**[Compra & Ahorra]**

**(SAD) Software Architecture Document**

**Versión 1.1**

**Identificación de Documento**

| **Identificación** |  |
| --- | --- |
| **Proyecto** | Byte & Beat |
| **Versión** | 1.1 |

| **Documento mantenido por** | Ruben Mansilla Villena |
| --- | --- |
| **Fecha de última revisión** | 07-03-2025 |
| **Fecha de próxima revisión** | 10-03-2025 |

| **Documento aprobado por** | Betel Villena |
| --- | --- |
| **Fecha de última aprobación** | 08-03-2025 |

**Historia de Revisiones**

| **Fecha** | **Versión** | **Descripción** | **Autor** |
| --- | --- | --- | --- |
| 07-03-2025 | 1.1 | Inicio de toma de requisitos con el cliente | Ruben Mansilla |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Tabla de Contenidos**

[**1**](#_heading=h.gjdgxs) **Introducción 3**

[1.1](#_heading=h.30j0zll) Contexto del Problema 3

[1.2](#_heading=h.1fob9te) Propósito 3

[1.3](#_heading=h.dre6zxhju4ka) Ámbito 3

[1.4](#_heading=h.2et92p0) Definiciones, acrónimos y abreviaciones 3

[1.5](#_heading=h.tyjcwt) Referencias 3

[1.6](#_heading=h.3dy6vkm) Resumen ejecutivo 4

[**2**](#_heading=h.3whwml4) **Representación de la Arquitectura 5**

[2.1](#_heading=h.1t3h5sf) Representación 5

[**3**](#_heading=h.4d34og8) **Metas y Restricciones de la Arquitectura 6**

[A continuación se revisan las metas y restricciones de la arquitectura. 6](#_heading=h.2s8eyo1)

[3.1](#_heading=h.17dp8vu) Metas de la arquitectura 6

[3.2](#_heading=h.3rdcrjn) Restricciones de la Arquitectura 6

[3.3](#_heading=h.26in1rg) Otros antecedentes y consideraciones 6

[**4**](#_heading=h.lnxbz9) **Vista de Casos de Uso y Escenarios de Calidad 8**

[4.1](#_heading=h.35nkun2) Modelo de Casos de Uso 8

[4.2](#_heading=h.1ksv4uv) Especificación de Casos de Uso Relevantes 8

[4.3](#_heading=h.44sinio) Especificación de los Escenarios de Calidad Relevantes 9

[**Vista Lógica 11**](#_heading=h.2jxsxqh)

[**5**](#_heading=h.1y810tw) **Vista de Procesos 13**

[**4**](#_heading=h.4i7ojhp) **Vista de Implementación 14**

[**5**](#_heading=h.2xcytpi) **Vista de Despliegue 15**

[**6**](#_heading=h.1ci93xb) **Decisiones de Diseño y Selección de Alternativas 16**

1. **Introducción**
   1. **Contexto del Problema**

El cliente es una pequeña empresa que actualmente gestiona sus ventas utilizando únicamente WhatsApp e Instagram. A pesar de su crecimiento, el proceso de ventas se ha mantenido de forma manual, lo que genera dificultades de control tanto para la empresa como operativas para los clientes. Esta falta de un sistema eficiente ha afectado la agilidad, la precisión en el manejo de pedidos y la capacidad de la empresa para escalar su modelo de negocio de manera efectiva.

* 1. **Propósito**

El propósito de este documento es definir la arquitectura del sistema web Byte & Beat, diseñado para mejorar la eficiencia y efectividad de la gestión de ventas, reportes financieros y facilitar el crecimiento de la empresa. El sistema propuesto tiene como objetivo principal de reemplazar el proceso manual y desorganizado de gestión de ventas, que actualmente se realiza únicamente por WhatsApp, mediante la implementación de una plataforma digitalizada que integre todas las operaciones de la empresa.

* 1. **Ámbito**El alcance de este documento de arquitectura principalmente abarca el diseño, desarrollo y despliegue de un sistema web de gestión de pedidos y reportes de ventas para la pequeña empresa Byte & Beat. Este sistema está pensado para optimizar y automatizar las operaciones comerciales, proporcionando una plataforma integral que facilita la toma de pedidos, la administración de inventarios y el análisis de las ventas de manera eficiente.  
       
     El sistema estará diseñado para operar en un entorno web, permitiendo el acceso desde múltiples dispositivos conectados a internet, como computadoras, tablets y smartphones. Dentro del alcance igualmente se encuentra el uso de infraestructura en la nube para el almacenamiento de los datos, asegurando la disponibilidad y escalabilidad de la información en un entorno seguro y confiable.

* 1. **Definiciones, acrónimos y abreviaciones**

| **Abreviatura** | **Descripción** |
| --- | --- |
| WebPay | Plataforma de pago en línea |
| BD | Base de datos |
| Hosting | Alojamiento de sitio web |
| Render | Servidor que alojara el sitio web |

* 1. **Referencias**
* **Acta Constitución**
* **ERS (Especificación de Requisitos)**
* **Visión y Roles**
* **Matriz de Requerimientos**
* **Cronograma**
  1. **Resumen ejecutivo**

Este documento describe la arquitectura del sistema web Byte & Beat, una plataforma diseñada para digitalizar y optimizar la gestión de pedidos y los procesos administrativos de una pequeña empresa que actualmente realiza pedidos mediante WhatsApp. Con el método actual presenta limitaciones en cuanto a agilidad y eficiencia, el sistema propuesto busca reemplazar este proceso manual, proporcionando una solución escalable y eficiente que permita el acceso en tiempo real a la información crítica para la toma de decisiones más rápida y precisas.

El diseño de la arquitectura sigue el enfoque 4+1, que estructurará el sistema en diferentes vistas: Lógica, Procesos, Despliegue y Física. Esto nos ayuda a asegurarnos que el sistema cumpla con los requisitos claves, como la seguridad de los datos, la facilidad de mantenimiento, la escalabilidad para soportar un volumen mayor de transacciones y la adaptabilidad a nuevas funcionalidades. Además, facilita el manejo eficiente y orden adecuado de los pedidos, mejorando la operación general del sistema.

El sistema se desarrollará utilizando Django como framework backend. La BD será PostgreSQL manejará el almacenamiento de la información y la integración con WebPay permitirá realizar transacciones electrónicas seguras. El servidor estará alojado en Render, que proporcionará el hosting para el sitio web en cambio la base de datos está alojada en un servidor de Supabase únicamente para almacenamiento de datos.

Lo que se espera en esta solución es la eficiencia operativa, reducción de errores en el manejo de inventarios, acceso rápido y seguro a los datos financieros y una experiencia de usuario mejorada tanto para clientes como para administrador.

* 1. **Representación**

La arquitectura del sistema web Byte & Beat está utilizando el modelo 4+1 y que nos dará una estructura clara y comprensiva desde diferentes perspectivas. Cada vista aborda aspectos específicos del sistema asegurando que los requisitos funcionales y de calidad sean cubiertos en el diseño y desarrollo.

* **Vista de Escenarios (Diagrama de Casos de Uso)**: Describe los casos de uso principales y los escenarios de calidad que influyen en la arquitectura. Esta vista establece el contexto funcional del sistema web al presentar interacciones entre actores y el sistema
* **Vista Lógica (Diagrama de Clases)**: Representa la estructura lógica del sistema mediante clases y relaciones. Esta vista muestra la organización de los módulos de software, sus responsabilidades y las dependencias lógicas entre ellos, formando la base del modelo de negocio
* **Vista de Procesos (Diagrama de Actividades)**: Define los flujos de procesos principales dentro del sistema web, asegurando que la ejecución de los procesos sea coherente y sincronizada. Esta vista se enfoca en los aspectos dinámicos y de concurrencia de la arquitectura.
* **Vista Física (Diagrama de Despliegue)**: Representa la disposición de los componentes físicos que soportan la arquitectura, incluyendo dispositivos de usuario y servidores. Esta vista garantiza que el sistema web cuente con la infraestructura necesaria para operar de forma eficiente y escalable.
* **Vista de Despliegue (Diagrama de Componentes)**: Describe cómo se implementan los componentes del sistema web en el entorno de producción y las dependencias entre ellos. Esta vista facilita la implementación y despliegue en un entorno de producción seguro y estable.

1. **Metas y Restricciones de la Arquitectura**

La meta principal es terminar todas las funcionalidades requeridas para que el software funcione correctamente según lo que solicitó el cliente, el sistema al finalizar sus funcionalidades, diferentes interfaces, integración de las APIs correspondientes. A lo largo de cada Sprint, se realizan pruebas continuas para garantizar que cada funcionalidad implementada esté libre de errores y cumpla con los requisitos del cliente.

La mayor restricción es el tiempo de plazo que se dio ya que este proyecto debe completarse dentro de un plazo de 4 meses desde el 1 de marzo de 2025 hasta el 20 de junio de 2025, incluyendo el despliegue final, la ejecución de plan de pruebas y las posibles correcciones que puedan surgir durante la fase de pruebas o post-lanzamientos. Es fundamental que el sistema esté perfectamente funcionando junto con su BD y con los datos necesarios para respaldar las ventas en el sitio web.

* 1. **Metas de la arquitectura**

De acuerdo con las reuniones y al análisis de los requerimientos, se listan los principales conductores iniciales de la arquitectura los cuales corresponden a las metas arquitectónicas iniciales:

* **Desempeño**: El Sistema debe ser capaz de manejar un alto volumen de transacciones y consultas de manera eficiente. Esto implica optimizar tanto la velocidad de respuesta de las aplicaciones como la capacidad de manejar múltiples usuarios simultáneamente sin afectar la calidad del servicio. Se buscará un rendimiento escalable, tanto en la BD como en el servidor web
* **Tolerancia a fallos**: La arquitectura debe estar diseñada para minimizar el impacto de los fallos en el sistema. Esto incluye la implementación de mecanismos de respaldo y recuperación ante posibles fallos del servidor, la base de datos o la red, garantizando que continúe funcionando o recupere su estado sin pérdida de datos.
* **Seguridad**: El Sistema maneja datos sensibles como información de usuarios y transacciones, es crucial que se implementen protocolos de seguridad. Esto incluye la protección contra accesos no autorizados, el cifrado de datos sensible y autenticación segura.
* **Modificabilidad/Reuso**: El Sistema debe ser modular y flexible, permitiendo modificaciones, ampliaciones y mejoras a futuros sin afectar significativamente la infraestructura existente. La arquitectura debe estar diseñada para permitir la adición de nuevas funcionalidades con el mínimo esfuerzo.
* **Operatividad**: El Sistema debe ser fácil de operar y mantener. Esto incluye la creación de interfaces de administración intuitivas para los usuarios que utilizaran el sistema, así como el seguimiento constante del sistema y la generación de alertas ante posibles problemas. Se debe garantizar que los tiempos de inactividad sean mínimos y que las actualizaciones y el mantenimiento se puedan realizar de manera fluida.
  1. **Restricciones de la Arquitectura**

Existen restricciones que han sido levantadas con los stakeholders, las cuales se presentan a continuación:

* **Tiempo de construcción**: Se cuenta un plazo estrecho ajustado de 4 meses. La gestión del tiempo será crucial para cumplir con los entregables en cada fase, como los sprints de desarrollo, las pruebas de integración y revisión con el cliente.
* **Infraestructura**: La infraestructura disponible incluye servidores de aplicación replicados para asegurar la alta disponibilidad y el rendimiento del sistema. Además, la BD PostgreSQL estará implementada en Supabase, es una estructura escalable que permitirá soportar el crecimiento de datos sin comprometer el rendimiento del sistema.
* **Otros componentes de software**: No se considera la adquisición de licencia ni incorporación de otros componentes de software comerciales en este proyecto. Todo el sistema será construido utilizando tecnologías open-source, como Django para el desarrollo backend, PostgreSQL para la base de datos y WebPay para el procesamiento de pagos.
  1. **Otros antecedentes y consideraciones**

La empresa desarrolladora cuenta con un framework que considera los siguientes componentes que permiten satisfacer los requerimientos arquitectónicos del sistema Byte & Beat:

* Django de inyección de dependencias, permite la encapsulación y modularizar los componentes del sistema, lo que facilita su mantenibilidad a largo plazo. Al utilizar este enfoque, se asegura que cada módulo o componente sea independiente y fácilmente actualizable o reemplazable sin afectar a otros módulos del sistema
* Django soporta la meta de seguridad del sistema, implementando prácticas de autenticación y autorización segura, además de ofrecer protección contra vulnerabilidades comunes como inyección SQL. Garantiza la protección de datos sensibles y facilita la implementación de cifrado de contraseñas y gestión segura de sesiones de usuario.

1. **Escenarios de Calidad**

| Escenario de Calidad N°1 | |
| --- | --- |
| Atributo de Calidad Asociado (Característica): Usabilidad  Sub Característica: Estética de la interfaz de usuario | |
| Descripción: El sistema debe permitir que los usuarios naveguen fácilmente por la interfaz sin necesidad de instrucciones adicionales. | |
| Diagrama  El contenido generado por IA puede ser incorrecto. | |
| Fuente del Estímulo: | Usuario Final |
| Estímulo: | Solicitud de mejora en la facilidad de uso |
| Artefacto: | Interfaz de usuario |
| Ambiente: | Producción |
| Respuesta: | La interfaz es evaluada mediante pruebas de usabilidad con usuarios reales. |
| Medida de Respuesta: | El 90% de los usuarios deben poder completar una gestión (compras, reporte de ventas, editar perfil) en menos de 2 minutos sin ayuda |

| Escenario de Calidad N°2 | |
| --- | --- |
| Atributo de Calidad Asociado (Característica): Eficiencia de desempeño  Sub Característica: Utilización de recursos | |
| Descripción: El sistema debe mantener un tiempo de respuesta inferior a 2 segundos bajo una carga normal de usuario | |
| Diagrama  El contenido generado por IA puede ser incorrecto. | |
| Fuente del Estímulo: | Sistema |
| Estímulo: | Uso concurrente de la plataforma |
| Artefacto: | Servidor de aplicación |
| Ambiente: | Producción |
| Respuesta: | Se mide el tiempo de respuesta del servidor bajo diferentes cargas. |
| Medida de Respuesta: | Tiempo de respuesta inferior a 2 segundos en el 95% de los casos entre 100 usuarios |

| Escenario de Calidad N°3 | |
| --- | --- |
| Atributo de Calidad Asociado (Característica): Compatibilidad  Sub Característica: Adaptabilidad | |
| Descripción: La aplicación debe funcionar correctamente en múltiples dispositivos y navegadores. | |
| Diagrama  El contenido generado por IA puede ser incorrecto. | |
| Fuente del Estímulo: | Usuario final |
| Estímulo: | Acceder al sitio web desde diferentes dispositivos y navegadores |
| Artefacto: | Aplicación Web |
| Ambiente: | Producción |
| Respuesta: | Debe ser accesible y funcional en los principales navegadores (Chrome, FireFox, Safari, Edge) y dispositivos (móviles, tablets, computadores) |
| Medida de Respuesta: | La aplicación debe pasar todas las pruebas de compatibilidad con un 95% de éxito |

| Escenario de Calidad N°4 | |
| --- | --- |
| Atributo de Calidad Asociado (Característica): Seguridad  Sub Característica: Responsabilidad | |
| Descripción: El sistema debe garantizar que las cuentas de usuario sean gestionadas de manera segura, con mecanismos de autenticación robustos | |
| Diagrama  El contenido generado por IA puede ser incorrecto. | |
| Fuente del Estímulo: | Administrador del sistema |
| Estímulo: | Gestión de cuentas y acceso de usuarios |
| Artefacto: | Módulo de autenticación |
| Ambiente: | Producción |
| Respuesta: | El sistema aplica políticas de seguridad robustas y monitoriza accesos no autorizados |
| Medida de Respuesta: | 100% de los accesos deben ser identificados y monitoreados, menos del 1% de intentos de acceso autorizado. |

| Escenario de Calidad N°5 | |
| --- | --- |
| Atributo de Calidad Asociado (Característica): Eficiencia de Desempeño  Subcaracterística: Capacidad | |
| Descripción: El sistema debe ser capaz de soportar una cantidad mínima y máxima de usuarios simultáneamente sin degradar el desempeño | |
| Diagrama  El contenido generado por IA puede ser incorrecto. | |
| Fuente del Estímulo: | Usuarios concurrentes |
| Estímulo: | Alta concurrencia de usuarios en la plataforma |
| Artefacto: | Servidor de aplicación |
| Ambiente: | Producción |
| Respuesta: | El sistema mantiene el rendimiento y la capacidad operativa bajo condiciones de carga máxima. |
| Medida de Respuesta: | El sistema debe soportar hasta 150 usuarios concurrentes con un tiempo de respuesta inferior a 3 segundos. |

| Escenario de Calidad N°6 | |
| --- | --- |
| Atributo de Calidad Asociado (Característica): Adecuación Funcional  Sub Característica: Completitud | |
| Descripción: El sistema debe proporcionar todas las funcionalidades necesarias para satisfacer los requisitos del usuario | |
| Diagrama  El contenido generado por IA puede ser incorrecto. | |
| Fuente del Estímulo: | Usuario final |
| Estímulo: | Uso de diferentes funcionalidades del sistema |
| Artefacto: | Sistema completo |
| Ambiente: | Producción |
| Respuesta: | El sistema proporciona todas las funcionalidades requeridas |
| Medida de Respuesta: | El 98% de las funcionalidades requeridas requeridas por los usuarios deben estar implementadas y funcionando correctamente. |

| Escenario de Calidad N°7 | |
| --- | --- |
| Atributo de Calidad Asociado (Característica): Fiabilidad | |
| Sub Característica: Madurez | |
| Descripción: El sistema debe operar de manera confiable, minimizando fallos y errores durante periodos prolongados de uso. | |
|  |
| Diagrama  El contenido generado por IA puede ser incorrecto. | |  |
|  |
| Fuente del Estímulo: | Sistema en operación continua |  |
| Estímulo: | Uso prolongado del sistema |  |
| Artefacto: | Sistema completo |  |
| Ambiente: | Producción |  |
| Respuesta: | El sistema opera sin fallos significativos |  |
| Medida de Respuesta: | El sistema debe operar sin fallos críticos durante al menos 30 días consecutivos, con un máximo de 2 errores menores por semana. |  |

| Escenario de Calidad N°8 | |
| --- | --- |
| Atributo de Calidad Asociado (Característica): Seguridad | |
| Sub Característica: Completitud | |
| Descripción: El sistema debe proteger la información sensible de clientes y transacciones contra accesos no autorizados. | |
|  |
| Diagrama  El contenido generado por IA puede ser incorrecto. | |  |
|  |
| Fuente del Estímulo: | Atacante potencial |  |
| Estímulo: | Intento de acceso no autorizado a datos sensibles |  |
| Artefacto: | Base de datos de clientes y transacciones |  |
| Ambiente: | Producción |  |
| Respuesta: | El sistema previene el acceso no autorizado y registra el intento |  |
| Medida de Respuesta: | El 100% de los datos sensibles deben estar encriptados y el sistema debe prevenir todos los intentos de acceso no autorizados. |  |

* 1. **Especificación de Casos de Uso Relevantes**

Los casos de uso considerados los más relevantes para el desarrollo de la arquitectura fueron determinados. Los criterios usados para dicha determinación fueron:

* Su implementación es de alto riesgo.
* Incluye muchos conceptos y relaciones del dominio.
* Incluye posibles escenarios críticos de calidad.

| **Código** | **Nombre** | **Actores** | **Prioridad** |
| --- | --- | --- | --- |
| CU.1 | Autenticación de Usuario (Registro) | Cliente, Sistema | Alta |
| CU.2 | Autenticación de Usuario (Iniciar Sesión) | Cliente, Sistema | Alta |
| CU.3 | Perfil de usuario | Cliente, Sistema | Alta |
| CU.5 | Pedidos e Historial | Cliente, Sistema | Media |
| CU.6 | Carrito de compra | Cliente, Sistema | Alta |
| CU.8 | Método de pago | Cliente, Sistema | Alta |
| CU.9 | Gestión de productos | Administrador, Sistema | Alta |
| CU.10 | Dashboard de ventas | Administrador, Sistema | Baja |
| CU.11 | Reporte de ventas | Administrador, Sistema | Media |
| CU.12 | Gestión de pedidos | Administrador, Sistema | Alta |

* 1. **Especificación de los Escenarios de Calidad Relevantes**

Después de un análisis en conjunto con los stakeholders, los escenarios de calidad se expresan a continuación:

| **ID: QS1** | **Descripción** |
| --- | --- |
| **Nombre** | Navegación Intuitiva |
| **Sinopsis** | El sistema debe permitir que los usuarios naveguen fácilmente por la interfaz sin necesidad de instrucciones adicionales |
| **Entorno** | Producción |
| **Cambio en el entorno** | Ningún cambio esperado, pero se evalúa continuamente con usuarios reales para verificar la usabilidad |
| **Comportamiento esperado** | Los usuarios deben completar tareas comunes como compras, reportes de ventas y edición de perfil sin ayuda |
| **Medida** | El 90% de los usuarios deben poder completar estas tareas en menos de 2 minutos sin asistencia |
| **Prioridad Arquitectónica** | Alta |
| **Aplicación** | Global |
|  |  |
| **ID: QS2** | **Descripción** |
| **Nombre** | Eficiencia de desempeño |
| **Sinopsis** | El sistema debe mantener un tiempo de respuesta inferior a 2 segundos bajo una carga normal de usuario |
| **Entorno** | Producción |
| **Cambio en el entorno** | Aumento del tráfico de usuarios, especialmente durante picos de demanda. |
| **Comportamiento esperado** | El sistema debe responder rápidamente a las acciones del usuario, incluso con alta concurrencia de usuarios |
| **Medida** | El tiempo de respuesta debe ser inferior a 2 segundos en el 95% de los casos con 100 usuarios concurrentes |
| **Prioridad Arquitectónica** | Alta |
| **Aplicación** | Global |
|  |  |
|  |  |
| **ID: QS3** | **Descripción** |
| **Nombre** | Compatibilidad |
| **Sinopsis** | El sistema debe funcionar correctamente en múltiples dispositivos y navegadores |
| **Entorno** | Producción |
| **Cambio en el entorno** | Incremento de dispositivos y navegadores utilizados por los usuarios |
| **Comportamiento esperado** | La aplicación debe ser accesible y funcional en los principales navegadores (Chrome, Firefox, Safari, Edge) y dispositivos (móviles, tablets, PCs) |
| **Medida** | La aplicación debe pasar todas las pruebas de compatibilidad con un 95% de éxito |
| **Prioridad Arquitectónica** | Media |
| **Aplicación** | Global |
|  |  |
| **ID: QS4** | **Descripción** |
| **Nombre** | Seguridad |
| **Sinopsis** | El sistema debe garantizar que las cuentas de usuario sean gestionadas de manera segura, con mecanismos de autenticación robustos |
| **Entorno** | Producción |
| **Cambio en el entorno** | Ningún cambio esperado, pero debe estar siempre actualizado para prevenir vulnerabilidades |
| **Comportamiento esperado** | Los accesos a las cuentas deben ser seguros y los intentos de acceso no autorizado deben ser monitoreados |
| **Medida** | El 100% de los accesos deben ser identificados y monitoreados, con menos del 1% de intentos de acceso no autorizado |
| **Prioridad Arquitectónica** | Media |
| **Aplicación** | Global |

1. **Modelo 4 + 1**

**Vista Lógica**

La vista lógica de la aplicación Byte & Beat se presenta a través de un Diagrama de Clases, que representa relaciones entre las clases y sus interacciones dentro del sistema. Este diagrama organiza los componentes principales de permiten funcionalidad a la plataforma

En el Diagrama de Clases, se identifican los módulos y clases clave que componen la arquitectura lógica del sistema, como se describe a continuación:

* **Usuario**: Esta clase representa a los usuarios del sistema y contiene información básica como el nombre, correo electrónico y contraseña. Incluye métodos para registrar nuevos usuarios, iniciar sesión, recuperar contraseña y actualizar el perfil.
* **Administrador**: Se extiende de la clase Usuario. Los administradores tienen funciones adicionales como la gestión de productos, la visualización de reportes de ventas y gestión de pedidos. Esto permite que controlen aspectos operativos del sistema.
* **Cliente**: Esta clase igualmente se extiende de la clase Usuario, pero está orientada a la interacción con el sistema de compra. Los clientes pueden realizar compras, ver el historial de pedidos, confirmar compras y gestionar sus direcciones de envío.
* **Dirección**: Representa la dirección de usuario (cliente o administrador), con atributos como calle, número, comuna y ciudad. La dirección se utiliza en la creación de pedidos y para la determinación del lugar de envío.
* **Reporte De Ventas**: Esto se encarga de generar reportes detallados de ventas, como las ventas por fecha y el total de ventas. Los administradores pueden generar reportes y dashboard que les permite obtener información clave sobre el desempeño de la tienda.
* **Tipo Entrega:** Gestiona los métodos de entrega disponibles, como el retiro o él envió. Permite asociar una entrega con un pedido y ofrece método para agregar diferentes tipos de entrega en el sistema web.
* **Pedido:** Representa un pedido realizado por un cliente. Incluye detalles como el estado del pedido, el tipo de entrega y la relación con los productos seleccionados. Los administradores pueden actualizar el estado y obtener los detalles del pedido.
* **Detalle Pedido**: Detalle de cada producto dentro de un pedido, incluyendo precio total y cantidad. Esta clase ayuda a gestionar los productos individuales dentro de un pedido y calcula el costo total de la compra.
* **Producto**: Define los productos disponibles en el sistema, incluyendo atributos como SKU, nombre, descripción, precio, stock, categoría, marca e imagen. Permite la gestión de productos en el inventario, incluyendo la adición, modificación y eliminación de productos.

Diagrama

El contenido generado por IA puede ser incorrecto.

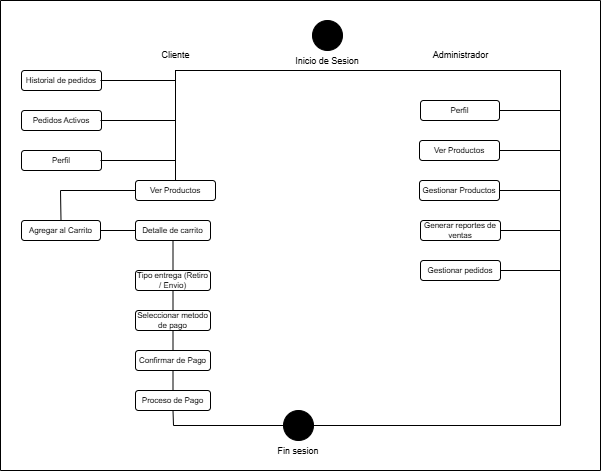
**Ilustración 1: Diagrama de Clases**

**Vista de Procesos**

La vista del proceso de la aplicación Byte & Beat se representa a través de un Diagrama de Actividades. Este diagrama describe los flujos de actividades claves para los usuarios del sistema, tanto clientes como administradores, desde el inicio de sesión hasta el cierre de sesión, y cubre las principales operaciones de gestión de pedidos, productos e interacciones de compras.

#### Descripción del Flujo de Actividades

1. **Inicio de sesión**: Los usuarios, tanto clientes como administradores, comienzan con el proceso de autenticación para acceder a sus funcionalidades respectivas. Esto permite a los usuarios acceder a sus respectivas cuentas, lo cual es el punto de partida para acceder a sus áreas personalizadas
2. **Flujo para clientes** :
   * **Historial de pedidos, Perfil y Pedidos activos**: Después de iniciar sesión, los clientes pueden revisar su historial de compras, pedidos que están pendiente por retiro o entrega y actualizar su perfil.
   * **Ver productos**: Los clientes tienen la opción de navegar por el catálogo de productos disponibles que desean comprar.
   * **Agregar al carrito**: Los productos seleccionados se agregan al carrito, donde pueden ver detalles del pedido.
   * **Detalle de carrito**: El cliente puede revisar los productos del carrito, modificar cantidades o eliminar productos si lo desea.
   * **Seleccionar tipo de entrega**: Cliente puede elegir entre las opciones de entrega disponibles (envió o domicilio)
   * **Seleccionar método de pago**: El cliente elige el método de pago, como tarjeta de crédito o débito
   * **Confirmar pago**: El cliente revisa su compra y confirma el pago
   * **Proceso de Pago**: El flujo termina con la verificación y finalización de la transacción de pago
3. **Flujo para Administradores**:
   * **Perfil**: El administrador tiene la posibilidad de acceder y gestionar su perfil personal.
   * **Ver Productos:** El administrador puede visualizar los productos listados en el sitio web para gestionar su disponibilidad.
   * **Gestionar productos:** El administrador tiene la opción de agregar, modificar o eliminar productos en el catálogo del sitio web.
   * **Generar reportes de ventas:** El administrador puede generar reportes de ventas, proporcionando datos importantes sobre el rendimiento de la tienda y las tendencias de los productos.
   * **Gestionar pedidos:** el administrador puede gestionar los pedidos realizados por los clientes asegurando que se procesen y se entreguen adecuadamente.
4. **Fin de Sesión**: Al completar sus actividades, tanto clientes como trabajadores pueden cerrar sesión, asegurando la seguridad de la información y finalizando su interacción con el sistema.



**Ilustración 2: Diagrama de Actividades**

**Vista Física**

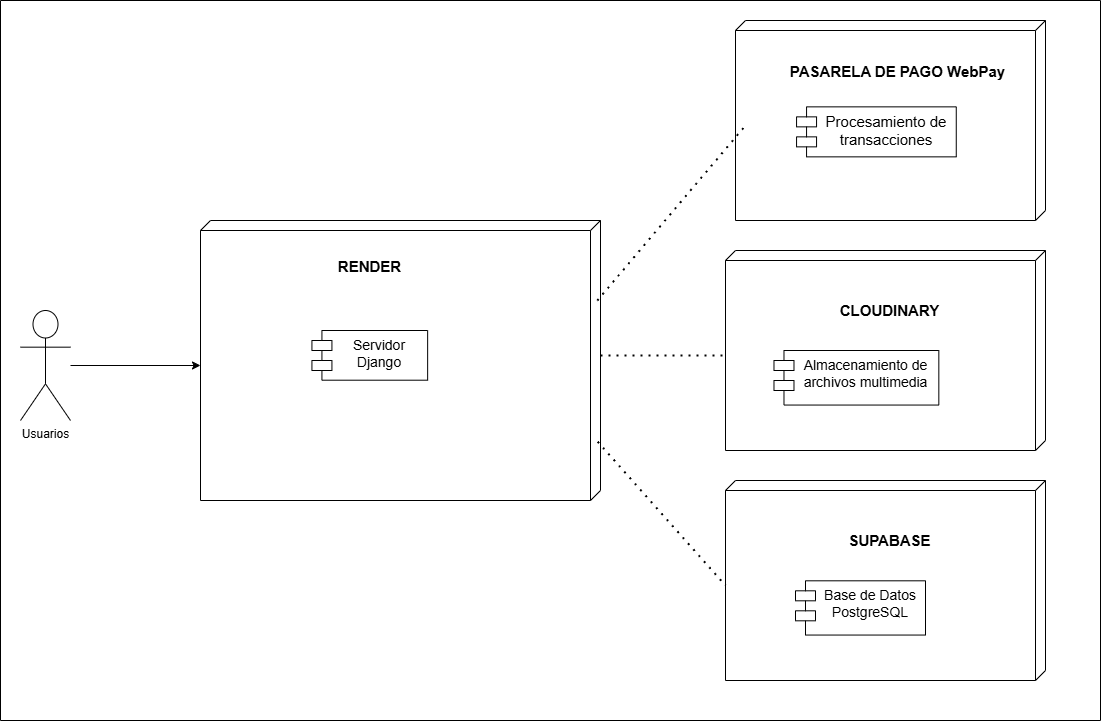
La vista física del sistema Byte & Beat se representa mediante un Diagrama de Despliegue, que muestra la disposición de los componentes físicos en el entorno de producción. Este diagrama destaca los principales nodos y cómo interactúan para garantizar el funcionamiento adecuado y seguro de la aplicación.

#### **Descripción de los Componentes Físicos**

1. **Actor (Usuario)**: Representa a los usuarios que interactúan con la aplicación a través de una interfaz web. Los usuarios pueden ser clientes o administradores que acceden al sistema mediante dispositivos conectados a internet.
2. **Servidor Web (Django Aplicación)**: Este es el servidor donde se ejecuta la aplicación principal, construida con el framework Django. El servidor maneja la lógica de negocio y se comunica con otros servicios como la BD PostgreSQL, WebPay y Almacenamiento de imágenes.
3. **Base de Datos**: Almacena toda la información relevante para la operación del sistema, incluyendo datos de usuarios, inventarios, y transacciones de compras. La base de datos es accedida por la aplicación Django para realizar operaciones CRUD y asegurar la persistencia de los datos.
4. **Almacenamiento en la Nube:** Este servicio se utiliza para almacenar archivos multimedia como imágenes de productos. En lugar de almacenar imágenes en la base de datos, solo se guarda la URL de la imagen, lo cual se gestiona mediante una API de comunicación desde Django.
5. **Sistema de Pago (WebPay):** El sistema de pago es integrado con el servidor Django. Este servicio maneja las transacciones de los usuarios, procesando pagos de manera segura.

#### **Interacciones y Conexiones:**

1. **Actor (Usuario) con Servidor Django**: Los usuarios interactúan con la aplicación web (servidor Django) a través de su navegador web.
2. **Servidor Django con la Base de Datos PostgreSQL:** El servidor Django interactúa con la base de datos PostgreSQL para realizar operaciones de lectura y escritura (guardar y recuperar datos como productos, pedidos, etc.).
3. **Servidor Django con Cloudinary:** El servidor Django se comunica con el almacenamiento en la nube para subir y recuperar imágenes. Solo se guarda la URL en la base de datos.
4. **Servidor Django con WebPay:** El servidor Django también interactúa con WebPay para procesar las transacciones de pago de los usuarios.

****

**Ilustración 3: Diagrama de Despliegue**

**Vista de Despliegue**

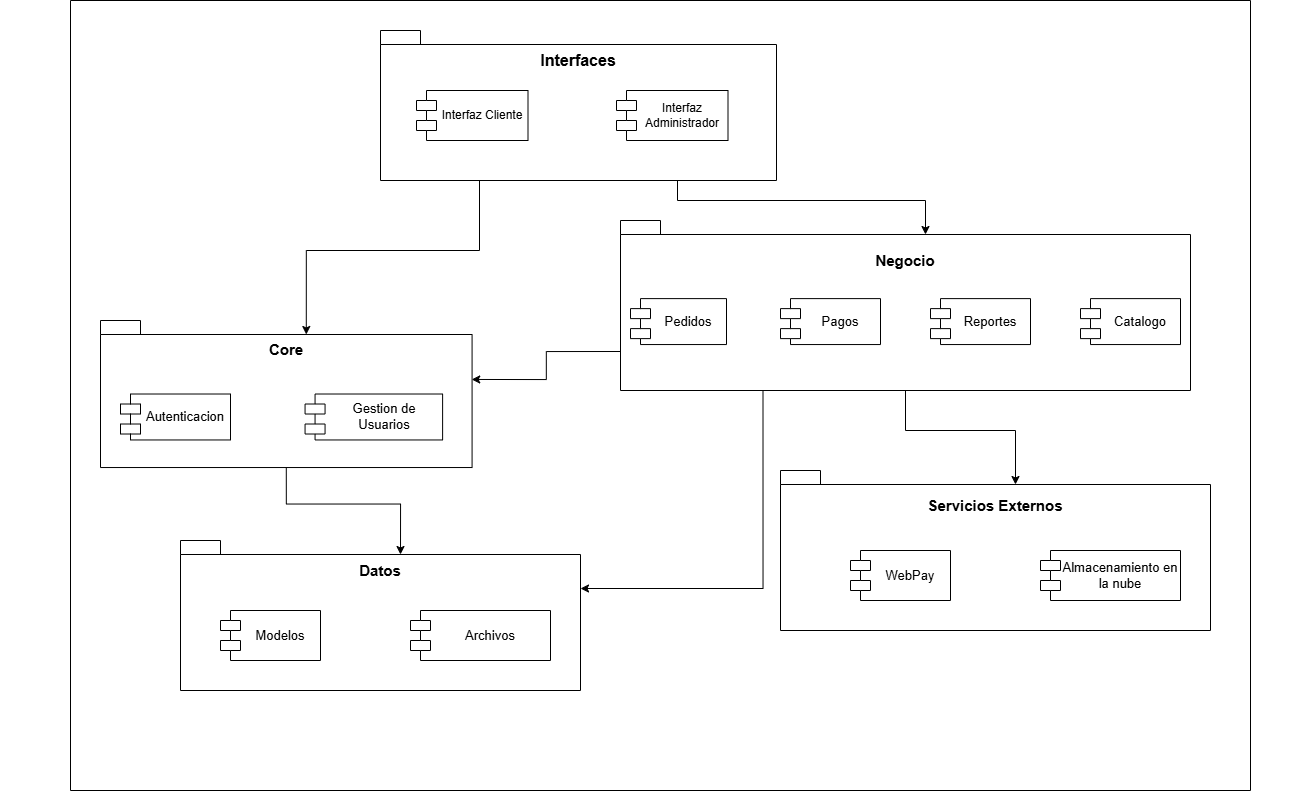
La vista de despliegue del sistema Byte & Beat se representa mediante un Diagrama de Paquetes que muestra los componentes físicos y virtuales involucrados en el funcionamiento de la aplicación. A continuación, se detallan los componentes principales que garantizan el acceso y la operación eficiente del sistema.

#### Componentes físicos

* **Dispositivos de acceso**: Los usuarios, incluyendo clientes y trabajadores, acceden al sistema web desde dispositivos como computadoras, tabletas o teléfonos inteligentes.
* **Conexiones de red**: Para asegurar el acceso continuo y sin interrupciones al sistema, es esencial contar con conexiones de red estables.

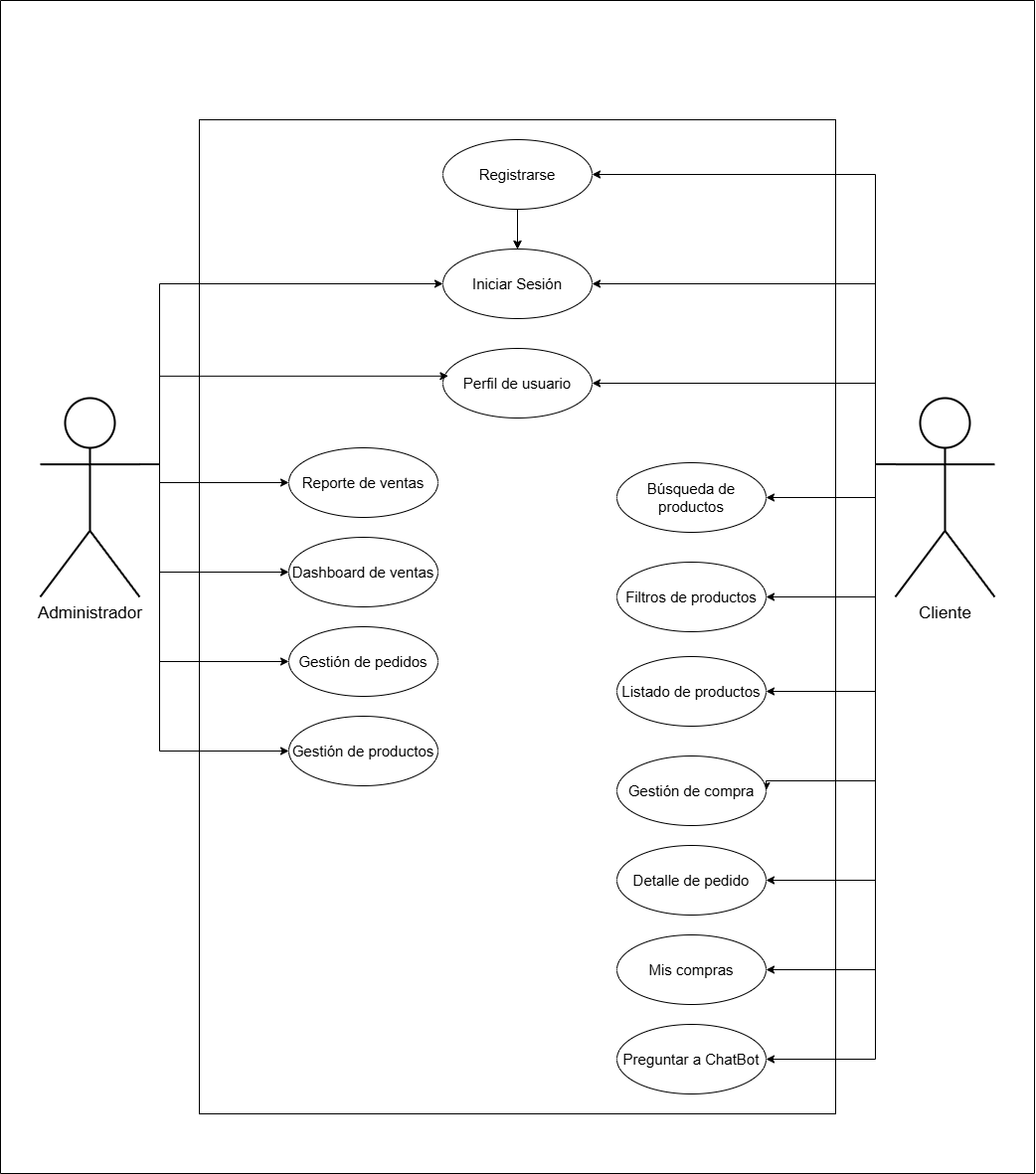
#### Componentes virtuales

* **Servidor Django**: Django es el servidor principal que gestiona la lógica del negocio, la autenticación de usuarios y la comunicación con la base de datos PostgreSQL y el almacenamiento en la nube. Está hospedado en Render, que se encargará de la gestión de la infraestructura y el despliegue de la aplicación.
* **Base de datos PostgreSQL (SupaBase)**: Es el sistema de gestión de BD que almacena de forma segura y eficiente los datos del sistema, como productos, usuarios y pedidos. Está alojada en Render para asegurar la disponibilidad y escalabilidad.
* **Almacenamiento en la nube**: El sistema utilizará almacenamiento en la nube para archivos multimedia, como las imágenes de productos, Este almacenamiento en la nube garantiza alta disponibilidad y seguridad de los archivos,
* **WebPay** : Este servicio externo es utilizado para procesar transacciones de pago de manera segura. El módulo de Pago de Django interactúa con WebPay para completar las compras de los usuarios.



**Ilustración 4: Diagrama de Paquetes**

**Caso de Uso del sistema (UML)**



**Ilustración 5: Diagrama de Casos de Uso**

1. **Decisiones de Diseño y Selección de Alternativas**

Las principales decisiones arquitectónicas para este proyecto se tomaron considerando la restricción de tiempo de construcción. Debido a que el proyecto debe implementarse en un tiempo limitado, se decidió adoptar una arquitectura Modelo-Vista-Template (MVT) que es la predeterminada con django y conocida por su facilidad de implementación, baja complejidad y probada efectividad en proyectos similares.

La arquitectura fue diseñada bajo el principio de separación de responsabilidades, permitiendo que las capas de la aplicación (Modelo, Vista y Template) se desarrollen de forma independiente, reduciendo el riesgo de errores y facilitando la paralelización del desarrollo.

Además, considerando la restricción de infraestructura y el escenario de calidad de Tolerancia a Fallos, se optó por una estructura modular.

En términos de mantenibilidad, el diseño MVT permite que el sistema sea extensible a futuro de forma que la integración con otros sistemas sea flexible y escalable.

En conclusión, la arquitectura MVT, combinada con un enfoque modular y patrones de diseño probados, garantiza que el sistema cumpla con las restricciones de tiempo, calidad y escalabilidad requeridas para este proyecto