**Week 4 Hands-On Assignment**

**Exercise 1: Create a Spring Web Project Using Maven**

**Steps:** 1. Go to <https://start.spring.io> 2. Change Group as com.cognizant 3. Change Artifact Id as spring-learn 4. Select “Spring Boot DevTools” and “Spring Web” 5. Create and download the project as a zip 6. Extract the zip in root folder to Eclipse Workspace 7. Build the project using the following command:

mvn clean package -Dhttp.proxyHost=proxy.cognizant.com -Dhttp.proxyPort=6050 -Dhttps.proxyHost=proxy.cognizant.com -Dhttps.proxyPort=6050 -Dhttp.proxyUser=123456

1. Import the project in Eclipse:
   * File > Import > Maven > Existing Maven Projects > Browse to extracted folder > Finish
2. Add logs in SpringLearnApplication to verify if the main() method is called.
3. Run the SpringLearnApplication class.

**Project Structure Overview:** - src/main/java – Folder with application code - src/main/resources – Folder for application configuration - src/test/java – Folder for test code - SpringLearnApplication.java – Main class - @SpringBootApplication – Indicates a configuration class that declares one or more @Bean methods and triggers auto-configuration and component scanning

**Output:**

Inside main

**Exercise 2: Spring Core – Load Country from Spring Configuration XML**

**Scenario:** An airline website supports booking in four countries. Country code and name should be displayed on a dropdown and are stored in Spring XML.

**Steps:** 1. Pick one country and configure it in country.xml.

<bean id="country" class="com.cognizant.springlearn.Country">  
 <property name="code" value="IN" />  
 <property name="name" value="India" />  
</bean>

1. Create Country.java:

public class Country {  
 private String code;  
 private String name;  
  
 public Country() {  
 LOGGER.debug("Inside Country Constructor.");  
 }  
  
 public String getCode() {  
 LOGGER.debug("Getting Code");  
 return code;  
 }  
  
 public void setCode(String code) {  
 LOGGER.debug("Setting Code");  
 this.code = code;  
 }  
  
 public String getName() {  
 LOGGER.debug("Getting Name");  
 return name;  
 }  
  
 public void setName(String name) {  
 LOGGER.debug("Setting Name");  
 this.name = name;  
 }  
  
 @Override  
 public String toString() {  
 return "Country{" + "code='" + code + '\'' + ", name='" + name + '\'' + '}';  
 }  
}

1. Add displayCountry() method in SpringLearnApplication.java:

ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  
Country country = context.getBean("country", Country.class);  
LOGGER.debug("Country : {}", country.toString());

1. Invoke displayCountry() in main().

**Output:**

Inside main  
Inside Country Constructor.  
Setting Code  
Setting Name  
Getting Code  
Getting Name  
Country : Country{code='IN', name='India'}

**Exercise 3: Hello World RESTful Web Service**

**Steps:** 1. Create controller: com.cognizant.springlearn.controller.HelloController

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

1. Access URL: http://localhost:8083/hello

**Sample Response:**

Hello World!!

**Exercise 4: REST - Country Web Service**

**Steps:** 1. Create controller: com.cognizant.springlearn.controller.CountryController

@RestController  
public class CountryController {  
  
 @RequestMapping("/country")  
 public Country getCountryIndia() {  
 ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  
 Country country = context.getBean("country", Country.class);  
 return country;  
 }  
}

**Sample Response:**

{  
 "code": "IN",  
 "name": "India"  
}

**Exercise 5: REST - Get Country by Code**

**Steps:** 1. Controller:

@GetMapping("/countries/{code}")  
public Country getCountry(@PathVariable String code) {  
 List<Country> countryList = context.getBean("countryList", List.class);  
 return countryList.stream()  
 .filter(c -> c.getCode().equalsIgnoreCase(code))  
 .findFirst()  
 .orElseThrow(() -> new CountryNotFoundException());  
}

1. Create service method getCountry(String code) in CountryService.

**Sample Request:**

http://localhost:8083/countries/in

**Sample Response:**

{  
 "code": "IN",  
 "name": "India"  
}

**Exercise 6: Create Authentication Service that Returns JWT**

**Steps:** 1. Send Request:

curl -s -u user:pwd http://localhost:8090/authenticate

1. Sample Response:

{  
 "token": "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiaWF0IjoxNTcwMzc5NDc0LCJleHAiOjE1NzAzODA2NzR9.t3LRvlCV-hwKfoqZYlaVQqEUiBloWcWn0ft3tgv0dL0"  
}

1. Steps to Implement:

* Create AuthenticationController
* Configure it in SecurityConfig
* Decode Authorization Header to retrieve credentials
* Generate token using JWT

**End of Week 4 Assignment**