



Instituto Politécnico Nacional



**Unidad Profesional Interdisciplinaria de
Ingeniería y Tecnologías Avanzadas**

Programación Avanzada

Profesor: Miguel Félix

Practica 5

Equipo No.3

García Acosta Abraham

Repositorio

Ejercicio 1.

Se emula el resultado de las API's de Spotify y geonames

```
desarrollo ~/Documents/upiita/aplicaciones-distribuidas/practica5 [main] $ /usr/local/bin/python3.11 /Users/desarollo/Documents/  
Datos de mex: {'countryCode': 'MEX', 'countryName': 'Mexico', 'numPostalCodes': 7, 'minPostalCode': 1010, 'maxPostalCode': 2021  
Datos de col: {'countryCode': 'COL', 'countryName': 'Colombia', 'numPostalCodes': 9, 'minPostalCode': 1010, 'maxPostalCode': 10  
Datos de eua: {'countryCode': 'EUA', 'countryName': 'Estados unidos', 'numPostalCodes': 79, 'minPostalCode': 3010, 'maxPostalC  
desarollo ~/Documents/upiita/aplicaciones-distribuidas/practica5 [main] $ []
```

```
/practicasyWebServiceClienteClimaUPIITA.py  
● desarrollo ~/Documents/upiita/aplicaciones-distribuidas/practica5 [main] $ /usr/  
bin/python3.11 /Users/desarollo/Documents/upiita/aplicaciones-distribuidas/prac  
WebServiceClienteClimaUPIITA.py  
Datos de lista Mix_2023: 283.7850132098037 escuchantes  
Datos de lista Daily_music: 286.96243153503093 escuchantes  
Datos de lista Liked_songs: 254.46655315039453 escuchantes  
Datos de lista Daily_mix_1: 346.38470539984723 escuchantes  
Datos de lista Daily_mix_2: 2090.644913237683 escuchantes  
Datos de lista Pa_trapear: 125.50863351727729 escuchantes  
○ desarrollo ~/Documents/upiita/aplicaciones-distribuidas/practica5 [main] $
```

Agregue varias listas de reproducción y el dato que almacenan es la cantidad de oyentes por mes.

Para esta petición estoy definiendo que la ruta sea utilizando el path `"/songs/"`, después de ese fragmento, recupero el nombre de la lista de reproducción de la que quiero recuperar sus oyentes. Una vez recuperada la lista, convierto ese dato en la información en el campo `"listas"` y lo mando codificado al cliente.

```
import http.server
import socketserver
import json
import random

listas = {
    "Mix_2023": random.uniform(10, 300),
    "Daily_music": random.uniform(100, 300),
    "Liked_songs": random.uniform(20, 300),
    "Daily_mix_1": random.uniform(100, 800),
    "Daily_mix_2": random.uniform(100, 3000),
    "Pa_trapear": random.uniform(10, 300),
}

# Clase personalizada para manejar las solicitudes
class MyHandler(http.server.SimpleHTTPRequestHandler):
    def do_GET(self):
        if self.path.startswith('/songs/'):
            lista = self.path[7:]
            print(lista)
            if lista in listas:
                data = {"listas": listas[lista]}
                self.send_response(200)
                self.send_header('Content-type', 'application/json')
                self.end_headers()
                self.wfile.write(json.dumps(data).encode()) # Codificar la cadena a bytes
            else:
                self.send_response(404)
                self.end_headers()
                self.wfile.write("Lista no encontrada.".encode()) # Codificar la cadena a byte
        else:
            super().do_GET()

# Configuración del servidor
with socketserver.TCPServer(("", 9090), MyHandler) as httpd:
    print("Servidor web en el puerto 9090")
    httpd.serve_forever()
```

Un proceso similar realizo para obtener la informacion de geonames, creo información por país y a cada pais le asigno una key para recuperar toda su información.

```
import http.server
import socketserver
import json

países = {
    "mex": {
        "countryCode": "MEX",
        "countryName": "Mexico",
        "numPostalCodes": 7,
        "minPostalCode": 1010,
        "maxPostalCode": 2020
    },
    "eua": {
        "countryCode": "EUA",
        "countryName": "Estados unidos",
        "numPostalCodes": 79,
        "minPostalCode": 3010,
        "maxPostalCode": 10020
    },
    "col": {
        "countryCode": "COL",
        "countryName": "Colombia",
        "numPostalCodes": 9,
        "minPostalCode": 1010,
        "maxPostalCode": 10020
    }
}

# Clase personalizada para manejar las solicitudes
class MyHandler(http.server.SimpleHTTPRequestHandler):
    def do_GET(self):
        if self.path.startswith('/countries/'):
            país = self.path[11:]
            print(país)
            if país in países:
                data = {"countries": países[país]}
                self.send_response(200)
                self.send_header('Content-type', 'application/json')
                self.end_headers()
                self.wfile.write(json.dumps(data).encode()) # Codificar la cadena a bytes
            else:
                self.send_response(404)
                self.end_headers()
                self.wfile.write("País no encontrado.".encode()) # Codificar la cadena a bytes
        else:
            super().do_GET()

# Configuración del servidor
with socketserver.TCPServer(("", 9090), MyHandler) as httpd:
    print("Servidor web en el puerto 9090")
    httpd.serve_forever()
```

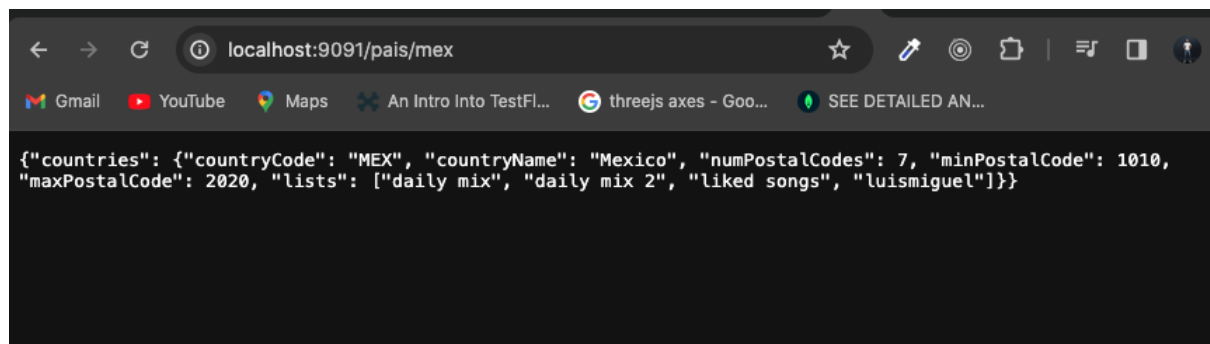
Ejercicio 2.

Obtener datos de cada pais junto con listas de reproducción

```
establish a new connection: [errno 61] Connection refused })
desarollo ~/Documents/upiita/aplicaciones-distribuidas/practica5 [main] $ /usr/local/
bin/python3.11 /Users/desarollo/Documents/upiita/aplicaciones-distribuidas/practica5/
WebserviceClienteClimaUPIITA.py
Datos de mex: {'countryCode': 'MEX', 'countryName': 'Mexico', 'numPostalCodes': 7, 'minPostalCode': 1010, 'maxPosta
lCode': 2020, 'lists': ['daily mix', 'daily mix 2', 'liked songs', 'luismiguel']}
Datos de col: {'countryCode': 'COL', 'countryName': 'Colombia', 'numPostalCodes': 9, 'minPostalCode': 1010, 'maxPos
talCode': 10020, 'lists': ['daily mix', 'daily mix 2', 'perreke', 'luismiguel']}
Datos de eua: {'countryCode': 'EUA', 'countryName': 'Estados unidos', 'numPostalCodes': 79, 'minPostalCode': 3010,
'maxPostalCode': 10020, 'lists': ['daily mix', 'daily mix 2', 'liked songs', 'juan gabriel']}
desarollo ~/
desarollo ~/Documents/upiita/aplicaciones-distribuidas/practica5 [main] $
```

Ejercicio 3.

Version navegador



```
localhost:9091/pais/mex
Gmail YouTube Maps An Intro Into TestFl... threejs axes - Goo... SEE DETAILED AN...
{"countries": {"countryCode": "MEX", "countryName": "Mexico", "numPostalCodes": 7, "minPostalCode": 1010,
"maxPostalCode": 2020, "lists": ["daily mix", "daily mix 2", "liked songs", "luismiguel"]}}
```

Conclusión

El código anterior emula peticiones https a un servidor utilizando un servidor local el cual recupera información dependiendo de un query o parametro que el cliente manda lo que en el ambito laboral puede ser un id o una key con la cual se puede recuperar información valiosa de una api.