Bases de datos I – Trabajo Obligatorio

Cafés Marloy

Joaquín Arzuaga

Martina Caetano

Samuel Casallas

Introducción

La empresa "Cafés Marloy" requiere un sistema de gestión integral para administrar sus máquinas expendedoras de café, clientes, insumos y mantenimientos. Este proyecto tiene como objetivo diseñar e implementar una base de datos relacional junto con python que permite gestionar, automatizar y generar reportes.

El sistema está enfocado al usuario administrador y clientes, donde se pueda tanto como por un lado modificar o consultar.

Tecnologías utilizadas:

Frontend:

Frameworks y Librerías principales:

- React (react, react-dom): Biblioteca principal para construir la interfaz de usuario.
- React Router DOM (react-router-dom): Enrutamiento de páginas en aplicaciones React.
- Axios: Cliente HTTP para hacer peticiones a la API.
- jwt-decode: Decodificador de tokens JWT (aunque el backend no usa JWT, está instalada en el frontend). falta implementar

Herramientas de desarrollo:

- Vite: Herramienta de build y desarrollo rápido para proyectos frontend modernos.
- ESLint y plugins: Linter para mantener la calidad del código.
- @vitejs/plugin-react: Integración de React con Vite.
- @types/react, @types/react-dom: Tipos para desarrollo con TypeScript (aunque el código parece estar en JS).

Backend:

Frameworks y Librerías principales:

- Flask: Framework web principal para construir la API.
- flask-cors: Soporte para CORS (Cross-Origin Resource Sharing).
- mysql-connector-python: Conector para interactuar con bases de datos MySQL.
- bcrypt: Para el hash y verificación de contraseñas.

Detalles Negocio Marloy:

Cafés Marloy es una empresa dedicada al alquiler de máquinas expendedoras de café para empresas y particulares, donde los clientes pueden rentar las máquinas disponibles. La compañía no solo suministra los insumos necesarios para su funcionamiento, sino que también gestiona el mantenimiento técnico de los equipos. Las ganancias obtenidas por las ventas se distribuyen entre la empresa y el arrendatario según un porcentaje previamente establecido, asegurando así un modelo de negocio sostenible y beneficioso para ambas partes.

Requisitos:

1) Se debe poder logear y registrar 2 tipos de usuarios con diferentes privilegios

Usuario administrador: tiene el poder de hacer consultas, modificar, borrar y añadir registros a la lista de proveedores, de técnicos y de máquinas disponibles para alquilar.

Usuario cliente: puede alquilar una máquina y consultar los costos y ganancias de las máquinas que tiene alquiladas

1.1) Requisitos Usuario cliente:

- Debe poder visualizar las máquinas que tiene contratadas
- Debe poder elegir una máquina de un catálogo de máquinas disponibles y alquilarla.
- Debe poder visualizar el gasto mensual que sus máquinas generan, tanto individualmente como en conjunto
- Debe poder visualizar las ganancias que generan sus máquinas tanto individualmente como en conjunto
- Debe poder solicitar mantenimiento de una máquina

1.2) Requisitos Usuario Administrador:

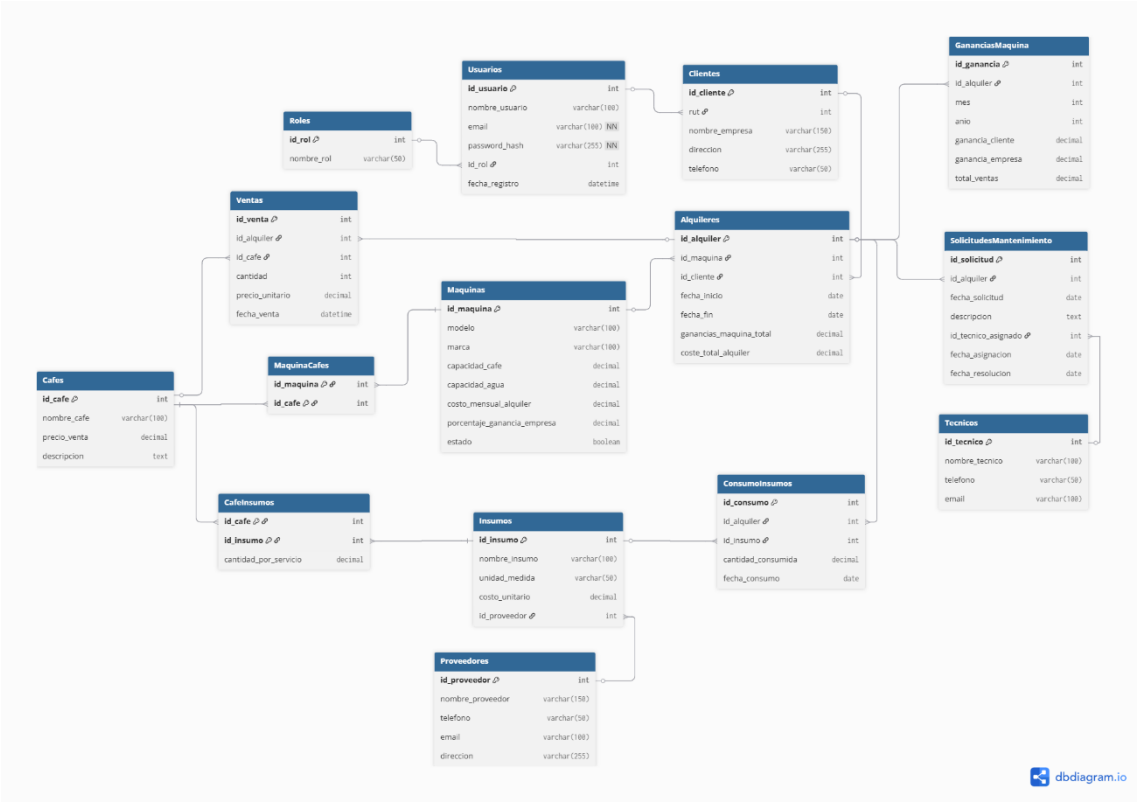
- Debe poder modificar una lista de proveedores (añadir, eliminar, modificar)
- Debe poder modificar la lista de máquinas que están disponibles para alquilar (añadir, eliminar, modificar)
- Debe poder modificar una lista de técnicos (añadir, eliminar, modificar)
- Debe poder "atender una solicitud de mantenimiento", debe poder visualizar una lista de pedidos de mantenimiento y asignar un equipo técnico
- Debe poder visualizar una lista de todas las máquinas ordenadas por cliente
- Debe poder visualizar las ganancias generadas, estas se deben poder visualizar de dos formas, cuánto generan las máquinas de un cliente y cuánto genera una máquina individualmente
- Análogamente para los costos mensuales de mantener las máquinas
- Análogamente debe poder visualizar el rendimiento de cada máquina

2) Las contraseñas de los usuarios deben estar encriptadas

3) La app debe incorporar métodos para evitar la inyección de código SQL

5) Se debe poder registrar el consumo de insumos por cada máquina

Diseño base de datos:



Implementación

En esta sección se detalla el desarrollo del sistema, incluyendo la estructura del frontend con sus páginas para administradores y clientes, así como sus funcionalidades. Además, se explica cómo se gestionan los procesos clave, como alquileres, ventas y mantenimientos.

Páginas para Administrador:

GestionClientes.jsx

Muestra una tabla con los datos de los clientes, cada fila de la tabla corresponde a un cliente, cada uno tiene al final 2 botones, uno que permite editar y otro que permite eliminar el cliente

GestionMáquinas.jsx

Muestra una tabla con los datos de cada máquina, cada fila de la tabla corresponde a una máquina, cada una tiene al final 2 botones, uno que permite editar y otro que permite eliminar la máquina. También cuenta con un botón que despliega un formulario que permite agregar una máquina al catálogo.

GestionCafés.jsx

Muestra una tabla con los datos de cada café, cada fila de la tabla corresponde a un café, cada una tiene al final dos botones, uno que permite editar y otro que permite eliminar el café. También cuenta con un botón que despliega un formulario que permite agregar un café al catálogo.

GestionInsumos.jsx

Muestra una tabla con los datos de cada insumo, cada fila de la tabla corresponde a un insumo, cada una tiene al final 2 botones, uno que permite editar y otro que permite eliminar el insumo. También cuenta con un botón que despliega un formulario que permite agregar un insumo al catálogo.

GestionProveedores.jsx

Muestra una tabla con los datos de cada proveedor, cada fila de la tabla corresponde a un proveedor, cada una tiene al final dos botones, uno que permite editar y otro que permite eliminar el proveedor. También cuenta con un botón que despliega un formulario que permite agregar un proveedor la lista

GestionTécnicos.jsx

Muestra una tabla con los datos de cada técnico, cada fila de la tabla corresponde a un técnico, cada una tiene al final 2 botones, uno que permite editar y otro que permite eliminar el técnico. También cuenta con un botón que despliega un formulario que permite agregar un técnico la lista.

GestionVentas.jsx

Permite simular las ventas de una máquina de café alquilada, con el fin de poder calcular correctamente las ganancias que genera la máquina.
Muestra una tabla que contiene los alquileres activos, donde cada fila corresponde a un alquiler, cada una tiene un botón que despliega un formulario en el que se puede simular una venta en la máquina.

GestionMantenimientos.jsx

Muestra una tabla que contiene las solicitudes de mantenimiento de los clientes, cada fila de la tabla corresponde a una solicitud, cada una tiene al final 3 botones, uno que permite editar, uno que permite eliminarla y uno que permite darla como completada.

GestionGanancias.jsx

Permite visualizar las ganancias tanto de la empresa como de los clientes por cada alquiler.
Contiene un formulario en el que se puede ingresar un periodo de tiempo.
Muestra una tabla donde cada fila corresponde a las ganancias generadas por cada máquina alquilada, contiene las ganancias en bruto de la máquina y la cantidad que corresponde al cliente y a Cafés marloy.

GestionCostosInsumos.jsx

Permite visualizar los costos de insumos consumidos por cada alquiler en un periodo de tiempo.

Contiene un formulario en el que se puede ingresar un periodo de tiempo.

Muestra una tabla que contiene los gastos en insumos de cada alquiler en un periodo de tiempo.

Paginas para Cliente:

MisAlquileres.jsx

Muestra los alquileres que tiene activos el cliente logueado, cada alquiler muestra datos relativos a este y contiene un botón que permite hacer una solicitud de mantenimiento, el cual despliega un formulario que permite ingresar una descripción del problema. La página también contiene un botón que permite al cliente alquilar una máquina disponible, este despliega un cuestionario que permite seccionales la máquina a alquilar junto con una fecha de inicio y una de fin.

MisVentas.jsx

Muestra tres items, que corresponden a ganancias totales generadas por el cliente, total generado por ventas de todas las máquinas del cliente y cantidad de unidades vendidas en un periodo de tiempo dado.
Muestra las ganancias de las máquinas alquiladas en un periodo de tiempo de forma individual.
Muestra un historial de ventas en periodo de tiempo.
Muestra un formulario en el que se puede ingresar un periodo de tiempo, al aplicar el filtro se actualizan los datos mostrados en la página según las ventas generadas en dicho periodo.

Mantenimientos.jsx

Muestra las solicitudes de mantenimiento que ha solicitado el cliente.

Cada solicitud muestra su estado, completada y pendiente, junto con los datos relativos a esta, alquiler, máquina, fecha de solicitud, fecha de asignación de técnico, fecha de resolución, técnico asignado y descripción.

Paginas comunes a los dos usuarios:

Dashboard.jsx

Muestra datos personales y un menú de accesos rápidos a las páginas correspondientes, diferenciando entre administrador y cliente según el rol del usuario.

Login.jsx

Presenta un formulario de login, valida las credenciales contra la API y, si son correctas, almacena la sesión del usuario y redirige al Dashboard.

Si las credenciales no coinciden con un usuario registrado se recomienda al usuario que se registre

Presenta un botón “Registrar” que muestra un formulario para registrar el usuario

HASH DE CONTRASEÑAS:

Para manejar el hash de las contraseñas utilizaremos la librería bcrypt, que es un estándar ampliamente utilizado para el almacenamiento seguro de contraseñas.

Al registrar o actualizar una contraseña, el backend no la almacena en texto plano, sino que utiliza las funciones bcrypt.gensalt() y bcrypt.hashpw() para generar un hash seguro para la contraseña y se guarda ese valor en lugar de la contraseña original. Al ingresar la contraseña en el login, el sistema la hashea y busca una coincidencia en los datos almacenados. De coincidir, la autenticación es exitosa.

Función de hasheo de contraseña:

```python
import bcrypt

def hash_password(password):
    password_bytes = password.encode('utf-8')
    salt = bcrypt.gensalt()
    hashed = bcrypt.hashpw(password_bytes, salt)
    return hashed.decode('utf-8')
```

Función de verificación de la contraseña:

```python
import bcrypt

def hash_password(password):
    password_bytes = password.encode('utf-8')
    salt = bcrypt.gensalt()
    hashed = bcrypt.hashpw(password_bytes, salt)
    return hashed.decode('utf-8')
```

PREVENCION DE INYECCIÓN EN EL CÓDIGO SQL:

Para prevenir la inyección de código SQL utilizaremos consultas parametrizadas en todas las operaciones de acceso a la base de datos. En lugar de concatenar directamente los valores de entrada en las sentencias SQL, se emplean placeholders (%s) en las consultas y los valores reales se pasan como parámetros separados a los métodos de ejecución del cursor, de esta forma garantizamos que la base de datos trate los parámetros únicamente como datos, sin interpretarlos como parte de la lógica SQL, lo que mitiga de forma efectiva el riesgo de inyección de código. Adicionalmente, se implementan validaciones de los datos de entrada en los endpoints del backend, reforzando la integridad y seguridad de las operaciones.

ejemplo:

```python
cursor.execute("""
    SELECT v.*, c.nombre_cafe, a.id_cliente, cl.nombre_empresa
    FROM Ventas v
    JOIN Cafes c ON v.id_cafe = c.id_cafe
    JOIN Alquileres a ON v.id_alquiler = a.id_alquiler
    JOIN Clientes cl ON a.id_cliente = cl.id_cliente
    WHERE v.fecha_venta BETWEEN %s AND %s
    ORDER BY v.fecha_venta DESC
""", (fecha_inicio, fecha_fin))
```

en este ejemplo extraído de clientes.py se utiliza el placeholder %s en lugar de insertar directamente el valor de *fecha_fin* y *fecha_inicio* en el string de la consulta. El valor real de *fecha_fin* y *fecha_inicio* se pasa como un parámetro aparte en la tupla, esto nos asegura que el motor de base de datos lo trate como un dato y no como parte de la consulta. De esta forma, aunque el usuario intente ingresar código SQL a través de estos valores.

BACKEND:

Endpoints:

Auth (/auth)

- POST /registro/cliente – Registra un nuevo cliente con sus datos y crea un usuario.
- POST /registro/admin – Registra un nuevo administrador.
- POST /login – Autentica a un usuario con email y contraseña.
- GET /roles – Obtiene todos los roles del sistema.
- GET /usuario/<id_usuario> – Obtiene datos de un usuario por ID.
- PUT /usuario/<id_usuario> – Actualiza datos de un usuario.
- DELETE /usuario/<id_usuario> – Elimina un usuario.

Cafés (/cafe)

- GET / – Lista todos los cafés.
- POST / – Crea un nuevo café.
- GET /<id_cafe> – Obtiene los detalles de un café por ID.
- PUT /<id_cafe> – Actualiza los datos de un café.
- DELETE /<id_cafe> – Elimina un café.
- GET /maquina/<id_maquina> – Obtiene cafés asociados a una máquina.
- POST /maquina/<id_maquina>/cafe/<id_cafe> – Asocia un café a una máquina.
- DELETE /maquina/<id_maquina>/cafe/<id_cafe> – Quita un café de una máquina.

Clientes (/cliente)

- GET /maquinas/disponibles – Lista máquinas disponibles para alquilar.
- POST /alquileres – Registra un nuevo alquiler de máquina.
- GET /id-cliente/<id_usuario> – Obtiene el ID de cliente según el ID de usuario.
- GET /alquileres-cliente/<id_cliente> – Obtiene los alquileres de un cliente.
- POST /solicitudes – Solicita mantenimiento para un alquiler.
- GET /ganancias-maquina/<id_cliente> – Lista ganancias por máquina para un cliente.
- GET /ganancias-totales/<id_cliente> – Total de ganancias para un cliente.
- GET / – Lista todos los clientes.
- POST / – Crea un nuevo cliente.
- PUT /<id_cliente> – Actualiza datos de un cliente.
- DELETE /<id_cliente> – Elimina un cliente.
- GET /alquiler-detalle/<id_alquiler> – Obtiene detalle del alquiler (máquina e insumos).
- GET /roles – Lista los roles disponibles.
- GET /total-mensual-cobrar – Monto total a cobrar por insumos y alquileres en un mes.

Insumos (/insumo)

- GET / – Lista todos los insumos junto con sus proveedores.
- POST / – Crea un nuevo insumo.
- GET /<id_insumo> – Obtiene los datos de un insumo.
- PUT /<id_insumo> – Actualiza un insumo.
- DELETE /<id_insumo> – Elimina un insumo.
- GET /proveedor/<id_proveedor> – Obtiene insumos por proveedor.

Mantenimientos (/mantenimiento)

- GET / – Lista todas las solicitudes de mantenimiento.
- POST / – Crea una nueva solicitud.
- PUT /<id_solicitud> – Actualiza una solicitud (opcional: técnico asignado).
- PUT /<id_solicitud>/completar – Marca una solicitud como completada.
- DELETE /<id_solicitud> – Elimina una solicitud.
- GET /<id_solicitud> – Obtiene datos de una solicitud por ID.
- GET /pendientes – Lista solicitudes pendientes.
- GET /tecnico/<id_tecnico> – Solicitudes asignadas a un técnico.
- GET /top-tecnicos – Ranking de técnicos con más mantenimientos.

Máquinas (/maquina)

- GET / – Lista todas las máquinas.
- POST / – Crea una nueva máquina.
- GET /<id_maquina> – Obtiene datos de una máquina.
- PUT /<id_maquina> – Actualiza los datos de una máquina.
- DELETE /<id_maquina> – Elimina una máquina.
- GET /alquiladas – Lista máquinas alquiladas.
- GET /disponibles – Lista máquinas disponibles.
- GET /cliente/<id_cliente> – Máquinas alquiladas por un cliente.
- GET /con-cafes – Máquinas con nombres de cafés asociados.
- GET /con-cafes-detalle – Máquinas con detalle de cafés.
- PUT /<id_maquina>/cafes – Actualiza lista de cafés asociados a una máquina.

Proveedores (/proveedor)

- GET / – Lista todos los proveedores.
- POST / – Crea un nuevo proveedor.
- GET /<id_proveedor> – Obtiene los datos de un proveedor.
- PUT /<id_proveedor> – Actualiza un proveedor.
- DELETE /<id_proveedor> – Elimina un proveedor.

Técnicos (/tecnico)

- GET / – Lista todos los técnicos.

- POST / – Crea un nuevo técnico.
- GET /<id_tecnico> – Obtiene los datos de un técnico.
- PUT /<id_tecnico> – Actualiza un técnico.
- DELETE /<id_tecnico> – Elimina un técnico.
- GET /disponibles – Técnicos no asignados actualmente.

Ventas (/venta)

- GET /periodo – Obtiene ventas realizadas en un período.
- GET /consumo-insumos – Consumos de insumos por período.
- GET /alquileres-activos – Alquileres aún vigentes.
- GET /estadisticas – Estadísticas generales de ventas.
- GET /costos-rendimiento – Costos por insumos y rendimiento por máquina.
- GET /ganancias – Ganancias detalladas por máquina y totales.
- GET /cafes-disponibles/<id_alquiler> – Cafés disponibles para un alquiler.
- POST / – Registra una venta y el consumo de insumos correspondiente.
- GET /costos-insumos – Costos totales de insumos por mes y alquiler.