Yale BIDS Technical Assessment Webapp Document

Provided by Niloofar Didar

[Niloofar.didar@gmail.com](mailto:Niloofar.didar@gmail.com)

**BASIC PROJECT DESCRIPTIONS**

Develop a web page designed to enable users to efficiently search for and retrieve information or data corresponding to specific term from PubMed.

Implemented tasks as required:

1. **Home page**

The homepage, depicted in the figure below, comprises a user interface (UI) facilitating access to the Pubmed website via search and fetch functions. It features a button directing users to the API documentation, along with two additional functions enabling direct term search and task ID retrieval stemming from search tasks.

A screenshot of a computer

Description automatically generated

1. **API Implementation/Documentation for Search and Fetch:**

AS the project requirement, I developed and implemented essential API endpoints to accommodate search queries tailored to user-specified terms

* Search:

This API enables users to search for any term (eg, kidney disease as shown in the figure below) on the Pubmed website. The results are displayed in a JSON file format, structured according to the specifications outlined in Yale's documentation provided below.

{

"records": 212111,

"query": "alzheimer disease",

"task\_id": "7fd381bf3cbe28e892e163db81b9e2cd"

}

A screenshot of a computer

Description automatically generated

* Fetch:

The fetch API allows users to retrieve additional details concerning the searched task ID. The results are presented in a JSON file format, containing information aligned with the specifications outlined in Yale's documentation.

{

"task\_id": "7fd381bf3cbe28e892e163db81b9e2cd"

"status":"completed",

"result":{

"pmids":[7952237,37506310,32397415,...]

},

"created\_time":"2024-03-12 13:51:02",

"run\_seconds": 75,

}

Below is a figure of API result for the fetch function using the kidney disease task ID:

A screenshot of a computer

Description automatically generated

1. **UI for User Search and Fetch:**

Additionally, this application provides users with the capability to conduct searches for terms without utilizing an API, as depicted in the figure below. For instance, when searching for the term "kidney disease," the results are displayed via the URL [localhost:5000/search?term=kidney+disease](http://localhost:5000/search?term=kidney+disease) through json data.

{

"query": "kidney disease",

"records": 752247,

"task\_id": " cae7eae5-a3a3-400e-821b-7ec045c339bb"

}

A screenshot of a web application

Description automatically generated

A screenshot of a computer

Description automatically generated

Similarly, when utilizing the task ID "cae7eae5-a3a3-400e-821b-7ec045c339bb" for the search function, the JSON result for that record will be displayed through the following URL format: [localhost:5000/fetch?taskID=cae7eae5-a3a3-400e-821b-7ec045c339bb](http://localhost:5000/fetch?taskID=cae7eae5-a3a3-400e-821b-7ec045c339bb) as is displayed in the image below:

A screenshot of a computer

Description automatically generated

1. **Containerization:**

The project has been containerized using Docker to facilitate easy deployment. To build your Docker image, open your terminal, navigate to the directory containing the Dockerfile, and execute the commands provided in the Readme.txt file. Further details on running a sample of the application through Docker can be found in the figure below.

A screenshot of a computer

Description automatically generated