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 2023.

PROYECTO 3 – ANALIZADOR DE SENTIMIENTOS

202100033 – Josue Daniel Chavez Portillo

Resumen

El proyecto consiste en la creación de un programa capaz de brindar una solución al inconveniente planteado por la facultad de ingeniería de la universidad de san Carlos de Guatemala.

El programa desarrollado como primer paso, contiene la lectura de un archivo xml lectura de mensajes que contienen hashtags, usuarios y también un archivo de configuración el cual tiene que analizar las palabras positivas y negativas que contiene.

Para la elaboración del proyecto se hizo uso del lenguaje de programación Python, junto a las estructuras de programación secuenciales, cíclicas y condicionales. Para la parte de generar reporte, se hizo uso de la herramienta Panda, se implementó la manipulación de archivos XML utilizando Dom.Minidom y se implemento Flask para el backend del proyecto y Django para el frontend.

Palabras clave

- TDA
- Memoria Dinamica

Abstract

he project consists of the creation of a program capable of providing a solution to the problem posed by the engineering faculty of the University of San Carlos de Guatemala.

The program developed as a first step, contains the reading of an xml file reading messages containing hashtags, users and also a configuration file which has to analyze the positive and negative words it contains.

For the elaboration of the project, the Python programming language was used, together with sequential, cyclic and conditional programming structures. For the report generation part, the Panda tool was used, XML file manipulation was implemented using Dom.Minidom and Flask was implemented for the backend of the project and Django for the frontend.

Keywords

- *TDA*
- *Dynamic Memory*

Introducción

Para entender mejor la resolución del problema del programa debemos de entender diferentes conceptos

para poder interpretar lo programado, como son las sintaxis básicas del lenguaje utilizado el cual fue Python, también las estructuras de programación que se utilizaron, lo que es la programación orientada a objetos y como utilizarla de la manera correcta y los errores frecuentes que se tienen a la hora de aplicarla a un programa, el como generar un reporte en este caso haciendo uso de graphviz el cual es un conjunto de herramientas de software para el diseño de diagramas, las lecturas de archivos usando minidom el cual es una implementación mínima de la interfaz Document Object Model con una API similar a la de otros lenguajes. Y el tema mas importante y talvez complicado de entender el cual es el TDA (Tipo de Dato Abstracto) el cual es un modelo matemático compuesto por una colección de operaciones definidas sobre un conjunto de datos para el modelo

Python

Python es un lenguaje de programación de alto nivel, es sencillo de leer y escribir debido a su similitud con el lenguaje humano. Además de esto se trata de un lenguaje multiplataforma de código abierto, y por lo tanto es gratuito lo cual permite el desarrollo de software sin ningún tipo de limite.

Como se menciona Python es un lenguaje de programación multiplataforma, lo cual permite desarrollar aplicaciones en cualquier sistema operativo con una facilidad asombrosa. Una gran cantidad de tecnología acepta como lenguaje Python debido a su sencillez y a su gran potencia para el tratamiento de datos, algo que sin duda ha hecho resurgir este lenguaje a nivel laboral, donde cada vez son más los usuarios que migran hacia este lenguaje.

Con esta información se concluye el por qué Python es uno de los lenguajes de programación más demandados en el mundo laboral. Debido a su relación con algunos de los campos con mayor relevancia de la actualidad, como la IA, el Machine Learning o el análisis de datos, se necesitan un gran número de programadores expertos en Python para desarrollar nuevas y emocionantes funciones.

Programación Orientada a Objetos

La Programación Orientada a Objetos (POO) es un paradigma de programación, es decir, un modelo o un estilo de programación que nos da unas guías sobre cómo trabajar con él. Se basa en el concepto de clases y objetos. Este tipo de programación se utiliza para estructurar un programa de software en piezas simples y reutilizables de planos de código (clases) para crear instancias individuales de objetos.

“A lo largo de la historia, han ido apareciendo diferentes paradigmas de programación. Lenguajes secuenciales como COBOL o procedimentales como Basic o C, se centraban más en la lógica que en los datos. Otros más modernos como Java, C# y Python, utilizan paradigmas para definir los programas, siendo la Programación Orientada a Objetos la más popular.” (Canelo, 2020)

Con el paradigma de Programación Orientado a Objetos lo que buscamos es dejar de centrarnos en la lógica pura de los programas, para empezar a pensar en objetos, lo que constituye la base de este paradigma. Esto nos ayuda muchísimo en sistemas

grandes, ya que, en vez de pensar en funciones, pensamos en las relaciones o interacciones de los diferentes componentes del sistema.

Importancia del POO

La Programación Orientada a objetos permite que el código sea reutilizable, organizado y fácil de mantener. Sigue el principio de desarrollo de software utilizado por muchos programadores DRY (Don't Repeat Yourself), para evitar duplicar el código y crear de esta manera programas eficientes. Además, evita el acceso no deseado a los datos o la exposición de código propietario mediante la encapsulación y la abstracción, de la que hablaremos en detalle más adelante.

Clases, Objetos e Instancias

Una clase es una plantilla. Define de manera genérica cómo van a ser los objetos de un determinado tipo. Por ejemplo, una clase para representar a animales puede llamarse "animal" y tener una serie de atributos, como "nombre" o "edad" y una serie con los comportamientos que estos pueden tener, como caminar o comer, y que a su vez se implementan como métodos de la clase (funciones).

Un ejemplo sencillo de un objeto, como decíamos antes, podría ser un animal. Un animal tiene una edad, por lo que creamos un nuevo atributo de "edad" y, además, puede envejecer, por lo que definimos un nuevo método. Datos y lógica. Esto es lo que se define en muchos programas como la definición de una clase, que es la definición global y genérica de muchos objetos.

Principios del POO

Encapsulación: contiene toda la información importante de un objeto dentro del mismo y solo expone la información seleccionada al mundo exterior.

Abstracción: contiene toda la información importante de un objeto dentro del mismo y solo expone la información seleccionada al mundo exterior.

Herencia: define relaciones jerárquicas entre clases, de forma que atributos y métodos comunes puedan ser reutilizados. Las clases principales extienden atributos y comportamientos a las clases secundarias.

Polimorfismo: consiste en diseñar objetos para compartir comportamientos, lo que nos permite procesar objetos de diferentes maneras. Es la capacidad de presentar la misma interfaz para diferentes formas subyacentes o tipos de datos.

Alrededor de estos principios de la programación orientada a objetos se construyen muchas cosas. Por ejemplo, los Principios SOLID, o los Patrones de diseño.

Django

Django es un popular framework de desarrollo web de código abierto escrito en Python. Fue creado para facilitar la creación de aplicaciones web de manera eficiente, permitiendo a los desarrolladores centrarse en la lógica de la aplicación en lugar de tener que escribir código repetitivo para manejar tareas comunes en el desarrollo web.

Algunas de las características clave de Django incluyen:

Sistema de enrutamiento: Django proporciona un sistema de enrutamiento que permite definir cómo se manejan las URL de una aplicación web, lo que facilita la creación de rutas y vistas.

ORM (Object-Relational Mapping): Django incluye un ORM que permite a los desarrolladores interactuar con la base de datos utilizando objetos Python en lugar de escribir consultas SQL directamente.

Plantillas: Django utiliza un sistema de plantillas que permite separar la lógica de presentación de una aplicación web, lo que facilita la creación de interfaces de usuario dinámicas.

Administrador de Django: Ofrece una interfaz administrativa preconstruida que facilita la gestión de datos en la base de datos de la aplicación.

Autenticación y autorización: Django proporciona herramientas para gestionar la autenticación de usuarios y controlar el acceso a las vistas y datos de la aplicación.

Seguridad: Django incluye medidas de seguridad incorporadas para proteger contra amenazas comunes, como ataques de inyección de SQL y ataques CSRF (Cross-Site Request Forgery).

Internacionalización y localización: Soporta la creación de aplicaciones web multilingües.

Extensibilidad: Django es altamente extensible y permite a los desarrolladores agregar funcionalidades personalizadas a través de aplicaciones y complementos.

Flask

lask es un microframework web de código abierto para Python. A diferencia de Django, que es un framework más completo y estructurado, Flask se centra en ser ligero y flexible, proporcionando las herramientas necesarias para construir aplicaciones web de manera sencilla y rápida sin imponer una estructura rígida.

Las características principales de Flask incluyen:

Ligereza: Flask está diseñado para ser minimalista y se enfoca en proporcionar solo lo esencial para el desarrollo web. Esto significa que es fácil de aprender y usar, y es una excelente elección para aplicaciones web pequeñas a medianas.

Extensibilidad: Flask es altamente extensible, lo que significa que los desarrolladores pueden agregar funcionalidades adicionales a través de extensiones y bibliotecas de terceros. Esto permite personalizar el framework según las necesidades del proyecto.

Enrutamiento: Flask ofrece un sistema de enrutamiento que permite asociar URL con funciones Python, lo que facilita la definición de rutas y vistas en la aplicación.

Sin capa de base de datos: A diferencia de Django, Flask no incluye una capa de abstracción de base de datos (ORM) ni una base de datos por defecto. Los desarrolladores pueden elegir la base de datos que mejor se adapte a sus necesidades.

Jinja2: Flask utiliza el motor de plantillas Jinja2 para generar contenido HTML dinámico, lo que permite separar la lógica de presentación de la aplicación.

Flexibilidad: Los desarrolladores tienen la libertad de estructurar sus aplicaciones de la manera que deseen. Flask no impone una estructura específica de proyecto, lo que brinda flexibilidad pero también requiere una mayor toma de decisiones por parte del desarrollador.

Pandas

Pandas es una biblioteca de código abierto de Python ampliamente utilizada en ciencia de datos y análisis

de datos. Esta biblioteca proporciona estructuras de datos y herramientas para manipular y analizar datos de manera eficiente. Algunos de los componentes clave de Pandas incluyen:

DataFrames: Un DataFrame es una estructura de datos bidimensional similar a una tabla o hoja de cálculo, donde se pueden almacenar y manipular datos en filas y columnas. Los DataFrames son especialmente útiles para trabajar con datos tabulares, como conjuntos de datos CSV o hojas de cálculo Excel.

Series: Una Serie es una estructura de datos unidimensional que puede considerarse como una columna o una fila de un DataFrame. Cada Serie tiene un índice que permite etiquetar y acceder a los datos.

Manipulación de datos: Pandas ofrece numerosas funciones para realizar operaciones de limpieza, filtrado, selección, transformación y agregación de datos. Esto incluye la capacidad de eliminar valores nulos, cambiar tipos de datos, agrupar datos y realizar cálculos estadísticos.

Lectura y escritura de datos: Pandas admite la importación y exportación de datos desde y hacia una variedad de formatos, incluyendo CSV, Excel, bases de datos SQL, JSON y más.

Integración con NumPy: Pandas se integra bien con la biblioteca NumPy, lo que facilita la manipulación de datos numéricos y la realización de operaciones matemáticas en los datos.

Visualización de datos: Aunque Pandas en sí mismo no proporciona herramientas de visualización, se puede combinar con bibliotecas de visualización como Matplotlib o Seaborn para crear gráficos y visualizaciones a partir de los datos almacenados en DataFrames y Series.

Conclusiones

• La Programación Orientada a Objetos es actualmente el paradigma que más se utiliza para diseñar aplicaciones y programas informáticos.
• Python es un lenguaje fácil de aprender e ideal para aquellos programadores que se están iniciando

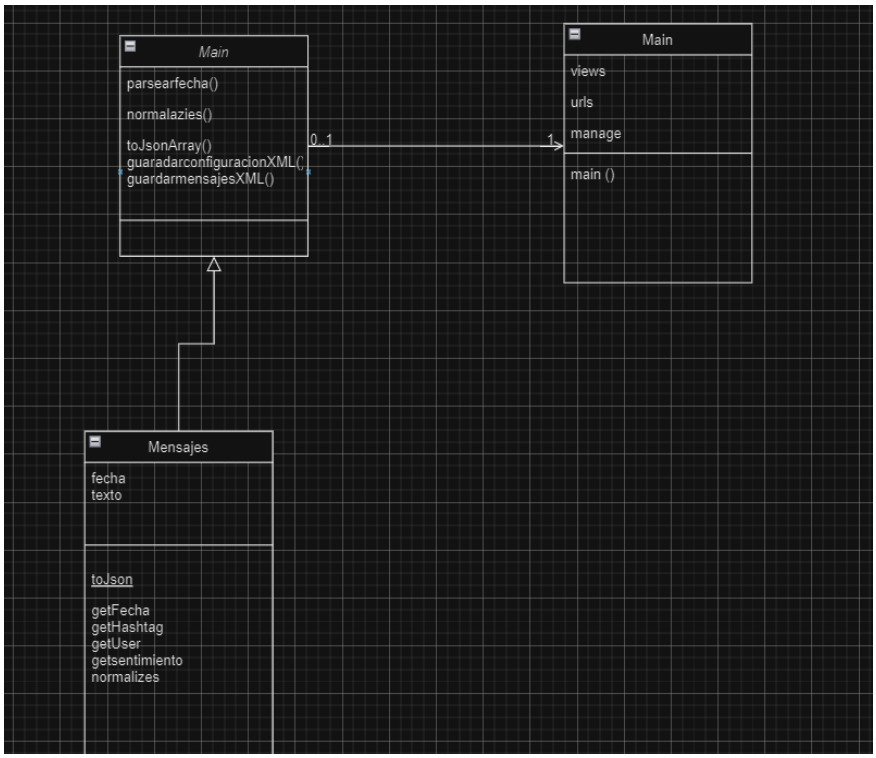
Referencias bibliográficas

- Autor: Vincent, W. S.
(2020). Django for Beginners: Build websites with Python and Django. Independently published.

Autor: Grinberg, M.
(2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.

Autor: McKinney, W.
(2017). Python for Data Analysis. O'Reilly Media.

Anexos


Main
parsearfecha()
normalazies()
toJsonArray()
guaradarconfiguracionXML()
guardarmensajesXML()
0..1
1
Main
views
urls
manage
main ()
Mensajes
fecha
texto
toJson
getFecha
getHashtag
getUser
getsentimiento
normalizes