Logotipo

Descripción generada automáticamente

DOCUMENTO DAS ARQUITECTURA DE SOFTWARE PROYECTO

Documento de Arquitectura de Software IEEE-1471-2000 Control del documento

Integrantes: Alvaro Arenas

Nombre Proyecto: Pedido Sugerido SaaS

Arquitectura Proyecto Capstone

Contenido

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

[**2.**](#_heading=h.30j0zll) **Propósito** 3

[**3.**](#_heading=h.1fob9te) **Alcance** 3

[**4.**](#_heading=h.3znysh7) **Equipo de Trabajo-Actores del Desarrollo** 4

[**5.**](#_heading=h.2et92p0) **Recomendaciones de conformidad con esta práctica.** 4

[Para asegurar que la arquitectura del sistema cumpla con los requisitos y estándares establecidos, se recomiendan las siguientes prácticas: 4](#_heading=h.tyjcwt)

[1. Implementar revisiones de arquitectura cada xxxx para asegurar que se ajusten a los cambios de requisitos. 4](#_heading=h.3dy6vkm)

[2. Utilizar patrones de diseño desacoplados como MVC para modularidad y escalabilidad o por capas. 4](#_heading=h.1t3h5sf)

[3. Adoptar metodologías ágiles como Scrum para la entrega continua de mejoras y funcionalidades. 4](#_heading=h.4d34og8)

[**6.**](#_heading=h.2s8eyo1) **Referencias y estándares aplicables a este documento:** 5

[**7.**](#_heading=h.17dp8vu) **Definiciones, acrónimos y abreviaciones.** 5

[**8.**](#_heading=h.3rdcrjn) **DESCRIPCIÓN DE ARQUITECTURA:**  6

[8.1 VISTAS: 6](#_heading=h.26in1rg)

[8.2 TIPOS DE VISTAS: 6](#_heading=h.lnxbz9)

[8.3 Framework Conceptual 7](#_heading=h.35nkun2)

[8.3.1 Descripción de la arquitectura en contexto 7](#_heading=h.1ksv4uv)

[8.3.1 Actores Usuarios y sus roles: 8](#_heading=h.44sinio)

[8.3.2 Actividades de arquitectura en el ciclo de vida 9](#_heading=h.2jxsxqh)

[8.3.3 Descripciones prácticas de arquitectura. 9](#_heading=h.z337ya)

[8.4 Documentación de la arquitectura 10](#_heading=h.3j2qqm3)

[8.5 Selección de puntos de vista de la arquitectura 10](#_heading=h.1y810tw)

[8.6 Vistas de la arquitectura 11](#_heading=h.4i7ojhp)

[UML-VISTA DE ESCENARIO 11](#_heading=h.2xcytpi)

[UML-VISTA LÓGICA-DIAGRAMA DE CLASES 12](#_heading=h.1ci93xb)

[UML-VISTA LÓGICA-DIAGRAMA DE BASE DE DATOS 14](#_heading=h.qsh70q)

[UML-VISTA DE PROCESO-DIAGRAMAS DE ACTIVIDAD 16](#_heading=h.1pxezwc)

[UML-VISTA DE DESPLIEGUE 18](#_heading=h.49x2ik5)

[UML-VISTA FÍSICA – DIAGRAMA INFRAESTRUCTURA 19](#_heading=h.2p2csry)

[Usabilidad y Calidad de software- Front-end -Vistas Principales 21](#_heading=h.147n2zr)

[Usabilidad y Calidad de software – Vistas específicas de usuario 22](#_heading=h.3o7alnk)

[8.7 Consistencia en la cantidad de vistas de la arquitectura. 23](#_heading=h.23ckvvd)

[8.7.1 DESCRIPCION DE MODULOS 23](#_heading=h.ihv636)

[8.7.2 DESCRIPCIÓN DE COMPONENTES 23](#_heading=h.32hioqz)

[8.7.3 DESCRIPCIÓN DE CONECTORES 23](#_heading=h.1hmsyys)

[9.1 Acceso a fuentes de información del proyecto 25](#_heading=h.41mghml)

# **Introducción**

El presente informe tiene como objetivo describir en detalle la arquitectura de un sistema de software como servicio diseñado mejorar los procesos de venta de los E-Commerces actuales más comunes.

El sistema desarrollado será implementado en WooCommerce, una de las plataformas más utilizadas de comercio electrónico. El sistema también proporcionará una interfaz intuitiva y accesible para todos los usuarios, asegurando que las gestiones se realicen de manera eficiente y organizada basada en los estándares de usabilidad y seguridad.

# **Propósito**

El propósito de este documento es servir como guía para el diseño e implementación de la arquitectura del sistema de recomendación de productos. En él se detallan las decisiones técnicas, los componentes principales y los puntos de integración que permitirán implementar una solución robusta y escalable, capaz de gestionar múltiples sitios de e-commerce bajo un esquema de administración unificado.

Este informe es esencial para los desarrolladores y stakeholders del proyecto, ya que les proporcionará una visión detallada de la estructura y comportamiento del sistema, así como las relaciones entre las diferentes funcionalidades y componentes. Asimismo, establece las bases para futuras expansiones y mejoras del sistema, considerando factores de disponibilidad, seguridad y usabilidad.

# **Alcance**

El sistema gestionará las siguientes áreas:

● **Análisis de datos:** Recopilación, procesamiento y análisis de información sobre compras, navegación de usuarios y características de productos para identificar patrones y relaciones.

● **Generación de recomendaciones:** Desarrollo y aplicación de algoritmos de recomendación para generar sugerencias personalizadas de productos.

● **Integración con WooCommerce:** Diseño e implementación de un plugin para WooCommerce que se encargará de mostrar las recomendaciones generadas por el sistema en la tienda online.

● **Gestión de la API:** Desarrollo y mantenimiento de la API de recomendaciones, que será la encargada de gestionar la comunicación entre el sistema de análisis y el plugin de WooCommerce.

● **Gestión de la base de datos:** Administración y mantenimiento de la base de datos MySQL que almacenará toda la información relevante para el sistema, incluyendo los datos de los productos, usuarios, compras y relaciones entre productos.

El sistema **NO** se encargará de los siguientes puntos:

● **Gestión de la tienda online:** El sistema se centrará en la generación de recomendaciones y su integración con WooCommerce. No se encargará de la gestión general de la tienda online, como la administración de productos, pedidos, clientes o pagos.

● **Atención al cliente:** El sistema no se encargará de la gestión de la atención al cliente o la resolución de consultas.

● **Diseño de la tienda online:** El sistema se centrará en la funcionalidad de las recomendaciones, no en la estética o diseño de la tienda online.

# **Equipo de Trabajo-Actores del Desarrollo**

El desarrollo del sistema requiere la participación de un equipo multidisciplinario con roles claramente definidos para asegurar que el proyecto se lleve a cabo de manera eficiente. A continuación, se presenta una tabla con los principales roles y sus responsabilidades:

# **Recomendaciones de conformidad con esta práctica.**

# Para asegurar que la arquitectura del sistema cumpla con los requisitos y estándares establecidos, se recomiendan las siguientes prácticas:

# Implementar revisiones de arquitectura cada sprint para asegurar que se ajusten a los cambios de requisitos.

# Utilizar patrones de diseño desacoplados como MVC para modularidad y escalabilidad o por capas.

# Adoptar metodologías ágiles como Scrum para la entrega continua de mejoras y funcionalidades.

# **Referencias y estándares aplicables a este documento:**

El documento sigue los siguientes estándares de calidad para asegurar la robustez y escalabilidad del sistema:

* **IEEE 830-1998 ST**: Estandarización de requisitos de software.
* **ISO 9126-2001**: Métricas de calidad del software.
* **Modelo 4+1 de Kruchten**: Modelo estructural basado en múltiples vistas de escenarios.
* **ATAM (Architecture Tradeoff Analysis Method)**: Metodología para evaluar y documentar decisiones arquitectónicas

# **Definiciones, acrónimos y abreviaciones.**

| Término | Definición |
| --- | --- |
| DAS | Documento de Arquitectura de Software |
| MVC | Modelo-Vista-Controlador, patrón de diseño utilizado para separar la lógica de negocio de la interfaz de usuario |
| UML | Unified Modeling Language, lenguaje gráfico para representar la arquitectura del sistema |
| CRUD | Operaciones básicas de Create, Read, Update, Delete para la gestión de datos. |
| SaaS | Software as a service, se presenta como un servicio en lugar de entregar el código fuente al cliente |
| A3S | Analytics as a service, los cálculos estadísticos son un servicio y no se entregarán al cliente |
| HTTPS | HyperText Transfer Protocol Secure, protocolo HTTP seguro para transferencia web. |

# **DESCRIPCIÓN DE ARQUITECTURA:**

La arquitectura del sistema se presenta mediante el uso de vistas basadas en el Modelo 4+1 de Kruchten, con el fin de proporcionar una visión completa y estructurada del sistema desde diferentes perspectivas. La arquitectura del sistema se presenta mediante el uso de vistas basadas en el Modelo 4+1 de Kruchten, con el fin de proporcionar una visión completa y estructurada del sistema desde diferentes perspectivas.

## VISTAS:

Cada vista se representará en un diagrama UML para describir la estructura y comportamiento del sistema. Estos diagramas serán esenciales para la comunicación entre los desarrolladores y stakeholders, asegurando una comprensión unificada de la arquitectura del sistema.

### TIPOS DE VISTAS:

1. **Vista de Escenarios (Diagrama de Caso de Uso)**: Describe las funcionalidades del sistema y cómo interactúan los diferentes actores con él.
2. **Vista Lógica (Diagrama de Clases)**: Detalla la estructura de datos y las relaciones entre las principales entidades del sistema.
3. **Vista de Procesos (Diagrama de Actividad)**: Muestra el flujo de actividades dentro del sistema y cómo se coordinan los diferentes procesos.
4. **Vista de Despliegue (Diagrama de Componentes)**: Representa la distribución de componentes en la infraestructura física.
5. **Vista Física (Diagrama Topológico)**: Representa la infraestructura de red y la comunicación entre los diferentes servicios del sistema

## Framework Conceptual

### Descripción de la arquitectura en contexto

El sistema será diseñado y desarrollado utilizando un enfoque ágil con metodologías como Scrum, lo que permitirá la entrega incremental de funcionalidades, la retroalimentación constante y la capacidad de adaptarse a cambios en los requisitos a lo largo del ciclo de vida del proyecto. Esto asegurará que se puedan abordar necesidades emergentes y maximizar el valor del producto para los usuarios.

El diseño del sistema se basará en la separación de vistas, que ofrecerán diferentes perspectivas de la arquitectura para una comprensión integral. A continuación, se describen las principales vistas:

* **Vista de Escenarios**: Representada mediante diagramas de Casos de Uso, que describen las interacciones del usuario con el sistema y los flujos principales.
* **Vista Lógica**: Incluye el Modelo de Clases, que muestra las entidades y relaciones del sistema, así como el Modelo de Datos que detalla la estructura de la base de datos.
* **Vista de Procesos**: Utiliza Diagramas de Actividad para ilustrar los flujos internos del sistema y las interacciones entre los distintos módulos.
* **Vista de Despliegue**: Se implementará a través de Diagramas de Componentes o Modelos de Capas para reflejar la distribución de la lógica de negocio, la interfaz de usuario y la capa de acceso a datos.
* **Vista Física**: Utiliza Diagramas Topológicos o de Servicios para mostrar la distribución del sistema en el entorno físico, como servidores, redes y otros componentes de infraestructura.

Cada una de estas vistas está interconectada para garantizar la coherencia y la alineación con los requisitos de negocio. No hay ninguna vista separada de una implementación específica, sino que todas contribuyen a una visión holística de la arquitectura del sistema descrita en este documento.

Estas vistas se han creado utilizando el Lenguaje de Modelado Unificado (UML) en su versión 2.0 y han sido desarrolladas empleando herramientas de modelado estándar como **StarUML** y Miro y en front-end **Canva, Figma u otro** para facilitar la creación de diagramas visuales que respalden la documentación arquitectónica.

El estilo arquitectónico se centra en una combinación de la arquitectura RESTful y patrones de diseño como MVC, en donde cada módulo del sistema se corresponde con un conjunto específico de casos de uso que resuelven los procesos definidos para la operación.

Estos procesos corresponden a:

● **Análisis de datos:** Recopilación, procesamiento y análisis de datos sobre compras, navegación de usuarios y características de productos.

● **Generación de recomendaciones:** Desarrollo y aplicación de algoritmos de recomendación para generar sugerencias personalizadas de productos.

● **Integración con WooCommerce:** Diseño e implementación de un plugin para WooCommerce que se encargará de mostrar las recomendaciones generadas por el sistema en la tienda online.

● **Gestión de la API:** Desarrollo y mantenimiento de la API de recomendaciones, que será la encargada de gestionar la comunicación entre el sistema de análisis y el plugin de WooCommerce.

● **Gestión de la base de datos:** Administración y mantenimiento de la base de datos MySQL que almacenará toda la información relevante para el sistema, incluyendo los datos de los productos, usuarios, compras y relaciones entre productos.

### Actores Usuarios y sus roles:

Este documento representa la identificación de Actores/Usuarios Stakeholders y sus roles a partir de la interpretación de los casos de uso del Negocio asociados.

### 8.3.2 Actividades de arquitectura en el ciclo de vida

El desarrollo de la arquitectura del sistema de recomendación de productos se abordó utilizando un ciclo de vida **iterativo incremental** basado en **Scrum**. Esta metodología fue seleccionada porque permite realizar entregas parciales y frecuentes de software funcional, facilitando la validación temprana de los requisitos por parte de los stakeholders y el product owner, además de permitir una mayor adaptación a los cambios en el entorno de negocio.

El ciclo de vida se organizó en sprints de **2 a 4 semanas**, con cada iteración centrada en la construcción y mejora de un módulo específico del sistema (por ejemplo, el módulo de análisis de datos, el módulo de API o el plugin de WooCommerce). Al final de cada sprint, se realiza una revisión detallada de los resultados para asegurar la alineación con las expectativas del cliente y se priorizan las tareas para el siguiente sprint.

### 8.3.3 Descripciones prácticas de arquitectura.

Durante el proceso de desarrollo, la arquitectura del sistema **de recomendación de productos** fue escalada desde un escenario principal de caso de uso hasta un diseño modular a través de los siguientes pasos:

1. **Análisis de Requerimientos:** Se analizaron los requisitos iniciales del sistema para identificar los casos de uso principales, tales como "Analizar datos de compras y navegación", "Generar recomendaciones personalizadas", "Integrar las recomendaciones con WooCommerce" y "Gestionar la API y la base de datos".
2. **Definición de la Vista Lógica:** A partir del escenario principal, se identificaron las entidades y relaciones claves en un diagrama de clases UML, representando el núcleo del sistema.
3. **Modelo de Datos:** Se desarrolló un diagrama de base de datos para definir la estructura de la información, manteniendo la coherencia con la vista lógica.
4. **Vista de Procesos:** Se modelaron los flujos de las actividades y transacciones principales del sistema mediante diagramas de secuencia.
5. **Despliegue:** Se definió la infraestructura física y lógica para alojar el sistema en la nube, detallando los servicios necesarios, como ISP, servidores de aplicaciones y bases de datos, así como servidores.
6. **Definición de planes de pruebas y técnicas de testing:** Se establecieron planes de pruebas para cada módulo del sistema, incluyendo pruebas unitarias y funcionales, así como de caja negra.

## Documentación de la arquitectura

* **ISO/IEC 25010**: Se utilizó para evaluar la calidad del sistema en términos de funcionalidad, seguridad y eficiencia.
* **IEEE 1471-2000**: Proporcionó la guía para la representación de vistas arquitectónicas y la identificación de stakeholders.
* **Modelo 4+1 de Kruchten**: Estructura la arquitectura en vistas lógicas, de desarrollo, de procesos y física, facilitando la visualización desde múltiples perspectivas.
* **Planes de prueba**: Se establecieron planes de pruebas para cada módulo del sistema, incluyendo pruebas unitarias y funcionales, así como de caja negra.

## Selección de puntos de vista de la arquitectura

|  |  |  |
| --- | --- | --- |
| VISTAS | UML | Cantidad |
| Escenario Principal | Diagrama de Caso de uso General | 1 |
| Vista Lógica | Diagrama de Clases  MER | 1  1 |
| Vista de Proceso | Diagrama de secuencias con 4 actividades:  -Obtención de token de usuario  -Recomendación por productos frecuentes  -Recomendación por productos relacionados  -Recomendación por productos según perfil | 1 |
| Vista Física | Diagrama topológico de infraestructura | 1 |
| Vista de Despliegue | Diagrama de componentes  Diagrama de capas (MVC) | 1  1 |

**\*\*\* Debido a confidencialidad, el documento no posee toda la información \*\*\***