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 2021.

# IPC MUSIC WEB

**201903909 – Diego Alexander Acetún Chicol**

**Resumen**

Este reproductor de música ahora puede ser consumido desde una página web, el cual permite al usuario cargar sus propias listas de reproducción por medio de archivos csv o xml, si se carga un archivo csv la aplicación analiza que no contenga errores para posteriormente convertirlo a formato xml.

Este contenido es el que se envía posteriormente a la api para que la aplicación pueda crear las listas de reproducción.

También tiene un área de texto en la que el usuario puede modificar las etiquetas xml. El reproductor tiene las funciones de avanzar y retroceder canción, mostrar album nombre y artista de la canción que se está reproduciendo y también tiene un apartado de peticiones en donde se pueden observar gráficas de las siguientes opciones: Canciones más escuchadas, Artistas más reproducidos, Listas más escuchadas, Listas más populares.

**Palabras clave**

Api, Flask, Django, Objeto, Json

***Abstract***

*This music player can now be consumed from a web page, which allows the user to upload their own playlists through csv or xml files, if a csv file is uploaded the application analyzes it for errors and then converts it to xml format.*

*This content is the one that is later sent to the api so that the application can create the playlists.*

*It also has a text area where the user can modify the xml tags. The player has the functions of forward and rewind song, show album name and artist of the song that is being played and also has a section of requests where you can see graphs of the following options: Most Played Songs, Most Played Artists, Most Played Lists, Most Popular Lists.*

***Keywords***

*Api, Flask, Django, Object, Json*

## Introducción

El siguiente proyecto trata de hacer un reproductor de música como aplicación web, desarrollado utilizando los frameworks flask y django utilizando la versión de python 3.9. Django se utilizó para hacer la parte de frontend en la cual se utilizaron las plantillas de django, html, css y javascript.

Se utilizó la librería re para el análisis del archivo csv y la librería Element Tree para la lectura de los archivos xml.

Para la parte de backend se utilizó flask, en esta parte se hizo el apartado de peticiones, en el cuál se enviaban datos en formato json para el intercambio de datos, el backend devolvía los datos necesarios para poder realizar las gráficas

## Desarrollo del tema

¿Qué es una API?

El término API es una abreviatura de Application Programming Interfaces, 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.

¿Para qué sirve una API?

Una de las principales funciones de las API es poder facilitar el trabajo a los desarrolladores y ahorrarles tiempo y dinero. Por ejemplo, si se está creando una aplicación que es una tienda online, no se necesitará crear desde cero un sistema de pagos u otro para verificar si hay stock disponible de un producto. Se podrá utilizar la API de un servicio de pago ya existente, por ejemplo PayPal, y pedirle al distribuidor una API que permita saber el stock que ellos tienen.

### ¿Qué es un Framework?

Actualmente en el desarrollo moderno de aplicaciones web se utilizan distintos Frameworks que son herramientas que nos dan un esquema de trabajo y una serie de utilidades y funciones que nos facilita y nos abstrae de la construcción de páginas web dinámicas.

### 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.

**Flask**

Flask es un "micro" Framework: Para desarrollar una App básica o que se quiera desarrollar de una forma ágil y rápida Flask puede ser muy conveniente, para determinadas aplicaciones no se necesitan muchas extensiones y es suficiente.

Incluye un servidor web de desarrollo: No se necesita una infraestructura con un servidor web para probar las aplicaciones sino de una manera sencilla se puede correr un servidor web para ir viendo los resultados que se van obteniendo.

Tiene un depurador y soporte integrado para pruebas unitarias: Si tenemos algún error en el código que se está construyendo se puede depurar ese error y se puede ver los valores de las variables. Además está la posibilidad de integrar pruebas unitarias.

**Peticiones**

**Canciones más escuchadas:**

Se mandaba desde frontend el nombre de la canción y las reproducciones en formato json, en backend se ordenaban las reproducciones y se respondía con las 5 canciones con más reproducciones. En frontend se hacía la gráfica y se mandaba a llamar desde la plantilla de django.

**Artistas más reproducidos:**

Se mandaba desde frontend el nombre del artista y las reproducciones en formato json, en backend se ordenaban las reproducciones y se respondía con los 3 artistas con más reproducciones. En frontend se hacía la gráfica y se mandaba a llamar desde la plantilla de django.

**Listas más populares:**

Se mandaba desde frontend el nombre de la lista y las reproducciones en formato json, en backend se ordenaban las reproducciones y se respondía con las 10 listas con más reproducciones. En frontend se mostraban en una tabla las 10 listas.

**Métodos HTTP**

Entre los métodos http más utilizados se encuentran los siguientes:

**GET:**

El método GET solicita una representación de un recurso específico. Las peticiones que usan el método GET sólo deben recuperar datos.

**POST:**

El método POST se utiliza para enviar una entidad a un recurso en específico, causando a menudo un cambio en el estado o efectos secundarios en el servidor.

**PUT:**

El modo PUT reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición.

**DELETE:**

El método DELETE borra un recurso en específico.

En este proyecto se utilizaron principalmente los métodos GET y POST.

**Análisis CSV**

Para esto se utilizó la librería csv de python, la cual recorre línea por línea y con la librería re se establecieron las expresiones regulares y se analizaron los datos que venían en el CSV. Si el archivo contenía algún error se hacía saber al usuario para corregirlo, si no se guardaban los datos para posteriormente convertirlos a formato xml.

**Objetos**

Se utilizaron 3 tipos de objetos, Canciones, Artistas y Listas de reproducción, debido a que de estás 3 se hacían las peticiones.

Para crearlos se leía el archivo xml generado y se recorrían las etiquetas correspondientes para ir creando cada objeto.

**Conclusiones**

Hay varias ventajas al utilizar apis como lo pueden ser las siguientes:

Trabajar con apis facilita el manejo de datos ya que proporciona flexibilidad en la entrega de servicios e información.

Con una API se puede crear una capa de aplicación que se puede utilizar para la distribución de información y servicios a nuevas audiencias que se pueden personalizar para crear experiencias de usuario a la carta.

Cuando se proporciona acceso a una API, el contenido que se genera se puede publicar automáticamente y está disponible para todos los canales. Permite que se comparta y se distribuya más fácilmente.

**Referencias bibliográficas**

https://openwebinars.net/blog/que-es-flask/

https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Introduction

https://developer.mozilla.org/es/docs/Web/HTTP/Methods

https://www.xataka.com/basics/api-que-sirve

**Link del repositorio**

https://github.com/DiegoAcetun/IPC2_PROYECTO2DICIEMBRE_201903909.git