Cindy R. Legaspino

BSI/T 3-E

Documentation

This backend API provides endpoints to interact with course data stored in a MongoDB database. It allows users to retrieve information about courses, including backend courses and BSIS/BSIT courses.

Installation and Setup

1. MongoDB Database Setup:

- Download and install the MongoDB Database tool from the official MongoDB website.
- Follow the installation instructions provided for your operating system

2. Import Course Data:

- Download the provided code repository.
- Navigate to the directory containing the `courses.json` file.
- Use the following command to import the course data into MongoDB:

```
mongoimport --db mongotest --collection courses --file courses.json --jsonArray
```

This command imports the data from the `courses.json` file into a collection named `courses` in the `mongo-test` database.

3. Node.js Setup:

- Ensure you have Node.js installed on your system. If not, download and install it from the official Node.js website.

4. Project Setup:

- Create a new directory for the project and navigate to it in the terminal.
- Run `npm init -y` to create a `package.json` file and follow the prompts.
- Install the required dependencies (`express`, `mongoose`, `fs`) using the command `npm install`.
- Start the server using the command `node app.js`. By default, the server runs on port `3220`.

API Endpoints

Retrieve All Backend Courses:

- Endpoint: `GET /backend-courses`
- URL: `http://localhost:3220/backend-courses`
- Description: Retrieves all published backend courses sorted alphabetically by their names.

Retrieve BSIS/BSIT Courses

- Endpoint: `GET /bsis-bsit-courses`
- URL: `http://localhost:3220/bsis-bsit-courses`
- Description: Retrieves all published BSIS (Bachelor of Science in Information Systems) and BSIT (Bachelor of Science in Information Technology) courses from the curriculum.

Data Validation

- Validation is conducted at every stage to guarantee the precision and reliability of the acquired data. Course data is stored in a MongoDB database, and Mongoose serves as an Object Data Modeling (ODM) tool for defining schemas and managing interactions with the database.

Error Handling

- Error handling is implemented to catch and log any errors that occur during database operations or API requests. Proper HTTP status codes are returned along with error messages to indicate the nature of the problem.

Challenges

In completing this activity, several challenges were encountered, especially with limited knowledge about MongoDB and Node.js. Connecting to MongoDB and understanding its operation was particularly challenging. Additionally, grappling with Node.js concepts required extensive online research and learning. The hardest challenge that I faced is the laptop that im using, it is lag and consumes so much time in doing this activity. The patience between me and this activity challenges me.

During development, issues arose when trying to retrieve data in Postman, resulting in empty responses. After troubleshooting, it was discovered that incorrect tags were being used, leading to incorrect data retrieval. Seeking help from peers and taking breaks were instrumental in resolving these challenges and completing the task successfully. Focusing in each and every details of words, commas, parenthesis, o=and other symbols were always included in why this activity give me challenge.