**SEMINARIO DE PRACTICA**

**LICENCIATURA EN INFORMATICA**

**Desarrollo de un Sistema de E-commerce para Tintorería Pardini**

****

**ALUMNO: ARMELLA MARCOS SANTIAGO**

**Legajo: VINF015042**

**DNI:33384528**

**PROFESOR: PABLO ALEJANDRO VIRGOLINI**

**CATEDRA: SEMINARIO DE PRÁCTICA DE INFORMATICA 05-AUG-2024 07-DEC-2024**

**FECHA DE ENTREGA:06/10/2024**

**Índice**

Introducción \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Justificacion\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Propuesta de Solución \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Definiciones del Proyecto y del Sistema\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Objetivo General del Proyecto\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Objetivo General del Sistema\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

Alcance del Sistema\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

Elicitación de Requerimientos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

Actividades de Elicitación Realizadas\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

Herramientas Utilizadas\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

Participantes del Proceso\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

DOMINIO\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6

Requerimientos Funcionales\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

Requerimientos No Funcionales\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

Caso de uso\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8

Etapa de Analisis\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_9

Etapa de Diseño\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_13

Etapa de implementacion\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_14

Etapa de pruebas\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_16

DER y Base de datos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_17

Definiciones de Comunicación\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_27

POO (Programación Orientada a Objetos)--------------------------------------------------------------------27

**Introducción**

Tintorería Pardini, fundada en 1969 por Eduardo Pardini y su esposa Susana Micaela Gómez de Pardini, ha sido un negocio icónico en la ciudad de Neuquén. Inicialmente ubicado en la calle Roca 191 y, desde 1978, en Belgrano 134 Neuquen capital, la tintorería ha ganado reputación por ofrecer servicios de alta calidad en la limpieza de prendas, alfombras y pieles. A lo largo de sus más de 50 años, han confiado en ella personalidades del ámbito político, cultural y artístico, siendo Sandro uno de los ejemplos más destacados. A pesar de su prestigiosa trayectoria, la empresa enfrenta desafíos en su modernización tecnológica, particularmente en la implementación de soluciones de comercio electrónico que optimicen su canal de ventas.

**Justificación**

El aislamiento social preventivo experimentado en 2020 puso de relieve la necesidad de diversificar los canales de venta, ya que los métodos tradicionales, como la atención presencial y telefónica, mostraron limitaciones importantes. Aunque Tintorería Pardini ha explorado plataformas de terceros como Facebook Marketplace, la empresa reconoce la necesidad de desarrollar un sistema de e-commerce propio. Este proyecto tiene como objetivo ampliar el alcance de la tintorería, mejorar la experiencia del cliente y ofrecer métodos de pago más modernos y eficientes, como Mercado Pago y tarjetas de crédito. El nuevo sistema también permitirá una mejor gestión de pedidos, evitando la dependencia de comprobantes físicos en papel.

**Propuesta de Solución**

Para abordar los desafíos actuales, la propuesta es implementar un sistema de comercio electrónico que permita a los clientes acceder a los servicios de Tintorería Pardini desde la comodidad de sus hogares. Este sistema permitirá gestionar pedidos, realizar pagos electrónicos, y automatizar la comunicación con los clientes a través de notificaciones. Además, se integrará un sistema de inventario para asegurar que el proceso de seguimiento de prendas sea más eficiente y transparente.

**Definiciones del Proyecto y del Sistema**

**Objetivo General del Proyecto**

**Objetivo General:**

Desarrollar un sistema de comercio electrónico para Tintorería Pardini que permita gestionar de manera eficiente la comercialización de sus servicios, ofreciendo una plataforma accesible y moderna para que los clientes puedan solicitar servicios en línea, realizar pagos electrónicos y recibir notificaciones automáticas, mejorando así la experiencia del cliente y optimizando los procesos internos de la empresa.

Este objetivo busca aumentar la competitividad del negocio en el entorno digital, atraer nuevos clientes, y facilitar el manejo de la relación con los usuarios actuales.

**Objetivo General del Sistema**

**Objetivo General:**

Desarrollar un sistema de comercio electrónico eficiente, seguro y fácil de usar que permita a los clientes de Tintorería Pardini gestionar sus pedidos de manera autónoma y digital, mientras optimiza los procesos internos de la empresa, como el control de inventario, la generación de reportes, y la gestión de pagos electrónicos.

Este sistema facilitará la interacción entre la empresa y sus clientes, mejorará la calidad del servicio, y ayudará a modernizar la gestión operativa de la tintorería.

**Alcance del Sistema:**

Implementación de un sistema de comercio electrónico que permita a los usuarios seleccionar servicios, agendar fechas de entrega y retiro, y gestionar pagos.

Integración de métodos de pago digitales como Mercado Pago y tarjetas de crédito.

Incorporación de un módulo de gestión de inventario para el seguimiento de prendas ingresadas y listas para retiro.

Creación de un sistema de notificaciones que avise al cliente cuando sus prendas estén listas para retirar o si necesitan alguna acción adicional.

**Elicitación de Requerimientos**

Para el desarrollo del Sistema de E-commerce para Tintorería Pardini, el proceso de elicitación se centró en comprender profundamente las necesidades del negocio, los procesos actuales y las expectativas de los usuarios finales. Esto se realizó mediante la combinación de encuestas y entrevistas, dos técnicas complementarias que permitieron recolectar tanto datos cuantitativos como cualitativos.

**Actividades de Elicitación Realizadas**

**Encuestas:**

**Objetivo:** Obtener una visión general de las necesidades y expectativas de un grupo más amplio de clientes, así como medir su disposición y familiaridad con el uso de plataformas de comercio electrónico.

**Metodología:** Se diseñaron encuestas con preguntas cerradas para recoger datos cuantitativos. Estas se distribuyeron entre clientes frecuentes de Tintorería Pardini y posibles nuevos usuarios que podrían beneficiarse del sistema de e-commerce.

**Resultados clave:**

Un 65% de los encuestados expresó interés en solicitar servicios de tintorería en línea, siempre que el proceso fuera sencillo y confiable.

Un 78% manifestó la necesidad de contar con múltiples opciones de pago digital.

50% consideró que la notificación automática de estado de pedidos sería una funcionalidad muy útil.

**Entrevistas:**

**Objetivo:** Profundizar en las necesidades específicas de los usuarios y comprender a fondo los problemas que enfrentan con los métodos actuales de solicitud de servicios, además de explorar las expectativas con respecto a un sistema de e-commerce.

**Metodología:** Se realizaron entrevistas semiestructuradas con los dueños de Tintorería Pardini, empleados encargados de la recepción de pedidos, y algunos clientes clave. Las preguntas fueron abiertas, permitiendo explorar en detalle temas específicos como la gestión de inventario, el seguimiento de prendas, y las preferencias de pago.

**Resultados clave:**

Los dueños del negocio destacaron la necesidad de un sistema que reduzca la dependencia del papel y facilite el seguimiento de pedidos.

El personal de la tintorería expresó su interés en una mejor organización de los pedidos para evitar errores y mejorar la eficiencia operativa.

Los clientes valoran la posibilidad de recibir notificaciones automáticas, pero también solicitaron una opción para realizar cambios en los pedidos sin necesidad de acudir físicamente al local.

**Herramientas Utilizadas**

**Google Forms:** Se utilizó para el diseño y distribución de las encuestas, permitiendo una recopilación eficiente de los datos cuantitativos.

**Entrevistas semiestructuradas en persona y vía Zoom:** Estas fueron útiles para explorar con más profundidad las necesidades de los stakeholders clave. Se grabaron y transcribieron las entrevistas para su posterior análisis.

**Técnicas de análisis de resultados:** Los datos obtenidos de las encuestas fueron analizados estadísticamente para identificar patrones y tendencias en las necesidades del cliente. Por otro lado, las entrevistas se analizaron mediante técnicas de codificación para identificar temas recurrentes y puntos críticos a considerar en el desarrollo del sistema.

**Participantes del Proceso**

**Propietarios de Tintorería Pardini:** Ofrecieron una visión clara sobre los problemas actuales de gestión y sus expectativas con respecto a la modernización del negocio.

**Personal de la Tintorería:** Brindaron información valiosa sobre los procesos internos y los desafíos operativos que enfrenta el negocio diariamente.

**Clientes:** Un grupo de clientes frecuentes y potenciales participó en encuestas y entrevistas, proporcionando retroalimentación sobre su experiencia con el negocio y sus expectativas hacia un sistema de e-commerce.

**Conclusiones del Proceso de Elicitación**

Existe una demanda latente de una plataforma digital que facilite la solicitud de servicios y el seguimiento de prendas.

La diversificación de métodos de pago es una prioridad, ya que los clientes buscan más opciones de pago electrónico.

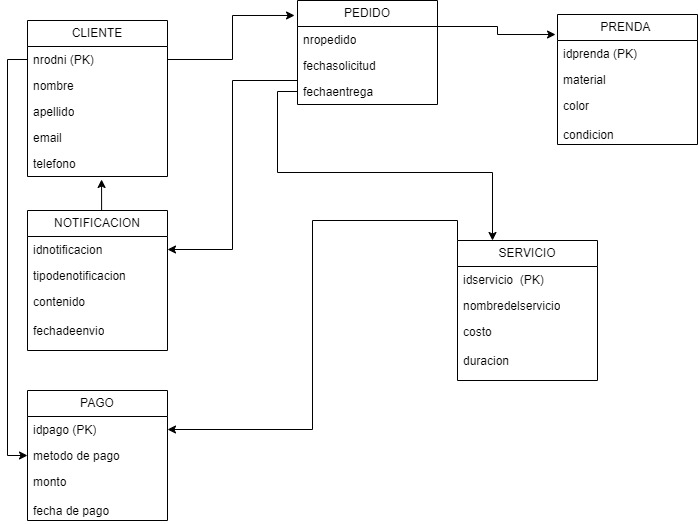
La notificación automática es vista como una mejora significativa en la experiencia del cliente, pero debe permitir flexibilidad para realizar cambios en el pedido.

Es esencial un sistema de gestión de inventario más robusto que permita al personal de la tintorería llevar un control detallado de los pedidos, minimizando errores y mejorando la eficiencia.

Estas actividades permitieron formular los requerimientos del sistema de manera precisa, asegurando que el desarrollo futuro cumpla con las expectativas tanto del negocio como de sus clientes.

**DOMINIO**

El diagrama de dominio es una representación visual que describe los conceptos clave, atributos y relaciones dentro del mundo real de un sistema específico, en este caso, Tintorería Pardini. A diferencia de un diagrama de clases en programación, el diagrama de dominio se centra en los elementos del negocio y no en la implementación técnica.



**Elicitación de Requerimientos**

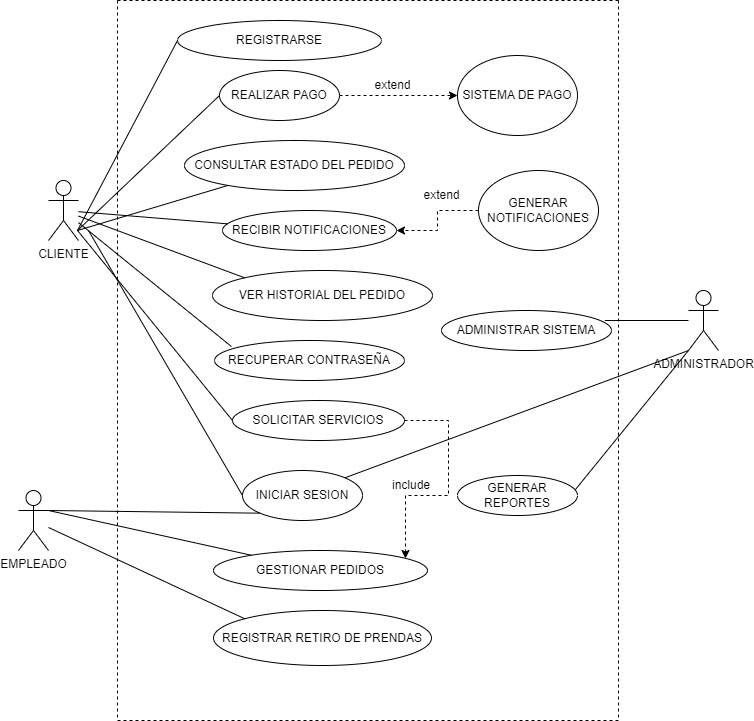
**Requerimientos Funcionales**

|  |  |  |
| --- | --- | --- |
| ID | Requerimiento Funcional | Descripción |
| RF01 | Solicitud de servicios en línea | El sistema debe permitir a los clientes seleccionar y solicitar servicios de tintorería a través del sitio web. |
| RF02 | Gestión de pedidos en tiempo real | Los empleados deben poder gestionar los pedidos (recibir, procesar y marcar como completados) de forma digital. |
| RF03 | Integración de métodos de pago | El sistema debe permitir el pago en línea mediante plataformas como Mercado Pago, tarjetas de crédito o débito. |
| RF04 | Notificaciones automáticas | El sistema debe enviar notificaciones automáticas a los clientes sobre el estado de sus pedidos (recibido, en proceso, listo para recoger). |
| RF05 | Panel de control para la administración | La administración debe poder supervisar los pedidos, gestionar el inventario y generar reportes de ventas y actividades. |
| RF06 | Historial de pedidos | Los clientes deben poder acceder al historial de pedidos realizados y ver detalles sobre cada uno. |
| RF07 | Seguimiento de estado de las prendas | El sistema debe permitir a los clientes consultar el estado actual de sus prendas en proceso de limpieza. |
| RF08 | Registro y gestión de clientes | El sistema debe gestionar un registro de clientes y permitir la actualización de información de contacto. |
| RF09 | Fidelización de clientes | El sistema debe ofrecer beneficios a los clientes frecuentes, como descuentos o promociones exclusivas. |

**Requerimientos No Funcionales**

|  |  |  |
| --- | --- | --- |
| ID | Requerimiento No Funcional | Descripción |
| RNF01 | Escalabilidad | El sistema debe ser capaz de manejar un creciente número de clientes y pedidos sin afectar el rendimiento. |
| RNF02 | Seguridad de los datos | Los datos personales y financieros de los clientes deben ser tratados con protocolos de seguridad como encriptación y SSL. |
| RNF03 | Interfaz fácil de usar | La interfaz del sistema debe ser intuitiva, tanto para los clientes como para los empleados de la tintorería. |
| RNF04 | Disponibilidad multiplataforma | El sistema debe ser accesible desde cualquier dispositivo, incluyendo computadoras, smartphones y tablets. |
| RNF05 | Rendimiento | El sistema debe procesar los pedidos y actualizar los estados de las prendas con una latencia mínima. |
| RNF06 | Disponibilidad | El sistema debe estar disponible al menos el 99% del tiempo, minimizando interrupciones. |
| RNF07 | Mantenimiento | El sistema debe ser fácil de mantener y actualizar, con capacidad para agregar nuevas funcionalidades. |
| RNF08 | Cumplimiento de normativas de pago | El sistema debe cumplir con las normativas de seguridad y privacidad exigidas por las plataformas de pago en línea. |
| RNF09 | Personalización | La plataforma debe permitir la personalización de la interfaz de usuario según las necesidades de la empresa. |

**Inicio del Análisis: Casos de Uso**



**Etapa de análisis**

En esta fase, se realiza un estudio detallado del modelo de negocio y las necesidades de los usuarios finales. Continuando con los casos de usos, se completan las fichas de los mismos.

**Caso de Uso 1: Registrar Pedido**

|  |  |
| --- | --- |
| Elemento | Descripción |
| Nombre del Caso | Registrar Pedido |
| Actor Principal | Cliente |
| Actores Secundarios | Empleado de la tintorería |
| Descripción | El cliente lleva prendas a la tintorería y el empleado registra el pedido en el sistema. Se ingresan detalles como tipo de prenda, servicio solicitado y cantidad. |
| Precondiciones | - El cliente debe estar registrado o ser nuevo.  - El empleado debe estar autenticado. |
| Postcondiciones | - El pedido queda registrado en la base de datos.  - Se emite un comprobante de pedido. |
| Flujo Principal | 1. El cliente lleva prendas a la tintorería.  2. El empleado selecciona "Registrar Pedido".  3. Se ingresa la información de las prendas y servicios.  4. El sistema calcula el costo.  5. El cliente confirma el pedido.  6. El sistema genera el comprobante. |
| Excepciones | - Si el sistema falla, el empleado debe registrar manualmente el pedido. |

**Caso de Uso 2: Consultar Estado de Pedido**

|  |  |
| --- | --- |
| Elemento | Descripción |
| Nombre del Caso | Consultar Estado de Pedido |
| Actor Principal | Cliente |
| Actores Secundarios | Empleado |
| Descripción | El cliente o el empleado consulta el estado de un pedido (en proceso, listo para retiro, entregado). |
| Precondiciones | - El pedido debe estar registrado.  - El cliente o empleado debe estar autenticado. |
| Postcondiciones | - El cliente o empleado puede ver el estado del pedido. |
| Flujo Principal | 1. El cliente solicita consultar estado.  2. El empleado busca el pedido en el sistema.  3. El sistema muestra el estado actual del pedido. |
| Flujo Alternativo | - Si el pedido no existe, se notifica al cliente y se solicita verificación. |
| Excepciones | - Si el sistema no está disponible, el estado no puede ser consultado en ese momento. |

**Caso de Uso 3: Realizar Pago**

|  |  |
| --- | --- |
| Elemento | Descripción |
| Nombre del Caso | Realizar Pago |
| Actor Principal | Cliente |
| Actores Secundarios | Empleado |
| Descripción | El cliente realiza el pago del servicio prestado, utilizando efectivo, tarjeta o Mercado Pago. |
| Precondiciones | - El pedido debe estar completado y listo para retiro.  - El cliente debe tener un saldo pendiente. |
| Postcondiciones | - El sistema registra el pago como completado.  - El pedido se actualiza como "Pagado". |
| Flujo Principal | 1. El cliente llega para retirar las prendas.  2. El empleado busca el pedido.  3. El cliente selecciona el método de pago.  4. El empleado registra el pago.  5. Se genera el comprobante de pago. |
| Flujo Alternativo | - Si el cliente no paga el total, se registra el pago parcial y se deja saldo pendiente. |
| Excepciones | - Si falla la terminal de pago, el cliente puede elegir otro método de pago. |

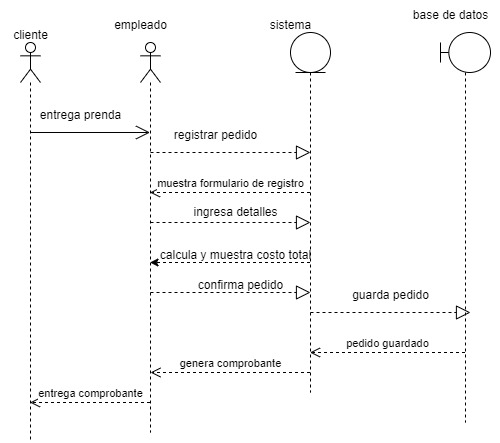
**Caso de Uso 4: Registrar Cliente**

|  |  |
| --- | --- |
| Elemento | Descripción |
| Nombre del Caso | Registrar Cliente |
| Actor Principal | Empleado |
| Descripción | El empleado registra a un nuevo cliente en el sistema con sus datos de contacto. |
| Precondiciones | - El empleado debe estar autenticado. |
| Postcondiciones | - El cliente queda registrado en la base de datos. |
| Flujo Principal | 1. El cliente solicita realizar un pedido.  2. El empleado selecciona "Registrar Cliente".  3. Se ingresa la información del cliente.  4. El sistema valida y guarda el registro. |
| Flujo Alternativo | - Si el cliente ya existe, el sistema notifica al empleado para evitar duplicados. |
| Excepciones | - Si faltan datos, el sistema solicita la información faltante. |

Vamos a detallar los diagramas de secuencia para algunos de los casos de uso mencionados. Estos diagramas mostrarán la interacción entre los actores y el sistema para llevar a cabo cada caso de uso.

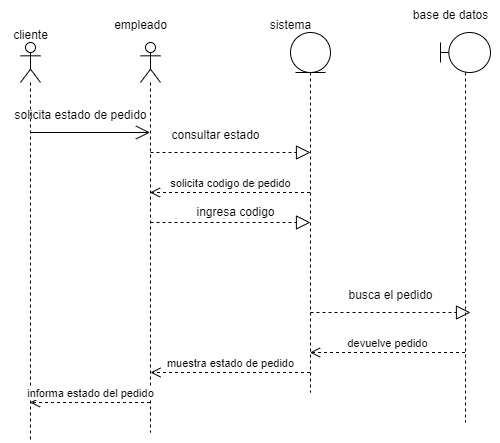
**Diagrama de Secuencia 1: Registrar Pedido**

El cliente lleva prendas a la tintorería, el empleado las registra en el sistema, se ingresan los detalles, y el sistema genera un comprobante.



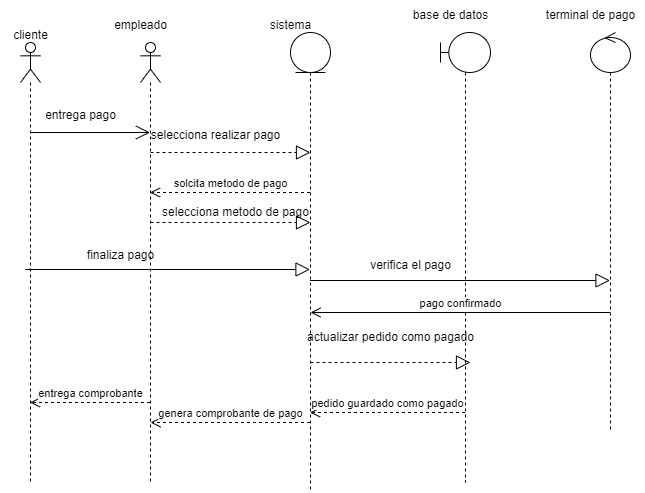
**Diagrama de Secuencia 2: Consultar Estado de Pedido**

El cliente solicita al empleado el estado de su pedido, y el sistema le proporciona la información requerida.



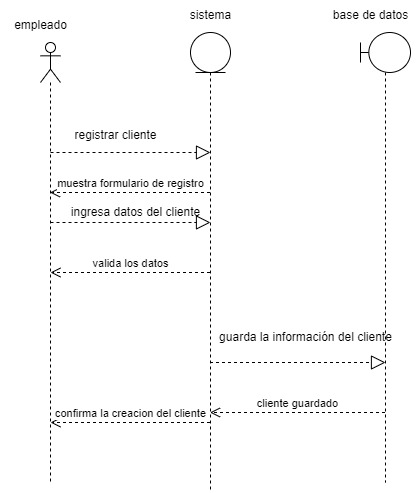
**Diagrama de Secuencia 3: Realizar Pago**

El cliente realiza el pago del pedido, y el sistema lo registra como completado.



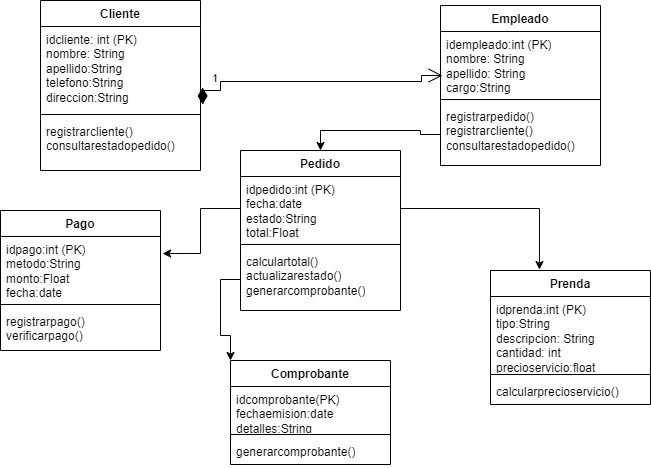
**Diagrama de Secuencia 4: Registrar Cliente**

El empleado registra a un nuevo cliente en el sistema con sus datos de contacto.



**Etapa de diseño**

El diseño es una fase crítica en el desarrollo del sistema, ya que implica traducir los requerimientos en una arquitectura de software adecuada. El enfoque principal es definir la estructura del sistema y la base de datos.

**Diagrama de Clases: Sistema de Gestión Tintorería Pardini**

Este esquema refleja las principales operaciones del sistema y sus relaciones, destacando la gestión de clientes, pedidos, pagos y el proceso de emisión de comprobantes.

**Etapa de implementación**

En esta etapa de implementación, vamos a representar cómo se desplegará el sistema en un entorno físico, utilizando un Diagrama de Despliegue, además de la integración de los Requerimientos No Funcionales (RNF) mencionados.

**Diagrama de Despliegue del Sistema de Tintorería Pardini**

Este diagrama representará los componentes clave de software y hardware que se utilizarán para implementar el sistema. Se identifican los servidores, dispositivos y bases de datos involucrados, junto con las conexiones entre ellos.

**Componentes Principales:**

**Servidor de Aplicaciones:**

* Descripción: Servidor donde se despliega la aplicación en Java.
* Componente de Software: Sistema de Gestión de la Tintorería (Aplicación Java).
* Relación: Interactúa con la Base de Datos MySQL y el servidor de pagos.

**Servidor de Base de Datos:**

* Descripción: Servidor dedicado a la base de datos MySQL.
* Componente de Software: Base de Datos MySQL.
* Relación: Conecta con el servidor de aplicaciones para la persistencia de los datos.

**Servidor Web (XAMPP):**

* Descripción: Utilizado como entorno de desarrollo web y servidor Apache, donde se despliega la interfaz web del sistema.
* Componente de Software: Apache Server, PHP, y MySQL (a través de XAMPP).
* Relación: Conecta con el servidor de aplicaciones para mostrar la interfaz web al usuario final.

**Terminal de Pago:**

* Descripción: Sistema de pagos que procesa transacciones con tarjeta y otros métodos electrónicos.
* Componente de Software: API de integración con Mercado Pago o similar.
* Relación: Conecta con el servidor de aplicaciones para validar y procesar pagos.

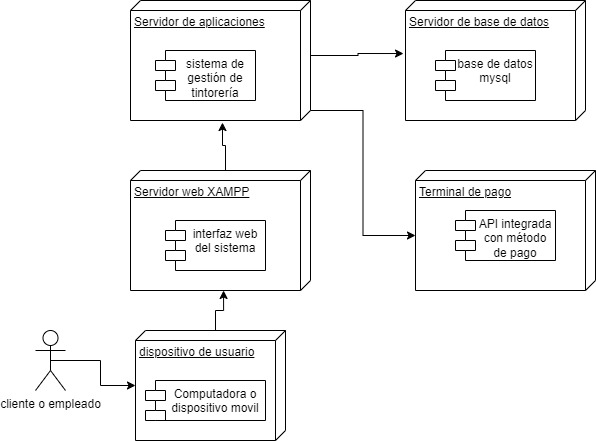
**Dispositivos de Usuario (Clientes y Empleados):**

* Descripción: Computadoras o dispositivos móviles utilizados por clientes y empleados para interactuar con el sistema.
* Relación: Acceden al servidor web a través de un navegador o aplicación móvil.

**Requerimientos No Funcionales (RNF) y Funcionales (RF)**

**Requerimientos del Sistema**

|  |  |
| --- | --- |
| Requerimiento | Descripción |
| RNF01 | El sistema debe estar desarrollado en Java. |
| RNF02 | El sistema debe contar con una base de datos MySQL. |
| RNF03 | El sistema debe utilizar XAMPP como entorno de desarrollo web de código abierto. |
| RF01 | El sistema debe permitir a los empleados registrar pedidos de clientes. |
| RF02 | El sistema debe permitir consultar el estado de los pedidos. |
| RF03 | El sistema debe generar comprobantes de pago. |
| RF04 | El sistema debe permitir a los empleados registrar nuevos clientes. |
| RF05 | El sistema debe gestionar el registro y procesamiento de pagos. |



Este despliegue garantiza que el sistema será escalable, modular y eficiente, cumpliendo con los requisitos tanto funcionales como no funcionales.

**Etapa de pruebas**

La **etapa de pruebas** para el sistema de e-commerce de Tintorería Pardini tiene como objetivo validar que todos los componentes del sistema cumplan con los requerimientos funcionales y no funcionales establecidos. Durante esta etapa, se verifican tanto las versiones intermedias como la versión final del sistema. Los principales artefactos que se consideran en esta etapa son:

1. **Plan de pruebas**: define los tipos de pruebas que se van a realizar, el orden en el que se ejecutan y los criterios de aceptación.
2. **Modelo de pruebas**: incluye los casos de prueba, los procedimientos y los componentes a testear.
3. **Tratamiento de defectos**: identifica y documenta los defectos encontrados y su corrección.
4. **Evaluación de pruebas**: realiza un análisis de los resultados obtenidos tras la ejecución de las pruebas.

### ****Plan de Pruebas****

### A continuación, se presenta el plan de pruebas básico para el caso de uso CU001, **Registrar Pedido**. Se empleará la siguiente nomenclatura:

* **CP**: Caso de Prueba.
* Tercera letra: Tipo de prueba (C: Componente, S: Sistema, I: Integración).
* Cuarta y quinta letras: Número de prueba.

#### ****Plan de Pruebas para Tintorería Pardini****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ****Caso de Uso**** | ****Código Prueba**** | ****Tipo de Prueba**** | ****Técnica Propuesta**** | ****Observaciones**** |
| **CU001 (Registrar Pedido)** | **CPC01** | Componente | Caja Blanca | Caja blanca aplicada al método verificarDatosPedido() de la clase Pedido. |
| **CU001 (Registrar Pedido)** | **CPS02** | Sistema | Pruebas Funcionales | Verificación de requerimientos funcionales: validar que los pedidos se registran correctamente. |
| **CU003 (Consultar Estado Pedido)** | **CPF03** | Componente | Partición de Equivalencia y Frontera | Caja negra al método consultarEstado() de la clase Pedido. |

### ****Casos de Prueba****

A continuación, se presenta un ejemplo de **caso de prueba** para el sistema de Tintorería Pardini, que aplica la técnica de **Partición de Equivalencia y Análisis de Frontera** para el método verificarDatosPedido() de la clase Pedido. Este caso de prueba cubre los siguientes requerimientos:

#### ****Requerimientos Funcionales del Caso de Uso "Registrar Pedido"****

|  |  |
| --- | --- |
| ****Requerimiento**** | ****Descripción**** |
| **RF01** | El sistema debe validar que todos los datos ingresados en el pedido sean correctos (nombre del cliente, tipo de prenda, fecha de entrega, etc.). |
| **RF02** | El sistema debe mostrar un mensaje de error si faltan datos obligatorios en el formulario del pedido. |

### ****Partición de Equivalencia y Análisis de Frontera****

#### ****Clases de Equivalencias Válidas****

|  |  |
| --- | --- |
| ****Clase**** | ****Descripción**** |
| **CEV1** | Nombre del cliente con longitud entre 1 y 50 caracteres. |
| **CEV2** | Fecha de entrega válida (posterior a la fecha actual). |

#### ****Clases de Equivalencias Inválidas****

|  |  |
| --- | --- |
| ****Clase**** | ****Descripción**** |
| **CEI1** | Nombre del cliente con longitud mayor a 50 caracteres. |
| **CEI2** | Fecha de entrega anterior a la fecha actual. |

### ****Valores de Frontera****

#### ****Análisis de Valores de Frontera****

|  |  |  |
| --- | --- | --- |
| ****Clase**** | ****Límite Inferior**** | ****Límite Superior**** |
| **CEV1** | 1 carácter | 50 caracteres |
| **CEV2** | Fecha actual | Fecha posterior |
| **CEI1** | 51 caracteres | - |
| **CEI2** | Fecha anterior | - |

### ****Comportamiento del Sistema****

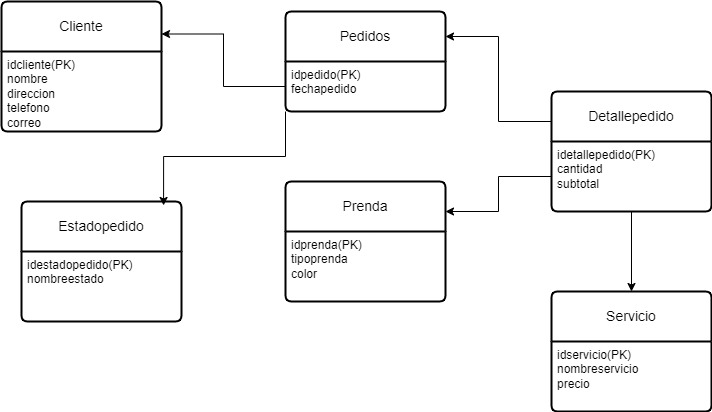
|  |  |  |
| --- | --- | --- |
| ****Dato de Entrada**** | ****Comportamiento Esperado**** | ****Mensaje del Sistema**** |
| Nombre vacío | No se registra pedido | "Nombre es obligatorio". |
| 1 carácter | Se registra el pedido | Pedido registrado. |
| 50 caracteres | Se registra el pedido | Pedido registrado. |
| 51 caracteres | No se registra pedido | "Nombre demasiado largo". |
| Fecha anterior | No se registra pedido | "Fecha de entrega inválida". |
| Fecha posterior | Se registra el pedido | Pedido registrado. |

Este plan de pruebas, con su respectivo modelo de pruebas y análisis de comportamiento, permite verificar que el sistema de e-commerce para Tintorería Pardini funcione según lo esperado en la gestión de pedidos y validaciones relacionadas. Se asegura que los datos ingresados por los usuarios sean consistentes y que el sistema emita mensajes claros en caso de errores, garantizando la calidad y la funcionalidad del sistema.

**Definición de la Base de Datos para el Sistema de Tintorería Pardini**

La base de datos para el sistema de e-commerce de Tintorería Pardini debe ser capaz de gestionar toda la información relacionada con los clientes, pedidos, prendas, servicios y estados de los pedidos. La base de datos relacional estará diseñada para garantizar la integridad de los datos, así como la eficiencia en las operaciones de consulta, inserción y actualización.

**Diagrama Entidad-Relación (DER)**



**Creación de las Tablas en MySQL para el Sistema de Tintorería Pardini**

Las siguientes instrucciones SQL crean las tablas necesarias para implementar la base de datos del sistema de gestión de la tintorería:

-- Creación de la tabla Cliente

CREATE TABLE Cliente (

idCliente INT AUTO\_INCREMENT PRIMARY KEY,

nombre VARCHAR(100) NOT NULL,

direccion VARCHAR(255),

telefono VARCHAR(20),

correo VARCHAR(100)

);

-- Creación de la tabla EstadoPedido

CREATE TABLE EstadoPedido (

idEstadoPedido INT AUTO\_INCREMENT PRIMARY KEY,

nombreEstado VARCHAR(50) NOT NULL

);

-- Creación de la tabla Pedido

CREATE TABLE Pedido (

idPedido INT AUTO\_INCREMENT PRIMARY KEY,

fechaPedido DATE NOT NULL,

idCliente INT,

idEstadoPedido INT,

FOREIGN KEY (idCliente) REFERENCES Cliente(idCliente),

FOREIGN KEY (idEstadoPedido) REFERENCES EstadoPedido(idEstadoPedido)

);

-- Creación de la tabla Prenda

CREATE TABLE Prenda (

idPrenda INT AUTO\_INCREMENT PRIMARY KEY,

tipoPrenda VARCHAR(50) NOT NULL,

color VARCHAR(50)

);

-- Creación de la tabla Servicio

CREATE TABLE Servicio (

idServicio INT AUTO\_INCREMENT PRIMARY KEY,

nombreServicio VARCHAR(100) NOT NULL,

precio DECIMAL(10, 2)

);

-- Creación de la tabla DetallePedido

CREATE TABLE DetallePedido (

idDetallePedido INT AUTO\_INCREMENT PRIMARY KEY,

idPedido INT,

idPrenda INT,

idServicio INT,

cantidad INT NOT NULL,

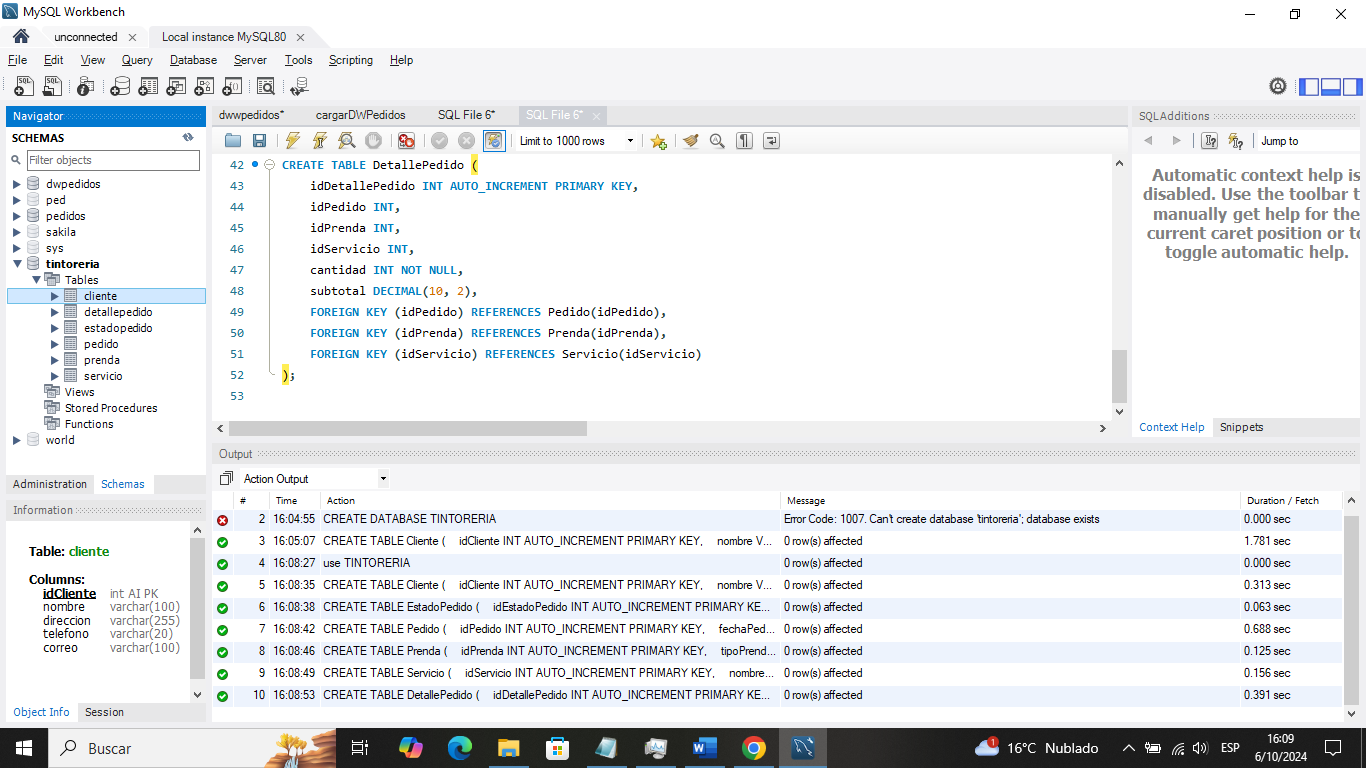
subtotal DECIMAL(10, 2),

FOREIGN KEY (idPedido) REFERENCES Pedido(idPedido),

FOREIGN KEY (idPrenda) REFERENCES Prenda(idPrenda),

FOREIGN KEY (idServicio) REFERENCES Servicio(idServicio)

);



**Inserción de Registros**

Para insertar datos de ejemplo en las tablas creadas:

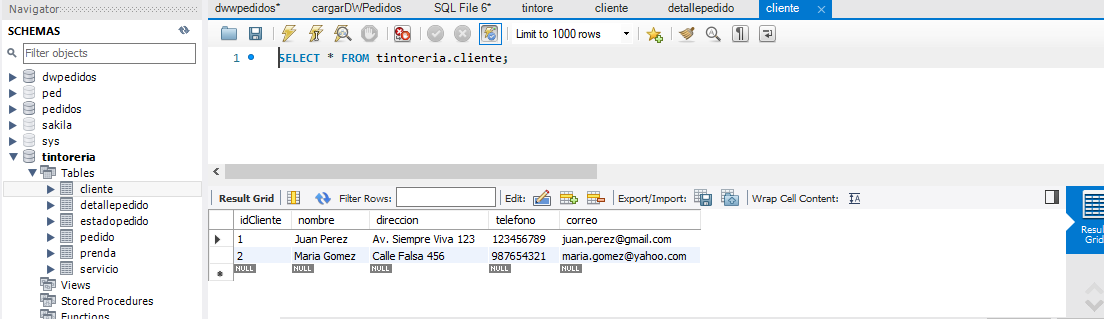
-- Insertando datos en la tabla Cliente

INSERT INTO Cliente (nombre, direccion, telefono, correo)

VALUES

('Juan Perez', 'Av. Siempre Viva 123', '123456789', 'juan.perez@gmail.com'),

('Maria Gomez', 'Calle Falsa 456', '987654321', 'maria.gomez@yahoo.com');



-- Insertando datos en la tabla EstadoPedido

INSERT INTO EstadoPedido (nombreEstado)

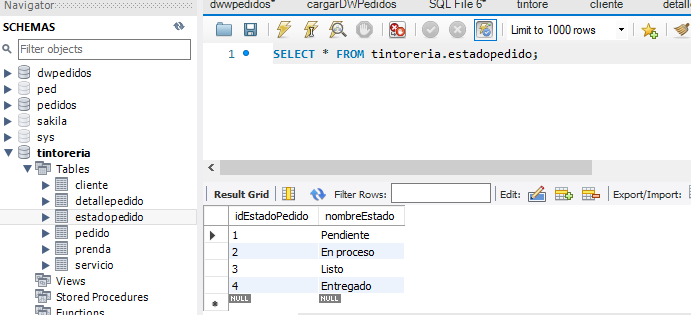
VALUES

('Pendiente'),

('En proceso'),

('Listo'),

('Entregado');



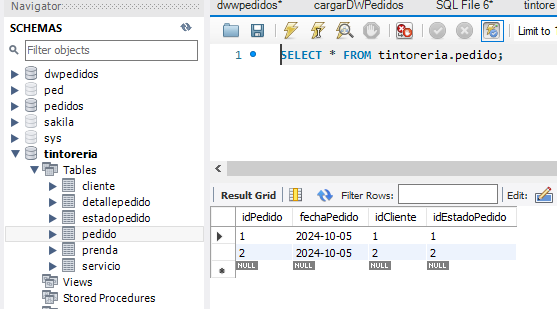
-- Insertando datos en la tabla Pedido

INSERT INTO Pedido (fechaPedido, idCliente, idEstadoPedido)

VALUES

('2024-10-05', 1, 1),

('2024-10-05', 2, 2);



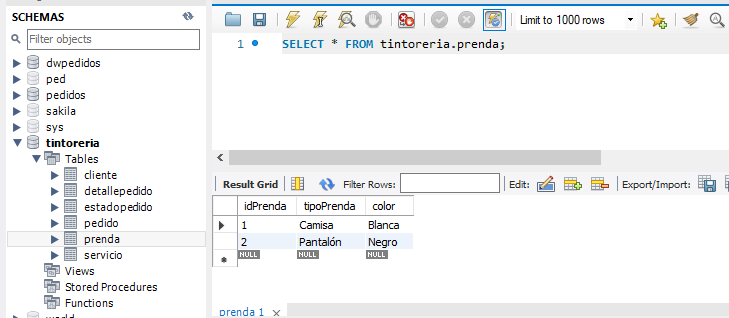
-- Insertando datos en la tabla Prenda

INSERT INTO Prenda (tipoPrenda, color)

VALUES

('Camisa', 'Blanca'),

('Pantalón', 'Negro');



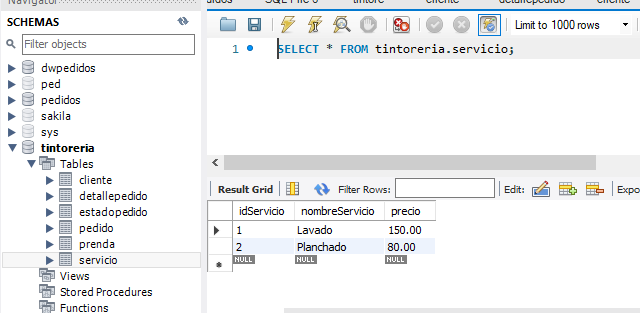
-- Insertando datos en la tabla Servicio

INSERT INTO Servicio (nombreServicio, precio)

VALUES

('Lavado', 150.00),

('Planchado', 80.00);



-- Insertando datos en la tabla DetallePedido

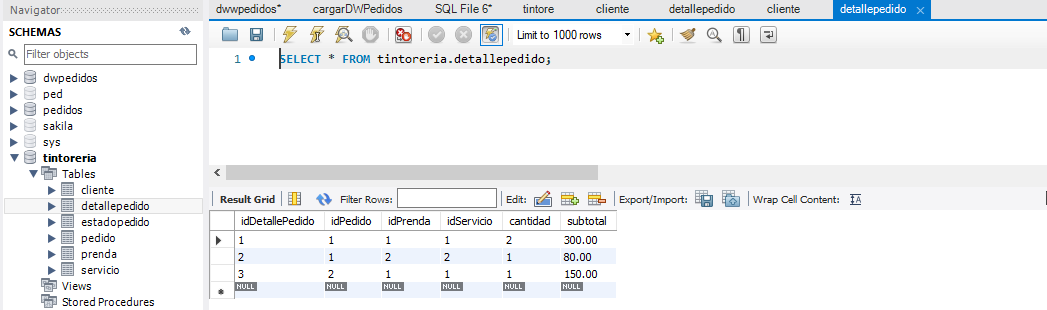
INSERT INTO DetallePedido (idPedido, idPrenda, idServicio, cantidad, subtotal)

VALUES

(1, 1, 1, 2, 300.00),

(1, 2, 2, 1, 80.00),

(2, 1, 1, 1, 150.00);



**Consultas SQL**

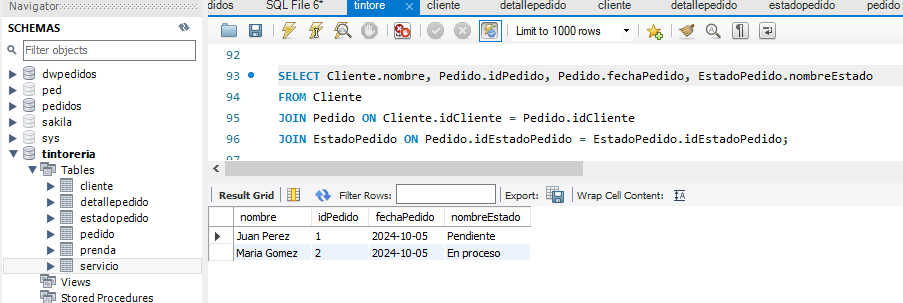
***Consulta de clientes y sus pedidos:***

SELECT Cliente.nombre, Pedido.idPedido, Pedido.fechaPedido, EstadoPedido.nombreEstado

FROM Cliente

JOIN Pedido ON Cliente.idCliente = Pedido.idCliente

JOIN EstadoPedido ON Pedido.idEstadoPedido = EstadoPedido.idEstadoPedido;



***Consulta de detalles de un pedido:***

SELECT Pedido.idPedido, Prenda.tipoPrenda, Servicio.nombreServicio, DetallePedido.cantidad, DetallePedido.subtotal

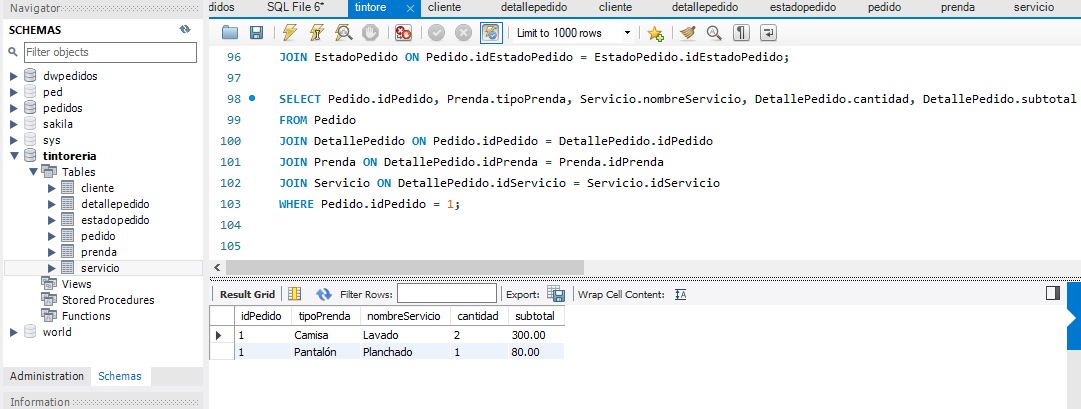
FROM Pedido

JOIN DetallePedido ON Pedido.idPedido = DetallePedido.idPedido

JOIN Prenda ON DetallePedido.idPrenda = Prenda.idPrenda

JOIN Servicio ON DetallePedido.idServicio = Servicio.idServicio

WHERE Pedido.idPedido = 1;



***Consulta de todos los servicios solicitados por un cliente específico:***

SELECT Cliente.nombre, Servicio.nombreServicio, SUM(DetallePedido.cantidad) AS total\_servicios

FROM Cliente

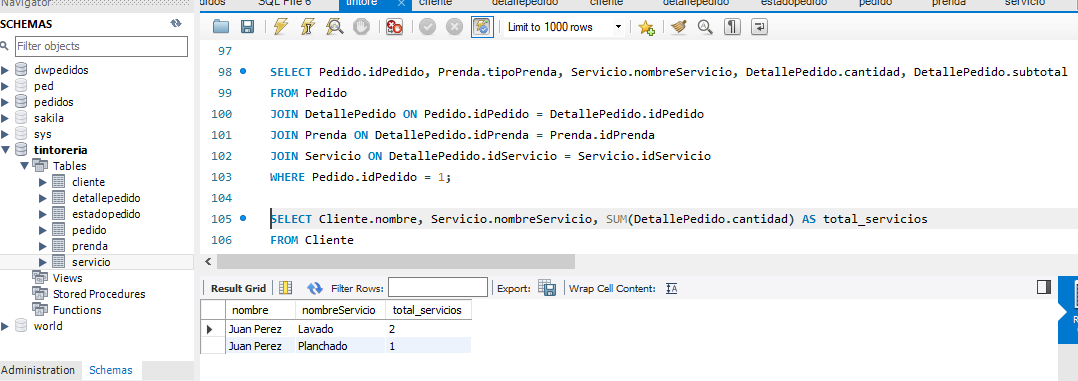
JOIN Pedido ON Cliente.idCliente = Pedido.idCliente

JOIN DetallePedido ON Pedido.idPedido = DetallePedido.idPedido

JOIN Servicio ON DetallePedido.idServicio = Servicio.idServicio

WHERE Cliente.idCliente = 1

GROUP BY Servicio.nombreServicio;



***Consulta para calcular el total a pagar por un pedido:***

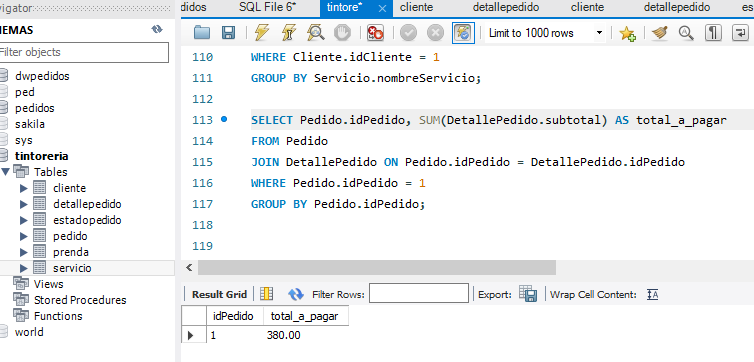
SELECT Pedido.idPedido, SUM(DetallePedido.subtotal) AS total\_a\_pagar

FROM Pedido

JOIN DetallePedido ON Pedido.idPedido = DetallePedido.idPedido

WHERE Pedido.idPedido = 1

GROUP BY Pedido.idPedido;



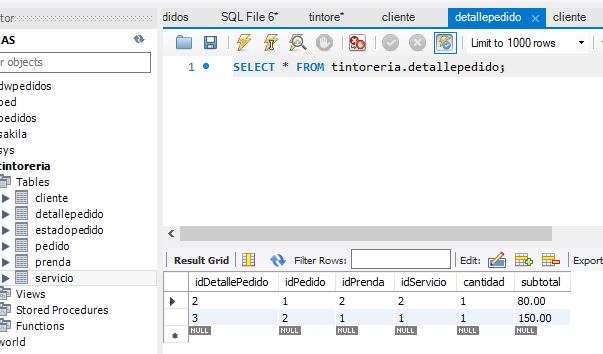
**Borrado de Registros**

Para borrar registros de la base de datos, se deben eliminar las entradas en las tablas relacionadas antes de eliminar en las tablas principales para mantener la integridad referencial.

**Eliminar un detalle de pedido:**

DELETE FROM DetallePedido

WHERE idDetallePedido = 1;



**Eliminar un pedido:**

DELETE FROM Pedido

WHERE idPedido = 1;--aca me salió un error, que supongo que es por las claves foráneas y que se borro anteriormente iddetallepedido=1.

**Eliminar un cliente**:

DELETE FROM Cliente

WHERE idCliente = 1; --mismo error

**Definiciones de Comunicación**

Para la comunicación del sistema con la base de datos, se utilizará JDBC (Java Database Connectivity) para interactuar con MySQL. Esto permitirá ejecutar operaciones de consulta, inserción, actualización y borrado de manera eficiente desde las aplicaciones Java.

**Flujo de Comunicación:**

Conexión a la Base de Datos: El sistema Java establece una conexión con el servidor MySQL utilizando la URL del servidor, nombre de usuario y contraseña.

Operaciones CRUD: Una vez establecida la conexión, el sistema puede realizar operaciones de creación, lectura, actualización y eliminación en la base de datos.

Cierre de Conexión: Tras completar las operaciones, el sistema cierra la conexión con la base de datos para liberar los recursos.

**En el siguiente repositorio:** <https://github.com/marcosBanducer/tintoreria_Pardini>

Esta cargado el sistema, a medida que el proyecto avance, se irán subiendo más código y documentación del mismo.

**POO (Programación Orientada a Objetos)**

**Consigna**

Esta tercera actividad te permitirá comenzar con la aplicación de los conceptos de programación orientada a objetos (POO). En esta instancia, es fundamental que tu propuesta considere las características principales del lenguaje de programación Java, incluyendo su sintaxis, tipos de datos, estructuras de control y manejo de excepciones.

Dentro de los pilares fundamentales del paradigma que se está empleando, debes realizar un proyecto que incluya correctamente el encapsulamiento, herencia, polimorfismo y abstracción. Estos pilares proporcionan una base sólida para el desarrollo de software modular, flexible y fácilmente mantenible.

Contempla la utilización de estructuras condicionales y de repetición, para controlar el flujo de ejecución de un programa. Adicionalmente, el desarrollo debe incluir un menú con opciones de selección, la declaración y creación de objetos en Java, así como el uso de constructores para inicializar objetos.

**Explicación del desarrollo en Java**

El desarrollo en Java para el sistema de gestión de pedidos que estamos implementando se basa en la Programación Orientada a Objetos (POO) y utiliza tecnologías como Swing para la interfaz gráfica y JDBC para conectar con la base de datos MySQL. A continuación, explico los aspectos clave de este desarrollo en Java, dividido en estructura, funcionalidades, principios de POO, y manejo de datos.

**Estructura del Proyecto**

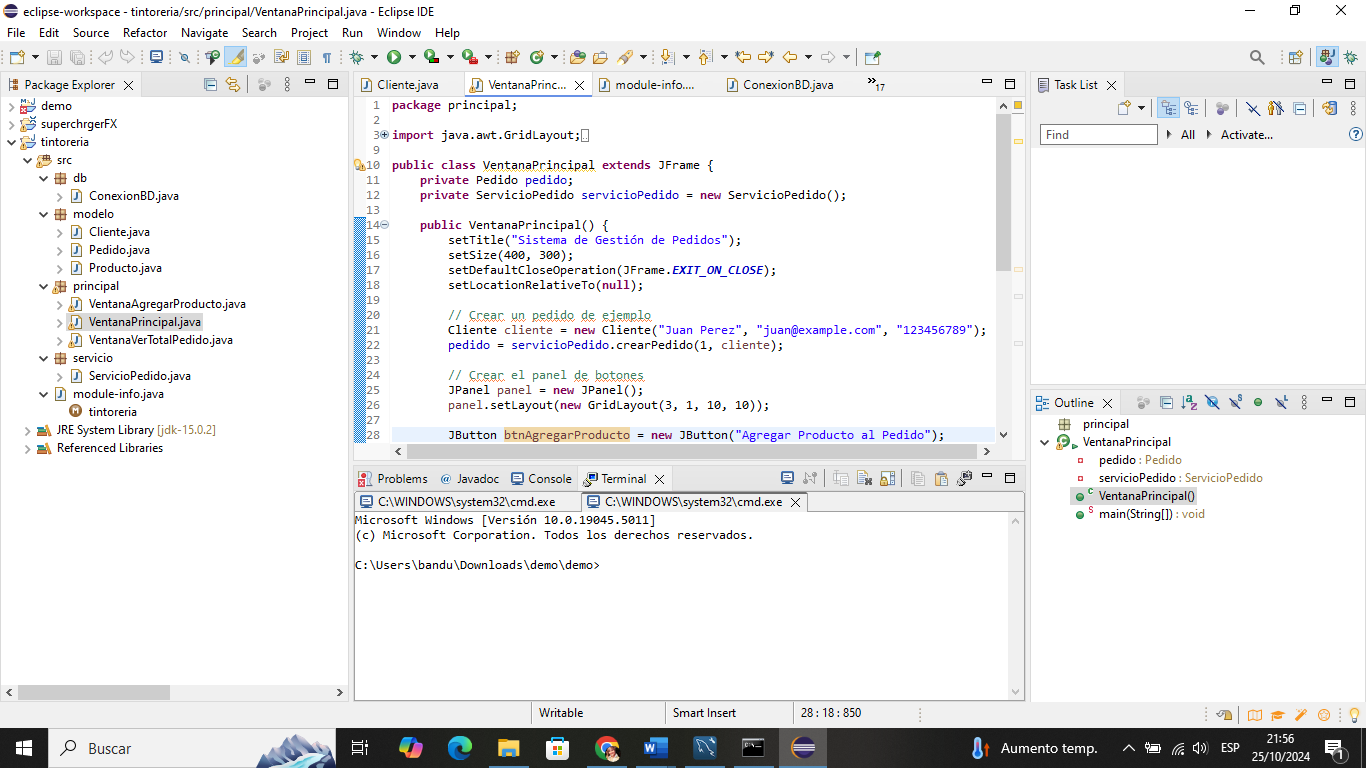
El proyecto se organiza en diferentes paquetes para facilitar la organización del código:

***modelo:*** Define las entidades principales (Cliente, Producto, Pedido) que representan los datos que maneja el sistema.

***servicio:*** Contiene la lógica de negocio del sistema (ServicioPedido), que realiza las operaciones principales sobre los datos.

***db:*** Administra la conexión con la base de datos a través de ConexionBD.

***principal:*** Contiene la interfaz gráfica principal del sistema (VentanaPrincipal), junto con las clases para interactuar con el usuario (VentanaAgregarProducto, VentanaVerTotalPedido).



Así es como se vería en el IDE Eclipse, el proyecto tintorería con su separación de paquetes: db, modelo, principal y servicio.

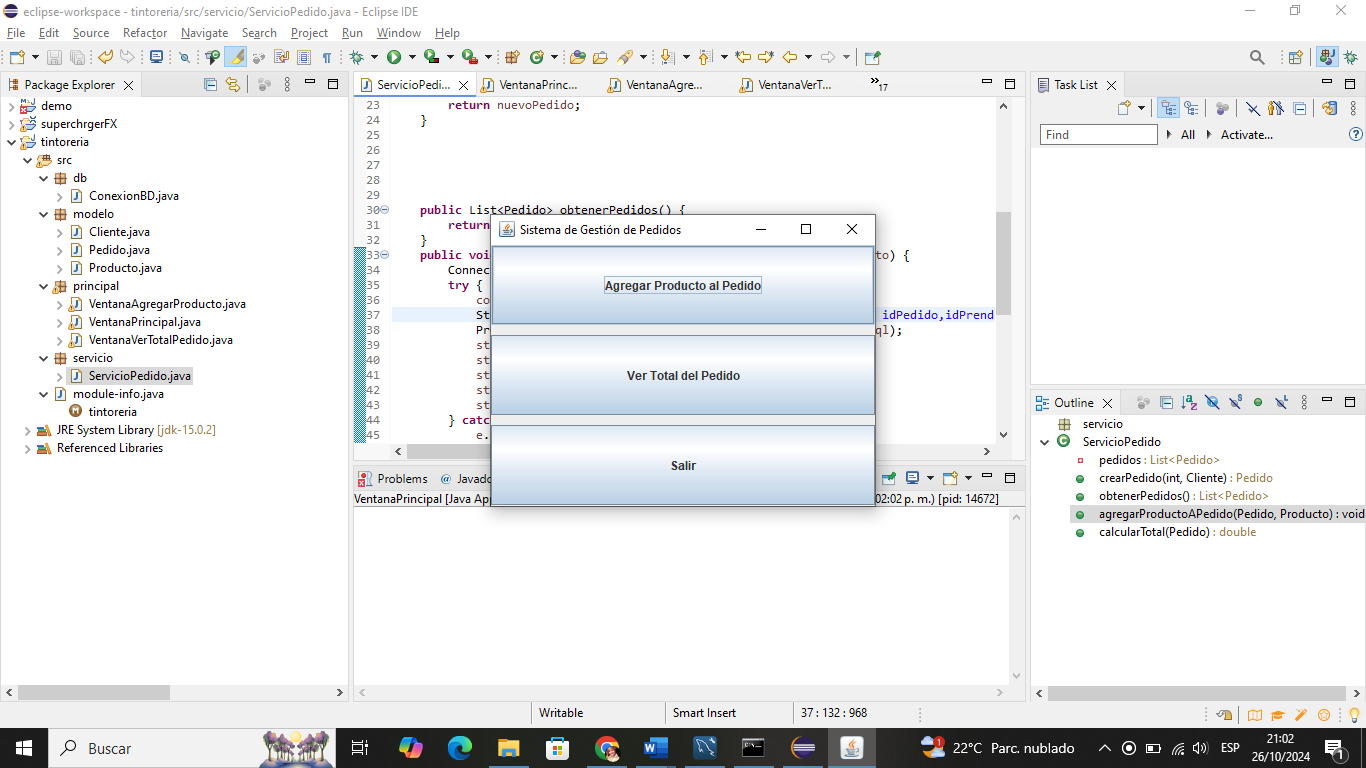
**Funcionalidades Principales**

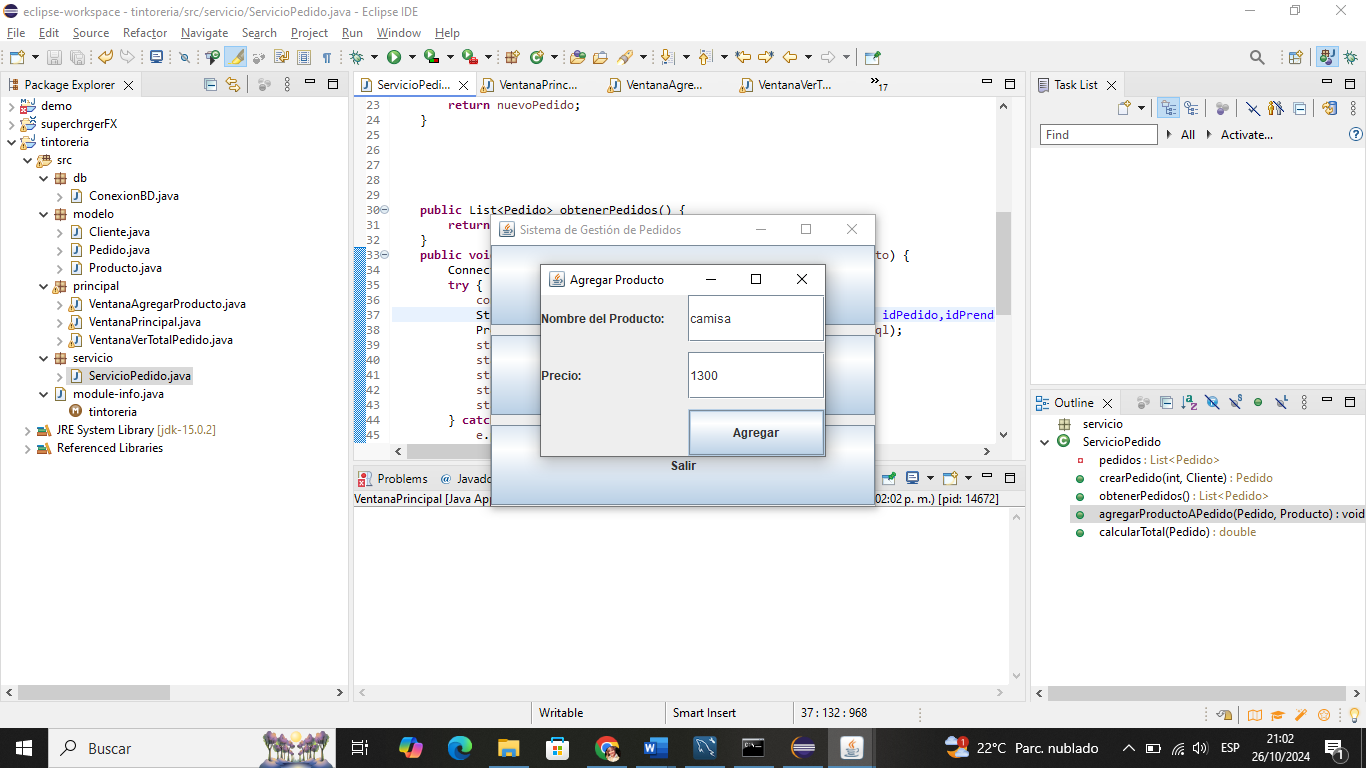
El sistema permite a los usuarios realizar las siguientes operaciones:

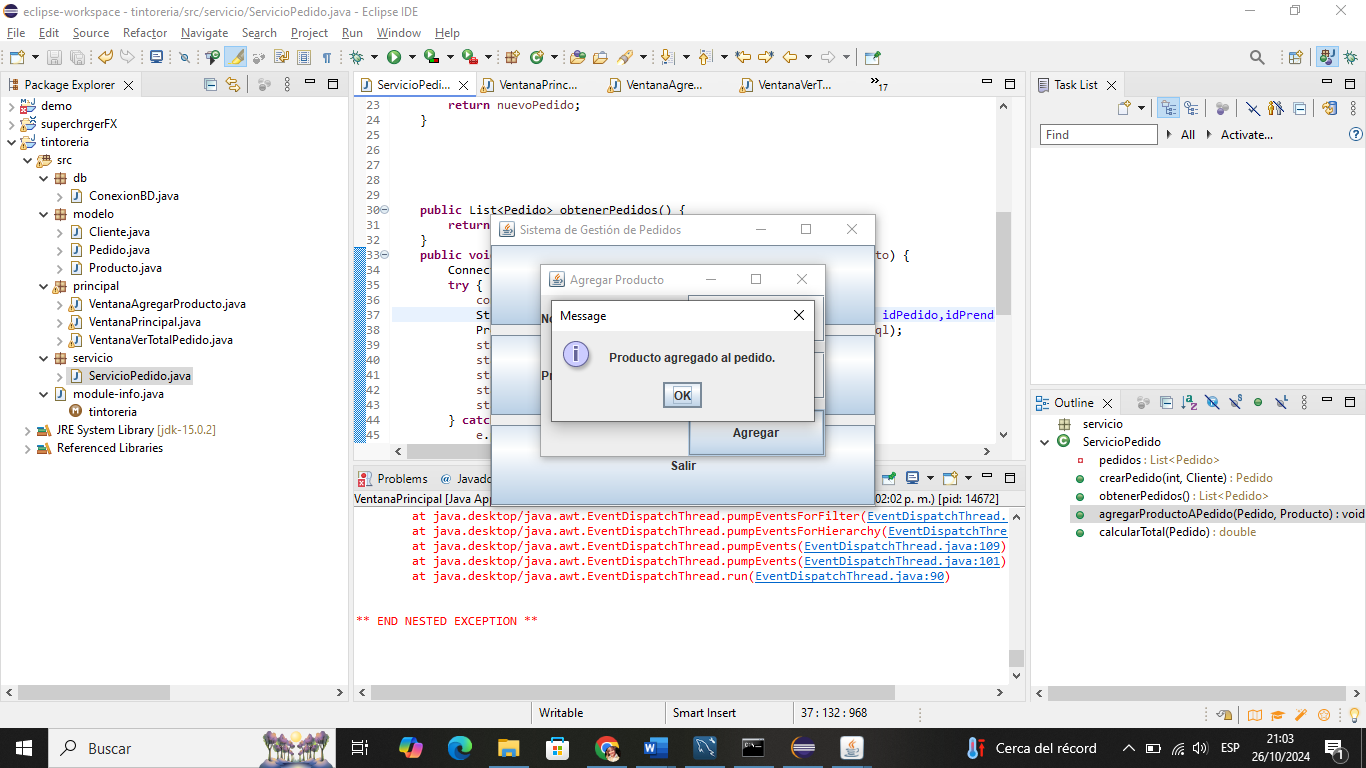
***Agregar Productos a un Pedido:*** Los productos se seleccionan y se agregan al pedido activo, almacenándolos en la base de datos.

***Ver Total del Pedido:*** Calcula el total del pedido consultando en la base de datos el precio acumulado de los productos seleccionados.

***Interfaz Gráfica de Usuario (GUI):*** El sistema ofrece un menú de selección en una interfaz de usuario construida con Swing.







**Principios de Programación Orientada a Objetos (POO)**

El desarrollo en Java utiliza los pilares de POO para mantener la modularidad, flexibilidad y mantenibilidad del código.

**a) Encapsulamiento**

Los atributos de las clases (Cliente, Producto, Pedido) están encapsulados y sólo son accesibles a través de métodos get y set.

Por ejemplo, la clase Producto tiene sus atributos id, nombre y precio privados y accedidos mediante métodos públicos para controlar cómo se manipulan esos datos.

**b) Herencia**

Aunque en este caso específico no se ha aplicado una jerarquía compleja de herencia, la estructura modular en paquetes permite que otras clases puedan heredar características comunes en caso de que sea necesario extender el sistema en el futuro.

**c) Polimorfismo**

En la clase ServicioPedido, el polimorfismo se aplica a través de métodos que permiten manejar diferentes comportamientos según el tipo de datos (por ejemplo, en agregarProductoAPedido que admite distintos productos y pedidos).

**d) Abstracción**

Las clases y métodos abstraen la complejidad de las operaciones, como el manejo de bases de datos y la lógica de negocio. El usuario final interactúa solo con la interfaz gráfica y no con los detalles internos de cómo se almacenan o calculan los datos.

**Interfaz Gráfica con Swing**

La interfaz gráfica se implementa usando Swing, lo cual permite que el usuario interactúe de forma intuitiva con el sistema a través de botones, etiquetas y cuadros de texto.

**Código implementado en la ventana principal**

**package** principal;

**import** java.awt.GridLayout;

**import** javax.swing.\*;

**import** modelo.Cliente;

**import** modelo.Pedido;

**import** servicio.ServicioPedido;

**public** **class** VentanaPrincipal **extends** JFrame {

**private** Pedido pedido;

**private** ServicioPedido servicioPedido = **new** ServicioPedido();

**public** VentanaPrincipal() {

setTitle("Sistema de Gestión de Pedidos");

setSize(400, 300);

setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

setLocationRelativeTo(**null**);

// Crear un pedido de ejemplo

Cliente cliente = **new** Cliente("Juan Perez", "juan@example.com", "123456789");

pedido = servicioPedido.crearPedido(1, cliente);

// Crear el panel de botones

JPanel panel = **new** JPanel();

panel.setLayout(**new** GridLayout(3, 1, 10, 10));

JButton btnAgregarProducto = **new** JButton("Agregar Producto al Pedido");

JButton btnVerTotalPedido = **new** JButton("Ver Total del Pedido");

JButton btnSalir = **new** JButton("Salir");

panel.add(btnAgregarProducto);

panel.add(btnVerTotalPedido);

panel.add(btnSalir);

add(panel);

// Acciones de los botones

btnAgregarProducto.addActionListener(e -> **new** VentanaAgregarProducto(pedido));

btnVerTotalPedido.addActionListener(e -> **new** VentanaVerTotalPedido(pedido));

btnSalir.addActionListener(e -> dispose());

}

**public** **static** **void** main(String[] args) {

SwingUtilities.*invokeLater*(() -> {

VentanaPrincipal ventana = **new** VentanaPrincipal();

ventana.setVisible(**true**);

});

}}

**Componentes Usados**

**JFrame:** Representa las ventanas principales (VentanaPrincipal, VentanaAgregarProducto, etc.).

**JButton:** Se utiliza para opciones de selección en el menú, como "Agregar Producto al Pedido" y "Ver Total del Pedido".

**JTextField**: Permite al usuario ingresar datos como el nombre y el precio del producto.

**JLabel:** Muestra el texto estático, como etiquetas para el nombre y el precio del producto.

La clase VentanaPrincipal actúa como la ventana de inicio, permitiendo la navegación entre las opciones de la aplicación.

**Conexión a la Base de Datos con JDBC**

Para conectar esta interfaz de usuario con la base de datos MySQL, utilice el conector JDBC de Java. JDBC permite realizar operaciones CRUD (Crear, Leer, Actualizar y Eliminar) en la base de datos desde tu aplicación Java. Muestro cómo configurar la conexión y modificar la lógica de la aplicación para que los datos se almacenen y recuperen de MySQL.

**Configurar el Conector MySQL en Eclipse**

Descargar el Conector JDBC:

Descarga el conector JDBC para MySQL desde el sitio oficial de MySQL.

Agrega el archivo mysql-connector-java.jar a tu proyecto en Eclipse.

**Agregar la Librería JDBC al Proyecto:**

clic derecho en tu proyecto en Eclipse, selecciona Build Path > Add External Archives, y selecciona el archivo .jar del conector JDBC.

**Clase ConexionBD:** Gestiona la conexión y desconexión con la base de datos. Usa un método conectar () para establecer la conexión y un método cerrarConexion() para cerrarla, asegurando un manejo eficiente de recursos.

**Clase ServicioPedido:** Realiza las operaciones de negocio sobre los datos en la base de datos, como agregar productos al pedido y calcular el total del pedido desde la base de datos.

**package** db;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.SQLException;

**public** **class** ConexionBD {

**private** **static** **final** String ***URL*** = "jdbc:mysql://localhost:3306/tintoreria?serverTimezone=America/Argentina/Buenos\_Aires";

**private** **static** **final** String ***USER*** = "root";

**private** **static** **final** String ***PASSWORD*** = "hyru567";

**public** **static** Connection conectar() **throws** SQLException {

**return** DriverManager.*getConnection*(***URL***, ***USER***, ***PASSWORD***);

}

**public** **static** **void** cerrarConexion(Connection conexion) {

**if** (conexion != **null**) {

**try** {

conexion.close();

} **catch** (SQLException e) {

e.printStackTrace();

}

}

}

}

**Manejo de Excepciones**

El manejo de excepciones es esencial en este desarrollo para gestionar posibles errores de usuario o de conexión con la base de datos. Se incluyen bloques try-catch en el manejo de la base de datos y en la entrada de datos para manejar errores como:

***Errores de conexión con la base de datos:*** Cada conexión con MySQL está rodeada por un bloque try-catch para capturar SQLException.

***Validación de entrada de usuario:*** Por ejemplo, al ingresar el precio de un producto, se usa un try-catch para capturar errores si el usuario introduce un valor no numérico.

**Ejemplo de Ejecución del Sistema**

Asegúrate de que estás usando una versión de Java que incluya las bibliotecas de Swing. Swing está incluido en el JDK (Java Development Kit) desde las versiones de Java SE, así que asegúrate de que tienes instalado un JDK y no solo el JRE (Java Runtime Environment).

**Instala JDK:** Si no tienes el JDK instalado, puedes descargarlo desde el sitio oficial de Oracle o Adoptium.

**Configuración el JDK en Eclipse:**

Ve a Window > Preferences.

Expande Java y selecciona Installed JREs.

Asegúrate de que tienes un JDK configurado. Si no, haz clic en Add y selecciona Standard VM, luego navega hasta la ubicación de tu JDK.

**Verificación la configuración del proyecto**

clic derecho en el proyecto en el Package Explorer y selecciona Properties.

Ve a Java Build Path y selecciona la pestaña Libraries.

Asegúrate de que esté listado un JDK (no un JRE). Si solo ves un JRE, quítalo y agrega el JDK.

Para agregar el JDK, haz clic en Add Library > JRE System Library y selecciona el JDK que configuraste anteriormente.

***Iniciar la aplicación ejecutando VentanaPrincipal en Eclipse.***

En la ventana principal, se despliegan opciones como "Agregar Producto al Pedido" y "Ver Total del Pedido".

Al seleccionar "Agregar Producto al Pedido", se abre una ventana donde el usuario puede ingresar el nombre y el precio del producto. Al hacer clic en "Agregar", el producto se inserta en la base de datos.

Al seleccionar "Ver Total del Pedido", se abre una ventana que muestra el total del pedido calculado al consultar los productos asociados al pedido en la base de datos.

***Importante tener la clase mudole-info.java de la siguiente manera:***

**module** tintoreria {

**requires** java.desktop; // Esto es necesario para usar Swing

**requires** java.sql;

**exports** principal; // Exporta tu paquete si es necesario

}

**Extensibilidad del Proyecto**

Gracias a la estructura modular y a la organización en paquetes y clases de este proyecto, resulta fácil extender el sistema para añadir más funcionalidades, como gestionar clientes, realizar búsquedas de pedidos o productos, o integrar nuevas formas de pago.

**En el siguiente repositorio:** <https://github.com/marcosBanducer/tintoreria_Pardini>

Esta cargado el sistema, a medida que el proyecto avance, se irán subiendo más código y documentación del mismo.

**Conclusión**

El sistema desarrollado en Java es una aplicación de gestión de pedidos que integra una interfaz gráfica de usuario con una base de datos MySQL. La estructura de POO facilita la mantenibilidad del código, mientras que el uso de Swing proporciona una interfaz gráfica interactiva. JDBC permite la persistencia de datos, garantizando que las operaciones realizadas en el sistema se almacenen en la base de datos.