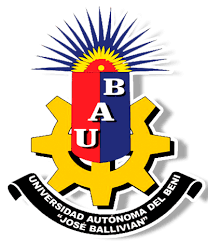
UNIVERSIDAD AUTÓNOMA DEL BENI

“JOSÉ BALLIVIÁN”

FACULTAD DE INGENIERÍA Y TECNOLOGÍA

CARRERA DE INGENIERÍA DE SISTEMAS

**Sistema de Venta y Control de Combustible para la Ciudad de Trinidad**

**Docente:Ing.Hermes Rodriguez Rivero**

**Materia: Tecnologia Emergentes**

**Semestre: 7mo**

**Integrantes:** Yammil Zarraga Pérez

Diego Alberto Montero Mamio

Mauricio Soria Patiño

**Trinidad,09 de Julio de 2025**

**1. Resumen del Proyecto**

Este informe detalla el análisis del código proporcionado para un sistema de venta y control de combustible, desarrollado utilizando Flask y SQL Server 2014.

El proyecto es una aplicación web básica diseñada para gestionar las operaciones de venta y control de gasolina, incluyendo el registro de usuarios (personas y vehículos), la realización de ventas de combustible, la consulta de saldos mensuales de consumo y el historial de ventas. Incorpora un sistema de autenticación y autorización para diferenciar entre administradores y vendedores ademas de un control de venta.

**2. Características Principales**

El sistema ofrece las siguientes funcionalidades:

* **Autenticación de Usuarios:**
* Permite a los usuarios (admin y vendedor) iniciar sesión con credenciales predefinidas.
* Gestiona sesiones de usuario para mantener el estado de autenticación.
* Proporciona mensajes flash para notificaciones de éxito o error en el inicio de sesión.
* **Gestión de Usuarios (Personas y Vehículos):**
* **Registro de Personas y Vehículos:** Los administradores pueden registrar nuevas personas (con cédula, nombre y fecha de nacimiento) y asociarles un vehículo (tipo, chasis, año, marca).
* **Validación de Registro:** Evita el registro duplicado de personas por número de cédula.
* **Gestión de Ventas de Combustible:**
* Permite a los usuarios (vendedores) registrar ventas de combustible.
* Requiere la cédula de la persona para identificar al cliente.
* **Control de Límite Mensual:** Limita el consumo de combustible a 120 litros por persona al mes.
* **Control de Capacidad por Tipo de Vehículo:** Restringe la cantidad de litros según el tipo de vehículo (13 litros para motos, 76 litros para autos).
* Calcula el monto a pagar (litros \* 3.740 Bs).
* Actualiza el stock de combustible.
* **Consulta de Saldo de Consumo:**
* Permite consultar el consumo mensual actual de una persona y el saldo de litros disponibles respecto al límite mensual.
* Muestra información del vehículo asociado.
* **Historial de Ventas:**
* Permite consultar el historial de ventas de una persona dentro de un rango de fechas específico.
* Calcula el total de litros y el monto total vendido en el período seleccionado.
* **Roles de Usuario:**
* **Administrador:** Acceso para registrar nuevos usuarios (personas y vehículos).
* **Vendedor:** Acceso para registrar ventas de combustible.
* El acceso a ciertas rutas está restringido según el rol.
* **Persona Natural:**acceso a Consulta de Saldo y su Historial.

**3. Pila Tecnológica**

* **Backend:** Python 3 con el microframework Flask.
* **Base de Datos:** SQL Server 2014 para el almacenamiento de datos.
* **Frontend:** HTML para la estructura de la interfaz de usuario, usando Framework como bootstrap v5.0.
* **Gestión de Sesiones:** Flask session para manejar el estado de la sesión de usuario.
* **Mensajes al Usuario:** Flask flash para mostrar notificaciones temporales.

**4. Esquema de Base de Datos (Inferido)**

Basado en las operaciones del código, se infieren las siguientes tablas y sus posibles columnas:

* **usuarios**:
* id (INTEGER PRIMARY KEY AUTOINCREMENT)
* usuario (TEXT UNIQUE NOT NULL)
* contrasena (TEXT NOT NULL)
* rol (TEXT NOT NULL - 'administrador' o 'vendedor')
* **personas**:
* id\_persona (INTEGER PRIMARY KEY AUTOINCREMENT)
* cedula (TEXT UNIQUE NOT NULL)
* nombre\_completo (TEXT NOT NULL)
* fecha\_nacimiento (TEXT - formato 'YYYY-MM-DD')
* **vehiculos**:
* id\_vehiculo (INTEGER PRIMARY KEY AUTOINCREMENT)
* id\_persona (INTEGER - FOREIGN KEY a personas.id\_persona)
* tipo (TEXT - 'moto', 'auto', etc.)
* chasis (TEXT)
* anio (TEXT)
* marca (TEXT)
* **ventas**:
* id\_venta (INTEGER PRIMARY KEY AUTOINCREMENT)
* id\_persona (INTEGER - FOREIGN KEY a personas.id\_persona)
* id\_vehiculo (INTEGER - FOREIGN KEY a vehiculos.id\_vehiculo)
* fecha (TEXT - formato 'YYYY-MM-DD')
* litros\_cargados (REAL)
* monto\_pagado (REAL)
* **stock**:
* id\_stock (INTEGER PRIMARY KEY AUTOINCREMENT)
* fecha\_registro (TEXT - formato 'YYYY-MM-DD')
* litros (REAL - stock actual de combustible)

**5. Limitaciones y Posibles Mejoras**

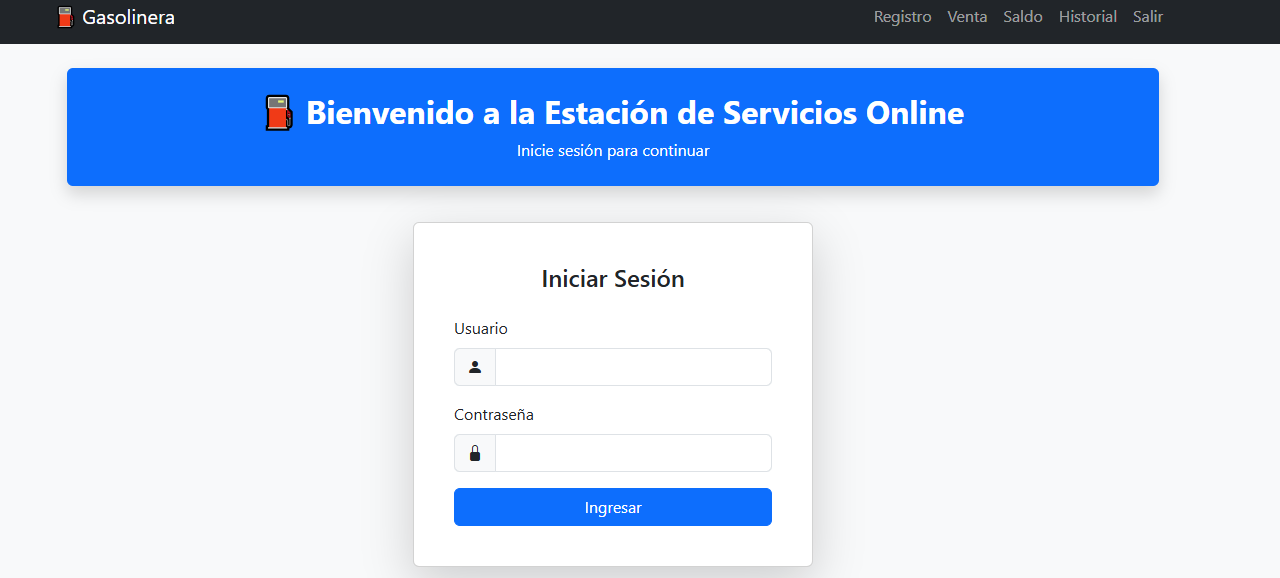
* **Seguridad:**
  + **Contraseñas:** Las contraseñas se almacenan en texto plano. Es crucial implementar el *hashing* de contraseñas (ej., con werkzeug.security.generate\_password\_hash y check\_password\_hash) para mejorar la seguridad.
  + **secret\_key:** La app.secret\_key está codificada directamente en el código. Debería cargarse desde una variable de entorno para producción.
* **Manejo de Errores:**
* Las funciones conectar() no tienen manejo de errores robusto para problemas de conexión a la base de datos.
* Los mensajes de error son genéricos ("⚠️ Persona no registrada."). Podrían ser más específicos para depuración.
* **Validación de Entrada:**
* La validación de entrada del usuario es mínima. Se debería validar que los campos numéricos sean realmente números, que las fechas tengan el formato correcto, y que los campos de texto no estén vacíos o contengan caracteres maliciosos.
* El límite de 13 litros para motos y 76 para autos está codificado. Podría ser configurable.
* **Interfaz de Usuario (UI/UX):**
* El código solo muestra la lógica del backend. La experiencia del usuario dependerá en gran medida de los archivos HTML y CSS asociados.
* La gestión de stock es muy básica (solo un registro de stock total). Un sistema más robusto manejaría diferentes tipos de combustible y sus cantidades.
* **Escalabilidad:**
* SQL Server es adecuado para aplicaciones con gran manero de Datos,es muy apropiada para manejar cantidades de registros como los que puede manejar el sistema de venta y control de combustible.
* La gestión de stock actual es un poco simplista. Cada venta resta del stock total, pero no hay un registro claro de entradas de combustible.
* **Módulos y Organización:**
  + El código está en un solo archivo. Para un proyecto más grande, se beneficiaría de una estructura modular (ej., separar rutas, modelos de base de datos, utilidades).
* **Funcionalidades Adicionales:**
  + Gestión de inventario de combustible (entradas y salidas).
  + Gestión de empleados (además de usuarios del sistema).
  + Sistema de precios dinámico para el combustible.

**6. Conclusión**

El proyecto de gestión de venta y control de combustible es una base funcional con las características esenciales para el registro de usuarios y ventas. Aunque cumple con su propósito básico, presenta oportunidades significativas de mejora en áreas como la seguridad, la robustez del manejo de errores, la validación de datos y la escalabilidad para un entorno de producción. La implementación de estas mejoras fortalecería considerablemente la aplicación.

**ANEXO**

PAGINA DE INICIO





MENU DEL SISTEMA