1. Definiciones y Especificaciones de Requerimientos

a) Definición General del Proyecto de Software

Idea General: Construmole es una plataforma de comercio electrónico diseñada específicamente para una ferretería. La idea principal es proporcionar un medio digital para la gestión de inventarios, ventas, y la interacción con clientes. El sistema permite a los usuarios explorar productos, agregar artículos al carrito de compras, confirmar pedidos y gestionar sus cuentas. El objetivo del proyecto es digitalizar las operaciones de la ferretería, mejorando la eficiencia en la gestión de productos y facilitando las compras para los clientes.

Propósitos y Objetivos del Sistema:

- **Automatización del Inventario:** Permitir que el inventario se actualice automáticamente con cada compra, optimizando la gestión de existencias.
- **Facilitar el Proceso de Compra:** Proveer una interfaz intuitiva para que los usuarios puedan realizar compras fácilmente desde cualquier dispositivo.
- **Mejorar la Experiencia del Cliente:** Ofrecer una plataforma accesible donde los clientes puedan ver productos, hacer preguntas, y realizar compras de manera eficiente.
- **Gestionar Pedidos de Manera Eficiente:** Permitir que la administración de la tienda gestione los pedidos recibidos y coordine las entregas de manera más organizada.

Usuarios: El sistema está dirigido a dos grupos principales de usuarios:

- **Clientes:** Personas que buscan comprar productos de la ferretería. Este grupo puede variar desde usuarios con experiencia mínima en tecnología hasta aquellos con un nivel intermedio.
- **Administradores:** Personal de la ferretería encargado de gestionar productos, pedidos, y la interacción con clientes a través del backend del sistema.

b) Especificación de Requerimientos del Proyecto

Requisitos Generales:

- El sistema debe estar disponible tanto para dispositivos móviles como para computadoras.
- Debe manejar múltiples usuarios concurrentes sin pérdida de datos o fallos en el sistema.
- Implementar medidas de seguridad para proteger la información personal y financiera de los usuarios.

Requisitos Funcionales:

- **Gestión de Usuarios:** Registrar, iniciar sesión, y gestionar cuentas de usuario.

- **Gestión de Productos:** Crear, editar, y eliminar productos desde un panel de administración.
- **Carrito de Compras:** Permitir que los usuarios agreguen y eliminen productos del carrito de compras.
- **Procesamiento de Pedidos:** Facilitar la confirmación y el seguimiento de pedidos tanto para clientes como para administradores.

Alcance del Sistema:

- **Limitaciones:** El sistema no incluirá inicialmente una pasarela de pagos en línea; las opciones de pago se limitarán a pagos contra entrega o en tienda física.
- **Alcances:** Se espera que el sistema gestione eficientemente la base de datos de productos y pedidos, así como la comunicación con los clientes a través de la plataforma.

c) Restricciones obligatorias del proyecto

El software debe cumplir con los siguientes requisitos obligatorios solicitados por el cliente para que sea factible y cumpla con su objetivo:

- El producto final debe ser una plataforma de comercio electrónico especializada para una ferretería, capaz de gestionar el inventario de productos, las órdenes de compra, y la información de los clientes.
- Debe permitir la gestión de categorías y productos, con opciones para añadir, modificar o eliminar productos desde un panel administrativo.
- No se incluirá una pasarela de pago; los usuarios podrán seleccionar únicamente la opción de pago al momento de la entrega o al recoger el producto en la tienda física.
- La plataforma debe incluir la capacidad de filtrar productos por categorías, asegurando una navegación eficiente para los usuarios.
- El sistema debe permitir que los administradores puedan ver las órdenes generadas, gestionar el inventario, y actualizar el estado de los pedidos.
- Debe ser una aplicación web accesible tanto en dispositivos móviles como en computadoras de escritorio.
- La interfaz de usuario debe estar en español, y toda la información debe ser gestionada por personal autorizado.

El objetivo principal es digitalizar y automatizar el proceso de ventas y gestión de inventario, eliminando los procedimientos manuales actuales.

Restricciones Cronológicas

Los usuarios que se registren como administradores o empleados de la ferretería solo tendrán acceso al sistema durante el horario laboral establecido por el cliente. Esto garantizará que las operaciones y la gestión del inventario se realicen únicamente durante el horario operativo de la tienda.

Ambiente de Implementación del Sistema Actual

Para implementar este proyecto, es necesario que el ambiente de trabajo cumpla con las siguientes condiciones:

- La ferretería debe contar con un computador en el área administrativa que esté conectado a internet.
- La plataforma debe estar alojada en un servidor confiable, preferiblemente en la nube, para garantizar disponibilidad y accesibilidad.
- Los navegadores compatibles deben incluir las versiones más recientes de Chrome, Firefox, Safari, y Edge.

d) Procedimientos de Instalación y Prueba

Procedimientos de Desarrollo:

Herramientas Utilizadas:

- **IDE:** Visual Studio Code.
- **Plataforma:** Django, un framework de desarrollo web en Python.
- **Bases de Datos:** MySQL.

Planificación:

- Se sigue la metodología en Cascada, comenzando con la fase de requerimientos, seguido del diseño, implementación, pruebas, y finalmente el despliegue.

Procedimiento de Instalación y Prueba:

Requisitos No Funcionales:

- **Facilidad de Uso:** El sistema debe ser fácil de navegar tanto para clientes como para administradores.
- **Confiabilidad:** El sistema debe ser robusto y capaz de manejar múltiples transacciones sin errores.
- **Rendimiento:** El sistema debe responder rápidamente a las solicitudes de los usuarios, incluso en condiciones de alta carga.

Obtención e Instalación:

1. **Clonación del Repositorio:** Obtener el código fuente desde el repositorio de GitHub.
2. **Configuración del Entorno Virtual:** Crear un entorno virtual en Python para aislar las dependencias del proyecto.
3. **Instalación de Dependencias:** Utilizar pip para instalar todas las librerías necesarias desde el archivo requirements.txt.

Especificación de Pruebas y Ejecución:

- El sistema se probará en un entorno de prueba que replica el entorno de producción para asegurar que funcione correctamente antes de ser desplegado.

2. **Arquitectura del Sistema**

a) Descripción Jerárquica:

El sistema está organizado de manera monolítica, con módulos separados para la gestión de productos, pedidos, y la gestión de usuarios. La estructura principal del código se divide en dos aplicaciones (store, accounts), y cada aplicación maneja una parte específica de la funcionalidad.

1. **Aplicación store:**

   Modelos: category, product, order, cart.

   Vistas: Permiten la visualización de productos, la gestión del carrito de compras, y la compra de productos.

   Urls: Definen las rutas que enlazan las vistas con las correspondientes acciones del usuario.

2. **Aplicación accounts:**

   Modelos: Gestión de usuarios, incluyendo registro y autenticación.

   Vistas: Manejan el registro, inicio de sesión, y eliminación de cuentas.

   Urls: Enlazan las vistas de autenticación y gestión de usuarios.

3. **Aplicación Raíz (root):**

   Configuración del Proyecto: Archivos como settings.py, urls.py, wsgi.py, y asgi.py, que manejan la configuración global del proyecto, incluyendo bases de datos, middleware, y rutas principales.

Descripción Individual de los Módulos

1. **Módulo store:**

   Descripción General y Propósito: Gestiona la lógica del negocio relacionada con los productos, el carrito de compras y las órdenes de compra.

   Responsabilidad y Restricciones: Este módulo maneja la lógica de inventario, la actualización de órdenes, y la interacción del usuario con los productos. No gestiona la autenticación de usuarios, que es manejada por el módulo accounts.

   Dependencias: Depende de los modelos de Django y de la base de datos para almacenar y recuperar información de productos y órdenes.

   Implementación: Implementado en archivos dentro de la carpeta store, principalmente en los archivos models.py, views.py, urls.py.

2. **Módulo accounts:**

   Descripción General y Propósito: Gestiona la autenticación y la administración de usuarios.

   Responsabilidad y Restricciones: Se encarga del registro, inicio de sesión, y eliminación de cuentas. No gestiona la lógica del negocio relacionada con los productos.

   Dependencias: Utiliza el modelo User de Django y depende de la autenticación y gestión de sesiones proporcionada por Django.

   Implementación: Implementado en archivos dentro de la carpeta accounts, como models.py, views.py, urls.py.

3. **Aplicación Raíz (root):**

   Descripción General y Propósito: Configura la aplicación general, incluyendo la base de datos, rutas, y el middleware.

   Responsabilidad y Restricciones: Configura el entorno de ejecución para las aplicaciones store y accounts. No maneja directamente la lógica del negocio ni la gestión de usuarios.

   Dependencias: Depende del framework Django para gestionar la configuración global.

   Implementación: Implementado en archivos como settings.py, urls.py, y otros archivos de configuración.

Dependencias Externas:

- **Librerías:** Django, MySQL Connector, Bootstrap para frontend.

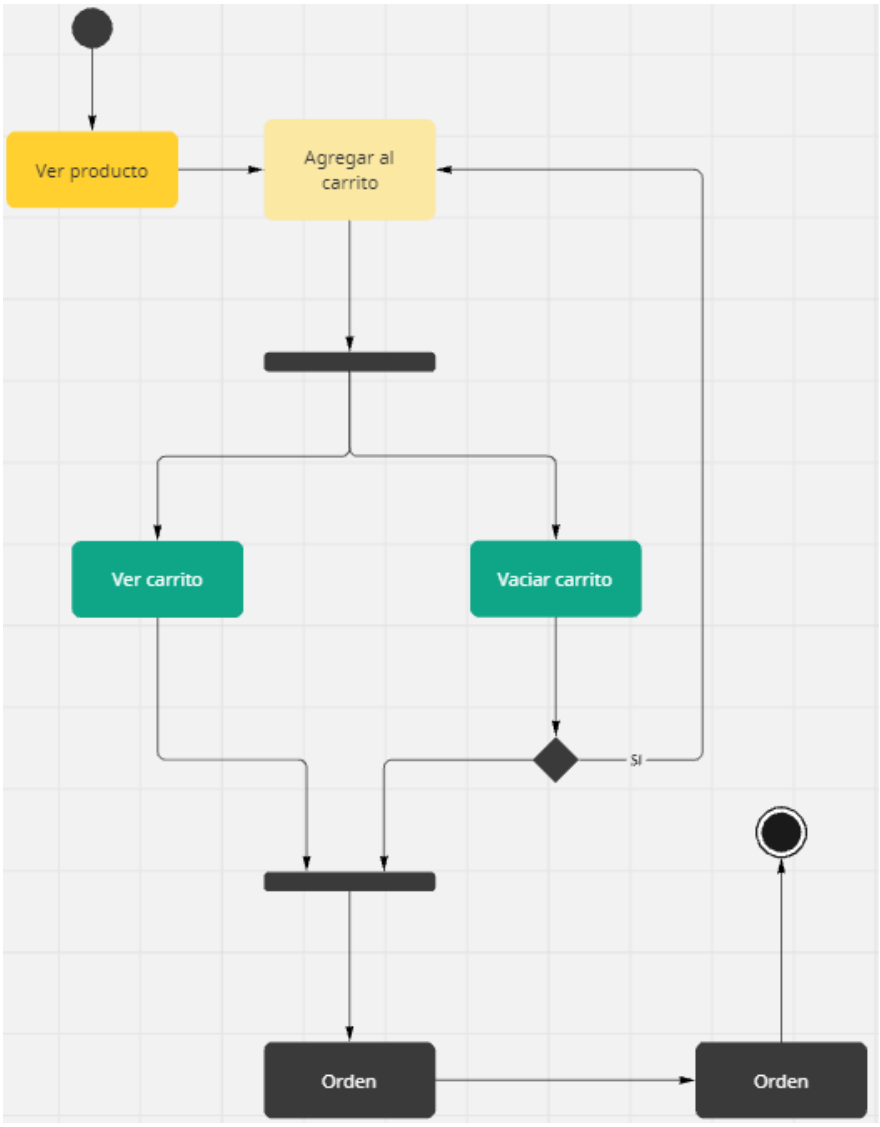
b) Vistas dinámicas

Las vistas dinámicas se centran en el comportamiento y la interacción de los componentes del sistema a lo largo del tiempo. Los diagramas de comportamiento en UML, como los de secuencia, actividad y estado, son herramientas clave para modelar estas vistas.

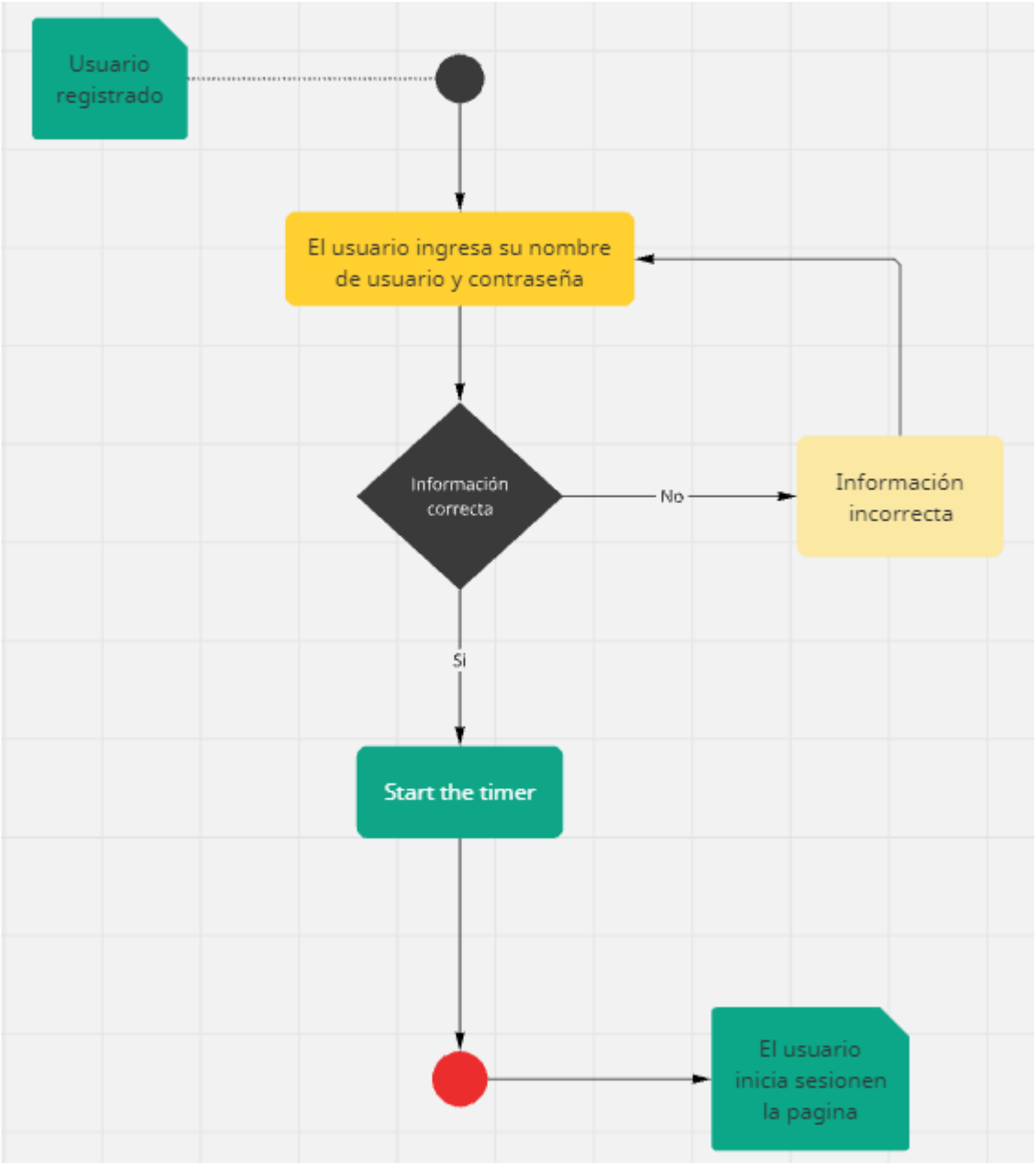
Diagrama de Actividad

- **Hacer un pedido:** Vamos a modelar el flujo de un pedido desde que es iniciado hasta que es confirmado, pasando por todas las validaciones y pasos intermedios.
- **Iniciar sesión:** Vamos a modelar el flujo de un usuario que intenta iniciar sesión con credenciales incorrectas / correctas.

Hacer un pedido

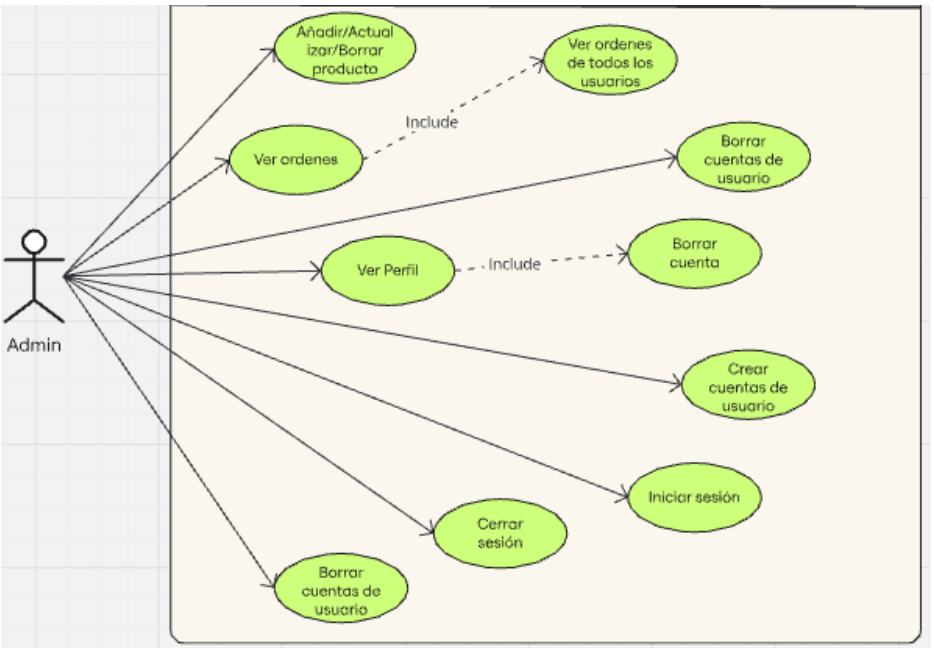


Iniciar sesión



Diagramas de casos de uso

Administrador: CU01, CU02, CU04, CU05, CU06, CU09, CU010, CU011



Cliente: CU03, CU04, CU05, CU06, CU07, CU08, CU11, CU12.

3. Análisis de Riesgo

Posibles Riesgos:

- **Fallo en la Integridad de Datos:** Pérdida o corrupción de datos durante las transacciones. **Mitigación:** Implementación de transacciones ACID en la base de datos.
- **Bajo Rendimiento Bajo Carga Alta:** El sistema podría no responder adecuadamente bajo alta demanda. **Mitigación:** Pruebas de rendimiento y optimización de consultas SQL.

4. Mecanismos de Control de Calidad

Proceso de Revisión de Documentación:

- Uso de la norma CCII-N2016-02 para asegurar que la documentación del proyecto sea clara, concisa, y completa.
- Revisión por pares de la documentación y código antes del despliegue final.