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.

SISTEMA WEB DE FACTURACIÓN PARA SERVICIOS DE INFRAESTRUCTURA DE NUBE

202100154 – Sergio André Lima Corado

Resumen

En este proyecto se ha realizado un sistema de 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.

Para esta solución en particular se llevó a cabo un proyecto dividido en dos partes, el frontend y el backend. En dónde se levantó una API a través del lenguaje Python con el framework de Flask que puede ser consumida utilizando el protocolo HTTP. Por otra parte, también se implementó por primera vez en el curso una base de datos para almacenar información de forma persistente, en mi caso basada en JSON.
.

Palabras clave

Recurso, Servicio, Remoto, Cliente, Servidor

Abstract

In this project, a system capable of billing in detail the cloud infrastructure services provided to its customers was developed. The structure of the cloud technology developed by Tecnologías Chapinas, S.A. consists of creating infrastructure configurations that group the necessary resources so that a company can build the application deployment architectures it requires.

For this particular solution we carried out a project divided into two parts, the frontend and the backend. In which an API was built through the Python language with the Flask framework that can be consumed using the HTTP protocol. On the other hand, we also implemented for the first time in the course a database to store information persistently, in my case based on JSON.

Keywords

Resource, Service, Remote, Client, Server

Introducción

Es importante entender el trabajo *frontend* y *backend* con una API pues es una de las dinámicas más encontradas en el trabajo con programas de código, conexión con APIs y desarrollo de aplicaciones. Ambos conceptos revelan características importantes acerca del lenguaje computacional y cómo trabajar con él.

En la solución que se presentó para el proyecto presenta precisamente ambas partes (frontend y backend). A continuación, se detalla la estructura del proyecto y su funcionalidad.

Desarrollo del tema

Cuando estamos usando una API nos encontramos con varios elementos dentro del programa para desarrollar una aplicación o plataforma web que conecte con ella. Este trabajo se divide en dos partes muy conocidas dentro del sector de la programación. Estos son el campo frontend y el campo backend. Este trabajo mancomunado es importante, pues ambos campos del programa se juntan en un solo sitio. Por lo tanto, el programa tendrá dos caras: la de Django en el cliente y la de Flask en el campo del servidor.

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.

Una de las principales funciones de las API es poder facilitarles 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.

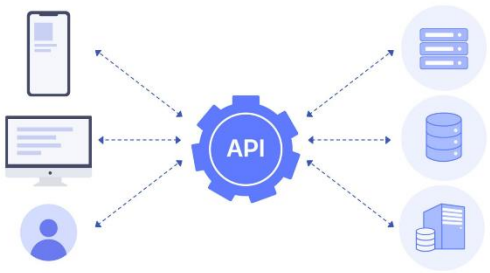


Figura 1. Ilustración de una API

Fuente: Fundamentos de API's, 2022.

A veces otros servicios crean API de forma deliberada para ser utilizados por terceros en tareas concretas, y así extender su uso y popularidad creando nuevas funciones. Por ejemplo, Google creó una para Google Docs con las que permite la creación automatizada de facturas o informes de ventas a otros servicios. De dónde se puede resaltar la creación de facturas como uno de los objetivos de este proyecto.

Flask

Flask es un "micro" Framework escrito en Python y desarrollado para simplificar y hacer más fácil la creación de Aplicaciones Web bajo el patrón MVC.

En cuanto al patrón MVC, este es una forma de trabajar que permite diferenciar y separar lo que es la vista (página HTML), el modelo de datos (los datos que va a tener la App), y el controlador (donde se gestionan las peticiones de la aplicación web).

Algunas ventajas del uso del framework Flask son:

- ✓ Micro Framework: Perfecto si se quiere desarrollar una App básica o que se quiera crear de manera rápida y ágil. Para según qué aplicaciones no se requieren muchas extensiones y con Flask es suficiente.

- ✓ Incluye un servidor web de desarrollo: Por lo tanto, no requiere una infraestructura con un servidor web para testar las aplicaciones web, simplemente se puede correr un servidor web de forma sencilla con el que se pueden ir observando los resultados que se van obteniendo.

- ✓ Tiene un depurador y soporte integrado para pruebas unitarias: Si hay algún error en el código que se está creando, se puede depurar ese error y también se pueden ver los valores de las variables. A su vez, existe la posibilidad de integrar pruebas unitarias.

- ✓ Buen manejo de rutas: Cuando se trabaja con Apps Web hechas en Flask Python se tiene el controlador que recibe todas las peticiones que hacen los clientes y se tiene que determinar a qué ruta está accediendo el cliente para ejecutar el código pertinente.

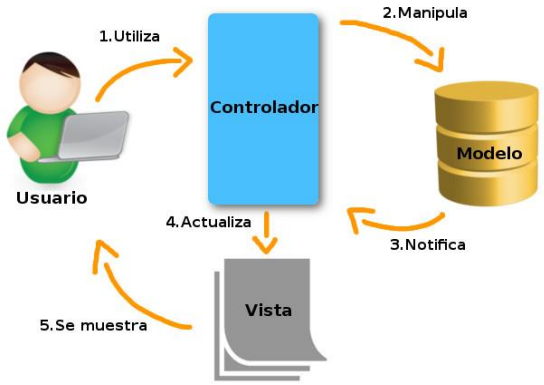


Figura 2. Ilustración del patrón MVC

Fuente: Fundamentos de API's, 2022.

- ✓ Soporta de manera nativa el uso de cookies.
- ✓ Sin ORMs: No tiene ORMs, pero fácilmente se puede usar una extensión.
- ✓ Óptima para construir servicios web o aplicaciones de contenido estático.
- ✓
- ✓ Documentación, lista de correos y código de GitHub.
- ✓ Open Source que se ampara bajo una licencia BSD.

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.

Desde la página web de Django podemos ver algunos destacados, en los que podemos ver algunas webs como la de National Geografic, Disqus, Instagram, Mozilla Foundation y Pinterest, que son webs con un tráfico altísimo y utilizan Django.

Los motivos principales para usar Django son:

- Es muy rápido: Si se tiene un startup, se tiene prisa por terminar algún proyecto o, simplemente, se quiere reducir costes, con Django se puede construir una aplicación muy buena en poco tiempo.

- Viene bien cargado: Cualquier cosa que se necesite realizar, ya estará implementada, sólo hay que adaptarla a las respectivas necesidades. Ya sea porque hay módulos de la comunidad, por cualquier paquete Python que se encuentre o las propias aplicaciones que Django trae, que son muy útiles.

- Es bastante seguro: Podemos estar tranquilos con Django, ya que implementa por defecto algunas medidas de seguridad, las más clásicas, para que no haya SQL Injection, no haya Cross site request forgery (CSRF) o no haya Clickjacking por JavaScript. Django se encarga de manejar todo esto de una manera realmente sencilla.

- Es muy escalable: Podemos pasar desde muy poco a una aplicación enorme perfectamente, una aplicación que sea modular, que funcione rápido y sea estable.

- Es increíblemente versátil: Es cierto que en un principio Django comienza siendo un Framework para almacenar noticias por sitios de prensa, blogs y este estilo de webs, pero con el tiempo ha ganado tanta popularidad que se puede usar para el propósito que se desee.

Arquitectura API

La arquitectura para la API se basa en 7 endpoints, que prestarán un servicio distinto cada una. La mayoría de estos endpoints son un método POST pues son para crear los distintos tipos de datos u objetos. Por otra parte, se encuentra un endpoint con método GET, para poder consultar todos los datos disponibles en la base de datos.

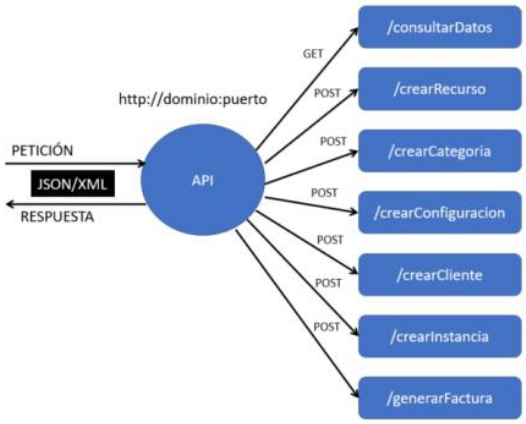


Figura 2. Arquitectura de la API

Fuente: Enunciado del Proyecto, ECYS, 2022.

Conclusiones

Flask es una excelente opción para todas aquellas personas que quieren comenzar en el desarrollo web con Python, y eso jamás implica limitarnos a solo desarrollar aplicaciones simples. Con Flask, si así lo deseamos podemos integrar la n cantidad de dependencias que necesitemos para tener proyectos profesionales, robustos y escalables.

Al ser sistemas independientes el fronted y el backend (solo se comunican con un lenguaje de intercambio como JSON) se pueden desarrollar como proyectos autónomos, equipos autónomos. Al cliente le da igual cómo está hecha la API y al servidor le da igual qué vas a hacer con los datos que te ha proporcionado.

Es decir, si se necesita evolucionar/refactorizar uno de los dos, back o front, se puede hacer de manera separada, siempre que se mantenga la interfaz del API. Además, se pueden hacer múltiples front, no necesariamente webs, que consuman un único backend.

Referencias bibliográficas

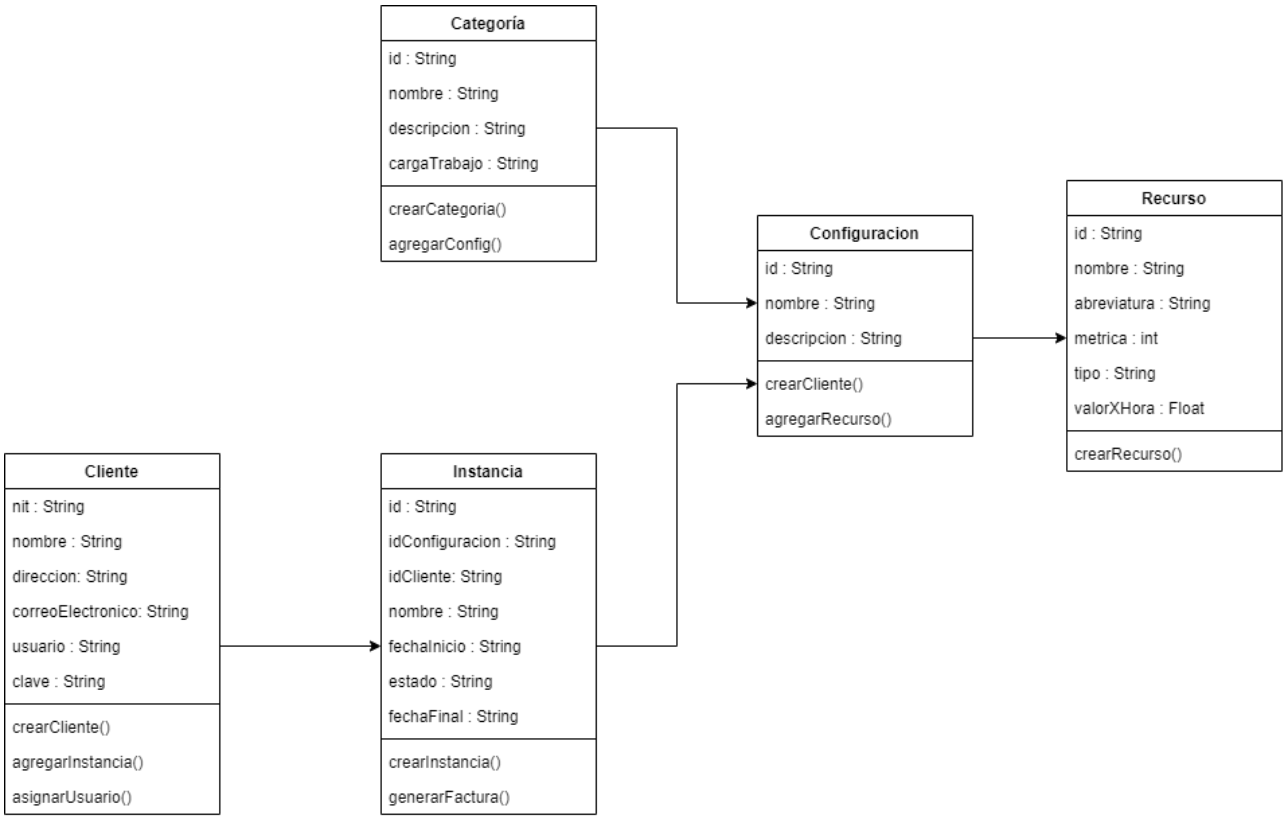
- Forcier, J., Bissex, P., & Chun, W. J. (2008). *Python web development with Django*. Addison-Wesley Professional.

- Grinberg, M. (2018). *Flask web development: developing web applications with python*. " O'Reilly Media, Inc.".

- Mora-Castillo, J. A. (2016). Serialización/deserialización de objetos y transmisión de datos con JSON: una revisión de la literatura. *Revista Tecnología en Marcha*, *29*(1), 118-125.

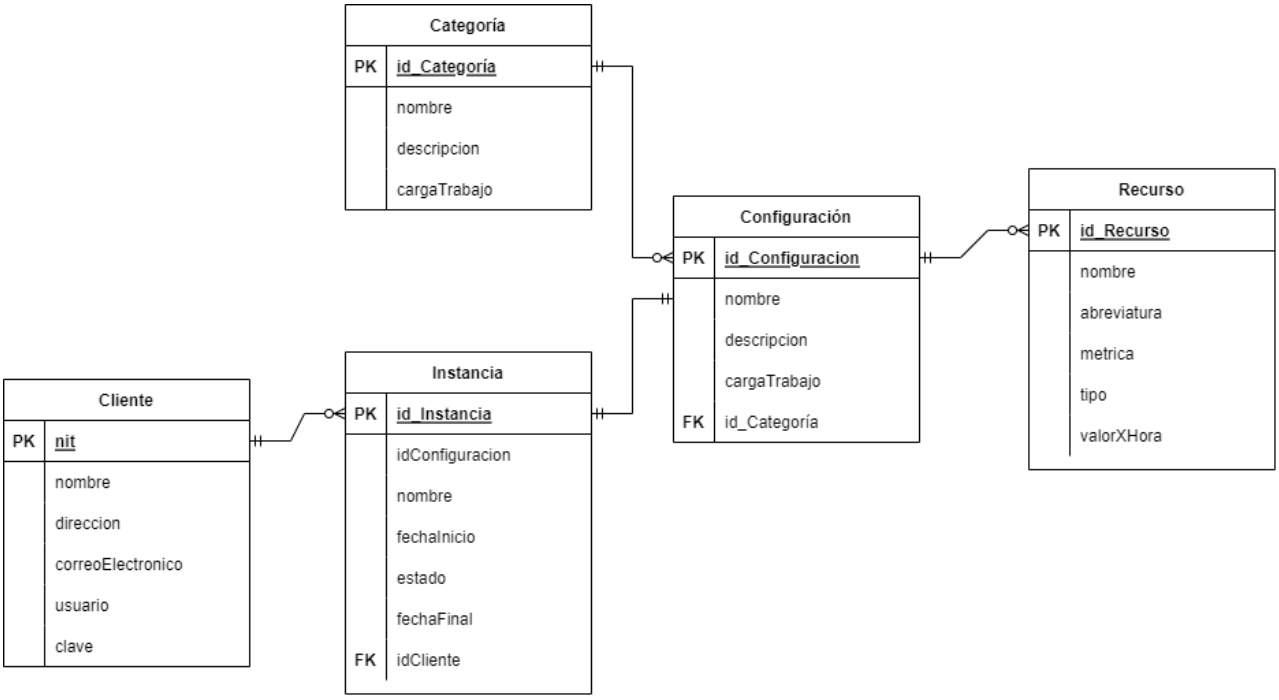
Apéndices

Apéndice A. Diagrama de Clases del Proyecto



Fuente: Elaboración Propia

Apéndice B. Modelo de Datos



Fuente: Elaboración Propia