**Plan de Arquitectura de Software con ATAM**

**Proyecto SGCalerías**

**Fecha*: 14-9-2024***

Control de Versiones

| Versión | Autor | Descripción |
| --- | --- | --- |
| 1.0 | Equipo de desarrollo | Creación |
|  |  |  |

***Contenido***

*1.* ***Introducción***

*1.1. Objetivo del Cliente ( Funcional )*

*1.2. Propósito del Sistema*

*1.3. Resumen Ejecutivo*

*2.* ***Fase I: Presentación***

*2.1. ¿Qué es ATAM?*

*2.2. Objetivo del Negocio ( No Funcional )*

*2.3. Presentar Arquitectura ( Enfoques sin analizar )*

*3.* ***Fase II: Investigación y Análisis***

*3.1. Propuestas Arquitectónicas sin evaluar*

*3.2. Árbol de Utilidad*

*3.3. Ranking de Arbol de utilidad*

*3.4. Análisis de Propuestas*

*4.* ***Fase III: Testing***

*4.1. Revisión y priorización del árbol de utilidad*

*5.*  ***Conclusión***

**Introducción**

La Arquitectura de Software establece las bases para construir un sistema que cumpla con los requerimientos funcionales y no funcionales, influenciando directamente atributos como desempeño, seguridad y mantenibilidad. El método Architecture Tradeoff Analysis Method (ATAM) permite evaluar arquitecturas de software basándose en los atributos de calidad definidos para el sistema. A continuación, se presentan las características de la arquitectura desarrollada para SGCalerías, una solución destinada a la gestión eficiente de un condominio.

**Fase 1: Presentación:**

**Objetivos del Negocio**

El objetivo principal del sistema **SGCalerías** es ofrecer una herramienta digital que facilite la administración de condominios, optimizando procesos relacionados con la gestión de gastos comunes, convenios de pago, ventas internas y comunicación entre residentes y administradores. Los atributos de calidad clave para alcanzar este objetivo incluyen:

* **Disponibilidad**: Garantizar el acceso al sistema 24/7 para administradores y residentes, asegurando continuidad en sus operaciones críticas.
* **Desempeño (Rapidez)**: Responder de forma ágil a las acciones de los usuarios, minimizando tiempos de carga al visualizar datos o realizar transacciones.
* **Interoperabilidad**: Integrar servicios externos, como pasarelas de pago (Webpay), para facilitar la gestión de pagos de cuotas y compras.
* **Confidencialidad**: Proteger la información sensible de los residentes y del condominio, cumpliendo con estándares de seguridad y privacidad.
* **Escalabilidad**: Garantizar que el sistema pueda ampliarse para gestionar múltiples condominios en el futuro.

El sistema busca simplificar y digitalizar tareas administrativas, mejorando la experiencia tanto de los residentes como de los administradores.

**Presentar Arquitectura**

La arquitectura de **SGCalerías** ha sido diseñada para soportar la gestión integral del condominio, con módulos funcionales específicos que incluyen:

* **Inicio de Sesión y Gestión de Usuarios**: Proveer autenticación y permisos diferenciados para residentes, administradores y otros usuarios.
* **Gestión de Gastos Comunes**: Registro, cálculo y visualización de gastos comunes asociados a las propiedades, permitiendo su prorrateo.
* **Convenios de Pago**: Permitir a los residentes establecer acuerdos para regularizar deudas, con aprobación del administrador.
* **Ventas Internas**: Implementar un catálogo en línea para la venta de productos del condominio, integrando opciones de carrito de compras y pago.
* **Reportes y Estadísticas**: Generar informes personalizados sobre gastos, pagos y transacciones para análisis administrativo.
* **Notificaciones Automáticas**: Informar a los usuarios sobre transacciones, vencimientos y actualizaciones mediante correo electrónico.

### Arquitectura Propuesta

Para cumplir con los objetivos mencionados, **SGCalerías** adopta una arquitectura **cliente-servidor** basada en tres capas principales:

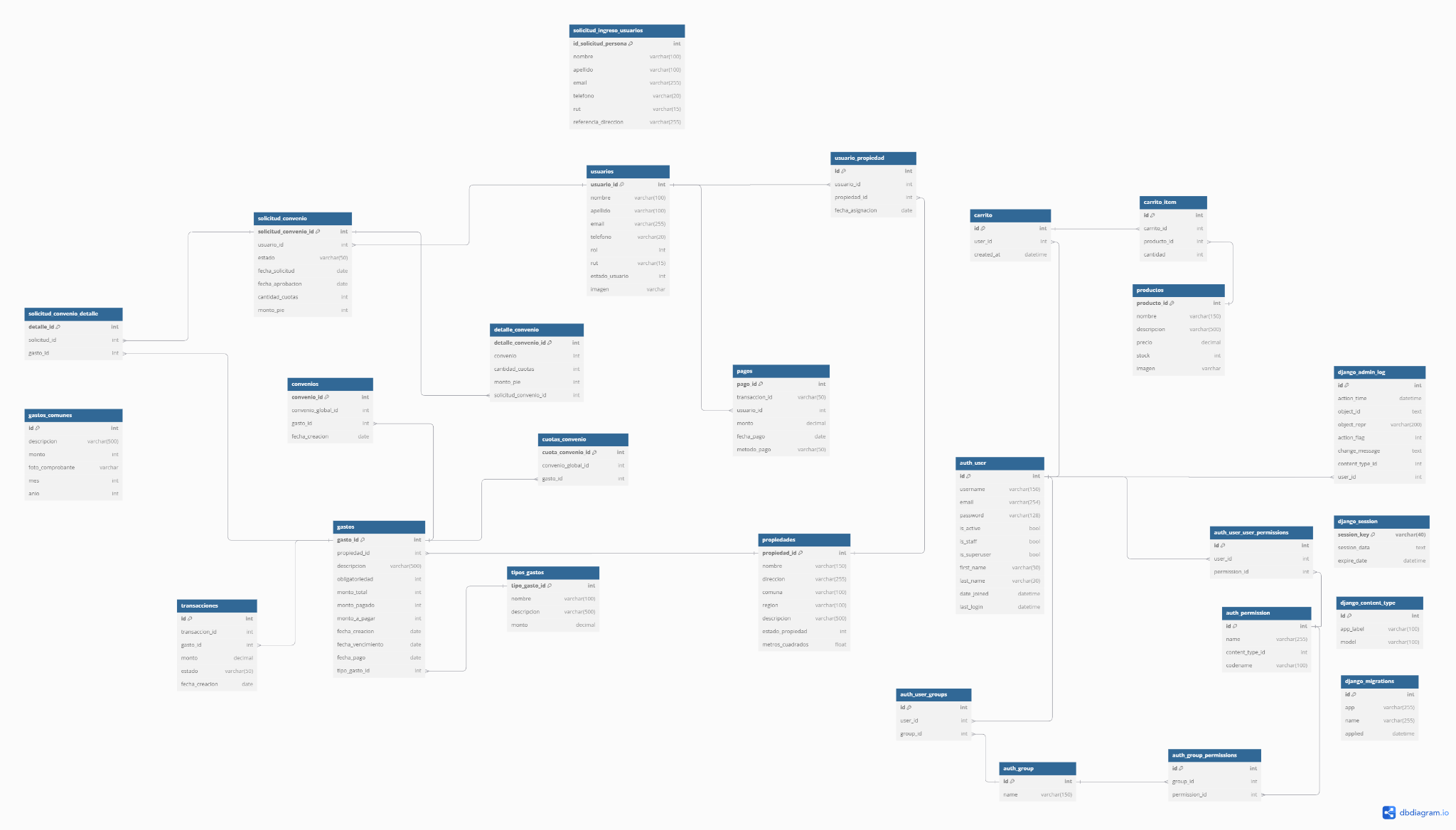
1. **Capa de Presentación**:
   * Frontend diseñado en HTML, CSS, JavaScript y Bootstrap para ofrecer una experiencia de usuario intuitiva y adaptable a diferentes dispositivos.
2. **Capa de Lógica de Negocio**:
   * Backend desarrollado con Django, implementando la lógica de gestión de usuarios, cálculos de prorrateo y transacciones.
3. **Capa de Persistencia**:
   * Base de datos relacional para almacenar de forma segura la información de residentes, propiedades, transacciones y reportes.

Además, la arquitectura incluye la integración con servicios externos:

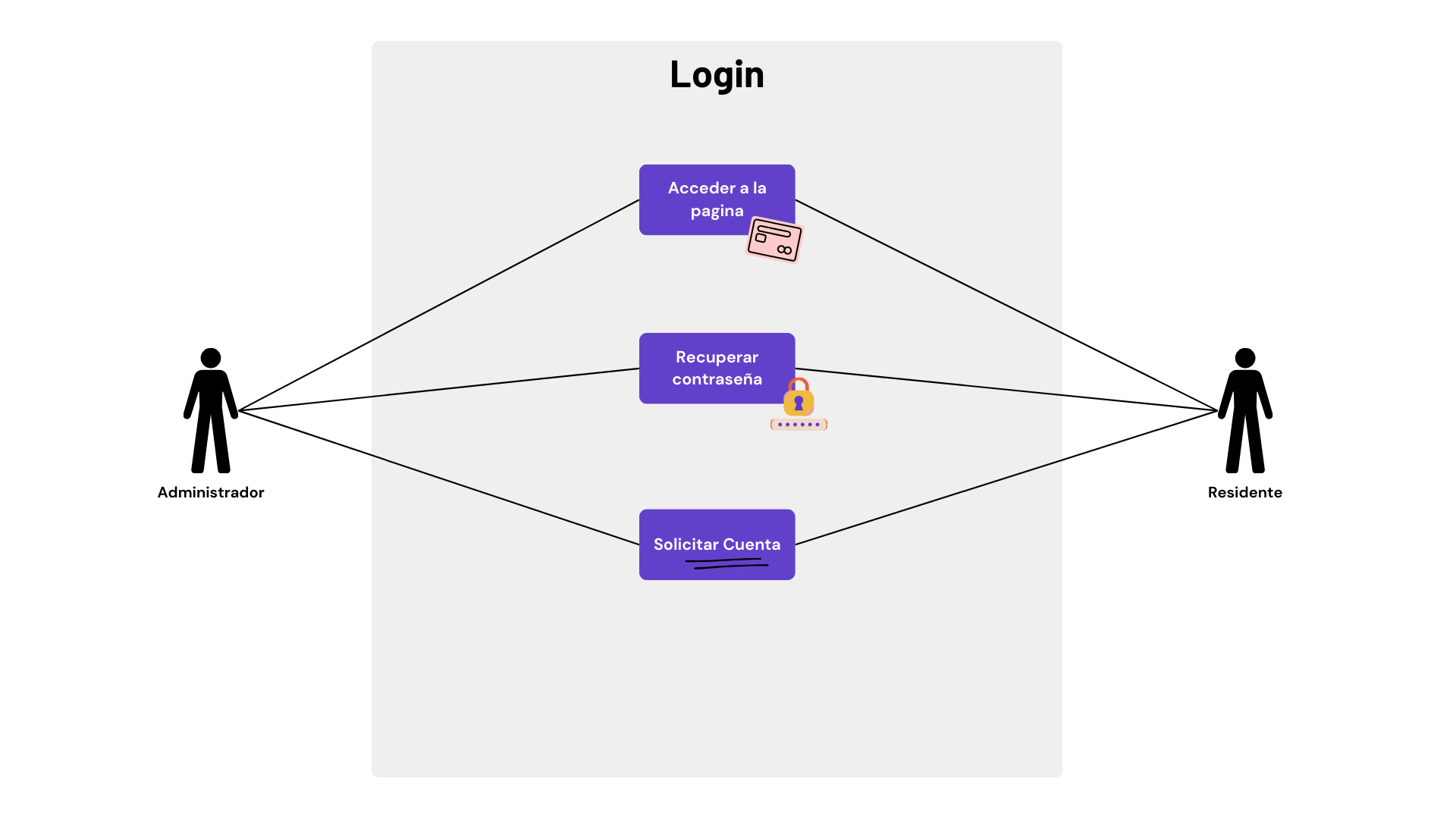
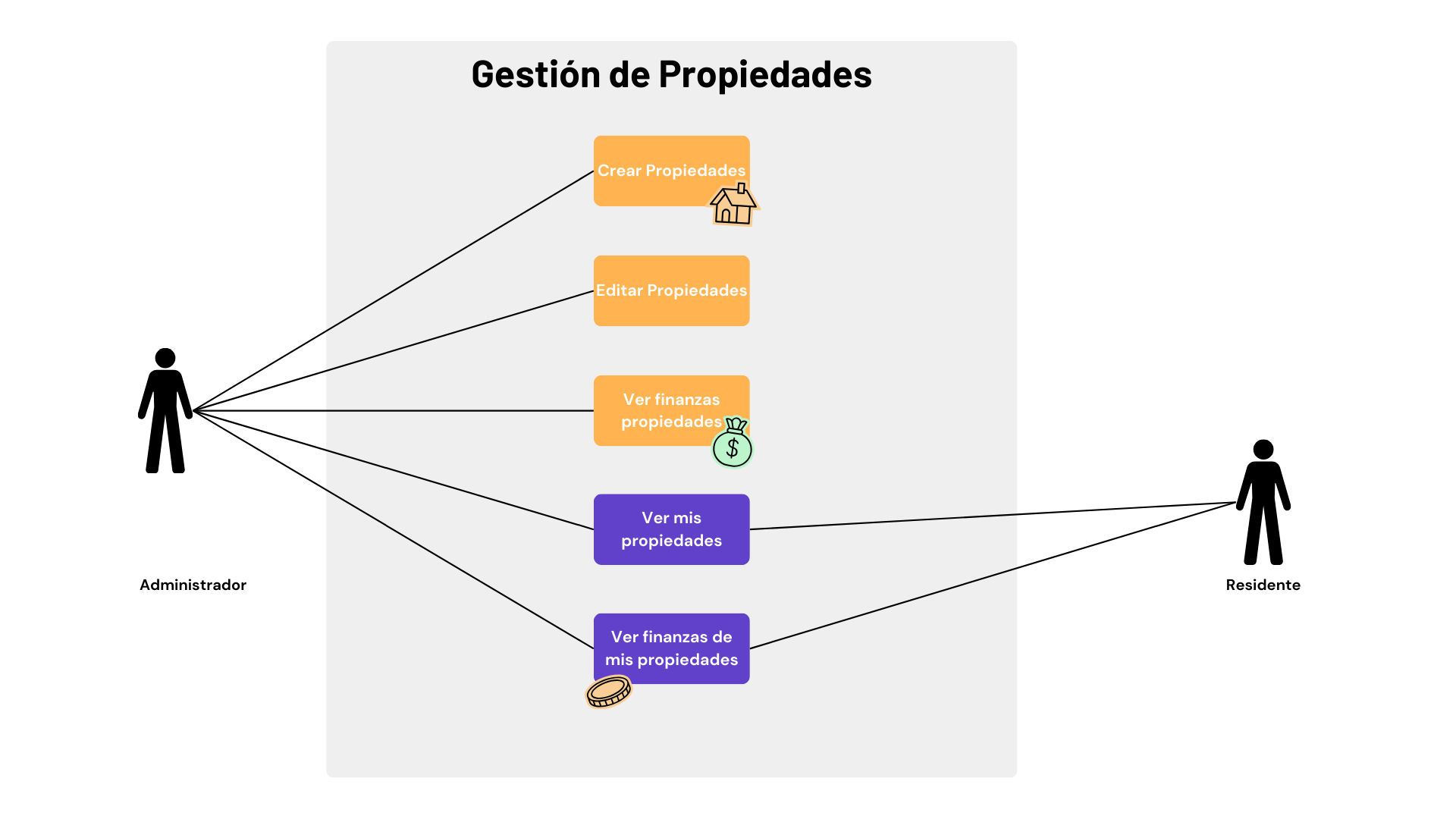
* **Pasarela de Pago Webpay** para pagos de cuotas y productos.
* **Servicio de Correo Electrónico** para envío de notificaciones y confirmaciones.
* **Servicios de Respaldo y Seguridad** para garantizar la integridad y protección de los datos.

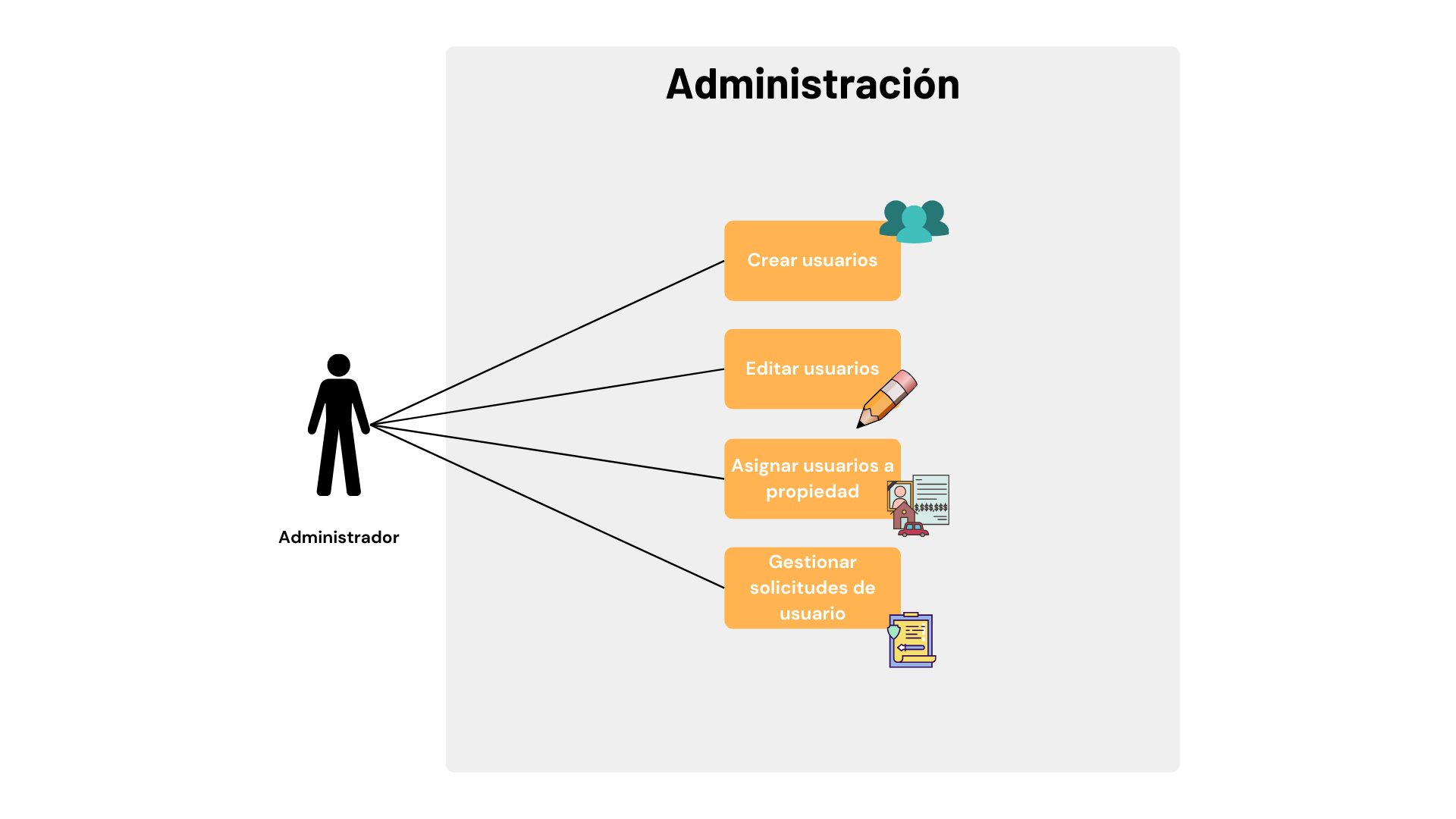
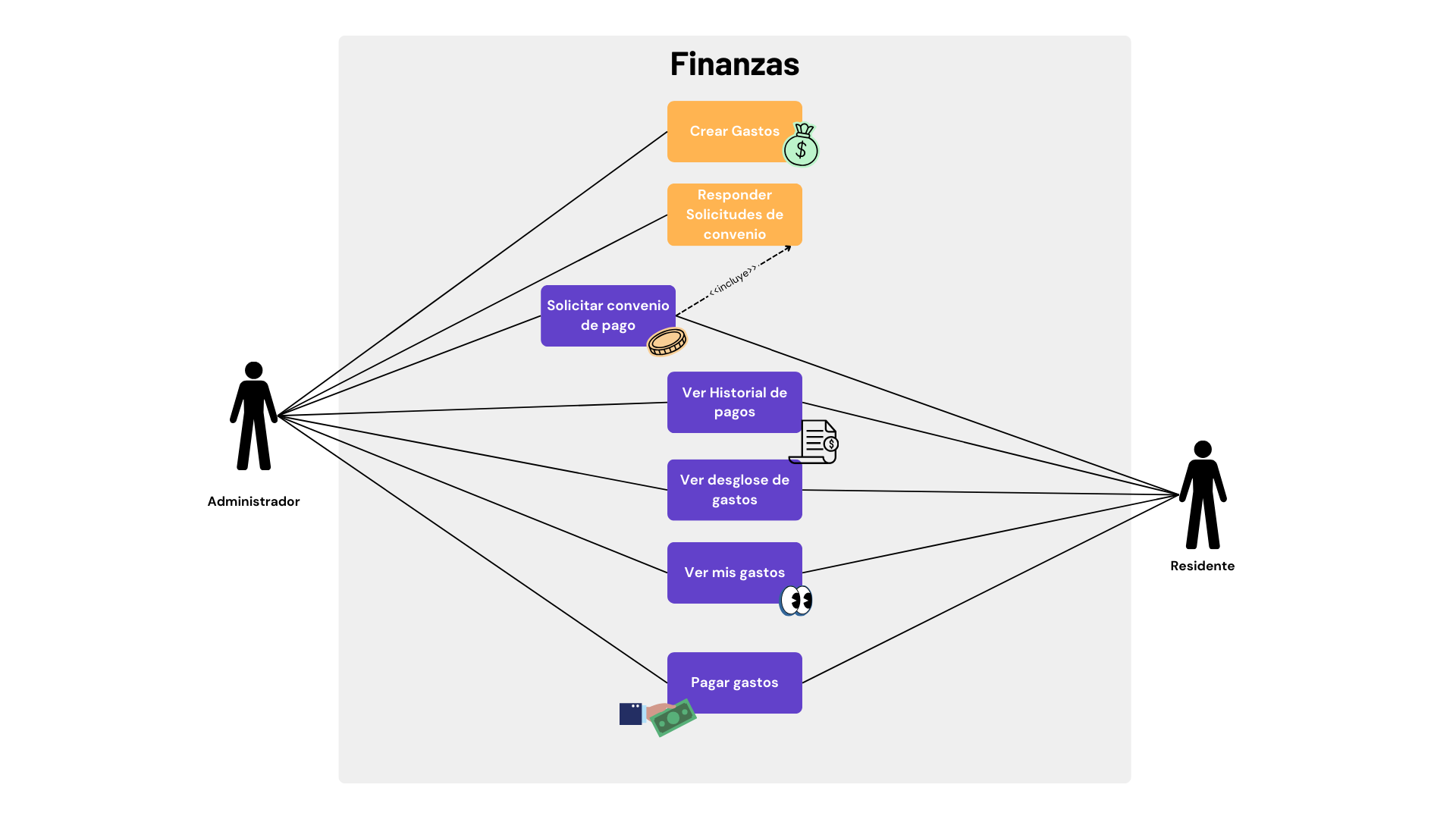
Esta arquitectura modular asegura la adaptabilidad del sistema a futuros cambios y su capacidad para manejar un creciente número de usuarios y condominios.

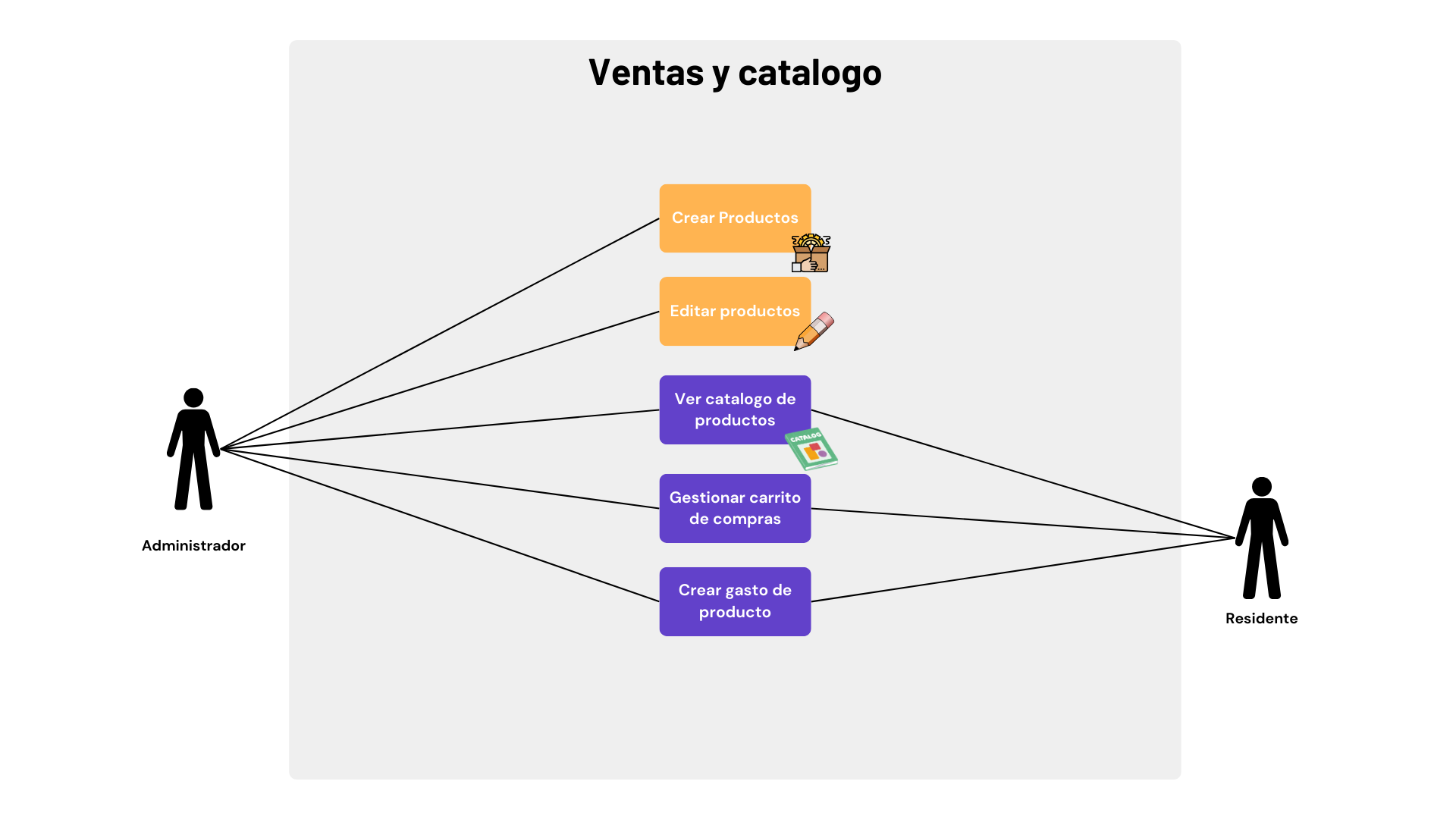
* ***Diagrama Entidad-Relación***

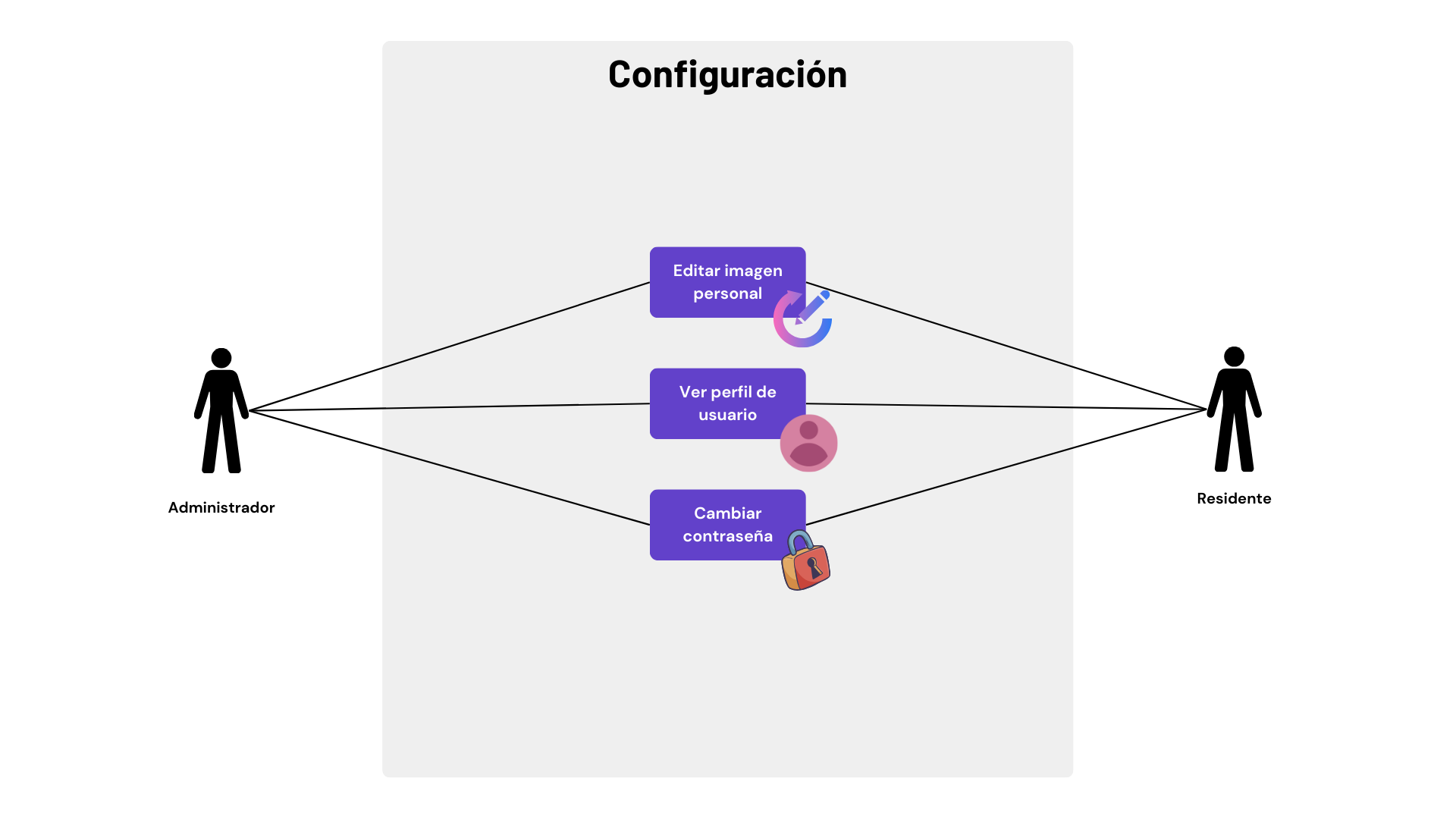
****

**Casos de Uso:**

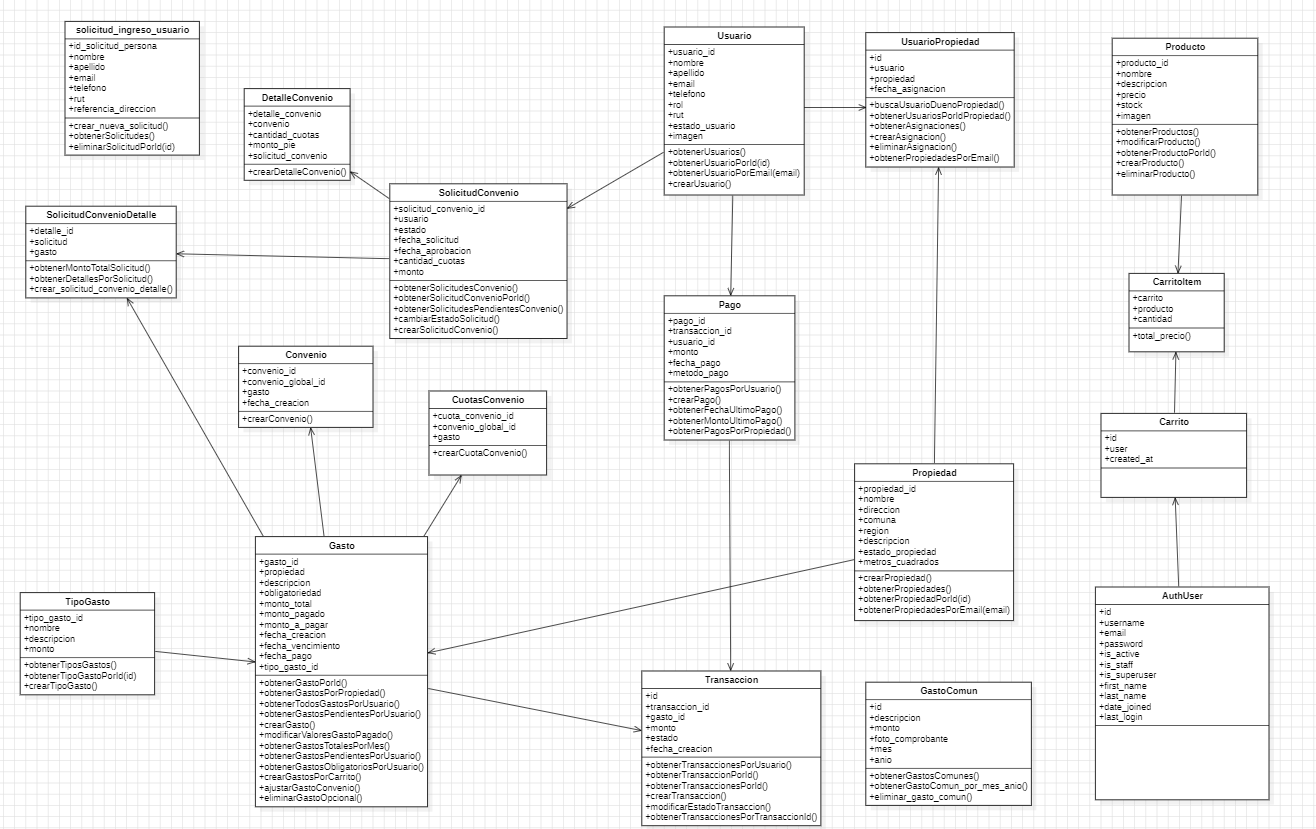
Los casos de uso son en gran parte CRUDs realizables por los clientes, empleados y administradores dependiendo del área, con los cuales se realizará la generación de informes a fin de mes para visualizar las ganancias y pérdidas de la empresa.







**Diagrama de Clases:**

****

### 

### Fase 3: Identificación y Análisis

#### 3.1 Identificar los enfoques (patrones) arquitectónicos

Los enfoques arquitectónicos seleccionados para el desarrollo de **SGCalerías** están alineados con los objetivos de negocio y los atributos de calidad identificados.

Para este sistema, se empleará el **patrón arquitectónico Modelo-Vista-Template (MVT)**, una adaptación del patrón **Modelo-Vista-Controlador (MVC)**, propio del framework Django. Este patrón organiza el sistema en tres componentes principales:

1. **Modelo (Model):**Representa la estructura y gestión de los datos, utilizando el ORM de Django para interactuar con la base de datos de manera eficiente. Esto asegura la escalabilidad y confidencialidad de la información, cumpliendo con el objetivo de mantener los datos seguros.
2. **Vista (View):**Contiene la lógica de negocio, gestionando las interacciones entre el modelo y las plantillas. Esto permite procesar solicitudes de los usuarios y devolver las respuestas apropiadas, asegurando un desempeño ágil y tiempos de respuesta mínimos.
3. **Template:**Responsable de la presentación visual del sistema, generando interfaces dinámicas basadas en las plantillas HTML. Esto asegura una experiencia de usuario consistente y eficiente en diferentes dispositivos y sistemas operativos.

#### Atributos de calidad cubiertos por el diseño arquitectónico:

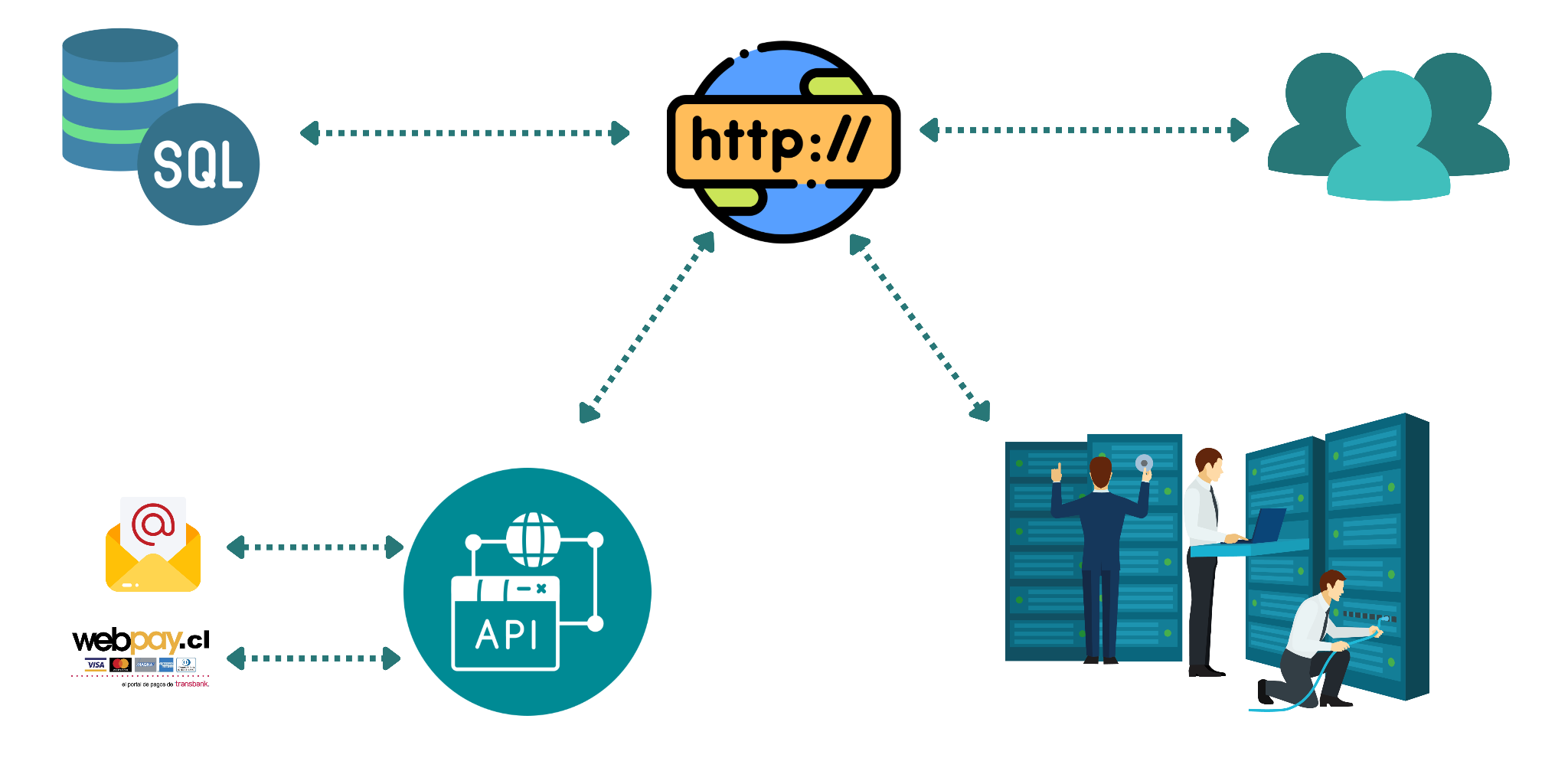
* **Disponibilidad:** El patrón MVT, en combinación con Django, soporta configuraciones en servidores que aseguran acceso al sistema 24/7.
* **Desempeño:** La estructura modular y la eficiencia del ORM de Django garantizan tiempos de respuesta rápidos.
* **Interoperabilidad:** La arquitectura facilita la integración con APIs y servicios externos esenciales, como pasarelas de pago y servicios de correo.
* **Confidencialidad:** La arquitectura permite implementar medidas de seguridad robustas, como autenticación, encriptación y validación de datos.

En conclusión, el uso del patrón MVT asegura que la arquitectura de **SGCalerías** sea eficiente, segura y preparada para satisfacer los objetivos de negocio y las necesidades de los usuarios.

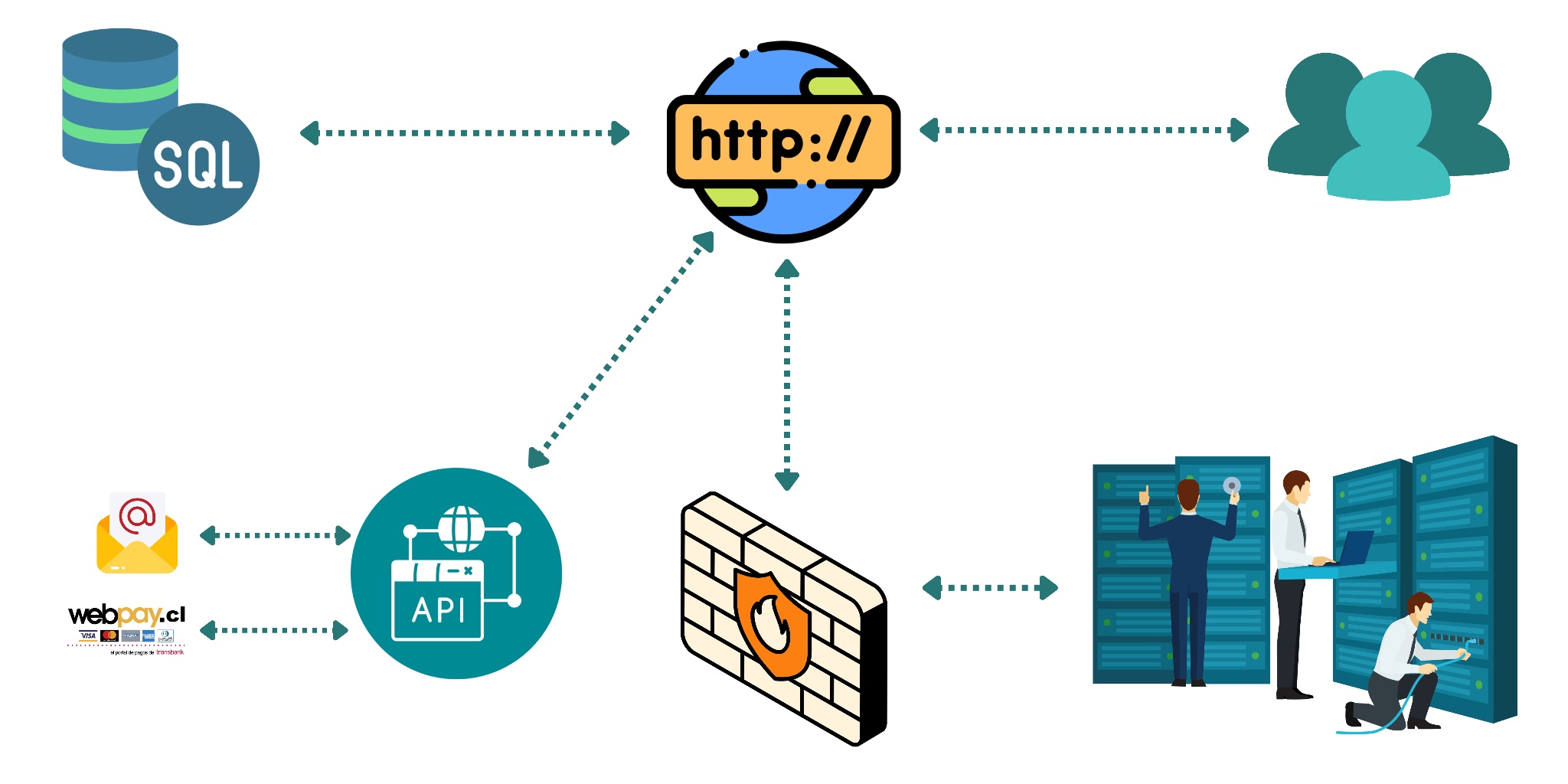
**3.2 Generar el árbol de utilidad:** Los factores de calidad (fiabilidad, disponibilidad, rendimiento…) son identificados y especificados hasta el nivel de escenarios, con estímulos y respuestas y después priorizados. Aplican los árboles de Utilidad de ATAM para permitir analizar los escenarios de calidad.

| Atributo de calidad | Subcaracteristica | Requisito | Escenario |
| --- | --- | --- | --- |
| Fiabilidad | Disponibilidad | Fiabilidad del sistema > 95% | A/M El sistema debe tener una disponibilidad superior o igual al 95% |
| Compatibilidad | Interoperabilidad | Integración sistemas de pago 100% | M/M El sitio web debe tener una integración de webpay para realizar las ventas a través del computador. |
| Interoperabilidad | Adaptabilidad | Disponibilidad Windows y Macintosh 100% | A/B El sitio web debe tener su acceso disponible para sistemas de Microsoft como de Apple |
| Eficiencia de Desempeño | Comportamiento temporal | Tiempo de respuesta < 2 segundos | M/A Tras realizar una acción el sistema debe demorar menos de 2 segundos en responder |
| Eficiencia de Desempeño | Comportamiento temporal  (Informes) | Tiempo de respuesta < 6 segundos | M/A Tras realizar un informe del negocio el sistema debe demorar menos de 6 segundos en responder |
| Usabilidad | Estética de la interfaz de usuario | Uso de colores de la empresa SGCalerías | B/B El sistema deberá estar constituido con los colores representativos de la empresa |
| Seguridad | No repudio | Evitar eliminar datos y en su lugar desactivarlos | A/A Capacidad del sistema de almacenar las acciones realizadas en el tiempo para evitar errores en el sistema |
| Usabilidad | Aprendizaje | Aprender a usar el sistema en menos de 3 días | M/A Un usuario debería poder aprender a ocupar el sistema en un corto periodo de tiempo |
| Usabilidad | Capacidad para ser Usado | Disposición intuitiva de botones | M/M La interfaz debe ser llamativa e intuitiva con el usuario , para mejorar su nivel interactivo. |
| Seguridad | Confidencialidad | Validación de datos para el ingreso del sistema | A/A El sistema debe brindar protección contra el acceso de datos e información no autorizados, ya sea accidental o deliberadamente. |

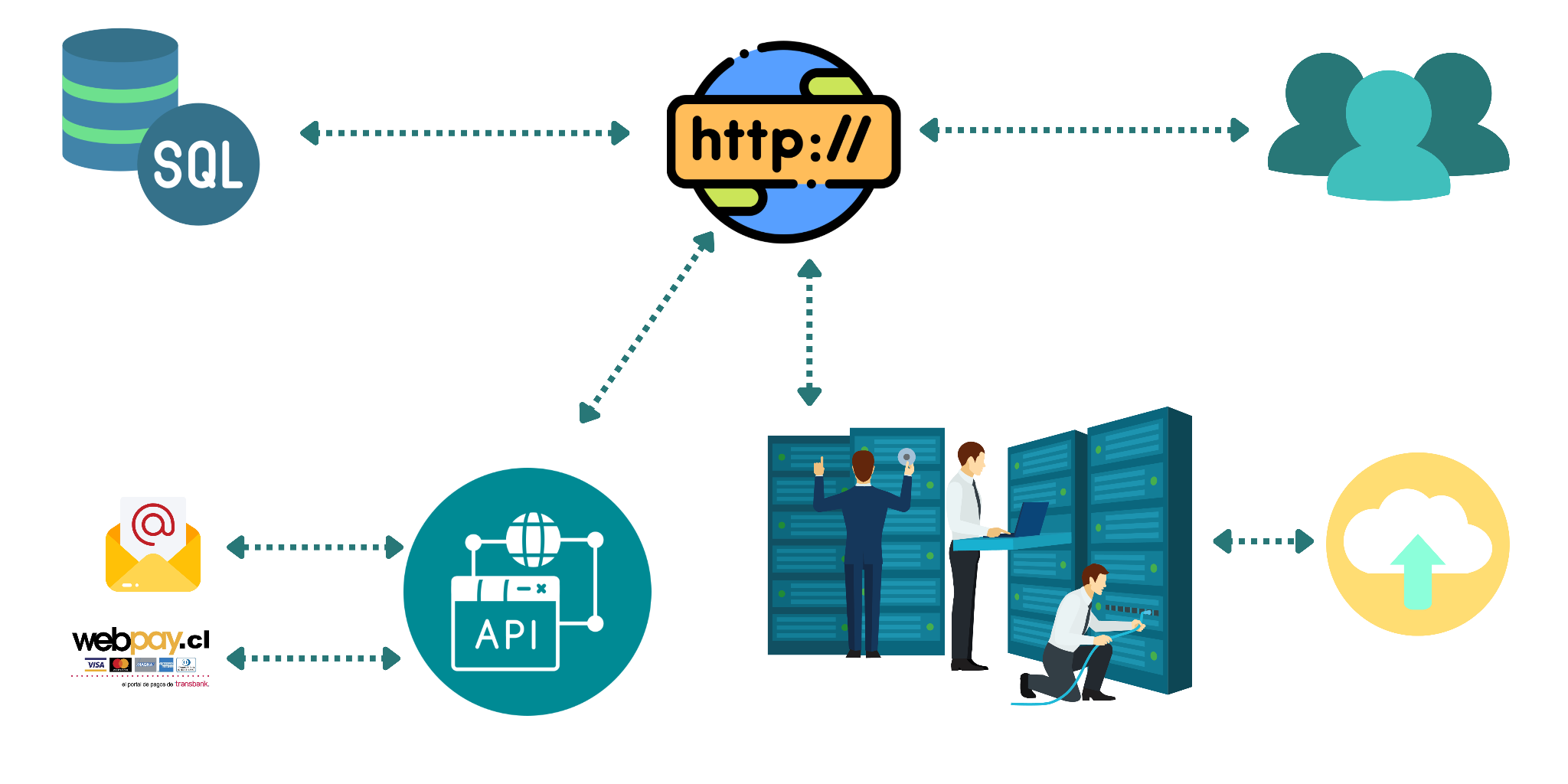
**3.3 Analizar los enfoques arquitectónicos:** los stakeholders y el arquitecto analizan cómo los enfoques arquitectónicos afectan a los factores identificados en el paso anterior

**3.3.1 Enfoque 1:**

**3.3.2 Enfoque 2:**



**3.3.3 Enfoque 3:**



### Argumentación para seleccionar la primera arquitectura

Aunque las propuestas presentadas en los enfoques 2 y 3 incluyen elementos como firewalls, backups y otros mecanismos de mejora, la selección de la primera arquitectura puede justificarse por las siguientes razones:

1. **Simplicidad y rapidez de implementación**
   * La primera arquitectura, al no incluir elementos como firewalls ni sistemas de respaldo, es inherentemente más sencilla y rápida de desplegar. Esto puede ser crucial si se busca llevar a cabo un proyecto piloto o validar la funcionalidad básica del sistema con recursos limitados.
2. **Optimización de recursos iniciales**
   * Implementar firewalls y sistemas de backup en las fases iniciales puede representar costos adicionales y una mayor complejidad técnica. Si el objetivo inicial es probar el concepto o realizar un despliegue a pequeña escala, estos mecanismos podrían posponerse hasta que el sistema madure y se disponga de un presupuesto mayor.
3. **Flexibilidad para iterar**
   * Al mantener una arquitectura más básica, el equipo de desarrollo puede concentrarse en mejorar la funcionalidad principal del sistema. Esto también permite recolectar datos reales sobre el rendimiento y las necesidades de seguridad/fiabilidad antes de decidir cómo y dónde incorporar elementos más avanzados.
4. **Respeto por el contexto operativo actual**
   * Aunque la adición de firewalls y respaldos incrementales es deseable, su ausencia en esta fase inicial no significa un riesgo inmediato insuperable. Por ejemplo, los comentarios de Víctor Otero y Víctor Bustos resaltan la importancia de la fiabilidad y disponibilidad, pero estas medidas pueden implementarse progresivamente una vez que se garantice el funcionamiento básico.

En resumen, la primera arquitectura se selecciona por su capacidad de ofrecer una solución funcional mínima viable, centrada en optimizar tiempo, costos y flexibilidad inicial. Los componentes de fiabilidad y seguridad se consideran importantes, pero su implementación se difiere para futuras iteraciones del sistema, en función de los datos obtenidos y los objetivos específicos de evolución del sistema.

**Priorización de escenarios:**

| **Atributo de calidad** | **Escenario** | **Necesidad** | **Dificultad** | **Total** |
| --- | --- | --- | --- | --- |
| Comportamiento temporal | A/A Tras realizar una acción el sistema debe demorar menos de 2 segundos en responder | 3 | 3 | 6 |
| No repudio | A/A Capacidad del sistema de almacenar las acciones realizadas en el tiempo para evitar errores en el sistema | 3 | 3 | 6 |
| Confidencialidad | A/A El sistema debe brindar protección contra el acceso de datos e información no autorizados, ya sea accidental o deliberadamente. | 3 | 3 | 6 |
| Fiabilidad | A/M El sistema debe tener una disponibilidad superior o igual al 95% | 3 | 2 | 5 |
| Comportamiento temporal (Informes) | M/A Tras realizar un informe del negocio el sistema debe demorar menos de 6 segundos en responder | 2 | 3 | 5 |
| Accesibilidad | M/A Tras realizar una acción el sistema debe demorar menos de 6 segundos en responder | 2 | 3 | 5 |
| Aprendizaje | M/A Un usuario debería poder aprender a ocupar el sistema en un corto periodo de tiempo | 2 | 3 | 5 |
| Interoperabilidad | M/M El sitio web debe tener una integración de webpay para realizar las ventas a través del computador. | 2 | 2 | 4 |
| Adaptabilidad | A/B El sitio web debe tener su acceso disponible para sistemas de Microsoft como de Apple | 3 | 1 | 4 |
| Capacidad para ser Usado | M/M La interfaz debe ser llamativa e intuitiva con el usuario, para mejorar su nivel interactivo. | 2 | 2 | 4 |
| Estética de la interfaz de usuario | B/B El sistema deberá  estar constituido con los colores representativos de la empresa | 1 | 1 | 2 |