API Documentation

Team A

Food Order Recommendation System

Dated: 11 May 2023

Prepared by: Ashwin Raguraj

Introduction

The API documentation describes a RESTful API that allows users to manage a collection of books. The API supports the following HTTP verbs:

- GET: Used to retrieve information about a book or a collection of books.
- POST: Used to create a new book in the collection.
- PUT: Used to update an existing book in the collection.
- DELETE: Used to delete a book from the collection.

To use the API, users can make HTTP requests to the appropriate API endpoints using one of these verbs. The API will then respond with the proper status code and response data.

1. `GET /api/users`: This endpoint allows you to retrieve a list of all the users in the system. You can use this endpoint to retrieve basic information about each user, such as their username, email address, and a link to their profile picture. You can also use query parameters to filter the list of users based on various criteria, such as username, email, or date created.

2. `POST /api/users`: This endpoint allows you to create a new user in the system. You must provide a username, email address, and password for the new user. Once the user is created, you will receive a response containing a JSON object with the user's details, including their unique ID in the system.

3. `GET /api/users/{id}`: This endpoint allows you to retrieve details about a specific user, identified by their unique ID in the system. You will receive a JSON object containing the user's details, including their username, email address, profile picture link, and date created.

4. `PUT /api/users/{id}`: This endpoint allows you to update the details of a specific user, identified by their unique ID in the system. You can update the user's username, email address, and profile picture link using this endpoint. You will need to provide a JSON object containing the updated user details in the request body.

5. `DELETE /api/users/{id}`: This endpoint allows you to delete a specific user from the system, identified by their unique ID. Once a user is deleted, their information is removed from the system and cannot be recovered. Note that this action cannot be undone, so use this endpoint with caution.

Endpoints

Signup

Request : **POST /signup/**

Field Name	Type	Required	Description
email	string	Yes	User's email address
password	string	Yes	User's password
first_name	string	Yes	User's first name
last_name	string	Yes	User's last name

Response

HTTP Status	Response Body	Description
200	{"success": true}	User account created
400	{"error": "Missing required field(s)"}	Required field missing
400	{"error": "User with this email already exists"}	User already exists

Login

Request: **POST /login/**

Field Name	Type	Required	Description
email	string	Yes	User's email address
password	string	Yes	User's password

Response

HTTP Status	Response Body	Description
200	{"success": true}	Login successful
400	{"error": "Missing required field(s)"}	Required field missing
400	{"error": "Invalid email or password"}	Authentication failed

Reset Password

Request: **POST /reset_password/**

Field Name	Type	Required	Description
email	string	Yes	User's email address
new_password	string	Yes	User's new password
confirm_password	string	Yes	Confirmation of new password

Response

HTTP Status	Response Body	Description
200	{"message": "Password reset successful"}	Password reset successful
400	{"error": "Missing required field(s)"}	Required field missing
400	{"error": "User with this email does not exist"}	User not found in database
400	{"error": "New password and confirm password do not match"}	Passwords don't match

User List

Request: **GET /api/users/**

Response

HTTP Status	Response Body	Description
200	[{user object}, {user object}, ...]	List of all user objects

User Detail

Request:**GET /api/users/:id/**

Response

HTTP Status	Response Body	Description
200	[{user object}	User object with corresponding id

Add User

Request:**POST /api/users/**

Field Name	Type	Required	Description
email	string	Yes	User's email address
password	string	Yes	User's password
first_name	string	Yes	User's first name
last_name	string	Yes	User's last name

Response

HTTP Status	Response Body	Description
201	{user object}	User account created
400	{"error": "Invalid data provided"}	Invalid

Update User

Request: **PUT /api/users/:id/**

Field Name	Type	Required	Description
email	string	Yes	User's email address
first_name	string	Yes	User's first name
last_name	string	Yes	User's last name

Response

200 OK	The user was updated successfully, and the updated user information is returned in the response body.
400 Bad Request	The request was invalid or malformed, and the error message is returned in the response body.

Delete User

Request: **DELETE /api/users/:id/**

Response

HTTP Status Code	Description
204	The user was successfully deleted. No response body is returned.
404	The specified user does not exist. An error message is returned.

Food Data

Request:**GET /api/food/**

Response

```
[
  {
    "id": 1,
    "restaurant": "Burger Corner",
    "price": "100",
    "food": "French Fries ",

    "linkImg": "https://www.thestatesman.com/wp-content/uploads/2019/05/french-fries.jpg",
"restaurantImg":"https://media.gettyimages.com/photos/bliss-hot-rod-grill-restaurant-in-florence-or
egon-picture-id520754024"

  },]
```