WEEK 4

Spring REST using Spring Boot 3

Create a Spring Web Project using Maven

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
        http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.cognizant</groupId>
  <artifactId>spring-learn</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>jar</packaging>

  <name>spring-learn</name>

  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.1.1</version>
  </parent>

  <properties>
    <java.version>17</java.version>
  </properties>

  <dependencies>
    <!-- Web support for building REST APIs -->
    <dependency>
      <groupId>org.springframework.boot</groupId>
```

```
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

  <!-- Developer tools for auto-restart and better dev experience -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-devtools</artifactId>
    <scope>runtime</scope>
  </dependency>

  <!-- Test support -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>
 </dependencies>

</project>

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) {
        SpringApplication.run(SpringLearnApplication.class, args);
        LOGGER.info("Application has been launched successfully.");
    }
}
package com.cognizant.springlearn.controller;

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

@RestController
public class HelloController {

    @GetMapping("/hello")
    public String displayMessage() {
        return "Hello from the Spring Learn project!";
    }
}
```

Output

http://localhost:8080/hello

Hello from the Spring Learn project!

Spring Core – Load Country from Spring Configuration XML

```
<?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.Country">
        <property name="code" value="IN"/>
        <property name="name" value="India"/>
    </bean>

</beans>
```

```
package com.cognizant.springlearn;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class Country {

    private static final Logger LOGGER = LoggerFactory.getLogger(Country.class);

    private String code;
    private String name;

    public Country() {
        LOGGER.debug("Country class constructor called.");
    }
```

```
    public String getCode() {
        LOGGER.debug("Accessed getCode() method.");
        return code;
    }

    public void setCode(String code) {
        LOGGER.debug("Setting code to: {}", code);
        this.code = code;
    }

    public String getName() {
        LOGGER.debug("Accessed getName() method.");
        return name;
    }

    public void setName(String name) {
        LOGGER.debug("Setting name to: {}", name);
        this.name = name;
    }

    @Override
    public String toString() {
        return "Country [code=" + code + ", name=" + name + "]";
    }
}
package com.cognizant.springlearn;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.context.ApplicationContext;
```

```
import org.springframework.context.support.ClassPathXmlApplicationContext;


public class SpringLearnApplication {


   private static final Logger LOGGER =
LoggerFactory.getLogger(SpringLearnApplication.class);


   public static void main(String[] args) {
      showCountryDetails();
   }


   public static void showCountryDetails() {
      ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");
      Country country = context.getBean("country", Country.class);
      LOGGER.debug("Retrieved Country Bean: {}", country);
   }
}
```

Output:

```
DEBUG com.cognizant.springlearn.Country - Country class constructor called.
DEBUG com.cognizant.springlearn.Country - Setting code to: IN
DEBUG com.cognizant.springlearn.Country - Setting name to: India
DEBUG com.cognizant.springlearn.Country - Accessed getCode() method.
DEBUG com.cognizant.springlearn.Country - Accessed getName() method.
DEBUG com.cognizant.springlearn.SpringLearnApplication - Retrieved Country Bean:
Country [code=IN, name=India]
```

Hello World RESTful Web Service

```
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()");
      String message = "Hello World!!";
      LOGGER.info("END - sayHello()");
      return message;
   }
}
```

```
package com.cognizant.springlearn;

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

@SpringBootApplication
public class SpringLearnApplication {
   public static void main(String[] args) {
      SpringApplication.run(SpringLearnApplication.class, args);
   }
}
```

Output:

Hello World!!

REST - Country Web Service

```
<?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.Country">
      <property name="code" value="IN" />
      <property name="name" value="India" />
   </bean>

</beans>

package com.cognizant.springlearn;

public class Country {
   private String code;
   private String name;

   public Country() {
      // No-arg constructor
   }

   public String getCode() {
      return code;
   }

   public void setCode(String code) {
      this.code = code;
   }

   public String getName() {
      return name;
   }

   public void setName(String name) {
      this.name = name;
   }
}
package com.cognizant.springlearn.controller;

import com.cognizant.springlearn.Country;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
```

```
@RestController
public class CountryController {

   private static final Logger LOGGER = LoggerFactory.getLogger(CountryController.class);

   @RequestMapping("/country")
   public Country getCountryIndia() {
      LOGGER.info("START - getCountryIndia()");

      ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");
      Country country = context.getBean("country", Country.class);

      LOGGER.info("END - getCountryIndia()");
      return country;
   }
}
```

Sample Request

http://localhost:8083/country

Response Output


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

REST - Get country based on country code

```
<?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="countryList" class="java.util.ArrayList">
      <constructor-arg>
         <list>
            <bean class="com.cognizant.springlearn.Country">
               <property name="code" value="IN"/>
               <property name="name" value="India"/>
            </bean>
            <bean class="com.cognizant.springlearn.Country">
               <property name="code" value="US"/>
               <property name="name" value="United States"/>
            </bean>
            <bean class="com.cognizant.springlearn.Country">
               <property name="code" value="DE"/>
               <property name="name" value="Germany"/>
            </bean>
            <bean class="com.cognizant.springlearn.Country">
               <property name="code" value="JP"/>
               <property name="name" value="Japan"/>
            </bean>
         </list>
      </constructor-arg>
   </bean>

</beans>


package com.cognizant.springlearn;


public class Country {

   private String code;

   private String name;


   public Country() {}


   public String getCode() {

      return code;
```

```
    }

    public void setCode(String code) {
        this.code = code;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}
package com.cognizant.springlearn.service;

import com.cognizant.springlearn.Country;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class CountryService {

    public Country getCountry(String code) {
        ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");
        List<Country> countryList = context.getBean("countryList", List.class);
```

```
        return countryList.stream()
                .filter(c -> c.getCode().equalsIgnoreCase(code))
                .findFirst()
                .orElse(null); // Optionally, you could throw an exception here
    }
}
package com.cognizant.springlearn.controller;

import com.cognizant.springlearn.Country;
import com.cognizant.springlearn.service.CountryService;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

@RestController
public class CountryController {

    private static final Logger LOGGER = LoggerFactory.getLogger(CountryController.class);

    @Autowired
    private CountryService countryService;

    @GetMapping("/countries/{code}")
    public Country getCountry(@PathVariable String code) {
        LOGGER.info("START - getCountry(): {}", code);
        Country country = countryService.getCountry(code);
        LOGGER.info("END - getCountry(): {}", country);
        return country;
    }}
```

Create authentication service that returns JWT

```
<!-- Spring Boot Starter Security -->
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-security</artifactId>
</dependency>

<!-- JWT dependency -->
<dependency>
   <groupId>io.jsonwebtoken</groupId>
   <artifactId>jjwt</artifactId>
   <version>0.9.1</version>
</dependency>
package com.cognizant.springlearn.util;


import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.stereotype.Component;


import java.util.Date;


@Component

public class JwtUtil {

   private final String SECRET_KEY = "secret";


   public String generateToken(String username) {

      return Jwts.builder()

            .setSubject(username)

            .setIssuedAt(new Date(System.currentTimeMillis()))

            .setExpiration(new Date(System.currentTimeMillis() + 10 * 60 * 1000)) // 10 mins

            .signWith(SignatureAlgorithm.HS256, SECRET_KEY)

            .compact();

   }

}

package com.cognizant.springlearn.controller;
```

```
import com.cognizant.springlearn.util.JwtUtil;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpHeaders;
import org.springframework.web.bind.annotation.*;

import java.nio.charset.StandardCharsets;
import java.util.Base64;
import java.util.HashMap;
import java.util.Map;

@RestController
public class AuthController {

    @Autowired
    private JwtUtil jwtUtil;

    @GetMapping("/authenticate")
    public Map<String, String>
authenticate(@RequestHeader(HttpHeaders.AUTHORIZATION) String authHeader) {
        // Remove "Basic " and decode base64
        String base64Credentials = authHeader.substring("Basic ".length());
        byte[] decodedBytes = Base64.getDecoder().decode(base64Credentials);
        String decodedCredentials = new String(decodedBytes, StandardCharsets.UTF_8);

        String[] values = decodedCredentials.split(":", 2);
        String username = values[0];
        String password = values[1];

        // Hardcoded check (in production, fetch from DB or UserDetailsService)
        if ("user".equals(username) && "pwd".equals(password)) {
```

```
            String token = jwtUtil.generateToken(username);
            Map<String, String> response = new HashMap<>();
            response.put("token", token);
            return response;
        } else {
            throw new RuntimeException("Invalid Credentials");
        }
    }
}
package com.cognizant.springlearn.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.web.SecurityFilterChain;

@Configuration
public class SecurityConfig {

    @Bean
    public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
        http.csrf().disable()
            .authorizeHttpRequests()
                .requestMatchers("/authenticate").permitAll()
                .anyRequest().authenticated()
            .and()
            .httpBasic(); // Enable Basic Auth
        return http.build();
    }
}
```