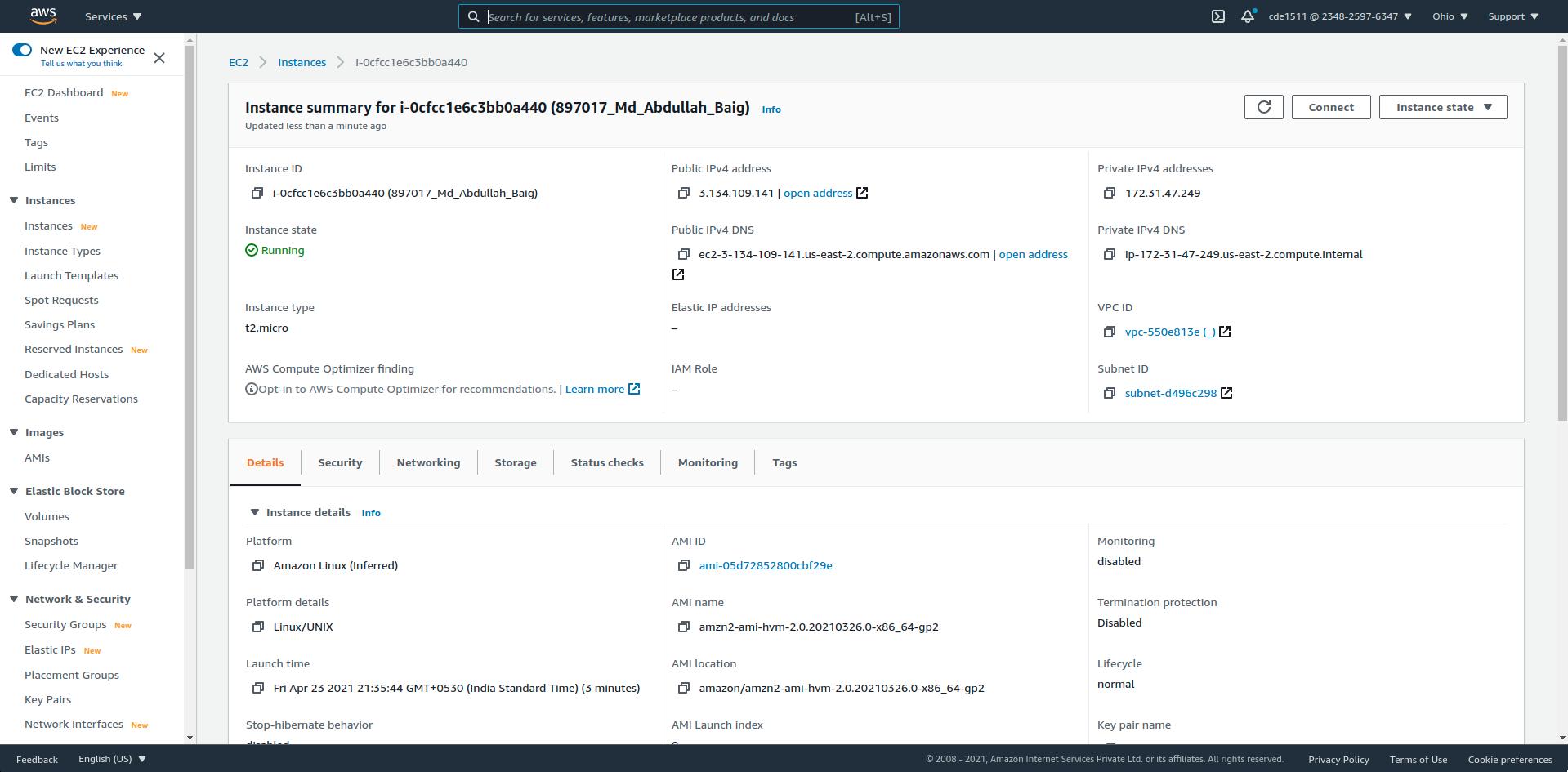
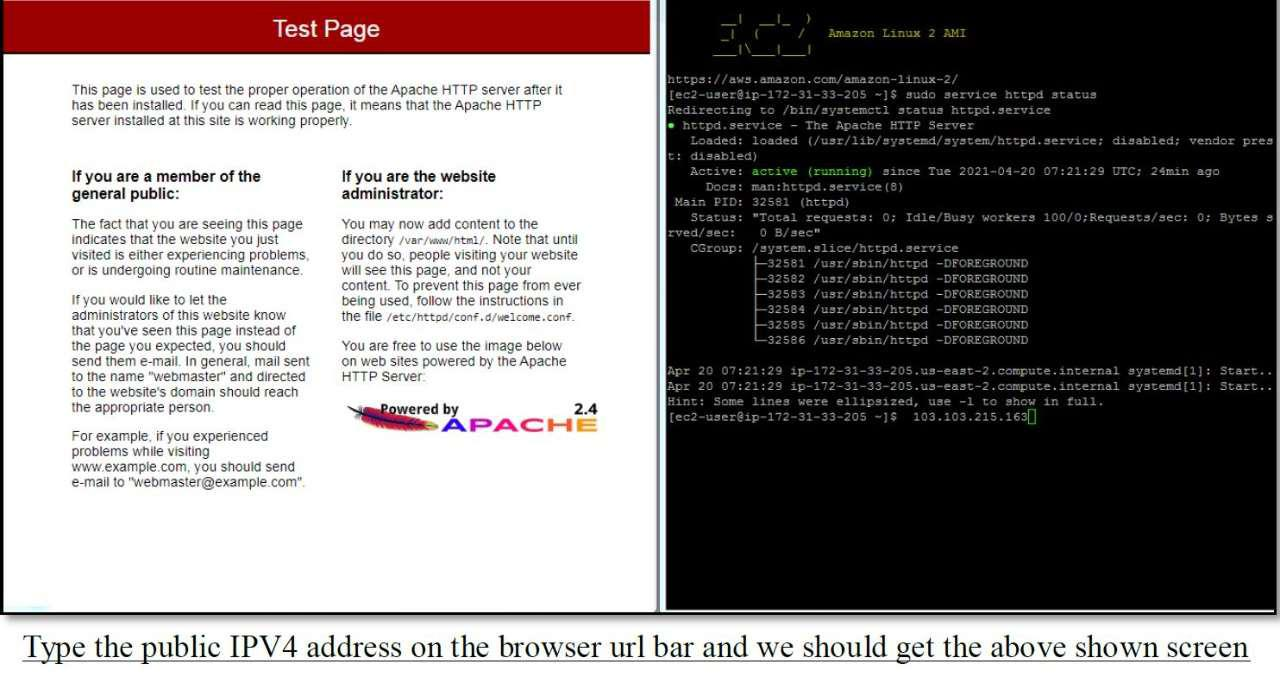
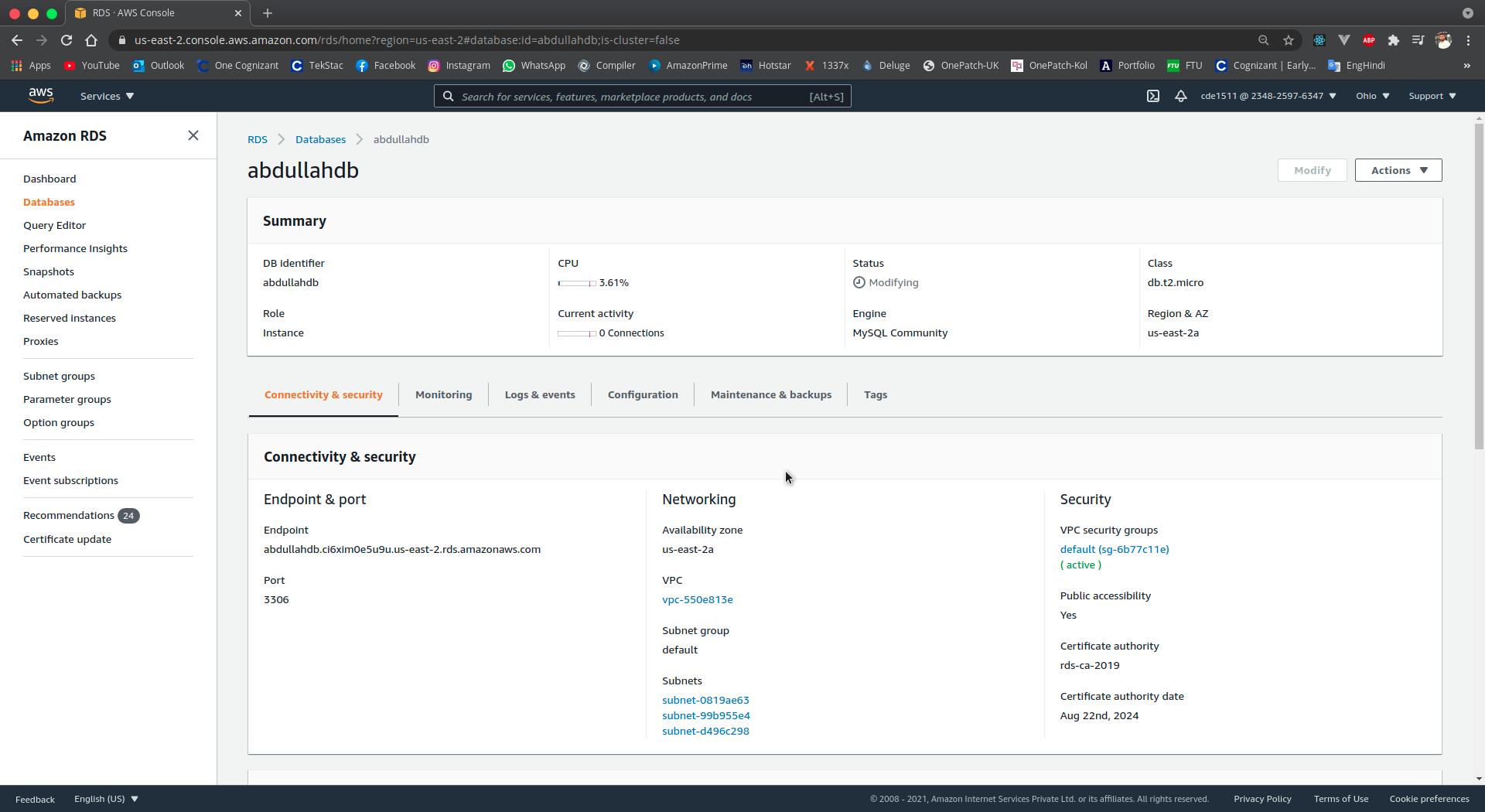
**AWS Hands-on**

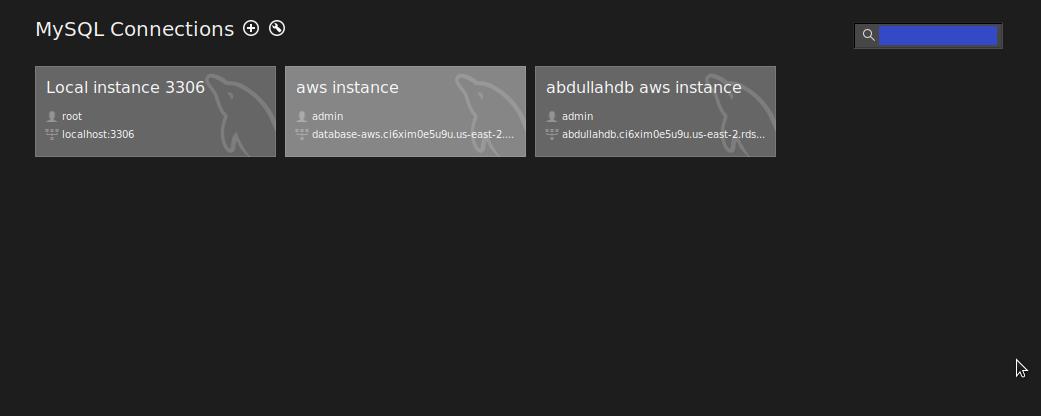
**EC2 Hands-on**

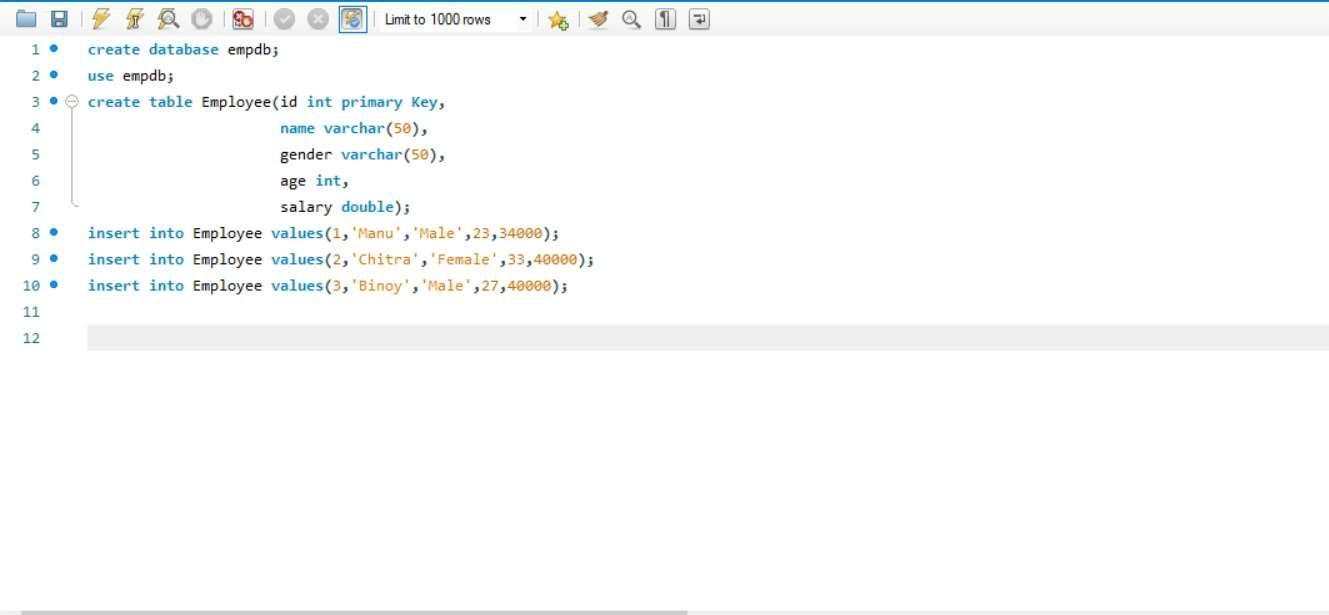
****



**RDS Hands-on**

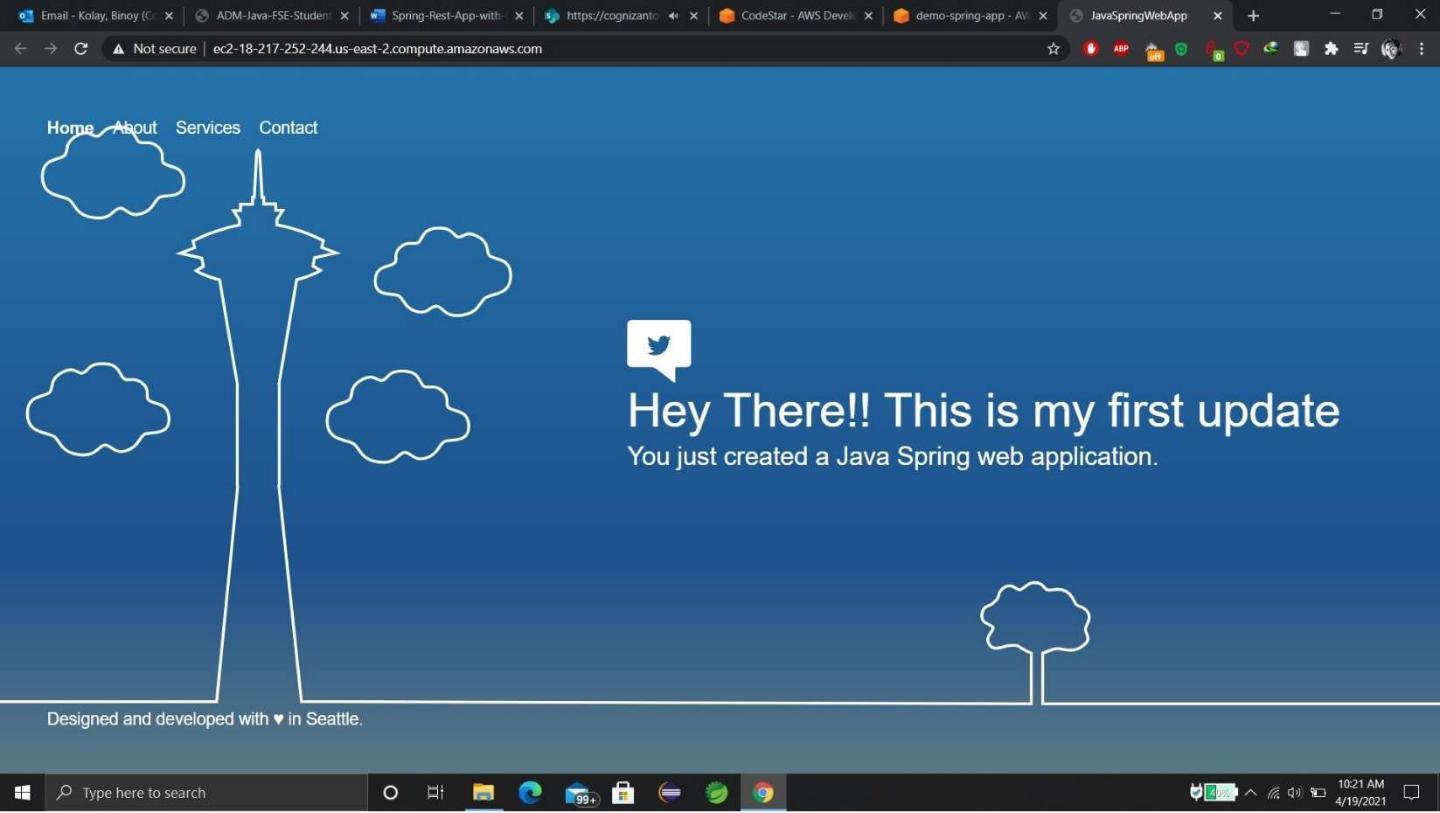
****

****

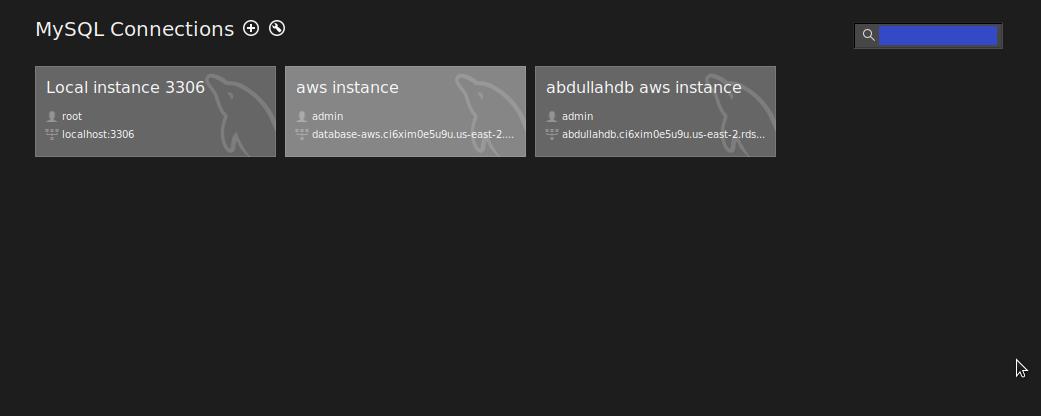


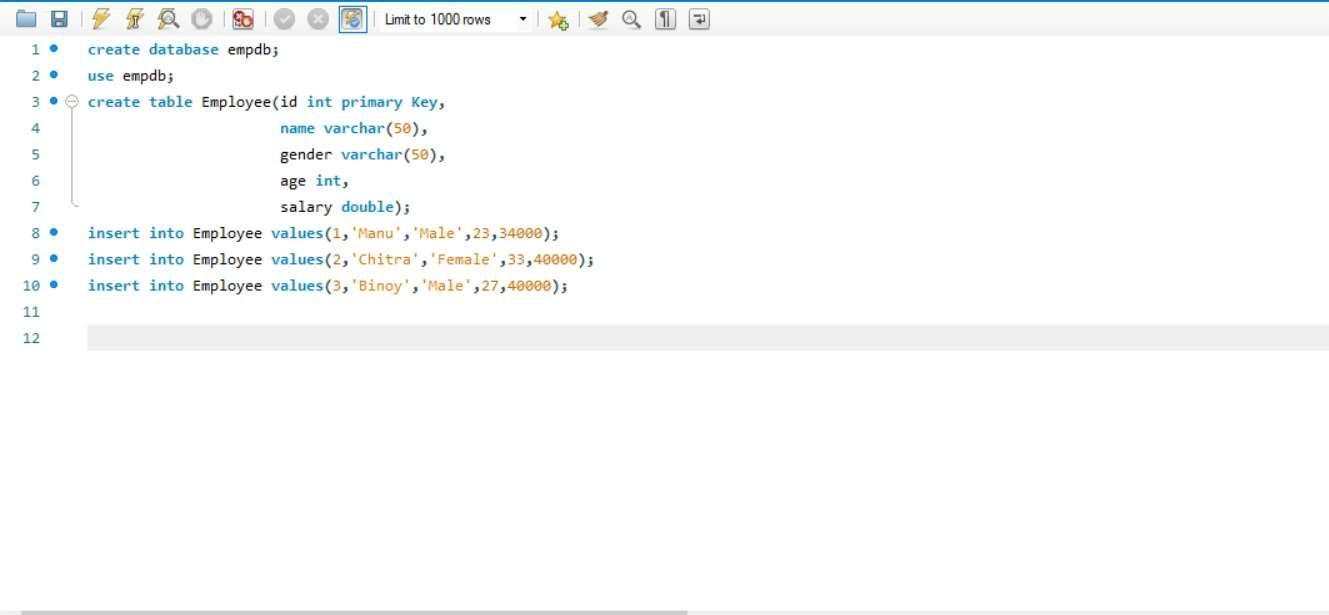
**CI/CD Hands-on**

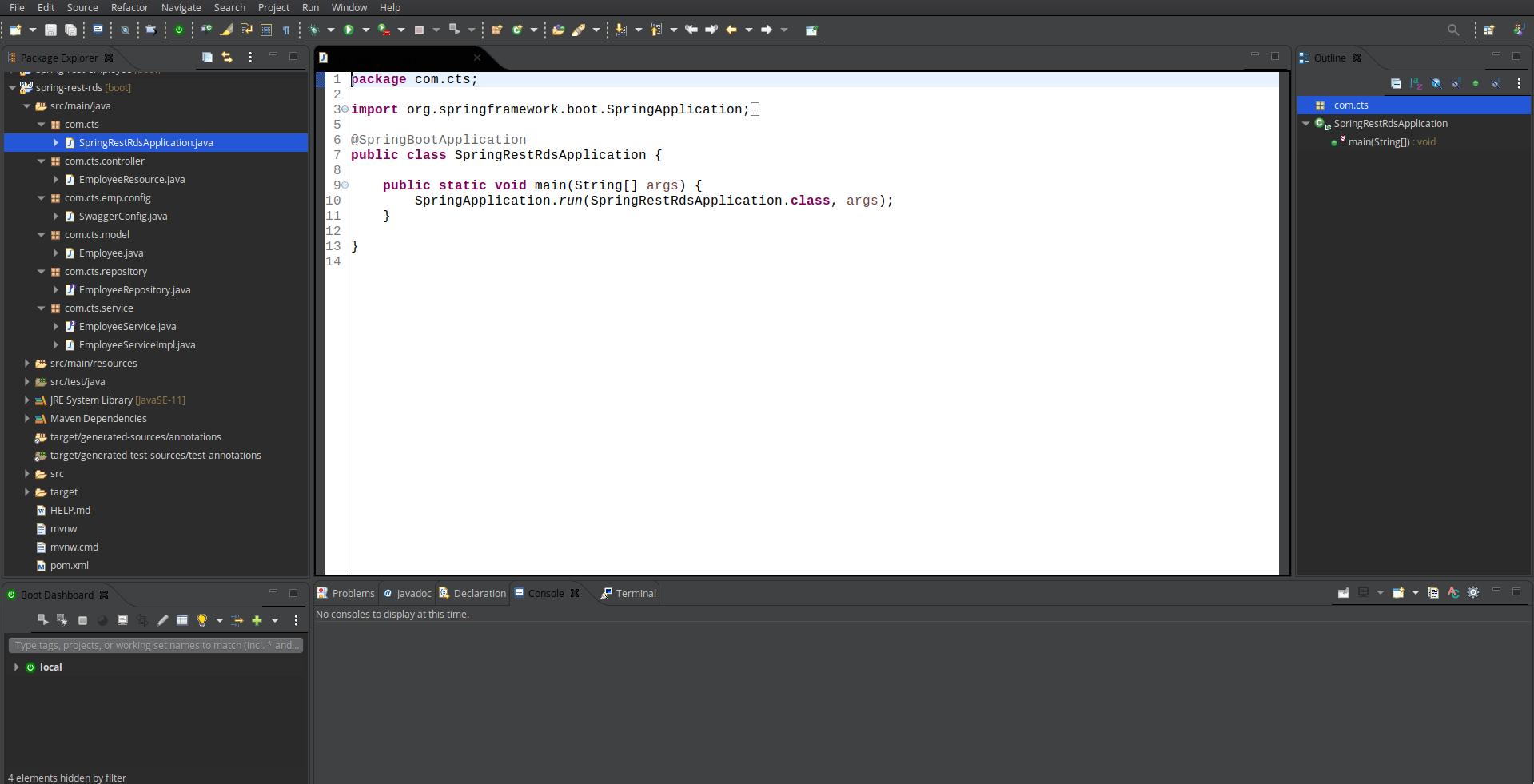
**Output:**



**Spring-Rest-with-RDS-Backend Hands-on**

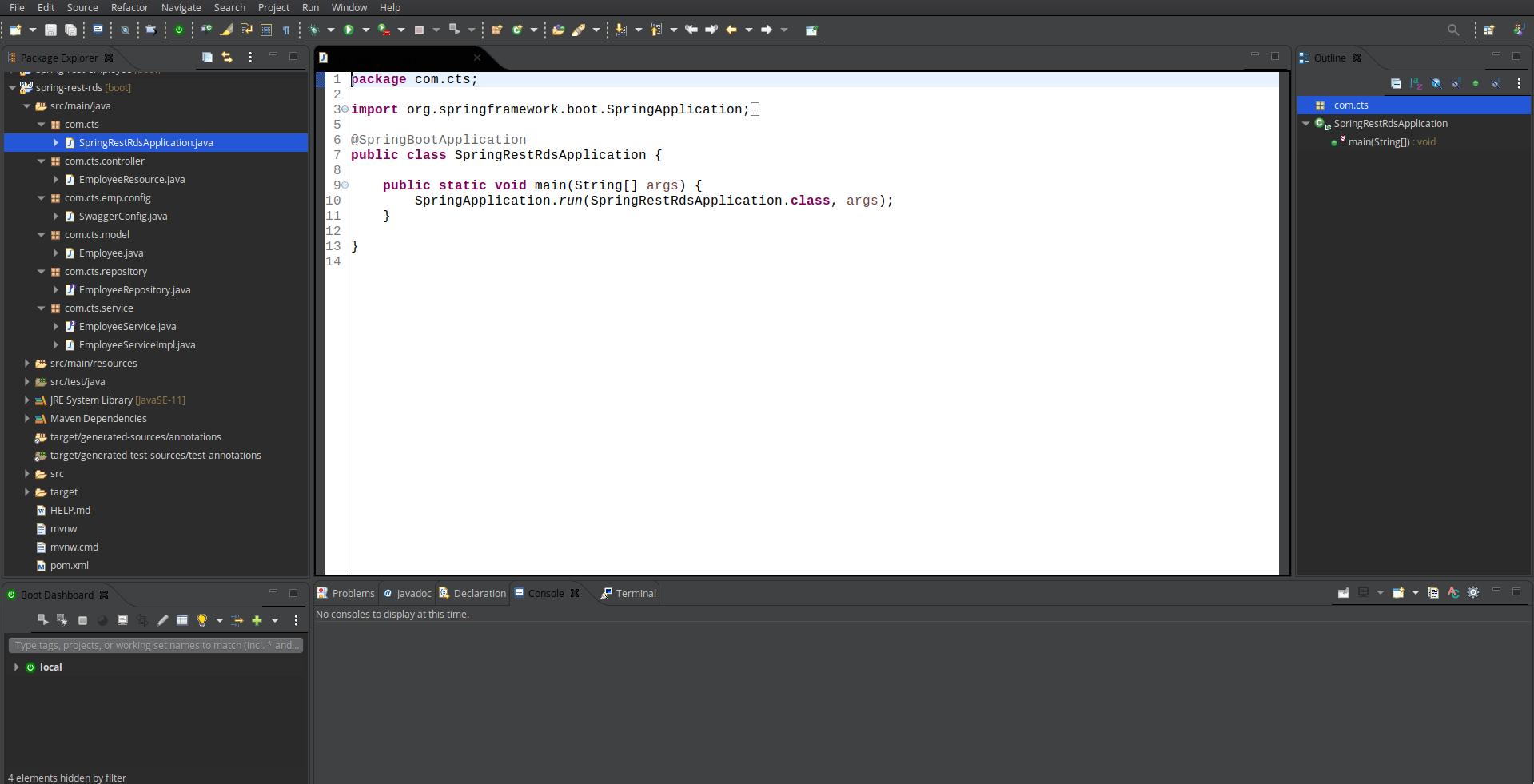
****



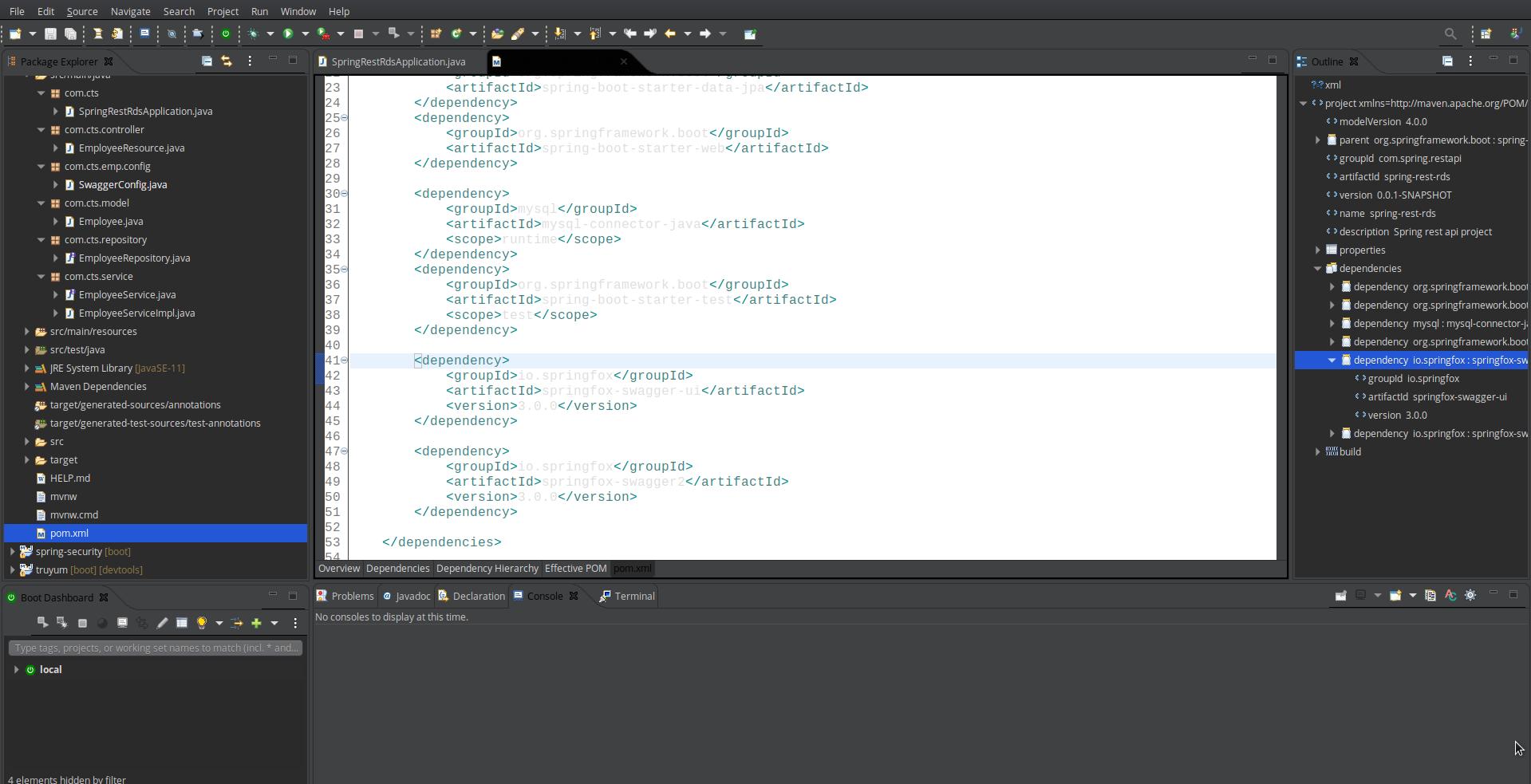
****

**Swagger Hands-on**

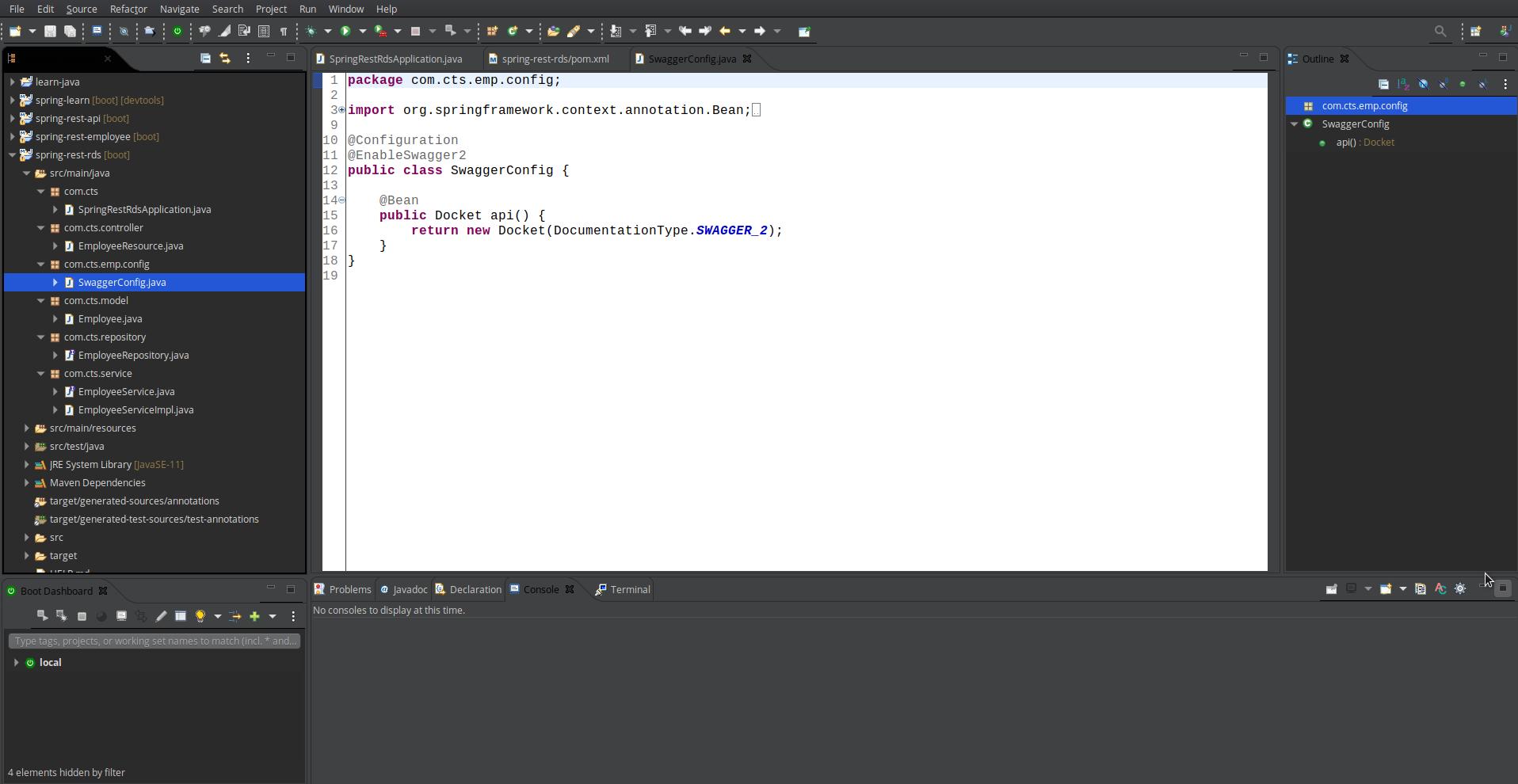
**Step 1:- Create a simple RESTful service using Spring BOOT**

****

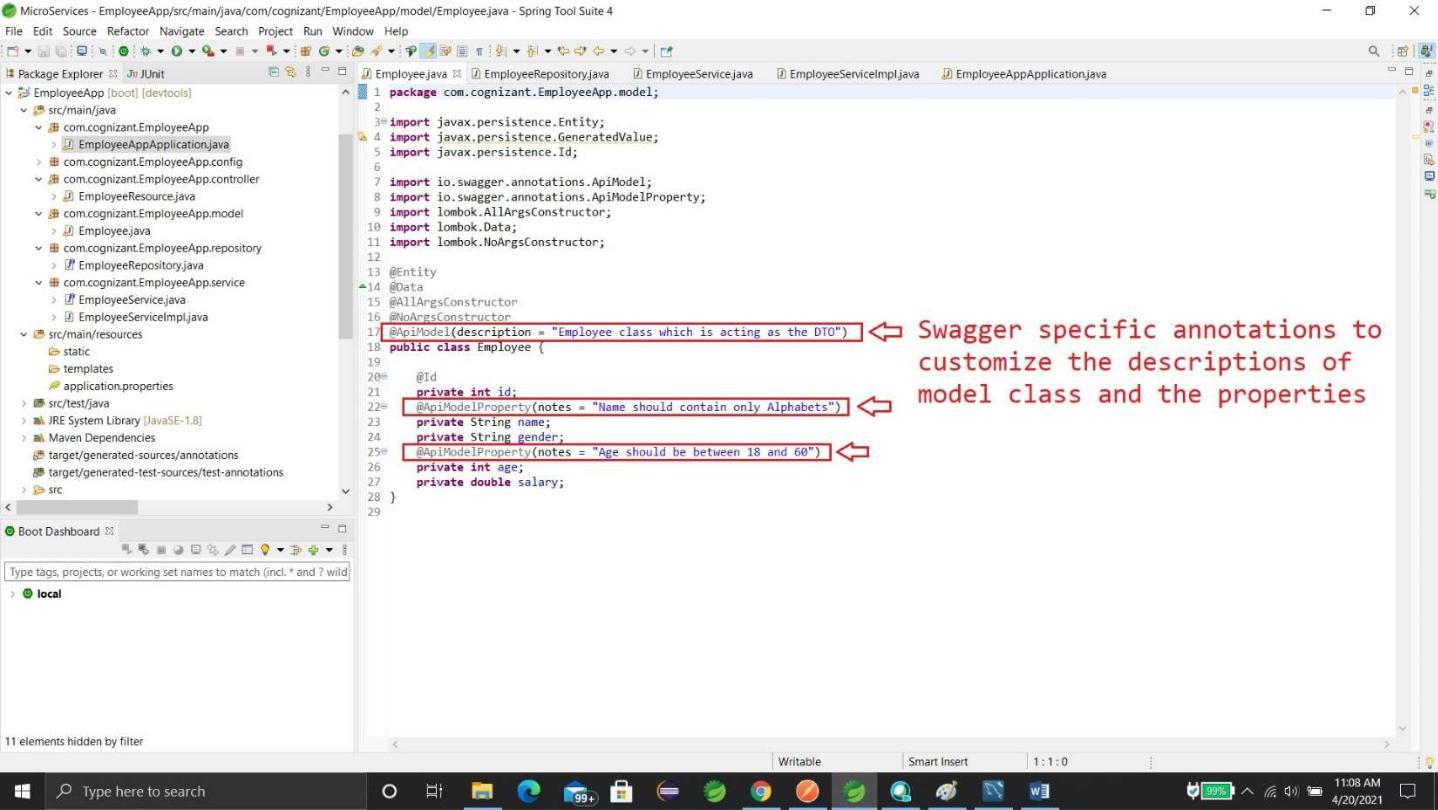
**Step-2:- Add Dependencies**

****

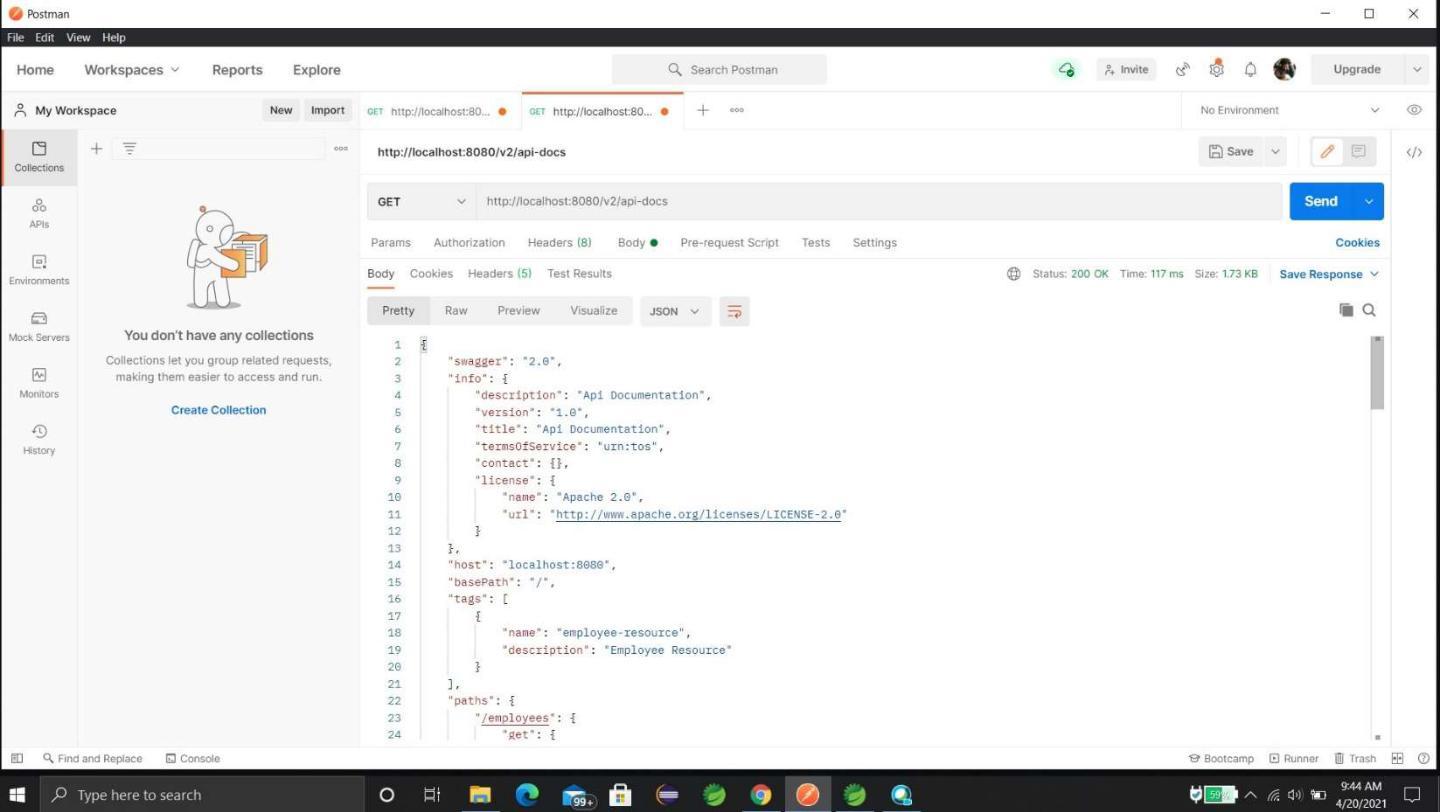
**Step 3:- Create a Swagger configuration class**

****

**Step 4:- use Swagger specific annotations to customize the descriptions of model class and the properties.**

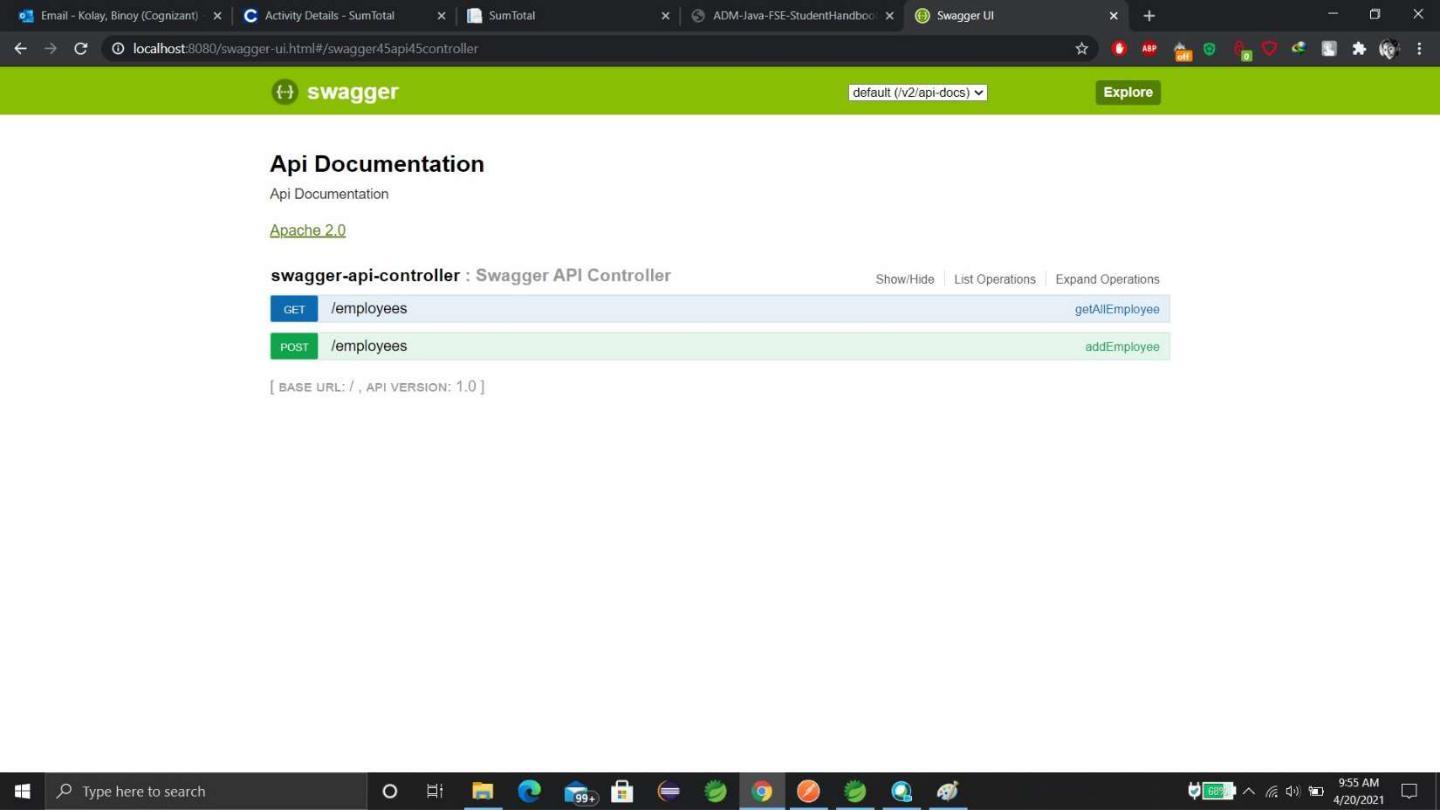


“localhost:8080/v2/api-docs” and you can see the complete API documentation of your service.



Now, hit the URL in your web browser and see the Swagger API functionalities.

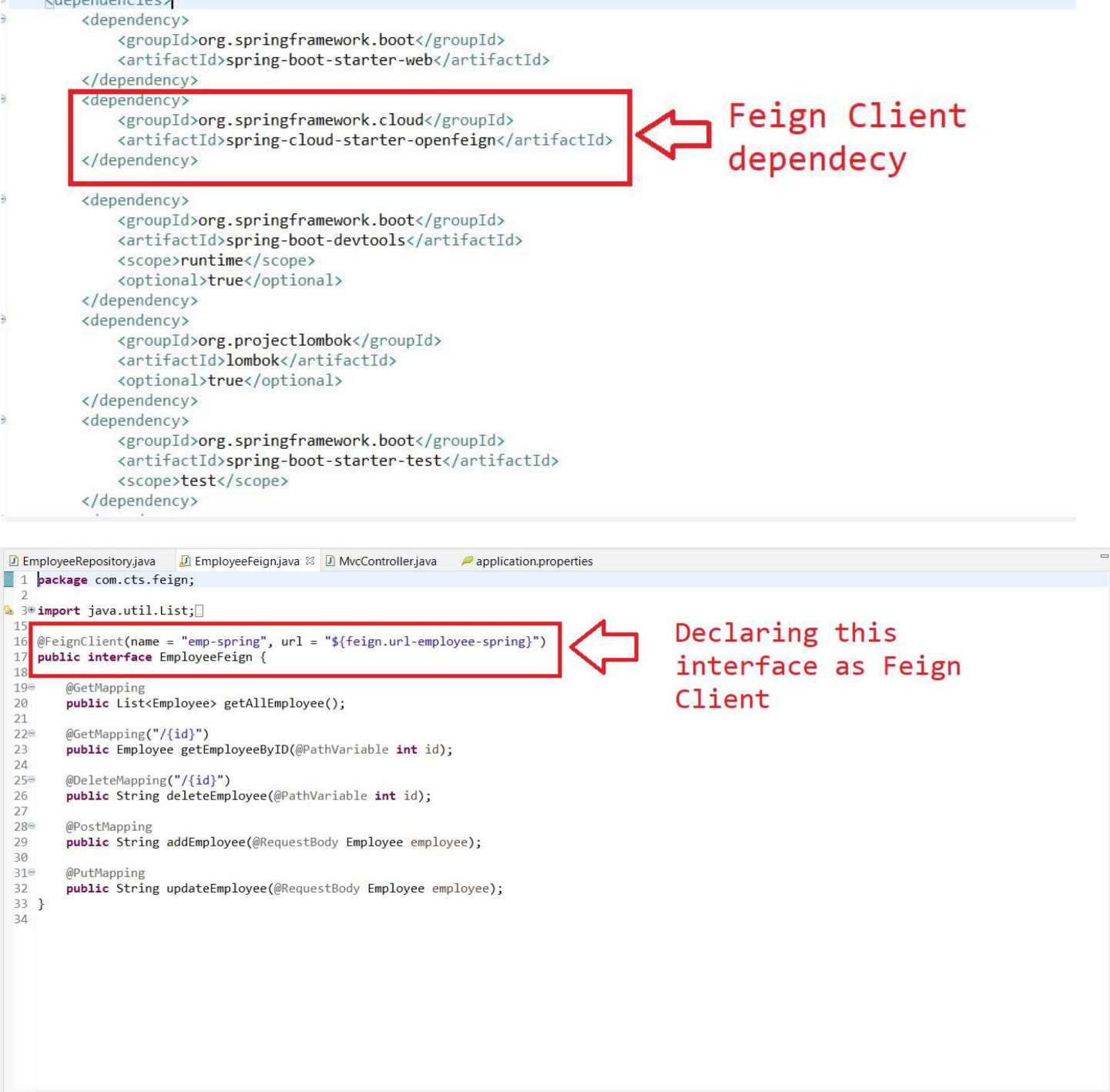
**http://localhost:8080/swagger-ui.html**



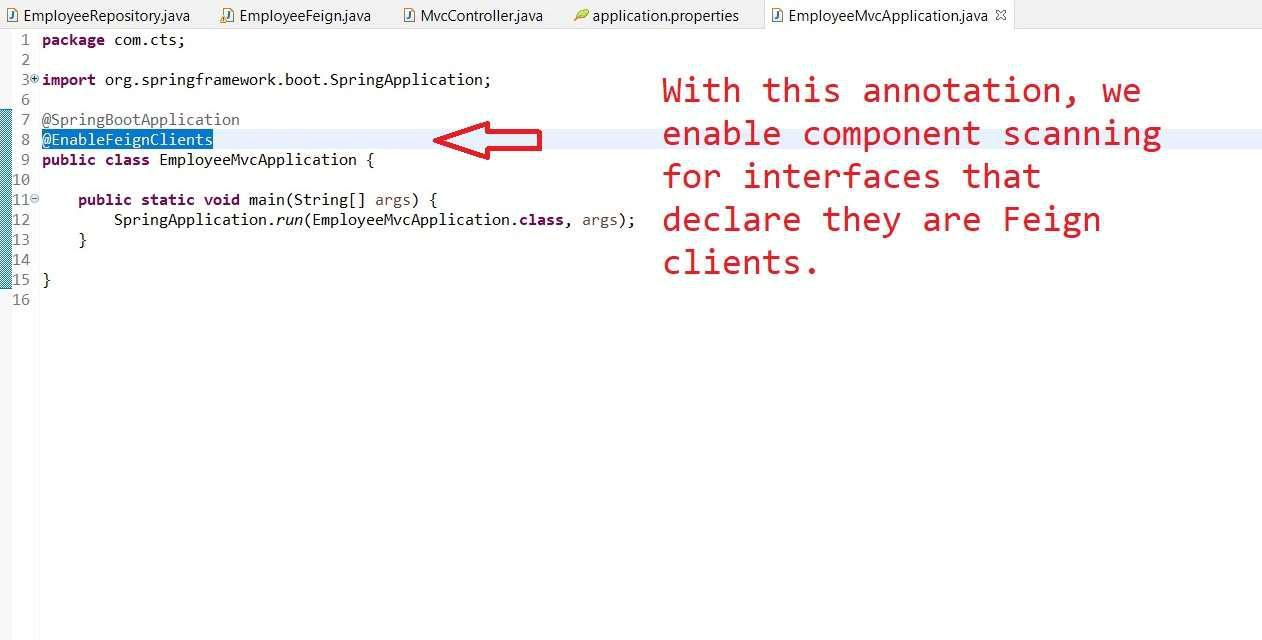
**Spring MVC Client For Spring REST Service**

**Note:- We have already created a microservice(employee) in our local System. Now, we are just creating another microservice which will consume the rest service of our previous employee microservice.**

**First we have to add “openfeign” dependency**



**Note:-The value argument passed in the @FeignClient annotation is a mandatory, arbitrary client name, while with the url argument, we specify the API base URL. Furthermore, since this interface is a Feign client, we can use the Spring Web annotations to declare the APIs that we want to reach out to.**

**base URL.**

With this annotation, we enable component scanning for interfaces that declare they are Feign clients.

**Output:**

