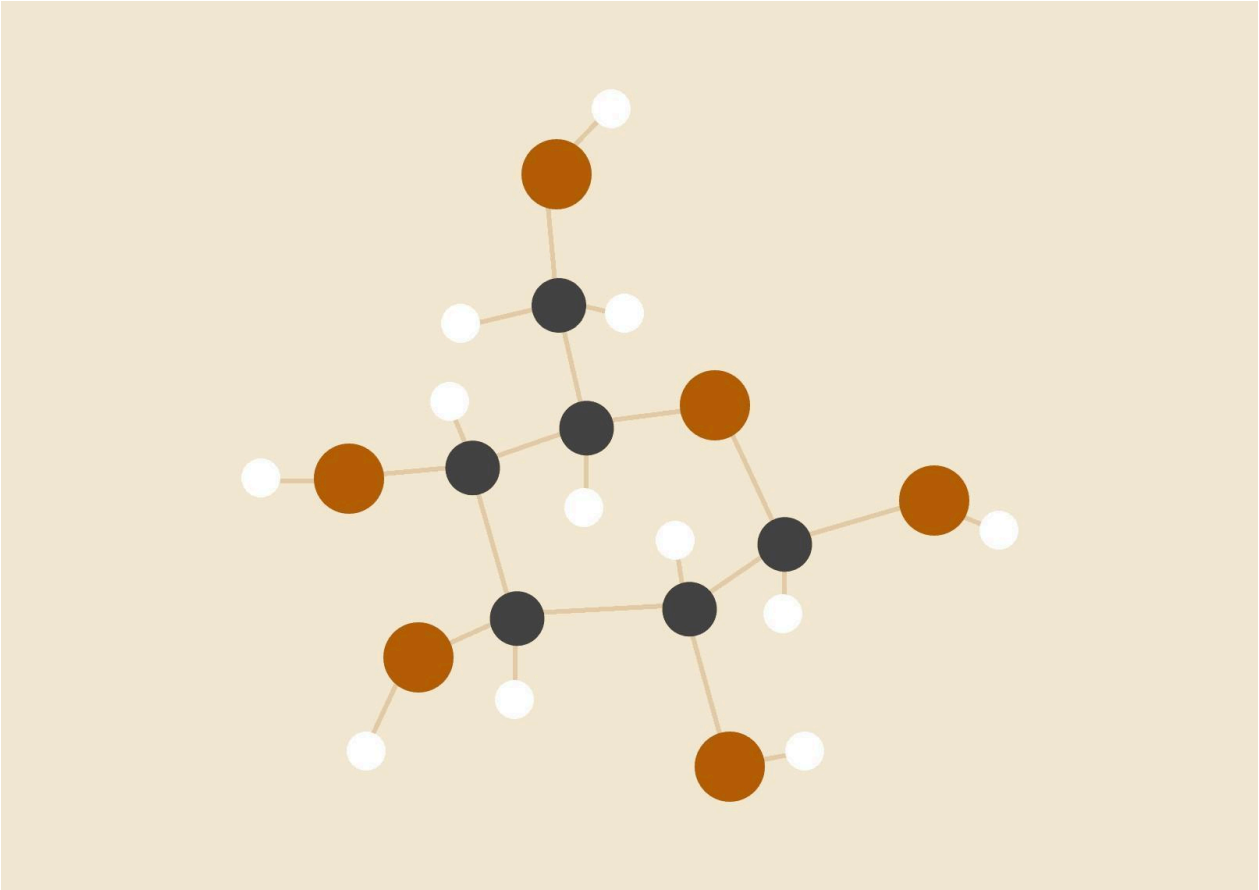
TRABAJO PRÁCTICO INTEGRADOR

IA3.5 Redes de Datos

Tecnicatura Universitaria en Inteligencia Artificial



Martinez Dufour, Caterina
Porcelli, Fabricio

30/11/2024

Universidad Nacional de Rosario

Facultad de Ciencias Exactas, Ingeniería y Agrimensura

INTRODUCCIÓN

Este trabajo consiste en desarrollar una API Cliente - Servidor. El servidor debe ser capaz de almacenar información en un archivo con formato JSON, gestionar consultas y modificaciones sobre este. Sin embargo, el cliente debe ser el que realiza esas consultas y modificaciones.

La base de datos que elegimos de las propuestas fue la lista de los 100 mejores libros de un repositorio público de github en formato JSON. Este archivo incluye información detallada de cada libro (autor, país, enlace de imagen, idioma, enlace con información del libro, cantidad de páginas, título y año de publicación). El tipo de datos utilizado son cadenas de texto y números enteros.

El sistema operativo utilizado fue Windows y para la implementación no usamos un entorno virtual, sino el entorno predeterminado del sistema.

DESCRIPCIÓN DEL ENTORNO DE TRABAJO DEL SERVIDOR

Desarrollamos el servidor en Python 3.12 utilizando FastAPI. Utilizamos Uvicorn como servidor ASGI para manejar las solicitudes.

- Dependencias instaladas: Fastapi, requests, Uvicorn (pip install fastapi uvicorn requests).

- Librerías importadas: os, json.
- Archivos principales: books.json (almacena los datos) y TP_api.py (contiene el servidor).
- Ejecución local:

```
uvicorn TP_api:app --reload --host 0.0.0.0 --port 8000
```

Un inconveniente que tuvimos al hacer el API fue al descargar el archivo que contiene los datos de los libros. En un principio estábamos utilizando: *requests.get("https://github.com/benoitvallon/100-best-books/blob/master/books.json")*
Esto realizaba la descarga del HTML de la página pero lo que necesitábamos era solamente el archivo JSON del repositorio. Es por eso que utilizamos el enlace raw (archivo sin procesar): *requests.get("https://raw.githubusercontent.com/benoitvallon/100-best-books/refs/heads/master/books.json")*
Aparte de eso, no tuvimos ningún otro inconveniente.

DESCRIPCIÓN DEL ENTORNO DE TRABAJO DEL DEL CLIENTE

El cliente fue desarrollado en Python 3.12, utilizando solo la librería requests para realizar solicitudes HTTP al servidor API.

- Dependencias instaladas: requests (pip install requests).
- Archivos principales: cliente.py (contiene el cliente).
- Ejecución local: *py cliente.py*

Al momento de abrir el servidor con Uvicorn, obtenemos la IPv4 de la pc que hará la parte de servidor. Esa IPv4 deberá agregarse en la línea 8 del programa cliente.py, para que las solicitudes HTTP se realicen a esa ip.
No tuvimos ningún inconveniente desarrollando el entorno del cliente.

DESCRIPCIÓN DE LA BASE DE DATOS ELEGIDA

La base de datos elegida es un archivo JSON que contiene información sobre la colección de libros.

El archivo books.json cuenta con un total de 100 objetos, cada uno representando un libro de la colección. Cada libro tiene los siguientes atributos:

Atributo	Tipo de Dato	Descripción
author	String (str)	El nombre del autor del libro.
country	String (str)	El país de origen del autor.
imageLink	String (str)	La URL relativa a la imagen de la portada del libro.
language	String (str)	El idioma en el que está escrito el libro.
link	String (str)	Enlace a la página de Wikipedia del libro.
pages	Integer (int)	La cantidad de páginas que tiene el libro.
title	String (str)	El título del libro.
year	Integer (int)	El año en que el libro fue publicado.

Ejemplo de un objeto libro:


```
{
"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
},
```

PROCEDIMIENTO

Descripciones y ejemplos de las instrucciones para comunicarse con la API

El cliente se ejecuta como una aplicación de consola, proporcionando un menú interactivo para realizar operaciones CRUD sobre la base de datos de libros.

Las operaciones disponibles son:

- Buscar libro por título:

  requests.get(f"http://{server_ip}:8000/books/title/{title}")

- Buscar libro por autor.

  `requests.get(f"http://{server_ip}:8000/books/author/{author}")`

- Agregar un libro.

  `requests.post(f"http://{server_ip}:8000/books", json = libro)`

- Eliminar un libro por título.

  `requests.delete(f"http://{server_ip}:8000/books/by-title?title={title}")`

- Eliminar un libro por autor.

  `requests.delete(f"http://{server_ip}:8000/books/by-author?author={author}")`

- Ver todos los libros.

  `requests.get("http://{server_ip}:8000/books")`

- Actualizar libro por título.

  `requests.put(f"http://{server_ip}:8000/books/{title}", json=book)`

INSTRUCCIONES DE USO DEL CLIENTE

1. Ejecutar el código del cliente:

   Una vez que el servidor esté activo, se deberá ejecutar el archivo

llamado “Cliente.py”.

2. Menú Interactivo:

   Luego de ejecutar el código, se mostrará en la consola un menú interactivo con el cual el cliente podrá realizar diferentes consultas y modificaciones en el servidor API. Las opciones son las siguientes:

   - 1. Buscar libro por título.
   - 2. Buscar libro por autor.
   - 3. Agregar un libro.
   - 4. Eliminar un libro por título.
   - 5. Eliminar un libro por autor.
   - 6. Ver todos los libros.
   - 7. Actualizar libro por título.
   - 8. Salir.

3. Realizar una consulta/modificación:

   Para poder seleccionar una de las opciones del menú, el usuario deberá escribir en la consola el número correspondiente a la acción que desee realizar. Si la opción es inválida, se le solicitará nuevamente al usuario que ingrese un valor.

4. Finalizar ejecución:

   Para salir del menú, el usuario deberá ingresar en la consola “8”. Esto terminará la ejecución del cliente, mientras que el servidor permanecerá activo con los cambios realizados.

REFERENCIAS

1. https://www.youtube.com/watch?v=J0y2tjBz2Ao&t=814s
2. https://www.w3schools.com/python/ref_requests_post.asp
3. https://fastapi.tiangolo.com/deployment/manually/#use-the-fastapi-run-command