Dream Home Rentals

A Project Report

Submitted for the partial fulfillment for the award of degree of

BACHELOR OF COMPUTER SCIENCE

By

PRASANTH P (BU200501)

Under the Guidance of

Prof P. KARTHIK M.Sc., BEd.,MPhil.,

DEPARTMENT OF COMPUTER SCIENCE

SACRED HEART COLLEGE (AUTONOMOUS)

Tirupattur - 635 601.

APRIL 2023.

CERTIFICATE

This is to certify that the project entitled "Dream Home Rentals" is being submitted to Sacred Heart College (Autonomous), Tirupattur, by PRASANTH P. for the partial fulfillment for the award of Degree of Bachelor of Science in Computer Science is a bonafide record of the work carried out by them, under my guidance and supervision.

Date : _____

Project Guide

Department Of Computer Science

Sacred Heart College (Autonomous)

Tirupattur – 635 061

Submitted for viva-voice examination on _____

Examiner:

1.

2.

Head of the Department

ACKNOWLEDGEMENT

At the very outset, I offer my sincere thanks to Almighty God for the grace and blessings that made me complete the project successfully.

I sincerely thank my parents, who have gifted me this life to attain many achievements.

As I submit the project report, it is a great pleasure to acknowledge my gratitude to various instrumental persons in completing this project.

I offer my humble gratitude to **Rev. Dr. John Alexandar, SDB**, the Rector and Secretary and **Rev. Dr. D. Maria Antony Raj, SDB**, Principal, Sacred Heart College Tirupattur, for permitting me to have my project work.

I wish to convey my deep sense of gratitude to **Prof**. **Mrs. A. Josephine Sahaya Mala, MCA., M.Phil.,** Head of the Department, belongs to Computer Science Shift-II, Sacred Heart College Tirupattur.

I express my deep sense of gratitude to my project guide **Prof. P. KARTHIK** who is the source of inspiration that makes me pursue this work successfully. I thank him for his valuable guidance, encouragement and support in doing this project.

I want to thank the entire teaching and non-teaching staff of the Department of Computer Science for helping me complete the project successfully.

Finally, I thank my friends, especially those who gave me motivation and ideas to complete my project work.

TABLE OF CONTENTS

CHAPTER-I

PROJECT PROPOSAL

Executive Summary (Abstract)

"**Dream Home Rentals**" is a website that allows users to look for available rental houses. The platform typically includes a user-friendly search module that allows users to enter different criteria such as location, state, district, city, house type and other specifics, to narrow down the search results. The results are presented in a clean and well-organized list format, complete with a detailed view button for each house, making it easy for users to navigate and filter through the results to find their ideal rental house. To ensure privacy and security for tenants and house owner, the website or application typically includes a registration module based on phone number and One-time password (OTP) authentication. To protect the platform from fraud and fake listings, the website also includes a validation module that can only be accessed by designated super users, such as site administrators. This module allows them to review, verify and approve or reject the listing of houses added by other users.

INTRODUCTION

Sacred Heart College (Autonomous) is affiliated first grade college of Thiruvalluvar university. It is a minority institution established and administrated by the Salesians of Don Bosco. The college is offering various programs at the UG, PG, M.Phil and Ph.D levels. This is Prasanth P working on this project, doing III year B.Sc (Computer Science) in Sacred Heart College(Autonomous), Tirupattur.

This document of "Dream Home Rentals" is designed to simplify the process of searching and renting a home. The platform streamlines the process, making it easier, faster, and more convenient for tenants to find their dream home, and for homeowners to rent out their properties securely. With a user-friendly search module, registration process based on phone number and OTP authentication, and a validation module accessible by designated super users, **"Dream Home Rentals"** is a comprehensive and secure solution for both tenants and homeowners.

PROBLEM STATEMENT

- Finding and renting a home can be a challenging and time-consuming process for tenants.
- Homeowners also struggle with finding suitable tenants and managing their properties.
- The current process of finding and renting a home involves multiple steps and parties, making it cumbersome and inefficient.
- There is a lack of transparency in the rental market, with hidden fees and unclear terms and conditions causing frustration for both tenants and homeowners.
- The lack of standardization in the rental market leads to confusion and inefficiency in the rental process.
- There is a need for a platform that brings all stakeholders in the rental process together, providing a simplified and transparent process for finding and renting a home.

PROPOSED SOLUTION(S)

The proposed solution is:

- The "Dream Home Rentals" website offers an efficient and secure solution for home rental needs.

- Tenants can easily search for available properties using the platform's user-friendly search module, which allows for searching based on location, house type, and other criteria.

- Homeowners can list their properties and manage the rental process. Once a tenant books the home, the homeowner will receive an SMS notification and can then manage the rental process accordingly. This allows for a seamless and convenient experience for both homeowners and tenants.

- The website includes a phone number and OTP-based registration process and a validation module accessible only by designated super users to ensure privacy and security for both tenants and homeowners.

- The aim of the website is to simplify the home rental process and make it secure for all parties involved.

TECHNICAL REQUIREMENTS

1. **Project Title** : Dream Home Rentals
2. **Modules** : User, Admin, House Listing, House Search, Booking, Payment
3. **Database** : MYSQL
4. **Packages/GUI Tools** : VS Code, Mysql Work bench, Postmon
5. **Server Side** : Python(Django Framework) and MySQL database
6. **Client Side** : A GUI based JavaScript supported web browser.

Hardware/Software Requirements

1. **Hardware** : AMD Ryzen 7 with 8 GB RAM.
2. **Operating System** : Windows 10
3. **Software** : Visual Studio code

A. Web Server Specification

Hardware

Intel Pentium/512MB RAM with an Internet connection.

Operating System

Windows or Linux

B. Client Specification

Hardware

Intel Pentium/512MB RAM with an Internet connection.

Operating System

Platform Independent – Run on any browser.

Software

A GUI based JavaScript supported web browser.

TERMS AND CONDITIONS

1. **Project Duration -** The duration of the academic project is five months.

2. **Project Initiation -** The student will begin the requisite study and analysis after getting approval from the Project Guide and complete it according to the Department's agreed-upon project schedule.

3. **Development -** The project development also can be done in the college.

Signature:

Guide Name: **P. KARTHICK**

Date:

Signature:

Proposer Name: **P. PRASANTH**

Date:

CHAPTER-II

SYSTEM REQUIREMENTS

1. Introduction

The purpose of this document is to outline the vision for the Dream Home Rentals platform, which aims to simplify the process of finding and renting a home for both tenants and homeowners. The platform will provide a user-friendly interface for tenants to browse and search for available homes, as well as for homeowners to list and manage their properties. Through the use of modern technology and streamlined processes, Dream Home Rentals seeks to revolutionize the home rental industry and become the go-to platform for tenants and homeowners alike. This document outlines the key objectives, requirements, and strategies for achieving this vision.

2. Problem Statement

The problem of	inefficient and time-consuming house searching process not only involves the need for a broker, but also requires manual effort of searching houses
Affects	both tenants and the house owners
The impact of which is	spending more time and brokerage to find houses based on their expectations and also the house owners may experience a potential loss of rental income due to the extended vacancy of the property
A successful solution would be	Dream Home Rentals website

3. Problem Position Statement

For	both tenants and the house owners
Who	are seeking an efficient and convenient way to find rental houses, as well as for house owners who are looking for a platform to list their rental properties
The Dream Home Rentals	is a website
That	Provides a platform for house owners to advertise their houses for renting and provide a convenient way for tenants to find houses for rent
Unlike	the traditional manual searching process which can be time consuming, unreliable and also needs brokers to find rental houses which may lead to spend more money on brokerage
Our product	Dream Home Rentals allows house owners to easily upload their rental properties, while tenants can search and book houses based on their requirements. With high-quality photos and detailed descriptions, tenants have a clear understanding of the rental properties before booking.

4. Stakeholder Summary

Name	Description	Responsibilities
Website Admin	The website admin is responsible for managing the Dream Home Rental website.	Verify the information provided by the users before it is posted on the website. Respond to user inquiries and resolve any issues or concerns. Monitor the website's performance and make any necessary improvements.
Home Owners	Home owners are individuals who have a property they would like to rent out.	Provide accurate information about their property when posting it on the website. Respond to inquiries from potential tenants. activating and deactivating the house's visibility to tenants.
Tenants	Tenants are individuals looking for a home to rent.	Browse the website to find properties that meet their needs. Contact Home Owners to inquire about properties they are interested in. Communicate with Home Owners to arrange a rental agreement.

5. User Summary

Name	Description	Responsibilities	Stakeholder
Tenants	People looking for a place to rent for their home.	Search for available homes on the platform. Contact the home owners to request information or schedule a visit. Submit rental applications and payment. Provide feedback on the properties they have rented.	Tenants
Home Owners	People who own properties that they want to rent out.	List their properties on the platform Respond to inquiries from tenants. Manage the rental process including accepting applications and rent payments. Maintain the property and resolve any issues that arise during the rental period.	Home Owners
Administrator	People who manage the platform and enforce the terms and conditions.	Verify the authenticity of the properties listed by the home owners. Approve or reject rental listings based on the information provided. Respond to user complaints and resolve disputes. Monitor the platform to ensure that users are following the terms and conditions.	Administrator

1. Introduction

The Dream Home Rentals project aims to simplify the process of finding and renting a home for both tenants and homeowners. The system will provide a platform for users to list and search for available houses, manage bookings, and make payments. The project will be developed to provide an efficient and convenient way for all stakeholders to interact and manage their activities. This system study will provide a detailed analysis of the requirements, objectives, and scope of the Dream Home Rentals project. It will outline the functionalities and features of the system, as well as the technologies and tools that will be used to implement it. The study will also identify the potential challenges and risks associated with the project and propose strategies to mitigate them.

1.1 Reference

This section contains a complete list of all documents mentioned in the Vision document. Each should be identified by its title, report number, date, and publishing organization. Indicate where the references can be obtained. This information could be provided by linking to an appendix or another document.

2. List of modules

1. User Management Module
2. Admin Module
3. House Listing Module
4. House Update Module
5. House Search Module
6. House Booking Module
7. Payment Management Module

2.1 Description of modules

Module 1:

User Management

Purpose:

The purpose of this module is to manage user accounts, user authentication, and authorization.

Description:

In this module, users can register and log in to their accounts. Once logged in, they can view their profile, update their personal information, and change their password.

Entry Criteria:

The user needs to access the application to register and login.

Input:

1. User's personal information including name, email address, and phone number.
2. User's desired username and password.

Output:

1. User accounts are created and stored in the database.
2. User authentication and authorization are managed.

Exit Criteria:

Users can log out of their accounts.

Module 2:

Admin

Purpose:

The purpose of this module is to manage the application's content and functionality.

Description:

In this module, administrators can view and manage all users' accounts, including creating, editing, and deleting user accounts. They can also view and manage house listings and bookings. Administrators can validate the post of house listings before they are made public to ensure they meet the platform's standards.

Entry Criteria:

The administrator needs to access the application and authenticate themselves.

Input:

1. Admin's username and password for authentication.
2. New house listing details for validation.

Output:

1. Admin can view and manage all user accounts.
2. Admin can view and manage house listings and bookings.
3. Admin can create, edit, and delete user accounts.
4. Admin can validate new house listings before they are made public.

Module 3:

House Listing

Purpose:

The purpose of this module is to allow homeowners to upload details of their houses for rent.

Description:

In this module, homeowners can upload their house details including house images, location, rent price, and additional information.

Entry Criteria:

1. The homeowner needs to access the application and authenticate themselves.
2. Home owner needs to be a Premium user to upload their house details for rent.

Input:

1. House details including images, location, rent price, and additional information.
2. Homeowner's username and password for authentication.

Output:

1. House details are stored in the database.
2. House listing is visible in the house search module.

Exit Criteria:

Homeowner logs out of their account.

Module 4:

House Update Module:

Purpose:

The purpose of this module is to allow homeowners to update their house details.

Description:

In this module, homeowners can edit their house details including house images, location, rent price, and additional information.

Entry Criteria:

The homeowner needs to access the application and authenticate themselves.

Input:

1. House details including images, location, rent price, and additional information.
2. Homeowner's username and password for authentication.

Output:

1. House details are updated in the database.
2. House details are updated in the house search module.

Exit Criteria:

Homeowner logs out of their account.

Module 5:

House Search

Purpose:

The purpose of this module is to allow users to search for houses based on their preferences.

Description:

In this module, users can search for houses based on location, rent price, number of rooms, and other filters.

Entry Criteria:

The user needs to access the application and log in to their account.

Input:

User's search criteria including location, rent price, number of rooms, and other filters.

Output:

List of houses based on the user's search criteria.

Exit Criteria:

The user logs out of their account.

Module 6:

House Booking

Purpose:

The purpose of this module is to allow users to book a house for rent.

Description:

In this module, users can select a house from the house search module and book it for rent.

Entry Criteria:

The user needs to access the application and log in to their account.

Input:

1. User's selected house for rent.
2. User's payment information.

Output:

1. House booking details are stored in the database.
2. User's payment is processed.

Exit Criteria:

The user logs out of their account.

Module 7:

Payment Management

Purpose:

The purpose of this module is to handle all payment transactions between tenants and homeowners using the Cash free payment gateway.

Description:

In this module, users can make payments for their bookings and homeowners can receive payments for their listed properties. The module integrates with the Cashfree payment gateway to process payments using various payment methods, including credit/debit cards, UPI, and net banking.

Entry Criteria:

The user must have a valid booking ID, and the homeowner must have a valid property listing ID to make or receive payments.

Input:

1. Booking ID or Property listing ID
2. Payment amount
3. Payment method (credit/debit card, UPI, net banking, etc.)
4. Payment gateway credentials

Output:

1. Payment confirmation message
2. Transaction ID
3. Receipt generation for the payment made

Exit Criteria:

Payment made and received successfully.

Use Case No: DHR_1

Use Case Name: User Registration

Use Case Description:

This use case describes how a user can register for an account on Dream Home Rentals.

Actor:

User

Precondition:

User should be in the registration page.

Basic Flow:

- Enter personal information.
- Enter email address and password.
- Click the "Register" button.
- System will verify the user's information and creates a new account.

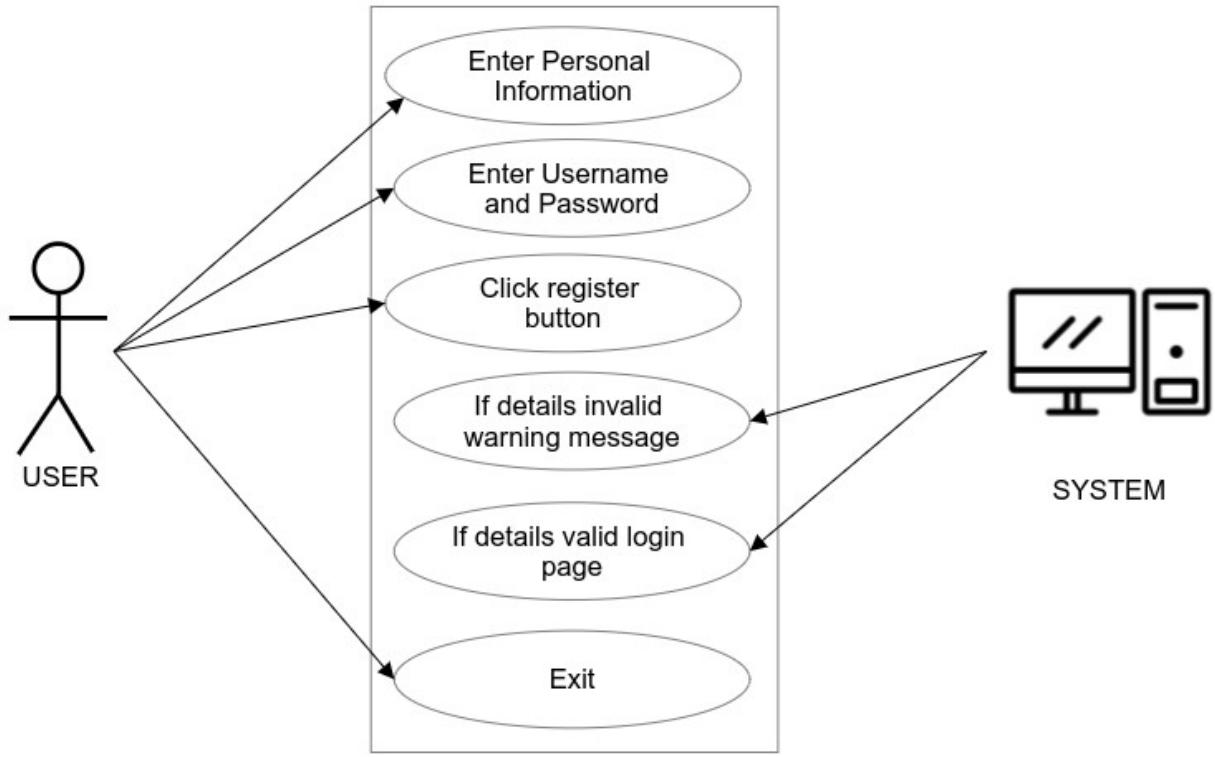
Alternative Flow:

The user can exit the registration page.

Post Condition:

The user can view the details of all available houses and also list their own house.

Use Case Diagram:



Use Case No: DHR_2

Use Case Name: User Login

Use Case Description:

This use case describes how a user can log in to their account on Dream Home Rentals.

Actor:

User

Precondition:

User should be in the login page.

Basic Flow:

- Enter phone number and password.
- Click the "Log In" button.
- System will verify the user's credentials and redirect to home page.

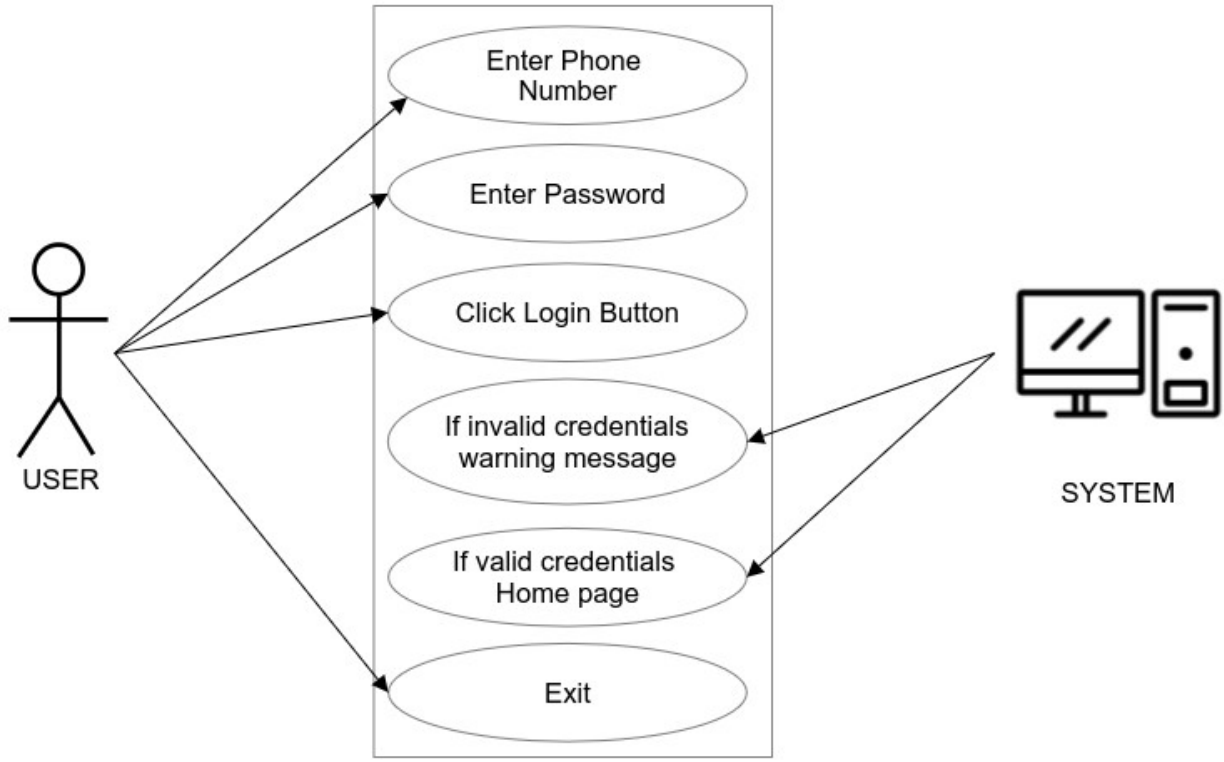
Alternative Flow:

The user can exit the login page.

Post Condition:

The user can view the details of all available houses and also list their own house.

Use Case Diagram:



Use Case No: DHR_3

Use Case Name: Search Houses

Use Case Description:

This use case describes how a user can search for houses on Dream Home Rentals.

Actor:

User

Precondition:

User should be logged in.

Basic Flow:

- Enter search criteria, such as location and price range.
- Click the "Search" button.
- System will display a list of houses that match the search criteria.

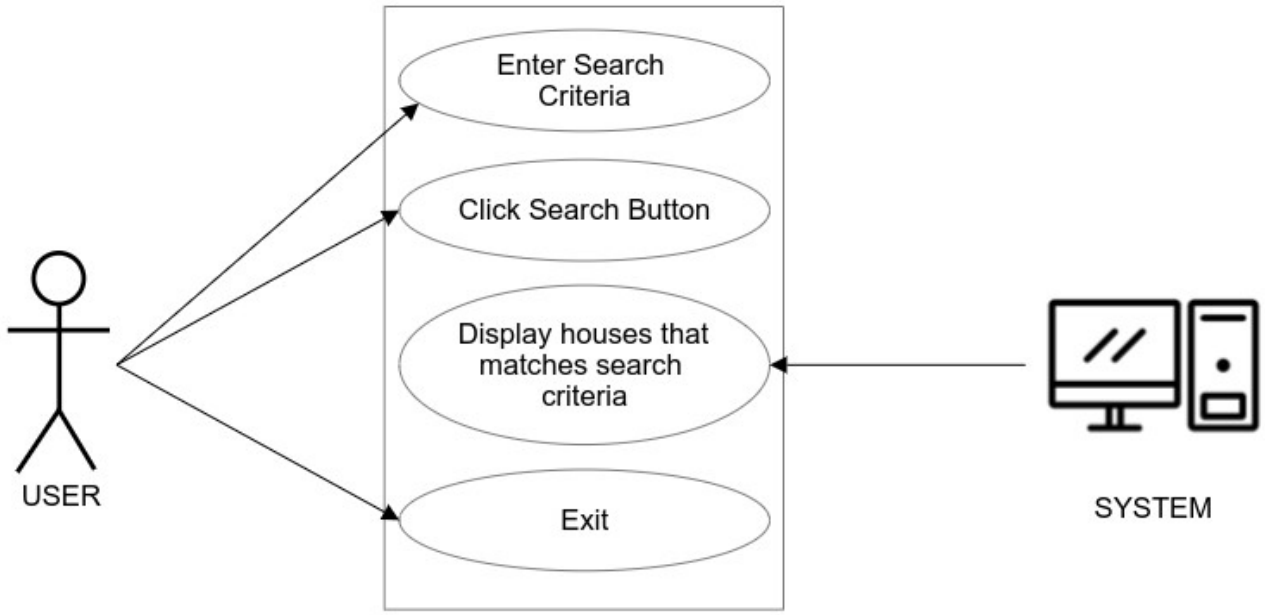
Alternative Flow:

The user can exit from the search page.

Post Condition:

The user can see the list of houses that match their search criteria.

Use Case Diagram:



Use Case No: DHR_4

Use Case Name: View House Details

Use Case Description:

This use case describes how a user can view the details of a house on Dream Home Rentals.

Actor:

User

Precondition:

The user should be logged in.

Basic Flow:

- Click on a house from the search results.
- System will display the details of the house, such as description, photos, and location.

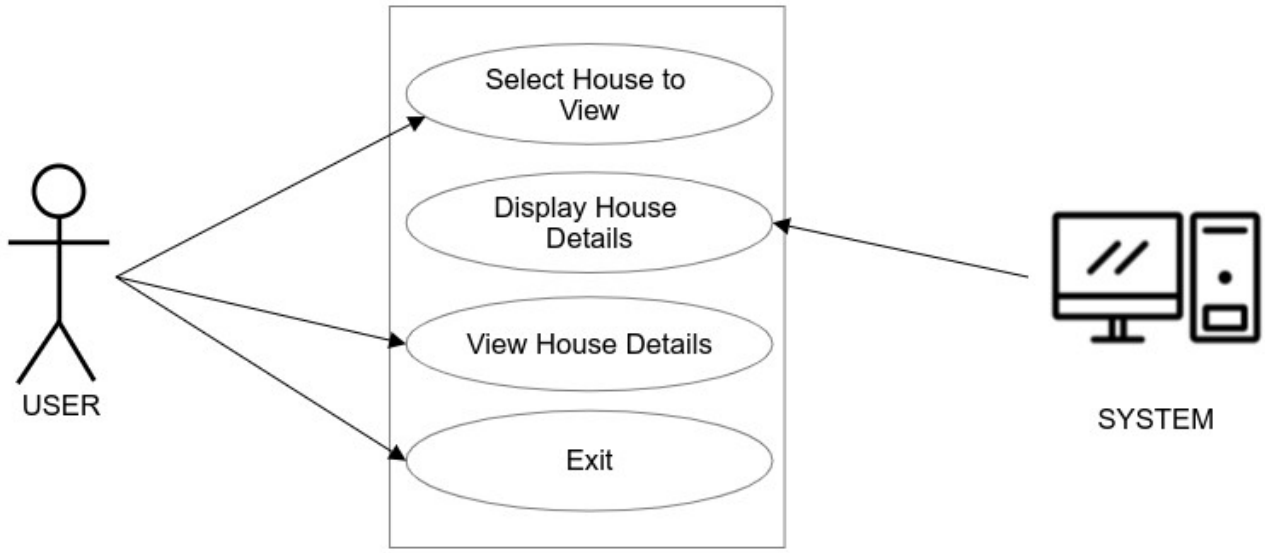
Alternative Flow:

The user can exit from the house details page.

Post Condition:

The can view the details of the house.

Use Case Diagram:



Use Case No : DHR_5

Use Case Name: Book a house

Use Case Description:

This use case describes how the user can book a house from the available listing.

Actor:

User

Precondition:

The user should be logged in.

Basic Flow:

- Select the desired house from the list of houses.
- Click Book Now button.
- Confirm the booking by making the payment.
- System will display the confirmation message and sends a confirmation SMS to the user.

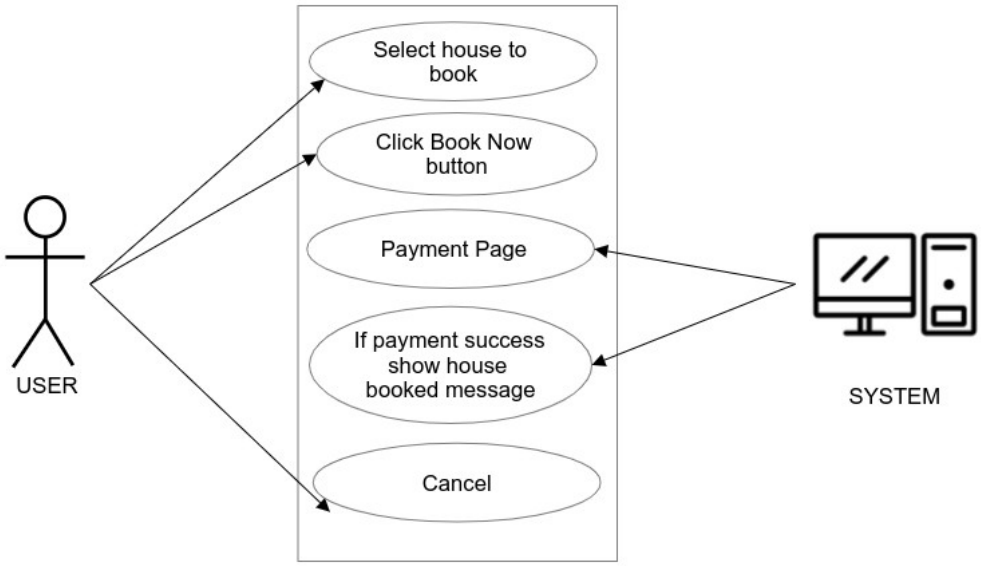
Alternative Flow:

The user can cancel the booking before making the payment.

Post Condition:

The house is booked and the user receives a confirmation message.

Use Case Diagram:



Use Case No: DHR_6

Use Case Name: Add House Listing

Use Case Description:

This use case describes how a house owner can add a new house listing to the website for rental.

Actor:

House Owner

Precondition:

House owner should be logged in

Basic Flow:

- Enter the required information about the house.
- The system will save the information to the database.

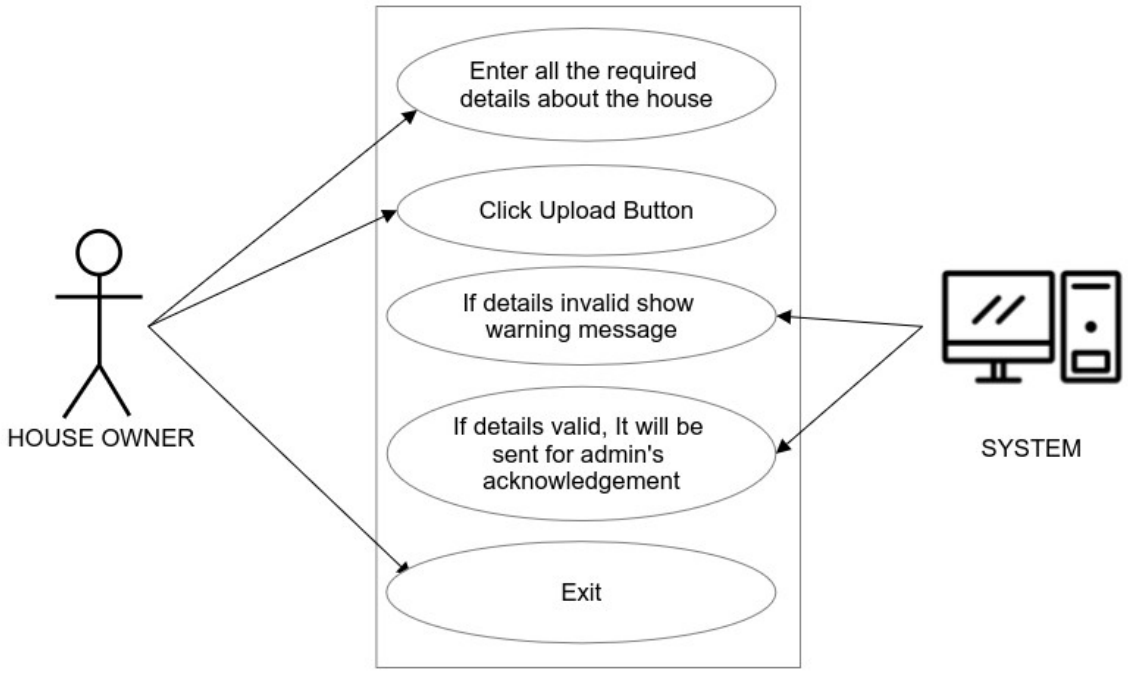
Alternative Flow:

House owner can exit from the Add House Listing page.

Post Condition:

The new house listing will be added to the website and can be searched and viewed by users.

Use Case Diagram:



Use Case No: DHR_7

Use Case Name: Edit House Listing

Use Case Description:

This use case describes how the user can edit an existing house listing.

Actor:

House owner

Precondition:

House owner must be logged in and have at least one house listing available.

Basic Flow:

- Select "Edit House Listing" option from the menu.
- System will display a list of the user's existing house listings.

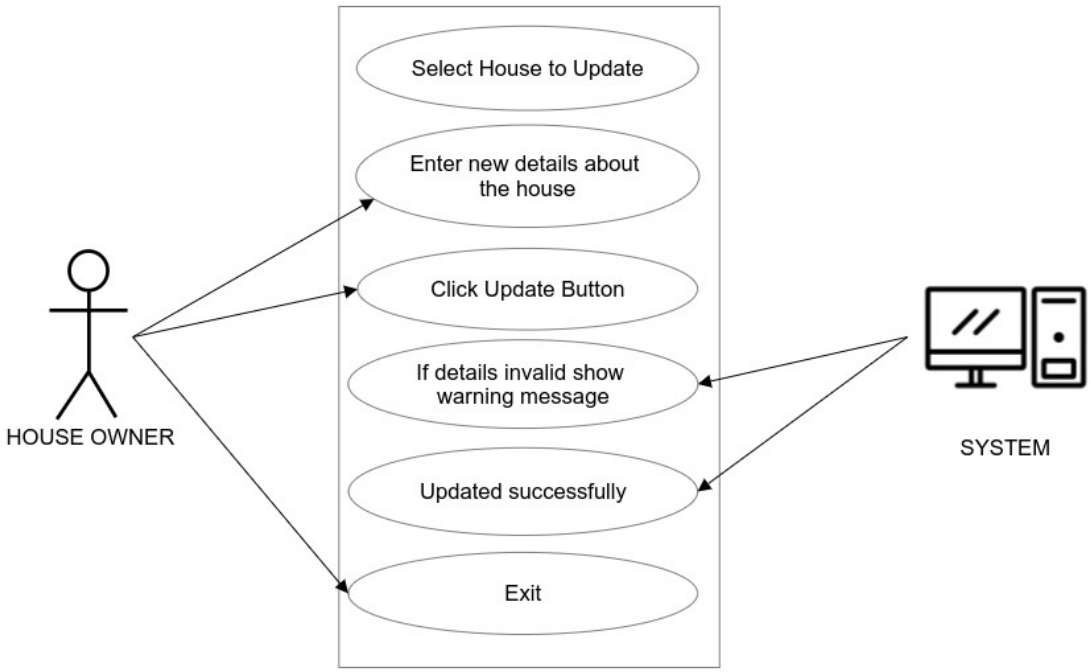
Alternative Flow:

The user can cancel the editing process.

Post Condition:

The user can makes changes to the existing house details and saves the changes.

Use Case Diagram:



Use Case No: DHR_8

Use Case Name: Delete House Listing

Use Case Description:

This use case describes how the owner can delete their listed house from the platform.

Actor:

House Owner

Precondition:

User must have logged in to their account.

Basic Flow:

- Select the desired house to be deleted.
- Click the delete button.

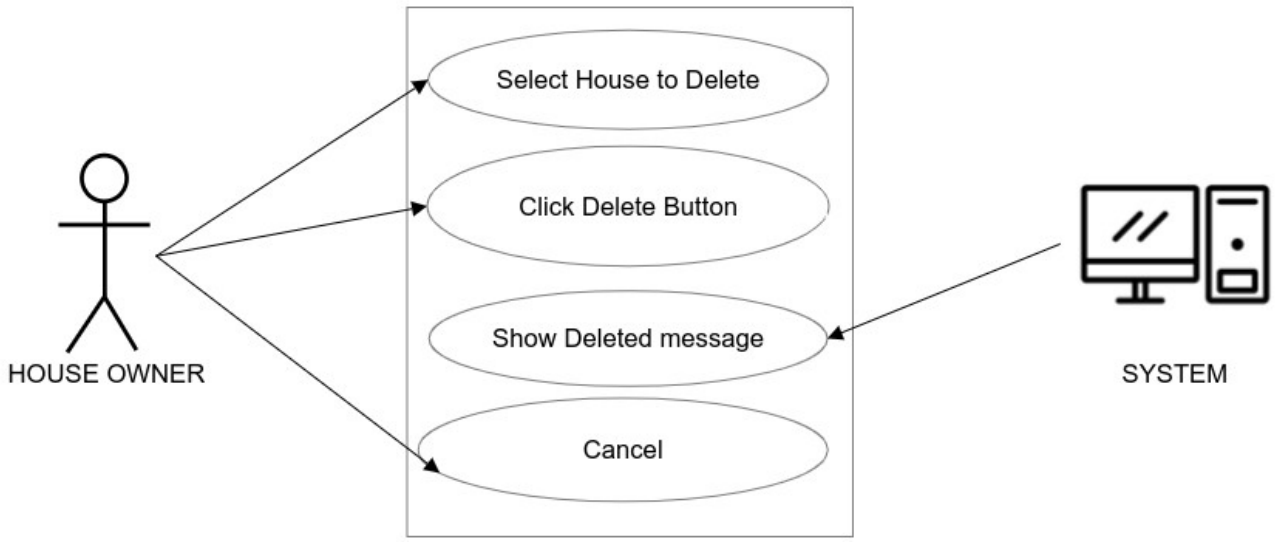
Alternative Flow:

The user can cancel the deletion process.

Post Condition:

User's house is deleted from the platform.

Use Case Diagram:



Use Case No: DHR_9

Use Case Name: Validate house

Use Case Description:

This use case describes how the admin validates the houses listed on the platform.

Actor:

Admin

Precondition:

Admin must be in the login page.

Basic Flow:

- Enter admin username and password.
- Click login button.

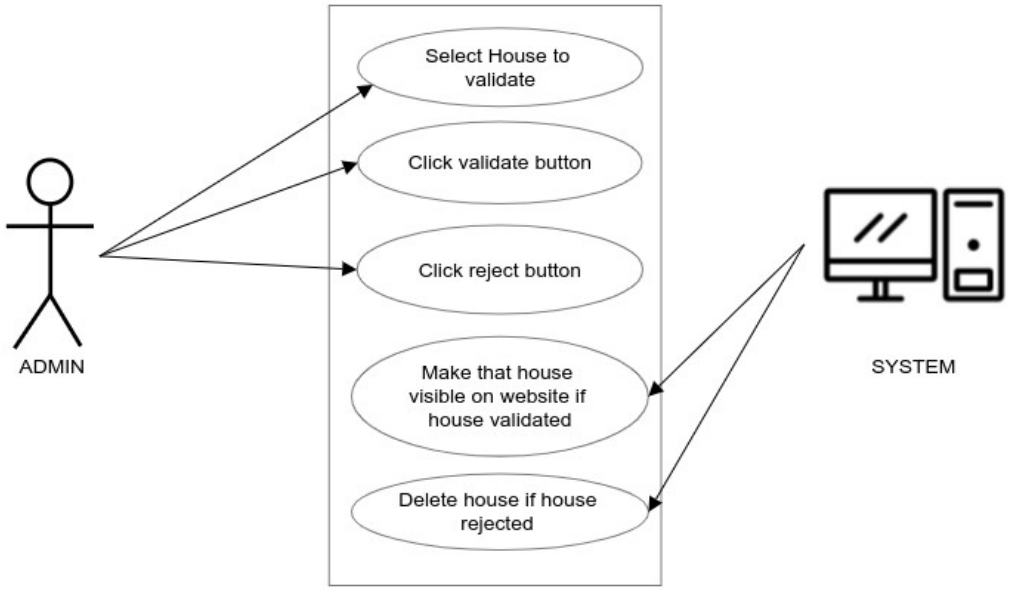
Alternative Flow:

The admin can exit from the login page.

Post Condition:

Admin can verify the house details and validate the house.

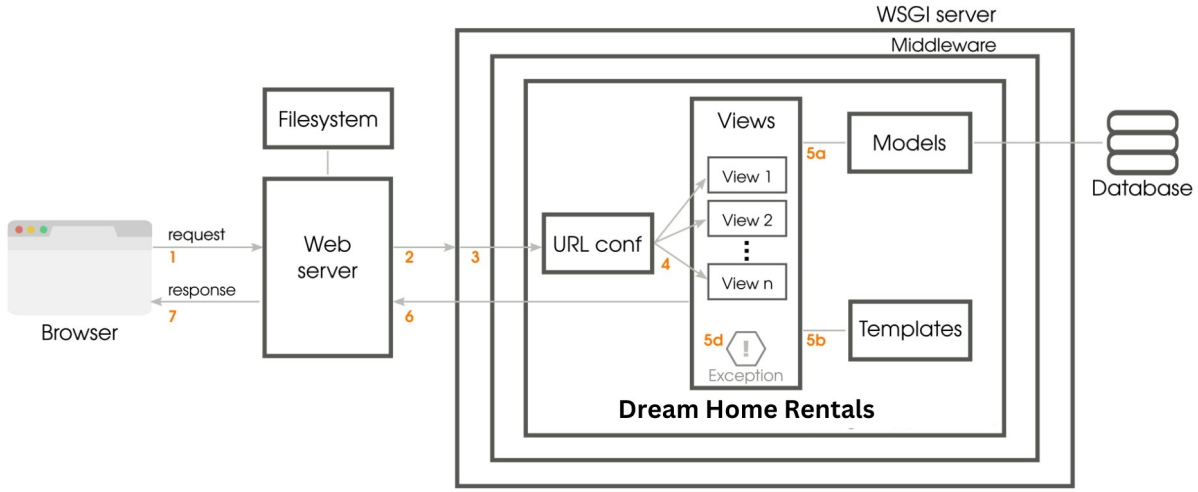
Use Case Diagram:

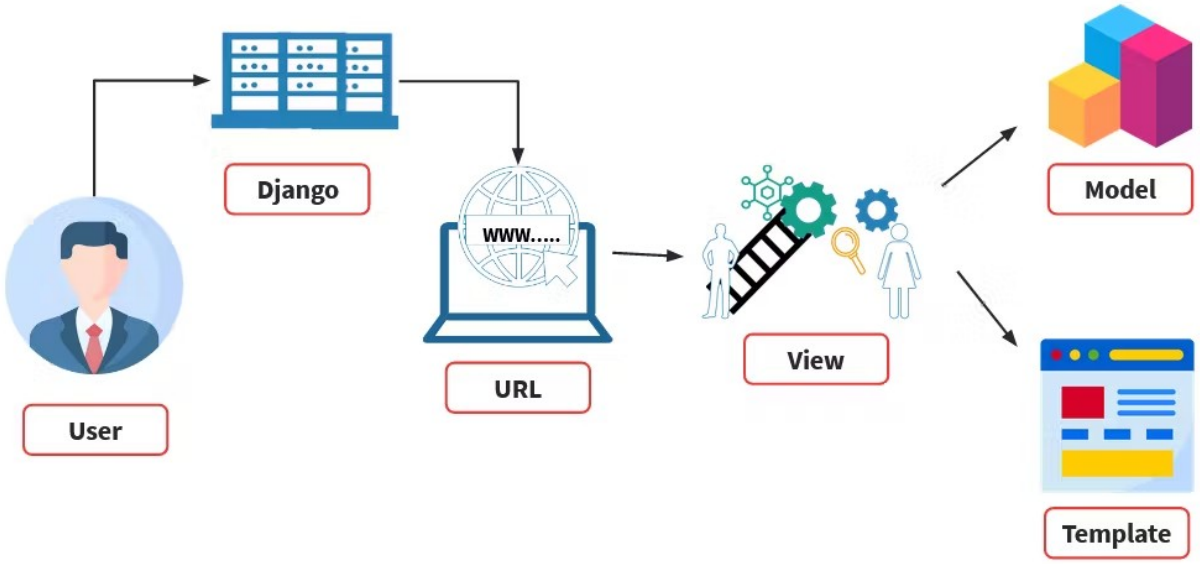


CHAPTER-III

ANALYSIS AND DESIGN

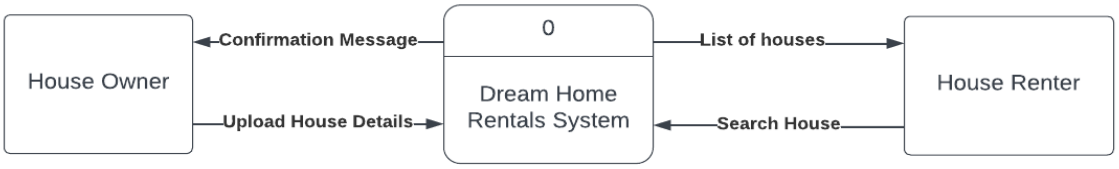
3.1 ARCHITECTURAL DIAGRAM



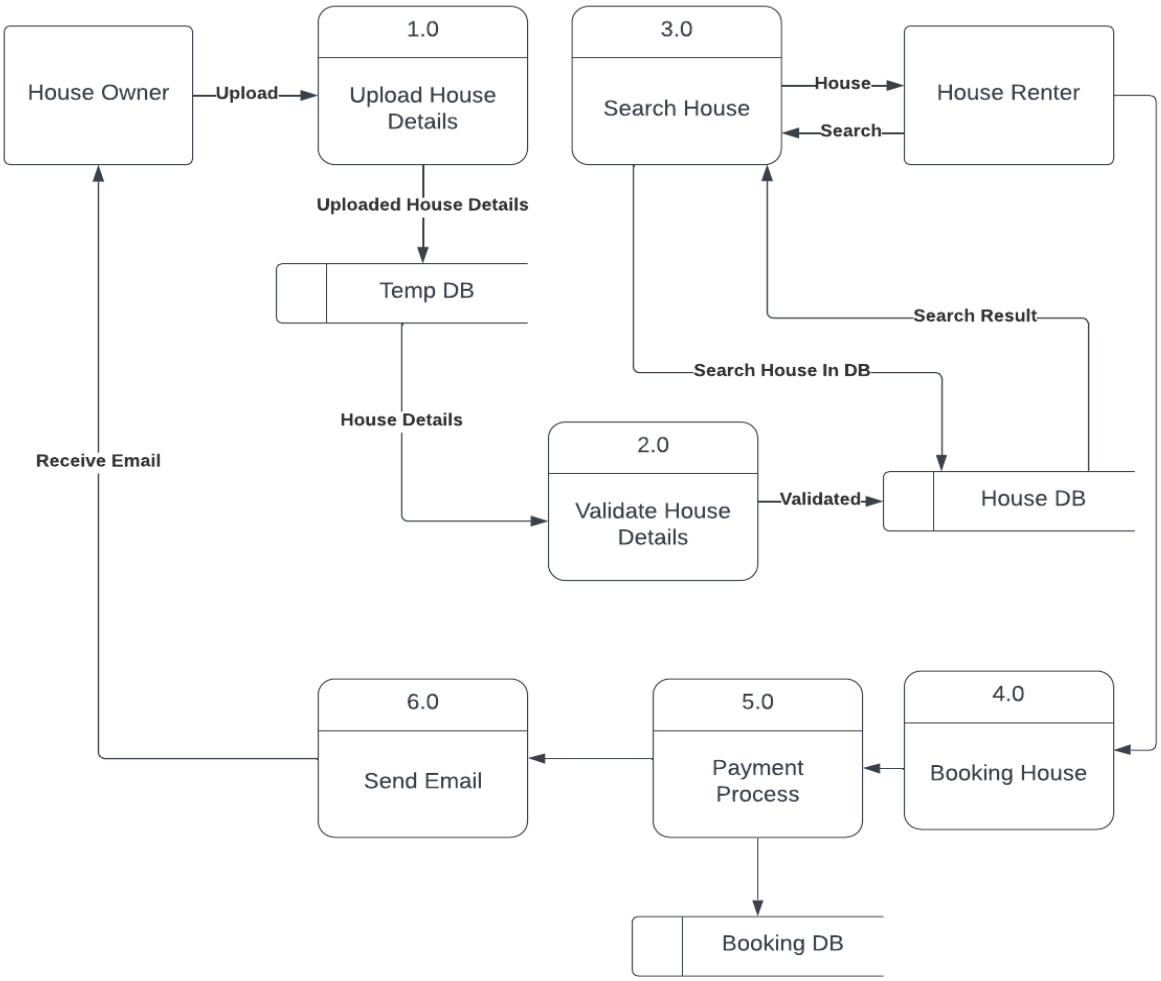


3.2 DATA FLOW DIAGRAM

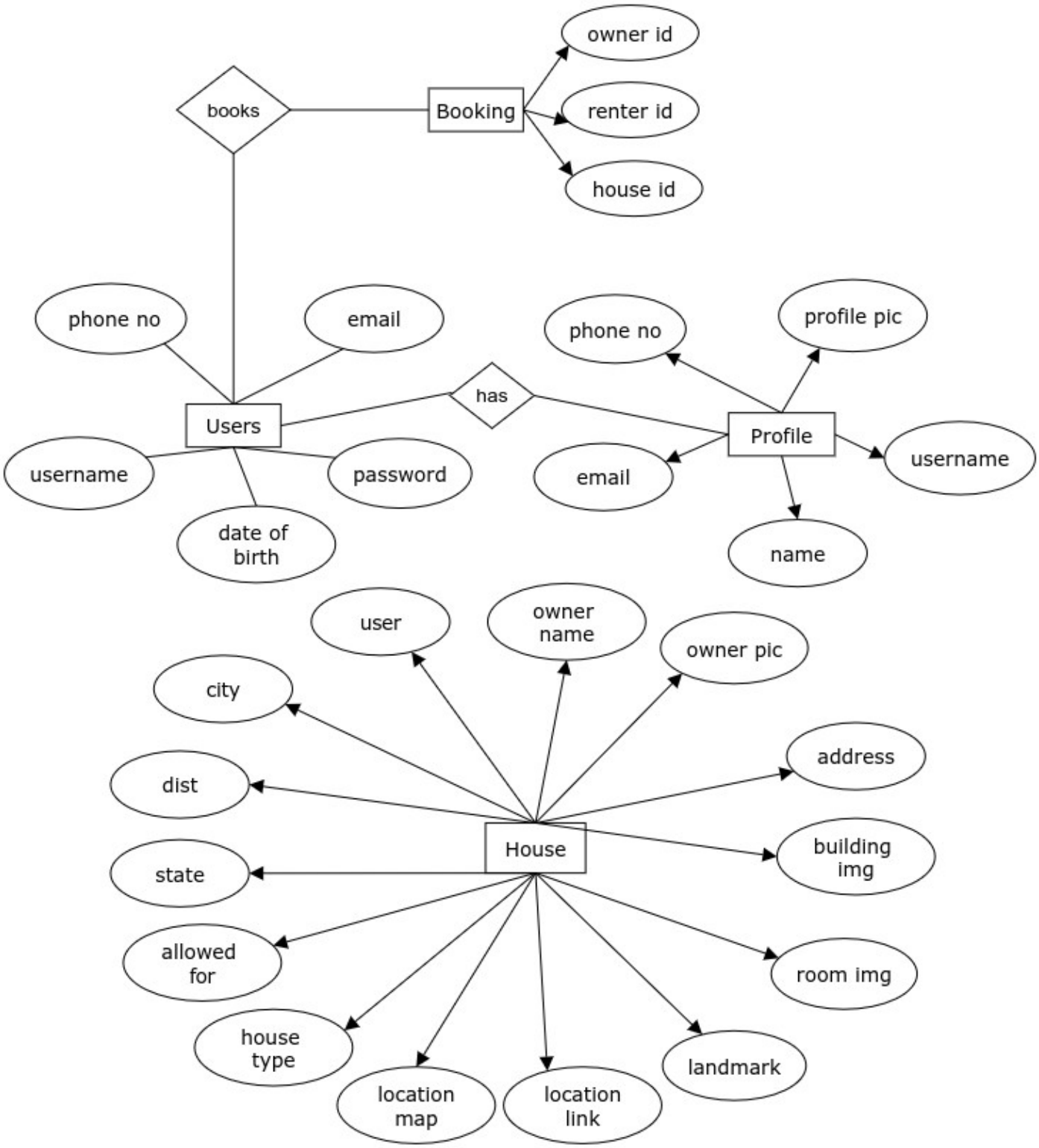
Level 0:



Level 1:



3.3 ER DIAGRAM



3.4 USE CASE DIAGRAM

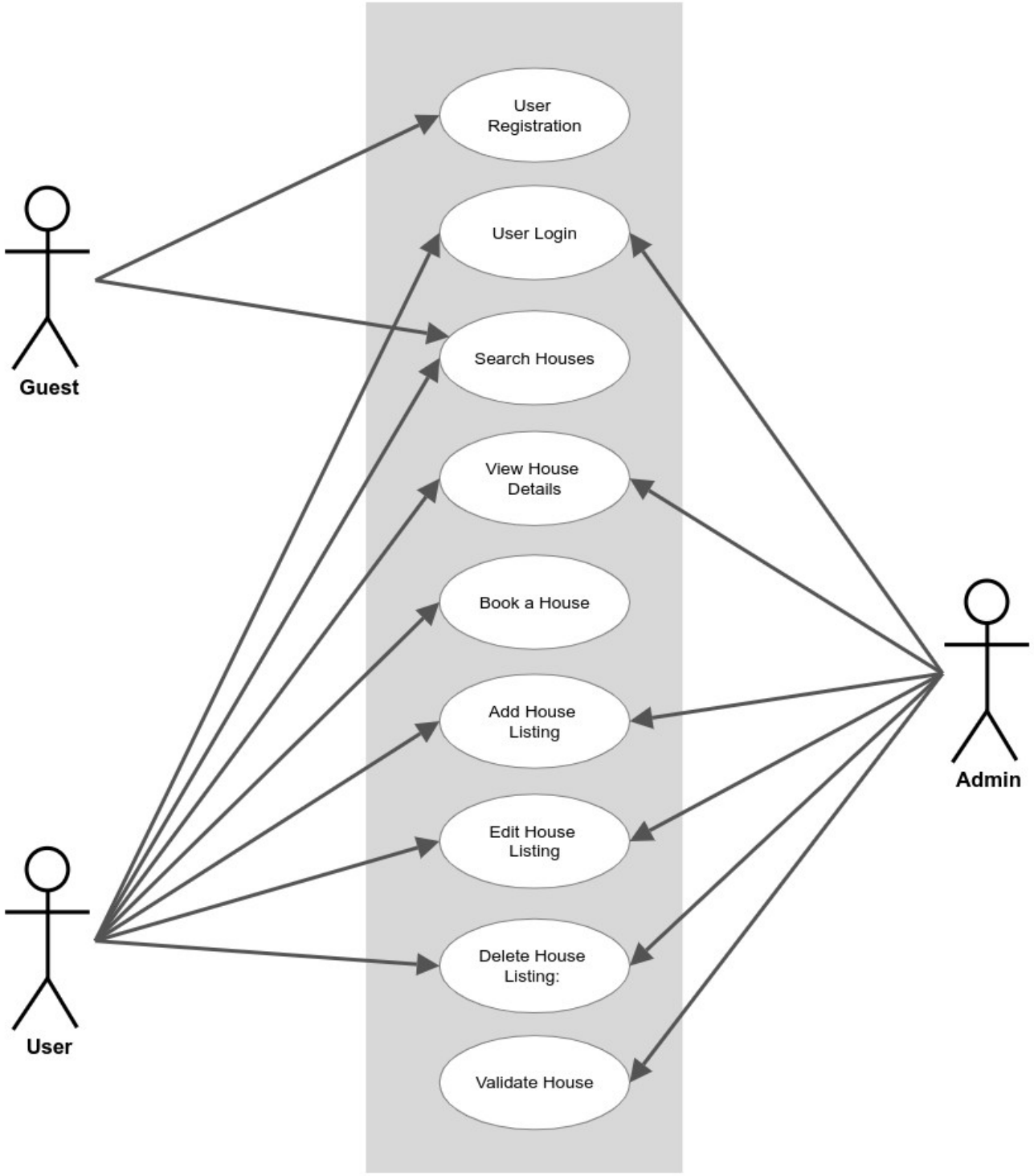


Table Name: Account_users

Description:

This table to store the user details.

S. No	Field Name	Field Type	Size	Constraints	Description
1	username	Varchar	100	Not Null	User Name
2	password	Varchar	128	Not Null	Password
3	email	Varchar	255	Not Null	Email
4	phone	Varchar	10	Not Null	Phone number
5	date_of_birth	Datetime	-	Not Null	Date of Birth

Table Name: Account_profile

Description:

This table to store the profile details.

S. No	Field Name	Field Type	Size	Constraints	Description
1	user_id	Int	11	Not Null	User Name
2	FullName	Varchar	100	Not Null	Name
3	Email	Varchar	100	Not Null	Email
4	Profile_pic	Varchar	255	Not Null	Profile picture
5	address	Longtext	-	Not Null	Address

Table Name: Rent_room

Description:

This table to store the house details.

S. No	Field Name	Field Type	Size	Constraints	Description
1	Id	Int	11	Not Null	Id
2	Owner_Name	Varchar	20	Not Null	Owner Name
3	Owner_pic	Varchar	50	Not Null	Owner Picture
4	House_address	Varchar	30	Not Null	Address
5	Building_img1	Varchar	50	Not Null	Building Image
6	Room_img1	Varchar	50	Not Null	Room Image
7	Landmark	Varchar	20	Not Null	Landmark
8	Location_link	Varchar	30	Not Null	Google Map link
9	location_map	Varchar	50	Not Null	Google Map Embed Code
10	House_type	Varchar	20	Not Null	House Type
11	AllowedFor	Varchar	20	Not Null	Allowed for
12	state	Varchar	15	Not Null	State
13	district	Varchar	15	Not Null	District
14	city	Varchar	15	Not Null	City

Table Name: Booking

Description:

This table to store the booking details.

S. No	Field Name	Field Type	Size	Constraints	Description
1	tenant_id	Int	4	Not Null	Tenant Id
2	owner_id	Int	4	Not Null	Owner Id
3	house_id	Int	4	Not Null	House Id

Introduction:

This purpose of these documents is providing the case specification for the user, cases captured in the case specification.

Scope:

This document applies to the “Dream Home Rentals” website.

Overview:

The following document contains the case’s specification for the various use case identified in software requirement specification.

Test Case 1: (User Management Module)

Test Case	Scenario	Selected Option	Condition	Expected Result
1	User attempts to log in to their account.	Login button	If the username and password entered are incorrect or not recognized by the system	Display an error message indicating that the username or password is incorrect and prompt the user to try again
			If the username and password entered are correct and recognized by the system	Display the home page of the website, allowing the user to access their account and begin using the site's features

Test Case 2: (Admin Module)

Test Case	Scenario	Selected Option	Condition	Expected Result
2	Admin attempts to access the admin panel	Admin Login button	If the admin credentials entered are incorrect or not recognized by the system	Display an error message indicating that the admin username or password is incorrect and prompt the admin to try again
			If the admin credentials entered are correct and recognized by the system	Grant access to the admin panel and display the validate navigation link to validate the post

Test Case 3: (House Listing Module**)**

Test Case	Scenario	Selected Option	Condition	Expected Result
3	House owner adds a new property listing	Add Property button	If the property details entered are incomplete or invalid	Display an error message indicating that the property details are incomplete or invalid and prompt the house owner to enter valid details
			If the property details entered are complete and valid	Add the new property listing to the system and display a success message to the house owner

Test Case 4: (House Update Module**)**

Test Case	Scenario	Selected Option	Condition	Expected Result
4	House owner updates an existing property listing	Edit Property button	If the property details entered for the update are incomplete or invalid	Display an error message indicating that the property details are incomplete or invalid and prompt the house owner to enter valid details
			If the property details entered for the update are complete and valid	Update the existing property listing in the system and display a success message to the house owner

Test Case 5: (House Search Module**)**

Test Case	Scenario	Selected Option	Condition	Expected Result
5	User searches for rental houses	Search button	If the search criteria entered by the user are invalid or incomplete Expected result	Display an error message indicating that the search criteria are invalid or incomplete and prompt the user to enter valid criteria
			If the search criteria entered by the user are valid and complete	Display a list of rental properties matching the search criteria and allow the user to view detailed information about each property

Test Case 6: (House Booking Module**)**

Test Case	Scenario	Selected Option	Condition	Expected Result
6	User books a rental property	Book Property button	If the booking information entered by the user is incomplete or invalid	Display an error message indicating that the booking information is incomplete or invalid and prompt the user to enter valid information
			If the booking information entered by the user is complete and valid	Book the rental property for the specified dates and display a confirmation message to the user

Test Case 7: (Payment Management Module)

Test Case	Scenario	Selected Option	Condition	Expected Result
7	User makes a payment for a rental property booking	Make Payment button	If the payment information entered by the user is incomplete or invalid	Display an error message indicating that the payment information is incomplete or invalid and prompt the user to enter valid information
			If the payment information entered by the user is complete and valid	Process the payment and confirm the successful completion of the transaction to the user

CHAPTER-IV

IMPLEMENTATION

User Management Module:

```
from django.shortcuts import render,redirect
from rest_framework.views import APIView,Response
from rest_framework import permissions,status,generics
from django.views.decorators.csrf import csrf_exempt
from django.utils.decorators import method_decorator
from django.contrib.auth import login,logout
from .serializers import ChangePasswordSerializer
from rest_framework.permissions import IsAuthenticated
from rest_framework.parsers import MultiPartParser,FormParser
from datetime import date
from django.core.mail import send_mail
from datetime import datetime,timedelta
import calendar
import re
# Create your views here.
from twilio.rest import Client
from decouple import config
from django.core.mail import send_mail
# for paymnet handling
import stripe
from decouple import config
from django.http import JsonResponse
```

```
from .serializers import
CreateUserSerializer ,LoginSerializer,EditProfileSerializer,UserAvatarSerializer,PremiumPlanSerialize
r,AddPremiumSerializer

import json

import requests

from . models import User,OTP,Profile,PremiumPlan

from rent.models import room

import random

from django.contrib.auth.decorators import login_required

from knox.views import LoginView,LogoutView

from knox.auth import TokenAuthentication

from knox.models import AuthToken


@method_decorator(csrf_exempt,name='dispatch')

class SendOTPphone(APIView):

    def send_otp(self, phone,otp):

        print("8"*70)

        print("Ye Send OTP TO PHONE NUMBER CALL HUA HAI ")

        Account_sid = config('Account_sid')

        print(Account_sid,"SID no")

        auth_token = config('auth_token')

        print(auth_token,"auth TOken ")

        client = Client(Account_sid, auth_token)

        print(Client)

        message = client.messages \

            .create(
```

```
        body="Your One Time Password For  DreamHomeRentals.com is {} Please do not share your
OTP with Any one ".format(otp),
        to='+91{}'.format(phone),
        from_=config('from'),
    )
    print(message)
    print(message.sid)
    print("8"*70)
    return
  def post(self,request,*args,**kwargs):
    data = request.body
    dict_data = json.loads(data)
    print(dict_data)
    phone_number = dict_data["phone"]
    print(phone_number)
    if phone_number:
        user = User.objects.filter(phone__iexact = phone_number)
        if user.exists():
            return Response({
                'status': False,
                'Detail':"Failed to enroll as phone number already taken "
            })
        else:
           key =  SendOTP(phone_number)
           if key:
               print(key)
```

```
old_otp = OTP.objects.filter(phone = phone_number)
if old_otp.exists():
    old = old_otp.first()
    count = old.count
    print(count)
    if count > 4:
        return Response({
            'status': False,
            'Detail': 'OTP sending limit is crossed contact to customer care on 8340312640 '
        })
    else:
        old.count = count+1
        old.otp = key
        old.save()
        self.send_otp(phone_number,key)
        return Response({
            "status": True,
            "OTP": key,
            "Detail": "OTP sent Successfully "


        })
OTP.objects.create(
    phone = phone_number,
    otp = key,
    count =1
)
```

```
            self.send_otp(phone_number, key)
            return Response({
                "status": True,
                "OTP" : key,
                "Detail": "OTP sent Successfully "
            })
        else:
            return Response({
                'status': False,
                'Detail': 'Something Went Wrong please contact customer support
            })
    else:
        return Response({
            'status': False,
            'Detail': 'Phone Number Not Given plz input valid phone number'


        })
def SendOTP(phone):
    if phone :
        key = random.randint(999,9999)
        return key
    else:
        return False
class validateOTP(APIView):
    def post(self,request,*args,**kwargs):
        data = request.body
```

```
        data_dict = json.loads(data)
        phone = data_dict["phone"]
        sent_otp = data_dict["otp"]
        if phone and sent_otp:
            old = OTP.objects.filter(phone__iexact = phone)
            old = old.first()
            if str(sent_otp) == old.otp :
                old.validated = True
                old.save()
                return Response({
                    "status" : True,
                    "Message " : "OTP Matched proceed for registration "
                })
            else:
                return Response({
                    "status": False,
                    "Message ": "OTP Not Matched Try Again with valid otp "
                })
        else:
            return Response({
                "status": False,
                "Message ": "Enter valid phone number in valid json format "
            })
class Register(APIView):
    def post(self,request,*args,**kwargs):
        print("Register call hua hai ")
```

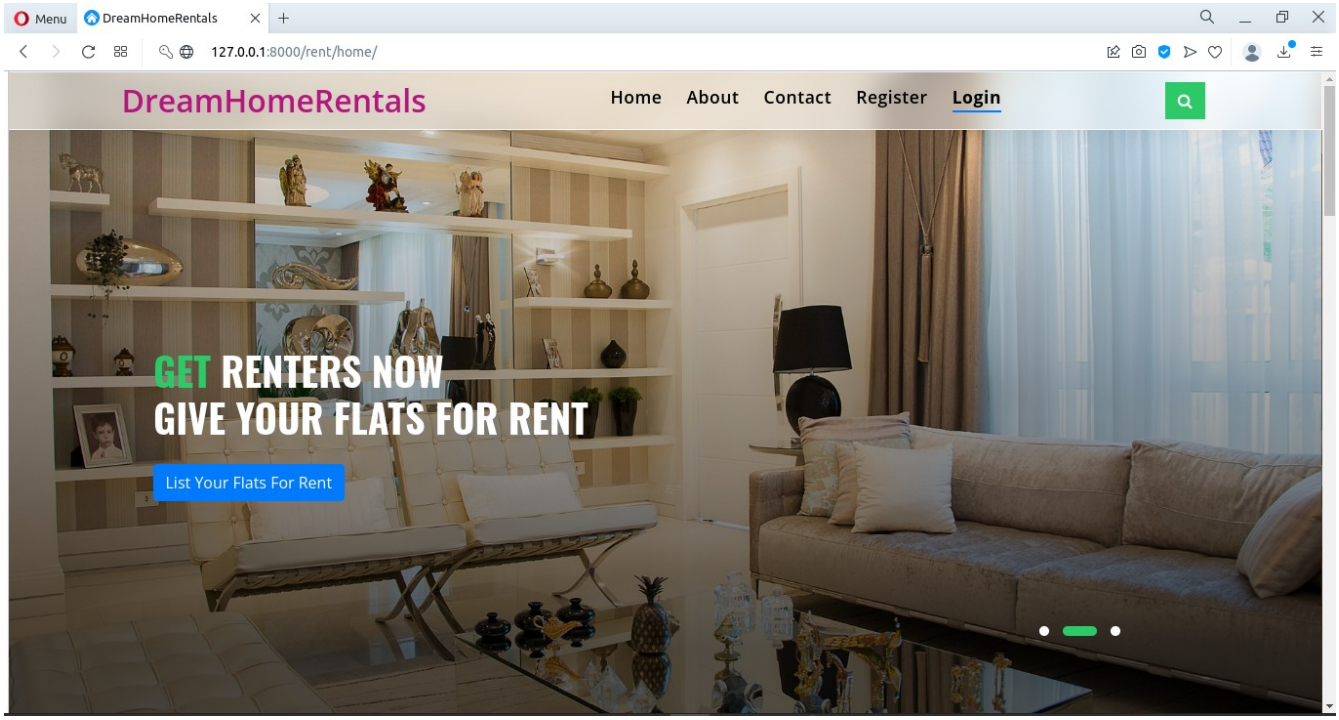
```
data = json.loads(request.body)
phone = data["phone"]
password = data["password"]
print(data ,"register form se aaya hai ")
if phone and password :
    old = OTP.objects.filter(phone__iexact = phone)
    if old.exists() :
        old = old.first()
        if old.validated:
            temp_data = {
                "phone":data['phone'],
                "password":data['password'],
                "email" :data['email'],
                "date_of_birth":data['DOB'],
                "username":data["username"]
            }
            serializer =  CreateUserSerializer(data=temp_data)
            serializer.is_valid(raise_exception=True)
            user = serializer.save()
            old.delete()
            return Response({
                "status" : True,
                "message" : "Account created",
                "token": AuthToken.objects.create(user)[1]
            })
        else:
```

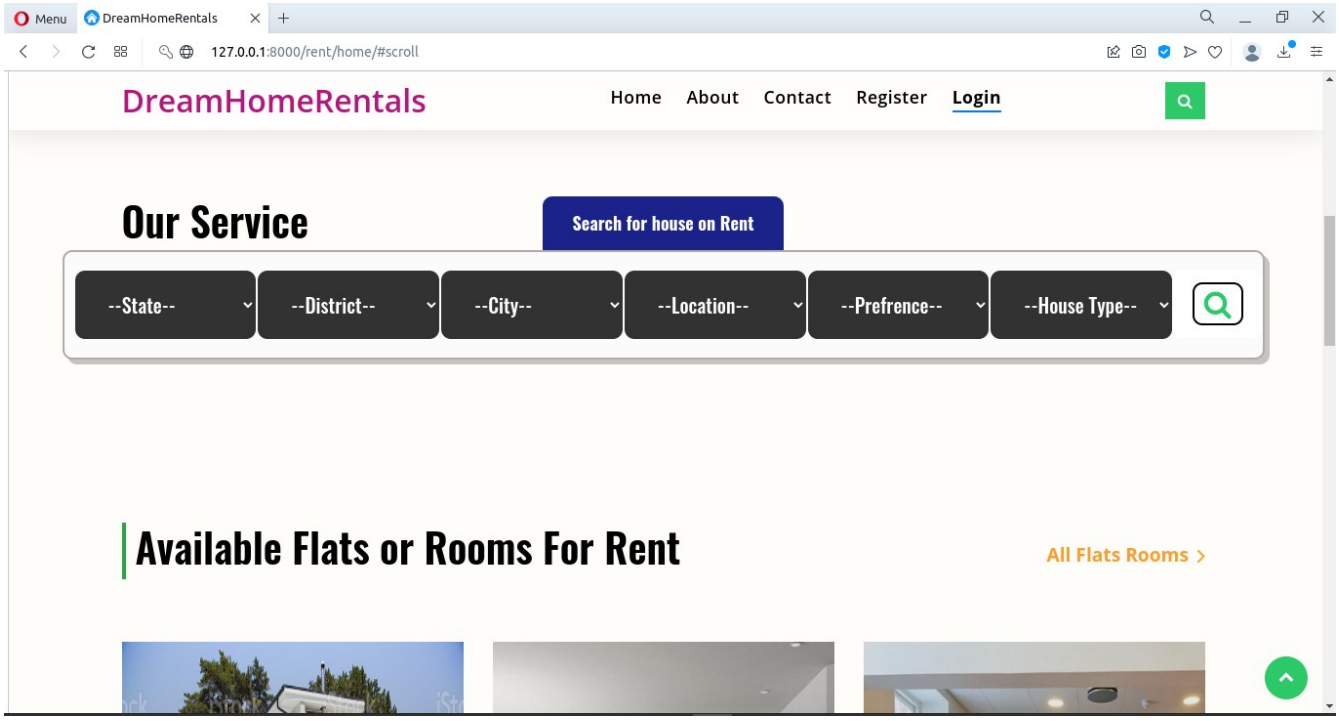
```
            return Response({
                "status": False,
                "message": "Account not created First verify ur phone"
            })
        else:
            return Response({
                "status": False,
                "message": "Account not created First verify ur phone"
            })
    else:
        return Response({
            "status": False,
            "message": "Enter valid phone or password in json format"
        })
class LoginAPI(LoginView):
    permission_classes = (permissions.AllowAny,)
    def post(self,request, format = None):
        data = request.body
        data = json.loads(data)
        serializer = LoginSerializer(data=data)
        serializer.is_valid(raise_exception=True)
        user = serializer.validated_data['user']
        login(request,user)
        res =super().post(request, format=None)
        print(res.data)
        print(super().post(request, format=None),"ye super method wala ha")
```

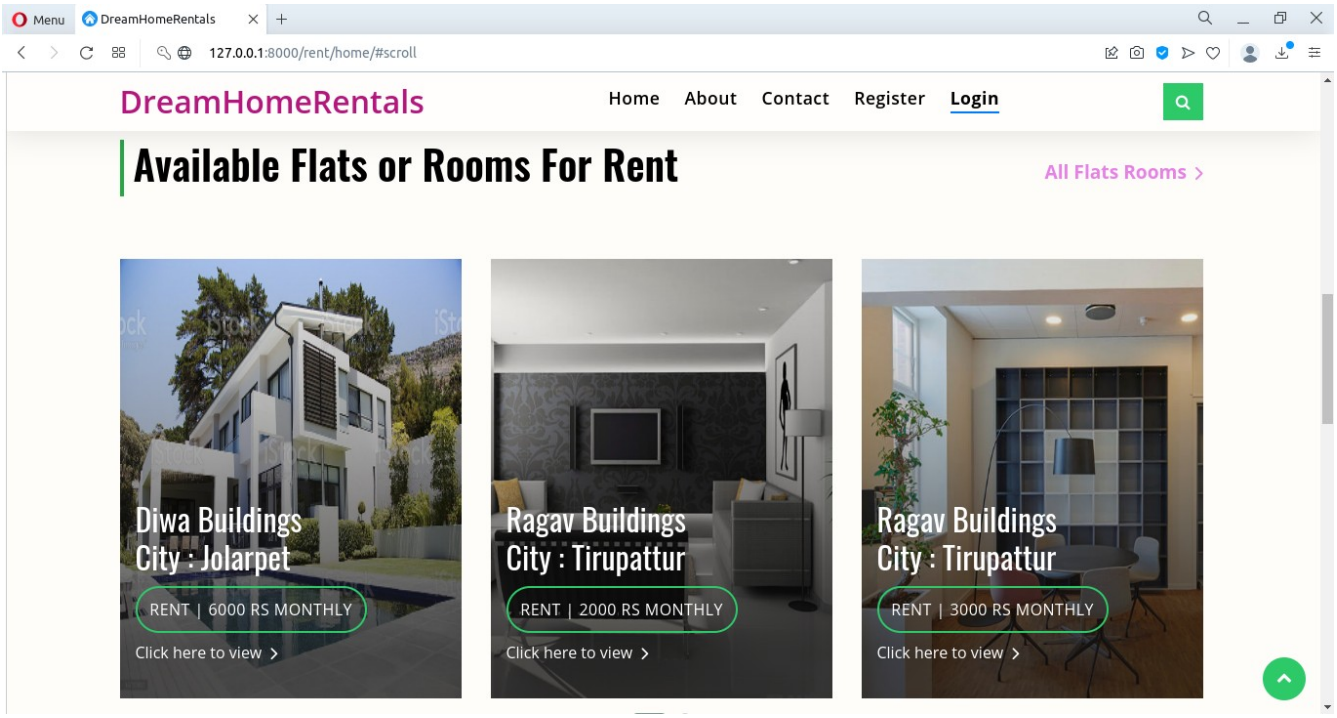
```
        if user.is_authenticated:
            print("user is logged in ")
            return Response({
                'status': True,
                'token': res.data['token']
            })
        else:
            return Response({'status' : False})
@method_decorator(login_required, name ="dispatch")
class ChangePasswordView(generics.UpdateAPIView):
    serializer_class = ChangePasswordSerializer
    model = User
    permission_classes = (IsAuthenticated,)
    def get_object(self, queryset=None):
        obj = self.request.user
        print(obj)
        return obj
    def put(self, request, *args, **kwargs):
        self.object = self.get_object()
        data = json.loads(request.body)
        print(data,"cOMING FROM CHANGE PASSW FORM")
        serializer = self.get_serializer(data=data)
        if serializer.is_valid():
            # Check old password
            if not self.object.check_password(serializer.data.get("old_password")):
```

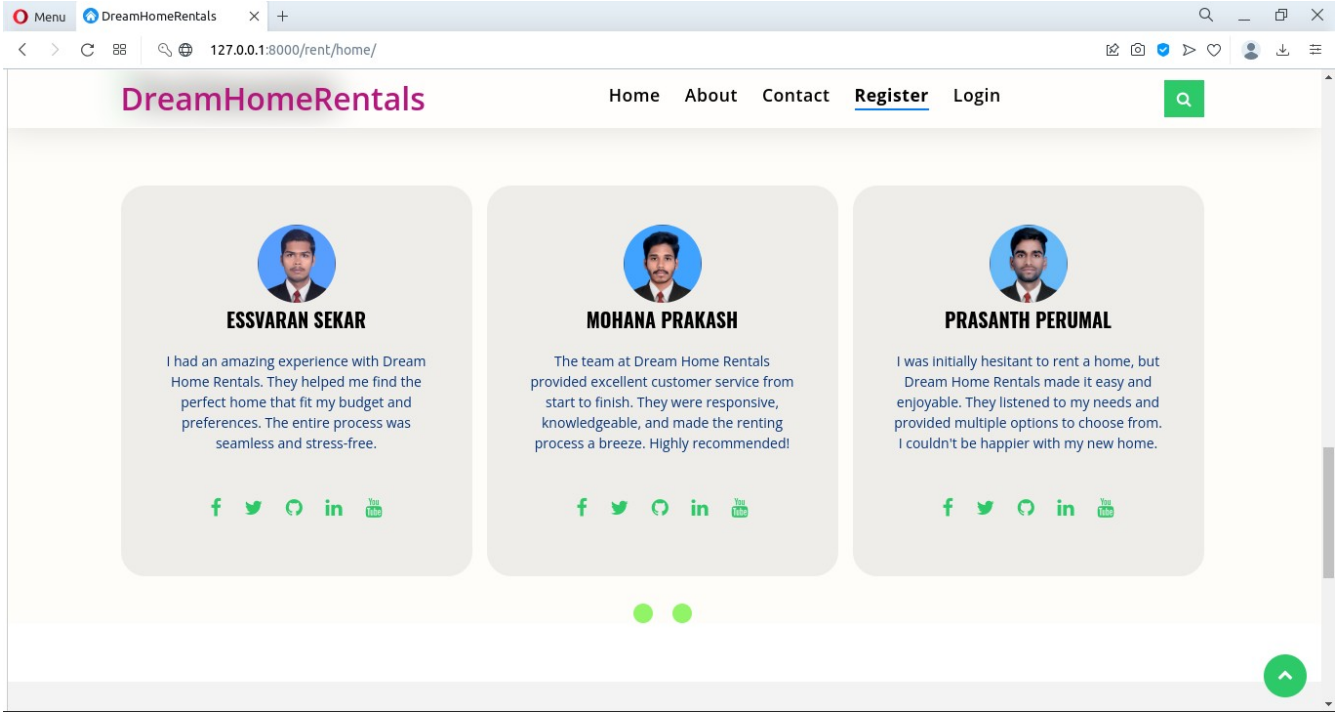
```
            return Response({"old_password": ["Wrong password."]},
status=status.HTTP_400_BAD_REQUEST)

        # set_password also hashes the password that the user will get

        self.object.set_password(serializer.data.get("new_password"))

        self.object.save()

        response = {

            'status': True,

            'code': status.HTTP_200_OK,

            'message': 'Password updated successfully',

            'data': []

        return Response(response)

    return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)

@method_decorator(login_required, name ="dispatch")

class EditProfile(APIView):

  permission_classes = (IsAuthenticated,)

  parser_classes = (MultiPartParser,FormParser)

  def put(self, request, format=None):

    data = request.body

    data = json.loads(data)

    # print(type(data['Profile_pic']))

    queryset = Profile.objects.get(id =data['id'])

    serializer = EditProfileSerializer(queryset,data=data)

    serializer.is_valid(raise_exception=True)

    if serializer.is_valid():

        serializer.save()

        return Response({
```

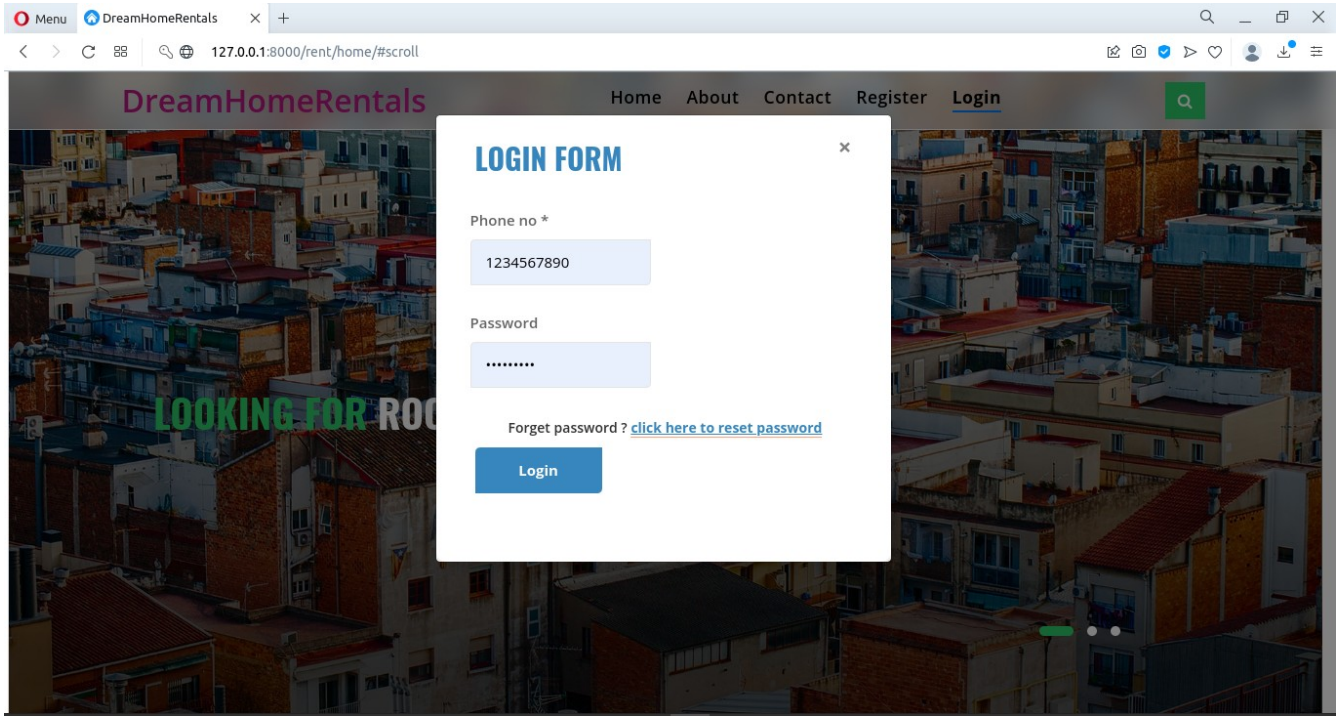
```
            "UpdatedData":serializer.validated_data,
            'status': True,
            "msg" : "Data successfully updated"
        else:
            return Response({
            "msg": "something went wrong",
            "status": False
            })
def logoutview(request):
    logout(request)
    return redirect('/')
@login_required()
def ProfileView(request):
    user = Profile.objects.filter(id = request.user.id)
    context = {'data': room.objects.all().filter(user=request.user.profile).order_by("-id"),
            'users': user
            }
    return render(request,'UserProfile.html',context)
```

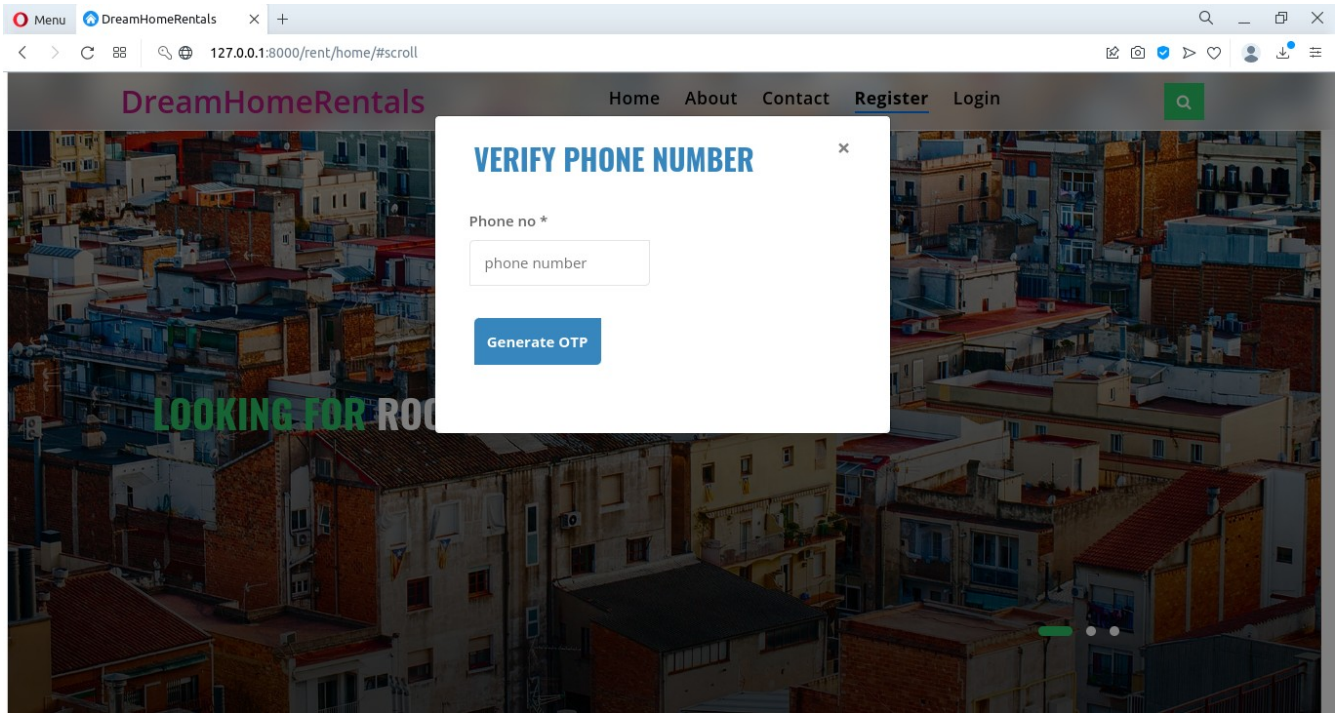

Menu
DreamHomeRentals
127.0.0.1:8000/rent/home/
DreamHomeRentals
Home
About
Contact
Register
Login
GET RENTERS NOW
GIVE YOUR FLATS FOR RENT
List Your Flats For Rent

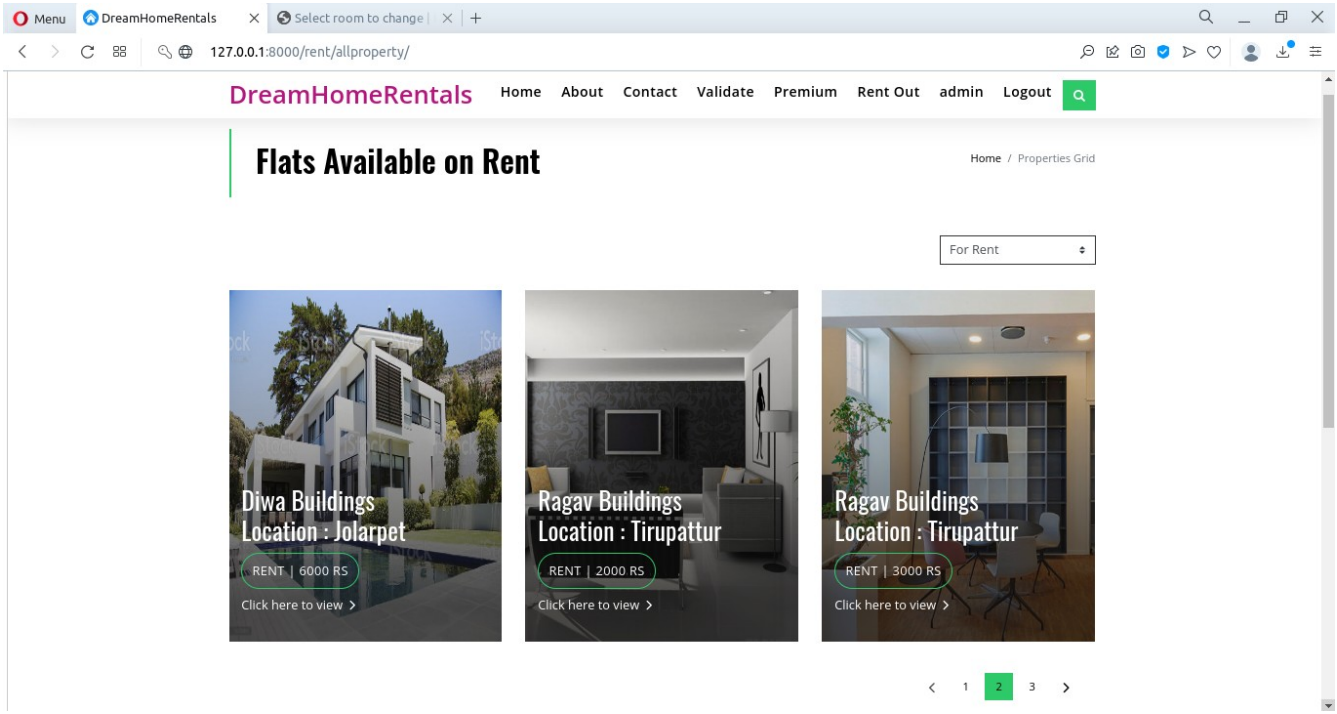

Menu
DreamHomeRentals
127.0.0.1:8000/rent/home/#scroll
DreamHomeRentals
Home
About
Contact
Register
Login
Our Service
Search for house on Rent
--State--
--District--
--City--
--Location--
--Prefrence--
--House Type--
Available Flats or Rooms For Rent
All Flats Rooms

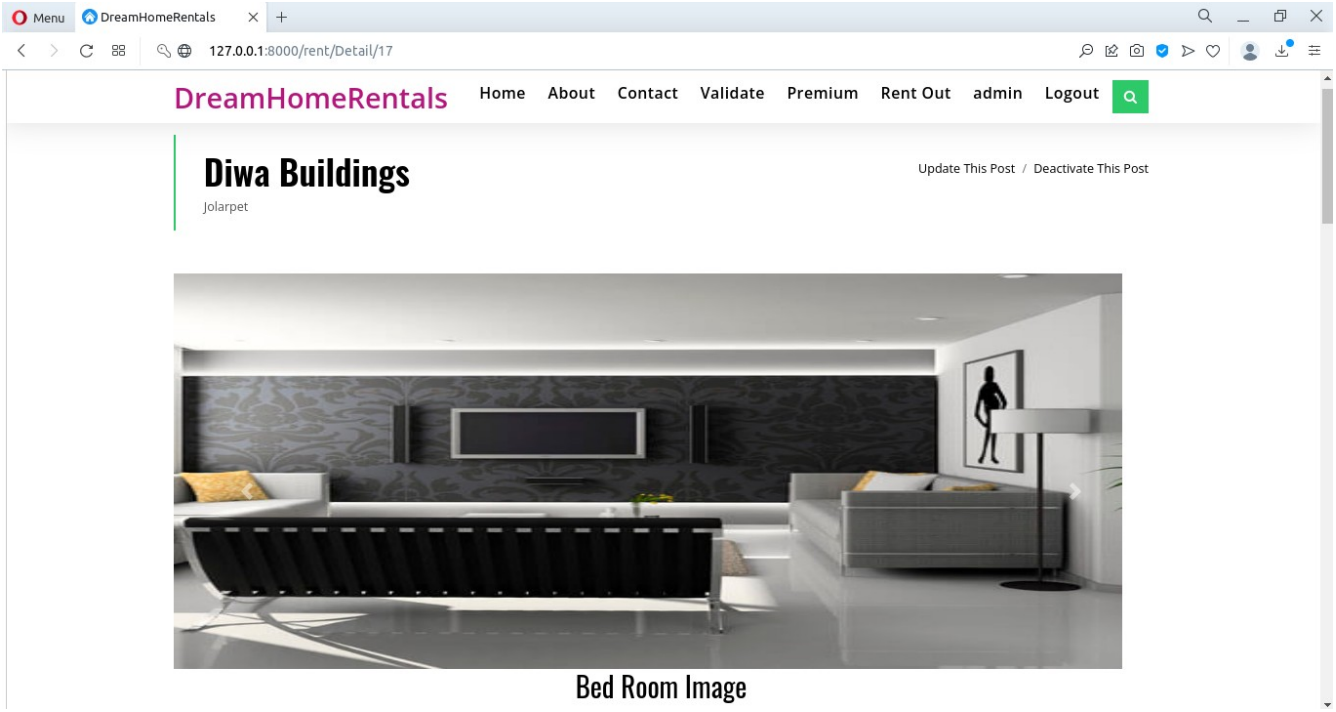

Menu
DreamHomeRentals
127.0.0.1:8000/rent/home/#scroll
DreamHomeRentals
Home
About
Contact
Register
Login
Available Flats or Rooms For Rent
All Flats Rooms
Diwa Buildings
City : Jolarpet
RENT | 6000 RS MONTHLY
Click here to view
Ragav Buildings
City : Tirupattur
RENT | 2000 RS MONTHLY
Click here to view
Ragav Buildings
City : Tirupattur
RENT | 3000 RS MONTHLY
Click here to view

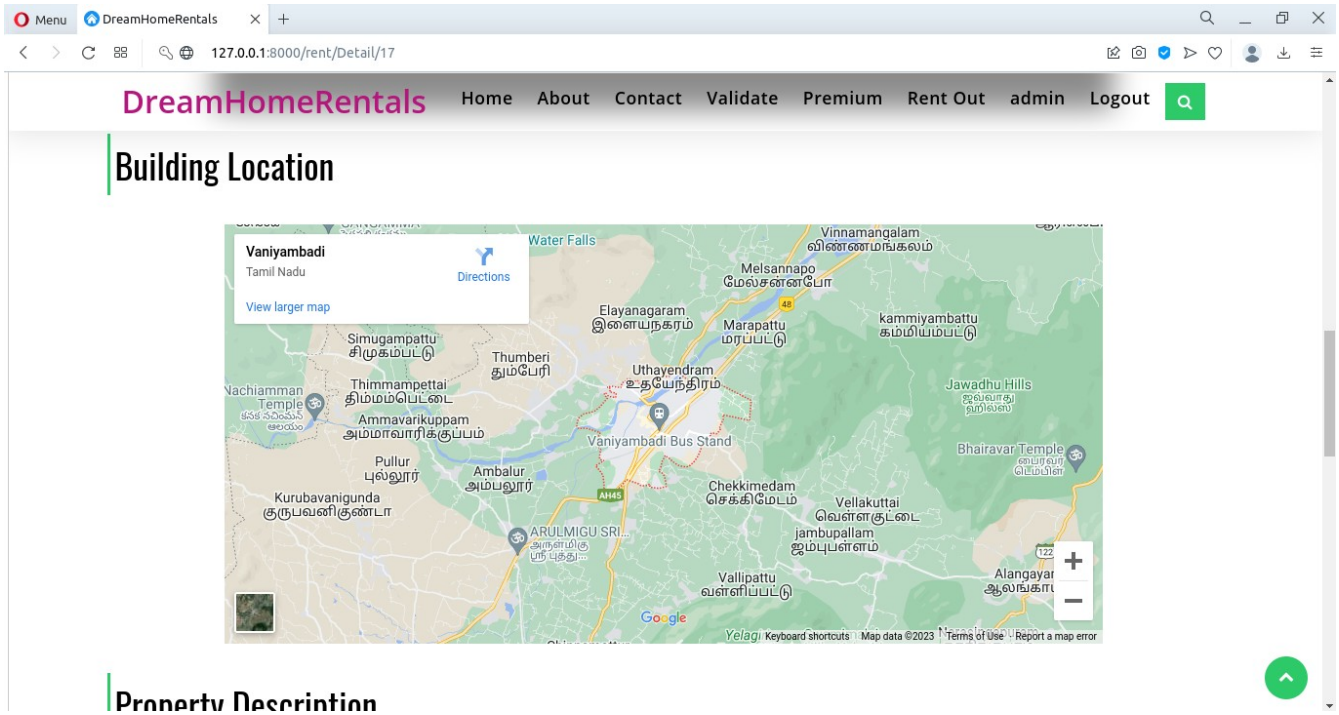

Menu
DreamHomeRentals
127.0.0.1:8000/rent/home/
DreamHomeRentals
Home
About
Contact
Register
Login
ESSVARAN SEKAR
I had an amazing experience with Dream Home Rentals. They helped me find the perfect home that fit my budget and preferences. The entire process was seamless and stress-free.
MOHANA PRAKASH
The team at Dream Home Rentals provided excellent customer service from start to finish. They were responsive, knowledgeable, and made the renting process a breeze. Highly recommended!
PRASANTH PERUMAL
I was initially hesitant to rent a home, but Dream Home Rentals made it easy and enjoyable. They listened to my needs and provided multiple options to choose from. I couldn't be happier with my new home.

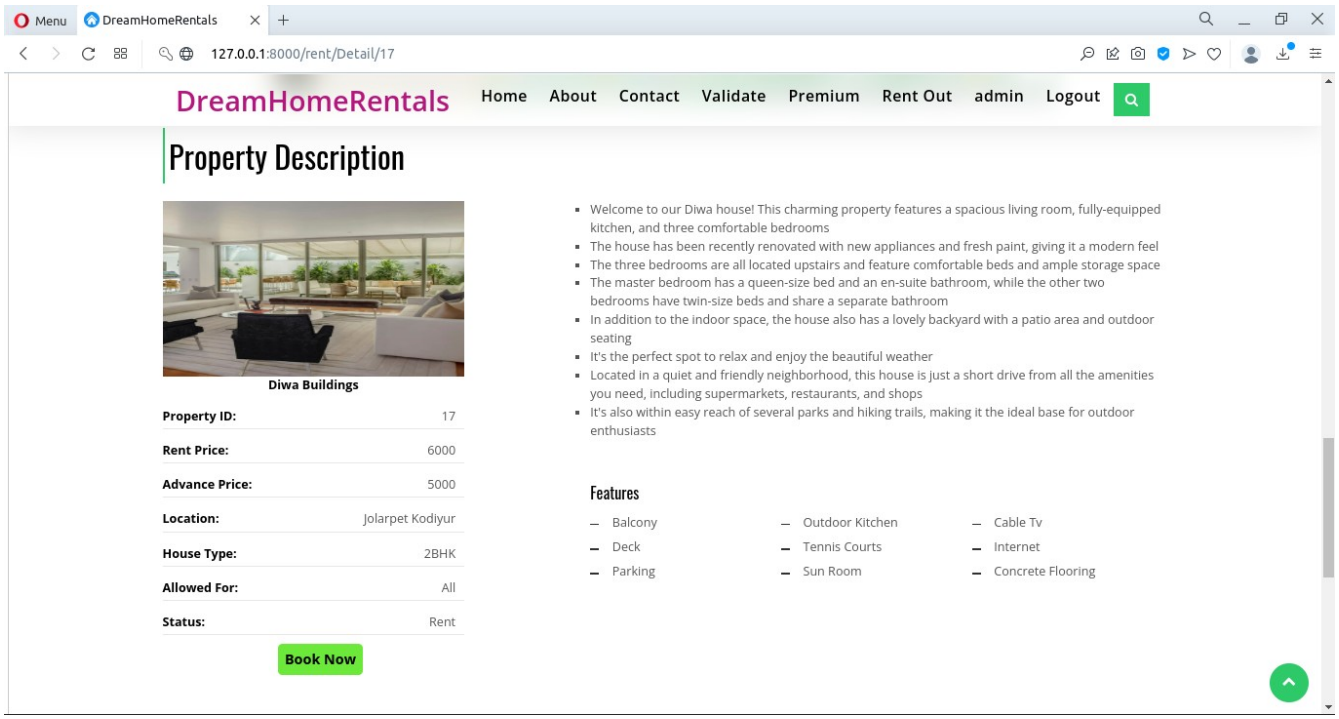

Menu
DreamHomeRentals
127.0.0.1:8000/rent/home/#scroll
DreamHomeRentals
Home
About
Contact
Register
Login
LOOKING FOR ROO
LOGIN FORM
Phone no *
1234567890
Password
Forget password ? click here to reset password
Login

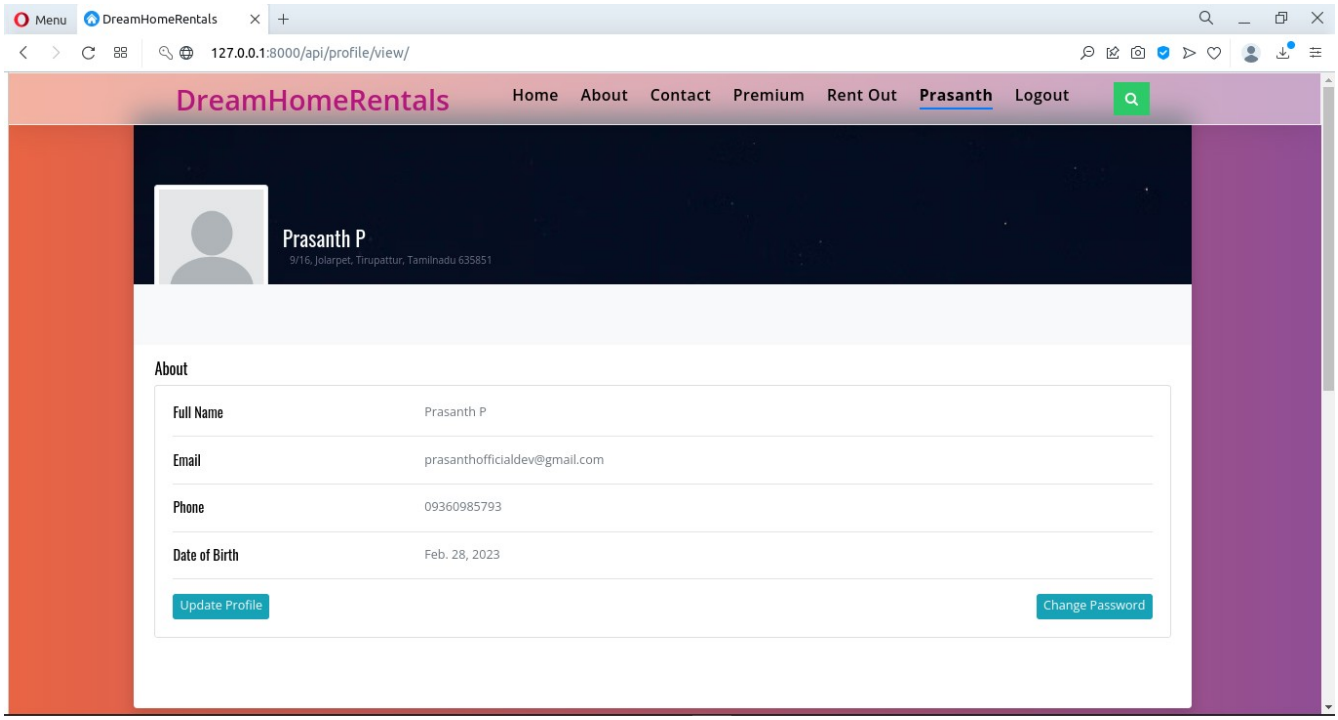

Menu
DreamHomeRentals
127.0.0.1:8000/rent/home/#scroll
DreamHomeRentals
Home
About
Contact
Register
Login
LOOKING FOR ROO
VERIFY PHONE NUMBER
Phone no *
phone number
Generate OTP

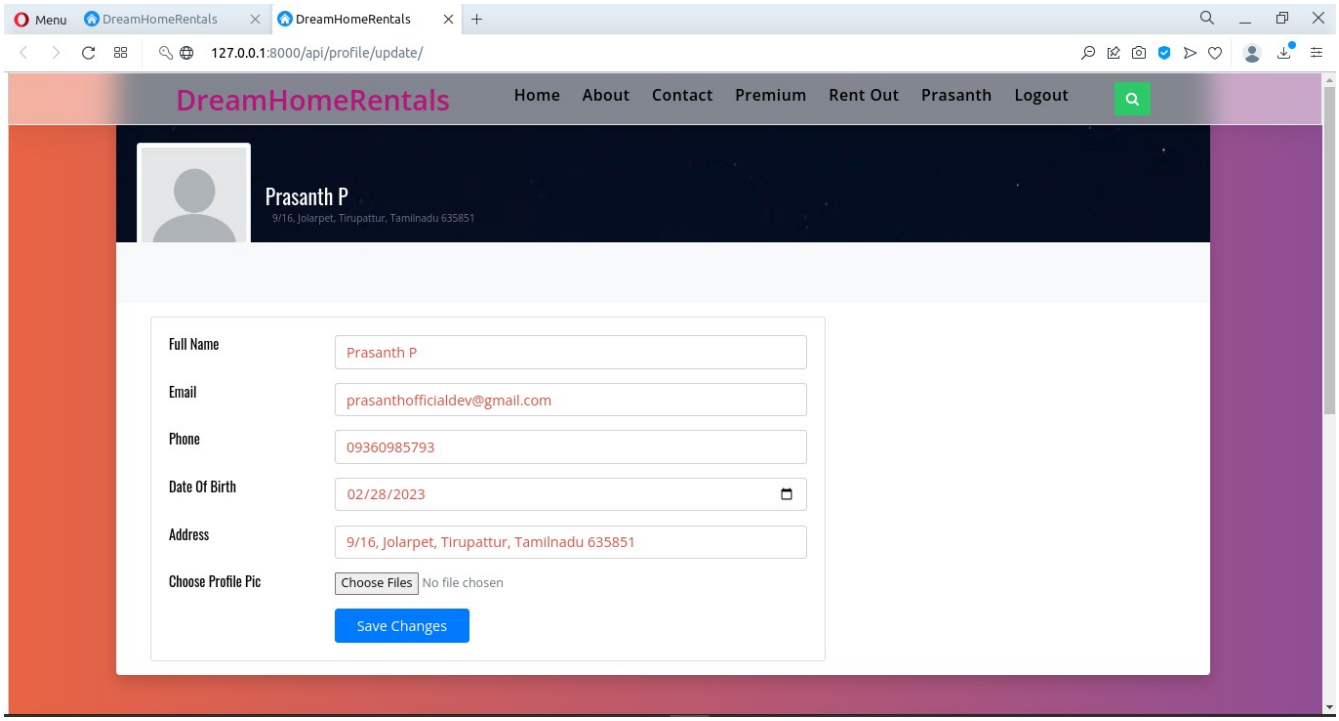

Menu
DreamHomeRentals
Select room to change
127.0.0.1:8000/rent/allproperty/
DreamHomeRentals
Home
About
Contact
Validate
Premium
Rent Out
admin
Logout
Flats Available on Rent
Home / Properties Grid
For Rent
Diwa Buildings
Location : Jolarpet
RENT | 6000 RS
Click here to view
Ragav Buildings
Location : Tirupattur
RENT | 2000 RS
Click here to view
Ragav Buildings
Location : Tirupattur
RENT | 3000 RS
Click here to view
1
2
3

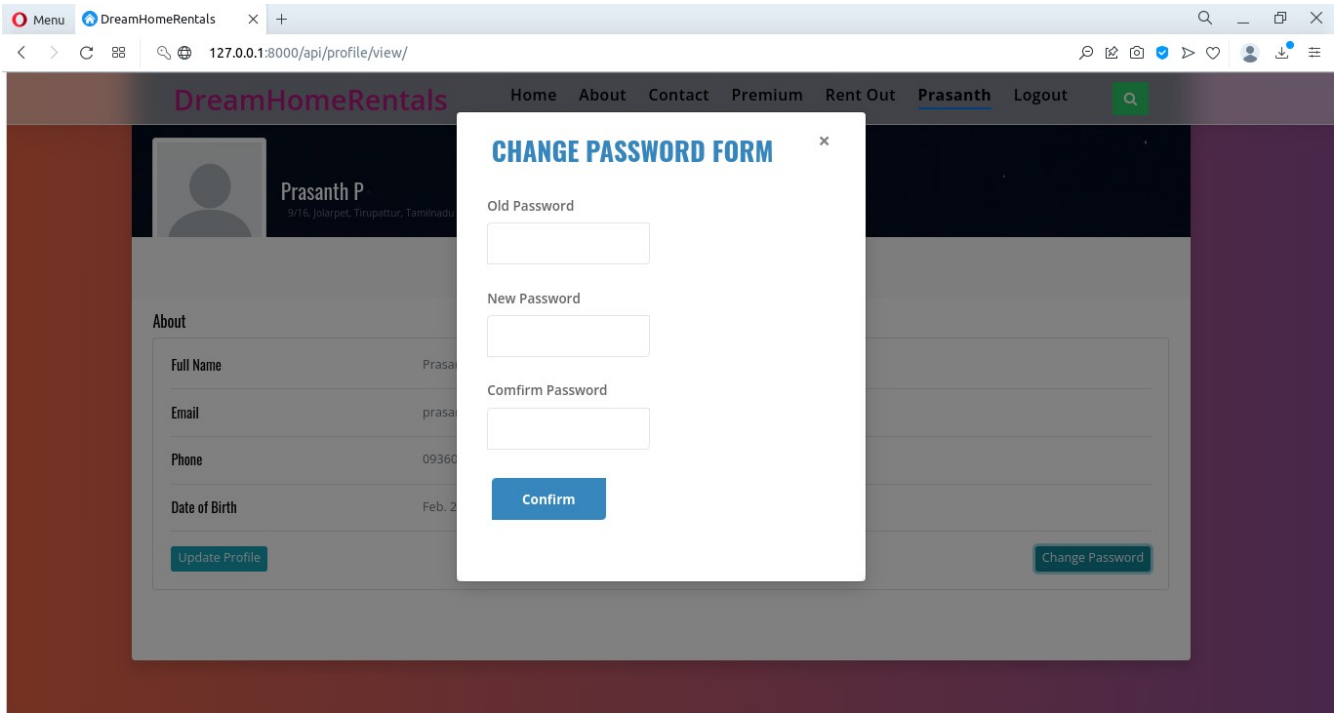

Menu
DreamHomeRentals
127.0.0.1:8000/rent/Detail/17
DreamHomeRentals
Home
About
Contact
Validate
Premium
Rent Out
admin
Logout
Diwa Buildings
Jolarpet
Update This Post / Deactivate This Post
Bed Room Image

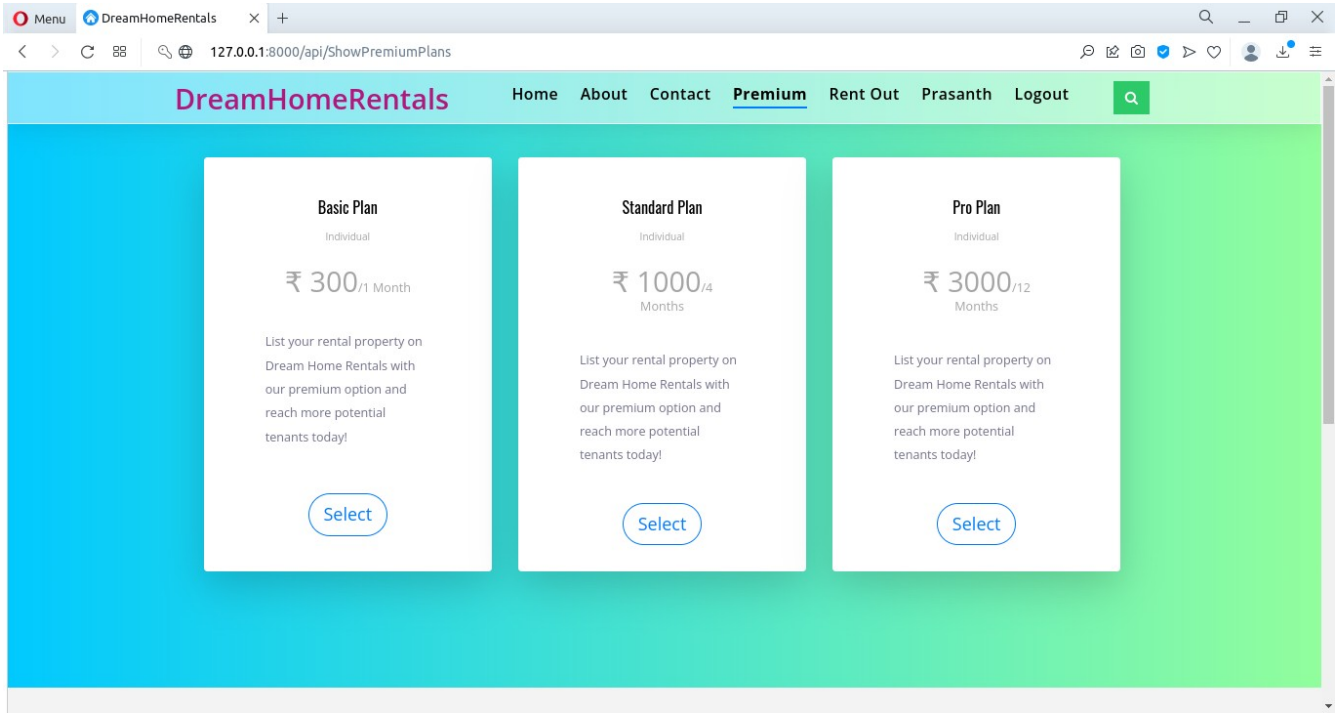

Menu
DreamHomeRentals
127.0.0.1:8000/rent/Detail/17
DreamHomeRentals
Home
About
Contact
Validate
Premium
Rent Out
admin
Logout
Building Location
Vaniyambadi
Tamil Nadu
Directions
View larger map
Water Falls
Vinnamangalam
Melsannapo
Elayanagaram
Marapattu
kammiyambattu
Simugampattu
Thumberi
Uthayendram
Thimmampettai
Jawadhu Hills
Ammavarikuppam
Vaniyambadi Bus Stand
Bhairavar Temple
Pullur
Ambalur
Kurubavanigunda
Chekkimedam
Vellakuttai
jambupallam
Vallipattu
Google
Keyboard shortcuts
Map data ©2023
Terms of Use
Report a map error
Property Description

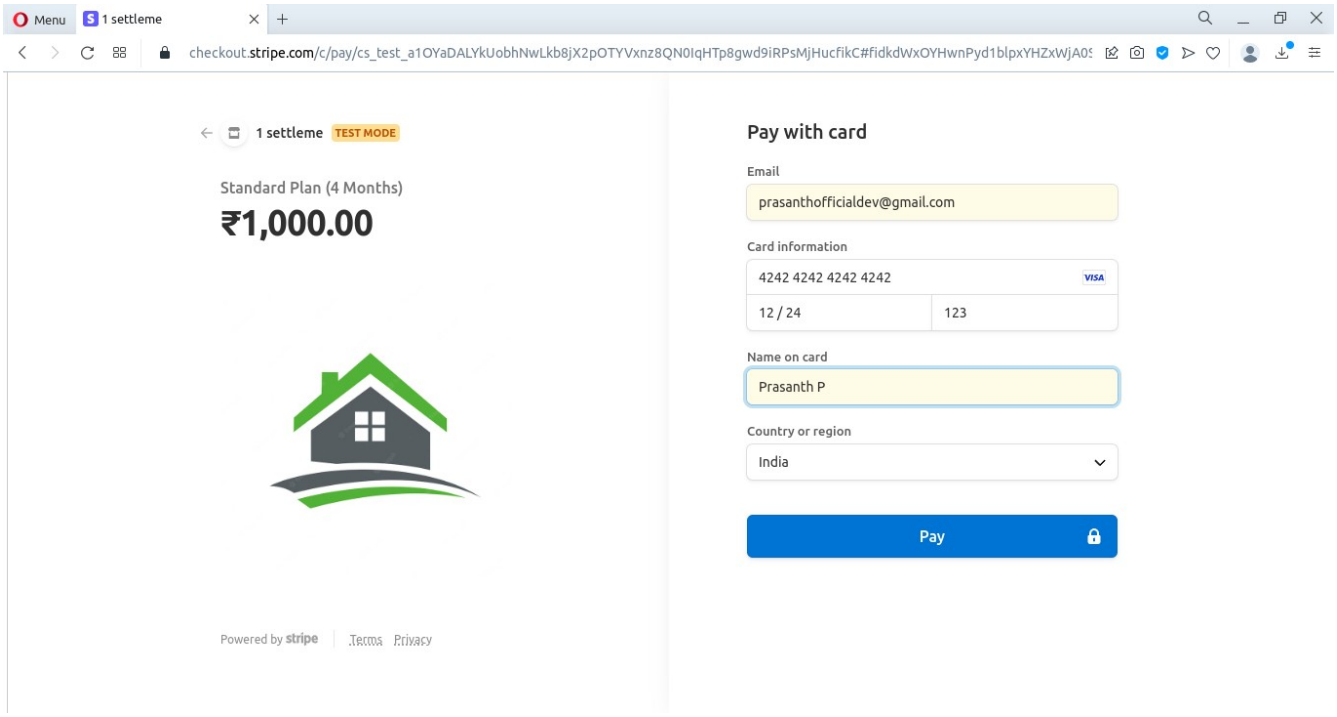

Menu
DreamHomeRentals
127.0.0.1:8000/rent/Detail/17
DreamHomeRentals
Home
About
Contact
Validate
Premium
Rent Out
admin
Logout
Property Description
Diwa Buildings
Property ID: 17
Rent Price: 6000
Advance Price: 5000
Location: Jolarpet Kodiyur
House Type: 2BHK
Allowed For: All
Status: Rent
Book Now
Welcome to our Diwa house! This charming property features a spacious living room, fully-equipped kitchen, and three comfortable bedrooms
The house has been recently renovated with new appliances and fresh paint, giving it a modern feel
The three bedrooms are all located upstairs and feature comfortable beds and ample storage space
The master bedroom has a queen-size bed and an en-suite bathroom, while the other two bedrooms have twin-size beds and share a separate bathroom
In addition to the indoor space, the house also has a lovely backyard with a patio area and outdoor seating
It's the perfect spot to relax and enjoy the beautiful weather
Located in a quiet and friendly neighborhood, this house is just a short drive from all the amenities you need, including supermarkets, restaurants, and shops
It's also within easy reach of several parks and hiking trails, making it the ideal base for outdoor enthusiasts
Features
Balcony
Deck
Parking
Outdoor Kitchen
Tennis Courts
Sun Room
Cable Tv
Internet
Concrete Flooring

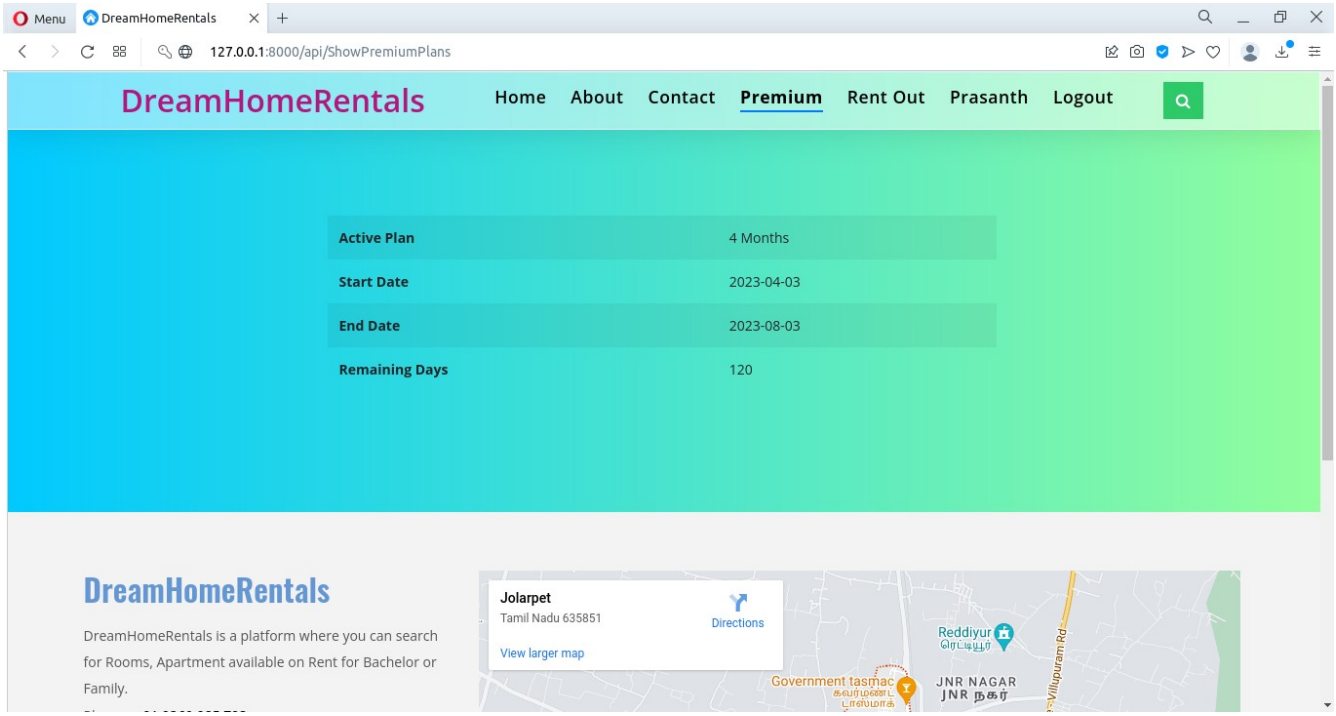

Menu
DreamHomeRentals
127.0.0.1:8000/api/profile/view/
DreamHomeRentals
Home
About
Contact
Premium
Rent Out
Prasanth
Logout
Prasanth P
9/16, Jolarpet, Tirupattur, Tamilnadu 635851
About
Full Name
Prasanth P
Email
prasanthofficialdev@gmail.com
Phone
09360985793
Date of Birth
Feb. 28, 2023
Update Profile
Change Password

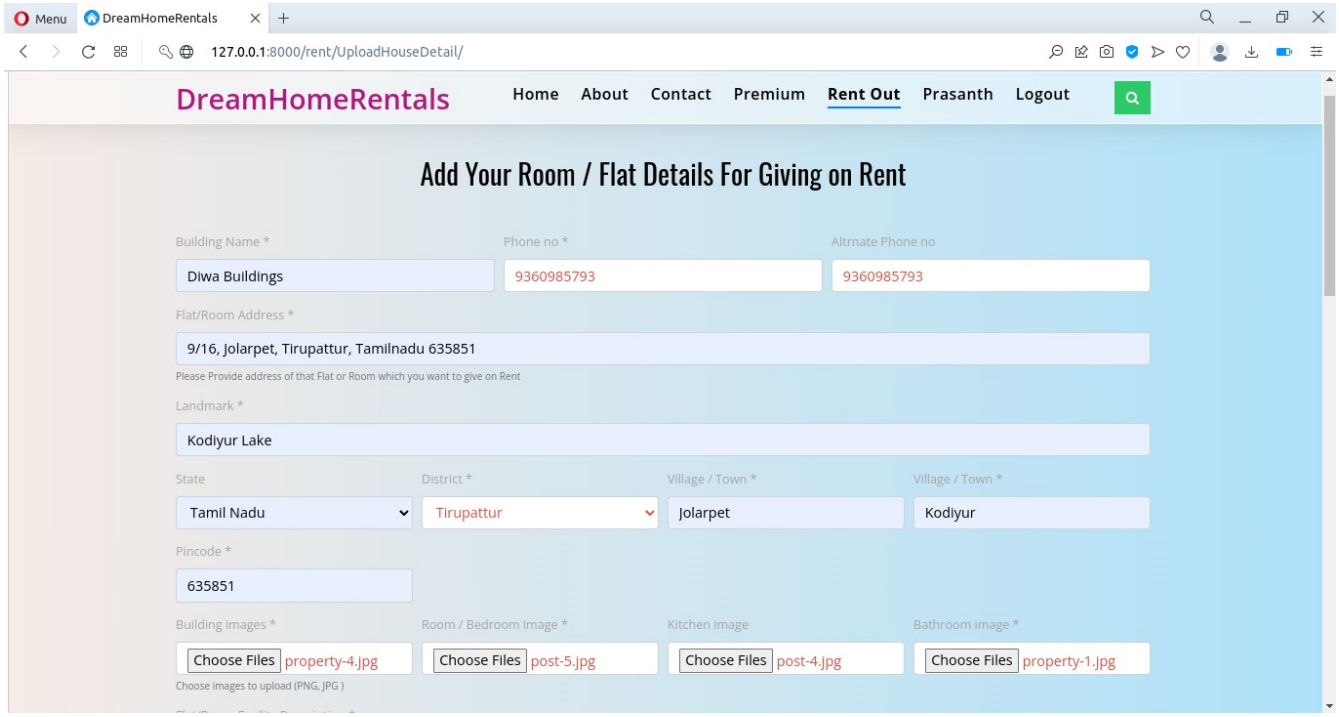

Menu
DreamHomeRentals
DreamHomeRentals
127.0.0.1:8000/api/profile/update/
DreamHomeRentals
Home
About
Contact
Premium
Rent Out
Prasanth
Logout
Prasanth P
9/16, Jolarpet, Tirupattur, Tamilnadu 635851
Full Name
Prasanth P
Email
prasanthofficialdev@gmail.com
Phone
09360985793
Date Of Birth
02/28/2023
Address
9/16, Jolarpet, Tirupattur, Tamilnadu 635851
Choose Profile Pic
Choose Files
No file chosen
Save Changes

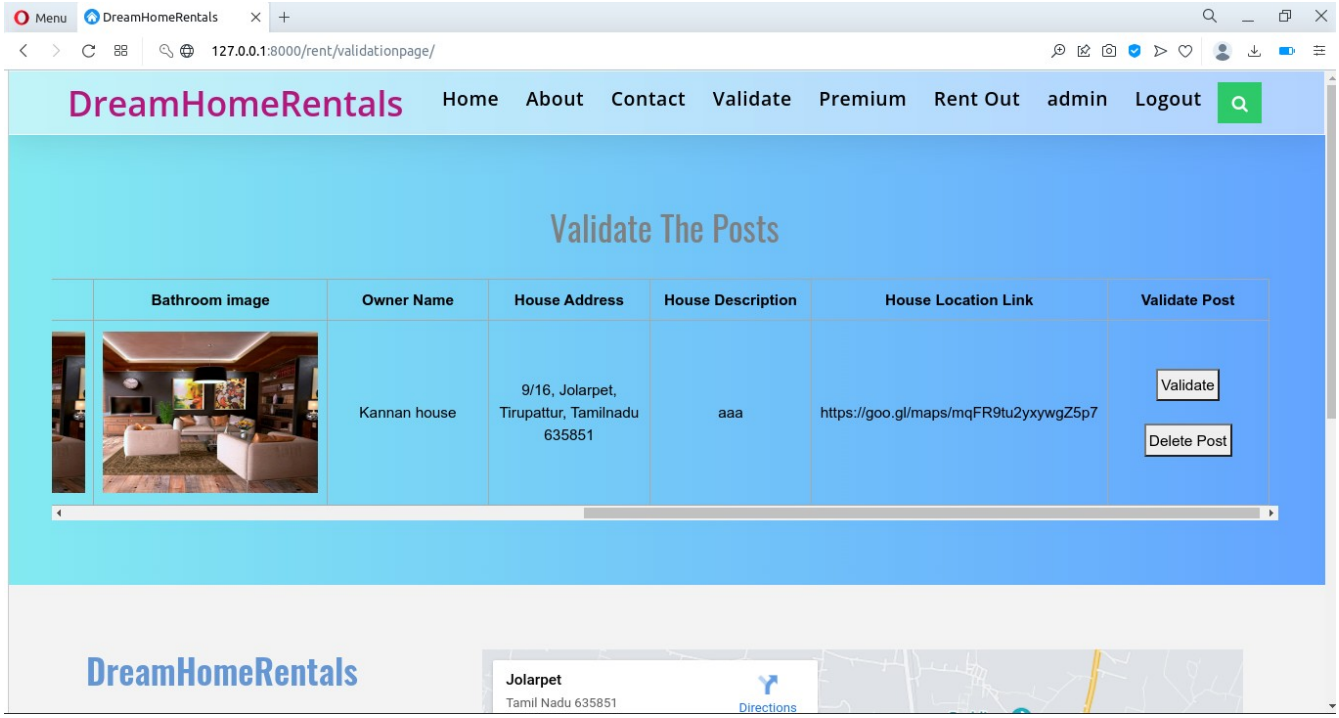

Menu
DreamHomeRentals
127.0.0.1:8000/api/profile/view/
DreamHomeRentals
Home
About
Contact
Premium
Rent Out
Prasanth
Logout
Prasanth P
About
Full Name
Email
Phone
Date of Birth
Update Profile
Change Password
CHANGE PASSWORD FORM
Old Password
New Password
Comfirm Password
Confirm


Menu
DreamHomeRentals
127.0.0.1:8000/api/ShowPremiumPlans
DreamHomeRentals
Home
About
Contact
Premium
Rent Out
Prasanth
Logout
Basic Plan
Individual
₹ 300/1 Month
List your rental property on Dream Home Rentals with our premium option and reach more potential tenants today!
Select
Standard Plan
Individual
₹ 1000/4 Months
List your rental property on Dream Home Rentals with our premium option and reach more potential tenants today!
Select
Pro Plan
Individual
₹ 3000/12 Months
List your rental property on Dream Home Rentals with our premium option and reach more potential tenants today!
Select


Menu
1 settleme
checkout.stripe.com/c/pay/cs_test_a1OYaDALYkUobhNwLkb8jX2pOTYVxnz8QN0IqHTp8gwd9iRPsMjHucfikC#fidkdWxOYHwnPyd1blpxYHZxWjA0
1 settleme
TEST MODE
Standard Plan (4 Months)
₹1,000.00
Powered by stripe
Terms
Privacy
Pay with card
Email
prasanthofficialdev@gmail.com
Card information
4242 4242 4242 4242
12 / 24
123
Name on card
Prasanth P
Country or region
India
Pay


Menu
DreamHomeRentals
127.0.0.1:8000/api/ShowPremiumPlans
DreamHomeRentals
Home
About
Contact
Premium
Rent Out
Prasanth
Logout
Active Plan
4 Months
Start Date
2023-04-03
End Date
2023-08-03
Remaining Days
120
DreamHomeRentals
DreamHomeRentals is a platform where you can search for Rooms, Apartment available on Rent for Bachelor or Family.
Jolarpet
Tamil Nadu 635851
Directions
View larger map
Reddiyur
Government tasmac
JNR NAGAR
Villupuram Rd


Menu
DreamHomeRentals
127.0.0.1:8000/rent/UploadHouseDetail/
DreamHomeRentals
Home
About
Contact
Premium
Rent Out
Prasanth
Logout
Add Your Room / Flat Details For Giving on Rent
Building Name *
Diwa Buildings
Phone no *
9360985793
Altrnate Phone no
9360985793
Flat/Room Address *
9/16, Jolarpet, Tirupattur, Tamilnadu 635851
Please Provide address of that Flat or Room which you want to give on Rent
Landmark *
Kodiyur Lake
State
Tamil Nadu
District *
Tirupattur
Village / Town *
Jolarpet
Village / Town *
Kodiyur
Pincode *
635851
Building images *
Choose Files property-4.jpg
Room / Bedroom image *
Choose Files post-5.jpg
Kitchen image
Choose Files post-4.jpg
Bathroom image *
Choose Files property-1.jpg
Choose images to upload (PNG, JPG)


Menu
DreamHomeRentals
127.0.0.1:8000/rent/validationpage/
DreamHomeRentals
Home
About
Contact
Validate
Premium
Rent Out
admin
Logout
Validate The Posts
Bathroom image
Owner Name
House Address
House Description
House Location Link
Validate Post
Kannan house
9/16, Jolarpet, Tirupattur, Tamilnadu 635851
aaa
https://goo.gl/maps/mqFR9tu2yxywgZ5p7
Validate
Delete Post
DreamHomeRentals
Jolarpet
Tamil Nadu 635851
Directions

The Dream Home Rentals project is developed to provide an easy and convenient platform for tenants to find their next rental house and for house owners to list their rental house. It offers a secure login system and a validated post module to ensure the accuracy of house information, saving both tenants and house owners valuable time. The system has been developed based on the specifications and requirements of the target users, with a focus on user-friendly interface and efficient functionality. The project has been developed in such a way that it can be easily updated and expanded to include new features.

Future Enhancement:

- Implementation of a recommendation system to suggest properties to users based on their search history and preferences.
- Integration with social media platforms to provide users with more options for sharing and promoting their rental listings.
- Use of machine learning algorithms to optimize search results and provide more relevant property listings to users.
- Implementation of a chat bot system to provide 24/7 support to users and answer their questions in a timely manner.

BOOKS:

1. Django for APIs: Build web APIs with Python & Django by **William S. Vincent**

ONLINE REFERENCE:

1. https://chat.openai.com/chat/
2. https://getbootstrap.com
3. https://www.w3schools.com
4. https://docs.djangoproject.com/en/4.1/
5. https://www.twilio.com