

Testing eCommerce Using Postman

Step 1: Creating a new user

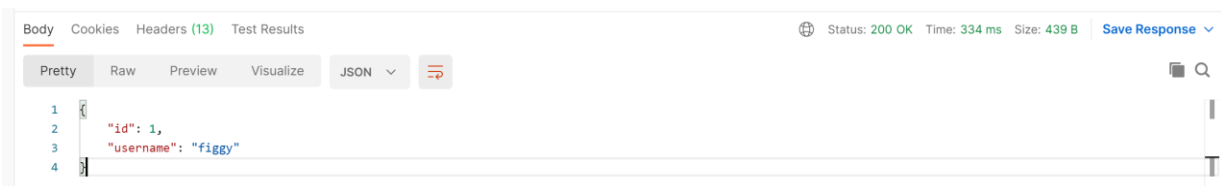
Method: POST

URL: <http://localhost:8080/api/user/create>

BODY:

```
{
  "username" : "figgy",
  "password" : "goodpassword",
  "confirmPassword" : "goodpassword"
}
```

RESPONSE:



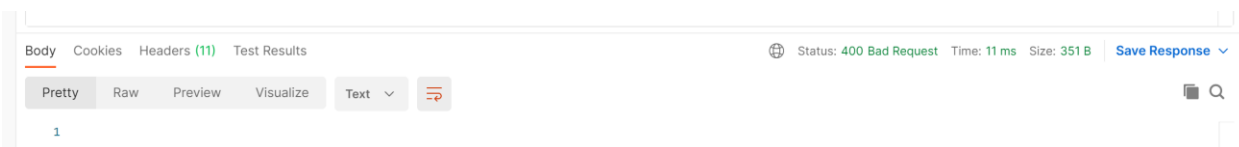
Step 2: Try to create the same user (different password)

Method: POST

URL: <http://localhost:8080/api/user/create>

BODY:

```
{
  "username" : "figgy",
  "password" : "interestingpassword",
  "confirmPassword" : "interestingpassword"
}
```



Step 3: Create a different user with short password

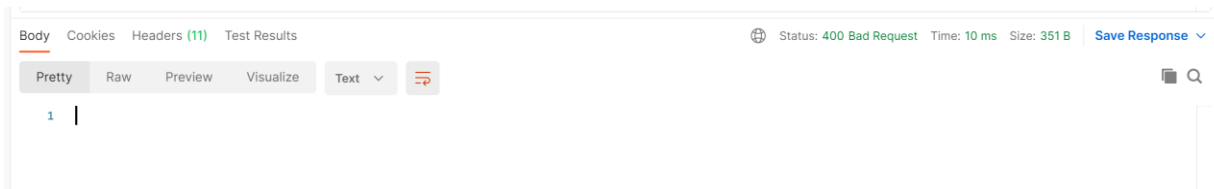
Method: POST

URL: http://localhost:8080/api/user/create

Body:

```
{
  "username" : "felix",
  "password" : "one",
  "confirmPassword" : "one"
}
```

Response:



Step 4: Create user but password and confirm password don't match

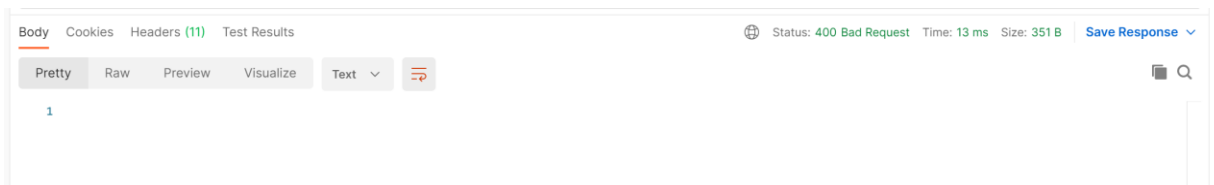
Method: POST

URL: http://localhost:8080/api/user/create

Body:

```
{
  "username" : "felix",
  "password" : "onetwothree",
  "confirmPassword" : "onetwothreefour"
}
```

Response:



Step 5: Login user created

Method: POST

URL: http://localhost:8080/login

Body:

```
{
  "username" : "figgy",
  "password" : "goodpassword"
}
```

Response:

Body		Cookies	Headers (13)	Test Results	Status: 200 OK Time: 209 ms Size: 562 B Save Response	
KEY				VALUE		
Vary				Origin		
Vary				Access-Control-Request-Method		
Vary				Access-Control-Request-Headers		
Authorization				Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJmaWdneSIsImV4cCI6MTYxNzU3Mzk0Nn0PS7_gg6Oq4BZp1XADso5kg50aHSoRFvUcNiixMUa-GPhfHITeMWRhp0ubD-iBznG7vm2DNJ7gmYoD7oOC1XQA		
X-Content-Type-Options				nosniff		
X-XSS-Protection				1; mode=block		
Cache-Control				no-cache, no-store, max-age=0, must-revalidate		
Pragma				no-cache		
Expires				0		

JWT: Bearer

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJmaWdneSIsImV4cCI6MTYxNzU3Mzk0Nn0PS7_gg6Oq4BZp1XADso5kg50aHSoRFvUcNiixMUa-GPhfHITeMWRhp0ubD-iBznG7vm2DNJ7gmYoD7oOC1XQA

Step 6: Get this user by id (1) (to JWT used)

Method: GET

URL: http://localhost:8080/api/user/id/1

Body: None

Response:

Body

Cookies

Headers (13)

Test Results

Status: 403 Forbidden

Time: 18 ms

Size: 534 B

Save Response

Pretty

Raw

Preview

Visualize

</

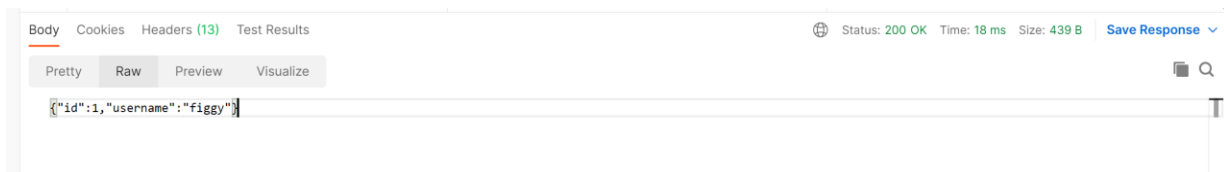
Step 7: Get user 1 by id with JWT

Method: GET

URL: http://localhost:8080/api/user/id/1

Body: None

Response:



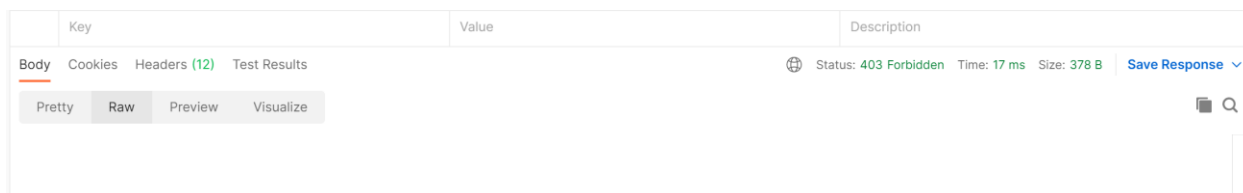
Step 8: Get User Id for non-existing Id use JWT for user 1

Method: GET

URL: `http://localhost:8080/api/user/id/100`

Body: None

Response:



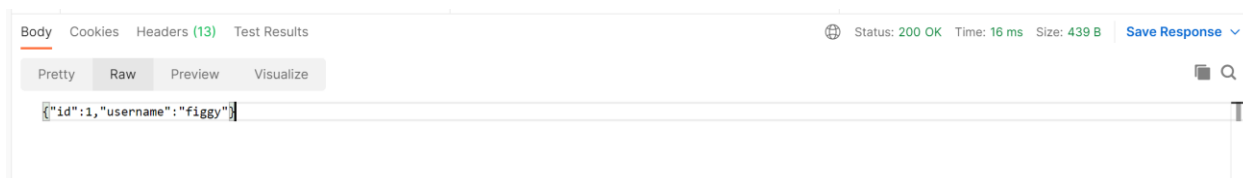
Step 9: Get User by username (figgy) using JWT

Method: GET

URL: `http://localhost:8080/api/user/figgy`

Body: None

Response:



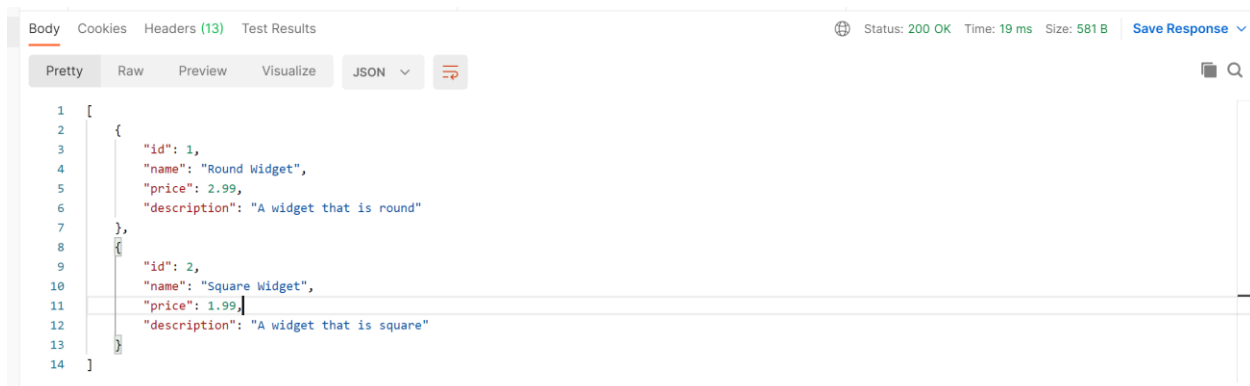
Step 10: Get All items + JWT

Method: GET

URL: `http://localhost:8080/api/item`

Body: None

Response:



```
1 [
2   {
3     "id": 1,
4     "name": "Round Widget",
5     "price": 2.99,
6     "description": "A widget that is round"
7   },
8   {
9     "id": 2,
10    "name": "Square Widget",
11    "price": 1.99,
12    "description": "A widget that is square"
13  }
14 ]
```

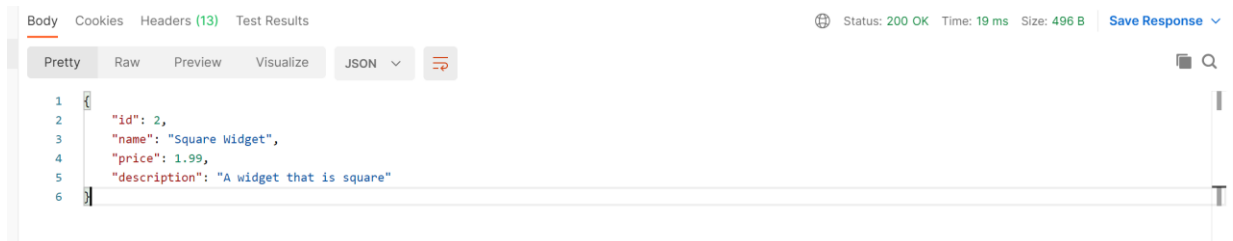
Step 11: Get Item by Id

Method: GET

URL: `http://localhost:8080/api/item/2`

Body: None

Response:



```
1 {
2   "id": 2,
3   "name": "Square Widget",
4   "price": 1.99,
5   "description": "A widget that is square"
6 }
```

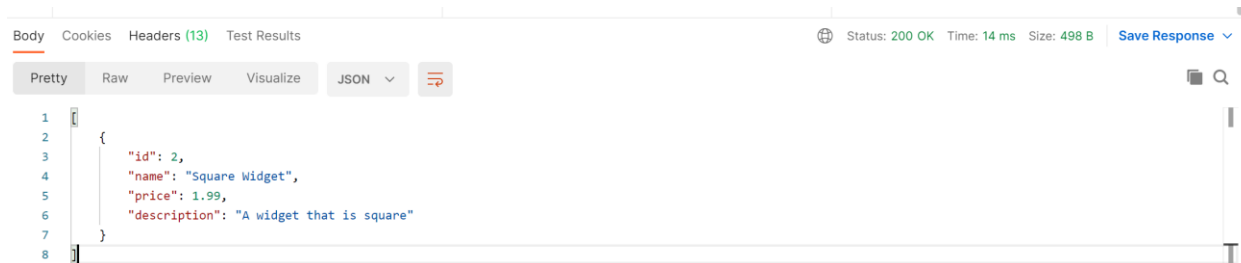
Step 12: Get Item By Name

Method: GET

URL: `http://localhost:8080/api/item/name/Square Widget`

Body: None

Response:



```
1 {
2   "id": 2,
3   "name": "Square Widget",
4   "price": 1.99,
5   "description": "A widget that is square"
6 }
```

Step 13: Add 2 Round Widgets to the user cart

Method: POST

URL: http://localhost:8080/api/cart/addToCart

Body:

```
{
  "username" : "figgy",
  "itemId" : 1,
  "quantity" : 2
}
```

Response:

Pretty Raw Preview Visualize JSON ↕

```
1  {
2    "id": 1,
3    "items": [
4      {
5        "id": 1,
6        "name": "Round Widget",
7        "price": 2.99,
8        "description": "A widget that is round"
9      },
10     {
11       "id": 1,
12       "name": "Round Widget",
13       "price": 2.99,
14       "description": "A widget that is round"
15     },
16   ]
17 }
```

Step 14: Add 3 Square Widgets to the car

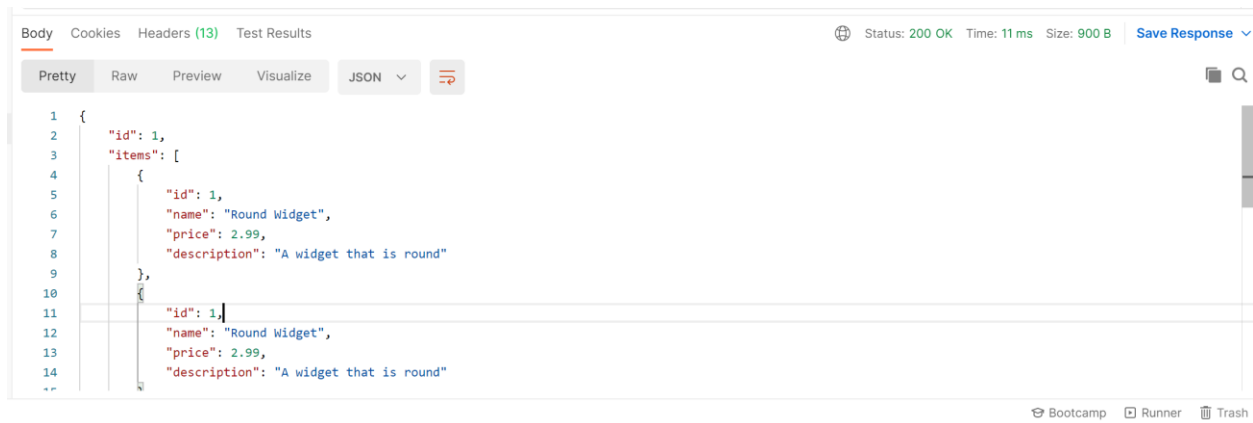
Method: POST

URL: http://localhost:8080/api/cart/addToCart

Body:

```
{
  "username" : "figgy",
  "itemId" : 2,
  "quantity" : 3
}
```

Response:



Step 15: Remove one Round Widget from the cart

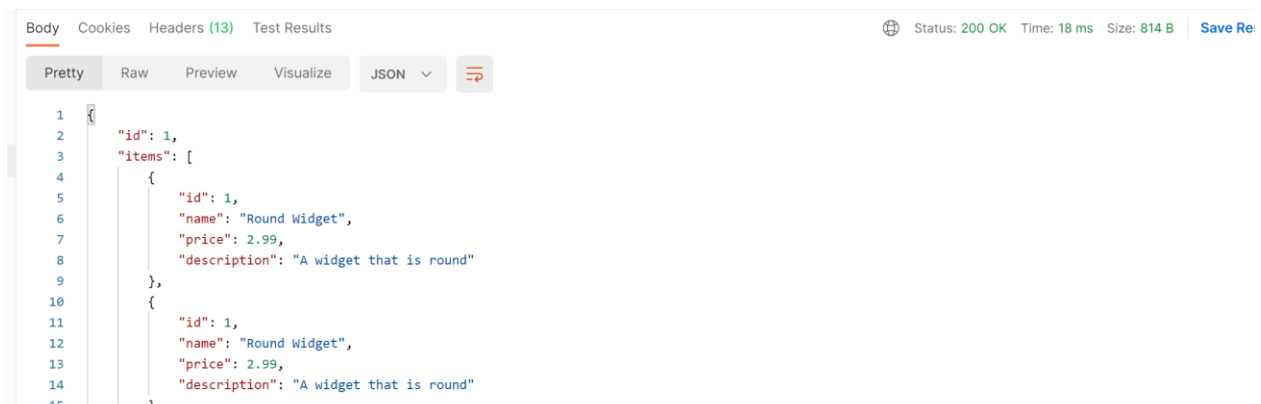
Method: POST

URL: <http://localhost:8080/api/cart/removeFromCart>

Body:

```
{
  "username" : "figgy",
  "itemId" : 2,
  "quantity" : 1
}
```

Response:



Only 2 Round Widgets remaining in my list

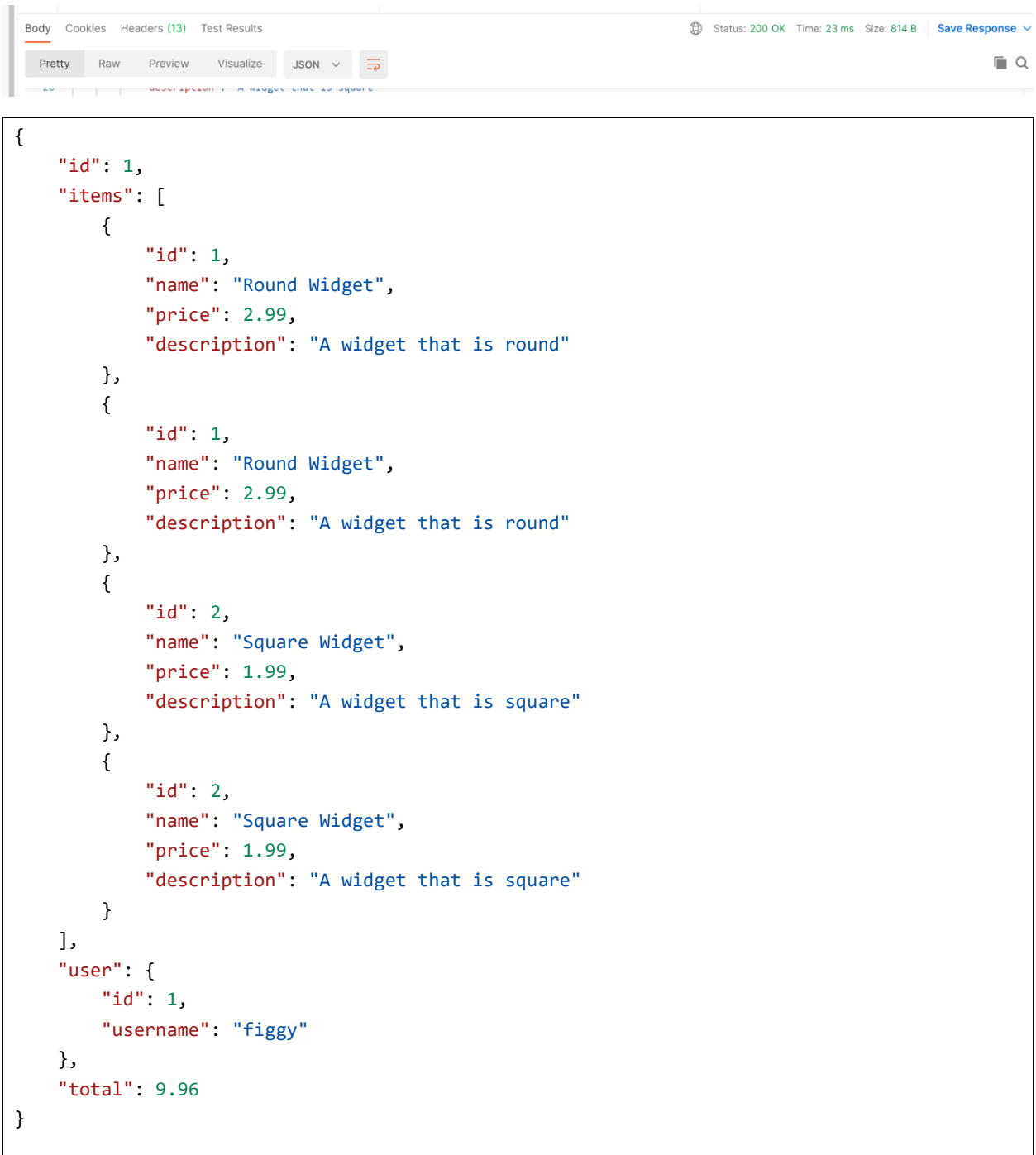
Step 16: Submit Order for figgy

Method: POST

URL: http://localhost:8080/api/order/submit/figgy

Body: None

Response:



The screenshot shows a web browser interface with a tab labeled 'Body'. The status bar at the top indicates 'Status: 200 OK', 'Time: 23 ms', and 'Size: 814 B'. Below the status bar, there are tabs for 'Pretty', 'Raw', 'Preview', 'Visualize', and 'JSON'. The 'JSON' tab is selected, and the response body is displayed as a JSON object. The JSON object contains an order summary for user 'figgy'.

```
{
  "id": 1,
  "items": [
    {
      "id": 1,
      "name": "Round Widget",
      "price": 2.99,
      "description": "A widget that is round"
    },
    {
      "id": 1,
      "name": "Round Widget",
      "price": 2.99,
      "description": "A widget that is round"
    },
    {
      "id": 2,
      "name": "Square Widget",
      "price": 1.99,
      "description": "A widget that is square"
    },
    {
      "id": 2,
      "name": "Square Widget",
      "price": 1.99,
      "description": "A widget that is square"
    }
  ],
  "user": {
    "id": 1,
    "username": "figgy"
  },
  "total": 9.96
}
```


Step 17: Get all orders for user

Method: GET

URL: <http://localhost:8080/api/order/history/figgy>

Body: None

Response:



```
[
  {
    "id": 1,
    "items": [
      {
        "id": 1,
        "name": "Round Widget",
        "price": 2.99,
        "description": "A widget that is round"
      },
      {
        "id": 1,
        "name": "Round Widget",
        "price": 2.99,
        "description": "A widget that is round"
      },
      {
        "id": 2,
        "name": "Square Widget",
        "price": 1.99,
        "description": "A widget that is square"
      },
      {
        "id": 2,
        "name": "Square Widget",
        "price": 1.99,
        "description": "A widget that is square"
      }
    ],
    "user": {
      "id": 1,
      "username": "figgy"
    },
    "total": 9.96
  }
]
```

--