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.

INTRODUCCIÓN A LA PROGRAMACIÓN Y COMPUTACIÓN 2
PROYECTO 3

201709149 – Percy Juventino Duarte Gálvez

Resumen

Se ha analizado las palabras clave de ciertos perfiles predefinidos dentro de la simulación de una red social ChapinChat así como de mensajes publicados por los distintos usuarios registrados dentro de un sitio creado a partir de servidores locales por Django y Flask. Se han calculado el peso de los perfiles en dicha red basado en la probabilidad de que el usuario genere mensajes dentro de la red, promediado los porcentajes de todos los mensajes cuyo porcentaje de probabilidad sea mayor a cero asignando así un peso para el perfil dado dentro de la red social. El servidor proporciona respuestas sobre la mensajería cargada por los usuarios.

Palabras clave

API: interfaz de programación de aplicaciones
Red Social: Página web en la que los internautas intercambian información personal y contenidos multimedia de modo que crean una comunidad de amigos virtual e interactiva.

TDA: Tipo de Dato Abstracto (modelo de datos en memoria dinámica).
XML: Xtensible Markup Language (Lenguaje de marcado extensible).

Abstract

The keywords of certain predefined profiles have been analyzed within the simulation of a ChapinChat social network as well as messages published by different registered users within a site created from local servers by Django and Flask. The weight of the profiles in said network have been calculated based on the probability that the user generates messages within the network, averaging the percentages of all the messages whose probability percentage is greater than zero, thus assigning a weight for the given profile within from the social network. The server provides responses on messages uploaded by users..

Keywords

API: *Application Programming Interface*

Social Network: *Web page in which Internet users exchange personal information and multimedia content so as to create a virtual and interactive community of friends.*
Orthogonal: *unidirectional path with initial and final position between organisms.*

XML: *Xtensible Markup Language.*

Introducción

Se ha buscado crear un algoritmo que pueda calcular el peso promedio de los mensajes publicados por los usuarios de la red social ChatChapin. El algoritmo se basa en los porcentajes de probabilidad de los perfiles en los mensajes que el usuario ha generado en la red social. Dados sus mensajes y obteniendo las palabras clave de sus mensajes se promedia y asigna un porcentaje de pertenecer a cierto perfil para determinar en qué perfil puede catalogarse ese usuario.

a) Estructura del programa

El programa se ha construido en una arquitectura que cuenta con dos programas, el primer programa correspondiendo al Frontend quien hace peticiones al segundo programa siendo el Backend quien se encarga de gestionar las peticiones y la lógica del programa.

b) Vistas

En Django, una vista es una función de Python que procesa una solicitud HTTP y devuelve una respuesta HTTP. En otras palabras, una vista es el código que se ejecuta cuando un usuario hace una petición a una URL en una aplicación Django.

Las vistas de Django se definen en los archivos views.py de una aplicación. Una vista toma una solicitud HTTP como entrada y devuelve una respuesta HTTP como salida. La respuesta puede ser una página HTML, un archivo de imagen, un archivo PDF o cualquier otro tipo de contenido que se pueda servir a través de HTTP.

Las vistas de Django se pueden escribir de muchas formas diferentes, pero generalmente siguen un patrón similar. Primero, la vista realiza cualquier procesamiento necesario de los datos enviados con la solicitud. Luego, se genera la respuesta HTTP adecuada, que puede incluir una plantilla HTML para mostrar en el navegador del usuario.

Las vistas también pueden interactuar con modelos y bases de datos en Django para realizar operaciones como leer o escribir datos. En resumen, las vistas son la parte central de una aplicación Django, ya que definen cómo se manejan las solicitudes y se generan las respuestas.

c) Servicio 1

Desde un archivo de entrada XML seleccionado por el usuario a través del frontend se carga la información de perfiles, palabras clave y palabras descartadas que se envía como solicitud al backend para que este lo lea y procese para tener almacenada toda la información mencionada para su manejo. El

manejo de archivos consiste en una solicitud de tipo POST que se genera en el momento en que el usuario carga el archivo .XML entonces la petición la recibe el backend quien procesa y genera una respuesta previamente definida la cual consiste en un formato XML que especifica la cantidad de perfiles leídos, palabras leídas así como de modificaciones hechas a la información previa.

d) Interfaz e interacción

Los servicios de los programas están diseñados para ser consumidos desde el frontend del programa desarrollado en Django, pero pueden ser consumidos por otros clientes como Postman. El usuario desde el frontend interactúa con los programas seleccionando los servicios y cargando los archivos de entrada en formato XML que contienen toda la información manejada y acumulativa.

e) Servicio 2

Consiste en brindar servicios utilizando el protocolo HTTP para procesar los archivos enviados por el programa 1, los datos son almacenados en archivos XML con el formato previamente definido. Este servicio devuelve también los datos almacenados para ser mostrados por el programa 1 en el frontend.

f) Solicitudes y respuestas

En el contexto de una API (Application Programming Interface), una solicitud se refiere a un mensaje que se envía a un servidor para solicitar información o realizar una acción específica. Una solicitud generalmente contiene información como el tipo de operación que se está realizando (por ejemplo, GET, POST, PUT, DELETE), la URL a la que se está haciendo la solicitud y, en algunos casos, datos adicionales que se envían al servidor.

Por otro lado, una respuesta es la información que el servidor devuelve en respuesta a una solicitud. La respuesta puede contener datos en diferentes formatos, como JSON, XML o HTML, y puede incluir un código de estado HTTP que indica si la solicitud se procesó correctamente o no.

En resumen, las solicitudes y respuestas son los componentes fundamentales de una API. Las solicitudes envían información al servidor, y las respuestas devuelven la información solicitada o indican si la solicitud se procesó correctamente o no. Las APIs son una forma común de comunicación entre diferentes sistemas y aplicaciones, y son esenciales para permitir que diferentes servicios y plataformas interactúen entre sí de manera eficiente y coherente.

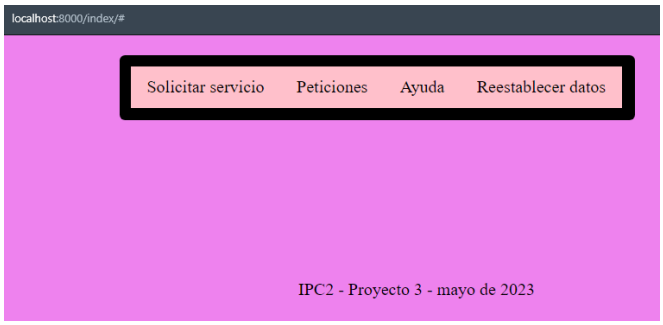
g) Almacenamiento de datos.

Para guardar datos extraídos de un archivo XML con se han seguido a grandes rasgos la siguiente secuencia.

- Importar el módulo "xml.etree.ElementTree" de Python para procesar el archivo XML.
- Carga el archivo XML usando la función "xml.etree.ElementTree.parse()".
- Accedido a los elementos del archivo XML usando la función "xml.etree.ElementTree.getroot()".
- Procesa los datos extraídos del archivo XML y guárdalos en una estructura de datos de Python, como una lista o un diccionario, se ha optado por archivos XML.
- Usa una biblioteca de almacenamiento de datos de Python, como SQLAlchemy, para almacenar los datos en una base de datos.

Anexos:

Figura 1: Página principal del programa



Interfaz inicial de la API donde el usuario realiza las solicitudes.

Figura 2: Ejemplo de respuesta 1

```xml
<?xml version="1.0"?>
<respuesta>
    <perfilesNuevos>Se han creado 3 perfiles nuevos</
    perfilesNuevos>
    <perfilesExistentes>Se han actualizado 2 perfiles
    existentes</perfilesExistentes>
    <descartadas>Se han creado 20 nuevas palabras a
    descartar</descartadas>
</respuesta>
```

Ejemplo del formato de salida que obtiene el usuario como respuesta del servidor tras solicitar el servicio 1.

Figura 3: Ejemplo de entrada

```xml
<configuracion>
 <perfiles>
    <perfil>
        <nombre> Miriam </nombre>
        <palabrasClave>
            <palabra>pm1</palabra>
            <palabra>pm2</palabra>
            <palabra>pm3</palabra>
        </palabrasClave>
    </perfil>
    <perfil>
        <nombre>Otro</nombre>
        <palabrasClave>
            <palabra>po1</palabra>
            <palabra>po2</palabra>
            <palabra>po3</palabra>
        </palabrasClave>
    </perfil>
 </perfiles>
 <descartadas>
    <palabra>P1</palabra>
    <palabra>P2</palabra>
    <palabra>P3</palabra>
    <palabra>P4</palabra>
 </descartadas>
```

Estructura de la entrada que es consumida por el servidor y procesado para generar una respuesta y almacenar la información.

Referencias bibliográficas

Graphviz. *Documentation*, 2010.
Disponible en: graphviz.org

Serrano. M. *Estructura de datos*. Universidad de Valladolid. Segovia.
Disponible en: www.infor.uva.es