PROYECTO 3 – IPC2

Estudiante

201115455 – José Ernesto Pajoc Raymundo.

Resumen

El proyecto se enfocó en el manejo de archivos XML y el uso de frameworks para la creación de API´s Rest en el entorno de Python, se utilizó dos servidores para alojar a Flask y Django donde cada servicio funcionó de manera independiente.

El usuario es el encargado de seleccionar el archivo para cargar los reportes, este debe cumplir con la estructura definida para el funcionamiento adecuado del programa, si la información es correcta el programa mostrará en el navegador la información que se procesará.

Palabras clave

- Ruta
- Expresión
- Regular
- Servicio
- vista

Abstract

The project focused on the handling of XML files and the use of frameworks for the creation of Rest APIs in the Python environment, two servers were used to host Flask and Django where each service worked independently.

The user is in charge of selecting the file to load the reports, it must comply with the structure defined for the proper operation of the program. If the information is correct, the program will display the information to be processed in the browser.

Keywords

- *Path*
- *phrase*
- *Regular*
- *service*
- *View*

Introducción

El programa que corresponde al proyecto 3 es una aplicación desarrollada en el lenguaje de programación Python, el cual cuenta con una página web inicial desarrollada en Django, en él se puede cargar un archivo con extensión XML, el cual luego puede ser enviado a un servidor el cual se encargará de procesar los datos con Flask, para luego enviar una respuesta al usuario y poder filtrar la información que fue cargada en el XML y analizarla fácilmente.

Desarrollo del software

El lenguaje utilizado para desarrollar el software es Pyhton en su versión 3.8.1 y el editor de texto utilizado es Visual Studio Code 1.54.1

La finalidad del programa es crear una herramienta que permita cargar un archivo XML el cual contará con información recopilada sobre errores producidos por usuarios los cuales afectan a otros usuarios, donde se detalla la fecha y el tipo de error producido, para que el cliente pueda cargar su archivo se utilizó el Framework Django en el cual se creo plantillas que facilitan el uso de la App e interacción con el servidor, los datos son enviados a un segundo servidor a través de Json para trasladar la información entre los servicios, en este segundo servidor se utilizó el Framework de Flask, donde se crearon diversas rutas o endpoint's para cumplir varias tareas como el procesado de datos, visualización de documentación, reinicio de la App y las filtraciones solicitadas por el usuario en el servidor 1.

Para crear la app se divido las tareas en dos servidores, el primero encargado del frontend lo cual es la interacción con los usuarios y el segundo se encarga del backend el cual almacena la información y la procesa.

A continuación, se muestran las vistas y rutas utilizadas en el servidor 1.

Tabla I.

Rutas utilizadas en Django

Ruta	Función
/inicio/	Esta es la ruta inicial en la cual se puede cargar el archivo XML, visualizar los datos de salida y la ayuda.
/ayuda/	Al ingresar a la ruta se muestran los datos del estudiante como también un botón el cual permite ver este documento en formato PDF.

Fuente: elaboración propia.

Tabla II.

Vistas utilizadas en Django

NOMBRE DE LA VISTA	FUNCIÓN
Inicio	Permite renderizar la plantilla llamada index.html la cual es la primera página que visualiza el usuario.
Ayuda	Permite renderizar la plantilla llamada ayuda.html la muestra los datos del estudiante y la documentación.

Fuente: elaboración propia.

Para unir ambos Framework's se utilizó JavaScript con la función de fetch.

Tabla III.

Rutas utilizadas en Flask

RUTA	FUNCIÓN
/	Muestra una respuesta Json indicando que el servidor se encuentra funcionando correctamente.
/recibirXML	Recibe en formato Json los datos del archivo XML enviado desde el frontend, para crear una nueva lista donde se separará por eventos de los errores y luego se procesará cada uno.
/verDocumentacion	Llama a la función verDocumentacion la cual se encuentra en el archivo procesoEvento.py
/reiniciarApp	Llama a la función reiniciar la cual se encuentra en el archivo procesoEvento.py
/filtrar/<string:tipo>	La ruta permite obtener un parámetro de la url la cual es utilizada para poder realizar una filtración de los datos según lo solicitado por el usuario.

Fuente: elaboración propia.

El segundo servidor el cual utiliza Flask cuenta con varios archivos, para organizar el código y facilitar la comprensión del mismo, entre los archivos se encuentra el proceso de los eventos y otro el cual crea una clase llamada evento.

A continuación, se muestra una tabla donde se especifica las funciones utilizada dentro del archivo llamado procesoEvento.py

Tabla IV.

Funciones del archivo procesoEvento.py

NOMBRE	FINALIDAD
recibirDatos	Almacena los datos dentro de una lista para luego ser recorrida y así analizar cada elemento a través de expresiones regulares obteniendo los datos fundamentales necesarios para crear objetos de la clase evento.
Reiniciar	Elimina la información de la lista de eventos si esta posee datos almacenados.
crearXML	Esta función crea un nuevo archivo con extensión XML la cual posee los datos que fueron procesador por la App, estos datos son enviados de nuevo al usuario para que los pueda visualizar en la página de inicio del frontend.
verDocumentación	Abre el archivo PDF de la documentación para que el usuario lo pueda visualizar.

Fuente: elaboración propia.

Existe un último archivo llamado eventos.py la cual crea una clase de tipo evento con los atributos de fecha, reportado por, usuarios afectados y tipo de error. Con ello cada evento se transforma en un objeto el cual es almacenado en una lista.

Con lo anterior se logra cargar, procesar y mostrar los eventos ocurridos en los errores producidos por los

usuarios a través de dos servicios independientes alojados en distintos servidores.

El sistema operativo que se utilizo es Windows 10 y el instalador de paquetes de Python fue pip para poder hacer uso de webbrowser, io.

Conclusiones

El lenguaje de Python es muy sencillo de utilizar y entender comparado con otros lenguajes de programación de alto nivel, lo cual permite enfocarse en la solución del problema.

Los Frameworks facilitan la creación API´s para poder crear software en la web y proporcionar servicios desde diversos dispositivos.

Al hacer uso de las listas de Python se utiliza únicamente la memoria necesaria en cada evento y no se desperdicia hardware lo cual permite mayor rendimiento en el equipo.

Referencias bibliográficas

(s.f.). Obtenido de http://facundoq.github.io/courses/images/res/03_numpy.html

Programador clic. (s.f.). Obtenido de https://programmerclick.com/article/8282763126/

Severance, C. (2009). *Python para informáticos.* Michigan.

Anexos