

To run the code, do the following -

1. Clone the repository and go inside the folder
2. Run the command "npm install"
3. Run the command "npm start"

1. Adding food items using POST API

The screenshot displays a REST client interface with a dark theme. At the top, the method is set to 'POST' and the URL is 'http://localhost:8000/foodItem/add'. Below this, there are tabs for 'Query', 'Headers', 'Auth', 'Body', and 'Tests'. The 'Body' tab is selected and highlighted with a red underline. Under the 'Body' tab, there are sub-tabs for 'Json', 'Xml', 'Text', 'Form', 'Form-encode', 'Graphql', and 'Binary'. The 'Json' sub-tab is selected and highlighted with a red underline. The main area shows the JSON content of the request body, which is a single object with the following properties: 'name' (value: 'Beer'), 'calories' (value: 30), 'protein' (value: 0.3), 'carb' (value: 2), 'acceptedUnits' (value: ['g']), and 'itemWeight' (value: 100). The JSON is formatted with line numbers 1 through 8 on the left. Below the JSON content, there is a 'Format' button. At the bottom of the interface, the status bar shows 'Status: 200 OK', 'Size: 11 Bytes', and 'Time: 17 ms'.

```
POST http://localhost:8000/foodItem/add

Query Headers Headers Headers Headers Headers Headers Headers
Body Body Body Body Body Body Body Body
Tests Tests Tests Tests Tests Tests Tests Tests

Json Xml Text Form Form-encode Graphql Binary

Json Content

1 {
2   "name": "Beer",
3   "calories": 30,
4   "protein": 0.3,
5   "carb": 2,
6   "acceptedUnits": ["g"],
7   "itemWeight": 100
8 }

Format

Status: 200 OK Size: 11 Bytes Time: 17 ms
```

2. Creating meals by referencing food items into the meal items array using POST API

POST

http://localhost:8000/meal/create

Query

Headers²

Auth

Body¹

Tests

Json

Xml

Text

Form

Form-encode

GraphQL

Binary

Json Content

```
1 {  
2   "name": "Meal 2",  
3   "category": "Lunch",  
4   "foodItems": ["62dfb89c8d437a3656b4c7cf", "62dfb8b08d437a3656b4c7d1",  
5                 "62dfb8c48d437a3656b4c7d3", "62dfb8d98d437a3656b4c7d5"]  
6 }
```

Format

Status: 200 OK Size: 11 Bytes Time: 73 ms

3. Creating user using dummy data using POST API

POST

http://localhost:8000/user/create

Query

Headers²

Auth

Body¹

Tests

Json

Xml

Text

Form

Form-encode

Graphql

Binary

Json Content

```
1 {  
2   "name": "Soumyadeep Paul",  
3   "calorieRequirement": 1200,  
4   "mealPlan": [{  
5     "date": "2020-05-11",  
6     "meal": "62dfc3c0eca044e9058917bc"  
7   }]  
8 }
```

Format

Status: 200 OK Size: 20 Bytes Time: 130 ms

4. Updating meals using PATCH API

PATCH 

http://localhost:8000/meal/update

Query

Headers ²

Auth

Body ¹

Tests

Json

Xml

Text

Form

Form-encode

Graphql

Binary

Json Content

1 

2 "name": "Meal 5",

3 "category": "Lunch",

4 "foodItems": ["62dfb89c8d437a3656b4c7cf", "62dfb8b08d437a3656b4c7d1",

5 "62dfb8c48d437a3656b4c7d3", "62dfb8d98d437a3656b4c7d5"]

5 

Format

Status: 200 OK Size: 11 Bytes Time: 52 ms