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 2022.

PROYECTO 3- API Servicios de Infraestructura de Nube

202111718 – Walter Javier Santizo Mazariegos

Resumen

Para la realización del proyecto No. 3 se realizó una aplicación web mediante django y flask a una empresa de infraestructuras en la nube mediante la cual se crea y se gestiona información de los recursos y clientes que una empresa maneja, los cuales cargan al sistema los por ejemplo, el envió de un mensaje de éxito a los celulares de los clientes informando el consumo de los recursos de las distintas categorías y configuraciones. Se permite cargar una configuración inicial por medio de un XML el cual contiene los recursos, categorías y clientes que la empresa posee

Para la solución del problema se hizo uso de las rest apis y del framework django y flask, usando listas nativas de Python y las herramientas de web services que brinda los 2 frameworks en este caso se trabaj el modelo MVT Modelo Vista Template que es el que utiliza djanf¿go en el disño de paginas web además usamos notación jinja para las plantillas de nuestra web.

La api lee archivos xml y los almacena en listas simulando una base de datos mediante estos archivos además mediante los endpoints se puede listar y crear información por medio los métodos “POST” y “GET” sobre los cuales giro la base de este proyecto así como sus alcances y objetivos

Palabras clave

Protocolos ,HTTP , API , nube , arquitecturas, entorno, dominio, puerto, xml, Flask, Django

Abstract

To carry out project No. 3, a web application was made using django and flask for a cloud infrastructure company through which information on the resources and clients that a company manages is created and managed, which load the data into the system. for example, the sending of a success message to the clients' cell phones informing the consumption of the resources of the different categories and configurations. It is allowed to load an initial configuration through an XML which contains the resources, categories and clients that the company has

To solve the problem, the rest apis and the django and flask framework were used, using native Python lists and the web services tools provided by the 2 frameworks. In this case, the MVT model Model View Template was used, which is the one that uses djanf¿go in the design of web pages and we also use jinja notation for our web templates.

The api reads xml files and stores them in lists, simulating a database through these files, and through the endpoints it is possible to list and create information through the "POST" and "GET" methods on which the basis of this project is based, as well as its scope and objectives

Keywords

Protocols, HTTP, API, cloud, architectures, environment, domain, port, xml, Flask, Django

Introducción

La empresa Tecnologías Chapinas, S.A. está desarrollando una herramienta que sea capaz de facturar detalladamente los servicios de infraestructura de nube que aprovisiona a sus clientes.

La estructura de la tecnología de nube desarrollada por Tecnologías Chapinas, S.A. consiste en crear configuraciones de infraestructura que agrupan recursos necesarios para que una empresa pueda construir las arquitecturas de despliegue de aplicaciones que requiera.

Debido a que existen muchos recursos tecnológicos y que éstos deben configurarse de acuerdo con las cargas de trabajo para la cual será aprovisionada la infraestructura, Tecnologías Chapinas, S.A. ha creado categorías

Cada categoría define "N" configuraciones, estas configuraciones agrupan recursos que son integrados para brindar máquinas virtuales que puedan satisfacer los requerimientos de carga de trabajo definidos en la categoría.

Actualmente, los clientes de Tecnologías Chapinas, S.A. se registran ingresando su nombre, su NIT, su dirección física y su dirección de correo electrónico, una vez registrados reciben en su correo electrónico un usuario y una clave con la cual pueden ingresar a una pagina web donde pueden crear, modificar y eliminar instancias. Una vez que encienda su instancia, ésta empezará a calcular el tiempo de uso de cada recurso involucrado, y mensualmente, deberá facturar según las tarifas vigentes por cada recurso.

Desarrollo del tema

Api

La expresión Application Programming Interface.
Hace referencia a un traductor, cuya función es conectar sistemas, software y aplicaciones. Con las API es posible ofrecerle una experiencia de uso más familiar a las personas.

Las API le permiten al usuario final utilizar una aplicación, software o incluso una simple hoja de cálculo, consultando, cambiando y almacenando datos de diferentes sistemas, sin que el usuario tenga que ingresar a ellos, directamente.

El propósito de una API es intercambiar datos entre diferentes sistemas, la mayoría de las veces estos intercambios de datos tienen como objetivo automatizar procesos manuales y / o permitir la creación de nuevas funcionalidades.

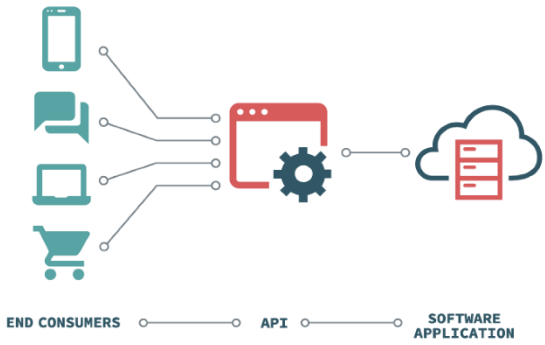


Figura 1. Modelo de un API

Flask

Flask es un framework minimalista escrito en Python que permite crear aplicaciones web rápidamente y con un mínimo número de líneas de código. Está basado en la especificación WSGI de Werkzeug y el motor de templates Jinja2 y tiene una licencia BSD

Figura 2. Logo de Flask

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.

Figura 3. Logo de Django

API Rest

la API utiliza requisiciones HTTP responsables de las operaciones básicas necesarias para la manipulación de datos.

Siendo un conjunto de restricciones que se utilizan para que las solicitudes HTTP cumplan con las directrices definidas en la arquitectura. Básicamente, las restricciones determinadas por la arquitectura Rest son:

- **Cliente-servidor:** las aplicaciones existentes en el servidor y el cliente deben estar separadas.
- **Sin estado:** las requisiciones se realizan de forma independiente, es decir, cada una ejecuta solo una determinada acción
- **Caché:** la API debe utilizar la caché para evitar llamadas recurrentes al servidor.
- **Interfaz uniforme:** agrupa otros cuatro conceptos en los que se determina que los recursos deben ser identificados, la manipulación de los recursos debe ser a través de la representación.

Las principales solicitudes son:

- POST: crea datos en el servidor;
- GET: lectura de datos en el host;
- DELETE: borra la información; • PUT: registro de actualizaciones.

Protocolo HTTP

El Protocolo de Transferencia de HiperTexto (Hypertext transfer Protocol) es un sencillo protocolo cliente-servidor que articula los intercambios de información entre los clientes Web y los servidores HTTP. La especificación completa fue propuesta por Tim Berners-Lee, atendiendo a las necesidades de un sistema global de distribución de información como el World Wide Web.

Cada vez que un cliente realiza una petición a un servidor, se ejecutan los siguientes pasos:

- Un usuario accede a una URL, seleccionando un enlace de un documento HTML o introduciéndola directamente en el campo Location del cliente Web.
- El cliente Web descodifica la URL, separando sus diferentes partes. Así identifica el protocolo de acceso, la dirección DNS o IP del servidor, el posible puerto opcional (el valor por defecto es 80) y el objeto requerido del servidor.
- Se abre una conexión TCP/IP con el servidor, llamando al puerto TCP correspondiente. Se realiza la petición. Para ello, se envía el comando necesario (GET, POST, HEAD,…), la dirección del objeto requerido (el contenido de la URL que sigue a la dirección del servidor), la versión del protocolo HTTP empleada (casi siempre HTTP/1.0) y un conjunto variable de información, que incluye datos sobre las capacidades del browser, datos opcionales para el servidor.

- El servidor devuelve la respuesta al cliente. Consiste en un código de estado y el tipo de dato MIME de la información de retorno, seguido de la propia información.

- Se cierra la conexión TCP.

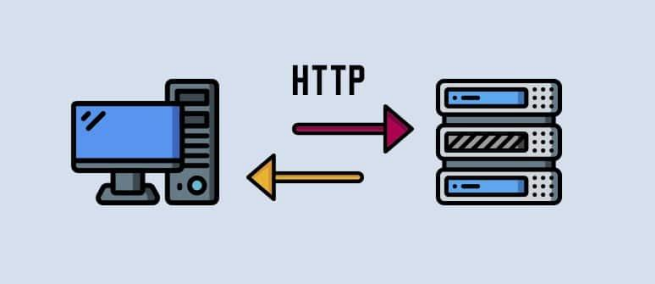


Figura 4. petición/Respuesta

Regex

Las expresiones regulares son patrones que se utilizan para hacer coincidir combinaciones de caracteres en cadenas. Las expresiones regulares son patrones utilizados para encontrar una determinada combinación de caracteres dentro de una cadena de texto. Las expresiones regulares proporcionan una manera muy flexible de buscar o reconocer cadenas de texto.

Arquitectura del Proyecto

La API brindara servicios utilizando el protocolo HTTP, su funcionalidad principal es procesar los datos recibidos del programa 1, luego de procesar los datos es necesario que estos sean almacenados en uno o varios archivos xml que representan la base de datos, este servicio también tiene la funcionalidad de devolver los datos que fueron almacenados para que sean mostrados como respuesta a las solicitudes realizadas por el “Programa 1 – Frontend”.

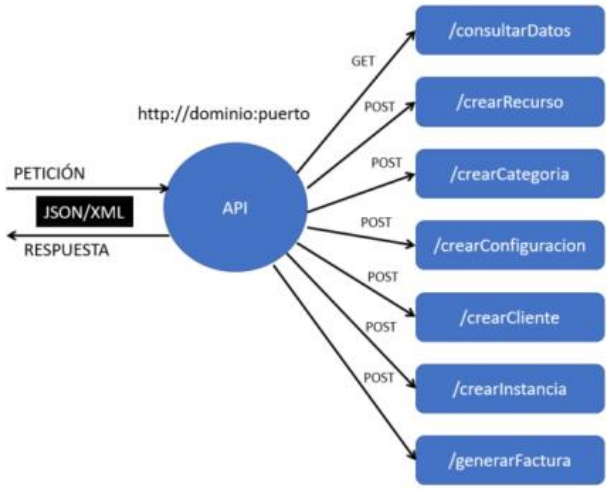


Figura 5. Arquitectura del API

Conclusiones

- El modelo MVT (modelo vista template) facilita al desarrollador brindar una solución escalable y de calidad a los problemas cotidianos

- Python es un lenguaje de programación fácil de entender y programar con variedad de librerías de las cuales nos ha servido para la resolución del problema y construcción de clases necesarios para el funcionamiento del algoritmo.

- Los archivos xml y json son una forma fácil, de interpretar los objetos provenientes de una web y procesarlos en información que el cliente pueda observar

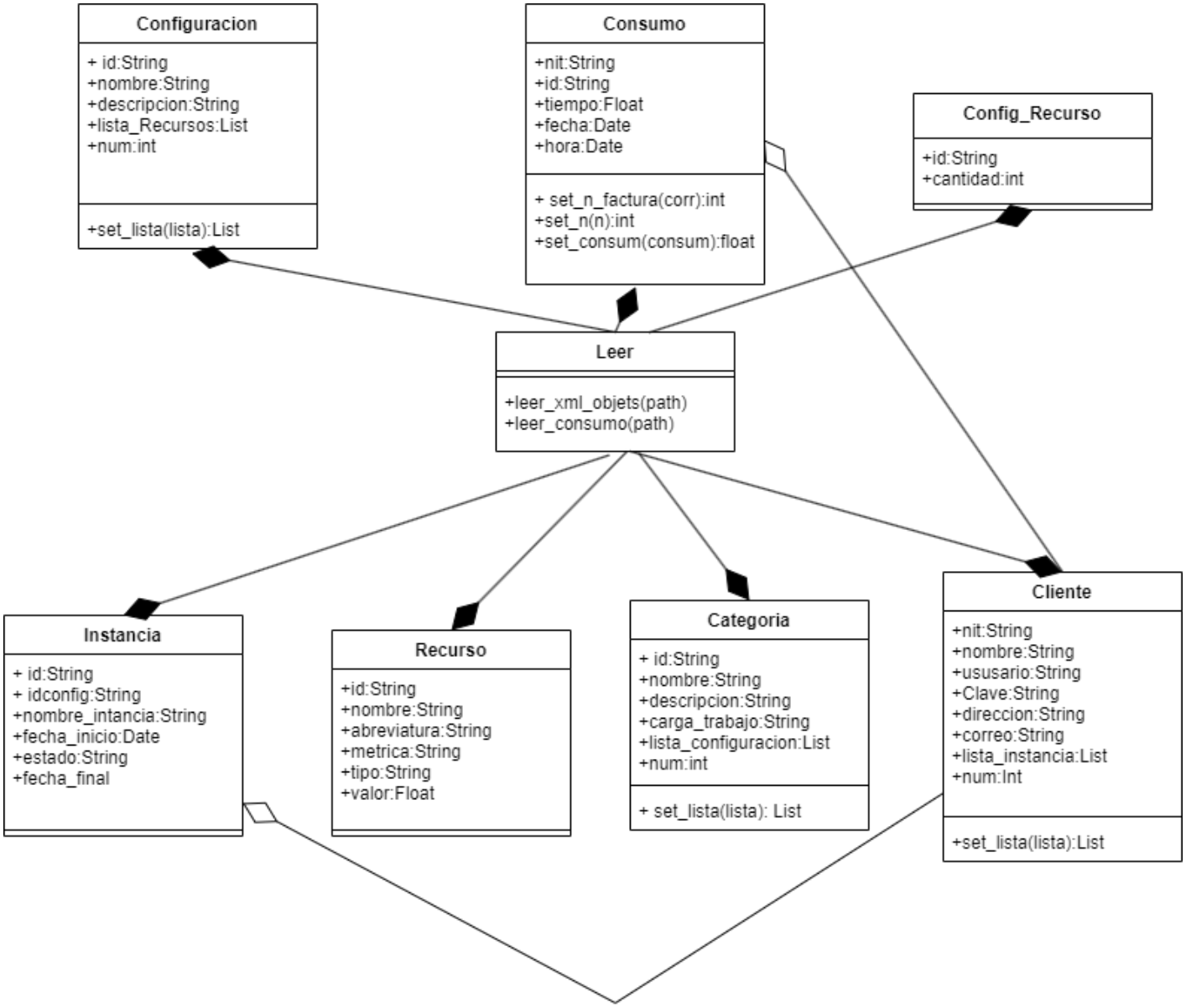
- Las Peticiones HTTP son una forma de estandarizar los web services y las aplicaciones que estas pueden brindar

- La implementación de las APIs a las aplicaciones web actualmente son una de las tecnologías mas usadas, en producción por las empresas

Referencias bibliográficas

- C. S. Severance, (2009). Python para informaticos. Explorando la informacion, version 2.7.2, Inc.
- C. Larman, (2003). Introducción al análisis y diseño orientado a objetos, Hall Prentice
- . The web framework for perfectionists with deadlines. (s/f). Djangoproject.com. Recuperado el 3 de noviembre de 2022, de https://www.djangoproject.com/
- Welcome to flask — flask documentation (2.2.X). (s/f). Palletsprojects.com. Recuperado el 3 de noviembre de 2022, de https://flask.palletsprojects.com/en/2.2.x/
- Bach, J. (2021). Django: Una guia completa para desarrollar sitios web con Django. Independently Published.
- Sebastian, J. (2009). The Art of XSD - SQL Server XML Schemas. Simple Talk Publishing.
- Studer, R., Grimm, S., & Abecker, A. (Eds.). (2010). Semantic Web Services: Concepts, Technologies, and Applications. Springer.

Diagrama de Clases



Modelo de Datos