Academia Tehnică Militară "Ferdinand I“
Facultatea de Sisteme Informatice şi Securitate Cibernetică

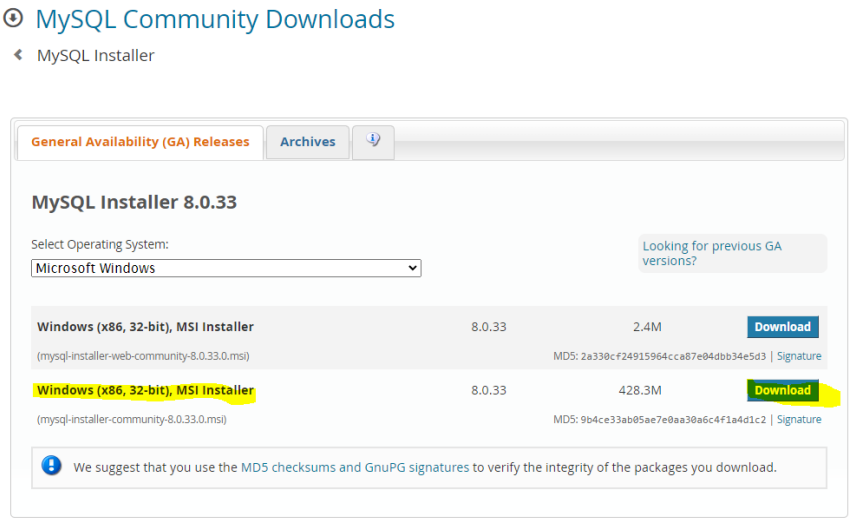
Paradigme de programare (în Java)

Lab 10/Curs 10- Servicii web REST

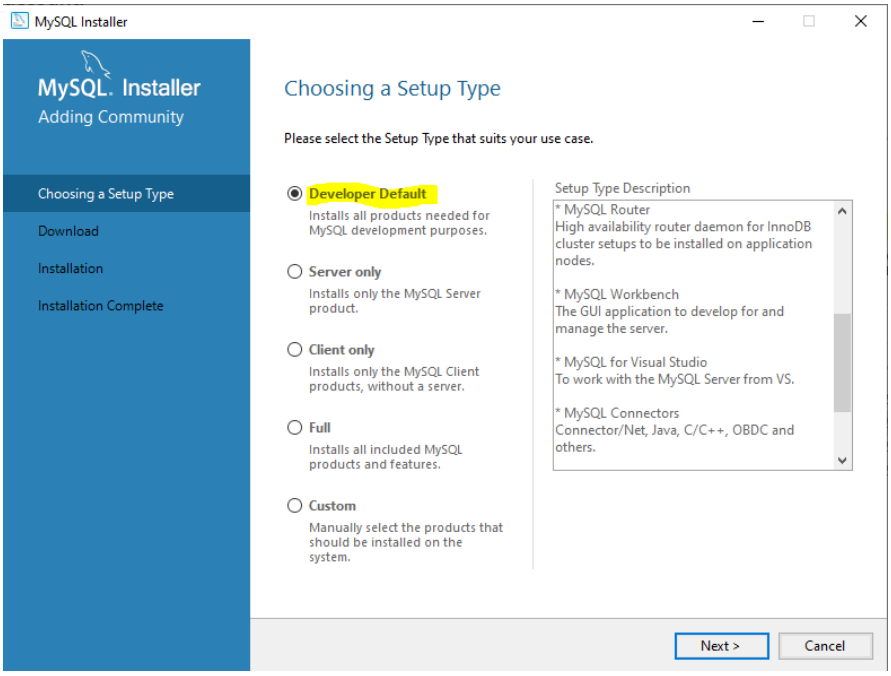
Col(r) Traian Nicula

A. Instalare MySQL

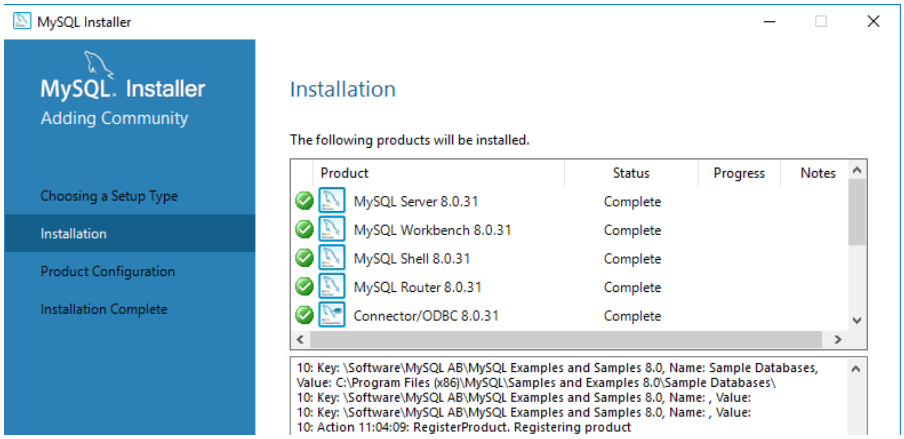
- Se descarcă kitul MySql de la https://dev.mysql.com/downloads/installer/



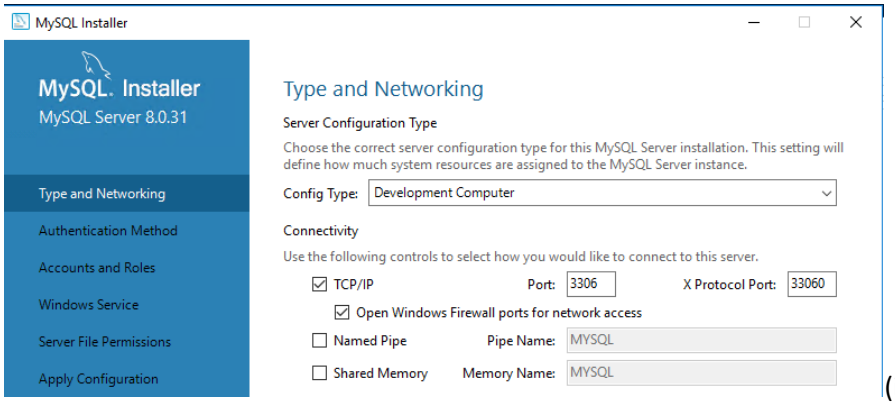
- Se instalează cu opțiunea *Developer Default*



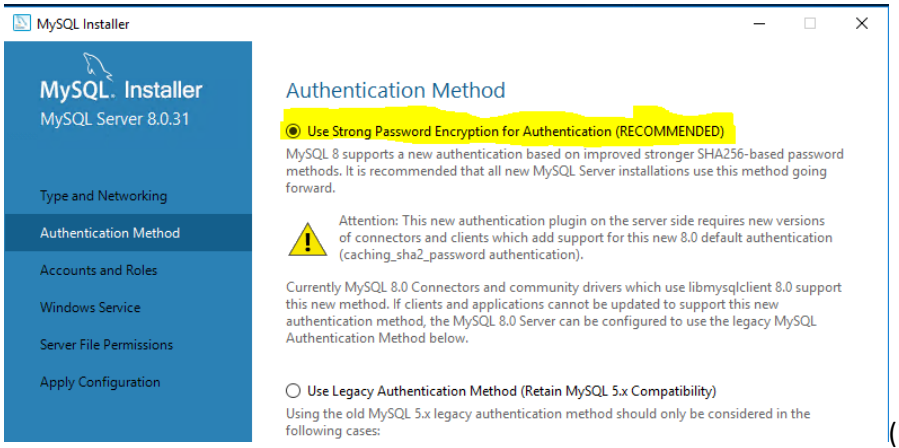
- Se instalează componentele



- Se configurează serviciul și alte componente

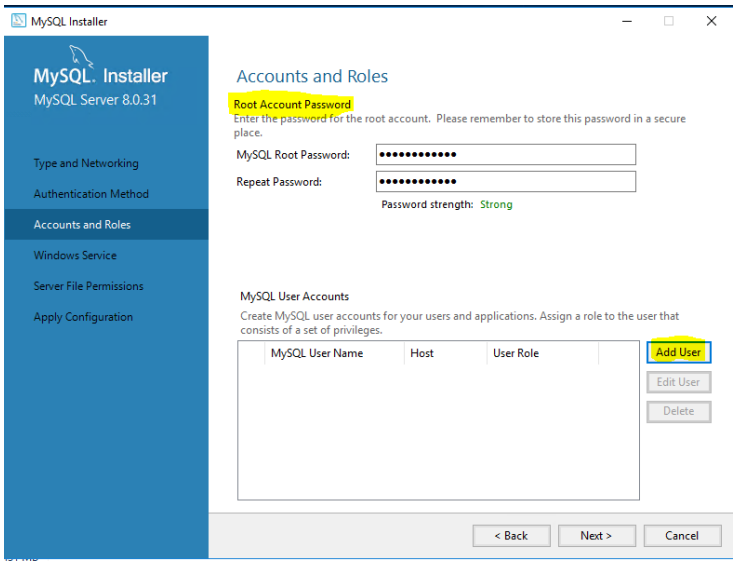


(implicit)

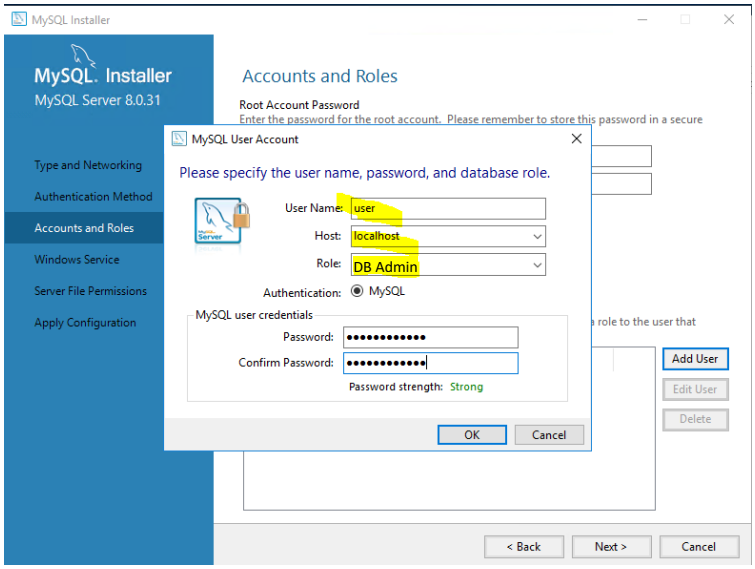


(implicit)

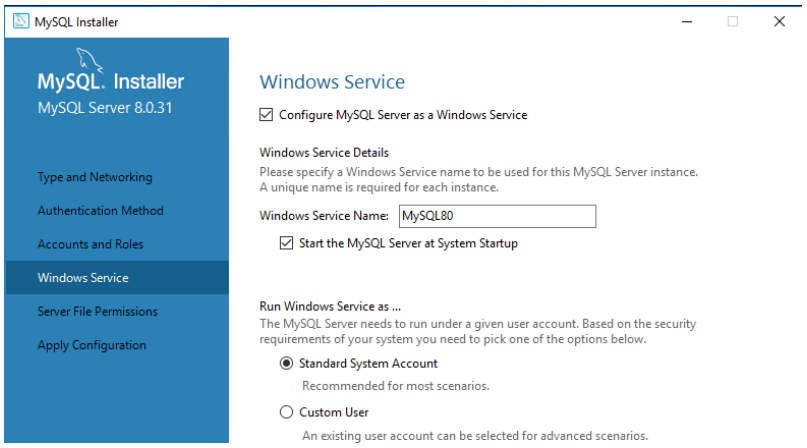
- Se stabilește parola pentru ***root*** (12 caractere)

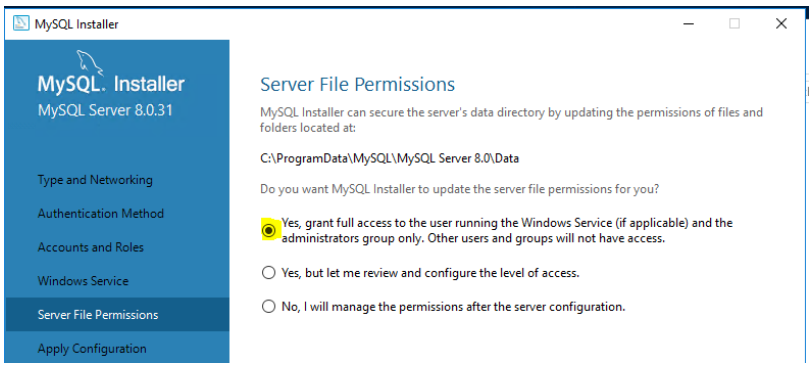


- Se adaugă un utilizator (user)

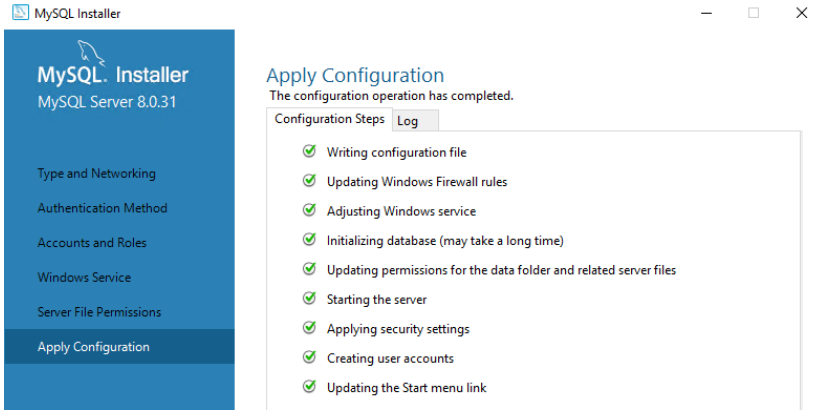


- Celelalte ecrane păstrează valorile implicite

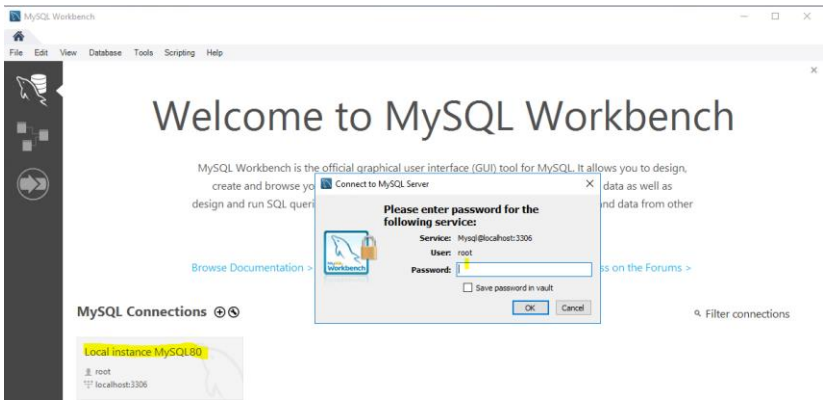




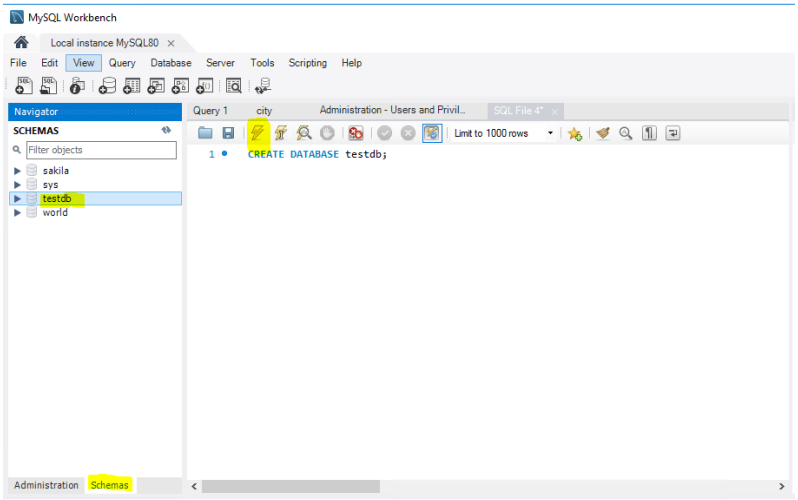
- Finalizarea configurării arată ca mai jos



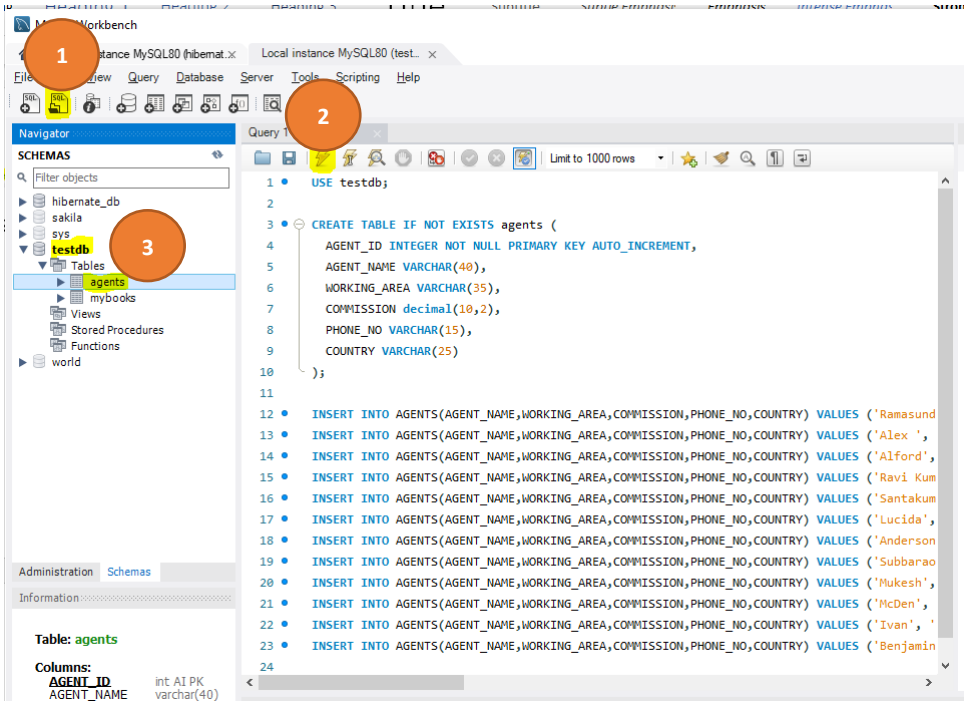
- Se lansează ***MySQL Workbench*** și ne conectăm ca ***root***



- Se creează baza de date ***testdb*** (CREATE DATABASE testdb;)



- Se rulează scriptul ***lab10.sql*** în ***MySQL Workbench*** urmând pașii 1 și 2 marcați pe imaginea de mai jos. Rezultatul se vede la 3.



B. Se creează proiectul Maven ***lab10*** folosind ca artefact ***jersey-quickstart-webapp*** din grupul ***org.glassfish.jersey.archetypes*** (la fel ca la laboratorul 9) care se salvează în folder-ul cu numele studentului.
   - Se deschide fișierul pom.xml, adaugă fragmentele xml care asigură dependențele pentru ***Jackson***, ***MyBatis***, conector ***MySQL*** și se salvează:

```
        <!-- uncomment this to get JSON support  -->
        <dependency>
            <groupId>org.glassfish.jersey.media</groupId>
            <artifactId>jersey-media-json-binding</artifactId>
        </dependency>
         <!--
https://mvnrepository.com/artifact/com.fasterxml.jackson.jaxrs/jackson-
jaxrs-json-provider -->
        <dependency>
            <groupId>com.fasterxml.jackson.jaxrs</groupId>
            <artifactId>jackson-jaxrs-json-provider</artifactId>
            <version>2.13.4</version>
        </dependency>
        <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->
        <dependency>
            <groupId>org.mybatis</groupId>
            <artifactId>mybatis</artifactId>
            <version>3.5.11</version>
        </dependency>
```

```
        <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java
-->
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.31</version>
        </dependency>
```

https://mvnrepository.com/artifact/com.fasterxml.jackson.jaxrs/jackson-jaxrs-json-provider/2.13.4

https://mvnrepository.com/artifact/org.mybatis/mybatis/3.5.11

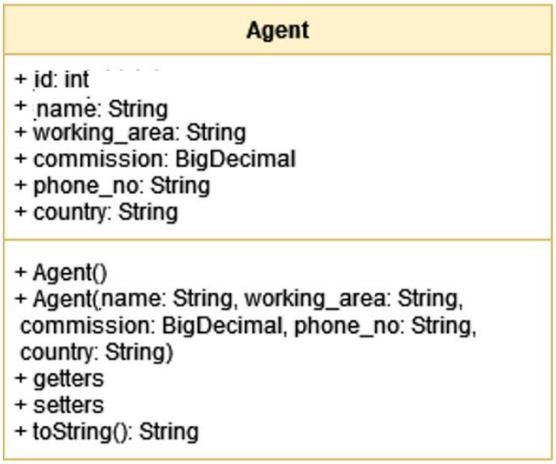
https://mvnrepository.com/artifact/mysql/mysql-connector-java/8.0.31

- În folder-ul *src/main/resources* se creează fișierul de configurare ***mybatis-config.xml*** cu datele de conectare la baza de date MySQL. Parola este cea stabilită la instalare.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
    <environments default="development">
        <environment id="development">
            <transactionManager type="JDBC"/>
            <dataSource type="POOLED">
                <property name="driver" value="com.mysql.jdbc.Driver"/>
                <property name="url"
value="jdbc:mysql://localhost:3306/testdb"/>
                <property name="username" value="user"/>
                <property name="password" value="qaz123QAZ!@#"/>
            </dataSource>
        </environment>
    </environments>
</configuration>
```

Se rezolvă următoarele exerciții:

1. Scrieți un program Java care: (**2.5p**)
   - creează clasa ***Agent*** conform diagramei UML de mai jos



- creează interfața Mapper care mapează metodele: *List<Agent> getAgents()*, *Agent getAgentById(int id)*, *void insertAgent(Agent agent)*, void *updateAgent(Agent agent)*, *void deleteAgentById(int id*)
  la operațiile SQL CRUD corespunzătoare
- creează clasa ***Tools*** cu metoda statică *SqlSessionFactory getSqlSessionFactory()* care returnează o instanță a *SqlSessionFactory.*

2. Scrieți un program Java care: (**2.5p**)
   - creează clasa resursă ***AgentsResource*** cu URI **/agents** având metoda *Response getAgents()* care:
     a. poate fi accesată cu metoda **GET** la URI **/agents**
     b. returnează reprezentarea JSON a listei agenților cu codul de stare 200 OK sau un mesaj de eroare în format JSON cu codul de stare 500 (vezi curs)
   - generează pachetul (***mvn clean package)*** și testează rezultatul cu în browser.
3. Scrieți un program Java care: (**2.5p**)
   - adaugă la clasa resursă ***AgentsResource*** metoda *Response getAgentById(int id)* care poate fi accesată cu metoda **GET** la URI **/ agents/{id}** și returnează reprezentarea agentului în format JSON cu codul de stare 200 OK sau un mesaj de eroare în format JSON cu codul de stare 500 (vezi curs)
   - generează pachetul (***mvn clean package)*** și testează rezultatul în browser.
4. Scrieți un program Java care: (**2.5p**)
   - adaugă la clasa resursă ***AgentsResource*** metoda *Response addAgent (Agent agent)* care poate fi accesată cu metoda **POST** la URI **/agents** care:
     a. adaugă un agent nou în baza de date
     b. returnează un mesaj de succes în format JSON cu codul de stare 201 CREATED sau un mesaj de eroare în format JSON cu codul de stare 500 (vezi curs)
   - generează pachetul (***mvn clean package)*** și testează rezultatul cu ***curl***.

Temă pentru acasă:

5. Scrieți un program Java care: (**2.5p**)
   - adaugă la clasa resursă ***AgentsResource*** metoda *Response modifyAgent (int id, Agent agent)* care poate fi accesată cu metoda **PUT** la URI **/agents/{id}** care:

   a. modifică un agent cu id specificat
   b. returnează un mesaj de succes în format JSON cu codul de stare 201 CREATED sau un mesaj de eroare în format JSON cu codul de stare 500 (vezi curs)
- generează pachetul (***mvn clean package)*** și testează rezultatul cu ***curl***.

6. Scrieți un program Java care: (**2.5p**)
   - adaugă la clasa resursă ***AgentsResource*** metoda *Response deleteAgent (int id)* care poate fi accesată cu metoda **DELETE** la URI **/agents/{id}** care:
     a. șterge din baza de date agentul cu id specificat
     b. returnează un mesaj de succes în format JSON cu codul de stare 200 OK sau un mesaj de eroare în format JSON cu codul de stare 500 (vezi curs)
   - generează pachetul (***mvn clean package)*** și testează rezultatul cu ***curl***.