API DOCUMENTATION

HAMIM SHAIKH
19901125
CSE4OAD

October 2020

1 Introduction

This Tomcat application provides a RESTful API that allows the user to interact with a database named Fridgedb. The database configuration is not part of this documentation. All replies to requests and expected inputs will be in JSON. Following is an example of Json format.

```
{
    "id": 6,
    "item": {
        "name": "Paddle Pop",
        "expires": false
    },
    "date": {
        "year": 2019,
        "month": 9,
        "day": 26
    },
    "quantity": 1,
    "section": "FREEZER"
}
```

2 GET Requests

2.1 Item

2.1.1 Get all Items

To get a JSON array of all the items issue a GET request to the base fridge/api/ endpoint with a item parameter added.

Example URL

```
http://<tomcat-server:port>/fridge/api/item/
```

Expected output

```
[
    {
        "name": "Beef",
        "expires": true
    },
    {
        "name": "Tangerines",
        "expires": false
    },
  ...
]
```

Expected HTTP status code: 200 OK

2.1.2 Get all items that can expire

To get a JSON array of all expired items issue a GET request to the base fridge/api/ endpoint with a parameter item/e added.

Example URL

```
http://<tomcat-server:port>/fridge/api/item/e
```

Expected output

```
[
    {
        "name": "Beef",
        "expires": true
    },
    {
        "name": "Broccoli",
        "expires": true
    },
    ...
]
```

Expected HTTP status code: 200 OK

2.2 Grocery

2.2.1 Get all grocery items

To get a JSON array of all the grocery items issue a GET request to the base url endpoint with a parameter grocery.

Example URL

```
http://<tomcat-server:port>/fridge/api/grocery/
```

Expected output

```
[
    {
        "id": 6,
        "item": {
            "name": "Paddle Pop",
            "expires": false
        },
        "date": {
            "year": 2019,
            "month": 9,
            "day": 26
        },
        "quantity": 1,
        "section": "FREEZER"
    },
    {
        "id": 19,
        "item": {
            "name": "Beef",
            "expires": true
        },
        "date": {
            "year": 2019,
            "month": 9,
            "day": 25
        },
        "quantity": 1,
        "section": "MEAT"
    },
    ...
]
```

Expected HTTP status code: 200 OK

2.2.2 Get grocery items that can expire

To get a JSON array of all expired items issue a GET request to the base fridge/api/ endpoint with a parameter grocery/e added.

Example URL

```
http://<tomcat-server:port>/fridge/api/grocery/e
```

Expected output

```
[
    {
        "id": 19,
        "item": {
            "name": "Beef",
            "expires": true
        },
        "date": {
            "year": 2019,
            "month": 9,
            "day": 25
        },
        "quantity": 1,
        "section": "MEAT"
    },
    {
        "id": 32,
        "item": {
            "name": "Beef",
            "expires": true
        },
        "date": {
            "year": 2019,
            "month": 9,
            "day": 26
        },
        "quantity": 2,
        "section": "MEAT"
    },..

]
```

Expected HTTP status code: 200 OK

2.3 GET Request Errors

	Error Message	Possible Reason	HTTP Status Code
1	Resource not found!	Incorrect base URL endpoint	400 Bad request
2	Resource target not defined	Base URL is not being targeted to the database	403 forbidden
3	Grocery with id 'id number' not found	Grocery id doesn't exist in Grocery table	404 not found

3 PUT

Whenever a grocery item is used an update operation is performed by PUT request which decreases the quantity of grocery item by 1 and returns back updated version of data in JSON format. If quantity of grocery item is 1 no update operation is performed and an empty JSON object is returned.

The operation is performed by giving grocery id as a parameter to the end point.

Example URL

```
http://<tomcat-server:port>/fridge/api/grocery/32
```

Expected Output

```
{
    "id": 32,
    "item": {
        "name": "Beef",
        "expires": true
    },
    "date": {
        "year": 2019,
        "month": 9,
        "day": 26
    },
    "quantity": 1,
    "section": "MEAT"
}
```

Expected HTTP status code: 200 OK

3.1 PUT Request Errors

	Error Message	Possible Reason	HTTP Status Code
1	Resource not found!	Incorrect base URL endpoint	400 Bad request
2	Resource target not defined	Base URL is not being targeted to the database	403 forbidden
3	Grocery with id 'id number' not found! Cannot Update!	Grocery id doesn't exist in Grocery table	404 Not found
4	Attribute id required! Cannot Update!	Grocery id hasn't been provided as a parameter in the base url	403 Forbidden
5	There is only one: Ice Cream (bought on 26/09/2019) - use DELETE instead.	Grocery quantity is 1 and cannot be reduced	405 Method not allowed

4 DELETE Request

Delete the grocery from fridge database using Grocery id. The operation is performed by giving grocery id as a parameter at the end point.

Example URL

```
http://<tomcat-server:port>/fridge/api/grocery/34
```

Expected Output

```
{"message":"Grocery with id 34 deleted!"}
```

Expected HTTP status code: 200 OK

4.1 DELETE Request Errors

	Error Message	Possible Reason	HTTP Status Code
1	Resource not found!	Incorrect base URL endpoint	400 Bad request
2	Resource target not defined	Base URL is not being targeted to the database	403 forbidden
3	Grocery with id 'id number' not found! Cannot Delete!	Grocery id doesn't exist in Grocery table	404 not found
4	Attribute id required! Cannot Delete!	Grocery id hasn't been provided as a parameter in the base url	403 Forbidden

5 POST Request

This creates a new Grocery item in Fridgedb database. The operation is performed by giving data in JSON format as an input. The following key value pairs must be provided

1. Key id : Takes value an integer number, However the id will be generated based on grocery id automatically.
2. Item : Item key takes two more key value pairs.
   (a) First key is the name which takes a values as a string present in Item table.
   (b) Second key is the expires which takes in Boolean value.
3. Date : takes following three key values pairs.
   (a) Year key takes an integer year value.
   (b) Month key takes an integer month value.

   (c) Day key takes an integer day value.

4. Quantity : Takes an integer value parameter

5. Section : Takes a string value representing fridge section.

Example URL

```
http://<tomcat-server:port>/fridge/api/grocery/
```

Example Input

```
{
       "id": 1 ,
       "item": {
           "name": "Milk",
           "expires": true
       },
       "date": {
           "year": 2019,
           "month": 9,
           "day": 26
       },
       "quantity": 3,
       "section": "CRISPER"
   }
```

Example Output

```
{"message":"created ok! Grocery id 34"}
```

Expected HTTP status code : 200 OK

5.1 POST request errors

	Error Message	Possible Reason	HTTP Status Code
1	Resource not found!	Incorrect base URL endpoint	400 Bad request
2	Resource target not defined	Base URL is not being targeted to the database	403 forbidden
3	{}	No input provided	412 Precondition failed
4	{message:java.io.EOFException: End of input at line 7 column 1}	Incorrect JSON format	412 Precondition failed