



Instituto Técnico Ricaldone

Tema: Manual de técnico - D&M Systems

Asignatura: Conversación en inglés sobre mantenimiento de sistemas informáticos.

Integrantes:

Jafet Aarón, Melara Paises	20220188
Rodrigo José, Díaz Hernández	20190380
Kevin Vladimir, Rodríguez Alvarado	20220286
Juan Pablo, Flamenco Pineda	20220671
Aleck Fernando, Rodríguez Nuñez	20190131

Docente: Daniel Wilfredo Granados

Grado/Sección: Tercer año 1B

Especialidad: Desarrollo de Software

Entrega: San Salvador, 30 de septiembre de 2024

Índice

Instituto Técnico Ricaldone	1
Índice	2
Introducción	3
Tecnologías y herramientas implementadas	4
Estructura de la base de datos	5
Diccionario de datos	7
Arquitectura del software.	16
Estructura del proyecto	18
Directorio general de carpetas	18
Carpeta API	19
Subcarpeta “Helpers”	20
Subcarpeta “images”	20
Subcarpeta “libraries”	20
Subcarpeta “models”	21
Subcarpeta “Reports”	21
Subcarpeta “services”	22
Subcarpeta “Vendor”	22
Carpeta controllers	23
Carpeta “resources”	23
Carpeta “views”	24
Diseño de la aplicación.	25
Buenas prácticas de desarrollo.	26
Requerimientos de hardware y software	29
Instalación y configuración	30

Introducción

El presente manual técnico describe las características y detalles del funcionamiento del Sistema de Gestión Integral implementado en la Farmacia Central de San Juan Opico, diseñado para optimizar los procesos clave de la operación farmacéutica. Este sistema tecnológico permite una administración centralizada y estructurada que responde a las necesidades operativas y normativas del sector.

El sistema ha sido desarrollado con el objetivo de asegurar el éxito y la organización de la farmacia, reduciendo el riesgo de errores humanos y asegurando un control riguroso sobre el inventario y las transacciones diarias.

Ventajas del sistema

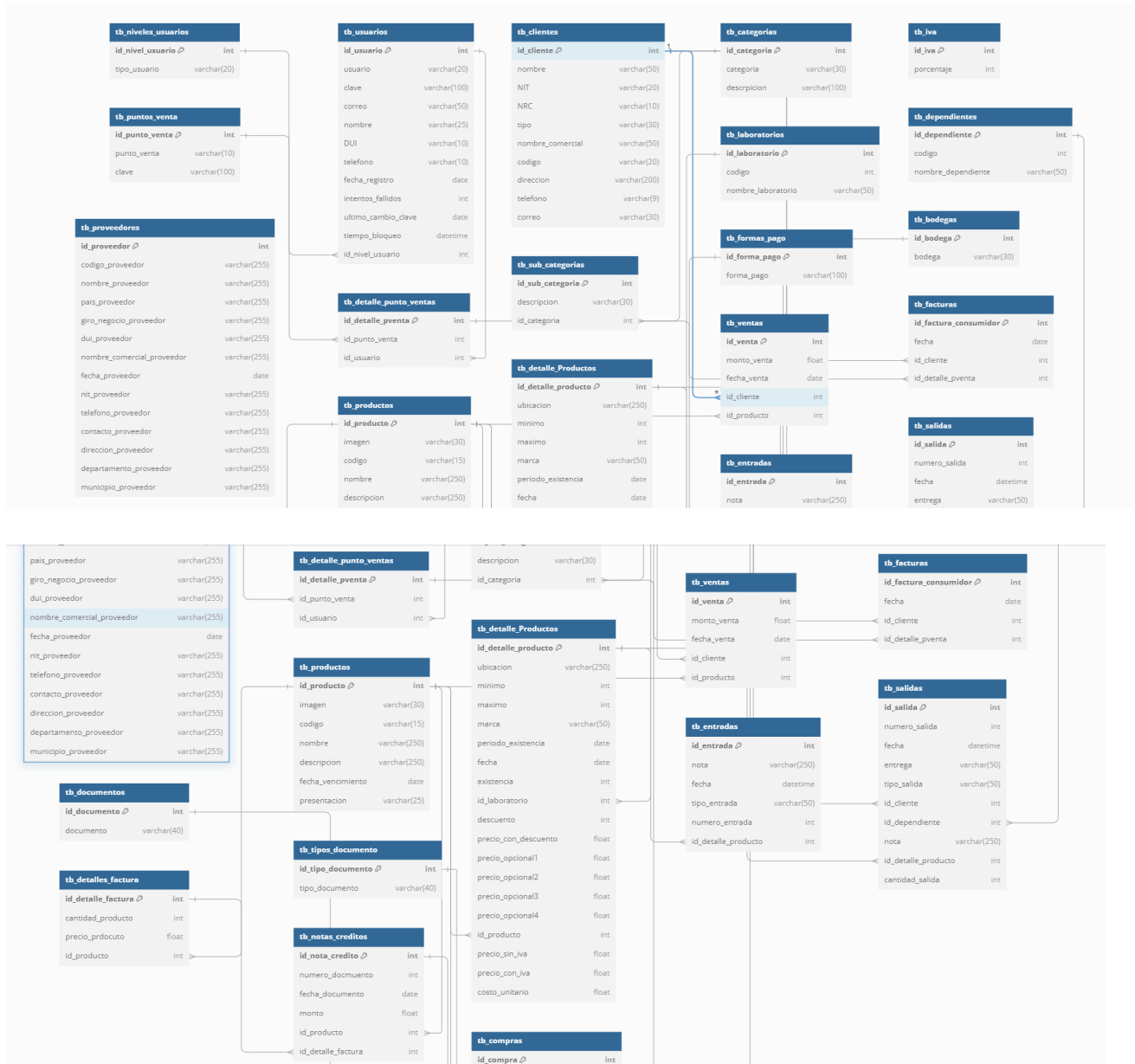
- **Automatización de procesos críticos:** El sistema permite automatizar tareas clave, como la actualización de inventario y la generación de reportes, liberando tiempo para que el personal se enfoque en actividades más estratégicas.
- **Seguridad y control de acceso:** La estructura de niveles de usuario garantiza que cada empleado solo pueda acceder a las funciones necesarias para su rol, protegiendo la integridad de los datos y cumpliendo con los estándares de seguridad informática además de contar con validaciones para obtener los datos necesarios y que cumplan los requisitos.
- **Mejora continua:** El sistema está diseñado para ser flexible y escalable, permitiendo futuras actualizaciones y mejoras que se adapten al crecimiento de la farmacia y a las demandas del entorno regulatorio y del mercado.

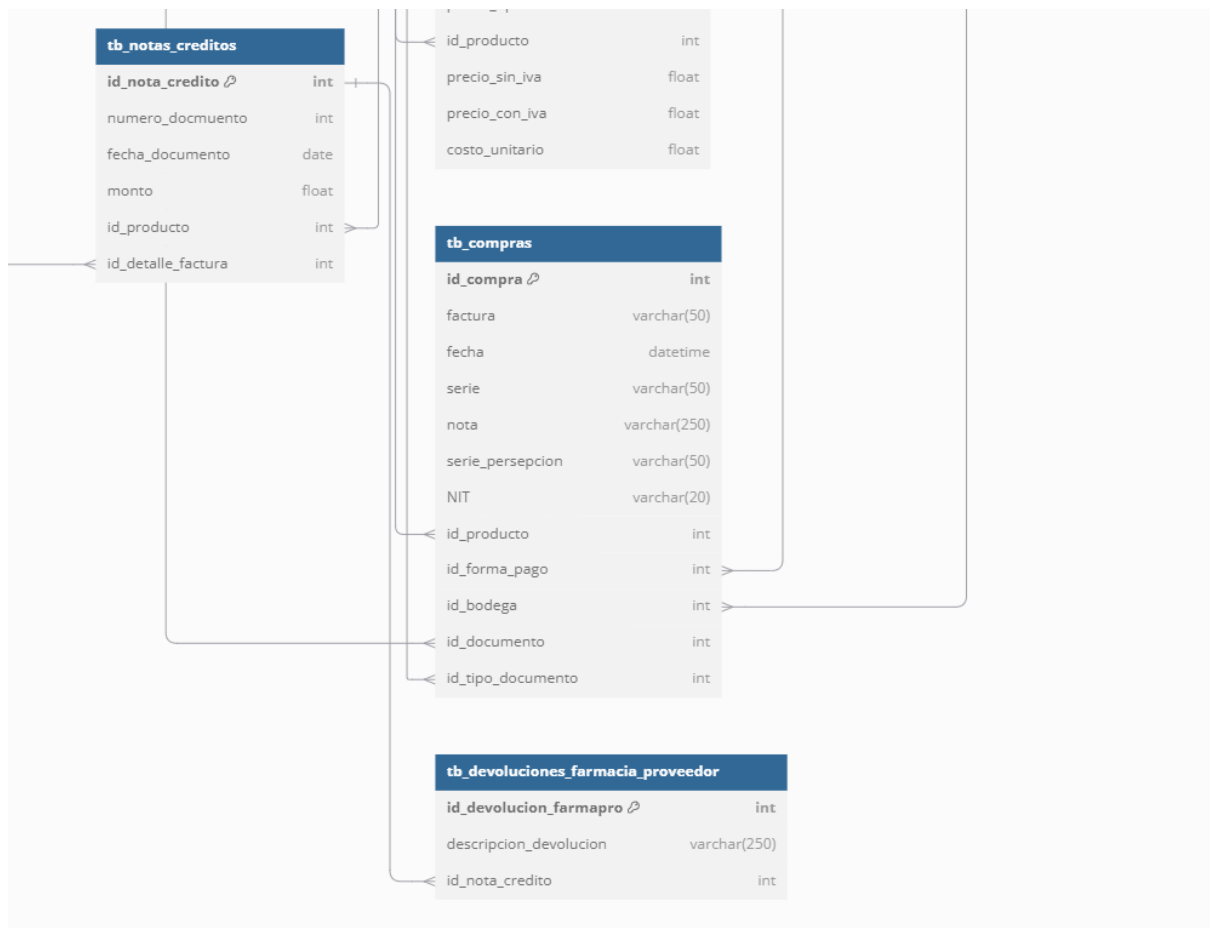
Tecnologías y herramientas implementadas

El presente proyecto utiliza Visual Studio Code como IDE. Los lenguajes de programación empleados para la realización del proyecto fueron HTML, JavaScript y PHP en el caso del sitio web. Para el apartado móvil se utilizó el mismo IDE pero empleando Node JS, React Native y librerías pertenecientes a dichas herramientas.



Para gestionar los datos se utilizaron gestores de bases de datos como HeidiSQL y SQL Workbench. El proyecto utiliza una estructura relacional de base de datos, esto permite mostrar resultados con información útil en el proyecto gracias a la capacidad de conectar diferentes tablas, potenciando el uso que se le puede dar a la base de datos en las dos presentaciones del sistema, sitio web y aplicación móvil

Estructura de la base de datos





Para acceder al código de la estructura de la base de datos debe acceder al siguiente [enlace](#). Una vez esté dentro de la carpeta de drive, acceda al archivo llamado **DM_systems (Tablas).sql**

 **DM_systems (Tablas).sql** 

Diccionario de datos

Tabla	Campo	Tipo de dato	Detalles
tb_niveles_usuarios	id_nivel_usuario	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	tipo_usuario	VARCHAR(20)	
tb_usuarios	id_usuario	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	usuario	VARCHAR(20)	UNIQUE
	clave	VARCHAR(100)	UNIQUE, NOT NULL
	correo	VARCHAR(50)	UNIQUE, NOT NULL
	nombre	VARCHAR(25)	
	DUI	VARCHAR(10)	UNIQUE, NOT NULL
	telefono	VARCHAR(10)	UNIQUE
	fecha_registro	DATE	DEFAULT NOW()
	intentos_fallidos	INT	DEFAULT 0
	ultimo_cambio_clave	DATE	DEFAULT NOW()
	tiempo_bloqueo	DATETIME	NULL

	id_nivel_usuario	INT	FOREIGN KEY REFERENCES tb_niveles_usuarios
tb_clientes	id_cliente	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	nombre	VARCHAR(50)	
	NIT	VARCHAR(20)	UNIQUE, NULL
	NRC	VARCHAR(10)	NULL
	tipo	VARCHAR(30)	NULL
	nombre_comercial	VARCHAR(50)	NULL
	codigo	VARCHAR(20)	UNIQUE, NOT NULL
	direccion	VARCHAR(200)	NULL
	telefono	VARCHAR(9)	UNIQUE, NULL
	correo	VARCHAR(30)	UNIQUE
tb_categorias	id_categoria	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	categoria	VARCHAR(30)	UNIQUE
	descripcion	VARCHAR(100)	NULL
tb_iva	id_iva	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL

	porcentaje	INT	CHECK(porcentaje > 0)
tb_puntos_venta	id_punto_venta	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	punto_venta	VARCHAR(10)	
	clave	VARCHAR(100)	UNIQUE, NOT NULL
tb_detalle_punto_ventas	id_detalle_venta	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	id_punto_venta	INT	FOREIGN KEY REFERENCES tb_puntos_venta
	id_usuario	INT	FOREIGN KEY REFERENCES tb_usuarios
tb_sub_categorias	id_sub_categoria	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	descripcion	VARCHAR(30)	NULL
	id_categoria	INT	FOREIGN KEY REFERENCES tb_categorias
tb_laboratorios	id_laboratorio	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	codigo	INT	UNIQUE

	nombre_laboratorio	VARCHAR(50)	
tb_dependientes	id_dependiente	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	codigo	INT	UNIQUE
	nombre_dependiente	VARCHAR(50)	
tb_proveedores	id_proveedor	INT	PRIMARY KEY, AUTO_INCREMENT
	codigo_proveedor	VARCHAR(255)	UNIQUE
	nombre_proveedor	VARCHAR(255)	
	pais_proveedor	VARCHAR(255)	
	giro_negocio_proveedor	VARCHAR(255)	
	dui_proveedor	VARCHAR(255)	UNIQUE
	nombre_comercial_proveedor	VARCHAR(255)	UNIQUE
	fecha_proveedor	DATE	
	nit_proveedor	VARCHAR(255)	UNIQUE
	telefono_proveedor	VARCHAR(255)	UNIQUE
	contacto_proveedor	VARCHAR(255)	UNIQUE
	direccion_proveedor	VARCHAR(255)	UNIQUE

	departamento_proveedor	VARCHAR(255)	
	municipio_proveedor	VARCHAR(255)	
tb_productos	id_producto	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	imagen	VARCHAR(30)	
	codigo	VARCHAR(15)	UNIQUE
	nombre	VARCHAR(250)	
	descripcion	VARCHAR(250)	
	fecha_vencimiento	DATE	
	presentacion	VARCHAR(25)	
tb_detalle_productos	id_detalle_producto	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	ubicacion	VARCHAR(250)	
	minimo	INT	
	maximo	INT	
	marca	VARCHAR(50)	
	periodo_existencia	DATE	
	fecha	DATE	

	existencia	INT	CHECK(existencia >= 0)
	id_laboratorio	INT	FOREIGN KEY REFERENCES tb_laboratorios
	descuento	INT	CHECK (descuento >= 0)
	precio_con_descuento	FLOAT	CHECK (precio_con_descuento >= 0)
	precio_opcional1	FLOAT	CHECK (precio_opcional1 >= 0)
	precio_opcional2	FLOAT	CHECK (precio_opcional2 >= 0)
	precio_opcional3	FLOAT	CHECK (precio_opcional3 >= 0)
	precio_opcional4	FLOAT	CHECK (precio_opcional4 >= 0)
	id_producto	INT	FOREIGN KEY REFERENCES tb_productos
	precio_sin_iva	FLOAT	CHECK (precio_sin_iva > 0)
	precio_con_iva	FLOAT	CHECK (precio_con_iva > 0)

	costo_unitario	FLOAT	CHECK (costo_unitario > 0)
tb_formas_pago	id_forma_pago	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	forma_pago	VARCHAR(100)	
tb_bodegas	id_bodega	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	bodega	VARCHAR(30)	
tb_documentos	id_documento	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	documento	VARCHAR(40)	
tb_tipos_documento	id_tipo_documento	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	tipo_documento	VARCHAR(40)	
tb_compras	id_compra	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	factura	VARCHAR(50)	UNIQUE
	fecha	DATE	
	fecha_registro	DATE	

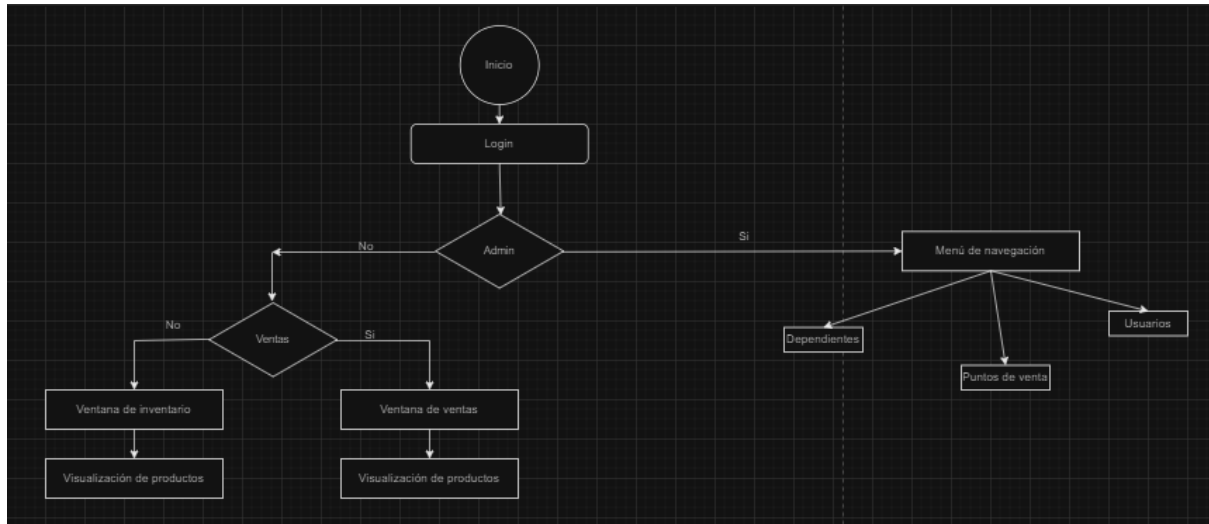
	id_proveedor	INT	FOREIGN KEY REFERENCES tb_proveedores
	id_tipo_documento	INT	FOREIGN KEY REFERENCES tb_tipos_documento
	id_usuario	INT	FOREIGN KEY REFERENCES tb_usuarios
	id_bodega	INT	FOREIGN KEY REFERENCES tb_bodegas
tb_detalle_compras	id_detalle_compras	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	cantidad	INT	CHECK(cantidad > 0)
	total	FLOAT	CHECK(total > 0)
	id_producto	INT	FOREIGN KEY REFERENCES tb_productos
	id_compra	INT	FOREIGN KEY REFERENCES tb_compras
tb_detalle_pagos	id_detalle_pago	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
	id_forma_pago	INT	FOREIGN KEY REFERENCES tb_formas_pago

	id_compra	INT	FOREIGN KEY REFERENCES tb_compras
--	-----------	-----	---

Arquitectura del software.

A continuación, se mostrarán los diferentes tipos de diagrama de flujo de datos para las acciones las cuales se realizan en el programa también el diagrama de la base de datos.

Login



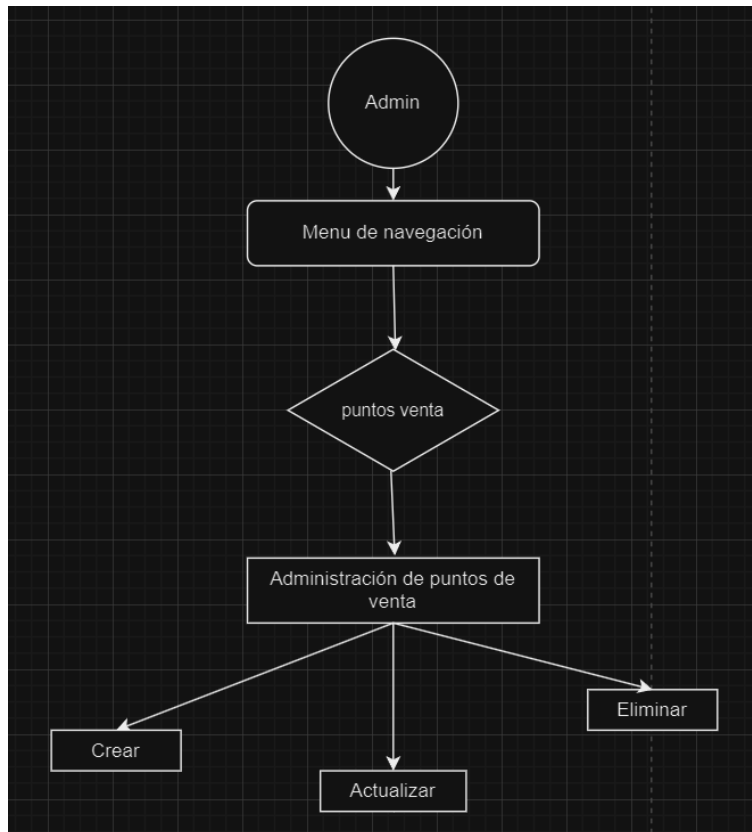
Usuarios



Usuarios

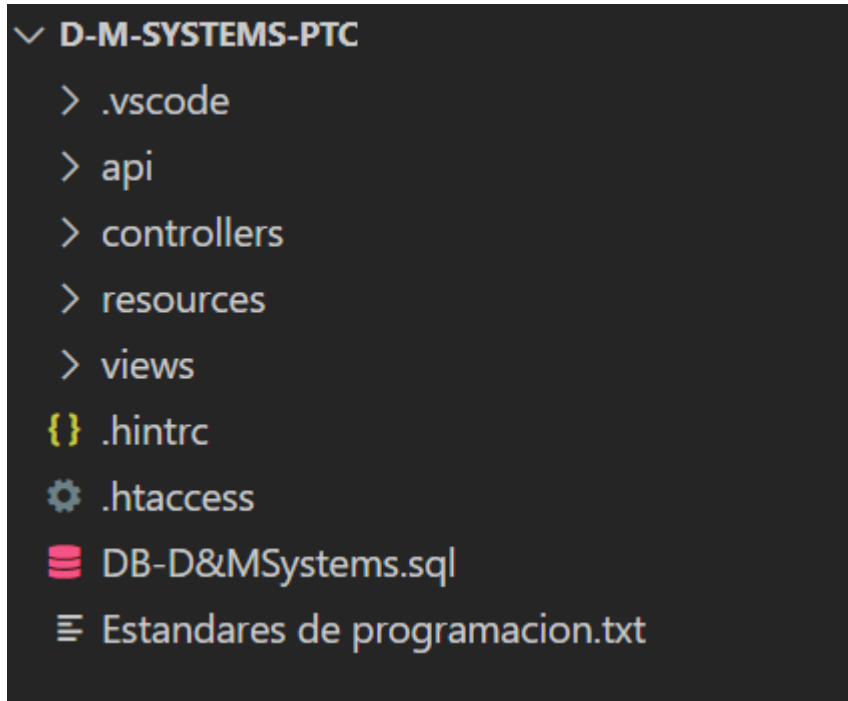


Puntos de venta



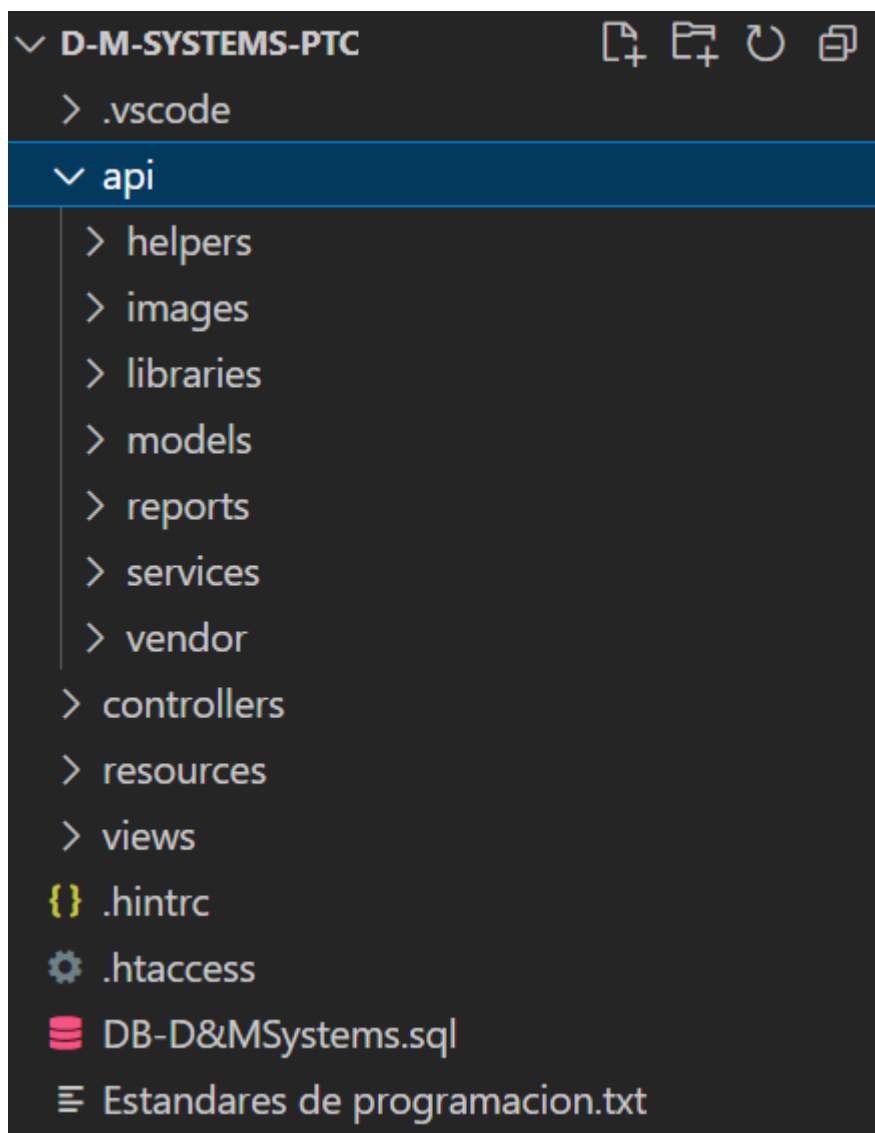
Estructura del proyecto

Directorio general de carpetas



Aca podemos observar todas las carpetas y archivos extra que conforman nuestro proyecto

Carpeta API

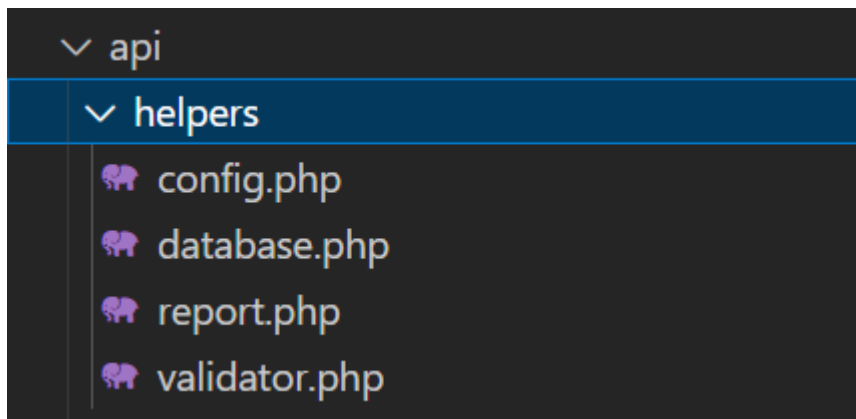


Al abrir esta carpeta podremos encontrar los archivos de programación mayoritariamente de php que son:

- helpers
- Models
- Services
- Reports
- Vendors

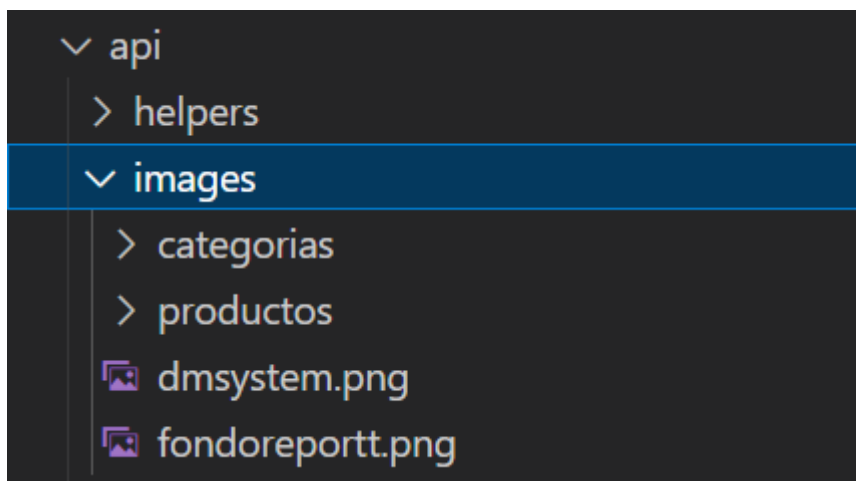
También encontramos la carpeta de imágenes y librerías que veremos más adelante.

Subcarpeta "Helpers"



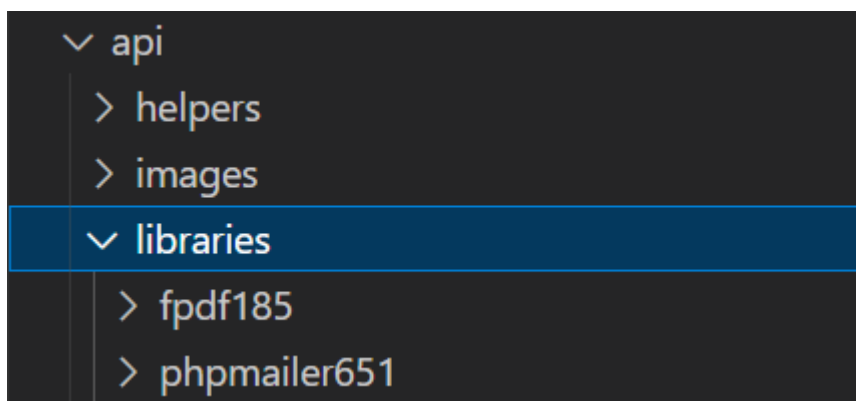
Aquí podemos observar los archivos que configuran el proyecto entero por ejemplo **"database.php"** que sirve para hacer la conexión a nuestra base de datos.

Subcarpeta "images"



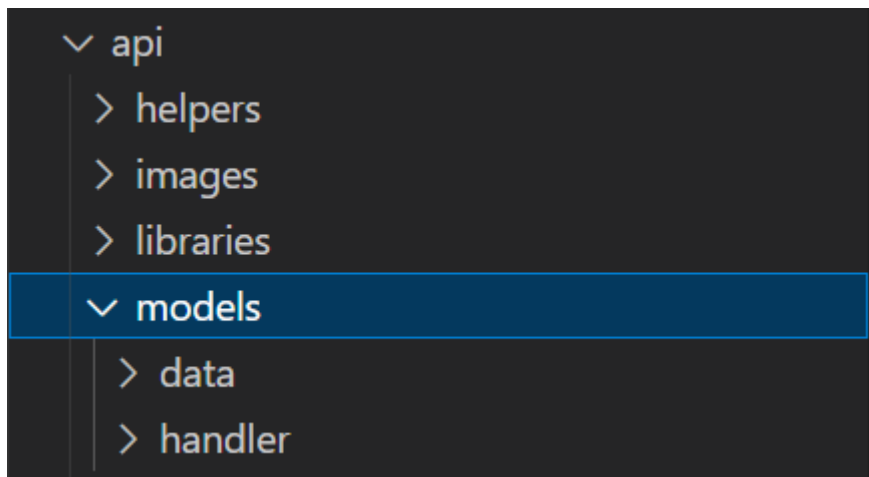
Aquí almacenamos las imágenes que utilizamos para nuestro proyecto.

Subcarpeta "libraries"



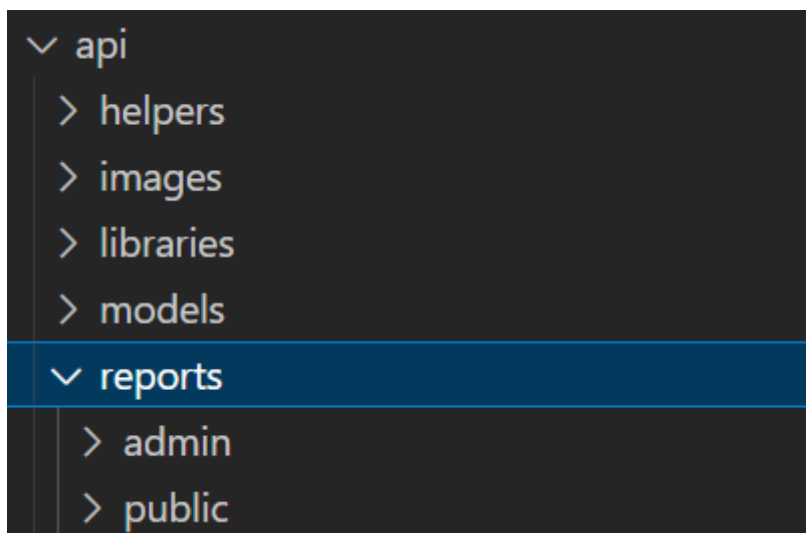
Aquí están las librerías que ocupamos para nuestro proyecto por ejemplo para los reportes y gráficos.

Subcarpeta “models”



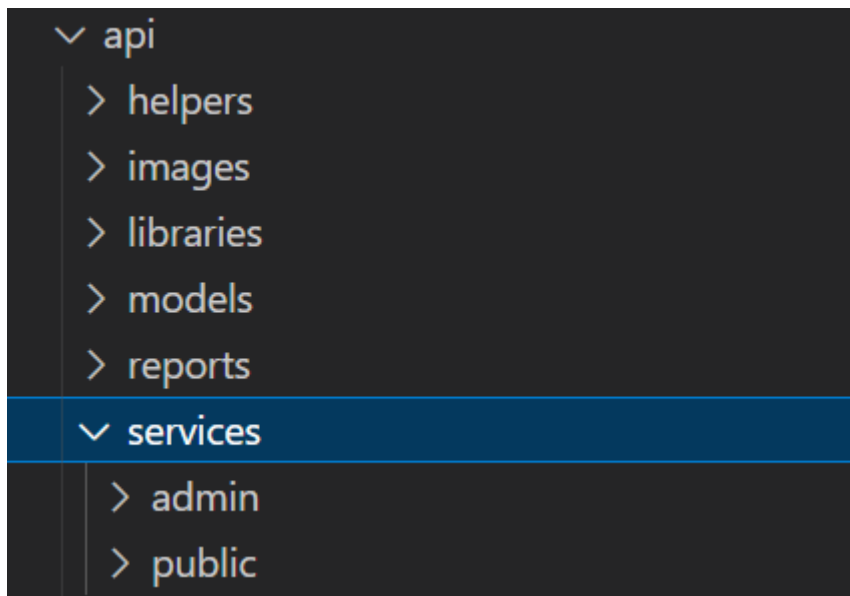
Aquí almacenamos todos los “data y handler” que utilizamos para darle funcionalidad a nuestro proyecto en la parte del backend.

Subcarpeta “Reports”



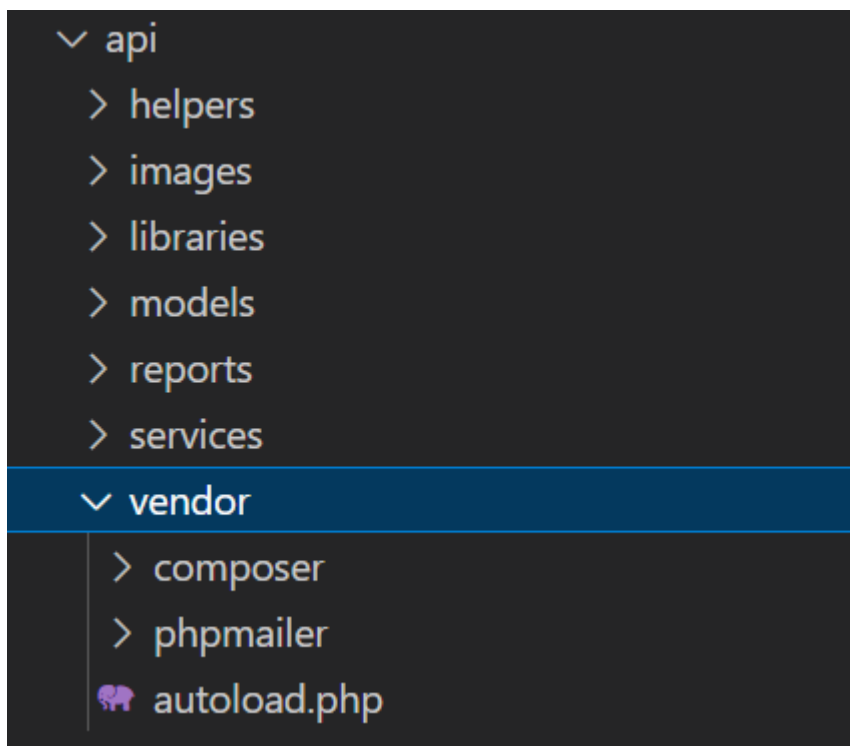
En esta carpeta podemos encontrar la configuración para los reportes.

Subcarpeta "services"



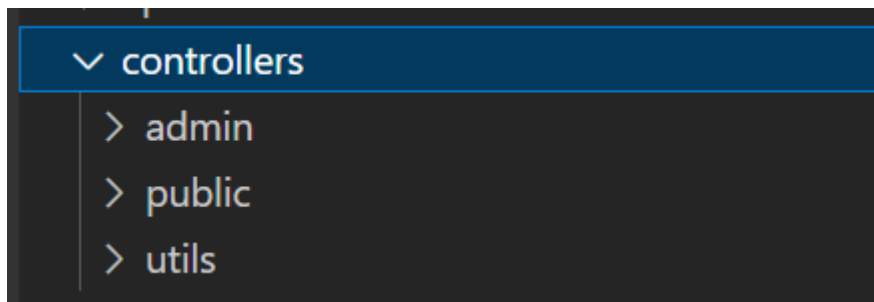
Es esta carpeta encontramos la carpeta de servicios en php en la cual se almacenan en 2 carpetas que corresponden a los dos sitios, público y privado

Subcarpeta "Vendor"



En esta carpeta se guardan librerías.

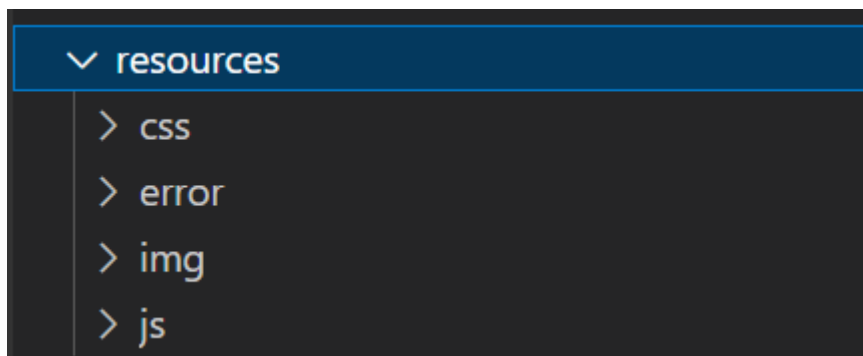
Carpeta controllers



En esta carpeta encontraremos los archivos de programación en Javascript que utilizamos para configurar las páginas web acá encontramos 3 subcarpetas.

- **Admin:** Aca están los controladores del sitio privado.
- **Public:** Acá están los controladores del sitio público.
- **Utils:** Acá se almacenan funciones, constantes y componentes de ambos sitios.

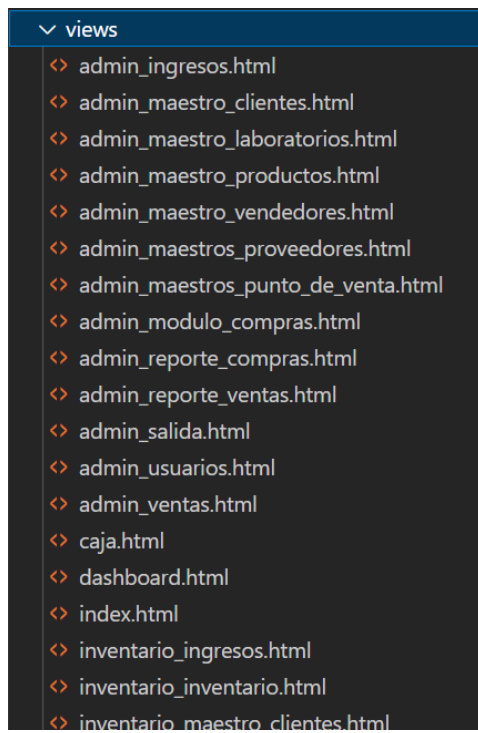
Carpeta “resources”



En esta carpeta encontramos los archivos:

- **Css:** Aquí tenemos los archivos css que ocupamos para nuestro proyecto.
- **Error:** Aquí tenemos los 2 archivos de errores 403 y 404.
- **img:** Aquí tenemos imágenes de fondo.
- **js:** Aquí tenemos archivos de javascript tales como alertas y plantillas.

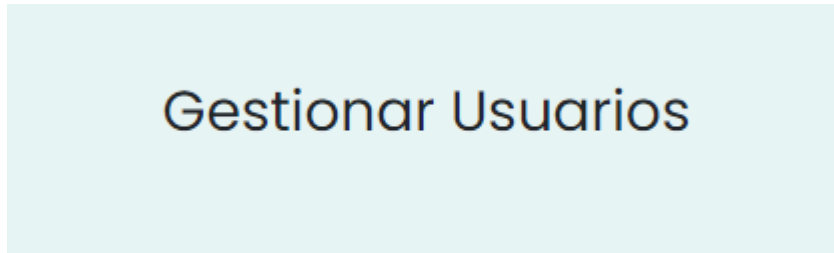
Carpeta “views”



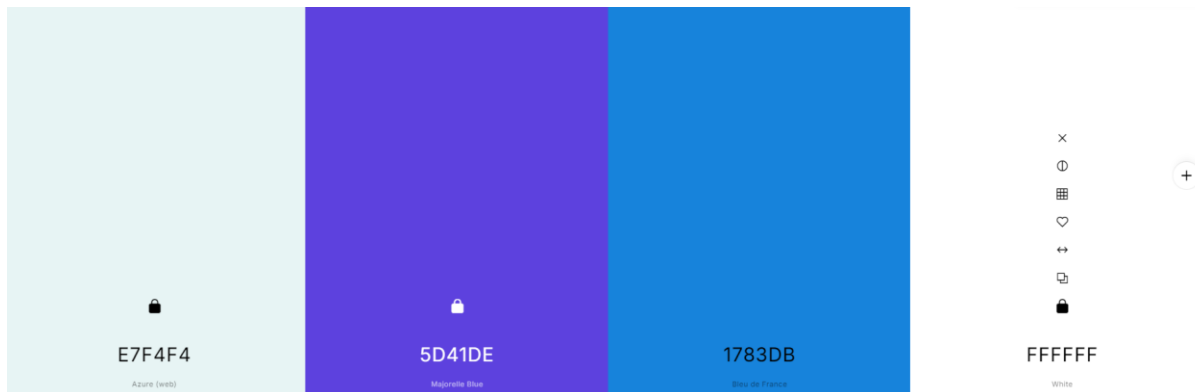
En esta carpeta están todas las vistas en html de todo el sistema.

Diseño de la aplicación.

Fuente: Poppins



Paleta de colores:



Dimensiones:

Botones (**.btn1**, **.btn-seleccionado**):

- Ancho: 200px.
- Alto: 65px.
- Bordes redondeados: 15px.

Contenedores:

- **.container:**
 - Ancho: 300px.
 - Margen superior: 15px.
 - Margen izquierdo: 10px.
- **.largo:**
 - Ancho: 500px.
- **.sinIVA y .largo1:**
 - Ancho: 300px.
 - Margen superior: -53px
 - **.largo1** tiene un margen izquierdo adicional de 300px.

Funcionalidades:

[Tutorial D&M.mp4](#)

Buenas prácticas de desarrollo.

Buenas prácticas utilizadas en el lado del cliente (JavaScript)

- **Uso de Constantes:**
 - Hemos definido constantes para elementos del DOM y para la API, lo que ayuda a evitar errores y mejora la legibilidad.
- **Funciones Bien Definidas:**
 - Cada función tiene un propósito claro, facilitando así la comprensión y el mantenimiento del código.
- **Uso de `async/await`:**
 - Utilizamos `async/await` para manejar las peticiones asíncronas, mejorando la legibilidad en comparación con las promesas anidadas.
- **Manejo de Eventos:**
 - Hemos implementado correctamente el manejo de eventos para interacciones del usuario, asegurando una respuesta fluida a las acciones del usuario.
- **Comentarios en el Código:**
 - Los comentarios que hemos incluido son útiles y explican la lógica detrás de ciertas partes del código, lo que facilita su comprensión.
- **Mensajes de Respuesta:**
 - Utilizamos `sweetAlert` para manejar la retroalimentación visual al usuario, mejorando la experiencia y haciéndola más amigable.

Buenas prácticas utilizadas en el lado del servidor (PHP):

- **Uso de Clases y Objetos:**
 - Hemos implementado la **programación orientada a objetos (OOP)**, lo que permite una mejor organización del código y reutilización de componentes.
- **Validación de Datos:**
 - Antes de procesar cualquier entrada del usuario, realizamos una **validación** para asegurar que los datos sean seguros y en el formato correcto.
- **3. Consultas Preparadas:**
 - Utilizamos **consultas preparadas** para acceder a la base de datos. Esto ayuda a prevenir inyecciones SQL y mejora la seguridad de nuestra aplicación.
- **4. Manejo de Sesiones:**
 - Gestionamos las sesiones de usuario de manera segura, asegurándonos de que solo los usuarios autenticados puedan acceder a ciertas funcionalidades.
- **5. Estructura y Nombres Consistentes:**
 - Mantenemos una estructura clara y consistente en la base de datos y en el código. Usamos **nombres descriptivos** para tablas y columnas, siguiendo la convención de **snake_case**.
- **6. Manejo de Errores:**
 - Implementamos un manejo centralizado de errores usando bloques **try-catch** para garantizar que los errores sean capturados y gestionados adecuadamente.
- **7. Comentarios:**
 - Los comentarios en el código son claros y útiles, explicando la lógica detrás de las funciones y partes clave del código.

Buenas Prácticas Utilizadas en el Lado del Servidor (Base de Datos SQL):

- **Nombres Descriptivos:**
 - Se utilizan nombres claros y descriptivos para las tablas y columnas, facilitando la comprensión de su propósito (por ejemplo, **tb_usuarios**, **tb_clientes**, **tb_productos**).
- **Uso de Tipos de Datos Adecuados:**
 - Se seleccionan tipos de datos apropiados para cada columna, como **INT**, **VARCHAR**, y **DATE**, lo que optimiza el almacenamiento y mejora el rendimiento de las consultas.
- **Claves Primarias y Foráneas:**
 - Se definen claves primarias para cada tabla, asegurando la unicidad de los registros, y se utilizan claves foráneas para mantener la integridad referencial entre las tablas.
- **Constraints para Validación:**
 - Se implementan restricciones (constraints) en las tablas, como **UNIQUE**, **CHECK**, y **NOT NULL**, para garantizar la validez de los datos y evitar entradas no deseadas.
- **Estructura de Tablas Normalizada:**
 - La base de datos está estructurada de manera normalizada, lo que minimiza

la redundancia y mejora la consistencia de los datos (por ejemplo, separando los datos de productos, clientes, proveedores, etc.).

- **Manejo de Fechas y Tiempos:**
 - Se utilizan tipos de datos de fecha y hora (**DATE**, **DATETIME**) para almacenar correctamente las fechas y facilitar las operaciones de comparación y filtrado.
- **Uso de Comentarios:**
 - Se añaden comentarios claros y útiles en el código SQL, explicando la lógica detrás de la creación de tablas y sus relaciones, facilitando la comprensión para futuros desarrolladores.
- **Manejo de Inserciones y Actualizaciones:**
 - Se planifican cuidadosamente las inserciones y actualizaciones, considerando las dependencias entre tablas, para evitar errores de integridad referencial.
- **Seguridad y Prevención de Inyecciones SQL:**
 - Aunque esto se aplica principalmente en el código de la aplicación, la implementación de buenas prácticas en el manejo de consultas y validación de datos también contribuye a la seguridad general de la base de datos.
- **Establecimiento de Relaciones:**
 - Se establecen relaciones claras entre tablas mediante claves foráneas, lo que facilita la realización de uniones (joins) y consultas complejas.

Requerimientos de hardware y software

Requisitos mínimos:

- Sistema operativo:
Windows 10 (Última actualización)
Linux o macOS si planeas usar herramientas de desarrollo específicas o contenedores como Docker.
- Procesador:
Intel Core i5 o AMD Ryzen 5
- Memoria RAM:
8 GB de memoria RAM
- Almacenamiento:
256 GB de almacenamiento disponible

Requisitos recomendados:

- Sistema operativo
Windows 11
Linux o macOS si planeas usar herramientas de desarrollo específicas o contenedores como Docker.
- Procesador:
intel Core i5 o AMD Ryzen 5 o superior
- Memoria RAM:
16 GB de memoria RAM
- Almacenamiento:
512 GB de almacenamiento disponible

Requisitos para trabajar de forma local:

- Tener instalado Visual Studio Code para poder trabajar el código del sitio
- Tener instalado XAMPP o cualquier otro paquete de software libre que permita montar servidores locales.
- Tener instalado un gestor de bases de datos como HeidiSQL, WorkBench o phpmyadmin (se puede ingresar desde xampp)
- Tener descargado el script de la base de datos

Otros requisitos:

Conexión a internet para poder acceder a todas las funcionalidades del sistema, descargar actualizaciones y trabajar con repositorios en la nube.

Instalación y configuración

1. Abrir el el link de la página web
2. crear cuenta del primer usuario (administrador):

Registrar primer usuario



Nombre

Dui

Correo

Alias

Telefono

Contraseña:

Confirmar Contraseña:

© 2024 D-M SYSTEM. Todos los derechos reservados.

Recomendaciones:

- El nombre debe ser corto, no se solicita el nombre completo.
 - El correo electrónico debe ser real, ya que con este se mandará el código de recuperación de contraseña
 - El teléfono debe tener el formato (2, 6, 7)###-####
 - La contraseña no debe contener información personal (nombre).
 - La contraseña debe incluir mayúsculas, minúsculas, números y caracteres especiales.
 -
 -
3. iniciar sesión con la cuenta creada anteriormente
 4. Aparecerá una pantalla donde debes verificar el correo que añadiste a la cuenta creada, el código será enviado a tu correo :

Iniciar sesión



Código de verificación


○

 Traducir al español



Tu código de verificación es: 454351

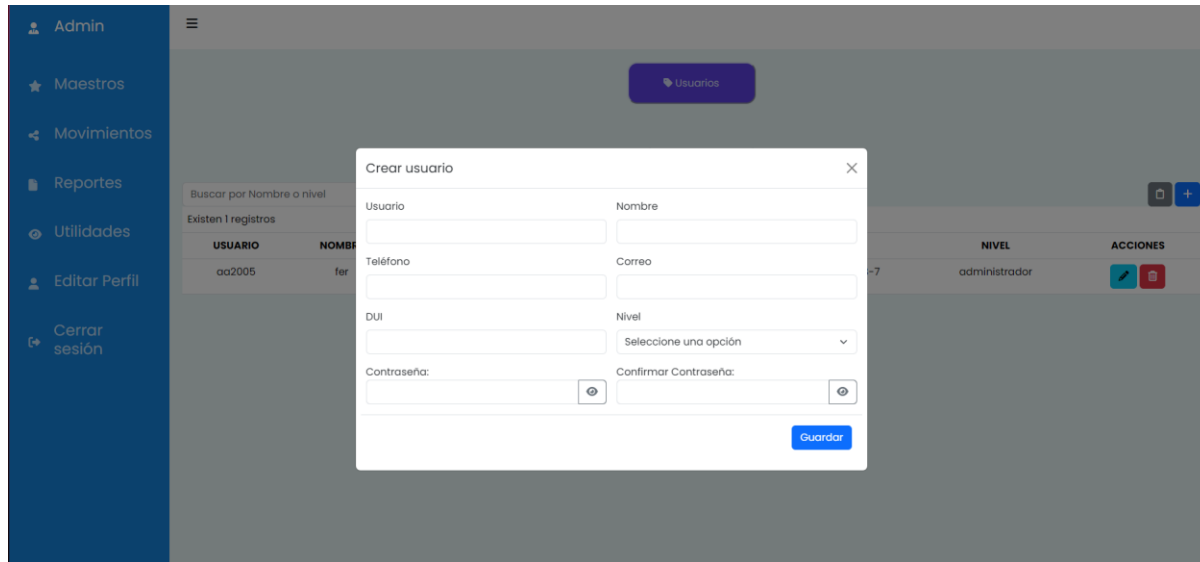
 Responder

 Reenviar



 Respuesta de IA

-
- 5. Si se necesita agregar, modificar o eliminar más usuarios ya sea para administrador, inventario, caja o ventas se puede hacer desde el apartado de utilidades:
-



Admin

Maestros

Movimientos

Reportes

Utilidades

Editar Perfil

Cerrar sesión

Usuarios

Buscar por Nombre o nivel

Existen 1 registros

USUARIO	NOMBRE
aa2005	fer

Crear usuario

Usuario

Nombre

Teléfono

Correo

DUI

Nivel

Seleccione una opción

Contraseña

Confirmar Contraseña

Guardar