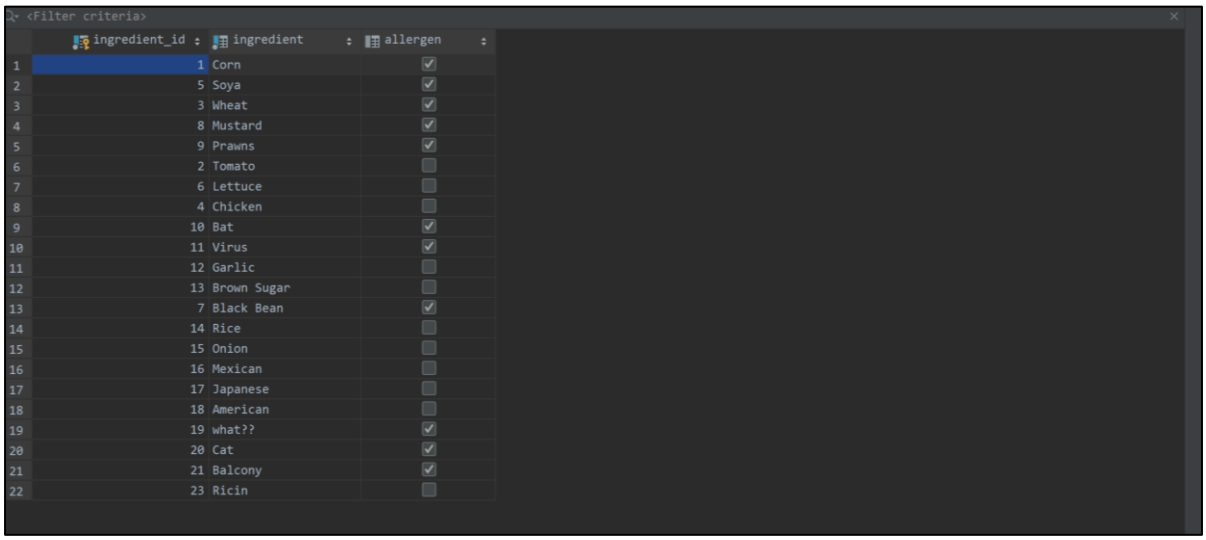
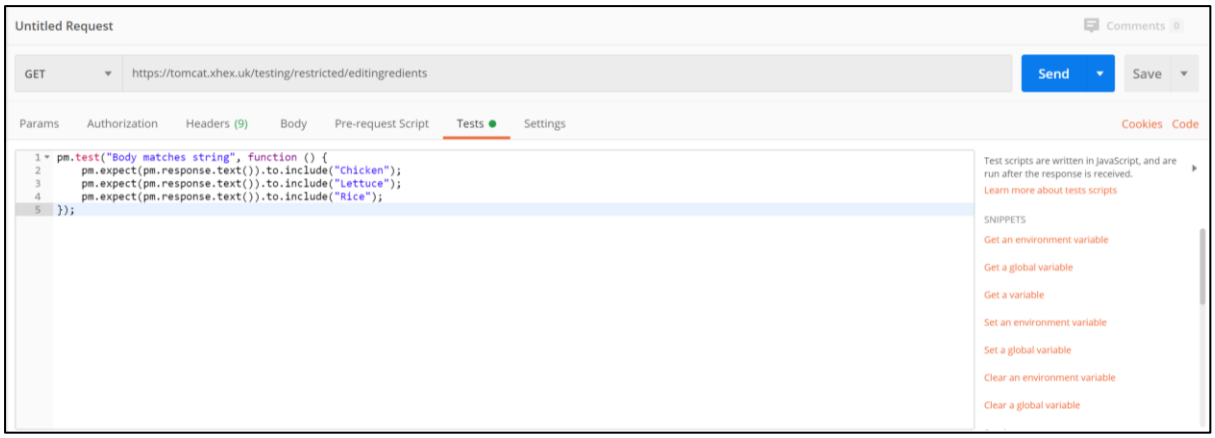
Author: Anas Choudhury

## TEST 4 (Using Postman)

### Test – In this test report I will be testing the 'EditIngredients' endpoint by manipulating its GET and POST method

For the first test I will be testing if the endpoint allows the user to obtain all the ingredients that are stored on the database. This can help them see what ingredients can be added and what should be added to what item on the menu. Also, test whether any other information is showed such as if it is an allergen or not. As this particular test is very simple, I will also be using the testing feature in Postman. Here is the current database, showing the ingredients table.



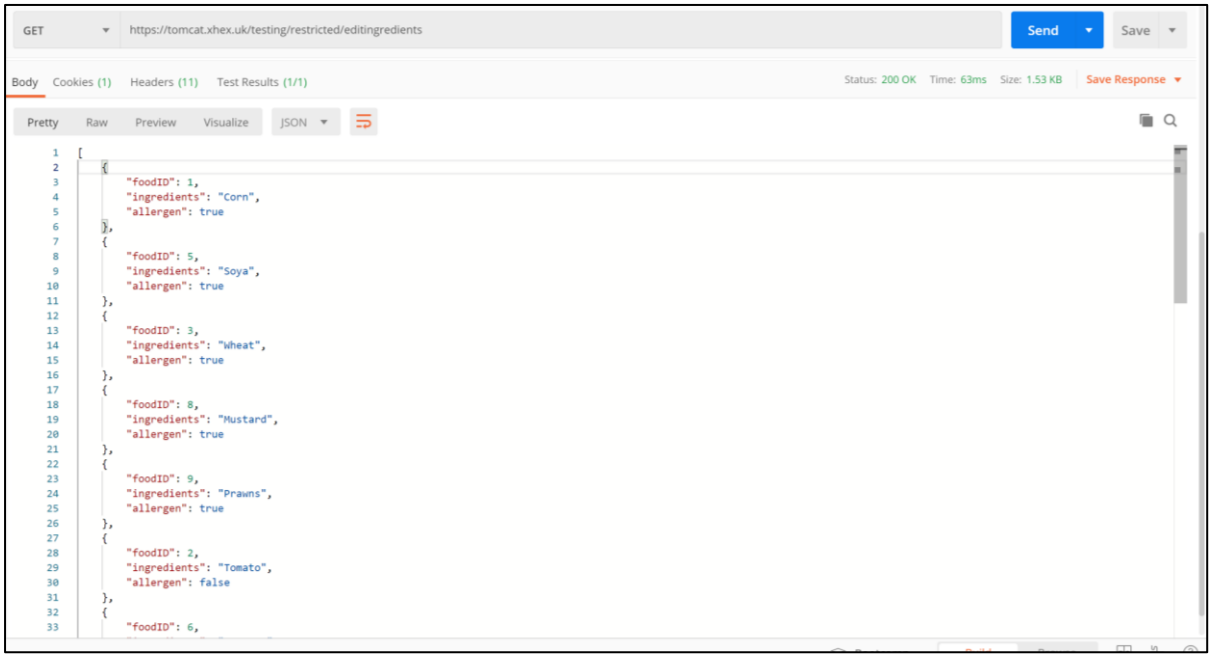

As you can see from the setup on Postman, I have used the GET request type but didn't use any params as they weren't needed as I am simply getting all the ingredients. I have written test scripts in the tests tab in Postman. As you can see, I am testing whether the response body contains the three strings which are chicken, lettuce and rice.

### Expected Result

From this test I expect an array of ingredients which are stored on the database and potentially a Boolean value if it is an allergen or not. Also, I expect the test script to

pass as these ingredients are shown on the menu screen on the application. Therefore, this test should go as expected.

**Result**





Here is the result from the tests. The first image shows that all the ingredients from the database were returned in the response body as expected. For each ingredient it also included its food id and Boolean value of the allergen status. The second screenshot shows confirms that the test scripts had passed which means the ingredients which I stated were in fact in the ingredients list in the database.

**Test 2**

For the next test I will be testing the same point whether it can allow me to create an ingredient and add it to the database. I will also use a test script to prove whether the ingredient is found in the response body once I get all the ingredients after it has been stored. I will be using the POST request type as I am sending data to the application rather than requesting to get information.

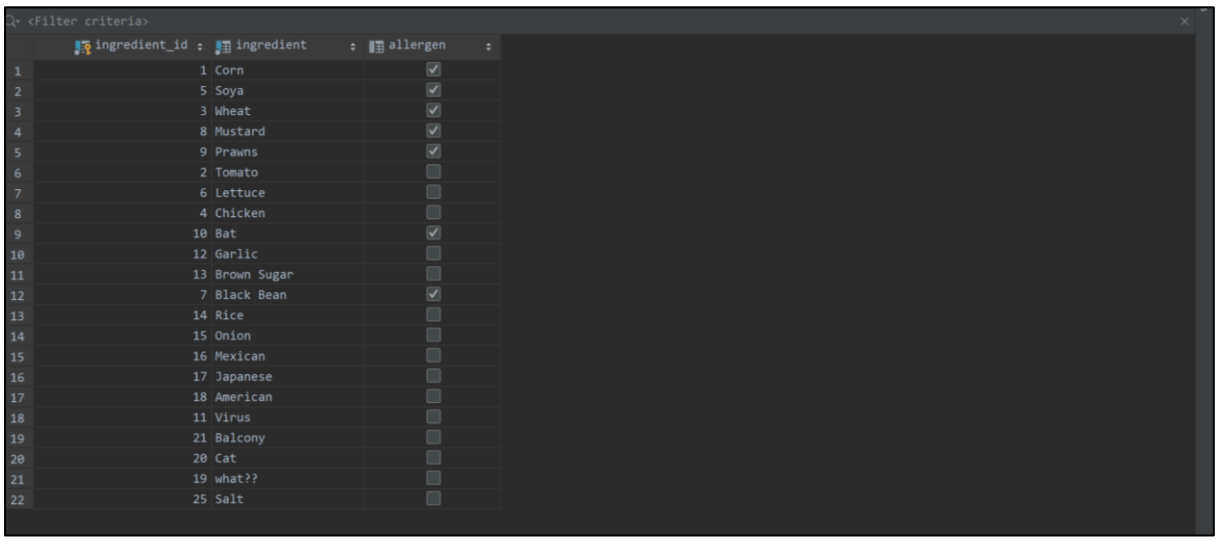
Author: Anas Choudhury

| | ingredient_id | ingredient | allergen |
|---|---|---|---|
| 1 | 1 | Corn | ☑ |
| 2 | 5 | Soya | ☑ |
| 3 | 3 | Wheat | ☑ |
| 4 | 8 | Mustard | ☑ |
| 5 | 9 | Prawns | ☑ |
| 6 | 2 | Tomato | ☐ |
| 7 | 6 | Lettuce | ☐ |
| 8 | 4 | Chicken | ☐ |
| 9 | 10 | Bat | ☑ |
| 10 | 12 | Garlic | ☐ |
| 11 | 13 | Brown Sugar | ☐ |
| 12 | 7 | Black Bean | ☑ |
| 13 | 14 | Rice | ☐ |
| 14 | 15 | Onion | ☐ |
| 15 | 16 | Mexican | ☐ |
| 16 | 17 | Japanese | ☐ |
| 17 | 18 | American | ☐ |
| 18 | 11 | Virus | ☐ |
| 19 | 21 | Balcony | ☐ |
| 20 | 20 | Cat | ☐ |
| 21 | 19 | what?? | ☐ |


## Expected Result

From this test, I expect the database to add salt as a new row at the bottom and state the allergen as false. I will also test if this happened by writing a script test to see if the total ingredients list contains the string ‘Salt’. This will prove that the test worked, and that the new ingredient has been added

## Result



As you can see from the bottom row in the updated database, the new ingredient has been added as expected including whether its an allergen or not. This can be proven as shown below. Using the simple java script test below allows me to search for a string from the body of the response. Also, the test has passed as it is labelled as 1/1 meaning the string was found. This concludes this endpoint is functioning as expected and planned.

```
pm.test("Body matches string", function () {
    pm.expect(pm.response.text()).to.include("Salt");
});
```