



Redes de Datos

Comunicación API Cliente-Servidor

Trabajo Práctico Integrador

Sancho Almenar, Mariano S-5778/9

26/07 2024.

Universidad Nacional de Rosario.

Facultad de Ciencias Exactas, Ingeniería y Agrimensura.

Carrera: Tecnicatura Universitaria en Inteligencia Artificial.

Introducción

El trabajo consiste en la comunicación entre un cliente y un servidor por medio de una API. Elegí la base de datos 'Los mejores 100 libros', en la que se podrá acceder a la misma y modificarla agregando o quitando elementos.

Se trabajó sobre el sistema operativo Windows usando un entorno virtual. El entorno de trabajo está dividido en dos carpetas principales, una para el servidor y otra para el cliente. En cada carpeta se encuentran los archivos necesarios para el funcionamiento del programa. Además, cuenta con una carpeta adicional 'data' para ubicar allí la base de datos.

Entorno del Servidor

El servidor está configurado sobre el sistema operativo Windows. Se utiliza Visual Studio Code como el entorno de desarrollo para la programación en Python.

Se creó un entorno virtual venv para gestionar las dependencias del proyecto, las cuales son:

- FastAPI
- Uvicorn
- Requests
- Pydantic

El servidor cuenta con un archivo principal server.py donde se encuentran las funciones principales para la creación y manipulación de la API y un archivo secundario auxiliar.py con la base del modelo Libro y funciones secundarias como la descarga de la base de datos.

El principal problema que tuve fue no saber cómo descargar desde mi programa la carpeta 'images' de la base de datos, esta carpeta contiene las portadas de los libros que se encuentran en la base de datos. Mi solución, aunque no fuese óptima, fue incluir de entrada dicha carpeta, sin la necesidad de descargarla.

Entorno del cliente

La creación del entorno del cliente fue similar a la del servidor, con la diferencia en que solo se instaló la librería Request.

El entorno cuenta con varios archivos:

- cliente.py: Archivo principal donde se encuentra la estructura del cliente.
- menu.py: Archivo secundario para las funciones del menú.
- auxiliar_client.py: Archivo donde se encuentran las funciones que se comunican con el servidor

Base de datos elegida

Elegí la base 'Los 100 mejores libros' ya que es un tema que me resulta interesante y además me da la posibilidad de manipular tanto archivos de texto como de imágenes.

Cuenta con un archivo `books.json` donde se encuentran los 100 libros, donde cada libro cuenta con:

- Autor -> Tipo String
- País de origen -> Tipo String
- Imagen -> Tipo String (nos indica la ubicación de la misma)
- Lenguaje -> Tipo String
- Link de referencia a Wikipedia -> Tipo String
- Cantidad de páginas -> Tipo Entero
- Título -> Tipo String
- Año de publicación -> Tipo Entero

Además, se encuentra la carpeta 'images' con archivos .jpg donde se ubican las portadas de todos los libros.

Procedimientos

Para iniciar el servidor, será necesario primero crear y activar el entorno virtual en la carpeta descargada

```
python -m venv .venv
```

```
cd .\venv\Scripts
```

```
.\activate
```

Una vez activado, instalar las dependencias necesarias usando el comando

```
pip install -r requirements.txt
```

Ya se podría iniciar el servidor con el comando:

```
python .\src\server\server.py
```

De esta forma, estaríamos abriendo el servidor en la dirección <http://127.0.0.1:8000>.

Si se quiere acceder al servidor desde otra terminal, se deberá cambiar el valor de HOST en `server.py` por la IP del servidor.

El servidor contará con las siguientes funciones encargadas de la obtención y manipulación de datos:

- 1) `get_all_books()`: Devuelve todos los libros pertenecientes a la base de datos.
- 2) `get_book_by(**kargs)`: Devuelve los libros que cumplan con los parámetros
- 3) `post_book(book : BookSchema)`: Agrega un nuevo libro a la base de datos
- 4) `delete_book(author, tittle)`: Elimina un libro a partir de su autor y su título
- 5) `get_image(imageLink)`: Busca una portada a partir de su ubicación
- 6) `upload_image(image)`: Agrega una nueva portada a la base de datos
- 7) `delete_image(imageLink)`: Elimina una portada a partir de su ubicación.

Ejemplos de uso en el SwaggerUI:

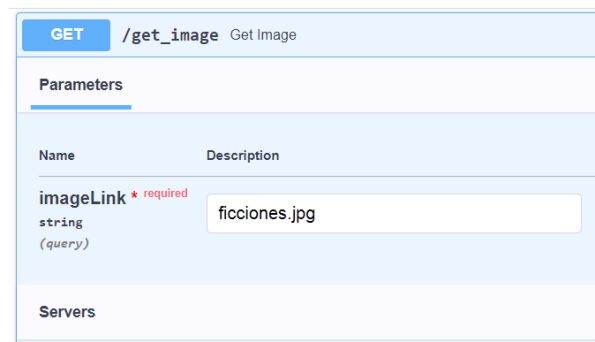
- 1) Buscar un libro en la base de datos, en este caso, alguno escrito por Jorge Luis Borges

Name	Description
author string (query)	Jorge Luis Borges
country string (query)	country
language string (query)	language
title string (query)	title
year integer (query)	year

Response body

```
[
  {
    "author": "Jorge Luis Borges",
    "country": "Argentina",
    "imageLink": "images/ficciones.jpg",
    "language": "Spanish",
    "link": "https://en.wikipedia.org/wiki/Ficciones\\n",
    "pages": 224,
    "title": "Ficciones",
    "year": 1965
  }
]
```

2) Buscar la portada de Ficciones:

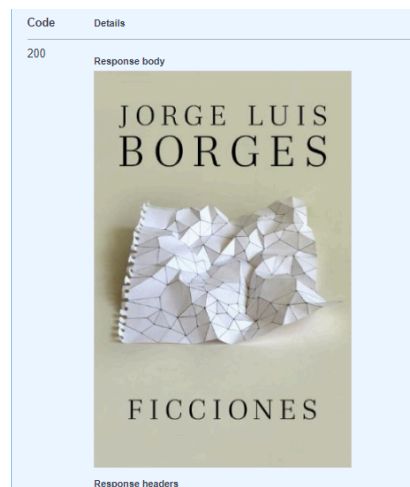


GET /get_image Get Image

Parameters

Name	Description
imageLink ★ required	
string (query)	<input type="text" value="ficciones.jpg"/>

Servers



Procedimiento cliente

El proceso de inicio del cliente es similar al del servidor. En una terminal, abrimos la carpeta descargada y creamos el entorno virtual.

```
python -m venv .venv
```

```
cd .\venv\Scripts
```

```
.\activate
```

Una vez activado, instalar las dependencias necesarias usando el comando

```
pip install -r requirements.txt
```

Ya se podría iniciar el cliente con el comando:

```
python .\src\client\client.py
```

Al igual que en el caso anterior, si se quiere acceder a un servidor ubicado en otra terminal, se deberá cambiar el valor de HOST en auxiliar.py por la IP del servidor.