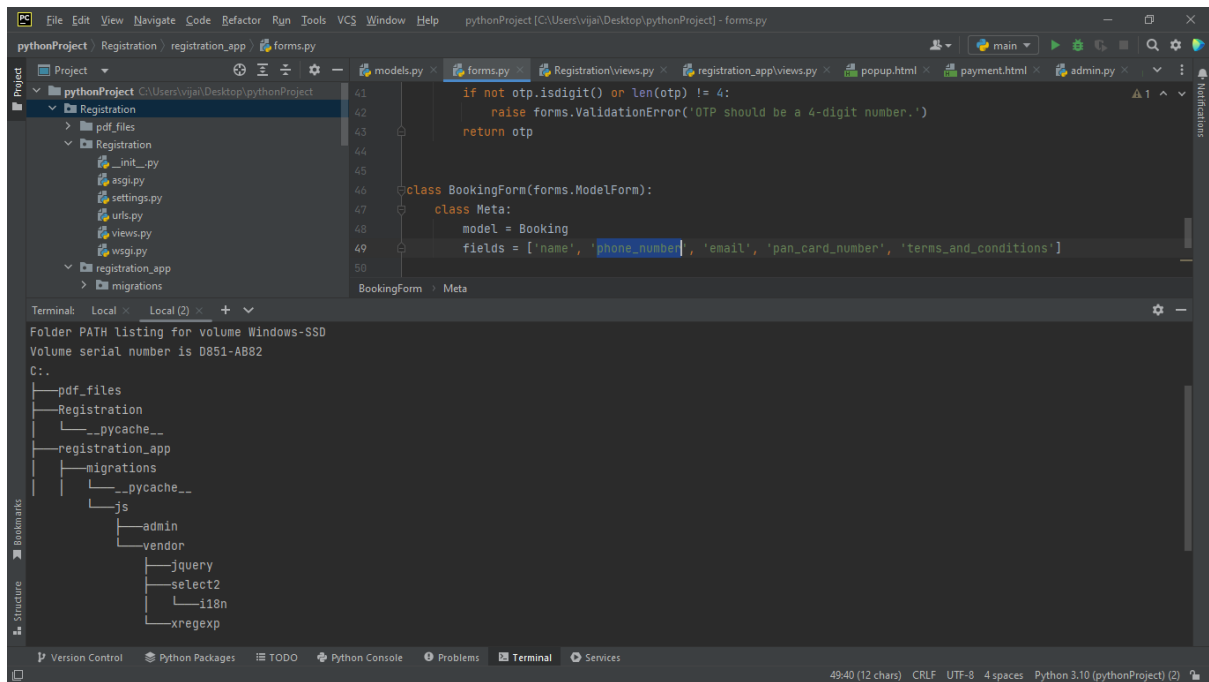
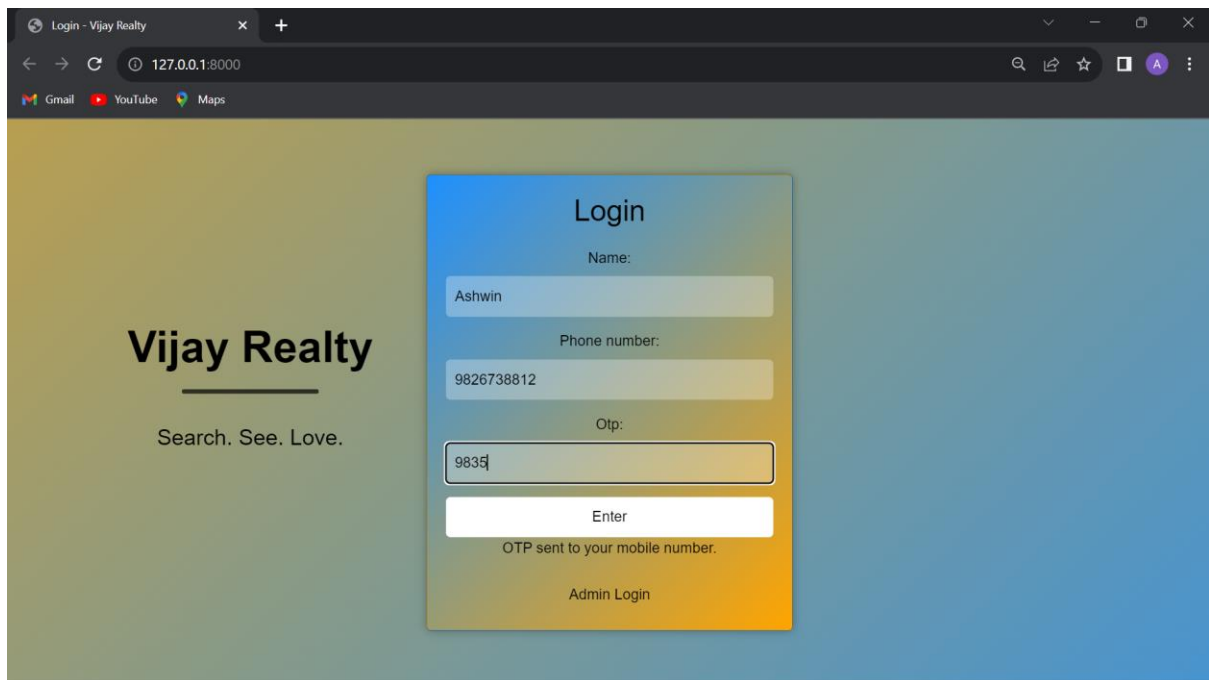


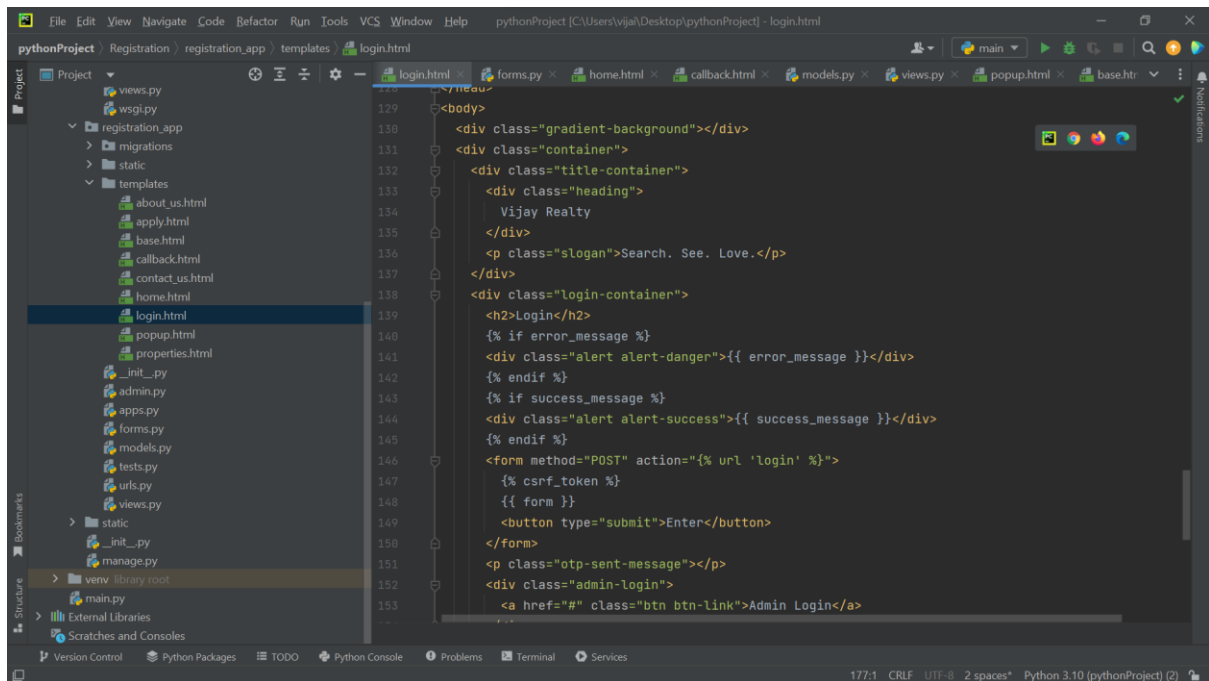
DJANGO WEBAPP UPDATE 03/06/2023

Tree of our project:



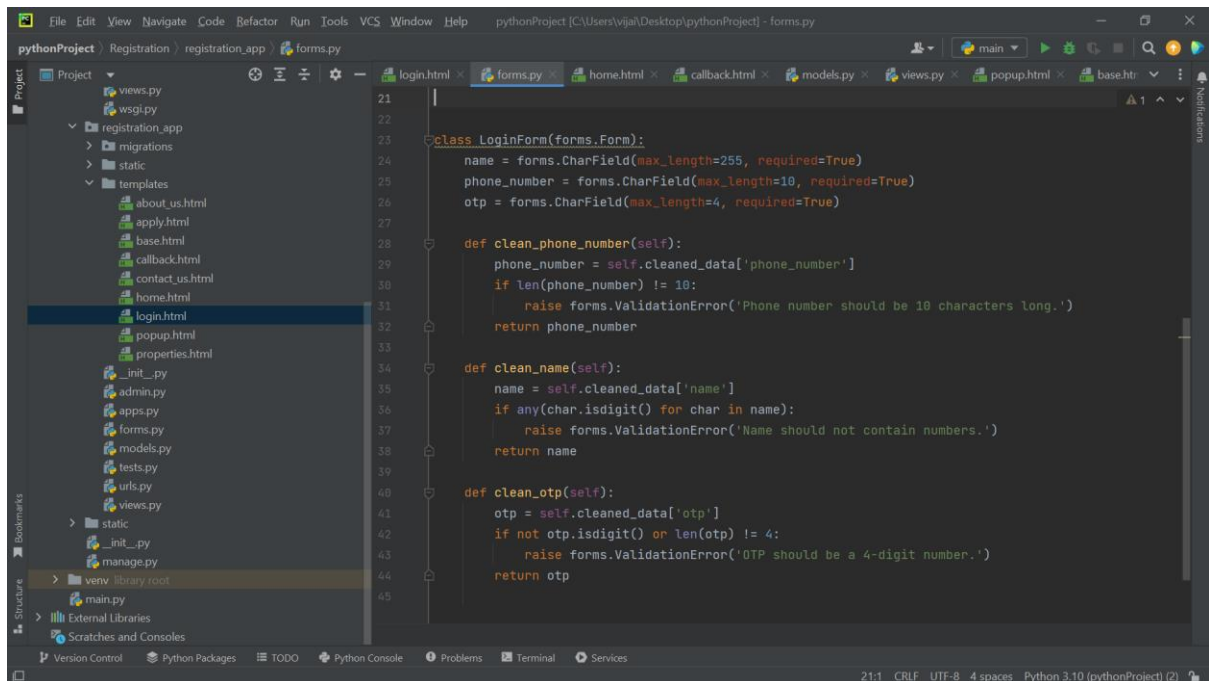
Authentication page





```
129 <body>
130 <div class="gradient-background"></div>
131 <div class="container">
132   <div class="title-container">
133     <div class="heading">
134       Vijay Realty
135     </div>
136     <p class="slogan">Search. See. Love.</p>
137   </div>
138   <div class="login-container">
139     <h2>Login</h2>
140     {% if error_message %}
141     <div class="alert alert-danger">{{ error_message }}</div>
142     {% endif %}
143     {% if success_message %}
144     <div class="alert alert-success">{{ success_message }}</div>
145     {% endif %}
146     <form method="POST" action="{% url 'login' %}">
147       {% csrf_token %}
148       {{ form }}
149       <button type="submit">Enter</button>
150     </form>
151     <p class="otp-sent-message"></p>
152     <div class="admin-login">
153       <a href="#" class="btn btn-link">Admin Login</a>
154     </div>
155   </div>
156 </div>
157 </body>
```

Forms.py



```
21
22
23 class LoginForm(forms.Form):
24     name = forms.CharField(max_length=255, required=True)
25     phone_number = forms.CharField(max_length=10, required=True)
26     otp = forms.CharField(max_length=4, required=True)
27
28     def clean_phone_number(self):
29         phone_number = self.cleaned_data['phone_number']
30         if len(phone_number) != 10:
31             raise forms.ValidationError('Phone number should be 10 characters long.')
32         return phone_number
33
34     def clean_name(self):
35         name = self.cleaned_data['name']
36         if any(char.isdigit() for char in name):
37             raise forms.ValidationError('Name should not contain numbers.')
38         return name
39
40     def clean_otp(self):
41         otp = self.cleaned_data['otp']
42         if not otp.isdigit() or len(otp) != 4:
43             raise forms.ValidationError('OTP should be a 4-digit number.')
44         return otp
45
```

Every login will be updated in our DB with Date and time

MySQL Workbench

SQL File 3* SQL File 4*

Limit to 50000 rows

```
select * from registration_app_logindetails;
```

#	id	name	phone_number	login_time
1	Vijay	9786788833	2023-05-28 16:14:30.144286	
2	Vijay	9786788833	2023-05-28 16:16:03.863462	
3	Anand	9899732145	2023-05-28 16:16:34.960144	
4	Vijay	9786788833	2023-05-28 16:31:12.403177	
5	Shiby	9942514497	2023-05-28 16:32:02.144710	
6	Prakash	9899741234	2023-05-28 16:40:16.319395	
7	Pandian	9943514489	2023-05-28 16:53:22.160295	
8	Priya	9123514489	2023-05-28 16:54:19.009214	

registration_app_logindetails 17

Action Output

#	Time	Action	Message	Duration / Fetch
22	21:40:09	show tables	23 row(s) returned	0.000 sec / 0.000 sec
23	21:40:31	select * from registration_app_logindetails LIMIT 0, 50000	0 row(s) returned	0.000 sec / 0.000 sec
24	21:42:52	select * from registration_app_logindetails LIMIT 0, 50000	0 row(s) returned	0.000 sec / 0.000 sec
25	21:44:45	select * from registration_app_logindetails LIMIT 0, 50000	1 row(s) returned	0.000 sec / 0.000 sec
26	22:19:46	select * from registration_app_logindetails LIMIT 0, 50000	6 row(s) returned	0.000 sec / 0.000 sec
27	00:29:59	select * from registration_app_logindetails LIMIT 0, 50000	17 row(s) returned	0.016 sec / 0.000 sec

Model.py

pythonProject [C:\Users\vijai\Desktop\pythonProject] - models.py

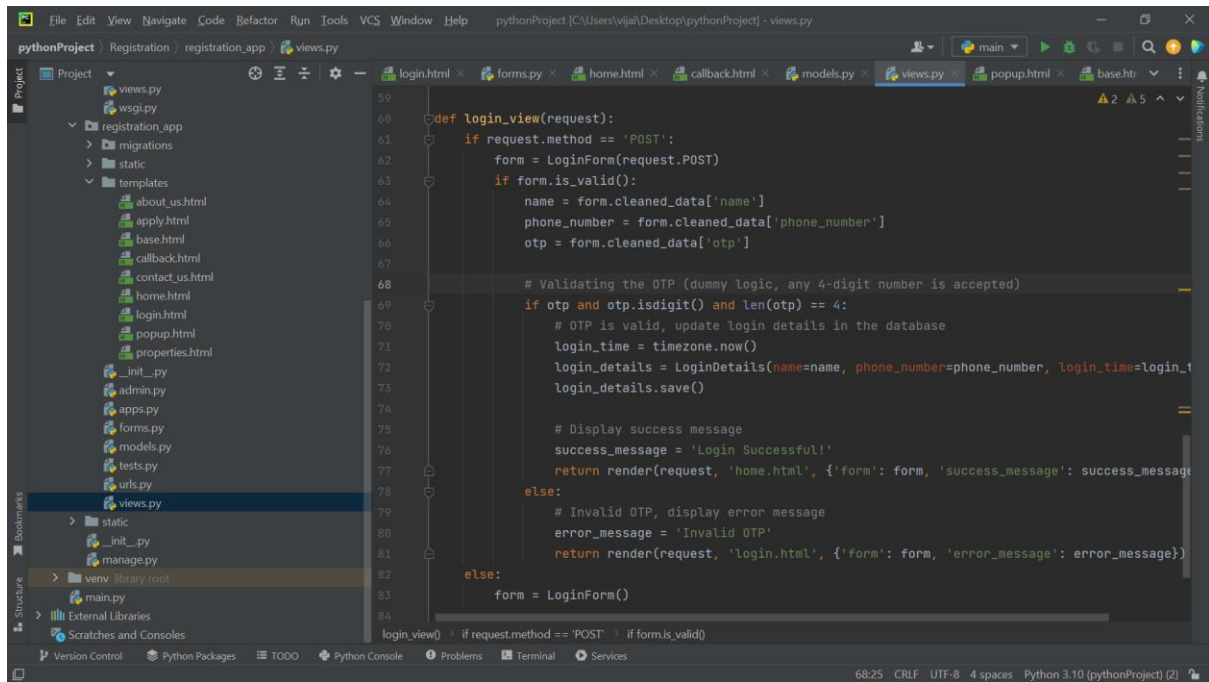
```
class CallbackRequest(models.Model):
    name = models.CharField(max_length=255)
    contact = models.CharField(max_length=20)
    property_type = models.CharField(max_length=20)
    budget = models.IntegerField()
    latitude = models.FloatField()
    longitude = models.FloatField()

class Property(models.Model):
    name = models.CharField(max_length=255)
    location = models.CharField(max_length=255)
    area = models.FloatField()
    property_type = models.CharField(max_length=100)
    price = models.DecimalField(max_digits=10, decimal_places=2)
    description = models.TextField()

    def __str__(self):
        return self.name

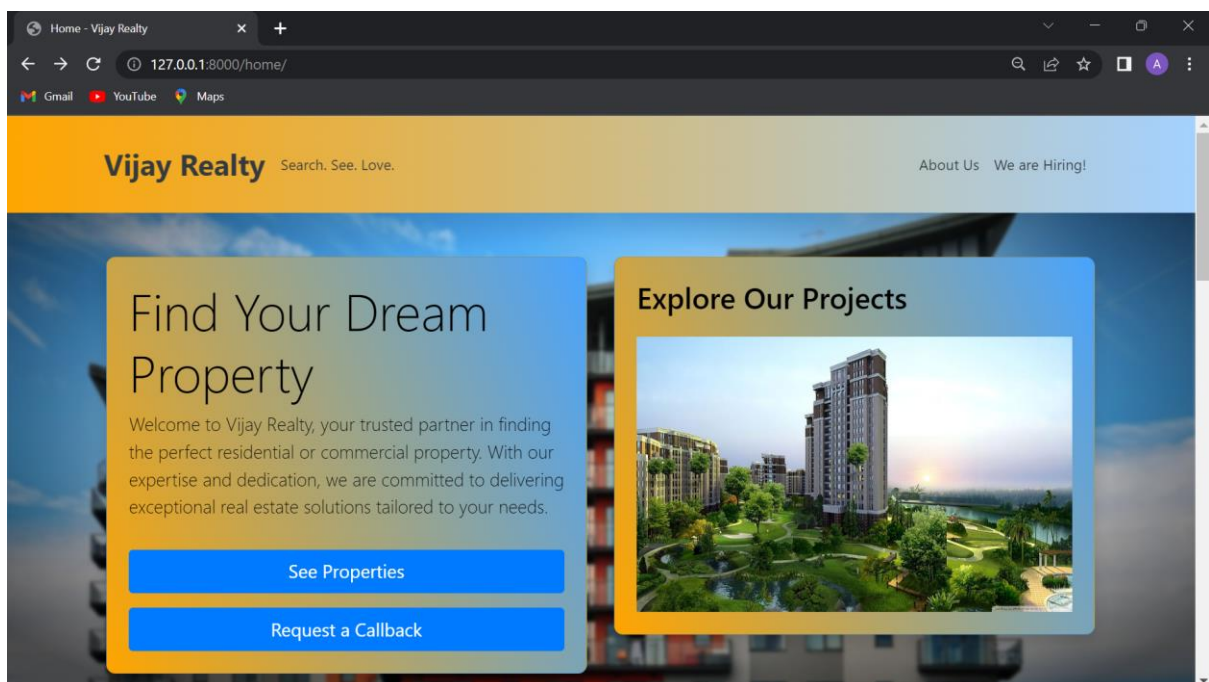
class LoginDetails(models.Model):
    name = models.CharField(max_length=100)
    phone_number = models.CharField(max_length=15)
    login_time = models.DateTimeField(auto_now_add=True)
```

views.py

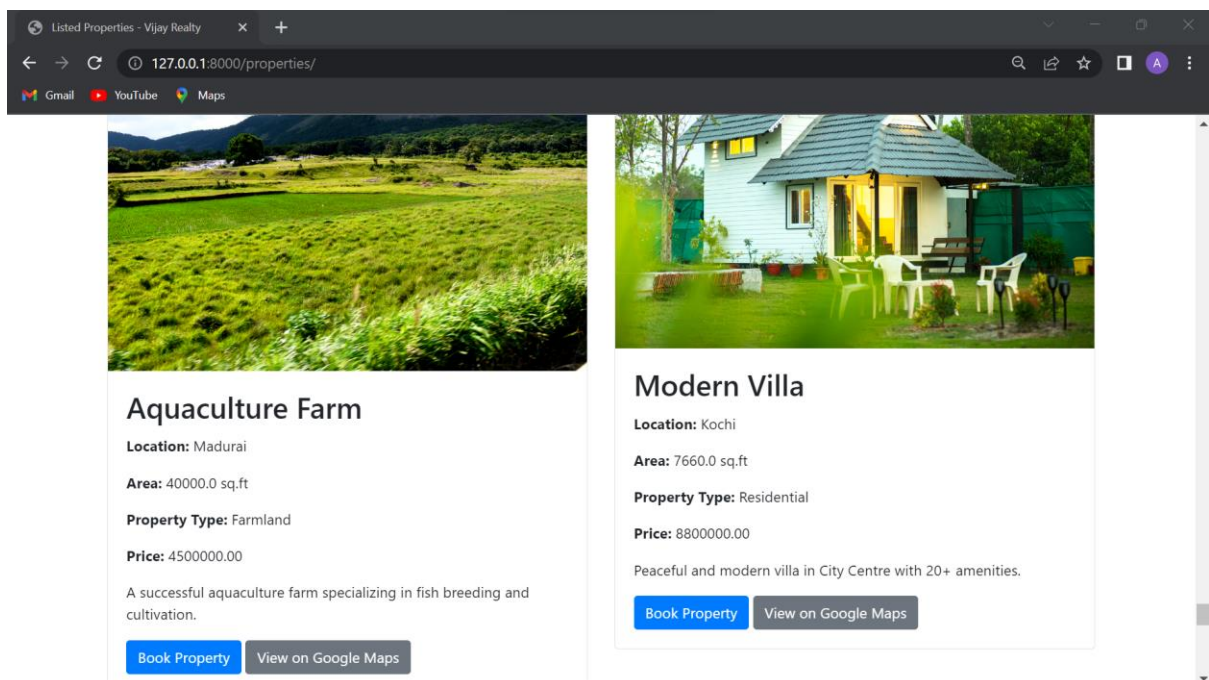
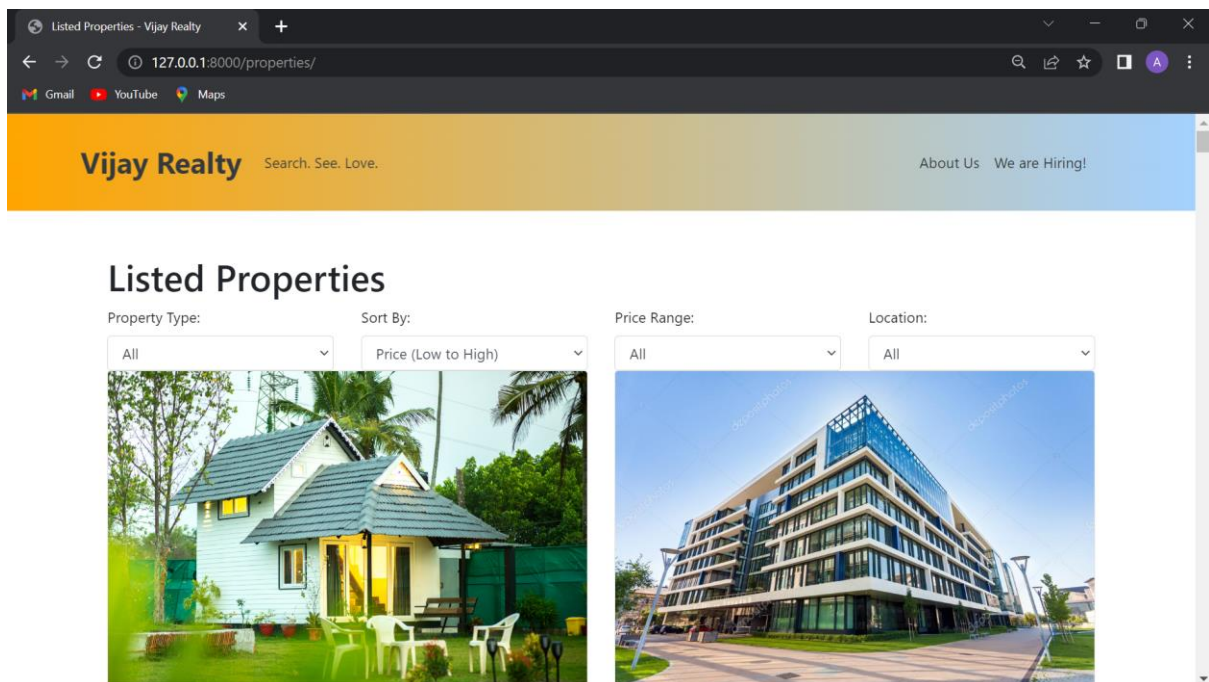


```
pythonProject [C:\Users\vijai\Desktop\pythonProject] - views.py
pythonProject | registration_app | views.py
Project | views.py | wsgi.py | migrations | static | templates | about_us.html | apply.html | base.html | callback.html | contact_us.html | home.html | login.html | popup.html | properties.html | _init_.py | admin.py | apps.py | forms.py | models.py | tests.py | urls.py | views.py | static | _init_.py | manage.py | venv | library root | main.py | External Libraries | Scratches and Consoles
60 def login_view(request):
61     if request.method == 'POST':
62         form = LoginForm(request.POST)
63         if form.is_valid():
64             name = form.cleaned_data['name']
65             phone_number = form.cleaned_data['phone_number']
66             otp = form.cleaned_data['otp']
67
68             # Validating the OTP (dummy logic, any 4-digit number is accepted)
69             if otp and otp.isdigit() and len(otp) == 4:
70                 # OTP is valid, update login details in the database
71                 login_time = timezone.now()
72                 login_details = LoginDetails(name=name, phone_number=phone_number, login_time=login_time)
73                 login_details.save()
74
75                 # Display success message
76                 success_message = 'Login Successful!'
77                 return render(request, 'home.html', {'form': form, 'success_message': success_message})
78             else:
79                 # Invalid OTP, display error message
80                 error_message = 'Invalid OTP'
81                 return render(request, 'login.html', {'form': form, 'error_message': error_message})
82         else:
83             form = LoginForm()
84
login_view() | if request.method == 'POST' | if form.is_valid()
```

Home page of our webapp



Listing page of our app with filter and sorting option:



Django forms:

The screenshot shows a web browser with three tabs: 'Apply - Vijay Realty', 'Flex - Bootstrap v5.3', and 'A Complete Guide to Flexbox | CSS'. The address bar shows '127.0.0.1:8000/apply/'. The page has a header with the 'Vijay Realty' logo, a search bar, and links for 'About Us' and 'We are Hiring!'. The main content area is divided into two columns. The left column contains job details for a 'Customer Care Executive' position (Job ID: 845, Date Published: May 28, 2023). It lists eligibility criteria (Excellent communication skills, Prior experience in customer service, Ability to handle customer inquiries and complaints, Strong problem-solving skills), responsibilities (Handle customer inquiries via phone, email, and live chat; Provide accurate information about products and services; Resolve customer complaints and ensure customer satisfaction; Maintain a high level of professionalism and empathy; Collaborate with other teams to address customer needs), and perks (Competitive salary, Healthcare benefits, Paid time off, Opportunity for growth and career advancement, Work From Home with paid amenities). The right column is titled 'Apply' and contains a form with fields for Name, Date of birth, Age, Email address, Phone number, LinkedIn URL, Address, and a PDF file upload section (with a 'Choose File' button and 'no file chosen' text). A 'Submit' button is at the bottom of the form. The footer contains 'Our Legacy' text and 'Contact' information for the Head Office and Branch Office.

Vijay Realty Search: See, Love. About Us · We are Hiring!

Customer Care Executive

Job ID: 845
Date Published: May 28, 2023

Eligibility:

- Excellent communication skills
- Prior experience in customer service
- Ability to handle customer inquiries and complaints
- Strong problem-solving skills

Responsibilities:

- Handle customer inquiries via phone, email, and live chat
- Provide accurate information about products and services
- Resolve customer complaints and ensure customer satisfaction
- Maintain a high level of professionalism and empathy
- Collaborate with other teams to address customer needs

Perks:

- Competitive salary
- Healthcare benefits
- Paid time off
- Opportunity for growth and career advancement
- Work From Home with paid amenities

Apply

Name:

Date of birth:

Age:

Email address:

Phone number:

LinkedIn URL:

Address:

PDF file: no file chosen

Our Legacy

With 25 years of real estate investment, development, and management experience, Vijay Realty has an unparalleled scale of delivery and an unmatched track record of customer-centric service excellence in India. We value your business and will provide you with the individual attention and

Contact

Head Office: 12, HM Plaza, Anna Nagar, Madurai, Tamilnadu - 625001
Contact: 0432 265 2277, +91 92333-92333
Branch Office: 1008, GK Towers, Guindy, Tamilnadu - 600012
Contact: +91 90244 60144

Admin Page :

The screenshot shows a web browser with a single tab titled 'Log in | Django site admin'. The address bar shows '127.0.0.1:8000/admin/login/?next=/admin/'. The page has a dark background. At the top, there is a blue header with the text 'Django administration' and a small icon. Below the header, there is a login form with two input fields: 'Username:' and 'Password:'. Below the password field is a blue 'Log in' button.

Django administration

Username:

Password:

Workbench:

The screenshot displays the MySQL Workbench interface. The left sidebar contains the 'MANAGEMENT' section with options like Server Status, Client Connections, Users and Privileges, Status and System Variables, Data Export, and Data Import/Restore. Below it is the 'INSTANCE' section with 'Administration' and 'Schemas' tabs. The main area shows a SQL editor with a query that inserts data into a table and then selects all records from 'registration_app_logindetails'. The 'Result Grid' shows the output of the query, displaying columns 'id', 'name', 'phone_number', and 'login_time' with 58 rows of data. The 'Output' section at the bottom shows the execution log with three actions: 'use Registration', 'show tables', and 'select * from registration_app_logindetails LIMIT 0, 50000'.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SQL File 4*

Limit to 50000 rows

```
45 ('Coffee Plantation', 'Kochi', 60000.00, 'Farmland', 6000000.00, 'A well-managed coffee plantation known for its premium beans.'),
46 ('Aquaculture Farm', 'Madurai', 40000.00, 'Farmland', 4500000.00, 'A successful aquaculture farm specializing in fish breeding and cult
47
48 • select * from registration_app_logindetails;
49
50 • use Registration;
```

Result Grid

	id	name	phone_number	login_time
43	Sri	8755754778	2023-05-29 10:58:14.697647	
44	Sri	8755754778	2023-05-29 10:59:26.537253	
45	Vijay	9878834122	2023-05-30 01:59:48.895858	
46	Anand	7855541224	2023-05-30 04:38:34.889213	
47	Naveen	8744524458	2023-05-30 04:31:02.687786	
48	Vijay	9786788833	2023-05-31 05:16:59.091685	
49	Priya	8554255689	2023-06-01 04:16:46.005876	
50	Vijay P...	9786788833	2023-06-01 09:09:01.731441	
51	Vijay P...	9786788833	2023-06-01 09:10:10.731441	

registration_app_logindetails 2 x

Output

#	Time	Action	Message	Duration / Fetch
1	07:04:25	use Registration	0 row(s) affected	0.000 sec
2	07:04:29	show tables	26 row(s) returned	0.015 sec / 0.000 sec
3	07:04:53	select * from registration_app_logindetails LIMIT 0, 50000	58 row(s) returned	0.016 sec / 0.000 sec

Object Info Session