CEGEP VANIER COLLEGE
CENTRE FOR CONTINUING EDUCATION
Web Services
420-941-VA

Teacher: Samir Chebbine **Lab 5** **Oct 30, 2024**

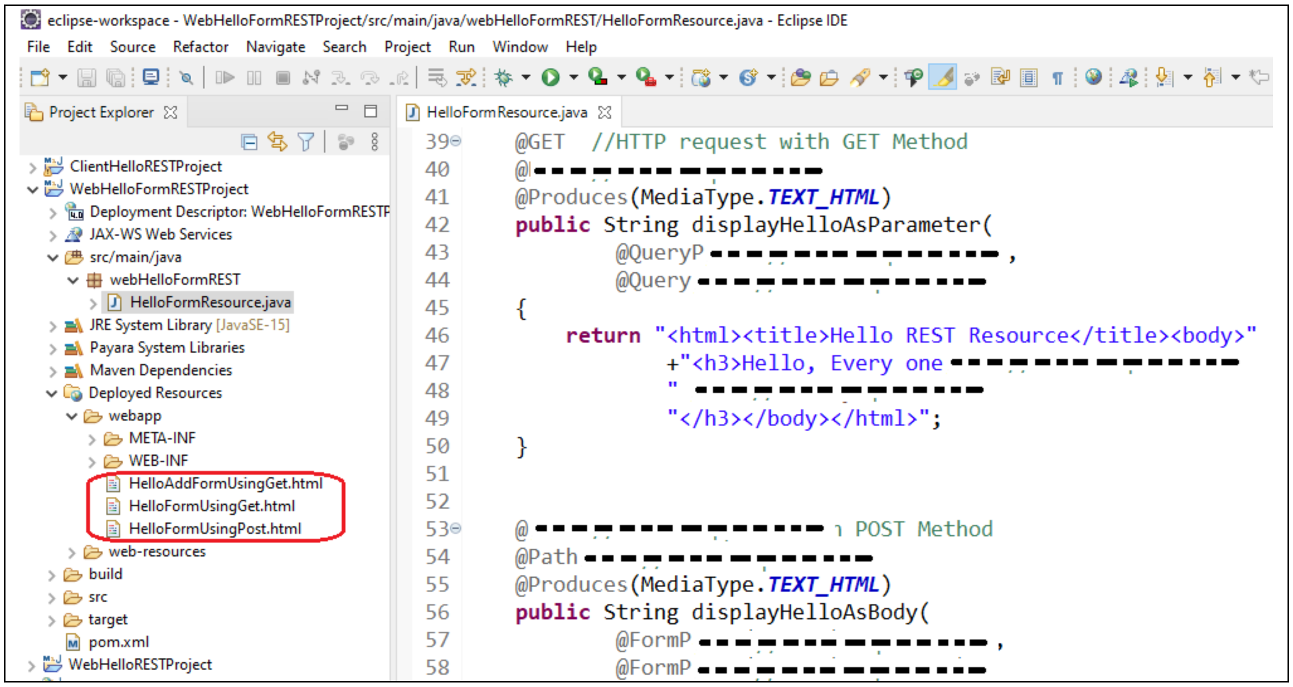
Lab 5: REST Web Services Using different HTTP Methods and HTML Form Processing

Complete all these following programs in class**.** All *missing coding statements* are presented during class time and in Presentation 5.

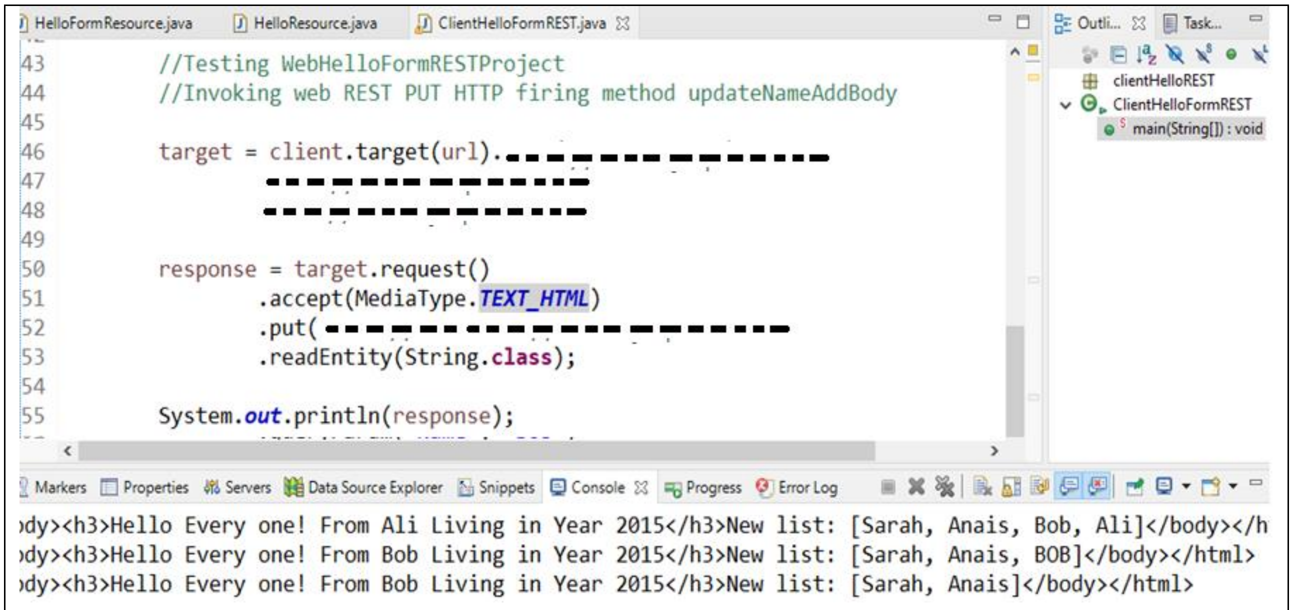
Create and Submit a Word file ***Lab5WebServicesYourName.doc*** which contains Answers of theory questions if any and output screenshots for every Java EE Project. Submit the Java projects too and submit the whole Lab 5 as compressed zip file

1. Using Parameters: Query String Parameters and Form Parameters

a) Create a new Dynamic Web project called **WebHelloFormRESTProject** and convert it into Maven Project.
b) Deploy **WebHelloFormRESTProject** within GalssFish Server.
c) Using Postman, display screenshots testing each HTTP request method included in my YouTube Video Lab 5 related to HTTP GET, POST, PUT and DELETE as shown in Figure 1.
d) Using HTML Forms, display screenshots testing each HTML form included in my YouTube Video Lab 5.
e) Create a Maven project Client Application called **ClientHelloFormRESTProject** to test different HTTP methods used in REST-based server application **WebHelloFormRESTProject**.



```
    @GET  //HTTP request with GET Method
    @[redacted]
    @Produces(MediaType.TEXT_HTML)
    public String displayHelloAsParameter(
            @QueryP[redacted] ,
            @Query[redacted]
    {
        return "<html><title>Hello REST Resource</title><body>"
               +"<h3>Hello, Every one [redacted]
               " [redacted]
               "</h3></body></html>";
    }


    @[redacted] POST Method
    @Path[redacted]
    @Produces(MediaType.TEXT_HTML)
    public String displayHelloAsBody(
            @FormP[redacted] ,
            @FormP[redacted]
```



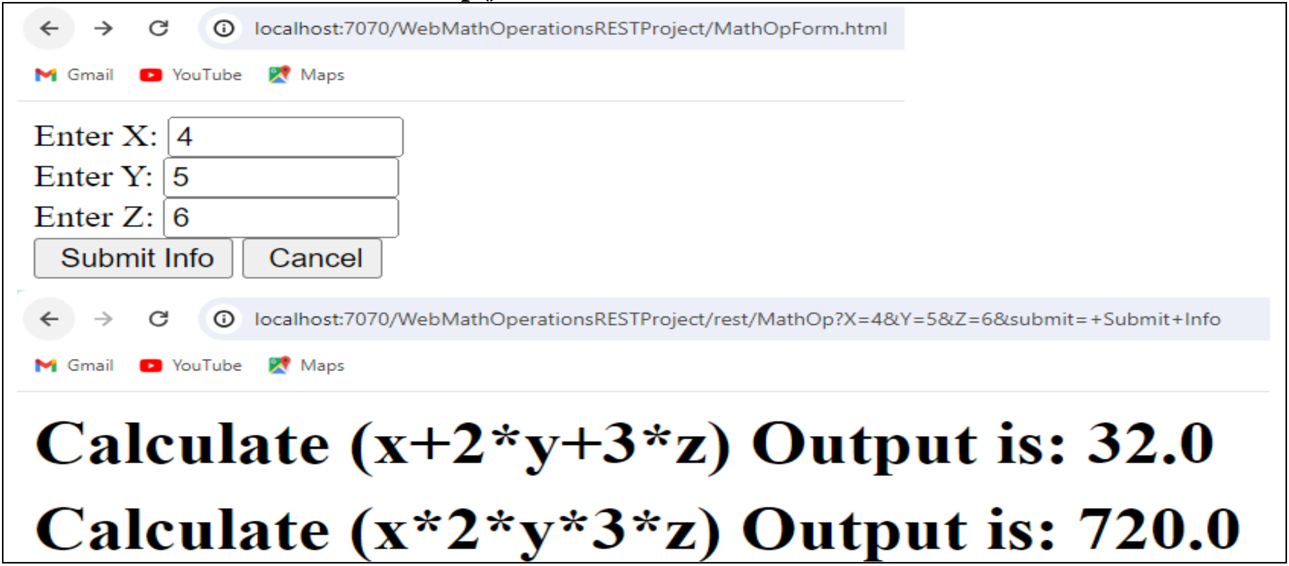
```
        //Testing WebHelloFormRESTProject
        //Invoking web REST PUT HTTP firing method updateNameAddBody

        target = client.target(url).[redacted]

        response = target.request()
                .accept(MediaType.TEXT_HTML)
                .put([redacted]
                .readEntity(String.class);

        System.out.println(response);
```

```
dy><h3>Hello Every one! From Ali Living in Year 2015</h3>New list: [Sarah, Anais, Bob, Ali]</body></h
dy><h3>Hello Every one! From Bob Living in Year 2015</h3>New list: [Sarah, Anais, BOB]</body></html>
dy><h3>Hello Every one! From Bob Living in Year 2015</h3>New list: [Sarah, Anais]</body></html>
```

Figure 1

2. Complete Maven Dynamic Web Project: WebMathOperationsRESTProject from Lab 3

a) Create HTML form **MathOpForm.html** that includes three inputs x, y, z to test the REST resource URL mapping ("/MathOp") with three **query string parameters** firing method **calculateHTMLOp()** as shown hereafter.


localhost:7070/WebMathOperationsRESTProject/MathOpForm.html
Gmail
YouTube
Maps
Submit Info
Cancel
localhost:7070/WebMathOperationsRESTProject/rest/MathOp?X=4&Y=5&Z=6&submit=+Submit+Info
Gmail
YouTube
Maps

Enter X:	4
Enter Y:	5
Enter Z:	6

Calculate (x+2*y+3*z) Output is: 32.0

Calculate (x*2*y*3*z) Output is: 720.0

b) Add a new path URL mapping ("/OpFormHashMap…") with form parameter x as search parameter to access REST resource searching into Hash Map.

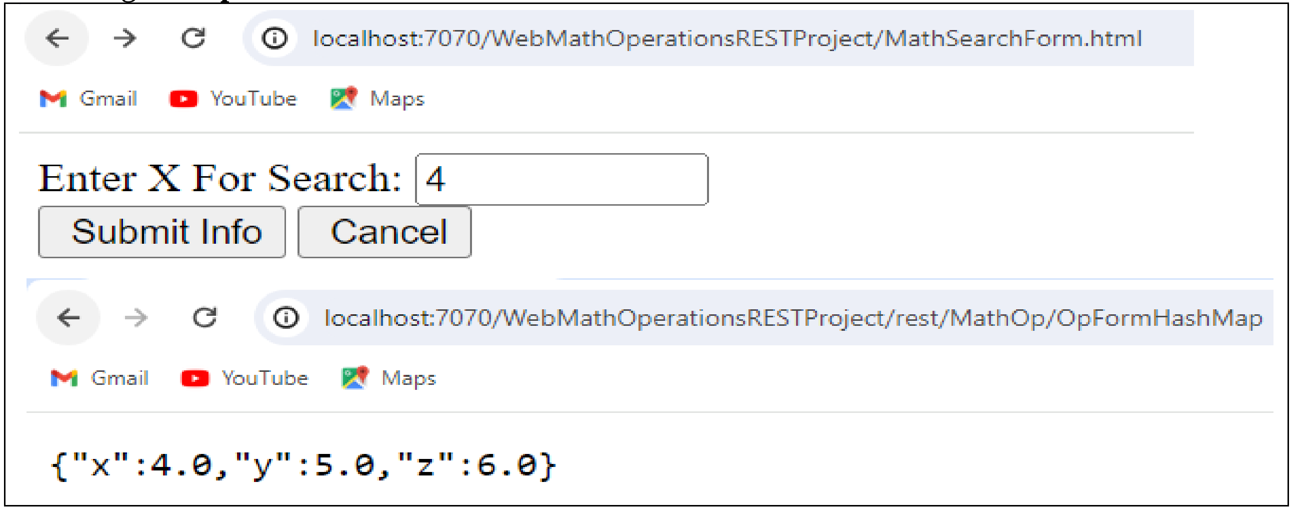
-Add a method **searchHashMapListZYZ()** using **form parameter** `x` and returns a JSON media type using appropriate Java REST annotations and will be fired upon using URL mapping ("/OpFormHashMap…").

-Add appropriate statements in **searchFormHashMapListZYZ ()** to instantiate a data structure **HashMap** `of MathOp class type` to be referenced by (`opHashMap`) where hash map key represents `x (form parameter)` and value hash map of `MathOp class type`. Set every component of hash map to the following values:

`x = 1,y = 2,z = 3/ x = 4,y = 5,z = 6 / x = 7,y = 8,z = 9`

-Skip through `Hash Map collection` (`opHashMap`) and display the result of Hash Map search in JSON media type using **form parameter** `x`.

-Create HTML form **MathSearchForm.html** that includes one search input x to test the REST resource URL mapping ("/OpHashMap") firing **searchFormHashMapListZYZ()** using **form parameter** x as shown hereafter.

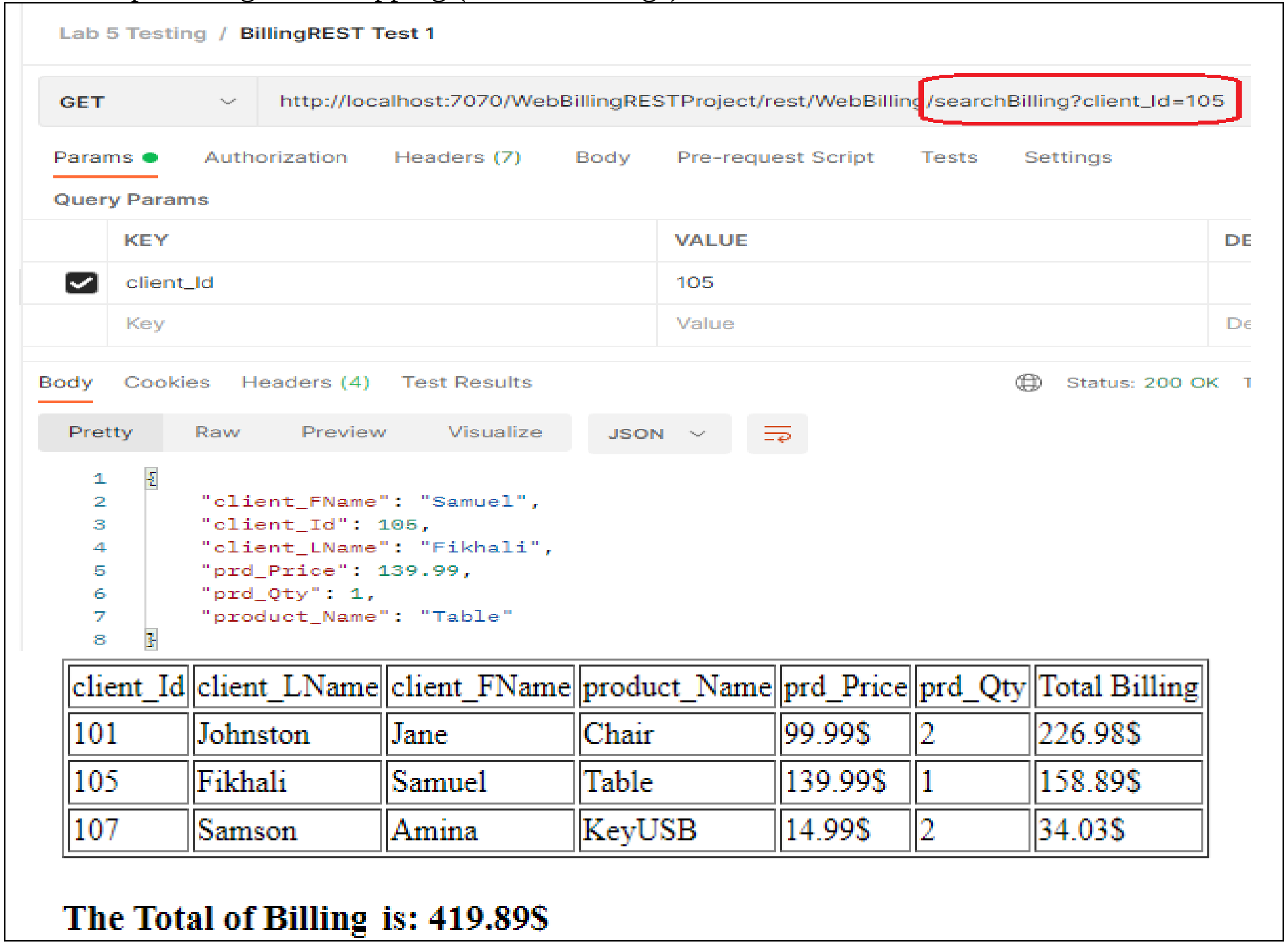

localhost:7070/WebMathOperationsRESTProject/MathSearchForm.html
Gmail
YouTube
Maps
Enter X For Search: 4
Submit Info
Cancel
localhost:7070/WebMathOperationsRESTProject/rest/MathOp/OpFormHashMap
Gmail
YouTube
Maps

```
{"x":4.0,"y":5.0,"z":6.0}
```

3. Complete Maven Dynamic Web Project: WebBillingRESTProject from Lab 3

a) Add a new path URL mapping ("/searchBilling") with **query string parameter** client_id as search integer parameter to access REST resource searching into Array List using appropriate Java REST annotation. Save your own Postman screenshot in word document.

b) Add a method **searchAsQPBillingInfo(int client_id)** that returns the result of Array List search as JSON media type using appropriate Java REST annotations and will be fired upon using URL mapping ("/searchBilling") as shown hereafter.



```
http://localhost:7070/WebBillingRESTProject/rest/WebBilling/searchBilling?client_Id=105
```

Query Params

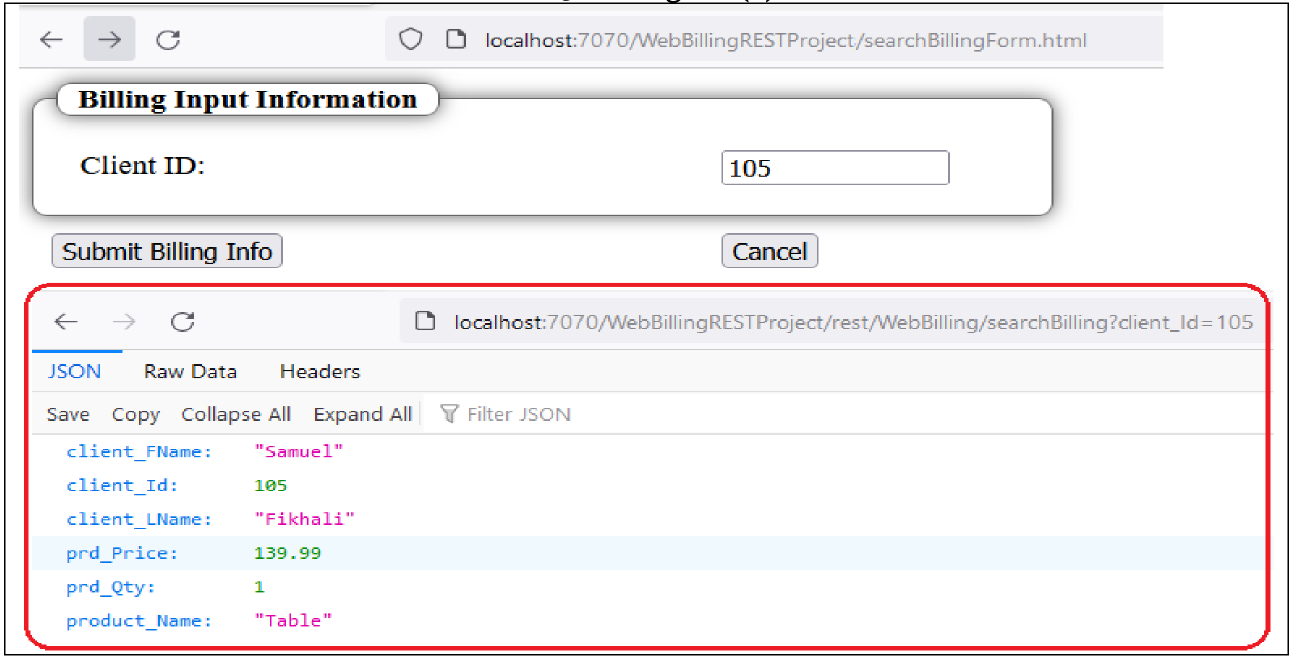
KEY	VALUE	DE
client_Id	105	
Key	Value	De

```
{
    "client_FName": "Samuel",
    "client_Id": 105,
    "client_LName": "Fikhali",
    "prd_Price": 139.99,
    "prd_Qty": 1,
    "product_Name": "Table"
}
```

client_Id	client_LName	client_FName	product_Name	prd_Price	prd_Qty	Total Billing
101	Johnston	Jane	Chair	99.99$	2	226.98$
105	Fikhali	Samuel	Table	139.99$	1	158.89$
107	Samson	Amina	KeyUSB	14.99$	2	34.03$

The Total of Billing is: 419.89$

c) Create HTML form **searchBillingForm.html** that includes one input client ID to test the REST resource method **searchAsQPBillingInfo(..)** as shown hereafter.



localhost:7070/WebBillingRESTProject/searchBillingForm.html

Billing Input Information

Client ID: 105

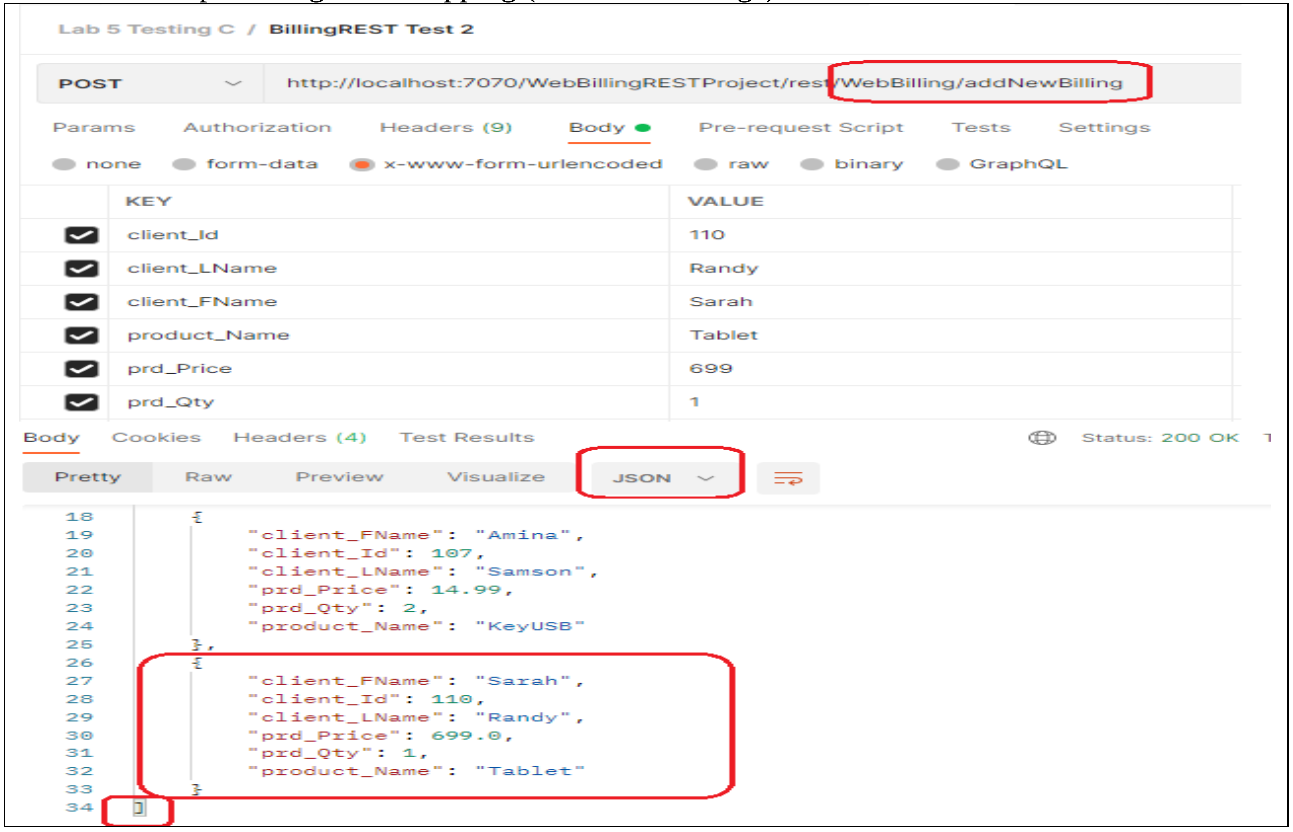
Submit Billing Info | Cancel

localhost:7070/WebBillingRESTProject/rest/WebBilling/searchBilling?client_Id=105

```
client_FName:    "Samuel"
client_Id:       105
client_LName:    "Fikhali"
prd_Price:       139.99
prd_Qty:         1
product_Name:    "Table"
```

d) Add a new path URL mapping ("/addNewBilling") with **form parameters** client_Id, client_LName, client_FName, product_Name, prd_Price, prd_Qty to add new Billing info to **ArrayList** data structure using appropriate Java REST annotation. Save your own Postman screenshot in word document.

e) Add a method **addNewBillingInfo (…)** that returns the Array List after adding new billing info as JSON media type using appropriate Java REST annotations and will be fired upon using URL mapping ("/addNewBilling") as shown hereafter.



Lab 5 Testing C / BillingREST Test 2
Params Authorization Headers (9) Body Pre-request Script Tests Settings
none form-data x-www-form-urlencoded raw binary GraphQL
Body Cookies Headers (4) Test Results
Pretty Raw Preview Visualize JSON

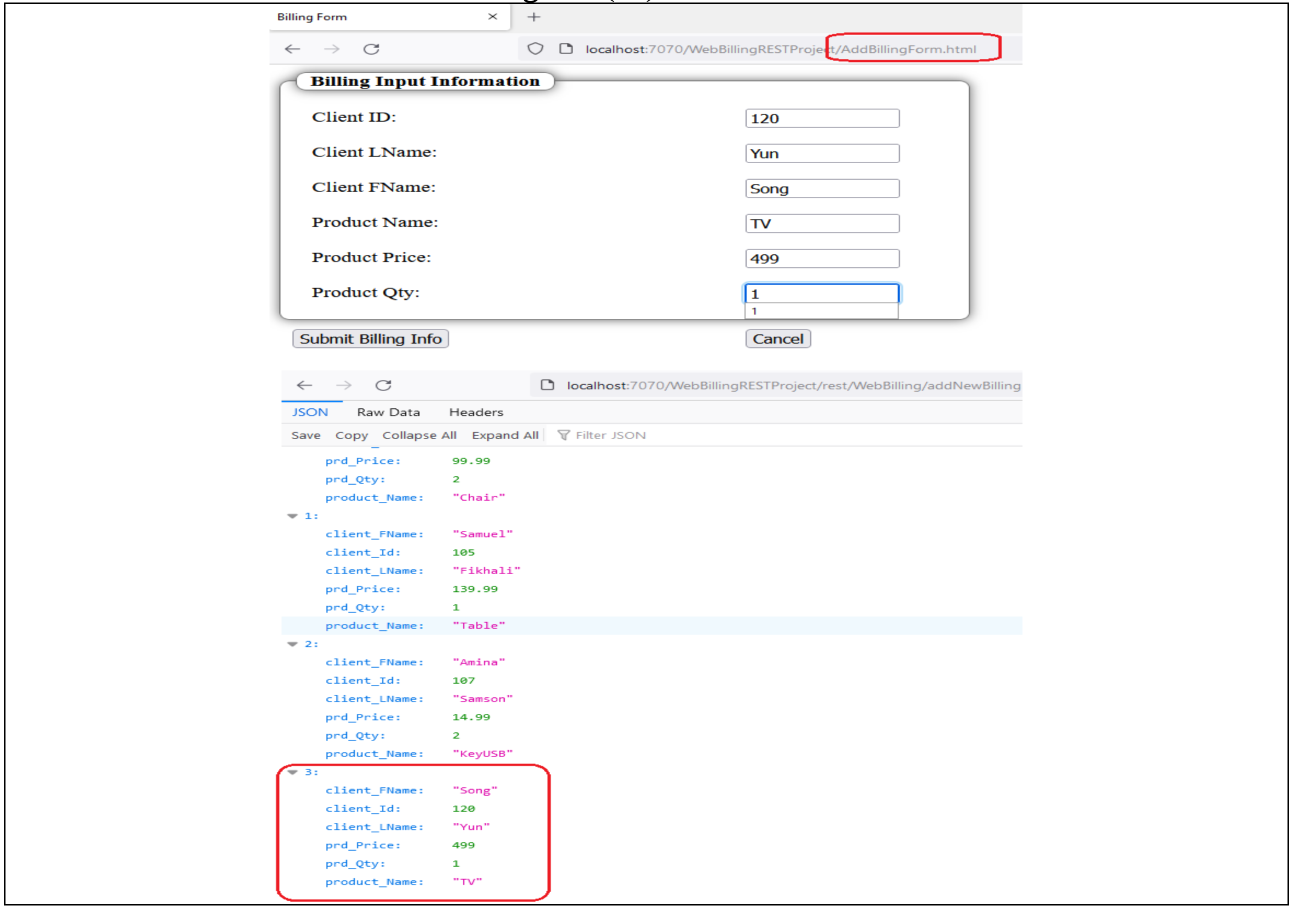
POST http://localhost:7070/WebBillingRESTProject/rest/WebBilling/addNewBilling

KEY	VALUE
client_Id	110
client_LName	Randy
client_FName	Sarah
product_Name	Tablet
prd_Price	699
prd_Qty	1

Status: 200 OK

```
        {
            "client_FName": "Amina",
            "client_Id": 107,
            "client_LName": "Samson",
            "prd_Price": 14.99,
            "prd_Qty": 2,
            "product_Name": "KeyUSB"
        },
        {
            "client_FName": "Sarah",
            "client_Id": 110,
            "client_LName": "Randy",
            "prd_Price": 699.0,
            "prd_Qty": 1,
            "product_Name": "Tablet"
        }
]
```

f) Create HTML form **AddBillingForm.html** that includes six inputs to test the REST resource method **addNewBillingInfo (…)** as shown hereafter.



Billing Form

localhost:7070/WebBillingRESTProject/AddBillingForm.html

Billing Input Information

Client ID:	120
Client LName:	Yun
Client FName:	Song
Product Name:	TV
Product Price:	499
Product Qty:	1

Submit Billing Info   Cancel

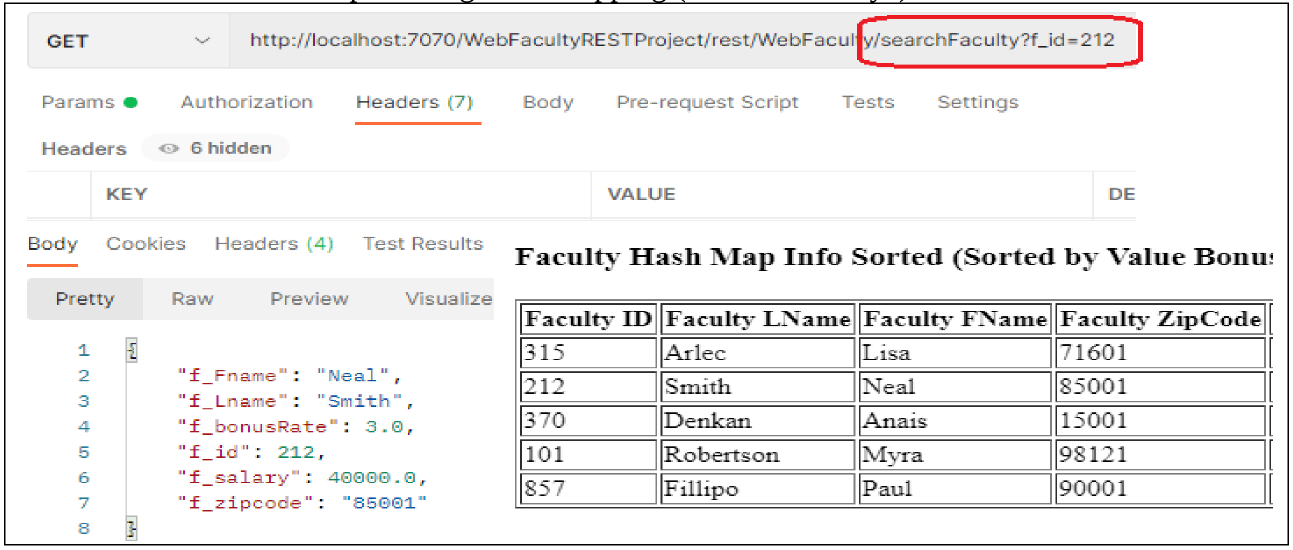
localhost:7070/WebBillingRESTProject/rest/WebBilling/addNewBilling

```
    prd_Price:      99.99
    prd_Qty:        2
    product_Name:   "Chair"
▼ 1:
    client_FName:   "Samuel"
    client_Id:      105
    client_LName:   "Fikhali"
    prd_Price:      139.99
    prd_Qty:        1
    product_Name:   "Table"
▼ 2:
    client_FName:   "Amina"
    client_Id:      107
    client_LName:   "Samson"
    prd_Price:      14.99
    prd_Qty:        2
    product_Name:   "KeyUSB"
▼ 3:
    client_FName:   "Song"
    client_Id:      120
    client_LName:   "Yun"
    prd_Price:      499
    prd_Qty:        1
    product_Name:   "TV"
```

4. Complete Maven Dynamic Web Project: WebFacultyRESTProject from Lab 4

a) Add a new path URL mapping ("/searchFaculty") with **query string parameter** `f_id` as search string parameter to access REST resource searching into **faculty hash map** using appropriate Java REST annotation. Save your own Postman screenshot in word document.

b) Add appropriate method that returns the result of hash map search as JSON media type and will be fired upon using URL mapping ("/searchFaculty") as shown hereafter.



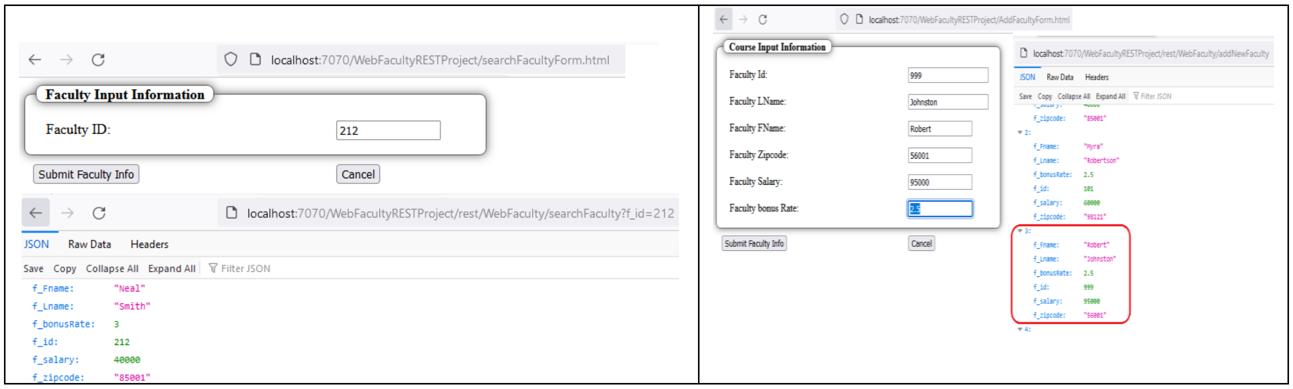
GET   http://localhost:7070/WebFacultyRESTProject/rest/WebFaculty/searchFaculty?f_id=212

```
{
    "f_Fname": "Neal",
    "f_Lname": "Smith",
    "f_bonusRate": 3.0,
    "f_id": 212,
    "f_salary": 40000.0,
    "f_zipcode": "85001"
}
```

Faculty Hash Map Info Sorted (Sorted by Value Bonu

Faculty ID	Faculty LName	Faculty FName	Faculty ZipCode
315	Arlec	Lisa	71601
212	Smith	Neal	85001
370	Denkan	Anais	15001
101	Robertson	Myra	98121
857	Fillipo	Paul	90001

c) Create HTML form **searchFacultyForm.html** that includes one input faculty id to test the REST resource search method as shown hereafter.



localhost:7070/WebFacultyRESTProject/searchFacultyForm.html

Faculty Input Information

Faculty ID:	212

localhost:7070/WebFacultyRESTProject/rest/WebFaculty/searchFaculty?f_id=212

```
f_Fname:      "Neal"
f_Lname:      "Smith"
f_bonusRate:  3
f_id:         212
f_salary:     40000
f_zipcode:    "85001"
```

localhost:7070/WebFacultyRESTProject/AddFacultyForm.html

Course Input Information

Faculty Id:	999
Faculty LName:	Johnston
Faculty FName:	Robert
Faculty Zipcode:	56001
Faculty Salary:	95000
Faculty bonus Rate:	2.5

localhost:7070/WebFacultyRESTProject/rest/WebFaculty/addNewFaculty

```
    f_zipcode:    "85001"
▼ 2:
    f_Fname:      "Myra"
    f_Lname:      "Robertson"
    f_bonusRate:  2.5
    f_id:         101
    f_salary:     60000
    f_zipcode:    "98121"
▼ 3:
    f_Fname:      "Robert"
    f_Lname:      "Johnston"
    f_bonusRate:  2.5
    f_id:         999
    f_salary:     95000
    f_zipcode:    "56001"
▼ 4:
```

d) Add a new path URL mapping ("/addNewFaculty") with appropriate **form parameters** to add new Faculty info into **HashMap** using appropriate Java REST annotation.

e) Add a method **addNewFacultyInfo (…)** that returns hash map collection as JSON media type after adding new faculty info and will be fired upon using URL mapping ("/addNewFaculty") as shown hereafter. Save your own Postman screenshot in word document.

f) Create HTML form **AddFacultyForm.html** that includes six inputs to test the REST resource method **addNewFacultynfo (…)** as shown above.