Rest Web Service with Spring boot

The goal of this lab work is to develop a small application for cars renting.

The functionalities to be implemented are:

- Get a list of unrented cars
- Rent a car
- Get back a car

Required software

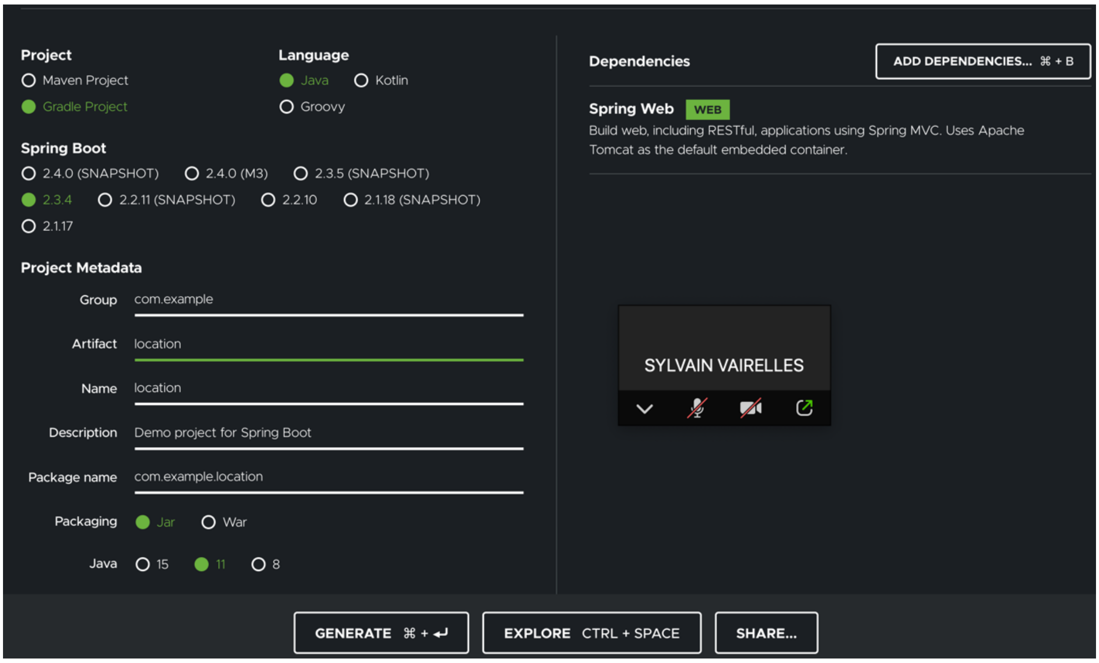
Java (JDK version)

Eclipse, Intellij

Gradle project creation

Use the Spring Initializer (https://start.spring.io/) to create a project:

- Don't forget to choose Gradle Project
- Give it a name (Articfact)
- Add Spring Web as dependencies (library)
- Don't forget to select the Java version installed on your machine (version 11 recommended) – Open a terminal and type "java -version" to know which version is installed on your machine.



Project
○ Maven Project
● Gradle Project

Language
● Java ○ Kotlin
○ Groovy

Spring Boot
○ 2.4.0 (SNAPSHOT) ○ 2.4.0 (M3) ○ 2.3.5 (SNAPSHOT)
● 2.3.4 ○ 2.2.11 (SNAPSHOT) ○ 2.2.10 ○ 2.1.18 (SNAPSHOT)
○ 2.1.17

Project Metadata

Group: com.example
Artifact: location
Name: location
Description: Demo project for Spring Boot
Package name: com.example.location
Packaging: ● Jar ○ War
Java: ○ 15 ● 11 ○ 8

Dependencies

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Download and unzip the project (outside the Eclipse workspace)

Open the project inside Eclipse (import gradle project) or Intellij (open project).

Create a new class (along with the main program into src/main/java/package…):

```
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HelloService {

	@GetMapping (value="/")
	public String hello() {
		return "hello";
	}
}
```

Launch the main program: /src/main/java/package…/*Application.java

Open your web browser at: http://localhost:8080

Spring boot coding

Write a class annotated with Controller implementing the car rental service (see https://github.com/charroux/CarService for an example).

This class provides two methods:

- Http Get on URI …/cars and return a list of cars
- Http Post on URI …cars with a json Http boby {"plateNumber":"55DD77", "brand": "Mazerati", "price":2000}

Then code the following features (find at the next paragraph examples):

- Get the features of a car:
  - URI: …/cars/plateNumber
  - http GET
  - Json response: { " plateNumber" : "11AA22" , " brand" : “Ferrari, "price" : 100 }
- Rent a car:
  - URI: …/plateNumber?rent=true
  - http PUT
  - Send Json inside the http body: { "begin" : "11/11/2017" , " end" : "1/1/2018" }
- Get back the car:
  - URI: …/plateNumber?rent=false
  - http PUT

Advice: you can use the following templates.

```
        @RequestMapping(value = "/cars", method = RequestMethod.GET)
        @ResponseStatus(HttpStatus.OK)
        @ResponseBody
        public List<Car> listOfCars(){
        }

        @RequestMapping(value = "/cars/{plateNumber}", method = RequestMethod.GET)
        @ResponseStatus(HttpStatus.OK)
        @ResponseBody
        public Car aCar(@PathVariable("plateNumber") String plateNumber) throws Exception{
        }

        @RequestMapping(value = "/cars/{plateNumber}", method = RequestMethod.DELETE)
        @ResponseStatus(HttpStatus.OK)
        public void getBack(@PathVariable("plateNumber") String plateNumber) throws Exception{
        }

        @RequestMapping(value = "/cars/{plateNumber}", method = RequestMethod.PUT)
        @ResponseStatus(HttpStatus.OK)
        public void rent(@PathVariable("plateNumber") String plateNumber) throws Exception{
        }

        @RequestMapping(value = "/voiture/{plateNumber}", method = RequestMethod.PUT)
        @ResponseStatus(HttpStatus.OK)  public void
rentAndGetBack(@PathVariable("plateNumber") String plateNumber,
@RequestParam(value="rent", required = true)boolean rent) throws Exception{
        }

        @RequestMapping(value = "/cars/{plateNumber}", method = RequestMethod.PUT)
        public void rent(@PathVariable("plateNumber") String plateNumber, @RequestParam(value="rent",
required = true)boolean rent, @RequestBody Dates dates){

        }
```

Test your application

Test the web service inside a web browser: http://localhost:8080/cars

Use a plugin for web browser like RestClient or Postman to test your application.