WeRent Homes – Real Estate Management Website

Author : Neha Kiran Nayak

Website Link: https://werent-homes-app.onrender.com/

Github Repo: https://github.com/NEHAKIRANNAYAK/WeRent-Homes/tree/main

---

1. Introduction

WeRent Homes is a web-based real estate management system designed to streamline interactions between renters and property agents while maintaining a well-structured relational database backend. The primary goal of this project is to demonstrate the practical application of database organization concepts such as schema design, relationships, constraints, and dynamic data updates within a real-world system.

The platform allows users to register, authenticate, browse properties, and perform role-specific actions. Agents can manage listings and renter interactions, while renters can view properties and submit requests. The system ensures data consistency, integrity, and scalability through proper database design and structured queries.

---

2. System Overview

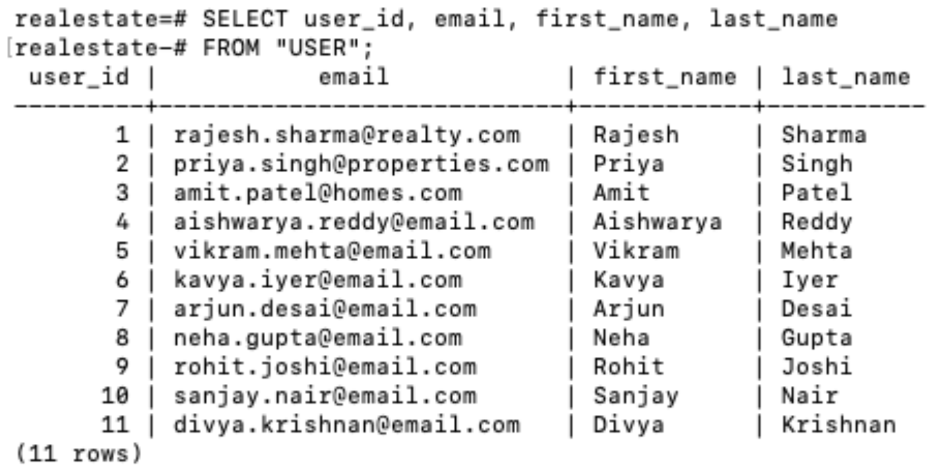
The WeRent Homes website consists of three main user roles:

- **Renters:** Registered users who can log in, view available properties, and interact with listings.
- **Agents:** Authorized users responsible for managing property listings and renter-related actions.
- **New Users:** Visitors who can register and become renters by creating an account.

The application is built using a client–server architecture where the frontend interacts with a backend server that communicates with a relational database. All user actions are reflected dynamically in the database.

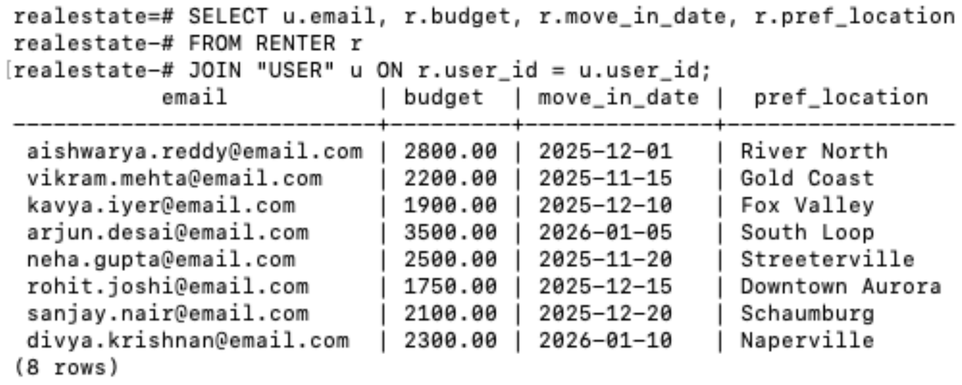
Sample Postgres queries to show the same:

1. Query to show all users:



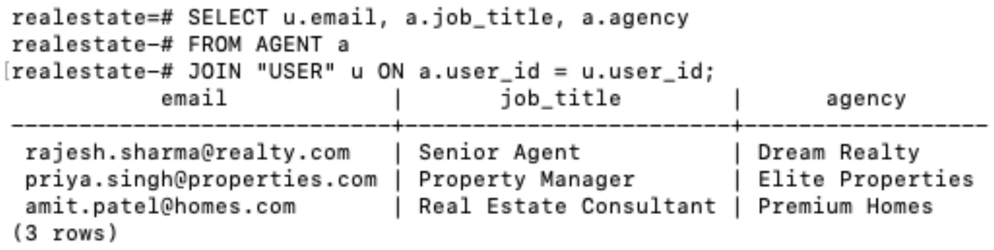
```
realestate=# SELECT user_id, email, first_name, last_name
[realestate-# FROM "USER";
 user_id |           email            | first_name | last_name
---------+----------------------------+------------+-----------
       1 | rajesh.sharma@realty.com   | Rajesh     | Sharma
       2 | priya.singh@properties.com | Priya      | Singh
       3 | amit.patel@homes.com       | Amit       | Patel
       4 | aishwarya.reddy@email.com  | Aishwarya  | Reddy
       5 | vikram.mehta@email.com     | Vikram     | Mehta
       6 | kavya.iyer@email.com       | Kavya      | Iyer
       7 | arjun.desai@email.com      | Arjun      | Desai
       8 | neha.gupta@email.com       | Neha       | Gupta
       9 | rohit.joshi@email.com      | Rohit      | Joshi
      10 | sanjay.nair@email.com      | Sanjay     | Nair
      11 | divya.krishnan@email.com   | Divya      | Krishnan
(11 rows)
```

2. Query to show all renters and their move in dates:



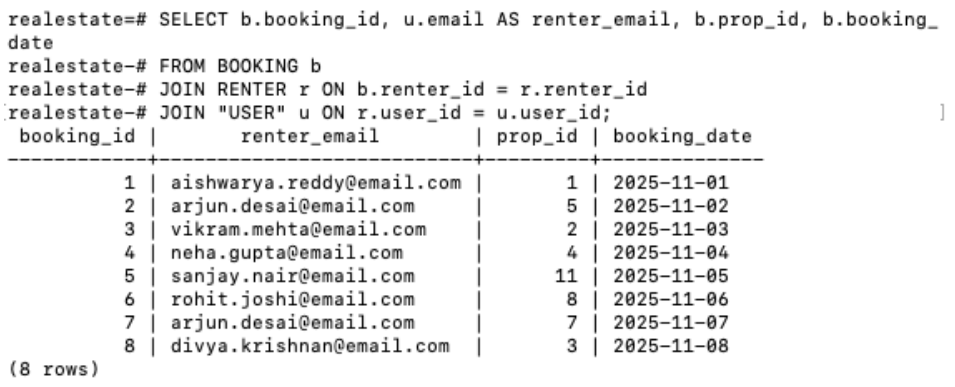
```
realestate=# SELECT u.email, r.budget, r.move_in_date, r.pref_location
realestate-# FROM RENTER r
[realestate-# JOIN "USER" u ON r.user_id = u.user_id;
           email           | budget  | move_in_date |  pref_location
---------------------------+---------+--------------+-----------------
 aishwarya.reddy@email.com | 2800.00 | 2025-12-01   | River North
 vikram.mehta@email.com    | 2200.00 | 2025-11-15   | Gold Coast
 kavya.iyer@email.com      | 1900.00 | 2025-12-10   | Fox Valley
 arjun.desai@email.com     | 3500.00 | 2026-01-05   | South Loop
 neha.gupta@email.com      | 2500.00 | 2025-11-20   | Streeterville
 rohit.joshi@email.com     | 1750.00 | 2025-12-15   | Downtown Aurora
 sanjay.nair@email.com     | 2100.00 | 2025-12-20   | Schaumburg
 divya.krishnan@email.com  | 2300.00 | 2026-01-10   | Naperville
(8 rows)
```

3. Query to show agents and the agency they work for:



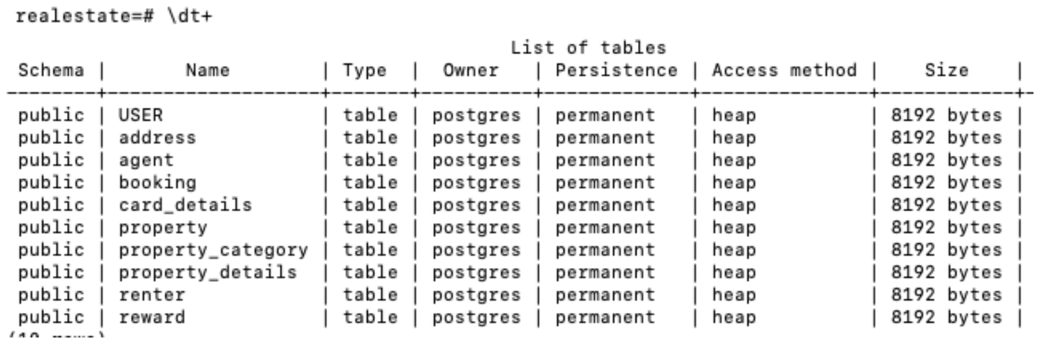
```
realestate=# SELECT u.email, a.job_title, a.agency
realestate-# FROM AGENT a
[realestate-# JOIN "USER" u ON a.user_id = u.user_id;
           email            |        job_title        |      agency
----------------------------+-------------------------+-------------------
 rajesh.sharma@realty.com   | Senior Agent            | Dream Realty
 priya.singh@properties.com | Property Manager        | Elite Properties
 amit.patel@homes.com       | Real Estate Consultant  | Premium Homes
(3 rows)
```

4. Query to show renter bookings:



```
realestate=# SELECT b.booking_id, u.email AS renter_email, b.prop_id, b.booking_
date
realestate-# FROM BOOKING b
realestate-# JOIN RENTER r ON b.renter_id = r.renter_id
realestate-# JOIN "USER" u ON r.user_id = u.user_id;
 booking_id |        renter_email         | prop_id | booking_date
------------+-----------------------------+---------+--------------
          1 | aishwarya.reddy@email.com   |       1 | 2025-11-01
          2 | arjun.desai@email.com       |       5 | 2025-11-02
          3 | vikram.mehta@email.com      |       2 | 2025-11-03
          4 | neha.gupta@email.com        |       4 | 2025-11-04
          5 | sanjay.nair@email.com       |      11 | 2025-11-05
          6 | rohit.joshi@email.com       |       8 | 2025-11-06
          7 | arjun.desai@email.com       |       7 | 2025-11-07
          8 | divya.krishnan@email.com    |       3 | 2025-11-08
(8 rows)
```

5. Query to get an overview of the schema



```
realestate=# \dt+
                                         List of tables
 Schema |        Name         | Type  |  Owner   | Persistence | Access method |    Size    |
--------+---------------------+-------+----------+-------------+---------------+------------+-
 public | USER                | table | postgres | permanent   | heap          | 8192 bytes |
 public | address             | table | postgres | permanent   | heap          | 8192 bytes |
 public | agent               | table | postgres | permanent   | heap          | 8192 bytes |
 public | booking             | table | postgres | permanent   | heap          | 8192 bytes |
 public | card_details        | table | postgres | permanent   | heap          | 8192 bytes |
 public | property            | table | postgres | permanent   | heap          | 8192 bytes |
 public | property_category   | table | postgres | permanent   | heap          | 8192 bytes |
 public | property_details    | table | postgres | permanent   | heap          | 8192 bytes |
 public | renter              | table | postgres | permanent   | heap          | 8192 bytes |
 public | reward              | table | postgres | permanent   | heap          | 8192 bytes |
```

3. Database Design

The database is designed using relational principles and normalization techniques to reduce redundancy and ensure data integrity. Core entities include users, renters, agents, properties, and transactions.

Key database concepts applied:

- Primary keys to uniquely identify records
- Foreign keys to establish relationships between tables
- Constraints to enforce valid data entries
- Separation of concerns between user roles

This structured design allows efficient querying and reliable updates while supporting multiple concurrent users.

4. Methodology

1. **Requirement Analysis**
   Functional requirements were identified for renters, agents, and new users. These requirements guided both the database schema and application logic.
2. **Database Schema Design**
   An Entity–Relationship (ER) model was created to represent entities and their relationships. The schema was then converted into relational tables following normalization rules.
3. **Backend Implementation**
   The backend handles authentication, role-based access, and database operations. SQL queries are used to perform Create, Read, Update, and Delete (CRUD) operations.
4. **Frontend Integration**
   User interfaces were connected to backend routes to ensure seamless interaction. Each user action triggers a corresponding database operation.
5. **Testing and Validation**
   The system was tested using multiple user accounts to verify correct role-based behavior and to confirm that database updates occur dynamically.

---

5. ER Diagram Explanation

The ER diagram models the core entities and relationships of the **WeRent Homes** real estate management system.

At the center is the **USER** entity, which represents all individuals using the system. A user can take on one of two specialized roles through *ISA* relationships: **AGENT** or **RENTER**, ensuring role-based behavior while avoiding data duplication.

An **AGENT** posts multiple **PROPERTY** listings (one-to-many relationship). Each property stores details such as location, size, price, availability, and amenities. Properties are further classified using the **PROPERTY_TYPE** entity, which captures category-specific attributes such as number of rooms, crime rate, and business type.

A **RENTER** can book properties through the **BOOKING** entity, representing the many-to-many relationship between renters and properties. Each booking records the renter, property, payment card, and booking date.

Renters make payments using **CARD_DETAILS**, where a renter can have multiple cards. Each booking references exactly one card, ensuring proper payment tracking.

The **REWARD** entity is linked one-to-one with **BOOKING**, representing reward points earned for each completed booking.

Overall, the design follows normalization principles, minimizes redundancy, and enforces referential integrity while clearly separating responsibilities across entities.

Implementation Highlights

- Secure login and registration functionality
- Role-based access control for renters and agents
- Dynamic database updates reflected immediately in the application
- Proper use of relational constraints to maintain data consistency

---

6. Results and Observations

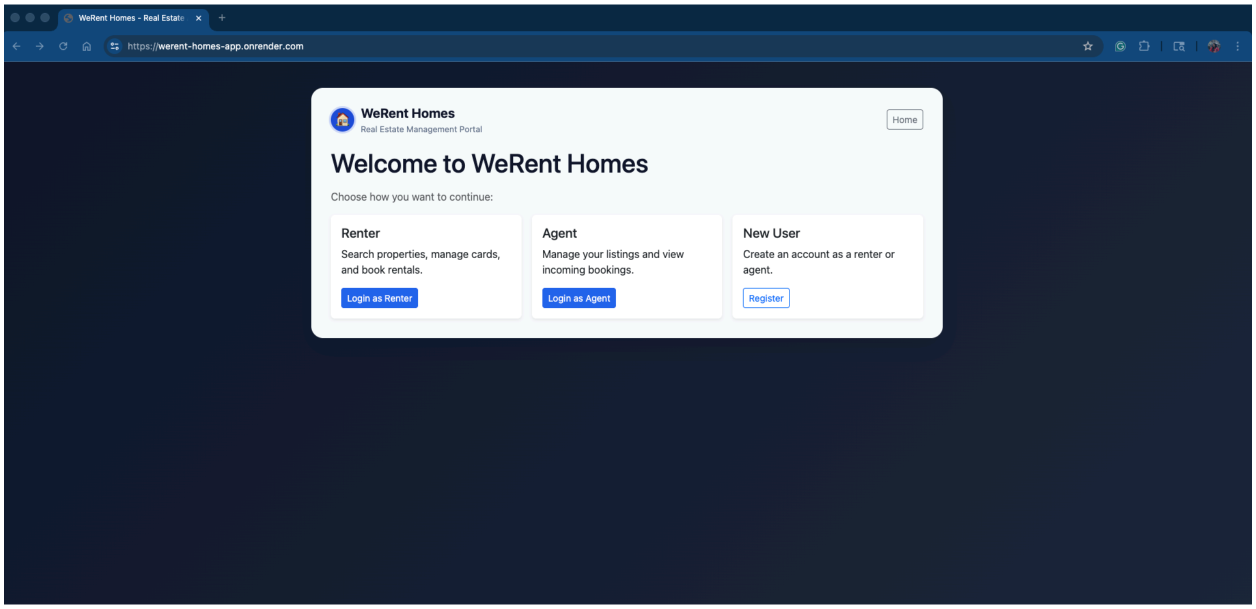
The implemented system successfully demonstrates core database organization principles. All user actions—such as registration, login, and data updates—are stored and retrieved accurately from the database. The relational design ensures minimal redundancy and efficient data access.

The project highlights how database design decisions directly affect application reliability and performance.

---

7. Screenshots

Screenshot 1: **Home Page and User Authentication (Login Interface)**


WeRent Homes - Real Estate
https://werent-homes-app.onrender.com
WeRent Homes
Real Estate Management Portal
Home
Login as Renter
Login as Agent
Register

Welcome to WeRent Homes

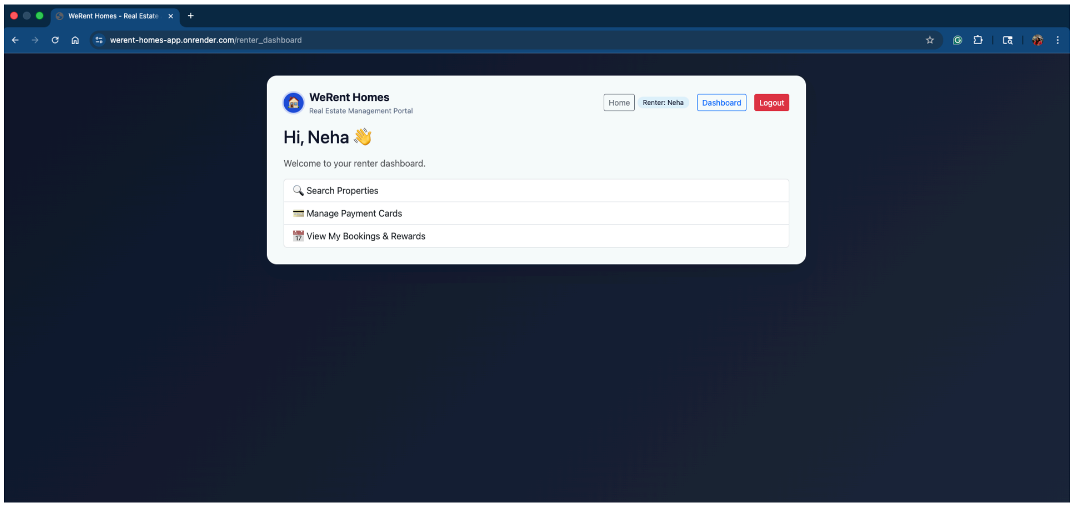
Choose how you want to continue:

Renter
Search properties, manage cards, and book rentals.

Agent
Manage your listings and view incoming bookings.

New User
Create an account as a renter or agent.

- Screenshot 2: Renter Dashboard
  - **Renter Dashboard Showing Property Search and Navigation Option**


WeRent Homes - Real Estate
werent-homes-app.onrender.com/renter_dashboard
Home
Renter: Neha
Dashboard
Logout

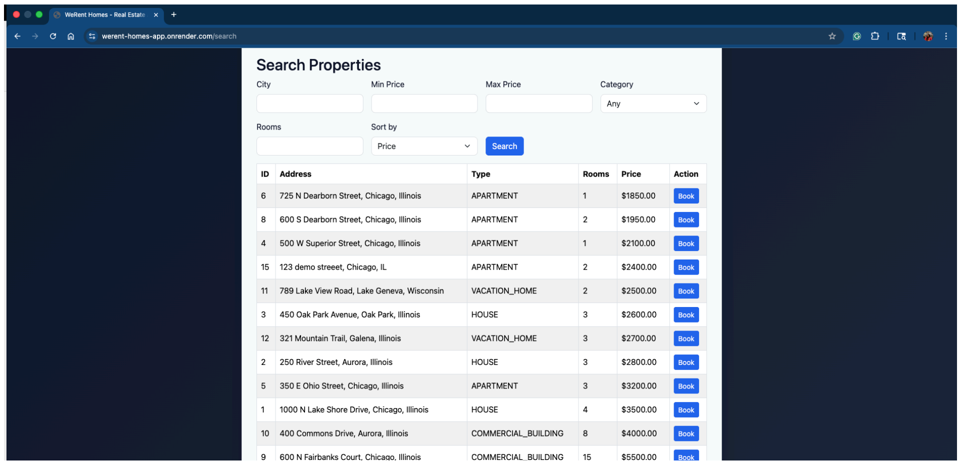
WeRent Homes
Real Estate Management Portal

Hi, Neha 👋

Welcome to your renter dashboard.

- 🔍 Search Properties
- 💳 Manage Payment Cards
- 📅 View My Bookings & Rewards

- **Property Search Results with Filters Applied (Renter View)**

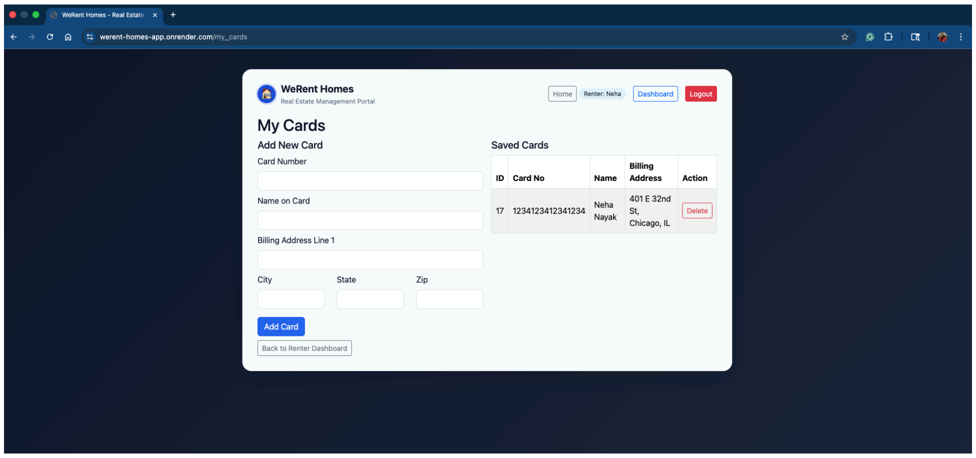

WeRent Homes - Real Estate
werent-homes-app.onrender.com/search
Search

Search Properties

City:
Min Price:
Max Price:
Category: Any
Rooms:
Sort by: Price

ID	Address	Type	Rooms	Price	Action
6	725 N Dearborn Street, Chicago, Illinois	APARTMENT	1	$1850.00	Book
8	600 S Dearborn Street, Chicago, Illinois	APARTMENT	2	$1950.00	Book
4	500 W Superior Street, Chicago, Illinois	APARTMENT	1	$2100.00	Book
15	123 demo streeet, Chicago, IL	APARTMENT	2	$2400.00	Book
11	789 Lake View Road, Lake Geneva, Wisconsin	VACATION_HOME	2	$2500.00	Book
3	450 Oak Park Avenue, Oak Park, Illinois	HOUSE	3	$2600.00	Book
12	321 Mountain Trail, Galena, Illinois	VACATION_HOME	3	$2700.00	Book
2	250 River Street, Aurora, Illinois	HOUSE	3	$2800.00	Book
5	350 E Ohio Street, Chicago, Illinois	APARTMENT	3	$3200.00	Book
1	1000 N Lake Shore Drive, Chicago, Illinois	HOUSE	4	$3500.00	Book
10	400 Commons Drive, Aurora, Illinois	COMMERCIAL_BUILDING	8	$4000.00	Book
9	600 N Fairbanks Court, Chicago, Illinois	COMMERCIAL_BUILDING	15	$5500.00	Book

- **Payment Card Management and Dynamic Card Addition (Renter View)**


WeRent Homes - Real Estate
werent-homes-app.onrender.com/my_cards
Home
Renter: Neha
Dashboard
Logout
Add Card
Back to Renter Dashboard

WeRent Homes
Real Estate Management Portal

My Cards

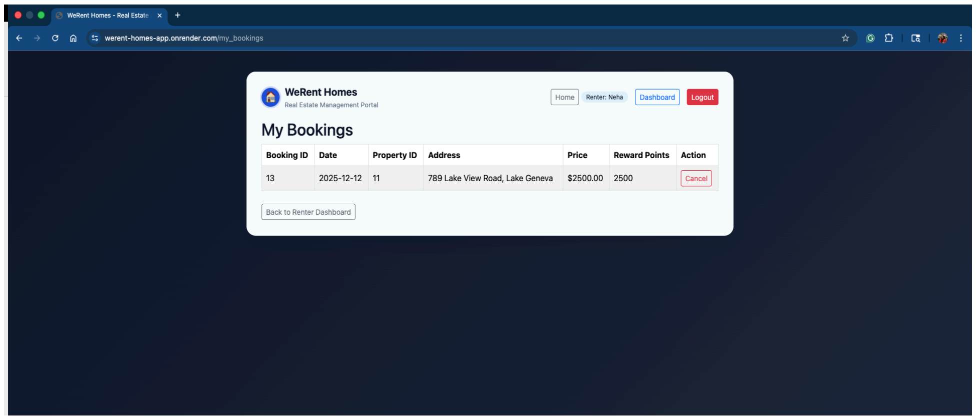
Add New Card

Card Number:
Name on Card:
Billing Address Line 1:
City:
State:
Zip:

Saved Cards

ID	Card No	Name	Billing Address	Action
17	1234123412341234	Neha Nayak	401 E 32nd St, Chicago, IL	Delete

- **Property Booking Confirmation and Reward Point Generation**



werent-homes-app.onrender.com/my_bookings

WeRent Homes
Real Estate Management Portal

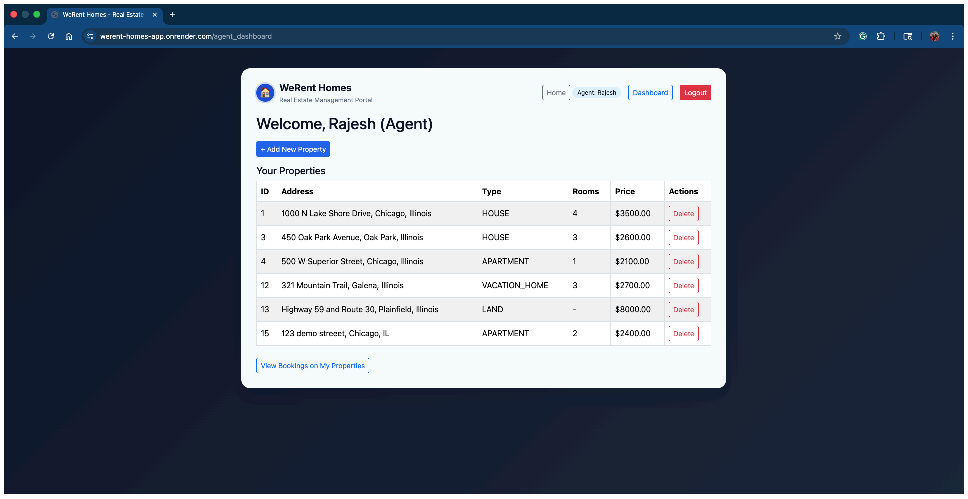
Home | Renter: Neha | Dashboard | Logout

My Bookings

Booking ID	Date	Property ID	Address	Price	Reward Points	Action
13	2025-12-12	11	789 Lake View Road, Lake Geneva	$2500.00	2500	Cancel

Back to Renter Dashboard

- Screenshot 3: Agent Dashboard
    - **Renter Booking History Showing Dynamic Updates**



werent-homes-app.onrender.com/agent_dashboard

WeRent Homes
Real Estate Management Portal

Home | Agent: Rajesh | Dashboard | Logout

Welcome, Rajesh (Agent)

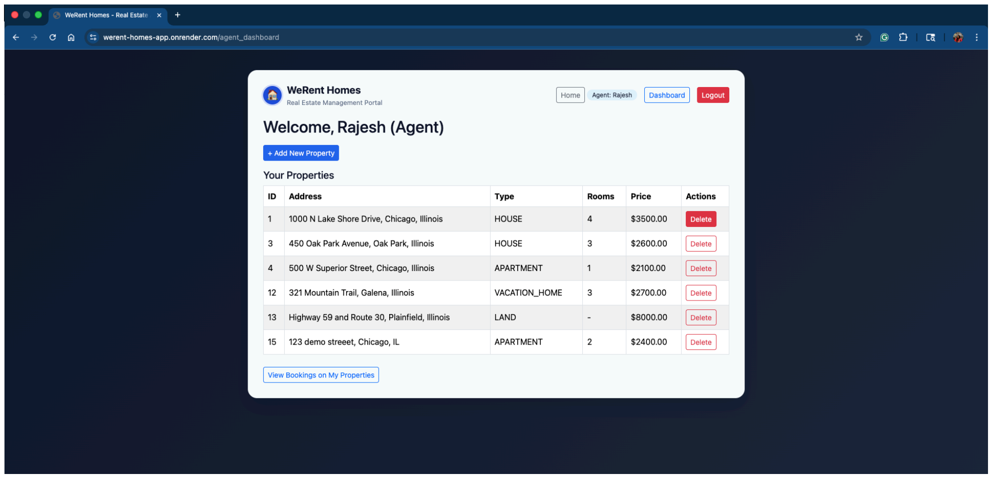
+ Add New Property

Your Properties

ID	Address	Type	Rooms	Price	Actions
1	1000 N Lake Shore Drive, Chicago, Illinois	HOUSE	4	$3500.00	Delete
3	450 Oak Park Avenue, Oak Park, Illinois	HOUSE	3	$2600.00	Delete
4	500 W Superior Street, Chicago, Illinois	APARTMENT	1	$2100.00	Delete
12	321 Mountain Trail, Galena, Illinois	VACATION_HOME	3	$2700.00	Delete
13	Highway 59 and Route 30, Plainfield, Illinois	LAND	-	$8000.00	Delete
15	123 demo streeet, Chicago, IL	APARTMENT	2	$2400.00	Delete

View Bookings on My Properties

- **Renter Booking History Showing first row deleted**



werent-homes-app.onrender.com/agent_dashboard

WeRent Homes
Real Estate Management Portal

Home | Agent: Rajesh | Dashboard | Logout

Welcome, Rajesh (Agent)

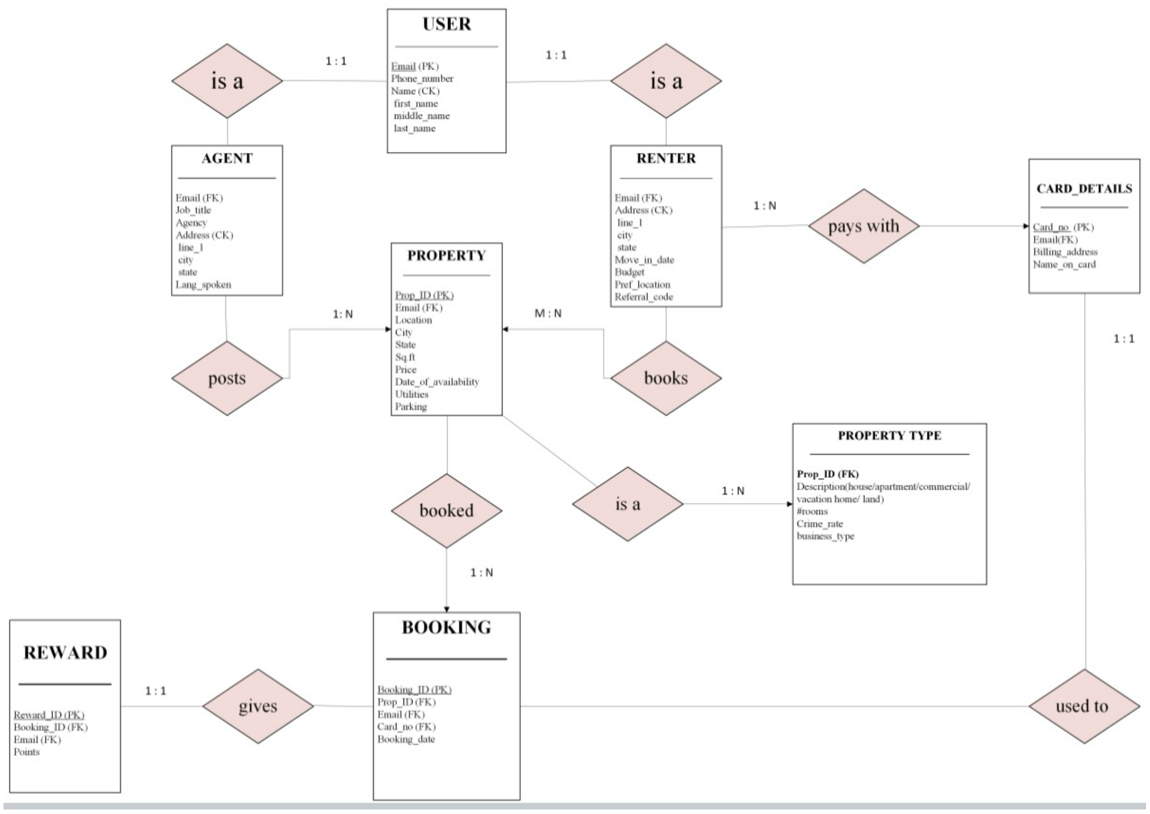
+ Add New Property

Your Properties

ID	Address	Type	Rooms	Price	Actions
1	1000 N Lake Shore Drive, Chicago, Illinois	HOUSE	4	$3500.00	Delete
3	450 Oak Park Avenue, Oak Park, Illinois	HOUSE	3	$2600.00	Delete
4	500 W Superior Street, Chicago, Illinois	APARTMENT	1	$2100.00	Delete
12	321 Mountain Trail, Galena, Illinois	VACATION_HOME	3	$2700.00	Delete
13	Highway 59 and Route 30, Plainfield, Illinois	LAND	-	$8000.00	Delete
15	123 demo streeet, Chicago, IL	APARTMENT	2	$2400.00	Delete

View Bookings on My Properties

- **Screenshot 4: ER Design**



---

8. Conclusion

WeRent Homes serves as a practical example of applying database organization concepts in a real-world web application. Through structured schema design, proper use of constraints, and dynamic data handling, the project demonstrates how databases support scalable and reliable systems.

This project reinforces the importance of thoughtful database design in building robust applications and provides hands-on experience with concepts covered in CS 425.
