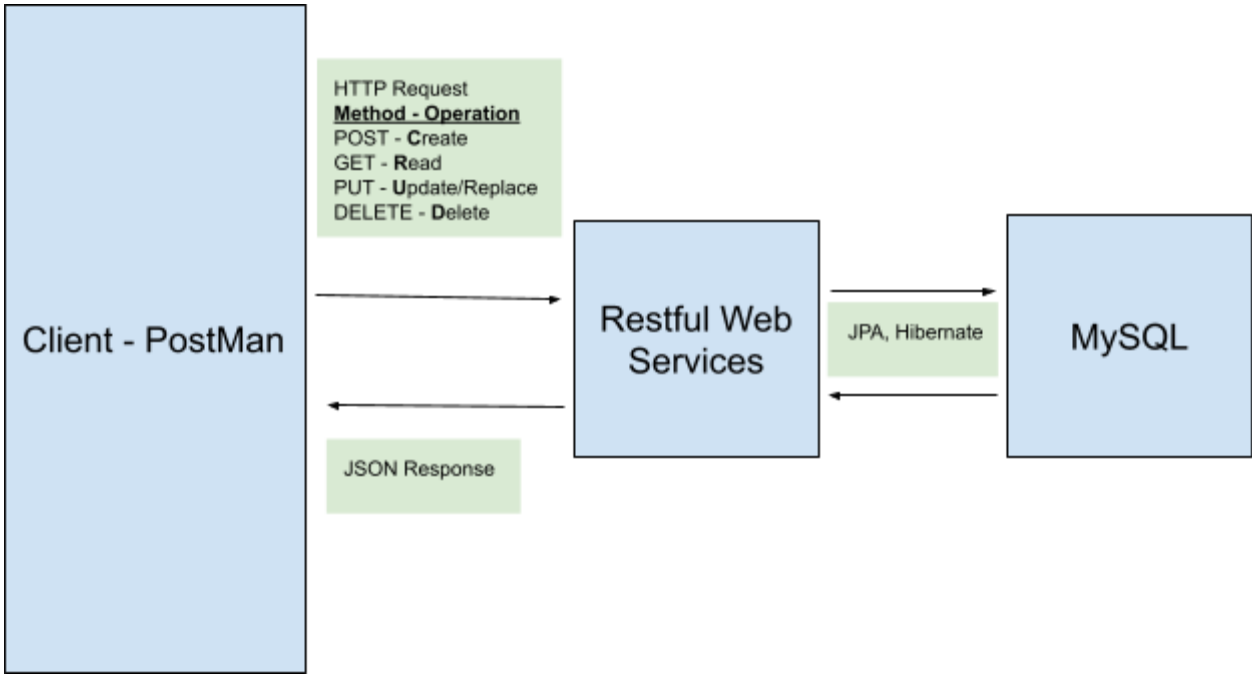
Requirement: To create RESTful Web services implementation using Java, Spring Boot, JPA, Hibernate and MySQL as back-end database.

Create the CRUD operation methods to perform actions like Save new Employee, Read existing Employees, Update existing Employee Information and finally Delete an existing Employee

Solution Specifications
For the POC solution to be accessible over the internet, both the application and the database should be hosted on a cloud environment. I have prefered to use **Amazon AWS** to get the benefits of **Elastic Beanstalk** for hosting the Spring boot application as JAR and the **AWS RDS** to host the MySQL database. The AWS Free Trial tier is being used to avoid any costs for hosting the POC solution.

Following are the components and the steps required to implement the solution.

Step 1
Solution Definition



Step 2

Create a **MySQL** database on AWS RDS
- DB identifier : ilinesoft-database-1
- Database: EmployeeDb
- Table : employee
- Endpoint - ilinesoft-database-1.c0vkclmcetpe.us-east-2.rds.amazonaws.com/

Add sample data manually using SQLStatement or MySql Client (HeidiSQL)

Host: ilinesoft-database-1.c0vkclmcetpe.us-east-2.rds.amazonaws.com | Database: EmployeeDb | Table: employee

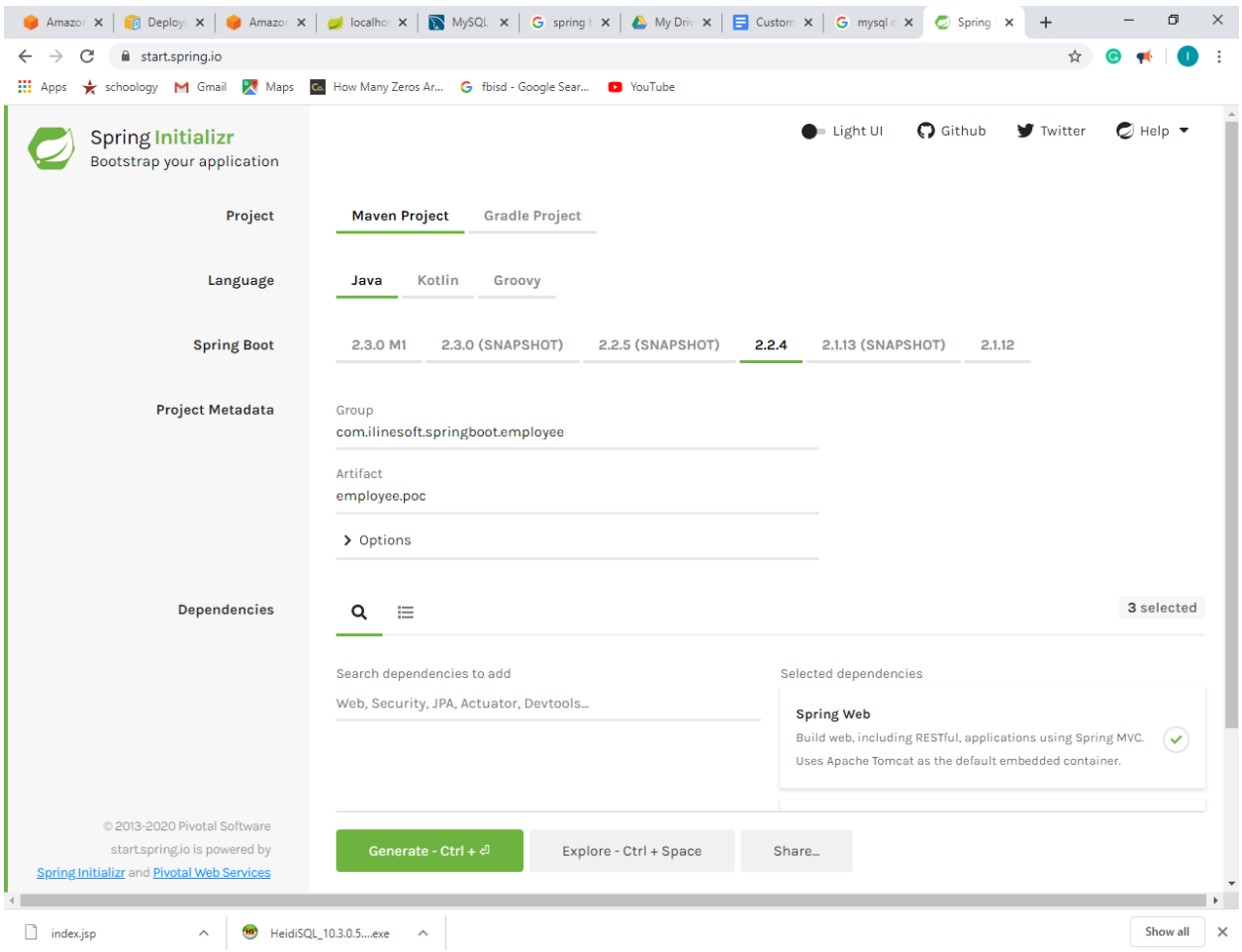
Data | Query* | Query #2*

EmployeeDb.employee: 3 rows total (approximately) | Next | Show all | Sorting | Columns (8/8) | Filter

id	first_name	last_name	work_location_city	technology	department	salary	date_of_joining
1,001	Ahmed	Shahab	Houston	Java, Spring Boot, MySql	Technology	40,000.67	2020-02-10
1,002	Raghu	Kalyan	Banglore	Java, Spring Boot, Angular, AWS	Technology	50,000.9	2020-02-26
1,003	Mohammed	Salman	Hyderabad	N/A	Human Resource	6,000,190	2014-01-25

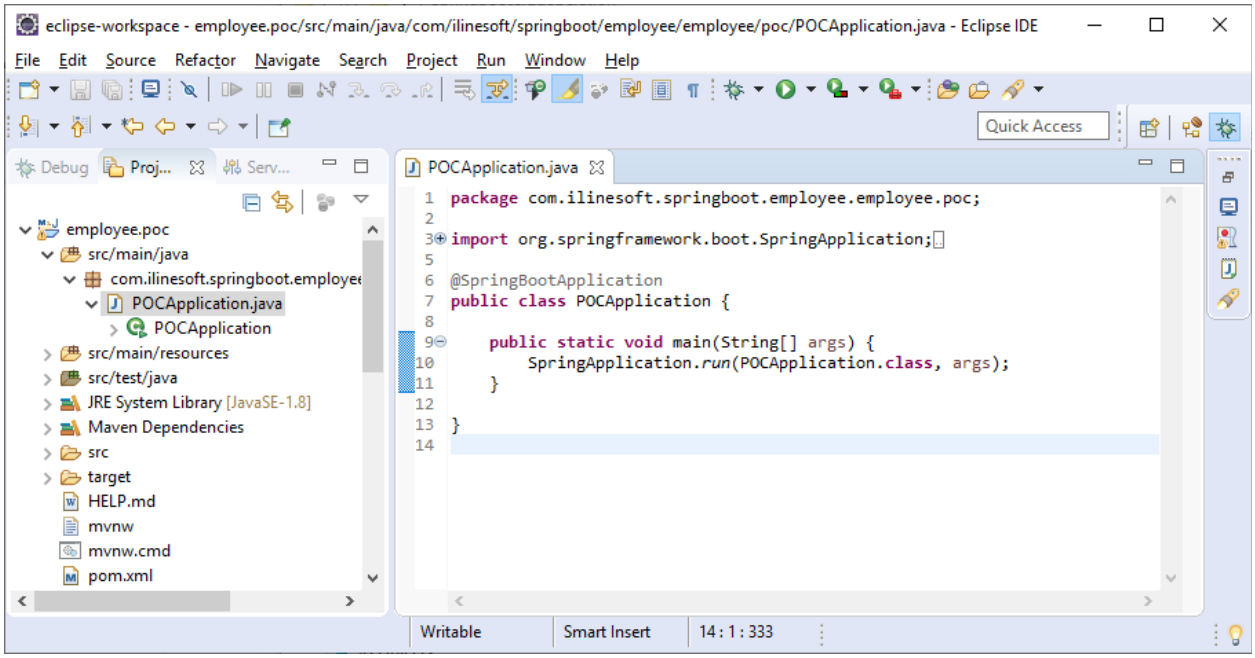
Step 3

Use **Spring Initializer** to generate spring boot projects by selecting Maven Project, Java, Spring Boot 2.2.4. Enter Project Metadata Group as *com.ilinesoft.springboot.employee* and Artifact *employee.poc*



Remember to select the dependency **Spring Web**, **Spring Data JPA** and **MySQL Driver**

Download the project and open in Eclipse IDE or IntelliJ IDEA Community Edition. I am comfortable using **Eclipse IDE**

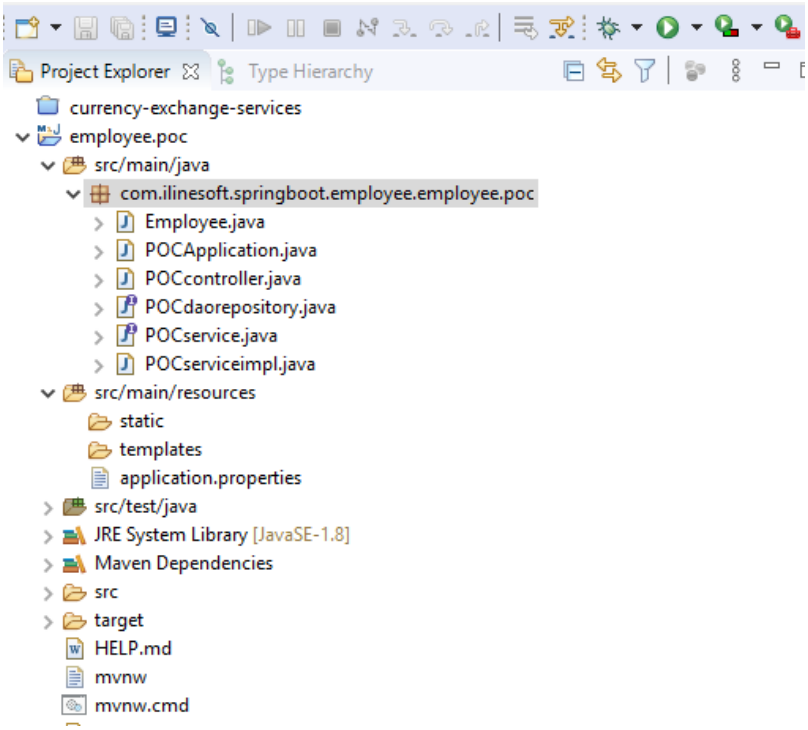


The Main class already exists as part of the Spring Initializer

Now create the below .java class files in the sequence order as shown below

a) Modal **Class** - Employee.java
b) Data-Access-Object **interface** - POCdaorepository.java
c) Service **interface** - POCservice.java
d) Service **Class** - POCserviceimpl.java
e) Controller **class** - POCcontroller.java
   Class designed to handle incoming HTTP requests
   Class is annotated with @RestController

Below screen shows files in the Project Explorer once the .java files are created



Step 4

Run the project using eclipse locally to test POC application is launching successfully by using a browser and typing localhost and port 5000

Note: We will test the remaining Rest API methods using postman tool in the subsequent steps after the application is deployed on AWS

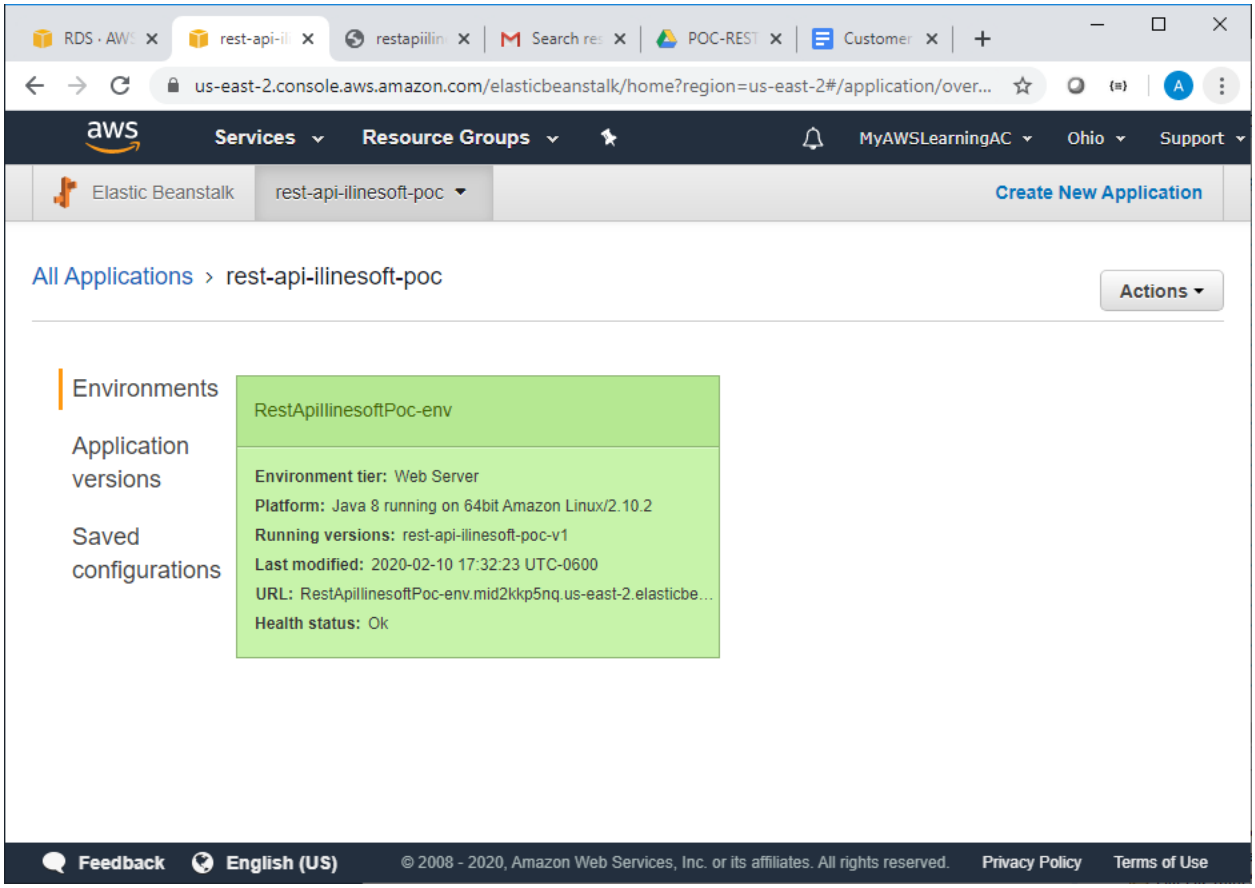
localhost:5000/employee/all


```
// 20200210181854
// http://localhost:5000/employee/all

[
  {
    "id": 1001,
    "technology": "Java, Spring Boot, MySql",
    "department": "Technology",
    "salary": 40000.67,
    "doj": "2020-02-10T00:00:00.000+0000",
    "lastName": "Shahab",
    "firstName": "Ahmed",
    "workLocationCity": "Houston"
  },
  {
    "id": 1002,
    "technology": "Java, Spring Boot, Angular, AWS",
    "department": "Technology",
    "salary": 50000.9,
    "doj": "2020-02-26T00:00:00.000+0000",
    "lastName": "Kalyan",
    "firstName": "Raghu",
    "workLocationCity": "Banglore"
  },
  {
    "id": 1004,
    "technology": "DevOps, Jenkins",
    "department": "Technology",
```

Step 5

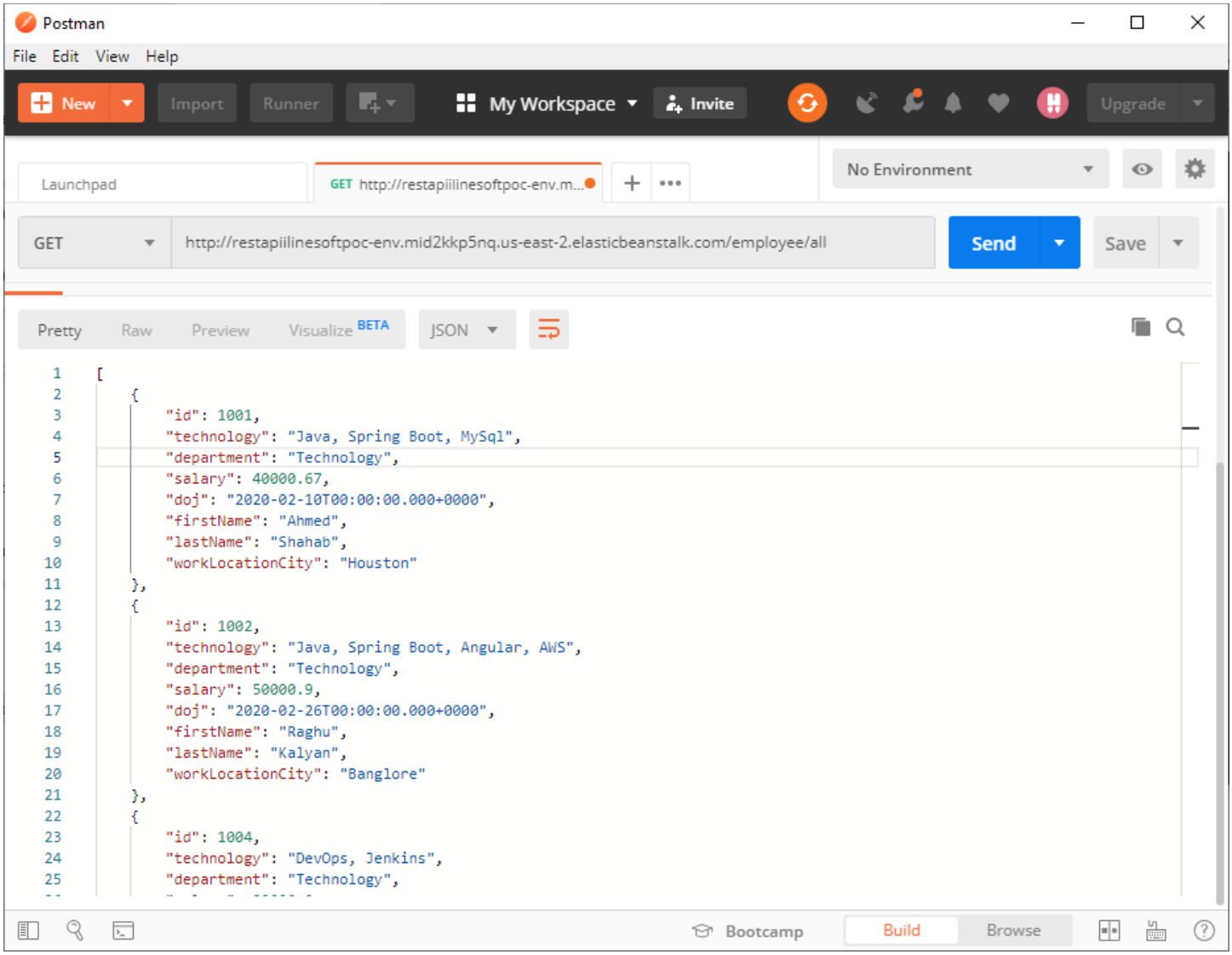
Export the project as JAR file and upload on AWS Elastic Beanstalk



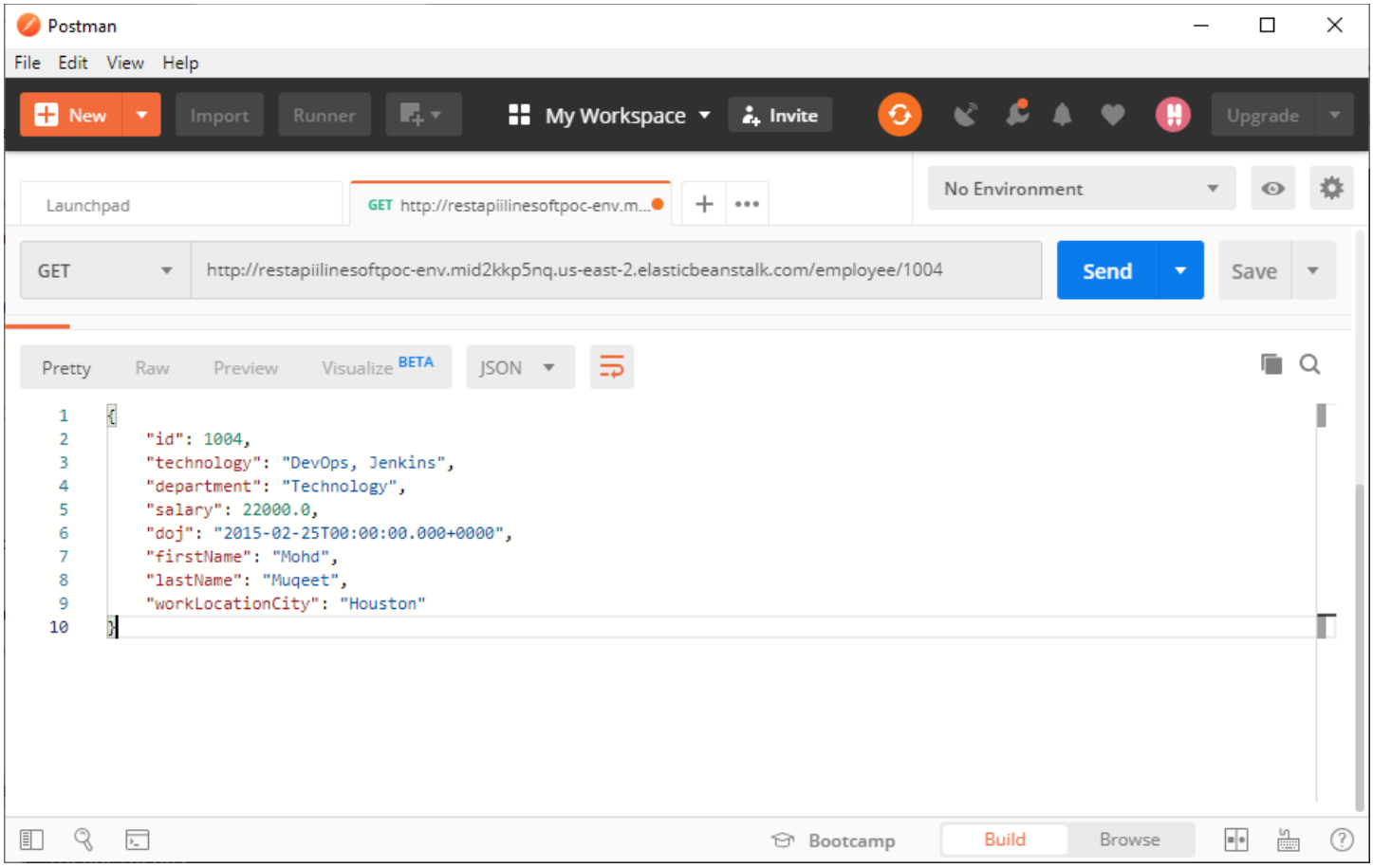
Step 6
Use the postman tool to test all the CRUD operations.

This step tests the **Read** operation

a) Testing the **GET method** to **Read** all employes using the Restful API
b) URL http://restapiilinesoftpoc-env.mid2kkp5nq.us-east-2.elasticbeanstalk.com**/employee/all**
c) Method **Get**



d) Testing the **GET method** with **parameter** to retrieve specific employe using the Restful API
e) URL http://restapiilinesoftpoc-env.mid2kkp5nq.us-east-2.elasticbeanstalk.com/**employee/1004**
f) Method Get



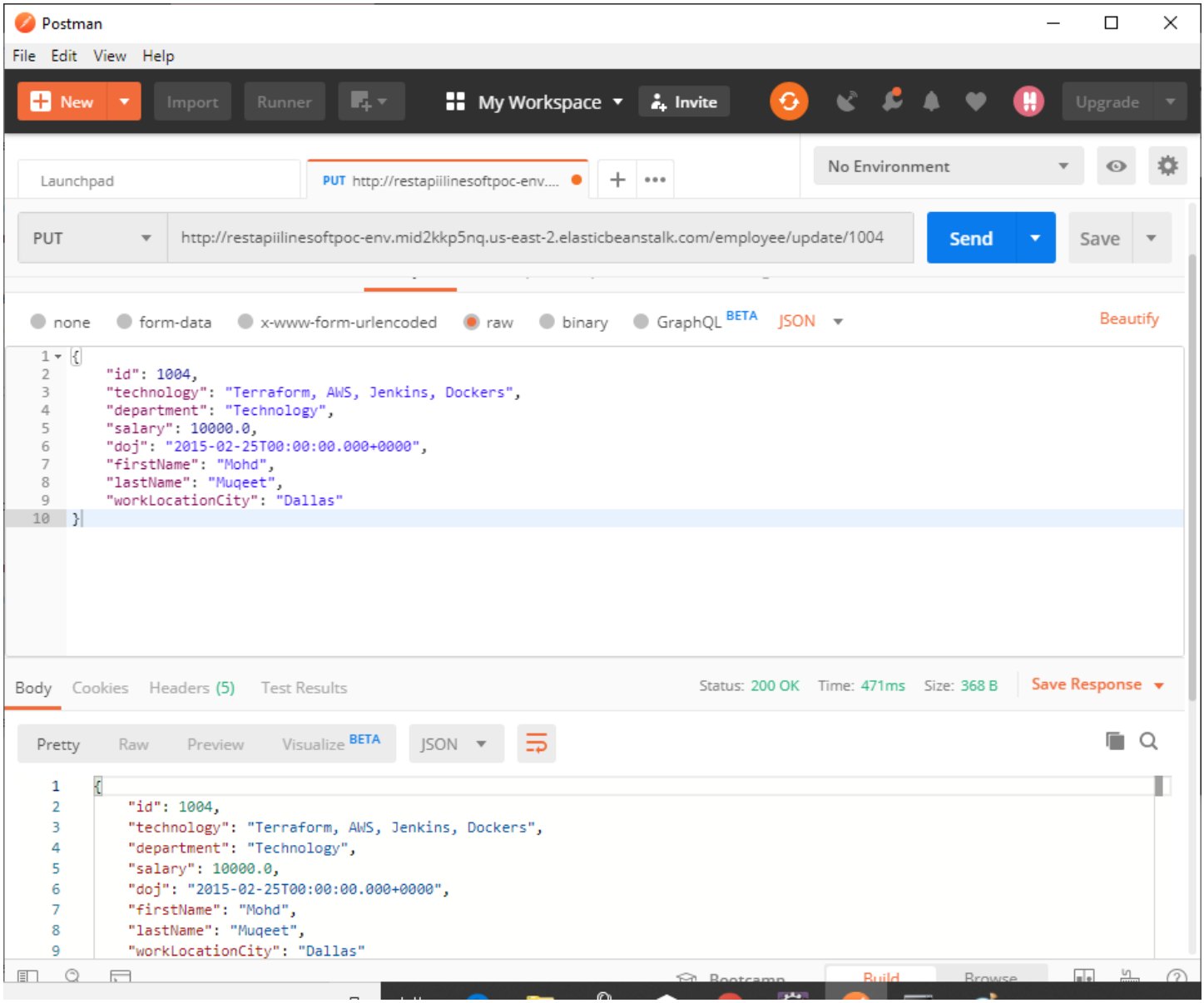
Step 7

This step tests the **Update** operation

g) Testing the **PUT RequestMethod** update specific employes using the Restful API

h) URL http://restapiilinesoftpoc-env.mid2kkp5nq.us-east-2.elasticbeanstalk.com**/employee/update/1004**

i) Method **PUT**



Step 8

This step tests the **Add** operation

j) Testing the **POST RequestMethod** add an employe detail using the Restful API

k) URL http://restapiilinesoftpoc-env.mid2kkp5nq.us-east-2.elasticbeanstalk.com/**employee/update/1004**

l) Method **Post**

Postman

File Edit View Help

New Import Runner My Workspace Invite Upgrade

Launchpad | POST http://restapiilinesoftpoc-env.... | No Environment

Untitled Request | Comments (0)

POST http://restapiilinesoftpoc-env.mid2kkp5nq.us-east-2.elasticbeanstalk.com/employee/add Send Save

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Code

none form-data x-www-form-urlencoded raw binary GraphQL BETA JSON Beautify

```
{
    "id": 1006,
    "technology": "Java Core, Spring Boot",
    "department": "Technology",
    "salary": 2500.00,
    "doj": "2017-04-28T00:00:00.000+0000",
    "firstName": "Nisar",
    "lastName": "Ahmed",
    "workLocationCity": "St Louis"
}
```

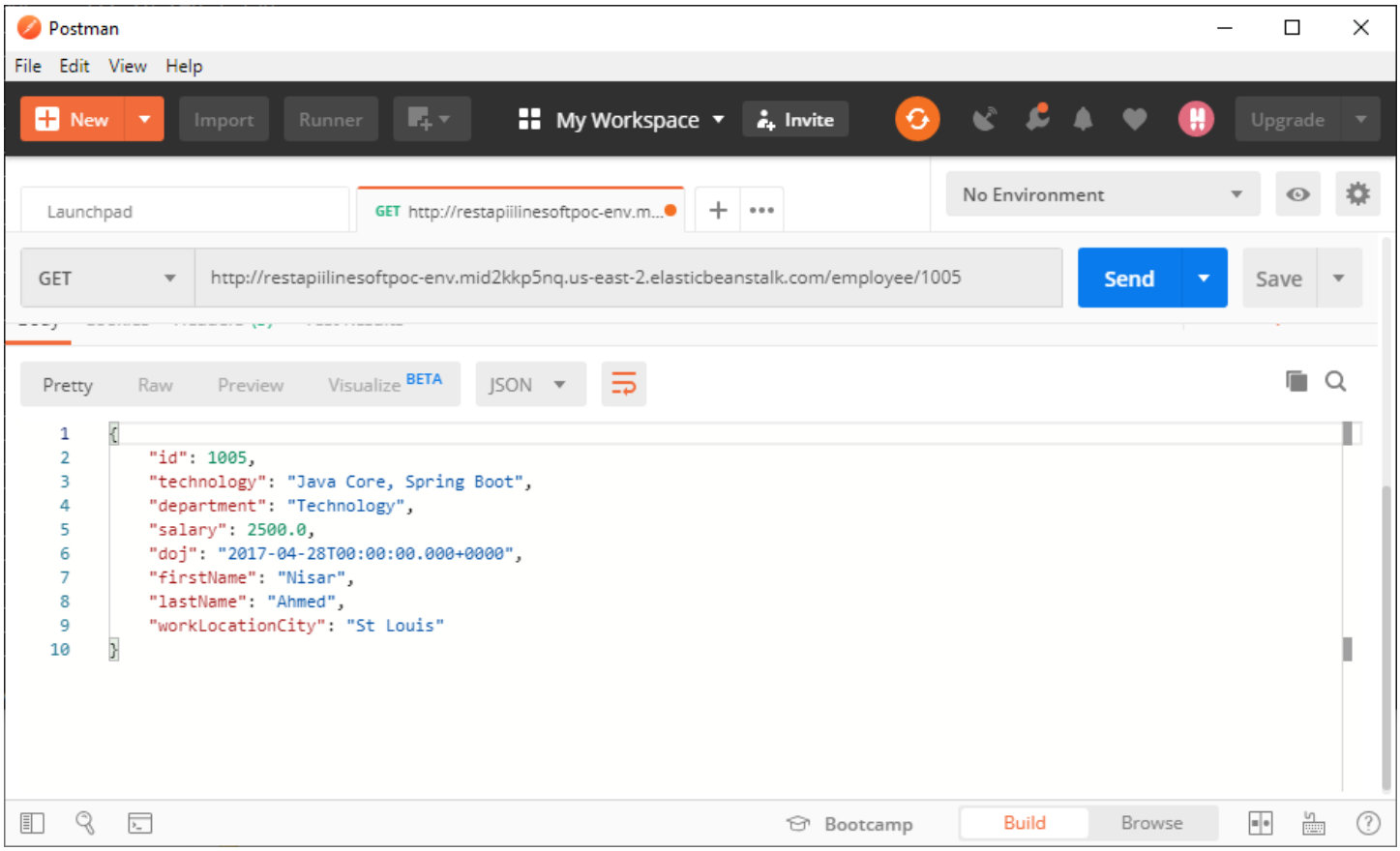
Body Cookies Headers (5) Test Results Status: 200 OK Time: 195ms Size: 359 B Save Response

Pretty Raw Preview Visualize BETA JSON

```
{
    "id": 1005,
    "technology": "Java Core, Spring Boot",
    "department": "Technology"
```

Bootcamp Build Browse

Use the GET method again to see the newly added Employee is pulled from the server



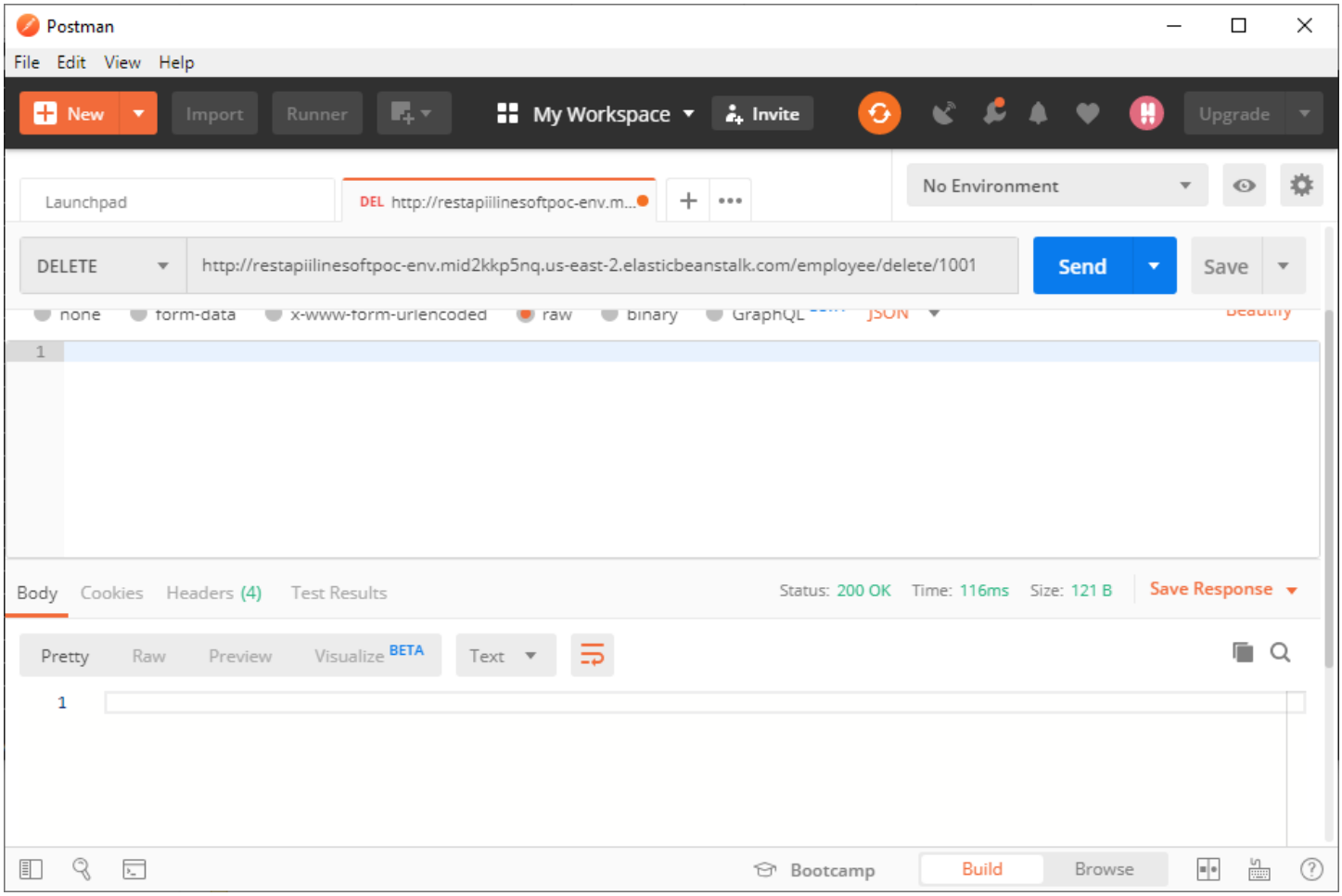
Step 9

This step tests the **Delete** operation

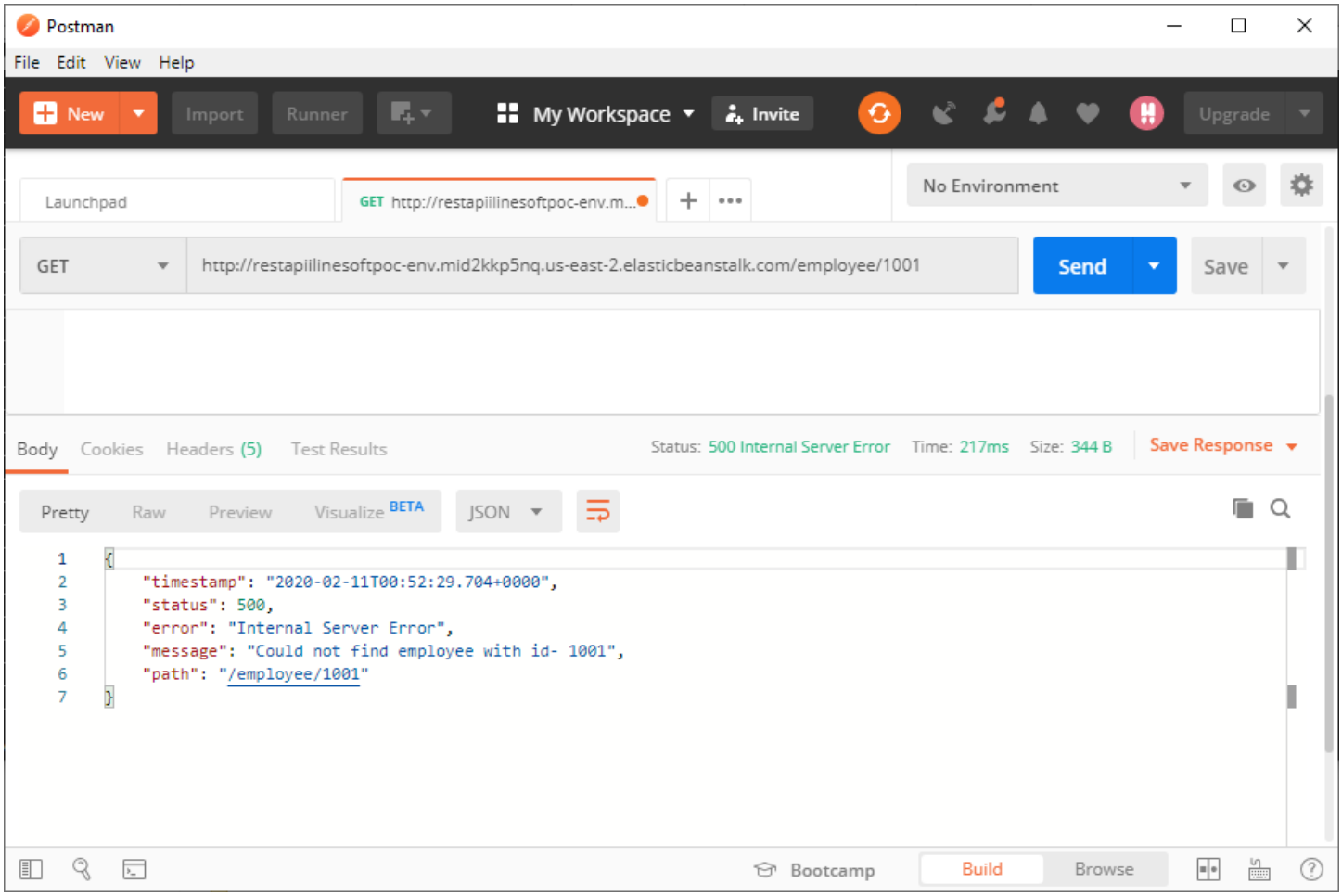
m) Testing the **DELETE RequestMethod** to delete an **specific employee** using the Restful API

n) URL http://restapiilinesoftpoc-env.mid2kkp5nq.us-east-2.elasticbeanstalk.com**/employee/delete/1001**

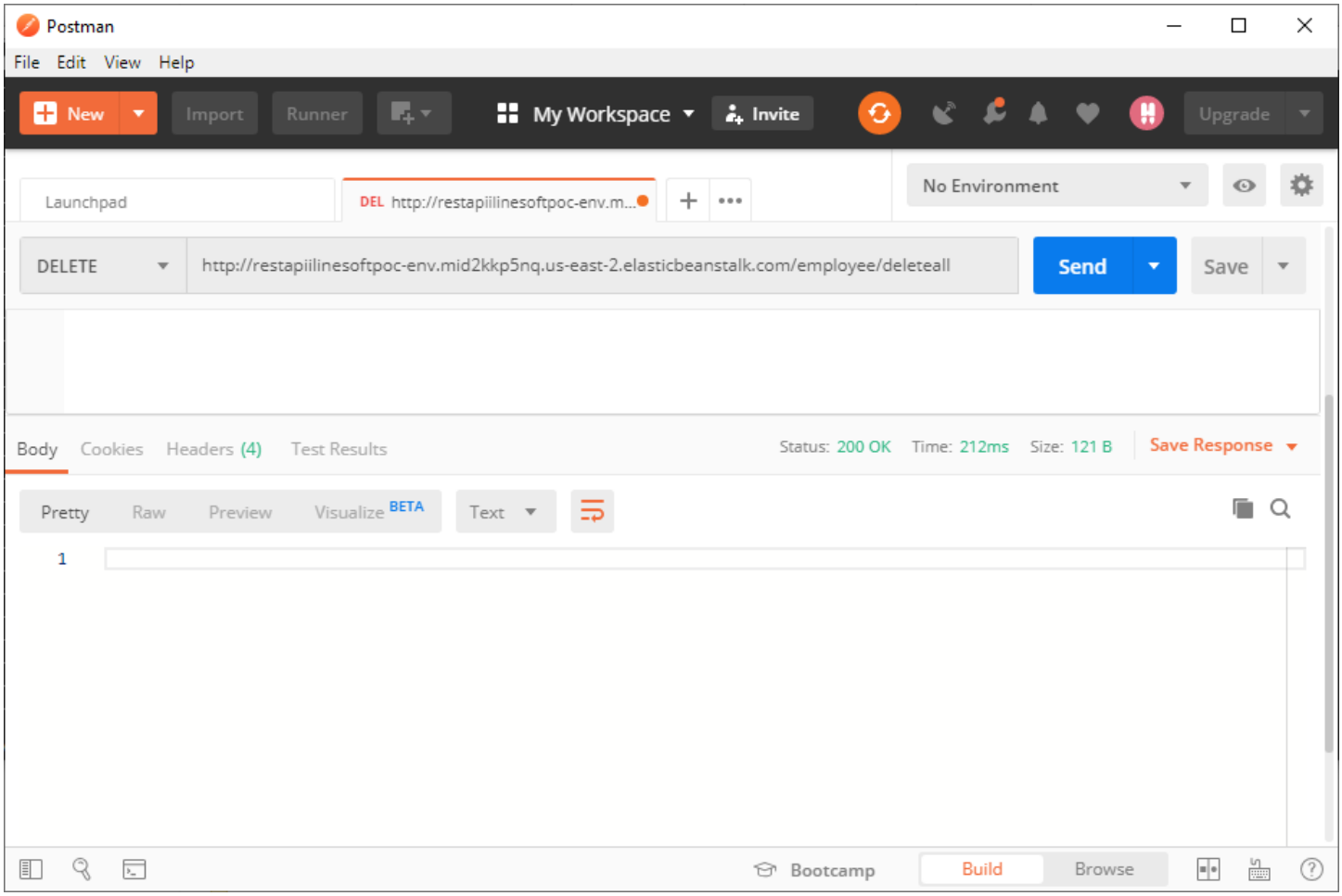
o) Method Delete



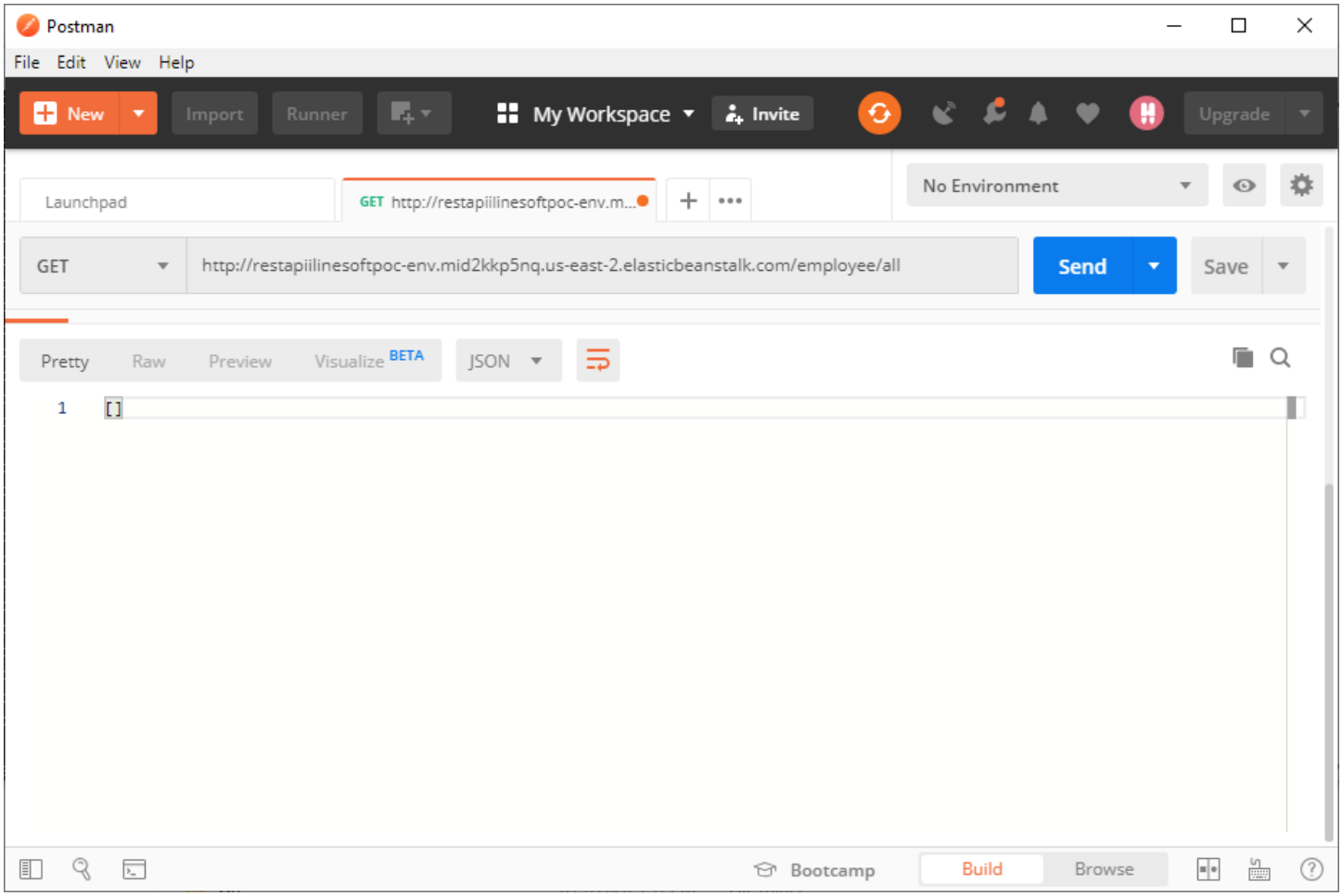
Check if the Employee (1001) has actually been deleted



p) Testing the **DELETE RequestMethod** to delete **all employees** using the Restful API
q) URL http://restapiilinesoftpoc-env.mid2kkp5nq.us-east-2.elasticbeanstalk.com/**employee/deleteall**
r) Method Delete



Executing the GET method again to see any employee record exists.



Conclusion

In the above steps, Restful API CRUD operations have been tested using the Postman tool to demonstrate a Restful application running Java and Spring Boot.

------ End of document-----