UNIVERSIDAD TECNOLÓGICA NACIONAL

CÁTEDRA: Soporte a la gestión de datos

Trabajo Práctico Integrador

Demo carro de compras

Año: 2022

Integrantes

Legajo	Apellido, nombre	Correo
44009	Kler, Nicolás	nicokler@hotmail.com
45711	Leones, Marcos	leonesmarcos96@gmail.com
42988	Tisera, Agustin	agustisera1@gmail.com

Narrativa

Se ha desarrollado un sistema que cumple en parte, las funcionalidades básicas de un sistema de ventas. El mismo se ha implementado con fines de la puesta en práctica de un stack tecnológico (sección y detalles en páginas siguientes) para evaluar su conveniencia, facilidad y tiempos de desarrollo.

La idea principal es probar una lista de casos de uso que cubren en aspectos generales la esencia de una aplicación de ventas con un listado de productos y usuarios con roles (administrador y cliente) capaces de popular un carro de compras donde se registre un histórico de las mismas. Pudiendo actualizar, agregar, eliminar productos y registrarse como usuario.

Requerimientos Funcionales

- El sistema debe ser capaz de registrar un usuario como cualquiera de los roles: Administrador o Cliente
- El sistema debe ser capaz de controlar la sesión del usuario.
- El sistema debe ser capaz de mostrar los productos registrados junto a sus detalles, independientemente del rol y sesión del usuario. (Stock, descripción, otro)
- El sistema debe ser capaz de registrar un nuevo producto en la base de datos.
- El sistema debe ser capaz de modificar un producto existente en la base de datos.
- El sistema debe ser capaz de eliminar un producto que ya no sea requerido en la base de datos.
- El sistema debe ser capaz de popular un carro de compras acorde a la sesión del usuario.
- El sistema debe ser capaz de registrar una venta/compra de productos para un usuario.
- El sistema debe ser capaz de limitar sus funcionalidades de acuerdo al rol del usuario.
- El sistema debe ser capaz de aplicar lógica de negocios al carro de compra. (Cálculo de totales, porcentajes IVA, otros).
- El sistema debe validar los campos de registro/edición para preservar la integridad.

Requerimientos no Funcionales

- La funcionalidad del sistema debe contar con tests unitarios.
- El sistema debe contar con integración continua

Stack Tecnológico

El sistema se ha implementado con la combinación de:

Flask framework: https://flask.palletsprojects.com/en/2.2.x

El cual incluye la siguiente lista de librerías basadas en **Python** (ver bibliografía):

> **Werkzeug** Implementa WSGI y el estándar python entre aplicaciones y servers
>
> **Jinja** Implementa los templates para las páginas que la aplicación sirve
>
> **MarkupSafe** Junto con Jinja, protege ante ataques inyectados al renderizar los templates
>
> **ItsDangerous** Paquetes de utilidad para asegurar la integridad de los datos (encriptadores, otros). Protege las variables de sesión de Flask y cookies.
>
> **Click** framework para definir comandos en de terminal (CLI).

En cuanto a la implementación del modelo de datos:

Sqlite3 y SQL syntax: Implementa una versión ligera de motor de base de datos SQL, siguiendo una versión particular de la sintaxis SQL

Git: sistema de versión de control para controlar la integridad del proyecto durante su desarrollo.

Github actions: Implementa el control de flujos de trabajo e integridad del proyecto en nuestro repositorio (CI - Integración continua)

Pytest y coverage: Define y lleva a cabo los unit tests y cobertura de la aplicación.

Reglas Negocio

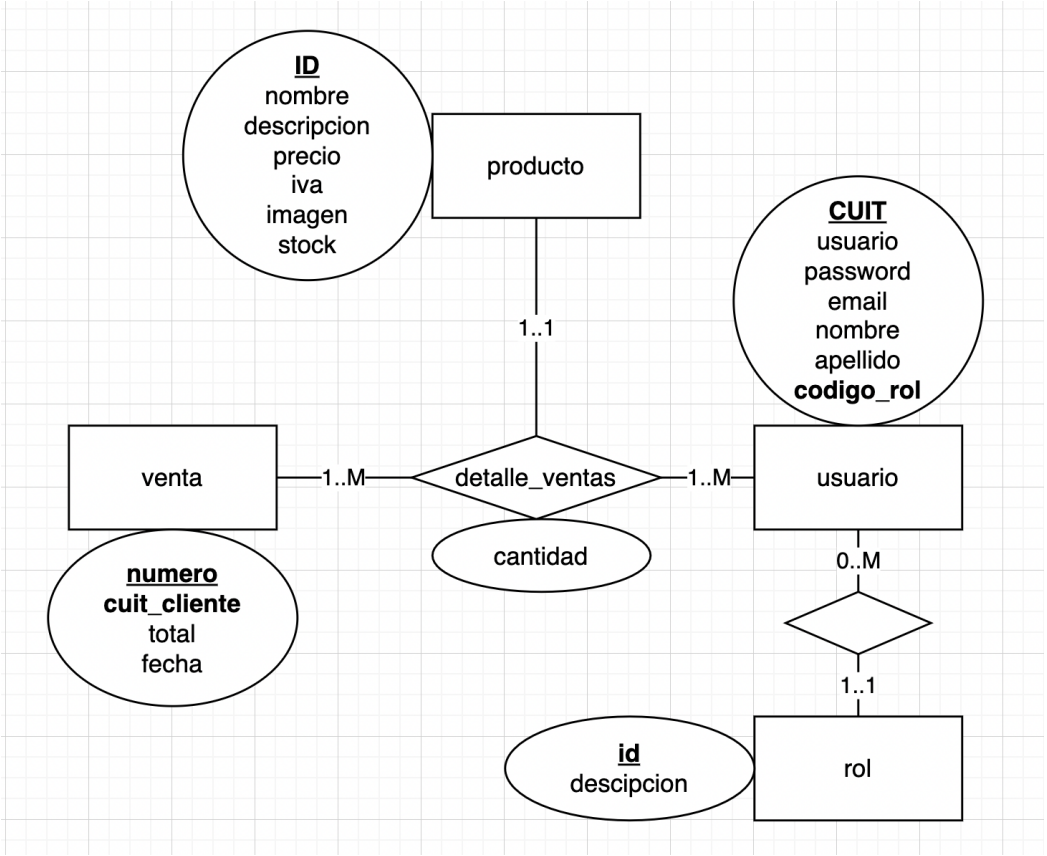
1. Una venta se identifica por un número único y se conoce el cliente, la fecha en que se realizó, y los detalles de la misma.
2. De un detalle de venta se conoce el producto, la cantidad y a que venta pertenece.
3. Los productos se identifican por un id único y se conoce el nombre, la descripción, el precio, el porcentaje de IVA que le corresponde y el stock actual.
4. Cada usuario tiene un rol (Cliente o Administrador) y se identifica por su CUIT. Se conoce el nombre, apellido, email. Tienen un nombre de usuario y una contraseña.
5. No se podrán registrar dos usuarios con el mismo número de CUIT.
6. Los usuarios Administradores tienen acceso a crear, modificar y eliminar productos. También cuentan con la posibilidad de consultar las ventas realizadas. Los usuarios del tipo Cliente no.
7. Los Clientes pueden consultar los productos que se encuentran en el sistema y realizar compras, también pueden consultar otras compras que hayan hecho previamente. Los Administradores no pueden registrar compras propias.
8. Al registrar una compra se debe validar que la cantidad solicitada para un producto sea menor o igual a la cantidad en existencia.
9. Cuando se registra una venta el stock de los productos pasará a ser igual al stock previo a la venta menos la cantidad vendida.
10. El precio de venta de un producto será el precio que figura más un porcentaje que corresponde al IVA.

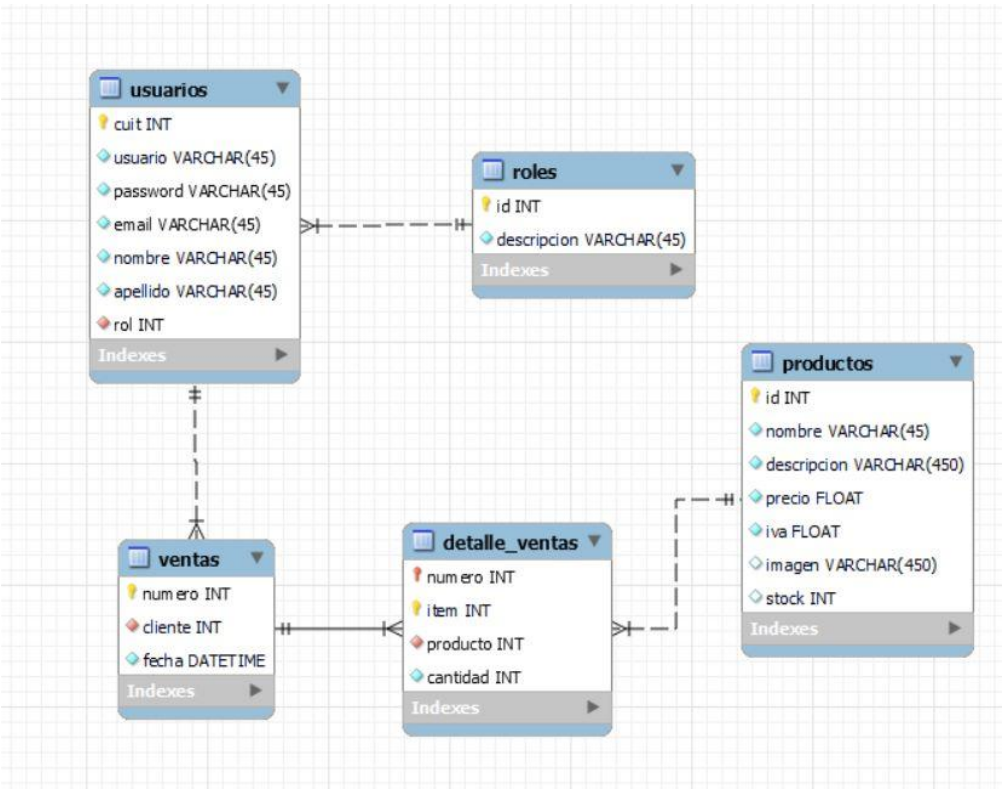
Casos de Usos Principales

El sistema en su versión actual cuenta con los casos de uso:

- Registro de usuarios
- Inicio de sesión de un usuario
- Listado de productos disponibles en el shop
- Alta de un producto en sistema
- Baja de un producto en sistema
- Modificación de un producto en sistema
- Listado del historial de compras de un usuario
- Crear/Actualizar el carro de compras de un usuario

Modelo datos

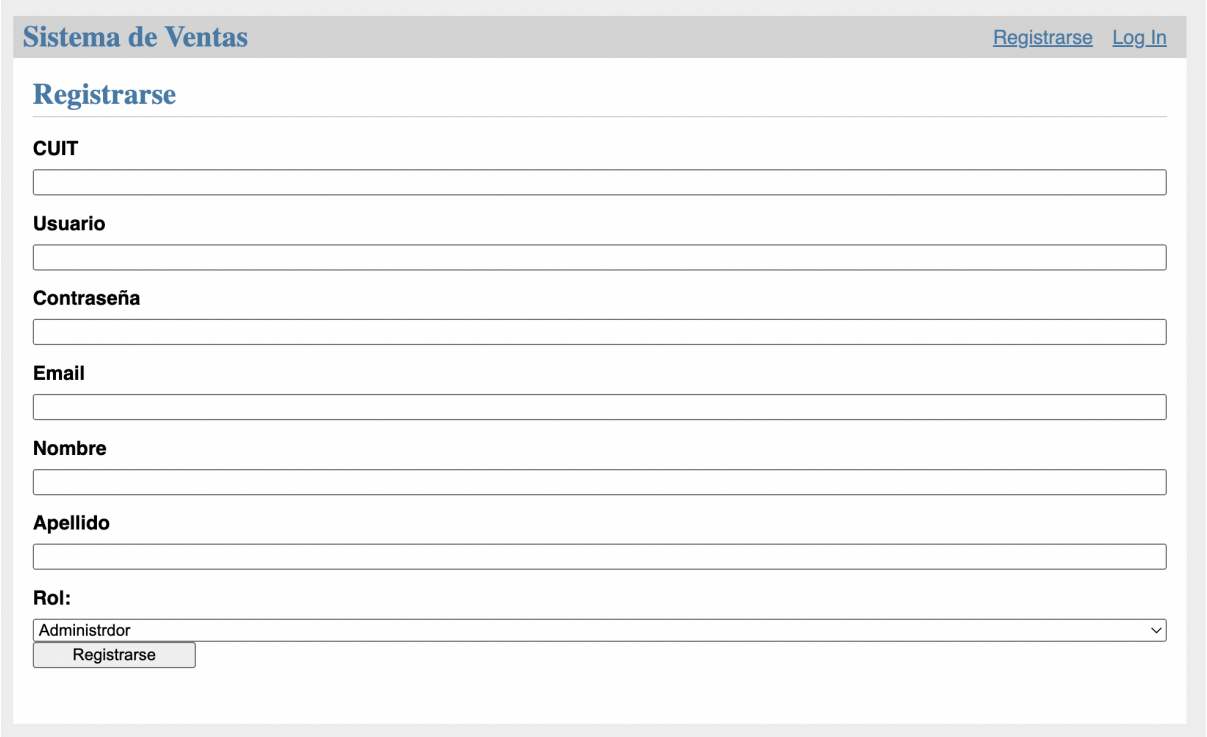




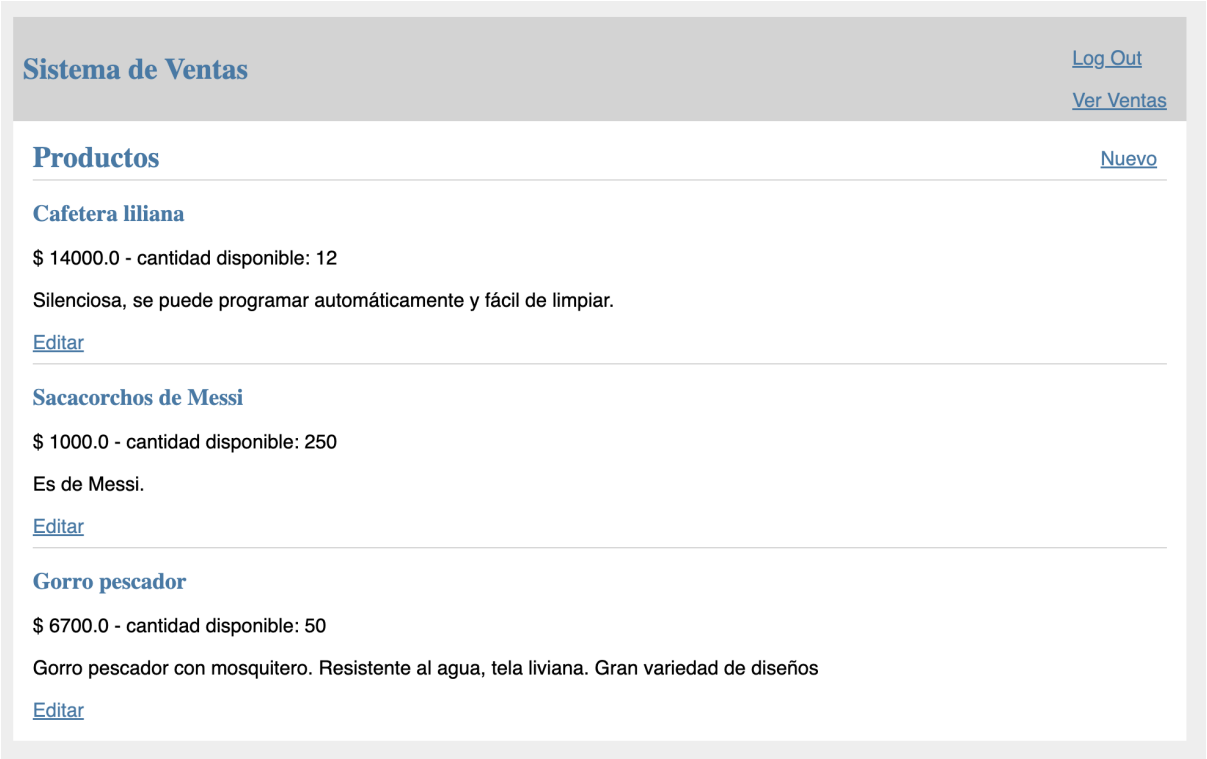
Vistas

Inicio sesión

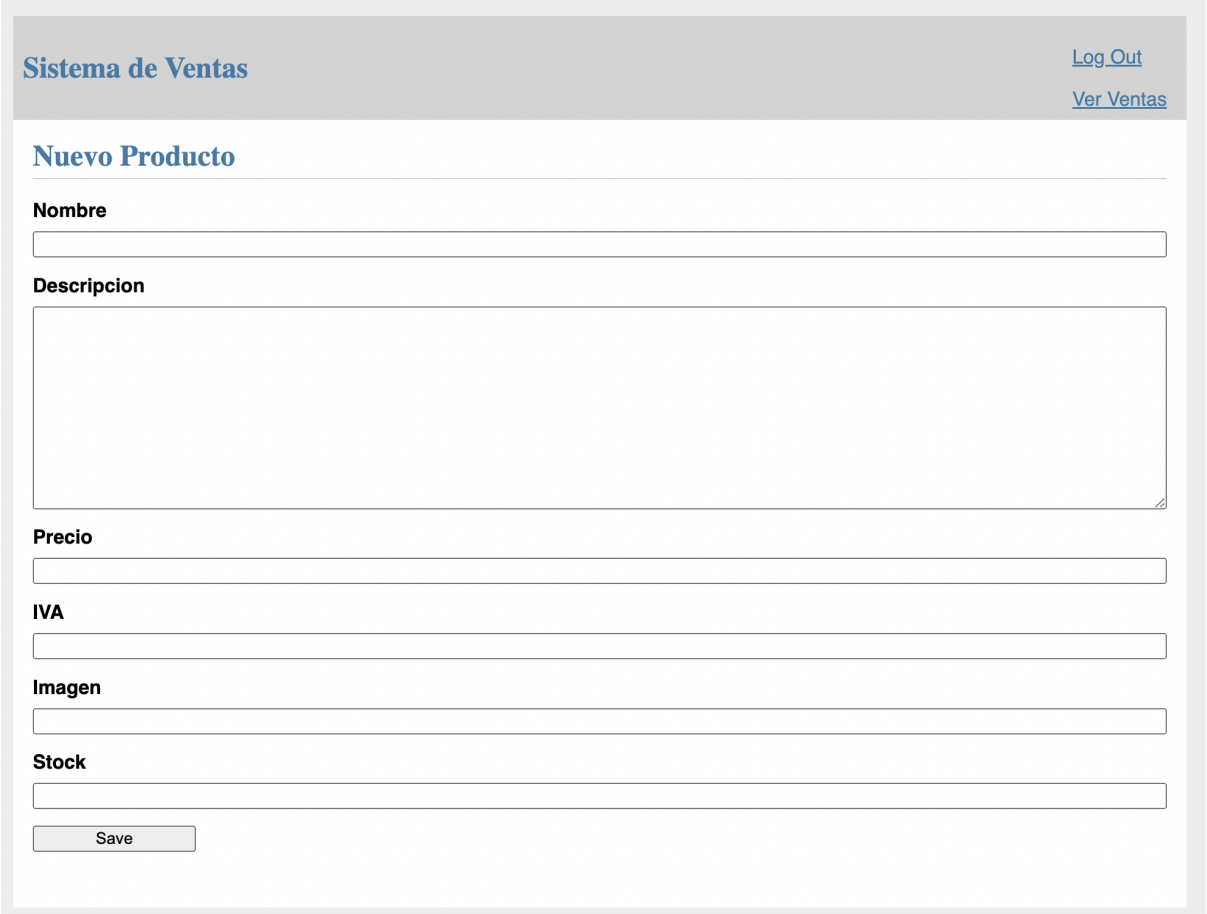
Registro usuario



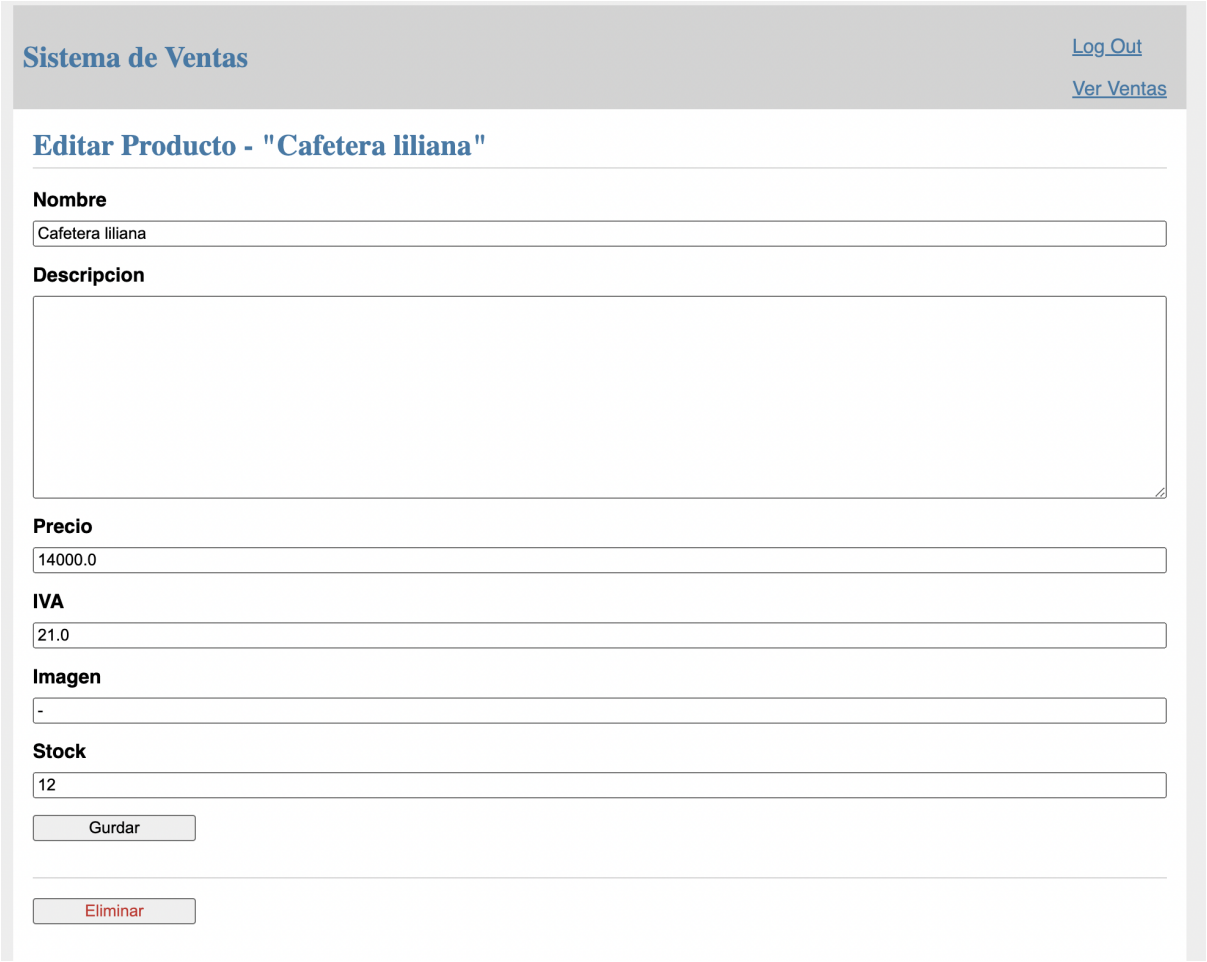
Listado de productos



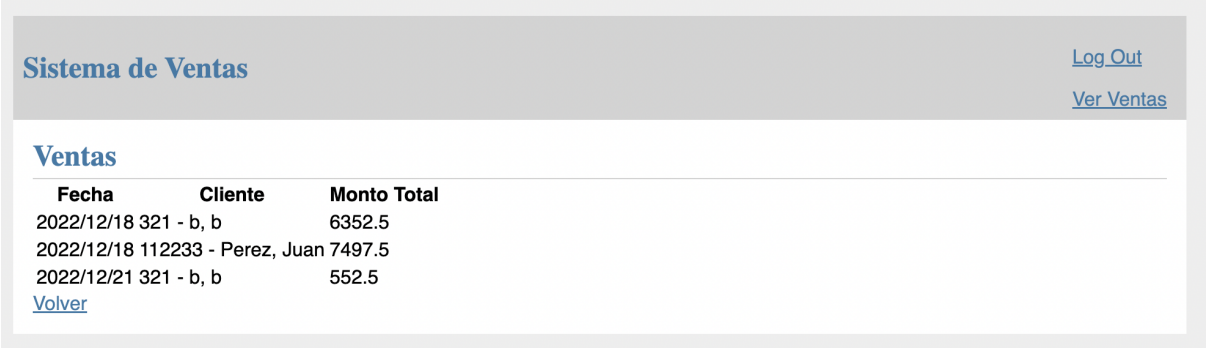
Alta producto



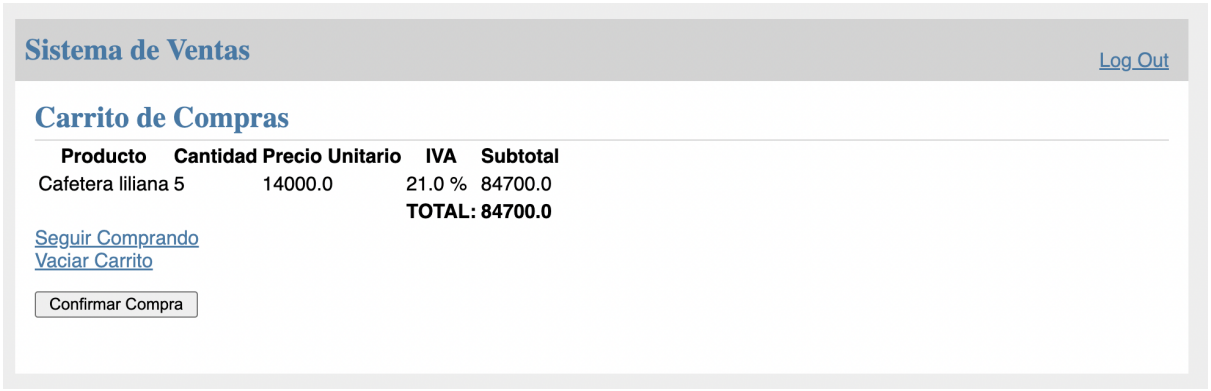
Edición y baja de producto:



Historico de compras de un usuario



Carrito de compras



Bibliografia

Secciones más relevantes flask:
https://flask.palletsprojects.com/en/2.2.x/tutorial/factory/

https://flask.palletsprojects.com/en/2.2.x/tutorial/database/

https://flask.palletsprojects.com/en/2.2.x/tutorial/views/

https://flask.palletsprojects.com/en/2.2.x/tutorial/install/

WSGI: https://wsgi.readthedocs.io/en/latest/what.html

werkzeug: https://palletsprojects.com/p/werkzeug

jinja: https://palletsprojects.com/p/jinja

markupsafe: https://palletsprojects.com/p/markupsafe/

itsdangerous: https://palletsprojects.com/p/itsdangerous/

click: https://palletsprojects.com/p/click/

sql syntax para sqlite3: https://www.sqlite.org/lang.html

sqlite3: https://www.sqlite.org/index.html

git: https://git-scm.com/

pytest: https://docs.pytest.org/en/7.2.x/

coverage: https://coverage.readthedocs.io/

githubActions https://docs.github.com/en/actions

Enlace repositorio:

Github: https://github.com/MarcosLeones/TPI-Soporte