**Rice Bowl Design Document**

**Objective**

The purpose of this assignment is to come up with a design to make “Rice Bowls” based OOP principles with following constraints.

1. Add white or brown rice
2. Add mixed vegetables or skip.
3. Add a choice of meat – chicken or beef
4. Add a choice of sauce – spicy or sweet
5. Add sour cream or skip
6. Add guacamole or skip

**Design:**

Java Restful APIs are implemented using Spring Boot to create rice bowls. The rice bowl object is represented by the below class

**public** **class** RiceBowl {

**private** RiceType riceType;

**private** **boolean** vegetablesRequired;

**private** MeatType meatType;

**private** SauceType sauceType;

**private** **boolean** sourCreamRequired;

**private** **boolean** guacamoleRequired;

**public** RiceBowl() {

}

**public** RiceBowl(RiceType riceType, **boolean** vegetablesRequired, MeatType meatType, SauceType sauceType,

**boolean** sourCreamRequired, **boolean** guacamoleRequired) {

**super**();

**this**.riceType = riceType;

**this**.vegetablesRequired = vegetablesRequired;

**this**.meatType = meatType;

**this**.sauceType = sauceType;

**this**.sourCreamRequired = sourCreamRequired;

**this**.guacamoleRequired = guacamoleRequired;

}

**public** RiceType getRiceType() {

**return** riceType;

}

**public** **void** setRiceType(RiceType riceType) {

**this**.riceType = riceType;

}

**public** **boolean** isVegetablesRequired() {

**return** vegetablesRequired;

}

**public** **void** setVegetablesRequired(**boolean** vegetablesRequired) {

**this**.vegetablesRequired = vegetablesRequired;

}

**public** MeatType getMeatType() {

**return** meatType;

}

**public** **void** setMeatType(MeatType meatType) {

**this**.meatType = meatType;

}

**public** SauceType getSauceType() {

**return** sauceType;

}

**public** **void** setSauceType(SauceType sauceType) {

**this**.sauceType = sauceType;

}

**public** **boolean** isSourCreamRequired() {

**return** sourCreamRequired;

}

**public** **void** setSourCreamRequired(**boolean** sourCreamRequired) {

**this**.sourCreamRequired = sourCreamRequired;

}

**public** **boolean** isGuacamoleRequired() {

**return** guacamoleRequired;

}

**public** **void** setGuacamoleRequired(**boolean** guacamoleRequired) {

**this**.guacamoleRequired = guacamoleRequired;

}

}

Each of the user input is the property of the class RiceBowl. Enumeration classes are created to represent Type of Rice, Meat and Sauce. User inputs for type of rice, meat and sauce are accepted as Strings which are then converted to respective enums. Valid values for each of these are as below

1. Type of Rice: brown, white, none
2. Type of Meat: chicken, beef, none
3. Type of Sauce: spicy, sweet, none

Vegetables, Sour Cream and Guacamole are accepted from the user as a Boolean input. A Boolean true indicates user wants to add vegetable or sour cream or guacamole and false indicates that user chooses to skip it.

APIs implemented are

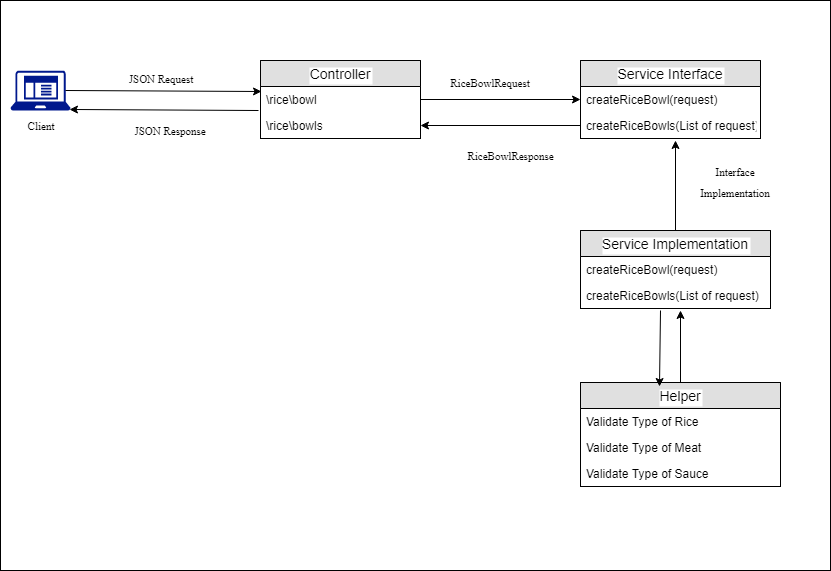
1. A POST API that creates a rice bowl. It takes input in the form of JSON which then, is Serialized to Java Object. The API returns the rice bowl that is created in the JSON format.

|  |
| --- |
| **Sample Input / Output:**  URL: http://localhost:8080/rice/bowl  Method: POST  Input:  {  "riceType":"brown",  "vegetablesRequired": true,  "meatType": "none",  "sauceType": "spicy",  "sourCreamRequired": false,  "guacamoleRequired": true    }  Output:  {  "message": "Success",  "status": "OK",  "rice": "brown",  "vegetables": true,  "meat": "none",  "sauce": "spicy",  "sourCream": false,  "guacamole": true  } |

1. A POST API that creates a list of rice bowls. The API takes a list of JSON where each JSON represents a rice bowl. The output is a list of rice bowls in JSON format.

|  |
| --- |
| **Sample Input / Output:**  URL: <http://localhost:8080/rice/bowls>  Method: POST  Input:  [{  "riceType":"white",  "vegetablesRequired": false,  "meatType":"none",  "sauceType":"spicy",  "sourCreamRequired": false,  "guacamoleRequired": true    },  {  "riceType":"brown",  "vegetablesRequired": true,  "meatType":"none",  "sauceType":"sweet",  "sourCreamRequired": false,  "guacamoleRequired": true    },  {  "riceType":"brown",  "vegetablesRequired": true,  "meatType":"none",  "sauceType":"sweets",  "sourCreamRequired": false,  "guacamoleRequired": true    }]  Output:  [  {  "message": "Success",  "status": "OK",  "rice": "white",  "vegetables": false,  "meat": "none",  "sauce": "spicy",  "sourCream": false,  "guacamole": true  },  {  "message": "Success",  "status": "OK",  "rice": "brown",  "vegetables": true,  "meat": "none",  "sauce": "sweet",  "sourCream": false,  "guacamole": true  },  {  "message": "Invalid Sauce Type",  "status": "BAD\_REQUEST",  "rice": null,  "vegetables": false,  "meat": null,  "sauce": null,  "sourCream": false,  "guacamole": false  }  ] |

**Workflow Diagram:**



There are 3 Java classes implemented to build each of the API

1. Controller – The rest APIs are exposed in this class.
2. Service – Interface that provides method signature and fields for implement the actual creation of rice bowls based on user input.
3. Helper – A helper class to validate and convert user inputs to suitable objects.

**Assumptions:**

1. Type of rice can be either brown, white or none. Any other values for type of rice would throw an exception.
2. Type of meat can be either chicken, beef or none. Any other values for type of meat would throw an exception.
3. Type of sauce can be spicy, sweet or none. Any other values for the type of sauce would throw an exception.
4. User can set only one value per request for type of rice, meat and sauce.
5. Either rice, vegetables or meat must be present. If all three inputs are none/false, throw an exception.

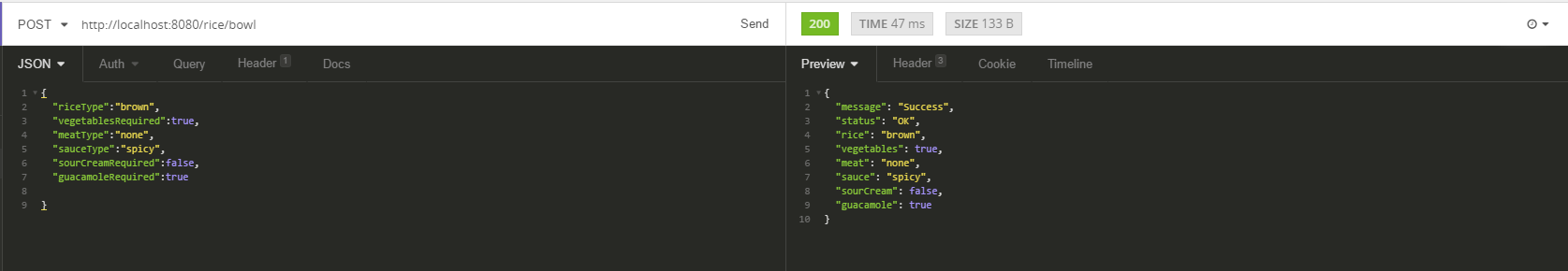
**Instructions to run on local environment:**

Execute the jar “ricebowl-0.1.0.jar” using the command below to host the application on Spring Boot’s embedded Tomcat server

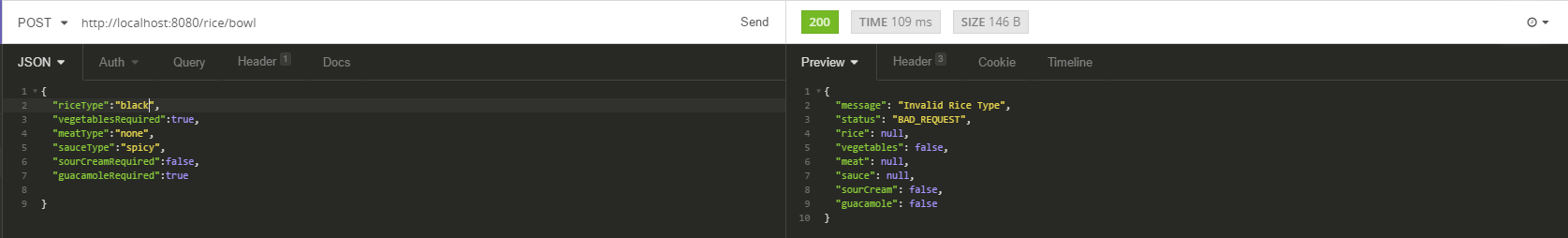
*java -jar ricebowl-0.1.0.jar*

API endpoint is <http://localhost:8080>

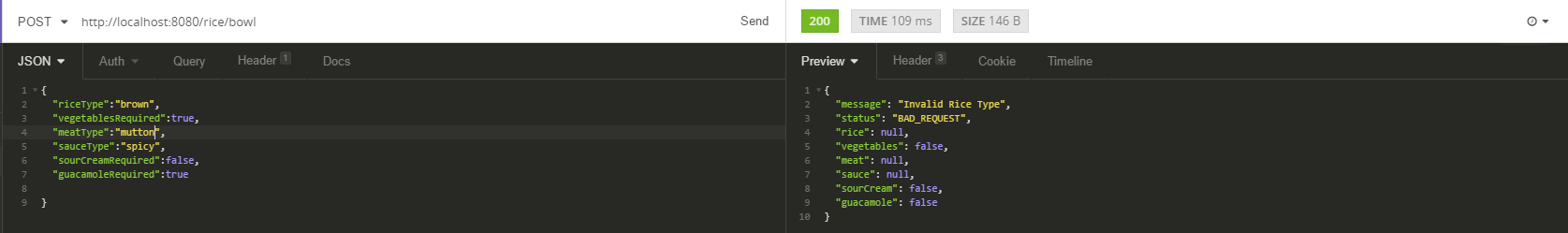
**Screenshots**



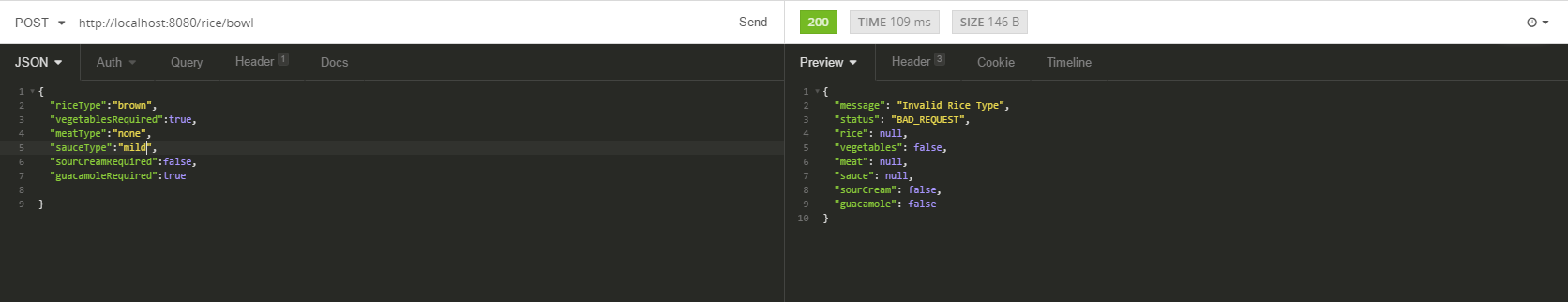
**Fig. 1 Valid Input and Output**



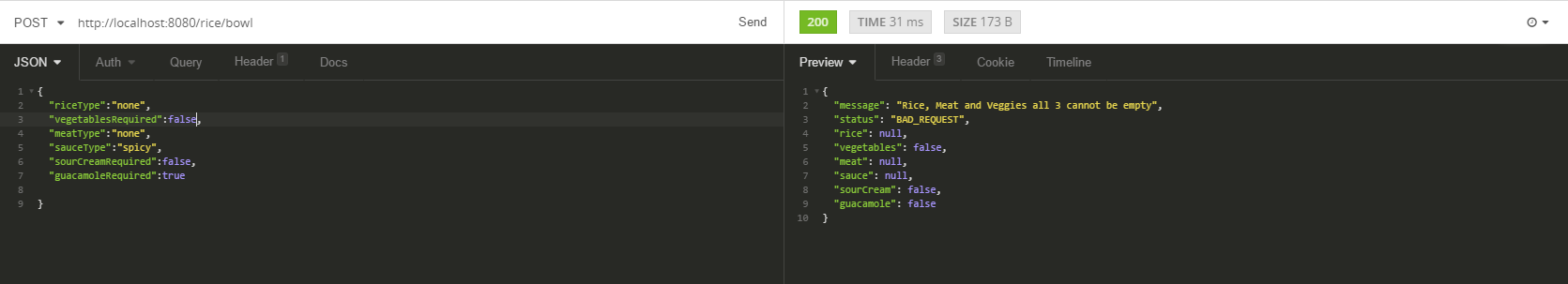
**Fig.2 Invalid Input for type of Rice**



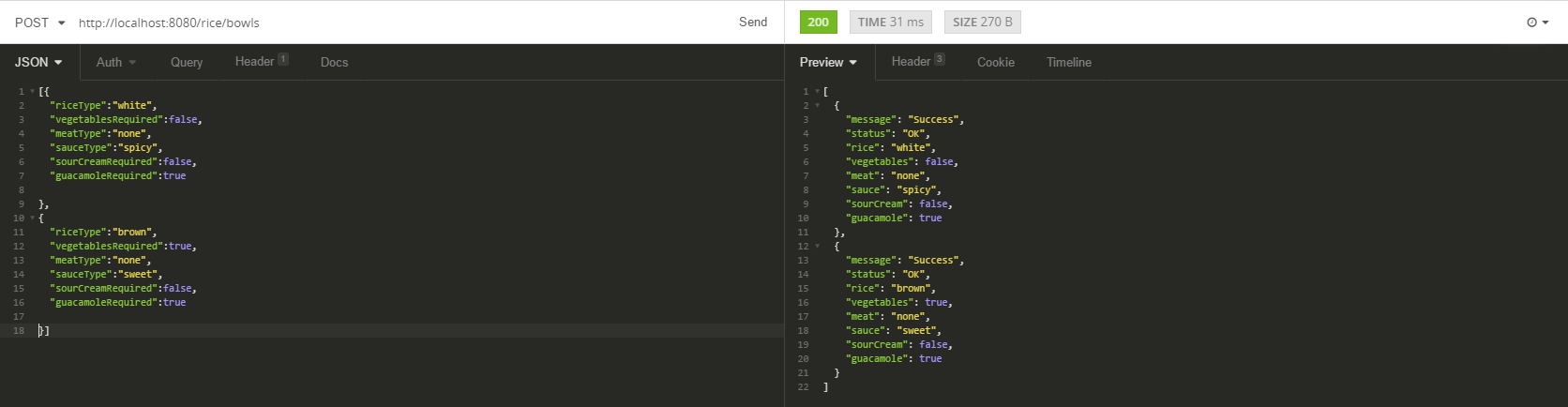
**Fig.3 Invalid Input for type of Meat**



**Fig.4 Invalid Input for Type of Sauce**



**Fig.5 Throws error when rice=none, meat=none and vegetables=false**



**Fig.6 Create List of Rice Bowls**