Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2021.

PROYECTO 3

201903873 – Joaquin Emmanuel Aldair Coromac Huezo

Resumen

El enunciado nos introduce al problema de una empresa que requiere un software que pueda ser consumido desde Internet como un servicio. Este software recibirá un mensaje de la bitácora del software principal de la compañía y producirá una serie de información estadística relacionada.

Para darle solución al software se utilizó el framework Django para el Frontend donde el usuario podrá visualizar información y enviar eventos ya que será consumida por una api en Flask conectada mediante métodos http.

Para el manejo de Datos entre los 2 servicios se utilizan archivos XML, que es donde vienen los datos requeridos de la empresa para poder procesarlos y mostrarlos de la forma deseada.

Palabras clave

>API > POST
> GET > Framework > Frontend.

Abstract

The statement introduces us to the problem of a company that requires software that can be consumed from the Internet as a service. This software will receive a message from the company's main software log and will produce a series of related statistical information.

To give solution to the software the Django framework was used for the Frontend where the user will be able to visualize information and send events since it will be consumed by an API in Flask connected through http methods.

For data management between the 2 services XML files are used, which is where the required data from the company comes to be processed and displayed as desired.

Keywords

> API > POST
> GET > Framework > Frontend.

Introducción

Para la solución del software requerido se utilizó el framework Django para realizar el servicio1, en el Frontend del programa se pueden visualizar los datos y enviar eventos para que en el servicio2 se procesen y almacenen los datos enviados.

Los métodos HTTP utilizados fueron GET, para poder enviar los datos procesados de vuelta al servicio1 y POST, que nos sirve para enviar los datos desde el servicio1 al servicio2 para que en este sean almacenados y procesados los archivos XML enviados.

Para el manejo de datos se utilizaron archivos XML y para poder verificar que estos sean correctos se utilizaron expresiones regulares para verificar que el programa pueda trabajar con estos y no tenga problema en procesar los datos requeridos.

Desarrollo del tema

Lógica Programa

Django

Django es un framework web de alto nivel que permite el desarrollo rápido de sitios web seguros y mantenibles. Desarrollado por programadores experimentados, Django se encarga de gran parte de las complicaciones del desarrollo web, por lo que puedes concentrarte en escribir tu aplicación sin necesidad de reinventar la rueda. Es gratuito y de código abierto, tiene una comunidad próspera y activa, una gran documentación y muchas opciones de soporte gratuito y de pago.

Flask

Flask es un micro framework utilizado para crear aplicaciones web. Se considera micro porque es un proyecto minimalista y no depende de muchas cosas para ser utilizado, lo que significa que como desarrollador debes encargarte de la base de datos que utilizarás, así como cualquier otra herramienta.

El mensaje de bitácora tendrá la siguiente información.

```
FECHA: dd/mm/yyyy
USUARIO: correo electrónico del usuario que genera el error
AFECTADO: lista de correos electrónicos separados por coma
ERROR: código numérico: descripción del error
```

El archivo de entrada tendrá la siguiente sintaxis

```
<EVENTOS>
  <EVENTO>
    Guatemala, 15/01/2021
    Reportado por: <"Nombre Empleado 1" xx@ing.usac.edu.gt>
    Usuarios afectados: aa@ing.usac.edu.gt, <bb@ing.usac.edu.gt>
    Error: 20001 - Desbordamiento de búfer de memoria RAM
    en el servidor de correo electrónico.
  </EVENTO>
  ...
</EVENTOS>
```

El archivo de salida tendrá la siguiente sintaxis

```
<ESTADISTICAS>
  <ESTADISTICA>
    <FECHA> 15/01/2021 </FECHA>
    <CANTIDAD_MENSAJES> 3 </CANTIDAD_MENSAJES>
    <REPORTADO_POR>
      <USUARIO>
        <EMAIL> xx@ing.usac.edu.gt </EMAIL>
        <CANTIDAD_MENSAJES> 1 </CANTIDAD_MENSAJES>
      </USUARIO>
      <USUARIO>
        <EMAIL> yy@ing.usac.edu.gt </EMAIL>
        <CANTIDAD_MENSAJES> 2 </CANTIDAD_MENSAJES>
      </USUARIO>
    </REPORTADO_POR>
    <AFECTADOS>
      <AFECTADO> aa@ing.usac.edu.gt </AFECTADO>
      <AFECTADO> bb@ing.usac.edu.gt </AFECTADO>
      ...
    </AFECTADOS>
    <ERRORES>
      <ERROR>
        <CODIGO> 20001 </CODIGO>
        <CANTIDAD_MENSAJES> 2 </CANTIDAD_MENSAJES>
      </ERROR>
      <ERROR>
        <CODIGO> 20002 </CODIGO>
        <CANTIDAD_MENSAJES> 1 </CANTIDAD_MENSAJES>
```

Especificaciones

Especificaciones de Editor de Texto, versión Python y Graphviz

Django

Flask

Visual Studio Code

python.3.8.1

Graphviz 2.46.0

Especificaciones de computadora

Procesador: Intel(R) Core™ i7-9750H CPU@

2.60 GHz 2.59GHz

RAM instalada: 16.0GB (15.9 GB usable)

Tipo de sistema: Sistema operativo de 64 bits,

Procesador basado en x64

Windows 10.

Control de versiones

Para el proyecto se trabajó con GitHub para controlar las versiones durante su realización.

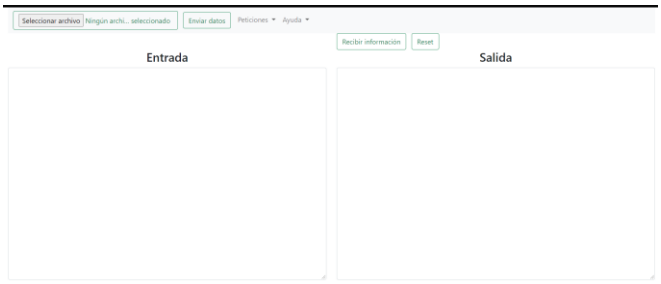
Enlace Repositorio.

https://github.com/jeach27/IPC2_Proyecto3_201903873.git

Forma de Uso Programa

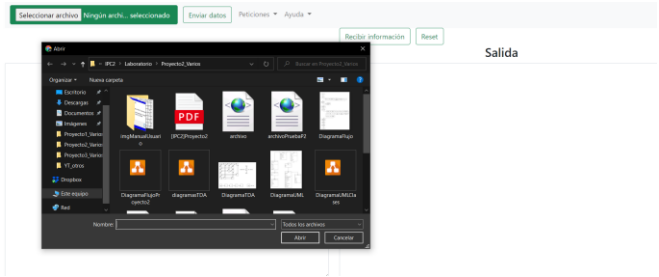
Principal

En la pantalla principal se pueden visualizar los datos en las áreas de texto de ENTRADA y SALIDA, así como poder seleccionar archivos y enviar eventos.



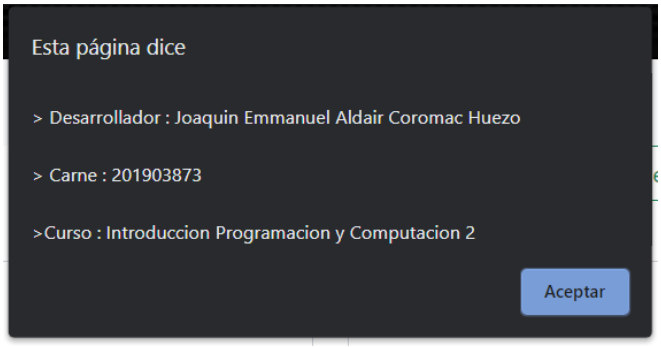
Selección de Archivo

Al presionar el botón de selección de archivos se abre una ventana para poder elegir el archivo deseado.



Visualización de Datos del estudiante

Mediante una alerta se muestra los datos del estudiante desarrollador.



Conclusiones

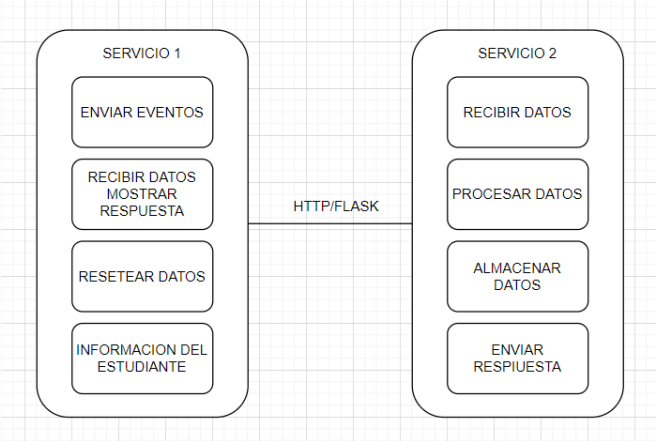
- El framework Django es de gran utilidad para realizar software consumidos desde internet.
- Para la comunicación entre el Frontend y Backend se utilizan de manera regular los métodos http.
- Para el manejo de datos el uso de archivos XML es una forma sencilla para hacer uso de estos.
-

Anexos

Métodos HTTP utilizados en la Api en Flask

```python
from flask import Flask, Response, request
from flask_cors import CORS


app = Flask(__name__)
cors = CORS(app, resources={r"/*": {"origin": "*"}})

@app.route('/events/', methods=['POST'])
def post_events():
    data = open('data.xml', 'w+')
    data.write(request.data.decode('utf-8'))
    data.close()

    return Response(response=request.data.decode('utf-8'),
                    mimetype='text/plain', content_type='text/plain')

@app.route('/events/', methods=['GET'])
def get_events():
    data = open('data.xml', 'r+')
    return Response(response=data.read(),
                    mimetype='text/plain', content_type='text/plain')


if __name__=='__main__':
    app.run(debug=True,port=5000)
```

Arquitectura de la aplicación



Referencias bibliográficas

9.3. Programación orientada a objetos — Materiales del entrenamiento de programación en Python - Nivel básico. (s. f.). Materiales del entrenamiento de programación en Python - Nivel básico. Recuperado 24 de febrero de 2021, de https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion9/poo.html

GeeksforGeeks. (s. f.). *Parse XML using Minidom in Python*. Recuperado 1 de marzo de 2021, de https://www.geeksforgeeks.org/parse-xml-using-minidom-in-python/

graphviz. (2020, 24 diciembre). PyPI. https://pypi.org/project/graphviz/

Xhunik Miguel. (2021, 9 abril). *Conectar 2 aplicaciones en el front-end con Django y Flask | Tutorial | Python | Flask y Django*. YouTube. https://www.youtube.com/watch?v=QoEbNGC0OYk&t=454s