**Program Code:**

**Settings.py:**

# Application definition

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'users.apps.UsersConfig',

    'services',

    'ratings',

    'chat',

    'dashboard.apps.DashboardConfig',

    'crispy\_forms',

    'crispy\_bootstrap5',

    'widget\_tweaks',

]

MIDDLEWARE = [

    'django.middleware.security.SecurityMiddleware',

    'django.contrib.sessions.middleware.SessionMiddleware',

    'django.middleware.common.CommonMiddleware',

    'django.middleware.csrf.CsrfViewMiddleware',

    'django.contrib.auth.middleware.AuthenticationMiddleware',

    'django.contrib.messages.middleware.MessageMiddleware',

    'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

ROOT\_URLCONF = 'homefix.urls'

TEMPLATES = [

    {

        'BACKEND': 'django.template.backends.django.DjangoTemplates',

        'DIRS': [],

        'APP\_DIRS': True,

        'OPTIONS': {

            'context\_processors': [

                'django.template.context\_processors.debug',

                'django.template.context\_processors.request',

                'django.contrib.auth.context\_processors.auth',

                'django.contrib.messages.context\_processors.messages',

            ],

        },

    },

]

WSGI\_APPLICATION = 'homefix.wsgi.application'

# Database

# https://docs.djangoproject.com/en/5.1/ref/settings/#databases

DATABASES = {

    'default': {

        'ENGINE': 'django.db.backends.sqlite3',

        'NAME': BASE\_DIR / 'db.sqlite3',

    }

}

**Views.py:**

def home(request):

    context = {}

    if request.user.is\_authenticated:

        print(f"User: {request.user}, Type: {request.user.user\_type}")  # Debug user info

        if request.user.user\_type == 'tenant':

            # Get recent service requests for tenant

            recent\_requests = ServiceRequest.objects.filter(

                tenant=request.user

            ).order\_by('-created\_at')[:4]

            context['recent\_requests'] = recent\_requests

            # Get direct service requests made by the tenant

            tenant\_direct\_requests = DirectServiceRequest.objects.filter(

                tenant=request.user

            ).order\_by('-created\_at')[:4]

            print(f"Direct Requests for {request.user}: {tenant\_direct\_requests}")  # Debug output

            context['tenant\_direct\_requests'] = tenant\_direct\_requests

        elif request.user.user\_type == 'serviceprovider':

            # Get upcoming jobs for service provider

            upcoming\_jobs = ServiceRequest.objects.filter(

                offers\_\_provider=request.user,

                offers\_\_status='accepted',

                status\_\_in=['assigned', 'in\_progress']

            ).order\_by('scheduled\_date')[:4]

            context['upcoming\_jobs'] = upcoming\_jobs

            # Get direct service requests assigned to the provider

            direct\_requests = DirectServiceRequest.objects.filter(

                provider=request.user

            ).order\_by('-created\_at')[:4]

            context['direct\_requests'] = direct\_requests

    return render(request, 'users/home.html', context)

def register\_tenant(request):

    if request.method == 'POST':

        form = TenantRegistrationForm(request.POST)

        if form.is\_valid():

            user = form.save(commit=False)

            user.is\_active = False

            user.save()

            send\_verification\_email(request, user)

            messages.success(request, 'Registration successful. Please check your email to verify your account.')

            return redirect('login')

    else:

        form = TenantRegistrationForm()

    return render(request, 'users/register\_tenant.html', {'form': form})

def register\_service\_provider(request):

    if request.method == 'POST':

        form = ServiceProviderRegistrationForm(request.POST)

        if form.is\_valid():

            user = form.save(commit=False)

            user.is\_active = False

            user.save()

            # Create provider profile

            provider\_profile = ProviderProfile.objects.create(

                user=user,

                rating=0.0,

                total\_reviews=0,

                is\_available=True,

                availability={}

            )

            # Add selected categories

            categories = form.cleaned\_data.get('categories', [])

            provider\_profile.categories.set(categories)

            send\_verification\_email(request, user)

            messages.success(request, 'Registration successful. Please check your email to verify your account.')

            return redirect('login')

    else:

        form = ServiceProviderRegistrationForm()

    return render(request, 'users/register\_service\_provider.html', {'form': form})

**urls.py:**

from django.urls import path

from django.contrib.auth import views as auth\_views

from . import views

from .forms import CustomPasswordResetForm

urlpatterns = [

    path('', views.home, name='home'),

    path('register/tenant/', views.register\_tenant, name='register\_tenant'),

    path('register/service-provider/', views.register\_service\_provider, name='register\_service\_provider'),

    path('login/', auth\_views.LoginView.as\_view(template\_name='users/login.html'), name='login'),

    path('logout/', views.user\_logout, name='logout'),

]

**Models.py:**

class CustomUser(AbstractUser):

    USER\_TYPE\_CHOICES = (

        ('tenant', 'Tenant'),

        ('serviceprovider', 'Service Provider'),

        ('admin', 'Admin'),

    )

    user\_type = models.CharField(max\_length=15, choices=USER\_TYPE\_CHOICES)

    email = models.EmailField(unique=True)

    phone\_regex = RegexValidator(

        regex=r'^[6-9]\d{9}$',

        message="Phone number must start with 6,7,8, or 9 and be exactly 10 digits long."

    )

    phone\_number = models.CharField(validators=[phone\_regex], max\_length=10)

    is\_email\_verified = models.BooleanField(default=False)

    is\_approved = models.BooleanField(default=False)

    def \_\_str\_\_(self):

        return self.username

class TenantProfile(models.Model):

    user = models.OneToOneField(

        "CustomUser", on\_delete=models.CASCADE, related\_name="tenant\_profile"

    )

    location = models.CharField(max\_length=255)

    phone\_number = models.CharField(max\_length=10, blank=True, null=True)

    move\_in\_date = models.DateField(blank=True, null=True)

    maintenance\_preferences = models.TextField(blank=True)  # Flexible storage

    def \_\_str\_\_(self):

        return f"{self.user.username}'s Tenant Profile"

    class Meta:

        verbose\_name = "Tenant Profile"

        verbose\_name\_plural = "Tenant Profiles"

class ServiceProviderProfile(models.Model):

    user = models.OneToOneField("users.CustomUser", on\_delete=models.CASCADE, related\_name='provider\_profile')

    service\_location = models.CharField(max\_length=255)

    service\_provided = models.ManyToManyField("services.ServiceCategory", related\_name='service\_provided')

    certifications = models.TextField(blank=True)

    def \_\_str\_\_(self):

        return f"{self.user.username}'s Service Provider Profile"

**forms.py:**

class TenantRegistrationForm(UserCreationForm):

    class Meta:

        model = CustomUser

        fields = ('username', 'email', 'phone\_number', 'password1', 'password2')

    def save(self, commit=True):

        user = super().save(commit=False)

        user.user\_type = 'tenant'

        if commit:

            user.save()

            TenantProfile.objects.create(

                user=user

            )

        return user

class ServiceProviderRegistrationForm(UserCreationForm):

    service\_location = forms.CharField(max\_length=255)

    categories = forms.ModelMultipleChoiceField(

        queryset=ServiceCategory.objects.all(),

        widget=forms.CheckboxSelectMultiple,  # Converts multi-select into checkboxes

        required=True,

        help\_text="Select the service categories you provide"

    )

    certifications = forms.CharField(widget=forms.Textarea, required=False)

    class Meta:

        model = CustomUser

        fields = ('username', 'email', 'phone\_number', 'password1', 'password2')

    def save(self, commit=True):

        user = super().save(commit=False)

        user.user\_type = 'serviceprovider'

        if commit:

            user.save()

            ServiceProviderProfile.objects.create(

                user=user

            )

        return user

**templates:**

**register.html:**

{% extends 'users/base.html' %}

{% block title %}Register as Tenant{% endblock %}

{% block content %}

<div class="max-w-md mx-auto">

    <div class="bg-white rounded-lg shadow-sm p-8">

        <h2 class="text-2xl font-bold text-center text-gray-900 mb-8">Register as Tenant</h2>

        <form method="post" novalidate class="space-y-6">

            {% csrf\_token %}

            {% for field in form %}

                <div>

                    <label for="{{ field.id\_for\_label }}" class="block text-sm font-medium text-gray-700 mb-1">

                        {{ field.label }}

                    </label>

                    {% if field.name == 'password1' or field.name == 'password2' %}

                        <input

                            type="{{ field.field.widget.input\_type }}"

                            name="{{ field.name }}"

                            id="{{ field.id\_for\_label }}"

                            required

                            class="appearance-none block w-full px-3 py-2 border {% if field.errors %}border-red-300{% else %}border-gray-300{% endif %} rounded-md shadow-sm placeholder-gray-400 focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm"

                        >

                    {% else %}

                        <input

                            type="text"

                            name="{{ field.name }}"

                            id="{{ field.id\_for\_label }}"

                            value="{{ field.value|default:'' }}"

                            required

                            class="appearance-none block w-full px-3 py-2 border {% if field.errors %}border-red-300{% else %}border-gray-300{% endif %} rounded-md shadow-sm placeholder-gray-400 focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm"

                        >

                    {% endif %}

                    {% if field.errors %}

                        <p class="mt-1 text-sm text-red-600">{{ field.errors|join:", " }}</p>

                    {% endif %}

                    {% if field.help\_text %}

                        <p class="mt-1 text-sm text-gray-500">{{ field.help\_text }}</p>

                    {% endif %}

                </div>

            {% endfor %}

            <button type="submit" class="w-full flex justify-center py-2 px-4 border border-transparent rounded-md shadow-sm text-sm font-medium text-white bg-indigo-600 hover:bg-indigo-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500">

                Register

            </button>

        </form>

        <div class="mt-6 text-center">

            <p class="text-sm text-gray-600">

                Already have an account?

                <a href="{% url 'login' %}" class="font-medium text-indigo-600 hover:text-indigo-500">

                    Login here

                </a>

            </p>

        </div>

    </div>

</div>

{% endblock %}

**Login.html:**

{% extends 'users/base.html' %}

{% block title %}Login{% endblock %}

{% block content %}

<div class="max-w-md mx-auto">

    <div class="bg-white rounded-lg shadow-sm p-8">

        <h2 class="text-2xl font-bold text-center text-gray-900 mb-8">Login to HomeFix</h2>

        <form method="post" novalidate class="space-y-6">

            {% csrf\_token %}

            {% for field in form %}

                <div>

                    <label for="{{ field.id\_for\_label }}" class="block text-sm font-medium text-gray-700 mb-1">

                        {{ field.label }}

                    </label>

                    {{ field.errors }}

                    <input

                        type="{{ field.field.widget.input\_type }}"

                        name="{{ field.name }}"

                        id="{{ field.id\_for\_label }}"

                        required

                        class="appearance-none block w-full px-3 py-2 border {% if field.errors %}border-red-300{% else %}border-gray-300{% endif %} rounded-md shadow-sm placeholder-gray-400 focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm"

                    >

                    {% if field.errors %}

                        <p class="mt-1 text-sm text-red-600">{{ field.errors|join:", " }}</p>

                    {% endif %}

                </div>

            {% endfor %}

            <button type="submit" class="w-full flex justify-center py-2 px-4 border border-transparent rounded-md shadow-sm text-sm font-medium text-white bg-indigo-600 hover:bg-indigo-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500">

                Login

            </button>

        </form>

        <div class="mt-6 space-y-4">

            <div class="text-center">

                <a href="{% url 'password\_reset' %}" class="text-sm font-medium text-indigo-600 hover:text-indigo-500">

                    Forgot your password?

                </a>

            </div>

            <div class="relative">

                <div class="absolute inset-0 flex items-center">

                    <div class="w-full border-t border-gray-300"></div>

                </div>

                <div class="relative flex justify-center text-sm">

                    <span class="px-2 bg-white text-gray-500">Don't have an account?</span>

                </div>

            </div>

            <div class="grid grid-cols-2 gap-3">

                <a href="{% url 'register\_tenant' %}" class="inline-flex justify-center items-center px-4 py-2 border border-gray-300 shadow-sm text-sm font-medium rounded-md text-gray-700 bg-white hover:bg-gray-50 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500">

                    <i class="fas fa-home mr-2"></i>

                    Register as Tenant

                </a>

                <a href="{% url 'register\_service\_provider' %}" class="inline-flex justify-center items-center px-4 py-2 border border-gray-300 shadow-sm text-sm font-medium rounded-md text-gray-700 bg-white hover:bg-gray-50 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500">

                    <i class="fas fa-tools mr-2"></i>

                    Register as Provider

                </a>

            </div>

        </div>

    </div>

</div>

{% endblock %}