

---

## PROYECTO 3 LABORATORIO – SISTEMA TECNOLOGIAS CHAPINAS, S.A

---

201906795 – Javier Ricardo Yllescas Barrios

### Resumen

El programa presentado propone el desarrollo un sistema de facturación para servicios en la nube, compuesto por un frontend en Django y un backend en Flask. El sistema procesa datos mediante archivos XML para gestionar recursos, clientes y configuraciones, calculando consumos y generando facturas y reportes.

El objetivo de la aplicación es usar paradigmas de programación orientada a objetos para la construcción del software, la utilización de base de datos XML y la interacción entre dos aplicaciones con diferentes tecnologías utilizando el protocolo HTTP para interactuar con el backend.

### Palabras clave

Proyecto2, Sistema Frontend (Django), Sistema Backend (Flask), USAC, IPC2.

### Introducción

El presente proyecto tiene como objetivo desarrollar una solución tecnológica integral para la empresa Tecnologías Chapinas, S.A., que le permita administrar su portafolio de servicios de nube y realizar los procesos de facturación correspondientes al consumo de recursos por parte de sus clientes.

La solución propuesta implementa una arquitectura moderna que combina un frontend web desarrollado con Django, que servirá como interfaz de gestión, y un backend construido con Flask que proveerá servicios API para el procesamiento de datos. El sistema utilizará archivos XML para la comunicación y persistencia de datos, aplicando los principios de la programación orientada a objetos y expresiones regulares para el procesamiento de información.

### Desarrollo del tema

Para una mejor compresión del proyecto se aplicaron los siguientes archivos para desarrollarlo.

### Estructura del Proyecto [Archivos]

- DOCUMENTACION\_IPC2\_Proyecto3\_20190695.pdf
  - >ArchivosdePrueba
  - >Backed
    - AppFlask.py
    - ArchivoConfiguraciones.xml
    - ArchivoConsumos.xml
    - entrada.xml

- entradaconsumos.xml

—>Sistemas

- SistemaCentral.py
- SistemaLecturaXML.py
- SistemaLecturaXMLconsumos.py
- SistemaSalidaXML.py
- SistemaSalidaXMLconsumos.py
- SistemaValidaciones.py

—>Clases

- ArchivoConfiguracion.py
- ArchivoConsumos.py

—>Frontend

—>app

- \_\_init\_\_.py
- admin.py
- apps.py
- models.py
- tests.py
- urls.py
- views.py

—>migrations

—>static

—>templates

—>Frontend

(Generados por Django)

### **Funcionamiento Frontend**

Donde:

—>**app**

Guarda todos los relacionado a la aplicacion en Django.

**app/urls.py**

Estan todas las Rutas y asignacion de sus funciones

**app/views.py**

Funciones para capturar datos y renderizar html

### **Funcionamiento Backend**

—>**Backed**

Almacena todo lo realcionado a la aplicacion con flask

**Backed/AppFlask.py**

Contiene todas las rutas para los endpoint para la interaccion con la aplicacion

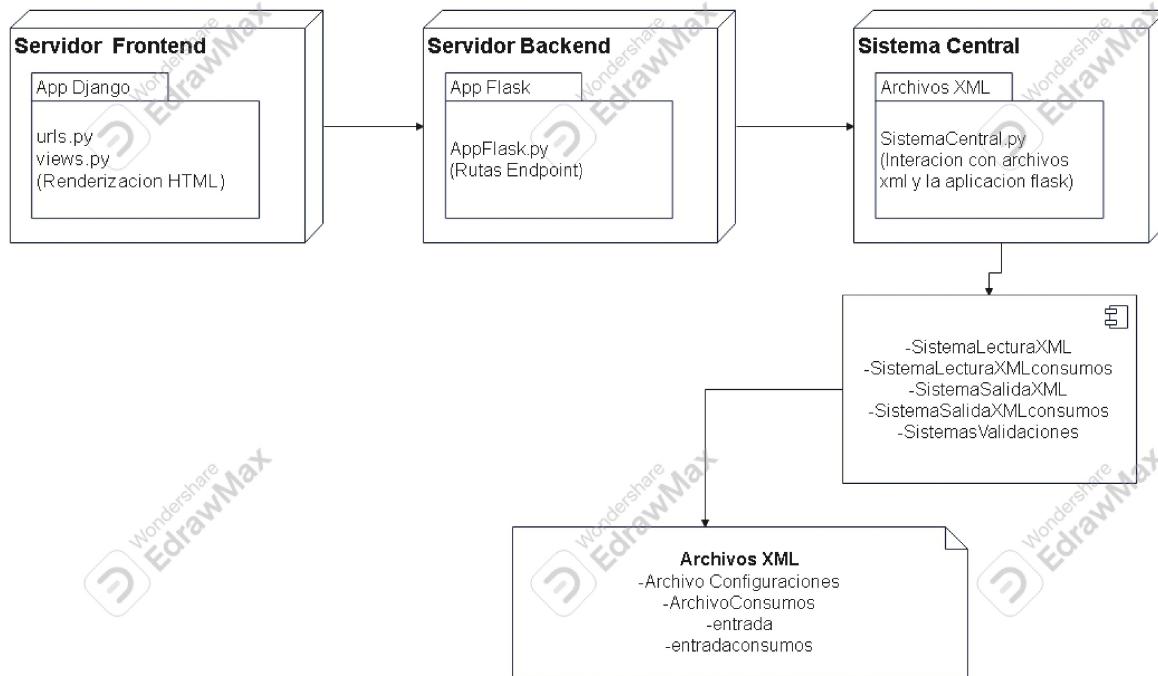
—>**Backed/Sistemas**

Estan todos los sistemas para ejecutar la aplicacion con la base de datos XML.

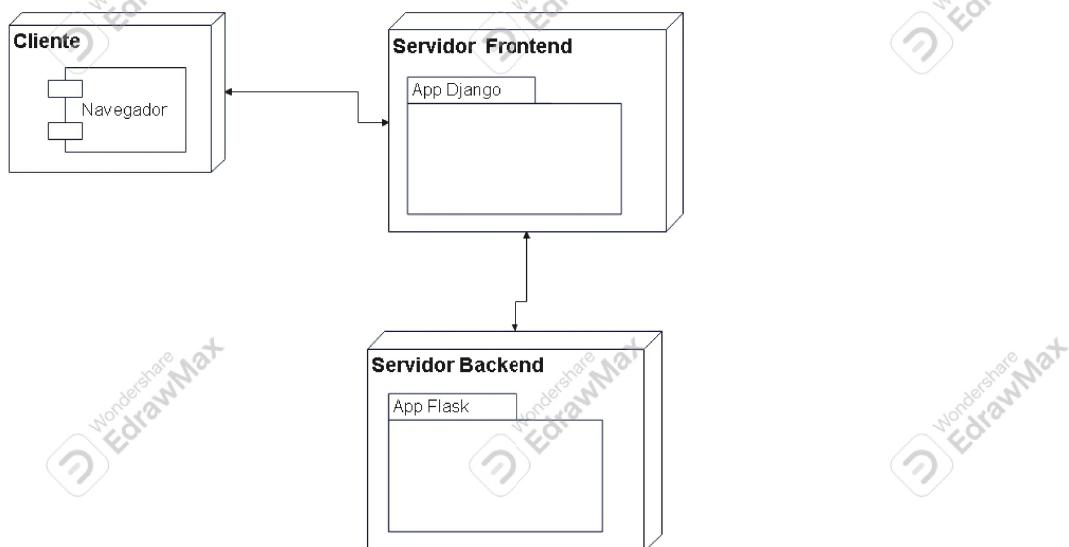
**Backed/Sistemas/SistemaCentral.py**

Es el encargado de recibir de flask las instrucciones para poder interactuar con los archivos XML y obtener las clases y creacion, eliminacion y modificacion de los archivos XML

## MODELO DE LA APLICACION

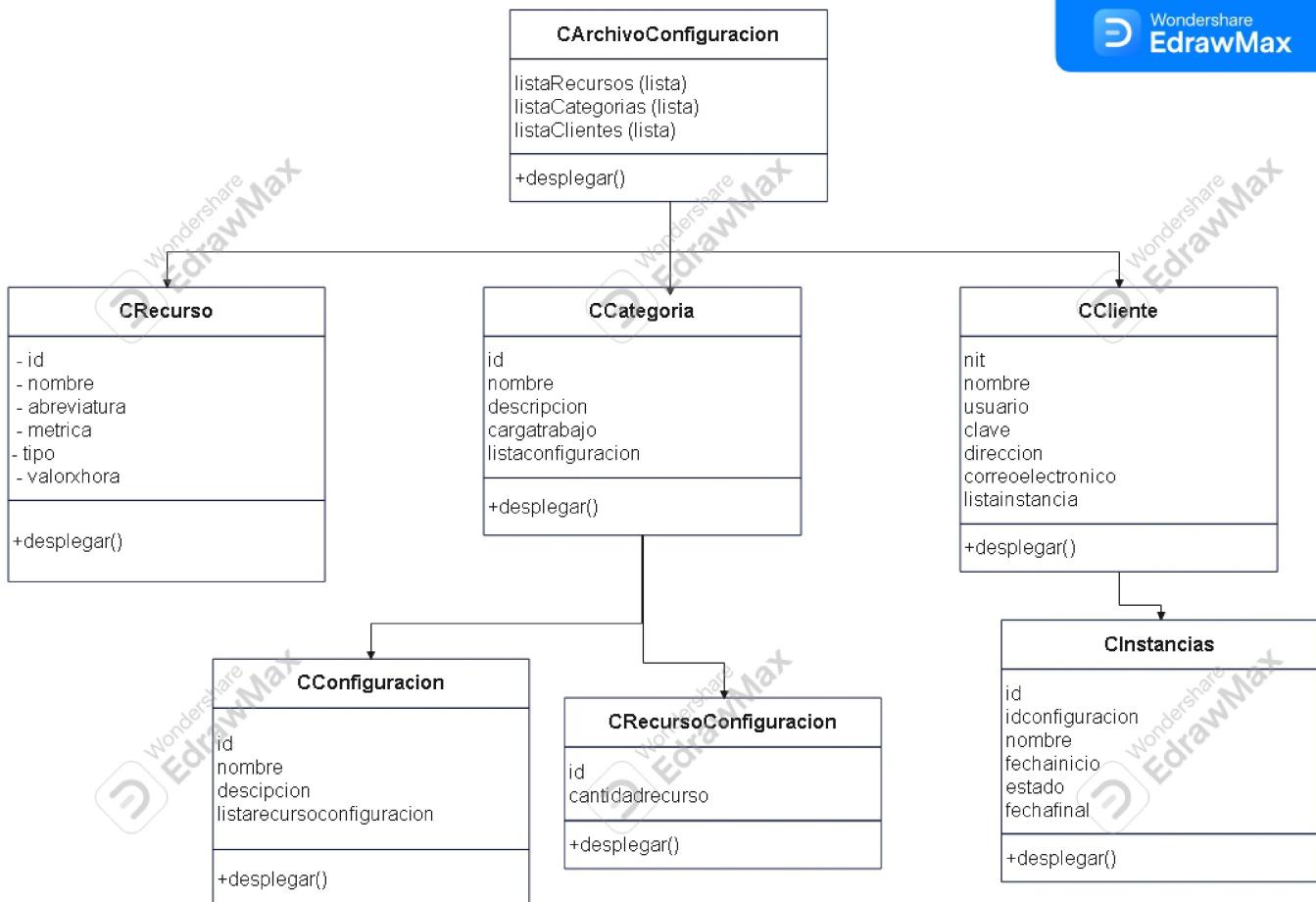


## DIAGRAMA DE DESPLIEGE



### CLASES SISTEMA CENTRAL (ARCHIVOS XML)

Wondershare  
**EdrawMax**



| SistemaCentral   |
|--|
| SisLeerArhvXML<br>ArchivoConfiguracion<br>SisSalidaXML<br>SisVal<br>mensajeErroresXML<br>SisLeerArhvXMLCons<br>ArchivoConsumos<br>SisSalidaXMLCons |

| SistemaLeerArchivosXML  |
|---|
| ruta<br>contenidoXML<br>domXML<br>XMLArchivonvofiguracion<br>CArchivoConfiguracion<br>ListaRecuros<br>ListaCategorias<br>ListaClientes<br><br>obtenerArchivoConfiguracion()<br>msg(mensaje, extra=None)<br>asignarruta(ruta)<br>SegmentarArchivo()<br>obtenerlistaclientes()<br>obtenerlistaCategorias()<br>obtenerlistaRecursos()<br>leerArchivo() |

| SistemaLeerArchivosXMLco<br>nsumos  |
|---|
| ruta<br>contenidoXL<br>domXML<br>ArchivoListaConsumos<br><br>obtenerArchivoListaConsumos(<br>)<br>msg(mensaje, extra=None)<br>leerArchivo()<br>asignarruta(ruta)<br>obtenerArchivoConsumo()<br>SegmentarArchivo() |

| SistemaSalidaXML   |
|--|
| rutasalida<br>ArchivoConfig<br>doc<br>root<br><br>asignarruta(ruta)<br>asignarArchivoConfiguraciones<br>(archivo)<br>GuardarArchivoConfiguraicione<br>s()<br>segmentar_archivo_XML()<br>creararchivoDOC()<br>crear_archivo()<br>msg(mensaje, extra=None) |

### SistemaSalidaXMLconsumos

rutasalida  
ArchivoCons  
doc  
root

GuardarArchivoConfiguracion()  
s()  
segmentar\_archivo\_XML()  
creararchivoDOC()  
crear\_archivo()  
msg(mensaje, extra=None)  
asignarruta(ruta)  
asignarArchivoConfiguraciones(archivo)

### SistemaValidaciones

msjErrores  
ArchivoConfiguracion  
ListaRecursos  
ListaCategorias  
ListaClientes  
ArchivoConsumos

obtenerArchivoConfiguracion()  
obtenerArchivoConsumos()  
obtenermensajeerrores()  
ValidarArchivoConsumos()  
validarfechaHora(txtfechahora)  
validatiempoconsumido(txtimepo)  
ValidarArchivoConfiguracion()  
ValidacionCliente(cliente)  
validainstancias(listainstancias)  
}  
validarfecha(txtfecha)  
validamit(nit)  
ValidacionRecurso(recurso)  
validarOpciones(opcionevaluar, opcionesvalidas)  
msg(mensaje, extra=None)  
asignarArchivoConfiguracion(Clase)  
asignarArchivoConsumoClientes(Clase)

## **Modelo de datos XSD**

### **Para el archivo de consumos clientes**

```
<?xml version="1.0" encoding="UTF-8"?>  
<xsd:schema  
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
    elementFormDefault="qualified">  
  
<!-- Elemento raíz -->  
<xsd:element name="listadoConsumos">  
    <xsd:complexType>  
        <xsd:sequence>  
            <xsd:element name="consumo"  
                type="consumoType" maxOccurs="unbounded"/>  
        </xsd:sequence>  
    </xsd:complexType>  
</xsd:element>  
  
<!-- Definición del tipo 'consumo' -->  
<xsd:complexType name="consumoType">  
    <xsd:sequence>  
        <xsd:element name="tiempo" type="xsd:string"/>  
        <xsd:element name="fechaHora"  
            type="xsd:string"/>  
    </xsd:sequence>  
    <xsd:attribute name="nitCliente" type="xsd:string"  
        use="required"/>  
    <xsd:attribute name="idInstancia" type="xsd:string"  
        use="required"/>  
</xsd:complexType>
```

</xsd:schema>

### **Descripción uso**

<listadoConsumos> - raíz - Contiene todos los registros de consumo  
<consumo> - complejo - Representa un consumo individual  
@nitCliente - string - Identificador del cliente  
@idInstancia - string - Instancia asociada al consumo  
<tiempo> - string - Duración del consumo (ej. "1.75 horas")  
<fechaHora> - string - Fecha y hora del consumo (ej. "02/04/2025 23:12")

### **Para el archivo configuracion**

```
<?xml version="1.0" encoding="UTF-8"?>  
<xsd:schema  
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
    elementFormDefault="qualified">  
  
<!-- Elemento raíz -->  
<xsd:element name="archivoConfiguraciones">  
    <xsd:complexType>  
        <xsd:sequence>  
            <xsd:element name="listaRecursos"  
                type="listaRecursosType"/>  
            <xsd:element name="listaCategorias"  
                type="listaCategoriasType"/>  
            <xsd:element name="listaClientes"  
                type="listaClientesType"/>
```

```
</xs:sequence>
</xs:complexType>
</xs:element>

<!-- ===== -->
<!-- LISTA DE RECURSOS -->
<!-- ===== -->
<xs:complexType name="listaRecursosType">
  <xs:sequence>
    <xs:element name="recurso"
      type="recursoType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="recursoType">
  <xs:sequence>
    <xs:element name="nombre" type="xs:string"/>
    <xs:element name="abreviatura"
      type="xs:string"/>
    <xs:element name="metrica" type="xs:string"/>
    <xs:element name="tipo" type="xs:string"/>
    <xs:element name="valorXhora"
      type="xs:decimal"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string"
    use="required"/>
</xs:complexType>

<!-- ===== -->
<!-- LISTA DE CATEGORÍAS -->
<!-- ===== -->
<xs:complexType name="listaCategoriasType">
  <xs:sequence>
    <xs:element name="categoria"
      type="categoriaType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="categoriaType">
  <xs:sequence>
    <xs:element name="nombre" type="xs:string"/>
    <xs:element name="descripcion"
      type="xs:string"/>
    <xs:element name="cargaTrabajo"
      type="xs:string"/>
    <xs:element name="listaConfiguraciones"
      type="listaConfiguracionesType"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string"
    use="required"/>
</xs:complexType>

<xs:complexType
  name="listaConfiguracionesType">
  <xs:sequence>
    <xs:element name="configuracion"
      type="configuracionType"
      maxOccurs="unbounded"/>
  </xs:sequence>
```

```
</xs:complexType>                                </xs:complexType>

<xs:complexType name="configuracionType">
  <xs:sequence>
    <xs:element name="nombre" type="xs:string"/>
    <xs:element name="descripcion"
      type="xs:string"/>
    <xs:element name="recursosConfiguracion"
      type="recursosConfiguracionType"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string"
    use="required"/>
</xs:complexType>

<xs:complexType
  name="recursosConfiguracionType">
  <xs:sequence>
    <xs:element name="recurso"
      type="recursoConfigRefType"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType
  name="recursoConfigRefType">
  <xs:simpleContent>
    <xs:extension base="xs:decimal">
      <xs:attribute name="id" type="xs:string"
        use="required"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

<!-- ===== -->
<!-- LISTA DE CLIENTES -->
<!-- ===== -->
<xs:complexType name="listaClientesType">
  <xs:sequence>
    <xs:element name="cliente" type="clienteType"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="clienteType">
  <xs:sequence>
    <xs:element name="nombre" type="xs:string"/>
    <xs:element name="usuario" type="xs:string"/>
    <xs:element name="clave" type="xs:string"/>
    <xs:element name="direccion"
      type="xs:string"/>
    <xs:element name="correoElectronico"
      type="xs:string"/>
    <xs:element name="listaInstancias"
      type="listaInstanciasType"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="nit" type="xs:string"
    use="required"/>
</xs:complexType>

<xs:complexType name="listaInstanciasType">
```

```
<xs:sequence>
  <xs:element name="instancia"
  type="instanciaType" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="instanciaType">
  <xs:sequence>
    <xs:element name="idConfiguracion"
    type="xs:string"/>
    <xs:element name="nombre" type="xs:string"/>
    <xs:element name="fechaInicio"
    type="xs:string"/>
    <xs:element name="estado" type="xs:string"/>
    <xs:element name="fechaFinal"
    type="xs:string"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string"
  use="required"/>
</xs:complexType>

</xs:schema>
```

### Como ejecutar el Proyecto.

Para ejecutar el proyecto necesitas tener python, flask, request, django, jsonify instalado y saber usar CMD

Paso 1: Ejecutar el Backend accede a la carpeta

Backend y ejecuta en CMD “python AppFlask.py”  
Paso 2: Ejecuta Frontend buscar la carpeta forntend y ejecuta en CMD “python manage.py runserver”  
Paso 3: Accede desde tu navegador a link <http://127.0.0.1:8000/>  
•Ingresa usuario admin y contraseña “1234”



Seleccion una opcion dle menu



Subir un archivo Configuracion XML al sistema

## Archivo Configuraciones

Archivo XML con las Configuraciones del sistema

Seleccionar archivo Sin archivos seleccionados

Cargar Procesar

Inicio



Subir un arhivo Consumos clientes XML

## Archivo Consumos Clientes

Archivo XML con las Consumos por cliente e instancia.

Seleccionar archivo Sin archivos seleccionados

Cargar Procesar

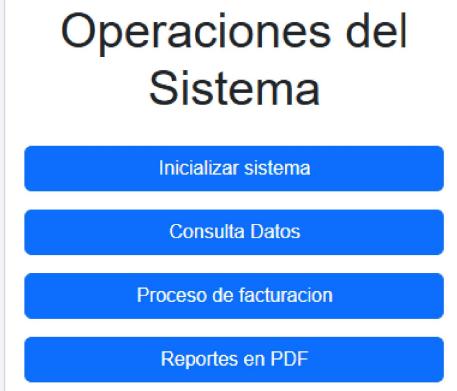
Inicio



Al ir al panel principal y selecciona operaciones del sistema aprecera las siguientes opciones

## Operaciones del Sistema

Inicializar sistema Consulta Datos Proceso de facturacion Reportes en PDF



Al consultar los datos se abrira un acordeon donde se puede visualizar todos los datos

### Consultar Datos

Víor centralizado de recursos, categorías de servicio y clientes del datacenter.

| ID     | Nombre            | Abreviatura | Métrica        | Tipo           | Valor por hora |
|--------|-------------------|-------------|----------------|----------------|----------------|
| RC-101 | Compute Optimized | C5          | Ghz reservados | Cómputo        | Q 4.50         |
| RC-204 | Storage IOPS Alto | STX         | IOPS           | Almacenamiento | Q 1.25         |
| RC-310 | Red 10Gbps        | NET10       | Gbps           | Networking     | Q 0.85         |



Si seleccionas agregar datos apreceran varios formularios para agregar datos

### Agregar Datos

Registra nuevos recursos, categorías, clientes e instancias para el datacenter.

Registrar Recurso Registrar Categoría Registrar Cliente Registrar Instancia

ID del recurso Nombre

Abreviatura Métrica

Tipo Valor por hora

Guardar recurso



Despues puede seleccionar la facturacion del cliente

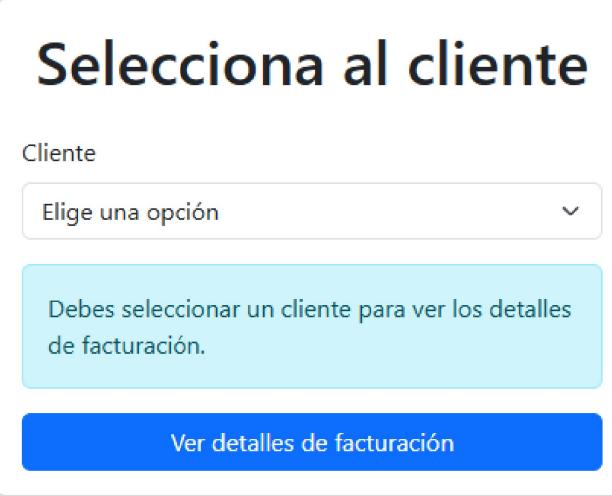
## Selecciona al cliente

Cliente

Elige una opción

Debes seleccionar un cliente para ver los detalles de facturación.

Ver detalles de facturación



Mostrara los gastos del cliente y la opcion de descargar la factura en un archivo PDF

Factura de consumo - Datacenter

Imprimir Descargar PDF

### Factura de Consumo

Detalle de instancias, horas consumidas y aporte de cada recurso al total facturado.

| CLIENTE                                  | HORAS TOTALES                 | IMPORTE TOTAL                                |
|--|-------------------------------|--|
| Soluciones Financieras GT<br>NTS-5487% K | 355 h<br>4 instancias activas | Q 11,408.00<br>Cómputo, almacenamiento y red |

Resumen por instancia

| Instancia | Configuración | Nombre             | Horas consumo | Recursos principales                      | Subtotal   |
|-----------|---------------|--------------------|---------------|---|------------|
| IN-9001   | DB-XL         | Core Bancario      | 120 h         | RC-101: 8 unidades<br>RC-204: 4 unidades  | Q 4,920.00 |
| IN-9034   | AN-CORE       | Analytics Regional | 95 h          | RC-101: 10 unidades<br>RC-310: 4 unidades | Q 4,598.00 |
| IN-9102   | AN-EDGE       | Seguimiento IoT    | 80 h          | RC-310: 6 unidades<br>RC-204: 3 unidades  | Q 708.00   |
| IN-9140   | DB-MED        | Catálogo Omnicanal | 60 h          | RC-101: 4 unidades<br>RC-310: 2 unidades  | Q 1,182.00 |

Instancia IN-9001 - Core Bancario

| Cliente: Soluciones Financieras GT | Periodo: 01-07 abr 2025 | Horas consumo: 120 h | Subtotal: Q 4,920.00 |                 |            |
|------------------------------------|-------------------------|----------------------|----------------------|-----------------|------------|
| Recurso                            | Métrica                 | Unidades             | Horas                | Tarifa por hora | Aporte     |
| Core Bancario                      | Horas consumo           | 120                  | 120                  | Q 41.00         | Q 4,920.00 |

El botón ayuda mostrara la información del estudiante y otro botón mostrara esta documentación

## Ayuda

Selecciona la acción que deseas realizar.

[Ver información del estudiante](#)

[Ver documentación del programa](#)

## Información del alumno

**Nombre:** Javier Ricardo Yllescas Barrios  
**Carné:** 201906795

## Conclusiones

Implementación de una Solución Integral: El proyecto permitió desarrollar un sistema completo de gestión y facturación para servicios en la nube, integrando exitosamente tecnologías modernas como Django para el frontend, Flask para el backend API, y XML como medio de persistencia de datos. La aplicación de los principios de programación

orientada a objetos aseguró la creación de un software modular, mantenable y escalable, capaz de manejar la complejidad de los procesos de negocio de Tecnologías Chapinas, S.A.

**Cumplimiento de Requerimientos Técnicos y Funcionales:** Se logró cumplir con todos los objetivos específicos planteados, incluyendo el consumo y procesamiento de mensajes XML, la validación de datos mediante expresiones regulares, la generación de reportes en PDF y la implementación de un sistema de facturación basado en consumo de recursos. El uso de control de versiones con GitHub facilitó un desarrollo organizado y metódico, resultando en una solución robusta que satisface las necesidades de gestión de infraestructura cloud y facturación requeridas por la empresa.

Flask al ser de un framework basado en python es muy sencillo para ejecutar toda nuestra lógica de back-end

## Comentarios:

Este proyecto representó una excelente oportunidad para aplicar conceptos clave de desarrollo de software en un contexto parecido al real, integrando backend, frontend, bases de datos y APIs, mientras se usa la programación orientada a objetos enseñada utilizando clases específicas para almacenar toda la información y tener un mejor manejo

## Referencias bibliográficas

Universidad de San Carlos de Guatemala. (s.f.).

Enunciado proyecto 1.

[https://uedi.ingenieria.usac.edu.gt/campus/pluginfile.php/270932/mod\\_resource/content/1/%5BIPC2%5DProyecto\\_3\\_2S2025-v2.pdf](https://uedi.ingenieria.usac.edu.gt/campus/pluginfile.php/270932/mod_resource/content/1/%5BIPC2%5DProyecto_3_2S2025-v2.pdf)

Argueta, Hesban. (s.f.). *Clases laboratorio IPC2*

[Repositorio en GitHub]. GitHub.

<https://github.com/Hes-007/IPC2-2S2025/tree/main>.

Ruiz Juarez, J. M. (s.f.). Contenido Unidad 2 y 3.

<https://uedi.ingenieria.usac.edu.gt/campus/course/view.php?id=2547>

Flask (s.f) Documentacion para uso framework

Flask

<https://flask.palletsprojects.com/en/stable/tutorial/>