**Project: Recipes**

# **1. Introduction**:

Recipe Project is developed for the User to manage recipes. User has privilege to add the recipe, delete the recipe, get the recipe, update the recipe and fetch the recipe based on filters.

# **2. Requirements:**

Create a standalone java application which allows users to manage their favourite recipes. It should allow adding, updating, removing and fetching recipes. Additionally users should be able to filter available recipes based on one or more of the following criteria:

1. Whether or not the dish is vegetarian

2. The number of servings

3. Specific ingredients (either include or exclude)

4. Text search within the instructions.

For example, the API should be able to handle the following search requests:

• All vegetarian recipes

• Recipes that can serve 4 persons and have “potatoes” as an ingredient

• Recipes without “salmon” as an ingredient that has “oven” in the instructions.

# **3. Technical Specifications**

This project is developed using SpringBoot REST API, JDBCTemplates and MySQL server.

Below table give detail information about technologies.

|  |  |
| --- | --- |
| **Technologies** | **Comments** |
| Java 1.8 |  |
| SpringBoot 2.7.0 |  |
| Maven |  |
| SpringBoot JDBC Templates |  |
| Tomcat Server | SpringBoot embedded Tomcat server |
| Log4j2 | for logging |

API’s can be tested using Testing Tools like Postman, SOAP UI.

## **3.1. REST API Specifications**

Recipes project was implemented using SpringBoot REST API endpoints and MySql database. All endpoints allow JSON request and provide JSON Response.

|  |  |  |
| --- | --- | --- |
| **Method** | **URL** | **Action** |
| POST | <http://localhost:8888/recipes/addrecipe> | Add recipe |
| PUT | [http://localhost:8888/recipes/updaterecipe/{recipe\_id}](http://localhost:8888/recipes/updaterecipe/%7brecipe_id%7d) | Update recipe |
| DELETE | [http://localhost:8888/recipes/removerecipe/{recipe\_id}](http://localhost:8888/recipes/removerecipe/%7brecipe_id%7d) | Remove Recipe |
| GET | [http://localhost:8888/recipes/getrecipe/{recipe\_id}](http://localhost:8888/recipes/getrecipe/%7brecipe_id%7d) | Get recipe |
| GET | <http://localhost:8888/recipes/filterrecipe?recipe_type=Vegtarian> | Filter Recipe by recipe type, number of servings, ingredients and instructions |

**Base URL** (differ according to each environment like DEV, UAT, PROD) : <http://localhost:8888/> (DEV)

Request and Response parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **constraints** | **Comments** |
| recipe\_id | Recipe Id | Long | Not null | Auto generation |
| recipe\_name | Recipe Name | String | Not null |  |
| Recipe\_type | Recipe Type | String | Not null |  |
| num\_of\_servings | Number of Servings | Int | Not null |  |
| ingredients | ingredients | String | Not null |  |
| instructions | instructions | String | Not null |  |

### **3.1.1. AddRecipe API Sevice:**

This API allows user to add new recipe to database. User should add recipe by providing all the request details.

User should construct JSON request using below details.

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Comments** |
| recipe\_name | Recipe Name | String | Mandatory. User must enter |
| Recipe\_type | Recipe Type | String | Mandatory. User must enter |
| num\_of\_servings | Number of Servings | int | Mandatory. User must enter |
| ingredients | ingredients | String | Mandatory. User must enter |
| instructions | instructions | String | Mandatory. User must enter |

**Url:** <http://localhost:8888/recipes/addrecipe>

**Method:** PUT

**Request:**

{

"recipe\_name":"cake",

"recipe\_type":"vegetarian",

"num\_of\_servings":1,

"ingredients":"wheat flour,sugar,salt",

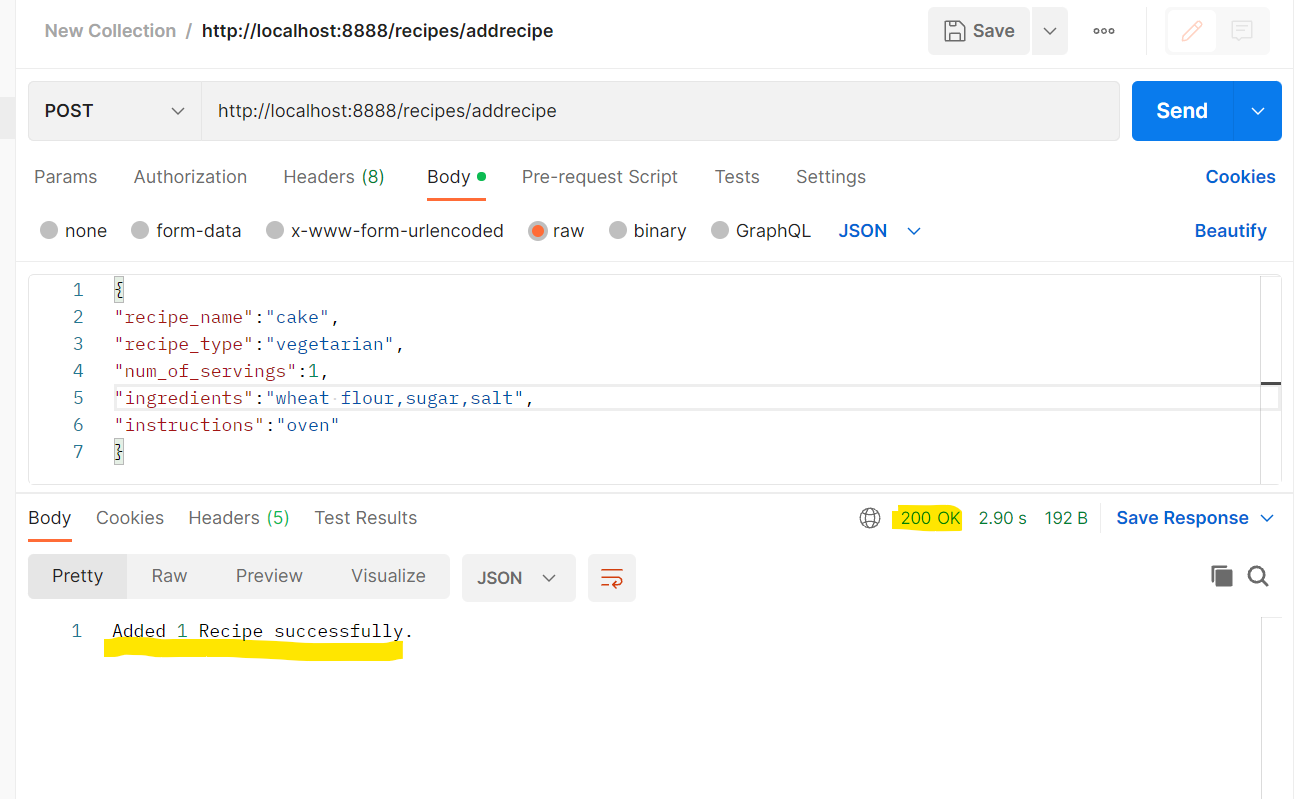
"instructions":"oven"

}

**Response:** Added 1 Recipe successfully.

**HTTP Status Code:** 200

**Sample screen:**

****

### **3.1.2. UpdateRecipe API Service:**

This API allows user to update recipe details based on recipe id. User must enter all the details along with recipe id as path parameter.

User should construct JSON request using below details and recipe\_id is path parameter.

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Comments** |
| recipe\_id | Recipe Id | Long | This path parameter. User must construct API endpoint using recipe\_id which he/she want to update. |
| recipe\_name | Recipe Name | String | Mandatory. User must enter |
| Recipe\_type | Recipe Type | String | Mandatory. User must enter |
| num\_of\_servings | Number of Servings | int | Mandatory. User must enter |
| ingredients | ingredients | String | Mandatory. User must enter |
| instructions | instructions | String | Mandatory. User must enter |

**Url:** [http://localhost:8888/recipes/updaterecipe/{recipe\_id}](http://localhost:8888/recipes/updaterecipe/%7brecipe_id%7d)

**Sample URL:** http://localhost:8888/recipes/updaterecipe/11

**Method:** PUT

**Request:**

{

    "recipe\_name": "cake",

    "recipe\_type": "vegetarian",

    "num\_of\_servings": 2,

    "ingredients": "wheat flour,sugar,salt,soda",

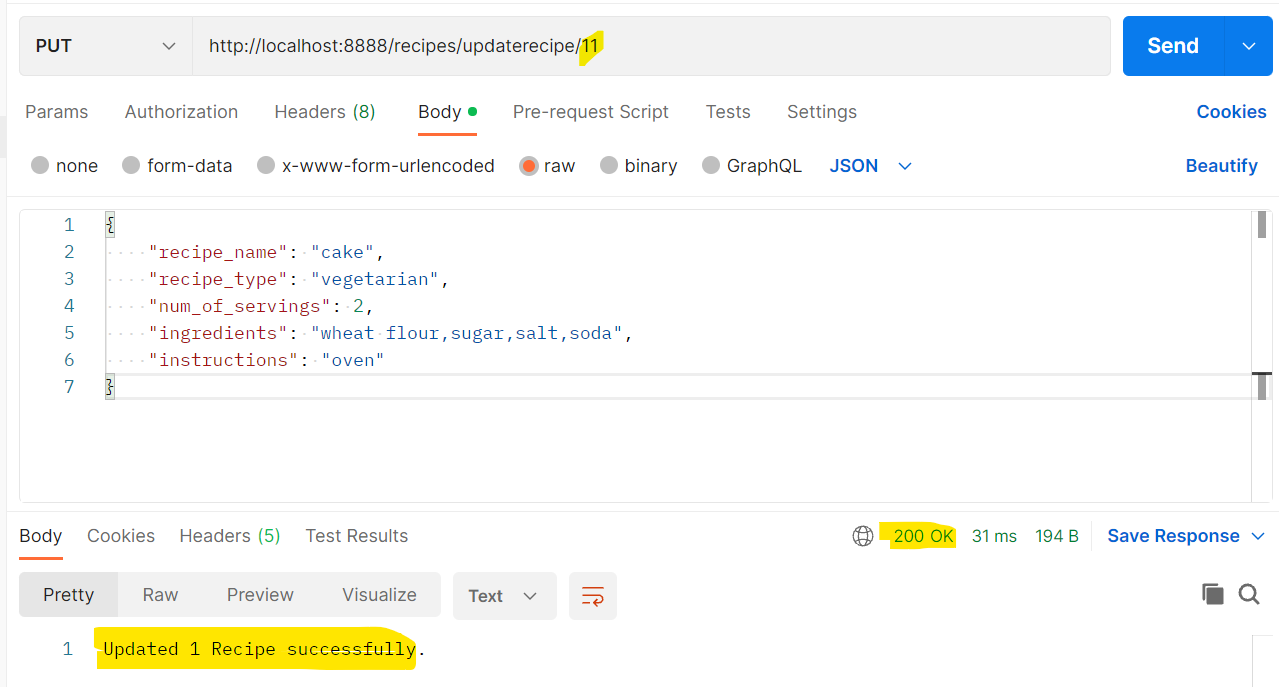
    "instructions": "oven"

}

**Response:** Updated 1 Recipe successfully.

**HTTP Status Code:** 200

**Sample screen:**

****

### **3.1.3. RemoveRecipe API Service:**

This API allows user to remove/delete the recipe from database based on recipe id. User must enter recipe id as path parameter.

User should construct JSON request using below details and recipe\_id is path parameter.

**Url:** [http://localhost:8888/recipes/removerecipe/{recipe\_id}](http://localhost:8888/recipes/removerecipe/%7brecipe_id%7d)

**Sample URL:** http://localhost:8888/recipes/removerecipe/12

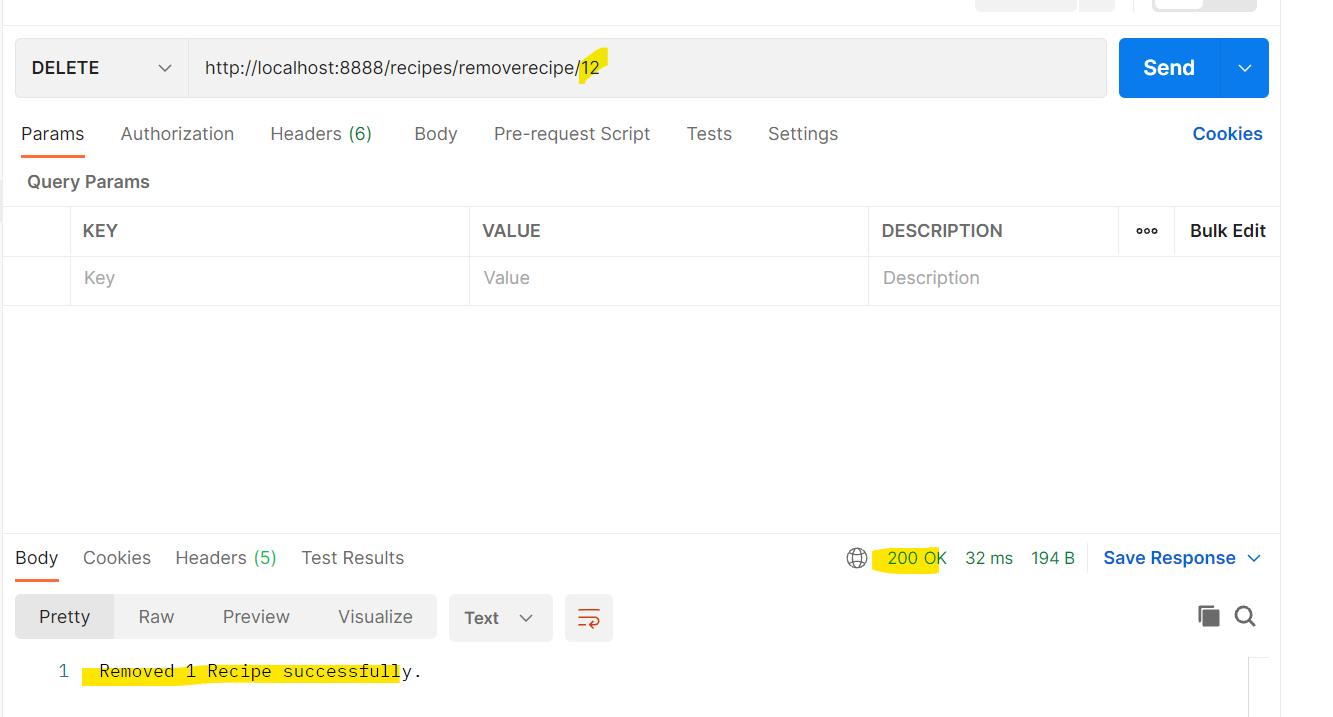
**Method:** DELETE

**Request:** Not Applicable

**Response:** Removed 1 Recipe successfully.

**HTTP Status Code:** 200

**Sample screen:**



### **3.1.4. GetRecipe API Service:**

This API allows user to get/fetch the recipe from database based on recipe id. User must enter **recipe id** as path parameter.

User should construct API request using below details and recipe\_id is path parameter.

**Url:** [http://localhost:8888/recipes/getrecipe/{recipe\_id}](http://localhost:8888/recipes/getrecipe/%7brecipe_id%7d)

**Sample URL:** http://localhost:8888/recipes/getrecipe/11

**Method:** GET

**Request:** Not Applicable

**Response:**

{

    "recipe\_id": 11,

    "recipe\_name": "cake",

    "recipe\_type": "vegetarian",

    "num\_of\_servings": 2,

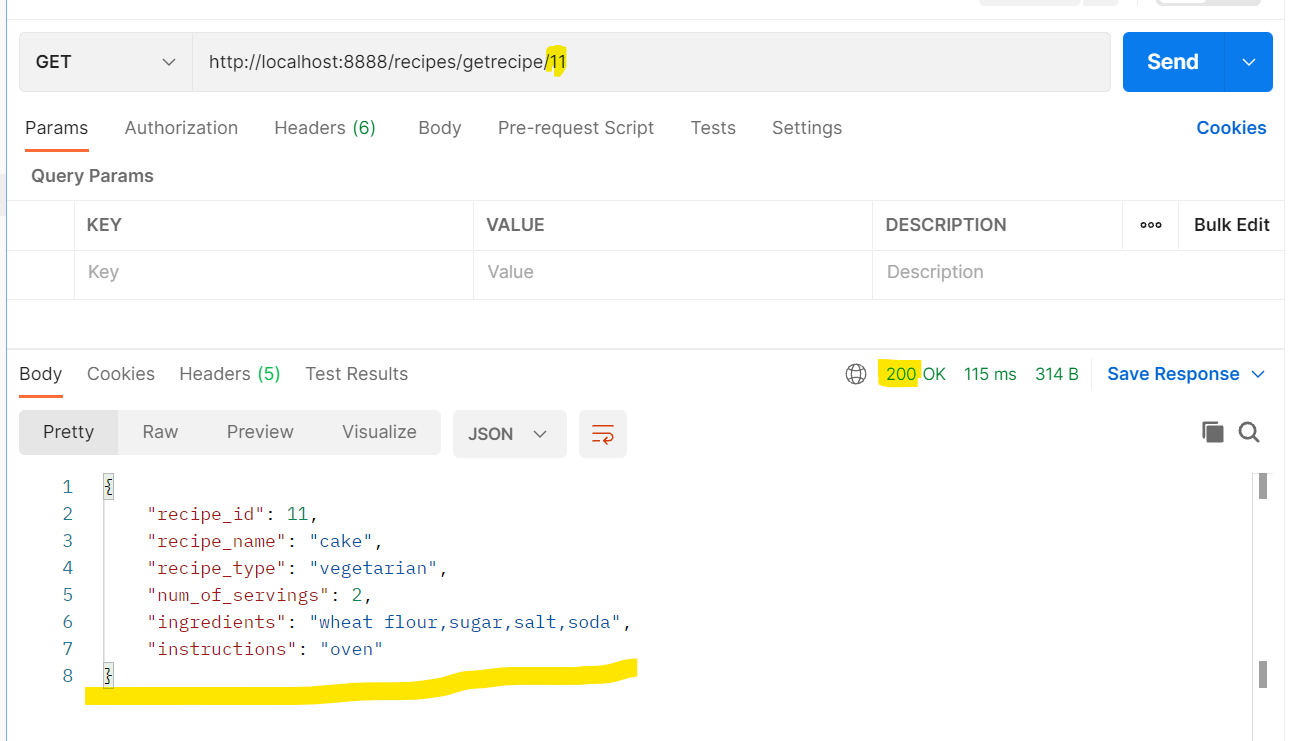
    "ingredients": "wheat flour,sugar,salt,soda",

    "instructions": "oven"

}

**HTTP Status Code:** 200

**Sample screen:**



### **3.1.5. FilterRecipe API Service:**

This API allows user to filter the recipe based on recipe type, number of servings, ingredients and instructions. User is allowed to apply one or more filters.

Filter is applied with AND condition when multiple filters applied. User should enter minimum one filter to get success response.

Filters like recipe\_type, num\_of\_servings, ingredients, instructions are Case Sensitive and Query Parameters.

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Comments** |
| Recipe\_type | Recipe Type | String | Optional. But minimum one Filter condition must be given. Query Parameter. |
| num\_of\_servings | Number of Servings | int | Optional. But minimum one Filter condition must be given. Query Parameter. |
| ingredients | ingredients | String | Optional. But minimum one Filter condition must be given. Query Parameter. |
| noingredients | no ingredient include in the recipe | String | Optional. But minimum one Filter condition must be given. Query Parameter. |
| instructions | instructions | String | Optional. But minimum one Filter condition must be given. Query Parameter. |

**Url:** <http://localhost:8888/recipes/filterrecipe>

**Sample URL 1:** <http://localhost:8888/recipes/filterrecipe?recipe_type=vegetarian>

**Sample URL 2:** <http://localhost:8888/recipes/filterrecipe?noingredients=salmon&instructions=oven>

**Method:** GET

**Request:** Construct with one or more Query Parameters recipe\_type, num\_of\_servings, ingredients, noingredients, instructions.

**Sample Response:**

[

    {

        "recipe\_id": 11,

        "recipe\_name": "cake",

        "recipe\_type": "vegetarian",

        "num\_of\_servings": 2,

        "ingredients": "wheat flour,sugar,salt,soda",

        "instructions": "oven"

    },

    {

        "recipe\_id": 15,

        "recipe\_name": "burger",

        "recipe\_type": "vegetarian",

        "num\_of\_servings": 1,

        "ingredients": "wheat flour,salt,potatoes,tamoto",

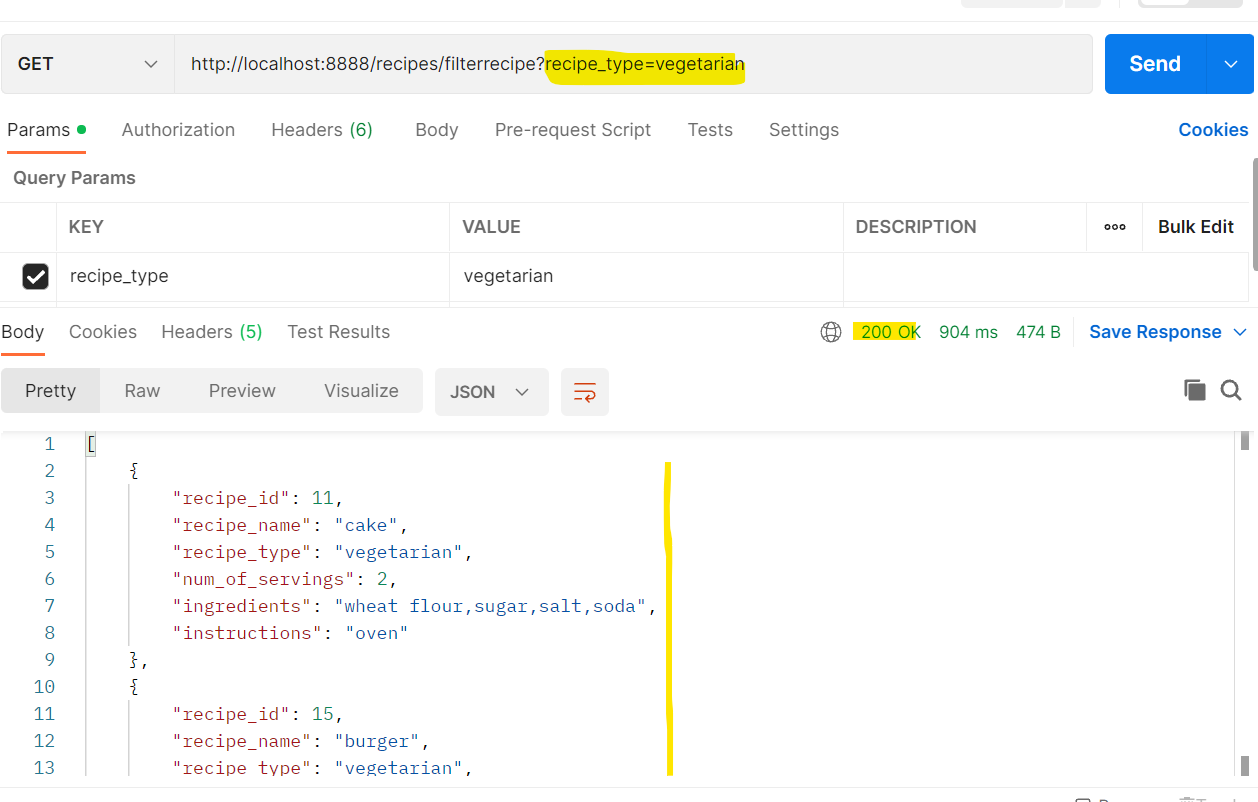
        "instructions": "oven"

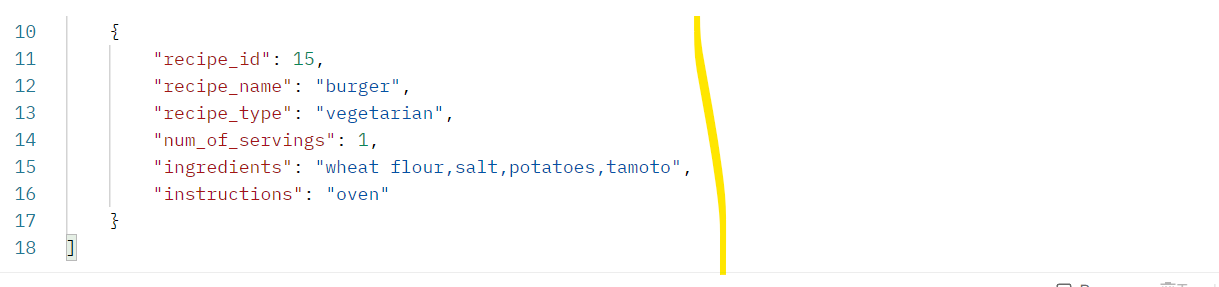
    }

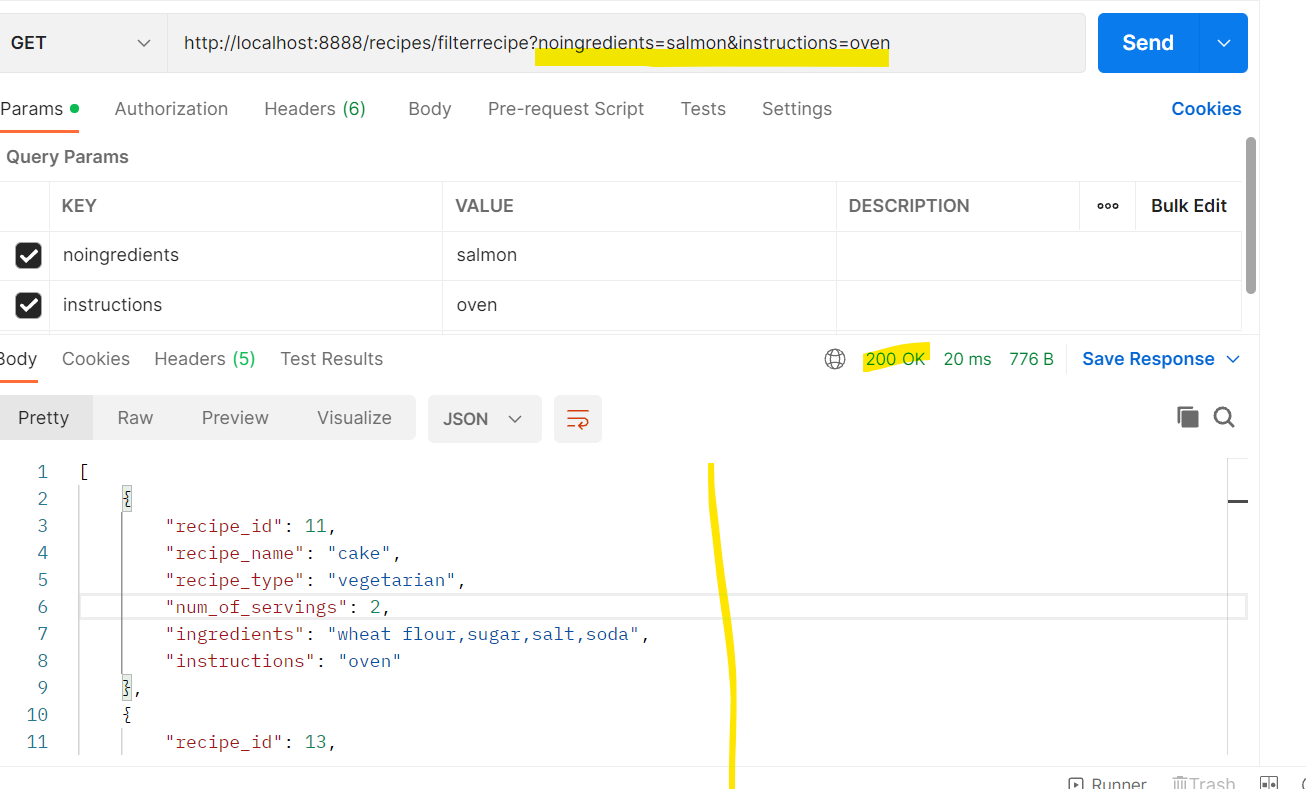
]

**HTTP Status Code:** 200

**Sample screen:**







### **3.2. Status Codes:**

REST API will give different HTTP status codes for success response and error response.

Below table gives brief information about different status codes.

|  |  |  |
| --- | --- | --- |
| Status Code | Description | Comments |
| 200 | OK/Successful | Response is success |
| 400 | Bad Request | Request is not correct |
| 404 | Not Found | Resource is not reachable/No Data Found |
| 500 | Internal Server Error | Error occurs internally in the server |