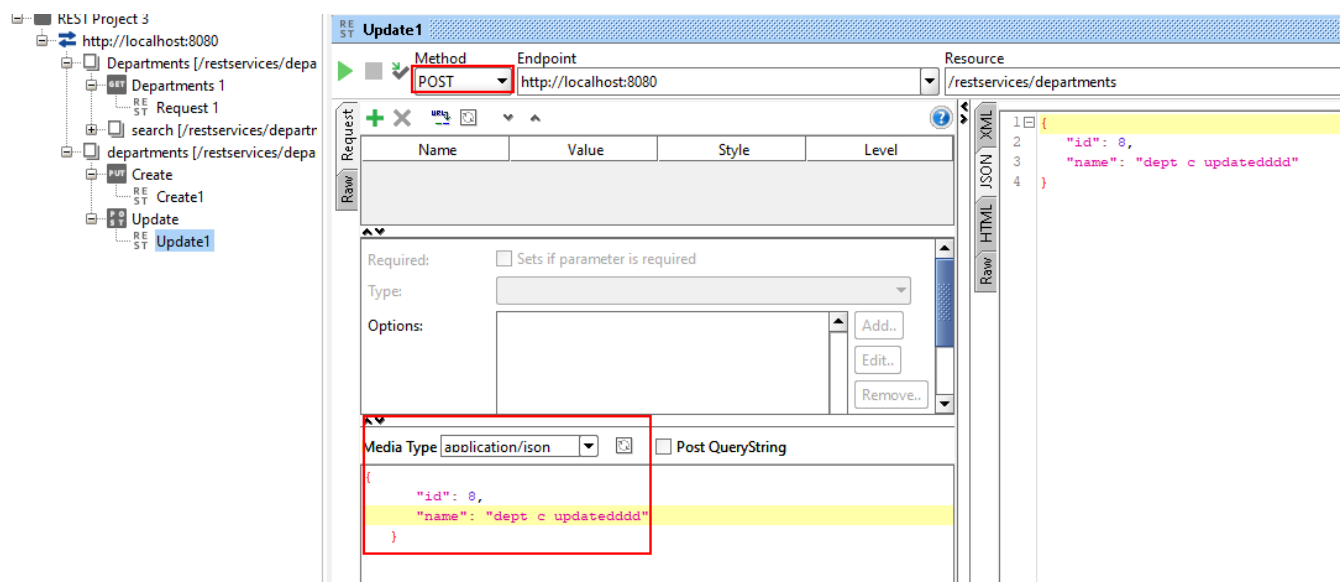
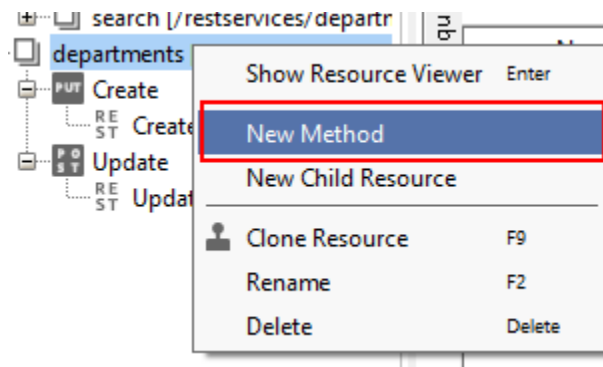


## 6. Laborator: Servicii REST partea a 2-a

### 6.1 Rest

Ne vom folosi de arhiva starter: laboratoree6\_starter.zip.

Este structura laboratorului 5 cu extensia citirii unei entitati dupa ID, dar si partea de update. Pentru soapUI trebuie sa adaugati o noua metoda (click dreapta > new method / sau clone method)



### 6.2 Servlet

In modulul "REST backend services" creati un nou pachet: ro.ulbs.ip.an3.servlet

Selectati acest nou pachet > click dreapta > New > Other > Web > Servlet

Numele clasei: ServiciiAditionaleServlet

## Lab 6

URL Pattern(s): /ServiciiAditionale

Observati codul generat.

Observati anotarea clasei: @WebServlet(name = "ServiciiAditionaleServlet", urlPatterns = {"/ServiciiAditionale"})

Observati metodele doGet si doPost.

Sugestie: Daca nu regastiti aceste metode, observati ca netbeans le comaseaza sub o singura linie la finalul sursei. Expandati aceasta linie.

Apoi modificati codul lor astfel:

```
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        out.println("<html><body>");
        out.println("<h2>Laborator 6</h2> <br> Raspuns pt GET");
        out.println("</body></html>");
    }
}

@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {

    Map<String, String[]> parametrii = request.getParameterMap();
    String paramsToString = "";
    for (String key: parametrii.keySet()) {
        String[] values = parametrii.get(key);
        String val = "";
        for (String v: values) {
            val += " " + v;
        }
        paramsToString += "<br>"+key+"="+val;
    }
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        out.println("<html><body>");
        out.println("<h2>Laborator 6</h2> <br> Raspuns pt POST.");
        out.println("<br> Parametrii trimisi sunt: ");
        out.println(paramsToString);
        out.println("</body></html>");
    }
}
```

Este nevoie sa importati java.util.Map.

Compilati si faceti deployment (<http://localhost:4848/> > application > deploy). Daca aveti deja laboratorul 5 depleoy-at atunci executati undeploy.

Accesati acest serviciu web la: **<http://localhost:8080/restservices/ServiciiAditionale>**

Pt GET puteti folosi browserul.

Pt POST trebuie sa folositi uneltele SoapUI/Postman sau incarcati un fisier html in browser care sa contina forma de mai jos:

```
<html>
<body>
Testare POST.
<form action="http://localhost:8080/restservices/ServiciiAditionale" method="post">

  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>

  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>

  <input type="submit" value="Submit">
</form>
</body>
</html>
```

## Sumar

Aplicatia de fata foloseste urmatoarele : Java EE, Maven, JPA, EJB, REST, Servlet

Serviciile WEB tip REST sunt expuse la <http://localhost:8080/restservices/departments>

Serviciile WEB tip Servlet sunt expuse la <http://localhost:8080/restservices/ServiciiAditionale>

Browserul web poate accesa url-ul de mai sus si face cerere de tip GET. Pentru alte tipuri de operatii (PUT, POST, DELETE) ne vom folosi de unelte specializate (SoapUI, PostMan, RestAssured).

## 6.3 Probleme

6.3.1 Adaugati un nou servlet cu numele ServiciiCalcul, configurat pentru

<http://localhost:8080/restservices/ServiciiCalcul>

si care pe GET sa raspunda cu cu help-ul de mai jos :

« POST : a=... b=... Rezultatul este suma acestor valori  
 PUT: a=... b=... Rezultatul este produsul acestor valori  
 “

Iar pentru POST si PUT sa execute suma parametrilor primiti respectiv produsul parametrilor primiti.

Daca parametrii nu pot fi convertiti la Integer, folositi valoarea 0 ;

6.3.2 Adaugati o noua clasa RestEmployee care sa ofere servicii REST pentru entitatile Employee. Trebuie sa oferiti support pentru operatii CRUD (Create – PUT, Read-GET, Update – POST, Delete – DELETE)

**Depanare / Troubleshooting**

1. Conversia unui String la int se face in mai multe metode. Va propun urmatoarea functie :

```
public int convert(String v) {
    int ret = 0 ;
    try{
        ret = Integer.parseInt(v) ;
    }catch(NumberFormatException nfe{
    }
    return ret;
}
```

2. Daca in timpul deploymentului logurile Glassfish-ului contin:

java.lang.RuntimeException: javax.naming.NameNotFoundException: CDIExtension

Atunci trebuie sa adaugam un parametru jdk-ului astfel :


- Glassfish admin console (<http://localhost:4848/>)  
Configurations > server-config > JVM Settings > tab 'JVM Options' >
- adaugati o noua optiune JVM:  
-Dcom.sun.jersey.server.impl.cdi.lookupExtensionInBeanManager=true
- restart glassfish

<input type="checkbox"/>	-Dcom.sun.jersey.server.impl.cdi.lookupExtensionInBeanManager=true
<input type="checkbox"/>	-XX:MaxPermSize=192m
<input type="checkbox"/>	-Djava.endorsed.dirs=\${com.sun.aas.installRoot}/modules/endorsed\${path.separator}\${com.sun.aas.ins
<input type="checkbox"/>	-Dcom.sun.enterprise.security.httpsOutboundKeyAlias=s1as
<input type="checkbox"/>	-client

3. Daca in timpul deployment-ului Glassfish-ul arunca erori ciudate. De ex. "Cannot find module restservices.war as defined by application.xml" atunci este necesar sa:

- opriti Glassfish
- stergeti continutul subdirectorului <...>\glassfish\domains\domain1\generated
- porniti Glassfish
- deploy din nou al aplicatiei
-

## Lab 6

 << glassfish > domains > domain1 > generated					▼	↺	🔍 Search generated	
s	⬆	⬆	⬆	⬆	⬆	⬆	⬆	⬆
Name				Date modified		Type		Size
📁 ejb				11/2/2021 10:59 PM		File folder		
📁 jsp				11/2/2021 10:58 PM		File folder		
📁 policy				11/2/2021 10:59 PM		File folder		
📁 xml				11/2/2021 10:59 PM		File folder		