

COMPETENCIA A EVALUAR: **DESARROLLO DE SOFTWARE**

Aksel Deneken Maldonado A01711966

Resumen integrador de los conceptos atendidos durante el curso.

Aprendimos a identificar qué necesita un sistema antes de desarrollarlo. Esto incluye tanto funciones que debe cumplir (como registrar usuarios o generar reportes), como condiciones bajo las que debe operar (velocidad, seguridad, compatibilidad). A partir de esos requerimientos, se definieron escenarios de uso que sirven como base para el diseño y validación posterior.

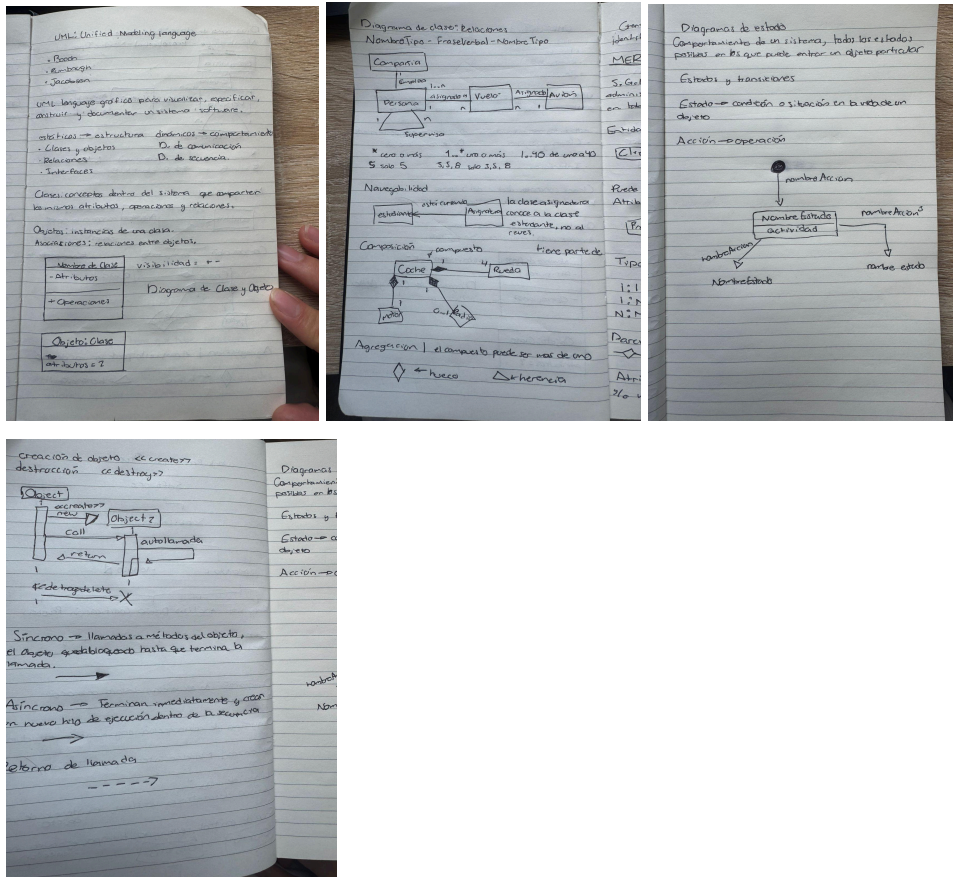
El diseño de los componentes se basa en que cada parte tenga una responsabilidad clara, y que el sistema esté estructurado de forma que los elementos no dependan innecesariamente entre sí. Separar responsabilidades; Aislar funcionalidades; Reducir dependencias.

Modelado con Diagramas uml

Utilizamos distintos tipos de diagramas UML que nos permitieron visualizar cómo debía funcionar el sistema antes de codificar:

- Componentes: muestran las partes principales y cómo se relacionan.
- Secuencia: detallan cómo se comunican los elementos en un flujo de acciones.
- Estados: representan cómo cambia un objeto según las acciones que ocurren.

APUNTES:



Manejo de Datos y Reglas

Diseñamos estructuras para que los datos se guarden de forma permanente, respetando reglas que aseguren su validez. Esto incluye:

- Crear tablas bien definidas
- Usar llaves primarias y foráneas

El manejo correcto de la persistencia es esencial para que el sistema funcione con datos reales.

Pruebas del Sistema

El sistema debe probarse para comprobar que lo que se diseñó cumple lo que se pidió. Las pruebas unitarias revisan funciones específicas, mientras que las pruebas

de integración validan cómo interactúan varias partes juntas. La clave es que no se hicieron pruebas al azar: se diseñaron a partir de los requerimientos.

Herramientas Técnicas

Durante el curso utilizamos varias herramientas que nos ayudaron a llevar a cabo el desarrollo de forma profesional:

- Para diseño: herramientas visuales como CANVA, LucidChart
- Para control de versiones: Git y GitHub
- Tecnologías comunes: Python, HTML, CSS, frameworks como Django

También fue clave el uso de canales de comunicación como Discord o WhatsApp para coordinar al equipo.

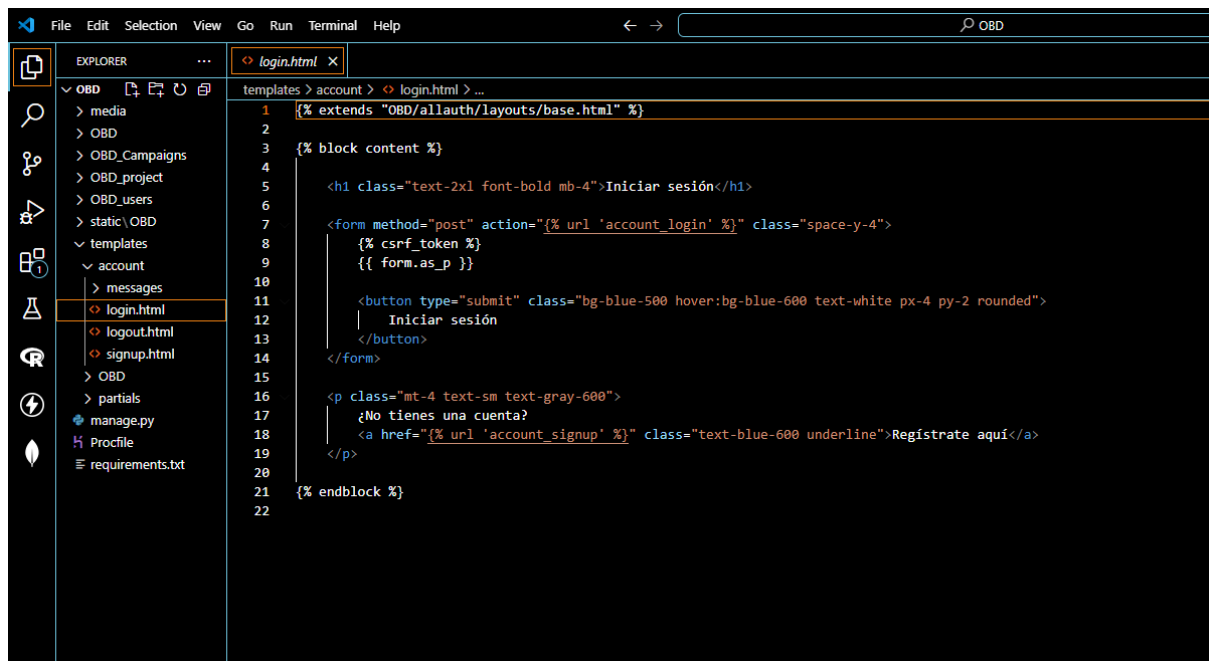
Repositorio:

<https://github.com/saucedor/OBD/blob/main>

Contribuciones individuales en el repositorio del proyecto

Durante el desarrollo del sistema, mi rol se centró principalmente en el frontend y en la estructura visual de las vistas principales, así como en el diseño de varios bosquejos base que guiaron la implementación. A la semana 9, el proyecto tenía un avance sólido, y ya se habían estructurado las apps necesarias dentro del proyecto de Django.

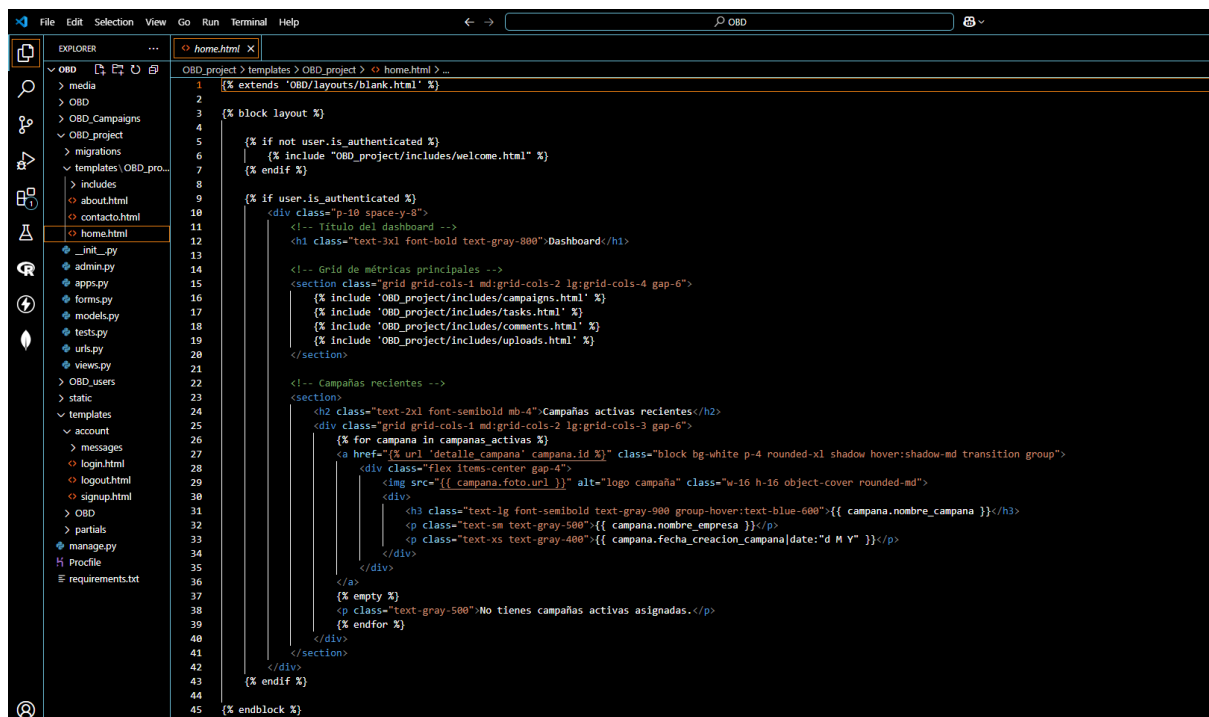
Al inicio del backend, César implementó el sistema de autenticación con Django AllAuth y HTMX, mientras que yo me encargué de crear las vistas para el inicio de sesión y el menú principal, enfocándome en que fueran visualmente claras, funcionales y coherentes con los bosquejos que habíamos definido como equipo.



The screenshot shows the Visual Studio Code editor with the 'login.html' file open. The Explorer sidebar on the left shows the project structure with 'templates' expanded. The main editor area displays the following code:

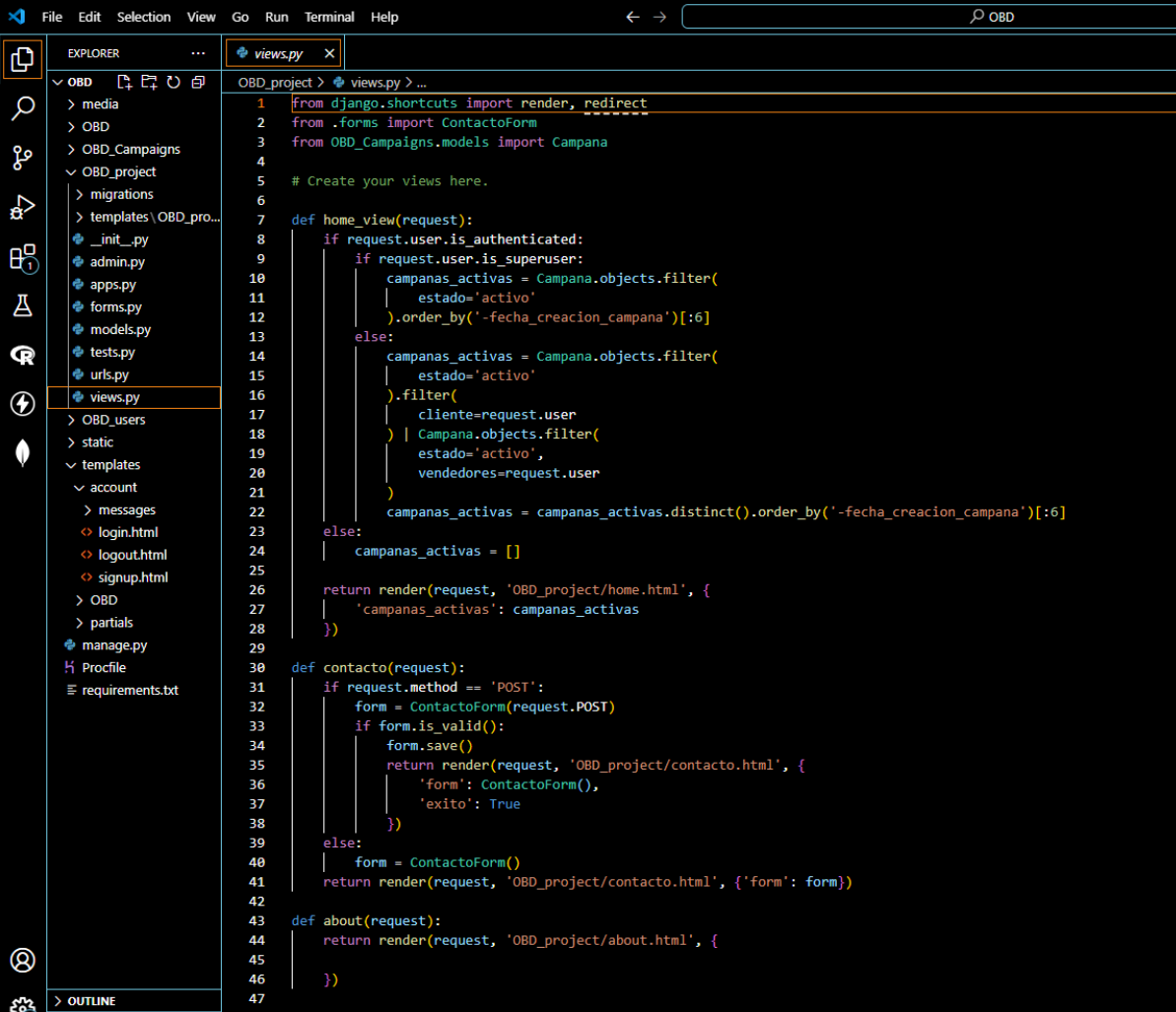
```
1 {% extends "OBD/allauth/layouts/base.html" %}
2
3 {% block content %}
4
5     <h1 class="text-2xl font-bold mb-4">Iniciar sesión</h1>
6
7     <form method="post" action="{% url 'account_login' %}" class="space-y-4">
8         {{ csrf_token %}}
9         {{ form.as_p }}
10
11         <button type="submit" class="bg-blue-500 hover:bg-blue-600 text-white px-4 py-2 rounded">
12             Iniciar sesión
13         </button>
14     </form>
15
16     <p class="mt-4 text-sm text-gray-600">
17         ¿No tienes una cuenta?
18         <a href="{% url 'account_signup' %}" class="text-blue-600 underline">Regístrate aquí</a>
19     </p>
20
21 {% endblock %}
22
```

También fui responsable del backend de las vistas básicas como home, contacto y about dentro de OBD_project. Posteriormente, César tomó estos bosquejos y desarrolló el frontend, siguiendo el estilo y la estructura que yo propuse desde el inicio del diseño.



The screenshot shows the Visual Studio Code editor with the 'home.html' file open. The Explorer sidebar on the left shows the project structure with 'templates' expanded. The main editor area displays the following code:

```
1 {% extends "OBD/layouts/blank.html" %}
2
3 {% block layout %}
4
5     {% if not user.is_authenticated %}
6         {% include "OBD_project/includes/welcome.html" %}
7     {% endif %}
8
9     {% if user.is_authenticated %}
10         <div class="p-10 space-y-8">
11             <!-- Título del dashboard -->
12             <h1 class="text-3xl font-bold text-gray-800">Dashboard</h1>
13
14             <!-- Grid de métricas principales -->
15             <section class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-6">
16                 {% include "OBD_project/includes/campaigns.html" %}
17                 {% include "OBD_project/includes/tasks.html" %}
18                 {% include "OBD_project/includes/comments.html" %}
19                 {% include "OBD_project/includes/uploads.html" %}
20             </section>
21
22             <!-- Campañas recientes -->
23             <section>
24                 <h2 class="text-2xl font-semibold mb-4">Campañas activas recientes</h2>
25                 <div class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-6">
26                     {% for campana in campanas_activas %}
27                         <a href="{% url 'detalle_campana' campana.id %}" class="block bg-white p-4 rounded-xl shadow hover:shadow-md transition group">
28                             <div class="flex items-center gap-4">
29                                 
30                                 <div>
31                                     <h3 class="text-lg font-semibold text-gray-900 group-hover:text-blue-600">{{ campana.nombre_campana }}</h3>
32                                     <p class="text-sm text-gray-500">{{ campana.nombre_empresa }}</p>
33                                     <p class="text-xs text-gray-400">{{ campana.fecha_creacion_campana|date:"d M Y" }}</p>
34                                 </div>
35                             </a>
36                             {% empty %}
37                                 <p class="text-gray-500">No tienes campañas activas asignadas.</p>
38                             {% endfor %}
39                         </div>
40                     </section>
41                 </div>
42             {% endif %}
43         </div>
44     {% endblock %}
45
```



```
1 from django.shortcuts import render, redirect
2 from .forms import ContactoForm
3 from OBD_Campaigns.models import Campana
4
5 # Create your views here.
6
7 def home_view(request):
8     if request.user.is_authenticated:
9         if request.user.is_superuser:
10             campanas_activas = Campana.objects.filter(
11                 | estado='activo'
12             ).order_by('-fecha_creacion_campana')[:6]
13         else:
14             campanas_activas = Campana.objects.filter(
15                 | estado='activo'
16             ).filter(
17                 | cliente=request.user
18             ) | Campana.objects.filter(
19                 | estado='activo',
20                 | vendedores=request.user
21             )
22             campanas_activas = campanas_activas.distinct().order_by('-fecha_creacion_campana')[:6]
23     else:
24         campanas_activas = []
25
26     return render(request, 'OBD_project/home.html', {
27         | 'campanas_activas': campanas_activas
28     })
29
30 def contacto(request):
31     if request.method == 'POST':
32         form = ContactoForm(request.POST)
33         if form.is_valid():
34             form.save()
35             return render(request, 'OBD_project/contacto.html', {
36                 | 'form': ContactoForm(),
37                 | 'exito': True
38             })
39     else:
40         form = ContactoForm()
41     return render(request, 'OBD_project/contacto.html', {'form': form})
42
43 def about(request):
44     return render(request, 'OBD_project/about.html', {
45
46     })
47
48
```

En la app OBD_campaigns, trabajé junto con Alex para desarrollar los modelos necesarios, cuidando que estuvieran alineados con la lógica del sistema. Ambos nos encargamos también del frontend de esta sección, construyéndolo con base en los bosquejos que diseñamos para visualizar los flujos de interacción del usuario. Mientras tanto, César se encargó del backend de campañas, conectando nuestras vistas y modelos.

