**E-BUSINESS-FERRETERÍA**

**DOCUMENTACIÓN DIA 2**

Fundamentos de Ingeniería de Software



Eduardo Pineda Almeda (193109)

Carlos Murillo Flores (202179)

Gabriel Alberto Vilchis Ríos (215460)

Emmanuel Rodríguez Franco (215822)

**12/11/24**

**INDICE**

Arquitectura de software-------------------------------------------------------------------------------1

Formalización de requisitos----------------------------------------------------------------------------3

Diagrama de clases--------------------------------------------------------------------------------------4

Clasificación de productos de catálogo---------------------------------------------------------------6

Diagrama de entidad relación--------------------------------------------------------------------------8

Diagrama lógico relacional----------------------------------------------------------------------------9

Diseño de tablas----------------------------------------------------------------------------------------11

**ARQUITECTURA DE SOFTWARE**

Con base al diseño Modelo Vista Controlador (MVC) definido anteriormente, se realizó un diagrama de arquitectura de software donde se indica el flujo de datos y funcionamiento.

Diagrama

Descripción generada automáticamente

El componente modelo se encarga de gestionar los datos y la lógica de negocio de la aplicación, el modelo está compuesto por las siguientes tecnologías:

* Como backend se utiliza Django como framework para la creación de la API RESTful, el cual, va a interactuar con la base de dichos.
* La base de datos se usará MySQL como herramienta para almacenar y gestionar la información del modelo.

En el modelo, realiza las siguientes acciones:

* El modelo contiene las definiciones de las tablas y relación de la base de datos, mediante el ORM que proporciona Django.
* El modelo realiza operaciones CRUD (Create, Read, Update, Delete), que consisten en la creación, la lectura, la actualización y la eliminación de datos.
* Y por último, el modelo proporciona datos procesados al controlador para que posteriormente sean enviados a la vista.

La vista es el componente responsable de mostrar la interfaz de usuario (UI) y mostrar la información de una manera que sea entendible al usuario. La vista esta compuesta por las siguientes tecnologías:

**1**

* Como frontend, se seleccionó React como framework para gestionar la UI, en conjunto a HTML, CSS y JavaScript para el diseño y el comportamiento de la aplicación.

En la vista, realiza las siguientes acciones:

* La vista recibe los datos que fueron procesados desde el backend, que fue gracias al controlador.
* La vista muestra los datos en la interfaz gráfica.
* Por último, captura las acciones que realiza el usuario, como podrían ser los clics o interacciones con formularios, para enviarlos al controlador.

El componente del controlador actúa como un intermediario entre la vista y el modelo, esta constituido en 2 partes:

* El controlador del backend, este, esta representado por los endpoints de la API RESTful de Django, que ayuda a recibir las solicitudes que provienen de la vista, interactúan con el modelo para obtener o actualizar los datos, para posteriormente retornar los resultados en formato JSON a la vista.
* El controlador del frontend, eta representado por los componentes que proporciona React, los cuales, ayudan a manejar los eventos del usuario (las interacciones del usuario), realizar solicitudes con el protocolo HTTP (Hypertext Transfer Protocol) al backend, y actualizar el estado de la aplicaciones para mostrar los datos en la vista.

**2**

**1**

**FORMALIZACIÓN DE REQUISITOS**

A los requisitos obtenidos y formalizados anteriormente, se le clasificaron como funcionales, y de igual forma se incluyeron los requisitos no funcionales que son de suma importancia para definir ciertos comportamientos del sistema:

* El sistema debe ser capaz de manejar diversos usuarios concurrentes realizando operaciones como registro, navegación en el catálogo, y pagos sin comprometer la velocidad de respuesta. Dicha velocidad de respuesta no debe exceder de los 2 segundos para cualquier acción que realicen los usuarios.
* El sistema debe garantizar la seguridad de los datos personales y de pago mediante el cifrado de la información sensible, como contraseñas y datos bancarios, utilizando un protocolo seguro como HTTPS. Este cifrado debe confirmar que no existan vulnerabilidades críticas en el manejo de datos sensibles.
* El sistema debe ser compatible con navegadores web más populares, como lo son Google Chrome, Brave, Microsoft Edge, Safari, Opera GX y Mozilla Firebox.
* El sistema debe ser escalable, permitiendo agregar más productos al catálogo, usuarios registrados y capacidad de procesamiento de transacciones sin necesidad de rediseño del sistema.
* El sistema debe estar diseñado de forma modular, permitiendo la adición de nuevas funcionalidades o la corrección de errores con un impacto mínimo en las funcionalidades existentes.
* El sistema debe ser intuitivo y fácil de usar, permitiendo que el usuario pueda completar tareas como el registro, navegación en el catálogo y la compra en menos de 5 minutos sin requerir soporte técnico.
* El sistema debe estar disponible para los usuarios al menos en un 99.9% del tiempo, exceptuando tiempos que involucren mantenimientos planificados previamente.

**3**

**DIAGRAMA DE CLASES**

Para el diseño de software, se elaboró un diagrama de clases Unified Modeling Language (UML), en el cual, proporcionará una explicación detallada de cada clase establecida en el diagrama, con sus respectivos, métodos, así como sus relaciones. Este sería el diagrama de clases:

Diagrama

Descripción generada automáticamente

Desglosando el diagrama:

* Existen 5 clases, las cuales serían cliente, producto, inventario, pago y pedido.

Cada clase esta compuesta de la siguiente manera:

* La clase ‘Cliente’ cuenta con todos sus atributos de tipo de protección (#), incluyendo el ‘id\_cliente’ de tipo entero, ‘nombre’, ‘apellido’, ‘num\_telefonico’, ‘direccion’, ‘email’ y ‘contraseña’ de tipo cadena de caracteres. Sus métodos serian 3 y todos de tipo de protección público (+), el método ‘creacionCuenta()’ retorna una cadena caracteres, y los métodos ‘realizarPedido()’ y ‘registrarCuenta()’ que son de tipo vacío.
* La clase ‘Producto’ se tienen atributos de visibilidad publica, los cuales son: ‘id\_producto’, y ‘stock’ de tipo entero, seguido de ‘nombre\_producto’, y ‘descripcion’ de tipo cadena de caracteres y ‘precio’ de tipo flotante. Los métodos que componen la clase son todos de visibilidad publica, ‘detallesProducto()’ retorna una cadena de caracteres, mientras que los métodos ‘restarStock()’, ‘agregarStock()’ y ‘actualizarPrecio()’ son de tipo vacio.

**4**

* La clase ‘Inventario’ este compuesto por un solo atributo de visibilidad privada (-), el cual es ‘productos’ que es un vector que contiene el objeto Producto. Y tiene 3 métodos, 2 de ellos de tipo vacío, los cuales serían: ‘agregarProducto()’ y ‘eliminarProducto()’, mientras que el método de ‘consultarStock()’ retorna un entero.
* La clase ‘Pago’ esta compuesto por atributos de visibilidad protegida, los cuales son: ‘id\_pago’ de tipo entero, ‘cliente’, ‘productos\_pedido’, ‘estado\_pedido’ y ‘metodo\_pago’ de tipo cadena de caracteres, teniendo de métodos ‘estadoPedido()’ y ‘totalPedido()’ de tipo vacio, ‘detallesPedido()’ que retorna una cadena de caracteres, y ‘fechaPedido()’ que retorna una variable de tipo fecha.
* Y, por último, la clase ‘Pedido’ esta compuesto por los atributos: ‘id\_pedido’ de tipo entero, ‘fecha\_pedido’ de tipo date, ‘estado’ de tipo cadena de caracteres, ‘total’ de tipo flotante y ‘productos’ un vector que contiene objetos de la clase Producto. Sus métodos de visibilidad publica y todos de tipo vacío: ‘agregarProductos()’, ‘actualizarPedido()’ y ‘actualizarEstado()’.

Una vez teniendo las clases definidas, las relaciones entre clases se definieron de la siguiente manera:

* La clase ‘Cliente’ está asociada con la clase ‘Pedido’.
* La clase ‘Pedido’ está asociada con la clase ‘Pago’ y la clase ‘Producto’.
* La clase ‘Inventario’ está asociada con la clase ‘Producto’.

Estas relaciones son bidireccionales, por lo que sus valores de multiplicidad serían los siguientes:

* Un cliente puede realizar varios pedidos, para cada pedido está asociada a un único cliente. Siendo una relación de 1 a muchos (1..).
* Un pedido puede contener múltiples productos, y un producto puede estar incluido en varios pedidos. Siendo una relación de muchos a muchos (n..m).
* Un inventario este compuesto por varios productos, pero cada producto pertenece a un único inventario. Siendo una relación de 1 a muchos (1..).
* Y, por último, cada pedido tiene asociada a un único pago, y cada pago corresponde a un único pedido. Siendo una relación de 1 a 1 (1..1).

**5**

**CLASIFICACIÓN DE PRODUCTOS DE CATÁLOGO**

Se les dio una clasificación a los productos pertenecientes al listado generado anteriormente, de modo que sea mucho más sencillo gestionar sus datos y disponibilidad, y poder implementarlos de forma más libre en una base de datos.

**Herramientas Manuales:**

* Destornillador plano, mediano – $50 MXN
* Destornillador Phillips, grande – $60 MXN
* Llave inglesa de 6" – $120 MXN
* Llave inglesa de 8" – $150 MXN
* Llave inglesa de 12" – $200 MXN
* Alicate universal, mediano – $120 MXN
* Alicate de corte, pequeño – $80 MXN
* Alicate de presión, grande – $250 MXN
* Martillo de carpintero, 400g – $90 MXN
* Martillo de goma, 300g – $70 MXN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Medición:**

* Cinta métrica de 5m – $40 MXN
* Cinta métrica de 10m – $70 MXN
* Cinta métrica de 25m – $200 MXN
* Cinta métrica láser, 50m – $800 MXN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Corte:**

* Serrucho para madera, 18" – $150 MXN
* Serrucho para metal, 12" – $120 MXN
* Cuchillo multiusos, retráctil – $50 MXN
* Cuchillo de hoja fija, 25mm – $80 MXN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Jardinería:**

* Tijeras de podar, pequeñas – $150 MXN
* Tijeras de podar, grandes – $250 MXN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**6**

**Accesorios de Taladro:**

* Brocas para madera, 3mm – $25 MXN
* Brocas para metal, 5mm – $30 MXN
* Brocas para concreto, 10mm – $40 MXN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Fijación:**

* Clavos de acero, 1" (paquete de 100) – $30 MXN
* Tornillos para madera, 2" (paquete de 200) – $80 MXN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Adhesivos:**

* Pegamento universal, 100g – $40 MXN
* Super glue, 20g – $25 MXN
* Silicona en tubo, 300ml – $100 MXN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Material Eléctrico:**

* Bombilla LED 10W, blanca fría – $50 MXN
* Extensión eléctrica, 5m, 3 tomas – $150 MXN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Accesorios Varios:**

* Cinta aislante negra, 19mm x 10m – $30 MXN
* Pintura en spray blanca, 400ml – $90 MXN
* Lija para madera, grano 80 – $40 MXN
* Lija para metal, grano 120 – $40 MXN

**7**

**DIAGRAMA DE ENTIDAD RELACIÓN**

Se definió el diagrama de entidad relación que tendrá la base de datos, identificando de qué forma interactuarán las tablas con base a sus contenidos.

Diagrama

Descripción generada automáticamente

**Componentes:**

* La entidad cliente, tiene como atributo el ID del cliente como llave primaria, nombre, apellido, correo electronico, direccion, contraseña y el numero telefonico como atributo multivaluado, ya que el cliente puede tener varios numeros telefonicos.
* La entidad pedido tiene como llave primaria el ID del pedido, seguido de los atributos monto total, fecha del pedido y el estado del pedido.
* La entidad pago tiene como llave priimaria el atributo el ID pago, seguido con el estado del pago, el metodo de pago y el monto total como atributos.
* La entidad carrito de compra tiene como llave primaria y atributo el ID del carrito de compra, el monto total y la cantidad productos que contiene el carrito.
* Y por ultimo, la entidad producto tiene el ID del producto como llave primaria y atributo, seguido del nombre del producto, precio, stock y la descripcion del producto.

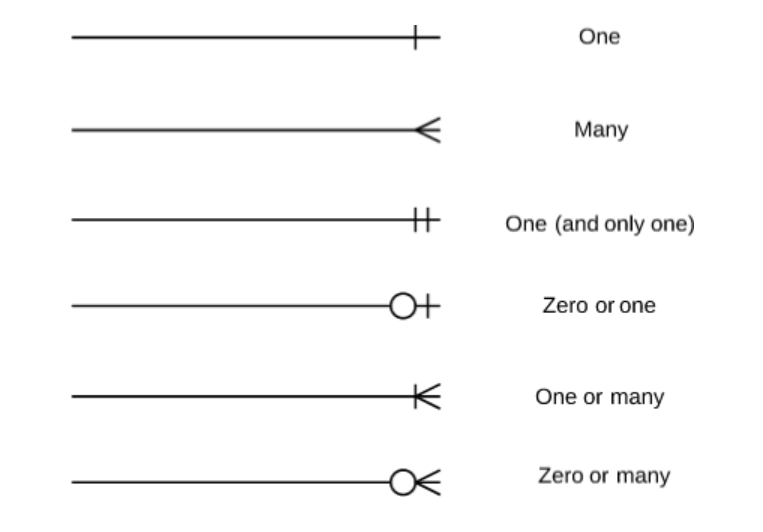
**Relaciones:**

* La entidad cliente tiene una relacion de 1 a 1 con el carrito de compras, ya que un cliente tiene un carrito de compras, y un carrito de compras tiene un cliente.
* Tambien, la entidad cliente tiene otra relacion con la entidad de pedido, su relacion es de 1 a muchos, debido a que 1 o varios clientes pueden realizar un solo pedido, y un pedido es realizado por 1 o varios clientes.
* La entidad pedido tiene una relacion de 1 a 1 con la entidad carrito, porque un pedido puede cofnfirmar el carrito de compra y el carrito de compra puede confirmar el pedido.
* Seguido, el pedido tiene una relacion de igual forma de 1 a 1 con la entidad pago, porque un pedido confirma un pago, y un pago confirma un pedido.
* Y por ultimo, la entidad producto tiene una relacion de muchos a muchos con la entidad carrito, debido a que 1 o varios productos tienen 1 o varios carritos de compras, y a su vez, 1 o varios carritos de compras tienen 1 o mas productos.

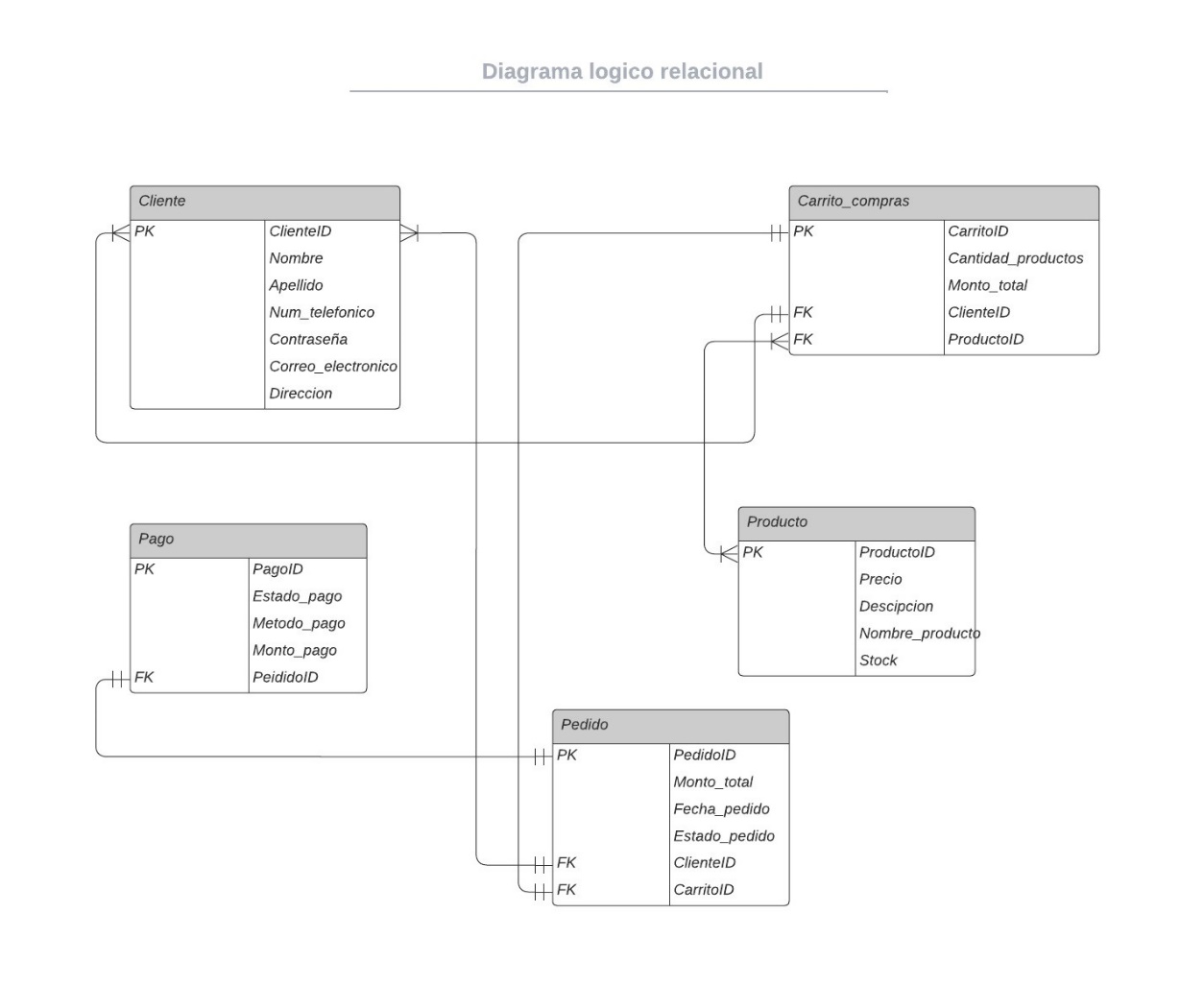
**8**

**DIAGRAMA LÓGICO RELACIONAL**

Una vez teniendo el modelo entidad relación para nuestra base de datos, se realizó el modelo lógico relacional que el cual, explica cuál es la conexión que tienen las tablas usando llaves primarias y llaves foráneas, o también dicho de forma casual, 'Primary Key' y 'Foreign Key'. Para explicar la conexión de estas tablas, se debe considerar que para este tipo de diagramas se usa una notación especifica, las relaciones u conexiones que hay son las siguientes:



Las relaciones son las mismas, pero en este caso, se utilizarán las cardinalidades de ambos lados correspondientes de las entidades. Para determinar una relación, se tienen en cuenta estos dos lados. Con estos principios, se creó el diagrama lógico relacional de la siguiente manera:



**9**

**Diagrama desglosado:**

* La tabla cliente cuenta con una llave primaria denominada como “ClienteID”.
* La tabla carrito\_compras cuenta con una llave primaria denominada como “CarritoID”.
* La tabla pago cuenta con una llave primaria denominada como “PagoID”.
* La tabla producto cuenta con una llave primaria denominada como “ProductoID”.
* La tabla pedido cuenta con una llave primaria denominada como “PedidoID”.
* La tabla de carrito\_compra posee 2 llaves foráneas, las cuales serian “ClienteID” y “ProductoID”, están conectadas a sus respectivas tablas de donde provienen, Indicando que los productos están en el carrito y que el cliente le pertenece o tiene el carrito.
* La tabla pago tiene una llave foránea “PedidoID” conectada con la tabla de producto, indicando que el pedido esta vinculado con el pago.
* La tabla pedido tiene 2 llaves foráneas, las cuales son “ClienteID” y “CarritoID”, de igual forma conectadas a sus respectivas tablas. Esto quiere decir que el cliente realiza los pedidos y que el carrito pertenece a los pedidos.

**10**

**DISEÑO DE TABLAS**

Una vez definidos los diagrama, se diseñan los tipos de datos que tendría cada tabla para al momento que se implemente la base de datos, a continuación, estos serían los diseños de las tablas:

**Clientes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Campo** | **Tipo de dato** | **Tamaño** | **default** |
| ClienteID | int |  | NOT NULL |
| Nombre | varchar | 50 |  |
| Apellido | varchar | 50 |  |
| Num\_telefonico | varchar | 14 |  |
| Contraseña | varchar | 15 |  |
| Correo\_electronico | varchar | 100 |  |
| Direccion | varchar | 100 |  |

**Pago:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Campo** | **Tipo de dato** | **Tamaño** | **default** |
| PagoID | int |  | NOT NULL |
| Estado\_pago | varchar | 15 |  |
| Método\_pago | varchar | 10 |  |
| Monto\_pago | decimal | 10,2 |  |

**Producto:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Campo** | **Tipo de dato** | **Tamaño** | **Default** |
| ProductoID | int |  | NOT NULL |
| Precio | int |  |  |
| Descripcion | varchar | 200 |  |
| Nombre\_producto | varchar | 100 |  |
| Stock | Int |  |  |

**Pedido:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Campo** | **Tipo de dato** | **Tamaño** | **Default** |
| PedidoID | int |  | NOT NULL |
| Monto\_total | decimal | 10,2 |  |
| Fecha\_pedido | date |  |  |
| Estado\_pedido | varchar | 15 |  |

**Carrito de compras:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Campo** | **Tipo de dato** | **Tamaño** | **Default** |
| CarritoID | int |  | NOT NULL |
| Cantidad\_productos | int |  |  |
| Monto\_total | decimal | 10,2 |  |

**11**