API TESTING ---LEARN AUTOMATION ONLINE DATE:02-09-2022

For set up json and NODEJS(Preconfiguration) 1.https://www.youtube.com/watch?v=4g2X7\_Z9BE4&t=0s

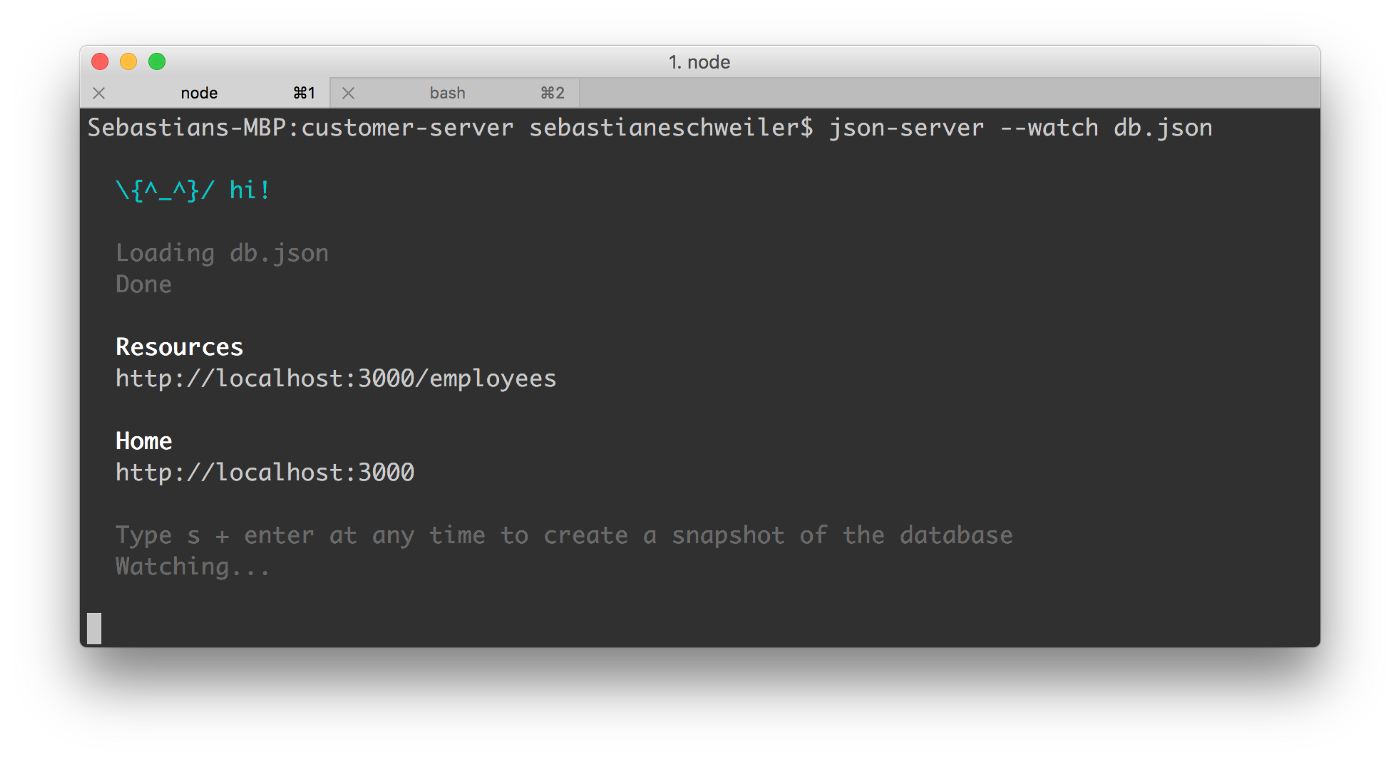
2.https://medium.com/codingthesmartway-com-blog/create-a-rest-api-with-json-server-36da8680136d

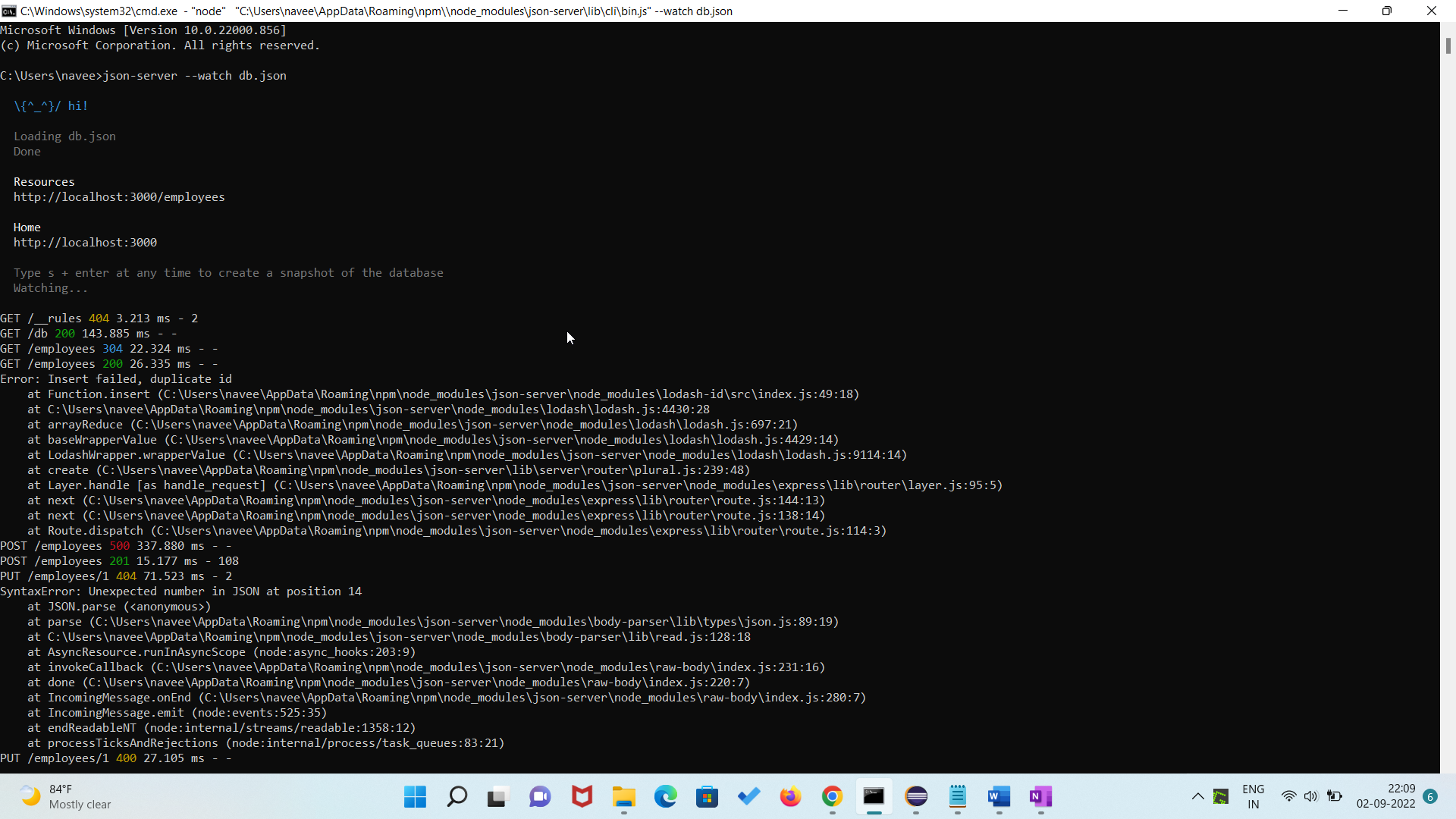
# Installing JSON Server

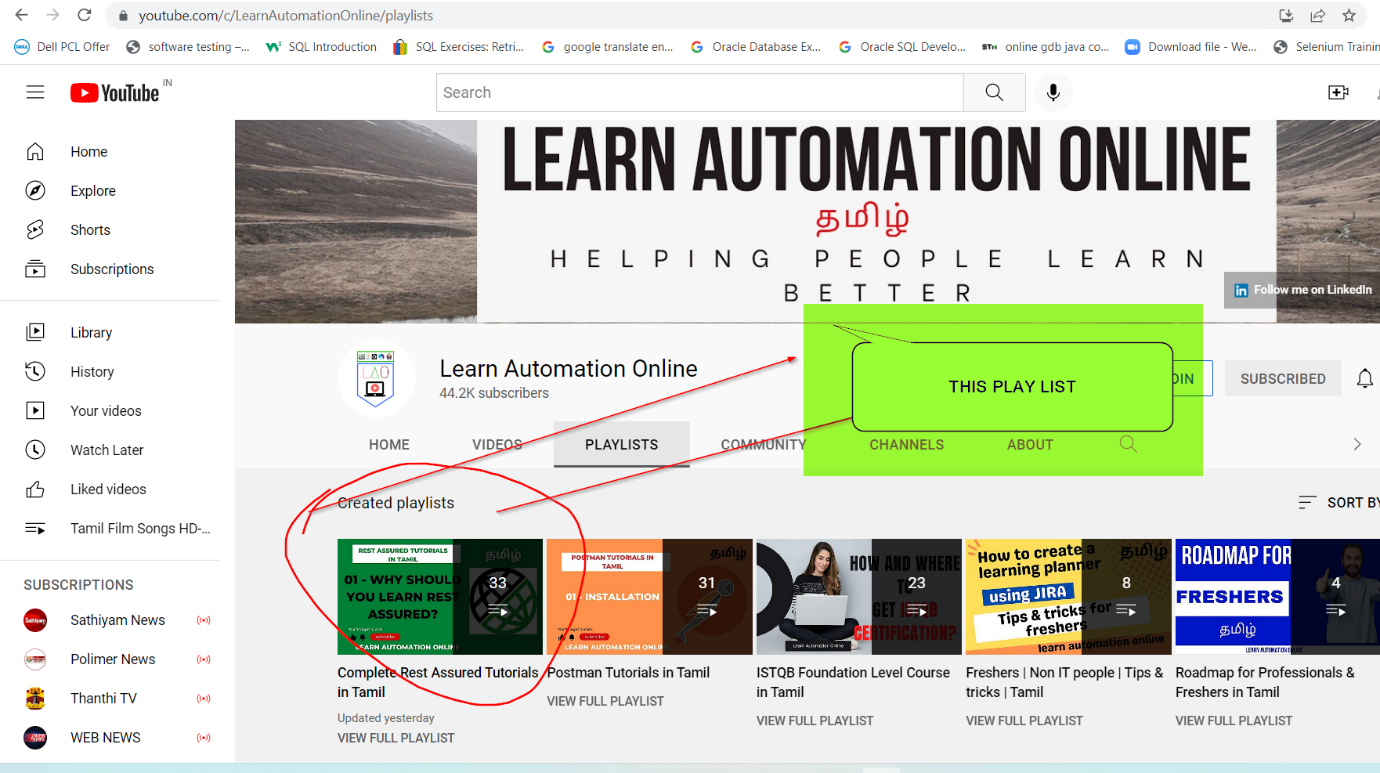
$ npm install -g json-server

**Running The Server**

$ json-server --watch db.json







DEPENDENCIES:

<dependencies>

<!-- https://mvnrepository.com/artifact/io.rest-assured/rest-assured -->

<dependency>

<groupId>io.rest-assured</groupId>

<artifactId>rest-assured</artifactId>

<version>5.0.1</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.6.1</version>

<configuration>

<source>1.8</source>

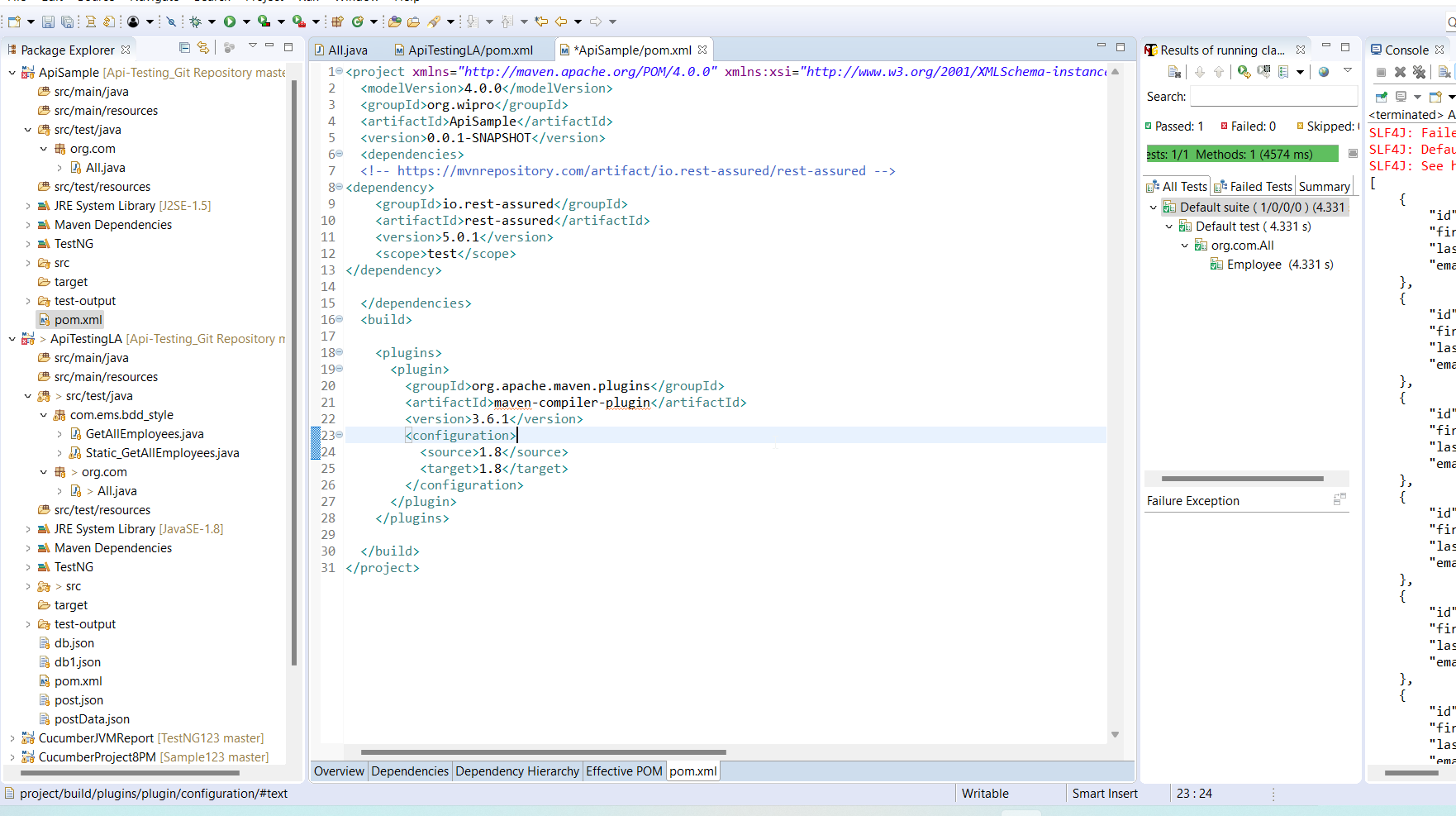
<target>1.8</target>

</configuration>

</plugin>

</plugins>

</build>



**package** org.com;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**import** io.restassured.http.Method;

**import** io.restassured.response.Response;

**import** io.restassured.specification.RequestSpecification;

**public** **class** All {

@Test

**private** **void** Employee() **throws** InterruptedException {

RestAssured.*baseURI*="http://localhost:3000/";

Thread.*sleep*(3000);

RequestSpecification requestSpecification= RestAssured.*given*();

Response response= requestSpecification.request(Method.***GET***,"employees");

System.***out***.println(response.asPrettyString());

System.***out***.println(response.getStatusLine());

}

@Test(enabled=**false**)

**private** **void** Employee1() {

RestAssured.*baseURI*="http://localhost:3000/";

RequestSpecification requestSpecification=

RestAssured.*given*()

.header("Content-Type","application/json")

.body("{\r\n" +

" \"id\": 5,\r\n" +

" \"first\_name\": \"Ann41\",\r\n" +

" \"last\_name\": \"Smith14\",\r\n" +

" \"email\": \"ann14@codingthesmartway.com\"\r\n" +

" }");

Response response= requestSpecification.request(Method.***POST***,"employees");

System.***out***.println(response.asPrettyString());

System.***out***.println(response.getStatusLine());

}

@Test(enabled=**false**)

**public** **void** Employee2() {

RestAssured.*baseURI*="http://localhost:3000";

RequestSpecification requestSpecification=

RestAssured.*given*()

.header("Content-Type","application/json")

.body("{\r\n" +

" \"id\": 3,\r\n" +

" \"first\_name\": \"Ann41\",\r\n" +

" \"last\_name\": \"Smith14\",\r\n" +

" \"email\": \"ann14@codingthesmartway.com\"\r\n" +

" }");

Response response= requestSpecification.request(Method.***PUT***,"/employees/1");

System.***out***.println(response.asPrettyString());

System.***out***.println(response.getStatusLine());

}

@Test(enabled=**false**)

**public** **void** deleteAnEmployee() {

RestAssured.*baseURI*="http://localhost:3000/";

RequestSpecification requestSpecification= RestAssured.*given*();

Response response= requestSpecification.request(Method.***DELETE***,"employees/1");

System.***out***.println(response.asPrettyString());

}

@Test(enabled=**false**)

**public** **void** getAnEmployee() {

RestAssured.*baseURI*="http://localhost:3000/";

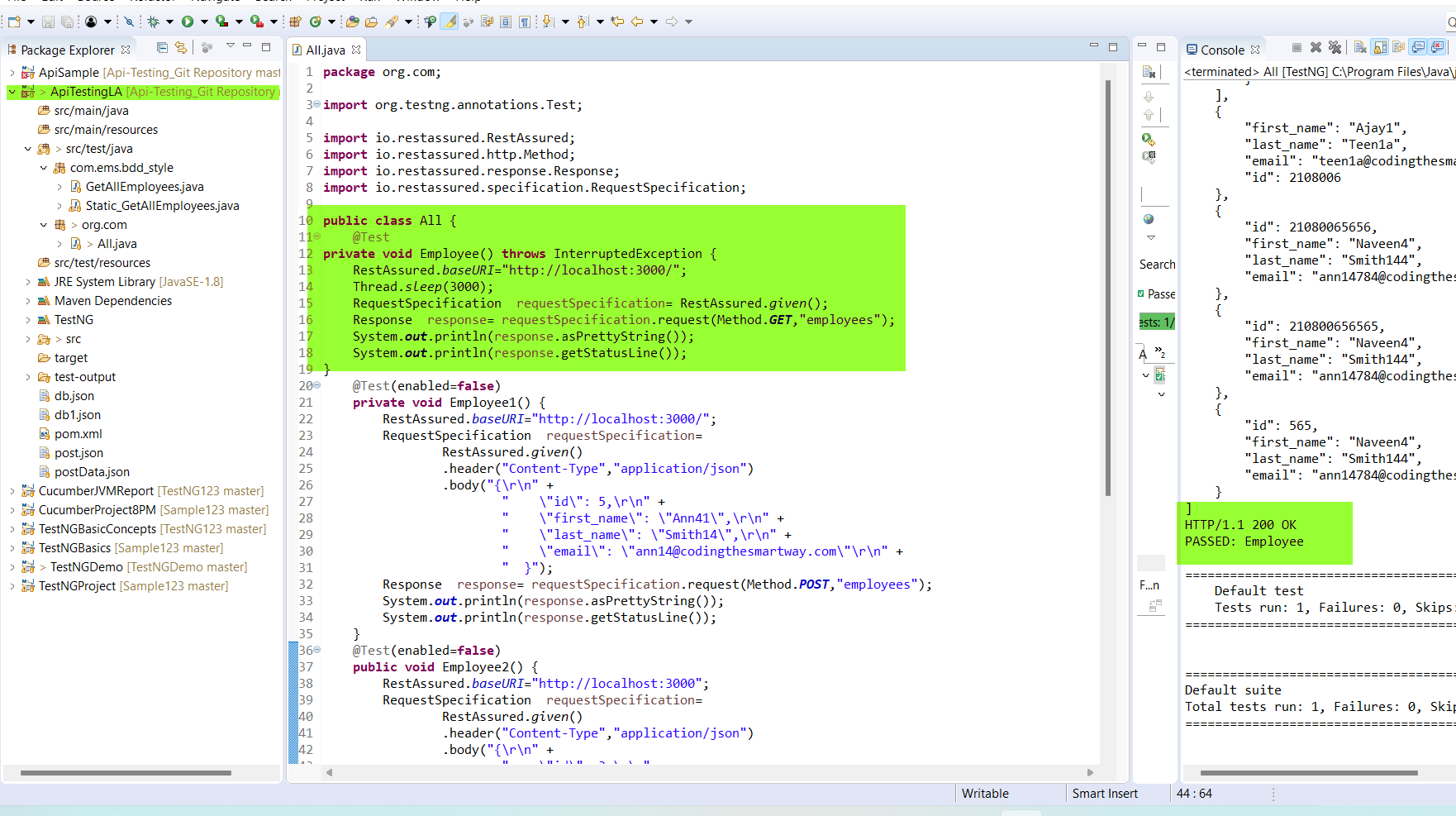
RequestSpecification requestSpecification= RestAssured.*given*();

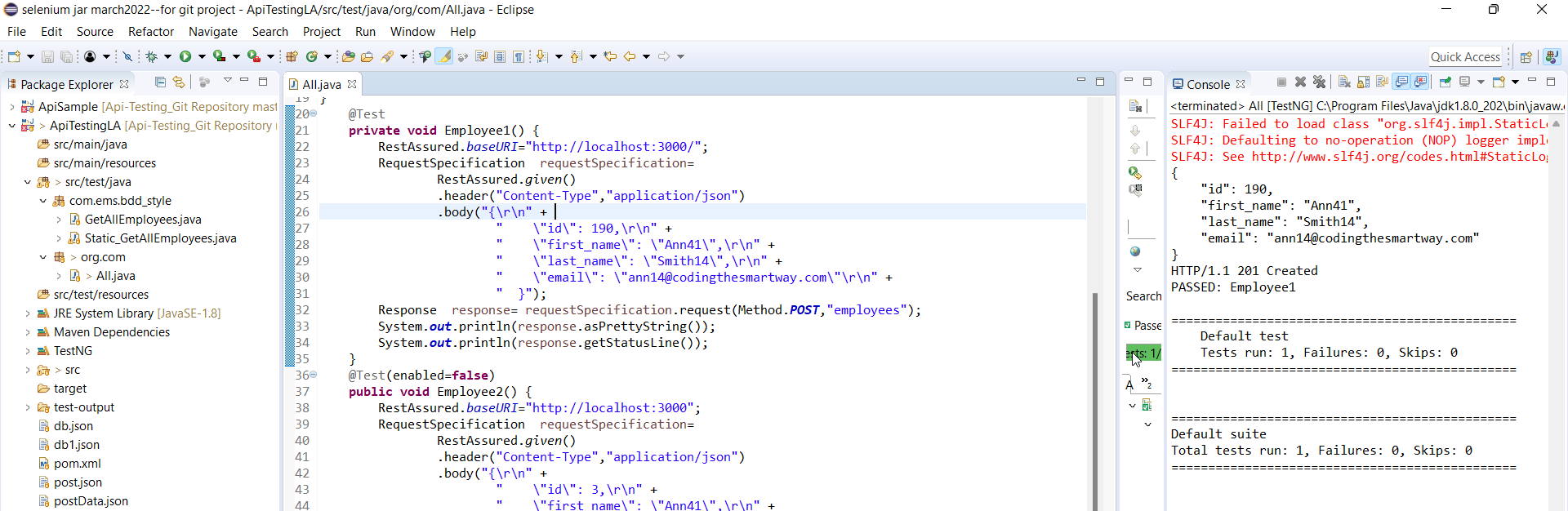
Response response= requestSpecification.request(Method.***GET***,"employees/5");

System.***out***.println(response.asPrettyString());

}

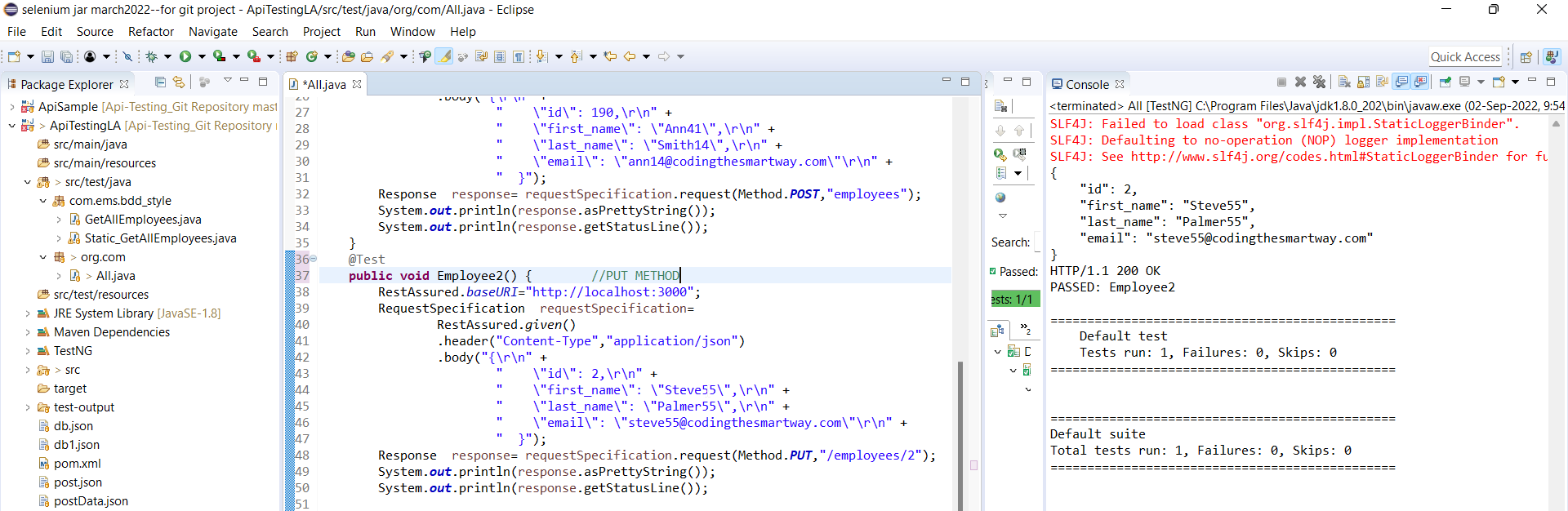
}



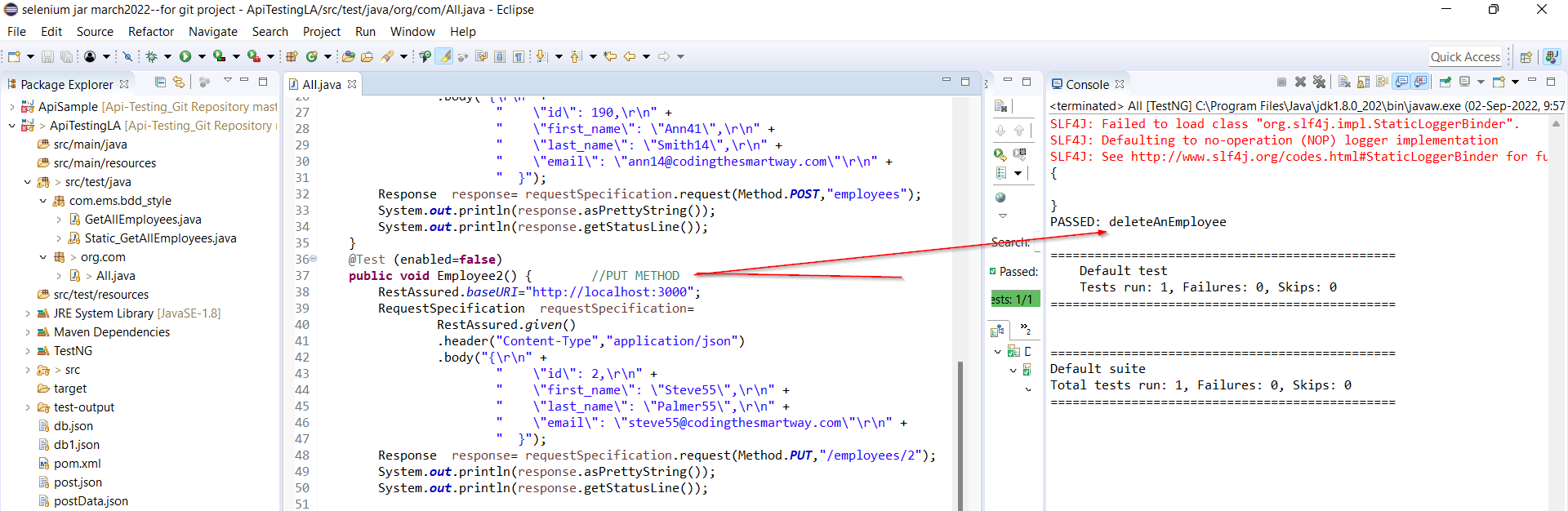


PUT METHOD:

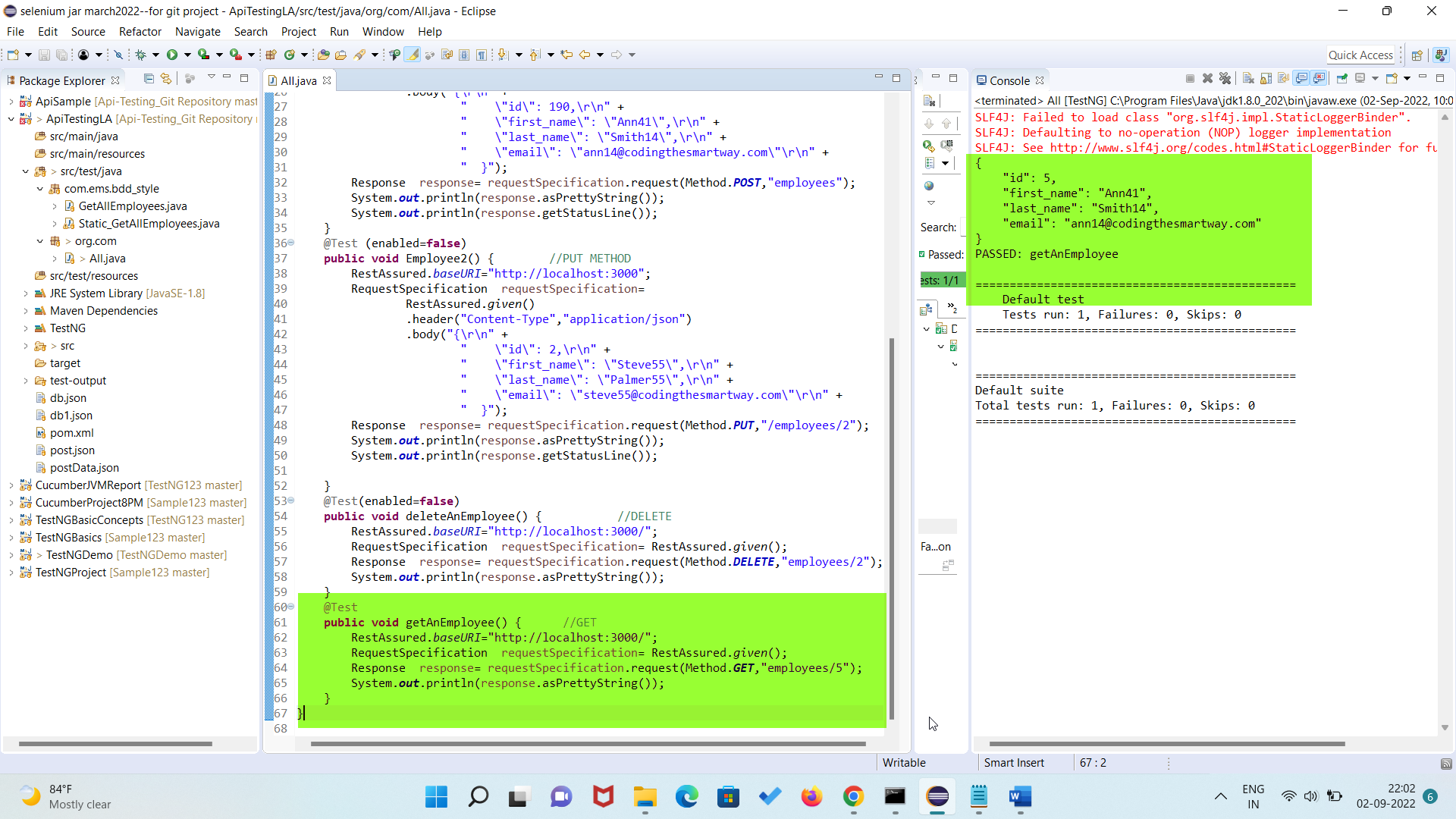
TO UPDATE THE VALUE



DELETE:

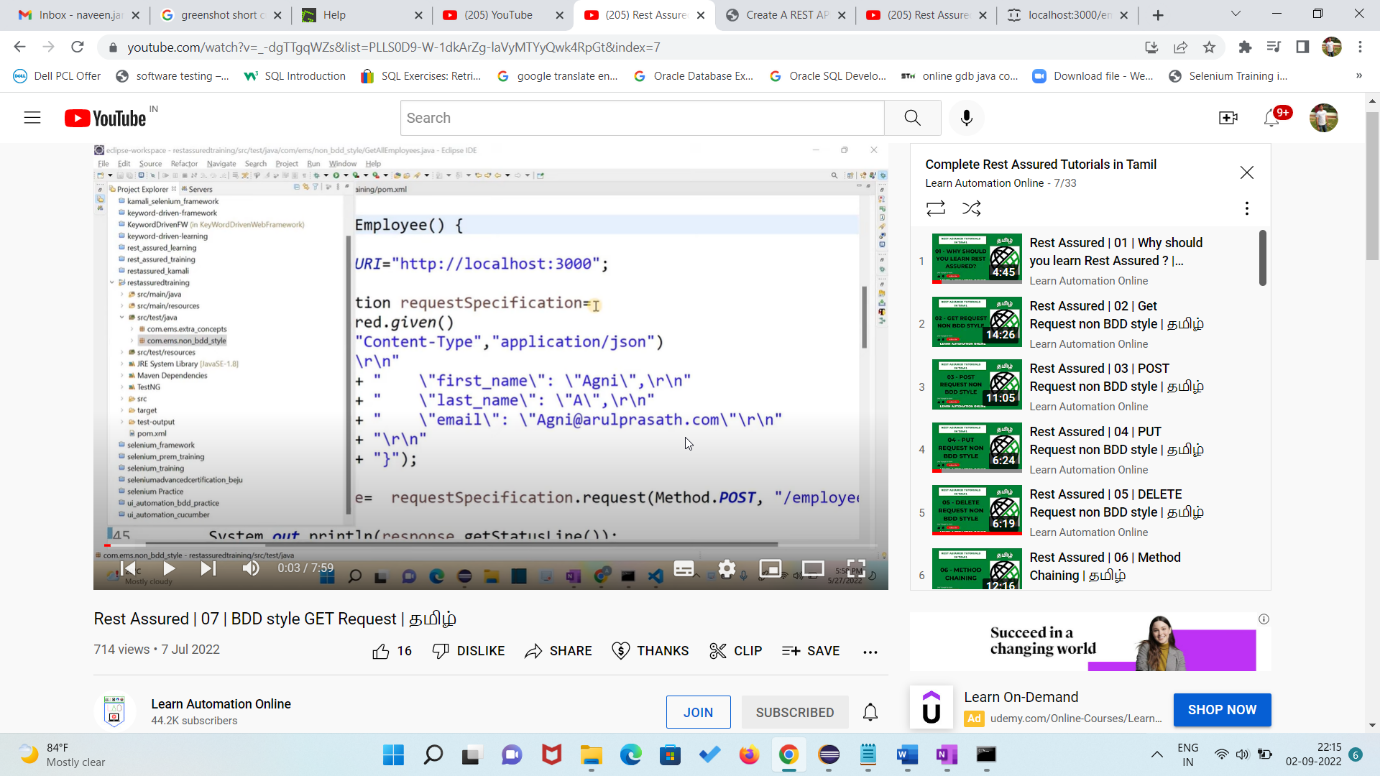


GET:



BDD STYLE-METHOD CHAINING:’

# Rest Assured | 07 | BDD style GET Request | தமிழ்



**package** com.ems.bdd\_style;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**public** **class** GetAllEmployees {

@Test

**public** **void** getAllEmployees() {

RestAssured

.*given*()

.baseUri("http://localhost:3000")

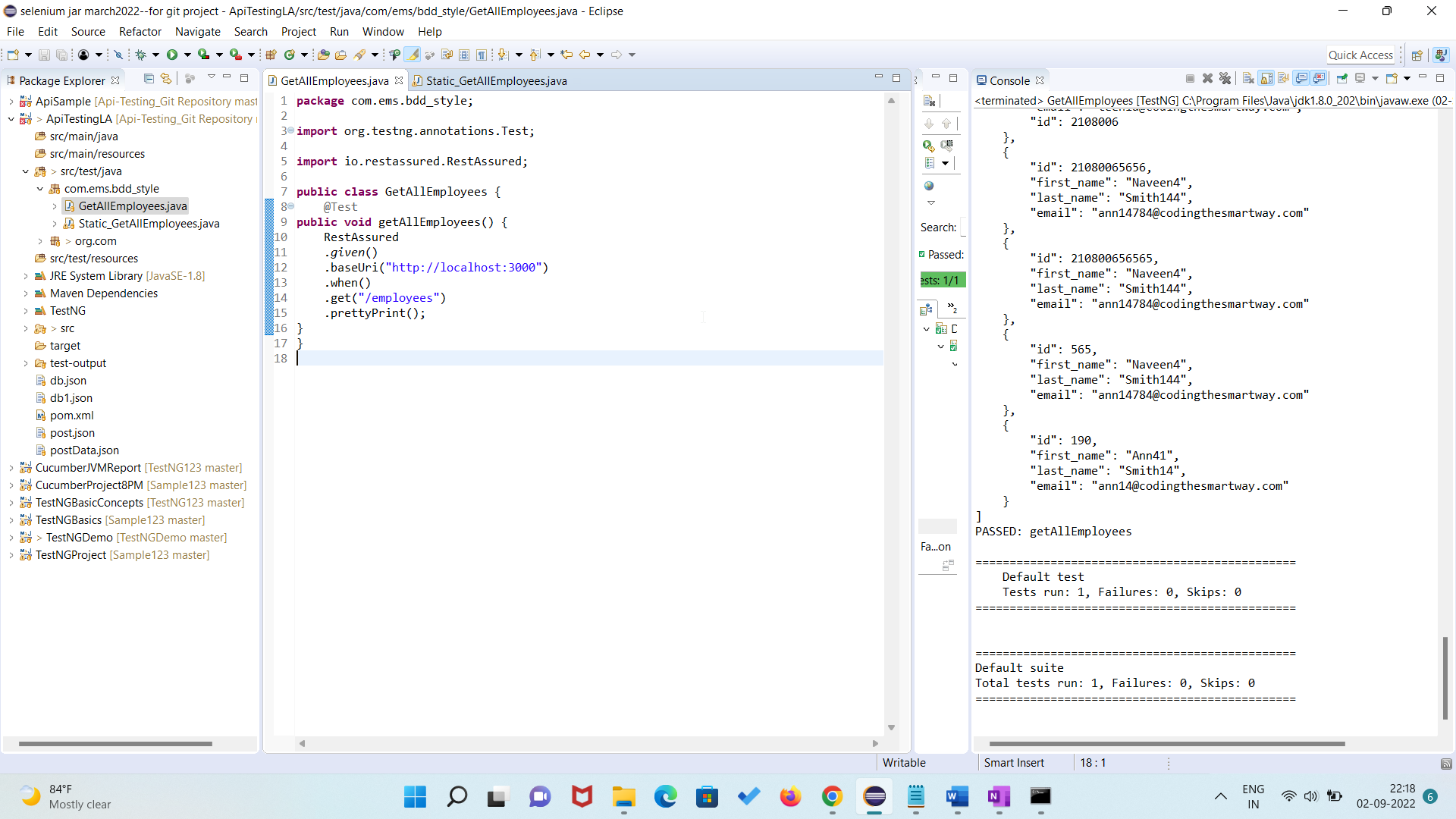
.when()

.get("/employees")

.prettyPrint();

}

}



==========

**package** com.ems.bdd\_style;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**import** io.restassured.specification.RequestSpecification;

**import** **static** io.restassured.RestAssured.\*;

**import** java.io.File;

**public** **class** Static\_GetAllEmployees {

@Test(enabled=**false**)

**public** **void** getAllEmployees() {

*given*()

.baseUri("http://localhost:3000")

.when()

.get("/employees")

.prettyPrint();

}

@Test

**private** **void** createAnEmployeeBDD() {

*given*()

.baseUri("http://localhost:3000")

.header("Content-Type","application/json")

.body("{\r\n" +

" \"id\": 565,\r\n" +

" \"first\_name\": \"Naveen4\",\r\n" +

" \"last\_name\": \"Smith144\",\r\n" +

" \"email\": \"ann14784@codingthesmartway.com\"\r\n" +

" }")

.when()

.post("/employees")

.prettyPrint();

}

@Test(enabled=**false**)

**private** **void** updateAnEmployee() {

*given*()

.baseUri("http://localhost:3000")

.header("Content-Type","application/json")

.body("{\r\n" +

" \"first\_name\": \"Ajay1\",\r\n" +

" \"last\_name\": \"Teen1a\",\r\n" +

" \"email\": \"teen1a@codingthesmartway.com\"\r\n" +

" }")

.when()

.put("/employees/2108006")

.prettyPrint();

}

@Test(enabled=**false**)

**public** **void** deleteAlnEmployees() {

*given*()

.baseUri("http://localhost:3000")

.when()

.delete("/employees/2108006")

.prettyPrint();

}

@Test(enabled=**false**)

**private** **void** createEmployeeFromJson() {

File jsonFile=**new** File("db1.json");

*given*()

.baseUri("http://localhost:3000")

.header("Content-Type","application/json")

.body(jsonFile)

.when()

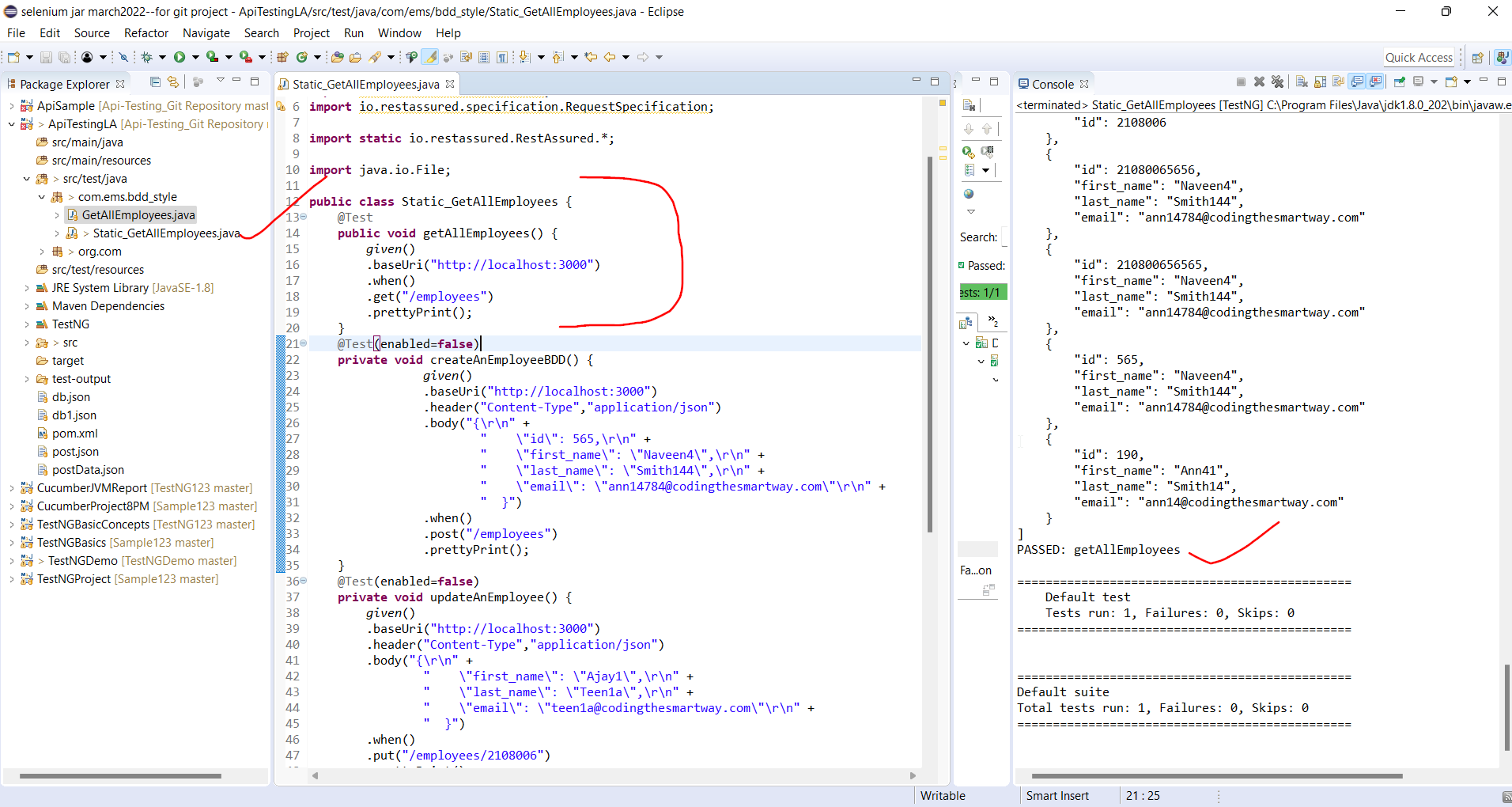
.post("/employees")

.prettyPrint();

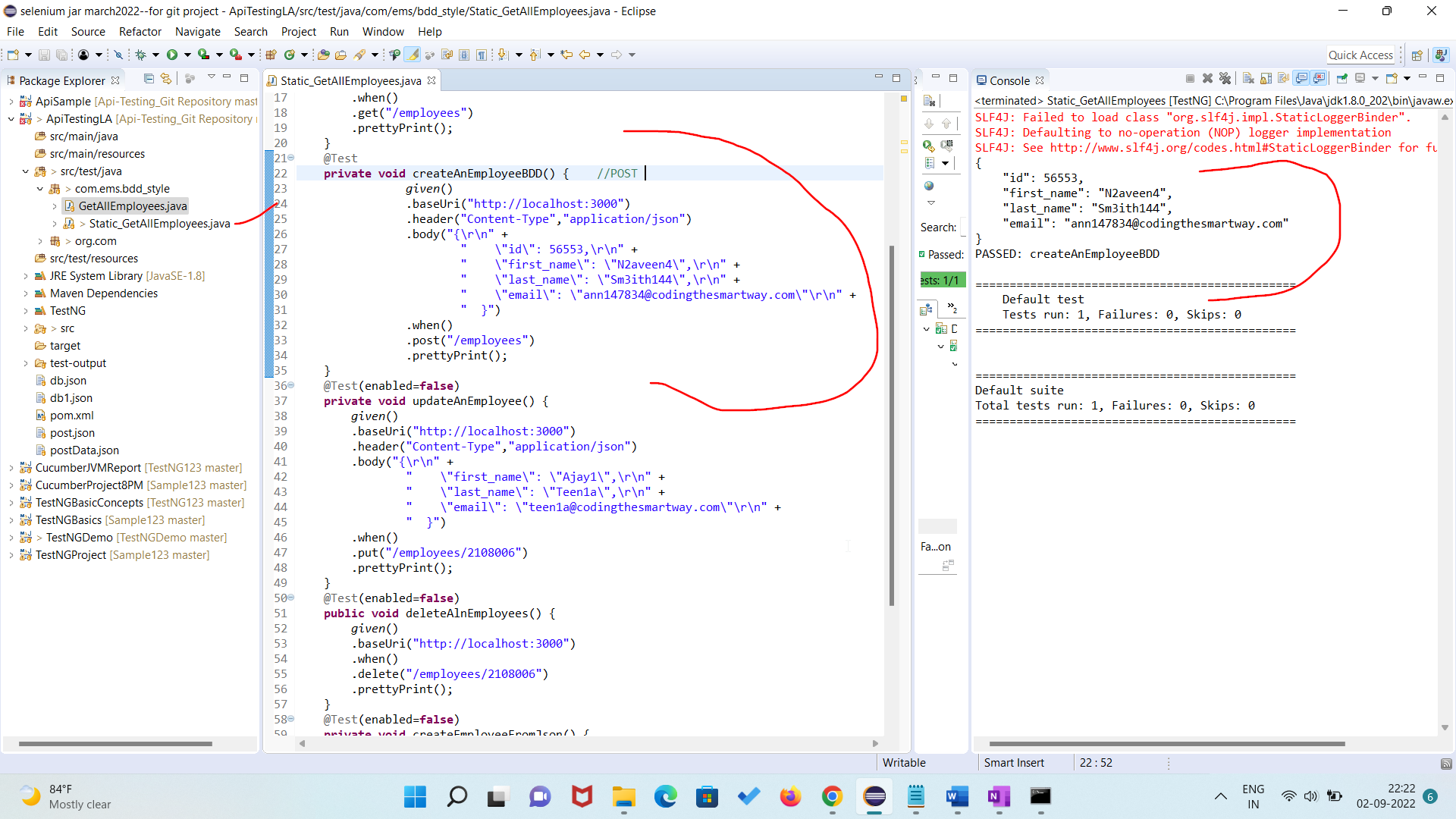
}

}

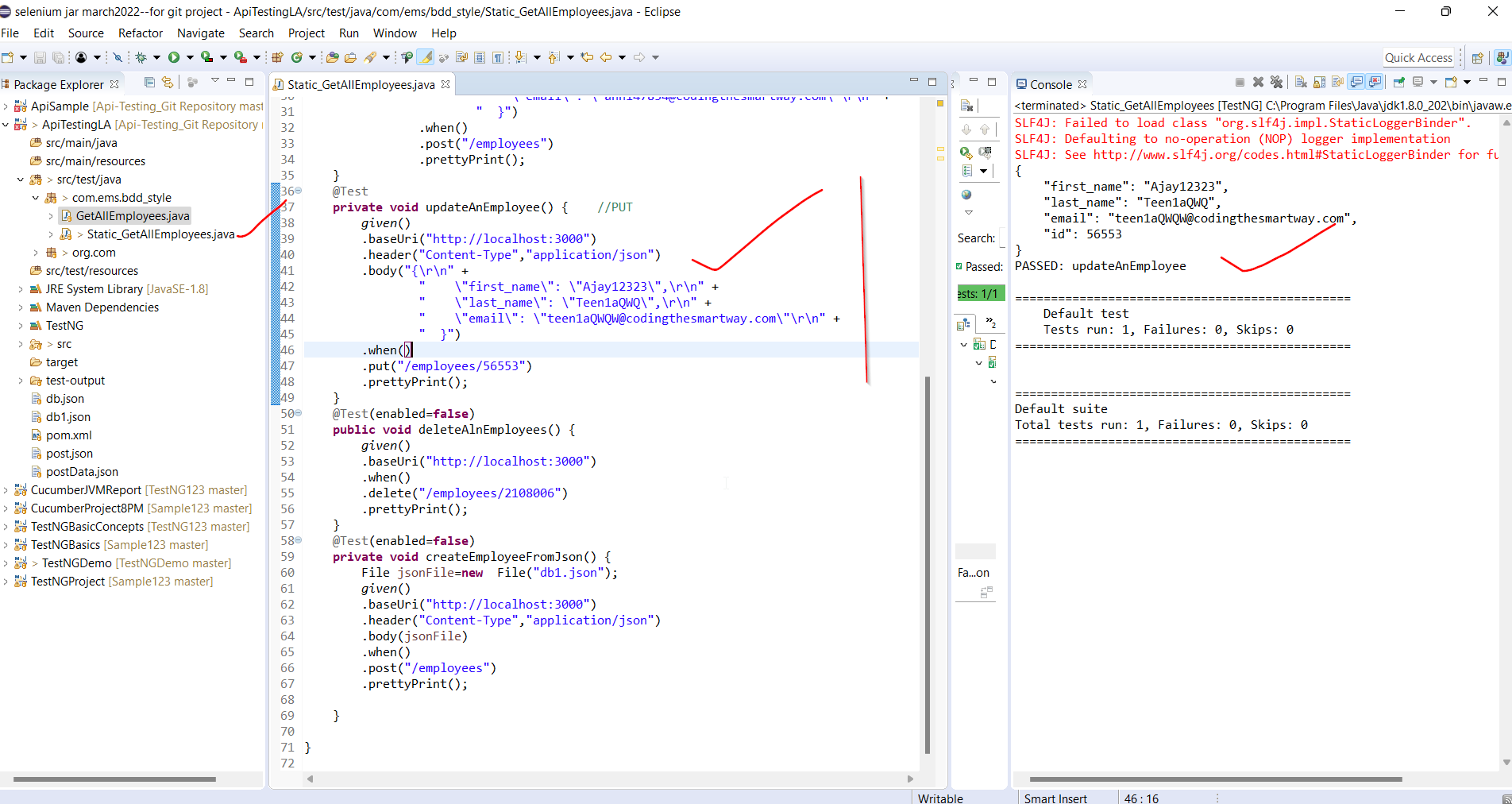
GET:



POST:



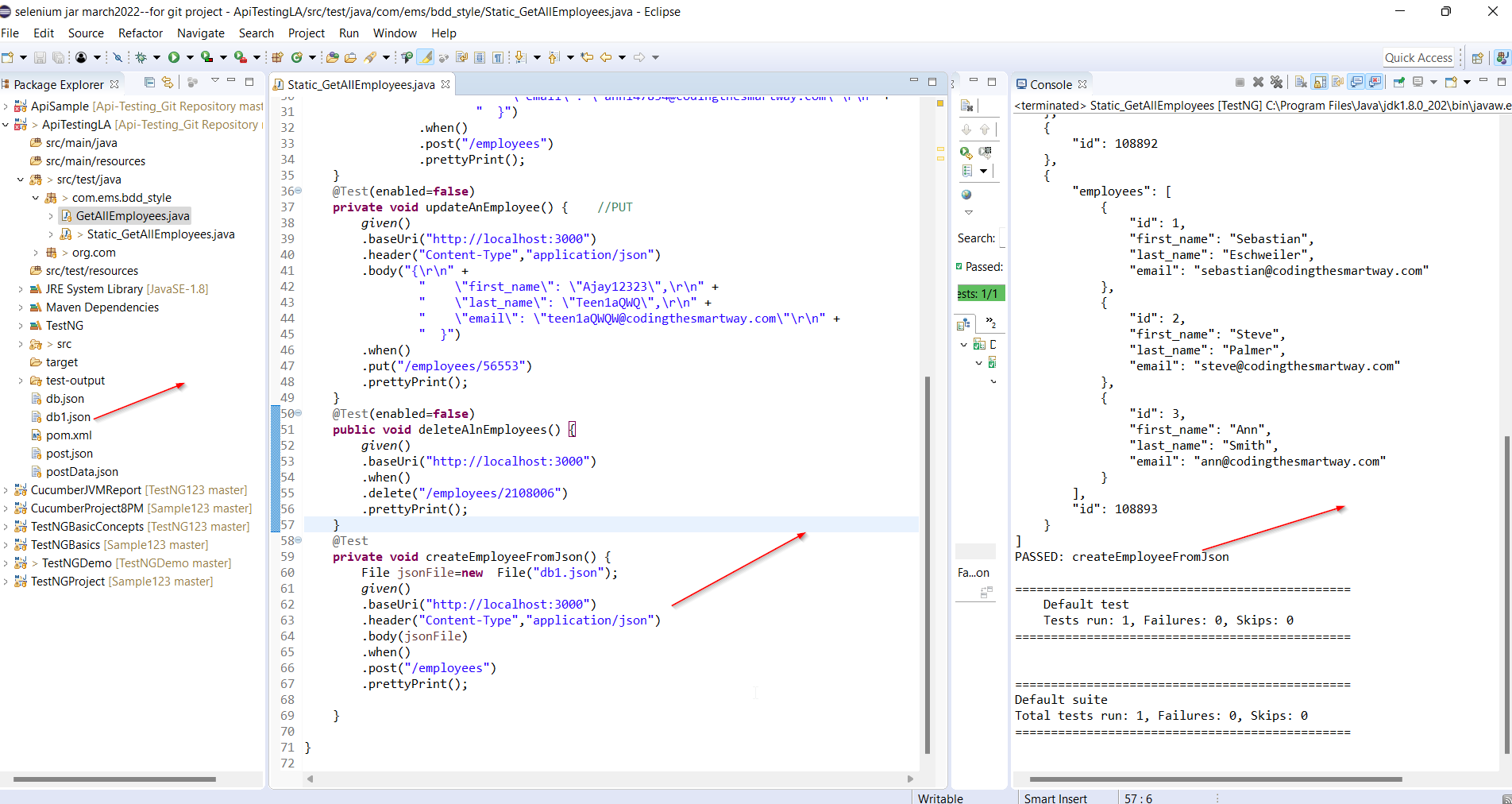
PUT:



DELETE:

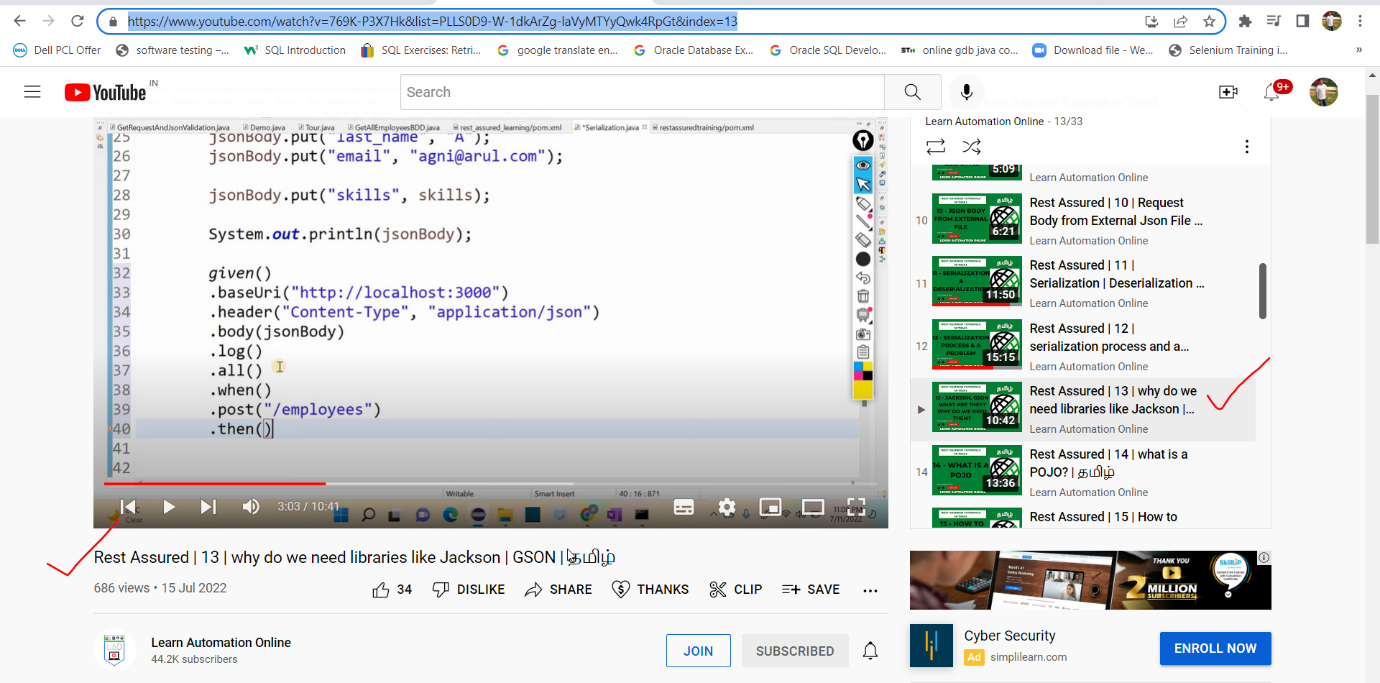


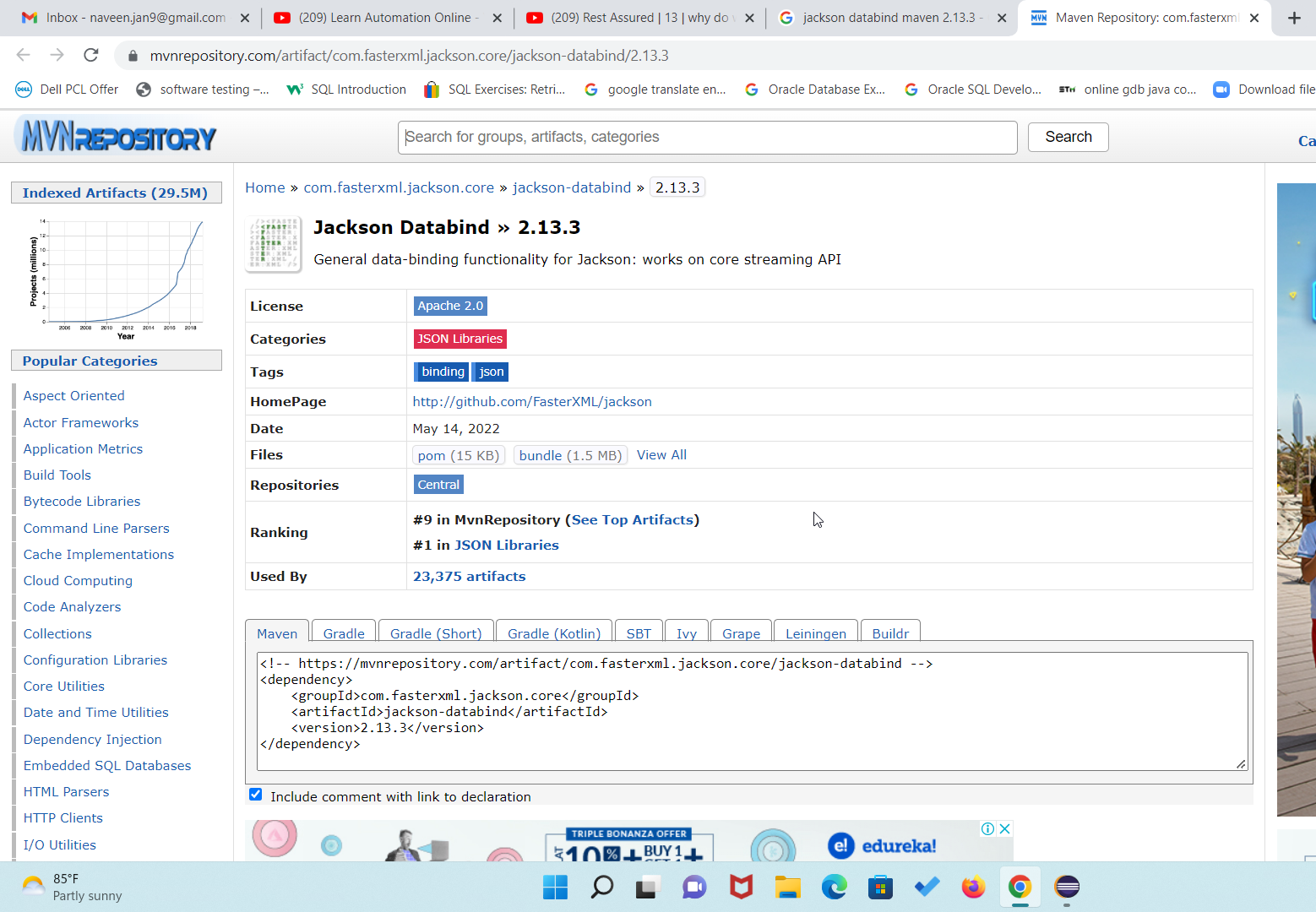
Sending data from json file:

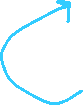


DAY02-- jackson databind maven 2.13.3 dependency

<https://www.youtube.com/watch?v=769K-P3X7Hk&list=PLLS0D9-W-1dkArZg-laVyMTYyQwk4RpGt&index=13>







Add the above dependency this program

**package** co.ems.extraConcepts;

**import** java.util.ArrayList;

**import** java.util.HashMap;

**import** java.util.List;

**import** java.util.Map;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**public** **class** Serialization {

@Test

**public** **void** Serial() {

Map<String,Object> jsonBody =**new** HashMap<String,Object>();

List<String> skills =**new** ArrayList<String>();

skills.add("java");

skills.add("selenium");

jsonBody.put("first name","Agni");

jsonBody.put("last name","A");

jsonBody.put("email","naveen.jan9@gmail.com");

jsonBody.put("skills",skills);

System.***out***.println(jsonBody);

RestAssured

.*given*()

.baseUri("http://localhost:3000")

.header("Content-Type","application/json")

.body(jsonBody)

.log()

.all()

.when()

.post("/employees")

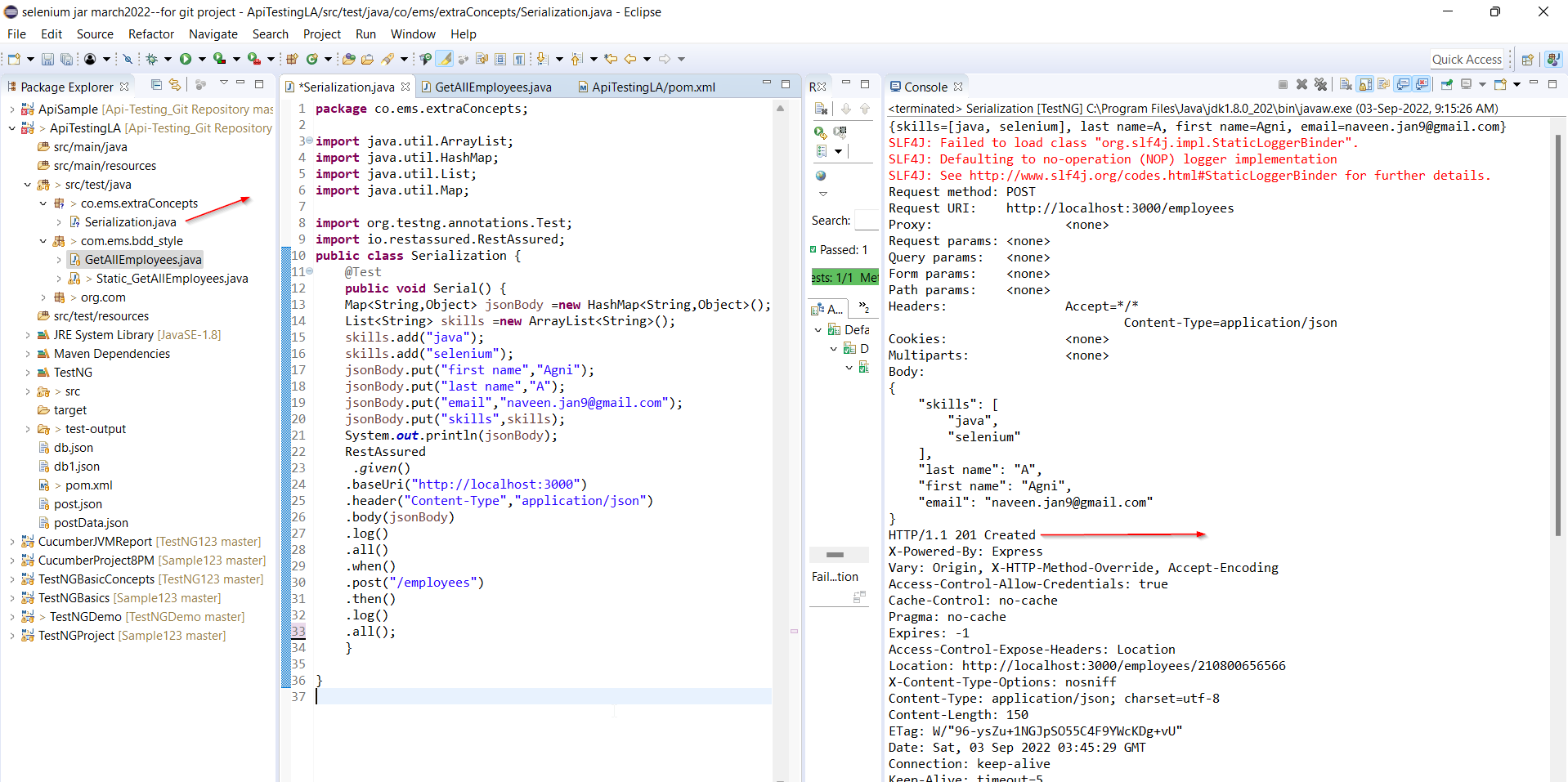
.then()

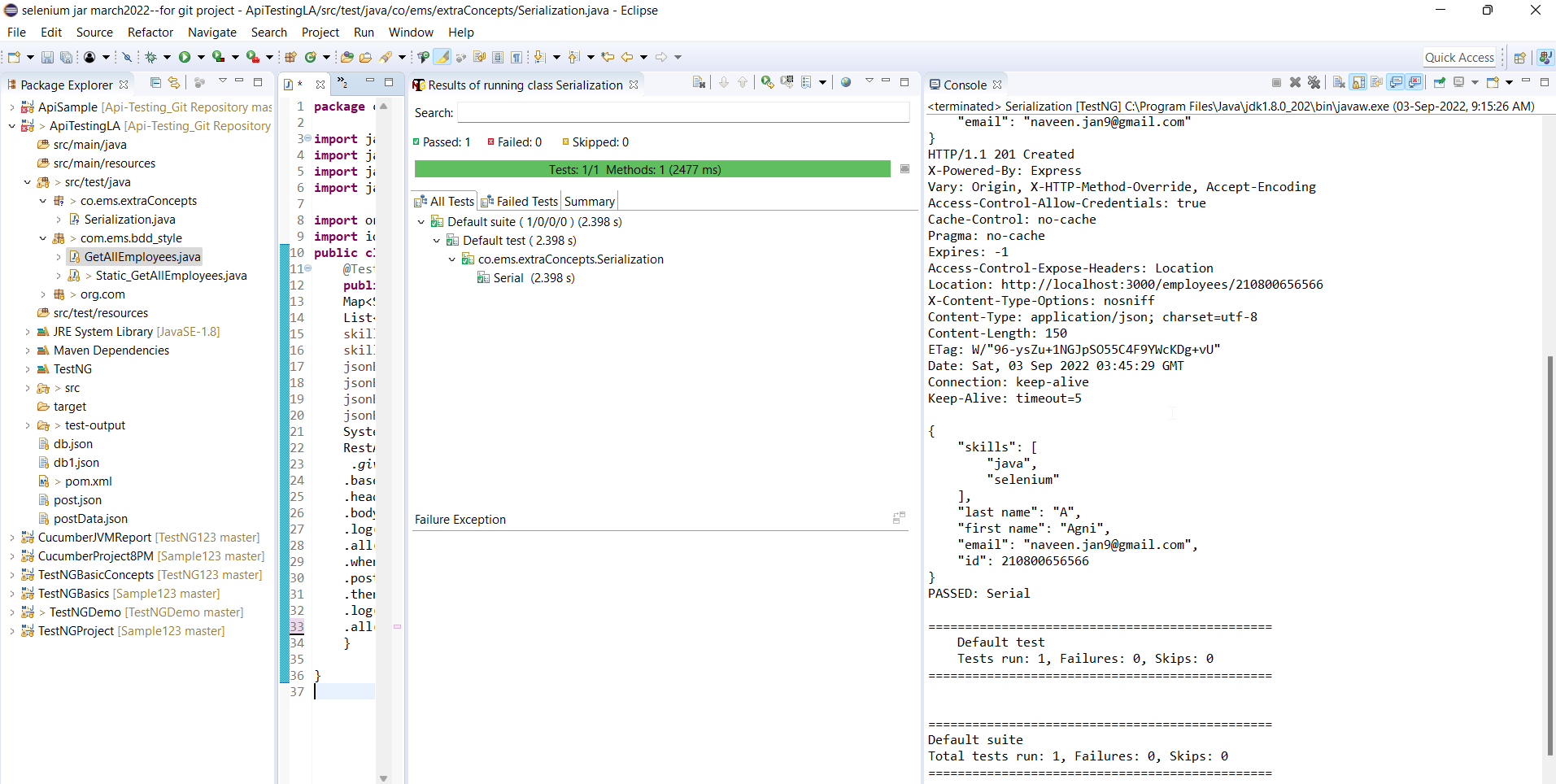
.log()

.all();

}

}





# Rest Assured | 14 | what is a POJO? | தமிழ்

**package** com.ems.PojoClass\_Concepts;

**import** java.util.List;

**public** **class** Employee {

**private** String FirstName;

**private** String LastName;

**private** String Email;

**public** String getFirstName() {

**return** FirstName;

}

**public** **void** setFirstName(String firstName) {

FirstName = firstName;

}

**public** String getLastName() {

**return** LastName;

}

**public** **void** setLastName(String lastName) {

LastName = lastName;

}

**public** String getEmail() {

**return** Email;

}

**public** **void** setEmail(String email) {

Email = email;

}

**public** List getSkills() {

**return** Skills;

}

**public** **void** setSkills(List skills) {

Skills = skills;

}

**private** List Skills;

}

**package** com.ems.PojoClass\_Concepts;

**import** java.util.Arrays;

**public** **class** Employee1 **extends** Employee {

**public** **static** **void** main(String[] args) {

Employee employee1=**new** Employee();

employee1.setFirstName("Agni");

employee1.setLastName("A");

employee1.setEmail("naveen.jan9@gmail.com");

employee1.setSkills(Arrays.*asList*("java","selenium"));

System.***out***.println(employee1.getFirstName());

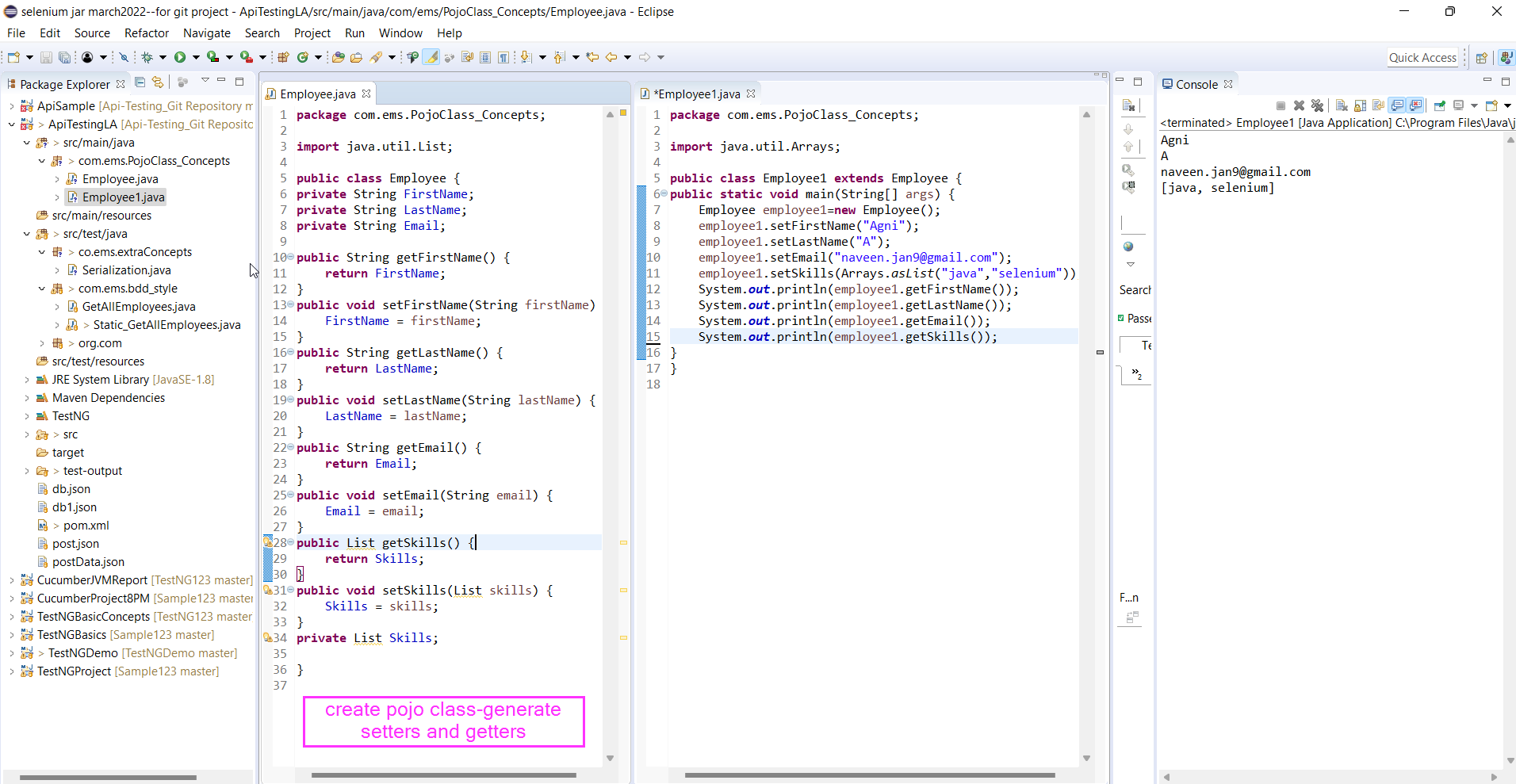
System.***out***.println(employee1.getLastName());

System.***out***.println(employee1.getEmail());

System.***out***.println(employee1.getSkills());

}

}



# Rest Assured | 15 | How to serialize a POJO | தமிழ்

**package** com.ems.PojoClass\_Concepts;

**import** java.util.List;

**public** **class** Employee {

**private** String FirstName;

**private** String LastName;

**private** String Email;

**public** String getFirstName() {

**return** FirstName;

}

**public** **void** setFirstName(String firstName) {

FirstName = firstName;

}

**public** String getLastName() {

**return** LastName;

}

**public** **void** setLastName(String lastName) {

LastName = lastName;

}

**public** String getEmail() {

**return** Email;

}

**public** **void** setEmail(String email) {

Email = email;

}

**public** List getSkills() {

**return** Skills;

}

**public** **void** setSkills(List skills) {

Skills = skills;

}

**private** List Skills;

}

**package** com.ems.PojoClass\_Concepts;

**import** java.util.Arrays;

**import** com.fasterxml.jackson.core.JsonProcessingException;

**import** com.fasterxml.jackson.databind.ObjectMapper;

**public** **class** Employee1 **extends** Employee {

**public** **static** **void** main(String[] args) **throws** JsonProcessingException {

Employee employee1=**new** Employee();

employee1.setFirstName("Agni");

employee1.setLastName("A");

employee1.setEmail("naveen.jan9@gmail.com");

employee1.setSkills(Arrays.*asList*("java","selenium"));

System.***out***.println(employee1.getFirstName());

System.***out***.println(employee1.getLastName());

System.***out***.println(employee1.getEmail());

System.***out***.println(employee1.getSkills());

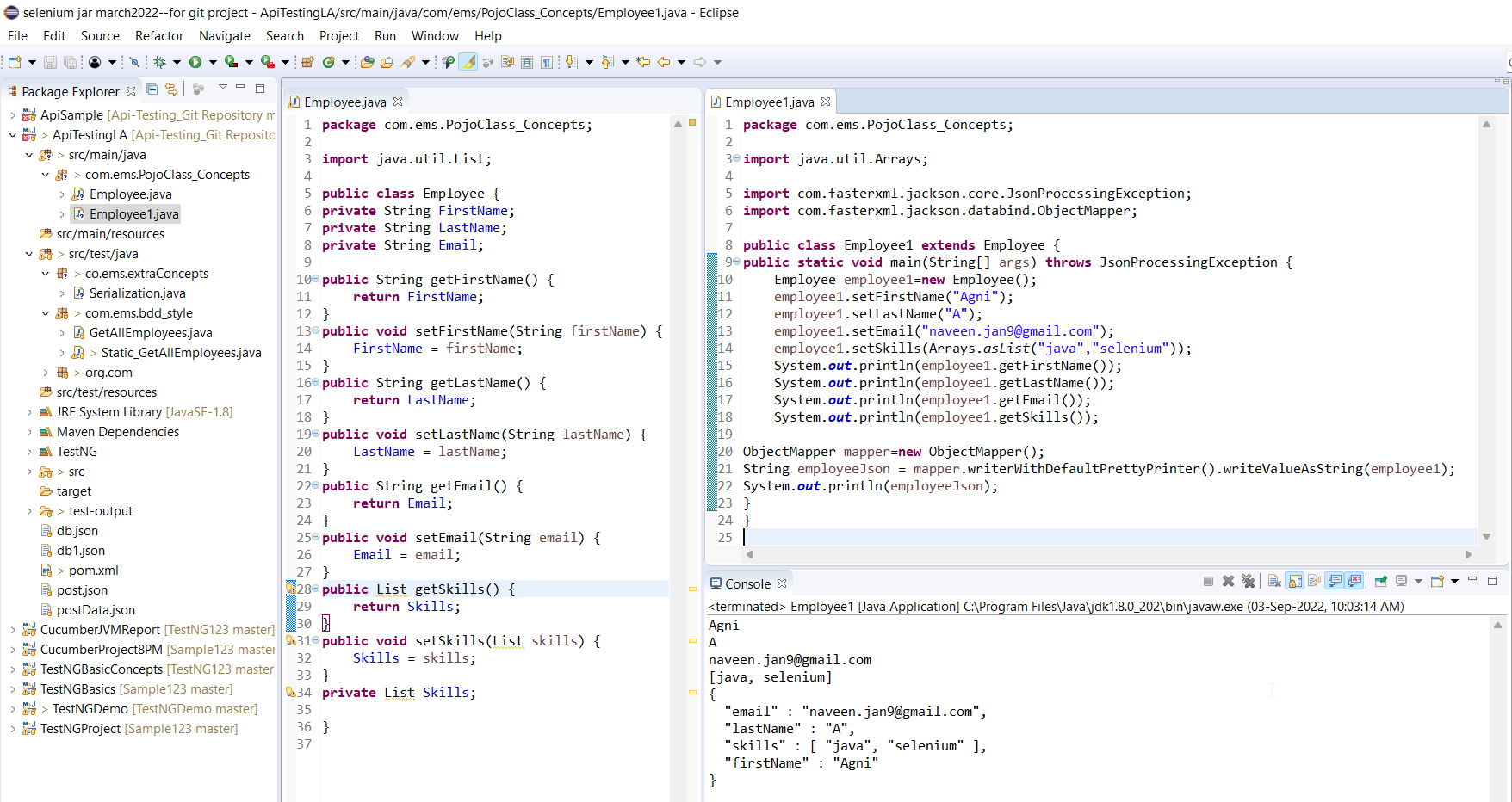
ObjectMapper mapper=**new** ObjectMapper();

String employeeJson = mapper.writerWithDefaultPrettyPrinter().writeValueAsString(employee1);

System.***out***.println(employeeJson);

}

}



===========================================================================

# Rest Assured |16 | what is json schema | JSON vs JSON Schema | தமிழ்

Watch this youtube vidoe-No Program

# Rest Assured | 17 | JSON schema validation using Rest assured | schema in class path | தமிழ்

**package** com.ems.Schema.Validation;

**import** java.io.File;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**import** io.restassured.module.jsv.JsonSchemaValidator;

**public** **class** SchemaValidation {

@Test

**private** **void** validate() {

File inputJson =**new** File("src/test/resources/input.json");

RestAssured

.*given*()

.baseUri("http://localhost:3000")

.header("Content-Type","application/json")

.body(inputJson)

.when()

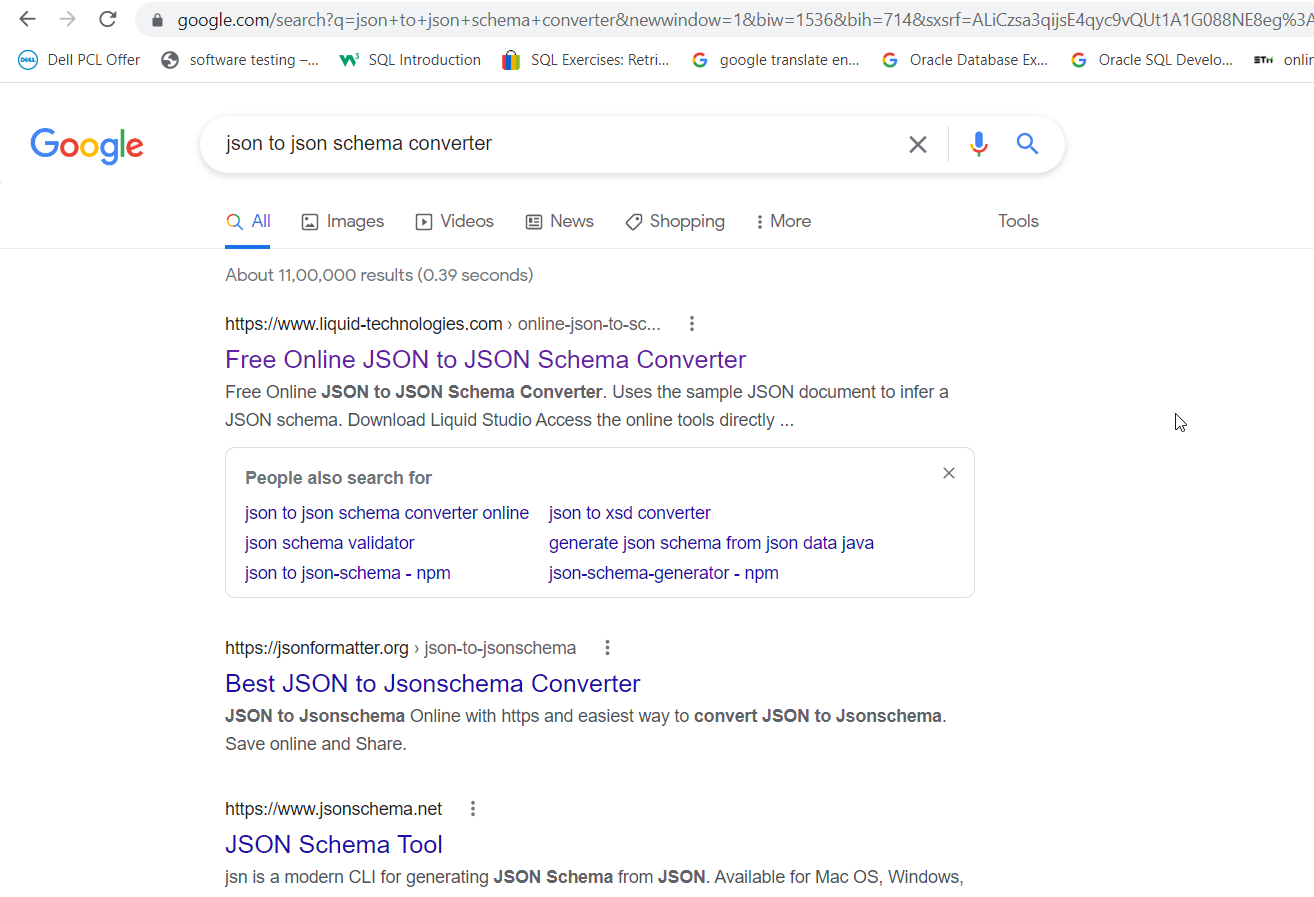
.post("/employees")

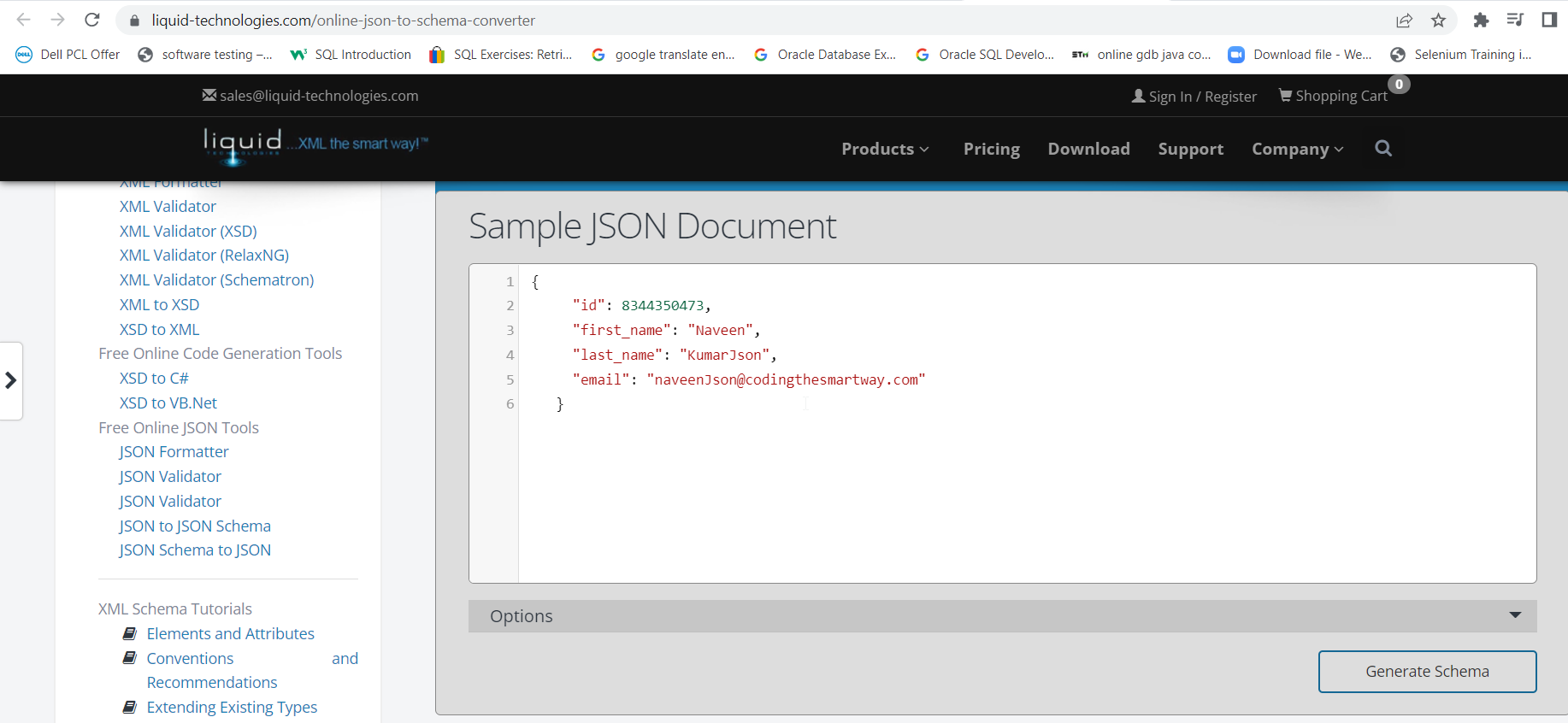
.then()

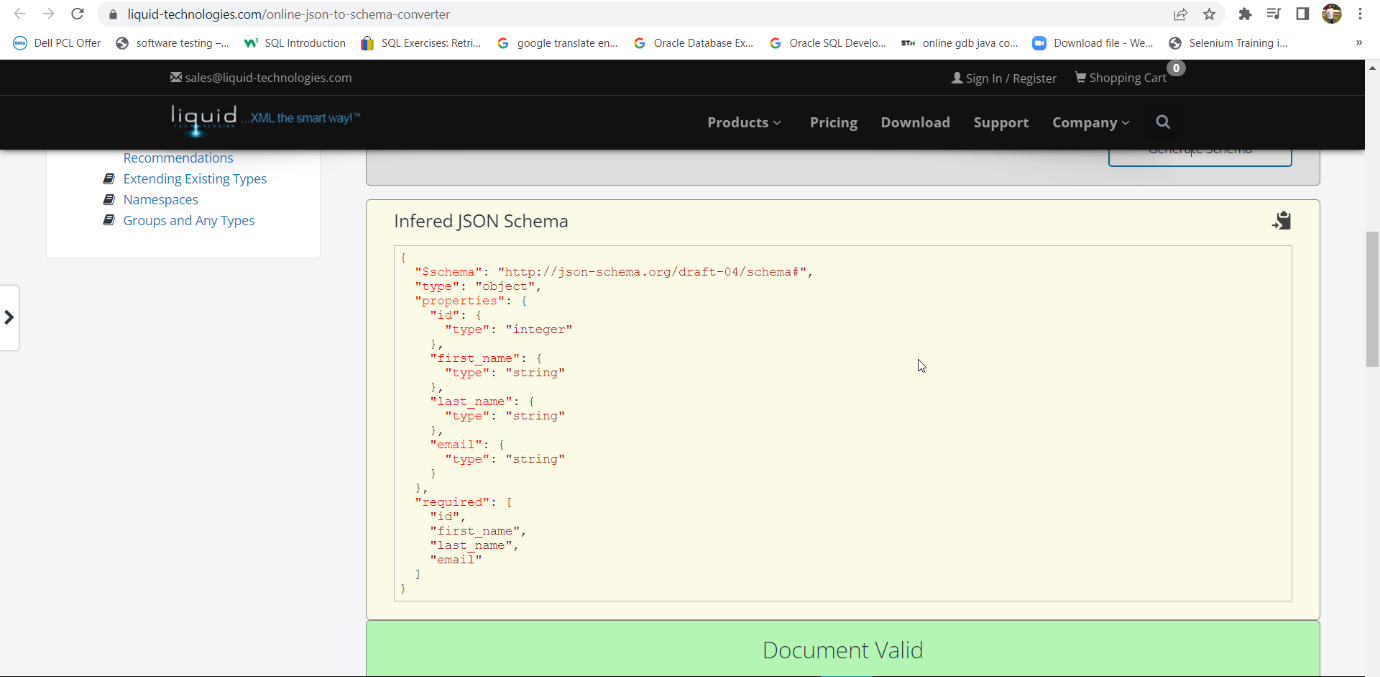
.body(JsonSchemaValidator.*matchesJsonSchemaInClasspath*("schema.json"));

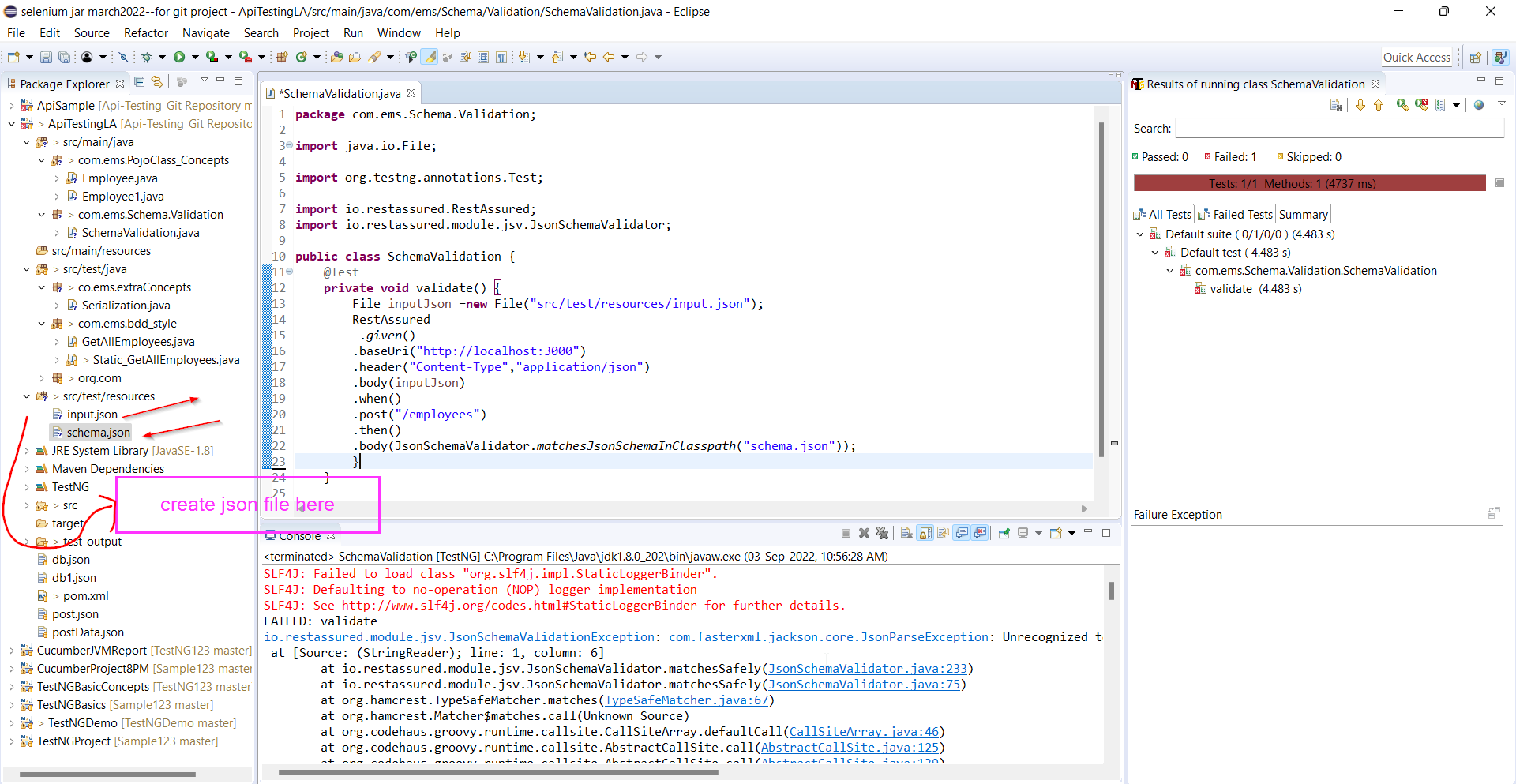
}

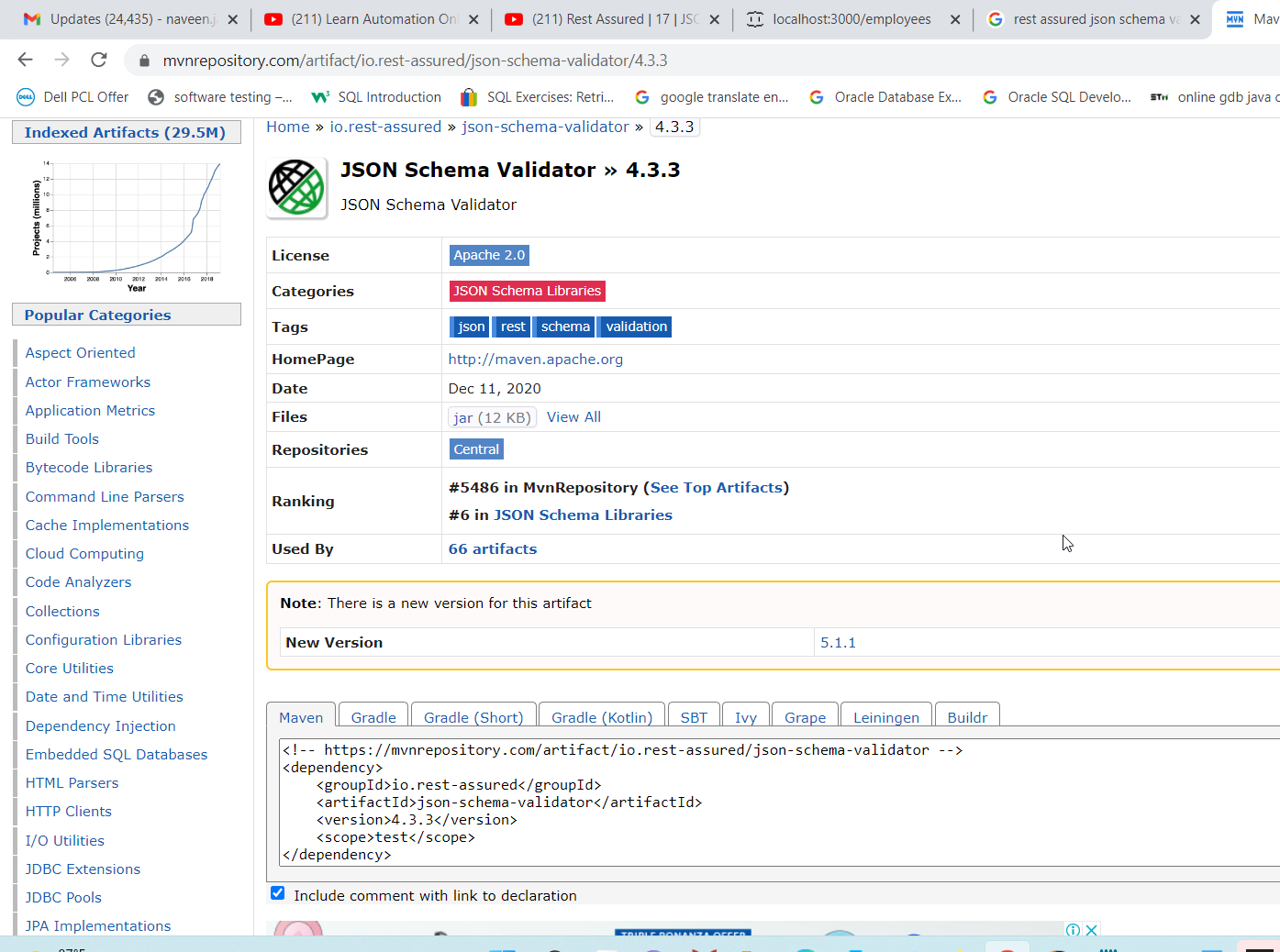
}











# Rest Assured |18 | JSON schema validation using Rest Assured | Without class path method | தமிழ்

**package** com.ems.Schema.Validation;

**import** java.io.File;

**import** java.io.FileNotFoundException;

**import** java.io.FileReader;

**import** java.io.Reader;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**import** io.restassured.module.jsv.JsonSchemaValidator;

**public** **class** SchemaValidation {

@Test(enabled=**false**)

**private** **void** validate() {

File inputJson =**new** File("src/test/resources/input.json");

RestAssured

.*given*()

.baseUri("http://localhost:3000")

.header("Content-Type","application/json")

.body(inputJson)

.when()

.post("/employees")

.then()

.body(JsonSchemaValidator.*matchesJsonSchemaInClasspath*("schema.json"));

}

@Test(enabled=**false**)

**private** **void** validateWithoutPath() {

File inputJson =**new** File("src/test/resources/input.json");

File inputSchema =**new** File("src/test/resources/schema.json");

RestAssured

.*given*()

.baseUri("http://localhost:3000")

.header("Content-Type","application/json")

.body(inputJson)

.when()

.post("/employees")

.then()

.body(JsonSchemaValidator.*matchesJsonSchema*(inputSchema));

}

@Test

**private** **void** validateWithoutPath1() **throws** FileNotFoundException {

File inputJson =**new** File("src/test/resources/input.json");

Reader inputSchema =**new** FileReader("src/test/resources/schema.json");

RestAssured

.*given*()

.baseUri("http://localhost:3000")

.header("Content-Type","application/json")

.body(inputJson)

.when()

.post("/employees")

.then()

.body(JsonSchemaValidator.*matchesJsonSchema*("{\r\n" +

"\r\n" +

" \"$schema\": \"http://json-schema.org/draft-04/schema#\",\r\n" +

"\r\n" +

" \"type\": \"object\",\r\n" +

"\r\n" +

" \"properties\": {\r\n" +

"\r\n" +

" \"id\": {\r\n" +

"\r\n" +

" \"type\": \"integer\"\r\n" +

"\r\n" +

" },\r\n" +

"\r\n" +

" \"first\_name\": {\r\n" +

"\r\n" +

" \"type\": \"string\"\r\n" +

"\r\n" +

" },\r\n" +

"\r\n" +

" \"last\_name\": {\r\n" +

"\r\n" +

" \"type\": \"string\"\r\n" +

"\r\n" +

" },\r\n" +

"\r\n" +

" \"email\": {\r\n" +

"\r\n" +

" \"type\": \"string\"\r\n" +

"\r\n" +

" }\r\n" +

"\r\n" +

" },\r\n" +

"\r\n" +

" \"required\": [\r\n" +

"\r\n" +

" \"id\",\r\n" +

"\r\n" +

" \"first\_name\",\r\n" +

"\r\n" +

" \"last\_name\",\r\n" +

"\r\n" +

" \"email\"\r\n" +

"\r\n" +

" ]\r\n" +

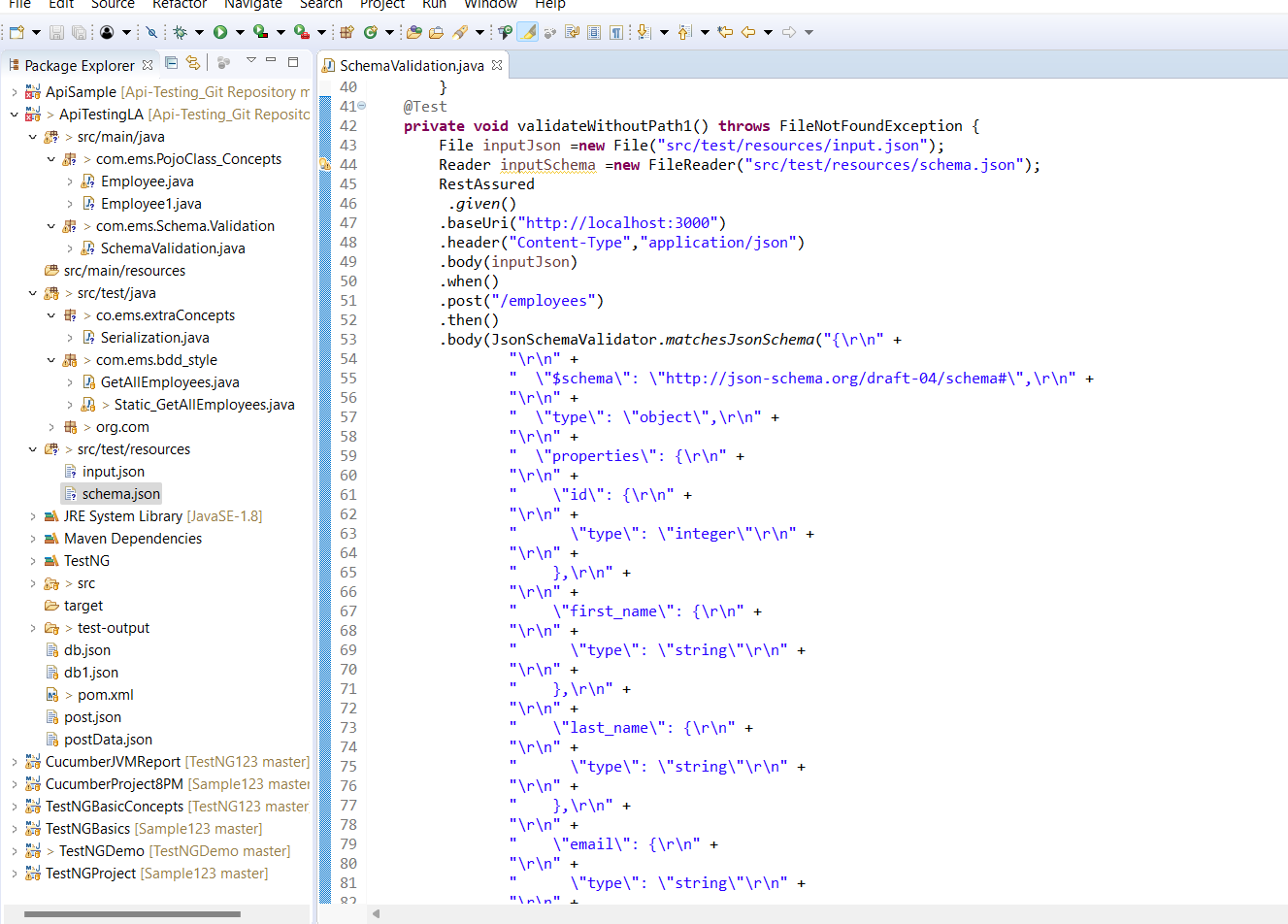
"\r\n" +

"}"));

}

}





# Rest Assured | 19 | JSON schema validation using networknt | தமிழ்