Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Introducción a la Programación y Computación 2

Inga. Claudia Liceth Rojas Morales
Ing. Marlon Antonio Pérez Türk
Ing. José Manuel Ruiz Juárez
Ing. Dennis Stanley Barrios Gonzalez
Ing. Edwin Estuardo Zapeta Gómez
Ing. Fernando José Paz González

Tutores de curso:
Angely Naomi Marroquín Tapaz
Diego Andrés Huite Alvarez
Hesban Amilcar Argueta Aguilar
Pedro Luis Pu Tavico
Angel Miguel García Urizar
Luis Antonio Castillo Javier

PROYECTO 3

OBJETIVO GENERAL

Desarrollar una solución integral que implemente un API que brinde servicios utilizando el Protocolo HTTP bajo el concepto de programación orientada a objetos (POO) y el uso de bases de datos.

OBJETIVOS ESPECÍFICOS

- Implementar un API a través de lenguaje Python que pueda ser consumida utilizando el protocolo HTTP.
- Utilizar el paradigma de programación orientada a objetos para construir software.
- Utilizar bases de datos para almacenar información de forma persistente.
- Utilizar archivos XML como insumos para la comunicación con el API desarrollado.
- Utilizar expresiones regulares para extraer contenido de texto.

ENUNCIADO

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 (núcleos[1], memoria RAM, almacenamiento secundario, sistemas operativos, bases de datos, etc.) 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 como, por ejemplo: Máquinas virtuales para desarrollo/pruebas, Máquinas virtuales económicas, etc. Cada categoría tiene un nombre, una descripción y la definición de la carga de trabajo para la cual estarán afinadas las configuraciones que se incluyan en la categoría.

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. Entre los recursos que una configuración puede incluir se encuentran: núcleos, memoria RAM, almacenamiento secundario, sistema operativo, base de datos, entre otros. Cada recurso que se puede incluir en una configuración posee una dimensional que se utiliza como métrica para el recurso y un costo por hora de este recurso, así, por ejemplo, para la memoria RAM su dimensional son GiB (Gigabytes) y su costo por hora $ 0.75, mientras que para los núcleos su dimensional es unidades y su costo por hora $.1.20.

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 consola donde pueden crear, modificar y eliminar instancias, las instancias representan el aprovisionamiento de una configuración que el usuario puede seleccionar en alguna de las categorías que brinda Tecnologías Chapinas, S.A. Una vez que el usuario ha aprovisionado las instancias que va a requerir, entonces, cada 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.

Usted ha sido contratado para crear un software que se ejecutará en el backend de la infraestructura de Tecnologías Chapinas, S.A., este software registrará, utilizando API's, todos los datos relacionados a configuraciones, clientes e instancias y, finalmente, calculará la facturación periódica asociada al uso de los recursos aprovisionados para las instancias que los clientes han registrado.

[1] Representa los núcleos (cores) de un CPU.

MENSAJE DE ENTRADA PARA CONFIGURAR LOS OBJETOS DEFINIDOS EN LA NUBE DE TECNOLOGÍAS CHAPINAS, S.A.

Para poder crear la información necesaria para brindar y cobrar por los servicios que presta Tecnologías Chapinas, S.A. se utilizará un mensaje XML con la siguiente estructura:

NOTAS:

Los valores y atributos antecedidos con el símbolo "$" se describen en el apartado "Reglas y consideraciones del formato para cada mensaje recibido".

Los IDs (idRecurso, idCategoria, etc) de los distintos conceptos manejados en este enunciado corresponden a valores numéricos enteros.

```
<?xml version="1.0"?>
<archivoConfiguraciones>
 <listaRecursos>
   <recurso id="idRecurso">
     <nombre> nombreRecurso </nombre>
     <abreviatura> abreviaturaRecurso </abreviatura>
     <metrica> NombreMetrica </metrica>
     <tipo> $tipoRecurso </tipo>
     <valorXhora> valorNumerico </valorXhora>
   </recurso>
   ...
 </listaRecursos>
 <listaCategorias>
   <categoria id="idCateoria">
     <nombre> nombreCategoria </nombre>
     <descripcion> descripcionCategoria </descripcion>
     <cargaTrabajo> descripcionCargaTrabajo </cargaTrabajo>
     <listaConfiguraciones>
        <configuracion id="idConfiguracion">
          <nombre> nombreConfiguracion </nombre>
          <descripcion> descripcionConfiguracion </descripcion>
          <recursosConfiguracion>
             <recurso id="idRecurso"> cantidadRecurso </recurso>
             ...
          </recursosConfiguracion>
        </configuracion>
        ...
     </listaConfiguraciones>
   </categoria>
   ...
 </listaCategorias>
 <listaClientes>
   <cliente nit="$nitCliente">
     <nombre> nombreCliente </nombre>
     <usuario> nombreUsuario </usuario>
     <clave> claveUsuario </clave>
     <direccion> direccionCliente </direccion>
     <correoElectronico> eMailCliente </correoElectronico>
     <listaInstancias>
       <instancia id="idInstancia">
          <idConfiguracion> idConfiguracion </idConfiguracion>
          <nombre> nombreInstancia </nombre>
          <fechaInicio> descripcionFecha </fechaInicio>
          <estado> $estado </estado>
          <fechaFinal> $descripcionFecha </fechaFinal>
       </instancia>
       ...
     </listaInstancias>
     ...
   </cliente>
   ...
 </listaClientes>
</archivoConfiguraciones>
```

MENSAJE DE ENTRADA PARA NOTIFICAR EL CONSUMO DE LOS RECURSOS INSTANCIADOS EN LA NUBE DE TECNOLOGÍAS CHAPINAS, S.A.

Una vez los recursos, clientes e instancias se encuentren definidos en la nube de Tecnologías Chapinas, S.A., se reciben mensajes que notifican el tiempo utilizado por cada instancia, de manera que sea posible establecer los montos a facturar a cada cliente, dependiendo del tiempo que utilice sus instancias y el valor definido para cada recurso.

```
<?xml version="1.0"?>
<listadoConsumos>
  <consumo nitCliente="$nit" idInstancia="id">
    <tiempo> $tiempoConsumido </tiempo>
    <fechaHora> $descripcionFechaHora </fechaHora>
  </consumo>
  ...
</listadoConsumos>
```

Reglas y consideraciones del formato para cada mensaje recibido:

- **Tipos de recurso ($tipoRecurso):** Los recursos que forman una configuración de infraestructura pueden ser de tipo “Hardware” o “Software”.
- **Fechas ($decripcionFecha):** Todos los datos que sean de tipo fecha podrán contener cualquier hilera, siempre y cuando contenga una secuencia de la forma dd/mm/yyyy que represente una fecha válida, cualquier otra información será descartada. Los campos fecha podrán traer valores como éstos: “01/01/2025”, “Guatemala, 01/01/2025”, “en la ciudad de Guatemala, 01/01/2025 se brinda el servicio…“, en todos los casos anteriores se debe extraer la fecha 01/01/2025. Si vinieran 2 o más fachas válidas en una hilera de tipo fecha, entonces se utilizará la primera fecha válida que se encuentre en la hilera.
- **Fechas y horas ($descripcionFechaHora):** Todos los datos que sean de tipo fecha y hora podrán contener cualquier hilera, siempre y cuando contenga una secuencia de la forma dd/mm/yyyy hh24:mi. Estas hileras aplican las mismas reglas que las hileras de tipo fecha.
- **Estado de la instancia ($estado):** Toda instancia registrada por un cliente podrá estar en estado “Vigente” o “Cancelada”. Toda instancia “Cancelada” deberá tener una fecha final de la vigencia, de otra forma, la fecha de final de vigencia no tendrá valor.
- **Atributo NIT ($NIT)**: Debe ser un NIT válido, es decir, una sucesión de números del 0 al 9 seguidos por un guion y un dígito de validación (valor entre 0 y 9 o K). Ejs. 34300-4, 110339001-K, etc.
- **Consumos ($tiempoConsumido):** Todos los consumos se reportan en horas (tag tiempo), este valor puede contener decimales, por ejemplo, si el recurso se utilizó por 1 hora y 45 minutos, se reportará un uso de 1.75 horas.

ARQUITECTURA

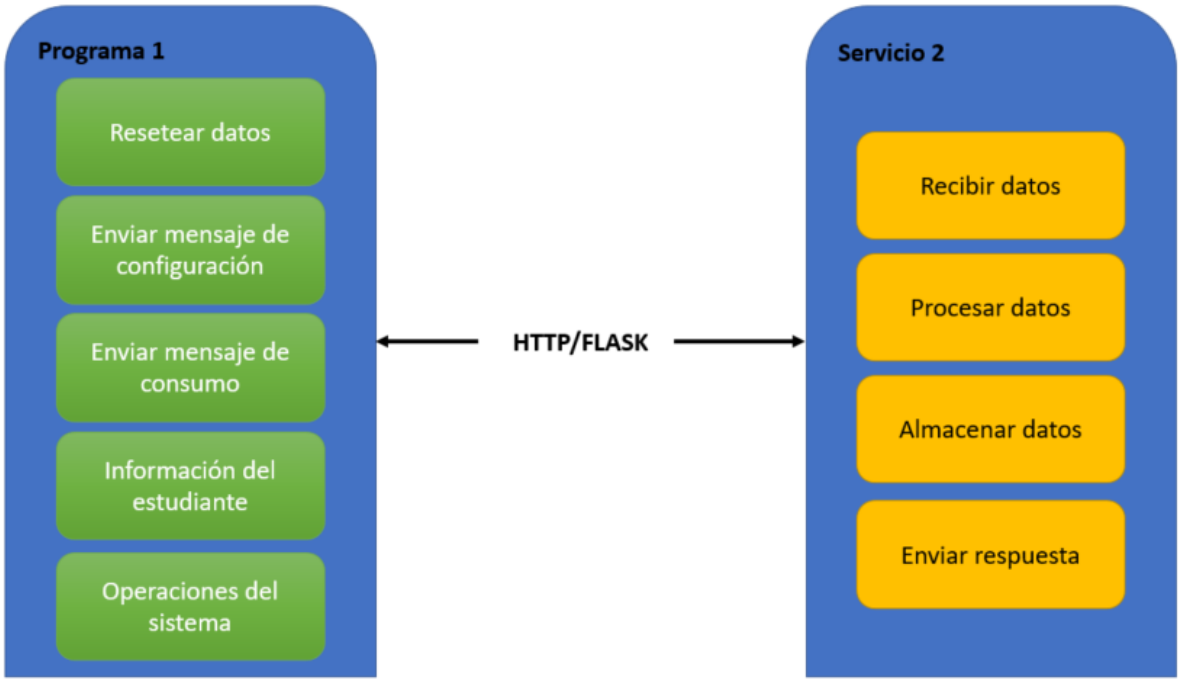

Fig. 1 – Arquitectura general de la aplicación

Programa 1 - Frontend

Este programa consiste en una aplicación Web y consistirá en un simulador de la aplicación principal, contendrá las funcionalidades necesarias para testear el buen funcionamiento de la API (Servicio 2) y la correcta funcionalidad de los procesos para gestionar la nube de Tecnologías Chapinas, S.A.; en esta aplicación se podrán mostrar los eventos que se procesarán y los datos estadísticos que fueron almacenados en la base de datos XML de salida.

Para realizar el frontend deberá utilizarse el framework **Django**, el cual trabaja con el patrón MVT (Modelo-Vista-Template).

Componentes:

- **Enviar mensaje de configuración:** Se desplegará una pantalla para gestionar el envío del mensaje de entrada con extensión .xml con una o varias solicitudes de creación de elementos del sistema. Deberá mostrar un mensaje indicando si el mensaje fue enviado exitosamente y el resultado de este, por ejemplo, 3 nuevos clientes creados, 7 nuevas instancias creadas, 2 nuevos recursos creados, 1 nueva categoría creada.
- **Enviar mensaje de consumo:** Se desplegará una pantalla para gestionar el envío del mensaje de entrada con extensión .xml con una o varias solicitudes de consumo de recursos del sistema. Deberá mostrar un mensaje indicando si el mensaje fue enviado exitosamente y el resultado de este, por ejemplo, 3 consumos procesados.
- **Operaciones del sistema:** En este apartado se debe de tener las siguientes opciones:

- ❖ **Inicializar sistema:** Al seleccionar esta opción se eliminarán todos los datos en la base de datos para iniciar una prueba sin datos previos.
- ❖ **Consultar Datos:** Al seleccionar esta opción se podrá chequear la estructura de información que actualmente maneja el sistema, es decir, podrá ver las categorías, recursos y configuraciones disponibles, o bien, los clientes e instancias registradas, así como los recursos pendientes de facturar por cada instancia.
- ❖ **Creación de nuevos datos:** Al seleccionar esta opción se podrá crear nueva información, es decir, podrá crear nuevas categorías, nuevos recursos, nuevas configuraciones, nuevos clientes o registrar nuevas instancias, cancelar instancias. Esta opción debe consumir los mismos API's que utiliza para dar servicio a los mensajes de configuración.
- ❖ **Proceso de facturación:** Al seleccionar esta opción se podrá elegir un rango de fechas y se generará una factura para cada cliente por todos los recursos consumidos no facturados previamente. La factura incluye la siguiente información: Número de factura (debe ser único), NIT del cliente al que se le emite la factura, fecha de la factura (último día incluido en el rango seleccionado), monto a pagar.
- ❖ **Reportes en PDF:** Deberá generar 2 reportes con las siguientes características:
  - ➢ **Detalle de factura:** Este reporte permitirá seleccionar una factura y presentará los datos de la factura y el detalle de lo que incluye la factura. Debe ser posible ver el monto a pagar por cada instancia, los tiempos en que fue consumida cada instancia y deberá mostrar el detalle de recursos, así como el aporte de cada recurso al cobro de la factura.
  - ➢ **Análisis de ventas:** Este reporte permitirá analizar la información de venta, permitiendo 2 opciones: a) Analizar las categorías y configuraciones que más ingresos generan para la empresa en un rango de fechas, b) Analizar los recursos que más ingresos generan para la empresa en un rango de fechas.

- **Ayuda:** desplegará 2 opciones, una para visualizar información del estudiante y otra para visualizar la documentación del programa.

Servicio 2 - Backend

Este servicio consiste en una o varias APIs que brindarán 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".

Para la realización de este servicio debe utilizarse el framework **Flask**. El estudiante deberá definir por su propia cuenta los métodos que necesitará para la realización de este servicio. Esto significa que debe implementar tantos métodos como necesite para consumir la API.

NOTA: Durante la calificación de este proyecto, el Servicio 2 podrá ser consumido desde otro cliente, por ejemplo, Postman.

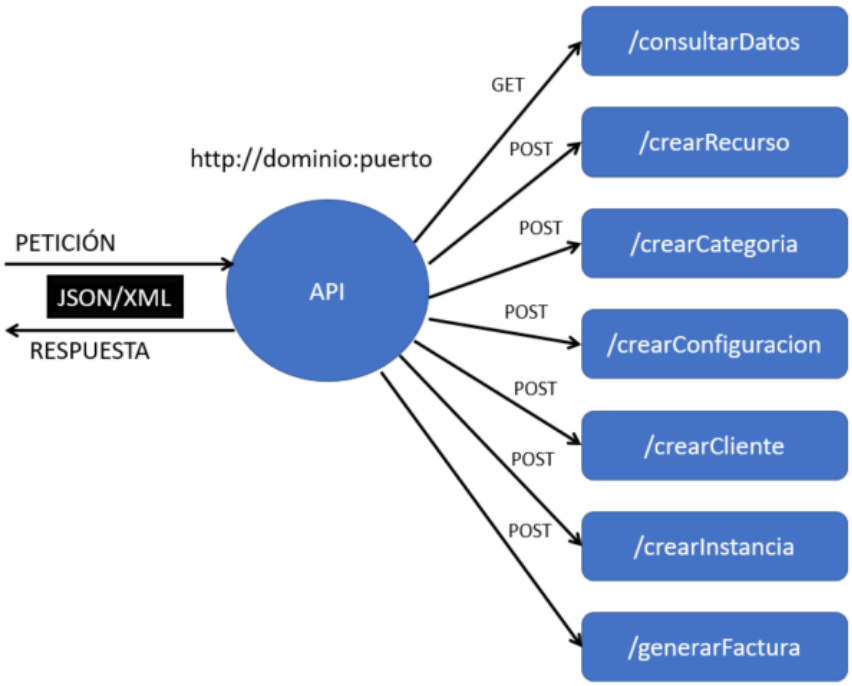

Fig. 4 - Ejemplo de la estructura de un API

CONSIDERACIONES

Debe utilizarse versionamiento para el desarrollo del proyecto. Se utilizará la plataforma Github en la cual se debe crear un repositorio en el que se gestionará el proyecto. Se deben realizar 4 releases o versiones del proyecto (se recomienda realizar una por semana del tiempo disponible). Se deberá agregar a su respectivo auxiliar como colaborador del repositorio. El cuarto release será el release final y se deberá de realizar antes de entregar el proyecto en la fecha estipulada.

DOCUMENTACIÓN

Para que el proyecto sea calificado, el estudiante deberá entregar la documentación utilizando el formato de ensayo definido para el curso. En el caso del proyecto, el ensayo debe tener entre 4 y 7 páginas de contenido, este máximo no incluye los apéndices o anexos donde se pueden mostrar modelos y diseños utilizados para construir la solución, como mínimo se debe incluir el diagrama de clases y el modelo de datos (relacional y XSD) utilizado para crear la solución al proyecto.

RESTRICCIONES

- Solo se permitirá la utilización de los IDEs y frameworks discutidos en el laboratorio.
- **Uso obligatori**o de programación orientada a objetos (**POO**).
- El nombre del repositorio debe de ser **IPC2_Proyecto3_#Carnet.**
- El estudiante debe entregar la documentación solicitada para poder optar a la calificación.
- Se calificará el cuarto release almacenado en el repositorio Github. Los cambios realizados después de ese release no se tomarán en cuenta.
- Para dudas concernientes al proyecto se utilizarán los foros en UEDI de manera que todos los estudiantes puedan ver las preguntas y las posteriores respuestas.
- **NO HABRÁ PRÓRROGA.**

ENTREGA

- La entrega será el **24 de octubre** a las 23:59 como máximo.
- La entrega será por medio de la UEDI.
- La documentación debe estar subida en el repositorio en una carpeta separada.
- Para entregar el proyecto en UEDI se deberá subir un archivo de texto con el link del repositorio.