

# Reporte Técnico de Actividades Práctico-Experimentales Nro. 0XX

## 1. Datos de Identificación del Estudiante y la Práctica

Nombre del estudiante(s)	Luis Alberto Medina Chamba		
Asignatura	Software Engineering Models		
Ciclo	7 A		
Unidad	2		
Resultado de aprendizaje de la unidad	R2. Describe los tipos de implementación, bajo los solidaridad, transparencia, honestidad.		
Título de la Práctica	Aplicación del Enfoque Dirigido MDD/MDA en Proyectos de Software		
Nombre del Docente	Wilman Patricio Chamba Zaragocín		
Fecha	04 de diciembre, 2025		
Horario	04 de diciembre, 2025		
Lugar	Aula		
Tiempo planificado en el Sílabo	4 horas		
Organización colaborativa	Individual / Equipo	Nro:	1

## 2. Objetivo(s) de la Práctica

- Comprender los principios de MDA (Model-Driven Architecture) y sus niveles de abstracción (CIM, PIM, PSM, IM).
- Aplicar transformaciones de modelos para generar código específico de plataforma a partir de modelos independientes.
- Validar la coherencia entre los diferentes niveles de modelos y su implementación final.
- Experimentar con el enfoque MDD en un proyecto integrador autónomo que demuestre el ciclo completo de transformación.

## 3. Materiales, Reactivos, Equipos y Herramientas

- Documentación: Manuales de herramientas, Guías de notación UML, Especificaciones MDA.
- Repositorios de Ejemplo: Proyectos de referencia disponibles en EVA Documentación: Guía rápida de MDA, Especificaciones UML, Material bibliográfico del EVA.
- Recursos Adicionales: Plantillas para documentación de modelos, ejemplos de transformaciones.



- Plataformas Tecnológicas: Java + Spring Boot, Python + Django, o cualquier stack tecnológico seleccionado.

## 4. Procedimiento / Metodología Ejecutada

Para el desarrollo de la práctica se aplicó la metodología MDA (Model Driven Architecture) siguiendo un enfoque de ingeniería de software basado en transformaciones de modelos progresivas, dividiendo el procedimiento en cuatro fases secuenciales. Inicialmente, en la fase CIM, se realizó el análisis de requisitos y el diseño del Diagrama de Casos de Uso en StarUML, definiendo los flujos para la gestión documental; seguido por la fase PIM, donde se elaboró el Diagrama de Clases del Dominio con tipos de datos abstractos para garantizar la independencia tecnológica. Posteriormente, en la fase PSM, se transformó el modelo hacia una arquitectura técnica basada en el framework Django, incorporando tipos específicos del ORM y diseñando los diagramas de Paquetes, Componentes y Despliegue bajo el patrón MVT. Finalmente, en la fase IM, se procedió a la generación automática de código fuente Python mediante StarUML, complementada con una refactorización manual para ajustes técnicos y la implementación de la lógica de negocio con la librería pypdf, culminando con el versionamiento del proyecto en GitHub y su validación funcional en un servidor local.

## 5. Resultados

### Modelo CIM (Computational Independent Model):

#### 1. Especificación de requisitos

Actores

Actor	Descripción	Responsabilidades
Usuario Operador	Persona encargada de ejecutar el prototipo, gestionar los documentos de entrada y salida, y verificar el correcto funcionamiento.	<ul style="list-style-type: none"><li>- Seleccionar los documentos de entrada.</li><li>- Subir documentos PDF.</li><li>- Ejecutar el proceso de conversión.</li><li>- Descargar los archivos convertidos.</li></ul>

#### Requisitos Funcionales

Número de requisito	RF01
Nombre de requisito	Cargar documento PDF
Descripción del Requerimiento	El sistema permitirá al usuario operador subir un archivo PDF como entrada para el proceso de conversión.
Prioridad del requisito	Alta

Número de requisito	RF02
---------------------	------

<b>Nombre de requisito</b>	Convertir a versión HTML accesible
<b>Descripción del Requerimiento</b>	El sistema debe convertir el documento PDF a una versión en formato HTML accesible, aplicando etiquetas semánticas conforme a los estándares WCAG 2.1 y PDF/UA
<b>Prioridad del requisito</b>	Alta

<b>Número de requisito</b>	RF03
<b>Nombre de requisito</b>	Convertir a versión EPUB accesible
<b>Descripción del Requerimiento</b>	El sistema debe permitir al usuario operador convertir el contenido del PDF a formato EPUB básico, manteniendo la estructura semántica y las etiquetas de accesibilidad.
<b>Prioridad del requisito</b>	Alta

<b>Número de requisito</b>	RF04
<b>Nombre de requisito</b>	Validar accesibilidad del contenido
<b>Descripción del Requerimiento</b>	El sistema debe permitir realizar pruebas automáticas de accesibilidad a los archivos convertidos utilizando herramientas externas (WAVE, EPUBCheck).
<b>Prioridad del requisito</b>	Alta

<b>Número de requisito</b>	RF05
<b>Nombre de requisito</b>	Descargar archivos generados
<b>Descripción del Requerimiento</b>	El sistema debe permitir al usuario operador descargar el archivo accesible generado en el formato previamente seleccionado (página HTML o EPUB).
<b>Prioridad del requisito</b>	Alta

<b>Número de requisito</b>	RF06
<b>Nombre de requisito</b>	Generar reportes de conversión
<b>Descripción del Requerimiento</b>	El sistema debe permitir al usuario operador generar un reporte sobre el estado de cada etapa indicando inconvenientes o advertencias.
<b>Prioridad del requisito</b>	Media

### **Requisitos no funcionales**

Usabilidad:



<b>Número de requisito</b>	RNF01
<b>Nombre de requisito</b>	Consistencia visual
<b>Descripción del Requerimiento</b>	El sistema debe tener una coherencia del diseño en todas las pantallas, siguiendo un mismo estilo gráfico en botones, colores y etiquetas.
<b>Prioridad del requisito</b>	Alta

<b>Número de requisito</b>	RNF02
<b>Nombre de requisito</b>	Frecuencia de clics menor a 3
<b>Descripción del Requerimiento</b>	El sistema debe tener una frecuencia de clics menor de 3 a 5 en el 80% de las funcionalidades
<b>Prioridad del requisito</b>	Alta

Compatibilidad:

<b>Número de requisito</b>	RNF03
<b>Nombre de requisito</b>	Versión aceptada de documentos PDF
<b>Descripción del Requerimiento</b>	El sistema debe ser compatible con archivos PDF creados bajo la especificación PDF 2.0 (ISO 32000-2).
<b>Prioridad del requisito</b>	Alta

Seguridad de los datos:

<b>Número de requisito</b>	RNF04
<b>Nombre de requisito</b>	Seguridad de los datos
<b>Descripción del Requerimiento</b>	El sistema debe eliminar automáticamente los archivos subidos por el usuario, no deben cargarse permanente en el servidor.
<b>Prioridad del requisito</b>	Alta

Rendimiento:

<b>Número de requisito</b>	RNF05
<b>Nombre de requisito</b>	Rendimiento en la conversión
<b>Descripción del Requerimiento</b>	El sistema debe convertir documentos PDF a páginas HTML o EPUB en un tiempo máximo de 3 minutos para archivo de hasta 5 MB de tamaño. Para archivos que superen el límite, el tiempo de conversión aumentara proporcionalmente al tamaño

		del documento.
<b>Prioridad requisito</b>	<b>del</b>	Alta

## 2. Diagrama de casos de uso del sistema

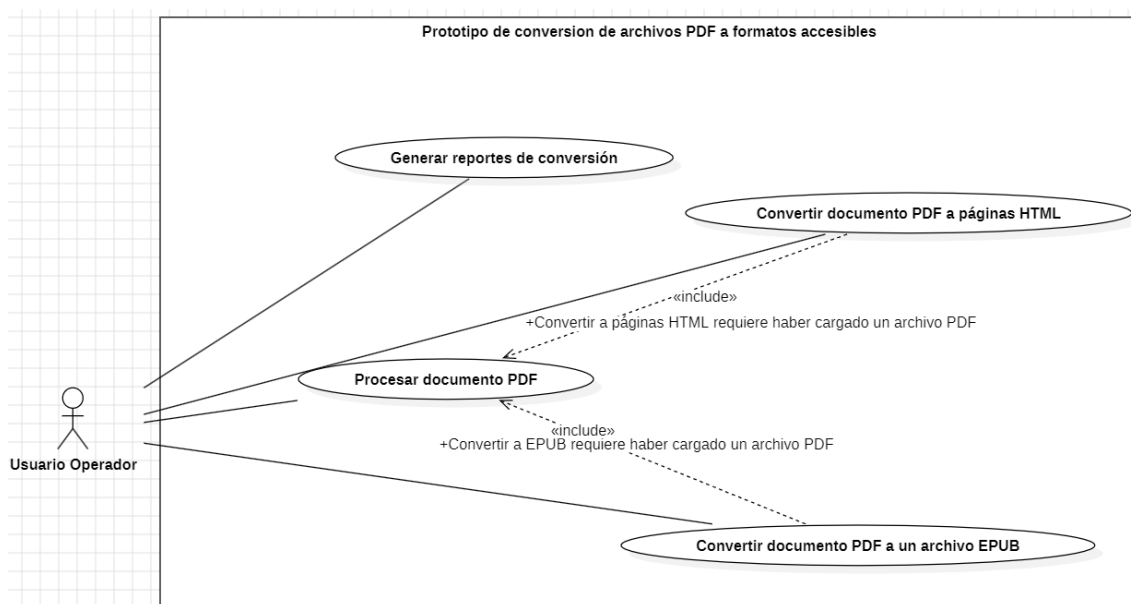


Figura 1. Diagrama de casos de uso.

## 3. Modelo conceptual del dominio

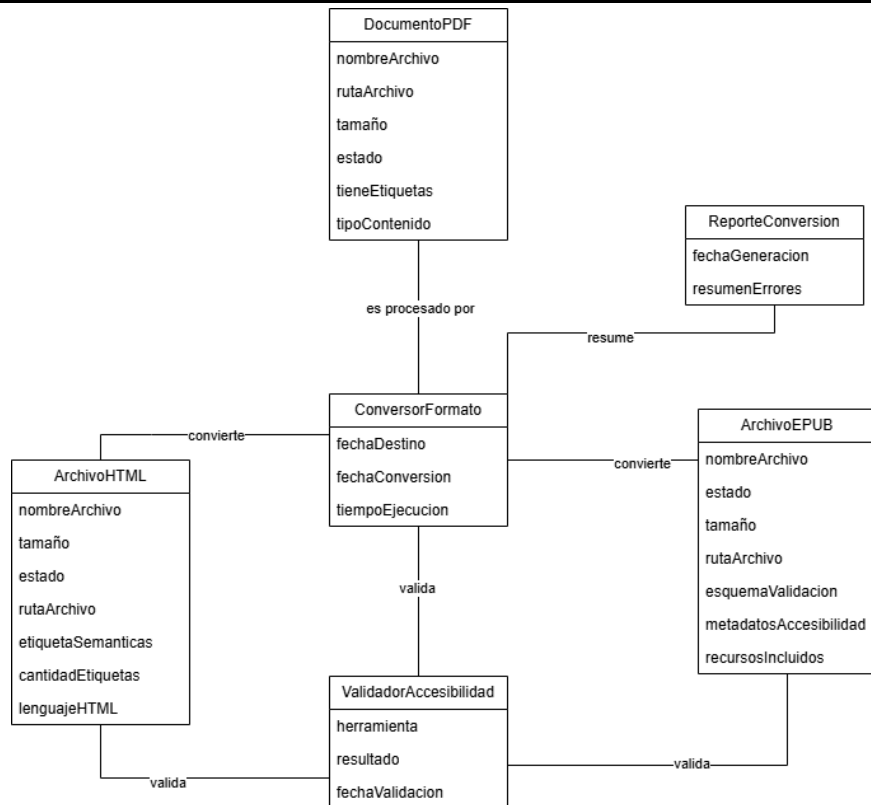


Figura 2. Modelo conceptual del dominio.

## Modelo PIM (Platform Independent Model):

### 1. Diagrama de clases del dominio de la aplicación.

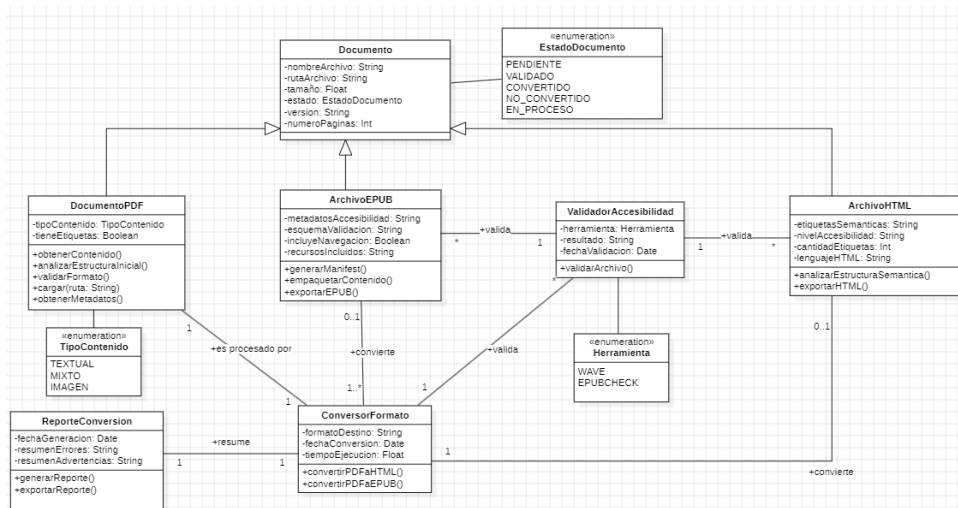


Figura 3. Diagrama de clases

### 2. Descripción detallada de al menos 2 casos de uso / historias de usuario.

Nro	1
-----	---

<b>Caso de uso</b>	Procesar documento PDF
<b>Actor principal</b>	Usuario Operador
<b>Descripción:</b>	El usuario carga un archivo PDF para su procesamiento con el fin de extraer el contenido y preparar la conversión a formatos accesibles (páginas HTML y EPUB).
<b>Precondiciones:</b>	<ul style="list-style-type: none"> <li>- El prototipo esté en funcionamiento.</li> <li>- El usuario ha accedido a la interfaz del prototipo.</li> </ul>
<b>Postcondiciones:</b>	<ul style="list-style-type: none"> <li>- El archivo PDF ha sido procesado correctamente por el sistema.</li> </ul>
<b>Flujo básico</b>	<ol style="list-style-type: none"> <li>1. El usuario selecciona la opción Procesar documento PDF.</li> <li>2. El prototipo solicita que se cargue un archivo PDF.</li> <li>3. El usuario carga el archivo PDF deseado.</li> <li>4. El prototipo analiza el documento y extrae su contenido (texto, etiquetas, metadatos).</li> <li>5. El prototipo muestra un mensaje indicando que el procesamiento fue exitoso.</li> </ol>
<b>Flujos alternativos:</b>	<p>1' El usuario no selecciona ningún archivo: El prototipo muestra un mensaje de error y solicita nuevamente la carga del PDF.</p> <p>5' El archivo PDF está dañado o no cumple con el formato: El prototipo muestra un mensaje de error y detiene el proceso.</p>

Tabla 1. Caso de uso: Procesar Documento PDF

<b>Nro</b>	2
<b>Caso de uso</b>	Convertir documento PDF a páginas HTML
<b>Actor principal</b>	Usuario Operador
<b>Descripción:</b>	El prototipo convierte un documento PDF previamente procesado a formato HTML, cumpliendo con los estándares WCAG 2.1.
<b>Precondiciones:</b>	<ul style="list-style-type: none"> <li>- Debe haberse ejecutado correctamente el caso Procesar documento PDF.</li> </ul>
<b>Postcondiciones:</b>	<ul style="list-style-type: none"> <li>- El prototipo ha convertido el contenido del PDF procesado a una o más páginas HTML accesibles.</li> </ul>
<b>Flujo básico</b>	<ol style="list-style-type: none"> <li>1. El usuario selecciona la opción Convertir documento PDF a versión HTML accesible.</li> <li>2. El prototipo verifica que exista un documento PDF</li> </ol>

	<p>previamente procesado.</p> <ol style="list-style-type: none"> <li>El prototipo convierte el contenido del PDF (texto, imágenes, tablas y estructura) al formato páginas HTML, aplicando etiquetas semánticas y atributos ARIA.</li> <li>El prototipo completa la conversión.</li> <li>El prototipo muestra un mensaje de confirmación con la ubicación del archivo generado.</li> </ol>
<b>Flujos alternativos:</b>	<p>2' No existe un PDF procesado: El prototipo muestra un mensaje indicando que se debe procesar un documento antes.</p> <p>4' Se produce un error durante la conversión: El prototipo notifica al usuario y registra el error para revisión técnica.</p>

Tabla 2. Caso de uso: Convertir documento PDF a páginas HTML

## Modelo PSM (Platform Specific Model):

### 1. Diagrama de clases técnico con anotaciones de plataforma.

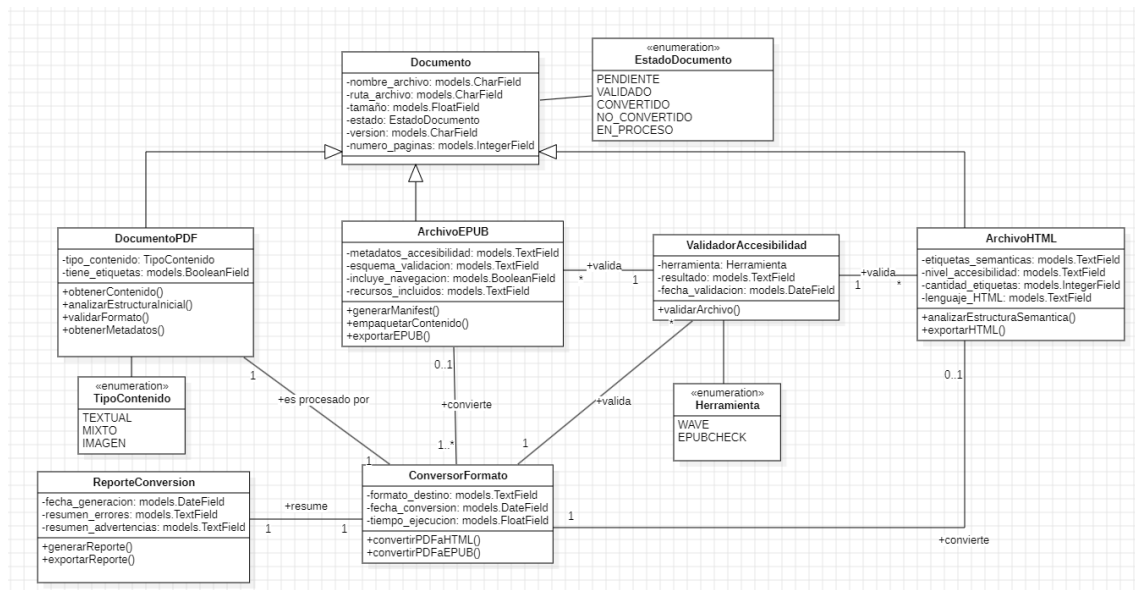


Figura 4. Diagrama de clases técnico

### 2. Diagrama de paquetes con organización arquitectónica



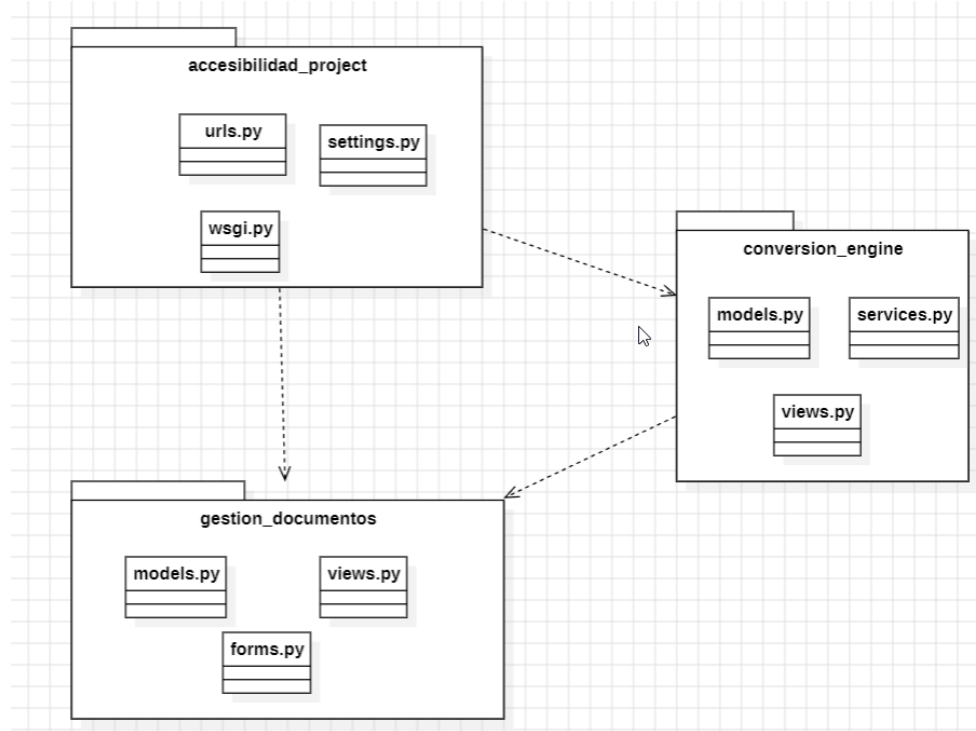


Figura 5. Diagrama de paquetes

### 3. Diagrama de componentes y de despliegue.

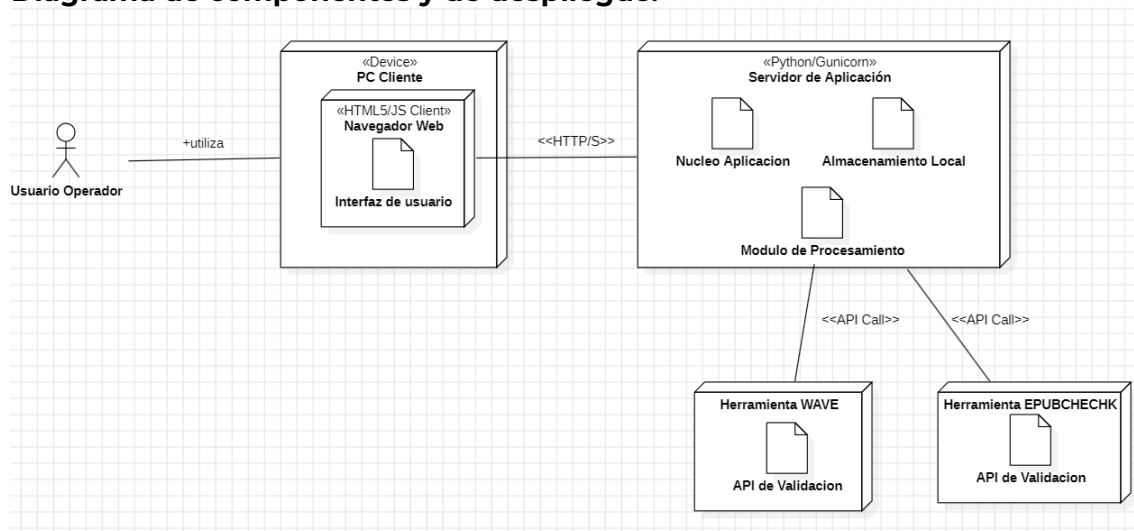


Figura 6. Diagrama de componentes

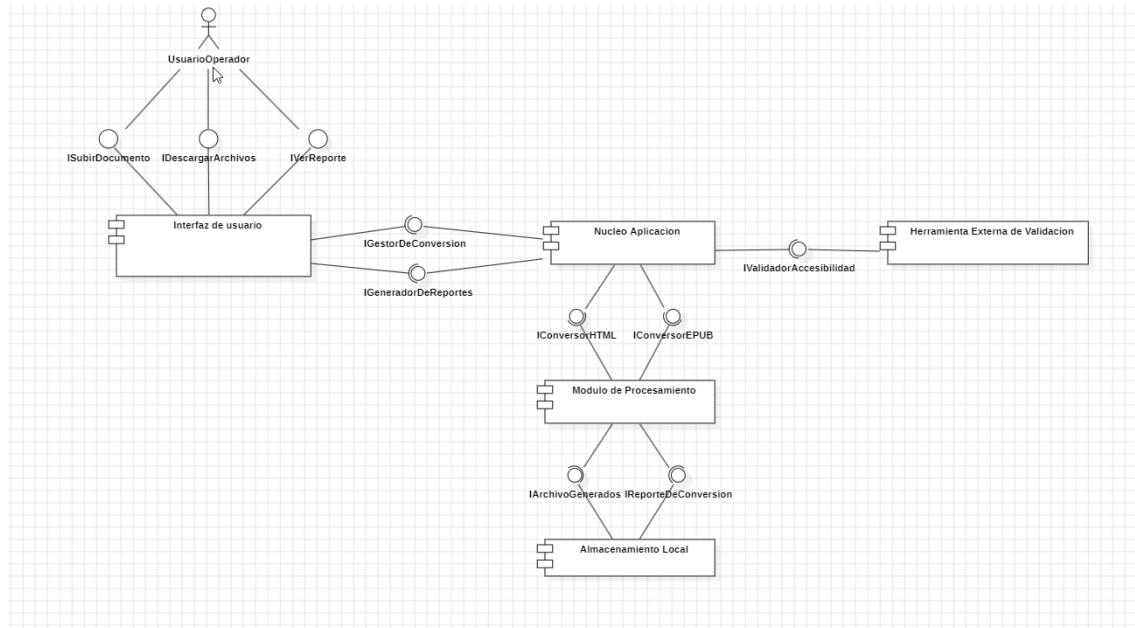


Figura 7. Diagrama de componentes

## Modelo IM (Implementation Model):

### 1. Código base generado automáticamente

Nombre	Fecha de modificación	Tipo	Tamaño
Al principio de esta semana			
archivoepub	9/12/2025 12:32	Archivo de origen ...	1 KB
archivohtml	9/12/2025 12:32	Archivo de origen ...	1 KB
conversorformato	9/12/2025 12:32	Archivo de origen ...	1 KB
documento	9/12/2025 12:32	Archivo de origen ...	1 KB
documentopdf	9/12/2025 12:32	Archivo de origen ...	1 KB
EstadoDocumento	9/12/2025 12:32	Archivo de origen ...	1 KB
Herramienta	9/12/2025 12:32	Archivo de origen ...	1 KB
reporteconversion	9/12/2025 12:32	Archivo de origen ...	1 KB
TipoContenido	9/12/2025 12:32	Archivo de origen ...	1 KB
validadoraccesibilidad	9/12/2025 12:32	Archivo de origen ...	1 KB

Figura 8. Estructura de archivos generados con StarUML

## Repositorio de GitHub:

<https://github.com/Medina23334444/StarUML-PDF>

### 2. Implementaciones manuales de funcionalidades clave

Funcionalidad Caso de Uso: Procesar Documentos

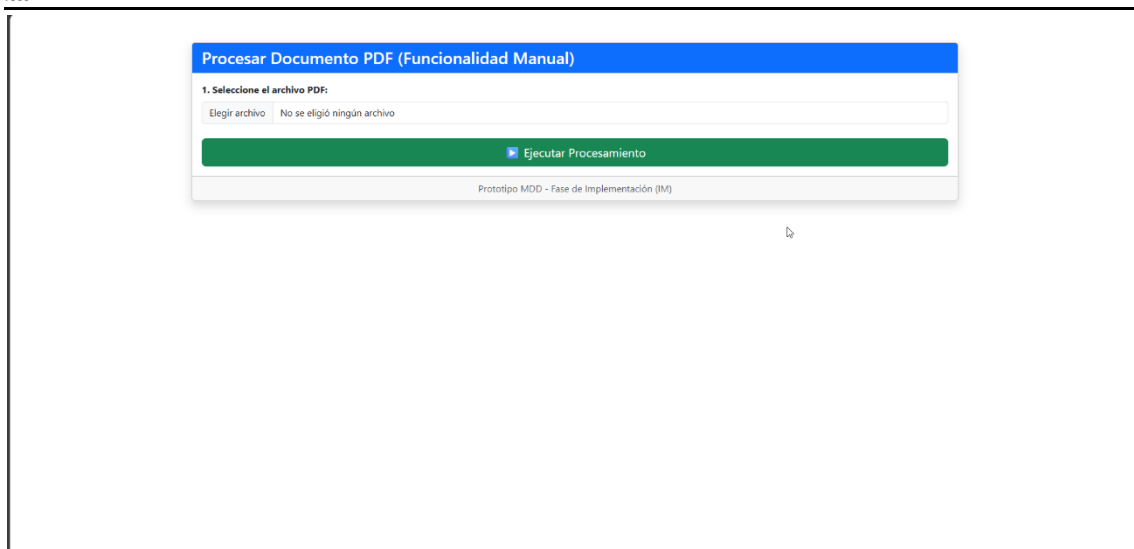


Figura 9. Implementación de la funcionalidad Procesar Documentos

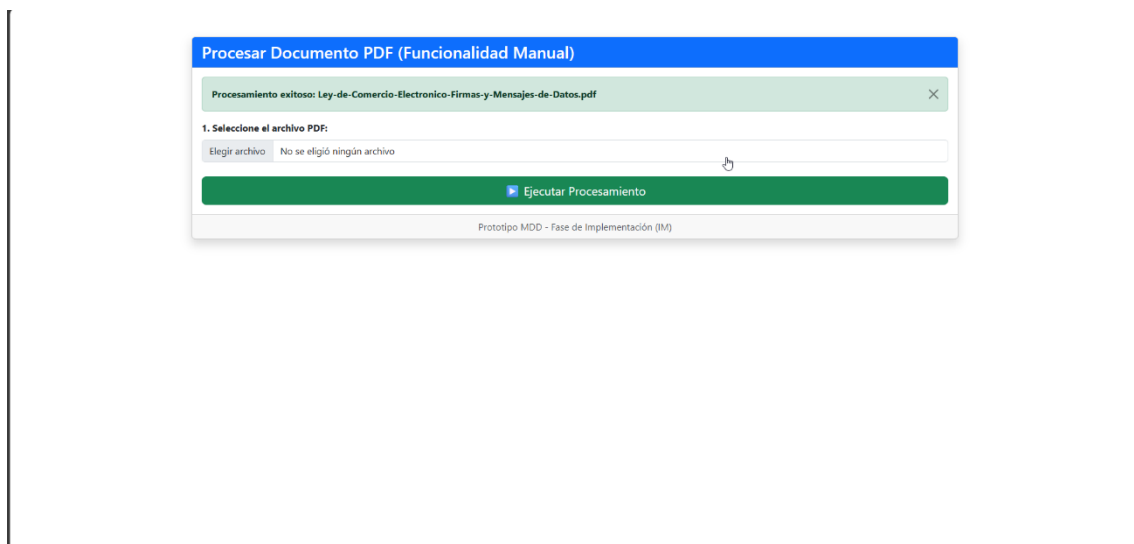


Figura 10. Implementación con éxito de la funcionalidad

### Repositorio de GitHub:

<https://github.com/Medina23334444/Funcionalidad-ProcesarPDF>

## 6. Preguntas de Control

- ¿Qué ventajas identificó en la transformación progresiva CIM->PIM->PSM->IM frente al desarrollo tradicional?

La principal ventaja es que el MDD permite organizar y pensar el sistema antes de programar. Mientras que en el desarrollo tradicional se suele pasar directo al código y la arquitectura se desordena, con MDD el CIM ayuda a aclarar qué necesita el usuario, el PIM define la lógica de negocio de forma clara y, al llegar al IM, herramientas como StarUML generan la estructura base (modelos, clases y relaciones). Esto reduce código repetitivo, ahorra tiempo y asegura que la implementación en las tecnologías siga el diseño previsto.

- ¿Cómo manejan las herramientas MDA la consistencia entre los diferentes niveles de modelos durante las transformaciones?  
La consistencia se asegura mediante la trazabilidad entre modelos. En la práctica, al transformar el PIM al PSM, los nombres de las clases se conservaron tal cual, los tipos de datos abstractos (como String) se convirtieron en tipos concretos de Django (como models.CharField) y las relaciones de herencia o dependencia se reflejaron en importaciones. Así, el código terminó siendo una traducción fiel del modelo de dominio.
- ¿Qué desafíos técnicos enfrentó durante las transformaciones de modelos cómo los resolvió?  
Un desafío técnico fue que el generador de código de StarUML producía las clases de Django con los atributos marcados como *undefined*, por lo que fue necesario ajustar el code generator para que reconociera y asignara correctamente esos atributos.

## 7. Conclusiones

El uso de herramientas como StarUML resultó útil para generar la estructura inicial del proyecto y establecer la base del diseño en Django. Sin embargo, la experiencia mostró que la intervención manual sigue siendo indispensable, especialmente porque algunos elementos, como los enums donde no se generaban en el código y requerían ajustes manuales por parte del desarrollador.

La implementación manual de la funcionalidad Procesar Documento PDF permitió confirmar la viabilidad técnica del modelo. Al llevarlo a la práctica, se comprobó que la estructura de clases definida en el PIM soportaba adecuadamente la funcionalidad requerida.

## 8. Recomendaciones

Se recomienda definir con mayor detalle los atributos en el diagrama PSM antes de generar el código. Esto minimizaría los errores de migración en lenguajes o frameworks estrictos como Django.

Al utilizar extensiones de generación de código en StarUML, es recomendable revisar el código generado para asegurarse de que el modelo resultante sea correcto, incluyendo la definición de atributos, los nombres de las clases y la estructura general.

## 9. Anexos

### Anexo 1: Repositorio del Código Base Generado (Modelo IM)

Link: <https://github.com/Medina23334444/StarUML-PDF>

### Anexo 2: Repositorio de la Implementación Funcional (Prototipo Django)

Link: <https://github.com/Medina23334444/Funcionalidad-ProcesarPDF>

### Declaración de IA

Parte del contenido de este documento fue asistido por ChatGPT para apoyar la redacción, mejorar la claridad del texto y optimizar la presentación de ideas. Todas las decisiones conceptuales, técnicas y académicas fueron revisadas, validadas y ajustadas manualmente por el autor, garantizando la precisión y coherencia del trabajo final.