WEEK4

# Hands on 1: Create a Spring Web Project using Maven

Question:

Create a Spring Web Project using Maven. Follow steps below to create a project:  
Go to https://start.spring.io/  
Change Group as “com.cognizant”  
Change Artifact Id as “spring-learn”  
Select Spring Boot DevTools and Spring Web  
Create and download the project as zip  
Extract the zip in root folder to Eclipse Workspace  
Build the project using mvn clean package ... command in command line  
Import the project in Eclipse  
Include logs to verify if main() method of SpringLearnApplication.  
Run the SpringLearnApplication class.

Code:

package com.cognizant.springlearn;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class SpringLearnApplication {  
 private static final Logger LOGGER = LoggerFactory.getLogger(SpringLearnApplication.class);  
  
 public static void main(String[] args) {  
 LOGGER.info("START");  
 SpringApplication.run(SpringLearnApplication.class, args);  
 LOGGER.info("END");  
 }  
}

Output:

Console:  
START  
...Spring Boot startup logs...  
END

# Hands on 4: Load Country from Spring Configuration XML

Question:

Write a program to store a country bean in a spring configuration XML and display the country details.   
Use ClassPathXmlApplicationContext to load and display the country.

Code:

country.xml:  
<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  
 https://www.springframework.org/schema/beans/spring-beans.xsd">  
 <bean id="country" class="com.cognizant.springlearn.Country">  
 <property name="code" value="IN"/>  
 <property name="name" value="India"/>  
 </bean>  
</beans>  
  
Country.java:  
public class Country {  
 private String code;  
 private String name;  
 public Country() {  
 LOGGER.debug("Inside Country Constructor");  
 }  
 // Getters and setters with logging  
}  
  
SpringLearnApplication.java:  
ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  
Country country = context.getBean("country", Country.class);  
LOGGER.debug("Country : {}", country.toString());

Output:

Console:  
Inside Country Constructor  
Country : Country{code='IN', name='India'}

# Hello World RESTful Web Service

Question:

Write a REST service in the spring-learn application that returns the text "Hello World!!" using Spring Web Framework.  
URL: /hello

Code:

@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!!";  
 }  
}

Output:

Browser/Postman:  
GET http://localhost:8083/hello  
Response: Hello World!!

# REST - Country Web Service

Question:

Write a REST service that returns India country details using a bean defined in Spring XML.  
URL: /country

Code:

@RestController  
public class CountryController {  
 private static final Logger LOGGER = LoggerFactory.getLogger(CountryController.class);  
  
 @RequestMapping("/country")  
 public Country getCountryIndia() {  
 LOGGER.info("START");  
 ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  
 Country country = context.getBean("country", Country.class);  
 LOGGER.info("END");  
 return country;  
 }  
}

Output:

Browser/Postman:  
GET http://localhost:8083/country  
Response: { "code": "IN", "name": "India" }

# Create authentication service that returns JWT

Question:

Create an authentication service that returns JWT token when user credentials are validated.  
URL: /authenticate

Code:

@RestController  
public class AuthenticationController {  
 private static final Logger LOGGER = LoggerFactory.getLogger(AuthenticationController.class);  
  
 @GetMapping("/authenticate")  
 public Map<String, String> authenticate(@RequestHeader("Authorization") String authHeader) {  
 LOGGER.info("START");  
 LOGGER.debug(authHeader);  
 String user = getUser(authHeader);  
 String token = generateJwt(user);  
 Map<String, String> map = new HashMap<>();  
 map.put("token", token);  
 LOGGER.info("END");  
 return map;  
 }  
  
 private String getUser(String authHeader) {  
 String encodedCredentials = authHeader.substring("Basic ".length());  
 byte[] decoded = Base64.getDecoder().decode(encodedCredentials);  
 return new String(decoded).split(":")[0];  
 }  
  
 private String generateJwt(String user) {  
 JwtBuilder builder = Jwts.builder();  
 builder.setSubject(user);  
 builder.setIssuedAt(new Date());  
 builder.setExpiration(new Date(System.currentTimeMillis() + 1200000));  
 builder.signWith(SignatureAlgorithm.HS256, "secretkey");  
 return builder.compact();  
 }  
}

Output:

curl -s -u user:pwd http://localhost:8090/authenticate  
Response: { "token": "<JWT token>" }