PROPERTY RENTAL MANAGEMENT SYSTEM

Course Number: 420-DW4-AS – Course Title: Web Server Applications Development II – Teacher: Quang Hoang Cao Session: Fall 2024 – Group 07422

DECEMBER 4, 2024

QUEEN SARAH ANUMU BIH

CONTENTS

I. Project Description

The **Property Rental Management System** (PRMS) is a web-based application designed to streamline the management of rental properties. The system allows property owners, managers, and tenants to efficiently handle property listings, rental agreements, maintenance requests, and payments.

II. Project Development

Phase I Analysis

1. Software Requirements

To develop the Property Rental Management System (PRMS) following technologies and tools used:

- **Visual Studio 2022**: An IDE for C# and ASP.NET Core, enhancing coding enhancing coding efficiency and debugging.
- **SQL Server 2022**: Provides robust data storage and management features, ensuring data integrity.
- **SQL Server Management Studio**: Offers effective tools for database management.
- **ASP.Net Entity Framework MVC**: A development framework that facilitates building web applications by combining ASP.NET MVC with Entity Framework.
- **ASP.NET MVC**: A framework for building web applications using the MVC design pattern. The Model represents the data, the View displays the data, and the Controller handles the user input and updates the model and view.
- **Entity Framework**: An ORM (Object-Relational Mapping) tool for .NET that simplifies database interactions by allowing you to work with relational data as objects in C# without needing to write raw SQL queries.
- **Bootstrap**: Ensure responsive design and a visually appealing user interface across various devices.
- **jQuery**: Enhances user interactivity through simplified AJAX and DOM manipulation and efficient event handling.

2. Functional Requirements

Who are the users of this web app?

- Property Owner (Landlord)
- Property Managers (Manager)
- Potential Tenants (Tenant)

User Operations:

Users	Operations
Property Owner	• Create/Update/Delete/Search/List any property manager account • Update/Delete/Search/List any potential tenant account • View reports/messages for building property that was reported by property manager.
Property Manager	• Perform CRUD operations related to buildings • Perform CRUD operations related to apartments • Keep track of apartments status • Schedule potential tenants ‘appointments • Respond to potential tenants ‘messages • Report any events to the property owner when necessary
Potential Tenants	• Create an on-line account through Property Rental Management Web Site • View any apartment suitable for their needs • Make an appointment with the property manager • View appointments approved/canceled by property manager • Send necessary messages to the property manager

3. Non-Functional Requirements

1) Security:

- User Authentication: The system implements user authentication to verify user identities.
- Authorization: The system ensure that only authorized users can perform specific actions, protecting sensitive data.

Phase II Design

Database Schema

Database Name: ***Property_Rental_DB***

Tables:

1) Statuses

Fields	**Data Types**	**Design Notes**
StatusId	int	PK (auto generated)
Description	nvarchar(60)	Status' Description

Example

StatusId	**Description**	**Category**
1	Available	Apartments
2	Occupied	Apartments
3	Maintenance	Apartments
4	Pending	Appointments, Payments
5	Confirmed	Appointments
6	Canceled	Appointments
7	Read	Messages
8	Unread	Messages
9	Fully Paid	Payments
10	Completed	Leases
11	Renewal Pending	Leases

2) Users

Fields	**Data Types**	**Design Notes**
UserId	int	PK
Password	nvarchar(50)	password

Example

UserId	**Password**
10003	manager10003
2000001	tenantpass1
3001	landlord3001

3) Landlords

Fields	**Data Types**	**Design Notes**	**Example**
LandlordId	int	PK	3001
FirstName	nvarchar(50)	User's first name	Alice
LastName	nvarchar(50)	User's last name	Smith
Email	nvarchar(100)	Unique Email	alicesmith@gmail.com
Phone	nvarchar(20)	Format xxx xxx-xxxx	234-567-8901

4) Managers

Fields	**Data Types**	**Design Notes**	**Example**
ManagerId	int	PK	10003
FirstName	nvarchar(50)	User's first name	Michael
LastName	nvarchar(50)	User's last name	Smith
Email	nvarchar(100)	Unique Email	michaelsmith@gmail.com
Phone	nvarchar(20)	Format xxx xxx-xxxx	789-012-3456

5) Tenants

Fields	**Data Types**	**Design Notes**	**Example**
TenantId	int	PK	2000001
FirstName	nvarchar(50)	User's first name	Jane
LastName	nvarchar(50)	User's last name	Doe
Email	nvarchar(100)	Unique Email	janedoe@gmail.com
Phone	nvarchar(20)	Format xxx xxx-xxxx	345-678-9012

6) Buildings

Fields	**Data Types**	**Design Notes**	**Example**
BuildingCode	nvarchar(3)	PK, format BXX	B01
LandlordId	int	FK Landlords	1
ManagerId	int	FK Managers	2
BuildingName	nvarchar(50)	Building Name	Royal's Building
Description	nvarchar(100)		The most unique building.
Address	nvarchar(50)		345 rue Laval
City	nvarchar(50)		Montreal
Province	nvarchar(50)		Québec
ZipCode	nvarchar(7)		T67 Y5G

7) Apartments

Fields	**Data Types**	**Design Notes**	**Example**
ApartmentNum	int	PK (auto generated)	1
BuildingCode	nvarchar(3)	FK Buildings	B01
Rooms	int		4
Bathrooms	int		2
Description	nvarchar(255)		This is a nice apartment
Rent	decimal	Currency	1000.00
StatusId	int	FK Statuses	1, 2, 3

8) Schedules

Fields	Data Types	Design Notes	Example
ScheduleId	int	PK (auto generated)	1
ManagerId	int	Managers	10002
ScheduleDate	date		2024-11-10
StartTime	time		11:00
EndTime	time		11:30

9) Appointments

Fields	Data Types	Design Notes	Example
AppointmentId	int	PK (auto generated)	1
TenantId	int	FK Tenants	2000001
ManagerId	int	FK Managers	10001
AppartmentNum	int	FK Apartments	1
ScheduleId	int	FK Schedules	1
StatusId	int	FK Statuses	4, 5, 6

10) Messages

Fields	Data Types	Design Notes	Example
MessageId	int	PK (auto generated)	1
SenderId	int	FK Users	2000001
ReceiverId	int	FK Users	10001
MessageSubject	nvarchar(50)		
MessageBody	nvarchar(255)		
MessageDate	datetime		2024-10-22 12:00PM
StatusId	int	FK Statuses	7, 8

11) Leases (Added Feature)

Fields	Data Types	Design Notes	Example
LeaseId	nvarchar(4)	PK	L001
TenantId	int	FK Tenants	2000001
AppartmentNum	int	FK Apartments	1
StartDate	date		2024-11-10
EndDate	date		2024-11-30
MonthlyRent	decimal(10, 2)		1200.00
StatusId	int	FK Statuses	10, 11

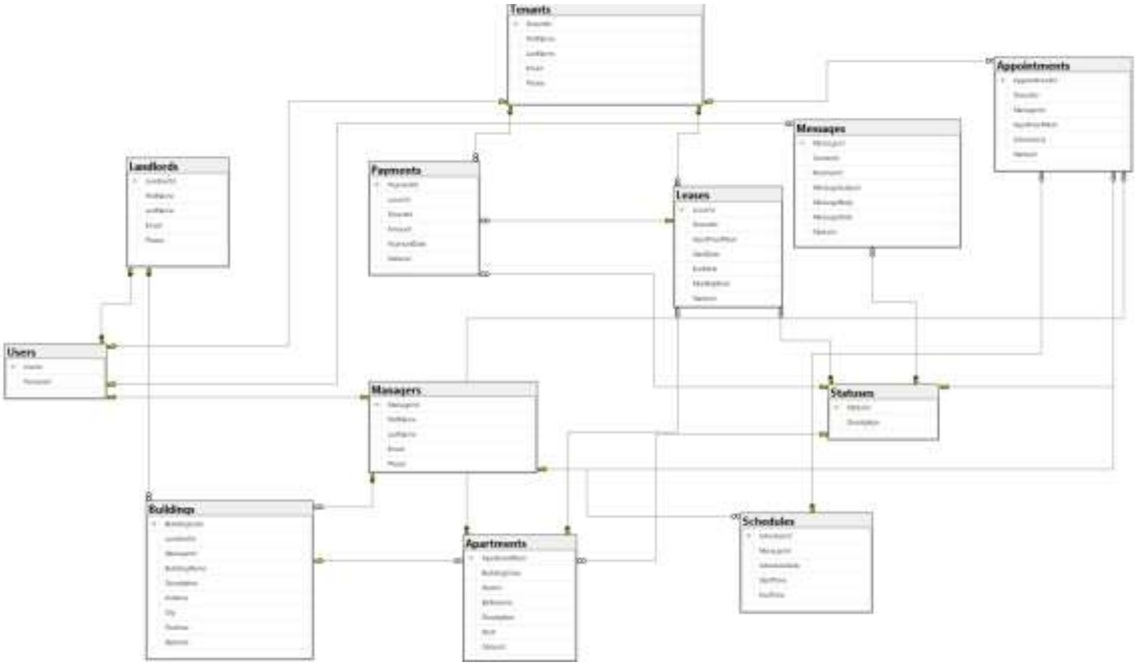
12) Payments (Added Feature)

Fields	Data Types	Design Notes	Example
PaymentId	int	PK	1
LeaseId	nvarchar(4)	FK Leases	L001
TenantId	int	FK Tenants	2000001
Amount	decimal		1000.00
PaymentDate	date		2024-11-20
StatusId	int	FK Statuses	4, 9

Associations Between Tables

Source Table	Target Table	Relationship	Description
Statuses	Apartments	One-to-Many	one status can be associated with many apartment
	Appointments	One-to-Many	one status can be associated with many appointments
	Messages	One-to-Many	one status can be associated with many messages
	Leases	One-to-Many	one status can be associated with many leases
	Payments	One-to-Many	one status can be associated with many payments
Users	Landlords	One-to-One	each user corresponds to exactly one landlord
	Managers	One-to-One	each user corresponds to exactly one manager
	Tenants	One-to-One	each user corresponds to exactly one tenant
	Messages	One-to-Many	a user can send or receive many messages
Landlord	Buildings	One-to-Many	a landlord can own multiple buildings
Managers	Buildings	One-to-Many	a manager can manage multiple buildings
	Schedules	One-to-Many	a manager can have multiple schedules
	Appointments	One-to-Many	a manager can handle multiple appointments
Tenants	Appointments	One-to-Many	a tenant can book multiple appointments
	Payments	One-to-Many	a tenant can make multiple payments
	Leases	One-to-Many	a tenant can have multiple leases
Buildings	Apartments	One-to-Many	a building can have multiple apartments
Apartments	Appointments	One-to-Many	an apartment can be part of multiple appointments
	Leases	One-to-Many	an apartment can be leased multiple times over its lifetime
Schedules	Appointments	One-to-Many	a schedule can have multiple appointments?
Appointments			
Messages			
Leases			
Payments			

ER Diagram


Tenants
Appointments
Messages
Landlords
Payments
Leases
Users
Managers
Statuses
Buildings
Apartments
Schedules

Phase III Implementation

1. **Creation of the Database**: Established the database in SQL Server Management Studio. Used database first approach.
2. **Creation of ASP.NET Entity Framework MVC**: Developed the web application using ASP.NET EntityFramework MVC in Visual Studio 2022 to facilitate a structured approach to building the application.
3. **Installation of Necessary Package**: Installed essential NuGet packages to enable Entity Framework MVC functionalities:
    - ***EntityFramework 6.4.4***: Entity Framework 6 is tried and tested object-relational mapper for .NET with many years of feature development and stabilization.
4. **Model Generation from database:** Utilized ADO.NET Entity Data Model, using EF Designer from database to generate models from database tables. Build project after generation of models.

    Connection String created in web.config:

    ```
    <connectionStrings>
     <add name="Property_Rental_DBEntities" connectionString="metadata=res://*/Models.PropertyRentalModel.csdl|res://*/Models.PropertyRentalModel.ssdl|res://*/Models.PropertyRentalModel.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=.;initial catalog=Property_Rental_DB;user id=sa;password=sysadm;trustservercertificate=True;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />
    </connectionStrings>
    ```

    Note: *If running on your server, you need to change the connection string based on you SQL Server credentials.*
5. **Creation of Controllers and Views:** Used ASP.NET Scaffolding to generate the controllers and views.

Login Authentication/Authorization

Authentication: This is handled by the FormsAuthentication system, which checks the user's credentials when they log in and sets a cookie for future requests.

Form authentication in the web.config:

```
<authentication mode="Forms">

        <forms loginUrl="Accounts/Login" timeout="30"></forms>

</authentication>
```

Authorization: After a user is authenticated, their role (stored in the *UserType* cookie) determines what parts of the application they can access. For example, a *Tenant* may only have access to tenant-specific pages, while a *Manager* might have broader access.

The code sets up cookie authentication with a designated login path (/Accounts/Login) and an expiration time of 30 minutes. The default route is established to direct requests to the **Home controller's Index action**.

The **Accounts controller** is responsive for managing user authentication-related actions, including login, signup, and logout processes. The **Login** action checks user authentication status, redirecting already authenticated users to their details page. The **SignUp** action validates user input – user id (ensures it is unique), first name, last name, password, email (ensures it is unique), and phone number (ensures it is unique), and then saves new tenant to the database if all criteria are met.

In the **Login** action, user credentials are validated against the database. Upon successful authentication, the user is signed in using cookie-based authentication. If the authentication fails, appropriate error messages are displayed.

Models

1. **Apartment**: Represents rental apartments with details like rent, number of rooms, description, availability status.
2. **Appointment**: Tracks meetings between tenants and managers for apartment viewing.
3. **Building**: Stores information about the rental properties, including location, landlord id, manager id.
4. **Landlord**: Captures personal account details of the landlord such as first name, last name, unique email, and unique phone number.
5. **Lease (Additional Feature)**: Has the contract details of tenant's apartments.
6. **Manager**: Captures personal account details of the manager such as first name, last name, unique email, and unique phone number.
7. **Message**: Manages communication between tenants and managers and managers and landlord via messaging system.
8. **Payment (Additional Feature)**: Has payment details made by tenant.
9. **Schedule**: Represents availability for property manager.
10. **Status**: Defines the current state of apartments, appointments, messages, leases, and payments.
11. **Tenant**: Captures personal account details of the tenant such as first name, last name, unique email, and unique phone number.
12. **User**: Creates a user. Create the user id and password of a user account before the users can fill up the rest of their account details.
13. **ApartmentDetailsViewModel**: Used specifically by the tenants to view apartments. (Added this to try out the image display differently. This is because I added image attribute in this class).
14. **SignupViewModel**: Used creating new tenant users.
15. **Metadata Classes**: Used for extending classes in the model with additional attributes or validation rules without modifying the original source code.
16. **Partial Classes**: Allow the definition of a class to be split across multiple files.

Controllers

1. **AccountsControllers**: Manages user login, signup, logout, and authorization for the system.
2. **ApartmentsController**: Handles apartment-related operations, including adding, updating, deleting and listing apartments.
3. **AppointmentsController**: Manages scheduling and handling of tenant-manager appointments such as creating.
4. **BuildingsController**: Facilitates CRUD operations for the managing property buildings in the system.
5. **HomeController**: Manages navigation for the home page and general application pages.
6. **LandlordsController**: View landlord details and manages pages of functionalities assigned to him.
7. **LeasesController (Additional feature)**: CRUD operations for leases. NOT IMPLEMENTED YET.
8. **ManagersController**: Allows landlord to perform CRUD and search operations for the managers in the system and manages pages of functionalities assigned to him.
9. **MessagesController**:
10. **PaymentsController (Additional feature)**: CRUD operations for payments. NOT IMPLEMENTED YET.
11. **SchedulesController**: Allows manager to perform CRUD operations on the system based on his availability.
12. **StatusesController**:
13. **TenantsController**: Allows landlord to perform read, update, delete, and search operations for the tenants in the system and manages pages of functionalities assigned to him.
14. **UsersController**: CRUD operations for the user in the system.

Views

1. **Accounts**: Provides pages for user login and sign up.
   a. Login: Shows the login page for users to authenticate themselves.
   b. Signup: Allows new tenants users to register for an account.
2. **Apartments**: Displays and manages apartment-related information and actions.
   a. Create: Form for adding a new apartment to the system.
   b. Delete: Form for deleting an apartment.
   c. Details: Form to view detailed information about a specific apartment.
   d. Edit: Form for editing an existing apartment’s information.
   e. Index: Form to list all apartments.
3. **Appointments**: Pages for managing tenant-manager appointments.
   a. Create: Form to schedule a new appointment.
   b. Details: Form to list all appointments.
4. **Buildings**: Pages for managing property buildings.
   a. Create: Form for adding a new building.
   b. Delete: Form for deleting a building.

   c. Details: Form to view a specific building.
   d. Edit: Form to edit an existing building.
   e. Index: Form to list all buildings.

5. **Home**: General application pages accessible to all users.
   a. Index: Welcome page to app; informs users what the app is all about.
   b. About: Shows building views for guests.
6. **Landlords**: Landlord pages.
   a. Index: View landlord logged in.
   b. ReceiveMessages: View messages sent by property managers.
   c. ViewMessage: See the details of received messages.
7. **Leases**: Pages for apartment contracts (DID NOT IMPLEMENT)
8. **Managers**: Manager pages.
   a. Apartments: View apartments belonging to the property manager logged in.
   b. AppointmentDetails: Appointment confirmation/cancellation.
   c. Buildings: View buildings belonging to the property manager logged in.
   d. Create: Form allows landlord to create a new property manager.
   e. Delete: Allows landlord to delete an existing property manager.
   f. Details: Allows landlord to view details of a specific property manager.
   g. Edit: Allows landlord to update an existing manager's information.
   h. Index: Allows landlord to view a property manager.
   i. MyDetails: Allows manager to view details after logging in.
   j. ReceiveMessages: Allows managers to view messages sent by tenants.
   k. Search: Allows landlord to search manager's credentials by filter.
   l. SendMessageToLandlord: Report events to the landlord.
   m. SendMessageToTenant: Allows managers to respond to tenants' messages.
   n. ViewApartmentsByStatus: Keep track of apartment status.
   o. ViewAppointments: View appointments requested by tenants.
   p. ViewMessage: View details of a specific message sent by tenants and status changes to read when viewed.
9. **Messages**: Pages for messages already implemented in the users' view.
10. **Payments**: Pages for apartment payments (DID NOT IMPLEMENT)
11. **Schedules**:
    a. Create: Allows property managers to add their schedule.
    b. Delete: Allows property managers to delete their schedule.
    c. Details: Allows property managers to view details of a specific schedule.
    d. Edit: Allows property managers to edit their schedule.
    e. Index: Allows property managers to view only their schedules.
12. **Shared**:
    a. _Layout: View for the all the app's pages.
    b. Error: Handles view for error if one occurred while processing request.
13. **Statuses**: View pages for statuses.
14. **Tenants**: Tenant pages
    a. ApartmentDetails: Allows tenant to view available apartments.
    b. BookAppointment: Allows tenant to book an appointment with the manager.
    c. Delete: Allows landlord to delete an existing potential tenant.
    d. Details: Allows landlord to view details of an existing potential tenant.

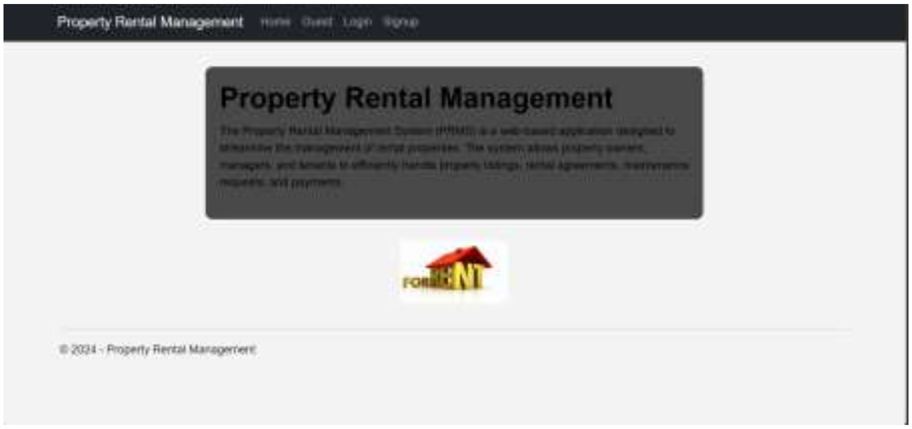
e. Edit: Allows landlord to update an existing tenant’s information.
f. Index: Allows landlord to view a property manager.
g. MyDetails: Allows tenant to view details after logging in.
h. ReceiveMessages: Allows tenants to view messages sent by managers.
i. Search: Allows landlord to search tenant’s credentials by filter.
j. SendMessageToManager: Form allows tenants to send messages to property managers.
k. ViewAppointments: View booked appointment and see the status of the appointment.
l. ViewMessage: View details of a specific message sent by property manager and status changes to read when viewed.
m. ViewSchedules: Allows tenants to view property managers’ schedules.

15. **Users**: Pages for users.

a. Create: Allows user to create a new tenant user during signup.

Pages Design

Home Page

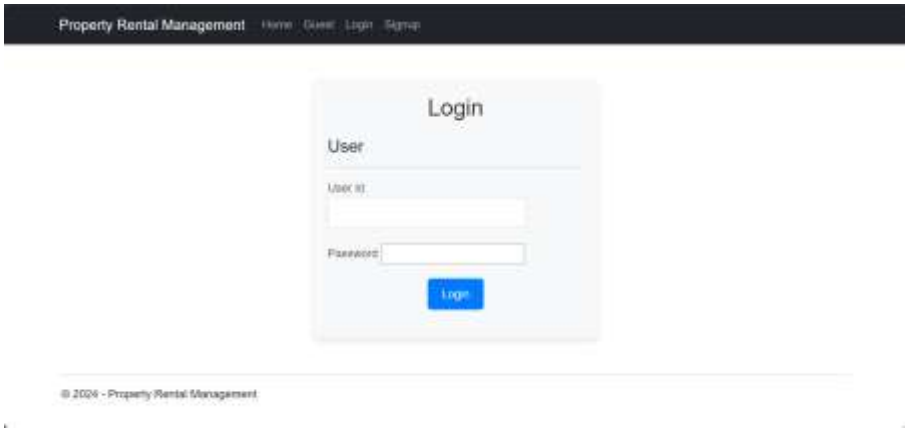


Property Rental Management

The Property Rental Management System (PRMS) is a web-based application designed to streamline the management of rental properties. The system allows property owners, managers, and tenants to efficiently handle property listings, rental agreements, maintenance requests, and payments.

© 2024 - Property Rental Management

Login Page



Login

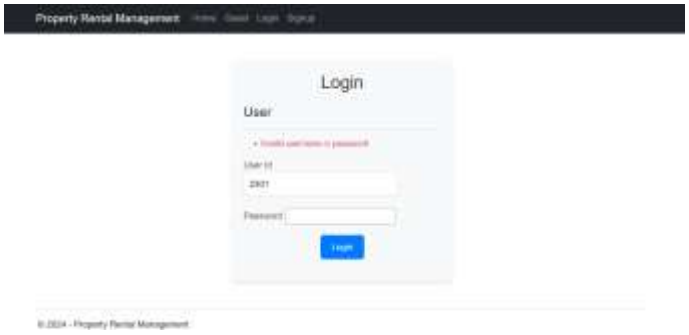
User

User Id

Password

© 2024 - Property Rental Management

Handled Exceptions for Signup page



Login

User

- Invalid user name or password

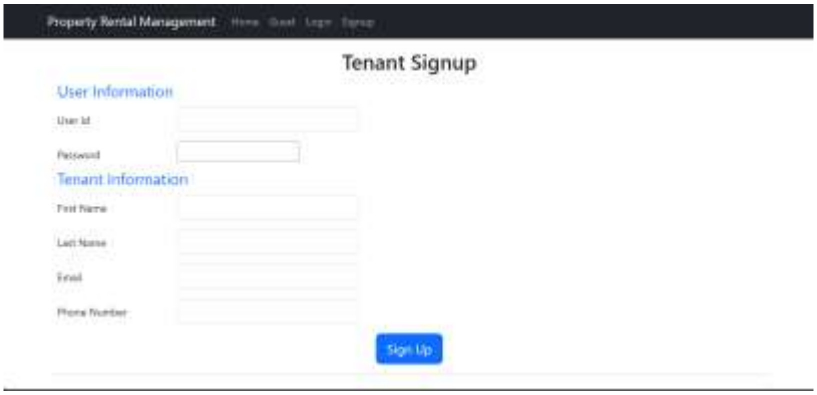
User Id

2021

Password

© 2024 - Property Rental Management

Signup Page



Tenant Signup

User Information

User Id

Password

Tenant Information

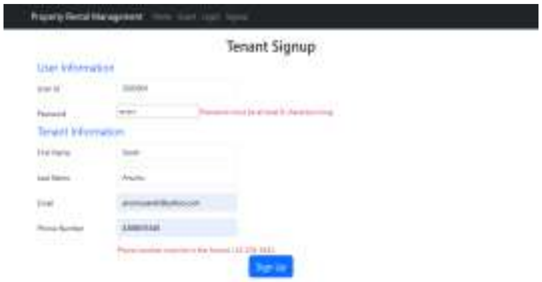
First Name

Last Name

Email

Phone Number

Handled Exceptions for Signup page



Tenant Signup

User Information

User Id [illegible]

Password [illegible] [illegible]

Tenant Information

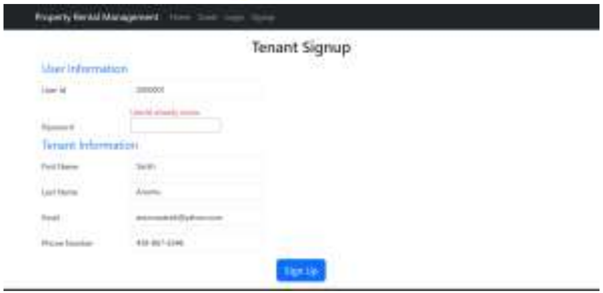
First Name [illegible]

Last Name [illegible]

Email [illegible]

Phone Number [illegible]

[illegible]



Tenant Signup

User Information

User Id [illegible]

UserId already exists

Password

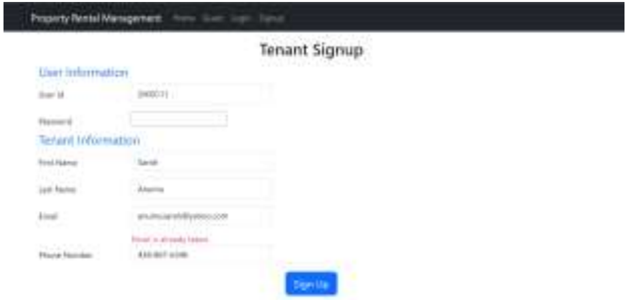
Tenant Information

First Name [illegible]

Last Name [illegible]

Email [illegible]

Phone Number [illegible]



Tenant Signup

User Information

User Id [illegible]

Password

Tenant Information

First Name [illegible]

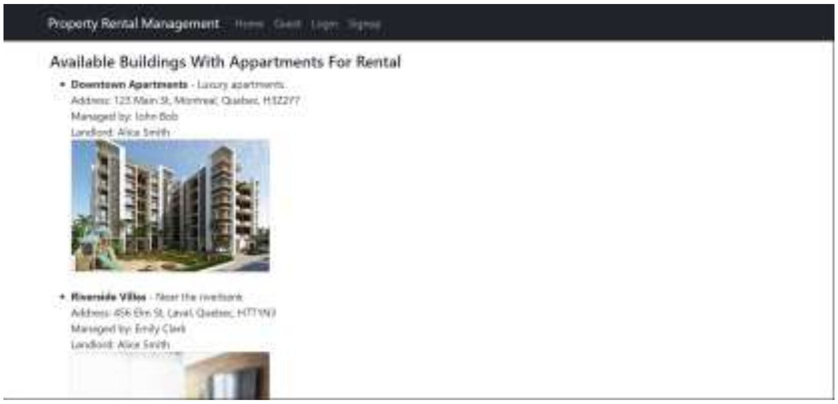
Last Name [illegible]

Email [illegible]

Email is already taken

Phone Number [illegible]

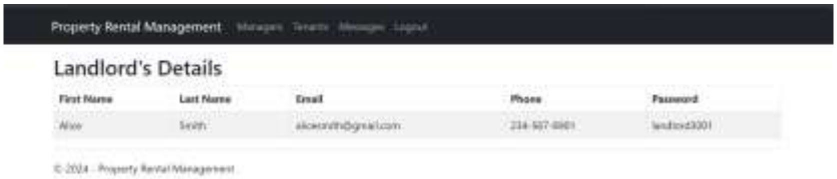
Guest Page



Available Buildings With Appartments For Rental

- **Downtown Apartments** - Luxury apartments.
  Address: 123 Main St, Montreal, Quebec, H3Z2Y7
  Managed by: John Bob
  Landlord: Alice Smith
- **Riverside Villas** - Near the riverbank
  Address: 456 Elm St, Laval, Quebec, H7T1N3
  Managed by: Emily Clark
  Landlord: Alice Smith

Landlord’s Dashboard

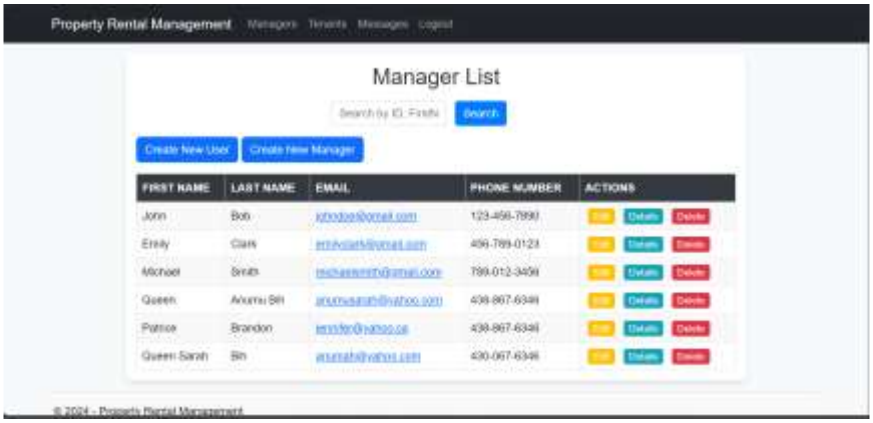


Landlord's Details

First Name	Last Name	Email	Phone	Password
Alice	Smith	alicesmith@gmail.com	234-567-8901	landlord3001

© 2024 - Property Rental Management

Landlord’s view for managing Property Managers

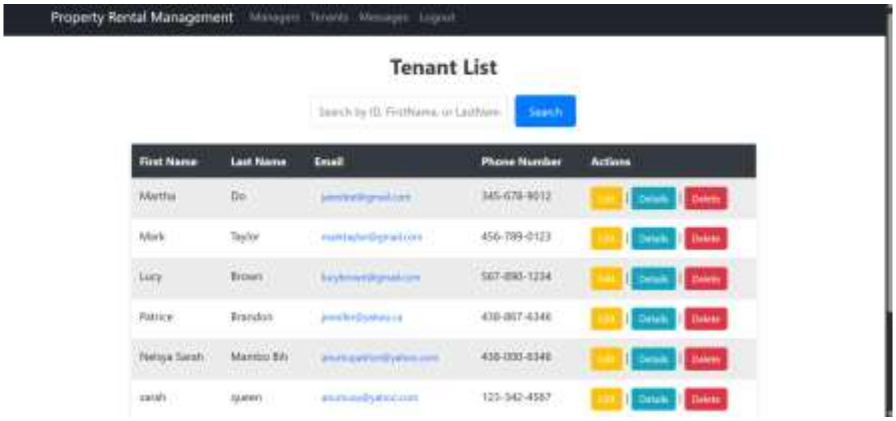


Manager List

FIRST NAME	LAST NAME	EMAIL	PHONE NUMBER	ACTIONS
John	Bob	johndoe@gmail.com	123-456-7890	Edit Details Delete
Emily	Clark	emilyclark@gmail.com	456-789-0123	Edit Details Delete
Michael	Smith	michaelsmith@gmail.com	789-012-3456	Edit Details Delete
Queen	Anumu Bih	anumusarah@yahoo.com	438-867-6346	Edit Details Delete
Patrice	Brandon	jennifer@yahoo.ca	438-867-6346	Edit Details Delete
Queen Sarah	Bih	anumsah@yahoo.com	430-067-6346	Edit Details Delete

© 2024 - Property Rental Management

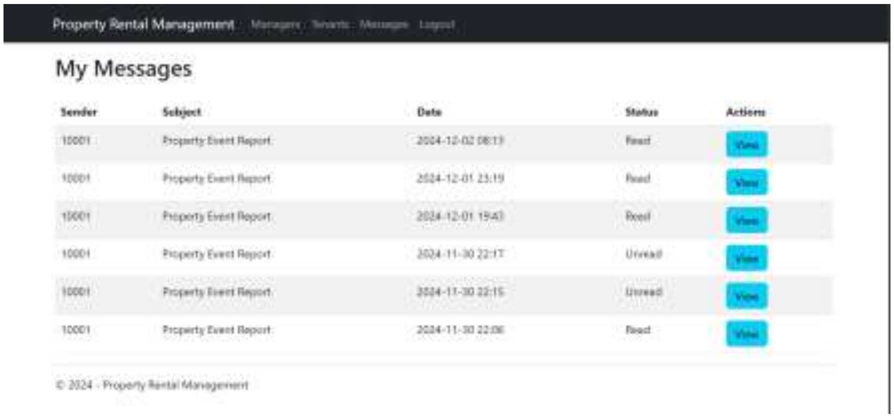
Landlord’s view for managing Potential Tenants


Property Rental Management
Managers
Tenants
Messages
Logout
Search by ID, FirstName, or LastName
Search

Tenant List

First Name	Last Name	Email	Phone Number	Actions
Martha	Do	[illegible]@gmail.com	345-678-9012	Edit | Details | Delete
Mark	Taylor	marktaylor@gmail.com	456-789-0123	Edit | Details | Delete
Lucy	Brown	lucybrown@gmail.com	567-890-1234	Edit | Details | Delete
Patrice	Brandon	[illegible]@yahoo.ca	438-867-6346	Edit | Details | Delete
Nelaya Sarah	Manmbo Bih	[illegible]@yahoo.com	438-000-6348	Edit | Details | Delete
sarah	queen	[illegible]@yahoo.com	123-342-4567	Edit | Details | Delete

Landlord’s view for received messages

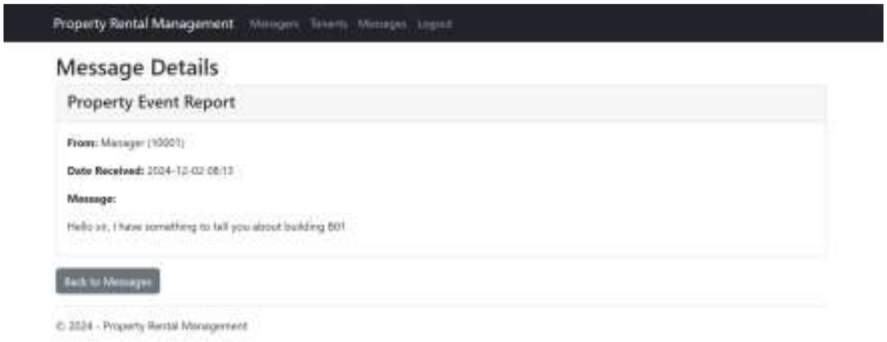


My Messages

Sender	Subject	Date	Status	Actions
10001	Property Event Report	2024-12-02 08:13	Read	View
10001	Property Event Report	2024-12-01 23:19	Read	View
10001	Property Event Report	2024-12-01 19:43	Read	View
10001	Property Event Report	2024-11-30 22:17	Unread	View
10001	Property Event Report	2024-11-30 22:15	Unread	View
10001	Property Event Report	2024-11-30 22:06	Read	View

© 2024 - Property Rental Management

Landlord’s view for message details



Message Details

Property Event Report

From: Manager (10001)

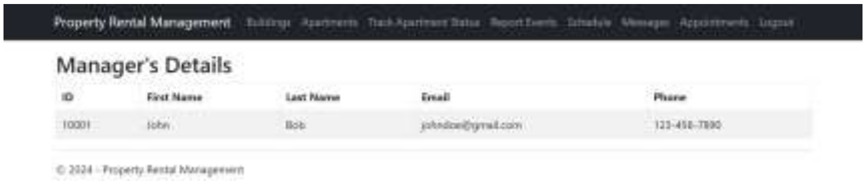
Date Received: 2024-12-02 08:13

Message:

Hello sir, I have something to tell you about building 801

© 2024 - Property Rental Management

Property Manager Dashboard – The logged in manager sees only his/her details.

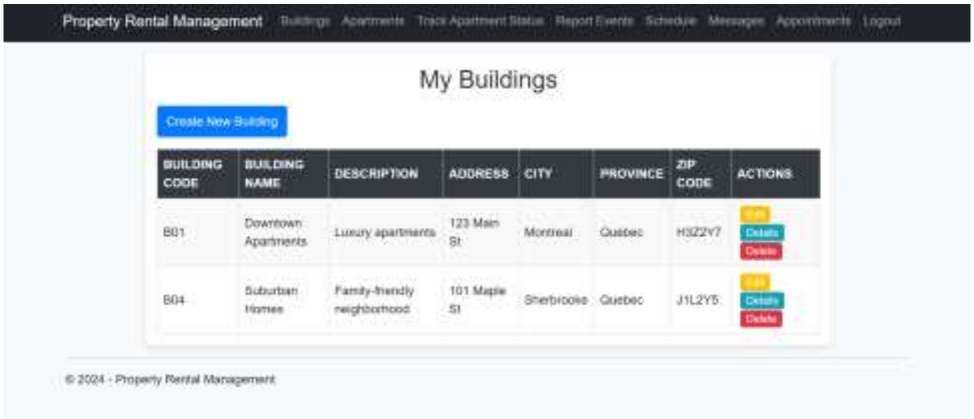


Manager's Details

ID	First Name	Last Name	Email	Phone
10001	John	Bob	johndoe@gmail.com	123-456-7890

© 2024 - Property Rental Management

Property Manager’s view for managing their buildings

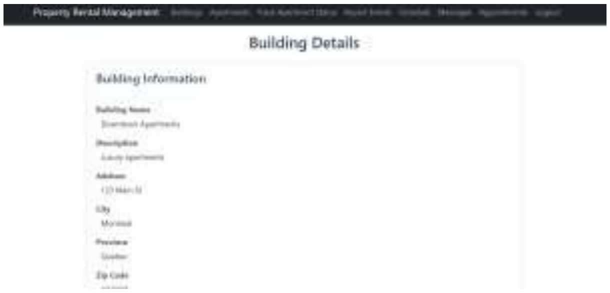


My Buildings

BUILDING CODE	BUILDING NAME	DESCRIPTION	ADDRESS	CITY	PROVINCE	ZIP CODE	ACTIONS
B01	Downtown Apartments	Luxury apartments	123 Main St	Montreal	Quebec	H3Z2Y7	Edit Details Delete
B04	Suburban Homes	Family-friendly neighborhood	101 Maple St	Sherbrooke	Quebec	J1L2Y5	Edit Details Delete

© 2024 - Property Rental Management

Property Manager’s View for building details with image



Building Details

Building Information

Building Name
Downtown Apartments

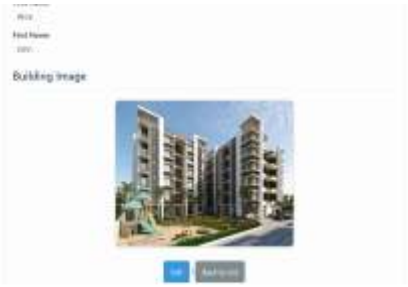
Description
Luxury apartments

Address
123 Main St

City
Montreal

Province
Quebec

Zip Code

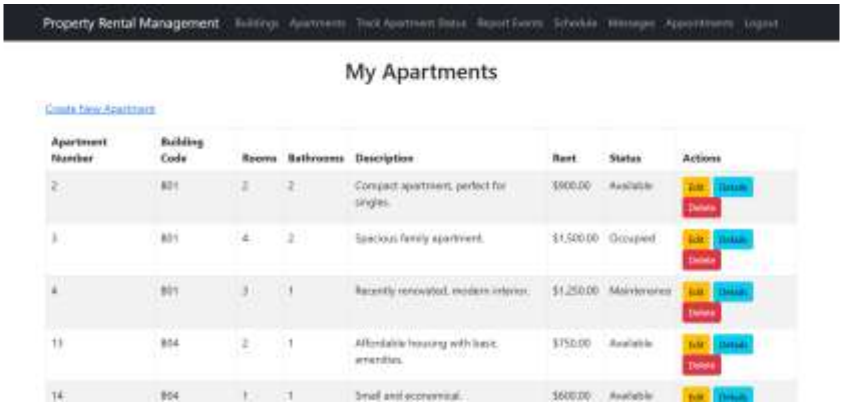


Bob

First Name
John

Building Image

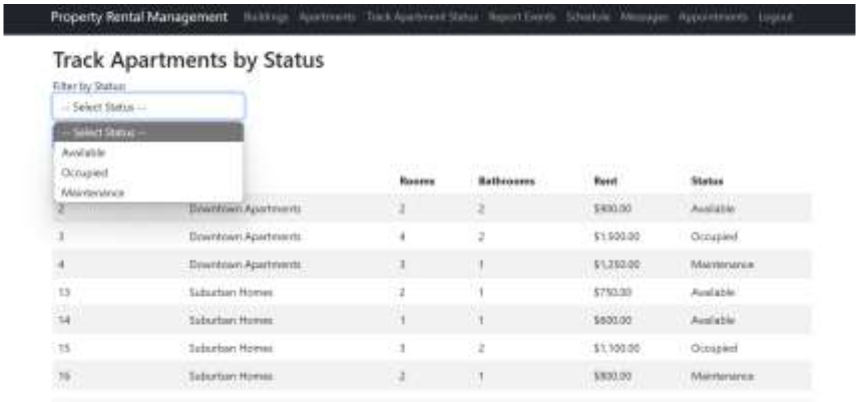
Property Manager’s view for managing their apartments



My Apartments

Apartment Number	Building Code	Rooms	Bathrooms	Description	Rent	Status	Actions
2	801	2	2	Compact apartment, perfect for singles.	$900.00	Available	Edit Details Delete
3	801	4	2	Spacious family apartment.	$1,500.00	Occupied	Edit Details Delete
4	801	3	1	Recently renovated, modern interior.	$1,250.00	Maintenance	Edit Details Delete
11	804	2	1	Affordable housing with basic amenities.	$750.00	Available	Edit Details Delete
14	804	1	1	Small and economical.	$600.00	Available	Edit Details

Property Manager View for tracking Apartment Status



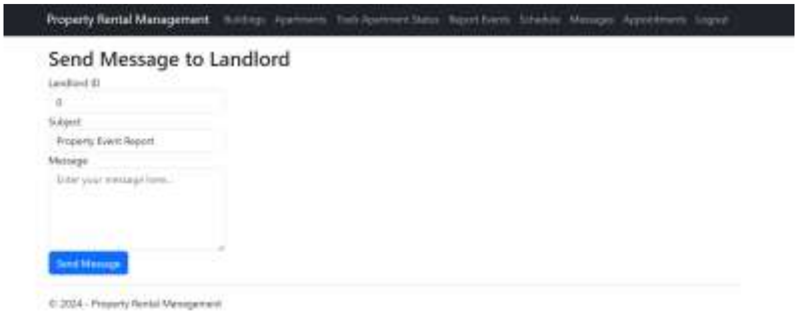
Track Apartments by Status

Filter by Status:

-- Select Status --

		Rooms	Bathrooms	Rent	Status
2	Downtown Apartments	2	2	$900.00	Available
3	Downtown Apartments	4	2	$1,500.00	Occupied
4	Downtown Apartments	3	1	$1,250.00	Maintenance
13	Suburban Homes	2	1	$750.00	Available
14	Suburban Homes	1	1	$600.00	Available
15	Suburban Homes	3	2	$1,100.00	Occupied
16	Suburban Homes	2	1	$800.00	Maintenance

Property Manager View to Report Events to the Landlord



Send Message to Landlord

Landlord ID

0

Subject

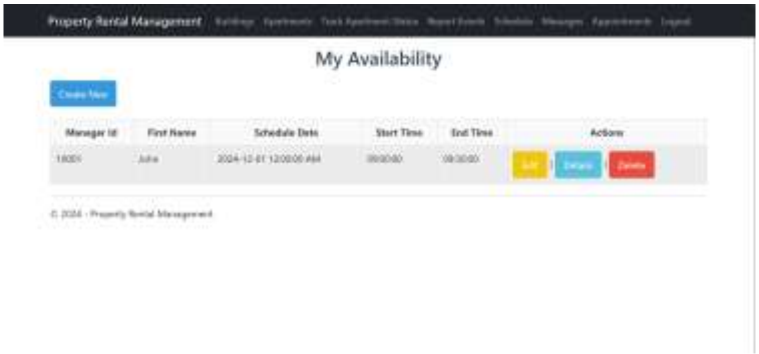
Property Event Report

Message

Enter your message here...

© 2024 - Property Rental Management

Property Manager View to create his created schedule for appointments

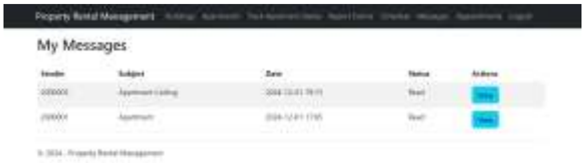


My Availability

Manager Id	First Name	Schedule Date	Start Time	End Time	Actions
18001	Julie	2024-12-01 12:00:00 AM	09:00:00	09:30:00	Edit | Details | Delete

© 2024 - Property Rental Management

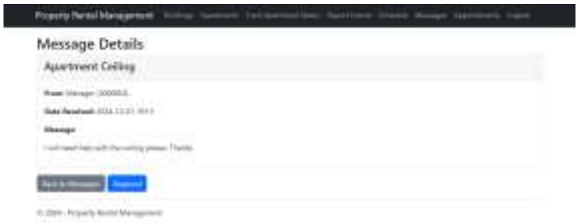
Property Manager View of their messages sent by potential tenants



My Messages

Sender	Subject	Date	Status	Actions
200001	Apartment Ceiling	2024-12-01 19:11	Read	View
200001	Apartment	2024-12-01 17:05	Read	View

© 2024 - Property Rental Management



Message Details

Apartment Ceiling

From: Manager (200001)

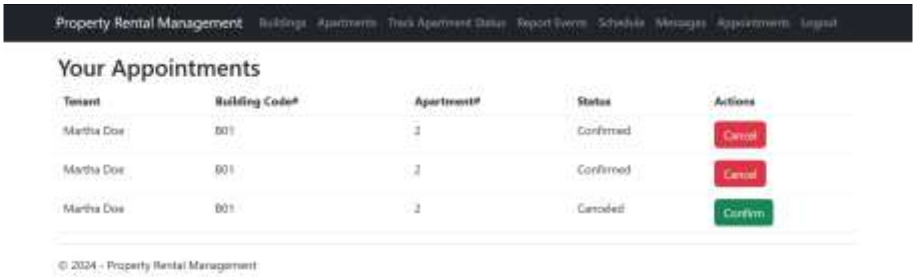
Date Received: 2024-12-01 19:11

Message:

I will need help with the ceiling please. Thanks

© 2024 - Property Rental Management

Property Manager View of their appointments

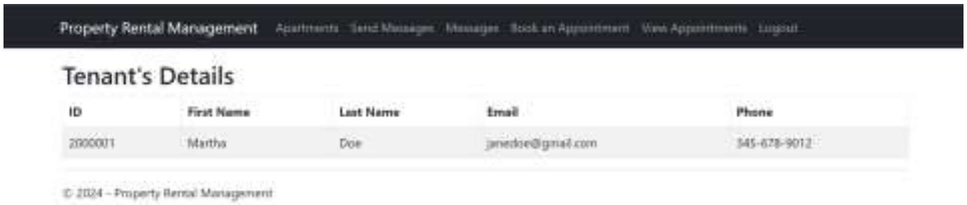


Your Appointments

Tenant	Building Code#	Apartment#	Status	Actions
Martha Doe	B01	2	Confirmed	Cancel
Martha Doe	B01	2	Confirmed	Cancel
Martha Doe	B01	2	Canceled	Confirm

© 2024 - Property Rental Management

Potential Tenant Dashboard – The logged in tenant sees only his/her details.

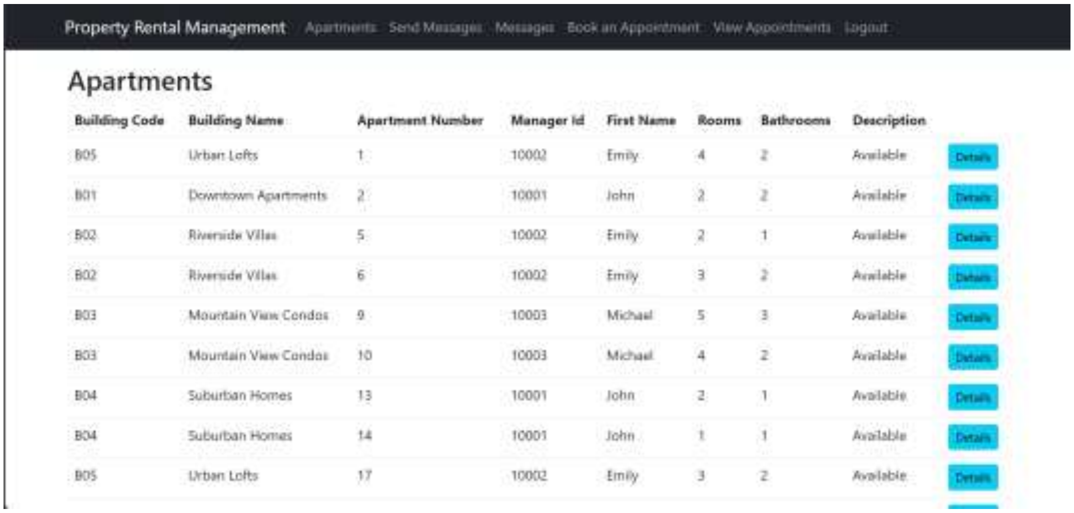


Tenant's Details

ID	First Name	Last Name	Email	Phone
2000001	Martha	Doe	janedoe@gmail.com	345-678-9012

© 2024 - Property Rental Management

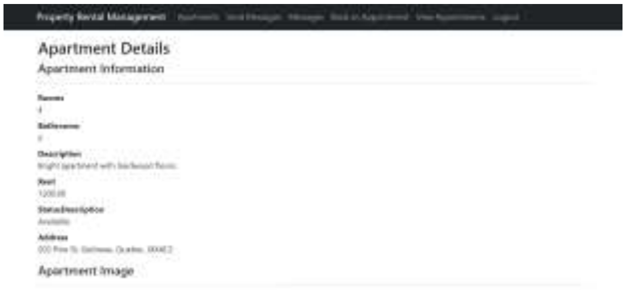
Tenant View of available apartments



Apartments

Building Code	Building Name	Apartment Number	Manager Id	First Name	Rooms	Bathrooms	Description	
B05	Urban Lofts	1	10002	Emily	4	2	Available	Details
B01	Downtown Apartments	2	10001	John	2	2	Available	Details
B02	Riverside Villas	5	10002	Emily	2	1	Available	Details
B02	Riverside Villas	6	10002	Emily	3	2	Available	Details
B03	Mountain View Condos	9	10003	Michael	5	3	Available	Details
B03	Mountain View Condos	10	10003	Michael	4	2	Available	Details
B04	Suburban Homes	13	10001	John	2	1	Available	Details
B04	Suburban Homes	14	10001	John	1	1	Available	Details
B05	Urban Lofts	17	10002	Emily	3	2	Available	Details

Tenant View of available apartments details



Apartment Details

Apartment Information

Rooms
4

Bathrooms
2

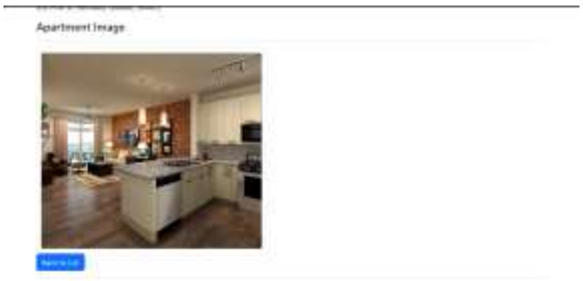
Description
Bright apartment with hardwood floors.

Rent
1200.00

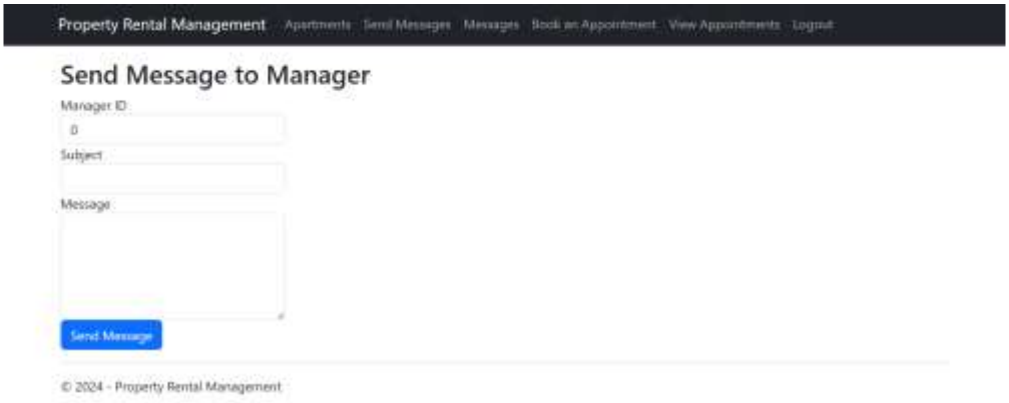
StatusDescription
Available

Address
[illegible] Pine St, Gatineau, Quebec, [illegible]

Apartment Image


Apartment Image

Tenant View to send messages to tenants



Send Message to Manager

Manager ID

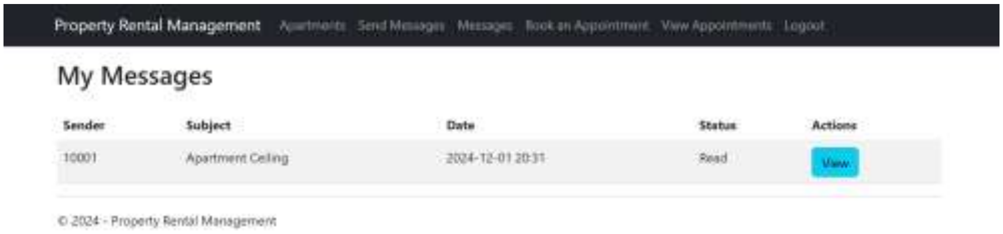
0

Subject

Message

© 2024 - Property Rental Management

Tenant View of messages sent to them by Property Manager

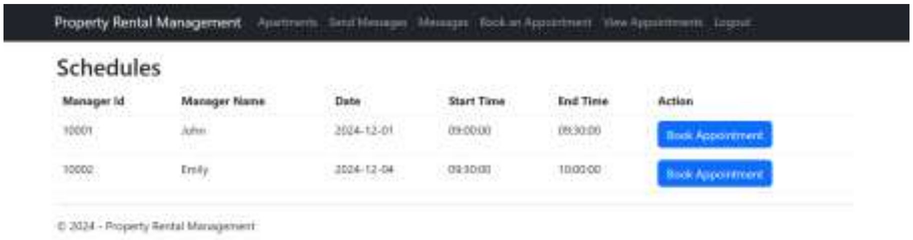


My Messages

Sender	Subject	Date	Status	Actions
10001	Apartment Ceiling	2024-12-01 20:31	Read	View

© 2024 - Property Rental Management

Tenant View of Manager’s availability for appointments

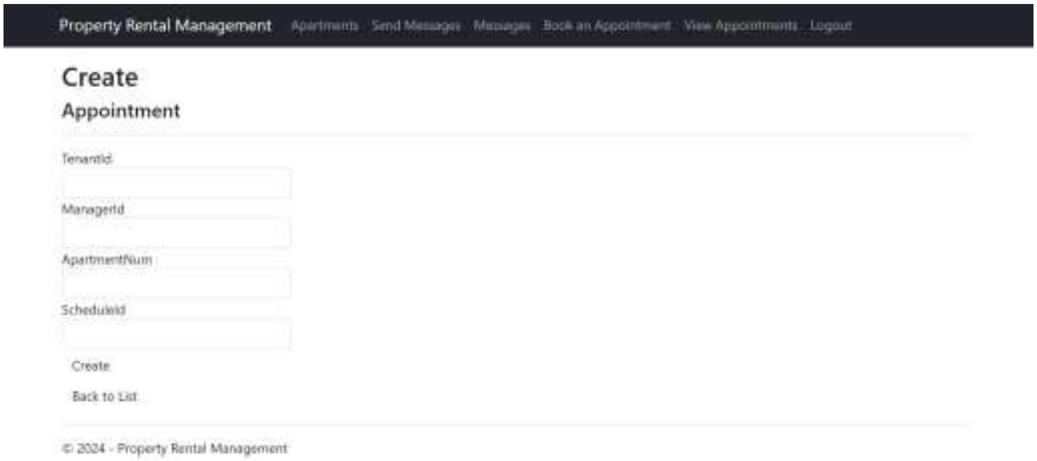


Schedules

Manager Id	Manager Name	Date	Start Time	End Time	Action
10001	John	2024-12-01	09:00:00	09:30:00	Book Appointment
10002	Emily	2024-12-04	09:30:00	10:00:00	Book Appointment

© 2024 - Property Rental Management

Tenant View to book/create an appointment based on manager’s schedule



Create

Appointment

TenantId

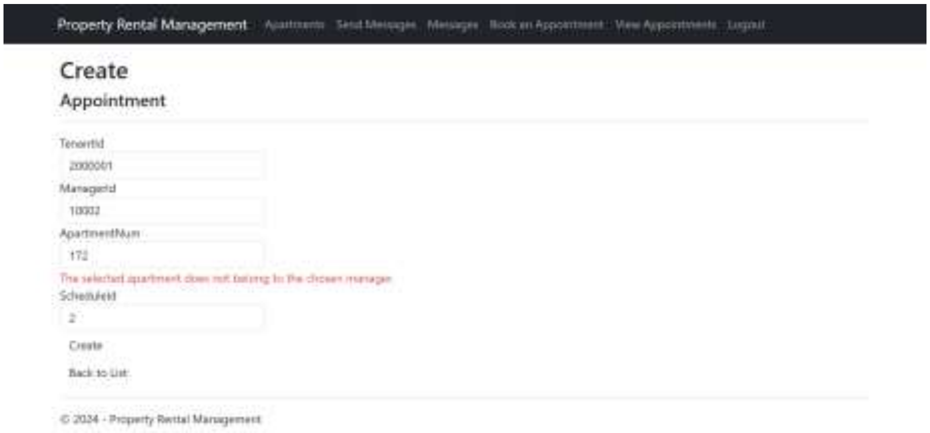
ManagerId

ApartmentNum

ScheduleId

© 2024 - Property Rental Management

Tenant View to exceptions handled when booking an appointment



Create

Appointment

TenantId: 2000001

ManagerId: 10002

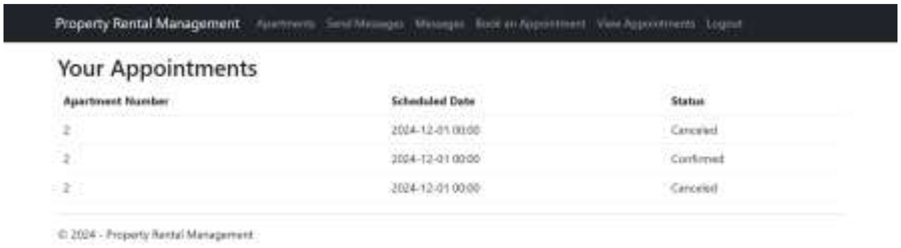
ApartmentNum: 172

The selected apartment does not belong to the chosen manager

ScheduleId: 2

© 2024 - Property Rental Management

Tenant View of appointment status



Your Appointments

Apartment Number	Scheduled Date	Status
2	2024-12-01 00:00	Canceled
2	2024-12-01 00:00	Confirmed
2	2024-12-01 00:00	Canceled

© 2024 - Property Rental Management

Users Login Credentials

1. Landlord:
   UserId: 3001
   Password: landlord3001

2. Manager:
   UserId: 10001
   Password: manager10001

3. Tenant:
   UserId: 2000001
   Password: tenantpass1

Phase IV Testing the Program

User	Functional Requirement	Test Result/Problem(s)
User (Potential Tenants, Managers, Landlords)	Sign in	Successful! Users can sign in and are redirected to their dashboards based on their user types.
Landlord	Access Restrictions	Successful! Landlord do not have access to managers and tenants' pages.
	Create/Update/Delete/Search/List any property manager account	Successful! Landlord can create, update, delete, list, and search property managers.
	Update/Delete/Search/List any potential tenant account	Successful! Landlord can update, delete, list, and search potential tenants.
	View Property Managers messages	Successful! Landlord can view property managers' messages, and the status of the message will change from unread to read. This is done through the Receive Messages and View Messages pages.

Property Manager	Access Restrictions	Successful! Property Managers do not have access to landlord and tenants' pages.
	Perform CRUD operations related to buildings	Successful! Property Managers can create, update, delete, and view their building details via the buildings page.
	Perform CRUD operations related to apartments	Successful! Property Managers can create, update, delete, and view their apartment details via the Apartments page.
	Keep track of apartments status	Successful! Property Managers can track the status of apartments (*available*, *maintenance*, or *occupied*) via the View Apartment by Status page using a filter.
	Schedule potential tenants' appointments	Successful! Property Managers can schedule appointments with tenants by making creating their schedule via the Create Schedule page and making it able for tenants to book appointment. Managers later see appointments booked by potential tenants and approves or denies via the View Appointments and Appointment Details pages.
	Respond to potential tenants' messages	Successful! Property Managers can view and respond to potential tenants' messages through the Receive Messages, View Message, and Respond to Tenants Messages page.
	Report any events to the property owner when necessary	Successful! Property managers can report events to the landlord through the Send Message to Landlord page.
Potential Tenants	Sign Up	Successful! New potential tenants can create an account using the signup page.
	Access Restrictions	Successful! Tenants do not have access to landlord and managers pages.

	View Available Apartments	Successful! Tenants can browse and view all available apartments directly from the Apartment Index page.
	Make an appointment with the Property Manager	Successful! Tenants can view the schedule of managers and book an appointment with them directly from the View Schedule and Book Appointment pages. Tenants can also view their appointment status if it is *pending*, *confirmed*, or *canceled*.
	Send necessary messages to the Property Manager	Successful! Tenants can send messages to property managers from the apartment from the Send Message to Manager page. They can also view the status of their messages if it is read or unread.
Guest	Access Restrictions	Successful! Guests do not have access to landlord, managers and tenants' pages.
	See Buildings	Successful! Guests can view some buildings and few details about the buildings via the About page. To see more details, they will need to sign up as potential tenants. They will be redirected to the sign-up page.

Summary of Test Results:

All functional requirements were tested and successful. The system handled operations well for all types of users. Though I faced minor issues like setting up a sign-up page, I overcame it by creating a SignupViewModel where user and tenant creation can be done at once. In addition, I had added to more tables in my database, that is: Leases and Payments. I did not have time to implement their functionalities in this project. Therefore, I will keep on working on this project to make it more personalized. No critical detects were identified, however, I plan on enhancing the user interface for a better experience.

III. Conclusion

The development of the Property Rental Management System (PRMS) offered invaluable learning experiences, increasing my proficiency in web development. Through this project, I gained a better understanding of key modern technologies including ASP.NET Entity Framework MVC and Razor Pages. The project reinforced my skills in database design, implementing Object-Relational Mapping (ORM), and building secure authentication and authorization mechanisms. The usage of Visual Studio 2022 and SQL Server Management Studio has refined my abilities to manage and develop complex applications.

IV. Bibliography

1. .NET Framework Documentation: What is .NET Framework? A software development framework.
2. Login Authentication/Authorization in ASP.NET Entity Framework MVC: Reference from Review for final exam.
3. ChatGPT: https://chatgpt.com/auth/login?sso