Property Rental System

Team:

1. **Chirag Poornamath - 49**
2. **Prakash Thapa - 60**
3. **Jai Udasi - 62**
4.

Introduction

The Property Rental System is designed to facilitate the rental process between property owners and tenants.
Built using Java, this system aims to streamline the search and management of rental properties, making it easier for users to find and book accommodation.

Problem Statement

Traditionally individuals seeking rental properties must physically search through physical listings, which is time consuming and inefficient. The proposed system eliminates this inconvenience by allowing users to search properties online.

Objectives

Payment Processing

Facilitate rent payments and issue reporting through the application.

Display Available Rentals

Showcase a list of houses available for rent in a specified area.

Property Selection

Enable users to select desired properties from the displayed list.

Project Features

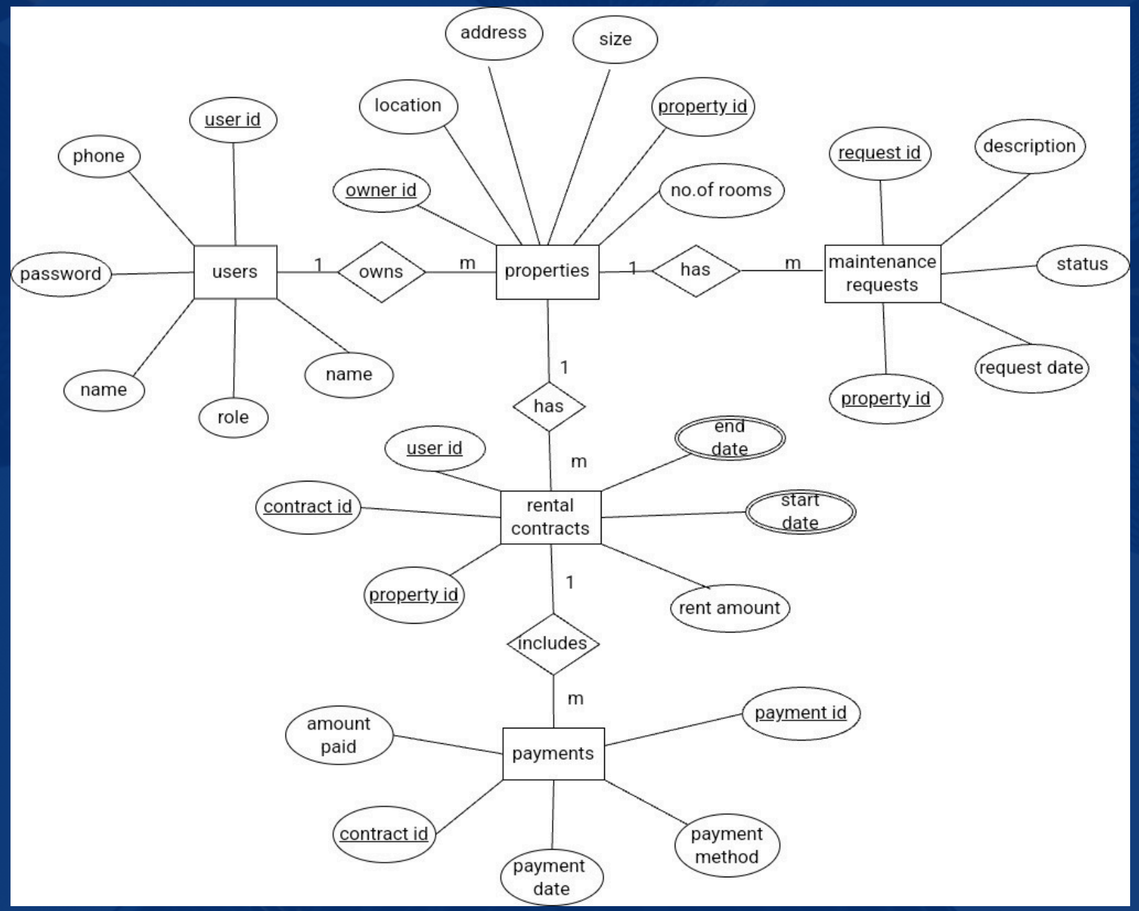
01 User registration and login

02 Rent payment processing

03 Search functionality for houses

04 Issue reporting feature

ER Diagram


address
size
location
property id
user id
phone
owner id
no.of rooms
request id
description
password
users
1
owns
m
properties
1
has
m
maintenance requests
status
name
name
role
1
has
request date
property id
user id
end date
m
contract id
rental contracts
start date
property id
1
rent amount
includes
m
amount paid
payments
payment id
contract id
payment date
payment method

Relational Schema

Users					
userid	name	email	password	role	phone

RentalContracts					
contractid	propertyid	userid	startdate	enddate	rentamount

Properties							
propertyid	ownerid	address	rent	size	numberofrooms	availabilitystatus	location

Payments				
paymentid	contractid	paymentdate	amountpaid	paymentmethod

MaintenanceRequests				
requestid	propertyid	requestdate	description	status

SQL Queries

select * from properties;

propertyid [PK] integer	ownerid integer	address character varying (255)	rent numeric (10,2)	size integer	numberofrooms integer	availabilitystatus character varying (20)	location character varying (100)

select * from users;

userid [PK] integer	name character varying (100)	email character varying (100)	password character varying (100)	role character varying (50)	phone character varying (15)

select * from RentalContracts;

contractid [PK] integer	propertyid integer	userid integer	startdate date	enddate date	rentamount numeric (10,2)

select * from Payments;

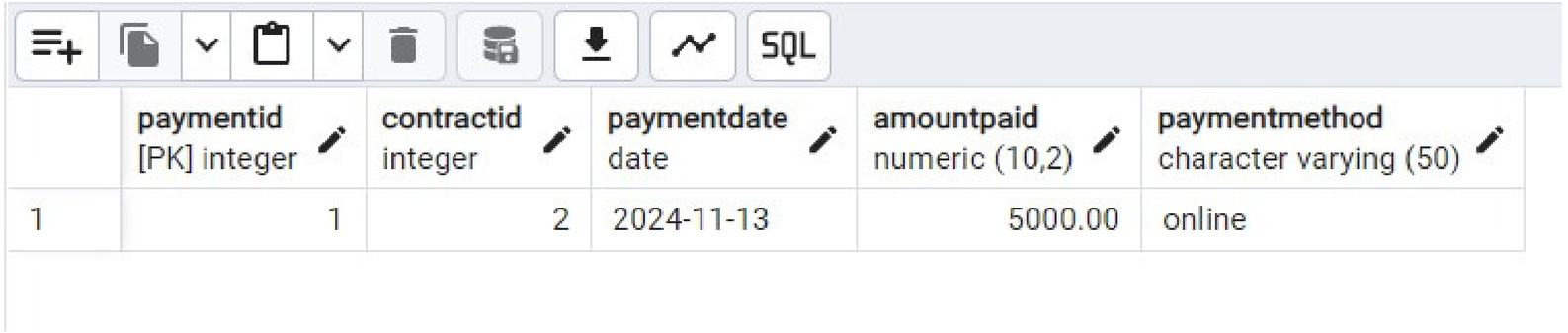
paymentid [PK] integer	contractid integer	paymentdate date	amountpaid numeric (10,2)	paymentmethod character varying (50)

select * from MaintenanceRequests;

requestid [PK] integer	propertyid integer	requestdate date	description text	status character varying (50)

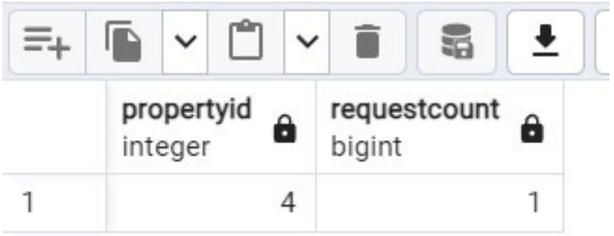
CORRELATED/COMPLEX QUERIES

```
SELECT P.*
FROM Payments P
WHERE P.ContractID IN (SELECT RC.ContractID FROM RentalContracts RC WHERE
RC.UserID = 4);
```



	paymentid [PK] integer	contractid integer	paymentdate date	amountpaid numeric (10,2)	paymentmethod character varying (50)
1	1	2	2024-11-13	5000.00	online

```
SELECT PropertyID, COUNT(*) AS RequestCount
FROM MaintenanceRequests
GROUP BY PropertyID;
```



	propertyid integer	requestcount bigint
1	4	1

NESTED QUERIES

SELECT * FROM Users
WHERE UserID IN (SELECT UserID FROM RentalContracts);

	userid [PK] integer	name character varying (100)	email character varying (100)	password character varying (100)	role character varying (50)	phone character varying (15)
1	4	Jai Udasi	jaiudasi21@gmail.com	goa	owner	5522110095

SELECT P.PropertyID, P.Address, RC.RentAmount
FROM Properties P
JOIN RentalContracts RC ON P.PropertyID = RC.PropertyID;

	propertyid integer	address character varying (255)	rentamount numeric (10,2)
1	4	789 Elm St, Village	20000.00

VIEWS

```
CREATE VIEW AvailableProperties AS
SELECT * FROM Properties WHERE AvailabilityStatus = 'Available';
```

Data Output | Messages | Notifications

```
CREATE VIEW

Query returned successfully in 171 msec.
```

```
CREATE VIEW UserPayments AS
SELECT U.Name, P.AmountPaid, P.PaymentDate
FROM Users U
JOIN RentalContracts RC ON U.UserID = RC.UserID
JOIN Payments P ON RC.ContractID = P.ContractID;
```

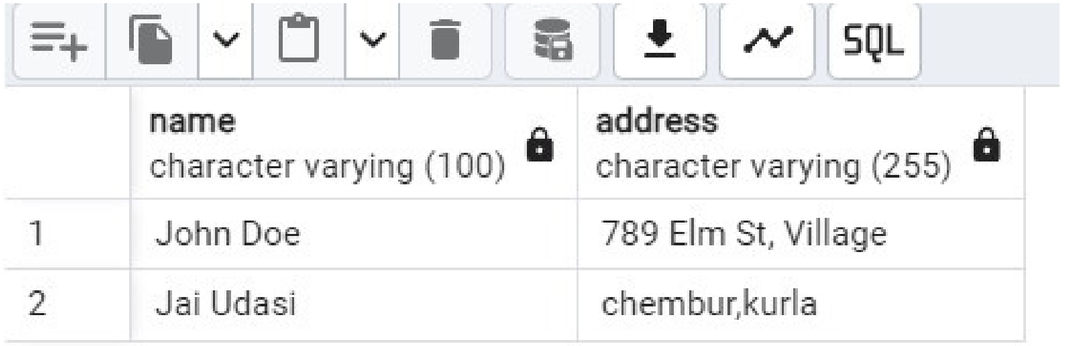
Data Output | Messages | Notifications

```
CREATE VIEW

Query returned successfully in 180 msec.
```

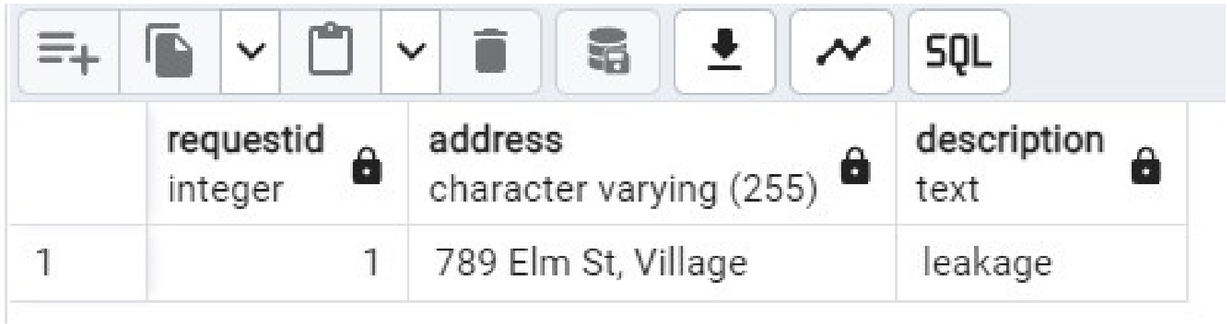
JOINS

```
SELECT U.Name, P.Address
FROM Users U
JOIN Properties P ON U.UserID = P.OwnerID;
```



	name character varying (100)	address character varying (255)
1	John Doe	789 Elm St, Village
2	Jai Udasi	chembur,kurla

```
SELECT MR.RequestID, P.Address, MR.Description
FROM MaintenanceRequests MR
LEFT JOIN Properties P ON MR.PropertyID = P.PropertyID;
```



	requestid integer	address character varying (255)	description text
1	1	789 Elm St, Village	leakage

Normalization

The given schema is generally in 3rd Normal Form (3NF), which means:

01 1.1NF (First Normal Form): All tables have atomic columns (e.g., a user's name is atomic).

02 2NF (Second Normal Form): All non-key attributes in a table are fully functionally dependent on the primary key

03 1.3NF (Third Normal Form): There are no transitive dependencies (i.e., no non-key attribute depends on another non-key attribute).

The structure has separate tables for Users, Properties, RentalContracts, Payments, and MaintenanceRequests, with foreign key constraints to maintain relationships, so the schema seems appropriately normalized up to 3NF.

Conclusion

The Property Rental System represents a significant advancement in how rental properties are managed and accessed. By leveraging technology, it addresses common pain points in the rental market, offering a user-friendly solution for both property owners and renters. This project not only showcases technical skills in Java programming but also provides a technical tool that meets real-world needs in property management.

Thank You