|  |  |
| --- | --- |
| INTRODUCCIÓN A LA PROGRAMACIÓN Y COMPUTACIÓN II | NOTA: |
| 2S2023 |

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE SISTEMAS

ING. JAIME FRANCISCO YUMAN RAMIREZ

Proyecto 2

**REPOSITORIO:** https://github.com/miguel-tubac/IPC2\_Proyecto2Diciembre\_1.git

|  |
| --- |
| **PROYECTO 2 – IPC2** |
| **201807086 – Nery José Barrientos Posadas**  **202200329 – Adler Alejandro Pérez Asensio**  **201901803 – Benjamin Alexander Torcelli Barrios**  **202101927 – Miguel Adrian Tubac Agustin**  **201900416 – Carlos René Coronado Sandoval** |

**Resumen**

El proyecto tiene como objetivo principal de desarrollar un sistema CRUD (Create, Read, Update, Delete) en la web, aplicando el patrón de diseño MVC (Model-View-Controller). Se enfoca en el uso de Programación Orientada a Objetos (POO) con Python, HTML para la capa de presentación, CSS para estilizar las páginas y el framework Django para el desarrollo.

La aplicación a desarrollar es un sistema para un punto de venta, con las siguientes funcionalidades:

1. **Menú amigable:** Debe permitir el acceso a las opciones de manera intuitiva.
2. **Operaciones CRUD:** Crear, editar, eliminar y listar clientes, productos y facturas.
3. **Campos mínimos para productos y clientes:** Detalles específicos para productos (Nombre, Descripción, Precio, Stock) y clientes (Nit, Nombre, Dirección).
4. **Maestro y detalle para facturas:** Almacenamiento de datos estructurados en maestro y detalle.
5. **Búsqueda y ordenamiento:** Implementación para productos, clientes y facturas.
6. **Templates basados en HTML y CSS:** Para la visualización de datos.
7. **Uso de Django Models, Forms y Views:** Definición de datos, creación/edición, y lógica del negocio.
8. **Almacenamiento en XML:** Se utilizará XML para el almacenamiento de datos.

En cuanto a la implementación, el proyecto incluye reportes en forma de un dashboard HTML, que visualice de manera gráfica los productos más vendidos y el top de clientes con más compras.

La interfaz gráfica, diseñada a través de Django, es intuitiva y presentar una estética atractiva. En resumen, se aplica conceptos de programación web, patrones de diseño, y utiliza tecnologías específicas para lograr un sistema funcional de punto de venta con capacidades de reporte y una interfaz de usuario agradable.

**Palabras clave**

Backend, Frontend, CRUD, MVC.

***Abstract***

*The main objective of the project is to develop a CRUD (Create, Read, Update, Delete) system on the web, applying the MVC (Model-View-Controller) design pattern. It focuses on the use of Object Oriented Programming (OOP) with Python, HTML for the presentation layer, CSS to stylize the pages and the Django framework for development.*

*The application to be developed is a system for a point of sale, with the following key functionalities:*

*1. Friendly menu: It should allow access to the options in an intuitive way.*

*2. CRUD Operations: Create, edit, delete and list customers, products and invoices.*

*3. Minimum fields for products and customers: Specific details for products (Name, Description, Price, Stock) and customers (Nit, Name, Address).*

*4. Master and detail for invoices: Storage of structured data in master and detail.*

*5. Search and sorting: Implementation for products, customers and invoices.*

*6. Templates based on HTML and CSS: For data visualization.*

*7. Use of Django Models, Forms and Views: Data definition, creation/editing, and business logic.*

*8. Storage in XML: XML will be used for data storage.*

*Regarding implementation, the project includes reports in the form of an HTML dashboard, which graphically displays the best-selling products and the top customers with the most purchases.*

*The graphical interface, designed through Django, is intuitive and presents an attractive aesthetic. In summary, it applies web programming concepts, design patterns, and uses specific technologies to achieve a functional point of sale system with reporting capabilities and a pleasant user interface.*

***Keywords***

*Backend, Frontend, CRUD and MVC.*

**Introducción**

En el marco de la creciente demanda de soluciones tecnológicas eficientes en el ámbito comercial, el presente proyecto tiene como objetivo general la implementación de un sistema CRUD (Create, Read, Update, Delete) en entorno web, siguiendo el esquema de diseño MVC (Modelo-Vista-Controlador). Este proyecto se enfoca en el desarrollo de una aplicación web para un punto de venta, utilizando como base el lenguaje de programación Python y el framework Django.

**Objetivos Específicos:**

1. Implementar Programación Orientada a Objetos (POO) mediante el uso del lenguaje Python.
2. Utilizar código HTML para la capa de presentación, asegurando una interfaz amigable.
3. Aplicar código CSS para personalizar los estilos de las páginas y mejorar la experiencia del usuario.
4. Desarrollar la aplicación utilizando el framework Django, aprovechando sus capacidades para simplificar el desarrollo web.

**Descripción General:** La aplicación web para punto de venta permitirá realizar operaciones clave, como la gestión de clientes, productos y facturas. Cada producto contendrá campos esenciales como nombre, descripción, precio y stock, mientras que cada cliente estará caracterizado por su NIT, nombre y dirección. La gestión de facturas se dividirá en maestro y detalle para un almacenamiento de datos eficiente. Se implementarán funciones de búsqueda y ordenamiento para productos, clientes y facturas, y se integrarán templates basados en HTML y CSS. Además, se utilizarán Django Models para la definición de datos, Forms para la creación/edición y Views para la lógica del negocio, con almacenamiento a través de archivos XML.

**Implementación:** En términos de informes, se incluirá un dashboard en HTML que visualizará gráficamente los productos más vendidos y el top de clientes con más compras.

**Interfaz Gráfica:** La interfaz gráfica se diseñará utilizando Django, prestando especial atención a la intuición y presentación del sistema. La facilidad de uso y la estética serán aspectos fundamentales en el desarrollo, asegurando una experiencia fluida y agradable para el usuario final.

**Desarrollo del tema**

*Análisis Sentimental:*

El proyecto presenta un enfoque claro y ambicioso al buscar desarrollar un sistema CRUD en la web bajo el esquema Modelo-Vista-Controlador (MVC). La elección de utilizar Python, especialmente con el marco de trabajo Django, indica una decisión bien fundamentada para aprovechar la eficiencia y la robustez de este entorno de desarrollo.

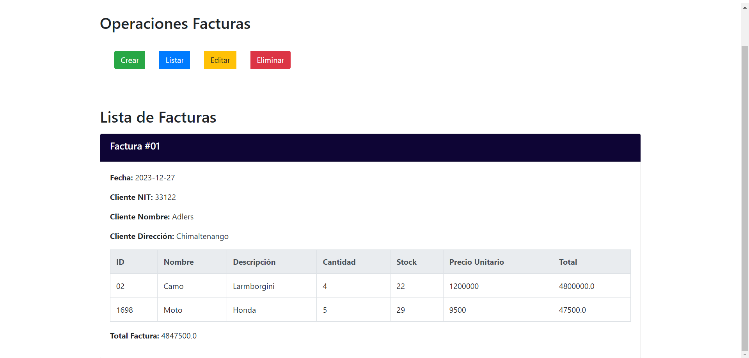
**Aspectos Positivos:**

1. **Objetivos Claros:** El proyecto tiene objetivos generales y específicos bien definidos, lo que facilita la comprensión de sus metas.
2. **Tecnologías Adecuadas:** La elección de utilizar Programación Orientada a Objetos (POO) con Python y el uso de Django para el desarrollo web sugieren una elección de tecnologías coherente y moderna.
3. **Funcionalidades Esenciales:** La inclusión de funciones clave, como la gestión de clientes, productos y facturas, así como la capacidad de búsqueda y ordenamiento, demuestra una comprensión sólida de las necesidades de un punto de venta.
4. **Diseño Gráfico y Presentación:** La atención al diseño de la interfaz gráfica mediante Django, HTML y CSS indica una preocupación por la usabilidad y la presentación visual, factores importantes para una aplicación web exitosa.
5. **Uso de XML:** La elección de almacenar datos mediante XML podría tener ventajas en ciertos contextos y puede ser una decisión interesante según las necesidades específicas del proyecto.

**Aspectos a Considerar:**

1. **Almacenamiento con XML:** Aunque la elección de XML puede ser válida, se debe considerar si es la mejor opción en términos de eficiencia y escalabilidad, especialmente comparada con otras bases de datos que Django y Python pueden soportar.
2. **Amplitud del Proyecto:** La implementación de un sistema de reportes y un dashboard agrega complejidad al proyecto. Es importante evaluar si esto se alinea con los recursos disponibles y el tiempo estimado para el desarrollo.
3. **Testing y Validación:** No se mencionan aspectos relacionados con pruebas y validación del código. Incorporar estrategias de prueba puede mejorar la calidad del software desarrollado.
4. **Seguridad:** No se aborda la seguridad en la descripción general. La seguridad de una aplicación web es crucial, y se debe considerar aspectos como la validación de datos de entrada, protección contra ataques de inyección, etc.

En general, el proyecto muestra un enfoque positivo hacia el desarrollo de una aplicación web para un punto de venta. Con la implementación cuidadosa de los objetivos específicos y la consideración de algunos aspectos a mencionar, es probable que resulte en una aplicación funcional y bien diseñada.



**Almacenamiento de Datos:**

Al desarrollar el proyecto descrito, el almacenamiento de datos se realizará utilizando Django Models y se ha especificado que se utilizará XML como formato de almacenamiento. A continuación, una descripción general de cómo se podría estructurar el modelo de datos en Django para cumplir con los requisitos del proyecto:

1. **Cliente:**
   * Nit (campo de texto)
   * Nombre (campo de texto)
   * Dirección (campo de texto)
2. **Producto:**
   * Nombre (campo de texto)
   * Descripción (campo de texto)
   * Precio (campo decimal)
   * Stock (campo entero)
3. **Factura (Maestro):**
   * Cliente (clave externa que se relaciona con el modelo Cliente)
   * Fecha de emisión (campo de fecha)
   * Total (campo decimal, calculado automáticamente)
4. **Detalle de Factura:**
   * Factura (clave externa que se relaciona con el modelo Factura)
   * Producto (clave externa que se relaciona con el modelo Producto)
   * Cantidad (campo entero)
   * Subtotal (campo decimal, calculado automáticamente)

En cuanto al almacenamiento por medio de XML, Django permite utilizar diferentes motores de base de datos, pero para este caso formato XML utilizando las bibliotecas proporcionadas por Django.

La implementación de búsqueda y ordenamiento de productos, clientes y facturas se realizará mediante el uso de consultas Django, y la presentación de datos se hará a través de templates basados en HTML y CSS.

Para los reportes, se implementa una vista en Django que recopile los datos necesarios y genere el HTML correspondiente para el dashboard.

En resumen, la estructura de almacenamiento de datos se basará en modelos Django con un motor de base de datos compatible y una implementación personalizada para la serialización/deserialización a XML.

**Conclusiones**

En conclusión, el proyecto tiene como objetivo principal desarrollar un sistema CRUD basado en la web, utilizando el esquema Modelo-Vista-Controlador (MVC). Se propone la implementación de Programación Orientada a Objetos (POO) en Python, junto con HTML y CSS para la presentación de la interfaz. La elección de utilizar el framework Django proporciona una base sólida para la creación de la aplicación web.

La aplicación está orientada a un punto de venta e incluye funcionalidades clave, como la gestión de clientes, productos y facturas. Se establecen requisitos específicos para los campos de productos y clientes, y se plantea la necesidad de un modelo maestro-detalle para las facturas. La búsqueda y ordenamiento de datos se consideran aspectos esenciales para una experiencia de usuario eficiente.

En cuanto a la implementación, se destaca el uso de Django Models para la definición de datos, Forms para la creación/edición y Views para la lógica del negocio. El almacenamiento de datos se propone mediante XML, proporcionando una opción de almacenamiento estructurado.

Además, se propone la generación de informes a través de un dashboard HTML que mostrará gráficamente los productos más vendidos y los principales clientes en términos de compras. La interfaz gráfica, diseñada con Django, se enfocará en la intuición y la presentación visual para mejorar la experiencia del usuario.

En resumen, el proyecto se presenta como una solución completa para la gestión de un punto de venta, integrando tecnologías web modernas y buenas prácticas de desarrollo para lograr un sistema robusto y amigable.

**Apéndices**

**Apéndice 1**

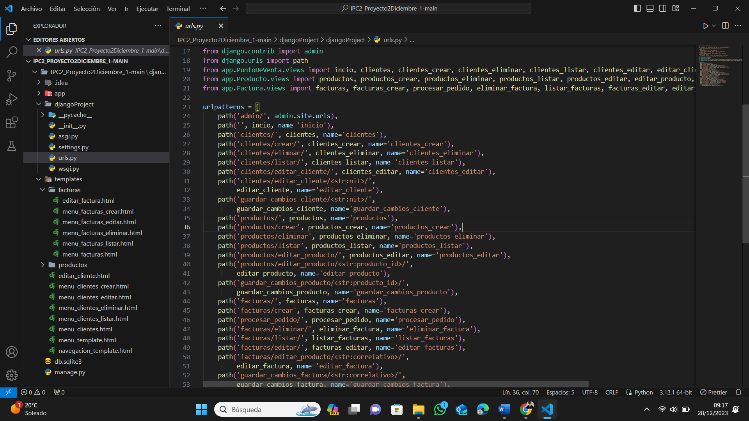


Imagen 1

Fuente: Dominio Propio

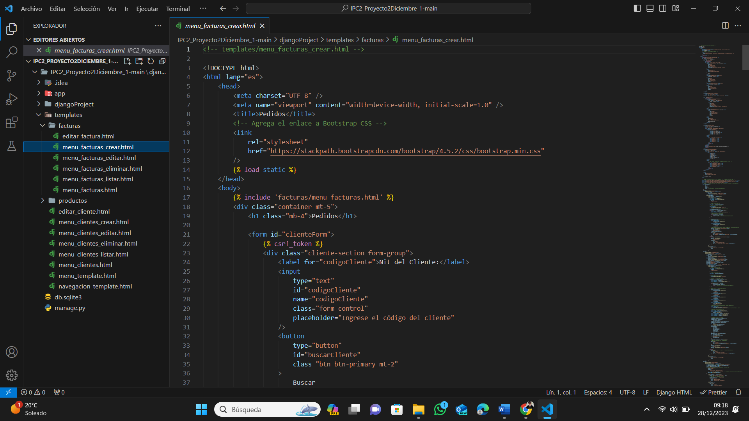
****

Imagen 2

Fuente: Dominio Propio

Imagen 1: Se observa el inicio de la carpeta principal con todas las templates que son todos los HTML que fueron utilizadas para la realización de la aplicación web, en la Imagen 2: la case urls.py donde se almacena los directorios de todo el programa.

**Apéndice 2**

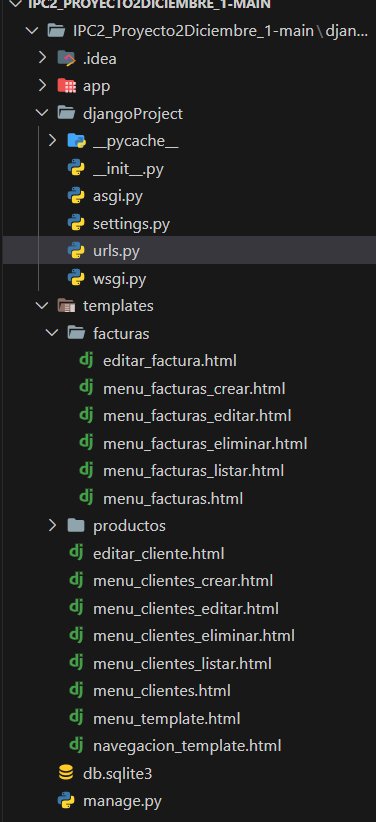


Imagen 3

Fuente: Dominio Propio

Imagen 3 donde se observa la estructuro de Punto de venta, con todos los templates y la clase principal manage.py, y las clases necesarias para almacenar la información de los templates en archivos XLM.

**Apéndice 3**

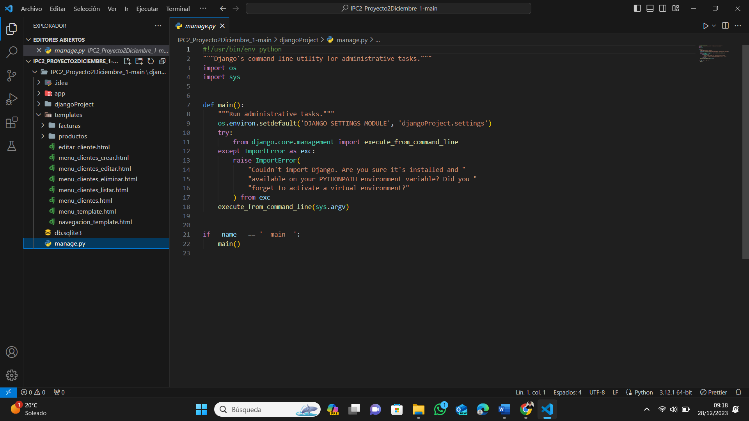


Imagen 4

Fuente: Dominio Propio

En la imagen 4 se observa manage.py donde se abre en la terminal para poder correr Punto de Venta.

**Anexos:**

**Punto de Venta - Manual de Instalación**

manual básico de instalación para el Punto de Venta. Importante tener Python y Django instalados en tu sistema. Si no es así, instálalos antes de comenzar. Con el siguiente comando: pip install django. Instalar Python desde el siguiente enlace: https://www.python.org/downloads/

Este manual proporciona los pasos necesarios para instalar y configurar el proyecto de Punto de Venta desarrollado con Django. Asegúrate de tener instalado Python y pip en tu sistema antes de comenzar.

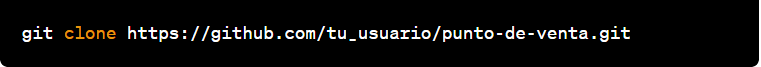
**Requisitos Previos**

1. **Python y pip:** Asegúrate de tener instalada la versión de Python adecuada para el proyecto. Puedes descargar Python desde python.org y pip se instala automáticamente con Python.
2. **Git:** Asegúrate de tener Git instalado para clonar el repositorio del proyecto.

**Instalación**

1. **Clonar el Repositorio:**

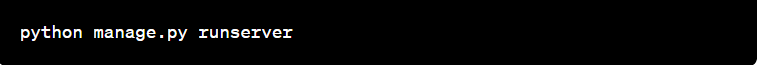
Abre una terminal y ejecuta el siguiente comando:



**Ejecución del Proyecto**

1. **Ejecutar el Servidor:**

Inicia el servidor de desarrollo:

****

**Almacenamiento en XML**

1. Asegúrate de manejar adecuadamente la lectura y escritura de archivos XML.

**Ejecución de la Aplicación**

Después de realizar estos pasos, tendrás la aplicación Punto de Venta en funcionamiento. Puedes acceder al panel de administración, realizar operaciones CRUD y explorar los informes y el dashboard.

Recuerda que este manual es una guía general y puede requerir ajustes según las características específicas de tu entorno de desarrollo

**Punto de Venta - Manual de Uso**

**Repositorio de GitHub**

El código fuente del proyecto se encuentra disponible en el siguiente repositorio de GitHub: https://github.com/miguel-tubac/IPC2\_Proyecto2Diciembre\_1.git

**Requisitos Previos**

1. Asegúrate de tener Python instalado en tu máquina. Puedes descargarlo desde python.org.
2. Instala Django ejecutando el siguiente comando en tu terminal o consola:

pip install django

**Ejecución del Proyecto**

1. Inicia el servidor de desarrollo de Django:

python manage.py runserver

**Uso de la Aplicación**

**Menú Principal**

El menú principal ofrece opciones para gestionar clientes, productos, facturas y generar reportes.

**Clientes**

1. **Listar Clientes:** Permite visualizar la lista de clientes registrados.
2. **Crear Cliente:** Agrega un nuevo cliente con su información.
3. **Editar Cliente:** Modifica la información de un cliente existente.
4. **Eliminar Cliente:** Elimina un cliente de la base de datos.

**Productos**

1. **Listar Productos:** Muestra la lista de productos disponibles.
2. **Crear Producto:** Añade un nuevo producto con detalles como nombre, descripción, precio y etc.
3. **Editar Producto:** Modifica la información de un producto existente.
4. **Eliminar Producto:** Elimina un producto de la base de datos.

**Facturas**

1. **Listar Facturas:** Visualiza la lista de facturas emitidas.
2. **Crear Factura:** Genera una nueva factura seleccionando productos y clientes.
3. **Editar Factura:** Realiza modificaciones en una factura existente.
4. **Eliminar Factura:** Elimina una factura registrada.

**Búsqueda y Ordenamiento**

* **Ordenamiento de Listas:** Ordenamiento en forma de Listas.

**Reportes**

1. **Dashboard:** Accede al panel de control que presenta información sobre los productos más vendidos y los clientes con más compras.

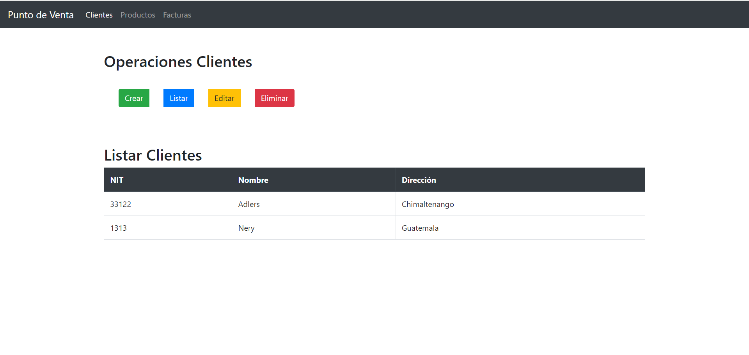
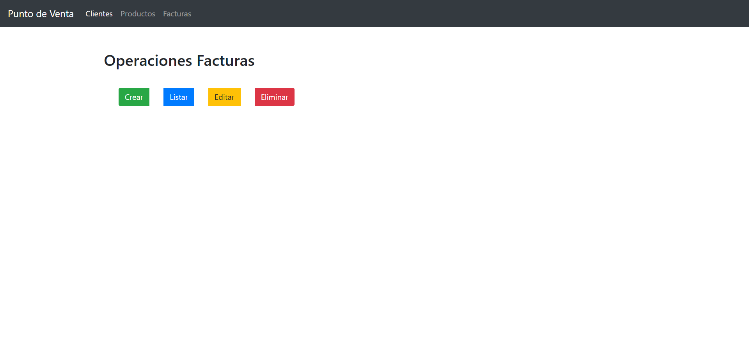
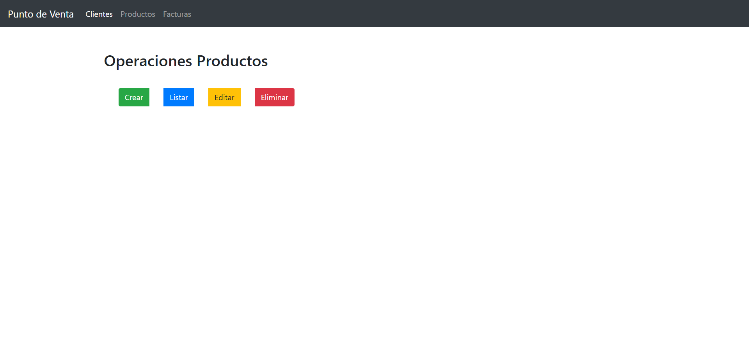
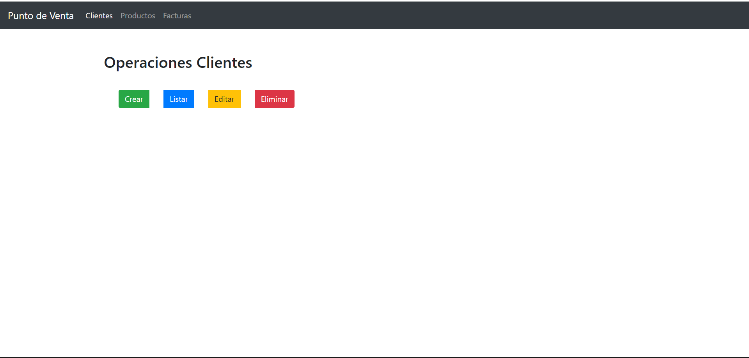
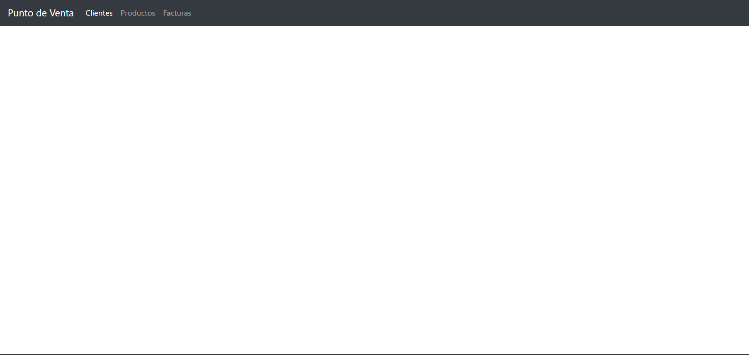
**Almacenamiento en XML**

Los datos de la aplicación se almacenan en archivos XML para facilitar la portabilidad y el respaldo de la información.

**Interfaz Gráfica**

La interfaz gráfica sigue un diseño intuitivo y amigable, priorizando la usabilidad y la presentación visual.

IMÁGENES DE LA INTERFAZ PARA GUIA DE USO:



**Referencias bibliográficas**

**Interfaz Gráfica:**

* https://www.sing-group.org/~lipido/blog/2013/05/07/interfaces-de-usuario-htmlcssjavascript-nuevo-estandar/

**HTMLS:**

* <https://www.w3schools.com/html/html_css.asp>
* https://developer.mozilla.org/es/docs/Learn/Getting\_started\_with\_the\_web/HTML\_basics

**Otros**

[Aprende A Programar En Python Para Principiantes, Escrito Por: Flynn Fisher](https://desarrolladoresweb.org/libros/libro/aprende-a-programar-en-python-para-principiantes-escrito-por-flynn-fisher/)