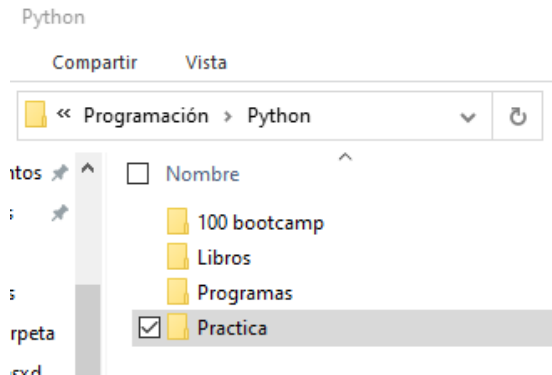
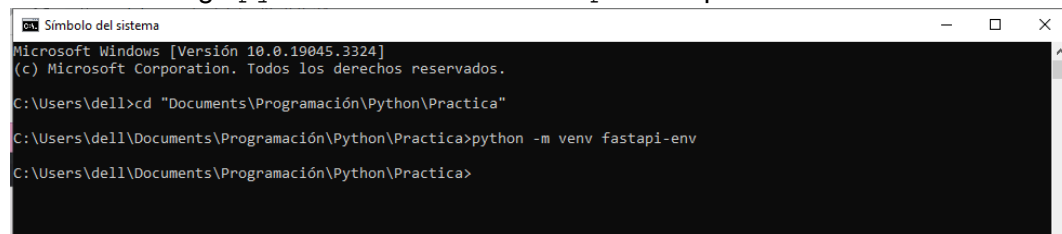


DOCUMENTACIÓN DE LA PRÁCTICA

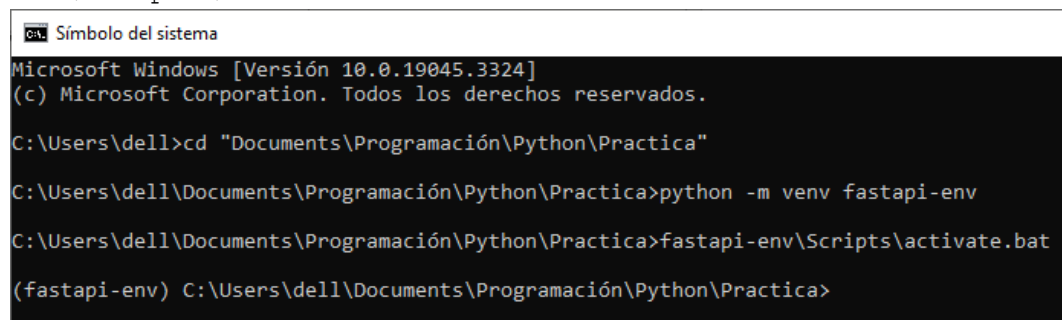
1. Como primer paso se eligió el lenguaje Python, por su versatilidad y una sintaxis sencilla.
2. Se crea una nueva carpeta llamada "Practica", esto para poder crear un entorno virtual.



3. Se entra a símbolo del sistema, y se dirige a la carpeta de practicas, y ahí dentro se escribe el código `python -m venv fastapi-env` para crear el entorno virtual.



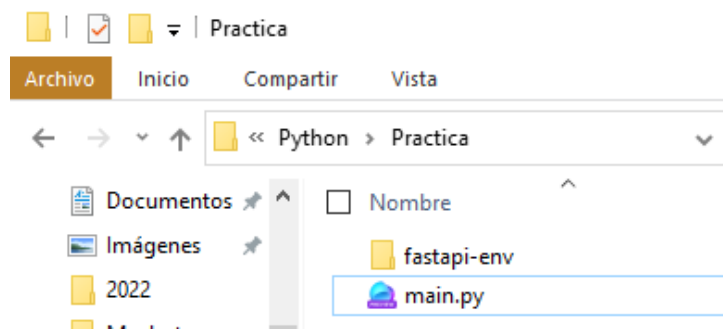
4. Posteriormente se activa el entorno con el código `fastapi-env\Scripts\activate.bat`.



5. Posteriormente se instala FastAPI con el código `pip install fastapi`.

```
Símbolo del sistema
(fastapi-env) C:\Users\dell\Documents\Programación\Python\Practica>pip install fastapi
Collecting fastapi
  Using cached https://files.pythonhosted.org/packages/09/ae/8378894f9fbd0297cdfdc79496ccd779166d675fec47cad8d2ca78273
9/fastapi-0.101.1-py3-none-any.whl
Collecting pydantic!=1.8,!1.8.1,!2.0.0,!2.0.1,!2.1.0,<3.0.0,>=1.7.4 (from fastapi)
  Using cached https://files.pythonhosted.org/packages/fd/35/86b1e7571e695587df0ddcf2937100436dce0caa277d2f016d4e4f7d3791
a/pydantic-2.2.1-py3-none-any.whl
Collecting typing-extensions>=4.5.0 (from fastapi)
  Using cached https://files.pythonhosted.org/packages/ec/6b/63cc3df74987c36fe26157ee12e09e8f9db4de771e0f3404263117e75b9
5/typing_extensions-4.7.1-py3-none-any.whl
Collecting starlette<0.28.0,>=0.27.0 (from fastapi)
  Using cached https://files.pythonhosted.org/packages/58/f8/e2cca22387965584a409795913b774235752be4176d276714e15e1a5888
4/starlette-0.27.0-py3-none-any.whl
Collecting pydantic-core==2.6.1 (from pydantic!=1.8,!1.8.1,!2.0.0,!2.0.1,!2.1.0,<3.0.0,>=1.7.4->fastapi)
  Using cached https://files.pythonhosted.org/packages/f7/18/b1c1909941731f042b1e486f4cb9f280234824d6072198fe63286fa7e9c
9/pydantic_core-2.6.1-cp37-none-win_amd64.whl
Collecting annotated-types>=0.4.0 (from pydantic!=1.8,!1.8.1,!2.0.0,!2.0.1,!2.1.0,<3.0.0,>=1.7.4->fastapi)
  Using cached https://files.pythonhosted.org/packages/d8/f0/a2ee543a96cc624c35a9086f39b1ed2aa403c6d355dfe47a11ee5c64a16
4/annotated_types-0.5.0-py3-none-any.whl
Collecting anyio<5,>=3.4.0 (from starlette<0.28.0,>=0.27.0->fastapi)
  Using cached https://files.pythonhosted.org/packages/19/24/44299477fe7dcc9cb58d0a57d5a7588d6af2ff403fdd2d47a246c91a324
6/anyio-3.7.1-py3-none-any.whl
Collecting sniffio>=1.1 (from anyio<5,>=3.4.0->starlette<0.28.0,>=0.27.0->fastapi)
  Using cached https://files.pythonhosted.org/packages/c3/a0/5dba8ed157b0136607c7f2151db695885606968d1fae123dc3391ecfdb
f/sniffio-1.3.0-py3-none-any.whl
Collecting exceptiongroup; python_version < "3.11" (from anyio<5,>=3.4.0->starlette<0.28.0,>=0.27.0->fastapi)
  Using cached https://files.pythonhosted.org/packages/ad/83/b71e58666f156a39fb29417e4c8ca4bc7400c0dd4ed9e8842ab54dc8c34
4/exceptiongroup-1.1.3-py3-none-any.whl
Collecting idna>=2.8 (from anyio<5,>=3.4.0->starlette<0.28.0,>=0.27.0->fastapi)
```

6. Se crea el archivo main.py en el mismo nivel a la carpeta fastapi-env.



7. Dentro del archivo main.py, con el editor Visual Studio Code, se escribirán las líneas de código siguientes

```
from fastapi import FastAPI
app = FastAPI()
```

Que significan la importación de FastAPI del módulo fastapi, y luego se instanciará FastAPI en app.

```
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5 |
```

8. Para `@app.get("/")`, ósea la función base con entrada "/", se programa una salida el diccionario con el mensaje de salida: "Bienvenidos al reventón musical, elije las canciones que quieras escuchar".

```
@app.get("/")
def index():
    return {"message": "Bienvenidos al reventón musical, elije las canciones que quieras escuchar."}
```

9. En símbolo de sistema, se ejecuta el siguiente comando `pip install uvicorn`

```

Simbolo del sistema
"uvicorn" no se reconoce como un comando interno o externo,
programa o archivo por lotes ejecutable.

(fastapi-env) C:\Users\dell\Documents\Programación\Python\Practica>pip install uvicorn
Collecting uvicorn
  Using cached https://files.pythonhosted.org/packages/ad/bd/d47ee02312640fcf26c7e1c807402d5c5eab468571153a94ec8f7ada0e4
6/uvicorn-0.22.0-py3-none-any.whl
Collecting h11>=0.8 (from uvicorn)
  Using cached https://files.pythonhosted.org/packages/95/04/ff642e65ad6b90db43e668d70ffb6736436c7ce41fcc549f4e947223412
7/h11-0.14.0-py3-none-any.whl
Requirement already satisfied: typing-extensions; python_version < "3.8" in c:\users\dell\documents\programación\python\
practica\fastapi-env\lib\site-packages (from uvicorn) (4.7.1)
Collecting click>=7.0 (from uvicorn)
  Using cached https://files.pythonhosted.org/packages/00/2e/d53fa4befbf2cfa713304affc7ca780ce4fc1fd8710527771b58311a322
9/click-8.1.7-py3-none-any.whl
Collecting colorama; platform_system == "Windows" (from click>=7.0->uvicorn)
  Using cached https://files.pythonhosted.org/packages/d1/d6/3965ed04c63042e047cb6a3e6ed1a63a35087b6a609aa3a15ed8ac56c22
1/colorama-0.4.6-py2.py3-none-any.whl
Collecting importlib-metadata; python_version < "3.8" (from click>=7.0->uvicorn)
  Using cached https://files.pythonhosted.org/packages/ff/94/64287b38c7de4c90683630338cf28f129decbbba0a44f0c6db35a873c73c
4/importlib_metadata-6.7.0-py3-none-any.whl
Collecting zipp>=0.5 (from importlib-metadata; python_version < "3.8"->click>=7.0->uvicorn)
  Using cached https://files.pythonhosted.org/packages/5b/fa/c9e82bbe1af6266adf08afb563905eb87cab83fde00a0a0896351062104
7/zipp-3.15.0-py3-none-any.whl
Installing collected packages: h11, colorama, zipp, importlib-metadata, click, uvicorn
Successfully installed click-8.1.7 colorama-0.4.6 h11-0.14.0 importlib-metadata-6.7.0 uvicorn-0.22.0 zipp-3.15.0
You are using pip version 19.0.3, however version 23.2.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

```

10. Después se ejecuta el comando `uvicorn main:app --reload` y así se lanza la API en el directorio <http://127.0.0.1:8000>, ósea el puerto 8000.

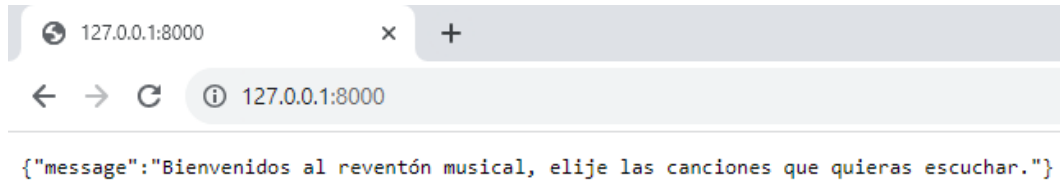
```

Simbolo del sistema - uvicorn main:app --reload
  Using cached https://files.pythonhosted.org/packages/95/04/ff642e65ad6b90db43e668d70ffb6736436c7ce41fcc549f4e947223412
7/h11-0.14.0-py3-none-any.whl
Requirement already satisfied: typing-extensions; python_version < "3.8" in c:\users\dell\documents\programación\python\
practica\fastapi-env\lib\site-packages (from uvicorn) (4.7.1)
Collecting click>=7.0 (from uvicorn)
  Using cached https://files.pythonhosted.org/packages/00/2e/d53fa4befbf2cfa713304affc7ca780ce4fc1fd8710527771b58311a322
9/click-8.1.7-py3-none-any.whl
Collecting colorama; platform_system == "Windows" (from click>=7.0->uvicorn)
  Using cached https://files.pythonhosted.org/packages/d1/d6/3965ed04c63042e047cb6a3e6ed1a63a35087b6a609aa3a15ed8ac56c22
1/colorama-0.4.6-py2.py3-none-any.whl
Collecting importlib-metadata; python_version < "3.8" (from click>=7.0->uvicorn)
  Using cached https://files.pythonhosted.org/packages/ff/94/64287b38c7de4c90683630338cf28f129decbbba0a44f0c6db35a873c73c
4/importlib_metadata-6.7.0-py3-none-any.whl
Collecting zipp>=0.5 (from importlib-metadata; python_version < "3.8"->click>=7.0->uvicorn)
  Using cached https://files.pythonhosted.org/packages/5b/fa/c9e82bbe1af6266adf08afb563905eb87cab83fde00a0a0896351062104
7/zipp-3.15.0-py3-none-any.whl
Installing collected packages: h11, colorama, zipp, importlib-metadata, click, uvicorn
Successfully installed click-8.1.7 colorama-0.4.6 h11-0.14.0 importlib-metadata-6.7.0 uvicorn-0.22.0 zipp-3.15.0
You are using pip version 19.0.3, however version 23.2.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

(fastapi-env) C:\Users\dell\Documents\Programación\Python\Practica>uvicorn main:app --reload
+-----+
| 32mINFO+0m:      Will watch for changes in these directories: ['C:\Users\dell\Documents\Programación\Python\Prac|
|               |tica']|
+-----+
| 32mINFO+0m:      Uvicorn running on +[1mhttp://127.0.0.1:8000+0m (Press CTRL+C to quit)|
+-----+
| 32mINFO+0m:      Started reloader process [+0m] using [+0m]|
+-----+
| 32mINFO+0m:      Started server process [+0m]|
+-----+
| 32mINFO+0m:      Waiting for application startup.|
+-----+
| 32mINFO+0m:      Application startup complete.|
+-----+

```

11. Se abre el navegador de preferencia (en este caso Google chrome) y en la barra de búsqueda se coloca el directorio ya desplegado. Como resultado, la API lanza como salida anteriormente descrita en el paso 8.



12. En este caso se realizará una API en base a un setlist de diversas canciones. Para eso primero se definirá un arreglo de diccionarios.

```
canciones = [
    {
        "id": 1,
        "name": "Remote Control",
        "artista": "Sussie 4 y Leon Larregui"
    },
    {
        "id": 2,
        "name": "Bohemian Rhapsody",
        "artista": "Queen"
    }
]
```

13. Definimos algunas funciones, con sus entradas y salidas. Una para la consulta de la lista de canciones, otra para consultar de acuerdo al id y la última para consulta con id y artista al mismo tiempo.

```
@app.get("/canciones")
def get_canciones():
    return canciones

@app.get("/canciones/{id}")
def get_cancion(id: int):
    return list(filter(lambda item: item['id'] == id, canciones))

@app.get("/canciones/")
def get_cancion(id: int, artista: str):
    return list(filter(lambda item: item['id'] == id, canciones))
```

Para desplegar cada una con el cliente REST (Google Chrome), debe tenerse actualizado `uvicorn main:app --reload` en símbolo de sistema, en chrome se busca <http://127.0.0.1:8000/docs> y de ahí se despliegan las funciones respectivas. Por cada una dar clic en la flecha abajo, luego en “try it out” y al final “execute” (Nota: este procedimiento es válido para las siguientes funciones. Y también, esta es una manera para documentar la API, y ejecutarla por su facilidad visual y lógica).

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:5000/canciones' \
  -H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:5000/canciones
```

Server response

Code

Details

200

Response body

```
[
  {
    "id": 1,
    "name": "Remote Control",
    "artista": "Sussie 4 y Leon Larregui"
  },
  {
    "id": 2,
    "name": "Supremassive black hole",
    "artista": "MUSE"
  }
]
```

Response headers

```
content-length: 130
content-type: application/json
date: Wed, 23 Aug 2023 05:37:38 GMT
server: uvicorn
```

get canciones()

```
id * required
integer
(path)
```

Execute

```
get Cancion(id: int)
```

artista * required
string
(query)

MUSE

Execute

Responses

Curl

```
curl -X 'GET' \
'http://127.0.0.1:5000/canciones/?id=2&artista=MUSE' \
-H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:5000/canciones/?id=2&artista=MUSE
```

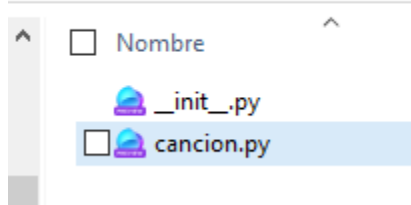
Server response

Code	Details
200	<p>Response body</p> <pre>[{ "id": 2, "name": "Supremassive black hole", "artista": "MUSE" }]</pre>

get_cancion(id: int, artista: str)

14. Ahora se van a crear unos esquemas, para ello se creará una nueva carpeta con el nombre de “models” (al nivel de main.py), y dentro de esa carpeta un archivo llamado `__init__.py` y también el archivo del modelo llamado `cancion.py`.

Python > Practica > models



15. Se edita el archivo `cancion.py` importando `BaseModel` de `Pydantic` y se declara la clase `Cancion` que hereda de `BaseModel`.

```
from pydantic import BaseModel

class Cancion(BaseModel):
    id: int
    name: str
    artista: str
```

16. Regresando al `main.py` se importa la clase `Cancion` de `models.cancion`.

```
from fastapi import FastAPI
from models.cancion import Cancion
```

17. En `main.py`, se crea una función para añadir canciones que es de tipo `post`.

```
@app.post("/canciones")
def create_cancion(cancion: Cancion):
    canciones.append(cancion)
    return canciones
```

Función desplegada.

Parameters

No parameters

Request body required

```
{
  "id": 3,
  "name": "Bohemian Rhapsody",
  "artista": "Queen"
}
```


Responses

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:5000/canciones' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "id": 3,
    "name": "Bohemian Rhapsody",
    "artista": "Queen"
  }'
```

Request URL

```
http://127.0.0.1:5000/canciones
```

Server response

Code

Details

200

Response body

```
[
  {
    "id": 1,
    "name": "Remote Control",
    "artista": "Sussie 4 y Leon Larregui"
  },
  {
    "id": 2,
    "name": "Supremassive black hole",
    "artista": "MUSE"
  },
  {
    "id": 3,
    "name": "Bohemian Rhapsody",
    "artista": "Queen"
  }
]
```

18. Ahora se define una función para editar canciones, que es de tipo put.

```
@app.put("/canciones/{id}")
def update_cancion(id: int, cancion: Cancion):
    for index, item in enumerate(canciones):
        if item['id'] == id:
            canciones[index]['name'] = cancion.name
            canciones[index]['artista'] = cancion.artista
    return canciones
```

Función desplegada

Parameters

Name	Description
id * required	
integer	2
(path)	

Request body required

```
{  
  "name": "Starlight",  
  "artista": "MUSE"  
}
```

Responses

Curl

```
curl -X 'PUT' \
  'http://127.0.0.1:5000/canciones/2' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "name": "Starlight",
    "artista": "MUSE"
  }'
```

Request URL

```
http://127.0.0.1:5000/canciones/2
```

Server response

Code

Details

200

Response body

```
[
  {
    "id": 1,
    "name": "Remote Control",
    "artista": "Sussie 4 y Leon Larregui"
  },
  {
    "id": 2,
    "name": "Starlight",
    "artista": "MUSE"
  }
]
```

19. Y también se define la función para eliminar elementos de la lista que es de tipo delete.

```
@app.delete("/canciones/{id}")
def delete_cancion(id: int):
    for item in canciones:
        if item['id'] == id:
            canciones.remove(item)
    return canciones
```

Función desplegada

DELETE

/canciones/{id} Delete Cancion

Parameters

Name	Description
id * required integer (path)	<input type="text" value="2"/>

Execute

Responses

Curl

```
curl -X 'DELETE' \
'http://127.0.0.1:5000/canciones/2' \
-H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:5000/canciones/2
```

Server response

Code	Details
200	<div>Response body<pre>[{ "id": 1, "name": "Remote Control", "artista": "Sussie 4 y Leon Larregui" }]</pre></div> <div>Response headers<pre>content-length: 71 content-type: application/json date: Wed, 23 Aug 2023 06:12:24 GMT server: uvicorn</pre></div>

20. Para hacerlo más práctico y en caso de que el usuario no sepa el id para poder editar, agregar o eliminar, solo se edita en `cancion.py` importando `Optional` de `typing` y dentro de la clase `canción` se asignaría a ese atributo como un entero opcional, y en caso de que no se asigne optaría por ser nulo.

```
from pydantic import BaseModel
from typing import Optional

class Cancion(BaseModel):
    id: Optional[int] = None
    name: str
    artista: str
```

21. Para realizar las validaciones, ya sea poner ciertas restricciones a los atributos de `Cancion`, se añade en la importación "`Field`" y después se editan los atributos

```
from pydantic import BaseModel, Field
from typing import Optional

class Cancion(BaseModel):
    id: Optional[int] = None
    name: str = Field(default="Nueva canción", min_length=2, max_length=20)
    artista: str = Field(default="Nuevo artista", min_length=2, max_length=20)
```

Para comprobarlo, se hará con la función `create_cancion(cancion: Cancion)`.

POST `/canciones` Create Cancion

Parameters

No parameters

Request body required

```
{
  "id": 3,
  "name": "Beat It",
  "artista": "Michael Jackson"
}
```

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:5000/canciones' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "id": 3,
    "name": "Beat It",
    "artista": "Michael Jackson"
  }'
```

Request URL

```
http://127.0.0.1:5000/canciones
```

Server response

Code	Details
200	<p>Response body</p> <pre>[{ "id": 1, "name": "Remote Control", "artista": "Sussie 4 y Leon Larregui" }, { "id": 2, "name": "Supremassive black hole", "artista": "MUSE" }, { "id": 2, "name": "Beat It", "artista": "Michael Jackson" }, { "id": 3, "name": "Beat It", "artista": "Michael Jackson" }]</pre>

22. En main.py se agregan a la importación Query y Path de fastapi.

```
from fastapi import FastAPI, Query, Path
from models.cancion import Cancion

app = FastAPI()
```

Para ello modificamos la función get_cancion(id: int = Path(gt=0))

```
@app.get("/canciones/{id}")
def get_cancion(id: int = Path(gt=0)):
    return list(filter(lambda item: item['id'] == id, canciones))
```

Y después lo comprobamos con la función ya desplegada

GET	/canciones/{id}	Get Cancion
Parameters		
Name	Description	
id * required		
integer		0
(path)		

Responses

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:5000/canciones/0' \
  -H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:5000/canciones/0
```

Server response

Code	Details
------	---------

422	Error: Unprocessable Entity
-----	-----------------------------

Response body

```
{
  "detail": [
    {
      "type": "greater_than",
      "loc": [
        "path",
        "id"
      ],
      "msg": "Input should be greater than 0",
      "input": "0",
      "ctx": {
        "gt": 0
      },
      "url": "https://errors.pydantic.dev/2.2/v/greater_than"
    }
  ]
}
```

23. Para poder consumir una API externa, solo basta con importar httpx y después escribir la función de la siguiente imagen.

```
@app.get("/consumir-api-externa")
async def consumir_api_externa():
    url = "https://pokeapi.co/api/v2/pokemon/ditto"
    async with httpx.AsyncClient() as client:
        response = await client.get(url)
        return response.json()
```


Curl

```
curl -X 'GET' \
'http://127.0.0.1:5002/consumir-api-externa' \
-H 'accept: application/json'
```

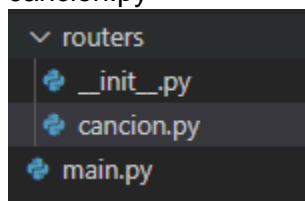
Request URL

```
http://127.0.0.1:5002/consumir-api-externa
```

Server response

Code	Details
200	<p>Response body</p> <pre>{ "abilities": [{ "ability": { "name": "limber", "url": "https://pokeapi.co/api/v2/ability/7/" }, "is_hidden": false, "slot": 1 }, { "ability": { "name": "imposter", "url": "https://pokeapi.co/api/v2/ability/150/" }, "is_hidden": true, "slot": 3 }], "base_experience": 101, "forms": [{ "name": "ditto", "url": "https://pokeapi.co/api/v2/pokemon-form/132/" }], "game_indices": [{</pre>

24. Con fines de optimización, se crea la carpeta routers y los archivos `__init__.py` y `cancion.py`



25. Dentro de `cancion.py` se importa `APIRouter`, y se instancia la clase `APIRouter` en `router`.

```
from fastapi import APIRouter

router = APIRouter()
```

26. Del archivo de main.py se cortan las rutas que tengan que ver con canciones, se reemplazan los app de las rutas con router, se corta el arreglo de canciones y se importan las clases necesarias como Path y el modelo de cancion.

```
from fastapi import Query, Path, APIRouter
from models.cancion import Cancion

router = APIRouter()

canciones = [
    {
        "id": 1,
        "name": "Remote Control",
        "artista": "Sussie 4 y Leon Larregui"
    },
    {
        "id": 2,
        "name": "Supremassive black hole",
        "artista": "MUSE"
    }
]

@router.get("/canciones")
def get_canciones():
    return canciones
#Nota
@router.get("/canciones/{id}")
def get_cancion(id: int = Path(gt=0)):
    return list(filter(lambda item: item['id'] == id, canciones))

@router.get("/canciones/")
def get_cancion(id: int, artista: str):
    return list(filter(lambda item: item['id'] == id and item['artista'] == artista, canciones))
```

27. Y ahora dentro de main.py se importa router renombrándola como song_router y se utiliza dentro de app.include_router(), esto para que se vea reflejado el cambio en la API.

```
from routers.cancion import router as song_router
import httpx

app = FastAPI()
app.include_router(song_router)
```

28. Si se desea correr la API desde el puerto 8090, solo basta con cerrar la actual ejecución, volver a realizar los pasos 3, 4 y 10, pero con la diferencia de que no se ejecuta el comando para instalar el entorno virtual y en la ejecución de reload es `uvicorn main:app --reload --port 8090`.

```

Microsoft Windows [Versión 10.0.19045.3324]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\dell>cd "Documents\Programación\Python\Practica"

C:\Users\dell\Documents\Programación\Python\Practica>fastapi-env\Scripts\activate.bat

(fastapi-env) C:\Users\dell\Documents\Programación\Python\Practica>uvicorn main:app --reload --port 8090
+ [32mINFO+ [0m: Will watch for changes in these directories: ['C:\\Users\\dell\\Documents\\Programación\\Python\\Prac
tica']
+ [32mINFO+ [0m: Uvicorn running on +[1mhttp://127.0.0.1:8090+ [0m (Press CTRL+C to quit)
+ [32mINFO+ [0m: Started reloader process [+ [36m+ [1m14428+ [0m] using +[36m+ [1mStatReload+ [0m]
+ [32mINFO+ [0m: Started server process [+ [36m4264+ [0m]
+ [32mINFO+ [0m: Waiting for application startup.
+ [32mINFO+ [0m: Application startup complete.

```

← → ↻ ⓘ 127.0.0.1:8090/docs#/

FastAPI 0.1.0 OAS 3.1

/openapi.json

default

GET	/	Index
GET	/canciones	Get Canciones
POST	/canciones	Create Cancion
GET	/canciones/{id}	Get Cancion
PUT	/canciones/{id}	Update Cancion
DELETE	/canciones/{id}	Delete Cancion
GET	/canciones/	Get Cancion
GET	/consumir-api-externa	Consumir Api Externa

Realizado por Joel Gómez - gjoel183@outlook.com