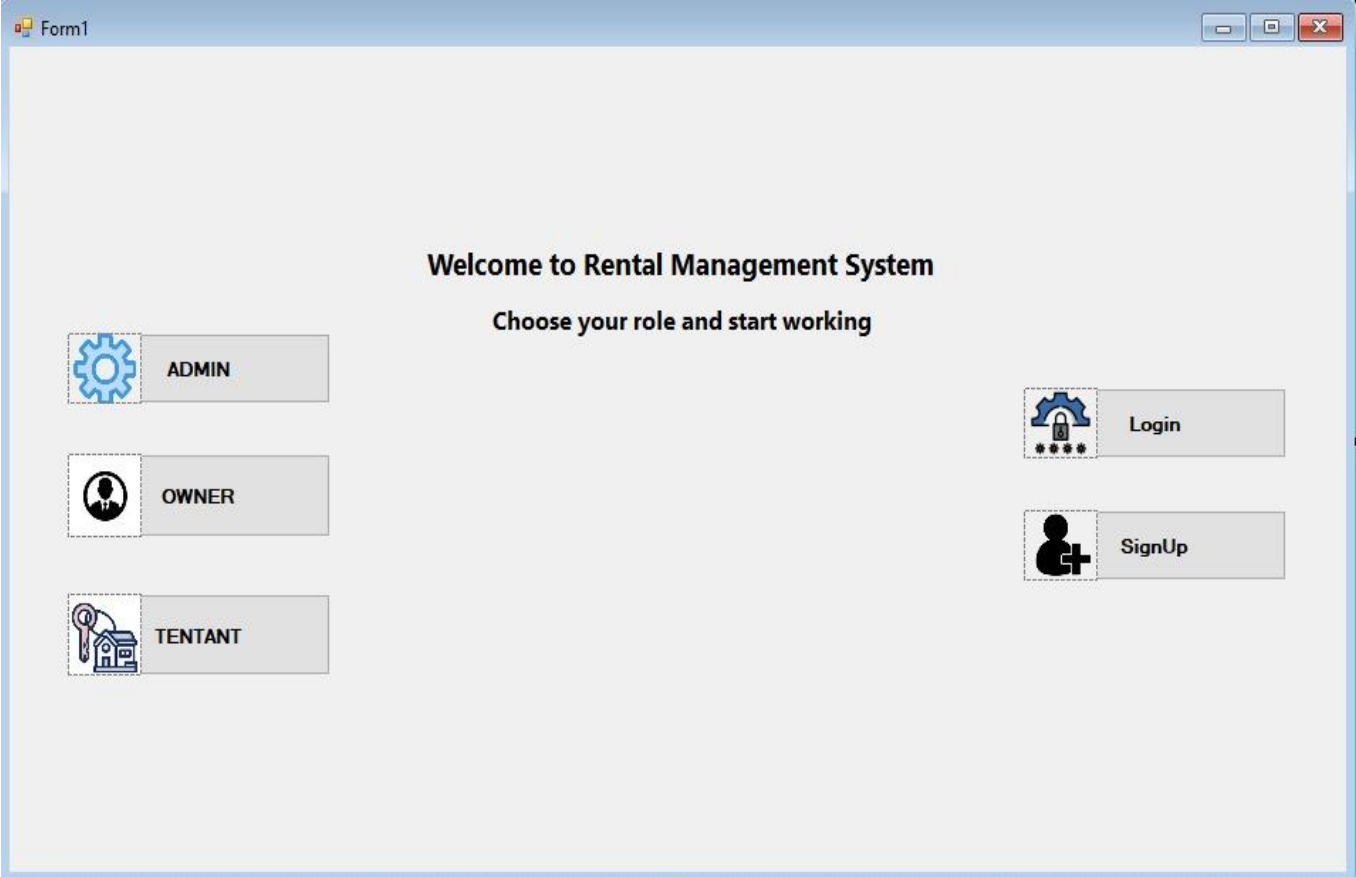
- Verify & activate Owner accounts.
- Deactivate abusive listings.
- View system logs & payments.

ACTIVITY DIAGRAM:

Sequence Diagram:

Form & Query

Form1 Form:



The screenshot shows a window titled "Form1" with standard Windows window controls (minimize, maximize, close) in the top right corner. The main content area has a light gray background. In the center, the text "Welcome to Rental Management System" is displayed in a bold, black font, with the instruction "Choose your role and start working" below it. On the left side, there are three buttons stacked vertically. Each button consists of a small square icon on the left and a rectangular text box on the right. The first button has a blue gear icon and the text "ADMIN". The second button has a black silhouette of a person's head and shoulders and the text "OWNER". The third button has a blue icon of a key and a house and the text "TENTANT". On the right side, there are two buttons stacked vertically. The first button has a blue gear icon with a padlock and four asterisks below it, and the text "Login". The second button has a black silhouette of a person with a plus sign and the text "SignUp".

Form1

Welcome to Rental Management System

Choose your role and start working

ADMIN

OWNER

TENTANT

Login

SignUp

Login Query:

```
query = "SELECT * FROM [owner] WHERE name=@name AND password=@password AND  
premiumPass=@premiumPass";
```

```
query = "SELECT * FROM [owner] WHERE name=@name AND password=@password";
```

```
query= "SELECT * FROM [user] WHERE name=@name AND password=@password";
```

Login Form:

Back

Name

Password

Role

Premium Password

Login

Sign Up Query:

```
string sql = @"INSERT INTO [user]
    (Id, name, dob, gender, password) VALUES (Id, @name, @dob, @gender, @password)";

string sql = @"INSERT INTO [owner]
    (Id, name, dob, gender, password, premium, normal, premiumPass)
    VALUES (@Id, @name, @dob, @gender, @password, @premium, @normal, @premiumPass)";
```

Sign Up Form:

SignUp

Back

ID

Name

DOB

Gender

Password

Confirm Password

☐ Normal ☐ Premium

Payment

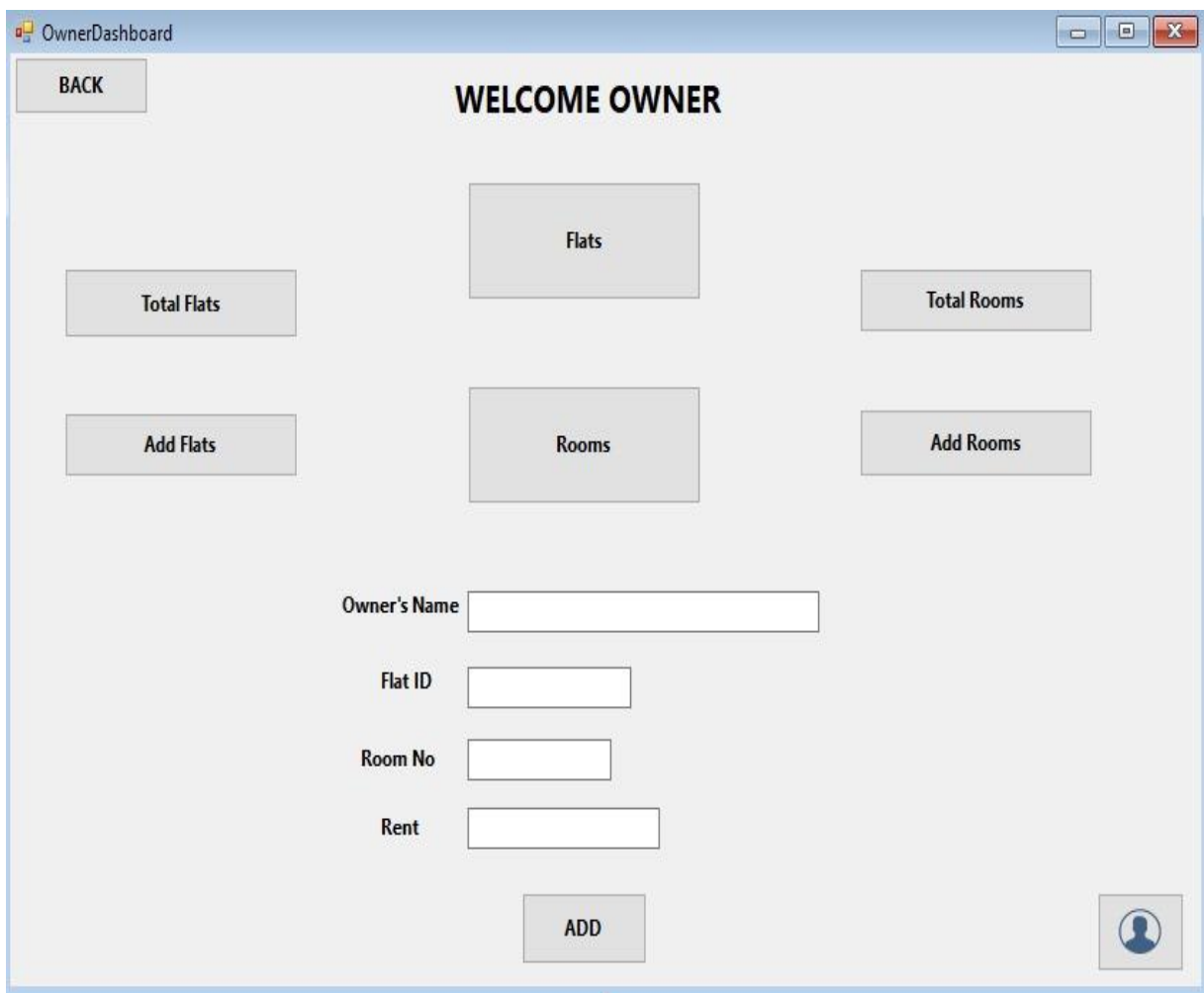
SignUp

Owner Query:

```
string queryRoom = "INSERT INTO Room(OwnerName, FlatID, RoomNo, Rent, Available)  
VALUES(@OwnerName, @FlatID, @RoomNo, @Rent, 1)";
```

```
string queryFlat = "INSERT INTO Flat(OwnerName, FlatID, Rent, Available)  
VALUES(@OwnerName, @FlatID, @Rent, 1)";
```

OwnerDashboard Form:

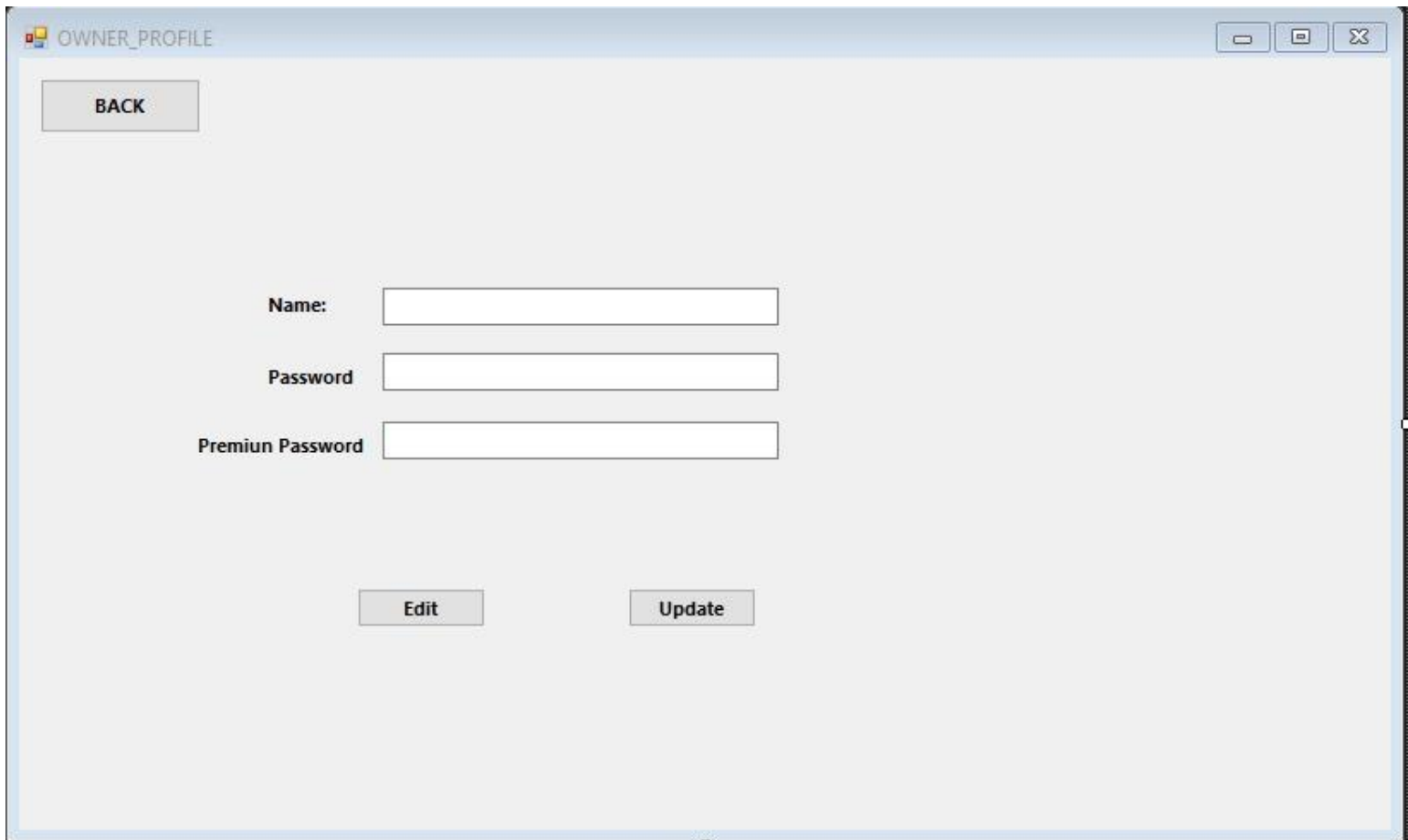


The screenshot shows a Windows-style application window titled "OwnerDashboard". The interface is light gray with a central "WELCOME OWNER" heading. A "BACK" button is in the top left. The main area contains six buttons arranged in a 2x3 grid: "Total Flats", "Flats", "Total Rooms", "Add Flats", "Rooms", and "Add Rooms". Below these is a form with four labels and text boxes: "Owner's Name", "Flat ID", "Room No", and "Rent". An "ADD" button is centered below the form. In the bottom right corner, there is a small square button with a user icon.

Owner Profile Query:

```
string query = "SELECT Id, name, password FROM owner WHERE Id = @Id";  
string query = "UPDATE owner SET name = @name, password=@password WHERE Id = @Id";
```

OwnerProfile Form:



The screenshot shows a Windows-style application window titled "OWNER_PROFILE". The window has a standard title bar with minimize, maximize, and close buttons. Inside the window, there is a "BACK" button in the top-left corner. Below it, there are three input fields with labels: "Name:", "Password", and "Premiun Password" (note the typo). At the bottom of the form, there are two buttons: "Edit" and "Update".

OWNER_PROFILE

BACK

Name:

Password

Premiun Password

Edit Update

Owner Flat Show Query:

```

string query = @"SELECT f.OwnerName, f.FlatID, f.Rent, f.Available
FROM Flat f
    INNER JOIN owner s ON f.OwnerName = s.name
WHERE f.OwnerName = @OwnerName
    ORDER BY CASE WHEN s.premium = 'Yes' THEN 0 ELSE 1 END";

string query = @"SELECT f.OwnerName, f.FlatID, f.Rent, f.RoomNo, f.Available
FROM Room f
    INNER JOIN owner s ON f.OwnerName = s.name
WHERE f.OwnerName = @OwnerName
    ORDER BY CASE WHEN s.premium = 'Yes' THEN 0 ELSE 1 END";

string searchQuery = @"
    SELECT *
    FROM Flat
    WHERE
        OwnerName LIKE @searchTerm
        OR FlatID LIKE @searchTerm
        OR CAST(Rent AS NVARCHAR) LIKE @searchTerm";

string query = "DELETE FROM Flat WHERE FlatID = @FlatID";

string query = "DELETE FROM Room WHERE FlatID = @FlatID";

```

OwnerFlatShow Form:

OWNERflats_RommsShow

Back

TOTAL FLATS AND ROOMS

Search

Delete

Admin Query:

```
string searchQuery = @"SELECT * FROM Flat
WHERE
    OwnerName LIKE @searchTerm
    OR FlatID LIKE @searchTerm
    OR CAST(Rent AS NVARCHAR) LIKE @searchTerm";
```

```
query = "DELETE FROM [dbo].[Flat] WHERE FlatID=@Id";
query = "DELETE FROM [dbo].[Room] WHERE RoomNo=@Id";
query = "DELETE FROM [dbo].[owner] WHERE Id=@Id";
query = "DELETE FROM [dbo].[user] WHERE Id=@Id";
```

```
string query = "SELECT f.OwnerName,f.FlatID,f.Rent,f.Available FROM Flat f INNER JOIN
owner s ON f.OwnerName = s.name ORDER BY CASE WHEN s.premium = 'Yes' THEN 0 ELSE 1 END;";
```

```
string query = "SELECT f.OwnerName,f.FlatID,f.Rent,f.RoomNo, f.Available FROM Room f INNER
JOIN owner s ON f.OwnerName = s.name ORDER BY CASE WHEN s.premium = 'Yes' THEN 0 ELSE 1 END;";
```

```
string query = "SELECT * from [owner]";
```

```
string query = "SELECT * from [user]";
```

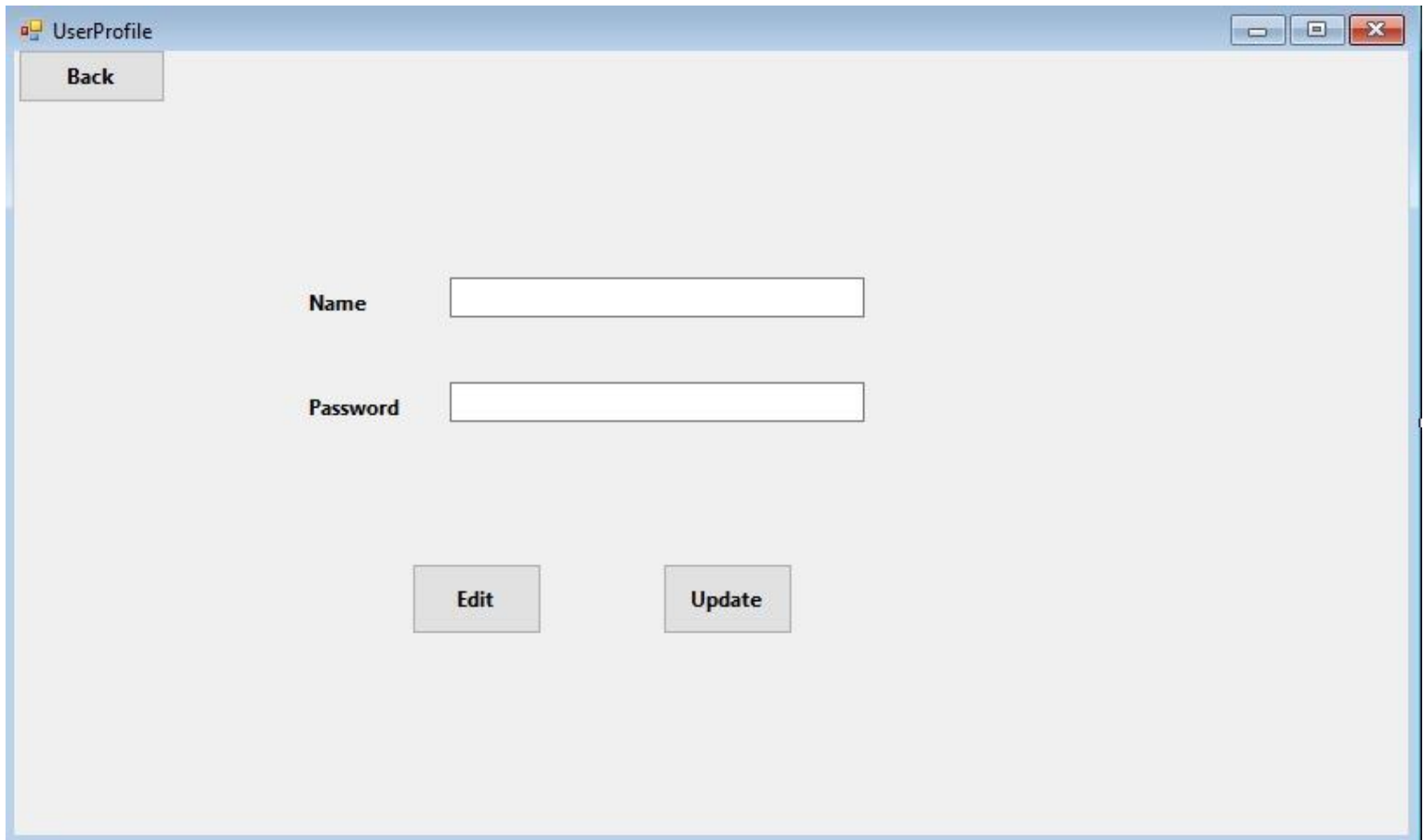
AdminDashboard Form:

The screenshot shows a web application window titled "ADMIN_DASHBOARD". The main content area has a light gray background. At the top, there's a "WELCOME ADMIN" message. Below it, four buttons are arranged horizontally: "Total_Flats", "Total_Rooms", "Total_OWNERS", and "Total_USERS". A large, semi-transparent gray rectangular area is positioned on the left side of the dashboard. On the right side, there is a "Search" button next to a text input field. Below the search field are two buttons: "SELECT" and "DELETE".

User Profile Query:

```
string query = "SELECT Id, name, password FROM [user] WHERE Id = @Id";  
string query = "UPDATE [user] SET name = @name, password=@password WHERE Id = @Id";
```

UserProfile Form:



The screenshot shows a Windows-style application window titled "UserProfile". In the top-left corner, there is a "Back" button. The main area of the window contains two text input fields. The first field is labeled "Name" and the second is labeled "Password". Below these fields, there are two buttons: "Edit" and "Update". The window has standard Windows window controls (minimize, maximize, close) in the top-right corner.

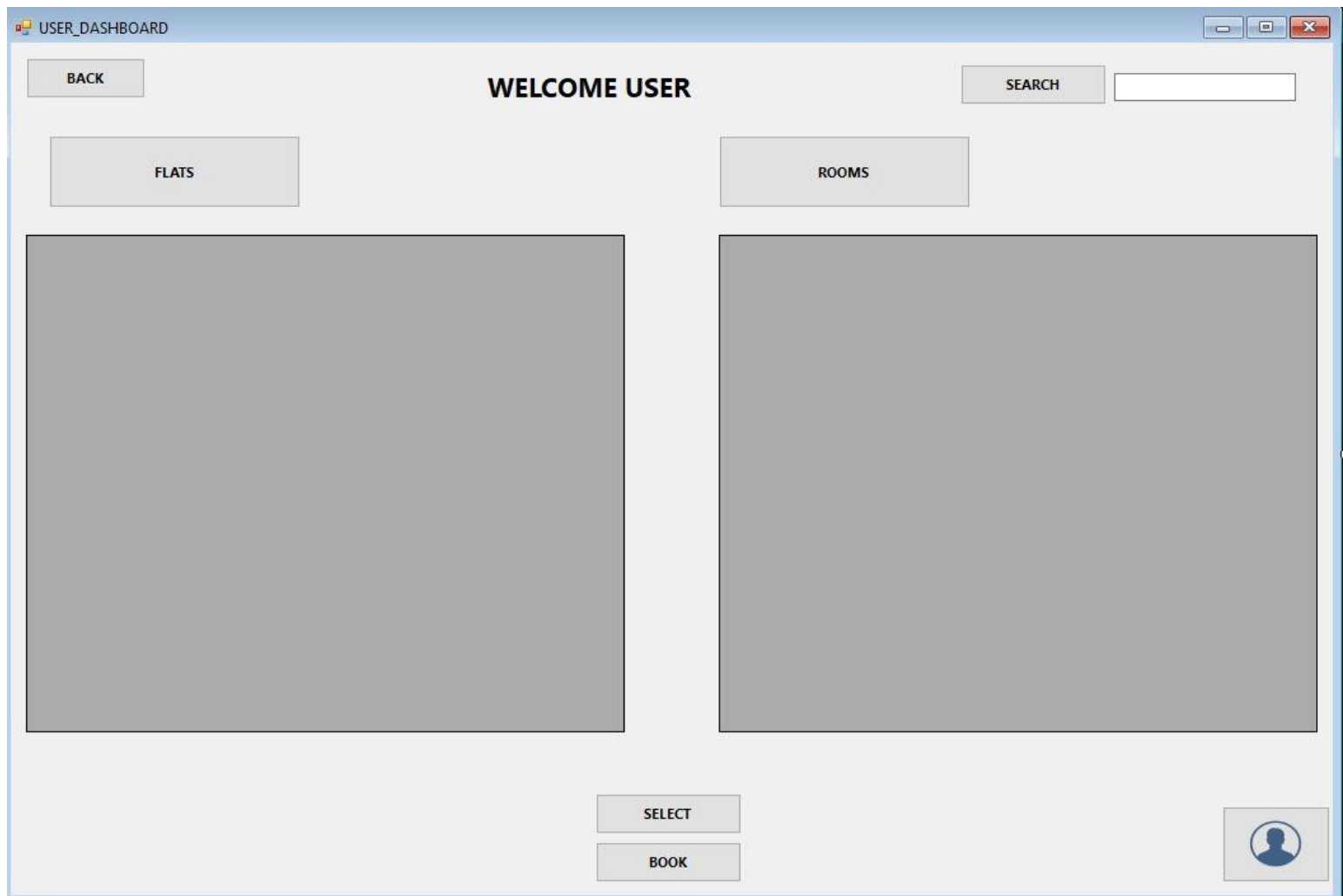
Back
Name <input type="text"/>
Password <input type="text"/>
<input type="button" value="Edit"/> <input type="button" value="Update"/>

User Dashboard Query:

```
string query = "SELECT f.OwnerName,f.FlatID,f.Rent,f.Available FROM Flat f INNER JOIN owner s ON
f.OwnerName = s.name ORDER BY CASE WHEN s.premium = 'Yes' THEN 0 ELSE 1 END;";
```

```
string query = "SELECT f.OwnerName,f.FlatID,f.Rent,f.RoomNo, f.Available FROM Room f INNER JOIN
owner s ON f.OwnerName = s.name ORDER BY CASE WHEN s.premium = 'Yes' THEN 0 ELSE 1 END;";
```

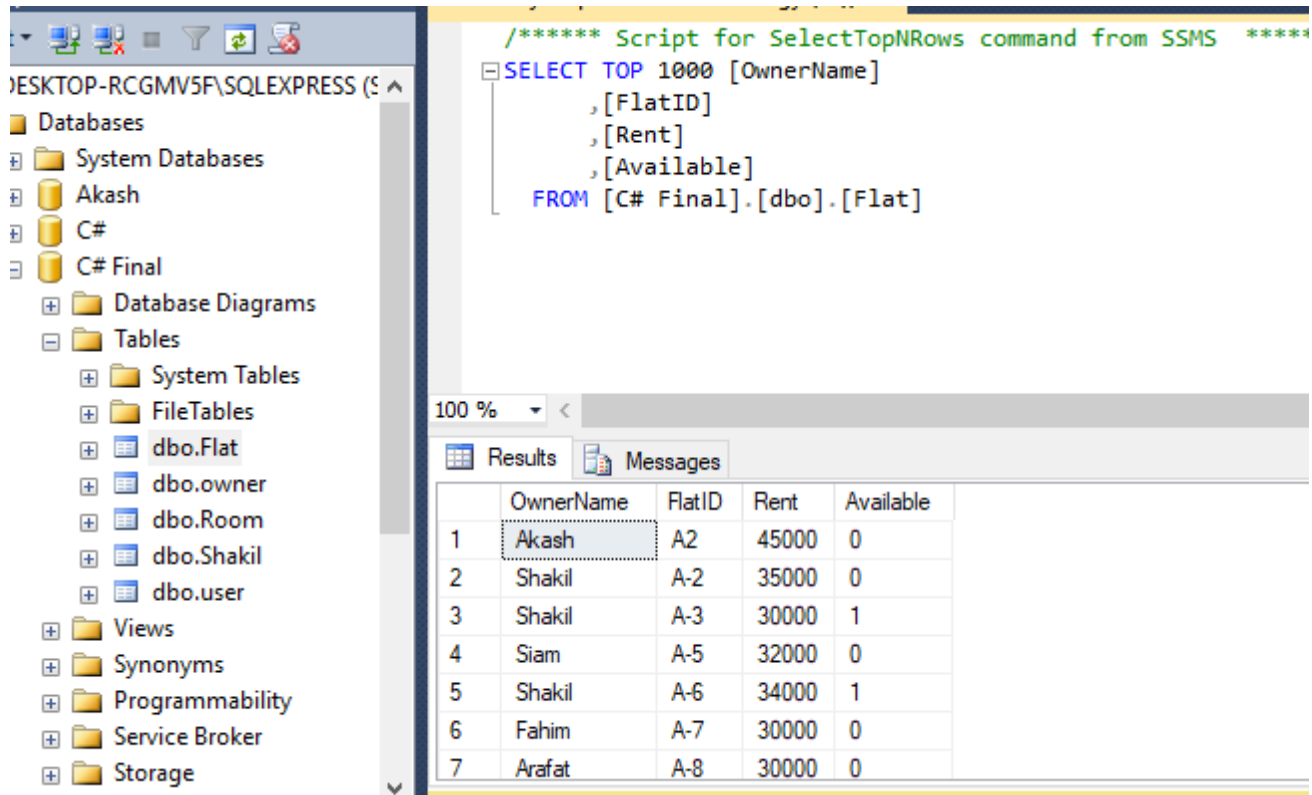
User Dashboard From:



The image shows a user dashboard window titled "USER_DASHBOARD". The dashboard has a light gray background and a blue border. At the top, there is a "WELCOME USER" message in the center. To the left of the welcome message is a "BACK" button. To the right is a "SEARCH" button followed by a text input field. Below the welcome message, there are two main sections: "FLATS" on the left and "ROOMS" on the right. Each section has a large gray rectangular placeholder for content. At the bottom of the dashboard, there are three buttons: "SELECT" and "BOOK" are stacked vertically in the center, and a user profile icon is on the right.

Database Table

Flat Table



The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Databases' folder is expanded, showing the 'C# Final' database. Under 'Tables', the 'dbo.Flat' table is selected. The main pane shows a SQL query script for the 'SelectTopNRows' command from SSMS. The query is as follows:

```
SELECT TOP 1000 [OwnerName]
      ,[FlatID]
      ,[Rent]
      ,[Available]
FROM [C# Final].[dbo].[Flat]
```

Below the query, the 'Results' tab is active, displaying a table with 7 rows and 5 columns: OwnerName, FlatID, Rent, and Available. The data is as follows:

	OwnerName	FlatID	Rent	Available
1	Akash	A2	45000	0
2	Shakil	A-2	35000	0
3	Shakil	A-3	30000	1
4	Siam	A-5	32000	0
5	Shakil	A-6	34000	1
6	Fahim	A-7	30000	0
7	Arafat	A-8	30000	0

Room Table

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Explorer' pane shows the database structure for 'DESKTOP-RCGMV5F\SQLEXPRESS'. The 'Tables' folder is expanded, showing the 'dbo.Room' table. The main pane shows a SQL query in 'SQLQuery2.sql' with the following text:

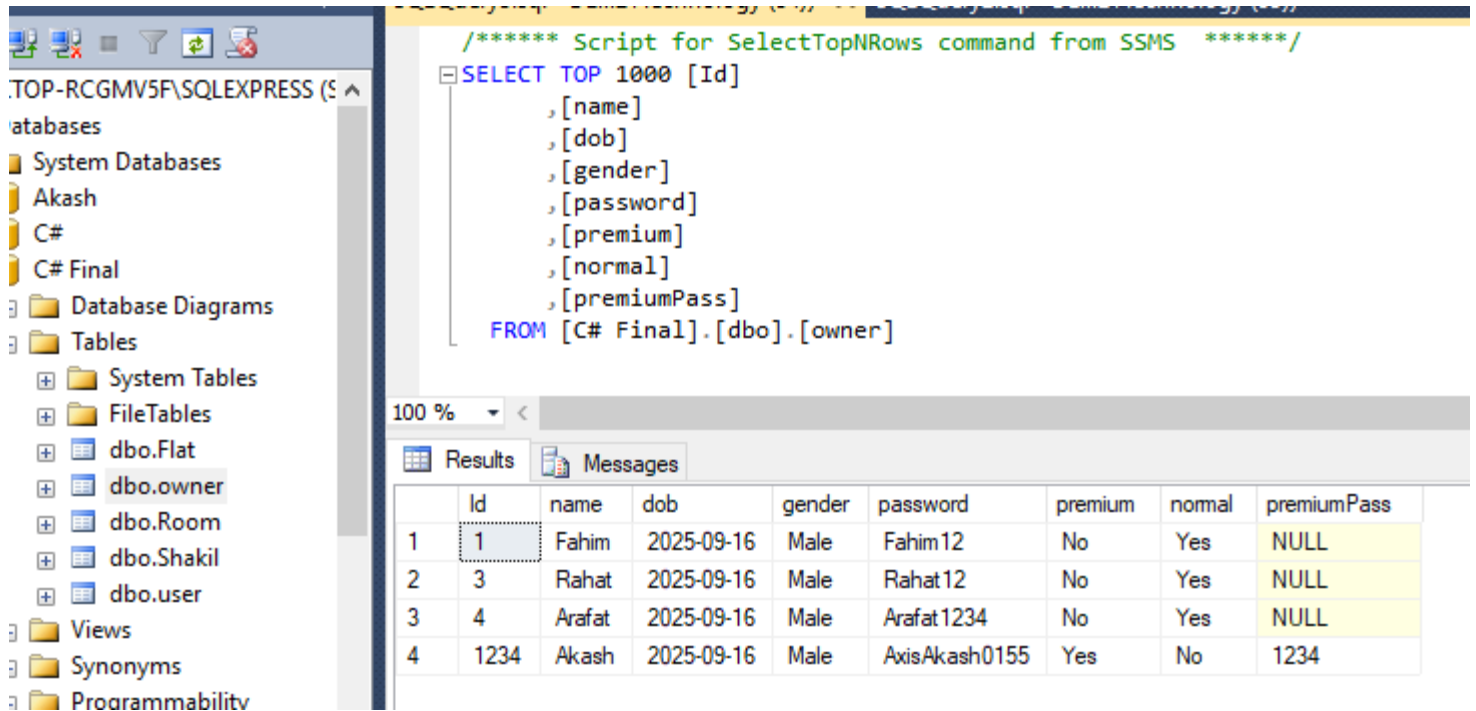
```

/***** Script for SelectTopNRows command from SSMS
SELECT TOP 1000 [OwnerName]
      ,[FlatID]
      ,[RoomNo]
      ,[Rent]
      ,[Available]
FROM [C# Final].[dbo].[Room]
  
```

Below the query, the 'Results' pane shows the output of the query, displaying the first 6 rows of the 'Room' table. The columns are OwnerName, FlatID, RoomNo, Rent, and Available.

	OwnerName	FlatID	RoomNo	Rent	Available
1	Fahim	A-12	2	16000	0
2	Shakil	A-2	3	15000	0
3	Shakil	A-5	2	15000	0
4	Shakil	A-6	3	19000	0
5	Arafat	A-8	2	15000	0
6	Akash	B2	5	6500	0

Owner Table



TOP-RCGMV5F\SQLEXPRESS (S)

atabases

- System Databases
- Akash
- C#
- C# Final
- Database Diagrams
- Tables
 - System Tables
 - FileTables
 - dbo.Flat
 - dbo.owner
 - dbo.Room
 - dbo.Shakil
 - dbo.user
- Views
- Synonyms
- Programmability

```

/***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP 1000 [Id]
      ,[name]
      ,[dob]
      ,[gender]
      ,[password]
      ,[premium]
      ,[normal]
      ,[premiumPass]
FROM [C# Final].[dbo].[owner]
  
```

100 %

Results Messages

	Id	name	dob	gender	password	premium	normal	premiumPass
1	1	Fahim	2025-09-16	Male	Fahim12	No	Yes	NULL
2	3	Rahat	2025-09-16	Male	Rahat12	No	Yes	NULL
3	4	Arafat	2025-09-16	Male	Arafat1234	No	Yes	NULL
4	1234	Akash	2025-09-16	Male	AxisAkash0155	Yes	No	1234

User Table

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Databases' folder is expanded, showing 'System Databases', 'Akash', 'C#', 'C# Final', 'Database Diagrams', 'Tables', 'System Tables', 'FileTables', 'dbo.Flat', 'dbo.owner', 'dbo.Room', 'dbo.Shakil', 'dbo.user', 'Views', 'Synonyms', and 'Programmability'. The 'dbo.user' table is selected. The main pane shows the SQL script for the 'SelectTopNRows' command from SSMS, which is a 'SELECT TOP 1000' query. The query selects columns [name], [dob], [gender], and [password] from the [C# Final].[dbo].[user] table. Below the script, the 'Results' tab is active, displaying a table with 4 rows and 6 columns: Id, name, dob, gender, and password. The data is as follows:

	Id	name	dob	gender	password
1	2	Rakib	2025-09-16	Male	Rakib12
2	3	Sakib	2025-09-16	Male	Sakib12
3	10	Riana	2025-09-16	Female	Riana12
4	12345	Rakib	2025-09-16	Male	Rakib12

1) Tables & Columns

user

- Id INT PRIMARY KEY
- name NVARCHAR(100) NOT NULL UNIQUE
- dob DATE NULL
- gender NVARCHAR(20) NULL
- password NVARCHAR(200) NOT NULL

owner

- Id INT PRIMARY KEY
- name NVARCHAR(100) NOT NULL UNIQUE
- dob DATE NULL
- gender NVARCHAR(20) NULL
- password NVARCHAR(200) NOT NULL
- premium NVARCHAR(10) NOT NULL DEFAULT 'No' -- e.g., 'Yes' / 'No' (used for ordering)
- normal NVARCHAR(10) NULL -- present in inserts; keep for compatibility
- premiumPass NVARCHAR(100) NULL -- used in one login variant

Flat

- FlatID NVARCHAR(50) PRIMARY KEY -- used for search & delete
- OwnerName NVARCHAR(100) NOT NULL -- joins to owner.name
- Rent DECIMAL(12,2) NOT NULL
- Available BIT NOT NULL DEFAULT 1

Room

- RoomNo NVARCHAR(50) PRIMARY KEY -- used for delete
- FlatID NVARCHAR(50) NOT NULL -- ties room to a flat
- OwnerName NVARCHAR(100) NOT NULL -- mirrors Flat for query convenience
- Rent DECIMAL(12,2) NOT NULL
- Available BIT NOT NULL DEFAULT 1

2) Keys & Relationships

- Flat.OwnerName → owner.name (FK)
- Room.FlatID → Flat.FlatID (FK)
- Room.OwnerName → owner.name (FK)
- Enforce owner.name uniqueness to support those JOIN ... ON f.OwnerName = s.name queries and login lookups.
- Keep Available as BIT since your UI shows “Available / Booked”.

3) Suggested Indexes

- owner(name) — UNIQUE (support joins & logins)
- Flat(OwnerName), Flat(Rent) — for search and premium-first ordering
- Room(FlatID), Room(OwnerName) — for owner views & joins

Conclusion:

The Rental Management System is a .NET desktop application designed to streamline the process of property rentals. It enables tenants to easily view and book available flats or rooms, supporting both family rentals (entire flat) and bachelor rentals (room-by-room). Owners are classified as Normal or Premium, with premium listings given display priority to enhance visibility. An integrated Admin Panel ensures efficient oversight of users, properties, and transactions. This system provides a structured, transparent, and reliable solution for managing house rentals.

References:

[1] Microsoft, *Windows Forms Overview*. [Online].

Available: <https://learn.microsoft.com/en-us/dotnet/desktop/winforms>

[2] Microsoft, *SQL Server Database Engine*. [Online].

Available: <https://learn.microsoft.com/en-us/sql>

[3] TutorialsTeacher, *C# .NET Framework Tutorial*. [Online].

Available: <https://www.tutorialsteacher.com/csharp>

[4] GeeksforGeeks, *Database Management System (DBMS) Concepts*. [Online].

Available: <https://www.geeksforgeeks.org/dbms>

[5] OpenAI, *Emma (ChatGPT) – Project guidance and system design assistance for Rental Management System*. [Online].

Available: <https://chat.openai.com>

GitHub Link:

<https://github.com/AxisAkash/lost-Found.git>