**Name: Sureshkumar V P**

**Company: MSG Global Solutions India Private Limited**

**Country:** **India**

**ASSIGNMENT**

**EU COUNTRY NAMES - TOP 3 HIGHEST STANDARD & TOP 3 LOWEST REDUCED VAT RATES**

**02-02-2023**

**Objective**

Build a microservice which exposes a REST API for:

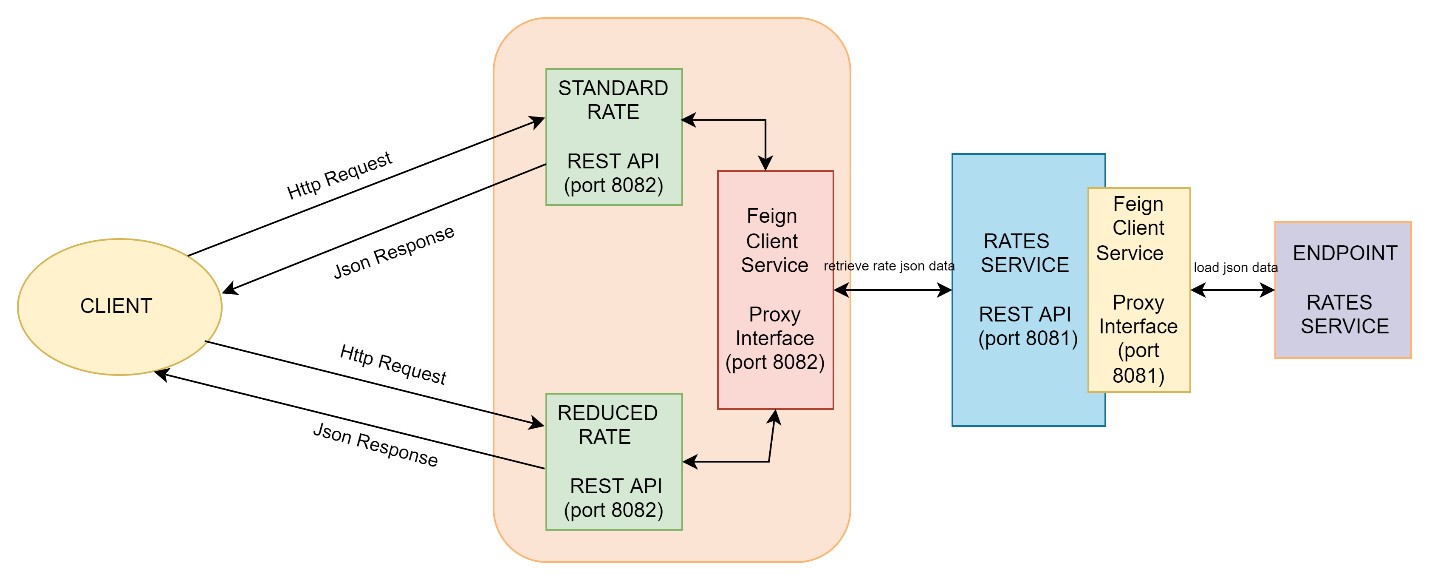
• Return three EU countries with the highest standard VAT rate

• Return three EU countries with the lowest reduced VAT rates

Input: [https://euvatrates.com/rates.json](https://eur03.safelinks.protection.outlook.com/?url=https%3A%2F%2Feuvatrates.com%2Frates.json&data=05%7C01%7C%7C291163ebb0834e1093ee08dad697583a%7Cc2350a4a56c64ec19195e636c90735d6%7C0%7C0%7C638058242808176820%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C3000%7C%7C%7C&sdata=Nqx2a3rDR3E0nwY%2FOKMsH0gfzG2BFNE5tS1MowXBdy8%3D&reserved=0)

**Guidelines: Keep it clean and simple (with some design for future extensions). Implement as a Maven project using Spring Boot or Quarkus. Provide source code, a way to run it and some kind documentation for the API.**

**High Level Diagram**

****

**Frameworks, Tools & Languages**

* Eclipse
* SpringBoot
* Feign Client
* Java
* JSON
* Postman

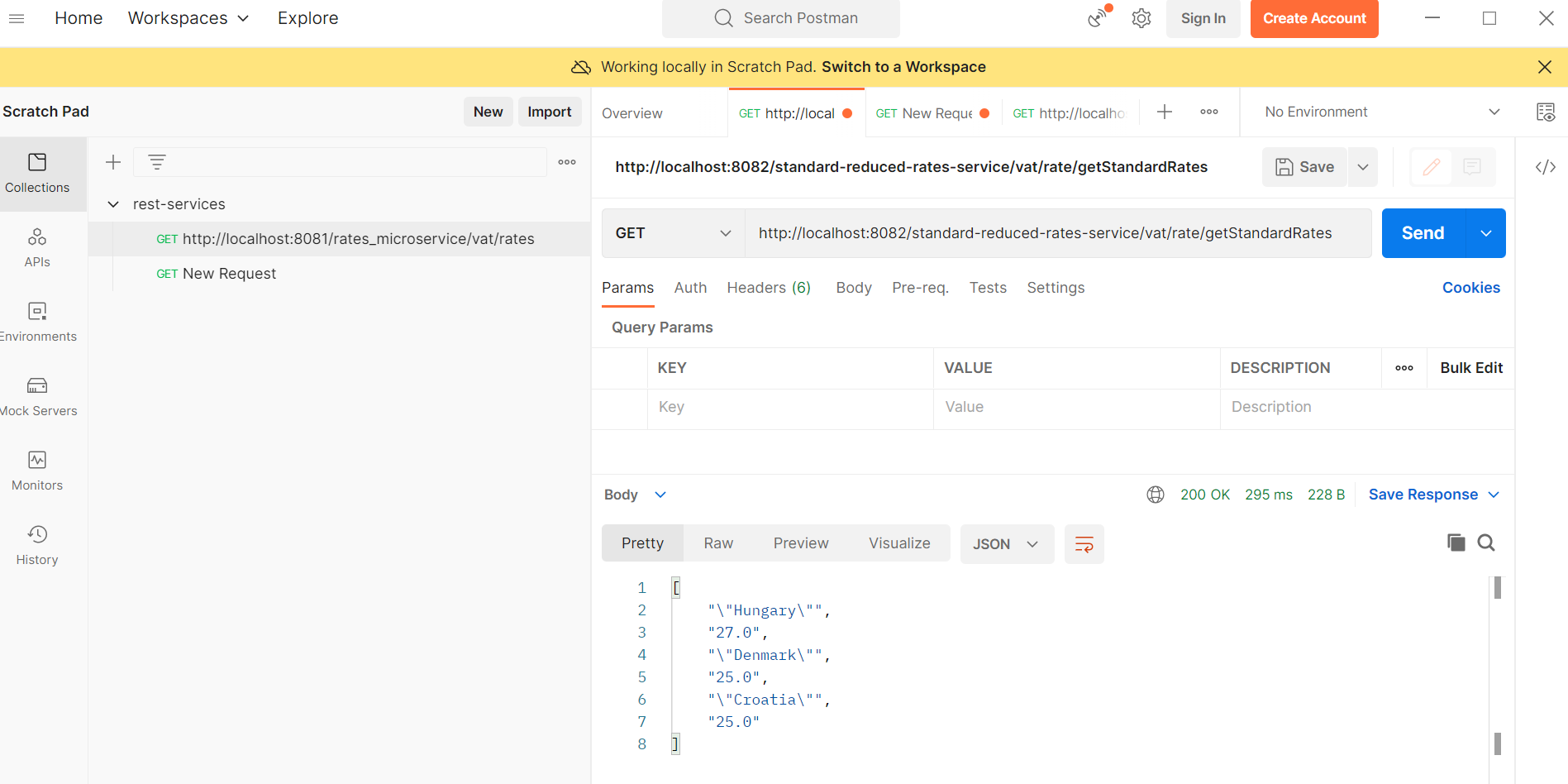
**Test Results**

* **Input:** [https://euvatrates.com/rates.json](https://eur03.safelinks.protection.outlook.com/?url=https%3A%2F%2Feuvatrates.com%2Frates.json&data=05%7C01%7C%7C291163ebb0834e1093ee08dad697583a%7Cc2350a4a56c64ec19195e636c90735d6%7C0%7C0%7C638058242808176820%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C3000%7C%7C%7C&sdata=Nqx2a3rDR3E0nwY%2FOKMsH0gfzG2BFNE5tS1MowXBdy8%3D&reserved=0)



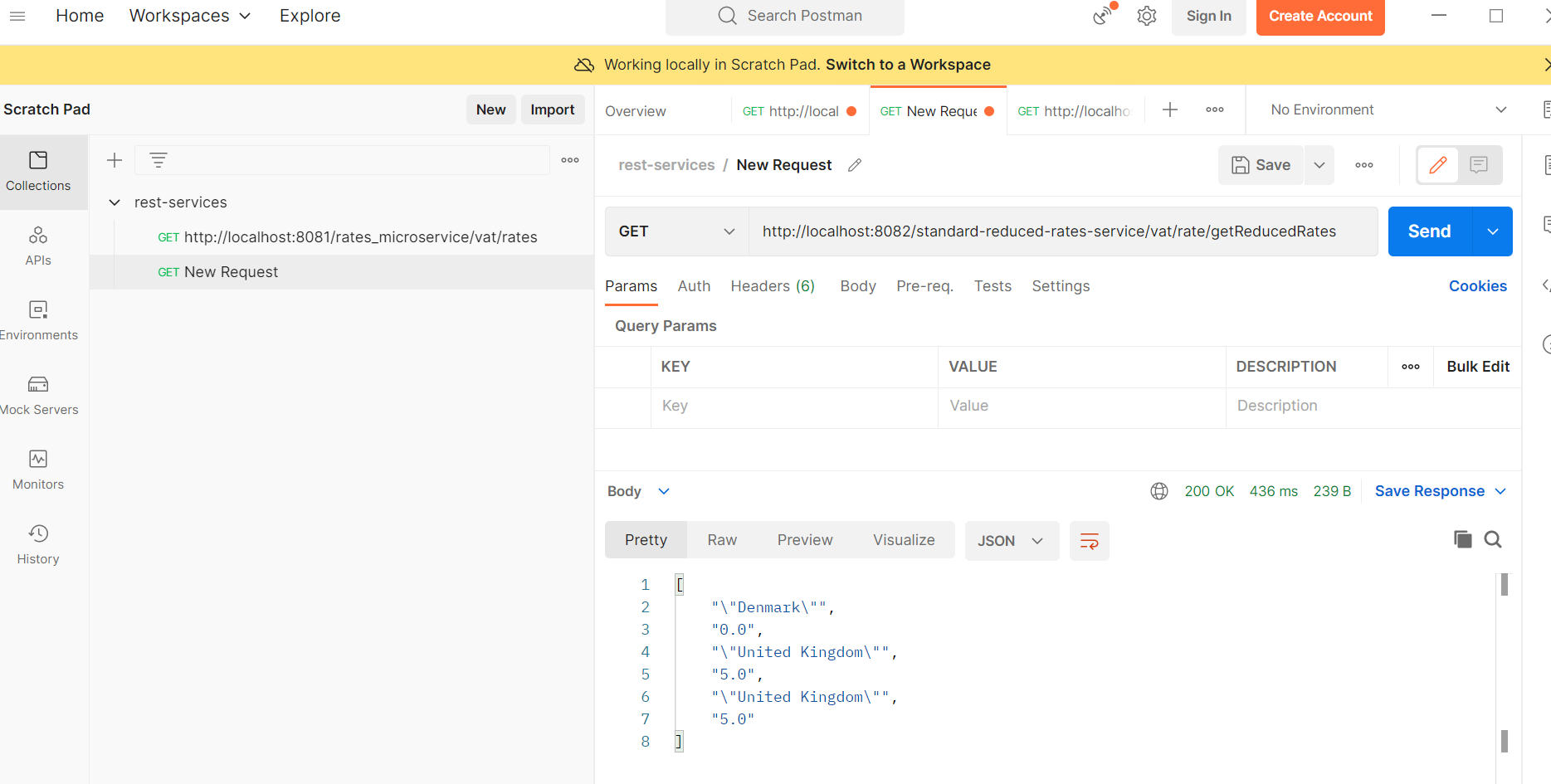
* Return three EU countries with the highest standard VAT rate

**Output URL:** <http://localhost:8082/standard-reduced-rates-service/vat/rate/getStandardRates>



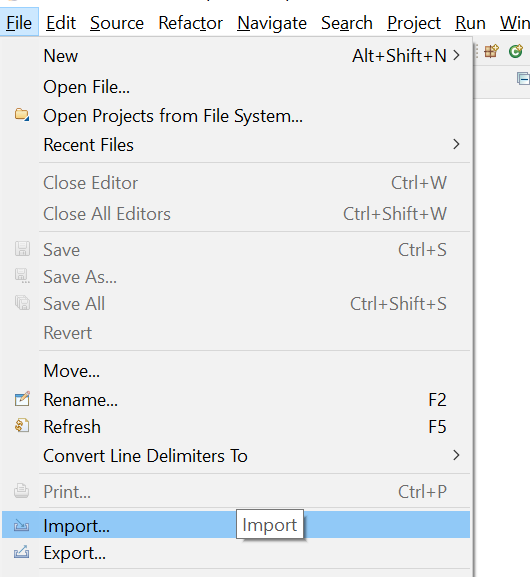
* Return three EU countries with the lowest reduced VAT rates

**Output URL:** <http://localhost:8082/standard-reduced-rates-service/vat/rate/getReducedRates>

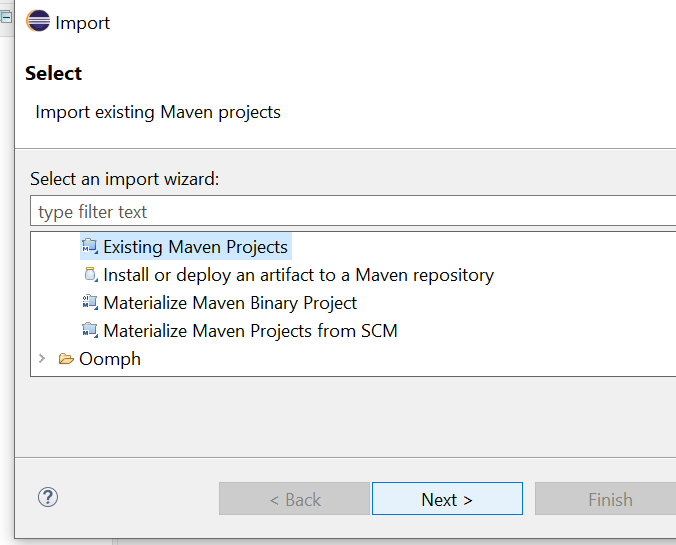


**Steps to configure and execute (in eclipse ide)**

1. Download archive files from one drive (link shared via email) and extract into local folders
2. Open eclipse tool
   1. File and select import as Maven project by choosing the extracted project folder

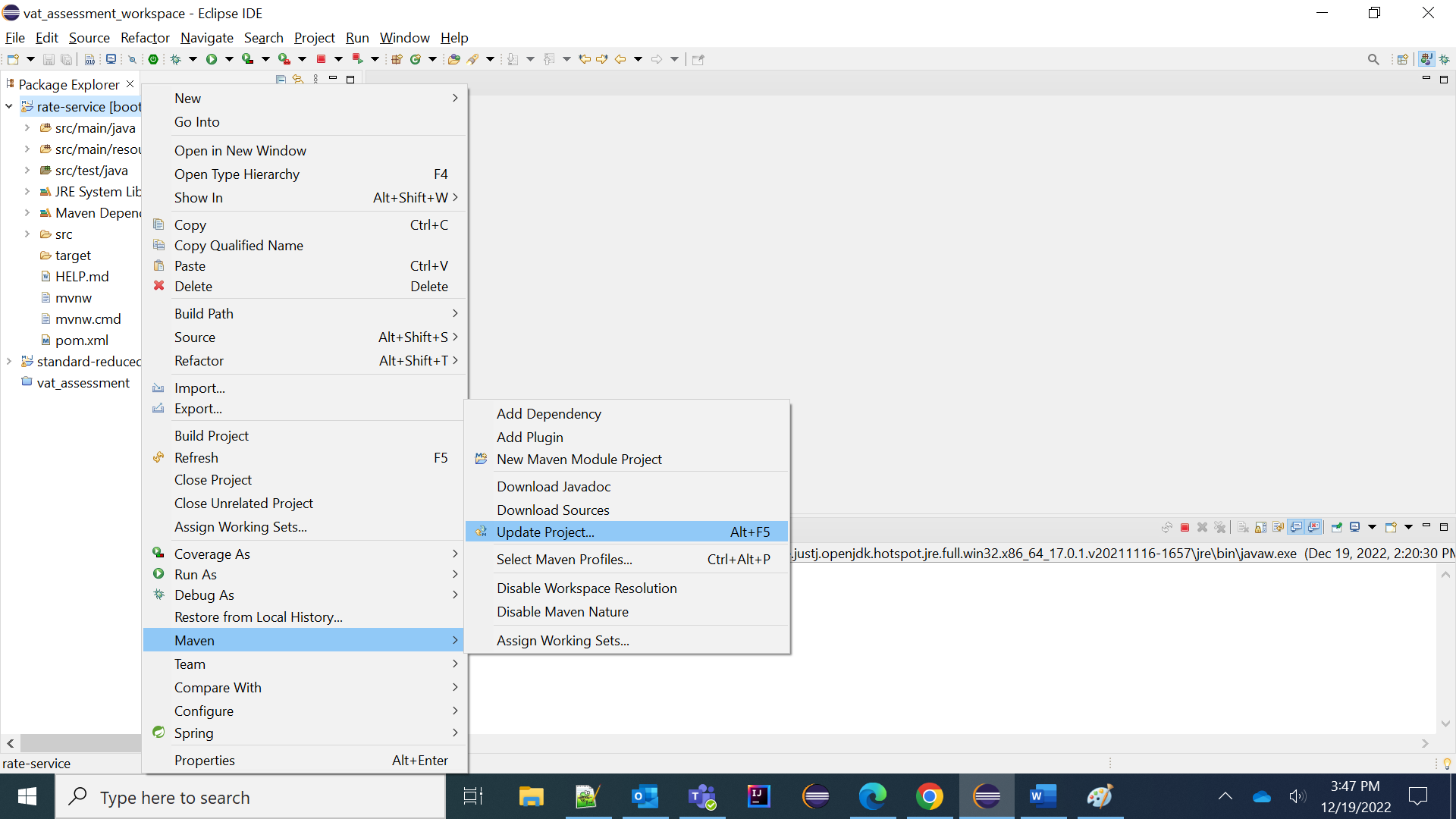
****

* 1. Select Existing Maven Projects

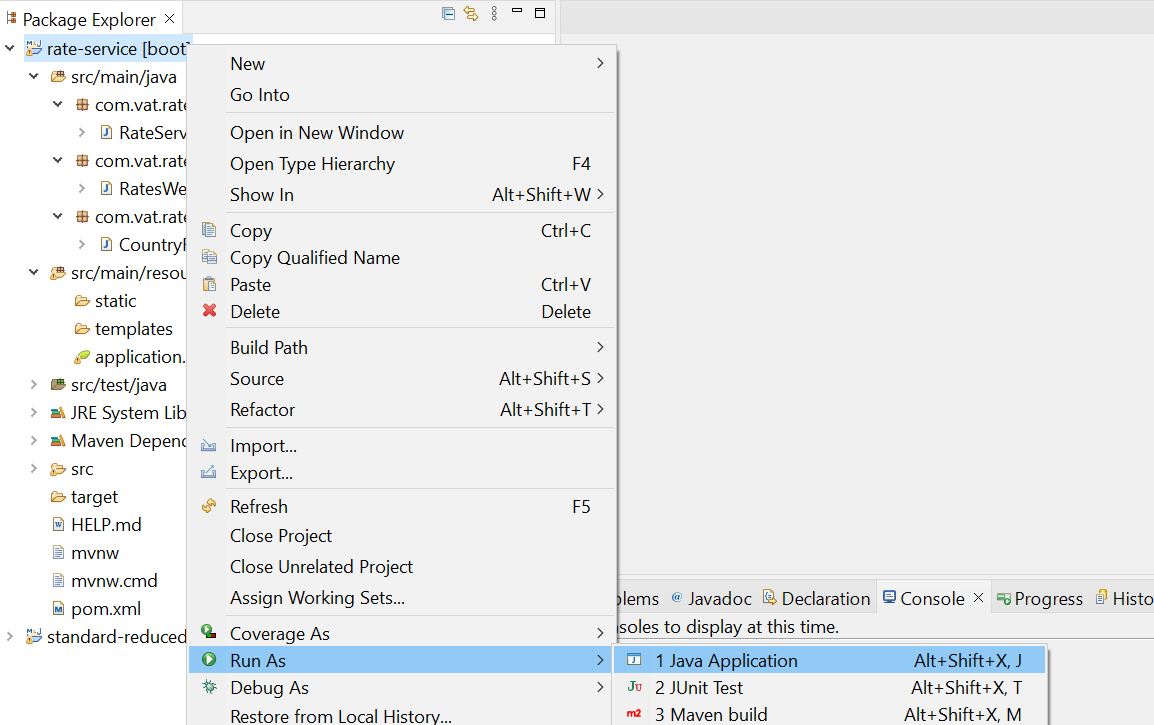
****

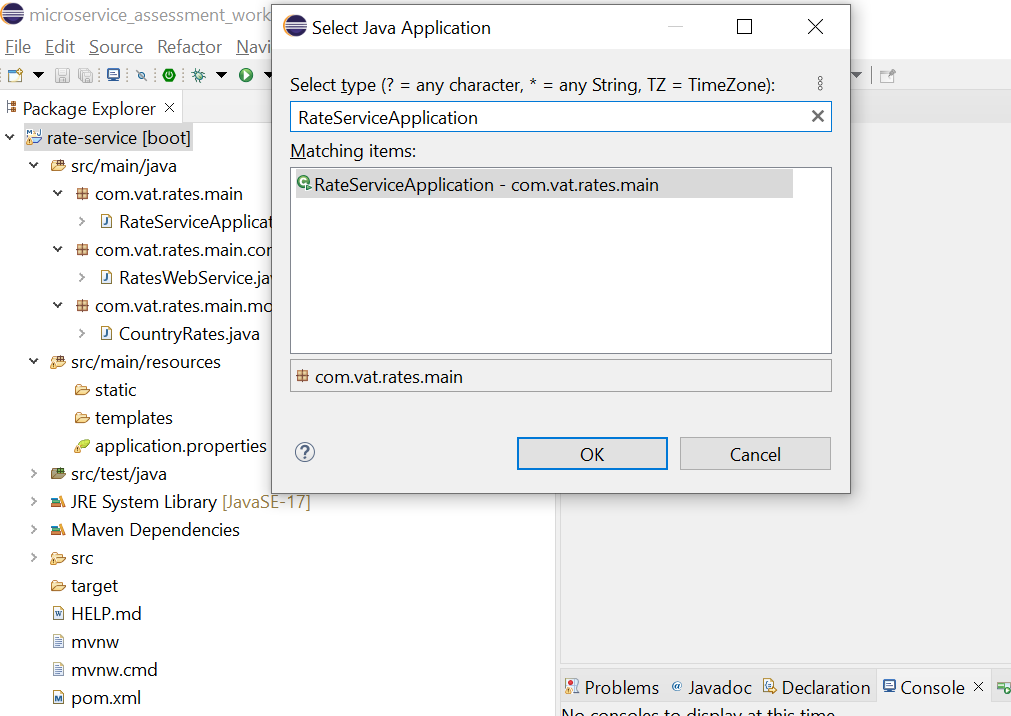
* 1. Select Root directory of the project extracted and select pom.xml check box and Finish

1. Update maven by selecting rate-service project



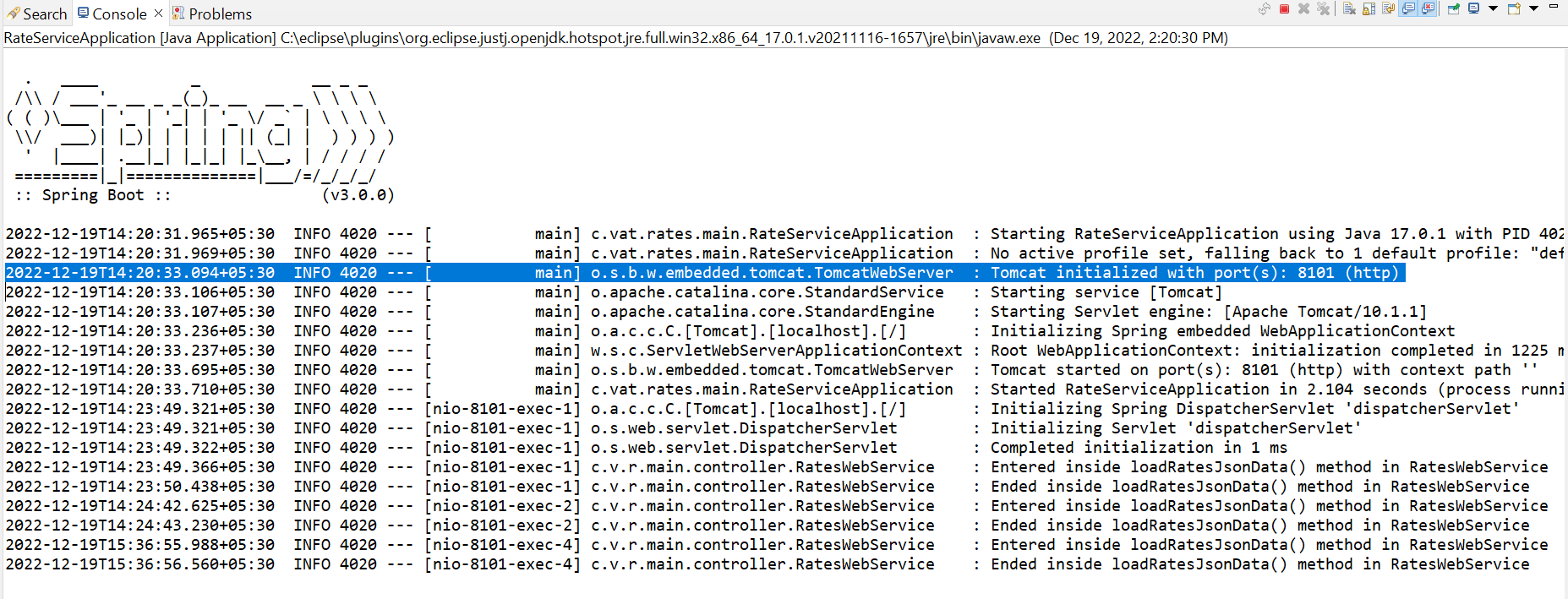
1. Select RateServiceApplication.java and click Run as Java Application with port 8081

****

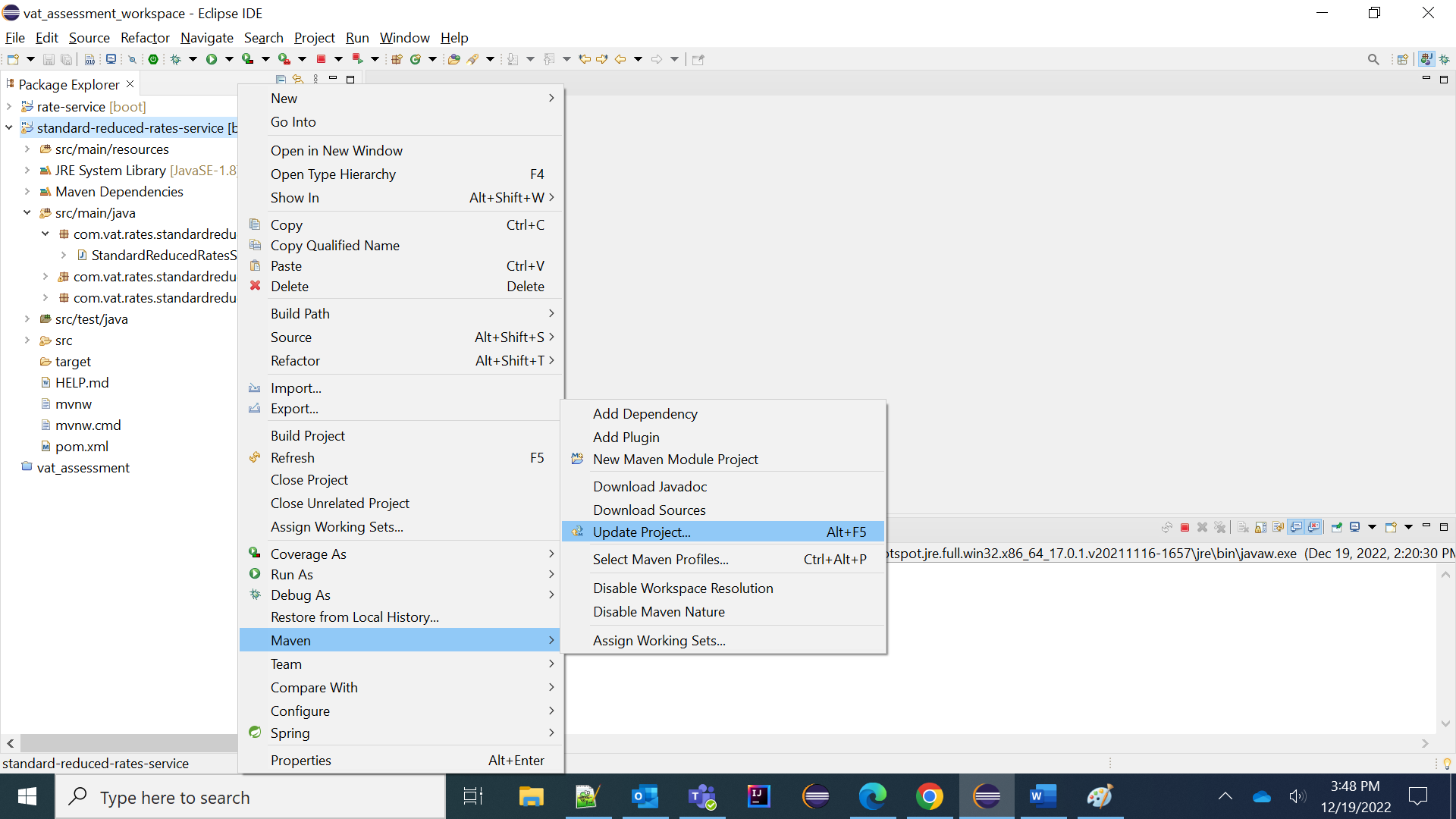
****

1. Once Spring Boot started by confirming in console

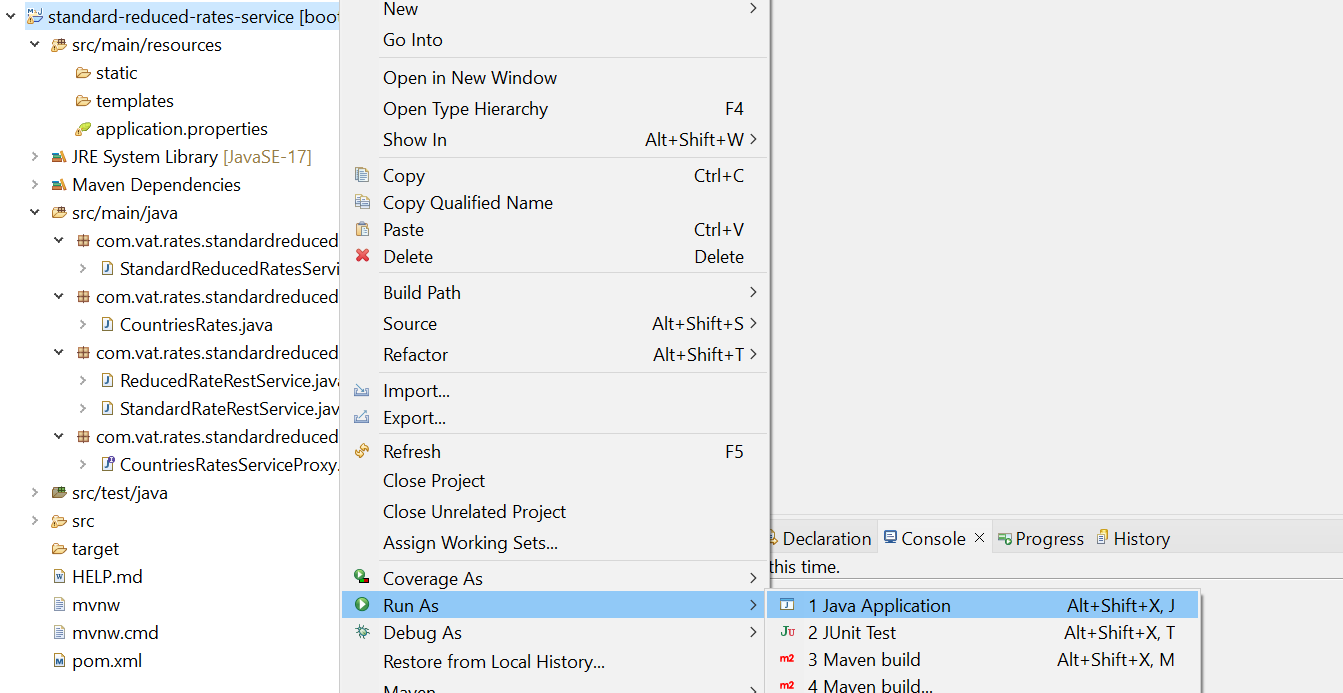
Localhost:8081

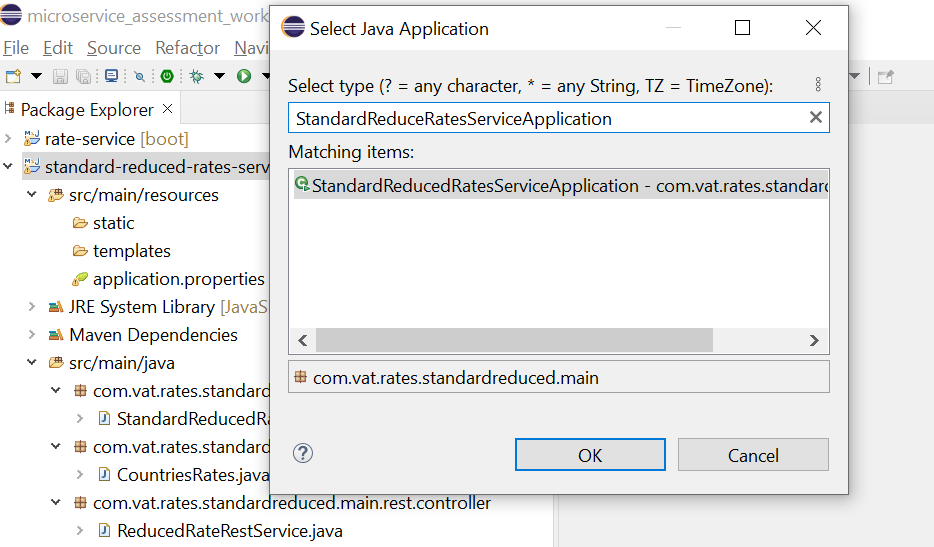


1. Update maven by selecting standard-reduced-rates-service project



1. Select StandardReducedRatesServiceApplication.java and click Run as Java Application with port 8082

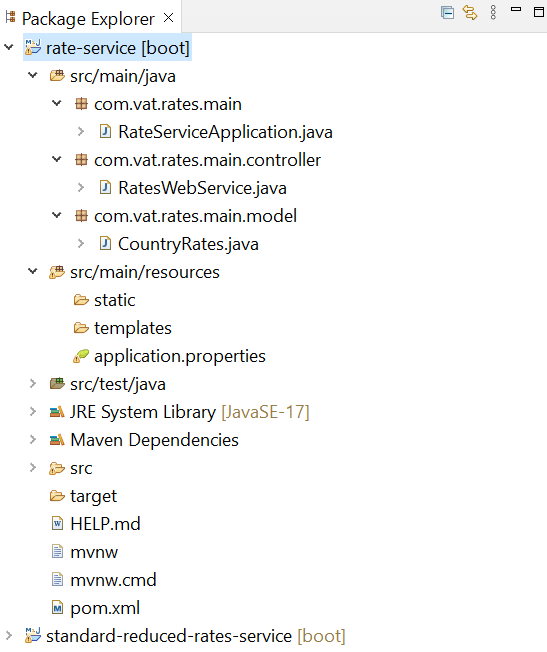
****

****

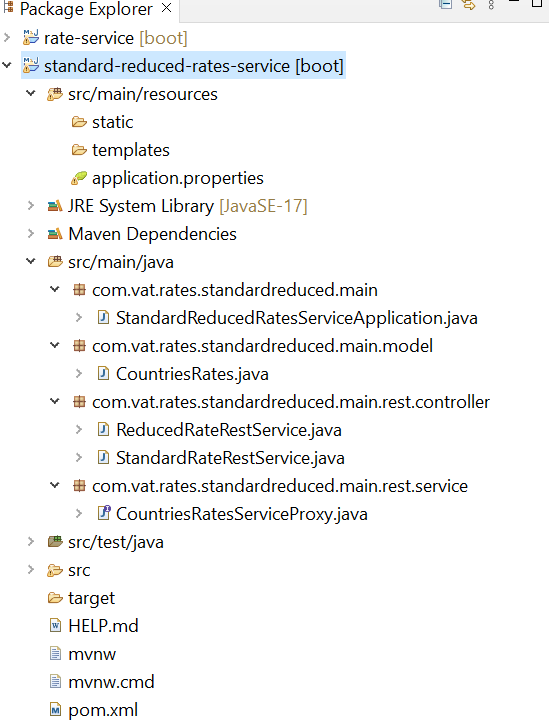
Rate\_Service - localhost:8081, Standard-reduced-rates-service – localhost:8082

**Project Structures**

RATE-SERVICE

****

STANDARD-REDUCED-RATES-SERVICE

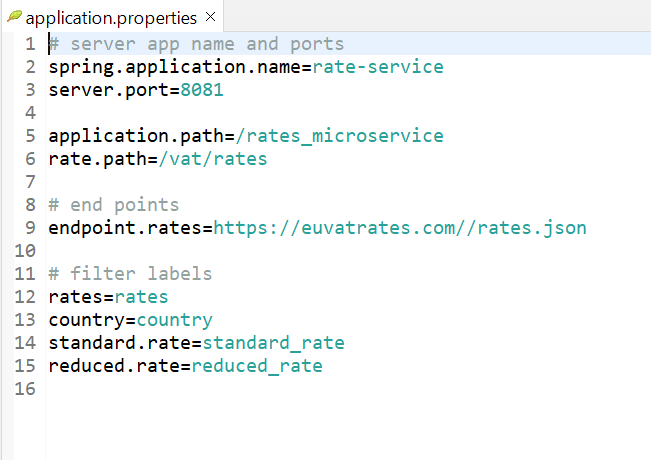


**Rate-Service:**

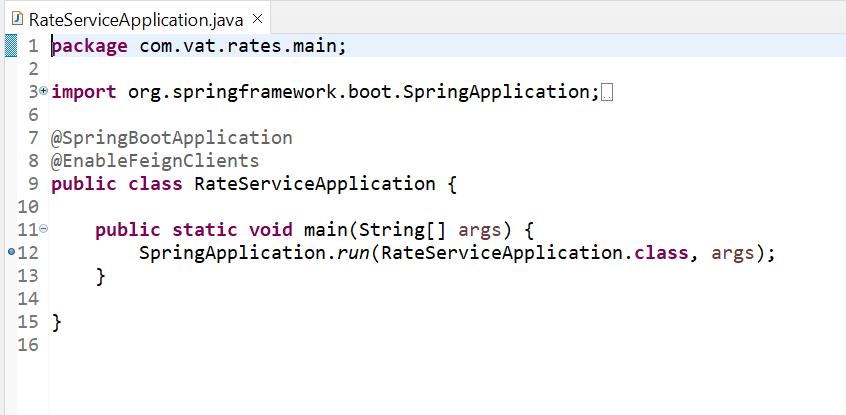
1. **Pom.xml** - Used to load dependencies
   1. **Spring-boot-starter-parent** – used to bootstrap the spring application
   2. **Spring-boot-starter-web** – used to start the web app and to support to make rest template call to the end point



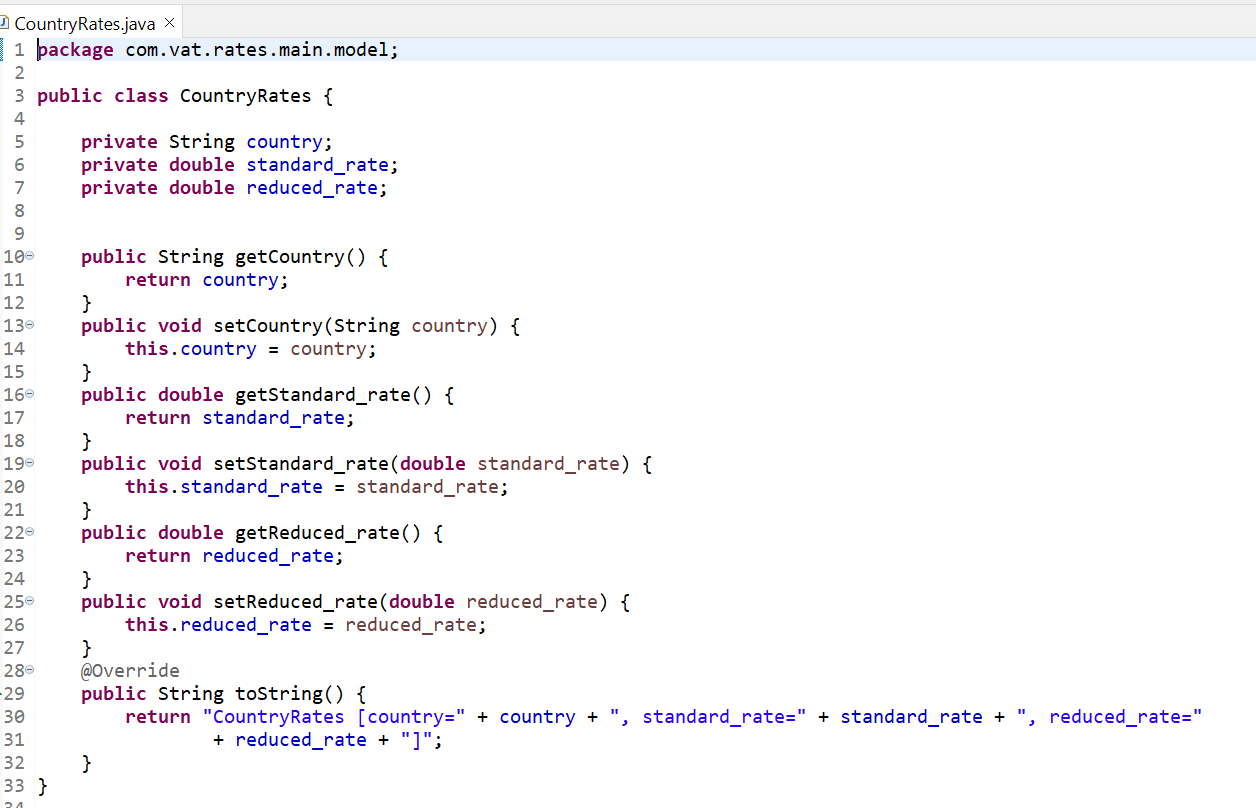
1. **Application.properties** – Used to maintain the endpoint as key/values and constant key values



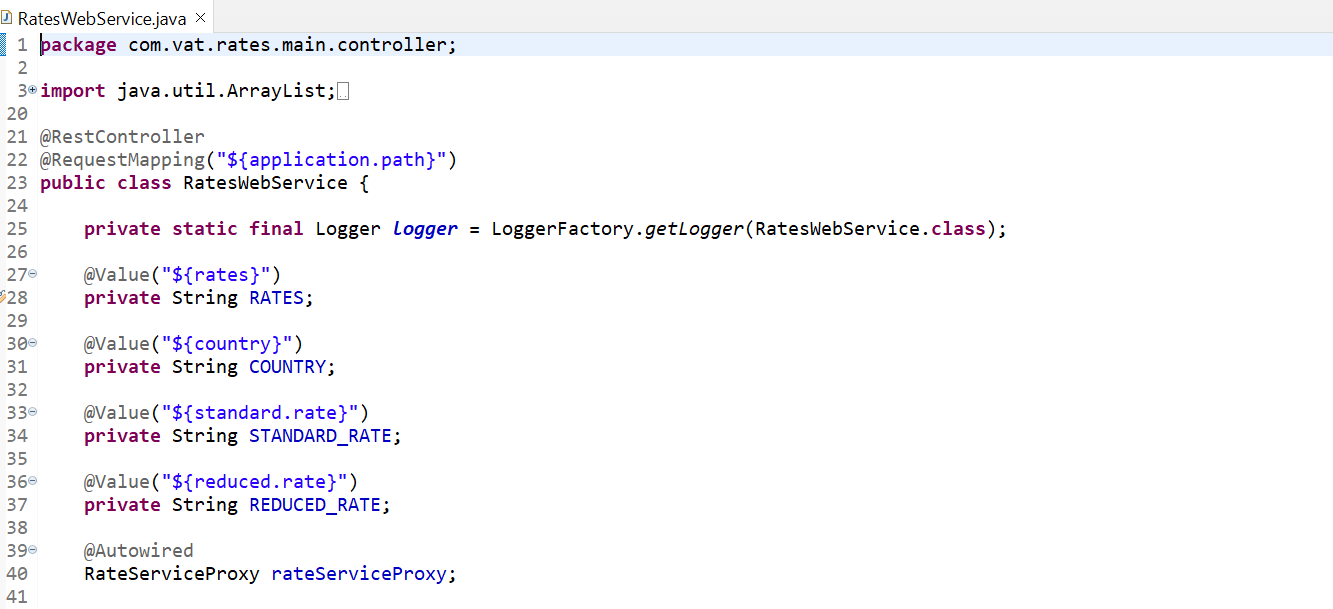
1. **RateServiceApplication.java** – Used to bootsrap the spring boot main class

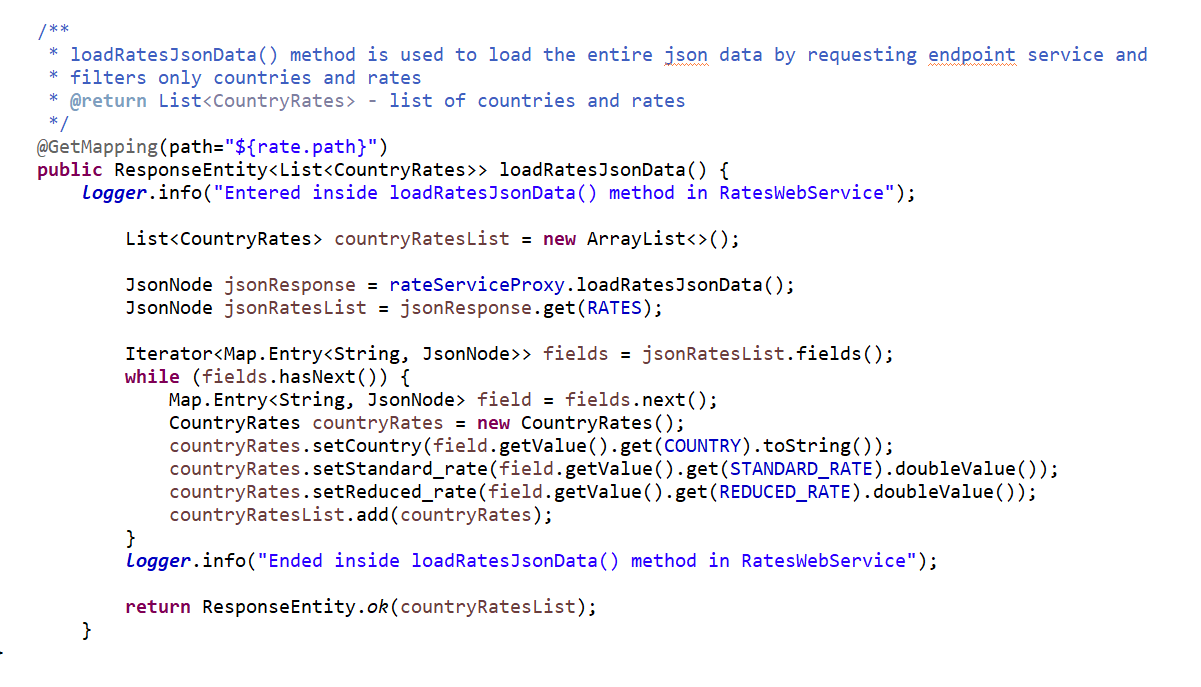


1. **CountryRates.java** – Used to maintain getter/setters in model bean class for the parameters Country, Standard Rate and Reduced Rate



1. **RatesWebService.java** – Used to inject rate service to load endpoint json data and apply the filter logic as per requirements
   1. loadRatesJsonData () method used to retrieve json data apply the filter to get countries and rates list





1. **RatesServiceProxy.java** – Used to inject rate service to load endpoint json data and apply the filter logic as per requirements

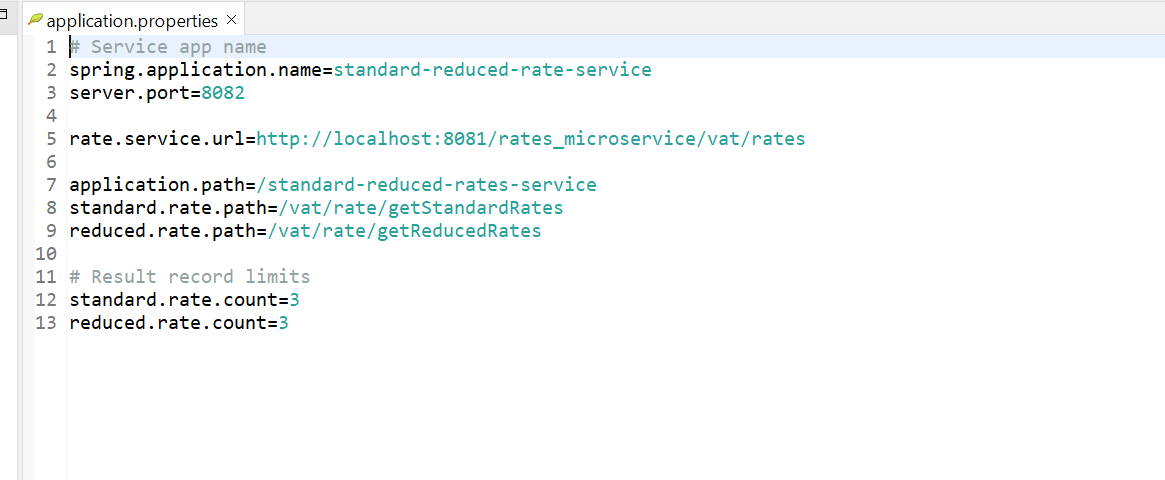


**standard-reduced-rates-service:**

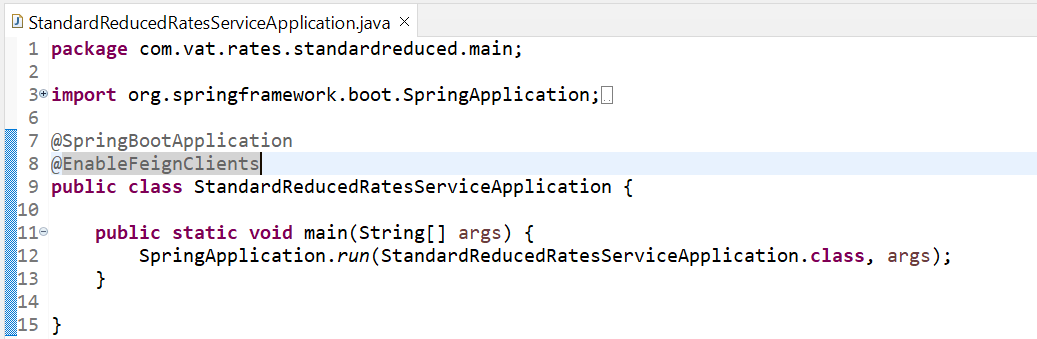
1. **Pom.xml** - Used to load dependencies
   1. **Spring-boot-starter-parent** – used to bootstrap the spring application
   2. **Spring-boot-starter-web** – used to start the web app and to support to make rest template call to the end point
   3. **spring-cloud-starter-openfeign** - to use a declarative way to build HTTP clients by means of creating annotated interfaces without writing any boilerplate code
   4. **spring-cloud-starter-bootstrap**
   5. **json**



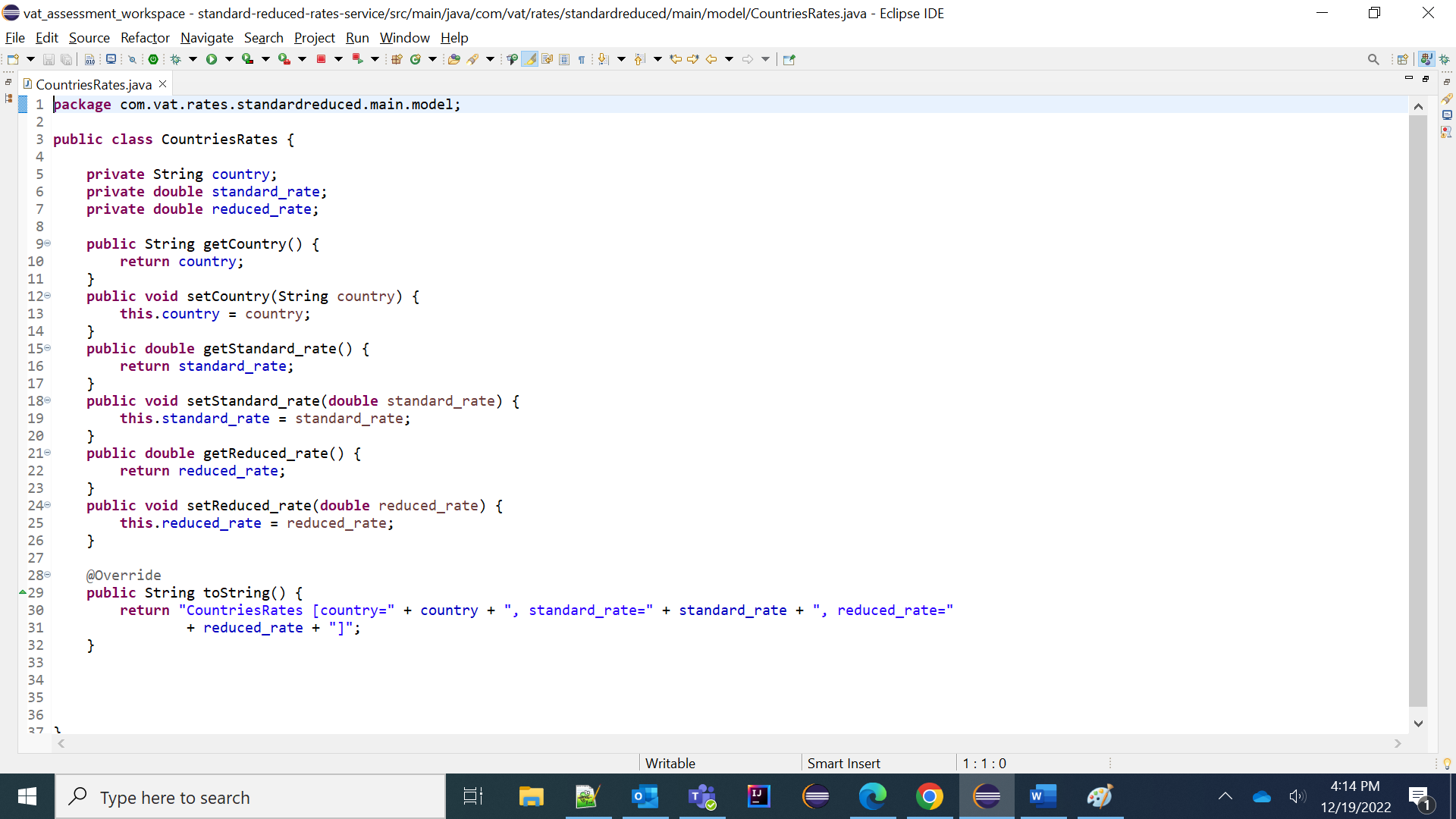
1. **Application.properties** – Used to maintain the endpoint as key/values and constant key values



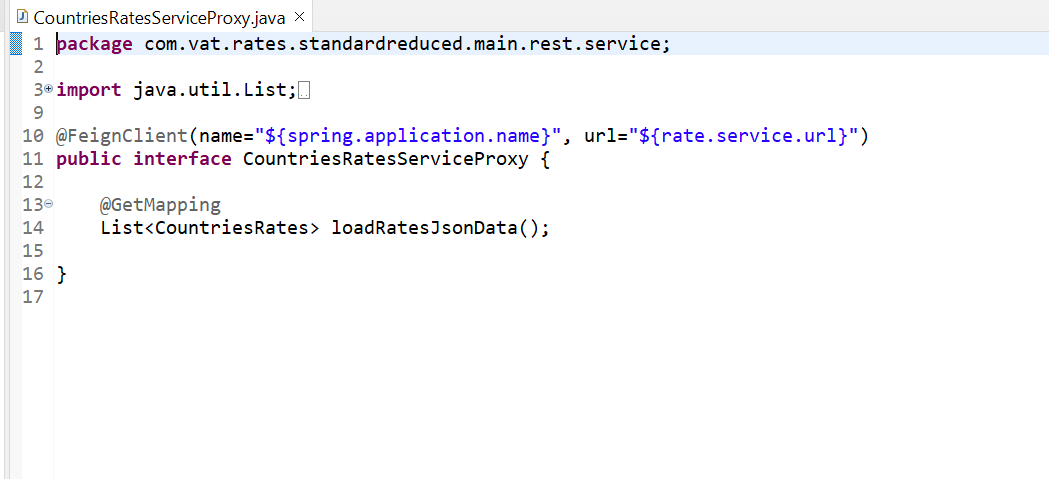
1. **StandardReducedRatesServiceApplication.java** – Used to bootsrap the spring boot main class



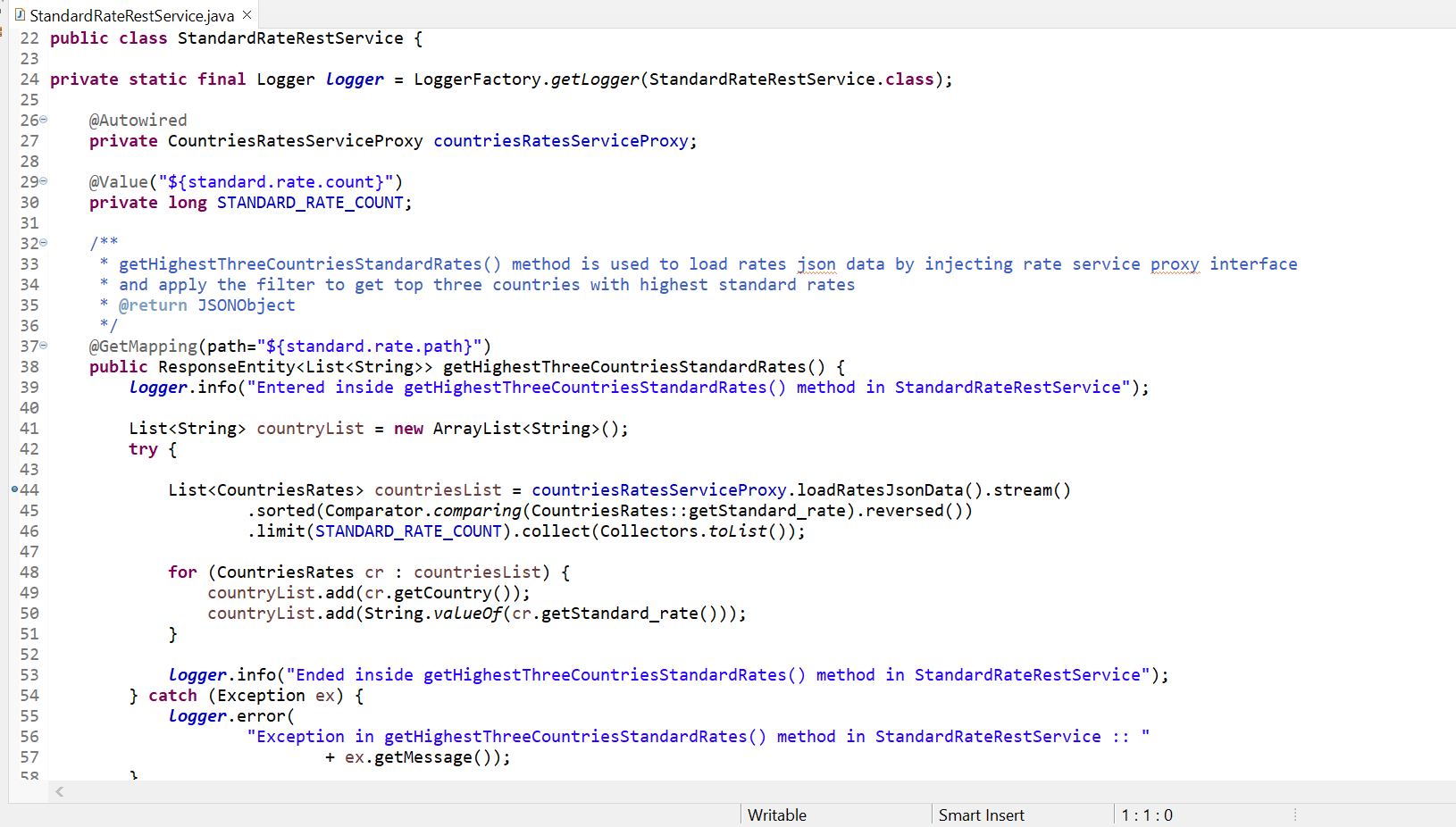
1. **CountryRates.java** – Used to maintain getter/setters in model bean class for the parameters Country, Standard Rate and Reduced Rate



1. **CountriesRatesServiceProxy.java** – This interface is used to invoke external client service to load json data with Feign Client interface



1. **StandardRateRestService.java** – This class is used to inject proxy interface and apply business logic for standard rates of top highest three countries from response



1. **ReducedRateRestService.java** – This class is used to inject proxy interface and apply business logic for top lowest three countries of reduced rates from response

