Spring REST Hands-on Solutions

Project Structure

```
spring-learn/
├── src/
│   ├── main/
│   │   ├── java/
│   │   │   └── com/
│   │   │       └── cognizant/
│   │   │           └── springlearn/
│   │   │               ├── SpringLearnApplication.java
│   │   │               └── Country.java
│   │   └── resources/
│   │       ├── application.properties
│   │       └── country.xml
│   └── test/
│       └── java/
└── pom.xml
```

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

pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
  <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 https://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>
   <parent> <groupId>org.springframework.boot</groupId>
     <artifactId>spring-boot-starter-parent</artifactId>
     <version>2.7.0</version>
     <relativePath/>
   </parent>
   <groupId>com.cognizant</groupId>
   <artifactId>spring-learn</artifactId>
   <version>0.0.1-SNAPSHOT</version>
   <name>spring-learn</name>
   <description>Demo project for Spring Boot</description>
   <properties>
     <java.version>11</java.version>
   </properties>
     <dependencies>
       <dependency>
           <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
     </dependency>
     <dependency>
       <groupId>org.springframework.boot</groupId>
       <artifactId>spring-boot-devtools</artifactId>
       <scope>runtime</scope> <optional>true</optional>
     </dependency>
     <dependency>
       <groupId>org.springframework.boot</groupId>
```

```
      <artifactId>spring-boot-starter-test</artifactId>
      <scope>test</scope>
    </dependency>
  </dependencies>
  <build>
    <plugins>
     <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
     </plugin>
    </plugins>
  </build>
 </project>
```

SpringLearnApplication.java

```
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:

```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.7.0)

 <terminated> MainApp [Java Application] C:\Users\Rachana\Downloads\eclipse-java-2020-06-R-win32-
 x86_64\eclipse\spring-learn
 Application started successfully!
```

Hands-on 2: Spring Core – Load Country from Spring Configuration XML

Project Configuration:

- Project Name: spring-learn
- Group ID: com.cognizant

- Artifact ID: spring-learn
- Tools Used: Eclipse IDE, Maven

Country.java

```
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("Inside Country Constructor."); }

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

    public void setCode(String code) {
        LOGGER.debug("Inside setCode method with value: {}", code);
        this.code = code;
    }

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

    public void setName(String name) {
        LOGGER.debug("Inside setName method with value: {}", name);
        this.name = name;
    }

    @Override
    public String toString() {
        return "Country [code=" + code + ", name=" + name + "]"; }
}
```

country.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
        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>
```

SpringLearnApplication.java

```
package com.cognizant.springlearn;

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

application.properties

```
logging.level.org.springframework=info
logging.level.com.cognizant.springlearn=debug
logging.pattern.console=%d{yyMMdd}|%d{HH:mm:ss.SSS}|%-20.20thread|%5p|%-25.25logger{25}|%25M|%m%n
```

OUTPUT:

```
<terminated> MainApp [Java Application] C:\Users\Rachana\Downloads\eclipse-java-2020-06-R-win32-x86_64\eclipse\spring-learn
Bean loaded successfully!
Country loaded from SpringConfiguration...
```

Spring REST using Spring Boot 3

Project Structure

```
spring-learn/
├── src/
│   ├── main/
│   │   ├── java/
│   │   │   └── com/
│   │   │       └── cognizant/
│   │   │           └── springlearn/
│   │   │               ├── SpringLearnApplication.java
│   │   │               ├── controller/
│   │   │               │   ├── HelloController.java
│   │   │               │   └── CountryController.java
│   │   │               ├── service/
│   │   │               │   └── CountryService.java
│   │   │               └── model/
│   │   │                   └── Country.java
│   │   └── resources/
│   │       ├── application.properties
│   │       └── country.xml
│   └── test/
│       └── java/
│           └── com/
│               └── cognizant/
│                   └── springlearn/
│                       └── SpringLearnApplicationTests.java
├── pom.xml
└── README.md
```

Hands-on 1: Hello World RESTful Web Service

HelloController.java

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

@RestController
public class HelloController {

    public HelloController() { System.out.println("HelloController
        constructor called");
    }

     @GetMapping("/hello")
    public String sayHello() {
        System.out.println("sayHello() method - START");
```

```
        String response = "Hello World!!";
        System.out.println("sayHello() method - END");
        return response;
    }
}
```

Output:

```
HelloController constructor called
sayHello() method - START
sayHello() method - END
```

Hands-on 3: REST - Get country based on country code

CountryNotFoundException.java (Exception Class)

```
package com.cognizant.springlearn.service.exception;

import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.ResponseStatus;

@ResponseStatus(value = HttpStatus.NOT_FOUND, reason = "Country not found") public
class CountryNotFoundException extends Exception {

    public CountryNotFoundException(String message) {
        super(message);
    }
}
```

Updated CountryService.java

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

package com.cognizant.springlearn.service;

import com.cognizant.springlearn.model.Country;
import com.cognizant.springlearn.service.exception.CountryNotFoundException;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationContext;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class CountryService {
```

```
    @Autowired
    private ApplicationContext context;

    public List<Country> getAllCountries() {
        System.out.println("CountryService.getAllCountries() - START");
        @SuppressWarnings("unchecked")
        List<Country> countries = (List<Country>) context.getBean("countryList");
        System.out.println("Loaded " + countries.size() + " countries from XML");
        System.out.println("CountryService.getAllCountries() - END");
        return countries; }

    public Country getCountry(String code) throws CountryNotFoundException {
        System.out.println("CountryService.getCountry() - START");
        System.out.println("Searching for country with code: " + code);

        List<Country> countries = getAllCountries();

        // Using lambda expression for case-insensitive search
        Country country = countries.stream()
                .filter(c -> c.getCode().equalsIgnoreCase(code))
                .findFirst()
                .orElse(null);

        if (country != null) {
            System.out.println("Country found: " + country); }
        else {
            System.out.println("Country not found for code: " + code);
            throw new CountryNotFoundException("Country not found for code: " + code); }

        System.out.println("CountryService.getCountry() - END");
        return country;
    }
}
```

Updated CountryController.java

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

import com.cognizant.springlearn.model.Country;
import com.cognizant.springlearn.service.CountryService;
import com.cognizant.springlearn.service.exception.CountryNotFoundException; import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationContext;  import
org.springframework.web.bind.annotation.GetMapping; import
org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping; import
org.springframework.web.bind.annotation.RestController;

import java.util.List;

@RestController
```

```
public class CountryController {

    @Autowired
    private ApplicationContext context;

    @Autowired
    private CountryService countryService;

    public CountryController() { System.out.println("CountryController
        constructor called");
    }

    @RequestMapping("/country") public
    Country getCountryIndia() {
        System.out.println("getCountryIndia() method - START");
        Country india = (Country) context.getBean("india");
        System.out.println("Retrieved India country: " + india);
        System.out.println("getCountryIndia() method - END"); return
        india;
    }

    @GetMapping("/countries")
    public List<Country> getAllCountries() {
        System.out.println("getAllCountries() method - START");
        List<Country> countries = countryService.getAllCountries();
        System.out.println("Retrieved " + countries.size() + " countries");
        System.out.println("getAllCountries() method - END");
        return countries; }

    @GetMapping("/countries/{code}")
    public Country getCountry(@PathVariable String code) throws CountryNotFoundException {
        System.out.println("getCountry() method - START");
            System.out.println("Country code received: " + code);
             Country country = countryService.getCountry(code);
           System.out.println("Retrieved country: " + country);
           System.out.println("getCountry() method - END");
           return country;
        }
    }
```

Hands On 2: REST - Country Web Service

Country.java (Model)

```
package com.cognizant.springlearn.model;

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

    public Country() {
    }

    public Country(String code, String name) {
        this.code = code;
        this.name = name; }

    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;
    }

    @Override
     public String toString() {
         return "Country{" +
                 "code='" + code + '\'' +
                 ", name='" + name + '\'' + '}';
    }
}
```

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

import java.util.List;

@RestController
public class CountryController {

    @Autowired
    private ApplicationContext context;

    @Autowired
    private CountryService countryService;

    public CountryController() {
        System.out.println("CountryController constructor called");
    }

    @RequestMapping("/country")
    public Country getCountryIndia() {
        System.out.println("getCountryIndia() method - START");
        Country india = (Country) context.getBean("india");
        System.out.println("Retrieved India country: " + india);
        System.out.println("getCountryIndia() method - END");
        return india;
    }

    @GetMapping("/countries")
    public List<Country> getAllCountries() {
        System.out.println("getAllCountries() method - START");
        List<Country> countries = countryService.getAllCountries();
        System.out.println("Retrieved " + countries.size() + " countries");
        System.out.println("getAllCountries() method - END");
        return countries; }

    @GetMapping("/countries/{code}")
    public Country getCountry(@PathVariable String code) {
        System.out.println("getCountry() method - START");
        System.out.println("Country code received: " + code);
        Country country = countryService.getCountry(code);
        System.out.println("Retrieved country: " + country);
```

```
        System.out.println("getCountry() method - END");
        return country;

    }
```

CountryService.java

```
package com.cognizant.springlearn.service;

import com.cognizant.springlearn.model.Country;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationContext;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class CountryService {

    @Autowired
    private ApplicationContext context;

    public List<Country> getAllCountries() {
        System.out.println("CountryService.getAllCountries() - START");
        @SuppressWarnings("unchecked")
        List<Country> countries = (List<Country>) context.getBean("countryList");
        System.out.println("Loaded " + countries.size() + " countries from XML");
        System.out.println("CountryService.getAllCountries() - END");
        return
    countries; }

    public Country getCountry(String code) {
        System.out.println("CountryService.getCountry() - START");
        System.out.println("Searching for country with code: " + code);

        List<Country> countries = getAllCountries();

        // Using lambda expression for case-insensitive search
        Country country = countries.stream()
                .filter(c -> c.getCode().equalsIgnoreCase(code))
                .findFirst()
                .orElse(null);

        if (country != null) {
            System.out.println("Country found: " + country);
        } else {
            System.out.println("Country not found for code: " + code); }

        System.out.println("CountryService.getCountry() - END");
        return country;

    }
```

country.xml (Configuration)

```
<?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="india" class="com.cognizant.springlearn.model.Country">
        <property name="code" value="IN"/>
        <property name="name" value="India"/>
    </bean>

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

</beans>
```

SpringLearnApplication.java

```
package com.cognizant.springlearn;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.ImportResource;

@SpringBootApplication
@ImportResource("classpath:country.xml")
public class SpringLearnApplication {

    public static void main(String[] args) {
```

```
        SpringApplication.run(SpringLearnApplication.class, args);
    }
```

application.properties

```
properties

server.port=8083
logging.level.com.cognizant.springlearn=DEBUG
```

Outputs

For GET /country request:

```
CountryController constructor called
getCountryIndia() method - START
Retrieved India country: Country{code='IN', name='India'}
getCountryIndia() method - END
```

For GET /countries request:

```
getAllCountries() method - START
CountryService.getAllCountries() -
START Loaded 4 countries from XML
CountryService.getAllCountries() - END
Retrieved 4 countries
getAllCountries() method - END
```

For GET /countries/{code} request (e.g., /countries/in):

```
getCountry() method - START
Country code received: in
CountryService.getCountry() -
START Searching for country with
code: in
CountryService.getAllCountries() -
START Loaded 4 countries from XML
CountryService.getAllCountries() - END
Country found: Country{code='IN', name='India'}
CountryService.getCountry() - END
Retrieved country: Country{code='IN', name='India'}
getCountry() method - END
```

For GET /countries/{code} request with invalid code (e.g., /countries/az):

```
getCountry() method - START
Country code received: az
CountryService.getCountry() -
START Searching for country with
```

```
code: az
CountryService.getAllCountries() -
START Loaded 4 countries from XML
CountryService.getAllCountries() - END
Country not found for code: az
```

Sample API Responses

1. GET /hello

Request: http://localhost:8083/hello **Response:**

```
Hello World!!
```

2. GET /country

Request: http://localhost:8083/country **Response:**


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


3. GET /countries

Request: http://localhost:8083/countries **Response:**


```
[

  "code": "IN",
  "name":
  "India"


  "code": "US",
  "name": "United
 States" },
 {
  "code": "JP",
  "name":
  "Japan"


  "code": "DE",
  "name":
  "Germany"

]
```


4. GET /countries/{code}

Request: http://localhost:8083/countries/in **Response:**

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


Request: http://localhost:8083/countries/us **Response:**


```
{
  "code": "US",
  "name": "United
States" }
```


5. GET /countries/{code} - Invalid Code (Error Response)

Request: http://localhost:8083/countries/az **Response:**


```
{
 "timestamp": "2025-07-12T10:27:54.521+0000",
 "status": 404,
 "error": "Not Found",   "message":
 "Country not found", "path":
 "/countries/az"
}
```

JWT Authentication Service Implementation

Project Structure

```
spring-learn/
├── src/
│   └── main/
│       └── java/
│           └── com/
│               └── cognizant/
│                   └── springlearn/
├── controller/
│                   │   └── AuthenticationController.java
│                   └── security/
│                       └── SecurityConfig.java
├── pom.xml
└── target/
```

Maven Dependencies

pom.xml

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

    <!-- JWT Library -->
    <dependency>
        <groupId>io.jsonwebtoken</groupId>
        <artifactId>jjwt</artifactId>
        <version>0.9.0</version>
    </dependency>

    <!-- Other existing dependencies -->
</dependencies>
```

Security Configuration

SecurityConfig.java

```
package com.cognizant.springlearn.security;

import org.slf4j.Logger;   import
org.slf4j.LoggerFactory;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
```

```
import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder; import
org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter; import
org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;

@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {

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

    @Override
    protected void configure(AuthenticationManagerBuilder auth) throws Exception {
        auth.inMemoryAuthentication()
            .withUser("admin").password(passwordEncoder().encode("pwd")).roles("ADMIN")
            .and() .withUser("user").password(passwordEncoder().encode("pwd")).roles("USER");
    }

    @Bean
    public PasswordEncoder passwordEncoder() {
        LOGGER.info("Start");
        return new BCryptPasswordEncoder(); }

    @Override
    protected void configure(HttpSecurity httpSecurity) throws Exception {
        httpSecurity.csrf().disable().httpBasic().and()
            .authorizeRequests() .antMatchers("/countries").hasRole("USER")
            .antMatchers("/authenticate").hasAnyRole("USER", "ADMIN");
    }
}
```

Authentication Controller

AuthenticationController.java

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

import java.util.Base64;
import java.util.Date; import
java.util.HashMap; import
java.util.Map;

import org.slf4j.Logger;   import
org.slf4j.LoggerFactory;
import org.springframework.web.bind.annotation.GetMapping; import
org.springframework.web.bind.annotation.RequestHeader; import
org.springframework.web.bind.annotation.RestController;

import io.jsonwebtoken.JwtBuilder;
import io.jsonwebtoken.Jwts;
import io.jsonwebtoken.SignatureAlgorithm;
```

```
@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("Authorization Header: {}", authHeader);

        Map<String, String> map = new HashMap<>();

        // Get user from authorization header
        String user = getUser(authHeader);
        LOGGER.debug("User: {}", user);

        // Generate JWT token
        String token = generateJwt(user);
        LOGGER.debug("Generated Token: {}", token);

        map.put("token", token);

        LOGGER.info("End");
        return map;
    }

    private String getUser(String authHeader) {
        LOGGER.info("Start");

        // Extract Base64 encoded credentials after "Basic " String
        encodedCredentials = authHeader.substring(6);
        LOGGER.debug("Encoded Credentials: {}", encodedCredentials);

            // Decode Base64
            byte[] decodedBytes = Base64.getDecoder().decode(encodedCredentials); String
            credentials = new String(decodedBytes);
            LOGGER.debug("Decoded Credentials: {}", credentials);

            // Extract username (text before colon) String
            user = credentials.split(":")[0];
            LOGGER.debug("Extracted User: {}", user);

            LOGGER.info("End")
            ; return user;
        }

        private String generateJwt(String user) {
            LOGGER.info("Start");

            JwtBuilder builder = Jwts.builder();
            builder.setSubject(user);
```

```
        // Set the token issue time as current time
        builder.setIssuedAt(new Date());

        // Set the token expiry as 20 minutes from now builder.setExpiration(new
        Date((new Date()).getTime() + 1200000));

        builder.signWith(SignatureAlgorithm.HS256, "secretkey");

        String token = builder.compact();
        LOGGER.debug("Generated JWT Token: {}", token);

        LOGGER.info("End")
        ; return token;
    }
}
```

Testing Commands

Get JWT Token

bash

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

Test with Valid Credentials

bash

```
curl -s -u admin:pwd http://localhost:8090/authenticate
```

Test with Invalid Credentials

```
curl -s -u user:wrongpwd http://localhost:8090/authenticate
```

OUTPUT

```
Authentication successful!
Token generated successfully...
{
  "token":
"eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiaWF0IjoxNTcwMzc5NDc0LCJleHAiOjE1NzAzODA2NzR9.t3LRvlCV-
hwKfoqZYlaVQqEUiBloWcWn0ft3tgv0dL0"
}
```