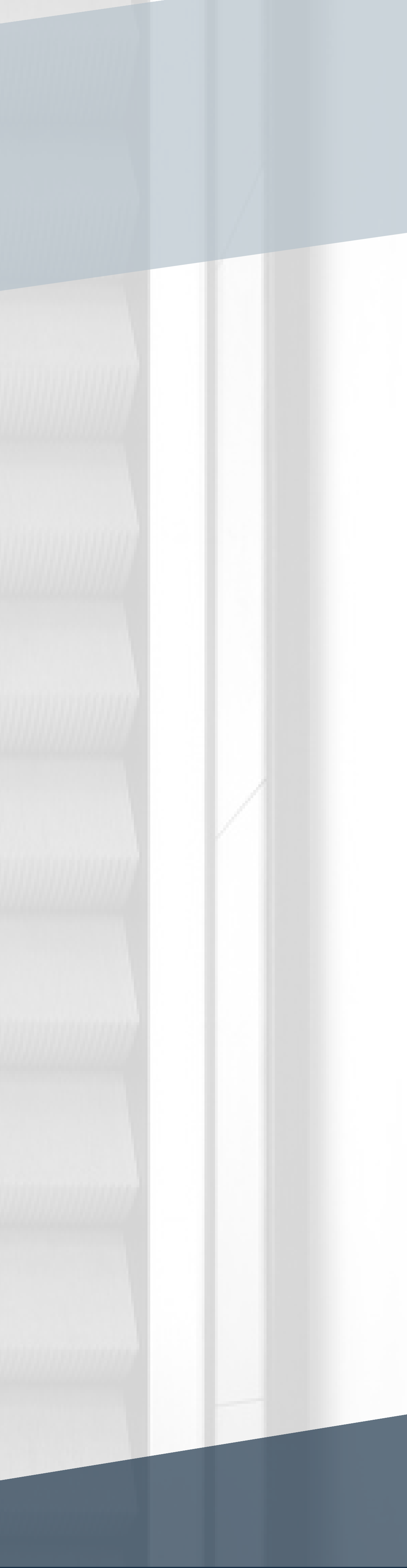


DREAM HOME

REPORT

DATA DYNAMICS

TABLE OF CONTENTS

INTRODUCTION

DreamHome is a property management company connecting property owners with clients who need to rent furnished properties for a fixed period. With 100 branches and over 2000 staff members, DreamHome offers services such as property management, rental services, and advertising to ensure that properties are rented out for maximum return. Key performance indicators show successful growth and increased market share. To continue its success, DreamHome must focus on providing high-quality services and investing in technology and software for streamlined operations and better customer service.

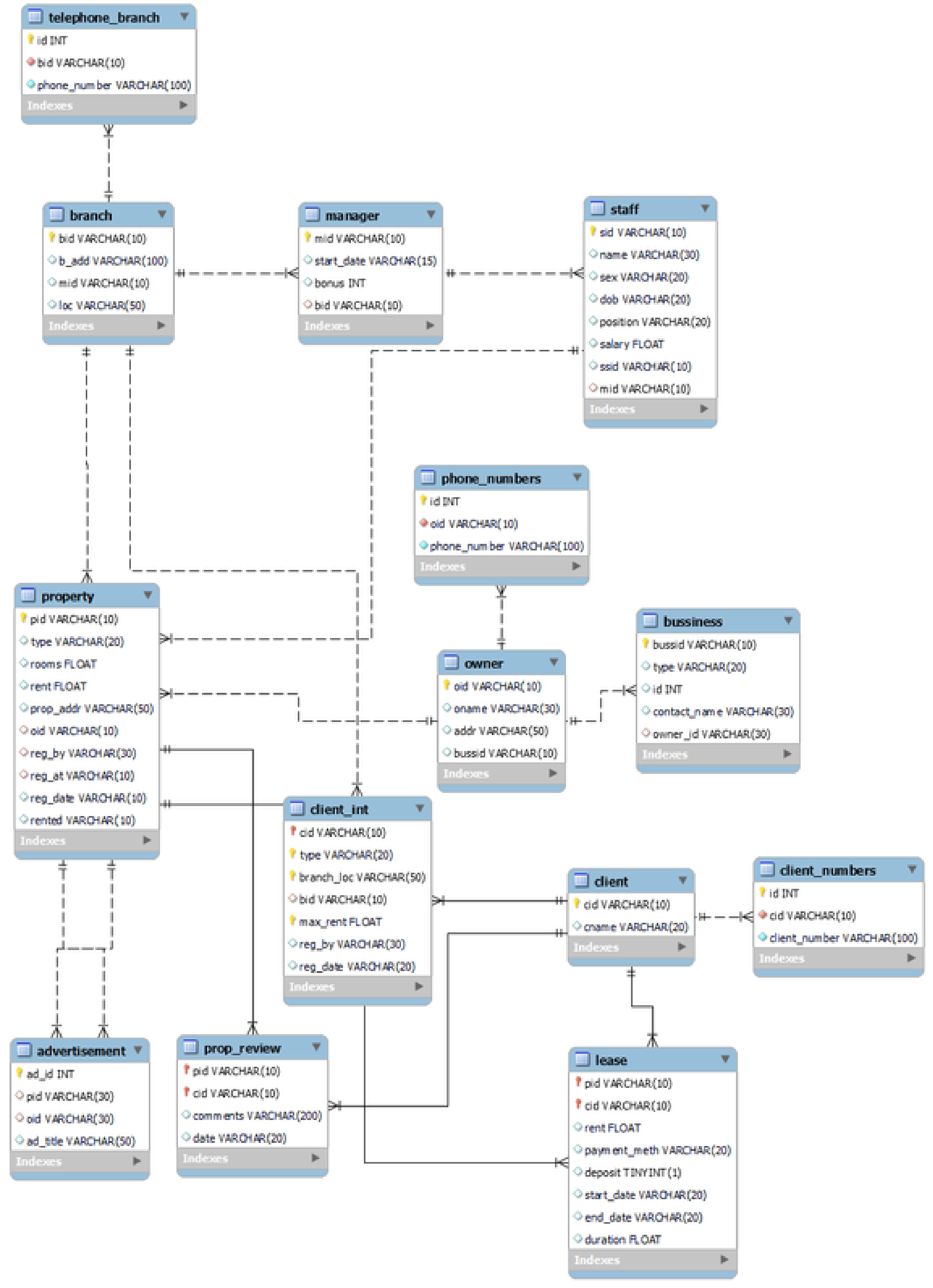
OUR GOALS

1. To provide high-quality property management services to property owners, ensuring that their properties are rented out for maximum return.
2. To offer a wide range of rental properties to clients, meeting their diverse needs and preferences.
3. To maintain a strong reputation in the property management industry through professionalism, integrity, and excellent customer service.
4. To expand the company's market share and increase profitability through strategic growth initiatives.
5. To leverage technology and software to streamline operations and enhance the customer experience.

PROBLEM STATEMENT

DreamHome needs an efficient database to store its large amount of data related to property owners, rental properties, and clients in a centralized system to improve operational efficiency, reduce errors, and enhance the customer experience, while enabling informed decision-making to drive growth and profitability.

Proposed Solution!!



Branch:

- Branch consists of Branch_Id(bid) and its address(b_add) the manager Id (mid) and Location of the branch
- mid and bid are unique for each branch

Telephone_branch:

- This table is to store multiple contact info of a branch
- It has auto-increment primary key as Id
- bid attribute references bid of the branch table
- phone_numbers to store contact Information of the branch

Manager:

- Manager table consists of Manager Id(mid), Start_Date of manager and the bonus of the manager Position and bid which is corresponding Branch Id for the manager
- Here mid is unique and the bid is the foreign key which references the attribute bid of Branch table.

Staff:

- Staff Table consists of sid, name, sex, DOB, position, salary, ssid, mid
- sid is the unique key for the staff table and ssid is the superior of the sid
- supervisor and manager has null as ssid .
- manager mid is same as itself , we cannot make it null as it is foreign key that references mid of Manager table.

Owner:

- Owner table consists of Oid, oName(Owner Name), Addr(address of the owner), bussid
- bussid by default 00 , if the owner has any bussiness it gets updated to 01
- Oid is the unique attribute of this Owner table .

Bussiness:

- If the owner has any Bussiness it gets redirected to this table
- It has bussid, type(Type of the bussiness), id, contact_name, and owner_id
- owner_id references to oid of the owner table .

Phone_numbers:

- This table is to store multiple contact info of a owners.
- It has auto-increment primary key as Id
- oid attribute references oid of the owner table
- phone_number to store contact Information of the owner

Property:

- This table consists of properties registered by owner.
- pid(property Id), type(type of the property), rooms, rent, prop_addr(Address of the property), oid(owner id), reg_by(the sid of the staff who registered the property), reg_at(at which branch it is registered), reg_date(date of property registration), rented(by default it is 'no' but if the property is in lease then it will be 'yes')
- pid is the unique attribute
- oid references owner table oid attribute and reg_at references branch bid and reg_by references staff sid

Client:

- Client table consists of cid and cName(ClientName)
- cid is the unique attribute of this client table.

Client_numbers:

- This table is to store multiple contact info of a clients.
- It has auto-increment primary key as Id
- cid attribute references cid of the client table
- client_number is to store contact Information of the client

Client_interest:

- Client interest table consists of cid, type(of property), branch_loc(where property is registered), bid (corresponding branch id), max_rent, reg_by(name of staff) and reg_date
- Here we have composite key of cid, type, branch_loc and max_rent so we get unique details about clients.
- bid is the foreign key that references to the unique bid of branch.

Advertisement:

- This table shows all advertisements published.
- Here, there is ad_id, pid, oid and ad_title (description).
- The ad_id is primary key where as pid and oid are foreign keys referencing to pid of property and oid of owner respetively.

Property_review:

- This table stores the reviews done by clients on different properties.
- it has pid, oid, comments, date(the date of review)
- pid and oid are both are used together to give a uniqueness to this table.
- pid references property pid and oid references owner oid.

Lease:

- This table consists of the lease drawn after the owner and client is satisfied with each other conditions.
- It consists of pid, cid, rent, payment_method, deposit, start_date, end_date, duration(duration of the lease).
- pid and oid are both are used together to give a uniqueness to this table.
- pid references property pid and oid references owner oid.

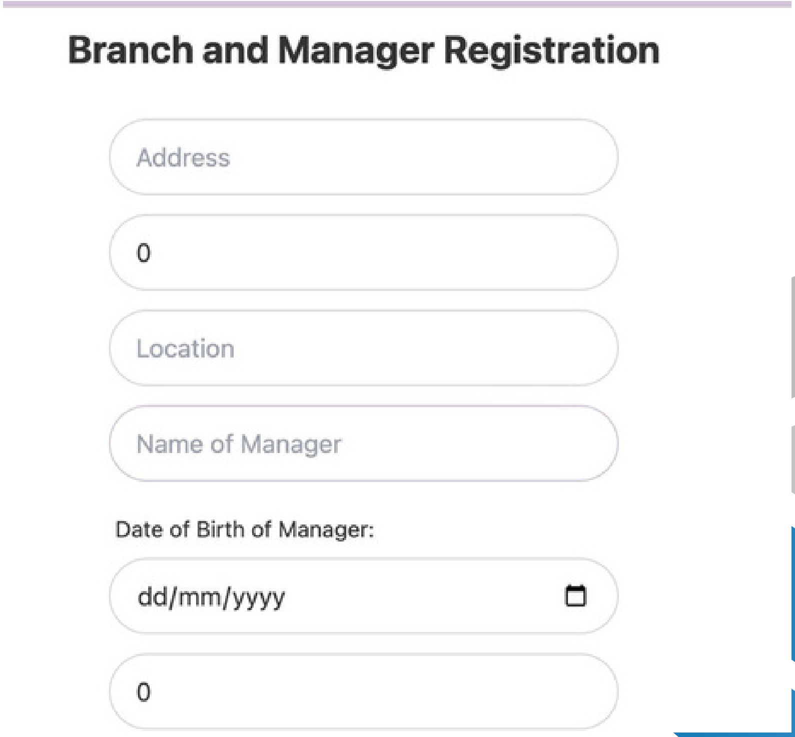
Interface

We created a interface with React and Tailwind css that will float throughout the branches of Dream Home, allowing them to simply make everything online and sophisticated, while the data of all branches will be in the hands of database administrators, reducing the fear of data loss and increasing data availability.

LINK: https://github.com/parikshith078/dream-home

To access our interface of our database .This is the git repository

- We've made it such that the ids are created at random; there's no need to take input anywhere except for property, and it's much more convenient that way.
- In our database you cannot add same property twice because we used the property id as input rather than a randomly generated number for only the property form, and the pid is the main key of the property table, adding the same property twice constitutes a violation.
- Furthermore, we designed our database schema with 14 tables to reduce data redundancy while also increasing database efficiency and flexibility.



Branch and Manager Registration

Address

0

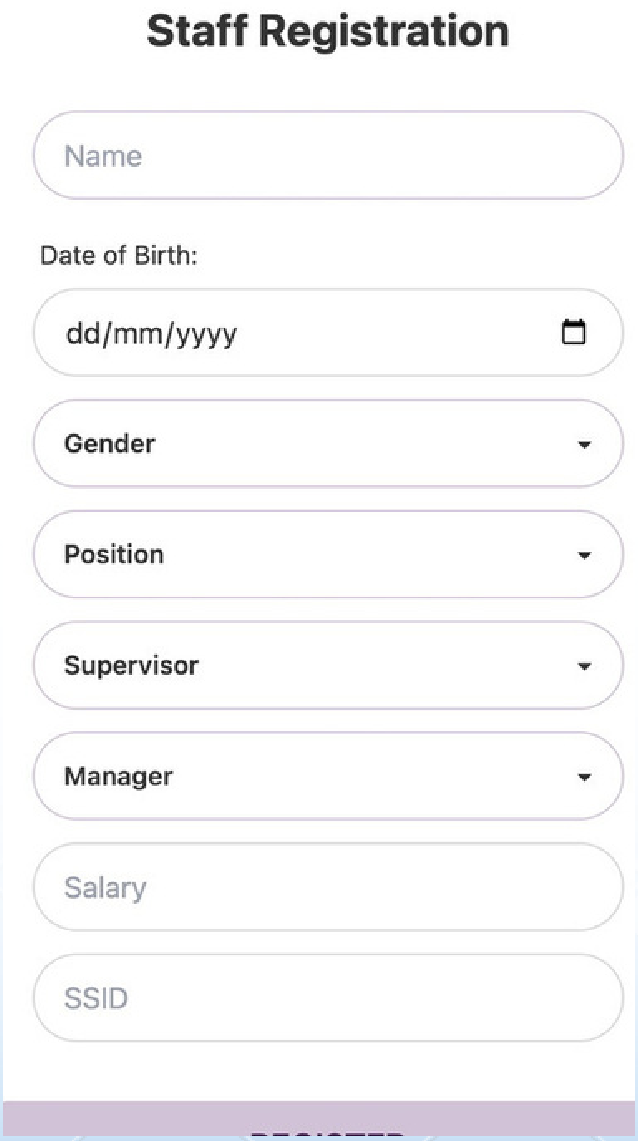
Location

Name of Manager

Date of Birth of Manager:

dd/mm/yyyy

0



Staff Registration

Name

Date of Birth:

dd/mm/yyyy

Gender

Position

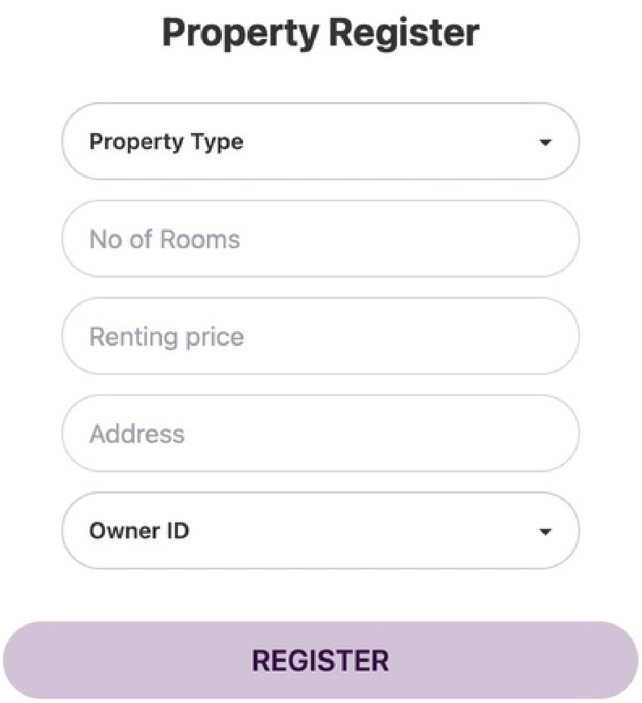
Supervisor

Manager

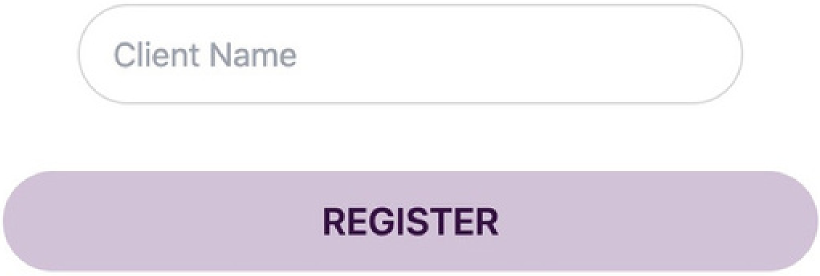
Salary

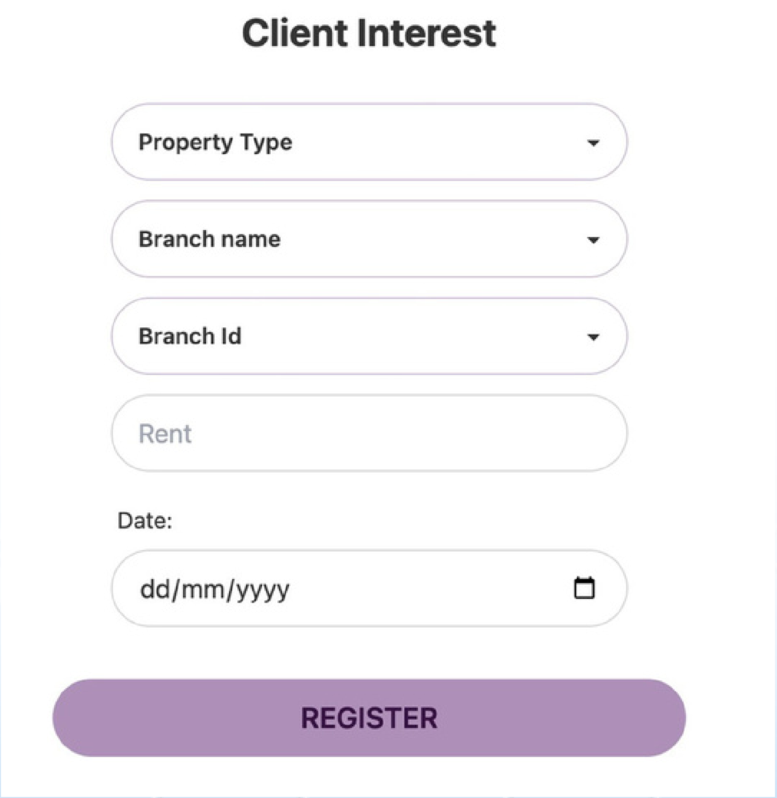
SSID

REGISTER

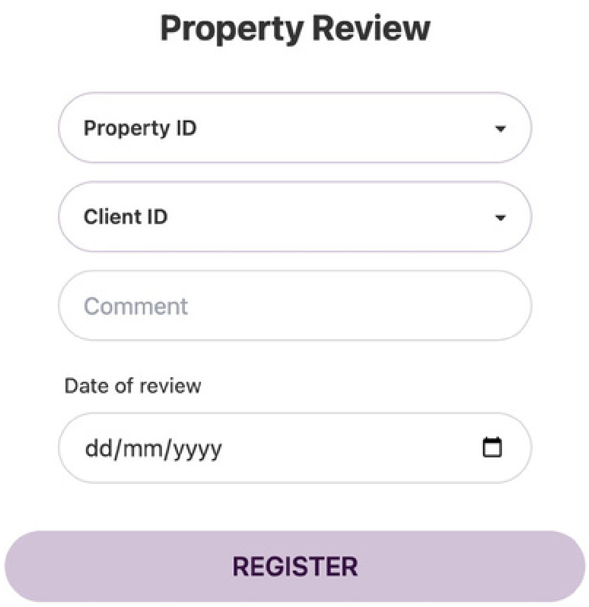


Property Register





Client Interest



Property Review

Property ID

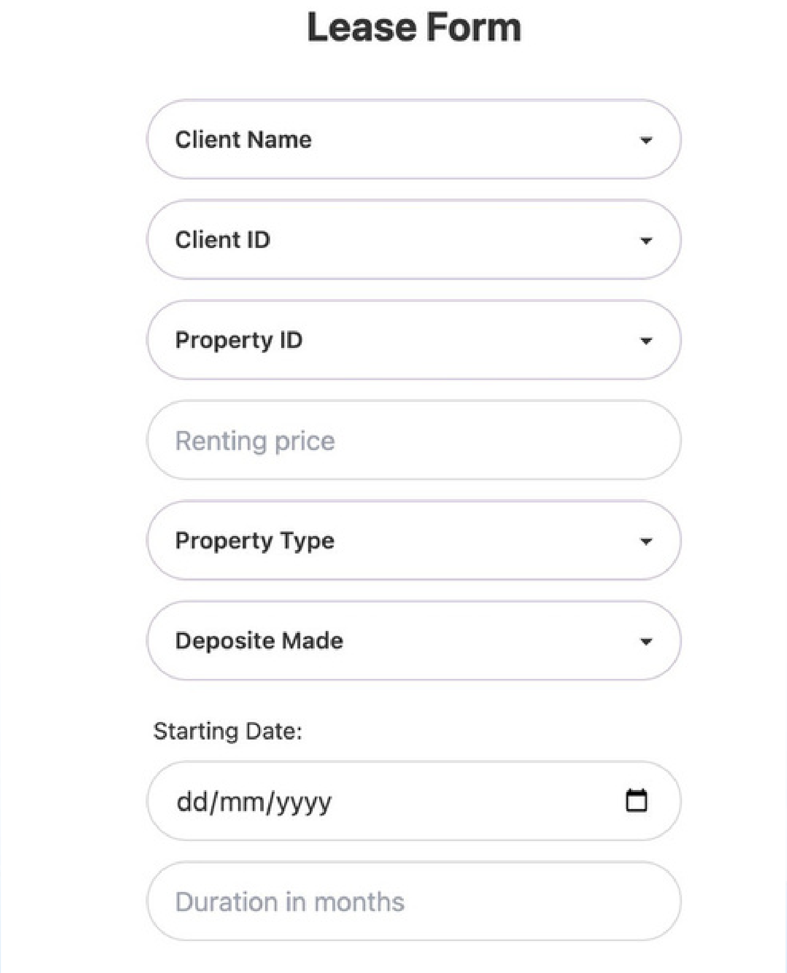
Client ID

Comment

Date of review

dd/mm/yyyy

REGISTER



Lease Form

Client Name

Client ID

Property ID

Renting price

Property Type

Deposite Made

Starting Date:

dd/mm/yyyy

Duration in months

NORMALIZATION

Based on the table structure, the conclusion can be drawn that the database is in at least third normal form (3NF).

- First normal form (1NF): Each column in a table contains atomic values and there are no repeating groups or arrays. The given tables have no repeating groups or arrays and each column contains atomic values.
- Second normal form (2NF): The table is in 1NF and all non-key attributes are dependent on the entire primary key. The given tables seem to be in 2NF since there are no partial dependencies in the primary keys.
- Third normal form (3NF): The table is in 2NF and all non-key attributes are not transitively dependent on the primary key. The given tables also seem to be in 3NF since there are no transitive dependencies in the attributes.

Therefore, the database appears to be in at least 3NF.

CONTRIBUTIONS

Parikshith Palegar (21bcs078) - Interface

Shashank G (21bcs107) - Interface

Patankar Chinmayee Nilesh (21bcs079) - Database Design, Data entry, Report Making

Sai Jyothirmai Survarapu (21bcs123) - Database Design, Data entry, Report Making

Tejas S (21bcs125) - Database Design, Queries