WEEK 4

Taniya G Remula

St. Joseph's Institute of Technology

Superset ID: 6376013

Spring-Boot

Exercise 1: Demonstrate creation of Spring Boot Application and explain benefits

Created using Spring Initializr https://start.spring.io

Group: com.cognizant | Artifact: spring-learn Dependencies: Spring Web, DevTools

Main Class: SpringLearnApplication.java

Benefits: Embedded Tomcat, minimal XML config, fast dev, no boilerplate

Exercise 2: Demonstrate loading bean from spring configuration file (country.xml)

Created country.xml in src/main/resources

Defined <bean> with id="country" and properties code, name

Used ClassPathXmlApplicationContext to load XML and get bean

Printed using toString()

Directory Structure:

src/main/java/com/cognizant/springlearn/...

src/main/resources/country.xml, application.properties

1. Country.java package

com.cognizant.springlearn.model; public

class Country { private String code;

private String name; public Country() {

System.out.println("DEBUG: Inside Country Constructor");

}

public String getCode() {

System.out.println("DEBUG: getCode() called");

return code;

}

```
    public void setCode(String code) {
        System.out.println("DEBUG: setCode() called");
this.code = code;
    }
    public String getName() {
        System.out.println("DEBUG: getName() called");
return name;
    }
    public void setName(String name) {
        System.out.println("DEBUG: setName() called");
this.name = name;
    }
    @Override     public
String toString() {
return "Country{" +
                "code='" + code + '\'' +
                ", name='" + name + '\'' +
                '}';
    }
}
```

2. country.xml (placed inside src/main/resources)

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">
    <bean id="country" class="com.cognizant.springlearn.model.Country">
        <property name="code" value="IN"/>
        <property name="name" value="India"/>
    </bean>
```

</beans>

3. SpringLearnApplication.java package com.cognizant.springlearn; import com.cognizant.springlearn.model.Country; import org.slf4j.Logger; import org.slf4j.LoggerFactory; import org.springframework.boot.SpringApplication; import org.springframework.boot.autoconfigure.SpringBootApplication; import org.springframework.context.ApplicationContext; import org.springframework.context.support.ClassPathXmlApplicationContext;

```
@SpringBootApplication public class
SpringLearnApplication {
   private static final Logger LOGGER =
LoggerFactory.getLogger(SpringLearnApplication.class);
public static void main(String[] args) {
      SpringApplication.run(SpringLearnApplication.class, args);
displayCountry();
   }
   public static void displayCountry() {
      ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");
      Country country = context.getBean("country", Country.class);
      LOGGER.debug("Country : {}", country.toString());
   }
}
```

Exercise:3 Write REST service that returns Hello World

Created HelloController.java

Mapped GET /hello to sayHello() method

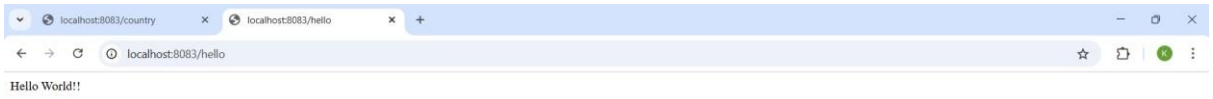
Returns "Hello World!!"

Tested via browser & Postman

4. HelloController.java package com.cognizant.springlearn.controller; import org.slf4j.Logger; import org.slf4j.LoggerFactory; import

```
org.springframework.web.bind.annotation.GetMapping; import
org.springframework.web.bind.annotation.RestController;
@RestController public class HelloController {     private static final Logger LOGGER =
LoggerFactory.getLogger(HelloController.class);
   @GetMapping("/hello")
public String sayHello() {
       LOGGER.info("Start sayHello()");
LOGGER.info("End sayHello()");         return
"Hello World!!";
   }
}
```

Output:



Exercise 4:Create REST API to return India country object from country.xml

Used ApplicationContext to load country.xml Retrieved

country bean and returned in JSON

Code:

CountryController.java package com.cognizant.springlearn.controller; import

com.cognizant.springlearn.model.Country; import

com.cognizant.springlearn.service.CountryService; import

```
org.springframework.beans.factory.annotation.Autowired; import
org.springframework.context.ApplicationContext; import
org.springframework.context.support.ClassPathXmlApplicationContext; import
org.springframework.web.bind.annotation.*; import java.util.List;
@RestController public class CountryController {
   @Autowired     private CountryService
countryService;
@RequestMapping("/country")     public
Country getCountryIndia() {
       ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");
return context.getBean("country", Country.class);
   }
   @GetMapping("/countries/{code}")     public Country
getCountry(@PathVariable String code) throws Exception {         return
countryService.getCountry(code);
   }
}
```

Output:

Exercise 5:Implement getCountry service with dynamic code (case-insensitive)

countryService.getCountry(code)

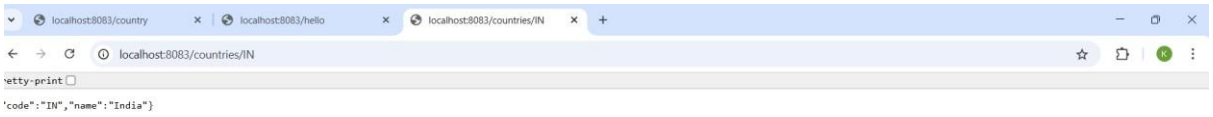
Filters country list based on input code ignoring case

Code:

6. CountryService.java package com.cognizant.springlearn.service; import com.cognizant.springlearn.model.Country; import org.springframework.context.ApplicationContext; import org.springframework.context.support.ClassPathXmlApplicationContext; import org.springframework.stereotype.Service; import java.util.ArrayList; import java.util.List;

```
@Service public class CountryService {     public Country
getCountry(String code) throws Exception {
      ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");
List<Country> countries = new ArrayList<>();         countries.add(context.getBean("country",
Country.class)); // Add more beans if defined         return countries.stream()
            .filter(country -> country.getCode().equalsIgnoreCase(code))
            .findFirst()
            .orElseThrow(() -> new Exception("Country Not Found"));
   }
}
```

Output:



7. application.properties server.port=8083

logging.level.root=DEBUG

8. pom.xml - dependencies section

```
<!-- Spring Web -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>
<!-- Logging -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-logging</artifactId>
</dependency>
<!-- Spring Boot DevTools -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-devtools</artifactId>
    <scope>runtime</scope>
```

```
</dependency>
<!-- Spring Context for XML config -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
</dependency>
```