

**Estructura de contenidos pág.**

2.1.1 alcance……………………………………………………………………………………3

2.1.2 Descripción de procesos……………………………………………………………4

2.1.3Diagramas UML ….………..………………………………………………………….8

2.1.4Administración de usuarios ……..……………………………………………….11

2.1.5Modelo relacional de base de datos …………..………………………………11

2.1.6 Descripción de plataforma……..…………………………………………………16

2.1.7 Documentación del Código fuente……………………………………………..16

**MANUAL TECNICO**

2.1.1 Alcance

Objetivo general del sistema:

* + El objetivo general del sistema es el correcto funcionamiento de la app web tanto como para administrador y usuario funcionen correctamente, proporcionando un uso fácil y eficaz de esta app web.

Objetivos específicos cumplidos:

* + El correcto funcionamiento del sistema de pago con un comprobante de compra.
  + El correcto funcionamiento de gestión, agregar, eliminar y mostrar productos.
  + Sistema de clasificación de productos por medio de las categorías.
  + Función de búsqueda de productos.
  + Pestañas de detalles para los productos.
  + Sistema de registro e inicio de sesión.
  + Perfil de uso administrador y usuario.
  + Vista de administrador y usuario.
  + Funciones de modificación del carrusel, categorías y productos.
  + Diagramas que evidencian las ventas desde menor a mayor.
  + Funciones de agregar e editar la información de vivienda del usuario.

Hay algunas varias funciones que aun no funcionan correctamente o no se han implementado, tales como la modificación de implementos visuales como los banners para hacer las vistas de los usuarios más atractivas desde la vista de administrador.

pero esos temas no serán descritos en este manual debido a que podrían cambiar o ser eliminados con el transcurso del desarrollo.

**2.1.2 Descripción de procesos**

- Mapa de procesos de la empresa:

- Diagramas de caso de uso:

**Diagramas de caso de uso agregar producto y categoría (admin):**

El **actor principal** es el **Administrador (Admin)**, quien interactúa con el sistema para **gestionar categorías y productos como se muestra en el diagrama a continuación**.

**1.** **Inicia sesión**

* El administrador accede al sistema ingresando sus credenciales.
* El servidor valida y devuelve la **vista home**.

**2.** **Gestión de categorías**

* El administrador selecciona el apartado **“categoría”**.
* El sistema devuelve el **formulario de categoría**.
* El administrador puede **crear una categoría**.
* El sistema guarda la información en la **base de datos**.
* El sistema devuelve la **vista “ver categoría”** mostrando la categoría creada.

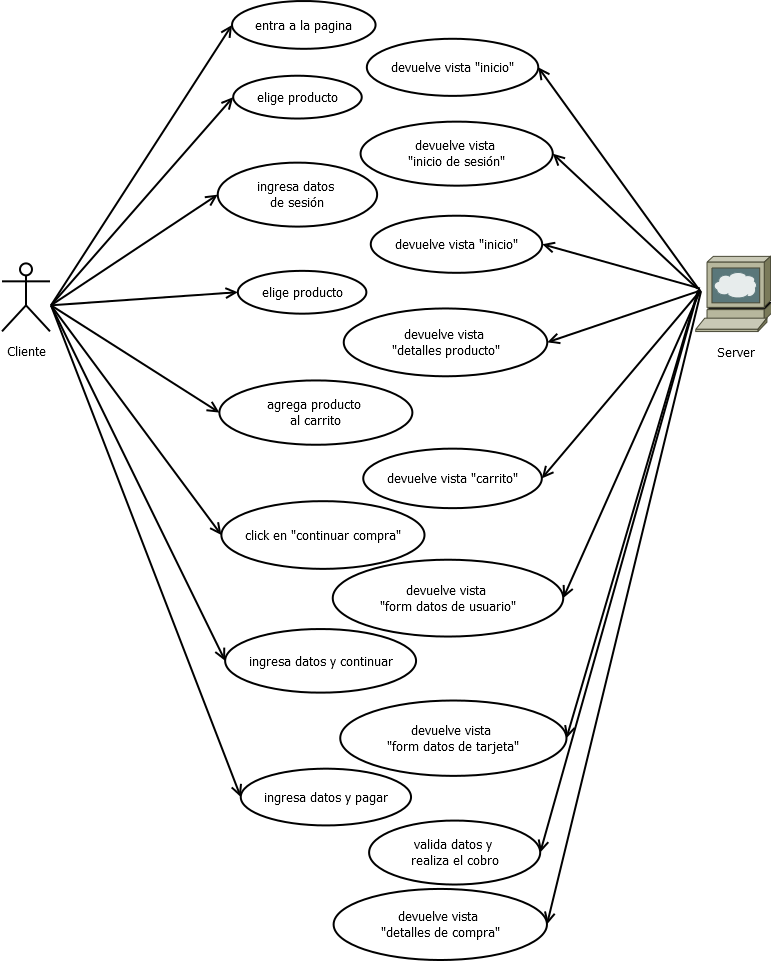
**3.** **Gestión de productos**

* El administrador selecciona el apartado **“Agregar producto”**.
* El sistema devuelve el **formulario de producto**.
* El administrador puede **crear un producto y asignar una categoría**.
* El sistema guarda la información en la **base de datos**.
* El sistema devuelve la **vista “ver producto”** mostrando el producto creado.

**Diagrama de caso de uso para comprar producto (usuario):**

El **actor principal** es el **Cliente**, quien interactúa con el sistema para **navegar, seleccionar productos y realizar una compra así como se ve en el diagrama a continuación**.

1. **Entra a la página**
   * El cliente accede al sitio web.
   * El servidor devuelve la **vista “inicio”**.
2. **Inicio de sesión (opcional)**
   * El cliente puede ingresar sus **datos de sesión**.
   * El servidor valida y devuelve nuevamente la **vista “inicio”**.
3. **Selección de producto**
   * El cliente elige un producto del catálogo.
   * El servidor devuelve la **vista “detalles producto”** mostrando la información completa.
4. **Agregar al carrito**
   * El cliente selecciona la opción de **agregar producto al carrito**.
   * El servidor devuelve la **vista “carrito”** con el listado de productos seleccionados.
5. **Continuar compra**
   * El cliente hace clic en **“continuar compra”**.
   * El servidor devuelve la **vista “form datos de usuario”** para completar información personal.
6. **Ingreso de datos de usuario**
   * El cliente ingresa sus datos (dirección, contacto, etc.) y continúa.
   * El servidor devuelve la **vista “form datos de tarjeta”**.
7. **Ingreso de datos de pago**
   * El cliente ingresa la información de la tarjeta.
   * El sistema valida los datos y realiza el cobro.
8. **Confirmación de compra**
   * El servidor devuelve la **vista “detalles de compra”**, mostrando el comprobante y resumen de la transacción.

****

- Lista de requerimientos funcionales:

Aquí se presenta una lista de las cosas que el sistema debe poder hacer

1. El sistema debe permitir el funcionamiento correcto del módulo de pagos, generando un comprobante de compra.
2. El sistema debe permitir la gestión de productos (agregar, eliminar y mostrar).
3. El sistema debe clasificar productos por categorías.
4. El sistema debe contar con una función de búsqueda de productos.
5. El sistema debe mostrar pestañas de detalle para cada producto.
6. El sistema debe permitir el registro e inicio de sesión de usuarios.
7. El sistema debe manejar perfiles de administrador y usuario.
8. El sistema debe mostrar vistas diferenciadas para administrador y usuario.
9. El sistema debe permitir modificar el carrusel, categorías y productos.
10. El sistema debe generar diagramas de ventas ordenadas de menor a mayor.
11. El sistema debe permitir agregar y editar la información de vivienda del usuario.
12. El sistema debe permitir agregar en favoritos productos seleccionados por el usuario.
13. El sistema debe permitir guardar productos en un carrito de compras.
14. El sistema debe mantener el perfil del usuario activo hasta que este cierre sesión.
15. El sistema debe permitir la modificación de banners o elementos de estilo desde la vista de administrador.
16. El sistema debe contar con una vista de “Sobre nosotros” con número de contacto y correo de asistencia.

**2.1.3 Diagramas UML**

**-** Diagramas de casos de uso:

**Diagramas de caso de uso agregar producto y categoría (admin):**

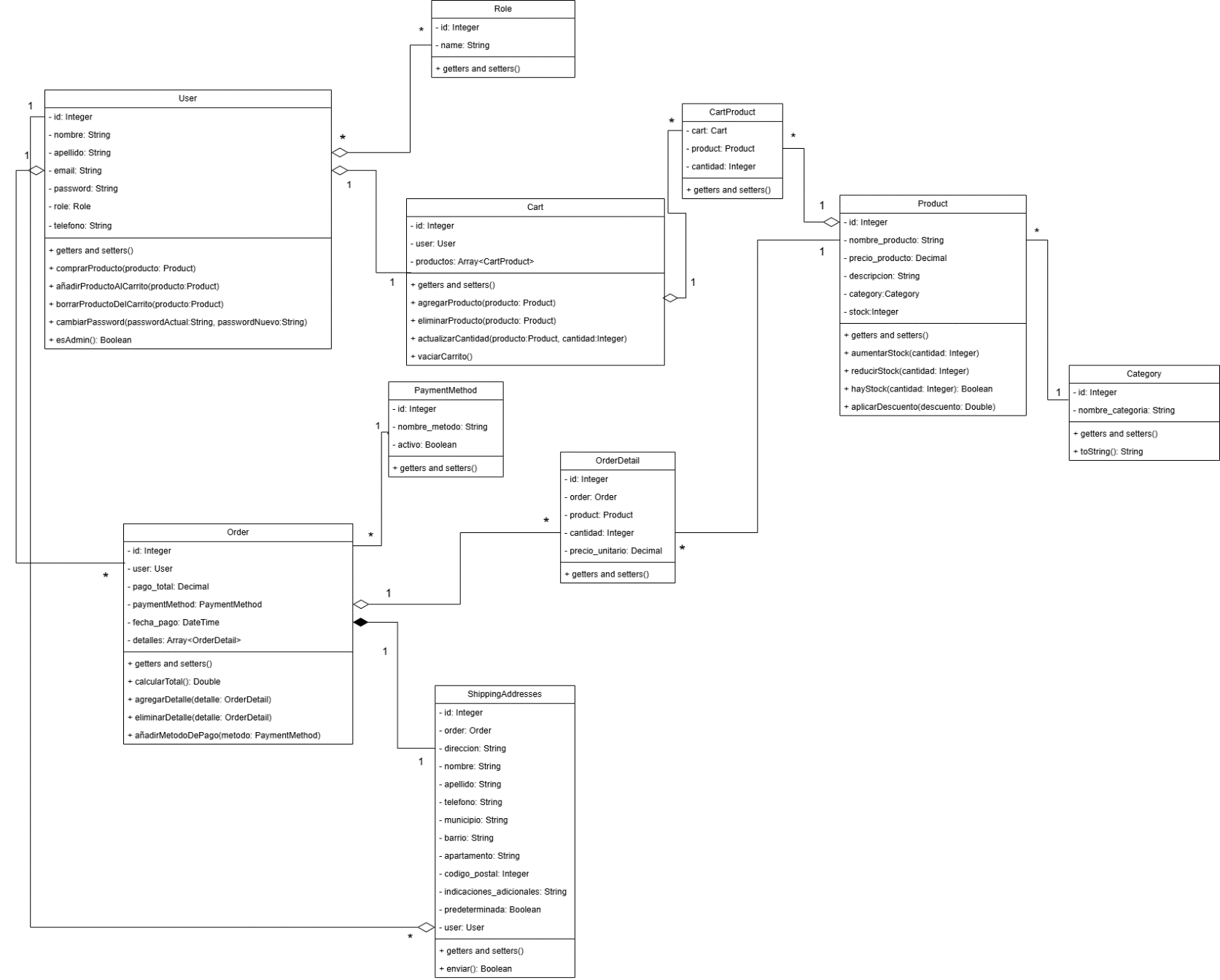
Diagrama

Descripción generada automáticamente

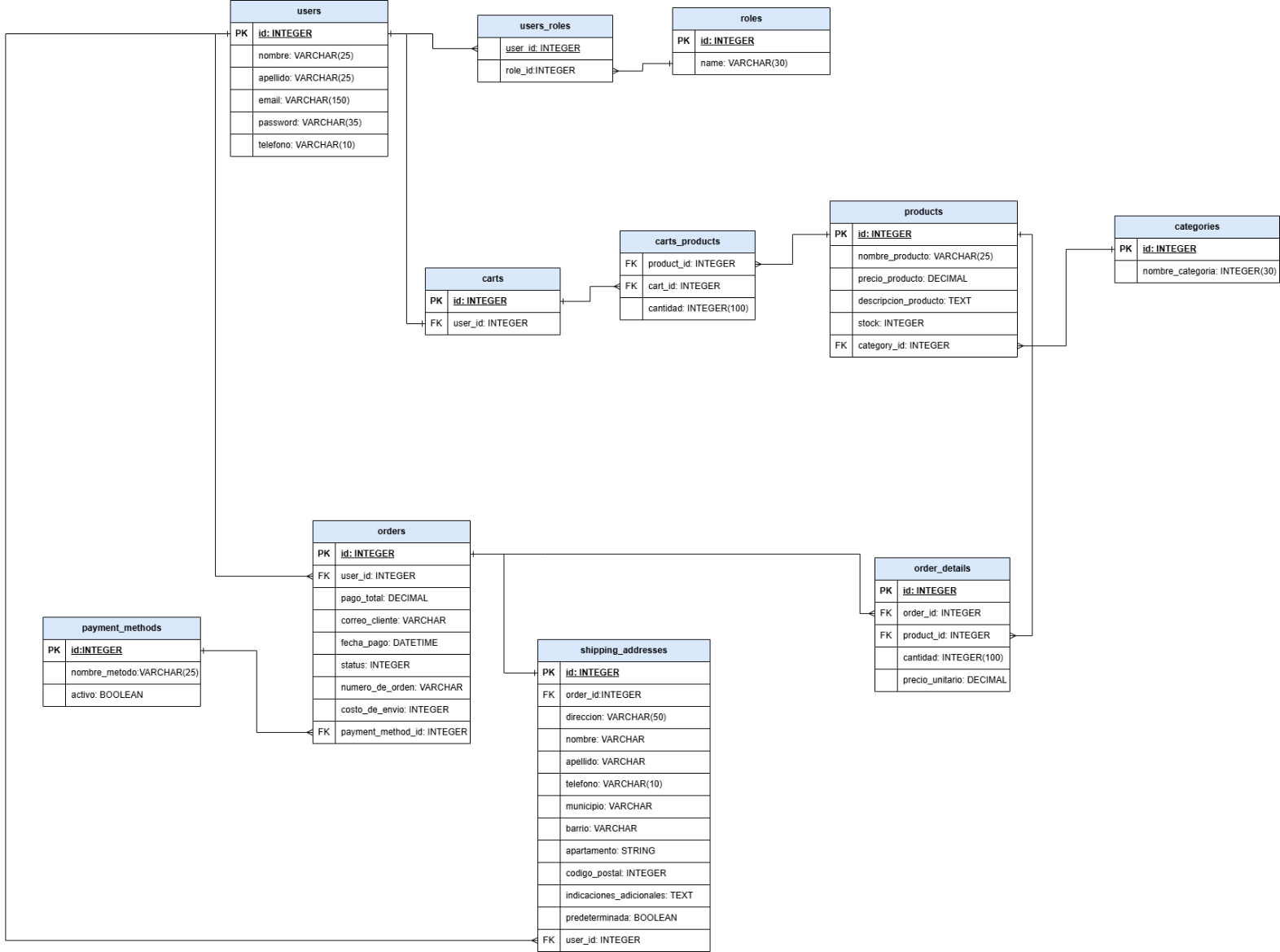
**Diagrama

Descripción generada automáticamenteDiagrama de caso de uso para comprar producto (usuario):**

**Diagrama de clases:**

****

**Diagramas de entidad relación:**



**2.1.4 Administración de usuarios**

Funciones para asignar el rol de admin a un no existen, estos roles deben asignarse de forma manual en la terminal del sistema, pero se agrega por forma predeterminada el rol de usuario al iniciar sesión o registrarse.

**Tabla

Descripción generada automáticamente2.1.5 Modelo relacional de base de datos**

**Diccionario de Datos**

### 1. **users (Usuarios)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único del usuario. |
| nombre | VARCHAR(25) | Nombre del usuario. |
| apellido | VARCHAR(25) | Apellido del usuario. |
| email | VARCHAR(150) | Correo electrónico del usuario (único). |
| password | VARCHAR(35) | Contraseña de acceso encriptada. |
| telefono | VARCHAR(10) | Teléfono de contacto del usuario. |

### 2. **roles (Roles de usuario)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único del rol. |
| name | VARCHAR(30) | Nombre del rol (ej: Admin, Cliente, Vendedor). |

### 3. **users\_roles (Relación Usuarios-Roles)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| user\_id | INTEGER (FK) | Usuario asociado. |
| role\_id | INTEGER (FK) | Rol asignado. |

### 4. **categories (Categorías)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único de la categoría. |
| nombre\_categoria | VARCHAR(30) | Nombre de la categoría de productos. |

### 5. **products (Productos)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único del producto. |
| nombre\_producto | VARCHAR(25) | Nombre del producto. |
| precio\_producto | DECIMAL | Precio unitario del producto. |
| description\_producto | TEXT | Descripción detallada del producto. |
| stock | INTEGER | Cantidad disponible en inventario. |
| category\_id | INTEGER (FK) | Categoría a la que pertenece el producto. |

### 6. **carts (Carritos de compra)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único del carrito. |
| user\_id | INTEGER (FK) | Usuario propietario del carrito. |

### 7. **carts\_products (Relación Carrito-Productos)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| product\_id | INTEGER (FK) | Producto agregado al carrito. |
| cart\_id | INTEGER (FK) | Carrito al que pertenece. |
| cantidad | INTEGER(100) | Cantidad de unidades de ese producto en el carrito. |

### 8. **orders (Órdenes de compra)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único de la orden. |
| user\_id | INTEGER (FK) | Usuario que realizó la orden. |
| pago\_total | DECIMAL | Monto total de la orden. |
| correo\_cliente | VARCHAR | Correo de contacto del cliente. |
| fecha\_pago | DATETIME | Fecha en que se realizó el pago. |
| status | INTEGER | Estado de la orden (ej: pendiente, pagada, enviada). |
| numero\_de\_orden | VARCHAR | Número de referencia de la orden. |
| costo\_de\_envio | INTEGER | Costo del envío. |
| payment\_method\_id | INTEGER (FK) | Método de pago utilizado. |

### 9. **order\_details (Detalles de orden)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único del detalle. |
| order\_id | INTEGER (FK) | Orden a la que pertenece. |
| product\_id | INTEGER (FK) | Producto incluido en la orden. |
| cantidad | INTEGER(100) | Cantidad de unidades compradas. |
| precio\_unitario | DECIMAL | Precio unitario del producto al momento de la compra. |

### 10. **shipping\_addresses (Direcciones de envío)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único de la dirección. |
| order\_id | INTEGER (FK) | Orden asociada a la dirección. |
| direccion | VARCHAR(50) | Dirección principal de envío. |
| nombre | VARCHAR | Nombre del destinatario. |
| apellido | VARCHAR | Apellido del destinatario. |
| telefono | VARCHAR(10) | Teléfono de contacto. |
| municipio | VARCHAR | Municipio o ciudad. |
| barrio | VARCHAR | Barrio o zona. |
| apartamento | STRING | Número de apartamento u oficina. |
| codigo\_postal | INTEGER | Código postal. |
| indicaciones\_adicionales | TEXT | Observaciones para el envío. |
| predeterminada | BOOLEAN | Si es la dirección predeterminada del usuario. |
| user\_id | INTEGER (FK) | Usuario propietario de la dirección. |

### 11. **payment\_methods (Métodos de pago)**

| **Atributo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único del método de pago. |
| nombre\_metodo | VARCHAR(25) | Nombre del método de pago (ej: Tarjeta, PayPal, Efectivo). |
| active | BOOLEAN | Indica si el método está habilitado. |

**2.1.6 Descripción de plataforma**

Las herramientas y lenguajes de programación usadas para el desarrollo de este sistema son:

* + **Aplicativos/programas:**

La mesa de trabajo o programa de escritorio que se uso en este proceso fue Visual Studio code.

* + para la creación y desarrollo de la base de datos fue usado MySQL y Docker.
  + **Lenguaje de programación:**
  + Ruby on rails.
  + **Lenguaje de secuencia:**
  + JavaScript.
  + **Framework:**
  + Bootstrap.
  + React.
  + **Lenguaje de marcado y estilos:**
  + HTML.
  + CSS.

Además de lo antes mencionado también trabajamos con el sistema operacional Windows11 y el posterior a este el Windows 10.

**2.1.7 documentación del Código fuente**

**Estructuras del Proyecto**

Aquí encontrarás la estructura base de nuestro proyecto como las estructuras generales de cada proceso.

├── 📁Backend # aqui se encuentra el servidor y las funciones principales

|

├── 📁Frontend # aqui se encuentra principalmente la aplicación movil

│ └── 📁ale\_beauty\_art\_app/ # aplicación movil

|

├── 📁Diagramas # Diagramas entidad relación y clases

|

├── 📁Mockup # diseño de interfaz grafica antes de implementarse

|

├── 📁docs # informacion detallada de los procesos,

| elementos, clases, etc... del aplicativo

|

└── 📄README.md # Este archivo README normalmente documenta todos los pasos

necesarios para poner la aplicación en funcionamiento.

#### **Backend**

|── 📁.github

│ ├── 📁workflows

│ └── 📄dependabot.yml

|

|── 📁.kamal

│ ├── 📁hooks

│ └── 📄secrets

|

|── 📁app

| ├── 📁assets

│ ├── 📁controllers

| ├── 📁helpers

| ├── 📁javascript

| ├── 📁jobs

| ├── 📁mailers

│ ├── 📁models

│ └── 📁views

|

├── 📁config

│ ├── 📄routes.rb

│ └── 📄database.yml

|

├── 📁db

│ ├── 📁migrate

│ └── 📄schema.rb

|

├── 📁lib

│ └── 📁tasks

│ └── 📄.keep

|

├── 📁log

│ ├── 📄.keep

└── 📄development.log

|

├── 📁public

│ ├── <> 400.html

│ ├── <> 404.html

│ ├── <> 406-unsupported-browser.html

│ ├── <> 422.html

│ ├── <> 500.html

│ ├── 📄icon.png

│ ├── 📄icon.svg

│ └── 📄robots.txt

|

├── 📁script

│ └── 📄.keep

|

├── 📁storage

|

├── 📁test

│ ├── 📁controllers

│ ├── 📁fixtures

| ├── 📁helpers

| ├── 📁integration

| ├── 📁mailers

│ ├── 📁models

│ ├── 📁system

│ ├── 📄aplication\_system\_test\_case.rb

│ └── 📄test\_helper.rb

|

├── 📁tmp

|

├── 📁test

│ ├── 📁javascript

│ └── 📄.keep

|

├── 📄.rubocop.yml

├── 📄.ruby-version

├── 📄config.ru

├── 📄docker-compose.yml

├── 📄Dockerfile

├── 📄Gemfile

├── 📄Gemfile.lock

└── 📄Rakefile