**Test Cases – Restful API Implementation – Online Store**

All tests have been performed using the Postman Extension for Google Chrome. All input for POST and PUT requests have to be given in JSON Format. Otherwise the API will return:

{

"error": "Input not in correct format"

}

Since the results are sometimes fetched from Memcache, you may sometimes not receive the updated expected response since sometimes cached responses may be returned. Items are cached for a default timespan of 30 seconds.

# Database online-store

## Table structure for table categories

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** |
| ***category\_id*** | int(11) | No |  |
| Name | varchar(30) | No |  |
| Description | Text | Yes |  |
| parent\_id | int(11) | No |  |

## Dumping data for table categories

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | Electronics | Sample Electronics Description | -1 |
| 2 | Books | Sample Books Description | -1 |
| 3 | Home and Kitchen | Sample Home and Kitchen Description | -1 |
| 4 | Music | Sample Music Description | -1 |
| 5 | Mobiles | Mobiles Description | 1 |
| 6 | Computers | Sample Computers Description | 1 |
| 7 | Spiritual | Sample Spiritual Books Description | 2 |
| 8 | Home Furnishing | Sample Home Furnishing Description | 3 |
| 9 | Home Appliances | Sample Home Appliances Description | 3 |
| 10 | Kitchen Appliances | Sample Kitchen Appliances Description | 3 |

### User Access

Only administrators can access the API. I have made a user with privileges to test the authorization functionality. The administrator user and password to access the API are:

Username: ‘kanishk’

Password: ‘password’

Test Case 1:

username: random

password: random

Output:

Please login with administrator credentials

Test Case 2:

username: kanishk2

password: password2

Output:

Only Administrators are allowed.

Test Case 3:

username:kanishk

password: password

Output:

Successfully Logged In

### Listing categories

Various query parameters can be given to retrieve the results intended in a particular way. Any of the query parameters can be combined in any combination to retrieve the result.

Test Case 1:

GET http://localhost/online-store/categories

Output:

The first ten categories with all fields order by their category id are returned. Ten records are returned since by default ten resources are fetched unless you pass a per\_page parameter in the query. And by default resources are ordered by category\_id.

Conclusion: Result as expected.

Test Case 2:

GET http://localhost/online-store/categories?fields=name,parent\_id&page=2&per\_page=5

Output:

Only name and parent\_id is returned for each category. The first five records are skipped and the next five are returned since per\_page parameter is given as 5 and page parameter is specified as 2. And by default resources are ordered by category\_id.

Conclusion: Result as expected.

Test Case 3:

GET http://localhost/online-store/categories?parent\_id=-1&sort=-name

Output:

Those records are returned which have parent id=-1 and displayed in the descending order of their names.

Conclusion: Result as expected.

Test Case 4:

GET http://localhost/online-store/categoriefs

Output:

STATUS 400

Content-Type: application/json

{

"error": “Bad Request”

}

Conclusion: Result as expected since ‘categories’ is misspelled.

### Listing a category

Test Case 1:

GET http://localhost/online-store/categories/1

Output:

STATUS 200

Content-Type: application/json

{

“category\_id”: “1”,

"parent\_id": "-1",

"name": "Electronics",

"description": "Sample Electronics Description "

}

Conclusion: Result as expected.

Test Case 2:

GET http://localhost/online-store/categories/1?fields=name,description

Output:

STATUS 200

Content-Type: application/json

{

"name": "Electronics",

"description": "Sample Electronics Description "

}

Conclusion: Result as expected since only the fields requested are returned.

Test Case 3:

GET http://localhost/online-store/categories/20

Output:

STATUS 404

Content-Type: application/json

{

"error": “resource not found”

}

Conclusion: Result as expected since the resource with category\_id = 20 does not exist

Test Case 4:

GET http://localhost/online-store/categoriefs/20

Output:

STATUS 400

Content-Type: application/json

{

"error": “Bad Request”

}

Conclusion: Result as expected since ‘categories’ is misspelled.

### Inserting a Category

POST method is only allowed on a collection. If it is attempted on a resource, it will always return “Method not allowed on a resource”.

Test Case 1:

POST http://localhost/online-store/categories

Content-Type: application/json

{

"parent\_id": "1",

"name": "Cameras",

"description": "Cameras Sample Description "

}

Output:

STATUS 201

Content-Type: application/json

{

“category\_id”: “11”,

"parent\_id": "1",

"name": "Cameras",

"description": "Cameras Sample Description "

}

Conclusion: Result as expected.

Test Case 2:

POST http://localhost/online-store/categories/11/

Content-Type: application/json

{

“category\_id”:11,

"parent\_id": "1",

"name": "Cameras",

"description": "Cameras Sample Description"

}

Output:

STATUS 400

Content-Type: application/json

{

"error": “Method not allowed on a resource”

}

Conclusion: Result as expected since the POST Method is not allowed on a Resource.

### Updating a Category

Update method is only allowed on a single resource. If it is attempted on a collection, it will always return “Method not allowed on a collection”.

Test Case 1:

PUT http://localhost/online-store/categories/10

Content-Type: application/json

{

“category\_id”:10,

"parent\_id": "3",

"name": "Kitchen Appliances",

"description": "Kitchen Appliances Sample Description Updated"

}

Output:

STATUS 200

Content-Type: application/json

{

“category\_id”:10,

"parent\_id": "3",

"name": "Kitchen Appliances",

"description": "Kitchen Appliances Sample Description Updated"

}

Conclusion: Result as expected. The resource gets updated.

Test Case 2:

PUT http://localhost/online-store/categories/20

Content-Type: application/json

{

“category\_id”:20,

"parent\_id": "3",

"name": "Kitchen Appliances",

"description": "Kitchen Appliances Sample Description Updated"

}

Output:

STATUS 404

Content-Type: application/json

{

"error": “Resource not found”

}

Conclusion: Result as expected since there is no category with category\_id = 20

Test Case 3:

PUT http://localhost/online-store/categories/10

Content-Type: application/json

{

“category\_id”:8,

"parent\_id": "3",

"name": "Kitchen Appliances",

"description": "Kitchen Appliances Sample Description Updated"

}

Output:

STATUS 400

Content-Type: application/json

{

"error": “Bad Request”

}

Conclusion: Result as expected since the category\_id in the URL(10) does not match the category\_id in the JSON Input(8)

Test Case 4:

PUT http://localhost/online-store/categories/

Content-Type: application/json

{

“category\_id”:8,

"parent\_id": "3",

"name": "Kitchen Appliances",

"description": "Kitchen Appliances Sample Description Updated"

}

Output:

STATUS 400

Content-Type: application/json

{

"error": “Method not allowed on a collection”

}

Conclusion: Result as expected since the PUT Method is not allowed on a collection.

### Deleting a Category

Delete method is only allowed on a single resource. If it is attempted on a collection, it will always return “Method not allowed on a collection”.

Test Case 1:

DELETE http://localhost/online-store/categories/10

Output:

STATUS 200

Content-Type: application/json

{

"deleted": true

}

Conclusion: Result as expected. The resource gets deleted

Test Case 2:

DELETE http://localhost/online-store/categories/20

Output:

STATUS 404

Content-Type: application/json

{

"error": “Resource not found”

}

Conclusion: Result as expected since there is no category with category\_id = 20

Test Case 3:

DELETE http://localhost/online-store/categories/

Output:

STATUS 404

Content-Type: application/json

{

"error": “Method not allowed on a collection”

}

Conclusion: Result as expected since the DELETE method is not allowed on a collection.