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 2023.

PROYECTO 3 IPC2

202001950 – Fernando Misael Morales Ortíz

El proyecto consiste en un API la cual trata de una aplicacion cliente servidor donde se crea un backend para almacenar toda la parte logica con el servidor flask y la parte del frontend se realizo con el framework Django donde se crearon las plantillas para las vistas y ademas trabaja con peticiones HTTP para poder conectarse con el servidor y realizar calculos de porcentajes para saber a que perfil encaja ya que se tiene una entrada donde analiza todo eso y nos da la respuesta.

En las plantillas se manejo toda la vista donde se usaron formularios intuitivos para que al usuario pueda usarlos sin ningun problema

The project consists of an API which deals with a client-server application where a backend is created to store all the logical part with the flask server and the frontend part was made with the Django framework where the templates for the views were created and also It works with HTTP requests to be able to connect to the server and perform percentage calculations to find out which profile it fits, since there is an entry where it analyzes all that and gives us the answer.

In the templates, the entire view was handled where intuitive forms were used so that the user can use them without any problem.

the data of an xml file.list which was storing the data from an xml file.

API: interfaz de programación de aplicaciones

Framework: es un marco o esquema de trabajo generalmente utilizado por programadores para realizar el desarrollo de software

Palabras clave:

Clase: elemento de la programación orientada a objetos que sirve como una plantilla para definir características de la misma.

Instancia: representa a una clase y se crea para poder obtener elementos desde otro archivo sin tener que crearlos de nuevo.

Interfaz: es un conjunto de herramientas que hacen una comunicación mas sencilla con el usuario y el contenido del programa.

Class: object-oriented programming element that serves as a template to define its characteristics.

Instance: represents a class and is created to be able to obtain

elements from another file without having to create them again.

API: application programming interface

Framework:
It is a framework or scheme of work generally used by programmers to carry out software development.

Interface: it is a set of tools that make communication easier with the user and the content of the program.

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 2023.

Desarrollo:

Se creo una aplicación basada en el modelo cliente servidor donde se tenia la parte de la lógica Backend y la parte visual Frontend en la parte de la lógica se manejo con el framework Flask el cual maneja peticiones y diferentes endpoints los cuales sirven par poder hacer diferentes peticiones y la parte visual se manejo con el framework el cual trabaja mediante plantilla y peticiones las cuales se fueron manejando para poder obtener información y esta se proceso en la parte del backend para poder tener una respuesta

Backend:

En esta parte se utilizaron dos archivos los cuales tenían el menú principal donde se manejaban todas apps con sus rutas y en archivo de procesar se realizaron las gestiones para poder leer archivos xml y tener sus datos que

```python
from flask import Flask,request,jsonify,Response
from procesar import ProcessData, ProceData

app = Flask(__name__)


@app.route("/")
def hello_world():
    return jsonify({"MSG":"Hola a todos esto es de parte de flask"})

@app.route('/analizar',methods=['POST'])
def analizar():
    if request.method=='POST':
        return Response(ProcessData(request.data),status=200)

@app.route("/mostrar",methods=['POST'])
def mostrar():
    if request.method=='POST':
        return Response(ProceData(request.data),status=201)

if __name__=='main':
    app.run(debug=True)
```

se utilizarían para poder hacer cálculos de porcentajes de perfiles y poder obtener todos los datos necesarios ya que se necesitaba hacer comparaciones para poder determinar los porcentajes.

Frontend:

Aca se trabajo con el framework Djando que trabaja con peticiones y plantillas en la cual se realizo en HTML para poder tener una mejor vista y que el usuario no tenga problemas para poder usarla y para poder ingresar en las diferentes direcciones únicamente es necesario cambiar la dirección la cual se conecta con el servidor en flask para poder tener respuestas de las peticiones que se están realizando pero al ser una framework que trabaja con modelo vista controlador cada cambio que se realizaba se necesitaba actualizar la parte de las peticiones y las URL para que no tuviera tropiesos y se manejara de la mejor manera.

Vistas: en esta parte es donde se unió la parte de flask con las peticiones que el usuario realizaba

```
def mis_datos(request):

    return render(request,'datos.html')

def perfil(request):
    if request.method=='POST':
        archivo=request.FILES['file']

        #headers={'content-Type': "aplicacion/xml"}
        response = requests.post('http://127.0.0.1:5000/analizar', data=archivo)

        if response.status_code==200:
            xml=response.content.decode("utf-8")
            print(str(xml))
            respuesta="El perfil es: "+ xml
            return render(request,'ocupacion.html',{'respuesta':respuesta})
    else:
        return render(request,'ocupacion.html')
```

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 2023.

Url: en esta parte es donde se tenían las diferentes peticiones las cuales se manejan y cambian en el navegador para poder hacer diferentes accione.

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.Welcome),
    path('archivo/',views.perfil),
    path('data/',views.mis_datos),
    path('vista/', views.mensajes)


]
```

Código de plantilla: esta parte usa comandos especiales los cuales los utiliza cuando el servidor manda una respuesta positiva y así poder continuar dando acceso a otra parte del código y saber si realmente se esta recibiendo alguna respuesta.

```
<!DOCTYPE html>
<html>
<head>
    <title>ChapinChat</title>
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/config.css' %}">

</head>
<body>
    <div>
        <h2>Carcar archivo {{titulo}}</h2>
        <form method="POST" enctype="multipart/form-data">

            {% csrf_token %}
            <input type="file" name="file"/>
            <button type="submit"> carga archivo</button>
```

Universidad de San Carlos de Guatemala

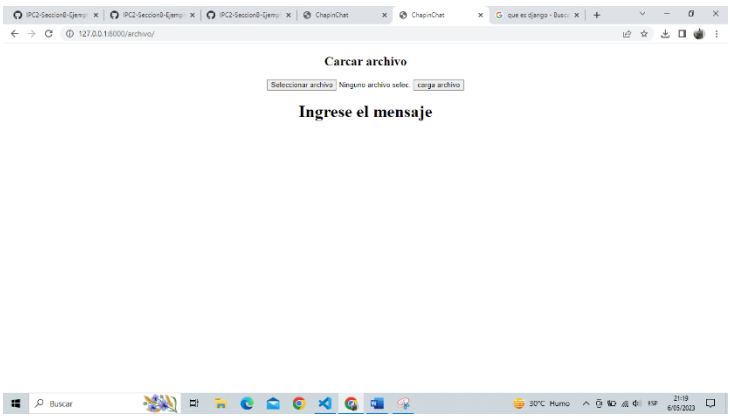
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería

Introducción a la programación y computación 2, 1er. Semestre 2023.

```
        </form>
        {% if respuesta %}
        <h3>{{respuesta}} </h3>
        {% endif %}
    </div>
     <h1>Ingrese el mensaje </h1>
     <form action="POST"></form>

</body>
</html>
```

Pantalla para cargar archivos: esta es vista donde el usuario puede ingresar un documeto en xml y lo analiza para poder tener toda información que será procesada

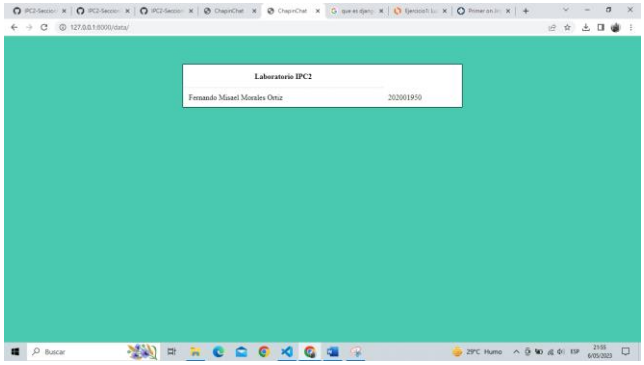


Pantalla de datos del estudiante: en esta parte se muestran los datos generales tales como el nombre del estudiante y el curso.

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 2023.



Conclusiones

- La programación orientada a objetos es la más adecuada para trabajar proyectos donde se necesitan diferentes módulos para almacenar datos
- Las listas enlazadas son la mejor opción para almacenar información variada ya que puede venir solo un dato o muchos además de ser versátil para poder crear mas espacios de almacenamiento.
- En una API cliente servidor se necesita trabajar de la mano para que las peticiones se puedan realizar de la mejor manera.
- Un framework permite agilizar los procesos de desarrollo ya que evita tener que escribir código de forma repetitiva.