1. **Introducción**

1.1. Propósito

Se desarrollará un software donde muestre los productos que se venden en una ferretería, así mismo como todo lo relacionado con dicho producto, es decir:

* Previsualización
* Descripción del producto
* Unidades disponibles
* Precio
* Forma de pago

Con esto se ayudará mayormente a personas que busquen algún tipo de herramienta en específico y/o material de construcción a encontrar mejor el artículo que necesite y ahorrar tiempo. A su vez, se ayudará a la ferretería a tener más clientes de lo habitual, aumentando sus ganancias y extendiendo su negocio a más lugares.

1.2. Alcance

El software podrá almacenar 500 tipos de herramientas y material de construcción y mediante conexión a internet el usuario podrá consultar los diferentes tipos de artículos que están a su disposición, a su vez la aplicación podrá mantener conectado a más de 2500 usuarios simultáneamente. En dado caso que el servidor no pueda mantener conectado a un usuario, se mantendrá en una lista de espera hasta que pueda entrar

1.3. Descripción general del documento

Este documento describe:

* Las principales funciones del sistema.
* Los usuarios finales y sus habilidades.
* Las limitaciones técnicas, de negocio o regulatorias.
* La lista de suposiciones que afectan al sistema y sus dependencias
* Las funcionalidades del sistema, cómo deben comportarse y las interacciones con los usuarios.
* Los requerimientos no funcionales como el rendimiento, la usabilidad, seguridad, disponibilidad, etc.
* Además se describirá cómo serán las interfaces de usuario, incluyendo pantallas y flujos.
* Cómo se conectará el sistema con otros sistemas o dispositivos.
* Detalla restricciones impuestas al diseño, como tecnologías específicas que deben utilizarse.
* Los distintos tipos de diagramas que se elaboraron
* Describe las medidas de seguridad y los controles de acceso necesarios.
* También se establecen requisitos de rendimiento, como tiempos de respuesta y capacidad.
* Describe cómo se mantendrá y actualizará el sistema.
* Proporciona un plan de pruebas para validar que el sistema cumple con los requisitos.

1. **Descripción general**

2.1. Perspectiva del producto

2.2. Funcionalidad del producto

* **Registro y autenticación de usuarios**: Permitir a los clientes registrarse y autenticarse.
* **Catálogo de productos**: Mostrar los productos de la ferretería con sus descripciones, precios y disponibilidad.
* **Carrito de compras**: Facilitar la selección y compra de productos.
* **Sistema de pagos**: Implementar un sistema de pagos en línea.
* **Gestión de inventario**: Para el administrador, gestión y actualización del stock.
* **Notificaciones**: Enviar confirmación de pedido y notificaciones de entrega.
* **Historial de compras**: Permitir a los usuarios consultar sus pedidos anteriores.
* **Opiniones y calificaciones**: Facilitar comentarios de usuarios sobre productos.

2.3. Características del usuario

El Software será dirigido mayormente para personas mayores de 18 años que busquen algún tipo de herramienta para su hogar o trabajo, también puede ser utilizado en menores de 18 años que deseen buscar alguno de los artículos de la ferretería

* Perfil: Trabajadores ……..
* Edad: mayormente mayores de 18 años
* Nivel socioeconómico: ingresos muy variados debido a las edades
* Nivel de educación: La educación necesaria para usar el producto es mínima, a pesar de ello se espera que mayormente los usuarios tengan los conocimientos suficientes para poder usar las herramientas que se están vendiendo.
* Limitaciones: No se cubren limitaciones en este caso, todo aquel que quiera usar la página tendrá que ser capaz de comprender y maniobrar un dispositivo electrónico, específicamente deberá poder “manejar” un navegador. Pero, se contempla en Android el TalkBack, en PC el Narrador de Windows y por parte de Apple el VoiceOver, herramientas que ayudan a personas invidentes a interactuar con entornos de software.

2.4. Restricciones

El sistema tiene que ser compatible con dispositivos Windows, Mac, Linux, Android e iOS.

2.5. Suposiciones y dependencias

Se asume que el usuario tendrá acceso a internet y un dispositivo con el que podrá acceder al software.

1. **Requerimientos específicos del sistema**

3.1. Requerimientos funcionales

* El sistema debe permitir que el usuario cree una cuenta personalizada con un correo propio.
* El usuario debe de poder visualizar los productos y su precio, también una breve descripción del producto.
* Cuando el usuario desee podrá agregar productos en un “Carrito” donde se mantendrán ahí hasta que se realice la compra.
* El software tiene que contar con un método de pago para las compras

3.2. Requerimientos no funcionales

* El tiempo de carga del software no debe de exceder los 5 segundos.
* En dado caso de que el correo o la contraseña del usuario sean incorrectos, se mostrará un mensaje en donde se le hará saber sobre el error.

3.3. Interfaz de usuario:  
  
Aquí agregare lo de Figma en cuanto lo termine a la brevedad

estos ultimos 3 puntos (interfaz de usuario, interfaz externa y restricciones de diseño) tambien los vas a hacer josé

3.4 Interfaces externas

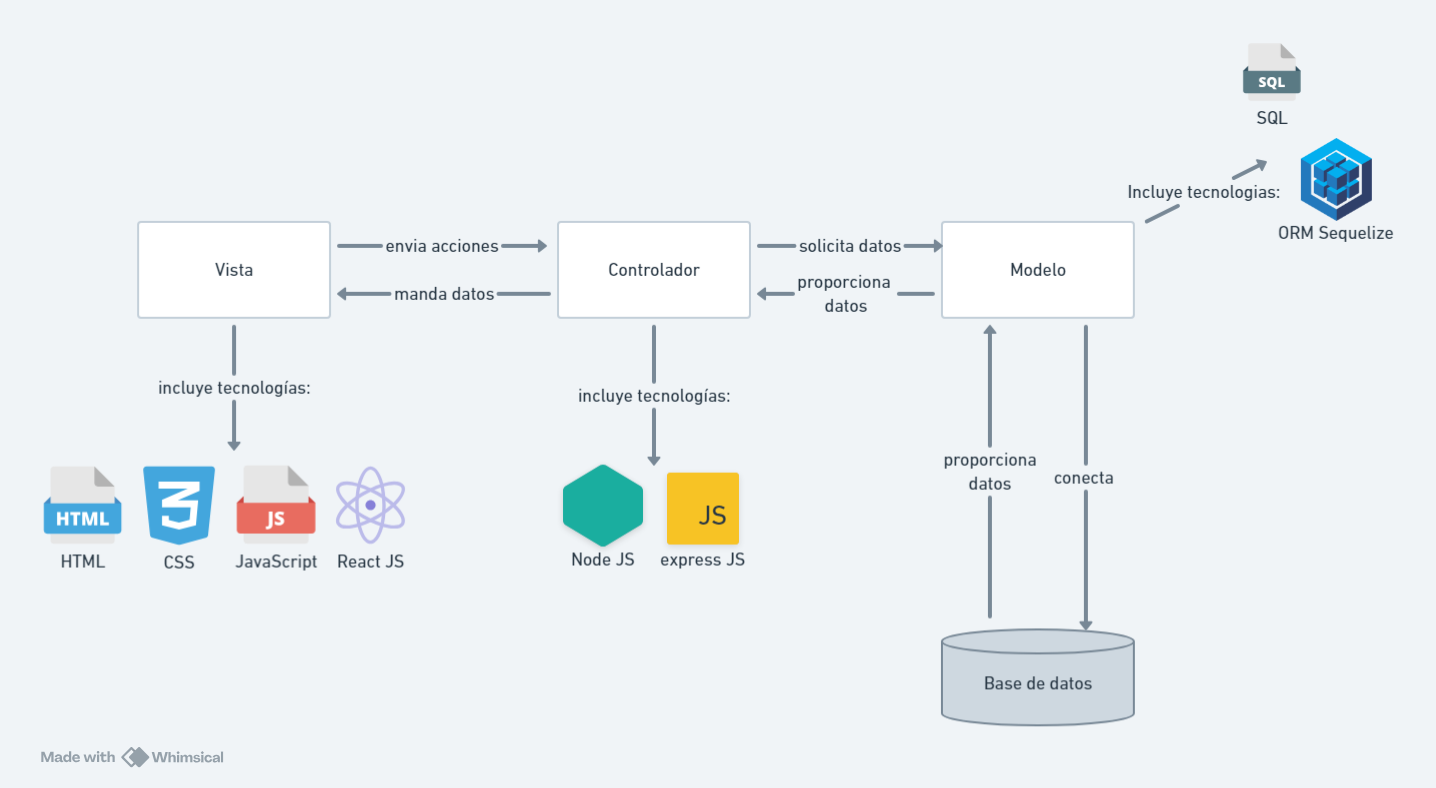
Define cómo se conectará el sistema con otros sistemas o dispositivos.

3.5 Restricciones de diseño

Detalla restricciones impuestas al diseño, como tecnologías específicas que deben utilizarse.

1. **Diagramas y modelos**

4.1. Diagrama de arquitectura

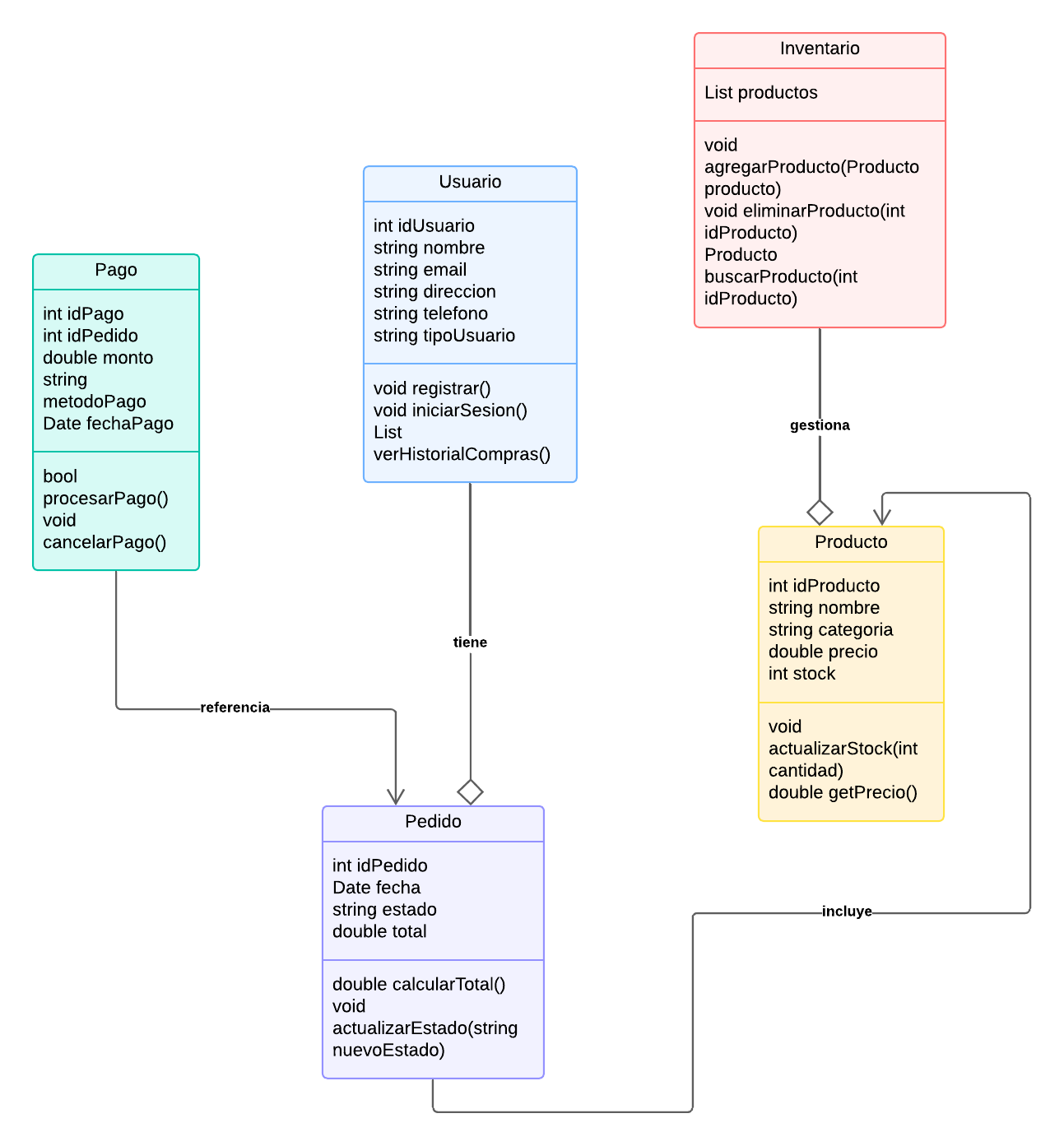


### Descripción del Diagrama MVC

1. Vista:  
   * Tecnologías: HTML, CSS, JavaScript, React JS.
   * Función: La vista se encarga de presentar datos al usuario y de enviar acciones del usuario al controlador. Es la interfaz de usuario que interactúa directamente con el usuario final.   
     En este diagrama, la vista utiliza tecnologías web para la presentación y la interacción.
2. Controlador:  
   * Tecnologías: Node JS, Express JS.
   * Función: El controlador actúa como intermediario entre la vista y el modelo. Recibe las entradas de la vista, solicita datos del modelo y devuelve los datos procesados a la vista para su presentación. Aquí, se utilizan Node JS y Express JS para manejar las solicitudes y la lógica del servidor.
3. Modelo:  
   * Tecnologías: SQL, ORM Sequelize.
   * Función: El modelo gestiona los datos del sistema. Se encarga de acceder a la base de datos y realizar operaciones de creación, lectura, actualización y eliminación (CRUD). En este diagrama, se utiliza SQL para la base de datos y Sequelize como ORM (Object-Relational Mapping) para interactuar con la base de datos de una manera más sencilla.
4. Base de datos:  
   * Función: La base de datos almacenará la información de la aplicación. Esta proporcionará datos al modelo cuando se solicitan.

### Interacciones en el Diagrama

* Vista a Controlador: La vista envía las acciones del usuario al controlador. Por ejemplo, cuando un usuario hace clic en un botón, la vista envía esta acción al controlador.
* Controlador a Modelo: El controlador solicita datos al modelo para procesar la solicitud del usuario. Por ejemplo, si el usuario quiere ver una lista de productos, el controlador pide estos datos al modelo.
* Modelo a Base de Datos: El modelo se conecta a la base de datos para recuperar o guardar datos. Por ejemplo, el modelo consulta la base de datos para obtener la lista de productos.
* Controlador a Vista: Una vez que el controlador recibe los datos del modelo, los procesa (si es necesario) y los envía de vuelta a la vista para su presentación.  
    
    
    
    
    
    
    
    
  4.2. Diagrama de clases:



El diagrama de clases muestra la estructura de un sistema de gestión de pedidos y pagos, con las siguientes clases y sus relaciones:

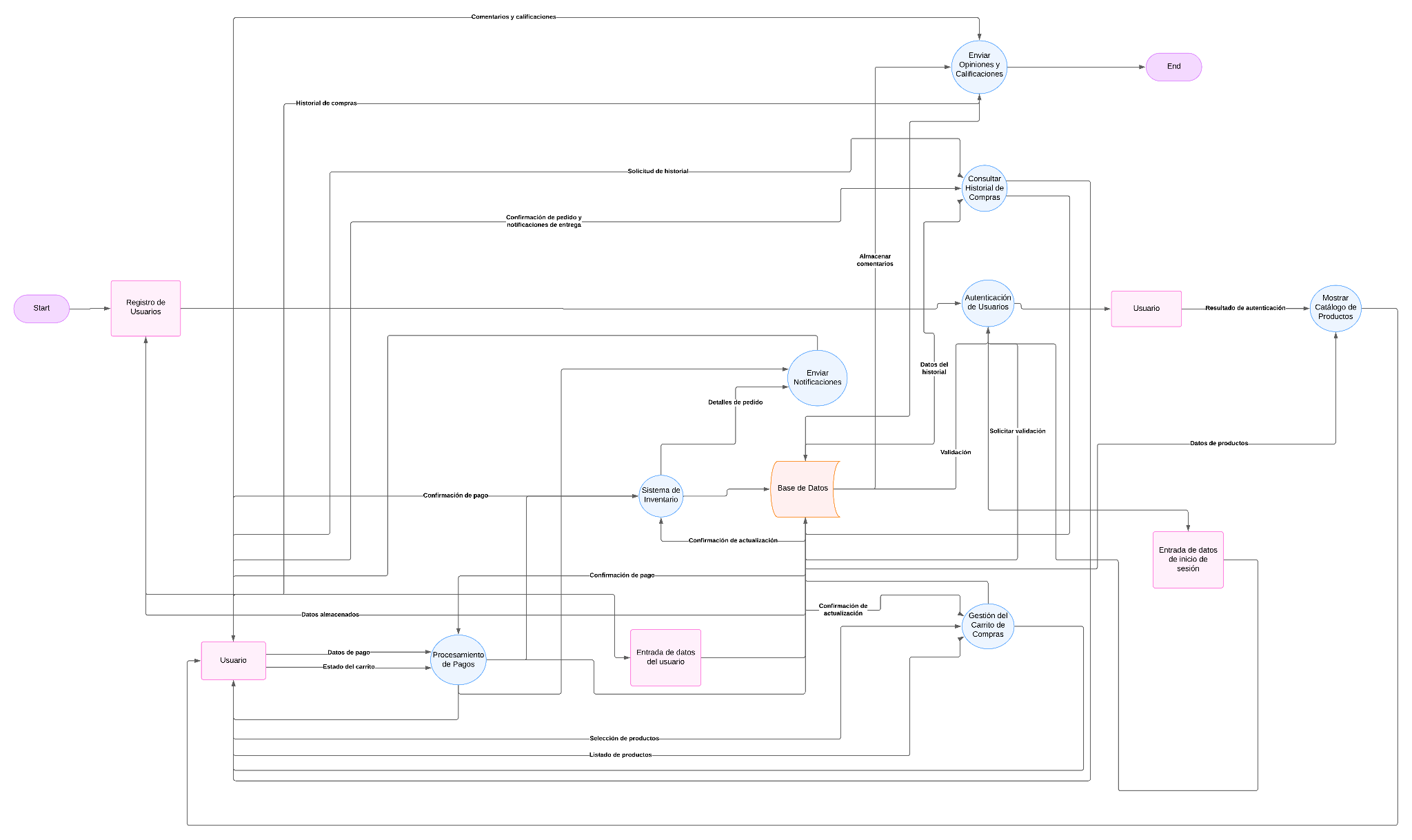
### Clases y Atributos

1. Pago:  
   * Atributos:
     + int idPago
     + int idPedido
     + double monto
     + string metodoPago
     + Date fechaPago
   * Métodos:  
     + bool procesarPago()
     + void cancelarPago()
   * Relación: Referencia a Pedido.
2. Usuario:  
   * Atributos:  
     + int idUsuario
     + string nombre
     + string email
     + string direccion
     + string telefono
     + string tipoUsuario
   * Métodos:  
     + void registrar()
     + void iniciarSesion()
     + List verHistorialCompras()
   * Relación: tiene un Pedido.
3. Pedido:  
   * Atributos:  
     + int idPedido
     + Date fecha
     + string estado
     + double total
   * Métodos:  
     + double calcularTotal()
     + void actualizarEstado(string nuevoEstado)
   * Relación: incluye Producto.
4. Producto:  
   * Atributos:  
     + int idProducto
     + string nombre
     + string categoria
     + double precio
     + int stock
   * Métodos:  
     + void actualizarStock(int cantidad)
     + double getPrecio()
5. Inventario:  
   * Atributos:  
     + List productos
   * Métodos:  
     + void agregarProducto(Producto producto)
     + void eliminarProducto(int idProducto)
     + Producto buscarProducto(int idProducto)
   * Relación: gestiona Producto.

### Relaciones:

* Pago tiene una relación con Pedido, indicando que un pago está asociado a un pedido.
* Usuario tiene una relación con Pedido, indicando que un usuario puede hacer múltiples pedidos.
* Pedido incluye Producto, indicando que un pedido puede contener múltiples productos.
* Inventario gestiona Producto, indicando que el inventario mantiene un registro de productos disponibles.

4.3. Diagrama de flujo de datos:

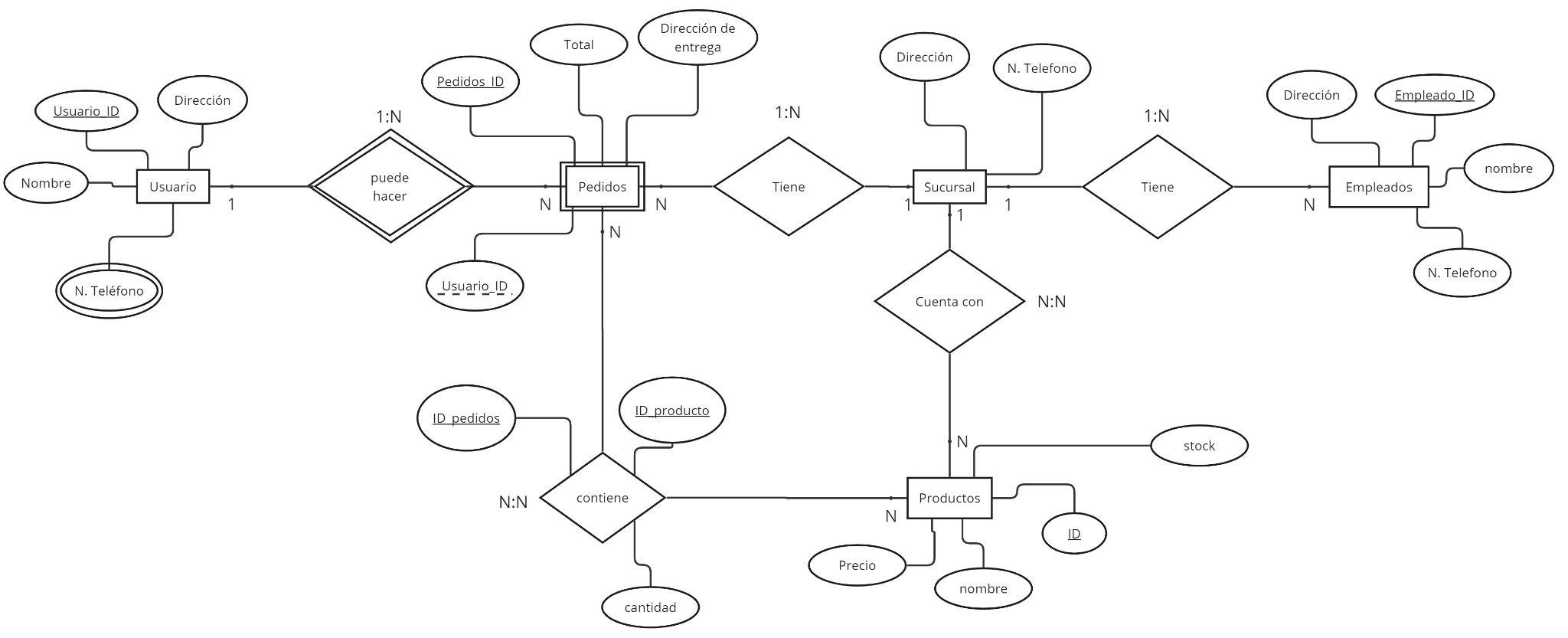


El diagrama de flujo de datos muestra el proceso de un sistema de compras en línea. A continuación, se describirán cada paso y su interacción:

1. Inició: El proceso comienza con el registro de usuarios.
2. Registro de Usuarios: Los usuarios se registran en el sistema.
3. Autenticación de Usuarios: Los usuarios autenticados pueden acceder al sistema.
4. Mostrar Catálogo de Productos: Los usuarios autenticados pueden ver el catálogo de productos.
5. Gestión del Carrito de Compras: Los usuarios seleccionan productos y gestionan su carrito de compras.
6. Entrada de Datos del Usuario: Los usuarios ingresan datos necesarios para la compra.
7. Procesamiento de Pagos: Se procesa el pago de los productos seleccionados.
8. Sistema de Inventario: Se actualiza el inventario con los productos comprados.
9. Base de Datos: Almacena y actualiza la información de usuarios, productos, y compras.
10. Enviar Notificaciones: Se envían notificaciones de confirmación de pedido y entrega.
11. Consultar Historial de Compras: Los usuarios pueden consultar su historial de compras.
12. Enviar Opiniones y Calificaciones: Los usuarios pueden enviar opiniones y calificaciones sobre los productos comprados.
13. Fin: El proceso finaliza.

El diagrama es relevante porque muestra de manera clara y detallada cómo interactúan los diferentes componentes de un sistema de compras en línea, desde el registro del usuario hasta la gestión del inventario y el envío de notificaciones.

4.4. Diagrama Entidad-Relación



El diagrama entidad-relación (ER) representa la estructura de una base de datos para un sistema de gestión de pedidos. A continuación, se explicarán los componentes y relaciones que se muestran en el diagrama:

### Entidades y Atributos

1. Usuario:  
   * Atributos: Usuario\_ID, Nombre, Dirección, N. Teléfono.
   * Relación: Un usuario puede hacer muchos pedidos (1:N).
2. Pedidos:  
   * Atributos: Pedidos\_ID, Total, Dirección de entrega.
   * Relación: Un pedido tiene una sucursal (N:1), un pedido contiene muchos productos (N:N).
3. Sucursal:  
   * Atributos: Dirección, N. Teléfono.
   * Relación: Una sucursal tiene muchos empleados (1:N), una sucursal tiene muchos pedidos (1:N).
4. Empleados:  
   * Atributos: Empleado\_ID, Nombre, Dirección, N. Teléfono.
   * Relación: Un empleado trabaja en una sucursal (N:1).
5. Productos:  
   * Atributos: ID, Nombre, Precio, Stock.
   * Relación: Un producto puede estar en muchos pedidos (N:N).
6. Otros requerimientos

5.1. Seguridad

5.2. Rendimiento

5.3. Mantenimiento y actualización

5.4. Pruebas y validación