Spring Boot and RESTful Web Services

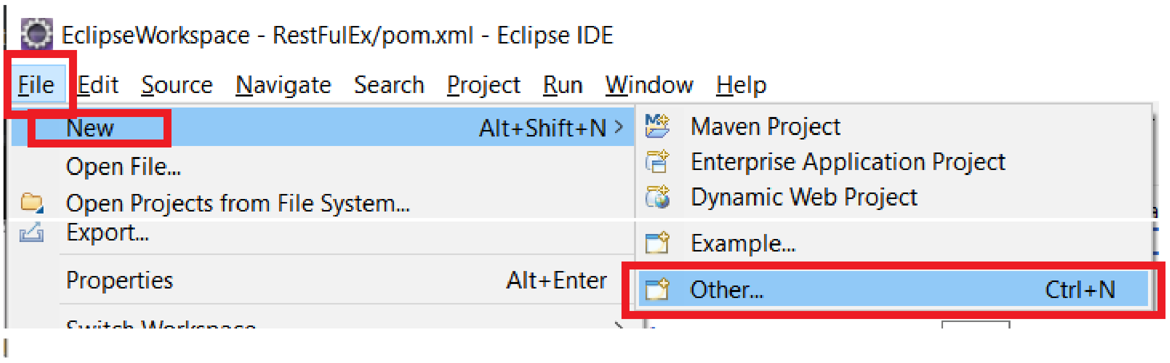
1. Write a program to create a simple Spring Boot application that prints a message.
2. Write a program to demonstrate RESTful Web Services with spring boot

Steps to Create a Spring Boot Project

Note : Make sure you have installed the Spring Plugin in Eclipse Itself.

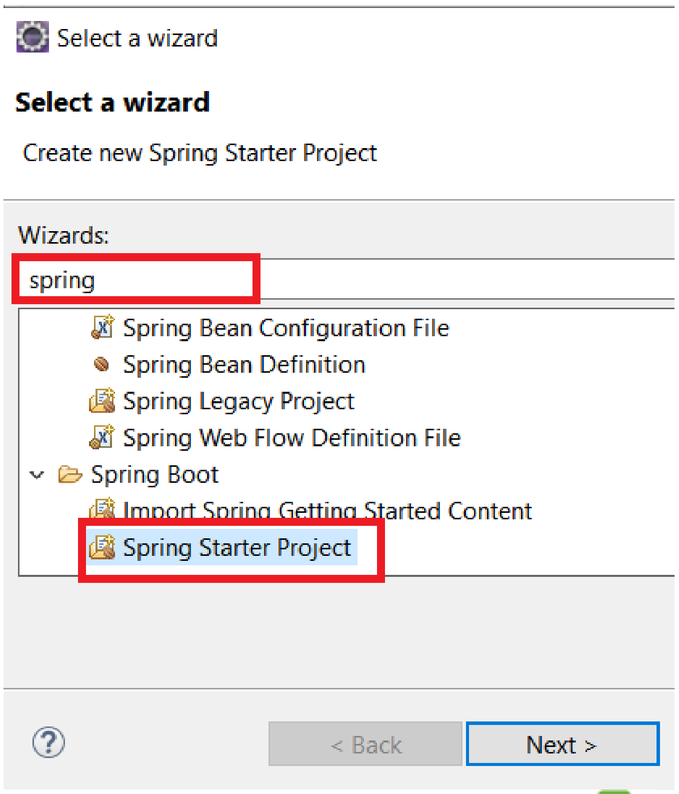
Step 1 :

1.1 : Open Eclipse. Go To File > New > Other.



1.2 : Search for ‘Spring’ and Select ‘Spring Starter Project’. Then Click on Next.

On Next Wizard, Choose your Project Name, and other parameters such as Group ID, Artifact ID. Then Choose Next.


Select a wizard

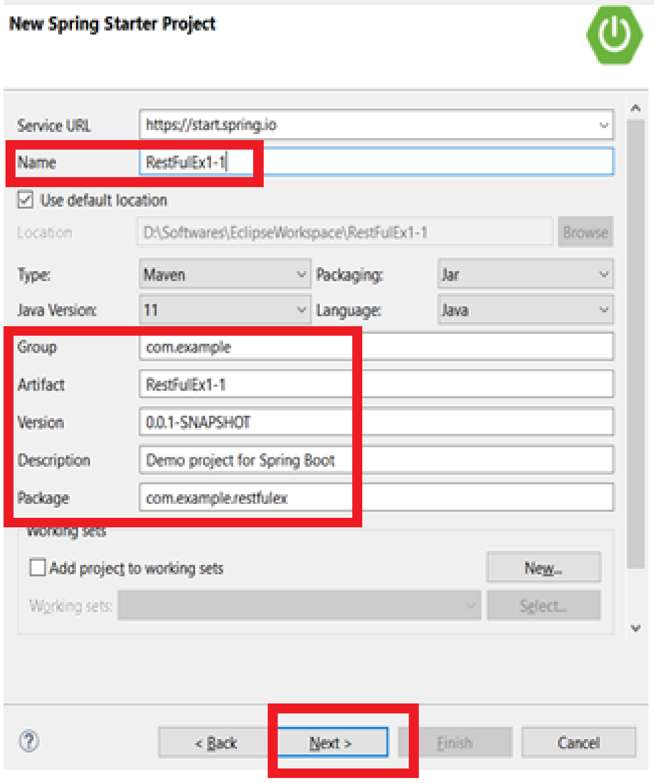
Select a wizard

Create new Spring Starter Project

Wizards:

spring

- Spring Bean Configuration File
- Spring Bean Definition
- Spring Legacy Project
- Spring Web Flow Definition File
- Spring Boot
  - Import Spring Getting Started Content
  - Spring Starter Project


New Spring Starter Project

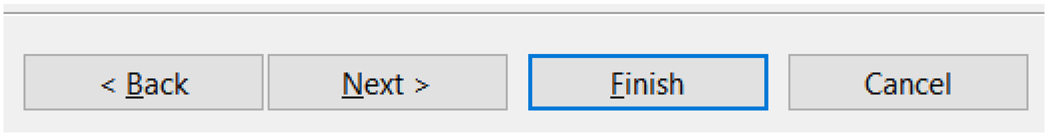
Service URL	https://start.spring.io
Name	RestFulEx1-1
☑ Use default location	
Location	D:\Softwares\EclipseWorkspace\RestFulEx1-1
Type:	Maven
Packaging:	Jar
Java Version:	11
Language:	Java
Group	com.example
Artifact	RestFulEx1-1
Version	0.0.1-SNAPSHOT
Description	Demo project for Spring Boot
Package	com.example.restfulex

Working sets

☐ Add project to working sets

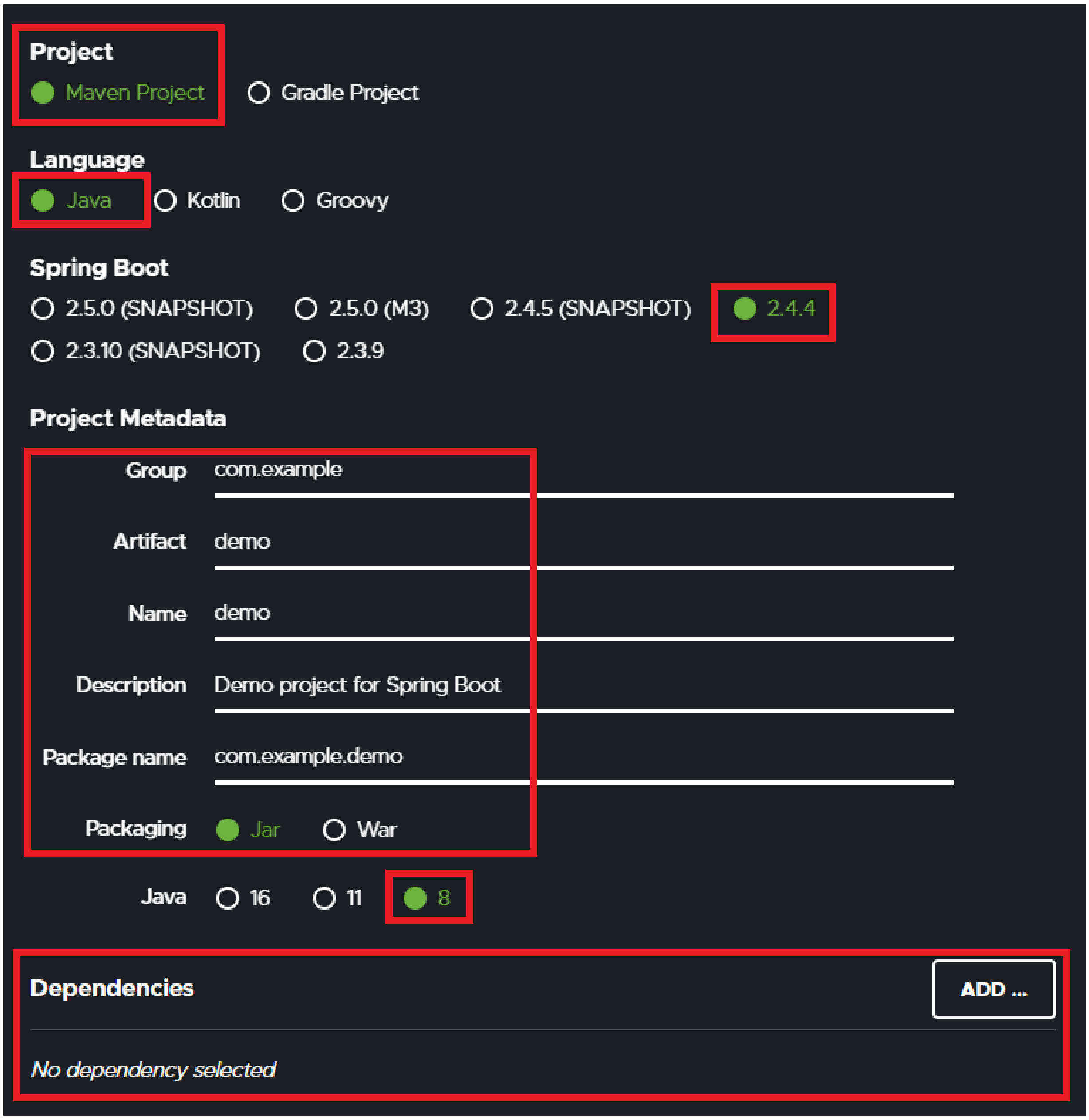
Working sets:

1.3 On next wizards, just click on “Finish”, once it is available.



Step 2 : Go to https://start.spring.io/

Select All the Options specific to your Machine and Java Version.



Project
[✓ Maven Project] Gradle Project

Language
[✓ Java] Kotlin Groovy

Spring Boot
2.5.0 (SNAPSHOT) 2.5.0 (M3) 2.4.5 (SNAPSHOT) [✓ 2.4.4]
2.3.10 (SNAPSHOT) 2.3.9

Project Metadata

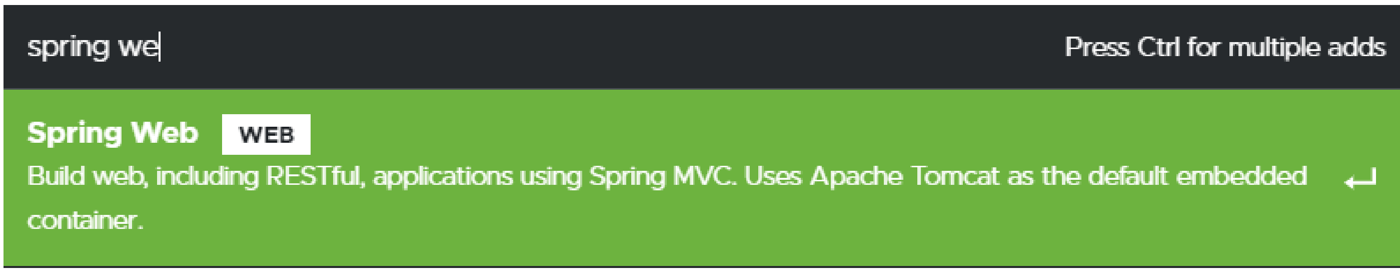
Group	com.example
Artifact	demo
Name	demo
Description	Demo project for Spring Boot
Package name	com.example.demo
Packaging	[✓ Jar] War
Java	16 11 [✓ 8]

Dependencies — ADD ...

No dependency selected

Selection of Dependencies is to be done as per Project Requirement :

For Ex. Lets add Spring Web Dependeny.

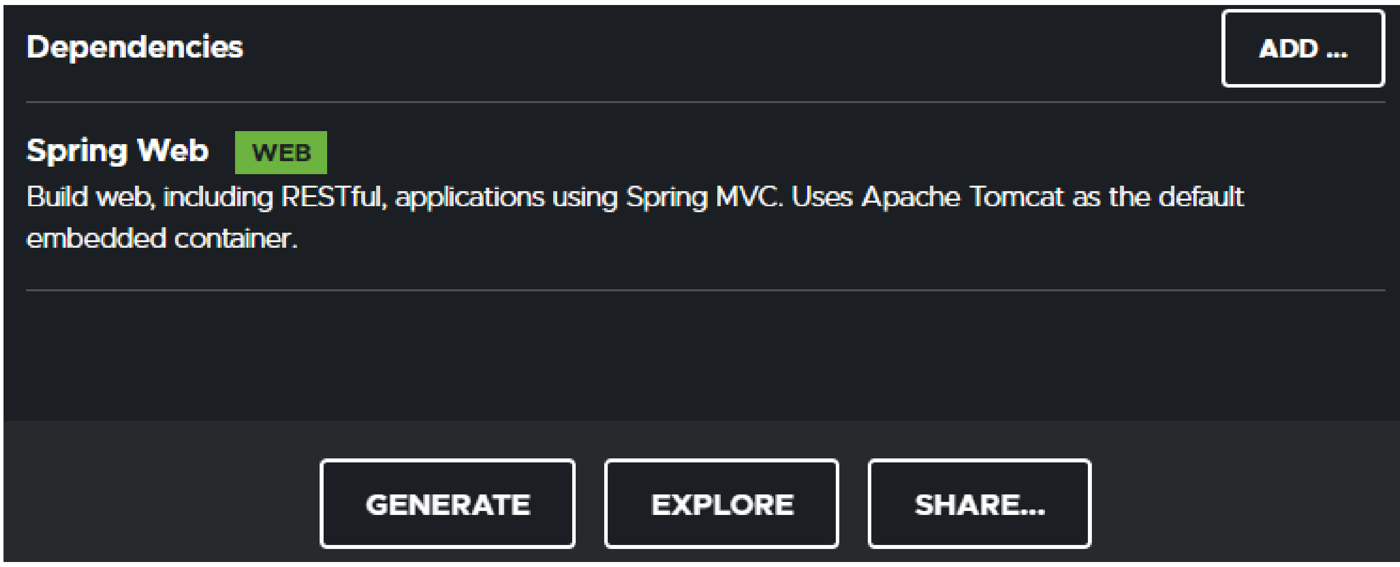


spring we

Press Ctrl for multiple adds

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

Click On Generate, a zip file will be downloaded.

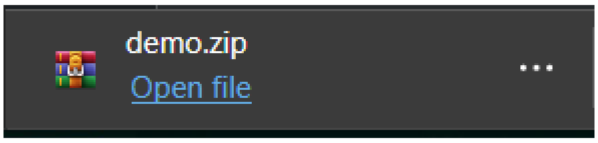

Dependencies

ADD ...

Spring Web WEB

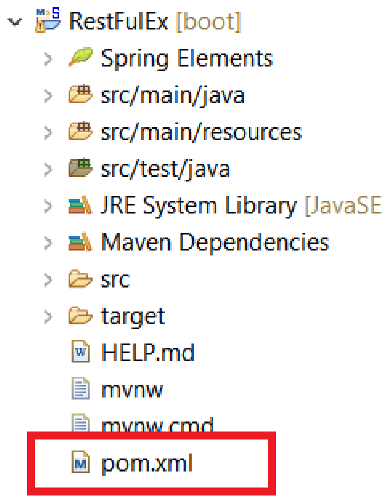
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

GENERATE EXPLORE SHARE...



Unzip the downloaded zip file and open the pom.xml file inside the demo folder.

Copy the Contents of the pom.xml file & paste it in the pom.xml file of our created project from step 1.


```
RestFulEx [boot]
  Spring Elements
  src/main/java
  src/main/resources
  src/test/java
  JRE System Library [JavaSE
  Maven Dependencies
  src
  target
  HELP.md
  mvnw
  mvnw.cmd
  pom.xml
```

Save the file, an automatic download process will start, wait till its completed.

Now you are good to go and develop Spring Boot Applications.

Problem Statement 1 : Write a program to create a simple Spring Boot application that prints a message.

Solution :

BoothelloApplication.java

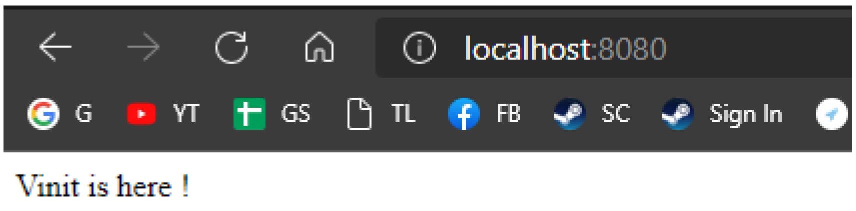
package com.example.demo;

```
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class BoothelloApplication {

	public static void main(String[] args) {
		SpringApplication.run(BoothelloApplication.class, args);
	}

}
```

HelloWorldController.java

```
package com.example.demo;

import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HelloWorldController {
	@RequestMapping("/")
	public String hello()
	{
		return "Vinit is here !";
	}
}
```

Output :



```
Vinit is here !
```

Problem Statement 2 : Write a program to demonstrate RESTful Web Services with spring boot

Solution :

HelloWorldBean.java

```
package com.example.demo;

publicclass HelloWorldBean {

	public String message;
	//constructor of HelloWorldBean
	public HelloWorldBean(String message)
```

```
        {
        this.message=message;
        }
        //generating getters and setters
        public String getMessage()
        {
        return message;
        }
        publicvoid setMessage(String message)
        {
        this.message = message;
        }
        @Override
        //generate toString
        public String toString()
        {
        return String.format ("HelloWorldBean [message=%s]", message);
        }
}
```

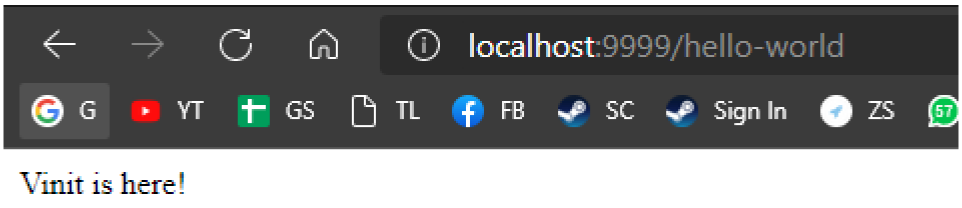
HelloWorldController.java

```
package com.example.demo;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
//Controller
@RestController
public class HelloWorldController
{
//using get method and hello-world as URI
        @GetMapping(path="/hello-world")
public String helloWorld()
{
return "Vinit is here!";
}
@GetMapping(path="/hello-world-bean")
public HelloWorldBean helloWorldBean()
{
return new HelloWorldBean("Kaise ho? xD"); //constructor of HelloWorldBean  }  }
```

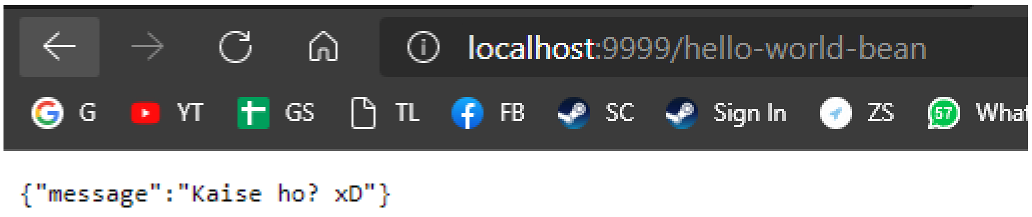
RestfulwebserviceApplication.java

```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class RestfulwebserviceApplication {

        public static void main(String[] args) {
                SpringApplication.run(RestfulwebserviceApplication.class, args);
        }

}
```

Output :


localhost:9999/hello-world

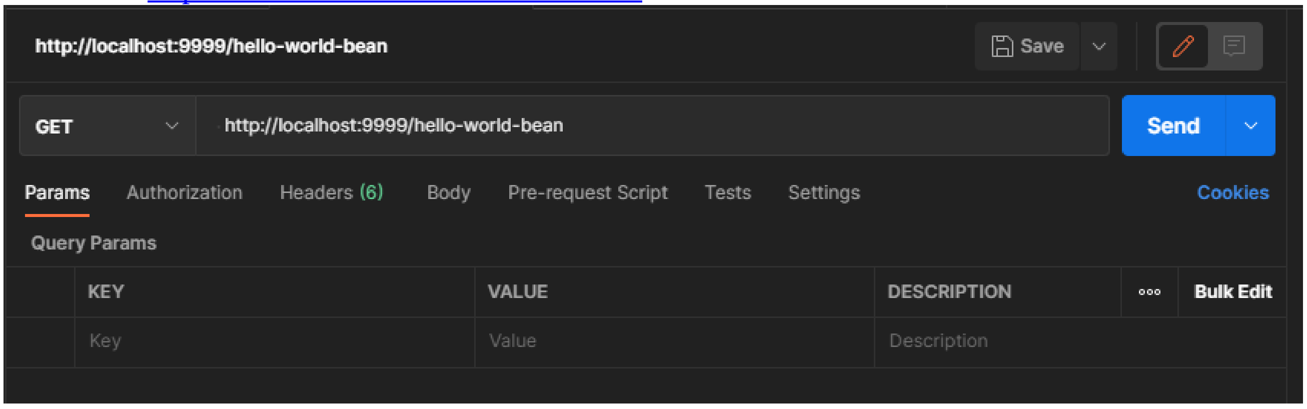
Vinit is here!


localhost:9999/hello-world-bean

```
{"message":"Kaise ho? xD"}
```

Testing API with PostMan.

EndPoint : http://localhost:9999/hello-world-bean


http://localhost:9999/hello-world-bean

GET http://localhost:9999/hello-world-bean

KEY	VALUE	DESCRIPTION
Key	Value	Description