|  |
| --- |
| **PROYECTO 4, FRONTEND, BACKEND** |
| **Carnet 1 – Nombre completo del estudiante** |

**Resumen**

Últimamente se está viendo mucho el tema de las api rest, cada vez siendo mas necesitada en lo que al mundo del internet se refiere, por lo tanto en este proyecto se buscó la manera de poner esos conceptos a trabajar y para ellos se utilizó lo también muy mencionado FrameWork, con esto y la ayuda de la herramienta llamada Flask se desarrollaron todas las funcionalidades requeridas para el proyecto, el cual consistía en tener dos servidores corriendo en paralelo, con la condición de que estos se tienen que comunicar el uno con el otro para poder obtener la información que se requiera.

**Palabras clave**

Parte delantera

Parte trasera

Flask

HTTP

Django

***Abstract***

*Lately we are seeing a lot the topic of the api rest, being more and more needed in the internet world, therefore in this project we looked for a way to put these concepts to work and for them we used the also mentioned FrameWork, with this and the help of the tool called Flask we developed all the functionality required for the project, which consisted of having two servers running in parallel, with the condition that they have to communicate with each other to obtain the information required.*

***Keywords***

*Frontend*

Backend

Flask

HTTP

**Introducción**

En este proyecto se desarrollaron varias ideas propuestas en el laboratorio, de las cuales la más importante es la comunicación entre servidores, tanto el frontend como el backend, por medio de lo que denominamos anteriormente como frontend realizamos una serie de peticiones la cual el servidor backend recibe y procesa para finalmente devolvernos una respuesta, todo dependiendo el tipo de método que sea. Hay varios tipos de métodos con los cuales podemos hacer peticiones, nombraremos los 4 más sonados en este mundo de la programación los cuales son: GET, POST, PUT, DELETE

**Desarrollo del tema**

Como mencionamos anteriormente existen muchos conceptos abarcados en el proyecto, por lo tanto procederemos a explicar a detalle cada uno de los mencionados, los cuales son:

a. Frontend

b. Backend

c. API REST

d. FrameWork

e. FLASK

f. Django

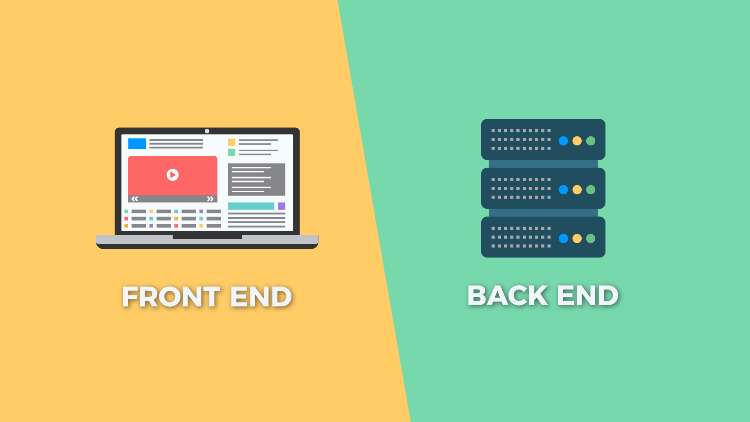
**Frontend:**

Es la parte de una aplicación que interactúa con los usuarios, es conocida como el lado del cliente. Básicamente es todo lo que vemos en la pantalla cuando accedemos a un sitio web o aplicación: tipos de letra, colores, adaptación para distintas pantallas (RWD), los efectos del ratón, teclado, movimientos, desplazamientos, efectos visuales… y otros elementos que permiten navegar dentro de una página web. Este conjunto crea la experiencia del usuario.

**Backend:**

Es el que está al interior de las aplicaciones que viven en el servidor y al que a menudo se le denomina “el lado del servidor”.

El back end del sitio web consiste en un servidor, una aplicación y una base de datos. Se toman los datos, se procesa la información y se envía al usuario. Los desarrolladores de Front end y Back end suelen trabajar juntos para que todo funcione correctamente.



*Figura 1.* Ilustración del frontend y backend

Fuente: Ken Ruiz Inoue, 2019.

**API REST:**

Es un conjunto de requisiciones que permite la comunicación de datos entre aplicaciones.

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

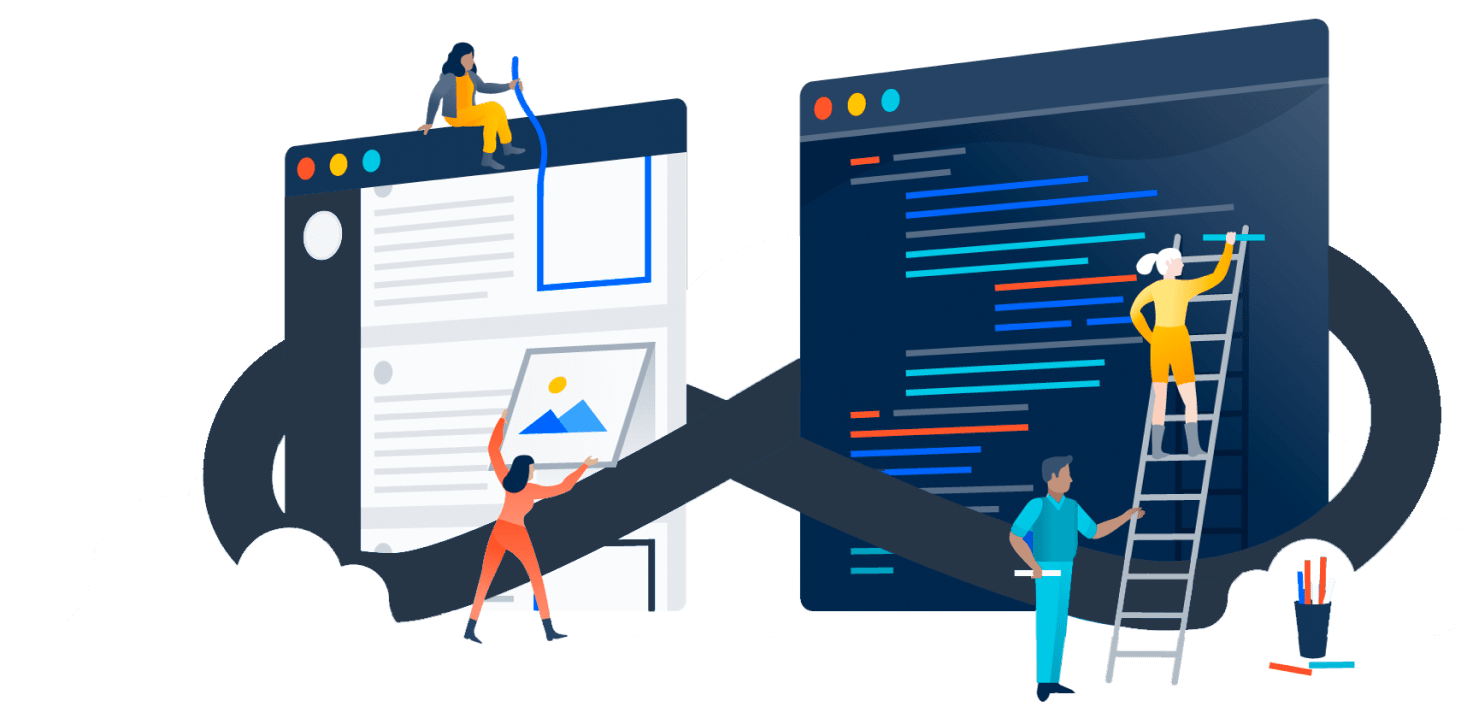


*Figura I1.* API REST

Fuente: Ivan de Souza, 2020.

**FrameWork:**

Es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.



*Figura II1.* FrameWork

Fuente: phalcon.io, 2020.

**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 IV.* Flask

Fuente: flask, 2021.

**Django:**

Django es un framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como modelo–vista–controlador.



*Figura IV.* Django

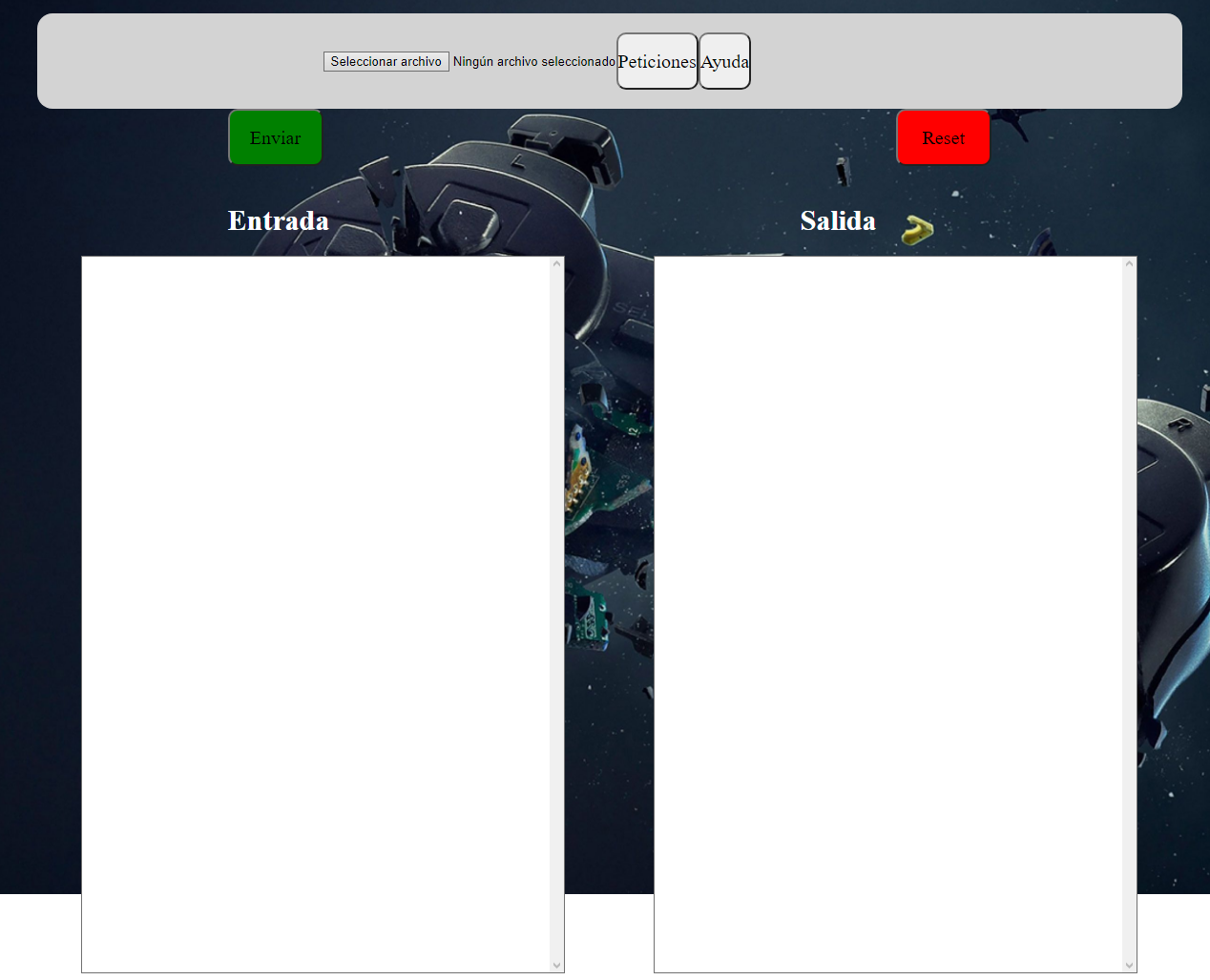
Fuente: django, 2021.

**Funcionalidades del programa:**

**Página Principal:**

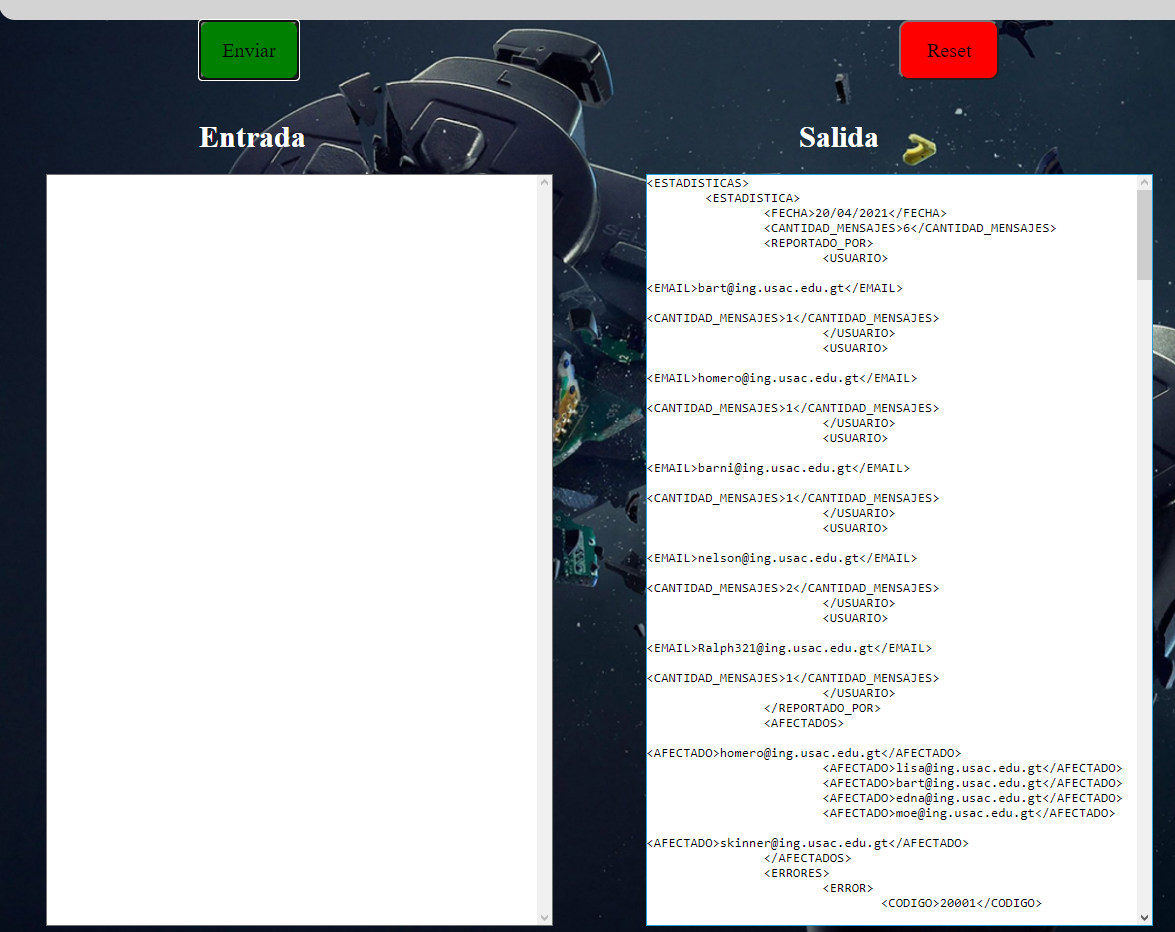
Hay varias opciones en esta pestaña, la de seleccionar archivo nos despliega una ventana para consultar en nuestra maquina el archivo que deseemos abrir.

**Nota:** *La estructura del archivo ingresado tiene que corresponder al de un archivo xml.*



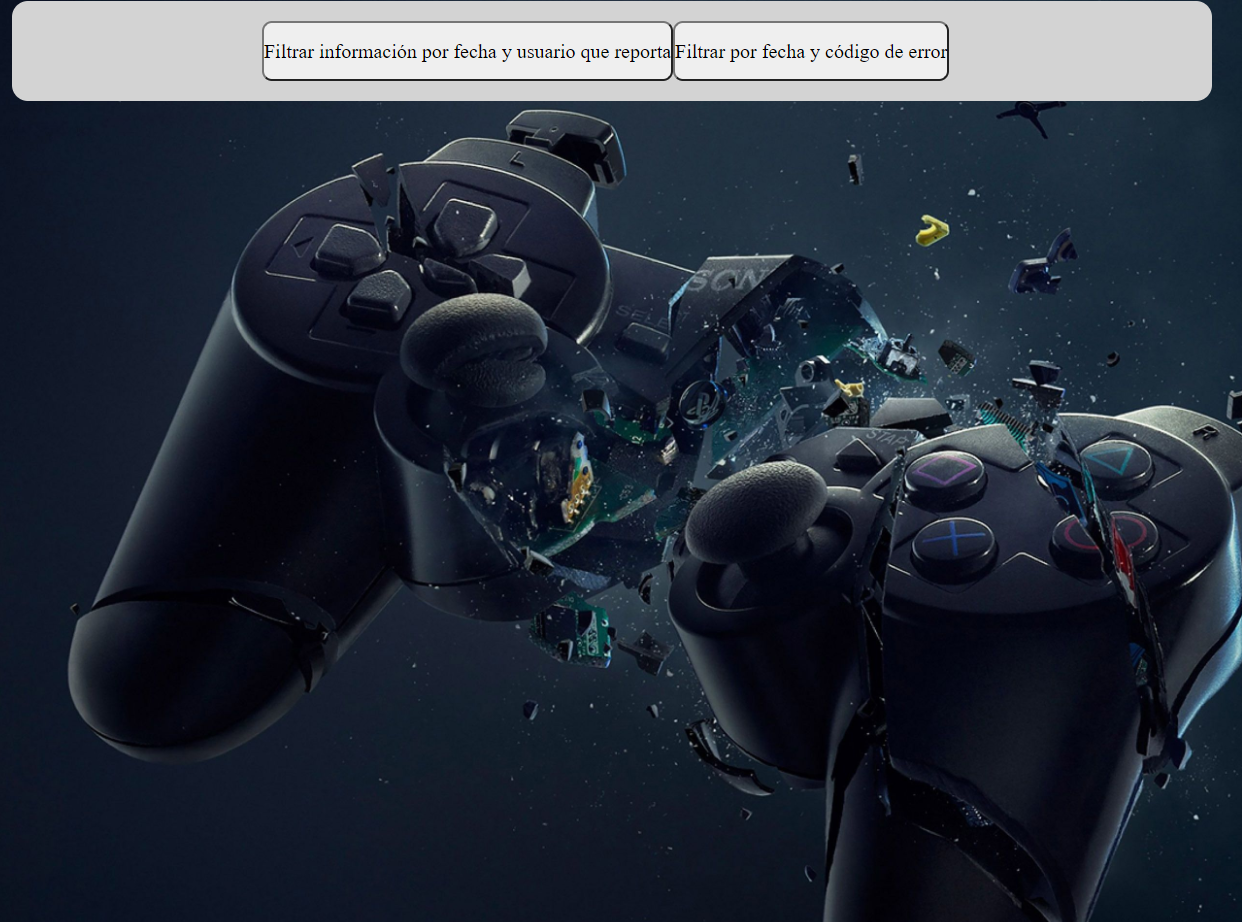
**Consultar Datos:**

Al presionar el botón enviar, el servidor frontend se comunicará con el backend solicitando la información requerida, la cual se mostrará en el recuadro de la derecha, debajo de la etiqueta salida.

****

**Peticiones:**

Al seleccionar esta opción se desplegará la siguiente ventana, la cual cuenta con dos opciones de filtrado:

****

**Conclusiones**

* La buena práctica de codificación es muy importante, tanto para entender mejor las funcionalidades del programa, como para evitar las ambigüedades en el código.
* Los framework son herramientas muy útiles para facilitarnos muchas de las tareas que se vuelven un tanto tediosas o confusas por sí mismas.
* Python nos proporciona un sinfín de posibilidades a la hora de trabajar proyectos, tanto frameworks como librerías muy útiles.

**Referencias bibliográficas**

Django, 2021:

<https://www.djangoproject.com>

Flask, 2021:

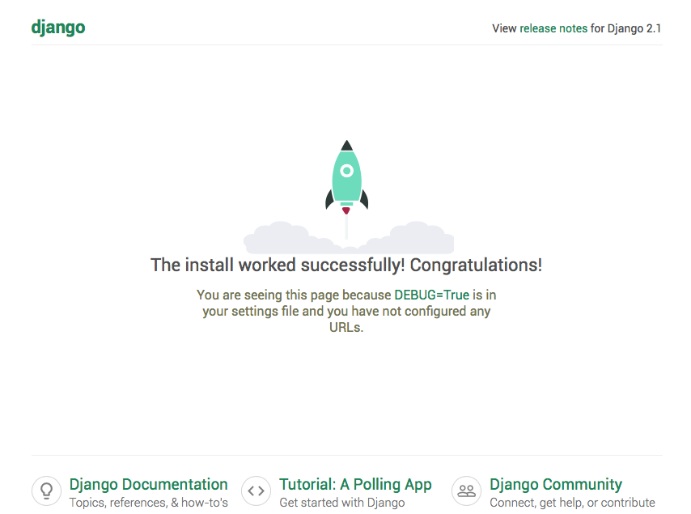
https://flask.palletsprojects.com/en/1.1.x/

FrameWork, Wikipedia 2021:

<https://es.wikipedia.org/w/index.php?title=Framework&oldid=135338706>

**ANEXOS:**

Servidor corriendo en Django:



Importaciones necesarias para el Proyecto:

