**Forma, Flecha, Rectángulo

El contenido generado por IA puede ser incorrecto.UNIVERSIDAD RAGIONAL I**

**Centro de Estudio: San Raymundo**

**Curso: Análisis y diseño de sistemas**

**Tutor: Ing. William Gonzales**

**Documentación análisis y sistemas (Semana 1)**

**Integrantes: Carnet:**

**Bryan Leonardo Rafael Santiagos Galicia 2349135**

**Andersón Osbelí Tzunux Tojín 2349023**

**Juan Jose Antonio Perez Garcia 2349044**

**Mario Adonias Galvez Yoc 2349105**

**Cesar Eduardo Gutiérrez Lima LIMAPRO01 2349071**

# Objetivos del proyecto

**Objetivo General:**

Desarrollar un sistema de ventas para la gestión eficiente de productos y clientes, utilizando:

* **Backend:** APIs en C# y .NET Core.
* **Frontend:** Java con programación orientada a objetos (POO) y una interfaz gráfica.
* **Base de Datos:** SQL Server o MySQL.

**Objetivos Específicos:**

1. Implementar **módulos CRUD** para:

* **Productos:** Crear, leer, actualizar y eliminar productos, controlar el stock, y consultar la lista de productos disponibles.
* **Clientes:** Crear, leer, actualizar y eliminar clientes, además de registrar las compras realizadas por cada cliente.

1. Establecer una arquitectura **Cliente-Servidor** donde el frontend en Java interactúe con las APIs RESTful en C# para realizar las operaciones CRUD.
2. Generar **documentación completa**, que incluya:

* Casos de uso y diagramas de UML.
* Diagramas de arquitectura del sistema y clases.
* Metodología de desarrollo justificada (ejemplo: Ágil, Cascada, etc.).
* Cronograma de desarrollo dividido por fases.
* Presupuesto estimado de costos de desarrollo e infraestructura.

# Alcance del proyecto

**1. Desarrollo de Módulos CRUD**

El sistema contemplará dos módulos principales que permitirán la gestión de **Productos** y **Clientes**, cada uno con sus respectivas operaciones CRUD (Create, Read, Update, Delete). Estas funcionalidades garantizarán la manipulación completa de los datos esenciales para la operación del sistema.

**a. Módulo de Productos:**

* **Creación de productos:** Permite registrar nuevos productos en el sistema, incluyendo nombre, descripción, precio, categoría y cantidad en stock.
* **Lectura/consulta de productos:** Visualización de una lista de productos disponibles, con capacidad para aplicar filtros por nombre, categoría o disponibilidad en stock.
* **Actualización de productos:** Modificación de datos específicos como el precio, la cantidad disponible, o cambios en la descripción.
* **Eliminación de productos:** Posibilidad de dar de baja un producto del inventario, evitando su visualización y operación dentro del sistema.
* **Control de stock:** Automatización en el ajuste de stock a partir de las compras o devoluciones, garantizando la precisión del inventario.

**b. Módulo de Clientes:**

* **Alta de clientes:** Registro de nuevos clientes con información personal como nombre, correo electrónico, número de contacto, y dirección.
* **Consulta de clientes:** Acceso a la lista de clientes registrados, con funciones de búsqueda y filtros.
* **Edición de datos de clientes:** Posibilidad de actualizar la información personal de un cliente registrado.
* **Eliminación de registros:** Opción para eliminar clientes inactivos o duplicados.
* **Registro de compras:** Asociación automática de cada compra realizada por el cliente al historial individual, permitiendo su posterior análisis o seguimiento.

Estos módulos estarán diseñados con una estructura modular y reutilizable, facilitando su mantenimiento y ampliación en versiones futuras del sistema.

**2. Arquitectura Cliente-Servidor**

El sistema utilizará una arquitectura **Cliente-Servidor** que permitirá una distribución clara entre la interfaz de usuario (frontend) y la lógica de negocio/datos (backend). Esta separación mejora la escalabilidad, seguridad y mantenimiento del sistema.

**a. Frontend:**

* **Lenguaje y entorno:** Aplicación de escritorio desarrollada en Java con Swing o JavaFX.
* **Responsabilidad:** Presentar una interfaz intuitiva para la interacción con el usuario final.
* **Funciones principales:** Enviar peticiones al servidor para realizar operaciones CRUD y mostrar los resultados en pantalla de manera clara.

**b. Backend:**

* **Lenguaje y entorno:** Desarrollo de servicios web utilizando C# y .NET para construir APIs RESTful.
* **Responsabilidad:** Ejecutar la lógica de negocio, validar datos, interactuar con la base de datos y devolver respuestas al frontend.
* **Seguridad y validación:** Implementación de validaciones de entrada y autenticación si es necesario.

**c. Comunicación:**

* **Protocolo:** HTTP/HTTPS para la transmisión de datos entre el cliente y el servidor.
* **Formato de datos:** Uso de JSON para enviar y recibir información de manera estructurada y ligera.

Este enfoque permite mantener una alta cohesión en cada componente y baja dependencia entre ellos, lo que facilita pruebas unitarias, integraciones futuras y mantenimiento general.

# Herramientas

**1. Java (Frontend)**

* **Descripción:** Lenguaje de programación orientado a objetos ampliamente utilizado para el desarrollo de aplicaciones de escritorio y móviles.
* **Uso en el proyecto:** Se utilizará para desarrollar la interfaz de usuario (cliente), permitiendo a los usuarios interactuar con el sistema de manera gráfica y amigable mediante componentes visuales creados con **JavaFX** o **Swing**.

**2. C# y .NET (Backend)**

* **Descripción:** Lenguaje moderno y potente de programación desarrollado por Microsoft. El framework .NET permite crear aplicaciones robustas, incluyendo servicios web.
* **Uso en el proyecto:** Desarrollo de APIs RESTful que ejecutarán la lógica del sistema, procesarán las operaciones CRUD y se comunicarán con la base de datos. Se utilizará **ASP.NET Core** para crear una arquitectura web escalable y segura.

**3. SQL Server (Base de Datos)**

* **Descripción:** Sistema de gestión de bases de datos relacional (RDBMS) desarrollado por Microsoft, ideal para aplicaciones empresariales por su rendimiento y fiabilidad.
* **Uso en el proyecto:** Almacenamiento y gestión de la información relacionada con productos, clientes y compras. El backend en C# se conectará a esta base para realizar consultas y actualizaciones.

**4. Postman (Pruebas de API)**

* **Descripción:** Herramienta para desarrollar, probar y documentar APIs RESTful.
* **Uso en el proyecto:** Validación de los endpoints del backend antes de integrarlos con el frontend. Permite enviar peticiones HTTP, revisar respuestas y comprobar el correcto funcionamiento de la API.

**5. Visual Studio (IDE para Backend)**

* **Descripción:** Entorno de desarrollo integrado desarrollado por Microsoft, especialmente optimizado para proyectos en C#.
* **Uso en el proyecto:** Desarrollo de las APIs en C#, pruebas locales del backend, depuración de errores y gestión del proyecto.

**6. IntelliJ IDEA / Eclipse (IDE para Java)**

* **Descripción:** Entornos de desarrollo integrados para Java. Ambos ofrecen herramientas para escribir, ejecutar y depurar código.
* **Uso en el proyecto:** Desarrollo del cliente en Java, integración con el backend y pruebas de interfaz.

**7. Draw.io o Lucidchart (Diagramación)**

* **Descripción:** Herramientas en línea para la creación de diagramas UML, arquitectónicos y de flujo.
* **Uso en el proyecto:** Creación de diagramas de casos de uso, clases, arquitectura del sistema, flujos de datos, entre otros.

**8. Git y GitHub (Control de versiones)**

* **Descripción:** Git es un sistema de control de versiones distribuido, y GitHub es una plataforma colaborativa basada en la nube.
* **Uso en el proyecto:** Gestión del código fuente de manera colaborativa, control de cambios, integración de ramas y documentación del repositorio.