PROYECTO III IPC2 APLICANDO PROGRAMACION WEB

201700603-Selvin Orlando Hernández Yumán

Resumen

Python es un lenguaje de programación de propósito general, cuya expansión y popularidad es relativamente reciente, brinda simplicidad, versatilidad y rapidez de desarrollo.
Es un lenguaje de scripting, independiente de plataforma y orientado a objetos, preparado para realizar cualquier tipo de programa, desde aplicaciones Windows a servidores de red o incluso, páginas web. Es un lenguaje interpretado, lo que significa que no se necesita compilar el código fuente para poder ejecutarlo, lo que ofrece ventajas como la rapidez de desarrollo e inconvenientes como una menor velocidad.
El paradigma de programación orientada a objetos POO y en el que se mencionan conceptos puntuales que nos permitirán comprender de qué forma es que un objeto o una idea abstracta puede plasmarse en un diagrama, así como en la forma que estos pueden relacionarse con otros objetos o ideas, también se describen algunos otros elementos en la cual permitirán comprender conceptos como lo son los tipos de relaciones, la Cardinalidad, encapsulamiento, etc.

Palabras clave
Programación
Web
Vistas
HTML

Abstract

Python is a general purpose programming language, whose expansion and popularity is relatively recent, providing simplicity, versatility, and speed of development.
It is a platform-independent, object-oriented scripting language prepared to perform any type of program, from Windows applications to network servers to even web pages. It's an interpreted language, which means you don't need to compile your source code to run it, which offers advantages such as rapid development and drawbacks such as lower speed.
The paradigm of programming OO objects and in which point concepts are mentioned that will allow us to understand how an abstract object or idea can be embodied in a diagram, as well as in the way that these can relate to other objects or ideas, also describes some other elements in which they will allow us to understand concepts such as the types of relationships , Cardinality, encapsulation.

Keywords

Programming

Web

Views

HTML

Introducción

A partir de la funcionalidad que la programación orientada a objetos (POO) se procede a dar solución a la simulación del alojamiento de objetos dentro de una base de datos a partir de la frecuencia de entrada en distintos sitios que se obtienen mediante su matriz de frecuencia y posteriormente realizar una serie de procesos y conversiones para la manipulación de la información que poseen los archivos de entrada

Otras de las bases fundamentales para la realización de este proyecto es la estructura XML el cual no es más que es un lenguaje de marcado que define un conjunto de reglas para la codificación de documentos.

Desarrollo del tema

En esta ocacion es necesario conocer los conceptos como servicio web, verbos html, API entre otros.

Un web service facilita un servicio a través de Internet: se trata de una interfaz mediante la que dos máquinas (o aplicaciones) se comunican entre sí. Esta tecnología se caracteriza por estos dos rasgos:

Multiplataforma: cliente y servidor no tienen por qué contar con la misma configuración para comunicarse. El servicio web se encarga de hacerlo posible.

Distribuida: por lo general, un servicio web no está disponible para un único cliente, sino que son diferentes los que acceden a él a través de Internet.

Cuando se utiliza un web service, un cliente manda una solicitud a un servidor, desencadenando una acción por parte de este. A continuación, el servidor devuelve una respuesta al cliente.

Un cliente es un ordenador o software que accede a un servidor y recupera servicios especiales o datos de él. Es tarea del cliente estandarizar las solicitudes, transmitirlas al servidor y procesar los datos obtenidos para que puedan visualizarse en un dispositivo de salida como una pantalla.

API es una abreviatura de Application Programming Interfaces, que en español significa interfaz de programación de aplicaciones. Se trata de un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, permitiendo la comunicación entre dos aplicaciones de software a través de un conjunto de reglas.

El proyecto ha sido desarrollado en dos partes una para el front end y otra para el backend, la parte de frontend se ha desarrollado mediante El Framework Django, ésta a su vez se compone de aplicaciones, el nombre de la aplicación utilizada para el frontend es “app”, dentro de ella encontramos las siguientes claces:

1. __init__
2. Admin
3. Apps
4. Models
5. Tests
6. Urls
7. Views

Estas son creadas por defecto dentro de la carpeta de la aplicación, para el desarrollo del frontend del proyecto únicamente se hizo uso de las clases “urls” y “views”

La clase “URLS” es el modulo que contiene configuración de las rutas donde se van a acceder desde el navegador, y esta a su vez invocaran un método que nos renderizará la pagina html que hayamos configurado previamente.

En la clase “VIEWS” encontraremos los métodos que verificaran el tipo de solicitud que hacemos desde el navegador y este se encargara de renderizar el html y los proceso que nosotros hayamos configurado, por ejemplo, si desde una url nosotros realizamos una petición POST este será verificado en un método desde views.py y dependiendo de lo que se haya programado este devolverá un elemento como respuesta

El Backend fue desarrollado a través del Framework Flask, esta parte se compone de las clases:

1. Main
2. crearEventos

a continuación, se describe el funcionamiento de las clases antes mencionadas

Clase Main

Como su nombre lo indica esta es la clase principal en donde encontraremos la configuración de las rutas (endponits) de nuestra api) que se consumirá con el frontend que previamente hemos desarrollado, aquí se tendrá contenida las direcciones como subir un xml o realizar operaciones sobre el xml que previamente se subio

XML es el acrónimo de Extensible Markup Language, el lenguaje de marcado es un conjunto de códigos que se pueden aplicar en el análisis de datos o la lectura de textos creados por computadoras o personas. El lenguaje XML proporciona una plataforma para definir elementos para crear un formato y generar un lenguaje personalizado.

Un archivo XML se divide en dos partes: prolog y body. La parte prolog consiste en metadatos administrativos, como declaración XML, instrucción de procesamiento opcional, declaración de tipo de documento y comentarios. La parte del body se compone de dos partes: estructural y de contenido (presente en los textos simples).

El diseño XML se centra en la simplicidad, la generalidad y la facilidad de uso y, por lo tanto, se utiliza para varios servicios web. Tanto es así que hay sistemas destinados a ayudar en la definición de lenguajes basados en XML, así como APIs que ayudan en el procesamiento de datos XML – que no deben confundirse con HTML.

.

Glosario

HTML:
Lenguaje Marcado de Híper-Texto por sus siglas en inglés, es una especie de lenguaje que los navegadores web interpretan

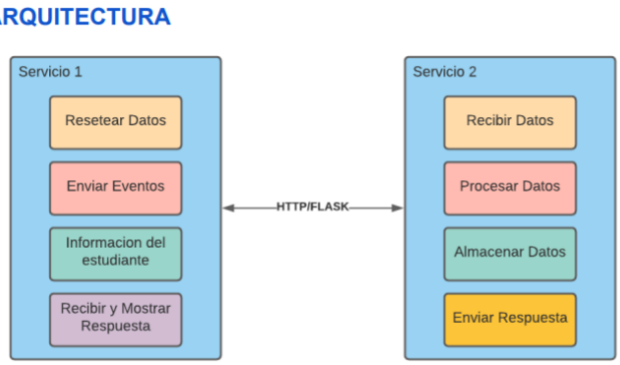


Figura 1. Arquitectura de la Aplicacion Fuente: Enunciado proyecto3

Conclusiones

Las estructuras de Datos permiten la manipulación de la memoria acorde a las necesidades del programador, a pesar de que las limitaciones de memoria en la actualidad son un tema que ya casi está superado no está demás mantener la funcionalidad optima de un programa así poder gestionar los recursos adecuadamente ya que al momento de realizar operaciones con una gran cantidad de datos es donde se pone a prueba que tan optimo es la gestión de nuestro trabajo.

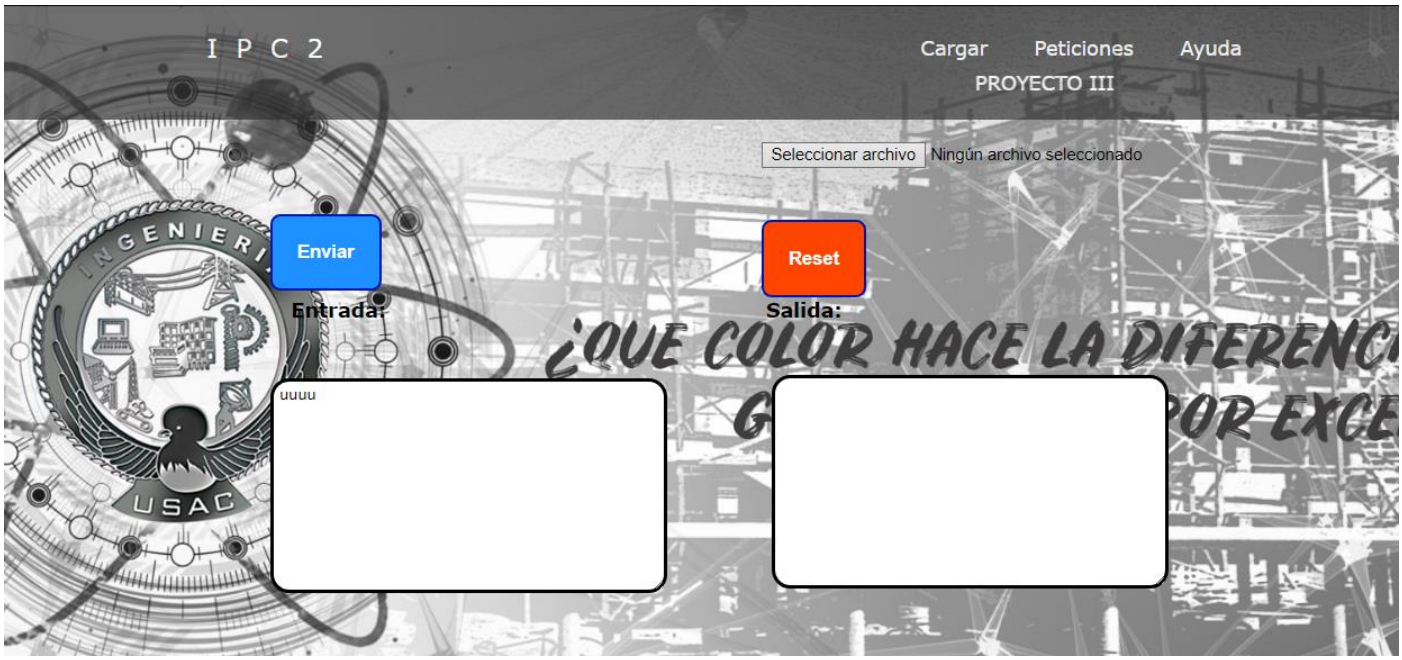
Referencias bibliográficas

Anexos

Clase main API

```
from products import products

@app.route('/ping')
def ping():
    return jsonify({'message':'pong'})

@app.route('/products')
def getProductos():
    return jsonify(products)
@app.route('/subir',methods=['POST'])
def subir():
    if request.method== 'POST':
        requestData=request.get_json()
        content=requestData['archivo']
        nArchivo=createEvents(content)



        return jsonify({'message': 'received'})



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

Imagen Frontend



Clase views del Frontend

```python
from django.shortcuts import render
import requests
import json

def index(request):

    if request.method == 'POST':
        #print("es post")
        #entrada= request.POST.get('contenido')
        archivo=request.FILES['archivo']
        file=str(archivo.read())
        print(file)
        fichero = {'archivo':file}
        response = requests.post('http://127.0.0.1:5000/subir',json=fichero)

        return render(request,'app/index.html')

    else:
        return render(request,'app/index.html')
```