**Spring REST using Spring Boot 3**

**Create a Spring Web Project using Maven**

**Code:**

//SpringLearnApplication.java

package com.cognizant.spring\_learn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

    public static void main(String[] args) {

        System.out.println("SpringLearnApplication started!");

        SpringApplication.run(SpringLearnApplication.class, args);

    }

}

**Output:**

A screenshot of a computer program

AI-generated content may be incorrect.

**Spring Core – Load Country from Spring Configuration XML**

**Code:**

**#application.properties**

spring.application.name=spring-learn

logging.level.root=DEBUG

//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

       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>

//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); // 🔥 This is required

        LOGGER.debug("START");

        displayCountry();

        LOGGER.debug("END");

    }

    public static void displayCountry() {

        ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

        Country country = context.getBean("country", Country.class);

        LOGGER.debug("Country: {}", country.toString());

    }

}

//Country.java

package com.cognizant.springlearn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

    private String code;

    private String name;

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

    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: {}", code);

        this.code = code;

    }

    public String getName() {

        LOGGER.debug("Getting name");

        return name;

    }

    public void setName(String name) {

        LOGGER.debug("Setting name: {}", name);

        this.name = name;

    }

    @Override

    public String toString() {

        return "Country [code=" + code + ", name=" + name + "]";

    }

}

**Output:**

**A screenshot of a computer screen

AI-generated content may be incorrect.**

**Hello World RESTful Web Service**

**Code:**

**#application.properties**

spring.application.name=spring-learn

logging.level.root=DEBUG

//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

       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>

//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);

    }

}

//Country.java

package com.cognizant.springlearn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

    private String code;

    private String name;

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

    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: {}", code);

        this.code = code;

    }

    public String getName() {

        LOGGER.debug("Getting name");

        return name;

    package com.cognizant.springlearn.controller;

//HelloController.java

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.debug("START sayHello()");

        LOGGER.debug("END sayHello()");

        return "Hello World!!";

    }

}

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

**Hello World RESTful Web Service**

**Code:**

**#application.properties**

spring.application.name=spring-learn

logging.level.root=DEBUG

//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

       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>

//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);

    }

}

//Country.java

package com.cognizant.springlearn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

    private String code;

    private String name;

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

    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: {}", code);

        this.code = code;

    }

    public String getName() {

        LOGGER.debug("Getting name");

        return name;

    package com.cognizant.springlearn.controller;

//CountryController.java

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.debug("START getCountryIndia()");

        ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

        Country country = context.getBean("country", Country.class);

        LOGGER.debug("Country: {}", country.toString());

        LOGGER.debug("END getCountryIndia()");

        return country;

    }

}

//HelloController.java

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.debug("START sayHello()");

        LOGGER.debug("END sayHello()");

        return "Hello World!!";

    }

}

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

**REST - Get country based on country code**

**Code:**

**#application.properties**

spring.application.name=spring-learn

logging.level.root=DEBUG

//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

       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>

//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);

    }

}

//Country.java

package com.cognizant.springlearn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

    private String code;

    private String name;

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

    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: {}", code);

        this.code = code;

    }

    public String getName() {

        LOGGER.debug("Getting name");

        return name;

//CountryService.java

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> countries = (List<Country>) context.getBean("countryList");

        return countries.stream()

                .filter(country -> country.getCode().equalsIgnoreCase(code))

                .findFirst()

                .orElse(null);

    }

}

//CountryController.java

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("/country")

    public Country getCountryIndia() {

        LOGGER.debug("START getCountryIndia()");

        Country country = countryService.getCountry("IN");

        LOGGER.debug("END getCountryIndia()");

        return country;

    }

    @GetMapping("/countries/{code}")

    public Country getCountry(@PathVariable String code) {

        LOGGER.debug("START getCountry() with code = {}", code);

        Country country = countryService.getCountry(code);

        LOGGER.debug("END getCountry()");

        return country;

    }

}

//HelloController.java

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.debug("START sayHello()");

        LOGGER.debug("END sayHello()");

        return "Hello World!!";

    }

}

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

**REST - Get country based on country code**

**Code:**

**#application.properties**

spring.application.name=spring-learn

#logging.level.root=DEBUG

server.port=8083

spring.security.user.name=user

spring.security.user.password=pwd

logging.level.root=DEBUG

//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

       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>

<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>3.1.0</version>

        <relativePath/>

    </parent>

    <groupId>com.cognizant</groupId>

    <artifactId>spring-learn</artifactId>

    <version>0.0.1-SNAPSHOT</version>

    <name>spring-learn</name>

    <description>JWT Auth Example</description>

    <properties>

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

    </properties>

    <dependencies>

        <!-- Web & Security -->

        <dependency>

            <groupId>org.springframework.boot</groupId>

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

        </dependency>

        <dependency>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-starter-security</artifactId>

        </dependency>

        <dependency>

    <groupId>org.springframework.boot</groupId>

    <artifactId>spring-boot-starter-test</artifactId>

    <scope>test</scope>

    <exclusions>

        <exclusion>

            <groupId>org.junit.vintage</groupId>

            <artifactId>junit-vintage-engine</artifactId>

        </exclusion>

    </exclusions>

</dependency>

        <!-- JWT (Java 11+ compatible) -->

        <dependency>

            <groupId>io.jsonwebtoken</groupId>

            <artifactId>jjwt-api</artifactId>

            <version>0.11.5</version>

        </dependency>

        <dependency>

            <groupId>io.jsonwebtoken</groupId>

            <artifactId>jjwt-impl</artifactId>

            <version>0.11.5</version>

            <scope>runtime</scope>

        </dependency>

        <dependency>

            <groupId>io.jsonwebtoken</groupId>

            <artifactId>jjwt-jackson</artifactId>

            <version>0.11.5</version>

            <scope>runtime</scope>

        </dependency>

        <!-- Dev tools (optional) -->

        <dependency>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-devtools</artifactId>

            <scope>runtime</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);

    }

}

//JwtUtil.java

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 = "secret123"; // use env variable in real projects

    public String generateToken(String username) {

        return Jwts.builder()

                .setSubject(username)

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

                .setExpiration(new Date(System.currentTimeMillis() + 1000 \* 60 \* 60)) // 1 hour

                .signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

                .compact();

    }

}

//CountryService.java

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> countries = (List<Country>) context.getBean("countryList");

        return countries.stream()

                .filter(country -> country.getCode().equalsIgnoreCase(code))

                .findFirst()

                .orElse(null);

    }

}

//SecurityConfig.java

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(csrf -> csrf.disable())

            .authorizeHttpRequests(auth -> auth

                .requestMatchers("/authenticate").permitAll()

                .anyRequest().authenticated()

            )

            .httpBasic(); // Enable basic auth for testing with Postman

        return http.build();

    }

}

//AuthenticationController.java

package com.cognizant.springlearn.controller;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import io.jsonwebtoken.security.Keys;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import jakarta.servlet.http.HttpServletRequest;

import java.nio.charset.StandardCharsets;

import java.util.Base64;

import java.util.Date;

@RestController

public class AuthenticationController {

    private static final String SECRET\_KEY = "MySuperSecretKeyForJwtThatShouldBeLongEnough123!";

    @GetMapping("/authenticate")

    public ResponseEntity<?> authenticate(HttpServletRequest request) {

        String authHeader = request.getHeader("Authorization");

        if (authHeader == null || !authHeader.startsWith("Basic ")) {

            return ResponseEntity.status(401).body("Missing or invalid Authorization header");

        }

        // Decode credentials

        String base64Credentials = authHeader.substring("Basic ".length());

        byte[] credDecoded = Base64.getDecoder().decode(base64Credentials);

        String credentials = new String(credDecoded, StandardCharsets.UTF\_8);

        String[] values = credentials.split(":", 2);

        String username = values[0];

        String password = values[1];

        // Very basic check, replace with actual authentication logic

        if (!username.equals("user") || !password.equals("pwd")) {

            return ResponseEntity.status(401).body("Invalid credentials");

        }

        // Generate token

        String token = Jwts.builder()

                .setSubject(username)

                .setIssuedAt(new Date())

                .setExpiration(new Date(System.currentTimeMillis() + 60 \* 60 \* 1000)) // 1 hour expiry

                .signWith(Keys.hmacShaKeyFor(SECRET\_KEY.getBytes(StandardCharsets.UTF\_8)), SignatureAlgorithm.HS256)

                .compact();

        return ResponseEntity.ok().body("{\"token\":\"" + token + "\"}");

    }

}

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.