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

Code:

|  |
| --- |
| package com.cognizant.spring\_learn;  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) {  displayCountry();  }  public static void displayCountry() {  ApplicationContext context = new ClassPathXmlApplicationContext("Country.xml");  Country country = context.getBean("country", Country.class);  System.out.println("Country: " + country); // Add this line for visible output  }  } |

Output:

A screenshot of a computer

AI-generated content may be incorrect.

**Hello World RESTful Web Service**

Code:

|  |
| --- |
| package com.cognizant.spring\_learn.controller;  import org.slf4j.Logger;  import org.slf4j.LoggerFactory;  import org.springframework.web.bind.annotation.\*;  *@RestController*  public class HelloController {  private static final Logger ***LOGGER*** = LoggerFactory.*getLogger*(HelloController.class);  *@GetMapping*("/hello")  public String sayHello() {  ***LOGGER***.info("START - sayHello()");  String response = "Hello World!!";  ***LOGGER***.info("END - sayHello()");  return response;  }  } |

Output:

A screenshot of a computer

AI-generated content may be incorrect.

**REST - Country Web Service**

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*  *https://www.springframework.org/schema/beans/spring-beans.xsd"*>  <**bean** id=*"country"* class=*"com.cognizant.spring\_learn.model.country"*>  <**property** name=*"code"* value=*"in"*/>  <**property** name=*"name"* value=*"india"*/>  </**bean**>  </**beans**>  @RequestMapping("/country")  public Country getCountryIndia() {  LOGGER.info("START - getCountryIndia()");  ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  Country country = (Country) context.getBean("country");  LOGGER.info("END - getCountryIndia()");  return country;  }  package com.cognizant.spring\_learn.model;  public class Country {  private String code;  private String name;  // Getters and Setters  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;  }  } |

Output:

A screenshot of a computer

AI-generated content may be incorrect.

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

Code:

|  |
| --- |
| *@Autowired*  private CountryService countryService;  *@GetMapping*("/countries/{code}")  public Country getCountry(*@PathVariable* String code) {  ***LOGGER***.info("START - getCountry() with code: {}", code);  Country country = countryService.getCountry(code);  ***LOGGER***.info("END - getCountry()");  return country;  }  *@Service*  public class CountryService {  public Country getCountry(String code) {  ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  List<Country> countryList = (List<Country>) context.getBean("countryList");  return countryList.stream()  .filter(country -> country.getCode().equalsIgnoreCase(code))  .findFirst()  .orElse(null);  }  } |

Output:

A screenshot of a computer

AI-generated content may be incorrect.

**Create authentication service that returns JWT**

Code:

|  |
| --- |
| package com.cognizent.Jwt;  import java.util.Base64;  import org.springframework.http.ResponseEntity;  import org.springframework.web.bind.annotation.GetMapping;  import org.springframework.web.bind.annotation.RequestHeader;  import org.springframework.web.bind.annotation.RestController;  @RestController  public class AuthController {  @GetMapping("/authenticate")  public ResponseEntity<?> authenticate(@RequestHeader("Authorization") String authHeader) {  if (authHeader != null && authHeader.startsWith("Basic ")) {  String base64Credentials = authHeader.substring("Basic ".length());  byte[] decodedBytes = Base64.getDecoder().decode(base64Credentials);  String decodedCredentials = new String(decodedBytes);  String[] values = decodedCredentials.split(":", 2);  String username = values[0];  String password = values[1];  if ("user".equals(username) && "pwd".equals(password)) {  String token = JwtUtil.generateToken(username);  return ResponseEntity.ok().body("{\"token\":\"" + token + "\"}");  }  }  return ResponseEntity.status(401).body("Invalid credentials");  }  } |

Output:  
A screenshot of a computer

AI-generated content may be incorrect.