Universidad Tecnológica de Durango

Tecnologías de la Información

Programación Estructurada

Informe Técnico

“Arreglos y archivos”

Alumnos:

- Barraza Torres Jesús Daniel
- Ceseñas Rivera Diana Laura

2°A BIS

Docente:

- Ing. Dagoberto Fiscal Gurrola, M.T.I.

Agosto 2025

Tabla de contenido

Ceseñas Rivera Diana Laura
Barraza Torres Jesús Daniel

Propósitos

General

- Desarrollar aplicaciones de software a través de técnicas de programación estructurada para implementar soluciones computacionales.

Específico

- Implementar diversas estructuras de datos como los arreglos, así como el manejo de archivos para gestionar la información.

Actividades del reporte: Farmacia en Base de datos

Explicación (Objetivo, alcance y limitaciones)

La base de datos nos permite respaldar y mostrar datos en tiempo real el cual pueden acceder ya sea de manera local como de manera remota mediante un servidor. Este servidor es conocido como un servidor MySQL, el cual puede ser accedido mediante el uso de metodologías de red (quiere decir que compone otros conceptos como la dirección IP).

Mediante este proyecto buscamos la implementación de base de datos MySQL en un proyecto el cual llama a un caso de la vida real.

Contexto (Descripción y justificación)

Durante el análisis del entorno de trabajo de una farmacia local, se identificaron múltiples fallas derivadas de la ausencia de control sistemático en el inventario. Los propietarios no llevaban registros precisos de cuántos medicamentos tenían en existencia, y aunque anotaban las ventas en una libreta, no existía un vínculo entre estas transacciones y la actualización automática del stock. Como resultado, medicamentos caducados permanecían en los estantes sin ser detectados a tiempo, otros se agotaban sin previo aviso, y el ingreso de nuevos productos no se sumaba adecuadamente al inventario existente.

Ante esta situación, se desarrolló un sistema de gestión de inventario que permite registrar entradas y salidas de productos, controlar fechas de caducidad y mantener una visión integral del estado actual de los medicamentos disponibles. Esta solución no solo mejora la eficiencia en las operaciones diarias, sino que también contribuye a la calidad del servicio y la sostenibilidad del negocio. Justamente, este desarrollo refleja la necesidad de digitalizar procesos críticos en sectores donde la precisión, el control y el seguimiento marcan la diferencia.

Código

Menú de inicio de sesión y BD

```python
def main():
    opcion = True
    while opcion:
        funciones.BorrarPantalla()
        opcion=funciones.menu_usuarios()

        if opcion == "1" or opcion=="RESGISTRO":
            funciones.BorrarPantalla()
            print("\n\t\t .:: Registro en el sistema ::.\n")
            email = input("Ingrese su correo electrónico: ").strip()
            contrasena = getpass.getpass("Ingrese su contraseña: ").strip()
            usuario = usuarios.registrar(email, contrasena)
            if usuario:
                print("\n✅ Su registro fue exitoso.")
                menu_principal()
            else:
                print("\n❌ No fue posible registrar su usuario. Intente de nuevo.")

            funciones.EsperarTecla()
        elif opcion == "2" or opcion=="LOGIN":
            funciones.BorrarPantalla()
            print("\n\t\t .:: Inicio de sesion ::.\n")
            email = input("Ingrese su correo electrónico: ").strip().lower()
            contrasena = getpass.getpass("Ingrese su contraseña: ").strip()
            registro = usuarios.iniciar_sesion(email, contrasena)
            if registro:
                menu_principal()
            else:
                print(f"\n\tCorreo o contraseña incorrectas")
                print("\n\tIntentelo de nuevo")
            funciones.EsperarTecla()

        elif opcion == "3" or opcion=="SALIR":
            print("\nGracias por usar el sistema. Hasta pronto.")
            opcion=False
            funciones.EsperarTecla()
        else:
            print("Opcion no valida")
            opcion=True
            funciones.EsperarTecla()
```

Ilustración 1. Función de Main

En esta ilustración podemos observar el menú interactivo de Main, el cual contine las opciones de registrar, iniciar sesión y salir, este utiliza datos de una tabla aparte. Esta función se encuentra en el módulo de Main.

```
try:
    conexion=mysql.connector.connect(
        host="127.0.0.1",
        user="root",
        password="",
        database="db_farmacia",
        use_pure=True
    )
    confirm=True
    print("Conexión exitosa")
    cursor=conexion.cursor(buffered=True)
except mysql.connector.errors.ProgrammingError:
    try:
        conexion=mysql.connector.connect(
            host="127.0.0.1",
            user="root",
            password="",
            use_pure=True
        )
        cursor=conexion.cursor(buffered=True)
    except:
        print(f"En este momento no posible comunicarse con el sistema, intentelo mas tarde ...")
        confirm=False
    else:
        cursor.execute("CREATE DATABASE db_farmacia")
        cursor.execute("USE db_farmacia")
        print("Se creó la base de datos")
        confirm=True
except:
    print(f"En este momento no posible comunicarse con el sistema, intentelo mas tarde ...")
    confirm=False
```

Ilustración 2. Conexion DB

En esta ilustración podemos observar la conexión a la base de datos. Si no se encuentra la base de datos entonces se ejecuta una creación de la base de datos y comienza desde allí. Esta parte se encuentra en el módulo de BD

```
if confirm:
    cursor.execute(
        """
    CREATE TABLE IF NOT EXISTS medicamentos (
        ID INT(10) AUTO_INCREMENT PRIMARY KEY,
        Nombre_marca VARCHAR(50),
        Compuesto_activo VARCHAR(200),
        Caducidad date,
        Concentracion INT(10),
        Presentacion VARCHAR(25),
        Cantidad INT,
        Precio DECIMAL(8,2),
        Laboratorio VARCHAR(25)
    ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Ilustración 4. Creación de tablas

En esta ilustración podemos observar que, si la conexión fue exitosa y si no se encuentra la tabla dentro de la base de datos, crea la tabla correspondiente. Esta parte se encuentra en el módulo de DB (cabe aclarar que esta captura muestra solo 1 de las tablas correspondientes, sin embargo, el resto de las tablas usan el mismo formato y se repite)

```
def registrar(email,contrasena):
    try:
        fecha=datetime.datetime.now()
        contrasena=hashlib.sha256(contrasena.encode()).hexdigest()
        cursor.execute(
            "INSERT INTO usuarios (id_usuario,  email, contrasena) VALUES (NULL,%s, %s)", (email, contrasena)
        )
        conexion.commit()
        return True
    except:
        print("Error al registrar")
        return False
```

Ilustración 3. Función de registrar

En esta ilustración podemos observar el registro de correos electrónicos y contraseña para poder iniciar sesión después. También registra la fecha de entrada y la ID de usuario Esta función se encuentra en el módulo de usuarios.

```python
def iniciar_sesion(email,contrasena):
    try:
        contrasena=hash_password(contrasena)
        cursor.execute(
            "select * from usuarios where email=%s and contrasena=%s", (email, contrasena)
        )
        registros=cursor.fetchone()
        if registros:
            return registros
        else:
            return None
    except:
        print("Error iniciar sesion")
        return None
```

Ilustración 6. Módulo de inicio de sesión

En esta ilustración podemos observar el módulo de inicio de sesión y contraseña, el cual juntos pueden iniciar sesión y entrar al siguiente menú. Esta función se encuentra dentro del módulo de Usuarios.

```python
def hash_password(contrasena):
    return hashlib.sha256(contrasena.encode()).hexdigest()
```

Ilustración 5. Encriptación

En esta ilustración podemos observar el módulo de hash_password, el cual encripta con un código sha256 escrito en hexadecimal para asegurar todo en la base de datos. Esta función se encuentra dentro del módulo de Usuarios

Menú principal y de exportación

```
def menu_principal():
    while True:
        funciones.BorrarPantalla()
        print("\n\t\t .:: Menú Principal ::.\n")
        print("1. Menú de venta")
        print("2. Menú de medicamentos")
        print("3. Exportar tabla de medicamentos")
        print("4. Importar tabla de medicamentos")
        print("5. Exportar ventas")
        print("6. Importar ventas")
        print("7. Exportar detalle de ventas")
        print("8. Importar detalle de ventas")
        print("9. Salir al menú de inicio")

        opcion = input("\nSeleccione una opción: ").strip()
```

Ilustración 7. Menú principal - Texto

En esta ilustración podemos ver las diversas opciones para impresión dentro de los archivos, estas opciones cubren los otros menús y la capacidad de exportar/importar los datos. Este archivo se encuentra dentro de main.

Ceseñas Rivera Diana Laura
Barraza Torres Jesús Daniel

```
if opcion == "1":
    funciones.BorrarPantalla()
    ventas.menu_ventas()
elif opcion == "2":
    funciones.BorrarPantalla()
    menu_medicamentos()
elif opcion == "3":
    funciones.BorrarPantalla()
    BD.exportar_tabla_med()
    funciones.EsperarTecla()
elif opcion == "4":
    funciones.BorrarPantalla()
    BD.importar_tabla_med()
    funciones.EsperarTecla()
elif opcion == "5":
    funciones.BorrarPantalla()
    BD.exportar_tabla_ventas()
    funciones.EsperarTecla()
elif opcion == "6":
    funciones.BorrarPantalla()
    BD.importar_tabla_ventas()
    funciones.EsperarTecla()
elif opcion == "7":
    funciones.BorrarPantalla()
    BD.exportar_tabla_detven()
    funciones.EsperarTecla()
elif opcion == "8":
    funciones.BorrarPantalla()
    BD.importar_tabla_detven()
    funciones.EsperarTecla()
elif opcion == "9":
    funciones.BorrarPantalla()
    print("\n🔙 Regresando al menú de inicio...")
    funciones.EsperarTecla()
    break
else:
    print("\n⚠️ Opción inválida. Intente de nuevo.")
```

Ilustración 8. Menú principal - Interacción

En esta ilustración podemos observar la siguiente parte de la función anterior, todas estas funciones pueden ser accedidas de la opción anterior. Este archivo se encuentra en “Main” y llama datos desde “Main” y desde BD.

```python
def importar_tabla_med():
    funciones.BorrarPantalla()
    print("\n\t\t .:: Importar tabla ::. \n\t\t \u26A0 AVISO \u26A0")
    opc=input(f"Para poder importar los datos de la tabla CSV es importante localizarlo en la 
    if opc=="Si":
        nom=input("Ingrese el nombre del archivo (procura que no tenga espacios, escribe bien,
        try:
            with open(nom, mode='r', encoding='utf-8') as archivo:
                lector_csv = csv.reader(archivo)
                fila1=next(lector_csv)
                nuevos = 0
                for fila in lector_csv:
                    sql = """INSERT INTO medicamentos
                    (Nombre_marca, Compuesto_activo, Caducidad, Concentracion, Presentacion, C
                    VALUES (%s, %s, %s, %s, %s, %s, %s, %s)
                    """
                    cursor.execute(sql, fila)
                    nuevos+=1
            conexion.commit()
            print(f"\n\t\t ::: {nuevos} medicamentos insertados correctamente :::")
        except:
            print("\n\t... Hubo un problema con el programa, inténtalo de nuevo ...")
    else:
        print("\n\t ...Operación abortada con éxito ...")

def exportar_tabla_med():
    funciones.BorrarPantalla()
    print("\n\t\t .:: Exportar tabla ::.")
    opc=input(f"La tabla se exportará en un archivo de tipo csv \n\n ¿Desea exportar la tabla 
    if opc=="Si":
        cursor.execute("SELECT * FROM medicamentos")
        dat=cursor.fetchall()
        nom=input("Ingrese el nombre de la tabla que se exportará: ").lower().strip() + ".csv"
        try:
            with open(nom, mode="w", newline="", encoding="utf-8") as archivo:
                escritor_csv = csv.writer(archivo)
                encabezados = [i[0] for i in cursor.description][1:]
                escritor_csv.writerow(encabezados)
                for fila in dat:
                    escritor_csv.writerow(fila[1:])
                print("\n\t\t :::Tabla de medicamentos exportada:::")
        except:
            print("\n\t... Hubo un problema con el programa, inténtalo de nuevo ...")
    else:
        print("\n\t ...Operación abortada con éxito ...")
```

Ilustración 9. Importar y exportar tablas

En esta ilustración podemos observar un archivo de importación y exportación de datos. Mediante un archivo .csv podemos guardar los datos de manera física en caso de un ataque directo al servidor. Esta parte se encuentra en el archivo BD (cabe aclarar que para las demás tablas el código se repite, por ende, es redundante mencionarlo)

Menú de Ventas

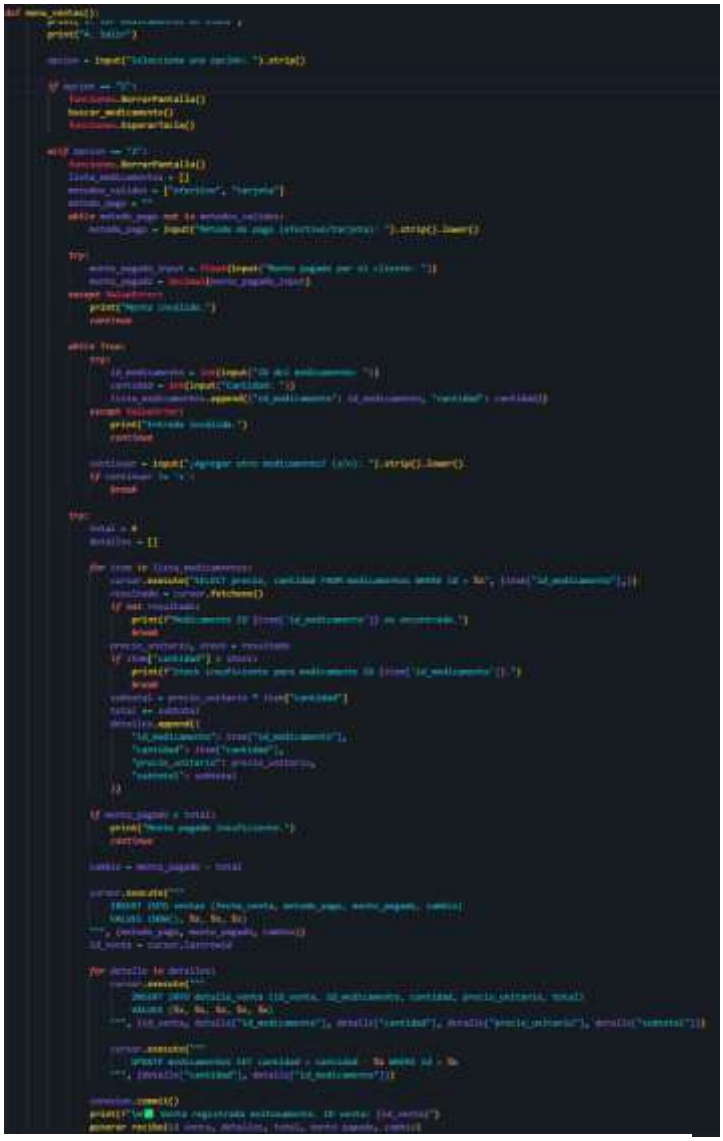

Ilustración 10. Menú de ventas

En esta ilustración se encuentra el megacódigo del menú de ventas, este no solamente muestra las opciones y te permite interactuar con ellas, si no que contiene el procedimiento entero de la opción 2 (capturar venta) y la opción 3 (mostrar stock). Esta megafunción se encuentra en el módulo de ventas.

```
def medicamentos_stock(): #Esta funcion le permite al vendedor ver la lista de todos los medicamentos disponibles
    cursor.execute("SELECT ID, Nombre_marca, Precio, Cantidad FROM medicamentos;")
    medicamentos = cursor.fetchall()
    return medicamentos
meds = medicamentos_stock()
print(f"Medicamentos en existencia: {len(meds)}")
for med in meds:
    print(f"{'ID':<10}{'Nombre':<15}{'Precio':<15}{'Cantidad':<15}")
    print(f"-"*60)
    print(f"{med[0]:<10} {med[1]:<15}{med[2]:<15}{med[3]:<15}")
    print(f"-"*60)
```

Ilustración 12. Función de stock

En esta ilustración se puede mostrar el medicamento que se encuentra junto a su cantidad disponible y su precio al público en forma de tabla. Esta función se encuentra en el módulo de Ventas

```
def buscar_medicamento(): #Esta funcion permite la busqueda del medicamento por nombre
      nombre=input("Dame el nombre del medicamento a buscar: ").upper().strip()
      sql="select id, nombre_marca, precio, cantidad from medicamentos where nombre_marca=%s"
      val=(nombre,)
      cursor.execute(sql,val)
      coincidencia=cursor.fetchall()
      if coincidencia:
        print(f"\n\tMostrar los medicamentos")
        print(f"-"*80)
        for med in coincidencia:
          print(f"{'ID':<10}{'Nombre':<15}{'Precio':<15}{'Cantidad':<15}")
          print(f"-"*60)
          print(f"{med[0]:<10}{med[1]:<15}{med[2]:<15}{med[3]:<15}")
          print(f"-"*60)
      else:
        print("\n\t .:: No hay medicamentos que coincidan con ese nombre::. ")
```

Ilustración 11. Función de buscar

En esta ilustración se puede mostrar la función que te sirve encontrar el medicamente por nombre comercial, mostrando cuánto de stock hay y cuál es el precio del producto. Esta función se encuentra en el módulo de ventas.

```python
def registrar_venta(metodo_pago, monto_pagado, lista_medicamentos):
        total = 0
        detalles = []

        # Calcular total y preparar detalles
        for item in lista_medicamentos:
            cursor.execute("SELECT precio, cantidad FROM medicamentos WHERE id = %s", (item["id_medicamento"],))
            resultado = cursor.fetchone()
            if not resultado:
                print(f"Medicamento ID {item['id_medicamento']} no encontrado.")
                return False
            precio_unitario, stock = resultado
            if item["cantidad"] > stock:
                print(f"Stock insuficiente para medicamento ID {item['id_medicamento']}.")
                return False
            subtotal = precio_unitario * item["cantidad"]
            total += subtotal
            detalles.append({
                "id_medicamento": item["id_medicamento"],
                "cantidad": item["cantidad"],
                "precio_unitario": precio_unitario,
                "subtotal": subtotal
            })

        # Validar pago
        if monto_pagado < total:
            print("Monto pagado insuficiente.")
            return False
        cambio = monto_pagado - total

        # Insertar en ventas
        cursor.execute("""
            INSERT INTO ventas (fecha_venta, metodo_pago, monto_pagado, cambio)
            VALUES (NOW(), %s, %s, %s)
        """, (metodo_pago, monto_pagado, cambio))
        id_venta = cursor.lastrowid

        # Insertar en detalle_venta y actualizar inventario
        for detalle in detalles:
            cursor.execute("""
                INSERT INTO detalle_venta (id_venta, id_medicamento, cantidad, precio_unitario, subtotal)
                VALUES (%s, %s, %s, %s, %s)
            """, (id_venta, detalle["id_medicamento"], detalle["cantidad"], detalle["precio_unitario"], detalle["subtotal"]))

            cursor.execute("""
                UPDATE medicamentos SET cantidad = cantidad - %s WHERE id = %s
            """, (detalle["cantidad"], detalle["id_medicamento"]))

        conexion.commit()
        print(f"Venta registrada exitosamente. ID venta: {id_venta}, Cambio: ${cambio:.2f}")
        return True

    except Exception as e:
        conexion.rollback()
        print("Error al registrar la venta1:", e)
        return False
```

Ilustración 13. Función de registrar la venta

En esta ilustración podemos ver el módulo de registro de ventas, el cual pregunta por medicamento y la cantidad de compra que, calculando el precio, este nos suelta el resultado directo. Esta función se encuentra en el menú de Ventas.

```python
def capturar_venta():
    lista_medicamentos = []
    metodos_validos = ["efectivo", "tarjeta"]
    metodo_pago = ""
    while metodo_pago not in metodos_validos:
        metodo_pago = input("Método de pago (efectivo/tarjeta): ").strip().lower()

    monto_pagado_input = float(input("Monto pagado por el cliente: "))
    Decimal(monto_pagado_input)


    while True:
        try:
            id_medicamento = int(input("ID del medicamento: "))
            cantidad = int(input("Cantidad: "))
            lista_medicamentos.append({"id_medicamento": id_medicamento, "cantidad": cantidad})
        except ValueError:
            print("Entrada inválida.")
            continue

        continuar = input("¿Agregar otro medicamento? (s/n): ").strip().lower()
        if continuar != 's':
            break
```

Ilustración 15. Módulo de captura de venta

En esta ilustración podemos observar el módulo donde se tiene que capturar la venta, dependiendo de cómo pagaste y que compraste se va a registrar la venta. Este módulo se encuentra en el módulo de ventas.

```python
def generar_recibo(id_venta, detalles, total, monto_pagado, cambio):
    print("\n=== Recibo de Venta ===")
    print(f"ID de Venta: {id_venta}")
    print(f"{'ID Medicamento':<15}{'Cantidad':<10}{'Precio':<10}{'Subtotal':<10}")
    print("-" * 50)
    for detalle in detalles:
        print(f"{detalle['id_medicamento']:<15}{detalle['cantidad']:<10}{detalle['precio_unitario']:<10.2f}{detalle['subtotal']:<10.2f}")
    print("-" * 50)
    print(f"Total: ${total:.2f}")
    print(f"Monto pagado: ${monto_pagado:.2f}")
    print(f"Cambio: ${cambio:.2f}")
    print("-" * 50)
```

Ilustración 14. Módulo del recibo

En esta ilustración podemos observar el módulo que te imprime el recibo en forma de tabla basado en normativas económicas y los datos de los anteriores 2 módulos. Esta función se encuentra en el módulo de ventas.

Menú de inventario

Ilustración 16. Menú de inventario - Impresión

En esta ilustración podemos ver la función de menú el cual muestra las opciones de modificación de la lista de medicamentos. Esta función se encuentra en el módulo de funciones y es llamado en main.

```python
def menu_medicamentos():
    opcion = ""

    while True:
        funciones.BorrarPantalla()
        opcion = funciones.menu_inventario()

        match opcion:
            case "1":
                Mod.AgregarMedicina()
                funciones.EsperarTecla()
            case "2":
                Mod.BorrarMedicina()
                funciones.EsperarTecla()
            case "3":
                Mod.MostrarMedicina()
                funciones.EsperarTecla()
            case "4":
                Mod.BuscarMedicina()
                funciones.EsperarTecla()
            case "5":
                Mod.FechaCaducidad()
                funciones.EsperarTecla()
            case "6":
                Mod.ModificarMedicina()
                funciones.EsperarTecla()
            case "7":
                funciones.BorrarPantalla()
                print("\t \U0001F6AA ...Regresando al menú de inicio...")
                funciones.EsperarTecla()
                menu_principal()
            case _:
                funciones.BorrarPantalla()
                print("\tOpcion invalida, por favor vuelva a intentarlo\n")
                funciones.EsperarTecla()
```

Ilustración 17. Menú de inventario - Interacción

En esta ilustración podemos encontrar la interacción de menú para el inventario; Todas estas opciones son para modificar los diversos datos de la tabla de inventario excepto el de salir, que utiliza el menú principal. Esta función se encuentra en main.

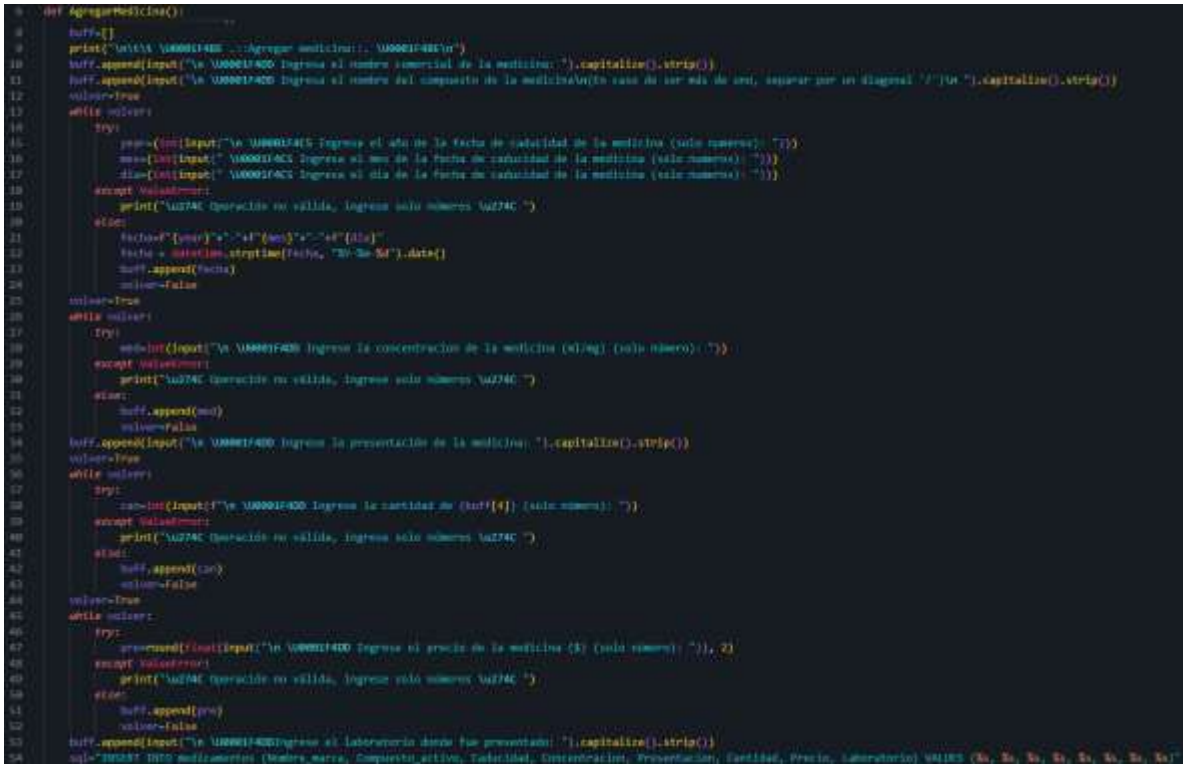

Ilustración 18. Agregar medicamento

En esta ilustración podemos observar la primera función, la cual es la función que agrega un medicamento; pidiendo el nombre comercial, nombre del activo, fecha de caducidad, concentración del medicamento, la presentación, la cantidad de la presentación, el precio y el laboratorio producido. Esta función se encuentra en el módulo de Mod.

```
def BorrarMedicina():
    funciones.BorrarPantalla()
    print("\n\t\t \U0001F4DB .::Borrar registros del medicamento::. \U0001F4DB")
    cursor.execute(f"SELECT Nombre_marca FROM medicamentos")
    dat1=cursor.fetchall()
    if len(dat1) <= 0:
        print("\n\t\t \u26A0 ...No hay medicamentos en la lista... \u26A0")
    else:
        dat1.sort()
        print(dat1)
        rep=True
        while rep:
            nom=input("\n \U0001F50D Ingrese el nombre comercial del medicamento a borrar: ").capitalize().strip()
            cursor.execute(f"SELECT * FROM medicamentos WHERE Nombre_marca = '{nom}'")
            dat=cursor.fetchall()
            nomb=(nom, )
            if nomb not in dat:
                print("\n\t\t \u26A0Este medicamento no se encuentra en la lista\u26A0")
                rep=True
            else:
                rep=False
        for pos in dat:
            if nom in pos:
                print("\n\t \U0001F4C2 Se encontró un medicamento \U0001F50D ")
                opc=input(f"\n¿Desea borrar el registro del medicamento {nom} {pos[4]}? (SI/No): ").capitalize().strip()
                if opc=="SI":
                    print(f"\n\t \U0001F4DB Se borró el medicamento {nom} {pos[4]} \U0001F4DB")
                    cursor.execute(f"DELETE FROM medicamentos WHERE ID = '{pos[0]}'")
                    conexion.commit()
        print("\n\n\t\t \u2705 :::La operación se ha realizado con éxito::: \u2705")
```

Ilustración 20. Borrar medicamento

En esta ilustración podemos observar la segunda función, siendo este el que te permite borrar un medicamento (que puede ser usado por si un medicamento queda en desuso). Esta función se encuentra en el módulo de Mod.

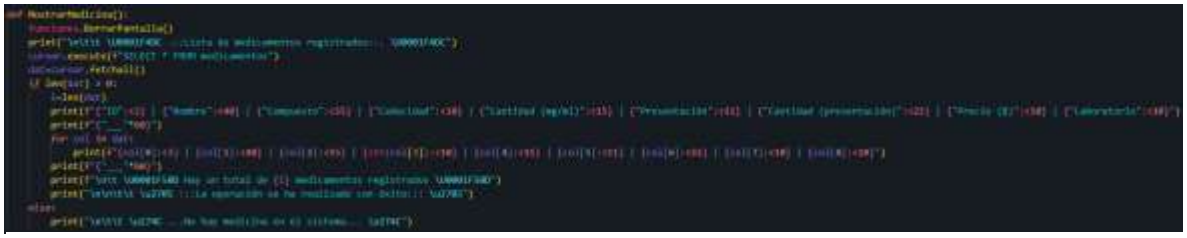

Ilustración 19. Mostrar medicamento

En esta ilustración podemos ver la tercera función, siendo este el que, llamando a todos los datos de la base de datos, muestra todos los datos en el formato de tabla. Esta función se encuentra en el módulo de Mod.

Ceseñas Rivera Diana Laura

Barraza Torres Jesús Daniel

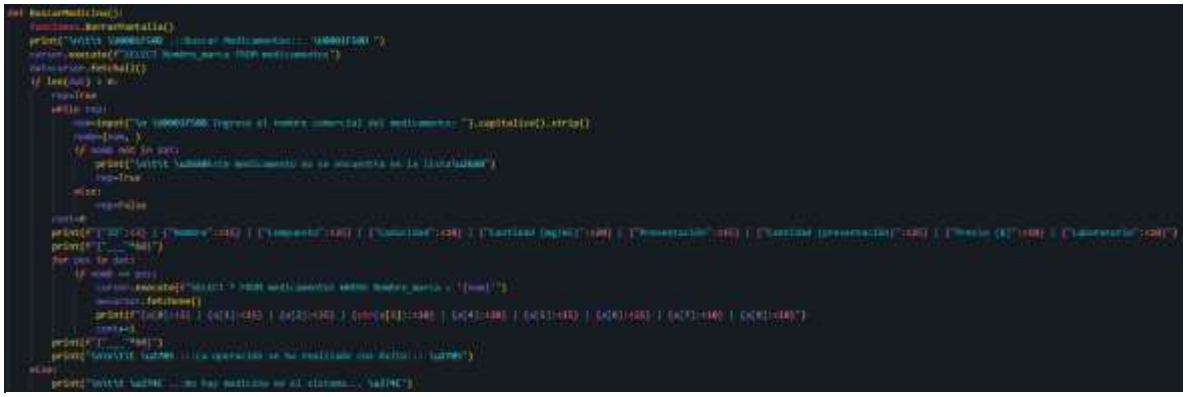

Ilustración 22. Buscar medicamento

En esta ilustración podemos observar la función de buscar la medicina, la cual, mediante el nombre comercial, puedes buscar los datos de cualquier medicamento. Esta función se encuentra en el Módulo de Mod.

```
def FechaCaducidad():
    funciones.BorrarPantalla()
    print("\n\t\t \U0001F4C5 :::Lista de fecha de medicamentos::: \U0001F4C5")
    cursor.execute(f"SELECT Nombre_marca, Caducidad FROM medicamentos")
    dat=cursor.fetchall()
    if len(dat) > 0:
        fecha=datetime.now().date()
        dat.sort(key = lambda x: x[1])
        cursor.execute(f"SELECT Caducidad FROM medicamentos where Caducidad < '{str(fecha)}'")
        lista1=cursor.fetchall()
        if len(lista1) > 0:
            print("\n\t \u203C :::ESTE MEDICAMENTO ESTÁ CADUCADO::: \u203C")
            for a, e in dat:
                if e < fecha:
                    e=str(e)
                    cursor.execute(f"SELECT Nombre_marca, Caducidad FROM medicamentos where Caducidad = '{e}' AND Nombre_marca = '{a}'")
                    enlista1=cursor.fetchone()
                    print(f"{enlista1[0]:<35} {str(enlista1[1]):<10}")
                else:
                    break
        cursor.execute(f"SELECT Caducidad FROM medicamentos where Caducidad > '{str(fecha)}'")
        lista2=cursor.fetchall()
        if len(lista2) > 0:
            print("\n\t \U0001F4C5 ...Este medicamento está todavía usable...")
            for a, e in dat:
                if e > fecha:
                    e=str(e)
                    cursor.execute(f"SELECT Nombre_marca, Caducidad FROM medicamentos where Caducidad = '{e}' AND Nombre_marca = '{a}'")
                    enlista2=cursor.fetchone()
                    print(f"{enlista2[0]:<35} {str(enlista2[1]):<10}")
        print("\n\t \u2705 :::La operación se ha realizado con éxito::: \u2705")
    else:
        print("\n\t\t \u274C ...No hay medicina en el sistema... \u274C")
```

Ilustración 21. Mostrar la fecha de caducidad

En esta ilustración podemos observar la función de fecha de caducidad, la cual muestra en una lista ordenada cuales medicamentos están caducados y cuales medicamentos todavía están en uso. Esta función se encuentra en el módulo de Mod.

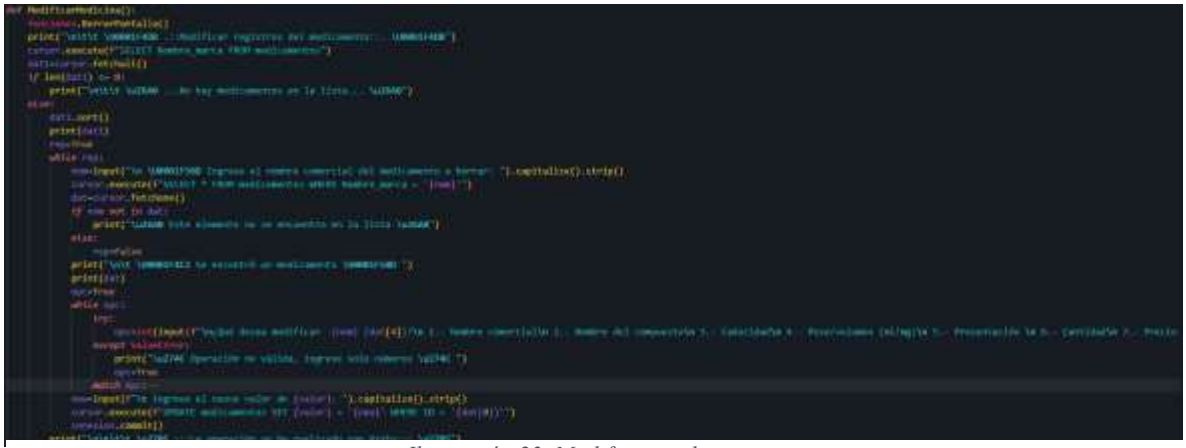

Ilustración 23. Modificar medicamento

En esta ilustración podemos observar la última función de la modificación de medicamentos, siendo este el de modificar un medicamento. Mediante un submenú podemos seleccionar, desde un medicamento, qué modificar en este. Esta función se encuentra en el módulo de Mod.

Diagrama entidad-relación

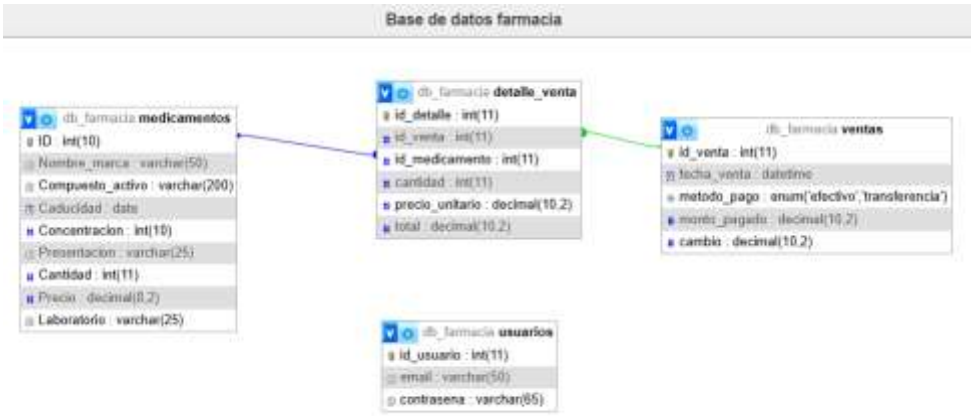

Ilustración 24. Diagrama entidad-relación

En esta ilustración podemos ver la relación de las tablas dentro de la base de datos, los cuales se relacionan por IDs para el posterior registro de la venta. Este archivo aisla a los usuarios debido a su falta de verificación obligatoria.

Conclusiones

Este proyecto ha sido uno de los más complejos que realicé, y la verdad no me siento tan satisfactorio como me lo esperaba (principalmente porque la presentación va a ser lo que defina si pierdo o continúo), pero gratificante, a fin de cuentas.

Quisiera tomarme un descanso de todo el rollo de lo que es la base de datos por el momento, ya que este tema de configuración de datos es sumamente agotador y con pocos resultados mostrados (al menos porque hicimos las pruebas en un servidor local).

Siento que con un poco de preparación podré sobrellevar todo esto, pero por lo pronto este proyecto deja cansado moralmente a las personas; si usted está leyendo esto quiero avisar que estoy cansado de este tipo de proyectos, de tener que superarme cada semana solamente porque no se encuentra conforme; estoy cansado.

Barraza Torres Jesús Daniel

Este proyecto fue retador en varios sentidos, desde lograr que la conexión a SQL funcionara correctamente, hasta construir los módulos que hicieran que todo fuera interactivo y funcional. Tuvimos que trabajar con varias tablas, organizar bien la información y asegurarnos de que cada parte del sistema respondiera como debía. Usamos listas, estructuras de control y validaciones para que el menú fuera intuitivo y permitiera navegar entre opciones sin errores.

Uno de los mayores aprendizajes fue lograr que las ideas de dos personas, con formas de pensar muy distintas, se integraran en un solo proyecto. Al principio fue complicado ponernos de acuerdo, pero con comunicación, paciencia y ajustes constantes, logramos que el sistema reflejara lo mejor de ambas perspectivas. El trabajo en equipo fue clave: cada quien aportó desde su experiencia y aprendimos a ceder, proponer y construir juntos.

Ver el menú funcionando y comprobar que las tablas en SQL se conectaban bien fue súper satisfactorio. Fue como ver cómo todo el esfuerzo cobraba vida. Más allá del código, este proyecto nos enseñó a organizarnos, a resolver problemas en tiempo real y a mantenernos enfocados en el objetivo.

En resumen, fue una experiencia completa: técnica, creativa y colaborativa. Nos llevamos no solo un sistema funcional de inventario, sino también herramientas para trabajar mejor en equipo y enfrentar retos con más seguridad.

Ceseñas Rivera Diana Laura

Bibliografía

- Jaime. (n.d.). Diccionarios y listas en Python. Tutorial Python. Retrieved junio 26, 2025, from https://tutorialpython.com/listas-en-python/
- Smith, Y. (2023). Cómo agregar, modificar y eliminar elementos de listas en Python. Platzi. Retrieved junio 27, 2025, from https://platzi.com/tutoriales/4227-python-fundamentos/24292-omo-agregar-modificar-y-eliminar-elementos-de-listas-en-python/
- Matthes, E. (2019). Python crash course: A hands-on, project-based introduction to programming (2nd ed.). No Starch Press.
- Beazley, D. M. (2021). Python cookbook: Recipes for mastering Python 3 (3rd ed.). O'Reilly Media.
- Modificar un archivo csv en python. (2018, Julio 20). Stack Overflow en español. Retrieved August 12, 2025, from https://es.stackoverflow.com/questions/182905/modificar-un-archivo-csv-en-python
- Python MySQL. (n.d.). W3Schools. Retrieved August 12, 2025, from https://www.w3schools.com/python/python_mysql_getstarted.asp