***Necesito tus conocimientos en Python con su framework django para que me guíes, necesito, primeramente la creación de un ambiente virtual en Python para luego continuar con la creación un proyecto en django, llamado Tienda, con las aplicaciones Tareas, Productos, Pedidos, Carrito, Categorías, Usuarios, etc., empecemos con la seguridad, los usuarios tendrán roles, las cuales serán administrador (administrador del sistema, acceso total), empleados (empleado de la tienda, acceso limitado), cliente (cliente de la tienda), la tabla usuario será creada utilizando AbstractUser.***

Te guiaré paso a paso en la creación del proyecto Django "Tienda". Empecemos:

## 1. Creación del Ambiente Virtual

```bash

# Crear el directorio del proyecto

mkdir Tienda

cd Tienda

# Crear ambiente virtual (elige una de estas opciones)

# Opción 1: Con venv (recomendado)

python -m venv venv

# Opción 2: Con virtualenv

# virtualenv venv

# Activar el ambiente virtual

# Windows:

venv\Scripts\activate

# Linux/Mac:

source venv/bin/activate

```

## 2. Instalación de Django

```bash

# Instalar Django

pip install django

# También instalar pillow para manejo de imágenes (útil para productos)

pip install pillow

```

## 3. Creación del Proyecto Django

```bash

# Crear el proyecto Tienda

django-admin startproject Tienda .

# Crear las aplicaciones

python manage.py startapp usuarios

python manage.py startapp productos

python manage.py startapp pedidos

python manage.py startapp carrito

python manage.py startapp categorias

python manage.py startapp tareas

```

## 4. Configuración del Modelo Usuario Personalizado

\*\*usuarios/models.py:\*\*

```python

from django.contrib.auth.models import AbstractUser

from django.db import models

class Rol(models.Model):

nombre = models.CharField(max\_length=50, unique=True)

descripcion = models.TextField(blank=True)

class Meta:

verbose\_name = 'Rol'

verbose\_name\_plural = 'Roles'

def \_\_str\_\_(self):

return self.nombre

class Usuario(AbstractUser):

ROLES = (

('administrador', 'Administrador'),

('empleado', 'Empleado'),

('cliente', 'Cliente'),

)

rol = models.CharField(max\_length=20, choices=ROLES, default='cliente')

telefono = models.CharField(max\_length=15, blank=True)

direccion = models.TextField(blank=True)

fecha\_nacimiento = models.DateField(null=True, blank=True)

fecha\_creacion = models.DateTimeField(auto\_now\_add=True)

fecha\_actualizacion = models.DateTimeField(auto\_now=True)

activo = models.BooleanField(default=True)

# Campos adicionales para clientes

dni = models.CharField(max\_length=20, blank=True, unique=True, null=True)

# Campos adicionales para empleados

fecha\_contratacion = models.DateField(null=True, blank=True)

salario = models.DecimalField(max\_digits=10, decimal\_places=2, null=True, blank=True)

class Meta:

verbose\_name = 'Usuario'

verbose\_name\_plural = 'Usuarios'

def \_\_str\_\_(self):

return f"{self.username} - {self.get\_rol\_display()}"

def es\_administrador(self):

return self.rol == 'administrador'

def es\_empleado(self):

return self.rol == 'empleado'

def es\_cliente(self):

return self.rol == 'cliente'

```

\*\*usuarios/forms.py:\*\*

```python

from django import forms

from django.contrib.auth.forms import UserCreationForm, UserChangeForm

from .models import Usuario

class UsuarioCreationForm(UserCreationForm):

class Meta:

model = Usuario

fields = ('username', 'email', 'rol', 'first\_name', 'last\_name')

def save(self, commit=True):

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

if commit:

user.save()

return user

class UsuarioChangeForm(UserChangeForm):

class Meta:

model = Usuario

fields = ('username', 'email', 'rol', 'first\_name', 'last\_name',

'telefono', 'direccion', 'fecha\_nacimiento', 'activo')

```

\*\*usuarios/admin.py:\*\*

```python

from django.contrib import admin

from django.contrib.auth.admin import UserAdmin

from .models import Usuario, Rol

class UsuarioAdmin(UserAdmin):

list\_display = ('username', 'email', 'first\_name', 'last\_name', 'rol', 'activo')

list\_filter = ('rol', 'activo', 'is\_staff', 'is\_superuser')

fieldsets = UserAdmin.fieldsets + (

('Información adicional', {

'fields': ('rol', 'telefono', 'direccion', 'fecha\_nacimiento',

'dni', 'fecha\_contratacion', 'salario', 'activo')

}),

)

add\_fieldsets = UserAdmin.add\_fieldsets + (

('Información adicional', {

'fields': ('rol', 'telefono', 'email')

}),

)

admin.site.register(Usuario, UsuarioAdmin)

admin.site.register(Rol)

```

## 5. Configuración del Settings

\*\*Tienda/settings.py:\*\*

```python

import os

from pathlib import Path

BASE\_DIR = Path(\_\_file\_\_).resolve().parent.parent

SECRET\_KEY = 'tu-clave-secreta-aqui' # Cambia esto en producción

DEBUG = True

ALLOWED\_HOSTS = []

# Aplicaciones instaladas

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

# Aplicaciones personalizadas

'usuarios',

'productos',

'pedidos',

'carrito',

'categorias',

'tareas',

]

# Modelo de usuario personalizado

AUTH\_USER\_MODEL = 'usuarios.Usuario'

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 = 'Tienda.urls'

TEMPLATES = [

{

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

'DIRS': [os.path.join(BASE\_DIR, 'templates')],

'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 = 'Tienda.wsgi.application'

DATABASES = {

'default': {

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

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

}

}

# ... resto de configuraciones por defecto ...

STATIC\_URL = '/static/'

STATICFILES\_DIRS = [os.path.join(BASE\_DIR, 'static')]

MEDIA\_URL = '/media/'

MEDIA\_ROOT = os.path.join(BASE\_DIR, 'media')

```

## 6. Migraciones Iniciales

```bash

# Crear migraciones

python manage.py makemigrations

# Aplicar migraciones

python manage.py migrate

# Crear superusuario

python manage.py createsuperuser

```

## 7. Middleware de Seguridad (Opcional)

Puedes crear un middleware para controlar el acceso por roles:

\*\*usuarios/middleware.py:\*\*

```python

from django.http import HttpResponseForbidden

from django.urls import reverse

from django.shortcuts import redirect

class RolMiddleware:

def \_\_init\_\_(self, get\_response):

self.get\_response = get\_response

def \_\_call\_\_(self, request):

response = self.get\_response(request)

return response

def process\_view(self, request, view\_func, view\_args, view\_kwargs):

# Aquí puedes agregar lógica de control de acceso por roles

if hasattr(view\_func, 'roles\_permitidos'):

if not request.user.is\_authenticated:

return redirect('login')

if request.user.rol not in view\_func.roles\_permitidos:

return HttpResponseForbidden("No tienes permisos para acceder a esta página")

return None

```

## 8. Decoradores para Control de Acceso

\*\*usuarios/decorators.py:\*\*

```python

from functools import wraps

from django.http import HttpResponseForbidden

from django.shortcuts import redirect

def rol\_requerido(roles\_permitidos):

def decorator(view\_func):

@wraps(view\_func)

def \_wrapped\_view(request, \*args, \*\*kwargs):

if not request.user.is\_authenticated:

return redirect('login')

if request.user.rol not in roles\_permitidos:

return HttpResponseForbidden("Acceso denegado")

return view\_func(request, \*args, \*\*kwargs)

# Agregar atributo para el middleware

\_wrapped\_view.roles\_permitidos = roles\_permitidos

return \_wrapped\_view

return decorator

# Decoradores específicos

def administrador\_required(view\_func):

return rol\_requerido(['administrador'])(view\_func)

def empleado\_required(view\_func):

return rol\_requerido(['administrador', 'empleado'])(view\_func)

def cliente\_required(view\_func):

return rol\_requerido(['cliente'])(view\_func)

```

## 9. URLs Principales

\*\*Tienda/urls.py:\*\*

```python

from django.contrib import admin

from django.urls import path, include

from django.conf import settings

from django.conf.urls.static import static

urlpatterns = [

path('admin/', admin.site.urls),

path('usuarios/', include('usuarios.urls')),

path('productos/', include('productos.urls')),

path('pedidos/', include('pedidos.urls')),

path('carrito/', include('carrito.urls')),

path('categorias/', include('categorias.urls')),

path('tareas/', include('tareas.urls')),

]

if settings.DEBUG:

urlpatterns += static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT)

```

## 10. Próximos Pasos

Para continuar, necesitaríamos:

1. \*\*Configurar las demás aplicaciones\*\* (productos, pedidos, carrito, categorías, tareas)

2. \*\*Crear los modelos correspondientes\*\* para cada aplicación

3. \*\*Implementar las vistas y templates\*\*

4. \*\*Configurar el sistema de autenticación\*\*

5. \*\*Agregar permisos y restricciones por roles\*\*

¿Te gustaría que continuemos con alguna aplicación específica a continuación?

***también necesito instalar el paquete django bootstrap 5.***

¡Perfecto! Vamos a instalar y configurar django-bootstrap5 para tu proyecto.

## 1. Instalación del Paquete

```bash

# Asegúrate de tener el ambiente virtual activado

pip install django-bootstrap5

```

## 2. Configuración en settings.py

\*\*Tienda/settings.py:\*\*

```python

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

# Aplicaciones de terceros

'django\_bootstrap5',

# Aplicaciones personalizadas

'usuarios',

'productos',

'pedidos',

'carrito',

'categorias',

'tareas',

]

# Configuración adicional de django-bootstrap5 (opcional)

BOOTSTRAP5 = {

'theme\_url': 'https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css',

'javascript\_url': 'https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js',

'css\_url': None, # Usar CDN por defecto

'javascript\_in\_head': False,

'include\_jquery': False,

'horizontal\_label\_class': 'col-md-3',

'horizontal\_field\_class': 'col-md-9',

'set\_placeholder': True,

'required\_css\_class': '',

'error\_css\_class': 'is-invalid',

'success\_css\_class': 'is-valid',

'formset\_renderers': {

'default': 'django\_bootstrap5.renderers.FormsetRenderer',

},

'form\_renderers': {

'default': 'django\_bootstrap5.renderers.FormRenderer',

},

'field\_renderers': {

'default': 'django\_bootstrap5.renderers.FieldRenderer',

'inline': 'django\_bootstrap5.renderers.InlineFieldRenderer',

},

}

```

## 3. Configuración de Templates Base

Crea la estructura de directorios para templates:

```bash

mkdir templates

mkdir templates/registration

```

\*\*templates/base.html:\*\*

```html

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>{% block title %}Tienda Online{% endblock %}</title>

{% load django\_bootstrap5 %}

{% bootstrap\_css %}

{% bootstrap\_javascript %}

{% block extra\_css %}{% endblock %}

</head>

<body>

<!-- Navbar -->

<nav class="navbar navbar-expand-lg navbar-dark bg-dark">

<div class="container">

<a class="navbar-brand" href="{% url 'home' %}">Tienda Online</a>

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNav">

<ul class="navbar-nav me-auto">

<li class="nav-item">

<a class="nav-link" href="{% url 'productos:lista' %}">Productos</a>

</li>

{% if user.is\_authenticated %}

{% if user.es\_administrador or user.es\_empleado %}

<li class="nav-item">

<a class="nav-link" href="{% url 'admin:index' %}">Administración</a>

</li>

<li class="nav-item">

<a class="nav-link" href="{% url 'tareas:lista' %}">Tareas</a>

</li>

{% endif %}

<li class="nav-item">

<a class="nav-link" href="{% url 'carrito:detalle' %}">Carrito</a>

</li>

<li class="nav-item">

<a class="nav-link" href="{% url 'pedidos:lista' %}">Mis Pedidos</a>

</li>

{% endif %}

</ul>

<ul class="navbar-nav">

{% if user.is\_authenticated %}

<li class="nav-item dropdown">

<a class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown">

{{ user.username }}

</a>

<ul class="dropdown-menu">

<li><a class="dropdown-item" href="{% url 'usuarios:perfil' %}">Perfil</a></li>

<li><hr class="dropdown-divider"></li>

<li><a class="dropdown-item" href="{% url 'logout' %}">Cerrar Sesión</a></li>

</ul>

</li>

{% else %}

<li class="nav-item">

<a class="nav-link" href="{% url 'login' %}">Iniciar Sesión</a>

</li>

<li class="nav-item">

<a class="nav-link" href="{% url 'usuarios:registro' %}">Registrarse</a>

</li>

{% endif %}

</ul>

</div>

</div>

</nav>

<!-- Contenido principal -->

<main class="container mt-4">

{% bootstrap\_messages %}

{% block content %}{% endblock %}

</main>

<!-- Footer -->

<footer class="bg-dark text-light mt-5 py-4">

<div class="container text-center">

<p>&copy; 2024 Tienda Online. Todos los derechos reservados.</p>

</div>

</footer>

{% block extra\_js %}{% endblock %}

</body>

</html>

```

## 4. Templates de Autenticación con Bootstrap

\*\*templates/registration/login.html:\*\*

```html

{% extends 'base.html' %}

{% load django\_bootstrap5 %}

{% block title %}Iniciar Sesión - Tienda Online{% endblock %}

{% block content %}

<div class="row justify-content-center">

<div class="col-md-6">

<div class="card">

<div class="card-header">

<h3 class="text-center">Iniciar Sesión</h3>

</div>

<div class="card-body">

<form method="post">

{% csrf\_token %}

{% bootstrap\_form form %}

<div class="d-grid gap-2">

{% bootstrap\_button "Iniciar Sesión" button\_type="submit" button\_class="btn-primary" %}

</div>

</form>

<hr>

<div class="text-center">

<a href="{% url 'usuarios:registro' %}" class="btn btn-outline-secondary">

¿No tienes cuenta? Regístrate

</a>

</div>

</div>

</div>

</div>

</div>

{% endblock %}

```

## 5. Formularios con Bootstrap

\*\*usuarios/forms.py\*\* (actualizado):

```python

from django import forms

from django.contrib.auth.forms import UserCreationForm, UserChangeForm

from .models import Usuario

class UsuarioCreationForm(UserCreationForm):

class Meta:

model = Usuario

fields = ('username', 'email', 'rol', 'first\_name', 'last\_name',

'telefono', 'direccion', 'fecha\_nacimiento')

widgets = {

'fecha\_nacimiento': forms.DateInput(attrs={'type': 'date'}),

}

def \_\_init\_\_(self, \*args, \*\*kwargs):

super().\_\_init\_\_(\*args, \*\*kwargs)

# Personalizar labels si es necesario

self.fields['fecha\_nacimiento'].label = 'Fecha de Nacimiento'

class UsuarioChangeForm(UserChangeForm):

class Meta:

model = Usuario

fields = ('username', 'email', 'first\_name', 'last\_name',

'telefono', 'direccion', 'fecha\_nacimiento')

widgets = {

'fecha\_nacimiento': forms.DateInput(attrs={'type': 'date'}),

}

class RegistroClienteForm(UserCreationForm):

class Meta:

model = Usuario

fields = ('username', 'email', 'first\_name', 'last\_name',

'telefono', 'direccion', 'fecha\_nacimiento', 'dni')

widgets = {

'fecha\_nacimiento': forms.DateInput(attrs={'type': 'date'}),

}

def save(self, commit=True):

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

user.rol = 'cliente' # Asignar rol automáticamente

if commit:

user.save()

return user

```

## 6. Vistas con Bootstrap

\*\*usuarios/views.py:\*\*

```python

from django.shortcuts import render, redirect

from django.contrib.auth import login

from django.contrib.auth.decorators import login\_required

from django.contrib import messages

from .forms import RegistroClienteForm

from .decorators import rol\_requerido

def registro\_cliente(request):

if request.method == 'POST':

form = RegistroClienteForm(request.POST)

if form.is\_valid():

user = form.save()

login(request, user)

messages.success(request, '¡Registro exitoso! Bienvenido a nuestra tienda.')

return redirect('home')

else:

form = RegistroClienteForm()

return render(request, 'usuarios/registro.html', {'form': form})

@login\_required

def perfil\_usuario(request):

return render(request, 'usuarios/perfil.html')

@login\_required

@rol\_requerido(['administrador', 'empleado'])

def panel\_empleado(request):

return render(request, 'usuarios/panel\_empleado.html')

@login\_required

@rol\_requerido(['administrador'])

def panel\_administrador(request):

return render(request, 'usuarios/panel\_administrador.html')

```

## 7. URLs para Usuarios

\*\*usuarios/urls.py:\*\*

```python

from django.urls import path

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

from . import views

app\_name = 'usuarios'

urlpatterns = [

path('registro/', views.registro\_cliente, name='registro'),

path('perfil/', views.perfil\_usuario, name='perfil'),

path('panel-empleado/', views.panel\_empleado, name='panel\_empleado'),

path('panel-admin/', views.panel\_administrador, name='panel\_administrador'),

# URLs de autenticación por defecto

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

path('logout/', auth\_views.LogoutView.as\_view(), name='logout'),

]

```

## 8. Template de Registro

\*\*templates/usuarios/registro.html:\*\*

```html

{% extends 'base.html' %}

{% load django\_bootstrap5 %}

{% block title %}Registro - Tienda Online{% endblock %}

{% block content %}

<div class="row justify-content-center">

<div class="col-md-8">

<div class="card">

<div class="card-header">

<h3 class="text-center">Registro de Cliente</h3>

</div>

<div class="card-body">

<form method="post">

{% csrf\_token %}

{% bootstrap\_form form %}

<div class="d-grid gap-2">

{% bootstrap\_button "Registrarse" button\_type="submit" button\_class="btn-primary" %}

</div>

</form>

<hr>

<div class="text-center">

<a href="{% url 'login' %}" class="btn btn-outline-secondary">

¿Ya tienes cuenta? Inicia Sesión

</a>

</div>

</div>

</div>

</div>

</div>

{% endblock %}

```

## 9. Aplicar Migraciones y Probar

```bash

# Aplicar migraciones

python manage.py makemigrations

python manage.py migrate

# Crear superusuario (si aún no lo has hecho)

python manage.py createsuperuser

# Ejecutar el servidor

python manage.py runserver

```

Ahora tu proyecto tiene django-bootstrap5 integrado con:

- ✅ Formularios estilizados con Bootstrap

- ✅ Navbar responsivo

- ✅ Mensajes de alerta bonitos

- ✅ Estructura base para templates

- ✅ Sistema de autenticación con estilo

¿Te gustaría que continuemos con alguna aplicación específica ahora?