

**SUMMER INTERNSHIP PROJECT REPORT**

**On**

**“Real Estate Portal”**

***Submitted in the partial fulfillment of the requirement for the award of***

***Master of Computer Applications Noida International University, Greater Noida***

###### 

###### Submitted by: Akash Rawat

***ACADEMIC SESSION (2022 – 2024)***

**Backend Developer MCA 2nd Semester**

**Premad Software Solutions Roll No: 12214142008**

**Location: India Date: 23/05/2023**

**Under the guidance of**

**Deepti Gautam**

**Asstt Professor**

**TO WHOMSOEVER IT MAY CONCERN**

**This is to certify that Akash Rawat a student of MCA from Noida International University, (Batch 2022-2024), Roll No: 12214142008 has undergone his Internship on the topic Real Estate Portal under my guidance and supervision of Deepti Gautam.**

**Internship Guide Name : Deepti Gautam**

**Designation : Asstt. Professor**

**Company : Sunstone**

**Date: 05/08/2023**

**Certificate for Internship**

**Dear Akash,**

**I am writing to congratulate you on completing your 2-month internship in Node.js at Premad Software Solutions. It is an outstanding achievement that speaks volumes about your dedication, hard work, and skills in the field.**

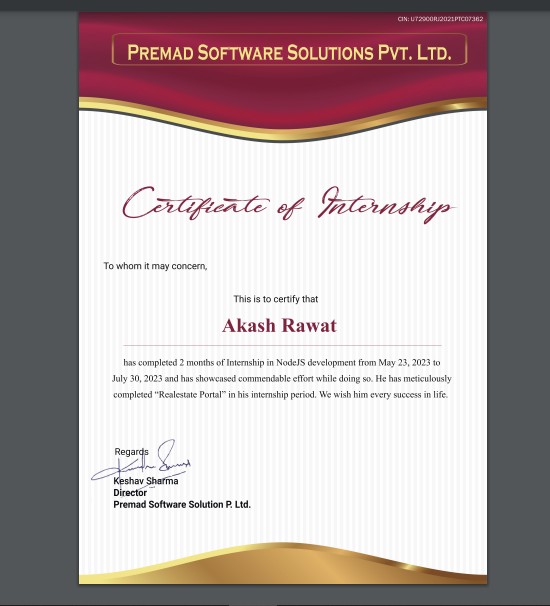
**You have shown exceptional enthusiasm and willingness to learn during your internship. Your ability to grasp complex concepts quickly and apply them efficiently has been commendable.**

**Once again, congratulations on successfully completing your internship. It has been a pleasure having you as part of our team, and we wish you all the best in your future endeavors. Your certificate for participation is attached to the same email. Please do not hesitate to reach out if you need any further assistance or support.**

Signature of Head of Institution:

Dr. Atul Sharma

Director



**Table of Contents:**

1. [Introduction 7](#_TOC_250005)
   * Background
   * Objectives
2. [Project Overview 8](#_TOC_250004)
   * Description
   * Technologies Used
3. [Role and Responsibilities 10](#_TOC_250003)
   * Backend Development
   * Database Design and Management
   * API Development
   * Testing and Debugging
4. [Challenges Faced 11](#_TOC_250002)
   * Scalability
   * Performance Optimization
   * Complex Queries
5. [Achievements 11](#_TOC_250001)
   * Successfully Developed RESTful APIs
   * Efficient Database Schema Design
   * Issue Resolution and Bug Fixes
6. [Skills Acquired 12](#_TOC_250000)
   * Node.js
   * MySQL
   * API Development
   * Database Optimization
   * Problem-Solving

###### Work 13

* + Training Tasks
  + FRD (Functional Requirement Documentation)
  + Database Schema
  + Database relations/tables
  + API Development
  + Admin Dashboard

###### Lessons Learned 66

* + Importance of Code Efficiency
  + Collaboration in Development
  + Testing and Debugging Best Practices

###### Future Recommendations 67

* + Scaling the System
  + Security Enhancements
  + Incorporating Advanced Features

###### Conclusion 68

* + Summary of Internship Experience
  + Acknowledgment

## Introduction:

**Background**: This internship report documents the experience and learning gained during the internship period as a Backend Developer at Premad Software Solutions. The role involved working on backend development tasks using Node.js and MySQL for a web application.

**Objectives**: The primary objectives of the internship were to gain practical experience in backend development, understand the integration of databases with web applications, and enhance skills in creating efficient and scalable APIs.

Embarking on a journey that merges the realms of technology and real estate, this internship report unveils the intricate workings of the backend development driving the dynamic property site website. Within these pages lies an exploration of the pivotal role played by backend development in shaping the digital landscape of property transactions and experiences.

As the real estate industry evolves, so too does the digital landscape that supports it. The backend development of our property site website stands as a testament to innovation and efficiency, where data intricacies,

server-side logic, and seamless integrations converge to create a platform that redefines how properties are discovered, evaluated, and acquired.

## Project Overview:

**Description**: The project aimed to develop the backend infrastructure for a web-based application. The application facilitated user registration, authentication, data retrieval, and manipulation. The primary focus was on creating RESTful APIs to serve client-side requests and managing the associated database interactions.

**Technologies Used**: The project heavily utilized Node.js for server-side scripting and API development, along with MySQL for data storage and retrieval. Express.js framework was employed to simplify routing and middleware handling.

**Key Deliverables:**

Throughout the course of this internship, key deliverables were achieved, showcasing the progress and impact of the backend development project:

Database Schema and Design: A comprehensive database schema was crafted to efficiently organize property data, enabling efficient querying and data retrieval.

RESTful APIs: A suite of RESTful APIs was developed to facilitate communication between the frontend and backend, allowing for smooth user interactions and data transfer.

User Authentication System: A secure user authentication system was implemented to safeguard user accounts and enable seamless logins and registrations.

Third-Party Integrations: External APIs were seamlessly integrated to enhance property listings with geolocation services, mapping functionalities, and up-to-date market insights.

Performance Optimization: Techniques such as caching, indexing, and query optimization were employed to ensure swift response times and an enhanced user experience.

Documentation: Comprehensive documentation detailing the backend architecture, APIs, and security protocols was generated to facilitate future development and maintenance.

Scalability and Performance: Building a scalable backend architecture capable of handling increasing user traffic and ensuring quick response times even during peak usage.

Collaboration and Communication: Fostering collaboration between cross-functional teams, including frontend developers, designers, and stakeholders, to ensure a seamless integration of backend and frontend components.

## Role and Responsibilities:

During the internship, the following responsibilities were undertaken:

**Backend Development**: Participated in designing and implementing the server-side logic using Node.js and Express.js. Created API endpoints for various functionalities such as user registration, login, data retrieval, and updates.

**Database Design and Management**: Designed the MySQL database schema, including tables, relationships, and constraints. Implemented CRUD (Create, Read, Update, Delete) operations using SQL queries.

**API Development**: Developed RESTful APIs that communicated with the frontend application. Ensured proper request handling, response formats, and error management.

**Testing and Debugging**: Conducted testing of API endpoints using tools like Postman. Identified and resolved bugs, improved error handling, and enhanced API performance.

## Challenges Faced:

**Scalability**: Ensuring that the backend architecture could handle an increasing number of users and requests while maintaining performance.

**Performance Optimization**: Optimizing database queries and server response times to provide a smooth user experience.

**Complex Queries**: Dealing with complex SQL queries involving multiple joins and aggregations to retrieve specific data from the database.

## Achievements:

Successfully developed and deployed several RESTful APIs to handle user registration, login, and data manipulation.

Designed an efficient MySQL database schema that facilitated organized data storage and retrieval.

Resolved critical issues and bugs, improving the overall reliability of the application.

## Skills Acquired:

\*\*1. Database Management: Expertise in designing and managing relational databases (e.g., MySQL, PostgreSQL), including data modeling, indexing, and optimizing queries for efficient data retrieval.Understanding of MySQL database design, optimization, and CRUD operations.

\*\*2. API Integration: Skill in integrating external APIs to enrich platform functionalities, such as geolocation services, mapping, and real-time market data, enhancing user experiences.

\*\*4. Server-Side Logic: Ability to develop robust server-side logic that processes user requests, handles data manipulation, and ensures efficient communication between different components.

\*\*5.Security Implementation: Knowledge of implementing security measures, including user authentication, encryption, and secure data transmission, to protect sensitive user information.

\*\*6. Version Control: Proficiency in using version control systems like Git to track code changes, collaborate with team members, and manage different code branches.

## Work:

· **Tasks-** I got various tasks from Company first to improve my knowledge about Nodejs.

|  |  |
| --- | --- |
| Day 1 | Install NodeJs ,npm package manager ,Node REPL |
| Day 2 | Export Modules ,Create a Web Server , CRUD Operation In fs |

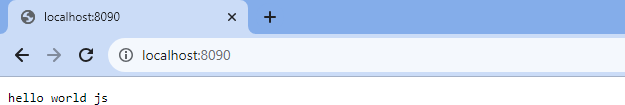
**//create a web server const http=require('http');**

**const server=http.createServer((req,res)=>{**

**//set the response headers res.writeHead(200,{'content-type':'text'});**

**//send the response res.end("hello world js");**

**}).listen(8090);**



**//The fs.createFile() method is used to create files on your computer.**

**//append**

**var fs=require('fs'); fs.appendFile('myfile6.txt','hello dear',function(err){**

**if (err) throw error; console.log('saved');**

**});**

**//write**

**const { error } = require('console'); var fs=require('fs');**

**fs.writeFile('myfile2.txt','hello Akash',function(err){ if (err) throw error;**

**console.log('saved')});**

**//rename**

**var fs=require('fs'); fs.rename('myfile2.txt','Akash.txt',function(err){**

**if (err) throw error; console.log('saved');});**

**//update fs.appendFile fs.writeFile**

**//delete**

**var fs=require('fs'); fs.unlink('myfile6.txt',function(err){**

**if (err) throw error; console.log('deleted') });**

asynchronous and synchronous in fs module, Debugging ,Event Emitter

Day 3

**//Synchronous operations are blocking, which means that they will halt the execution of the program until the operation is completed.**

**const fs=require('fs');**

**try{ const data=fs.readFileSync('test.txt','utf-8'); console.log(data);**

**console.log('file read operation completed');**

**}catch(error){console.error(error);}**

#### Output

**hello world all of you**

**file read operation completed**

**//Asynchronous operations, on the other hand, are non-blocking and allow the program to continue executing while the file operation is being performed.**

**const fs=require('fs'); fs.readFile('test.txt','utf-8',(err,data)=>{**

**console.log(data);}) console.log('operation completed') Output:**

**operation completed hello world all of you**

**//event emitter- EventEmitter class, which we'll use to handle our events.**

**const EventEmitter = require('events'); const eventEmitter = new EventEmitter();**

**//Create an event handler:**

**var myEventHandler = function () { console.log('I hear a scream!');}**

**Output:**

**I hear a scream!**

**//using arguments**

**eventEmitter.on('start', number => {console.log(`started ${number}`)}); eventEmitter.emit('start', 23);**

**//multiple eventEmitter.on('start',(start,end) => {**

**console.log(`started from ${start} to ${end}`)}); eventEmitter.emit('start', 1,5);**

Output:

**started 23**

**started 1**

**started from 1 to 5**

path module,Custom Module,nodemon,https handle Routing

Day 4

**//The path module is a built-in module in Node.js that provides utilities for working with file and directory paths.**

**const path=require('path');**

**var file1=path.basename('/Users/Refsnes/demo\_path.js'); console.log(file1)**

**var file2=path.dirname('/Users/Refsnes/demo\_path.js'); console.log(file2)**

**console.log(path.delimiter); //semicolon for windows var file3 = { dir: 'C:\\Users\\Refsnes', base: 'demo\_path.js' } console.log(path.format(file3));**

**Output: demo\_path.js**

**/Users/Refsnes C:\Users\Refsnes\demo\_path.js c\users\demo\_path.js**

**{ root: '',**

**dir: '',**

**base: 'demo\_path.js', ext: '.js',**

**name: 'demo\_path'**

**}**

**// http routing-route the endpoints at the http server const http=require('http');**

**const server=http.createServer((req,res)=>{ const url=req.url;**

**if(url=='/'){**

**res.writeHead(200,{'content-type':'text'}); res.end("homepage");**

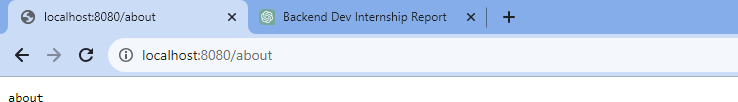
**}else if(url=='/about'){ res.writeHead(200,{'content-type':'text'}); res.end("about");**

**}else if(url=='/contact'){**

**res.writeHead(200,{'content-type':'text'}); res.end("contact");}**

**}).listen(8080);**





All About JSON, fetch data from api,event modules,

Day 5

**//read json file= helps us to read any json file const json=require("./user.json"); console.log(json);**

**Output:**

**[ { name: 'akash', age: '23' },**

**{ study: { ug: 'ignou', pg: 'niu' }, course: 'MCA' },**

**{ name: 'happy', age: '45', lang: [ 'eng', 'hindi', 'tamil' ] },**

**{ name: 'happy', age: '45', lang: [ 'eng', 'hindi', 'tamil' ] }]**

**//write json= helps to write any json file const fs=require('fs');**

**const user=require("./user.json"); //read**

**let users={"name":"happy","age":"45","lang":["eng","hindi","tamil"]}; user.push(users); //add data to user file fs.writeFile("user.json",JSON.stringify(user), err=>{**

**if (err) throw error; console.log("done",user)})**

**Output:**

**done [**

**{ name: 'akash', age: '23' },**

**{ study: { ug: 'ignou', pg: 'niu' }, course: 'MCA' },**

**{ name: 'happy', age: '45', lang: [ 'eng', 'hindi', 'tamil' ] },**

**{ name: 'happy', age: '45', lang: [ 'eng', 'hindi', 'tamil' ] },**

**{ name: 'happy', age: '45', lang: [ 'eng', 'hindi', 'tamil' ] }]**

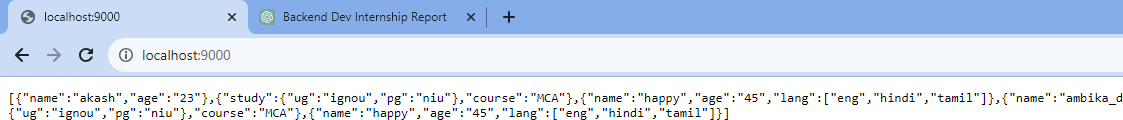
**//create a simple API**

**const http=require('http'); const user=require('./data.js'); http.createServer((req,res)=>{**

**res.writeHead(200,{'content-type':'application\json'}); res.write(JSON.stringify(user));**

**res.end();**

**}).listen(9000);**



Stream Buffer And Pipes scope and global variable

Day 6

**// buffer- It is a temporary memory in RAM that manages a block of data and sends it for processing**

**const buffer=new Buffer.from("Akash")**

**console.log(buffer); //represent code of akash in hexadecimal console.log(buffer.toJSON()); //represent code in decimal console.log(buffer.toString())**

**buffer.write('cod'); //replace first 3 letters with cod console.log(buffer);**

**console.log(buffer.toString())**

**//alloc() Method: It creates a Buffer object of the given length let buff=new Buffer.alloc(5);**

**console.log(buff) Output:**

**<Buffer 41 6b 61 73 68>**

**{ type: 'Buffer', data: [ 65, 107, 97, 115, 104 ] }**

|  |  |  |
| --- | --- | --- |
| **Akash** |  | |
| **<Buffer** | **63** | **6f 64 73 68>** |
| **codsh** |  |  |
| **<Buffer** | **00** | **00 00 00 00>** |
| Day 7 |  | callback function,promises |

**//callback function**

**//A callback is a function that is called when a task is completed, thus helping in preventing any kind of blocking and a callback function allows other code to run in the meantime.**

**// The readFileSync() function is synchronous and blocks execution until finished.**

**//The function blocks the program until it reads the file and then only it proceeds to end the program**

**const { error } = require('console'); const { promises } = require('dns'); const fs=require('fs');**

**const a=fs.readFileSync('./Akash.txt'); console.log(a.toString()); console.log("mission successfull") Output:**

**hello Akash**

**mission successfull**

**//Code for reading a file asynchronously (non-blocking code) in Node.js.**

**const fs=require('fs'); fs.readFile('./Akash.txt',function(err,data){**

**if (err) return console.log(error); console.log(data.toString())});**

**console.log("mission successfull");**

**//A callback function is passed which gets called when the task running in the background are finished.**

**Output:**

**mission successfull hello Akash**

**//promises- Promises are used to handle asynchronous operations in JavaScript.**

**console.log("hello one"); setTimeout(function(){**

**console.log("hello in two second");**

**},2000);**

**console.log('hello in three'); Output:**

**hello one**

**hello in three //it shows after 2 sec of code execution hello in two second**

**Ex.let promise=new Promise(function(resolve,reject){ console.log("hello");**

**resolve(23);});**

**console.log("hello one"); setTimeout(function(){**

**console.log("hello in two second");**

**},2000);**

**console.log('hello in three'); console.log(promise);**

**Output:**

**hello hello one**

**hello in three Promise { 23 }**

**hello in two second //shows at the end**

make connection between db and nodeJs, login form with validation,

Day 8

**//Installed express and sql modules. const express = require('express'); const mysql = require('mysql2'); const app = express();**

**// Database configuration- set database details to connect const db = mysql.createConnection({**

**host: 'localhost', user: 'root',**

**port:3306,**

**database: 'home'**

**});**

**//If connected successfully then it will show connected in console db.connect((err) => {**

**if (err) {**

**console.error('Error connecting to database:', err);**

**} else {**

**console.log('Connected to database') }});**

**// Start the server const port = 8000;**

**app.listen(port, () => {**

**console.log(`Server is listening on port ${port}`)}); Output:**

**Server is listening on port 8000 Connected to database**

**//for login Page**

**app.get('/', (req, res) => {**

**res.render('login')}); app.post('/login', (req, res) => {**

**const { User\_name, Password } = req.body;**

**const query = 'SELECT \* FROM user WHERE User\_name = ? AND Password**

**= ?';**

**db.query(query, [User\_name, Password], (err, results) => { if (err) {**

**console.error('Error executing query:', err); res.sendStatus(500)**

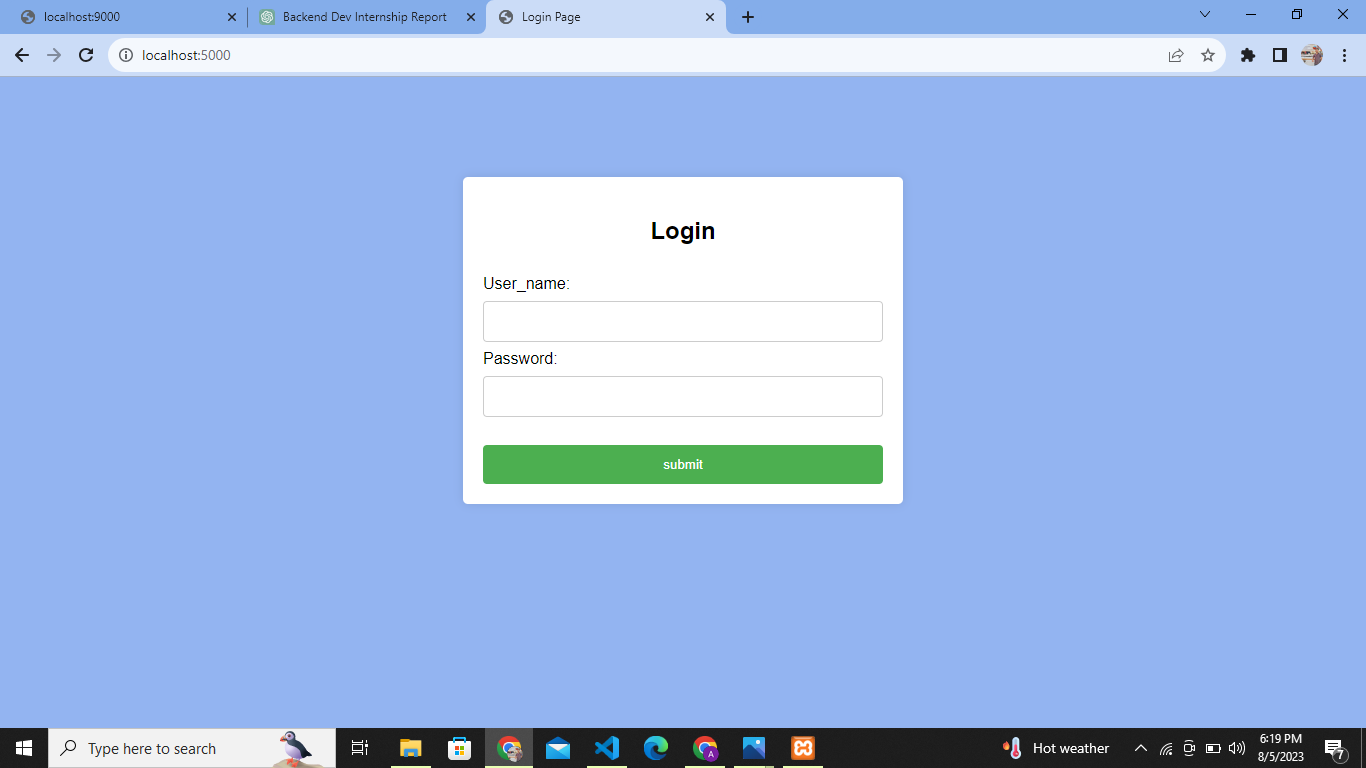
**} else if (results.length > 0) { res.send(“Login Successfully”);**

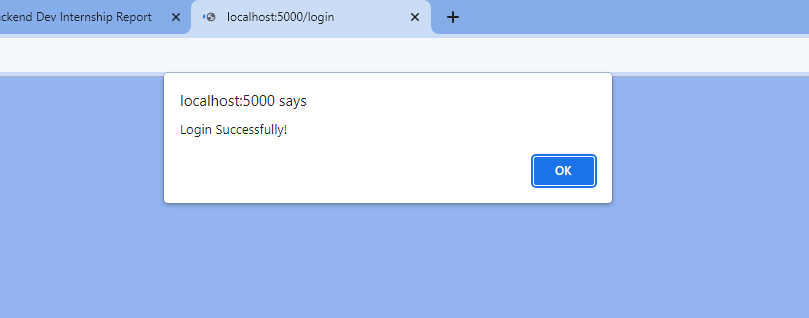
**}else{**

**res.send(“Invalid User Password”); }**

**})**

**});**





all of sql query ,CRUD operations

Day 5

**//Create**

**CREATE TABLE employee(ID INT PRIMARY KEY, First\_Name VARCHAR(20),**

**Last\_Name VARCHAR(20), Salary INT, Email\_Id VARCHAR(40));**

**1. INSERT INTO employee(ID, First\_Name, Last\_Name, Salary, Email\_Id) VALUES(1, "Neeta", "Korade", 59000, "**[**neetak12@gmail.com**](mailto:neetak12@gmail.com)**"), (2, "Sushma", "Singh", 62000, "**[**sushsingh67@gmail.com**](mailto:sushsingh67@gmail.com)**"), (3, "Kavita", "Rathod", 27000,**

[**"kavitar09@gmail.com**](mailto:kavitar09@gmail.com)**"), (4, "Mrunalini", "Deshmukh", 88000,**

**"**[**mrunald78@gmail.com**](mailto:mrunald78@gmail.com)**"), (5, "Swati", "Patel", 34000, "**[**swatip67@gmail.com**](mailto:swatip67@gmail.com)**");**

**//Read**

**SELECT \*FROM employee;**

**59000**

**ID**

**1**

**Neeta**

**Korade**

[**neetak12@gmail.com**](mailto:neetak12@gmail.com)

**2**

**Sushma**

**Singh**

[**sushsingh67@gmail.com**](mailto:sushsingh67@gmail.com)

**3**

**Kavita**

**Rathod**

[**kavitar09@gmail.com**](mailto:kavitar09@gmail.com)

**4**

**Mrunalini**

**Deshmukh**

[**mrunald78@gmail.com**](mailto:mrunald78@gmail.com)

**5**

**Swati**

**Patel**

[**swatip67@gmail.com**](mailto:swatip67@gmail.com)

**Salary**

**Last\_Name**

**First\_Name**

**34000**

**88000**

**27000**

**62000**

**Email\_Id**

**//Update**

**UPDATE employee SET Salary = "35000", Email\_Id= "**[**shwetawagh03@gmail.com**](mailto:shwetawagh03@gmail.com)**" WHERE ID = 5;**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **5** | **Swati** | **Patel** | **35000** | [**swatip67@gmail.com**](mailto:swatip67@gmail.com) |

**//Delete**

**DELETE FROM employee WHERE Salary = 27000;**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID First\_Name Last\_Name Salary Email\_Id** | | | | |
| **1** | **Neeta** | **Korade** | **59000** | [**neetak12@gmail.com**](mailto:neetak12@gmail.com) |
| **2** | **Sushma** | **Singh** | **62000** | [**sushsingh67@gmail.com**](mailto:sushsingh67@gmail.com) |
| **4** | **Mrunalini** | **Deshmukh** | **88000** | [**mrunald78@gmail.com**](mailto:mrunald78@gmail.com) |
| **5** | **Swati** | **Patel** | **34000** | [**swatip67@**gmail**.com**](mailto:swatip67@gmail.com) |
|  |  |  |  |  |

join in sql

Day 6

**// Inner join - Used to join 2 tables**

SELECT Orders.OrderID, Customers.CustomerName FROM Orders

INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;

#### Output;

**OrderID CustomerName**

10250 Hanari Carnes

10253 Hanari Carnes

10252 Suprêmes délices

10249 Tradição Hipermercados

* **FRD for Project:**

FUNCTIONAL REQUIREMENT DOCUMENT

Homeonline.com

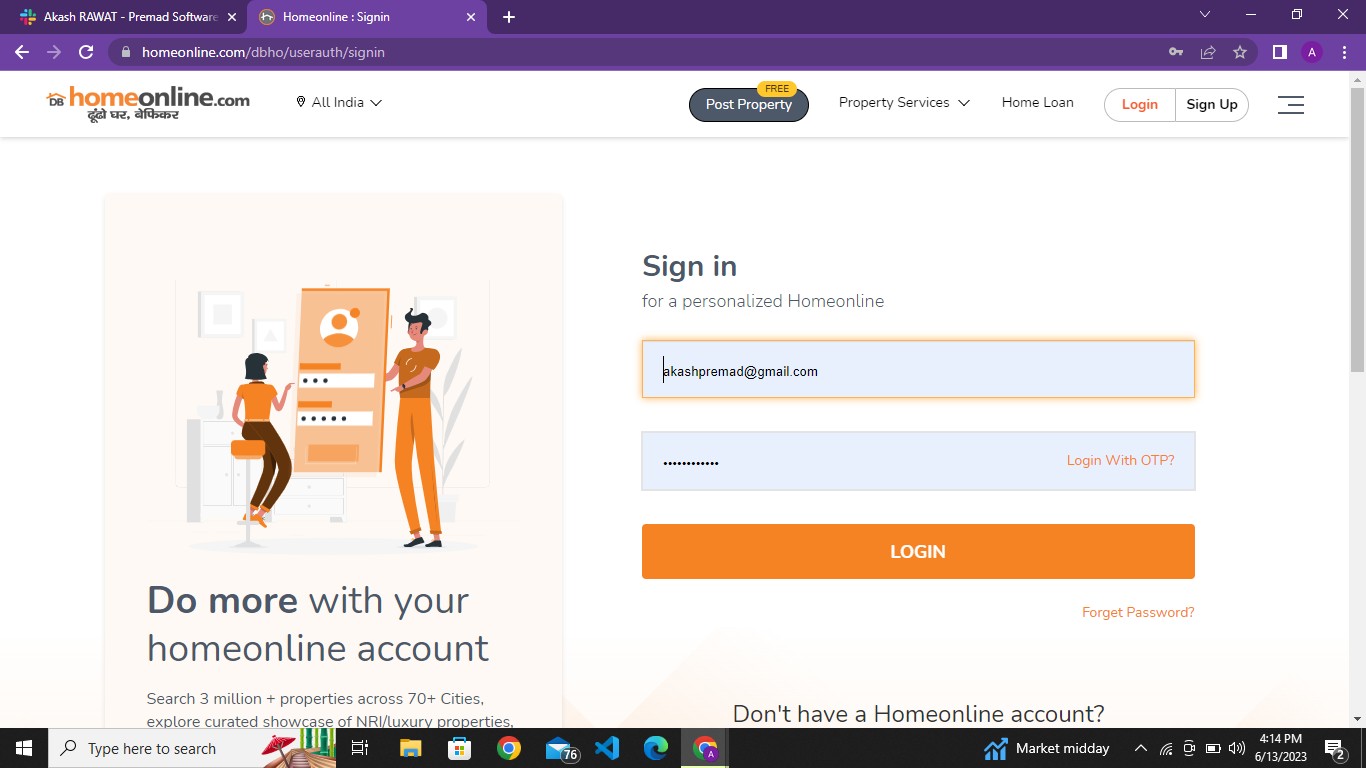
**HomeOnline.com is an online platform that connects homeowners with contractors, suppliers, and professionals in the home improvement and construction industry.**

Functionality-

# Login Module

* Users should be able to enter mobile numbers or email and the system should accept both registered and unregistered mobile numbers.
* CASE 1: If Invalid mobile number is entered
  + Show validation message as “**Please enter valid Email/Mobile number**”
* CASE 2: If valid mobile number is added,
  + Enter Password or Login with OTP.
  + Send 4 digit OTP to the entered mobile number and redirect the user to the OTP verification screen.

### Validation-

* Mobile Number
  + Allow Max 10 digits only
  + Allow valid mobile number format only
  + Allow numeric values only.

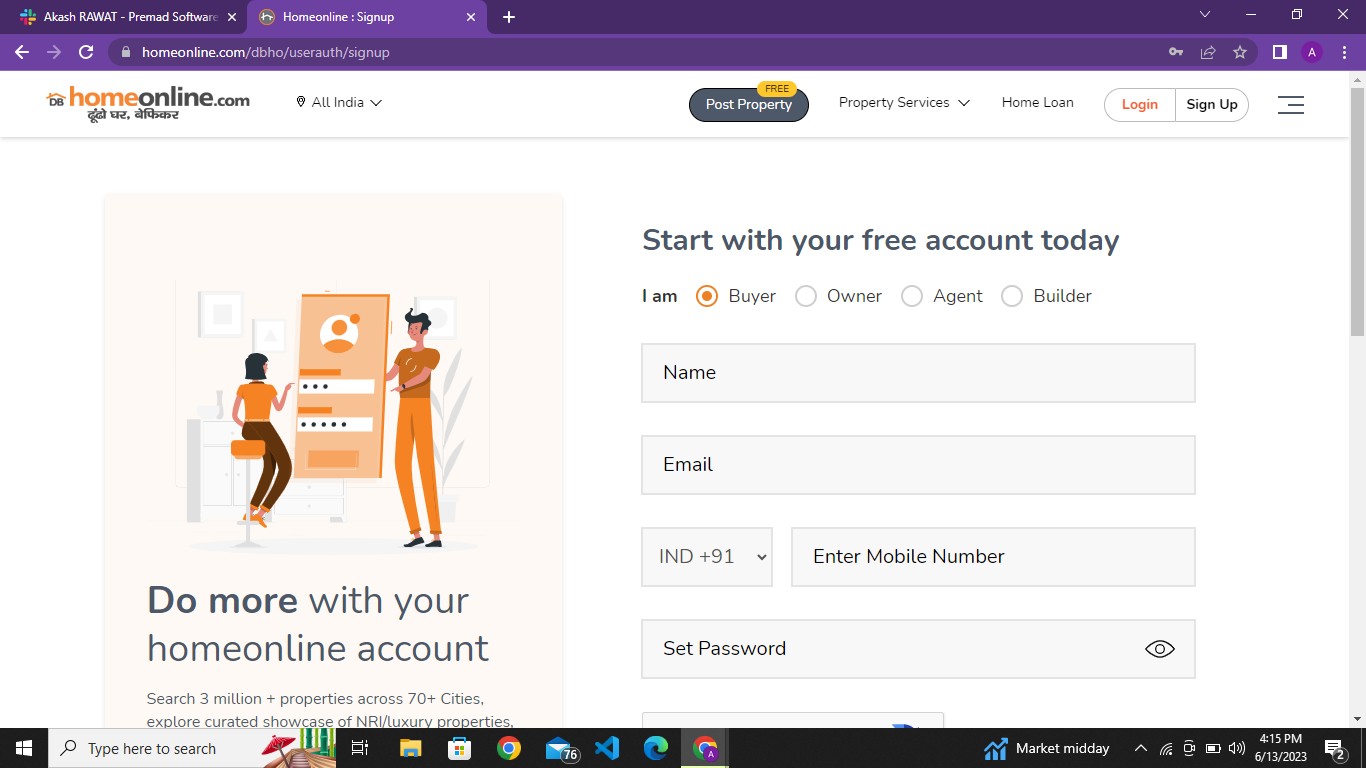
# Sign Up Module

### Functionality-

* User can make registration to the website or create his/her account at the website.
* User can select his role at the website like Buyer,Owner,Agent and Builder.
* User should have fill his name,email, Mobile number and set a strong password.
* Case 1: If Invalid mobile number is entered
  + Show validation message as “**Please enter valid Email/Mobile number**”
* CASE 2: If valid mobile number is added,
  + Send 4 digit OTP to the entered mobile number and redirect the user to the OTP verification screen.

### Validation-

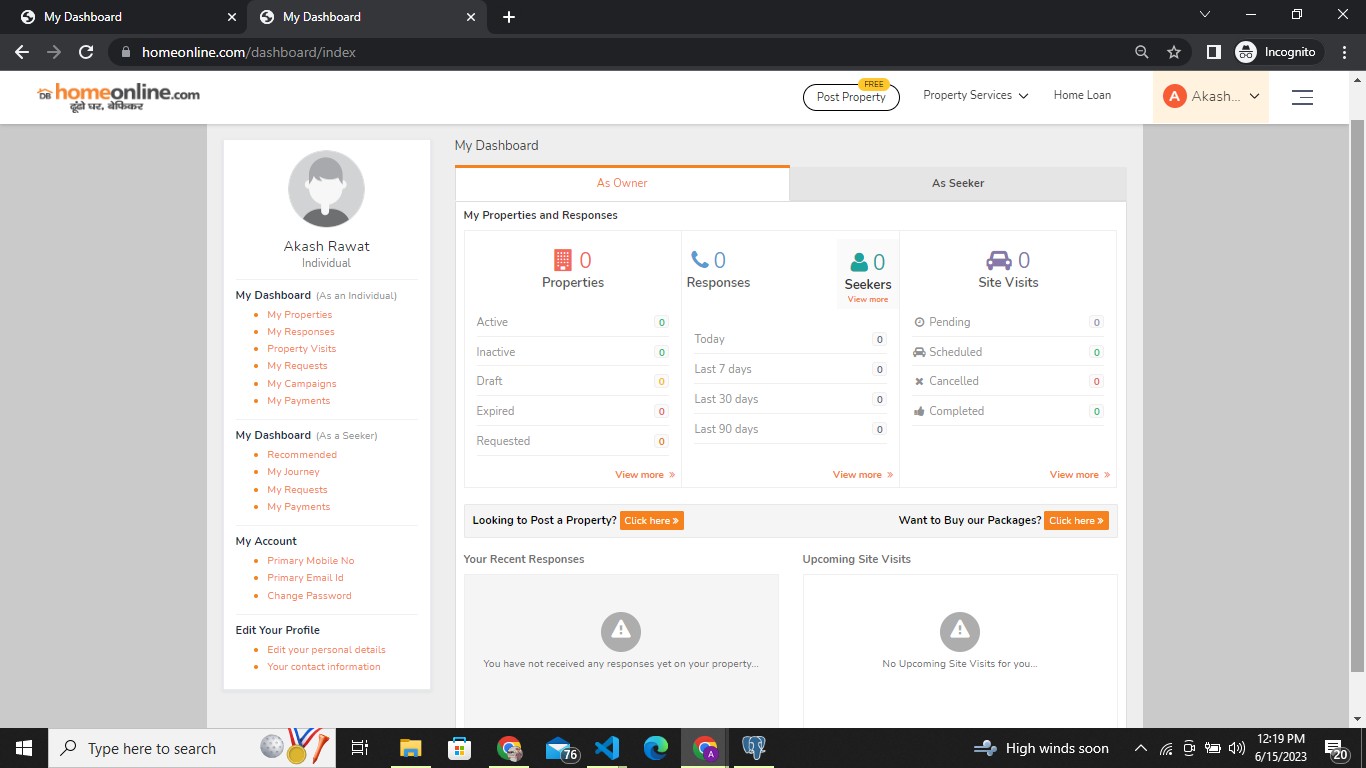
* Mobile Number
  + Allow Max 10 digits only
  + Allow valid mobile number format only
  + Allow numeric values only.



# My Dashboard Module

### Functionality-

We have multiple modules in My Dashboard Module.



1. As Owner Module- This module shows about the property which are posted by us.

Properties Module-

* + In this module it will show properties which are active, inactive, Draft,expired and requested by you.

Responses Module-

* In this module it will show Total number of seekers you got on current

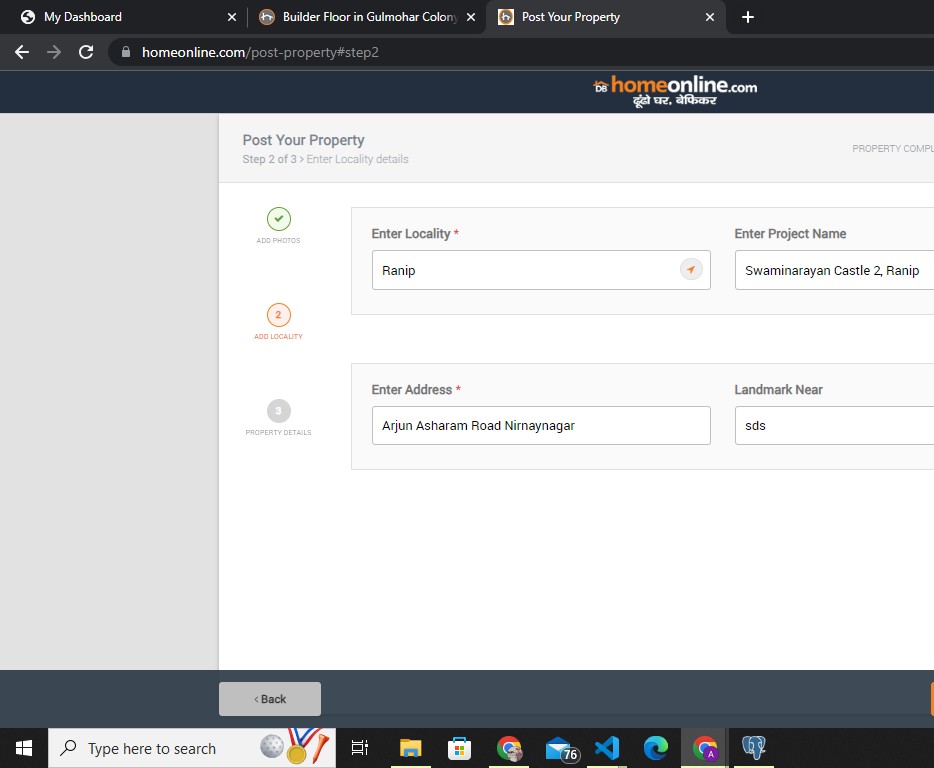
day or last 90 days.

Last Visit Module-

* In this module it will show How many site visits are completed by you, scheduled for you, pending and canceled visits.

Post Property free Module-

* In this module we can post our own property.
* We just have to select our profile type, and for what we are posting our property for(like sale, resale or rent.)
* We can also select our property type like apartment,villa,farm house etc.
* We have to provide some information like which city our property is located.
* As we are selecting all the columns and filling our details our property completeness column percentage is also increasing.
* When you select your requirements it will suggest you an option or plan(Basic), you have to select this and then it will ask for photos ,locality and some details of your property and then post

on the website.

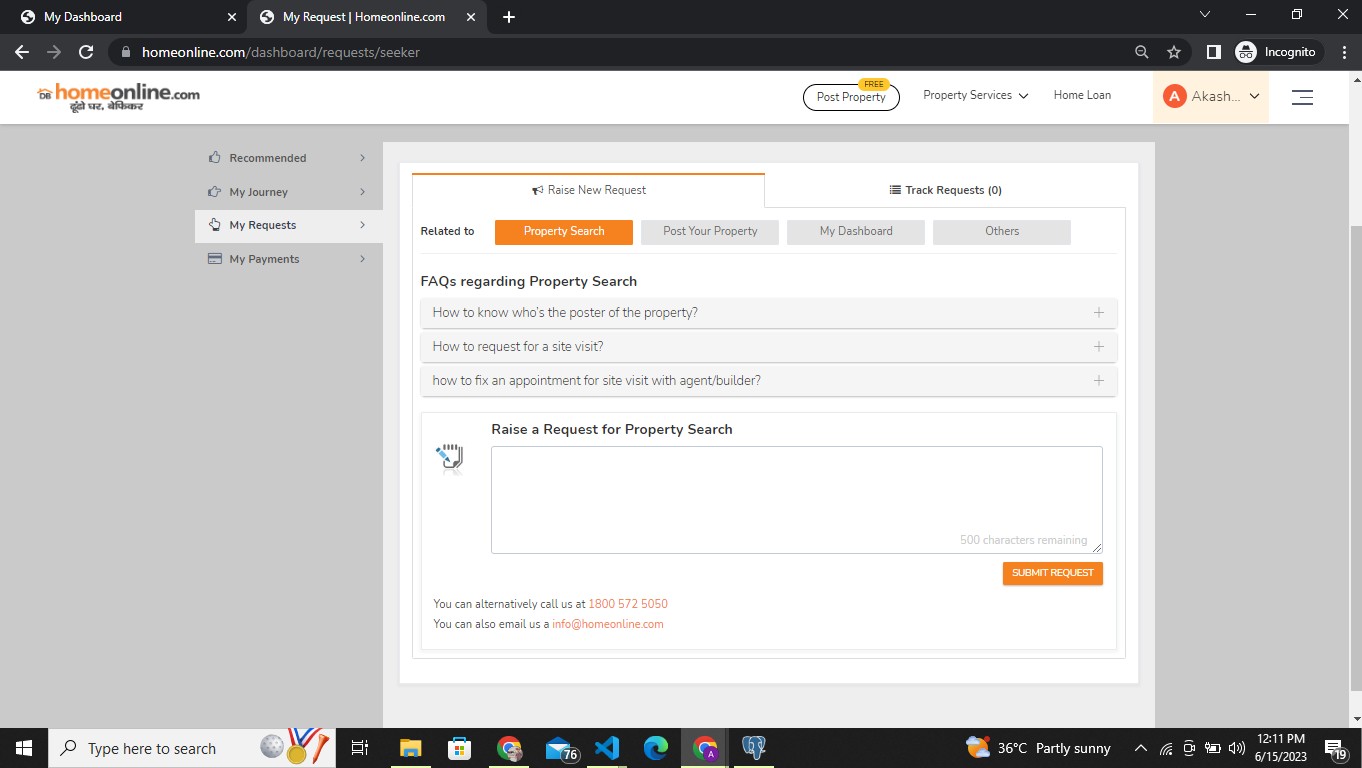
1. As Seeker Module- This module gives information about all the activities you performed from your id like property searches, viewed properties, shortlisted properties, contact properties etc.

It will also suggest some properties after you submit your requirements.

* + Property Searches-In this it shows all the property that we search on the app.
  + Viewed Properties- It shows all the properties that we view on our account.
  + Shortlisted Properties- It shows all the properties that we shortlisted for later on our account.
  + Contacted Properties- It shows all the properties that we contacted with their owner on our account..
  + Site Visits Properties- It shows all the properties that we visited.
  + Recommended Properties- It will help you to get properties as per your requirements.

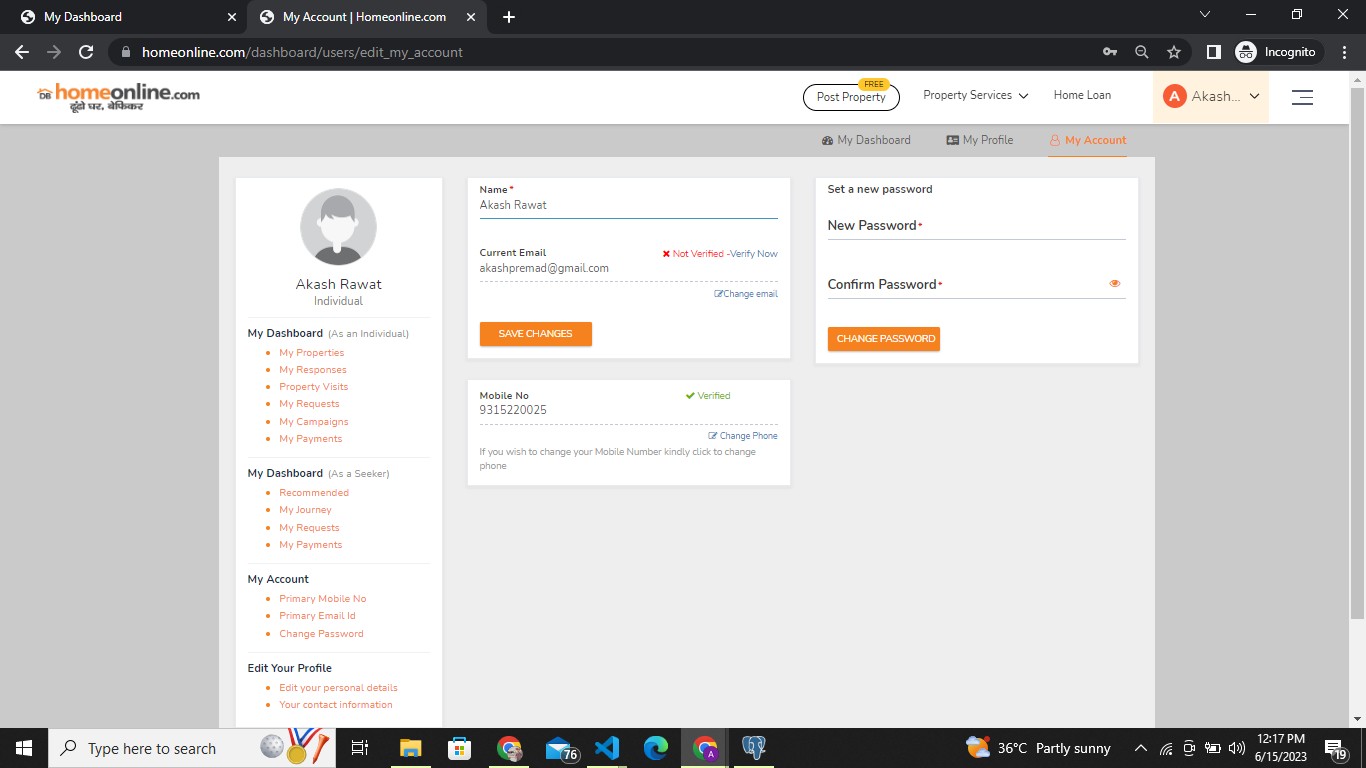
1. My Requests Module- This module gives information about any requests you raise regarding any issue.

Track request option will help you for tracking the request.

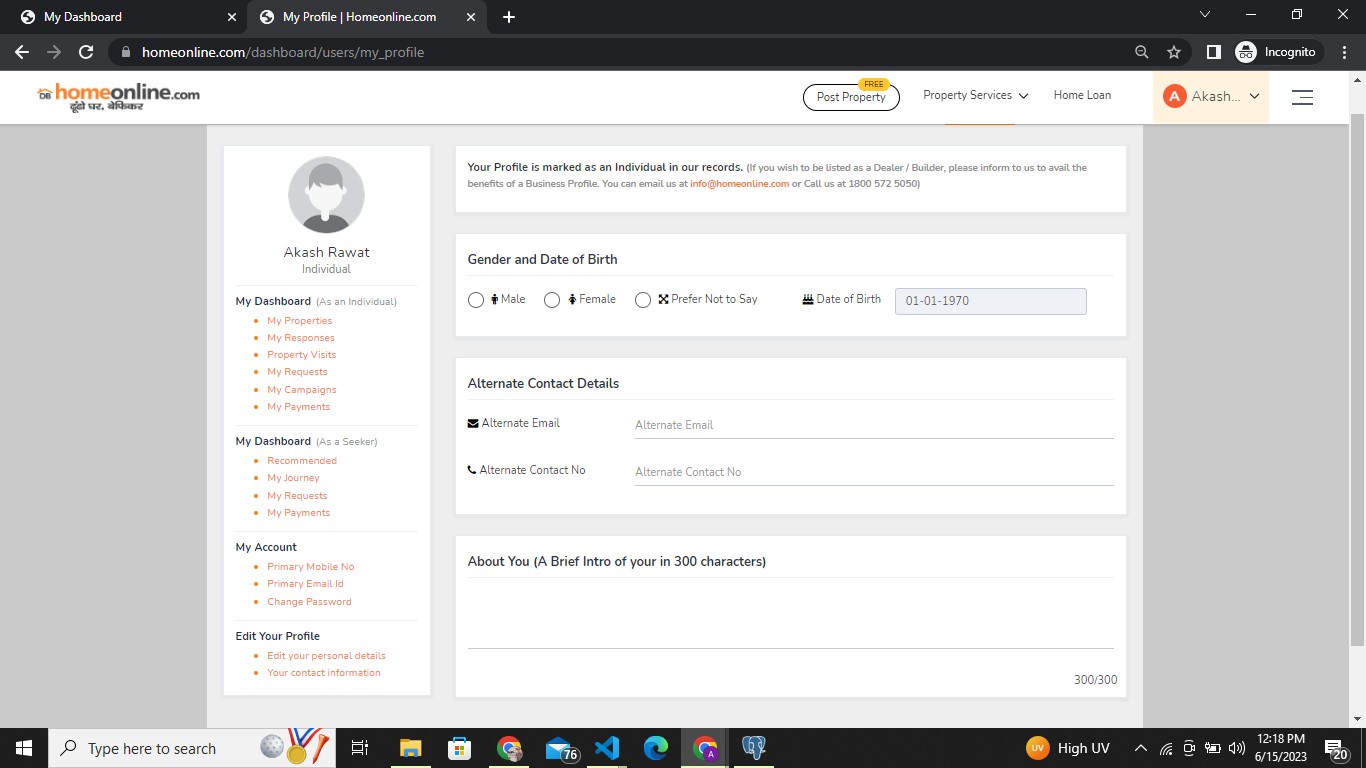


* It has 2 columns: Raise new request and Track Request.
* In Raise new Request it will ask some frequently asked questions and a option in which you can raise your query and submit your request
* If you face any issue you can call us or email us below given number or id.

1. My Payments Module-This module gives information about any payments you made to buy,sell or rent the property.
2. My Journey Module-This module gives information about your activities in your account like Your searches,viewed properties, shortlisted,contacted,site visited.
3. My Account Module-This module gives you functionality to change your Mobile number, email id and change password.



1. Edit Profile Module-This module gives you functionality to change your Date Of Birth, your gender,and your contact details. You can also write about you something so everyone will know about you.



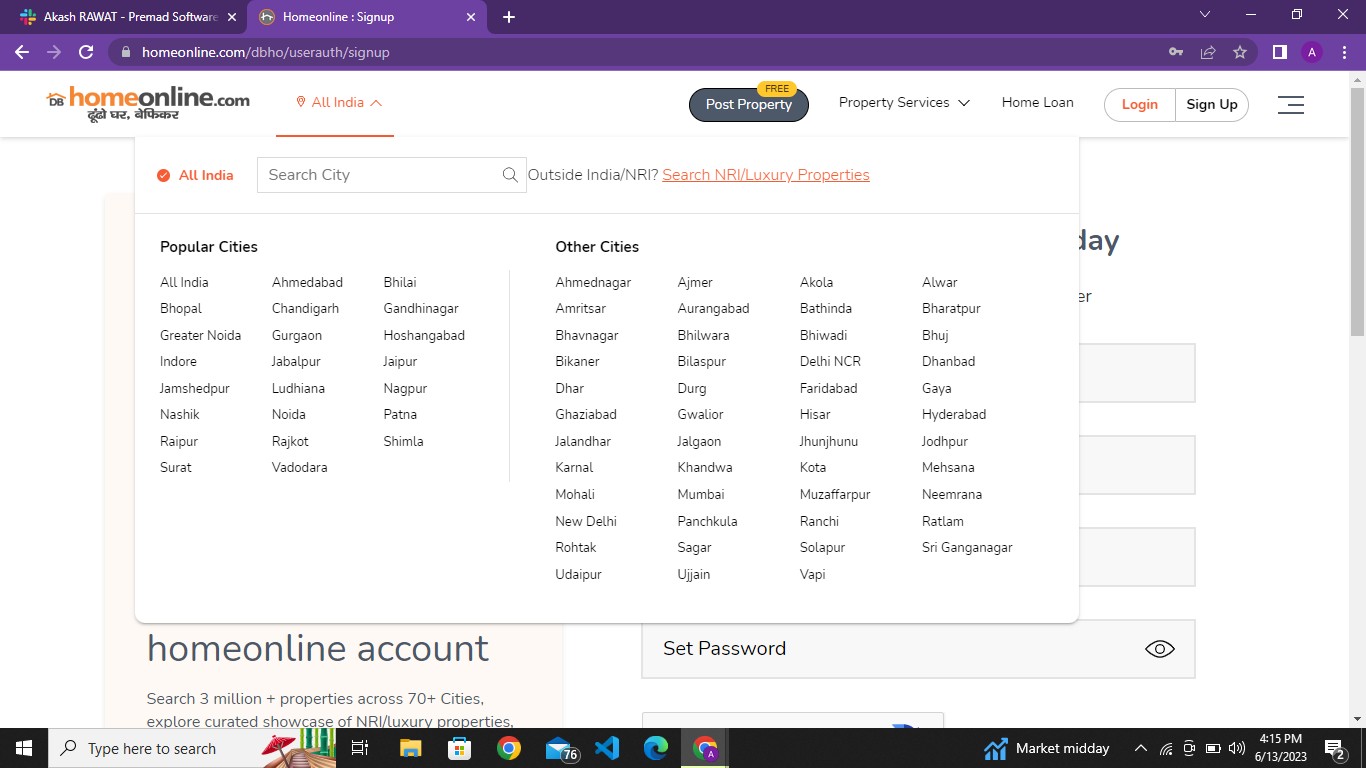
# Search City Module

### Functionality-

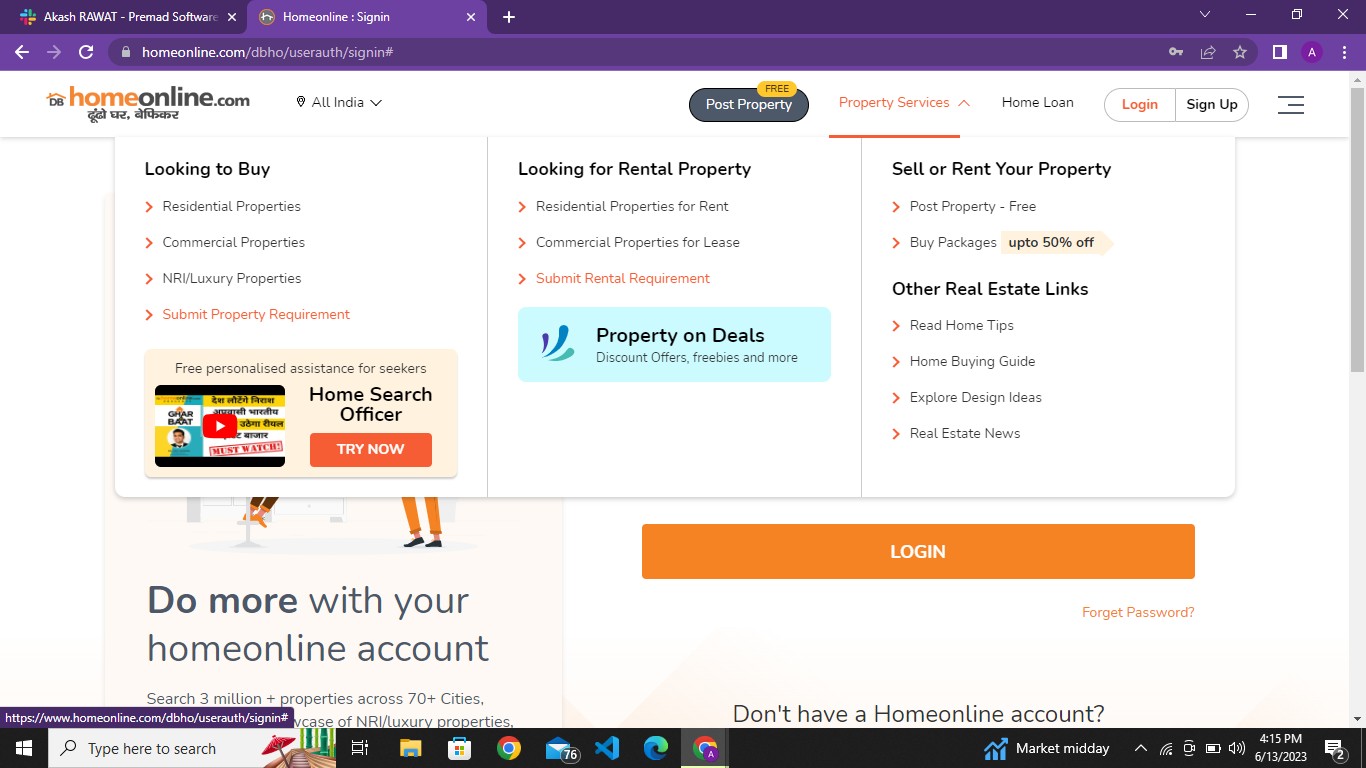
* It provides functionality to users looking for property for a particular location.
* It is already showing popular cities and some other cities.
* You just have to select your city and it will show all properties for that location.

### Validation-

* You can select only one city at a time.



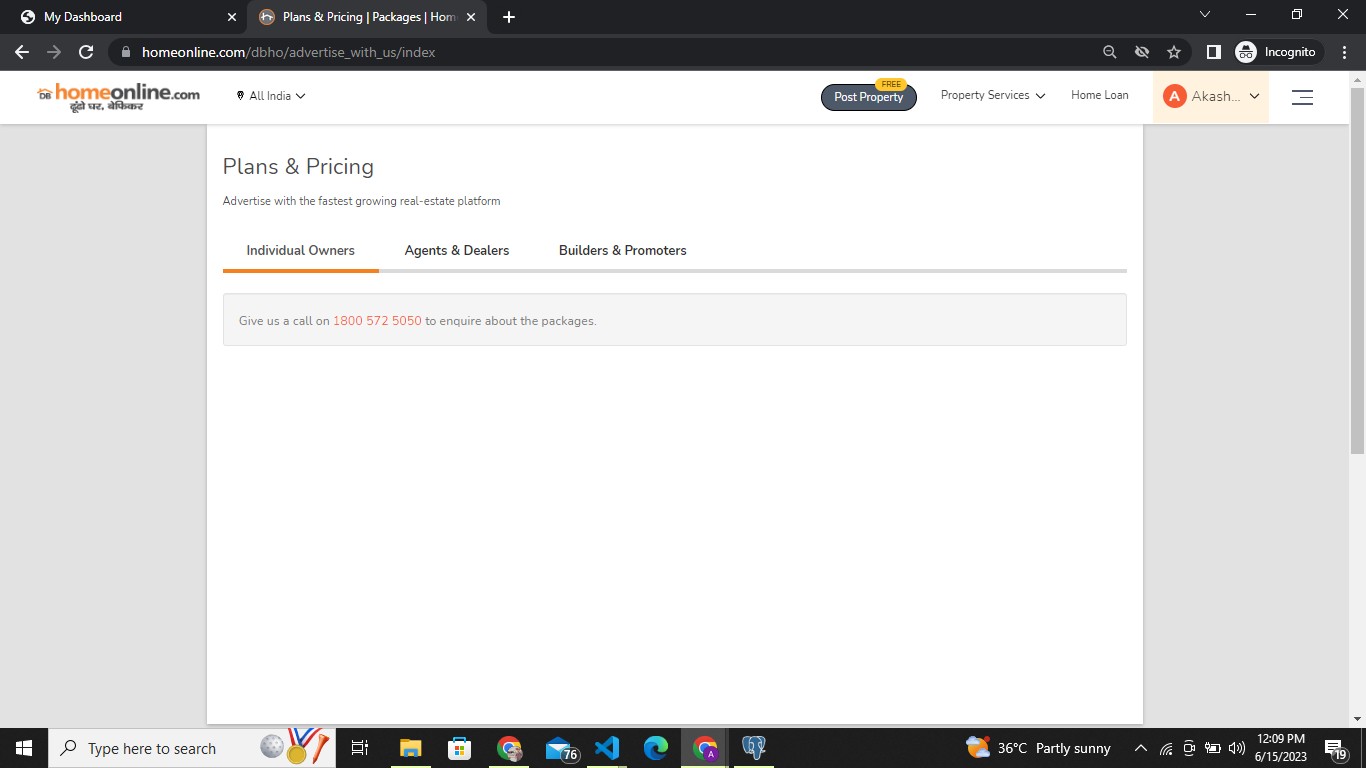
# Property Services Module



### Functionality-

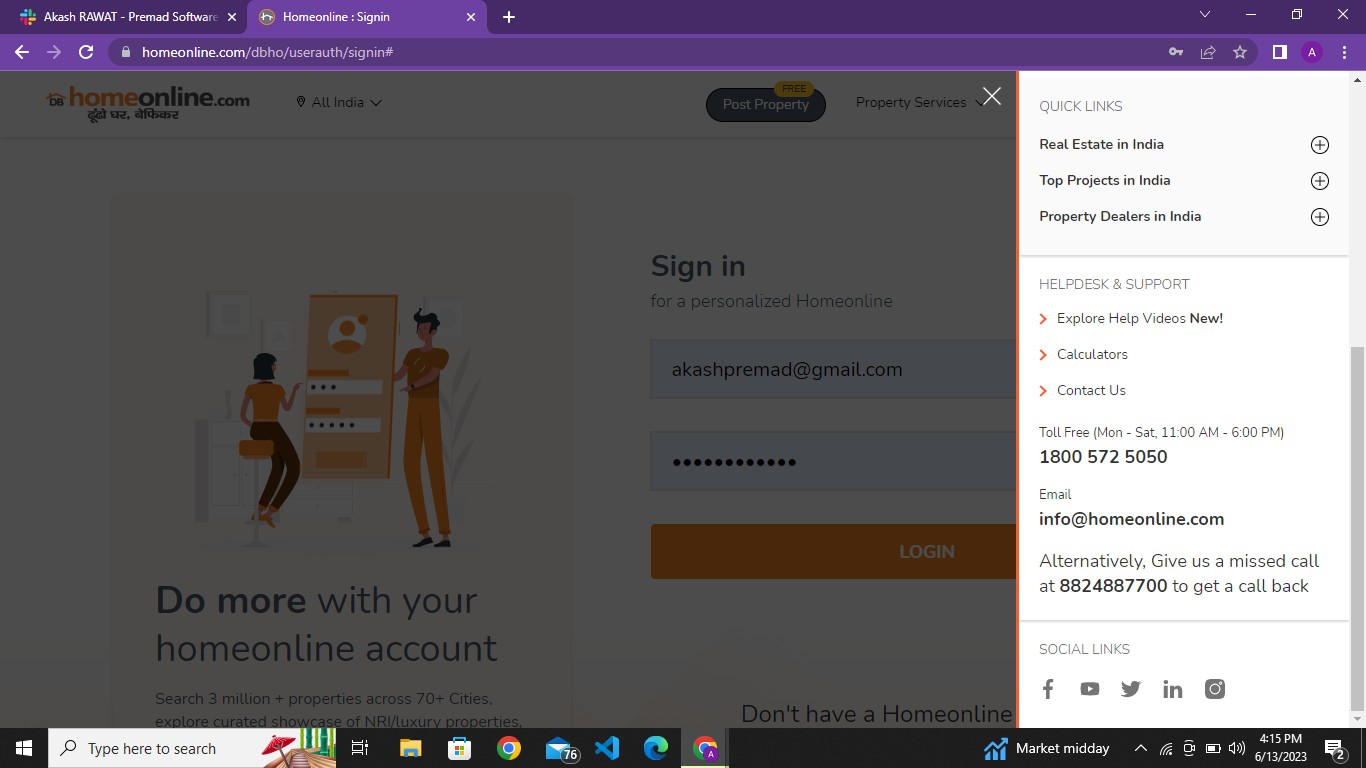
We have multiple modules in Property Services Module.

1. Apartment Module-
   * In this module we can find all Apartments for specific area.
   * It shows all Apartments, projects and Agents for apartments.
   * We have another module in this “sort by” which helps us to view the apartments.
2. Villas/Individual house Module-
   * In this module we can find all Villas and Individual houses for specific area.
   * It shows all Apartments, projects and Agents for Villas.
   * We have another module in this “sort by” which helps us to view the Villas
3. Residential plot-
   * In this module we can find all Residential Plots for specific area.
   * It shows all Residential Plots, projects and Agents for Villas.
   * We have another module in this “sort by” which helps us to view the Residential Plots.
4. Submit Property Requirement-
   * In this module we can select our specific Property Type.
   * We can select our specific location,Type..
   * We can select our bedroom preference according to our requirements.
5. Submit Rental Requirement-
6. In this module we can select our specific Property Type for rent.
7. We can select our specific location,Type..
8. We can select our bedroom preference according to our requirements.
9. Post Property free Module-
10. In this module we can post our own post our own property.
11. We just have to select our profile type, and for what we posting our property for(like sale, resale or rent.)
12. We can also select our property type like apartment,villa,farm house etc.
13. We have to provide some information like which city our property located.
14. When you select your requirements it will suggest you an option or plan(Basic), you have to select this and then it will ask for add photos,locality and some details of your property and then post on the website.
15. Buy Packages Module-
16. In this module we can advertise our property.
17. We can select plans or packages through individual owners,Agents and Dealers, and Builders and Promoters.
18. We can simply make a call on given number to learns more about these Packages.



1. Home Buying Guide Module-
2. In this module we can also take some help from related articles.
3. We can simply select our problem related topics like city and locality,Home loan, legal guidance, Vastu tips etc.
4. We can simply find related articles which will help us to buy a home.
5. Explore Design Module-
   * This Module provides us designs to decorate our rooms.
   * We can select our favorite design by our room type.
   * We can also select designs from Room Categories like guest Bedroom,Master Bedroom ,Dinner room etc.
6. Real Estate News Module-
   * In This Module we get Real Estate regarding all News and information.

# Slide Bar Module



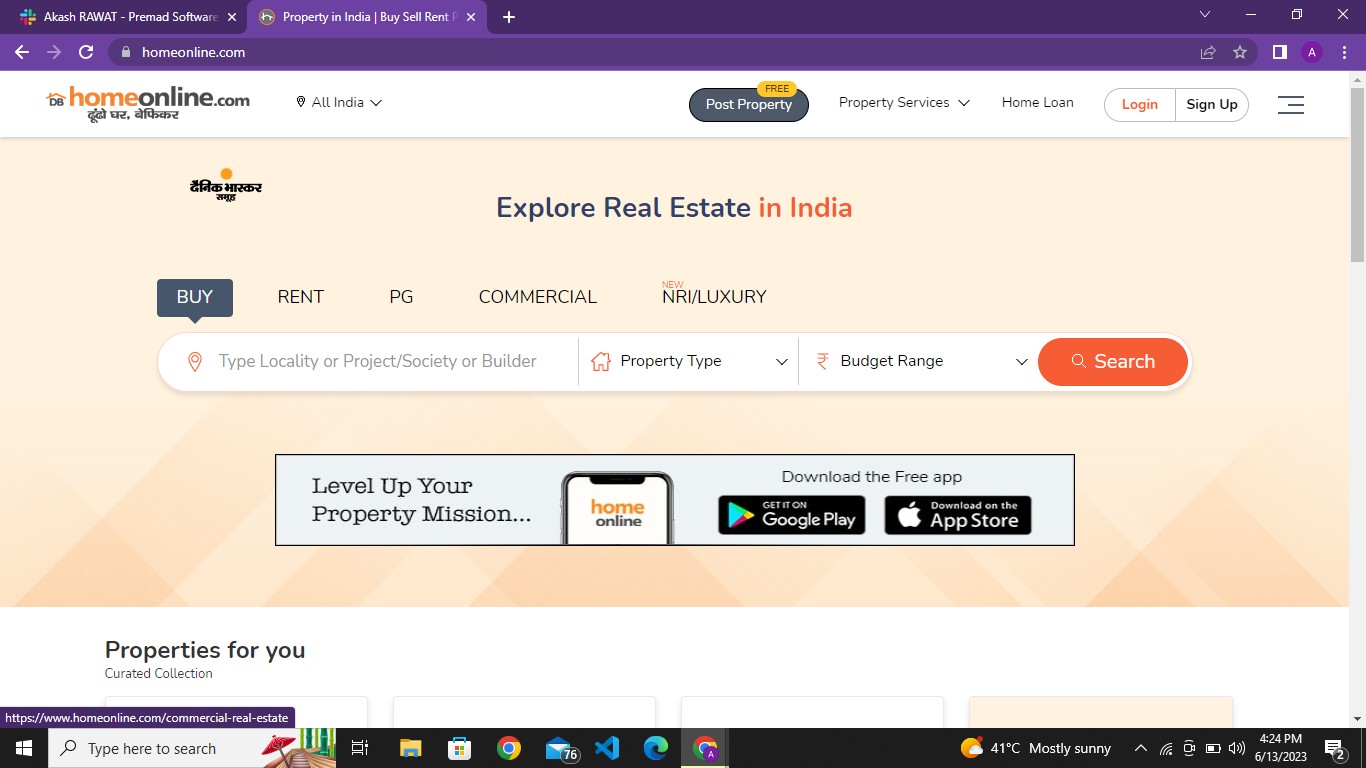
### Functionality-

It contains Multiple Modules

1. Quick Links Module -
   * Real Estate in india- This Module Provides Top Properties for the every particular city of India.
   * Top Projects In India- This Module gives a whole list of Top Projects in India for every City.
   * Property Dealers In India- This Module Shows Top Property Dealers of India from cities.
2. Helpdesk & Support Module -
   * Explore Help Videos- This Module Provides help videos for user so he can easily understand about website.
   * Calculators- This Module Opens up a calculator for customer so he can find price according to city,locality,property type,area, etc. We can also calculate EMI for Property.
   * Contact us- This Module provides a calling number, email Id and Mailing Address of the company.
3. Social Links Module -
   * This Module Provides Links of Social Media of the company so the user can contact the company via social media sites like Facebook,twitter,Youtube etc..

# Search bar Module

##### Let’s user search for city, area, property type and other things.



Buy module-

* Functionality -

This module lets the user search for particular locality, Property type and Budget range for buying them.

* Property type -

This module tells different types of properties which user can purchase

* Type Locality or Project/Society or Builder

This module asks user the type of locality, society or builder do they want

* Budget range -

By using this module, the user can set the range under which he wants to purchase property.

* Rent Module-

##### Functionality -

This module lets the user search for particular locality, Property type and Budget range for renting them.

##### Property type -

This module tells different types of properties which user take on rent.

##### Type Locality or Project/Society or Builder

This module asks user the type of locality, society or builder do they want

##### Budget range -

By using this module, the user can set the range under which he wants to take on rent property.

* PG Module-

##### Functionality -

This module lets the user search for pg in particular locality, Property type and Budget range to get an affordable pg.

##### Type Locality or Project/Society or Builder

This module asks user the type of locality, society or builder do they want

##### Property type -

This module tells different types of properties which user can get as PG

##### Budget range -

By using this module, the user can set the range under which he wants to PG.

* Commercial Module-

##### Functionality -

This module lets the user search for commercial property in particular locality, Property type and Budget range to get an affordable price.

##### Buy

To purchase commercial properties after searching and getting an affordable price.

##### Type Locality or Project/Society or Builder

This module asks user the type of locality, society or builder do they want

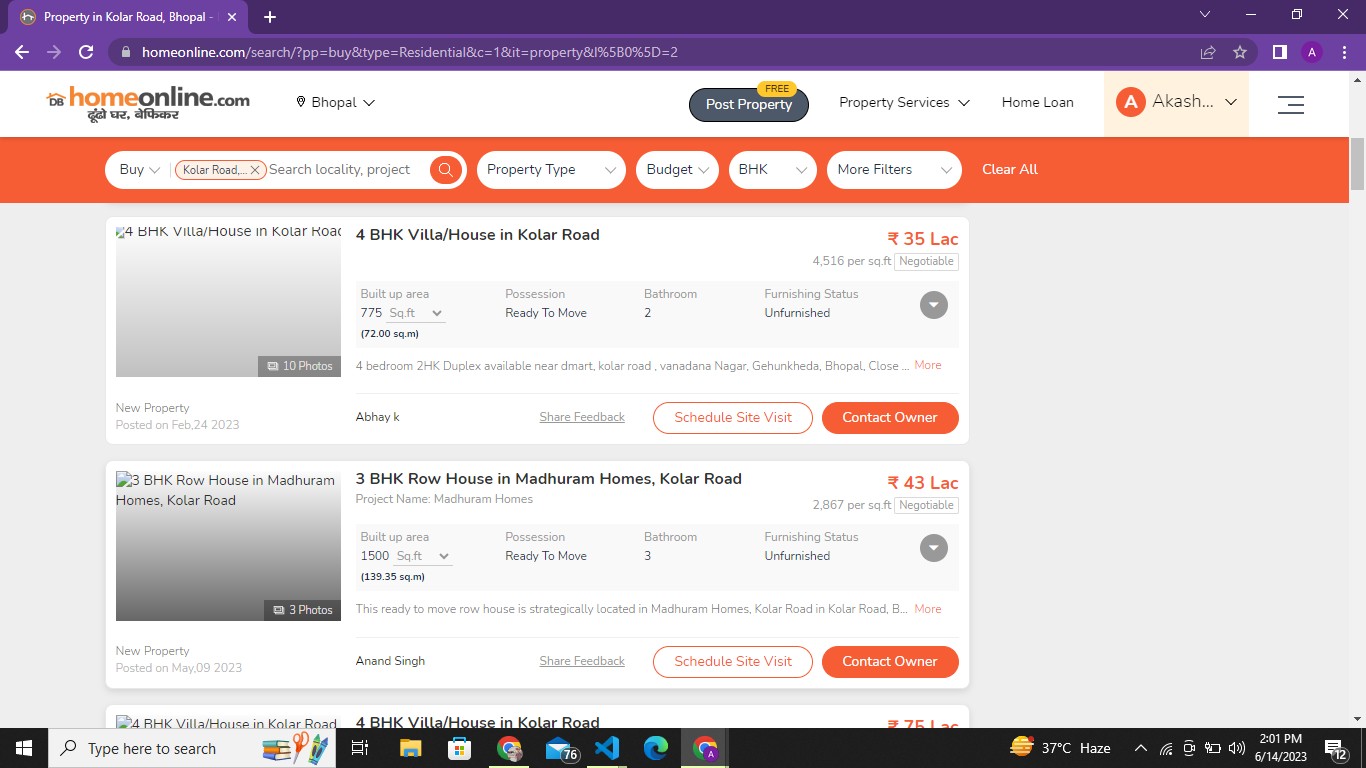
##### Property type -

This module tells different types of properties which user can get as PG

##### Budget range -

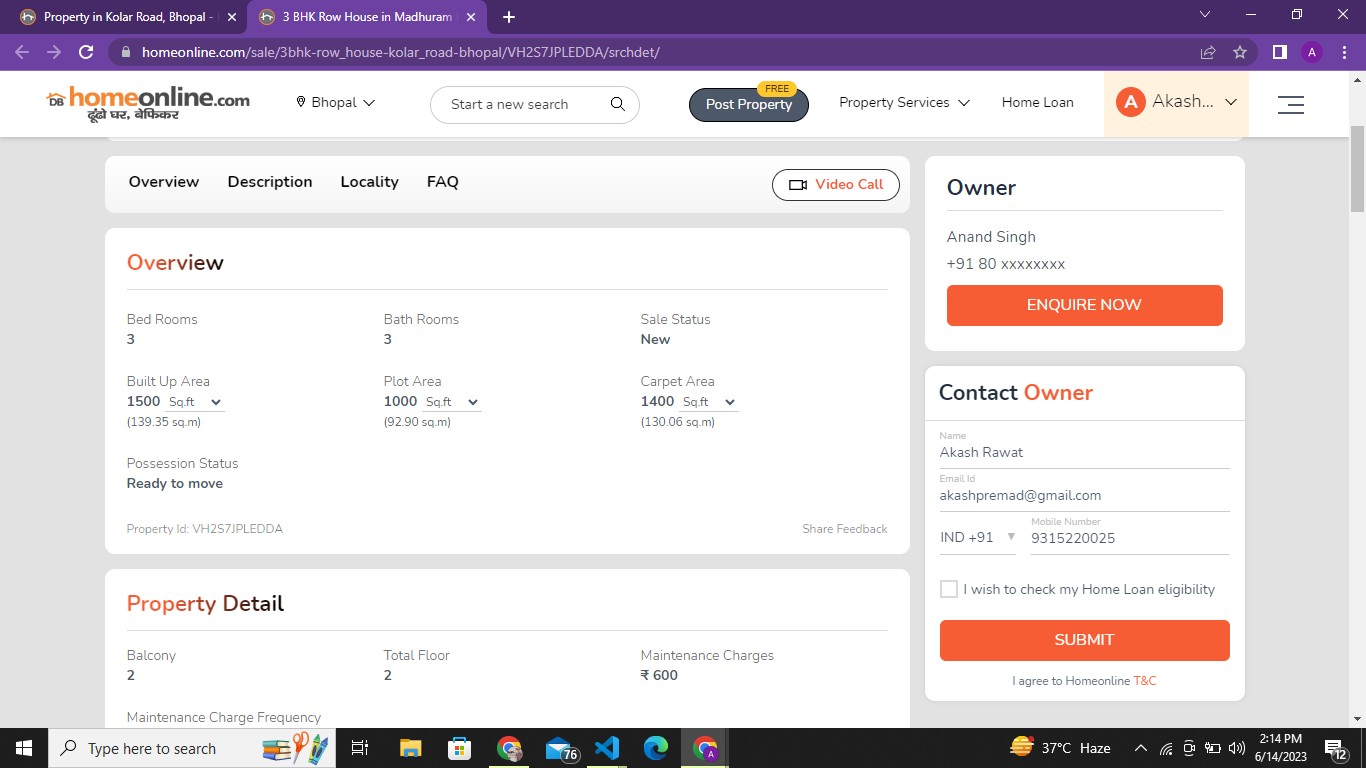
By using this module, the user can set the range under which he wants to PG. Whenever you search property in the search bar it shows properties in the

form of Cards.



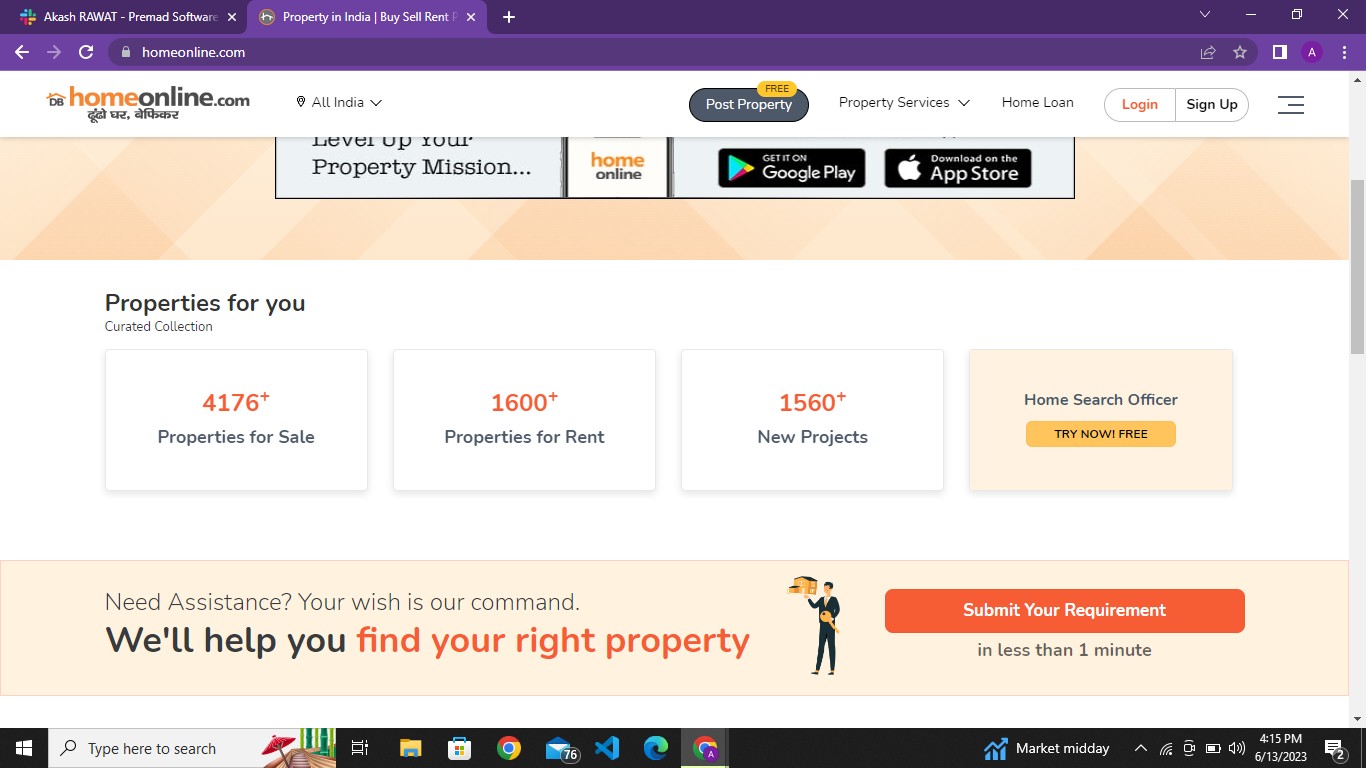
* Photos Module- It shows photos of that particular Property.
* Date Module- It shows the Date of that particular Property when it is posted on a website..
* Description Module- It shows the built up area, bathrooms, floor, furnishing status of the property.
* Price Module- It shows the price of that particular property.
* Schedule site visit Module- It gives you the option to book a date to visit that property site.
* Contact Owner Module- It gives you the Information of the owner who owns or posted the property.

Property Module-



* Property Module- It provides the information of the property detail,description,
* FAQ Module- It provides some property related questions and answers which are asked recently.
* EMI Calculator Module- It gives a functionality to calculate the loan for the property.
* Nearby Properties Module- It gives some suggested property to the user.
* Similar Searches Module- It gives some suggested related property to the selected one.
* Quick Links Module- It provides some shortcut links to the user.

# Properties for you Module



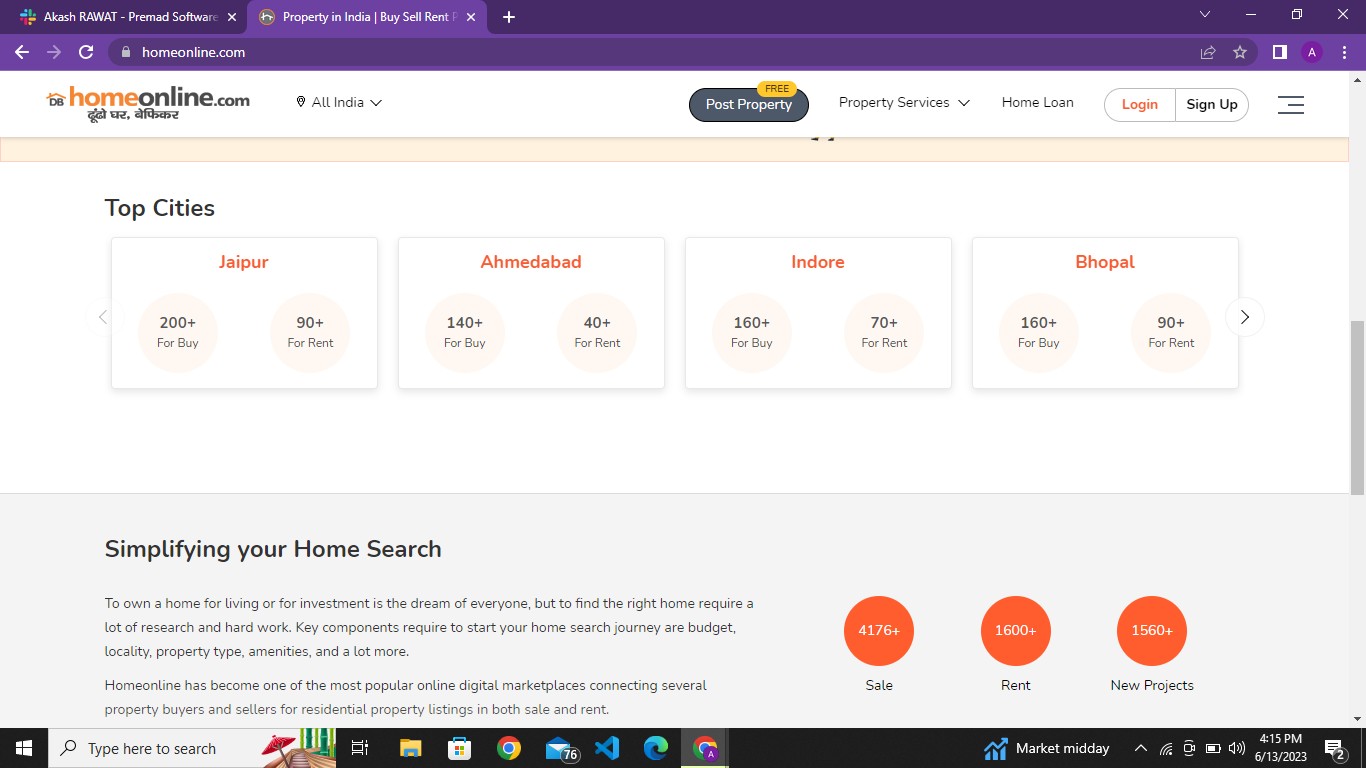
### Functionality-

* This Module provides information in the form of cards which tells us about properties for rent and buy ,New projects in particular locations.
* It is the Statiscal Overview for Users so he can understand through numbers easily.

# Top Cities Module

### Functionality-

* This Module shows Top cities in the form of Cards.
* This module shows stats of properties for buy and rent for particular cities.



# About Us Module

### Functionality-

* This Module Provide Information about Company HomeOnline.
* User can learn more about company, how it works and benefits etc.

# Google Play store Module

### Functionality-

This module lets users download applications for this website from google store.

Functionality-

# App store Module

This module lets users download applications for this website from the App Store.

# Sitemap Module

### Functionality-

This module gives information about all sites which are available under this company from which users can purchase, rent or get PG.

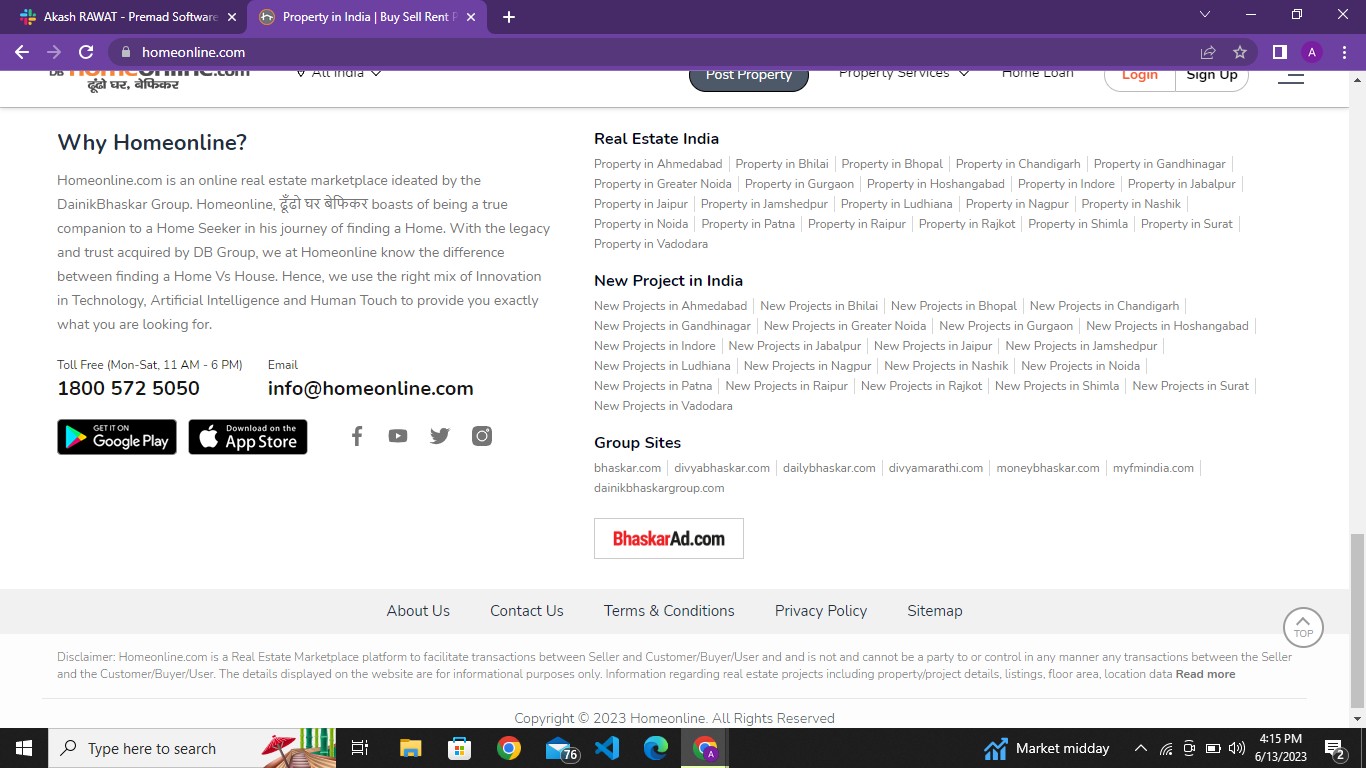
# Terms and Condition Module

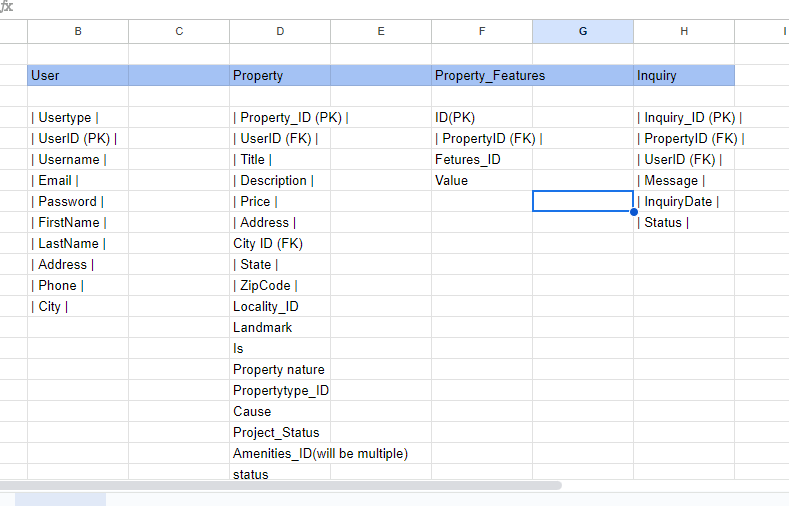
### Functionality-

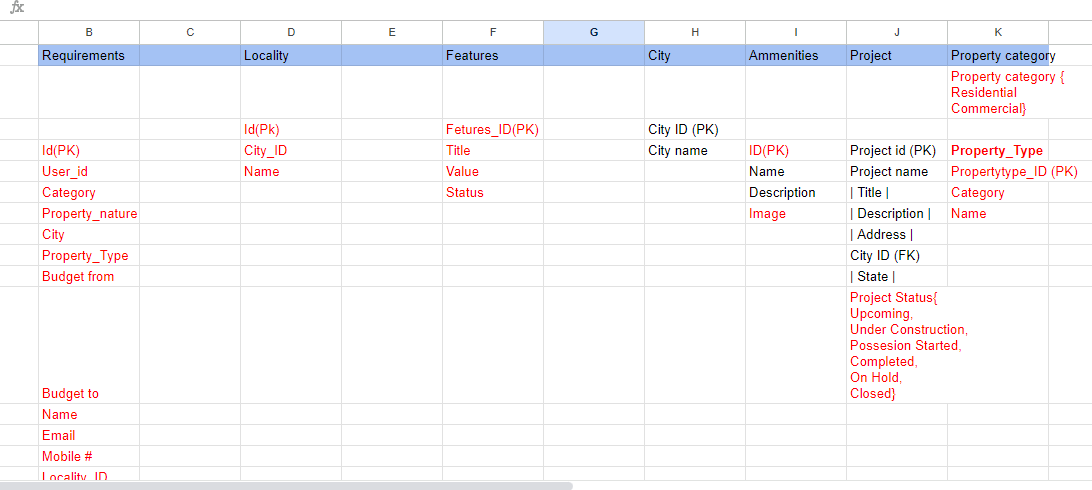
* This Module provides all rules and information about the company's Terms and conditions.
* It defines Terms,Services, Eligibility, fees and about Payment and refund clauses.

# Privacy Policy Module

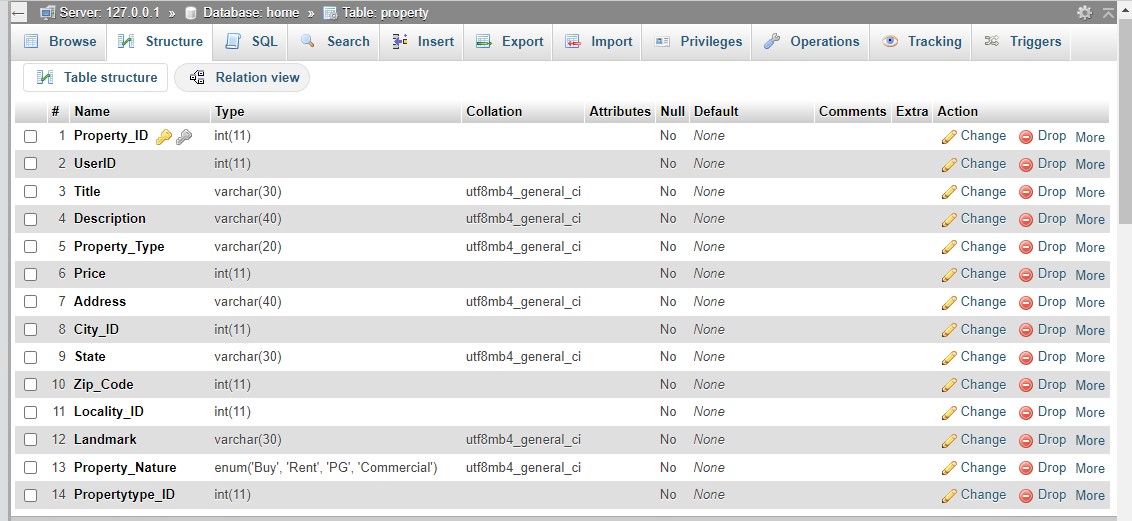
### Functionality-

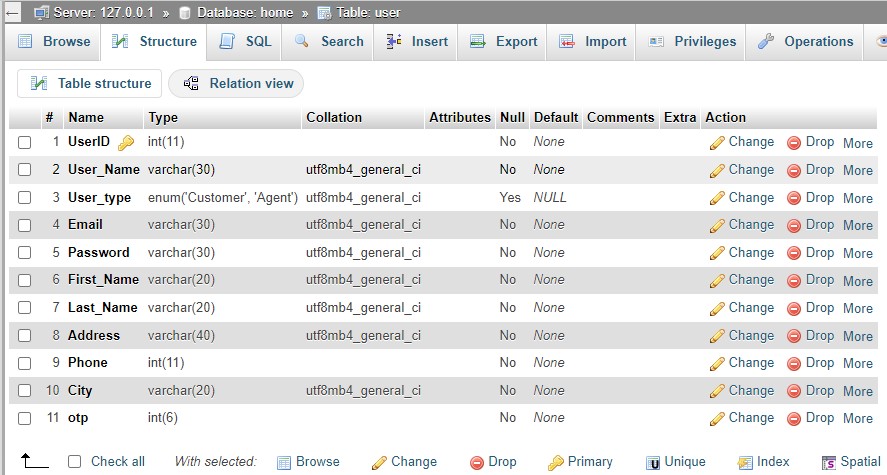
* This Module contains information about User and company private safety.
* It contains information about cookies, links,information use by company,sharing information,accessing , information security etc.
* **Schema for 5bhk Project:**





* **Database of 5bhk:**





* **API’s for Project :**

API is the acronym for application programming interface — a software intermediary that allows two applications to talk to each other. APIs are an

accessible way to extract and share data within and across organizations.

**Postman: Postman is an API(application programming interface) development tool which helps to build, test and modify APIs.**

1. **City Search :This API helps to get details of all cities from our database.**

**Code:**

**const express = require('express'); const mysql = require('mysql');**

**const app = express();**

**// Database configuration**

**const db = mysql.createConnection({ host: 'localhost',**

**user: 'root', port:3306, database: 'home'**

**});**

**db.connect((err) => { if (err) {**

**console.error('Error connecting to database:', err);**

**} else {console.log('Connected to database');}}); app.get('/fetch', (req, res) => {**

**db.query('SELECT \* FROM city', function(err,result,fields){**

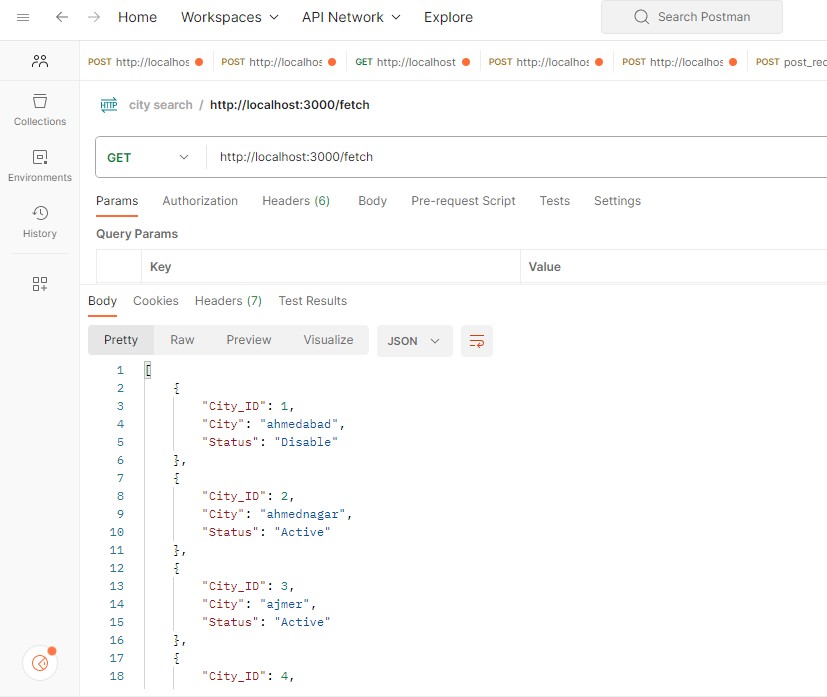
**if (err){ console.log(err)**

**}else{**

**res.send(result)}})})**

**// Start the server const port = 3000; app.listen(port, () => {**

**console.log(`Server is listening on port ${port}`)});**



1. **Login :** Verifies the User and gives a message if the user login successfully.

app.post('/login', (req, res) => {

const { username, password } = req.body;

const query = 'SELECT \* FROM user WHERE User\_name = ? AND Password

= ?';

db.query(query, [username, password], (err, results) => { if (err) {

console.error('Error executing query:', err); res.sendStatus(500);

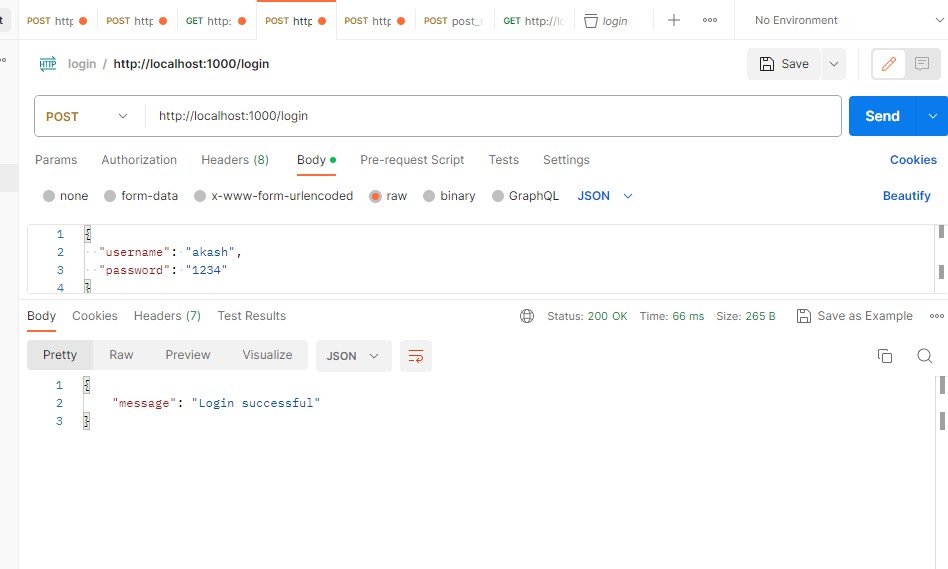
} else if (results.length > 0) {

res.json({ message: 'Login successful' }); console.log('username is: ',username)

} else {

res.status(401).json({ message: 'Invalid username or password'

})}})});



1. **Registration Page:** Helps to register user by taking details of user and post it to database.

app.post('/api/register', (req, res) => {

// Perform validation checks on the input data

if (!req.body.username || !req.body.email || !req.body.password) { return res.status(400).json({ success: false, message: 'Incomplete

registration data' });

}else{

const { username, First\_name,Last\_name,Phone,email, password } = req.body;

//generate a unique user ID

const userId = generateUserId();

// Return the registration success response res.status(200).send({

success: true,

message: 'Registration successful', data: {

user\_id: userId, First\_name:req.body.First\_name, Last\_name:req.body.Last\_name, Phone:req.body.Phone,

username: req.body.username, email: req.body.email} });

Function insertUser(user\_id,First\_name,Last\_name,Phone,username,email,password) { const query = 'INSERT INTO user (UserId,First\_Name,Last\_Name,Phone,User\_name,Email,Password) VALUES (?,?,?,?, ?, ?, ?)';

const values = [user\_id,First\_name,Last\_name,Phone,username,email,password];

db.query(query, values, (err, results) => { if (err) {

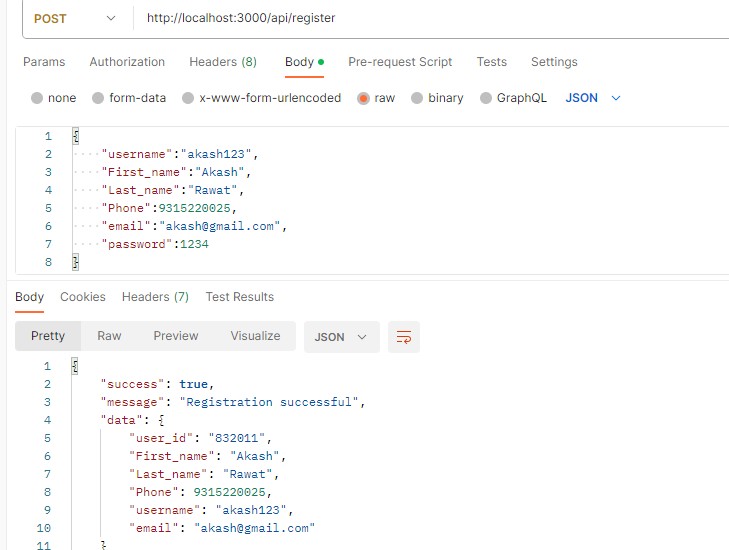
console.error('Error inserting user:', err); return; }

console.log('User inserted successfully'); })};

insertUser(userId,req.body.First\_name,req.body.Last\_name,req.body.Phone,re q.body.username,req.body.email,req.body.password);

function generateUserId() {

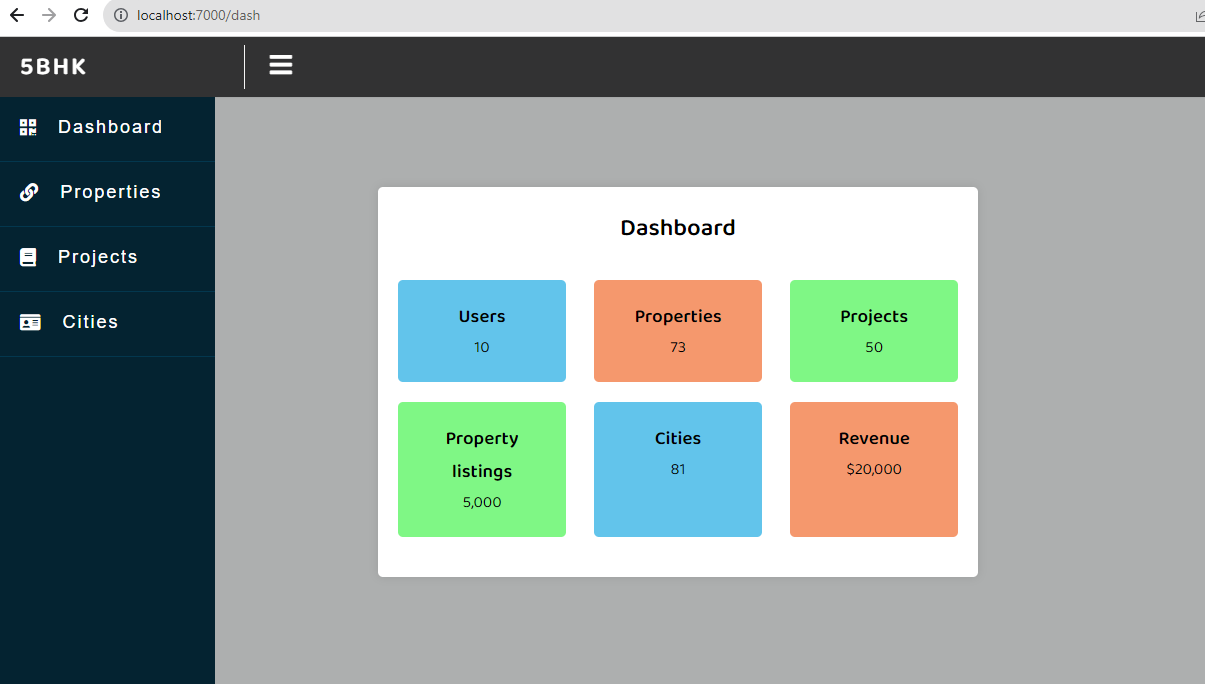
return Math.floor(Math.random() \* 1000000).toString()}}});



* **Admin Dashboard for Project:**

An admin dashboard allows at-a-glance access to the crucial information for the specific needs of a particular professional or a team. Dashboards provide users

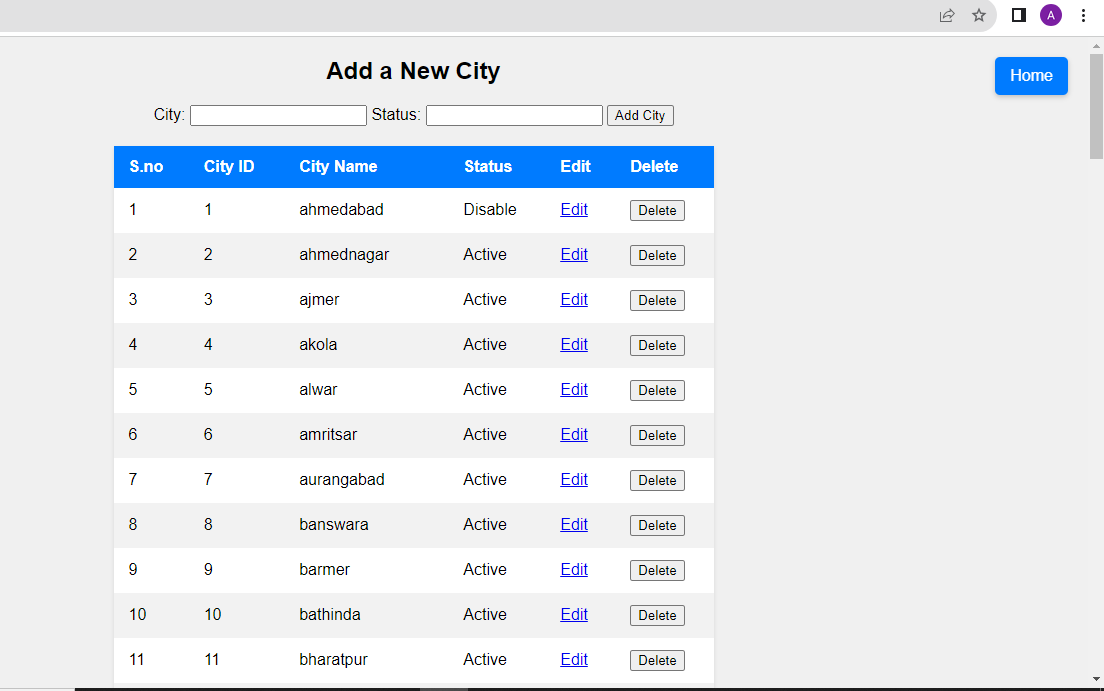
with a comprehensive overview of the key data, reports, KPIs, etc., and enable drilling down to more details if required.



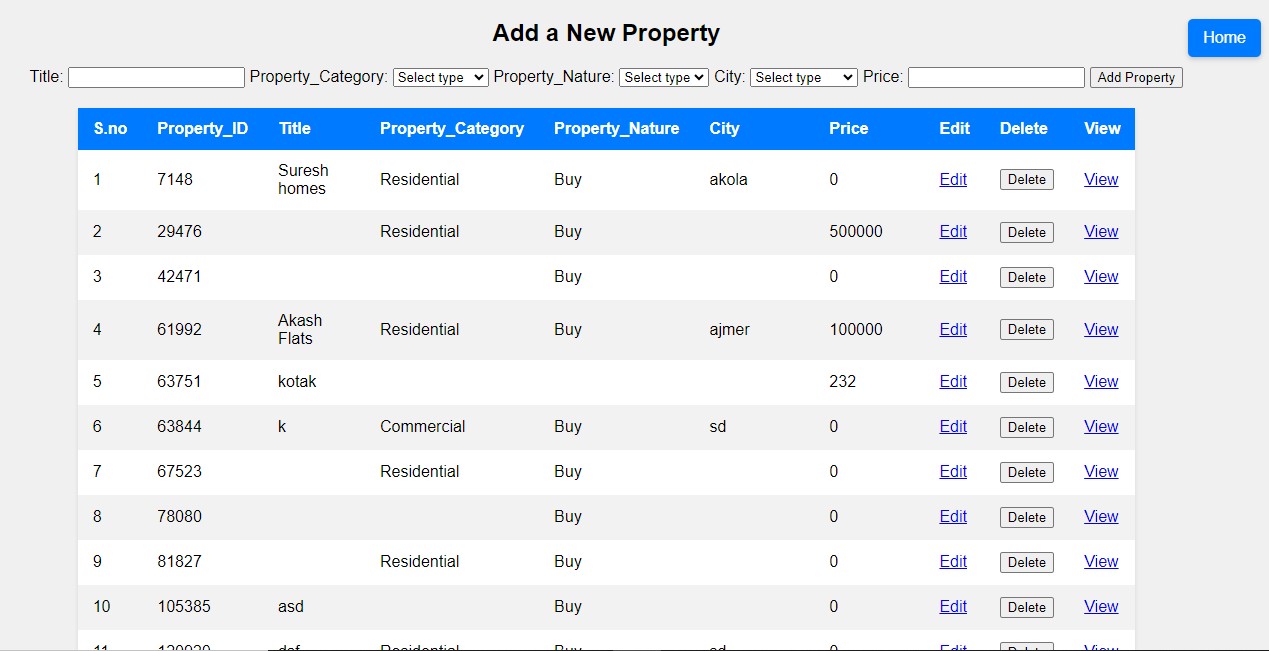
#### Here this Dashboard helps admin to show Data from the Database. It shows total users,projects,Properties,Cities,Revenue of project.

**In property menu from slidebar it gives three options like Add property,Property Listings and Search Property.**

1. **City:** In the city menu we have a table which shows the total listed Cities and its Status. It gives the option to add,delete and edit any particular city.

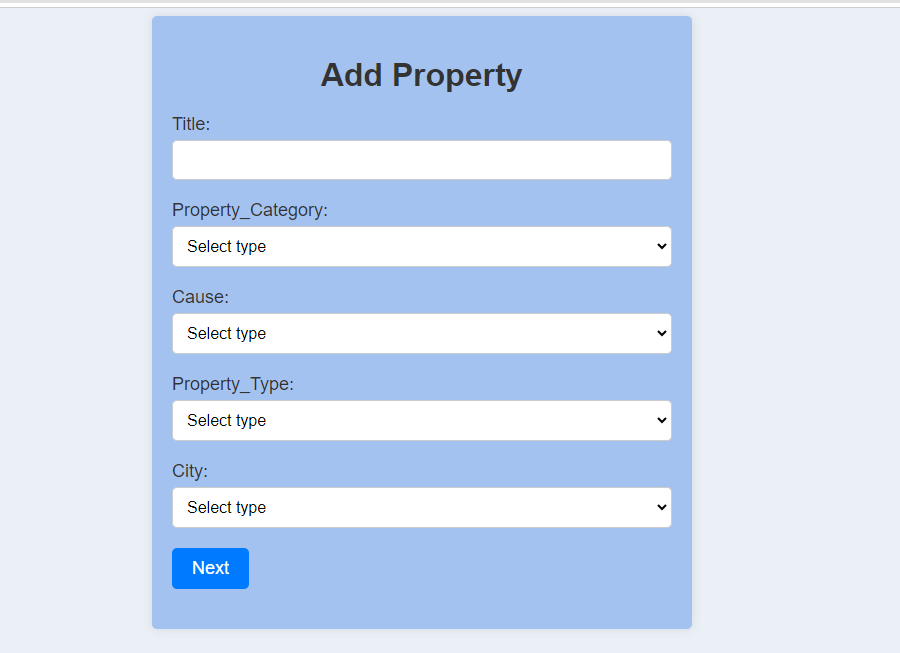


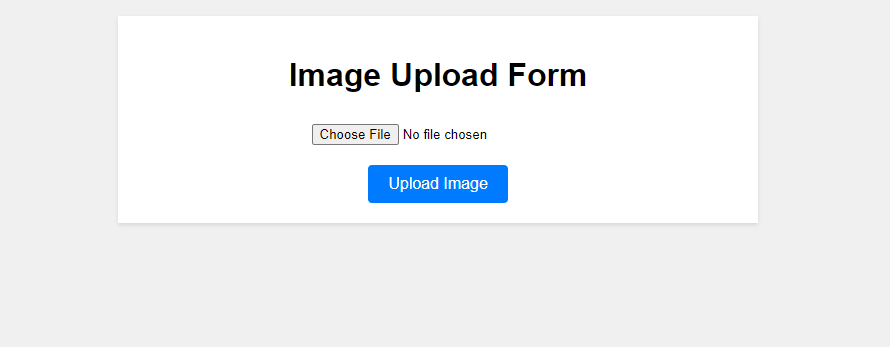
1. **Property List:** In the Property list menu it gives us functionality to add property instantly, and shows Property ID,Name,Category,Nature,Price etc. It Also has some features like delete property, Edit any property details,View any particular property details.

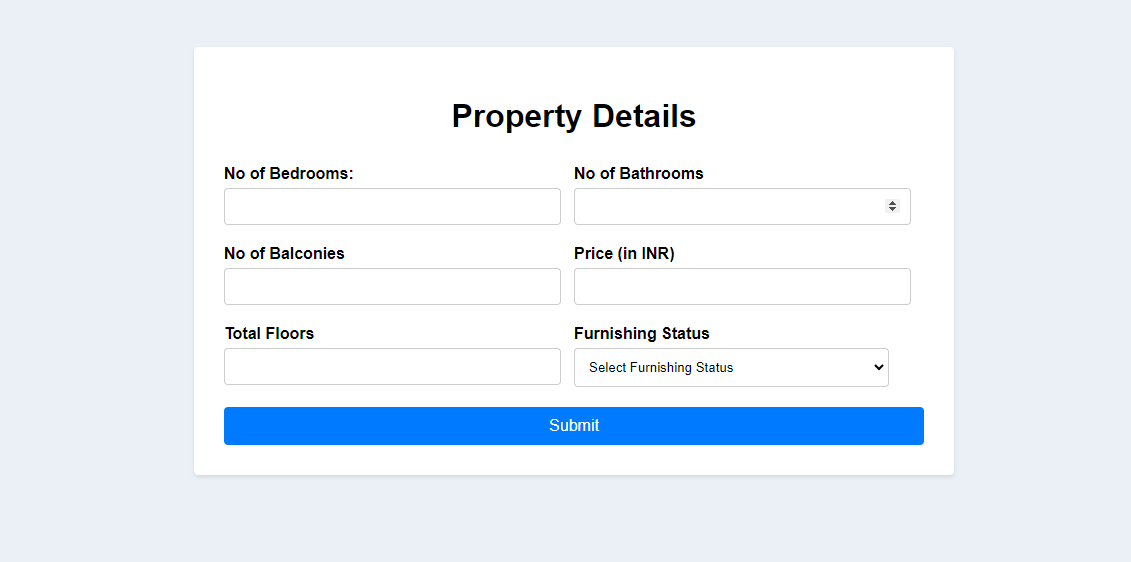


1. **Add Property:** In this menu we can add our property in Website with following details like Property\_Id,Title,Property\_Category,Cause,Property\_Type,C ity,Address,Landmark,Locality,Bedrooms,Bathrooms,Balconi es,Price,Total\_floors, Furnishing\_Status.

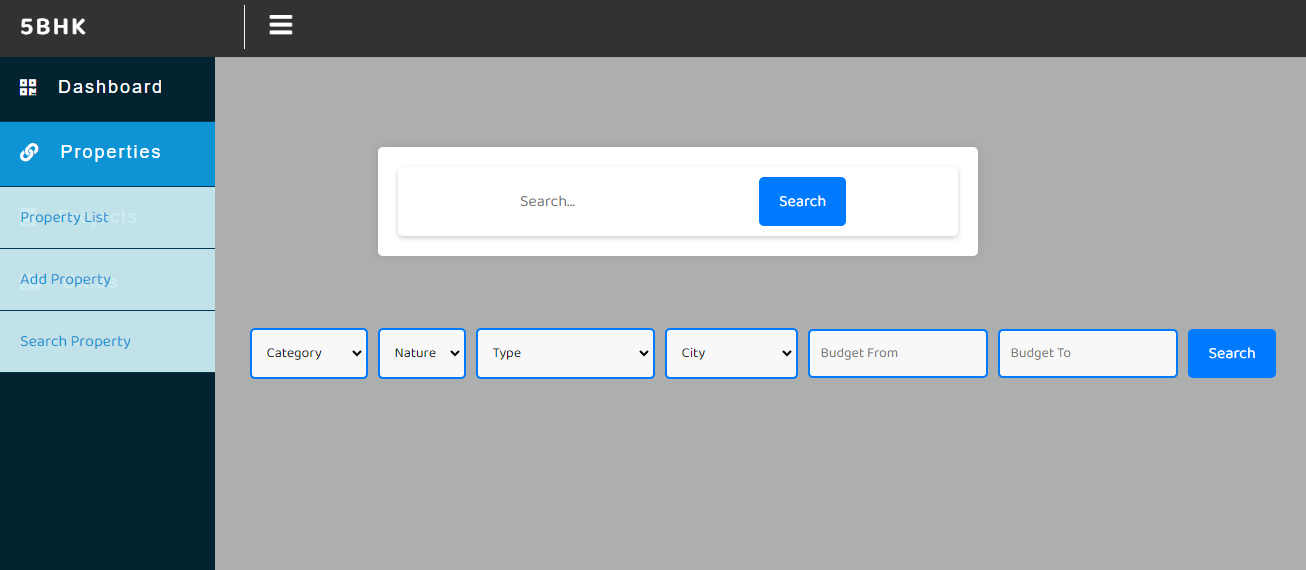
It gives us functionality to also add images of our property. It saves all data from the user through the website to our database and list on the website.







1. **Search Property :** In this menu we can search for property on the basis of city, property type,Category,and from the Budget.



## Lessons Learned:

The journey through the backend development of the property site website has been a dynamic learning experience, revealing invaluable insights into the realm of technology, real estate, and collaborative innovation. As the project unfolded, several key lessons were gleaned that have not only enhanced technical skills but also deepened the understanding of effective teamwork, problem-solving, and project management.

\*\*1. The Power of Collaboration: The development of a sophisticated property site website demands a harmonious collaboration between diverse teams, including backend developers, frontend designers, and stakeholders. Effective communication, mutual respect for expertise, and the willingness to learn from one another were pivotal in achieving seamless integration between various components.

\*\*2. User-Centric Design: Behind every line of code lies a real user seeking convenience, reliability, and meaningful interactions. Designing the backend with a user-centric approach requires empathy and an in-depth understanding of user behavior, which in turn drives decisions that enhance the overall user experience.

\*\*3. Scalability as a Priority: As the property site website gained traction, the importance of building a scalable backend architecture became evident. Anticipating and accommodating increased traffic while maintaining performance requires a proactive approach to design and coding, safeguarding against potential bottlenecks.

\*\*4. Data Security and Privacy: In an era of heightened data breaches, implementing stringent security measures is non-negotiable. Developing a robust user authentication system and encryption protocols became an invaluable lesson in safeguarding sensitive user data and instilling trust.

## Future Recommendations:

As the journey through the backend development of the property site website concludes, it paves the way for a future ripe with possibilities and improvements. The lessons learned and experiences gained offer valuable insights that can guide the evolution of the project and the enhancement of its impact. Here are some future recommendations that can shape the path ahead:

\*\*1. Enhanced Personalization: Consider implementing advanced personalization features based on user preferences and behavior. Utilize machine learning algorithms to suggest relevant properties, increasing user engagement and satisfaction.

\*\*2. Mobile-First Optimization: With the proliferation of mobile devices, optimizing the backend for seamless mobile experiences becomes paramount. Prioritize responsive design and performance on various screen sizes.

\*\*3. AI-Powered Insights: Harness the power of artificial intelligence to provide users with insightful data about property trends, investment opportunities, and market forecasts, offering them a deeper understanding of their choices.

\*\*4. Real-time Communication: Implement real-time communication channels within the platform, enabling users to interact with property owners, agents, and other users, fostering a sense of community.

\*\*5. Automated Property Management: Develop backend processes that enable property owners to manage listings, schedule viewings, and update property details with ease, streamlining their involvement in the platform.

## Conclusion:

This internship provided an invaluable opportunity to apply theoretical knowledge to real-world scenarios. The experience gained in backend development, particularly using Node.js and MySQL, was insightful. The challenges faced during the internship strengthened problem-solving skills, and the achievements highlight the progress made.