Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Introducción a la Programación y Computación 2

Inga. Claudia Liceth Rojas Morales
Ing. Marlon Antonio Pérez Türk
Ing. José Manuel Ruiz Juárez
Ing. Dennis Stanley Barrios González
Ing. Edwin Estuardo Zapeta Gómez
Ing. Fernando José Paz González

Tutores de curso:
Dayana Alejandra Reyes Rodríguez
Andrea María Cabrera Rosito
Paula Gabriela García Reinoso
Piter Angel Esaú Valiente de León
Mario Cesar Moran Porras
Denilson Florentín de León Aguilar

PRÁCTICA #3

Objetivo General

- Desarrollar habilidades en programación web mediante la implementación de aplicaciones utilizando los frameworks Django y Flask. Donde los estudiantes aplicarán principios de programación orientada a objetos (POO) en Python, consolidando sus conocimientos teóricos al enfrentar la complejidad añadida del desarrollo con frameworks.

Objetivos Específicos

- Que el estudiante desarrolle la habilidad de integrar de manera efectiva el frontend con el backend, asegurando una correcta comunicación entre ambos.
- Que el estudiante aprenda a realizar peticiones HTTP desde el frontend hacia una API backend, comprendiendo los principios de comunicación cliente-servidor..
- Que el estudiante sea capaz de procesar datos en el backend y generar respuestas en formato XML.

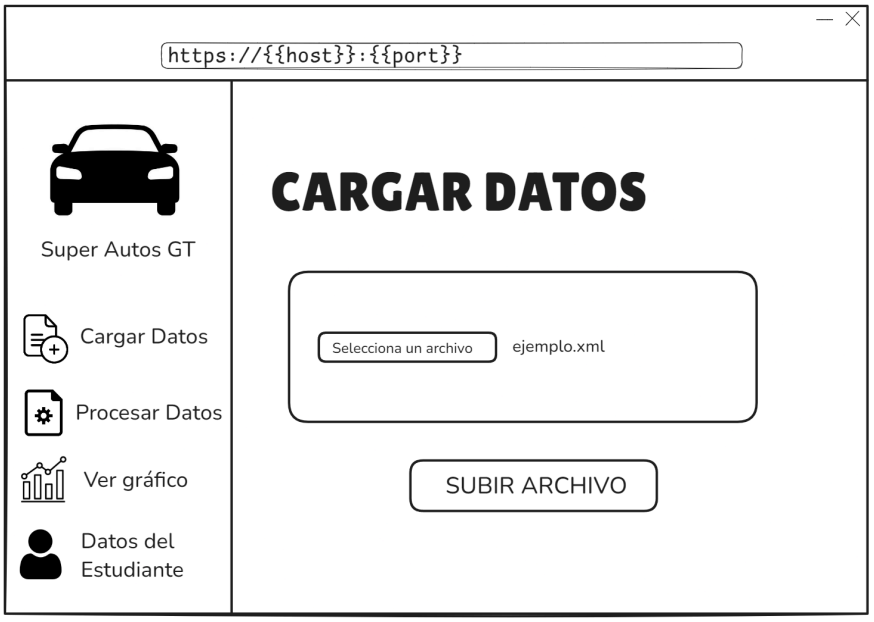
Descripción

Todas las soluciones que usted ha proporcionado a la agencia *Super Autos GT* han hecho que esta se sienta satisfecha con los resultados, es por esto mismo que ahora le piden un sistema de reportaje web utilizando Flask y Django. **Flask para backend y Django para frontend**. Se le pide que en la capa de cliente, el usuario sea capaz de cargar un archivo de tipo XML que contendrá la información de las ventas realizadas y, en base a esto, que igualmente reciba otro XML de respuesta que contendrá la información procesada para su fácil lectura y, que pueda visualizar gráficos. El funcionamiento es el siguiente:

Frontend - Django:

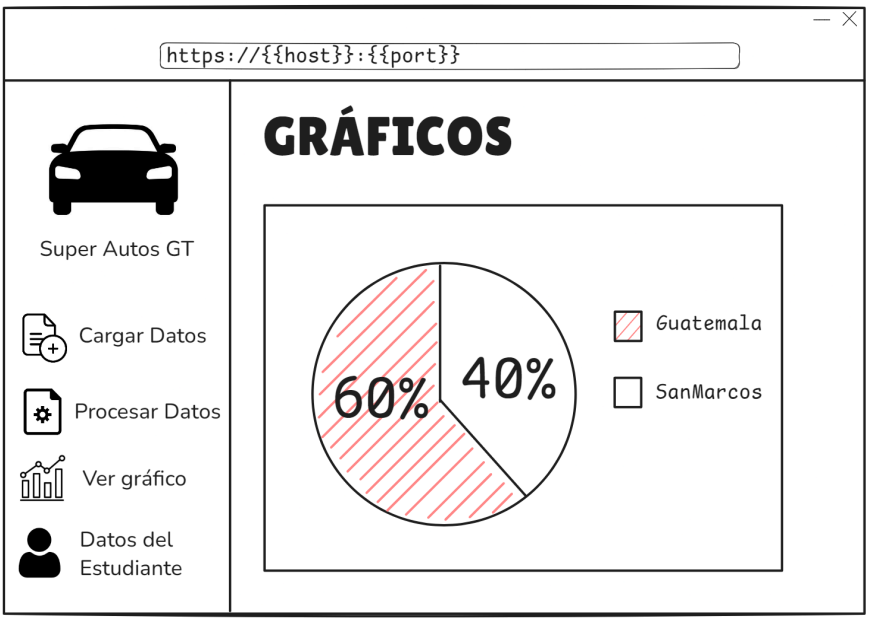
Ya que solo se le pide que implemente la parte de registros y revisión de datos, se le piden los siguientes componentes:

- **Cargar Archivo de Registros:** En esta pantalla el usuario que utilice la aplicación web podrá cargar lo que es el archivo XML con los registros de las ventas. (*Haga una revisión en la sección de archivos de entrada para visualizar el formato que estos archivos deberán tener).*

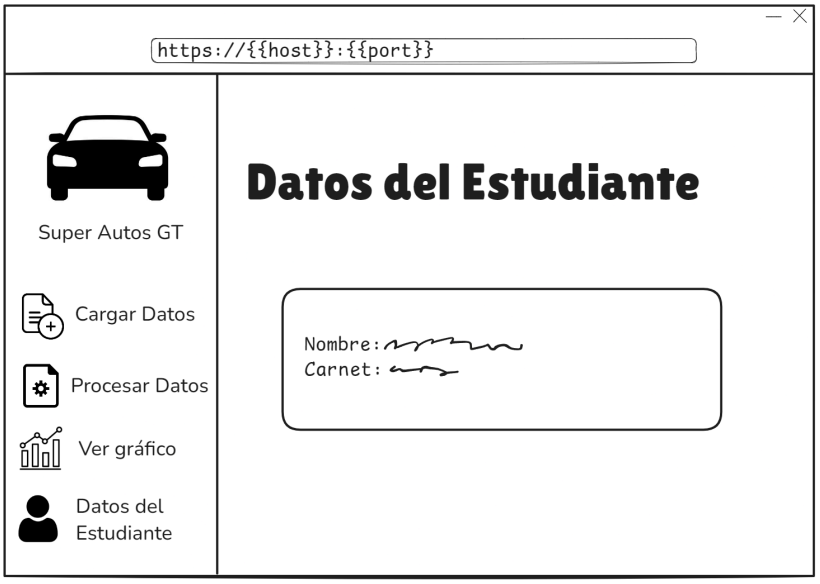


- **Revisión de datos procesados:** En esta pantalla el usuario podrá hacer revisión de los datos procesados en formato xml. (*Haga una revisión en la sección de archivos de salida para visualizar el formato que estos archivos deberán tener).*

- **Ver Gráfico:** Deberá mostrar un gráfico (Queda a elección del estudiante si de Barras o Pie), los datos del XML de salida, por ejemplo:

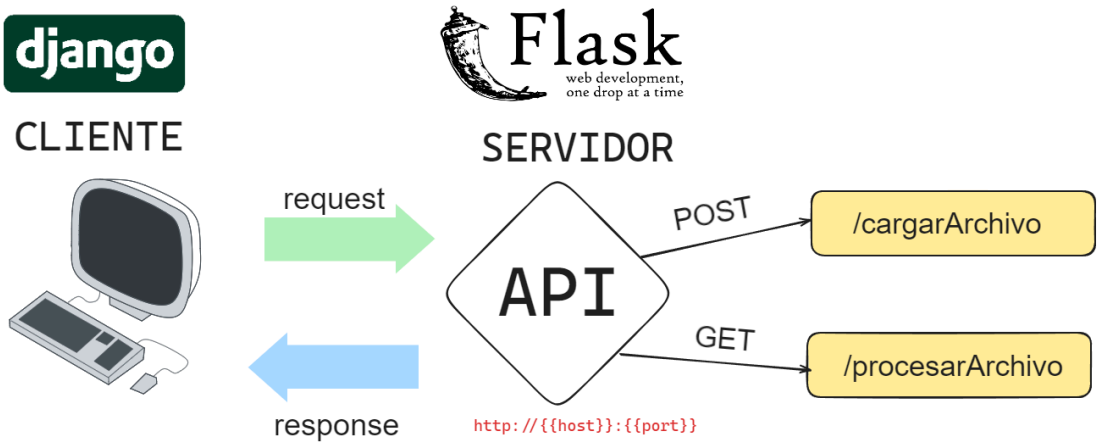


- **Datos del Estudiante:** Deberá de mostrar en frontend sus datos (nombre completo y carnet).



NOTA: Se proporcionan imágenes como sugerencia del frontend, queda siempre a <u>discreción del estudiante</u>.

Backend - Flask:
El backend consistirá en la API que hará uso de servicios HTTP, deberá de poder procesar los datos que reciba en formato xml.



Sugerencia para la arquitectura de la aplicación web.

NOTA: La cantidad de endpoints y nombre de estos queda a <u>discreción del estudiante</u>.

ARCHIVOS DE ENTRADA:

Super Autos GT solamente posee agencias dentro del país de Guatemala, por lo que nos interesa saber por departamento cuántas ventas han generado las sucursales de cada departamento. Por lo que un los archivos de entrada vendrán de esta manera:

```xml
<?xml version="1.0"? encoding="UTF-8"?>
<Resumen>
    <ListadoVentas>
        <Venta departamento="Guatemala">
            <Fecha>15/09/2024</Fecha>
        </Venta>
        <Venta departamento="Petén">
            <Fecha>15/09/2024</Fecha>
        </Venta>
        <Venta departamento="San Marcos">
            <Fecha>16/09/2024</Fecha>
        </Venta>
        <Venta departamento="Guatemala">
            <Fecha>16/09/2024</Fecha>
        </Venta>
    </ListadoVentas>
</Resumen>
```

Puede notarse que, los datos que ingresan a Super Autos GT vendrán desordenados, y se tendrá ‘n’ cantidad en total de ventas - donde ‘n’ es cualquier número entero positivo.
Tenga en cuenta que:

- Solamente se aceptarán las ventas de los 22 departamentos de la República. Si se encuentra otro, deberá omitirse.

ARCHIVO DE SALIDA:

El archivo de salida aparece cuando usted procesa los datos. En base a lo que ingrese, deberá mostrar en **formato xml** la cantidad total de ventas por departamento. Si un departamento no posee ventas (es decir, 0) **no deberán de mostrarse en el archivo de salida**. No es necesario que el archivo de salida se genere por aparte, si no que solamente pueda ser visible en la interfaz - por lo que es **obligatorio que sea legible y entendible dentro de esta**. Por ejemplo:

```xml
<?xml version="1.0"? encoding="UTF-8"?>
<resultados>
    <departamentos>
        <Guatemala>
            <cantidadVentas>15</cantidadVentas>
        </Guatemala>
        <SanMarcos>
            <cantidadVentas>10</cantidadVentas>
        </SanMarcos>
        <Quetzaltenango>
            <cantidadVentas>5</cantidadVentas>
        </Quetzaltenango>
        <Petén>
            <cantidadVentas>20</cantidadVentas>
        </Petén>
    </departamentos>
</resultados>
```

Note que solamente se muestran aquellos departamentos que poseen una cantidad de ventas mayor a 0.

Consideraciones:

- Se permite el uso de listas/estructuras nativas de Python.
- **Debe de hacer uso de la POO.**
- Para **derecho a calificación deberá de tener Frontend y Backend.**
- Debe realizar un **diagrama de clases** sobre la solución implementada.
- El frontend deberá de ser realizado en Django.
- El backend deberá ser realizado en Flask.

- Queda siempre a discreción del estudiante el estilo del frontend y la cantidad de endpoints a utilizar, no es obligatorio implementar exactamente las sugerencias que se le muestran en el enunciado.
- La fecha de entrega debe ser **17/10/2024.**
- **Para mostrar las gráficas en el frontend se permite el uso de ChartJS, el uso de Javascript para el procesamiento de datos o comunicación de frontend/backend no está permitido - debe realizarse con Python.**
- Se entrega el **link del repositorio** de su solución. Debe de agregar al auxiliar correspondiente para poder tener derecho a su calificación.
- No se permiten copias parciales o totales de la práctica.
- Nombre del Repositorio **IPC2_Practica3_carnet**.