Vicerrectoría Académica
Dirección de Servicios Académicos
Subdirección de Servicios a Escuelas

[Aplicación web “Focusa”]

(DAS) Documento Arquitectura Sistema
Versión 1.9

Identificación de Documento

Identificación	
Proyecto	Aplicación web “Focusa”
Versión	1.9

Documento mantenido por	Carla Adasme y Evelyn Ramos
Fecha de última revisión	05/11/2025
Fecha de próxima revisión	22/11/2025

Documento aprobado por	Stephen Lucic · Líder Técnico Proyecto Focusa
Fecha de última aprobación	22/11/2025

Historia de Revisiones

Fecha	Versión	Descripción	Autor
14/10/2025	1.0	Inicio del documento y definición de estructura base del DAS	Evelyn Ramos
16/10/2025	1.1	Incorporación de diagramas VISTAS 4+1 (alto nivel)	Stephen Lucic
19/10/2025	1.2	Desarrollo de Tablas ER, Diagrama de Clases y Casos de Uso	Carla Adasme
22/10/2025	1.3	Inclusión de flujos principales: Registro, Login, Creación de Tareas	Evelyn Ramos
26/10/2025	1.4	Integración del EDT, Flujo de Caja y primeros ajustes de arquitectura	Carla Adasme
30/10/2025	1.5	Actualización de la arquitectura AWS y estructura de despliegue	Stephen Lucic
05/11/2025	1.6	Incorporación de Casos de Uso Extendidos y Vista Front-End	Evelyn Ramos
15/11/2025	1.7	Mejoras visuales, revisión de dashboards y ajuste completo del módulo Kanban	Evelyn Ramos
20/11/2025	1.8	Revisión técnica de roles, permisos, endpoints y validaciones del sistema	Carla Adasme
22/11/2025	1.9	Revisión final del documento y cierre de la versión DAS para entrega	Evelyn Ramos

Tabla de Contenidos

1 Introducción

1.1 Contexto del Problema

El equipo de desarrollo del proyecto "Focusa" identificó la necesidad de una solución digital que permita gestionar tareas, actividades y productividad de manera centralizada. Actualmente, estudiantes, profesionales y equipos pequeños dependen de herramientas dispersas como notas, hojas de cálculo o aplicaciones no integradas. Esto genera:

- Falta de visibilidad de avances.
- Problemas en la priorización de actividades.
- Dificultad para hacer seguimiento a responsabilidades.
- Carencia de indicadores claros de productividad.
- Escasa trazabilidad de acciones y estados de tareas.

"Focusa" nace como una plataforma web desarrollada en Django que centraliza la gestión de tareas, permitiendo a los usuarios organizar su carga diaria mediante vistas de lista, tablero kanban, dashboard y herramientas de clasificación.

1.2 Propósito

El propósito de este documento es definir la arquitectura de software de la aplicación web "Focusa", describiendo sus componentes estructurales, tecnológicos y de interacción. El objetivo es asegurar que las decisiones de diseño:

- Cumplen con los requerimientos funcionales y no funcionales establecidos.
- Favorecen la escalabilidad y mantenibilidad de la aplicación web.
- Permiten una experiencia de usuario fluida y segura.
- Aseguran el soporte para el crecimiento futuro del proyecto.

Este documento constituye la base técnica para el desarrollo, validación y despliegue de la aplicación.

1.3 Ámbito

La aplicación web “Focusa” contempla las siguientes funcionalidades principales:

- Registro, autenticación y gestión de usuarios.
- Creación, edición, visualización y eliminación de tareas (CRUD).
- Gestión de estados (pendiente, en progreso, completada).
- Asignación de fechas límite y recordatorios.
- Clasificación de tareas mediante categorías o etiquetas.
- Visualización de tablero (Kanban) para organización por estados.
- Dashboard de productividad con métricas e indicadores.
- Búsqueda avanzada y filtrado por fechas, estado, prioridad o categoría.
- Adjuntar archivos o enlaces a una tarea.
- Compatibilidad con navegadores modernos.
- Módulo de respaldo manual de la base de datos (DBA).
- Panel administrativo básico para gestión interna.

1.4 Definiciones, acrónimos y abreviaciones

ACRONIMO	DESCRIPCION
MVC / MVT	Modelo-Vista-Controlador / Modelo-Vista-Template (Django)
CRUD	Create, Read, Update, Delete
ORM	Object Relational Mapping (mapeo objeto-relacional)
UI/UX	Interfaz de Usuario / Experiencia de Usuario
API	Application Programming Interface
DBA	Database Administrator
SLA	Service Level Agreement

1.5 Referencias

Este documento se complementa con los siguientes entregables del proyecto:

- ERS – Especificación de Requerimientos de la aplicación web
- Documento de Casos de Uso (Alto Nivel y Extendidos)
- Planilla de Requerimientos Funcionales y No Funcionales
- Matriz RACI del Proyecto “Focusa”
- Matriz de Riesgos del Proyecto
- Backlog y planificación del proyecto (Sprints)

1.6 Resumen ejecutivo

La aplicación web “Focusa” corresponde a una plataforma web diseñada para optimizar la gestión de tareas y la productividad personal o colaborativa, integrando herramientas de organización, visualización y seguimiento dentro de un entorno unificado y moderno.

El proyecto se desarrolla utilizando Python/Django, acompañado de una arquitectura estructurada bajo el enfoque 4+1, con énfasis en modularidad, mantenibilidad y escalabilidad. La aplicación se sustenta sobre una infraestructura técnica que incorpora una arquitectura DVB (Desarrollo – Base de Datos – Backend), implementada mediante contenedores Docker, lo que garantiza entornos reproducibles, desacoplados y fácilmente desplegables.

La base de datos de la aplicación web utiliza MySQL, administrada desde un contenedor aislado, lo que aporta robustez, consistencia transaccional y mayor rendimiento frente a soluciones de almacenamiento más simples. El backend Django se ejecuta igualmente en un contenedor independiente, estableciendo comunicación interna eficiente a través del ORM.

En términos funcionales, “Focusa” integra características claves tales como:

- Registro, autenticación y gestión de usuarios.
- Creación, modificación y seguimiento del ciclo de vida de tareas.
- Organización mediante categorías y estados.
- Visualización en tablero tipo Kanban.
- Dashboard con métricas de productividad utilizando componentes gráficos

interactivos.

- Búsqueda avanzada, filtrado por fechas y gestión de adjuntos.

- Respaldo manual de la base de datos y compatibilidad con navegadores modernos.

El diseño de la aplicación web prioriza la seguridad, la usabilidad, la trazabilidad de acciones, y la eficiencia operativa, apoyándose en buenas prácticas de UI/UX y pruebas de compatibilidad ejecutadas por el equipo QA. Adicionalmente, la arquitectura se mantiene preparada para un crecimiento futuro, permitiendo incorporar nuevos servicios, mejorar el pipeline de despliegue e integrar servicios externos según las necesidades del proyecto.

1.7 Representación

La arquitectura de la aplicación web “Focusa” está representada siguiendo el enfoque del framework 4+1 y las recomendaciones del Proceso Unificado, lo que permite estructurar la solución desde múltiples perspectivas para garantizar su calidad técnica y alineación con los requerimientos funcionales y no funcionales.

Las vistas consideradas en esta versión del documento son las siguientes:

- **Vista de Escenarios**: La vista de escenarios describe los comportamientos más relevantes de la aplicación web “Focusa” a través de sus casos de uso principales, permitiendo entender cómo interactúan los usuarios con las funcionalidades esenciales, como la gestión de tareas, autenticación, visualización del tablero y generación de métricas. Esta vista establece el contexto funcional y sirve como hilo conductor del resto de las vistas arquitectónicas, representando situaciones clave tanto funcionales como de calidad, tales como seguridad, rendimiento y usabilidad.

- **Vista de Procesos**: La vista de procesos representa el flujo interno de ejecución de la aplicación web, describiendo cómo se gestionan las solicitudes entre el frontend, el backend Django y la base de datos MySQL dentro del entorno Docker. Incluye validaciones, operaciones CRUD, sincronización entre módulos, comunicación mediante el ORM y el procesamiento de elementos como filtros, adjuntos y recordatorios, mostrando cómo todos los componentes colaboran para mantener la coherencia y el rendimiento de la aplicación web.

- **Vista Lógica**: La vista lógica organiza la solución en módulos y componentes internos de Django, detallando la estructura de la aplicación web mediante apps como autenticación, tareas, tablero Kanban, dashboard, categorías y administración. Define las responsabilidades, relaciones y dependencias entre cada módulo, asegurando cohesión interna, bajo acoplamiento y un diseño estructurado que facilita la mantenibilidad y escalabilidad del proyecto ”Focusa”.

- **Vista de Desarrollo o Despliegue**: La vista de desarrollo describe los componentes

concretos que conforman la implementación de la aplicación web, incluyendo el repositorio del proyecto, la estructura del código, el uso de contenedores Docker para backend y base de datos, el archivo docker-compose.yml y el control de versiones con Git. Esta vista evidencia cómo se construye, empaqueta y despliega la aplicación, manteniendo entornos reproducibles y consistentes entre desarrollo y producción.

- **Vista Fisica**: La vista física especifica la infraestructura donde se ejecuta ”Focusa”, destacando el uso de Docker para encapsular servicios, la base de datos MySQL con volúmenes persistentes y el backend Django dentro de contenedores independientes que se comunican mediante redes internas. Además, considera restricciones tecnológicas, normas de seguridad, disponibilidad y preparación para despliegues en entornos cloud, asegurando la estabilidad operativa de la aplicación web.

2 Metas y Restricciones de la Arquitectura

2.1 Metas de la arquitectura

Atributo de calidad	Descripción
Desempeño	Garantizar tiempos de respuesta óptimos en operaciones CRUD, carga del tablero, dashboard y filtrados en tiempo real, incluso con múltiples usuarios conectados.
Escalabilidad	Posibilitar la integración futura de módulos como API móvil, reportes avanzados, analítica y nuevas funcionalidades propias de un SaaS.
Seguridad	Implementar control de acceso por roles, autenticación segura, hashing robusto, uso de HTTPS (en despliegues cloud) y manejo adecuado de datos personales.
Mantenibilidad	Mantener una arquitectura modular basada en Django, utilizando el ORM, apps independientes y plantillas reutilizables para facilitar ampliaciones y mantenimiento.
Disponibilidad	Asegurar que el servicio esté disponible de manera continua mediante respaldos programados, logs de errores y estabilidad provista por Docker.
Usabilidad	Ofrecer una interfaz intuitiva, moderna y responsiva, basada en buenas prácticas de UI/UX y compatible con navegadores modernos.

2.2 Restricciones de la Arquitectura

Durante la definición del proyecto se identificaron las siguientes restricciones que

influyen en el diseño arquitectónico de Focusa:

- Tiempo de construcción: El desarrollo debe realizarse en un plazo máximo de 12 semanas, lo que obliga a priorizar funcionalidades esenciales.
- Infraestructura tecnológica obligatoria:
  - Backend en Python 3.12 + Django 5.0
  - Base de datos en MySQL 8.0
  - Contenedores Docker para Backend y Base de Datos
- Alcance funcional inicial: Se prioriza la versión inicial del sistema con autenticación, gestión de tareas, tablero Kanban, dashboard y funcionalidades básicas.

2.3 Otros antecedentes y consideraciones

La aplicación web Focusa, como solución SaaS, se apoya en el framework Django, el cual proporciona elementos clave para su funcionamiento:

- ORM robusto para interactuar eficazmente con MySQL.
- Sistema de autenticación nativo, integrado y extensible.
- Arquitectura modular, basada en apps separadas por responsabilidad.
- Integración natural con entornos Docker, permitiendo entornos replicables y consistentes.
- Compatibilidad con estándares web, facilitando la implementación de interfaces responsivas y dashboards interactivos.

Estas características proporcionan a Focusa una base sólida, flexible y preparada para evolucionar de acuerdo con las necesidades de sus usuarios.

3 Vista de Escenarios

La Vista de Escenarios describe en detalle los principales escenarios funcionales y no funcionales que fueron priorizados durante el análisis del sistema Focusa. Esta vista presenta los casos de uso más relevantes, los actores que intervienen y las situaciones en las que el sistema debe responder a atributos de calidad importantes, tales como seguridad, rendimiento, disponibilidad y usabilidad. A través de los diagramas de casos de uso, se representan las interacciones clave entre los usuarios y la plataforma, permitiendo visualizar el comportamiento general del sistema y las funcionalidades esenciales que soportan la operación del servicio SaaS.

3.1 Modelo de Casos de Uso

A continuación, se presentan las representaciones gráficas del modelo general de casos de uso correspondientes a Focusa. Estas ilustraciones permiten comprender de manera visual la interacción entre los actores y las principales funcionalidades del sistema.

Diagrama de Caso de Uso General de la aplicación web Focusa – Módulo Cliente

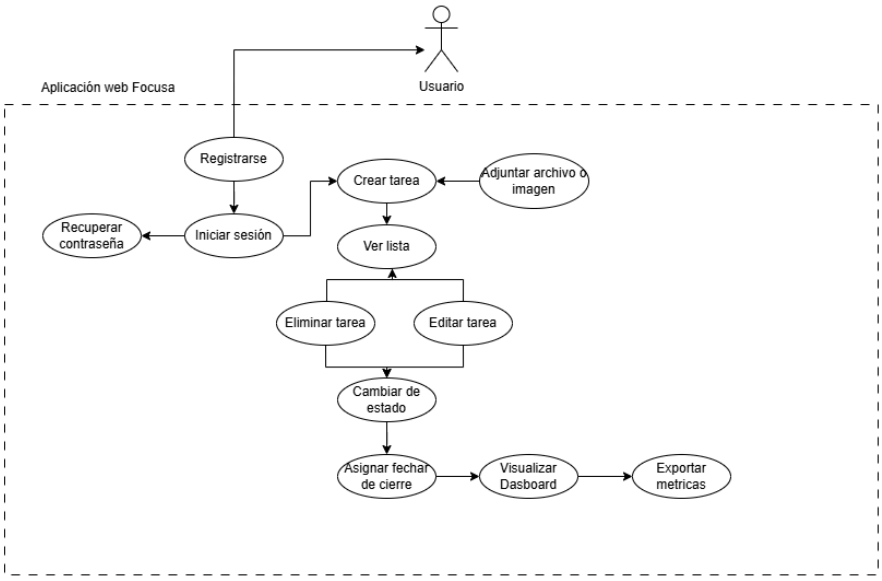
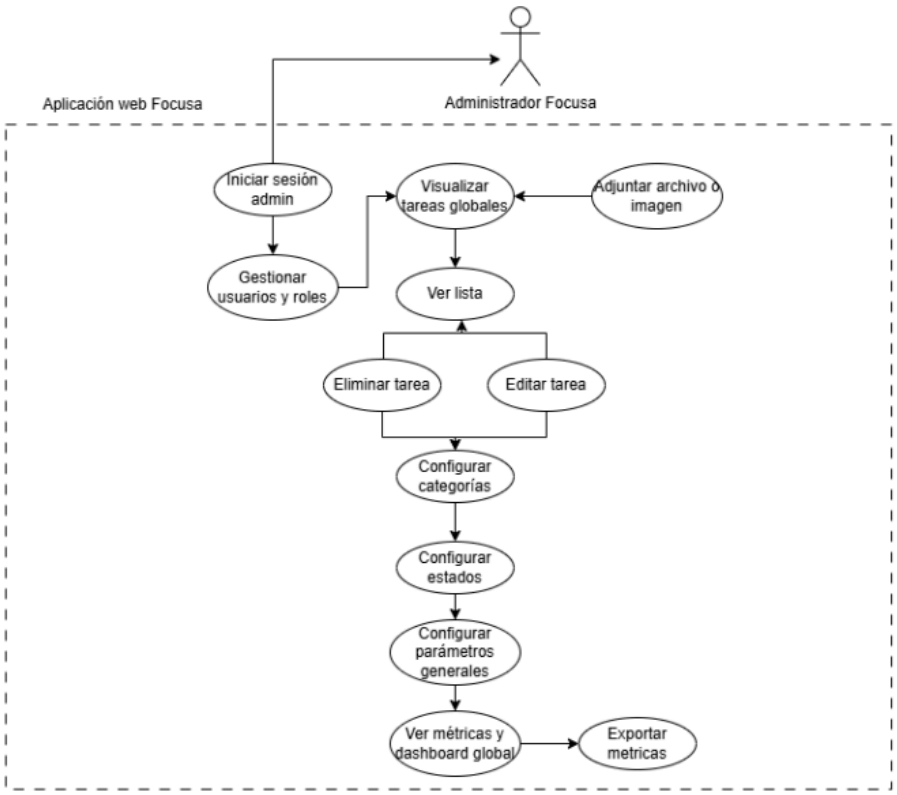


Diagrama de Caso de Uso del Módulo de Administrador de la aplicación web Focusa



3.2 Casos de Usos Extendidos

Los casos de uso considerados son los más relevantes para el desarrollo de la arquitectura. Se adjunta el documento o planilla caso uso.

Adjuntar la planilla caso uso extendido

Código	Nombre	Actores	Prioridad
CU-EX-001	Permitir registrar e iniciar sesión	Usuario, Sistema de Autenticación, Administrador	Alta
CU-EX-002	Permitir cerrar sesión de manera segura	Usuario, Sistema	Alta
CU-EX-003	Permitir crear, leer, actualizar y eliminar tareas (CRUD)	Usuario, Sistema	Alta
CU-EX-004	Permitir asignar fecha límite a una tarea	Usuario, Sistema	Alta
CU-EX-005	Permitir gestionar estados de una tarea	Usuario, Sistema	Alta
CU-EX-006	Permitir marcar tareas como completadas	Usuario, Sistema	Alta
CU-EX-007	Registrar automáticamente las fechas de creación y modificación	Usuario, Sistema	Media
CU-EX-008	Visualizar tareas en una lista ordenada	Usuario, Sistema	Alta
CU-EX-009	Implementar búsqueda y filtrado básico de tareas	Usuario, Sistema	Alta
CU-EX-010	Permitir recuperar la contraseña mediante correo electrónico	Usuario, Sistema de Autenticación	Alta

CU-EX-011	Gestionar roles y permisos avanzados de usuario	Administrador, Sistema	Alta
CU-EX-012	Integrar el sistema con fuentes de datos externas	Desarrollador Backend, Administrador BD, Sistema	Media
CU-EX-013	Visualizar gráficos interactivos avanzados	Usuario, Sistema de Reportes	Media
CU-EX-014	Garantizar disponibilidad del 99% con monitoreo y respaldos diarios	DevOps, Administrador, Sistema de Monitoreo	Alta
CU-EX-015	Permitir compartir tareas con otros usuarios	Usuario, Usuario Invitado, Sistema	Media
CU-EX-016	Permitir adjuntar archivos o enlaces a las tareas	Usuario, Sistema, Servicio de Almacenamiento	Media
CU-EX-017	Permitir filtrar tareas por rango de fechas	Usuario, Sistema	Media

3.3 Especificación de los Escenarios de Calidad Relevantes

Los siguientes escenarios de calidad fueron priorizados tras el análisis con los stakeholders del proyecto Focusa, considerando los atributos más críticos para una aplicación web SaaS: rendimiento, seguridad, disponibilidad, usabilidad y mantenibilidad.

Identificador: EC-01		
Escenario(s):		Varios usuarios crean, editan y completan tareas simultáneamente, y consultan el dashboard de productividad en horario de alta carga.
Atributos de Calidad relevantes:		Rendimiento / Desempeño – Escalabilidad
Componentes del Escenario	Estímulos:	Múltiples usuarios realizan operaciones de creación, actualización y filtrado de tareas, además de solicitar la carga del dashboard de productividad de forma concurrente.
	Fuente del estimulo	Usuarios finales conectados vía navegador web.
	Ambiente:	Entorno de producción de Focusa, en horario de uso intensivo (picos de jornada académica o laboral).
	Artefacto:	Módulo de tareas, módulo de filtros/búsqueda y módulo de dashboard dentro del backend Django, conectado a la base de datos MySQL en contenedor Docker.
	Respuesta:	La aplicación procesa las solicitudes concurrentes sin bloqueos ni errores, mostrando las tareas y métricas actualizadas dentro de tiempos aceptables.
	Medida de Respuesta	Tiempo promedio de respuesta menor o igual a 2 segundos por solicitud HTTP; máximo 5% de peticiones con demora superior a 3 segundos.

Identificador: EC-02		
Escenario(s):		Un usuario malintencionado intenta acceder a la cuenta de otro usuario de Focusa mediante un ataque de fuerza bruta o reutilización de sesión.
Atributos de Calidad relevantes:		Seguridad – Confidencialidad – Integridad de datos
Componentes del Escenario	Estímulos:	Múltiples intentos consecutivos de autenticación con credenciales incorrectas, o uso de tokens/sesiones

		caducadas o manipuladas.
	Fuente del estímulo	Usuario externo malintencionado.
	Ambiente:	Plataforma Focusa desplegada en entorno web seguro (HTTPS) con autenticación de Django y middleware de seguridad.
	Artefacto:	Módulo de autenticación (login/logout), manejo de sesiones, protección CSRF y lógica de bloqueo de cuentas.
	Respuesta:	Focusa bloquea temporalmente la cuenta tras un número definido de intentos fallidos, registra la IP y el evento en logs, invalida sesiones sospechosas y no permite el acceso no autorizado.
	Medida de Respuesta	Cero exposición de datos personales; tasa de mitigación de intentos maliciosos ≥ 99%; todos los eventos críticos registrados en logs de seguridad.

Identificador: EC-03		
Escenario(s):		Se produce una falla temporal en el servicio de base de datos MySQL mientras los usuarios están creando o actualizando tareas en Focusa.
Atributos de Calidad relevantes:		Disponibilidad – Tolerancia a fallos – Recuperabilidad
Componentes del Escenario	Estímulos:	Interrupción del contenedor de base de datos, pérdida de conexión entre el backend Django y MySQL o reinicio inesperado del servicio.
	Fuente del estímulo	Error de red, reinicio del host, mantenimiento no planificado o falla del motor.
	Ambiente:	Entorno de producción con usuarios activos gestionando tareas y consultando información.
	Artefacto:	Backend Django (ORM, capa de persistencia de tareas) y contenedor de base de datos MySQL.
	Respuesta:	Focusa detecta la falla, detiene temporalmente la operación crítica, muestra un mensaje de “servicio temporalmente no disponible” e intenta restablecer la conexión de forma automática; al recuperarse, sincroniza y asegura la consistencia de las transacciones.
	Medida de Respuesta	Tiempo máximo de recuperación ≤ 5 minutos; pérdida de transacciones = 0; disponibilidad global objetivo ≥ 99%.

Identificador: EC-04		
Escenario(s):		Un usuario de Focusa accede desde distintos dispositivos (móvil, tablet, notebook) para gestionar tareas, visualizar el tablero Kanban y revisar el dashboard de productividad.
Atributos de Calidad relevantes:		Usabilidad – Portabilidad – Accesibilidad
Componentes del Escenario	Estímulos:	Navegación desde dispositivos con diferentes tamaños de pantalla y navegadores (desktop, móvil, tablet).
	Fuente del estímulo	Usuario final que requiere una experiencia fluida, consistente y legible desde cualquier dispositivo.
	Ambiente:	Entorno de producción de Focusa, accesible vía navegador sobre HTTPS.

	Artefacto:	Frontend (plantillas Django, HTML, CSS responsivo y componentes de interfaz).
	Respuesta:	La interfaz de Focusa se adapta automáticamente al dispositivo, manteniendo legibilidad, estructura clara y permitiendo completar acciones clave (crear, editar, completar tareas, ver tablero) sin errores de formato ni desbordes.
	Medida de Respuesta	Tasa de tareas completadas sin error en dispositivos móviles ≥ 95%; tiempo medio para completar una acción típica (crear y marcar tarea) ≤ 3 minutos.

Identificador: EC-05		
Escenario(s):		El equipo de desarrollo debe incorporar un nuevo módulo en Focusa (por ejemplo, “Compartir tareas con otros usuarios” o “Integraciones externas”) sin afectar las funcionalidades ya operativas.
Atributos de Calidad relevantes:		Mantenibilidad – Modificabilidad – Reusabilidad
Componentes del Escenario	Estímulos:	Solicitud de ampliación del sistema para agregar nuevas capacidades a la plataforma SaaS Focusa.
	Fuente del estímulo	Equipo de proyecto / stakeholders internos (equipo de desarrollo, coordinación académica/profesional).
	Ambiente:	Entorno de desarrollo controlado, con uso de Git, entorno virtual, estructura modular en Django y despliegue mediante Docker.
	Artefacto:	Código fuente del proyecto (apps Django existentes, modelos, vistas, APIs internas), modelo de datos y configuración de contenedores.
	Respuesta:	El nuevo módulo se integra utilizando la arquitectura modular existente, el ORM y las vistas genéricas, sin introducir errores en funcionalidades ya desplegadas ni romper migraciones o procesos de despliegue.
	Medida de Respuesta	Tiempo máximo de implementación dentro del sprint planificado; incidencias críticas post-deploy = 0; cobertura de pruebas unitarias y funcionales ≥ 80–90% para el nuevo módulo.

Identificador: EC-06		
Escenario(s):		Un usuario intenta visualizar o editar una tarea perteneciente a otro usuario mediante manipulación manual del ID o acceso directo por URL.
Atributos de Calidad relevantes:		Seguridad – Confidencialidad – Integridad de datos
Componentes del Escenario	Estímulos:	Intento de acceso no autorizado a recursos privados manipulando el ID o URL del recurso.
	Fuente del estímulo	Usuario malintencionado (interno o externo).
	Ambiente:	Entorno de producción de Focusa, mediante navegador

		HTTPS.
	Artefacto:	Código fuente del proyecto (apps Django existentes, modelos, vistas, APIs internas), modelo de datos y configuración de contenedores.
	Respuesta:	Focusa valida automáticamente la propiedad del recurso, deniega la solicitud, registra el intento en logs y redirige a un mensaje de acceso restringido.
	Medida de Respuesta	0% de accesos no autorizados permitidos; 100% de intentos registrados en logs.

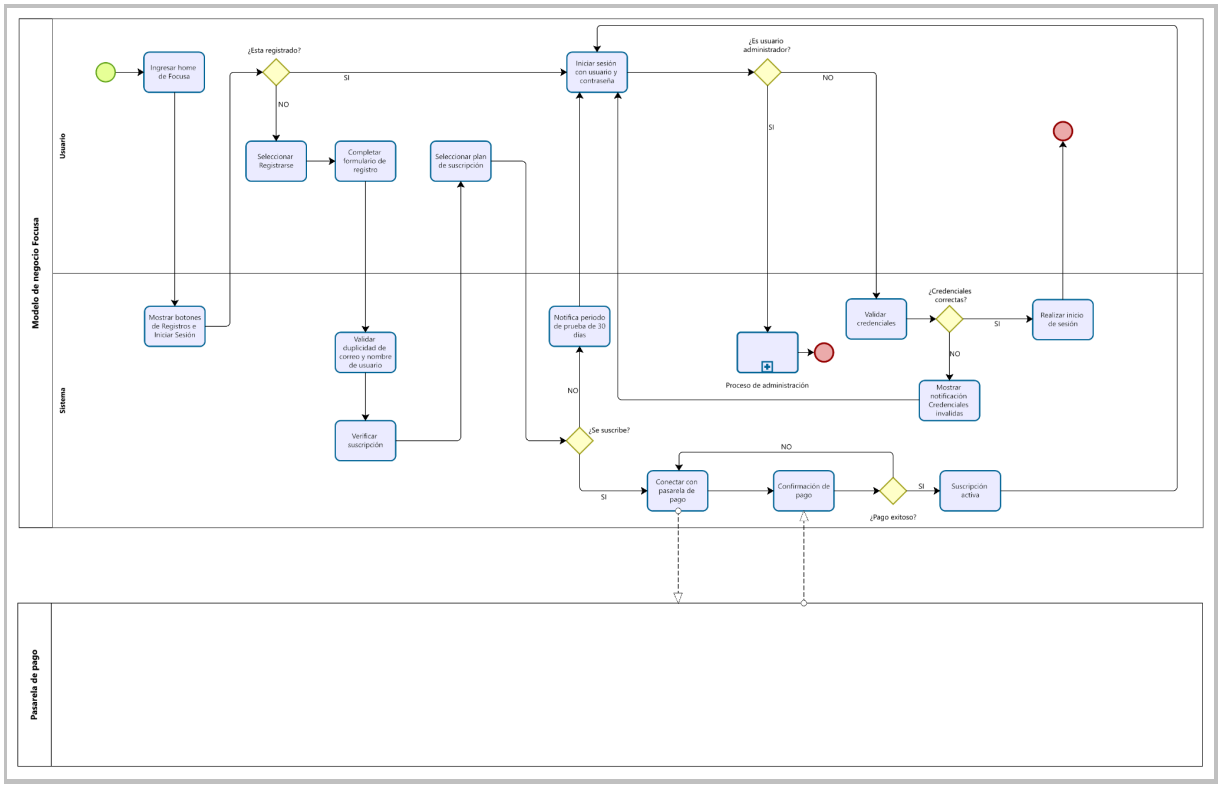
Identificador: EC-07		
Escenario(s):		El contenedor Docker que ejecuta el backend Django falla o se reinicia inesperadamente mientras usuarios están gestionando tareas.
Atributos de Calidad relevantes:		Disponibilidad – Recuperabilidad – Tolerancia a fallos
Componentes del Escenario	Estímulos:	Falla del contenedor, reinicio inesperado o caída del servicio backend.
	Fuente del estímulo	Error interno del contenedor, falta de recursos o reinicio inesperado del host.
	Ambiente:	Entorno de producción de Focusa, mediante navegador HTTPS.
	Artefacto:	Producción con múltiples usuarios activos en el sistema.
	Respuesta:	Docker reinicia automáticamente el contenedor; Focusa restablece el servicio sin pérdida de datos y permite continuar las operaciones normales del sistema.
	Medida de Respuesta	Tiempo máximo de recuperación ≤ 1 minuto; pérdida de transacciones = 0.

Identificador: EC-08		
Escenario(s):		El usuario abre la vista de “Lista de tareas” con más de 200 tareas almacenadas.
Atributos de Calidad relevantes:		Rendimiento del Frontend – Eficiencia – Fluidez en la UI
Componentes del Escenario	Estímulos:	Carga masiva de elementos visuales (tarjetas o filas de tareas).
	Fuente del estímulo	Usuario final con alto volumen de tareas creadas.
	Ambiente:	Navegador web moderno; sesión activa.
	Artefacto:	Frontend (DOM dinámico), paginación, scripts de optimización visual.
	Respuesta:	El listado usa técnicas como paginación, lazy load o virtualización para evitar saturar el DOM y mantener la fluidez de navegación.
	Medida de Respuesta	Render inicial ≤ 1.5 segundos; scroll sin lag perceptible.

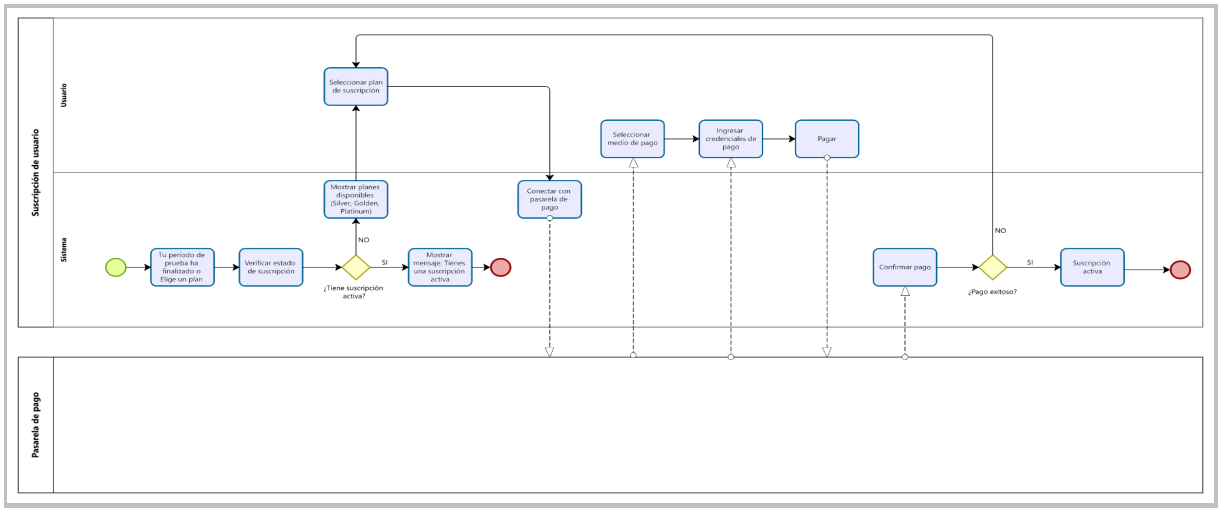
4. Vista de Procesos

Ilustración 2: Diagramas de Actividades

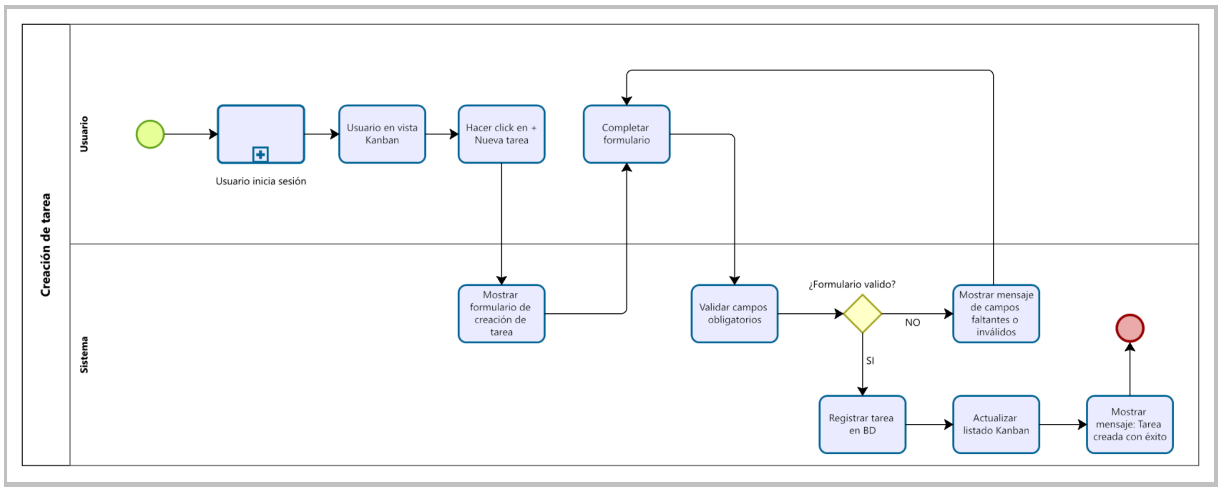
Registro y autenticación de usuario.



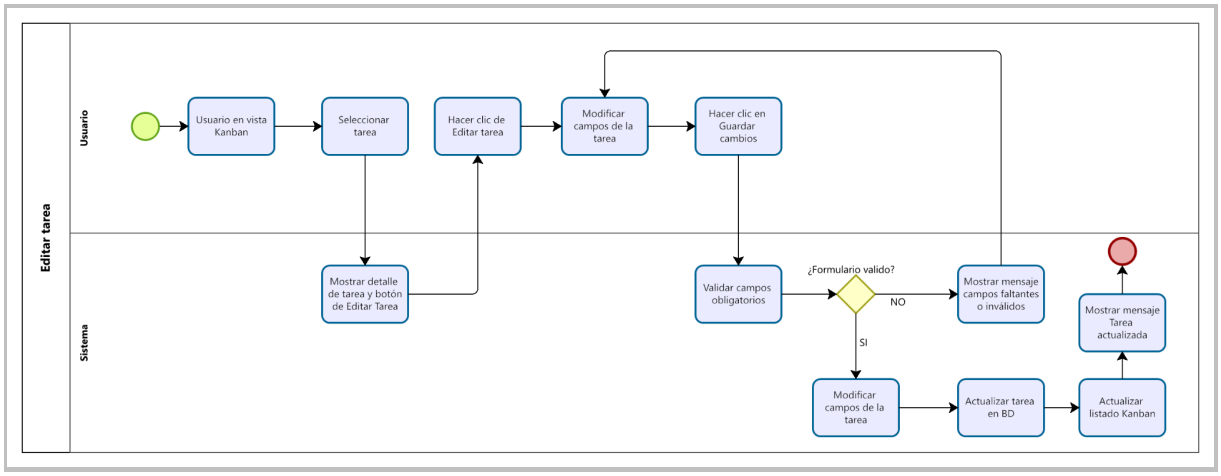
Suscripción



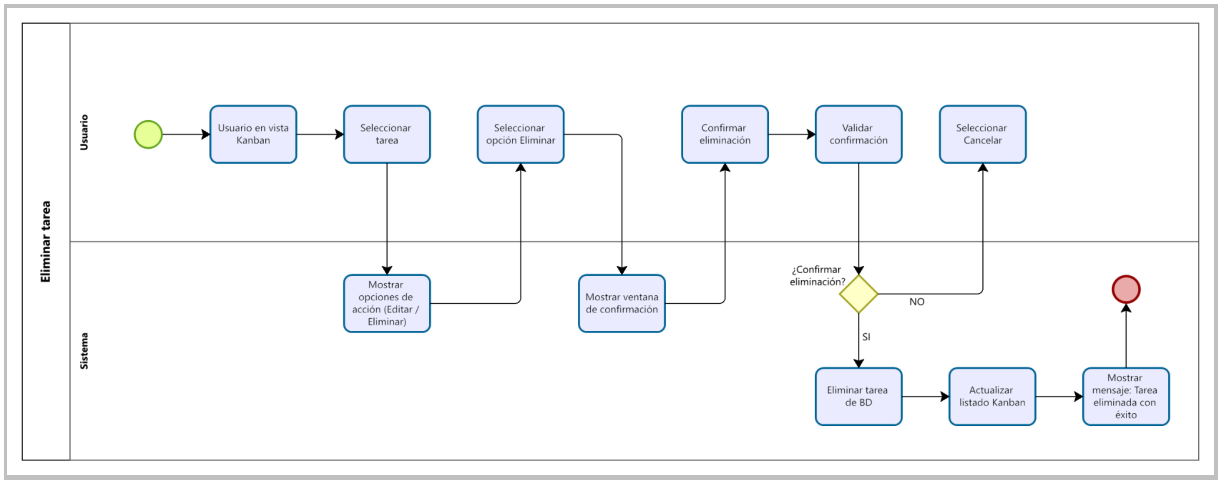
Creación Tarea



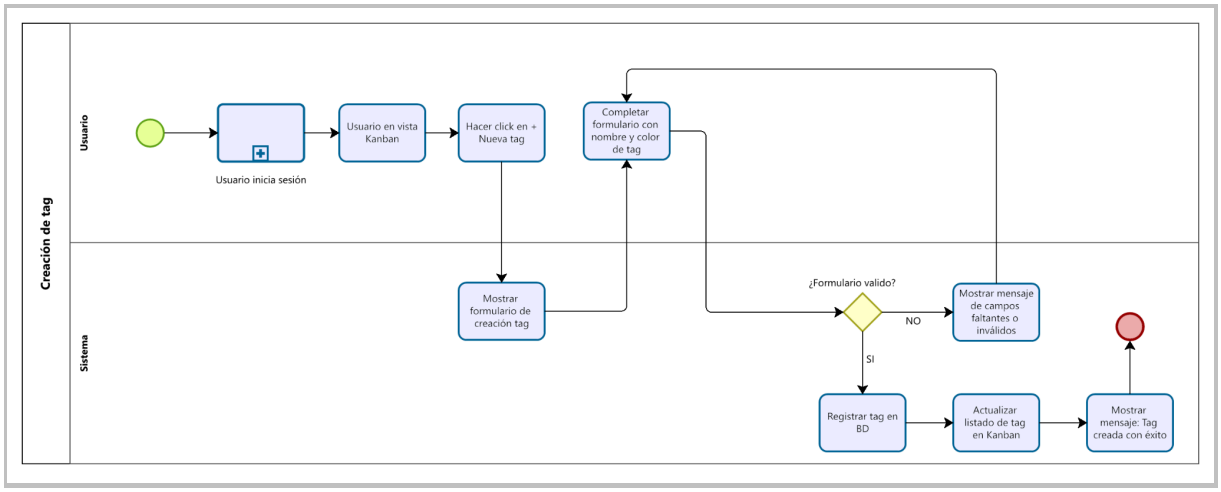
Edición Tarea



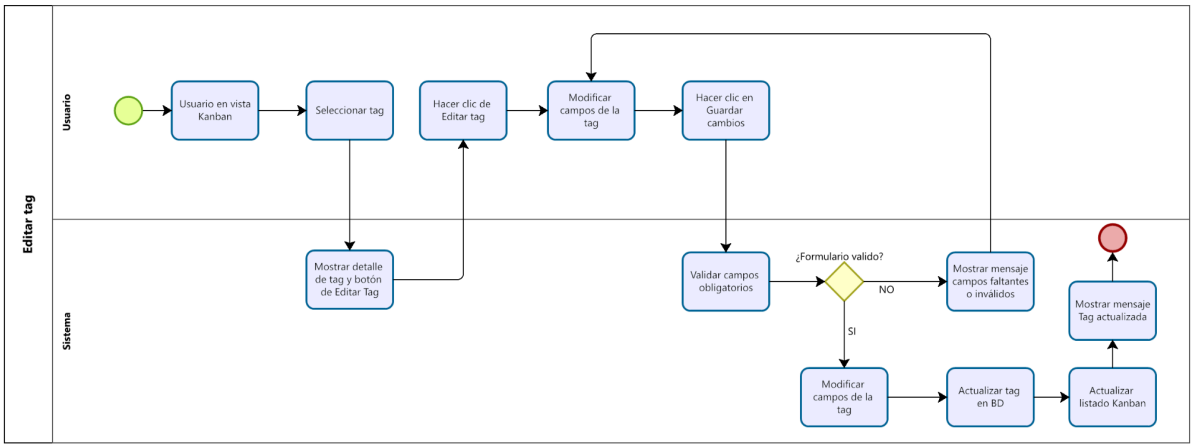
Eliminación Tarea



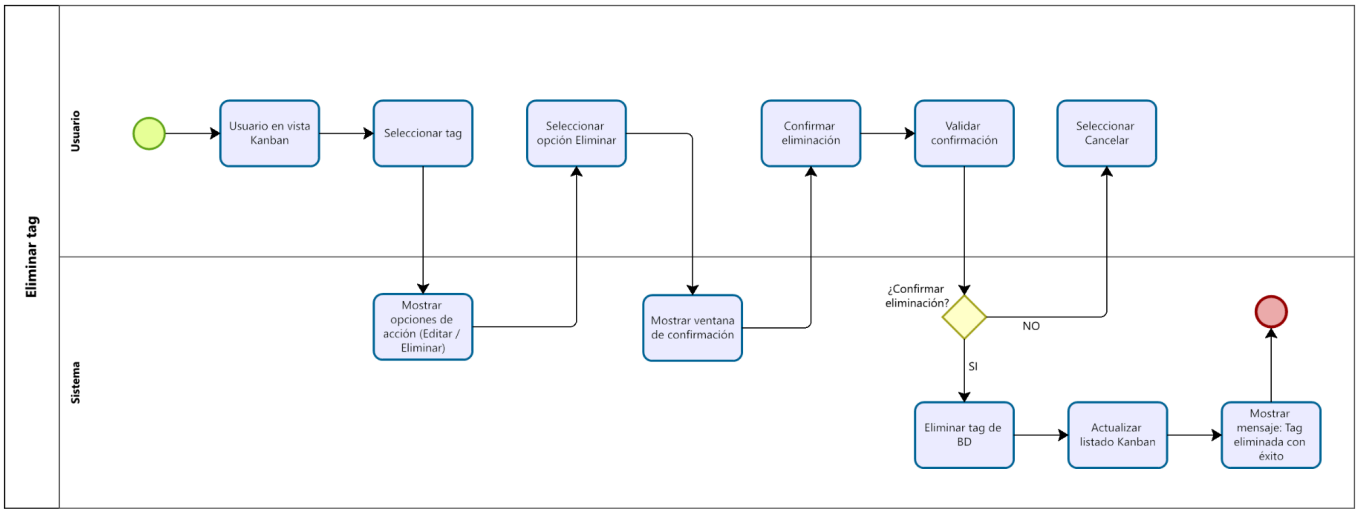
Creación Tag



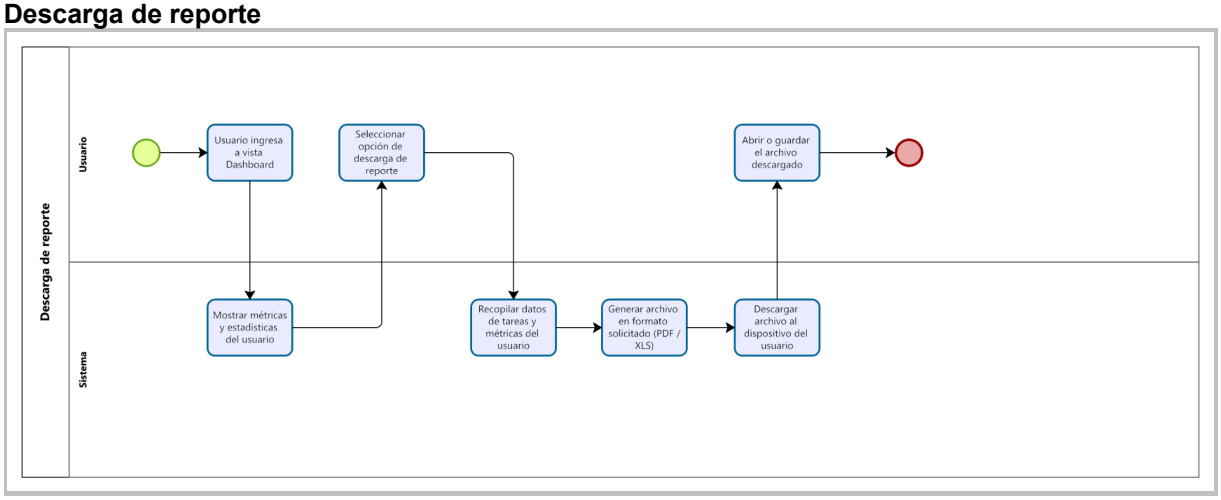
Edición Tag



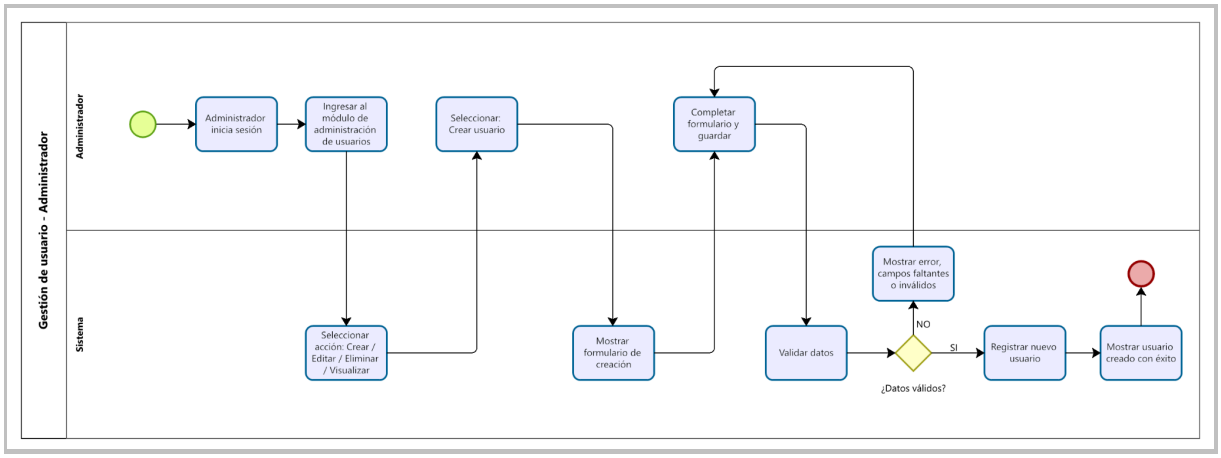
Eliminación Tag



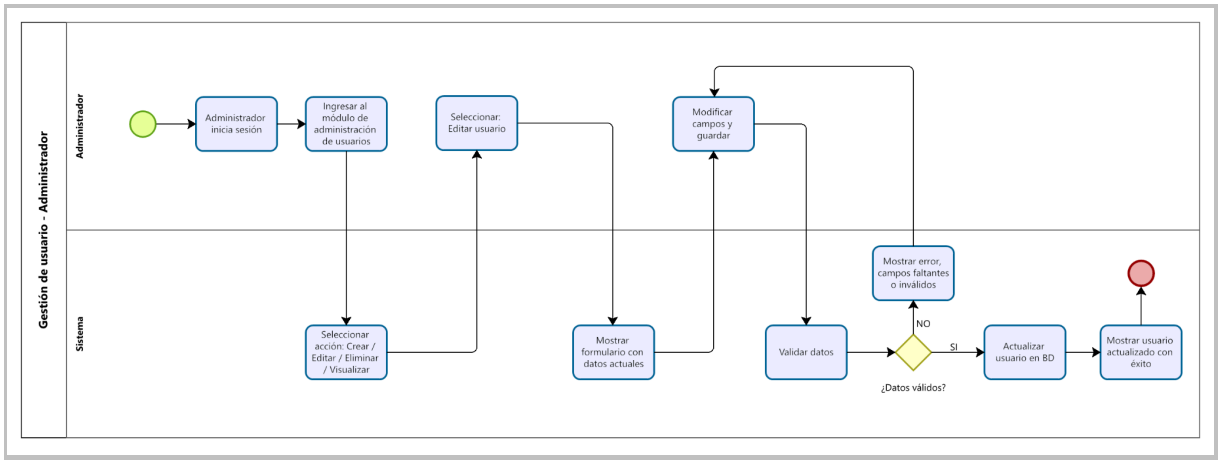
Descarga de reporte



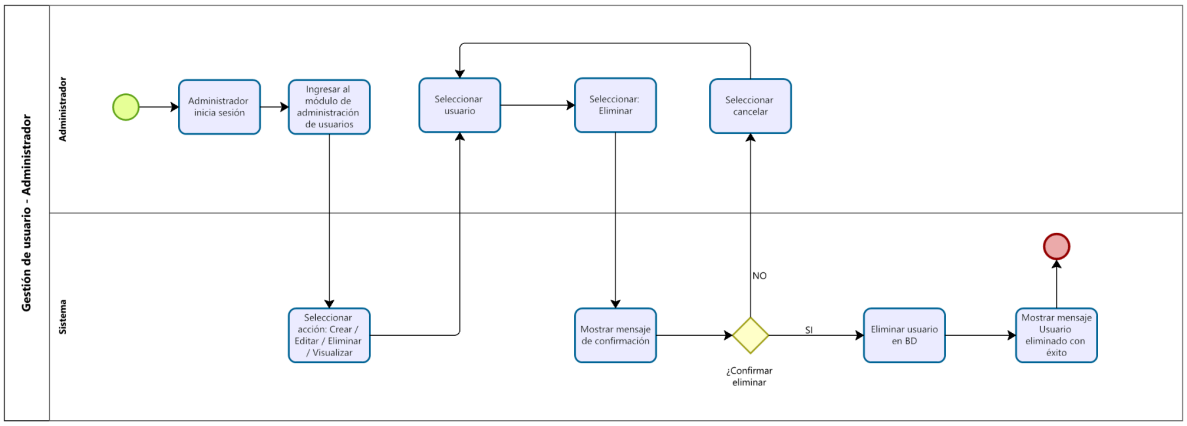
Administración Sistema Crear usuario



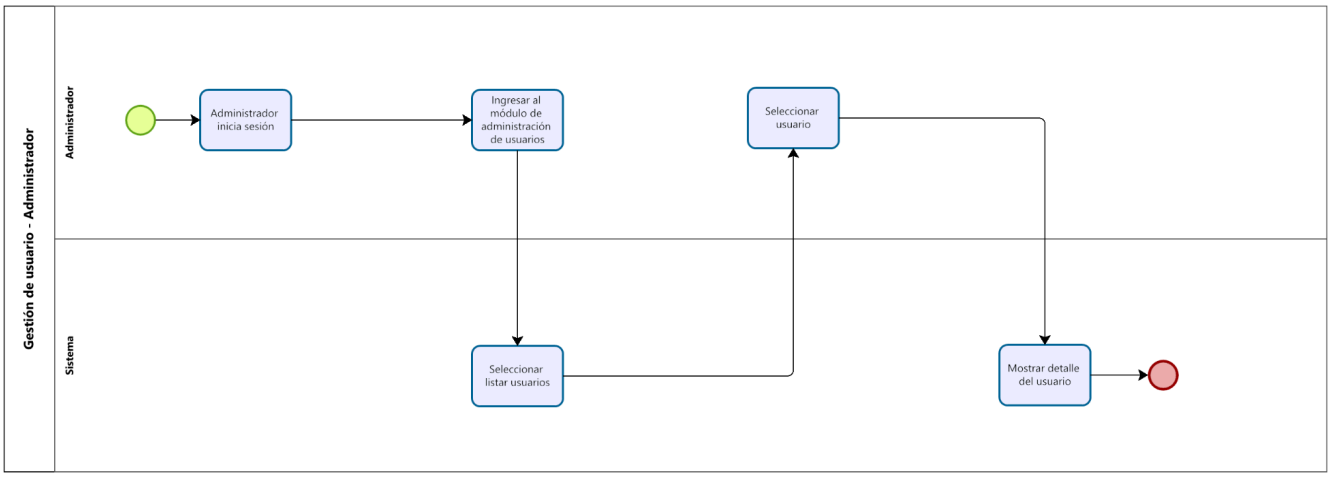
Administración Sistema Editar usuario



Administración Sistema Eliminar usuario



Administración Sistema Visualizar usuario



5. Vista Lógica

A continuación, se presenta una vista lógica de la aplicación expresado en tres diagramas, uno de ellos que muestra la parte estructural o estática de la aplicación (clases) y a la base de datos (modelo relacional).otra vista que representa la parte dinámica (secuencias).

☐ Parte Estructural (Diagrama de Clases y Diagrama Relacional)

Ilustración 3: Diagrama de Clases

1. Ilustración X.1 – Diagrama de Clases: Usuarios y suscripciones

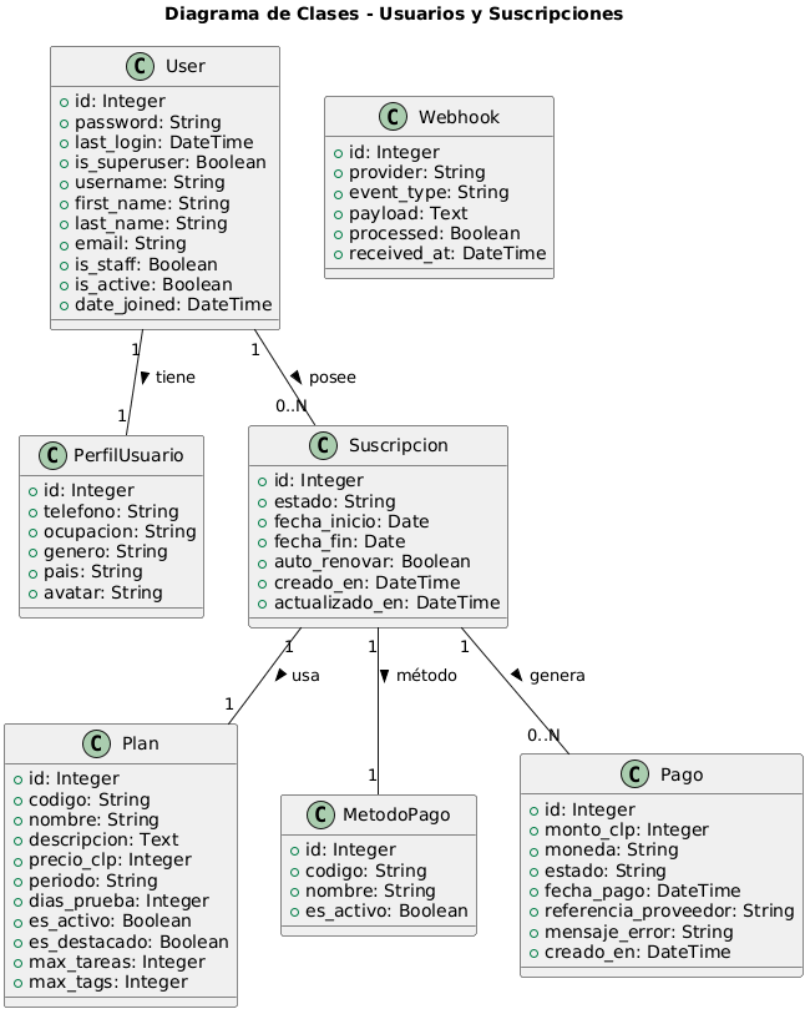
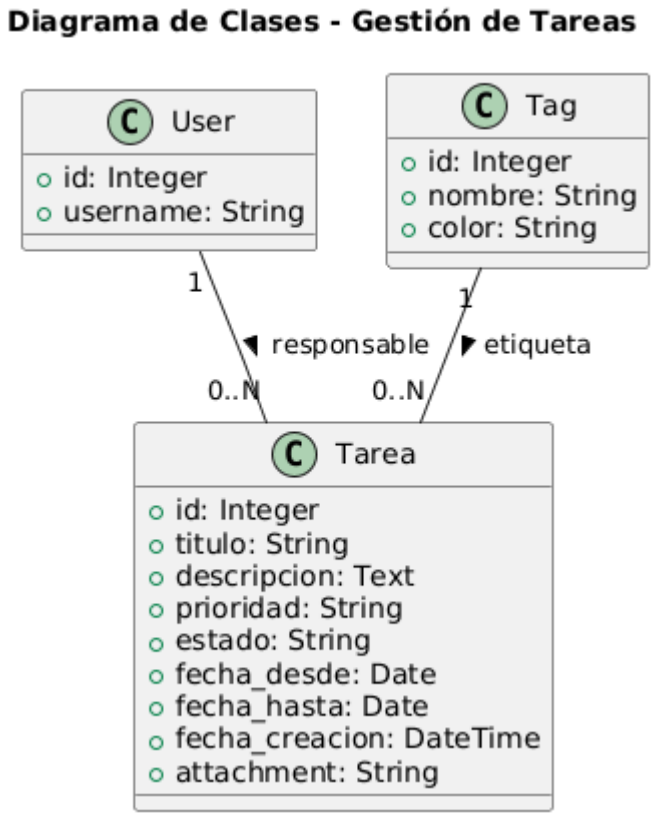
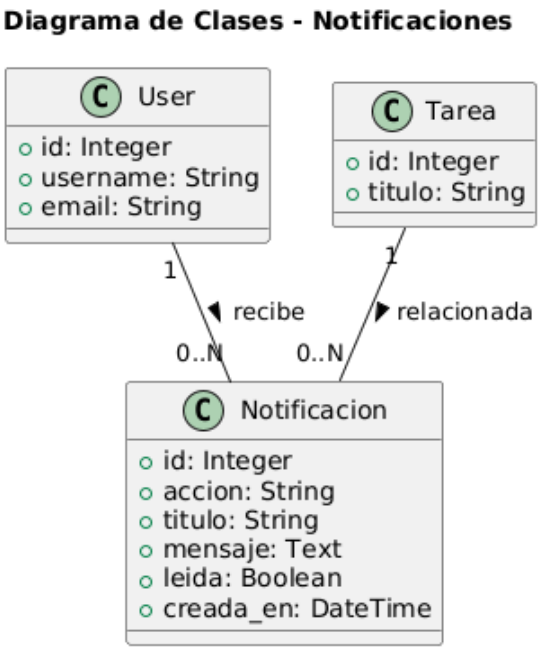


Ilustración X.2 – Diagrama de Clases: Gestión de tareas

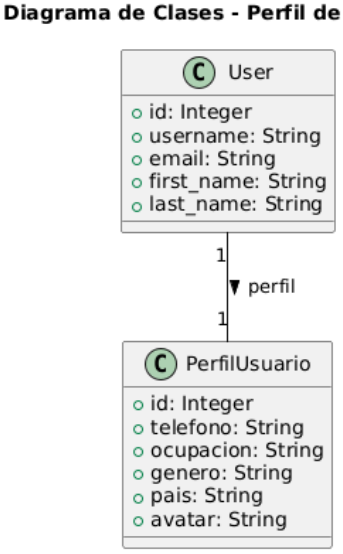


2. Ilustración X.3 – Diagrama de Clases: Notificación

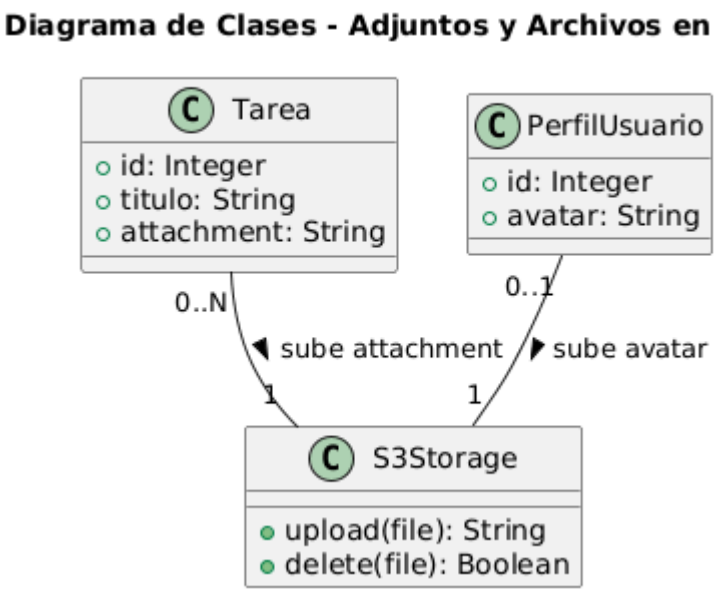


3.

4. Ilustración X.3 – Diagrama de Clases: Perfil de Usuario



5. Ilustración X.3 – Diagrama de Clases: Uso de Archivos en S3

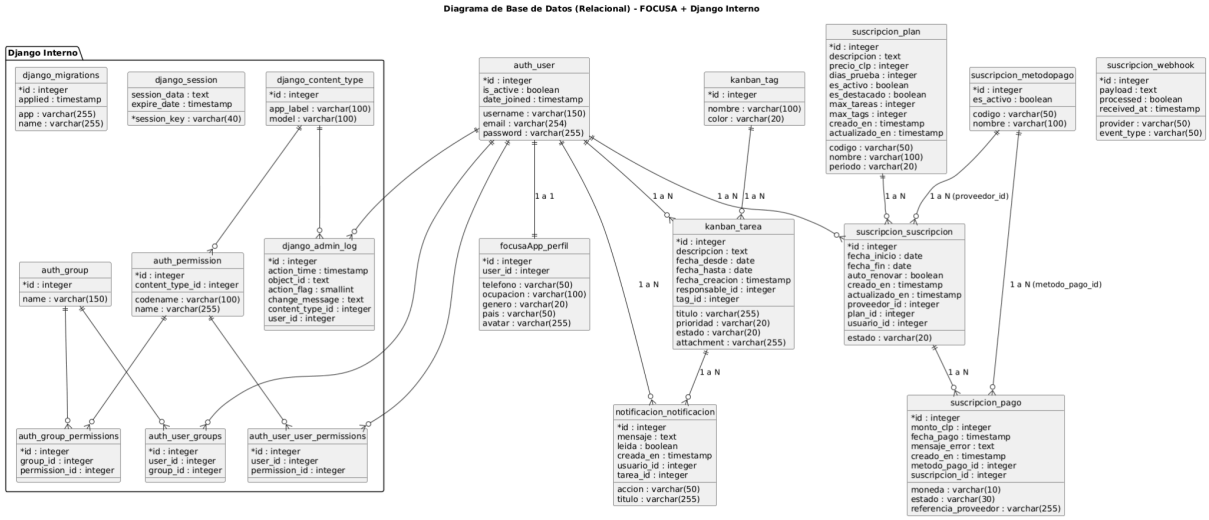


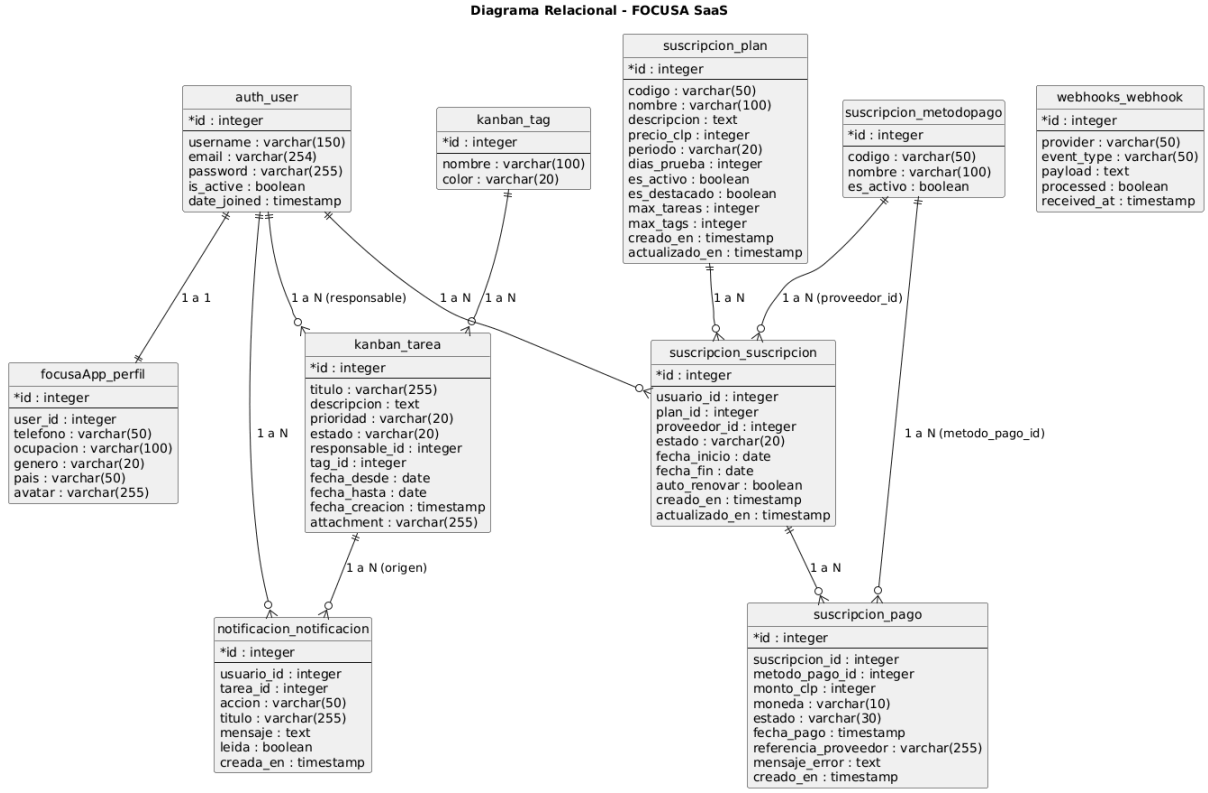
6. Descripción de Clases.

Código	Nombre	Descripción
CL-001	Usuario	Representa a los usuarios del sistema. Gestiona credenciales, autenticación, datos personales y permisos.
CL-002	PerfilUsuario	Extiende la información del usuario con datos adicionales como teléfono, país, ocupación y avatar almacenado en S3.
CL-003	Tarea	Unidad principal del sistema Kanban. Contiene título, descripción, prioridad, estado, fechas, tag y archivo adjunto en S3.
CL-004	Tag	Clasificación visual de tareas. Cada tarea tiene un único tag con nombre y color.
CL-005	Attachment	Archivo adjunto asociado a una tarea. Es un atributo dentro de Tarea y almacena la URL del archivo guardado en S3.
CL-006	Notificación	Alertas enviadas al usuario. Relacionadas a suscripciones, tareas o eventos del sistema.
CL-007	Suscripción	Representa el plan activo del usuario (Básico/Pro). Gestiona fechas, periodo de prueba y estado.
CL-008	MétodoPago	Métodos de pago registrados por el usuario, como tarjetas o proveedores externos.
CL-009	Pago	Registro de transacciones del usuario. Contiene monto, estado, fecha y método asociado.

CL-010	**ReporteAvance**	**Reportes automáticos o manuales disponibles para usuarios Pro, almacenados como PDF en S3.**
CL-011	**Administrador**	**Rol especial con permisos avanzados. No es una tabla separada, deriva de Usuario.**
CL-012	**LogSistema**	**Registro de acciones relevantes para auditoría y monitorización del sistema.**
CL-013	**APIKey**	**Claves generadas para integraciones externas, utilizadas para validación segura de solicitudes.**
CL-014	**Sesión**	**Registros de sesiones activas del usuario, implementadas con django_session.**
CL-015	**Webhook**	**Registra eventos recibidos desde proveedores externos de pago para actualizar transacciones.**

Ilustración 4: Diagrama de Base Datos (Relacional)





Descripción de las Tablas		
Código	Nombre	Descripción
TB-001	auth_user	Registra a los usuarios del sistema. Contiene username, correo, contraseña, estado y fecha de creación.
TB-002	focusaApp_perfil	Información complementaria del usuario: teléfono, ocupación, país, género y avatar almacenado en AWS S3.
TB-003	kanban_tarea	Contiene las tareas creadas por los usuarios: título, descripción, estado, prioridad, responsable, fechas y adjunto en S3.
TB-004	kanban_tag	Etiquetas que clasifican las tareas. Cada tag tiene un nombre y un color.
TB-005	notificacion_notificacion	Registro de notificaciones enviadas a los usuarios: acciones del sistema, mensajes, estado leído y fecha.
TB-006	suscripcion_plan	Catálogo de planes disponibles. Define precio, periodo, prueba, límites de tareas y tags, y estado del plan.
TB-007	suscripcion_suscripcion	Suscripciones del usuario: fechas, estado, auto-renovación, plan asociado y proveedor.
TB-008	suscripcion_metodopago	Métodos de pago registrados. Contiene código, nombre y estado del método.
TB-009	suscripcion_pago	Pagos realizados por el usuario: monto, moneda, estado, referencia externa y fecha de pago.

TB-010	WEBHOOKS	Registra eventos enviados por la pasarela de pago. Almacena el payload, tipo de evento, proveedor y estado de procesamiento.

7. Parte Dinámica (Diagrama de Secuencias)

Ilustración 1: Registro e Inicio de Sesión de Usuario

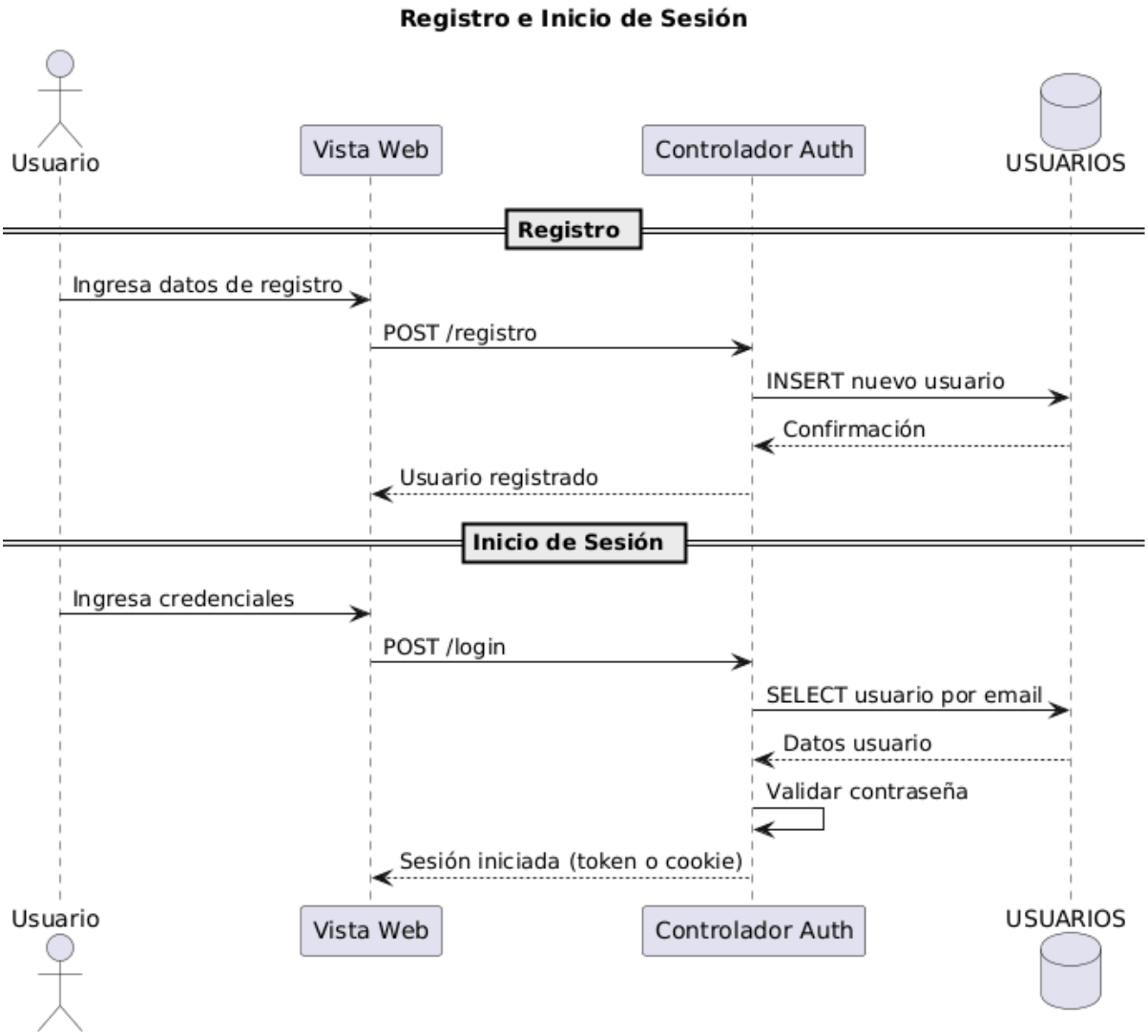


Ilustración 2: Crear Tarea

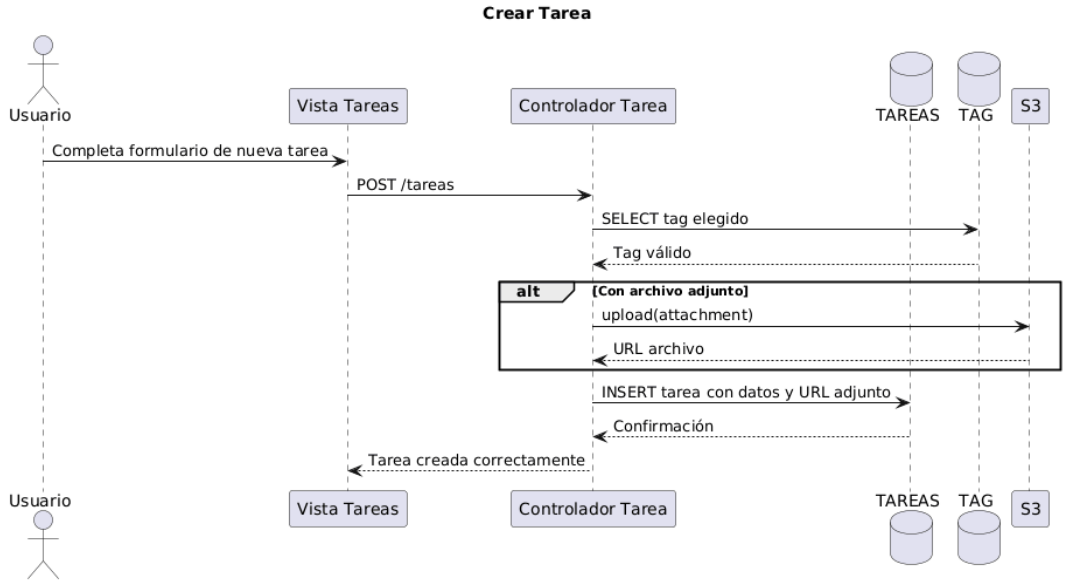


Ilustración 3: Editar Tarea

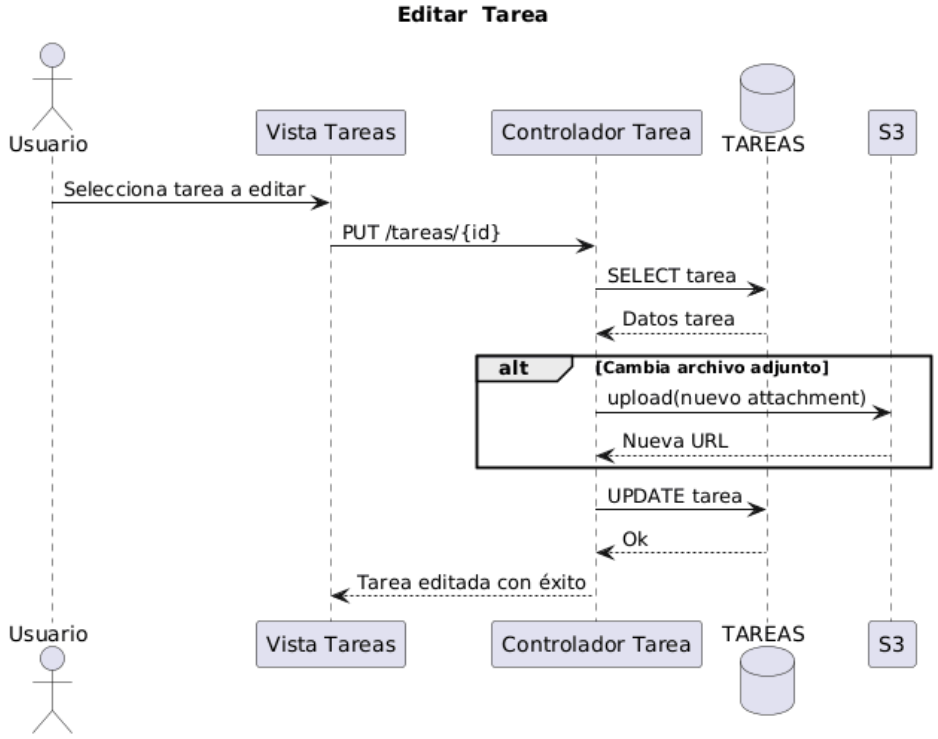


Ilustración 4: Eliminar Tarea

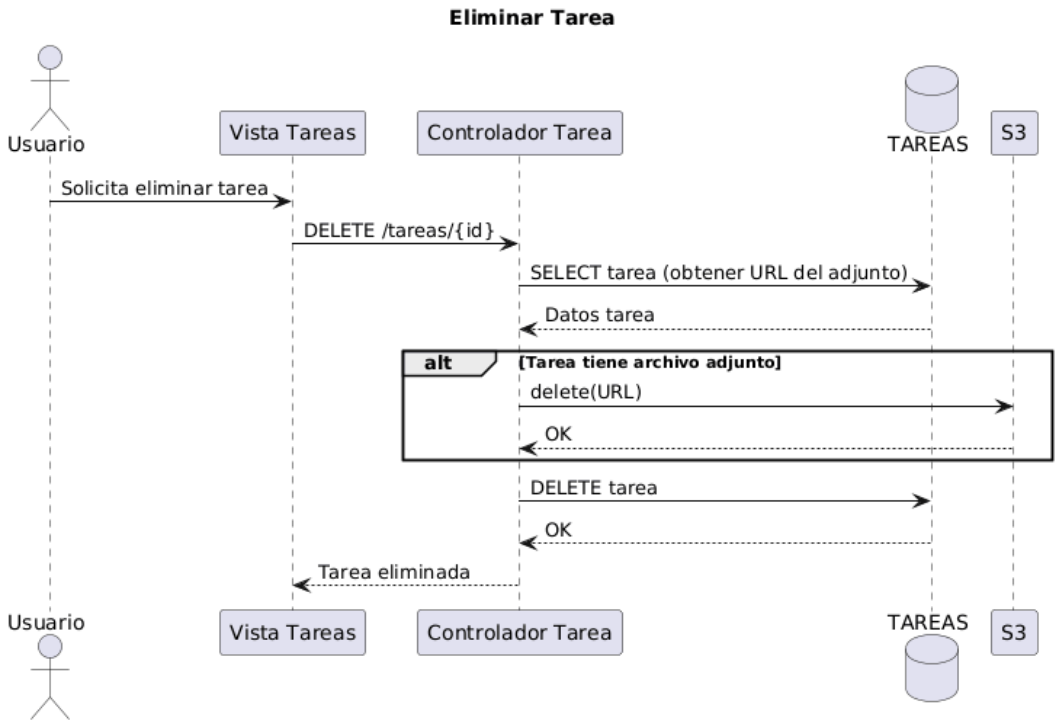


Ilustración 6: Generar Reporte de avance

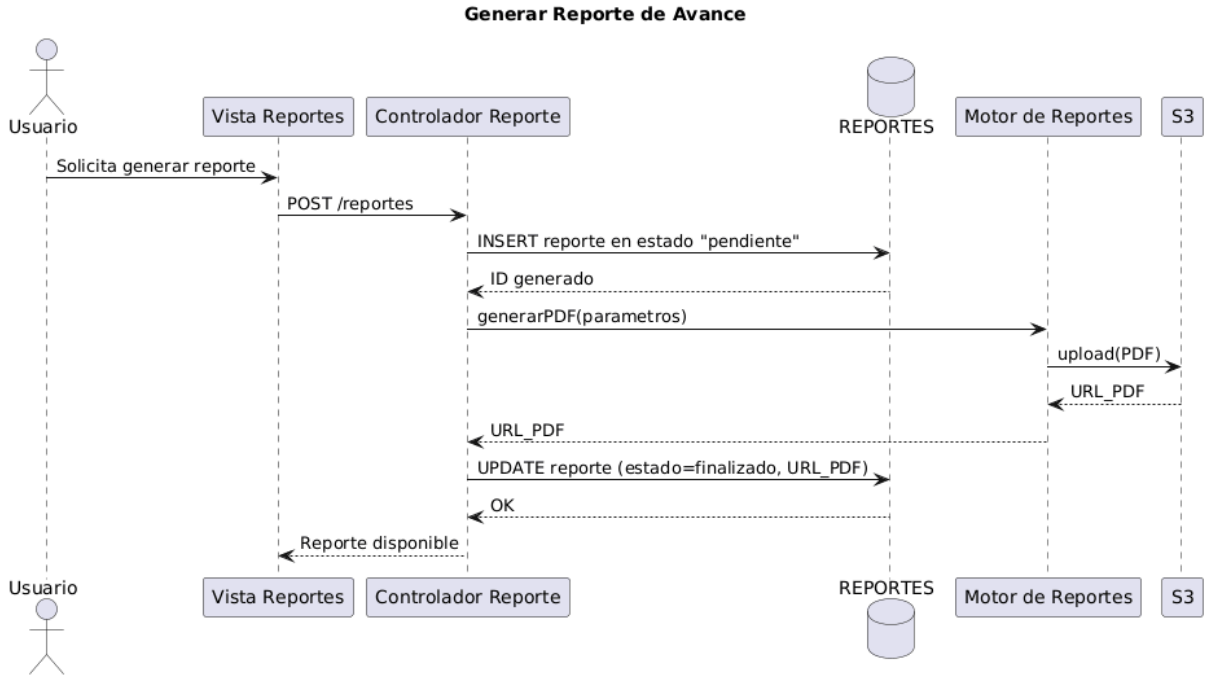


Ilustración 7 : Suscripciones y pagos

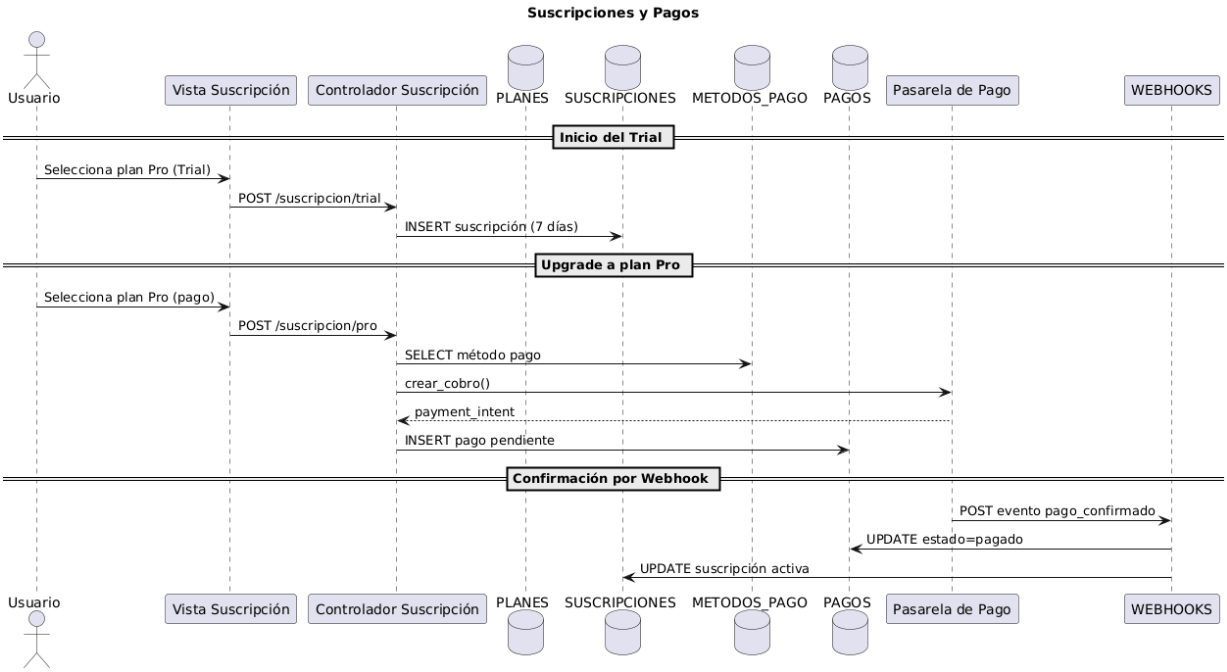


Ilustración 8 Notificaciones

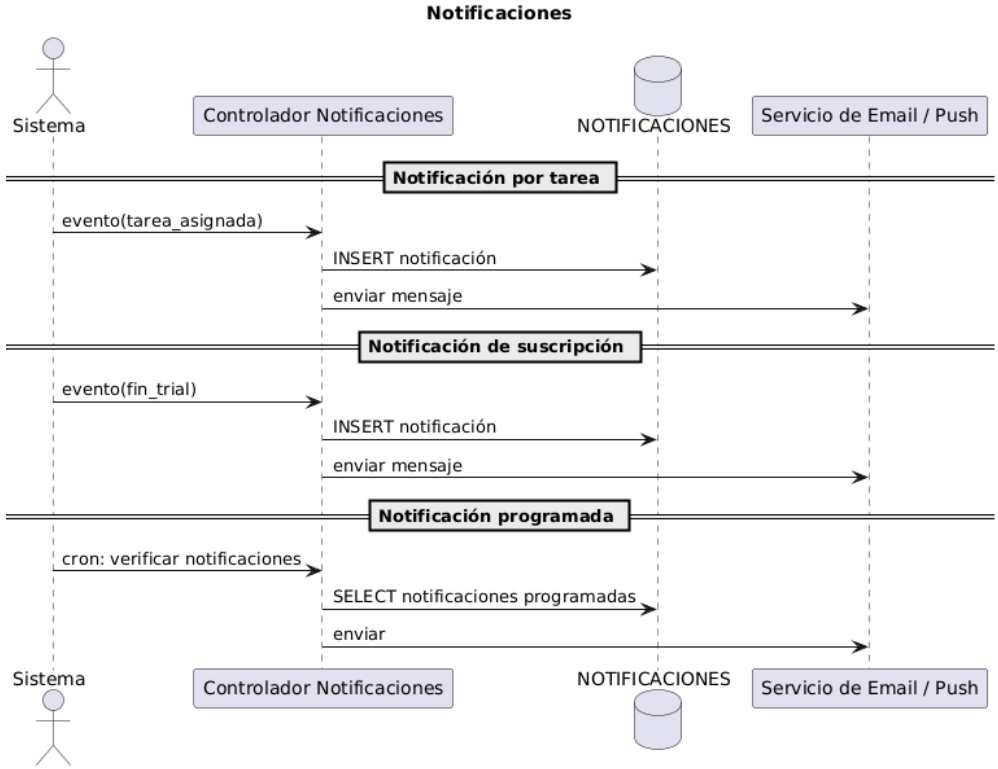
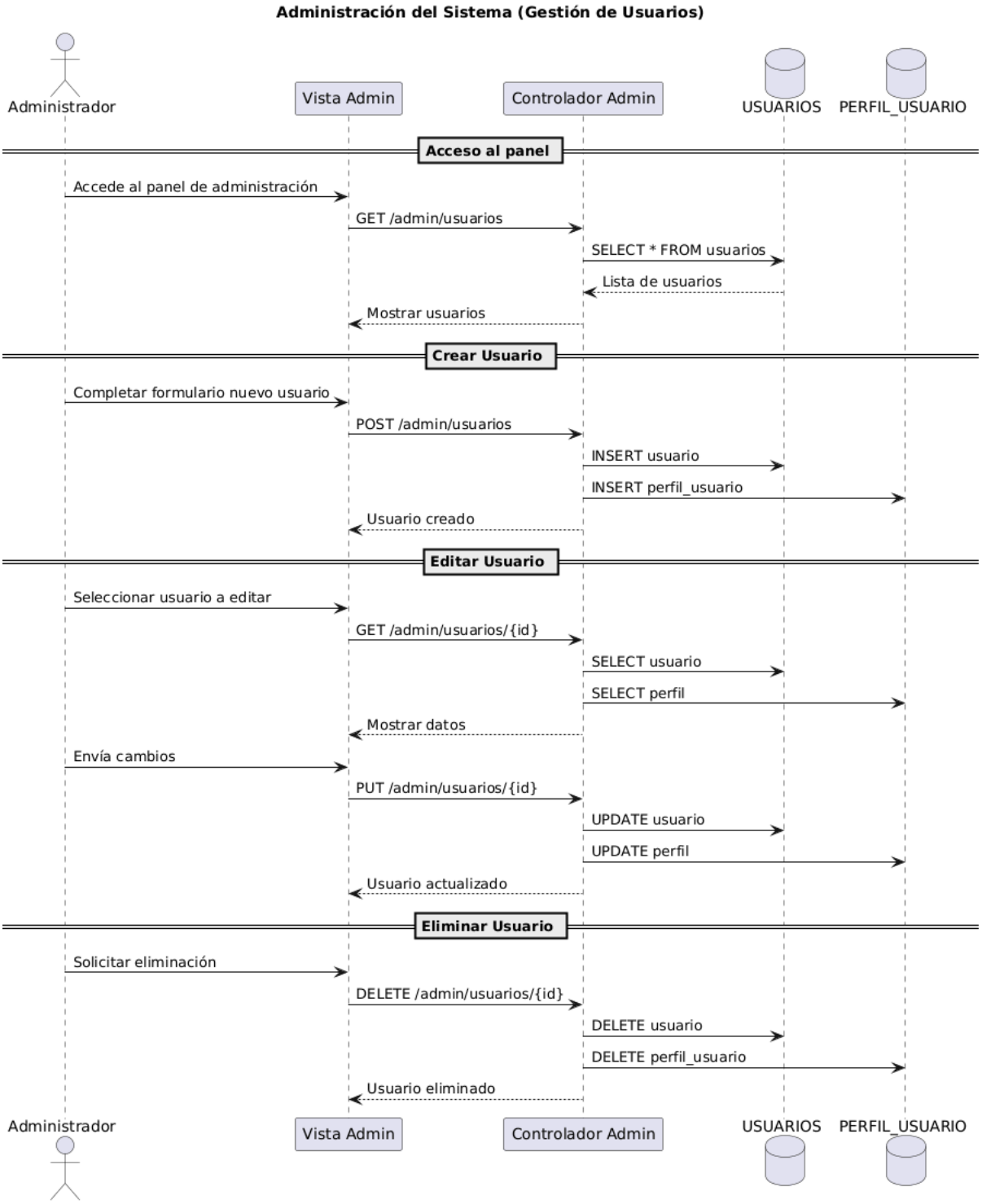


Ilustración 8.1: Administrador



8. Vista de Desarrollo o Despliegue

En esta vista se describen las componente o modulos en las cuales se dividirá o implementará el sistema

Ilustración 1: Diagrama de componentes general del sistema

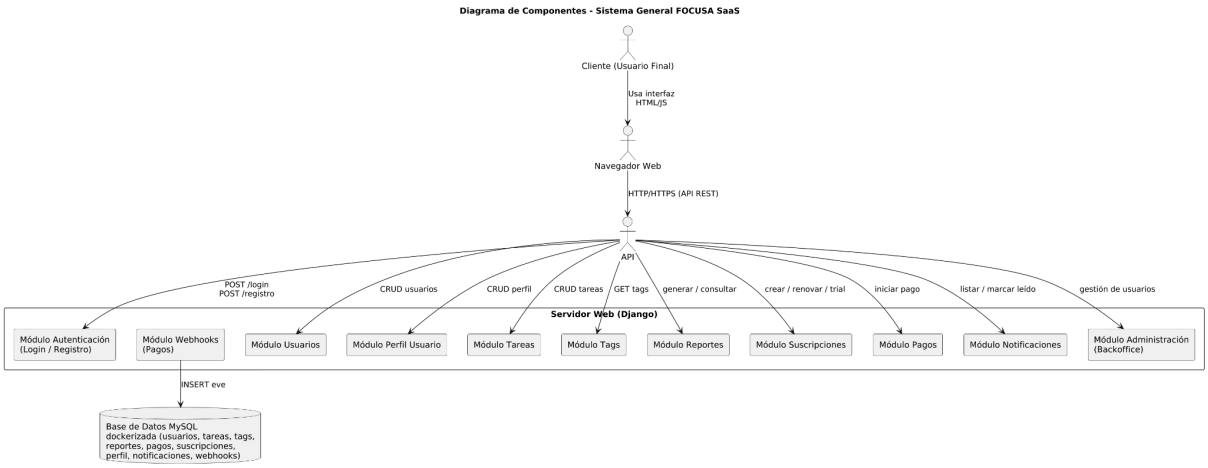


Ilustración 2: Diagrama de componentes modulo usuarios

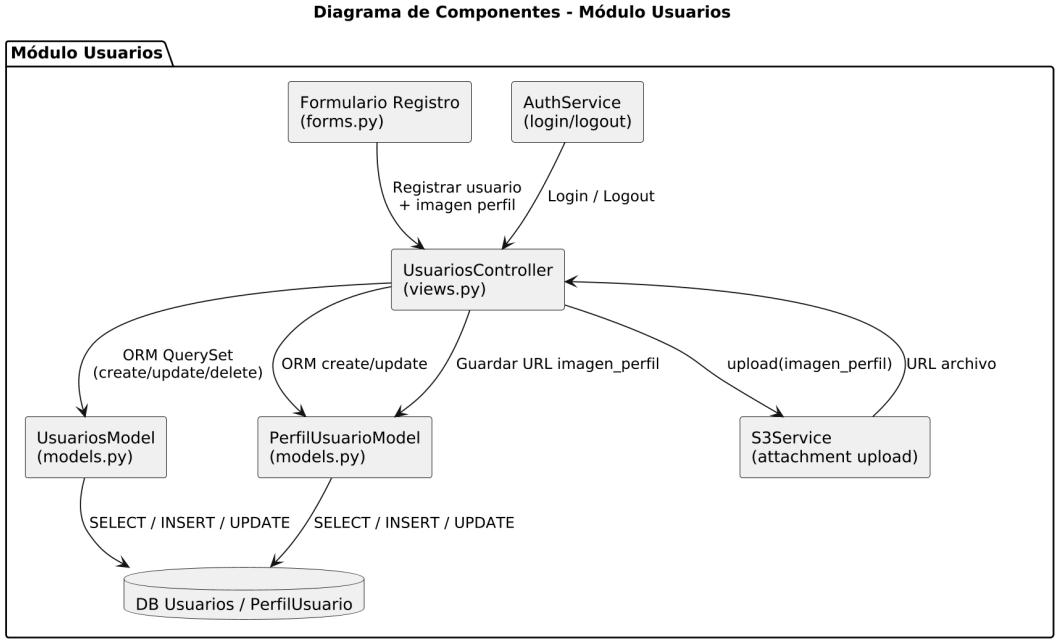


Ilustración 3: diagrama de componentes Tareas

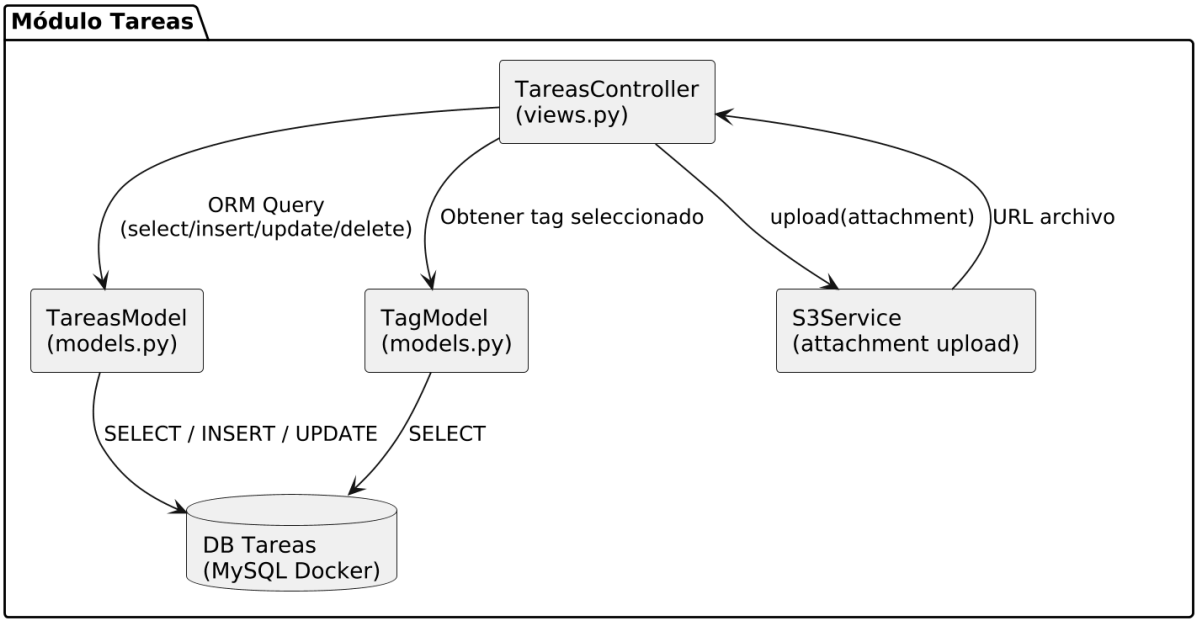


ilustración 4: diagrama de componentes TAG

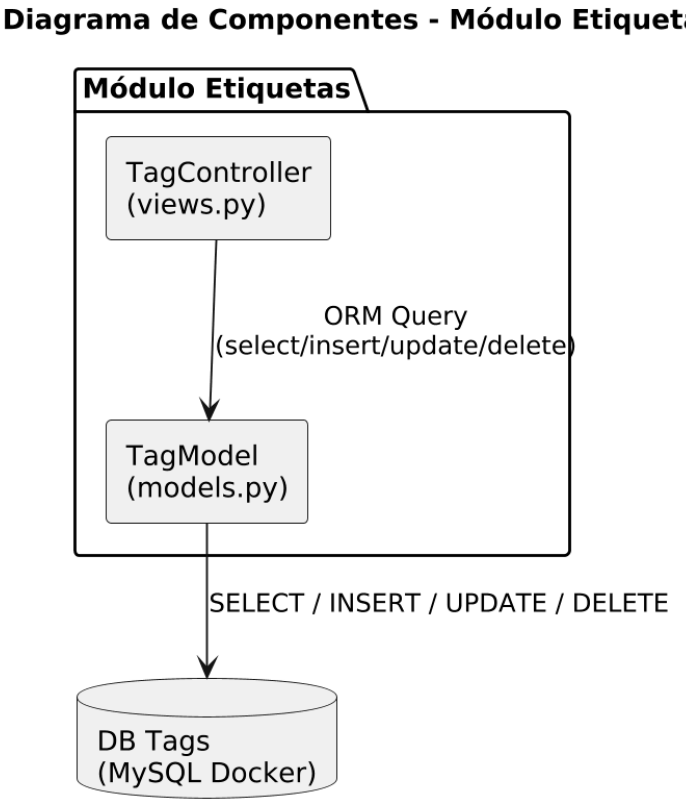


ilustración 5: diagrama de componente Suscripciones y pagos

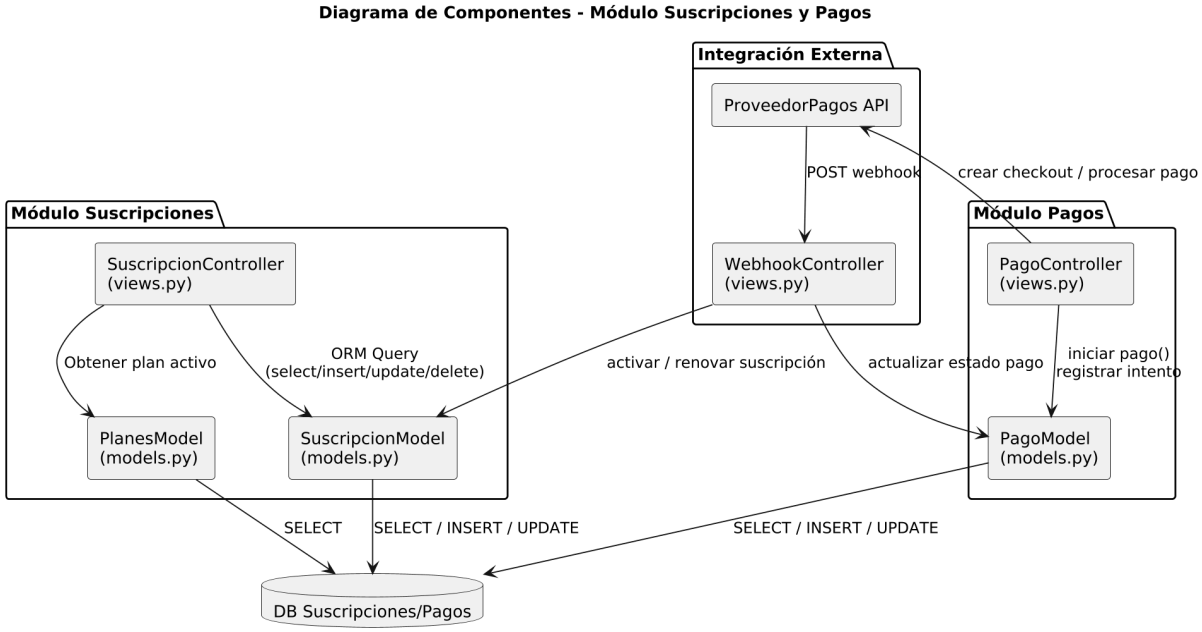


ilustración 6: diagrama de componente Notificaciones

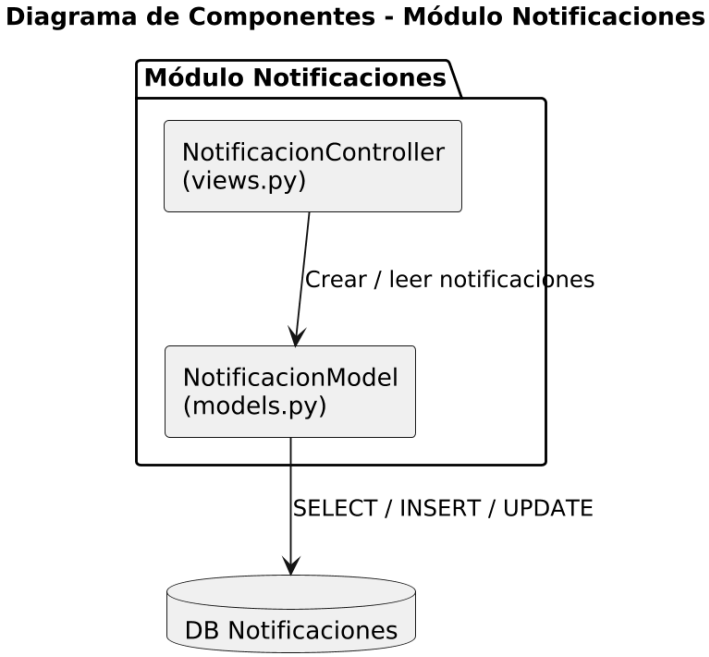
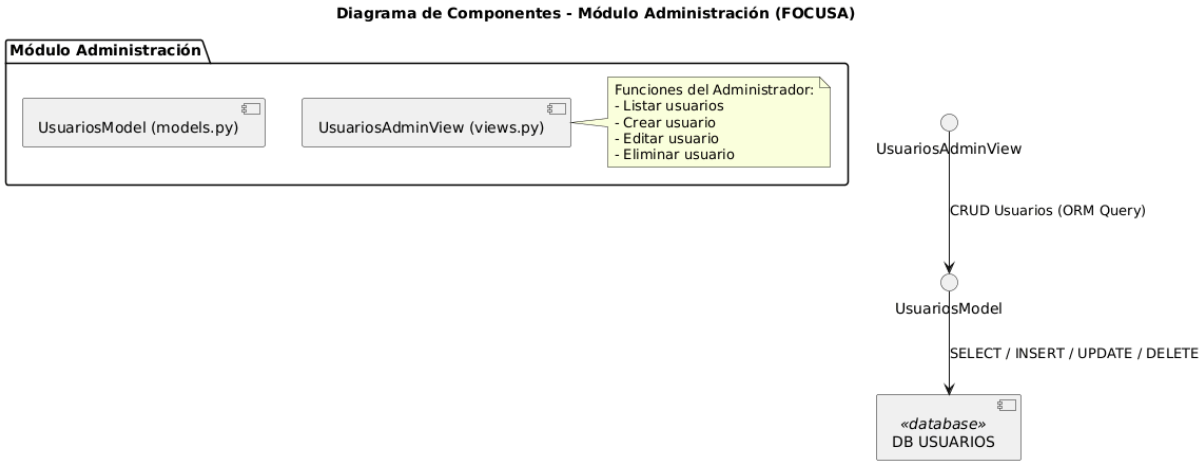


Ilustración 5: diagrama de componente BackOffice



9. Vista Física

En esta vista se despliegan los nodos que participan con el sistema.

Ilustración 1: diagrama despliegue general del sistema

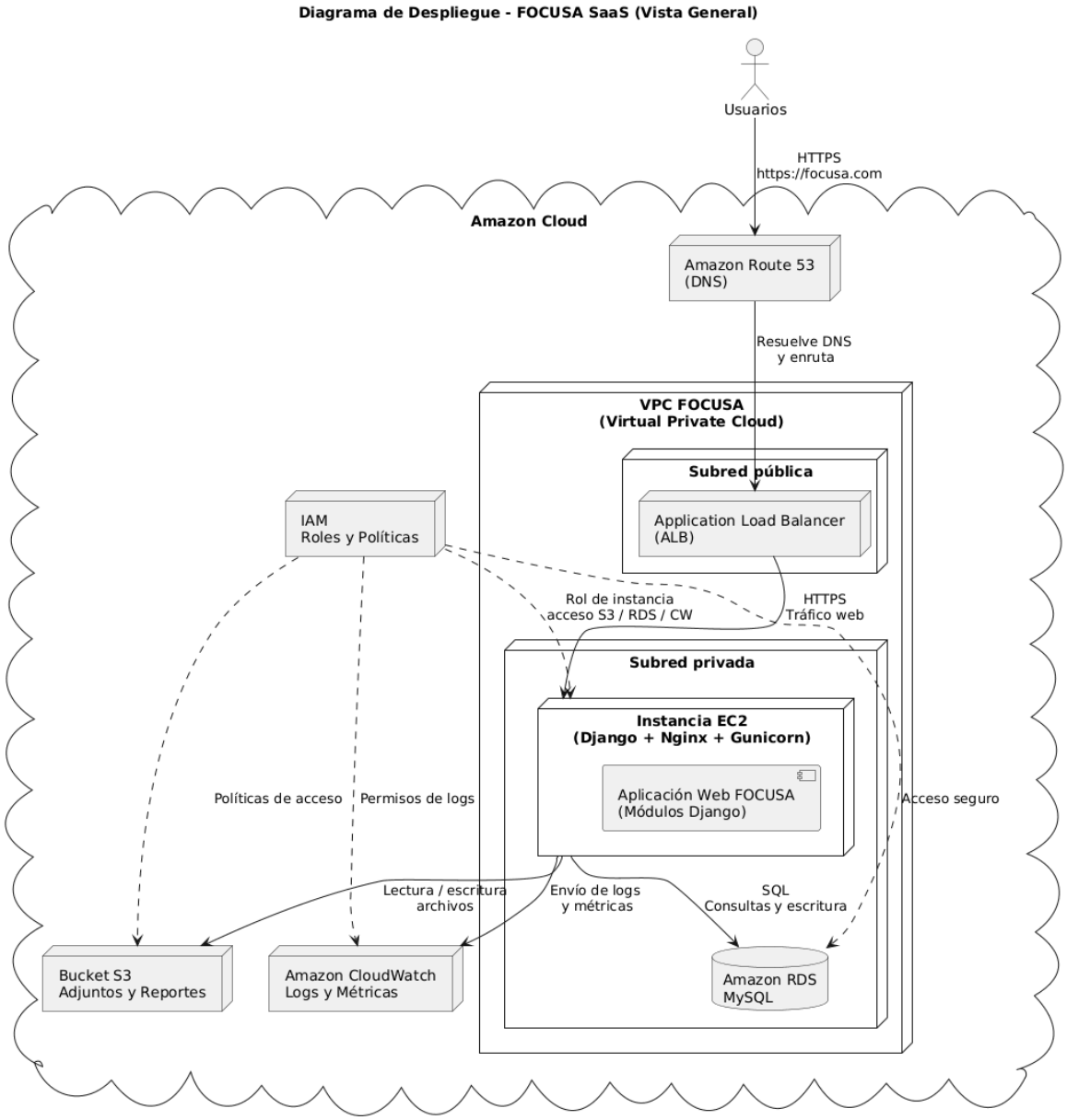


ilustración 2: diagrama de despliegue módulo de usuarios

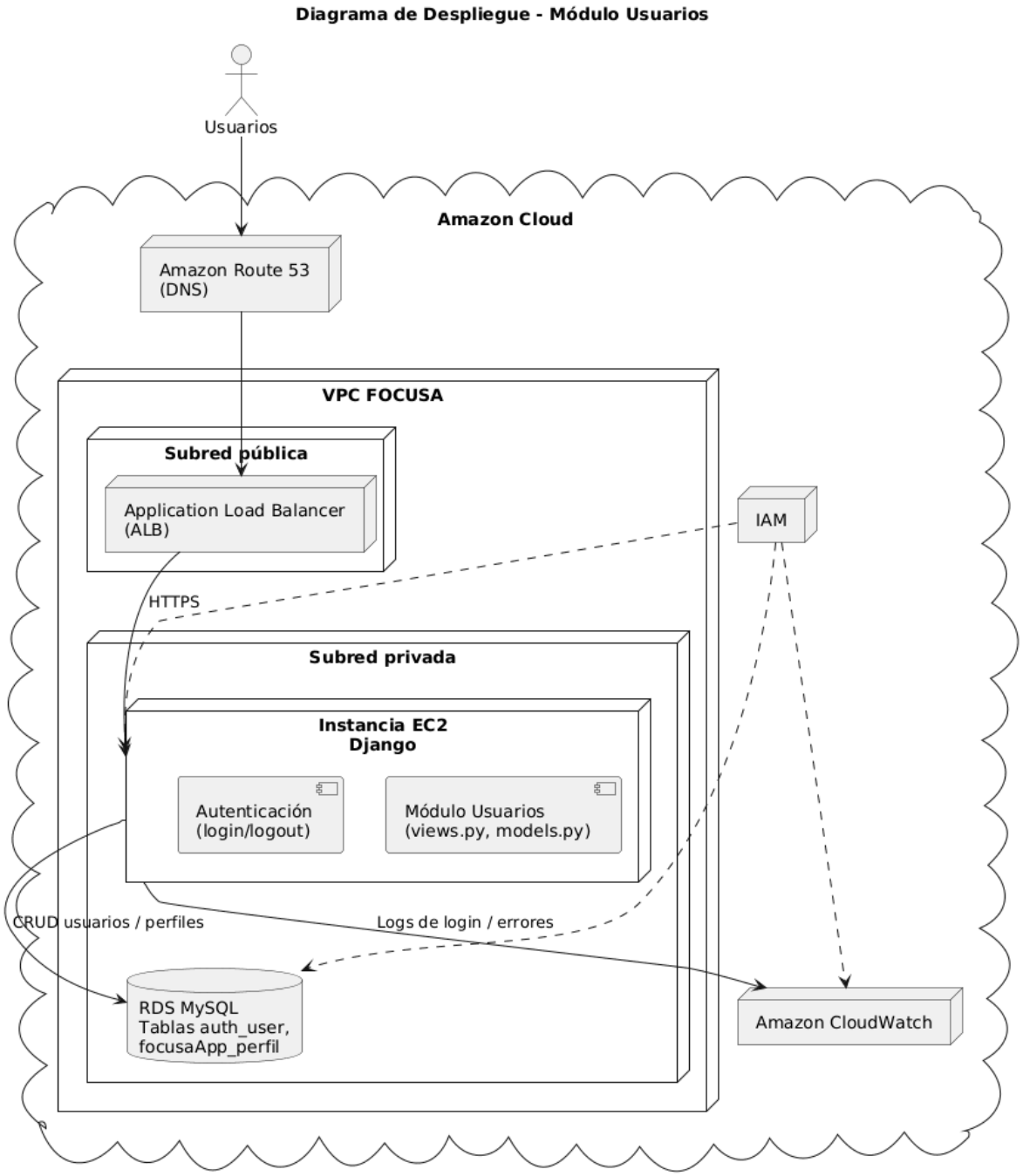


Ilustración 3: módulo Tareas, etiquetas, adjuntos

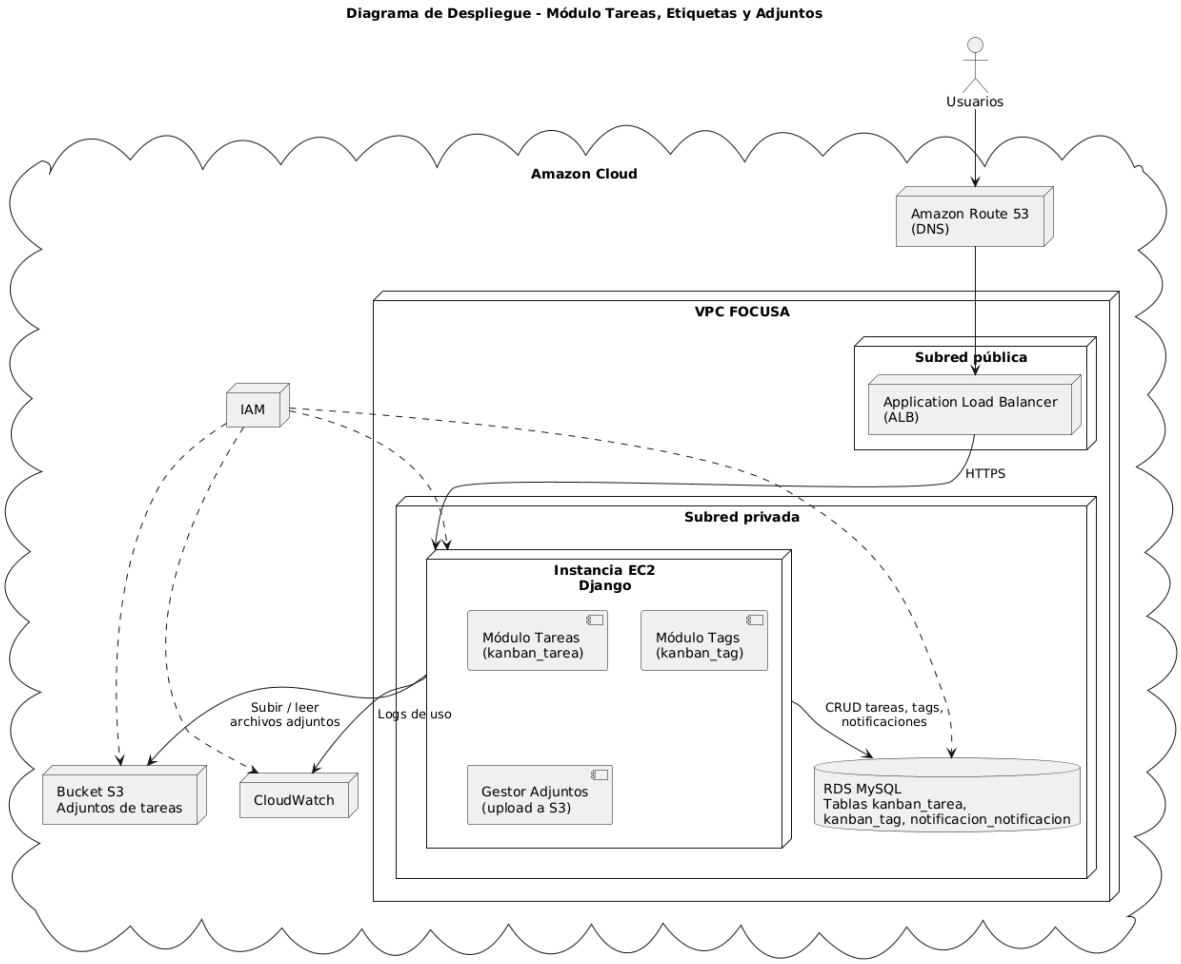
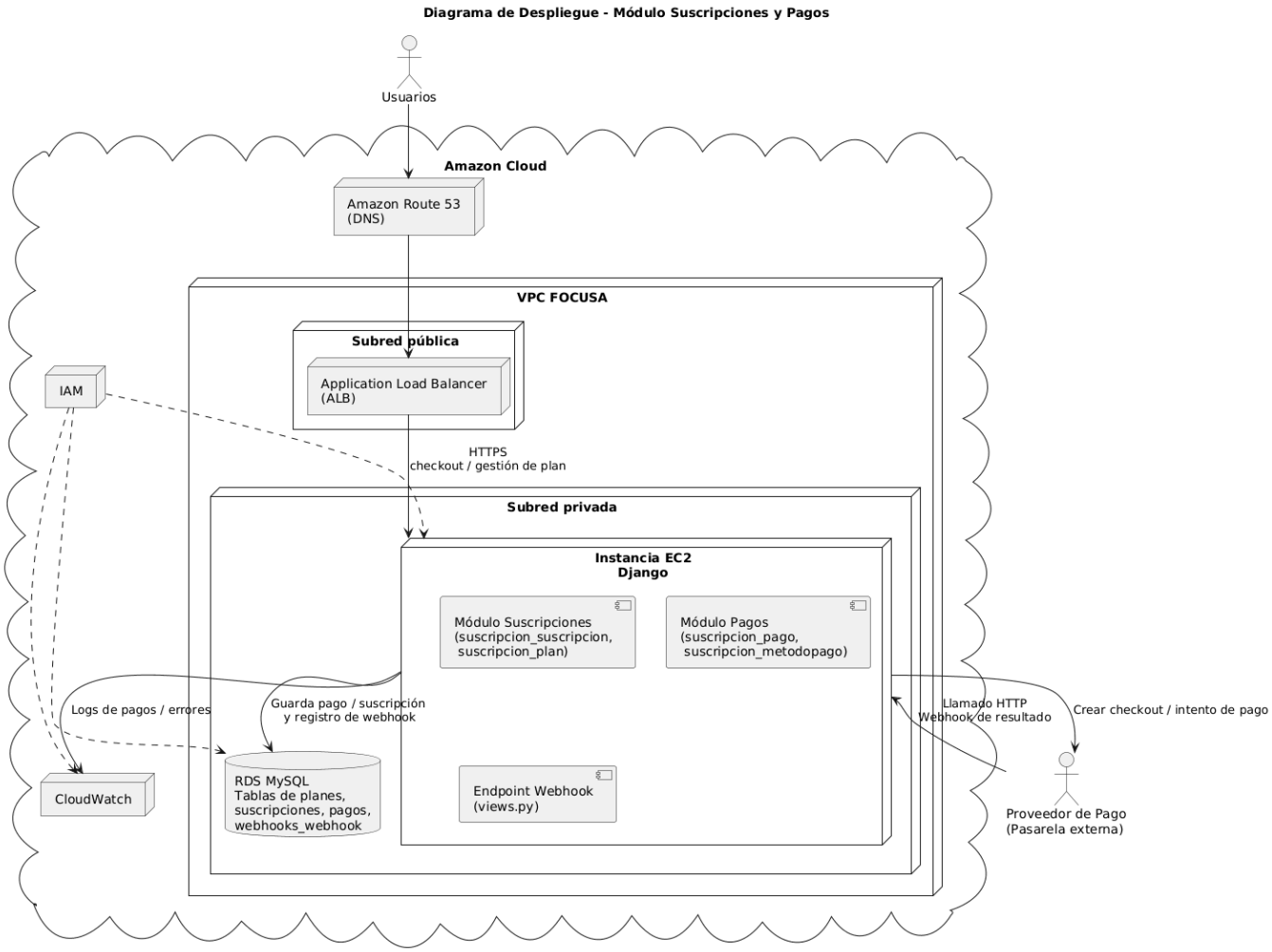


Ilustración 4: suscripciones y pagos



10. Decisiones de Diseño y Selección de Alternativas

Las decisiones arquitectónicas del sistema FOCUSA se tomaron considerando la restricción del tiempo de construcción y la necesidad de implementar una solución funcional, estable y de bajo riesgo. Dado que el desarrollo debía realizarse en un plazo acotado, se optó por tecnologías conocidas, confiables y que permitieran avanzar rápidamente sin comprometer la calidad.

Asimismo, la arquitectura se diseñó de forma modular, permitiendo que cada componente pueda ser desarrollado y probado de manera independiente. Esto reduce la probabilidad de fallas, facilita la mantención y permite la futura escalabilidad del sistema.

Las decisiones principales fueron:

– Uso del framework Django (Python):

Se seleccionó Django por ser un framework estable y ampliamente documentado, con una arquitectura clara basada en MTV (Model–Template–View). Permite avanzar rápido en el desarrollo manteniendo buenas prácticas, validaciones y seguridad básica.

– Base de Datos Relacional (mySQL):

Se eligió mySQL por su fiabilidad, fácil manejo y adecuado soporte para relaciones entre entidades como tareas, etiquetas, notificaciones y suscripciones.

– Arquitectura Modular:

El sistema se dividió en módulos independientes:

• Usuarios: registro, autenticación y gestión de cuentas.

• Tareas: creación, edición y seguimiento de tareas.

• Etiquetas: clasificación flexible de tareas.

• Reportes: generación de reportes en diferentes formatos.

• Suscripciones y Pagos: administración de planes y transacciones.

• Notificaciones: avisos internos del sistema.

• Backoffice: panel para la administración de usuario.

– Desarrollo basado en plantillas HTML reutilizables:

Se utilizó una estructura base común (`base.html`) para garantizar homogeneidad visual en todas las vistas.

– Despliegue adaptable:

La arquitectura permite operación tanto en un entorno local de desarrollo como en un ambiente de producción tradicional sin cambios significativos.

11. Análisis de Reutilización

Durante el desarrollo de FOCUSA se consideró la reutilización de componentes con el fin de optimizar los tiempos y mantener uniformidad en toda la aplicación. Esto se logró reutilizando tanto elementos de código como plantillas visuales.

Reutilización de código:

• Se reutilizó la lógica CRUD del módulo de tareas para implementaciones similares como etiquetas.

• Se replicaron estructuras de controladores (views) y modelos, ajustándolos únicamente a las necesidades de cada módulo.

• Las validaciones y uso del ORM siguieron un patrón común para mantener coherencia.

Reutilización de interfaz gráfica:

• Todas las vistas utilizan plantillas basadas en `base.html`.

• Se reutilizaron componentes gráficos como formularios, listas, tarjetas y botones, canvas (calendario) y otras librerías para la creación de de gráficos, apexcharts.

• Los módulos comparten la misma paleta de colores y estilos generales.

Beneficios de la reutilización:

• Reducción de tiempos de desarrollo.

• Menor probabilidad de errores al reutilizar código ya probado.

• Mayor coherencia visual y estructural en la aplicación.

• Facilidad de mantenimiento y escalabilidad futura.