Food App

Problem Statement 2

Solved By: Harshit Gupta (Swiggy IPP B1-033)

About Food App

- ★ Food App is a backend Application Programming Interface (API)
- ★ Built to provide common solutions like Registration and Managing Food Database
- ★ Built over MERN stack
- ★ Entire source available on GitHub

Technology Stack

- ★ Built over Node.js with Express.js
- ★ Express.js handles API endpoints, different requests and responses
- ★ Uses MongoDB as Database
- ★ Mongoose.js Node Module to handle MongoDB Object Modeling

API Endpoints

All API Endpoints get available at http://localhost:4000/api

★ Endpoints:
  - /register
  - /authenticate
  - /users
  - /users/:userID
  - /food
  - /food/:foodID

API Endpoint: /register

- Method: POST
- Takes data, registers it and returns data of format:

  { id: Number, username: String, email: String, password: String, address: {houseno: Number, street: String, city: String, state: String, zip: Number} }

- Username is checked into database prior to registering it, to ensure its uniqueness.
- Status codes:
  - If successful, returns 201 (success)
  - If username already exists, returns 302 (Found)
  - In any other case, returns 403 (Forbidden)

API Endpoint: /authenticate

- ★ Method: POST
- ★ Takes JSON data of format:

  { username: String, password: String, address: {houseno: Number, street: String, city: String, state: String, zip: Number} }

- ★ Username and passwords are matched within existing data, to check for prior registration.
- ★ Status codes:
  - ○ If username and password combination is found, returns 200 (OK)
  - ○ In any other case, returns 403 (Forbidden)

API Endpoint: /users

- ★ Method: GET
- ★ Returns an array collection of all users registered in the database.
- ★ Status code:
  - ○ returns 200 (OK)

API Endpoint: /users

- ★ Method: PUT
- ★ Updates the user with given ID (if found in database).
- ★ If given user ID is not found, returns a “User Not Found” message JSON object.
- ★ Status code:
    - If successful, returns 200 (OK)

API Endpoint: /users/:userID

- ★ Method: GET
- ★ Returns a specific user that’s found registered in the database with given :userID parameter.
- ★ Discarded username and ID is returned on successful.
- ★ If user with provided user ID is not found, returns “User Not Found” message JSON Object.
- ★ Status code:
  - ○ returns 200 (OK)

API Endpoint: /users/:userID

- ★ Method: DELETE
- ★ Returns a specific user that’s found registered in the database with given :userID parameter.
- ★ If user with provided user ID is not found, returns “User Not Found” message JSON Object.
- ★ Status code:
  - ○ returns 200 (OK)

API Endpoint: /food

- ★ Method: POST
- ★ Returns a specific user that’s found registered in the database with given :userID parameter.
- ★ If user with provided user ID is not found, returns “User Not Found” message JSON Object.
- ★ Status code:
  - ○ returns 200 (OK)

API Endpoint: /food/:foodID

- ★ Method: GET
- ★ Returns a food specified with given :foodID parameter.
- ★ Takes the following payload:

  { foodId: Number, foodName: String, foodCost: Number, foodType: Indian/Chinese/Mexican }

- ★ Returns the same payload with generated ID added to it
- ★ If user with provided user ID is not found, returns “User Not Found” message JSON Object.
- ★ Status code:
  - returns 200 (OK)

Scope of Improvements

- ★ No Authentication Token in use, directly exchanges username and password for every user authentication, which could lead to security issues
- ★ Food ID generated does not ensure uniqueness, deletions may lead to some repetitive IDs.
- ★ Needs to be online to access MongoDB Atlas Database.
- ★ Limited Functionality, additional endpoints can be used for added use cases on same set of data

Thank You!

- ★ Feedbacks and Suggestions are most welcome!