Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la Programación y Computación 2, Segundo Semestre 2023.

Implementación De Una API Utilizando El Protocolo HTTP

2022201524 – Carlos Manuel Lima y Lima

Resumen

El proyecto se centra en la creación de una API que ofrece servicios a través del Protocolo HTTP, utilizando la metodología de programación orientada a objetos (POO) y empleando el lenguaje de programación Python como su base. Para obtener la información necesaria y atender las solicitudes, el proyecto procesa archivos XML, lo que le permite extraer datos de manera eficiente.
La arquitectura del proyecto consta de un backend construido con Flask y un frontend desarrollado con Django, lo que brinda un enfoque completo y versátil para la implementación de la aplicación. Además de proporcionar servicios, la aplicación tiene la capacidad de generar gráficos mediante la herramienta Graphviz y elaborar informes en formato XML, lo que amplía sus capacidades de visualización y presentación de datos.
Un aspecto crucial del proyecto es su simulación de una base de datos, donde se almacena la información extraída de los archivos XML. Esto se hace para garantizar la integridad y disponibilidad de los datos, evitando pérdidas o corrupción. En conjunto, el proyecto combina la potencia de Python, la estructura de POO y tecnologías avanzadas para ofrecer una solución completa y robusta para la manipulación de datos, visualización y generación de informes.

Palabras Clave

- Aplicación
- Servicio
- Conexión
- Programación
- Bases De Datos

Abstract

The project focuses on the creation of an API that offers services through the HTTP Protocol, using the Object Oriented Programming (OOP) methodology and using the Python programming language as its base. To obtain the necessary information and fulfill requests, the project processes XML files, which allows it to extract data efficiently.

The project architecture consists of a backend built with Flask and a frontend developed with Django, which provides a complete and versatile approach to the implementation of the application. In addition to providing services, the application has the ability to generate graphs using the Graphviz tool and produce reports in XML format, which extends its data visualization and presentation capabilities.

A crucial aspect of the project is its simulation of a database, where the information extracted from the XML files is stored. This is done to ensure data integrity and availability, avoiding loss or corruption. Overall, the project combines the power of Python, the structure of OOP and advanced technologies to offer a complete and robust solution for data manipulation, visualization and reporting.

Keywords

- *Application*
- *Service*
- *Connection*
- *Programming*
- *Databases*

Introducción

En este proyecto se destaca la implementación de una API que utiliza el Protocolo HTTP como medio para brindar servicios, con un sólido fundamento en la programación orientada a objetos (POO) y desarrollada en el lenguaje de programación Python. Este proyecto se enfoca en la manipulación de archivos XML para la obtención de datos, los cuales son esenciales para satisfacer las diversas solicitudes de los usuarios.

La aplicación se compone de un backend construido con Flask y un frontend desarrollado con Django, lo que garantiza una experiencia completa y versátil para los usuarios. Además de ofrecer servicios, el proyecto tiene la capacidad de generar gráficos mediante Graphviz y crear informes en formato XML, brindando opciones adicionales para visualización y análisis de datos.

Un componente fundamental de este proyecto es la simulación de una base de datos, donde se almacena la información extraída de los archivos XML. Esto se hace con el propósito de asegurar la integridad y disponibilidad de los datos, evitando pérdidas o corrupción. En conjunto, este proyecto representa una solución completa y robusta que combina la potencia de Python, la estructura de POO y tecnologías avanzadas para la manipulación de datos, generación de informes y visualización.

1. Python

Python es un lenguaje de programación de propósito general que es compatible con una variedad de arquitecturas de sistemas y tiene aplicaciones versátiles en diversos campos, desde la creación de sitios web hasta el desarrollo de soluciones de aprendizaje automático. Su capacidad para adaptarse a diferentes áreas y su facilidad de uso lo convierten en uno de los lenguajes de programación más extendidos.

Las personas pueden usar y distribuir el código fuente del mismo de forma gratuita, incluso con fines comerciales. Hoy en día, cualquier persona con una computadora y una fuerte voluntad puede aprender a programar en este lenguaje. (Weisheim, 2023)

Adicionalmente a su facilidad de aprendizaje, su popularidad también deriva de su gran versatilidad. Este lenguaje encuentra aplicaciones en una amplia gama de campos, que incluyen la ciencia de datos, el desarrollo web y el aprendizaje automático. Además, su naturaleza multiplataforma permite que se ejecute en diversos sistemas operativos como Windows, Linux y macOS.

2. Programación Orientada a Objetos (POO)

La programación orientada a objetos representa un paradigma de programación en el cual la estructuración del software se fundamenta en torno a objetos o datos, en contraposición al uso de funciones y lógica. Su enfoque radica en los objetos que los programadores requieren manipular, en vez de centrarse en la lógica necesaria para llevar a cabo dicha manipulación. Un objeto se conceptualiza como un conjunto de atributos y comportamientos distintivos. (Universidad Europea, 2022)

2.1. Implementación de POO en Python

Python es un lenguaje multiparadigma: soporta la programación imperativa y funcional, pero también la programación orientada a objetos. En Python todo es un objeto. Cuando se crea una variable y se asigna un valor entero, ese valor es un objeto, lo mismo con las listas, conjuntos, tuplas, cadena de caracteres, etc.

Implementar POO en Python implica utilizar las características y sintaxis que ofrece dicho lenguaje.

- Utiliza la palabra reservada "class" para crear definiciones de clases, incluyendo atributos y métodos en su interior.

- Incorpora herencia al establecer la clase base entre paréntesis durante la creación de una nueva clase.

- Aprovecha la característica de herencia múltiple y el concepto de polimorfismo de forma orgánica, dado que Python posibilita estas prácticas.

3. API

Las API (Interfaz De Programación De Aplicaciones) son interfaces compuestas por reglas y llamados en lenguaje computacional, que sirven para programar el funcionamiento de una aplicación determinada dentro de un software.

Estas herramientas pueden extraer formatos predefinidos, información de una base de datos o partes de un programa a otro.

3.1 ¿Qué es un punto de terminación de API?

Al ser un canal que conecta la información con una herramienta que la trabajará, una API tiene un destino al cual debe dirigir los datos que ha extraído. A este programa de destino se le conoce como «punto de terminación de API» y puede tener la forma de una aplicación móvil, un sitio web o un programa de gestión de datos.

3.2 ¿Cómo usar una API?

Comprender el valor de una API en particular tiene que ver esencialmente con entender qué información hay disponible y cómo se puede acceder a ella. Para descubrir qué puede aportar una API en particular, es necesario identificar un proceso que se necesita simplificar o un aspecto del programa que puede mejorarse a partir de los desarrollos de otros programadores. (Coppola, 2022)

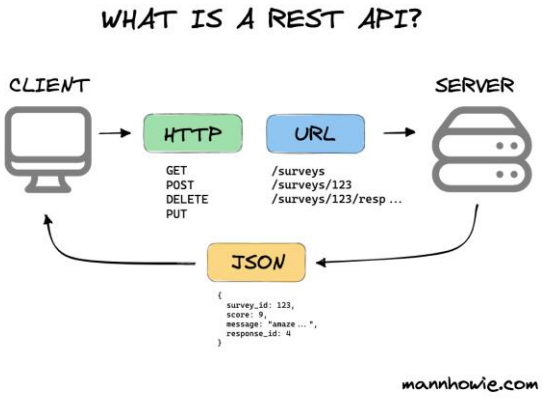

Figura 1. Estructura De Una API

4. Protocolo HTTP

El HTTP (Protocolo de Transferencia de Hiper Textos) es el protocolo de transmisión de información de la World Wide Web, es decir, el código que se establece para que el computador solicitante y el que contiene la información solicitada puedan "hablar" un mismo idioma a la hora de transmitir información por la red.

Con el HTTP se establecen criterios de sintaxis y semántica informática (forma y significado) para el establecimiento de la comunicación entre los diferentes elementos que constituyen la arquitectura web: servidores, clientes, proxies. (Concepto De, 2021)

4.1 ¿Cómo funciona el protocolo HTTP?

El funcionamiento del http se basa en un esquema de petición-respuesta entre el servidor web y el agente usuario o cliente que realiza la solicitud de transmisión de datos. Un cliente puede ser un explorador determinado, cuando intentamos abrir una página web, o los rastreadores web que las inspeccionan.

A ellos el servidor brinda una respuesta estructurada de modo puntual y dotada de una serie de metadatos, que establecen las pautas para el inicio, desarrollo y cierre de la transmisión de la información. Estos son los métodos de petición, es decir, los comandos que disparan la ejecución de recursos determinados, cuyos archivos residen en el servidor.

5. Base De Datos

Una base de datos es una herramienta que recopila datos, los organiza y los relaciona para que se pueda hacer una rápida búsqueda y recuperar con ayuda de un ordenador. Hoy en día, las bases de datos también sirven para desarrollar análisis. Las bases de datos más modernas tienen motores específicos para sacar informes de datos complejos. Cuando una empresa tiene una base de datos y quiere implementar un software, tiene que andarse con cuidado dependiendo del software que instala. (Tic Portal, 2022)

6. Graphviz

Graphviz constituye un software de código abierto de distribución gratuita orientado a la creación de representaciones gráficas, las cuales permiten visualizar datos estructurales en forma de diagramas. Su aplicabilidad es amplia y abarca campos tan diversos como el análisis de redes, la bioinformática, la ingeniería de software, las bases de datos, el diseño de páginas web y el aprendizaje automático. Además, proporciona interfaces gráficas diseñadas para diferentes dominios. (Takeyas, 2012)

El conjunto de herramientas Graphviz engloba una diversidad de programas diseñados para la creación de diagramas. Entre estas opciones, el lenguaje "dot" sobresale como la herramienta principal destinada a elaborar representaciones de estructuras jerárquicas o de capas dirigidas. Este lenguaje puede ser ejecutado como un programa desde la línea de comandos, a través de un servicio de visualización web o mediante una interfaz gráfica compatible. (Takeyas, 2012)

7. Archivos XML

XML consiste en un lenguaje de marcado creado por el W3C con la finalidad de definir una sintaxis para la codificación de documentos, que tanto los usuarios como las propias máquinas en sí puedan ser capaces de leer.

Para ello, lo hace mediante la utilización de una serie de etiquetas que definen la estructura que posee el documento en cuestión, además de cómo debe ser transportado y almacenado. (Pérez, 2020)

Podemos establecer una comparación con otro lenguaje de marcado con el cual es probable que estemos más familiarizados: el lenguaje de marcado de hipertexto (HTML), ampliamente empleado en la codificación de páginas web. HTML emplea una serie de símbolos de marcado predefinidos que delinean el formato del contenido de una página web.

8. Frontend

El frontend es la parte del desarrollo web que se dedica a la parte frontal de un sitio web, en pocas palabras del diseño de un sitio web, desde la estructura del sitio hasta los estilos como colores, fondos, tamaños hasta llegar a las animaciones y efectos.

Es esa parte de la página con la que interaccionan los usuarios de la misma, es todo el código que se ejecuta en el navegador de un usuario, al que se le denomina una aplicación cliente, es decir, todo lo que el visitante ve y experimenta de forma directa.

Un front-end, es la persona que se dedica básicamente al diseño web, pero esto no significa que no toque código, tanto el front-end como el back-end están en contacto con código todo el tiempo. (García, 2021)

9. Backend

Mientras que el frontend es la capa de programación ejecutada en el navegador del usuario, el backend procesa la información que alimentará el frontend de datos.

Es la capa de acceso a los datos, ya sea de un software o de un dispositivo en general, es la lógica tecnológica que hace que una página web funcione, lo que queda oculto a ojos del visitante.

El backend de una solución, determina qué tan bien se ejecutará la aplicación y qué experiencia, positiva o negativa, obtendrá el usuario de su uso.

Trabajar en este apartado supone algo totalmente diferente al frontend, ya que exige el dominio de otros términos de programación, lenguajes que requieren una lógica, ya que esta área es también la encargada de optimizar recursos, de la seguridad de un sitio y otros factores. (García, 2021)

Conclusiones

1. Integración de Tecnologías Avanzadas: El proyecto demuestra la capacidad de integrar tecnologías avanzadas en un solo sistema. Utiliza Python como lenguaje de programación, Flask para el backend y Django para el frontend, lo que proporciona una solución completa y versátil para el procesamiento de datos y la prestación de servicios web. Además, la inclusión de herramientas como Graphviz para generar gráficos y la generación de informes en formato XML agrega un valor significativo a la aplicación.

2. Enfoque en la Integridad de los Datos: La simulación de una base de datos para almacenar información extraída de archivos XML subraya la importancia de mantener la integridad de los datos. Esto garantiza que la información se conserve de manera segura y confiable, lo que es esencial en aplicaciones donde la precisión de los datos es fundamental.

3. Potencial para la Visualización y Análisis de Datos: La capacidad de generar gráficos con Graphviz y crear informes en formato XML abre la puerta a un potencial significativo para la visualización y el análisis de datos. Esto no solo mejora la capacidad de los usuarios para comprender los datos, sino que también brinda herramientas para tomar decisiones informadas basadas en la información procesada por el sistema.

4. Enfoque Estructurado y Orientado a Objetos: La adopción de la programación orientada a objetos (POO) como base estructural para el proyecto resalta la coherencia y la estructura lógica del código, lo que facilita su mantenimiento, escalabilidad y futuras expansiones del sistema de manera eficiente.

Bibliografía

Concepto De. (21 de Agosto de 2021). *HyperText Transfer Protocol*. Obtenido de Concepto De: https://concepto.de/http/

Coppola, M. E. (23 de Marzo de 2022). *¿Qué es una API? Definición, tipos y ejemplos*. Obtenido de HubSpot: https://blog.hubspot.es/website/que-como-usar-api

García, I. J. (20 de Marzo de 2021). *Backend y Frontend, ¿Qué es y cómo funcionan en la programación?* Obtenido de ServNet: https://www.servnet.mx/blog/backend-y-frontend-partes-fundamentales-de-la-programacion-de-una-aplicacion-web

Pérez, C. (18 de Diciembre de 2020). *Qué es y cómo abrir un archivo XML*. Obtenido de Muyinteresante: https://www.muyinteresante.es/tecnologia/23571.html

Tic Portal. (5 de Diciembre de 2022). *Base de datos*. Obtenido de Tic Portal: https://www.ticportal.es/glosario-tic/base-datos-database

Universidad Europea. (24 de Agosto de 2022). *Programación orientada a objetos*. Obtenido de Universidad Europea: https://universidadeuropea.com/blog/programacion-orientada-objetos/

Weisheim, R. (26 de Junio de 2023). *Qué es Python: conoce uno de los lenguajes de programación más populares*. Obtenido de Hostinger: https://www.hostinger.es/tutoriales/que-es-python

Anexos

1. Tabla de Actividades

Semana	Actividad
Primera Semana	Realización de la lectura de los archivos XML y guardar los datos en las listas.
Segunda Semana	Programación del backend, elaborando todos los enpoints necesarios y manejando las listas.
Tercera Semana	Realización de las gráficas en graphviz y los archivos de salida XML.
Cuarta Semana	Conexión con el Frontend con el Backend y diseño del HTML y el CSS.

Tabla 1. Cronograma de Actividades

Fuente: Elaboración Propia

2. Diagrama de Clases

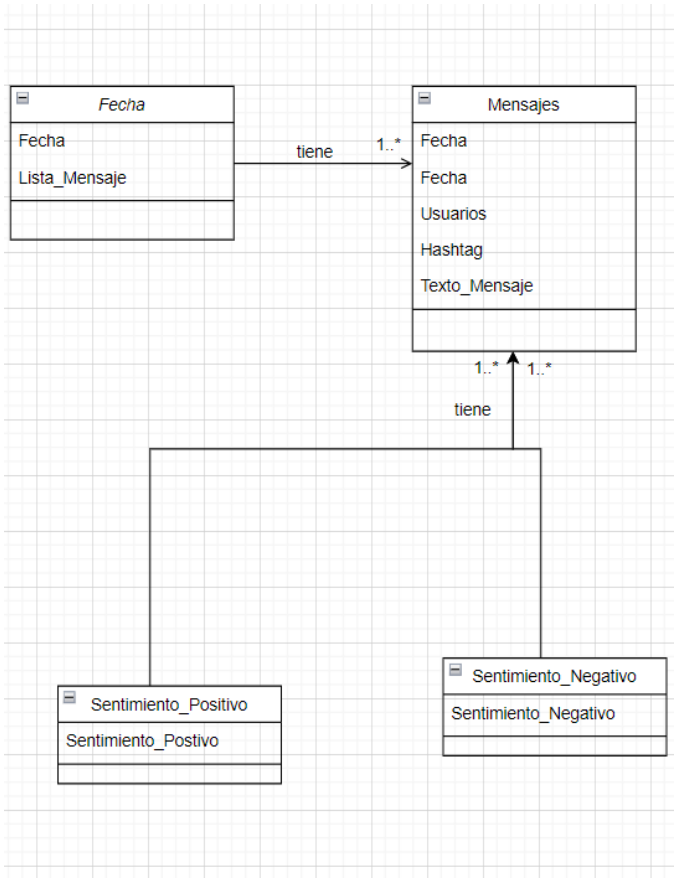

Figura 2. Diagrama de Clases

Fuente: Elaboración Propia

3. Diseño Frontend

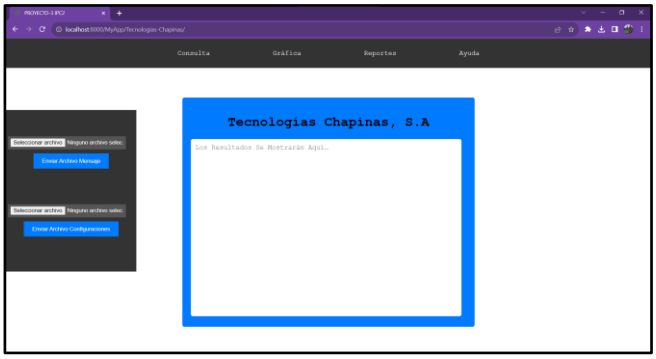

Figura 3. Diseño HTML

Fuente: Elaboración Propia