Name: Madhesh Hariharran S

Roll no.: 22110025

SECURIN ASSESSMENT DOCUMENTATION

To run the code, do the following:

1) Navigate to the backend folder.
2) Execute "npm init -y" in the terminal.
3) Install dotenv, express, fs, mysql, mysql2, path, nodemon -D and sequelize.
4) In package-json change the scripts with the following:

   "start": "node index.js",

   "dev": "nodemon index.js"

5) Open db.js and replace with MySQL username, password, database as mentioned in the comments in the file.
6) In terminal paste "npm run dev".

Things done in the Assessment:

1) Read and parsed the provided JSON file that contains recipe data. Then Stored the data into a MYSQL database named securing in a table named recipes.
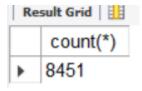
   Logic:

   - Read the raw json file.
   - Converted json objects into arrays.
   - Used sequelize to create a table and insert the data into the table.

```
1 •  use securin;
2
3 •  select * from recipes
4    order by id
5    limit 5;
```

id	cuisine	title	rating	prep_time	cook_time	total_time	description	nutrients
1	Southern Recipes	Sweet Potato Pie	4.8	15	100	115	Shared from a Southern recipe, thi...	{"calories": "389 kcal", "fatContent":...
2	Southern Recipes	Fresh Southern Peach Cobbler	4.7	20	40	60	This peach cobbler recipe makes t...	{"calories": "562 kcal", "fatContent":...
3	Southern Recipes	Best Fried Green Tomatoes	4.7	5	15	20	Fried green tomatoes are a quick a...	{"calories": "510 kcal", "fatContent":...
4	Southern Recipes	Best Jambalaya	4.8	20	45	65	A spicy jambalaya with chicken, an...	{"calories": "465 kcal", "fatContent":...
5	Southern Recipes	Authentic Louisiana Red Beans an...	4.5	25	185	690	This easy authentic Louisiana red b...	{"calories": "630 kcal", "fatContent":...

Result Grid

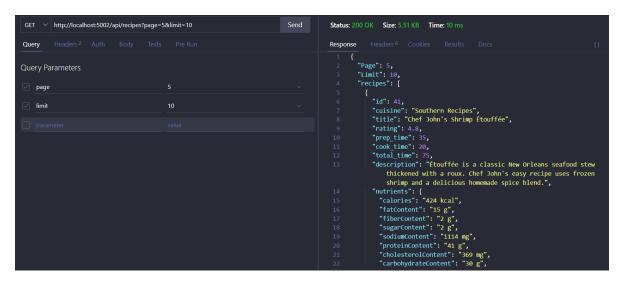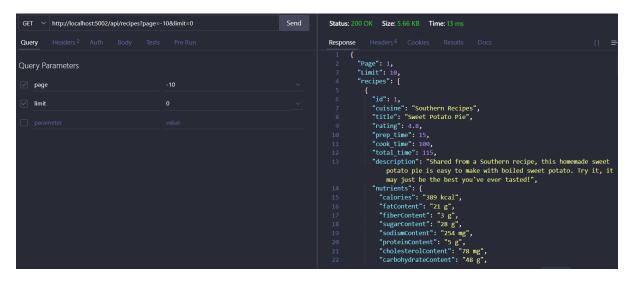|	count(*)
▶	8451

2) Developed two APIs
   a. To expose an endpoint to get all recipes in a paginated and sorted manner.
      Logic:
      - Created a route for getting all recipes and a controller for it.
      - Parsed the queries and handled errors like <=0 by changing the page and limit values to default values.
      - Found offset
      - Used findAll() method to get the recipes based on offset and limit.

I added error handling by making the page and limit parameters to their default values when a value <=0 is entered.

b. To expose an endpoint to search for recipes based on various fields
   Logic:
   - Created a route and controller to handle the search of recipes based on various fields
   - Parsed standard query parameters like title, cuisine, rating, and total_time, and used Sequelize operators for proper SQL WHERE conditions.
   - Fetched SQL-filtered recipes using the findAll() method from Sequelize with raw: true for better performance and raw data access.
   - Converted nutrient strings like "562 kcal" or "400 mg" to numeric values using regex to extract digits for accurate comparison.
   - Used a custom compare function to perform operations like <, <=, >, >=, = dynamically, based on the query.