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

201908359 – Alexis Marco Tulio López Cacoj

Resumen

Desarrollar una solución integral que implemente un API que brinde servicios utilizando el Protocolo HTTP bajo el concepto de programación orientada a objetos (POO) con el lenguaje de Python utilizando los frameworks Django y flask para la correcta programación del software .

Palabras clave
Backend
Frontend
HTTP
POST
GET

Abstract

Develop a comprehensive solution that implements an API that provides services using the HTTP Protocol under the concept of object-oriented programming (OOP) with the Python language using the Django and flask frameworks for the correct programming of the software.

Keywords
Backend
Frontend
HTTP
POST
GET

Introducción

Se desarrollara una aplicación que implemente un API que brinde servicios utilizando el Protocolo HTTP para poder realizar un solución integral para construir un software que pueda ser consumido desde Internet como un servicio. Este software recibirá un mensaje de la bitácora del software principal y producirá una serie de información estadística relacionada a. El mensaje que la bitácora enviará contendrá la siguiente información que se muestra en la figura1 :

```
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
```

Figura1: información de bitácora

El programa a desarrollar, luego de recibir el mensaje antes mencionado deberá almacenar la información

necesaria en un archivo XML que permitirá mostrar la siguiente información que se muestra en la figura2:

```
FECHA: dd/mm/yyyy
Cantidad total de mensajes recibidos en esta fecha
Listado de usuarios distintos que reportaron mensajes
Usuario1 cantidad mensajes generados
Usuario2 cantidad mensajes generados
...
Listado de usuarios afectados
UsuarioX1
UsuarioX2
...
Listado de errores distintos reportados
Código numérico del error: cantidad de mensajes recibidos
Código numérico del error: cantidad de mensajes recibidos
...
FECHA: dd/mm/yyyy
...
```

Figura2: xml de salida

Desarrollo del tema

API: que es y para qué sirve

¿Qué es un api?

El término API es una abreviatura de **A**pplication **P**rogramming **I**nterfaces, 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.

Así pues, podemos hablar de una API como una especificación formal que establece cómo un módulo de un software se comunica o interactúa con otro para cumplir una o muchas funciones. Todo dependiendo de las aplicaciones que las vayan a utilizar, y de los permisos que les dé el propietario de la API a los desarrolladores de terceros.

Para qué sirve una API

Una de las principales funciones de las API es poder facilitarle el trabajo a los desarrolladores y ahorrarles tiempo y dinero. Por ejemplo, si estás creando una aplicación que es una tienda online, no necesitarás crear desde cero un sistema de pagos u otro para verificar si hay stock disponible de un producto. Podrás utilizar la API de un servicio de pago ya existente, por ejemplo PayPal, y pedirle a tu distribuidor una API que te permita saber el stock que ellos tienen.

Con ello, no será necesario tener que reinventar la rueda con cada servicio que se crea, ya que podrás utilizar piezas o funciones que otros ya han creado. Imagínate que cada tienda online tuviera que tener su propio sistema de pago, para los usuarios normales es mucho más cómodo poder hacerlo con los principales servicios que casi todos utilizan.

Backend y Frontend

¿Qué es el frontend?

Frontend es la parte de un programa o dispositivo a la que un usuario puede acceder directamente. Son todas las tecnologías de diseño y desarrollo web que corren en el navegador y que se encargan de la interactividad con los usuarios.
Para convertirte en Frontend Developer debes saber HTML y CSS, los lenguajes de maquetación que nos permiten definir la estructura y estilos de una página web. Y también JavaScript, un lenguaje de programación para definir la lógica de nuestra aplicación, recibir las solicitudes de los usuarios y enviárselos al backend. Conoce más a fondo cómo aprender arquitectura frontend.

La mayoría de las aplicaciones de software tienen un frontend. Incluso con una base de datos donde la mayoría de los procesos se ejecutan en segundo plano y son invisibles para el usuario, existe una interfaz gráfica de usuario, como la tabla de salida después de una consulta de datos. Los sistemas de aplicaciones basados en la web casi siempre se dividen en un front-end y un back-end. En un sistema de gestión de contenidos, el front-end es la interfaz en la que los visitantes pueden ver el contenido publicado. En el caso de un sistema de tiendas, el frontend también ofrece la posibilidad de comprar productos, suscribirse a una newsletter, ser consultado y mucho más. En los foros y comunidades, los usuarios pueden ver la

información en el frontend y al mismo tiempo, ayudar a darle forma, participar activamente en ella y dar consejos.

¿Qué es el backend?

Backend es la capa de acceso a datos de un software o cualquier dispositivo, que no es directamente accesible por los usuarios, además contiene la lógica de la aplicación que maneja dichos datos. El Backend también accede al servidor, que es una aplicación especializada que entiende la forma como el navegador solicita cosas. Algunos de los lenguajes de programación para Backend son Python, Node.js, PHP, Go, Ruby y C#. Y así como en el frontend, todos estos lenguajes tienen diferentes frameworks que te permiten trabajar mejor según el proyecto que estás desarrollando,
como Django, Flask, Express.js, Laravel, Symphony Framework, Ruby on Rails y ASP.Net. Cada uno lo hemos elegido sobre todo porque tienen una gran comunidad que los respalda.

¿Qué es flask?

Flask es un “micro” Framework escrito en Python y concebido para facilitar el desarrollo de Aplicaciones Web bajo el patrón MVC.
La palabra “micro” no designa a que sea un proyecto pequeño o que nos permita hacer páginas web pequeñas, sino que al instalar Flask tenemos las herramientas necesarias para crear una aplicación web funcional, pero si se necesita en algún momento una nueva funcionalidad hay un conjunto muy
grande extensiones (plugins) que se pueden instalar con Flask que le van dotando de funcionalidad.

De principio en la instalación no se tienen todas las funcionalidades que se pueden necesitar, pero de una manera muy sencilla se pueden extender el proyecto con nuevas funcionalidades por medio de plugins.
El patrón MVC es una manera o una forma de trabajar que permite diferenciar y separar lo que es el modelo de datos (los datos que van a tener la App que normalmente están guardados en BD), la vista (página HTML) y el controlador (donde se gestiona las peticiones de la app web).

¿Qué es un Framework?
Actualmente en el desarrollo moderno de aplicaciones web se utilizan distintos Frameworks que son herramientas que nos dan un esquema de trabajo y una serie de utilidades y funciones que nos facilita y nos abstrae de la construcción de páginas web dinámicas.
En general los Frameworks están asociado a lenguajes de programación (Ruby on Rails (Ruby), Symphony (PHP)), en el mundo de Python el más conocido es Django pero Flask es una opción que quizás no tenga una curva de aprendizaje tan elevada pero nos posibilita la creación de aplicaciones web igual de complejas de las que se pueden crear en Django.

¿Qué es 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.

Django te ayuda a escribir software que es:

Completo

Django sigue la filosofía "Baterías incluidas" y provee casi todo lo que los desarrolladores quisieran que tenga "de fábrica". Porque todo lo que necesitas es parte de un único "producto", todo funciona a la perfección, sigue principios de diseño consistentes y tiene una amplia y actualizada documentación.

Versátil

Django puede ser (y ha sido) usado para construir casi cualquier tipo de sitio web — desde sistemas manejadores de contenidos y wikis, hasta redes sociales y sitios de noticias. Puede funcionar con cualquier framework en el lado del cliente, y puede devolver contenido en casi cualquier formato (incluyendo HTML, RSS feeds, JSON, XML, etc). ¡El sitio que estás leyendo actualmente está basado en Django!

Internamente, mientras ofrece opciones para casi cualquier funcionalidad que desees (distintos motores de base de datos , motores de plantillas, etc.), también puede ser extendido para usar otros componentes si es necesario.

Seguro

Django ayuda a los desarrolladores evitar varios errores comunes de seguridad al proveer un framework que ha sido diseñado para "hacer lo correcto" para proteger el sitio web automáticamente. Por ejemplo, Django, proporciona una manera segura de administrar cuentas de usuario y contraseñas, evitando así errores comunes como colocar informaciones de sesión en cookies donde es vulnerable (en lugar de eso las cookies solo contienen una clave y los datos se almacenan en la base de datos) o se almacenan directamente las contraseñas en un hash de contraseñas.

Un hash de contraseña es un valor de longitud fija creado al enviar la contraseña a una cryptographic hash function. Django puede validar si la contraseña ingresada es correcta enviándola a través de una función hash y comparando la salida con el valor hash almacenado. Sin embargo debido a la naturaleza "unidireccional" de la función, incluso si un valor hash almacenado se ve comprometido es difícil para un atacante resolver la contraseña original.

Django permite protección contra algunas vulnerabilidades de forma predeterminada, incluida la inyección SQL, scripts entre sitios, falsificación de solicitudes entre sitios y clickjacking (consulte Seguridad de sitios web para obtener más detalles sobre dichos ataques).

Escalable

Django usa un componente basado en la arquitectura “shared-nothing” (cada parte de la arquitectura es independiente de las otras, y por lo tanto puede ser reemplazado o cambiado si es necesario). Teniendo en cuenta una clara separación entre las diferentes partes significa que puede escalar para aumentar el tráfico al agregar hardware en cualquier nivel: servidores de cache, servidores de bases de datos o servidores de aplicación. Algunos de los sitios más concurridos han escalado a Django para satisfacer sus demandas (por ejemplo, Instagram y Disqus, por nombrar solo dos).

Mantenible

El código de Django está escrito usando principios y patrones de diseño para fomentar la creación de código mantenible y reutilizable. En particular, utiliza el principio No te repitas "Don't Repeat Yourself" (DRY) para que no exista una duplicación innecesaria, reduciendo la cantidad de código. Django también promueve la agrupación de la funcionalidad relacionada en "aplicaciones" reutilizables y en un nivel más bajo, agrupa código relacionado en módulos (siguiendo el patrón Model View Controller (MVC)).

Portable

Django está escrito en Python, el cual se ejecuta en muchas plataformas. Lo que significa que no está sujeto a ninguna plataforma en particular, y puede ejecutar sus aplicaciones en muchas distribuciones de Linux, Windows y Mac OS X. Además, Django cuenta con el respaldo de muchos proveedores de alojamiento web, y que a menudo proporcionan una infraestructura específica y documentación para el alojamiento de sitios de Django.

Conclusiones

La implementación de un framework siempre es importante para desarrollar algún software ya que estas nos facilitan el comienzo del mismo, ya que son un tipo de planilla o base y asì no empezar desde cero y tener un desarrollo del software mas escalable a la hora de codificar código.

Referencias bibliográficas

Titulo: Frontend Sitio: RYTE WIKI Fecha: Desconocido
Autor: Desconocido
URL: https://es.ryte.com/wiki/Frontend

Titulo: Que es Flask Sitio:OpenWebinars Fecha:17 de noviembre de 2017 Autor: José Domingo Muño URL: https://openwebinars.net/blog/que-es-flask/

Titulo: Qué es Frontend y Backend
Sitio: Fecha: Desconocido Autor: Maldeora URL: https://platzi.com/blog/que-es-frontend-y-backend/

Titulo: API: qué es y para qué sirve
Sitio: Xakayaka Fecha: Desconocido Autor: Yubal Fernandez URLhttps://www.xataka.com/basics/api-que-sirve