

API Contract for Supermarket Billing

In this the model/schema is having the field – items_id, name, category, subcategory and amount. Database used – sqlite3

GET request:

localhost:8000/api/items/

The screenshot displays the Postman interface for a GET request to the endpoint `localhost:8000/api/items/`. The request is configured with a JSON body containing an array of items. The response is a 200 OK status with a JSON body containing an array of three items.

Request Details:

- Method: GET
- URL: `localhost:8000/api/items/`
- Body (raw):

```
1 [{"name": "abc",
2   "category": "food",
3   "subcategory": "fruit",
4   "amount": 700
5 }]
```

Response Details:

- Status: 200 OK
- Time: 14 ms
- Size: 957 B
- Body (pretty):

```
1 {
2   {
3     "name": "Prabudh",
4     "category": "food",
5     "subcategory": "fruit",
6     "amount": 50
7   },
8   {
9     "name": "Goal",
10    "category": "sports",
11    "subcategory": "football",
12    "amount": 1000
13  },
14  {
15    "name": "Bat",
16    "category": "sports",
17    "subcategory": "cricket",
18  }
19 }
```

localhost:8000/api/items/?category=food

Scratch Pad

Overview

GET localhost:8000/api/items/?category=food

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 [
2   {
3     "name": "abc",
4     "category": "food",
5     "subcategory": "fruit",
6     "amount": 700
7   }
8 ]
```

Body Cookies Headers (10) Test Results

200 OK 116 ms 740 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "name": "Prabudh",
3   "category": "food",
4   "subcategory": "fruit",
5   "amount": 50
6 },
7 {
8   "name": "potato",
9   "category": "food",
10  "subcategory": "vegetables",
11  "amount": 20
12 },
13 {
14   "name": "apple",
15   "category": "food",
16   "subcategory": "fruit",
17   "amount": 20
18 }
```

localhost:8000/api/items/?subcategory=vegetables

Scratch Pad

Overview

GET localhost:8000/api/items/?subcategory=vegetables

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 [
2   {
3     "name": "abc",
4     "category": "food",
5     "subcategory": "fruit",
6     "amount": 700
7   }
8 ]
```

Body Cookies Headers (10) Test Results

200 OK 9 ms 390 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "name": "potato",
3   "category": "food",
4   "subcategory": "vegetables",
5   "amount": 20
6 }
```

localhost:8000/api/items/?name=ball

The screenshot shows the Postman interface with a GET request to `localhost:8000/api/items/?name=ball`. The request is in the "Body" tab, and the response is displayed in the "Body" tab. The response is a JSON object with the following structure:

```
{
  "name": "ball",
  "category": "sports",
  "subcategory": "fruit",
  "amount": 60
}
```

The status bar indicates a 200 OK response with a 6 ms latency and 385 B of data.

POST request – to create new data and storing it into the database

localhost:8000/api/items/

The screenshot shows the Postman interface with a POST request to `localhost:8000/api/items/`. The request is in the "Body" tab, and the response is displayed in the "Body" tab. The response is a JSON object with the following structure:

```
{
  "name": "stumps",
  "category": "sports",
  "subcategory": "cricket",
  "amount": 500
}
```

The status bar indicates a 201 Created response with a 25 ms latency and 393 B of data.

PUT request – to update the data in the database using primary key

localhost:8000/api/items/11/ -- where 11 is the primary key/id

The screenshot shows the Postman application interface. On the left is the 'Scratch Pad' sidebar with a 'You don't have any collections' message. The main workspace is titled 'localhost:8000/api/items/11/'. The request method is 'PUT' and the URL is 'localhost:8000/api/items/11/'. The 'Body' tab is selected, showing a JSON payload:

```
{  "name": "stumps",  "category": "sports",  "subcategory": "cricket",  "amount": 700}
```

. The 'Headers' tab shows 10 headers. The 'Test Results' tab shows a status of 200 OK, 21 ms, and 395 B. The 'Pretty' view of the response shows the same JSON object.

DELETE request - to delete the data in the database using primary key

localhost:8000/api/items/12/ -- where 11 is the primary key/id

The screenshot shows the Postman application interface. The workspace is titled 'localhost:8000/api/items/12/'. The request method is 'DELETE' and the URL is 'localhost:8000/api/items/12/'. The 'Body' tab is selected, showing a JSON payload:

```
{  "name": "stumps",  "category": "sports",  "subcategory": "cricket",  "amount": 700}
```

. The 'Headers' tab shows 10 headers. The 'Test Results' tab shows a status of 204 No Content, 14 ms, and 365 B. The 'Pretty' view of the response shows:

```
{  "delete": "deleted 12 successfully"}
```