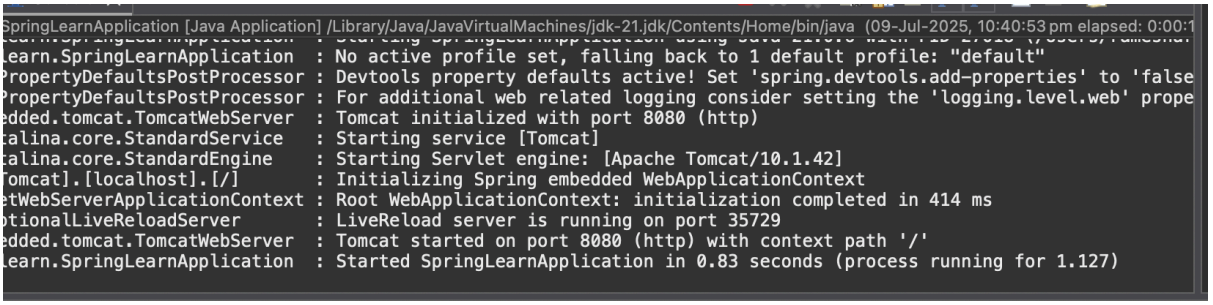
WEEK-4

Spring-rest handson

Exercise 1: Create a Spring Web Project using Maven

- Visit https://start.spring.io/
- Change Group as “com.cognizant”
- Change Artifact Id as “spring-learn”
- Select Spring Boot DevTools and Spring Web
- Download the project zip file
- Open Eclipse and open the existing maven project
- Click “Run as Java Application” on the SpringLearnApplication

Output:



Exercise 2: Spring Core – Load Country from Spring Configuration XML

SpringLearnApplication.java

```
package com.cognizant.spring_learn;
   import org.springframework.boot.SpringApplication;
   import org.springframework.boot.autoconfigure.SpringBootApplication;
   import org.slf4j.Logger;
   import org.slf4j.LoggerFactory;
   import org.springframework.context.ApplicationContext;
   import org.springframework.context.support.ClassPathXmlApplicationContext;
   import com.cognizant.spring_learn.model.Country;
   @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("SpringLearnApplication main() method finished execution.");
         displayCountry();
      }
      public static void displayCountry() {
         LOGGER.info("Starting displayCountry() method...");
         ApplicationContext context = new ClassPathXmlApplicationContext("country-config.xml");
         LOGGER.info("Loaded country-config.xml context.");
```

```
        Country country = context.getBean("in", Country.class);
        LOGGER.info("Retrieved 'in' country bean from context: {}", country);
        System.out.println("Country Code: " + country.getCode());
        System.out.println("Country Name: " + country.getName());
        ((ClassPathXmlApplicationContext) context).close();
        LOGGER.info("Closed country-config.xml context.");
        LOGGER.info("displayCountry() method finished.");
    }
}
```

- Create country.xml in src/main/resources

```
<?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="in" class="com.cognizant.spring_learn.model.Country">
      <property name="code" value="IN"/>
      <property name="name" value="India"/>
    </bean>
    <bean id="us" class="com.cognizant.spring_learn.model.Country">
      <property name="code" value="US"/>
      <property name="name" value="United States"/>
    </bean>

  </beans>
```

Output:

```
SpringLearnApplication [Java Application] C:\Users\HP\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.6.v20250130-0529\jre\bin\javaw.exe  (Jul 11, 2025, 8:24:52 PM elap
1T20:24:55.906+05:30  INFO 6576 --- [spring-learn] [  restartedMain] c.c.spring_learn.SpringLearnApplication  : Started SpringLearnApplicati
1T20:24:55.912+05:30  INFO 6576 --- [spring-learn] [  restartedMain] c.c.spring_learn.SpringLearnApplication  : SpringLearnApplication main(
1T20:24:55.913+05:30  INFO 6576 --- [spring-learn] [  restartedMain] c.c.spring_learn.SpringLearnApplication  : Starting displayCountry() me
Default Constructor called!
setCode called with: IN
setName called with: India
Default Constructor called!
setCode called with: US
setName called with: United States
1T20:24:56.031+05:30  INFO 6576 --- [spring-learn] [  restartedMain] c.c.spring_learn.SpringLearnApplication  : Loaded country-config.xml co
1T20:24:56.032+05:30  INFO 6576 --- [spring-learn] [  restartedMain] c.c.spring_learn.SpringLearnApplication  : Retrieved 'in' country bean
```

Exercise 3: Hello World RESTful Web Service

- We create a new package in src/main/java/com/cognizant/spring_learn/controller.
- Create a new java file HelloController.java

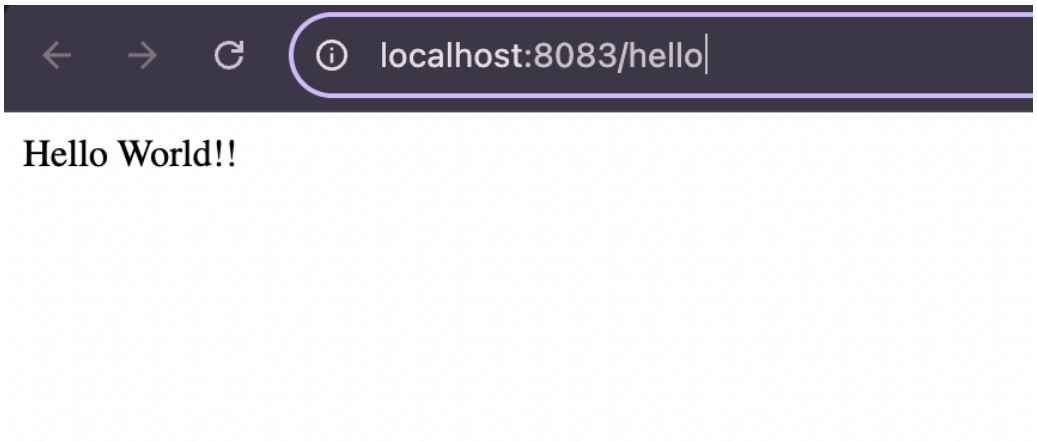
```
package com.cognizant.spring_learn.controller;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
@RestController
public class HelloController {
  private static final Logger LOGGER = LoggerFactory.getLogger(HelloController.class);
  @GetMapping("/hello")
  public String sayHello() {
```

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

- Update the port number in application.properties
  server.port=8083

- Output runs in the http://localhost:8083/hello

Output:



Exercise 4: REST - Country Web Service

- 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.spring_learn.model.Country">
       <property name="code" value="IN"/>
       <property name="name" value="India"/>
   </bean>
</beans>
```

- Create this class in com.cognizant.spring_learn.model

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

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

    public Country() {
```

```
        System.out.println("Inside Country Constructor.");
    }

    public String getCode() {
        System.out.println("Getting Code");
        return code;
    }

    public void setCode(String code) {
        System.out.println("Setting Code");
        this.code = code;
    }

    public String getName() {
        System.out.println("Getting Name");
        return name;
    }

    public void setName(String name) {
        System.out.println("Setting Name");
        this.name = name;
    }

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

- in com.cognizant.springlearn.controller

```
package com.cognizant.spring_learn.controller;
import com.cognizant.spring_learn.model.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;
   }
}
```

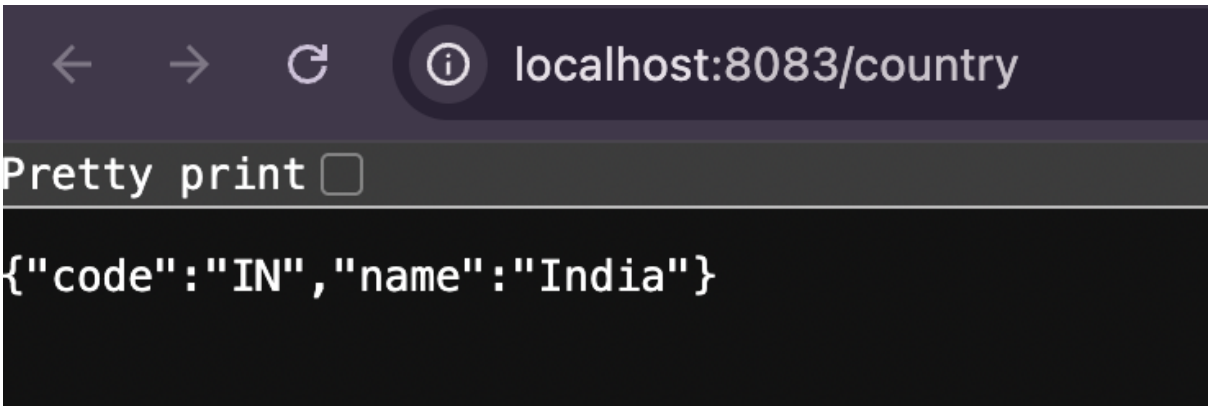
SpringLearnApplication.java

```
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:



Exercise 5 :REST - Get country based on country code

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.spring_learn.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.spring_learn.model.Country">
                    <property name="code" value="IN"/>
                    <property name="name" value="India"/>
                </bean>
                <bean class="com.cognizant.spring_learn.model.Country">
                    <property name="code" value="US"/>
                    <property name="name" value="United States"/>
```

```
            </bean>
            <bean class="com.cognizant.spring_learn.model.Country">
               <property name="code" value="UK"/>
               <property name="name" value="United Kingdom"/>
            </bean>
         </list>
      </constructor-arg>
   </bean>
</beans>
```

- In com.cognizant.spring_learn.service, create CountryService class

```
package com.cognizant.spring_learn.service;
import com.cognizant.spring_learn.model.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 = context.getBean("countryList", List.class);
      return countries.stream().filter(c->c.getCode().equalsIgnoreCase(code)).findFirst().orElse(null);
   }
}
```

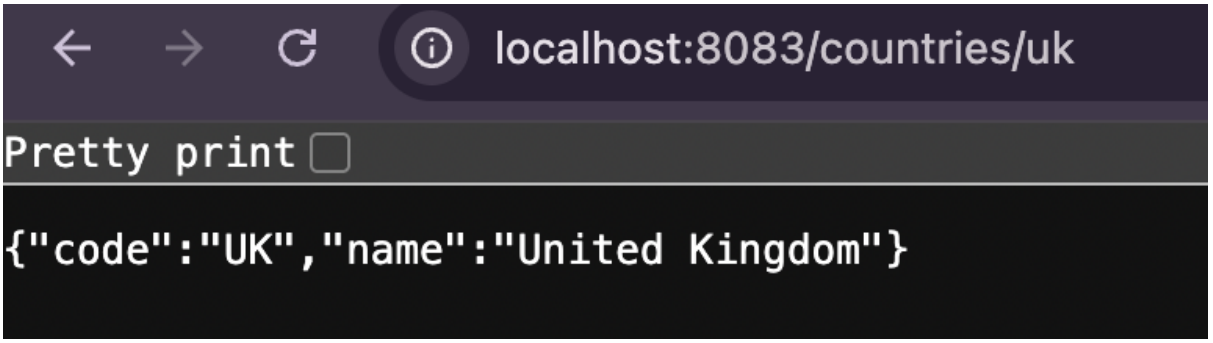
CountryController.java

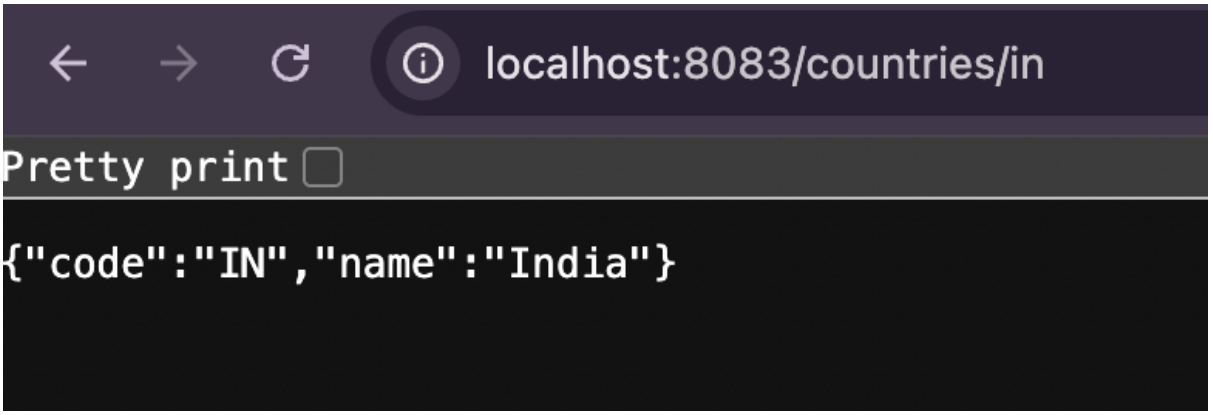
```
package com.cognizant.spring_learn.controller;
import com.cognizant.spring_learn.model.Country;
import com.cognizant.spring_learn.service.CountryService;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
@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()");
      Country country = countryService.getCountry(code);
      LOGGER.info("End getCountry()");
      return country;
   }
}
```

- In com.cognizant.spring_learn.model, create a class Country

```
package com.cognizant.spring_learn.model;
public class Country {
  private String code;
  private String name;
  public Country() {
     System.out.println("Inside Country Constructor.");
  }
  public String getCode() {
     System.out.println("Getting code");
     return code;
  }
  public void setCode(String code) {
     System.out.println("Setting code");
     this.code = code;
  }
  public String getName() {
     System.out.println("Getting name");
     return name;
  }
  public void setName(String name) {
     System.out.println("Setting name");
     this.name = name;
  }
  @Override
  public String toString() {
     return "Country [code=" + code + ", name=" + name +"]";
  }
}
```

Output:





Exercise 6 : Create authentication service that returns JWT

Pom.xml:

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-security</artifactId>
</dependency>
<dependency>
  <groupId>io.jsonwebtoken</groupId>
  <artifactId>jjwt</artifactId>
  <version>0.9.1</version>
</dependency>
```

Jwtutil.java:

```
package com.cognizant.spring_learn.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 = "mySecretKey123";
  public String generateToken(String username) {
     return Jwts.builder()
           .setSubject(username)
           .setIssuedAt(new Date(System.currentTimeMillis()))
           .setExpiration(new Date(System.currentTimeMillis() + 1000 * 60 * 10))
           .signWith(SignatureAlgorithm.HS256, SECRET_KEY)
           .compact();
  }
}
```

AuthenticationController.java:

```
   package com.cognizant.spring_learn.controller;
   import com.cognizant.spring_learn.util.JwtUtil;
   import org.springframework.beans.factory.annotation.Autowired;
   import org.springframework.http.ResponseEntity;
   import java.util.Base64;
   import org.springframework.web.bind.annotation.*;
   import jakarta.servlet.http.HttpServletRequest;
   @RestController
   public class AuthenticationController {
     @Autowired
     private JwtUtil jwtUtil;
     @RequestMapping(value = "/authenticate", method = RequestMethod.GET)
     public ResponseEntity<?> authenticate(HttpServletRequest request) {
        String authHeader = request.getHeader("Authorization");
        if (authHeader != null && authHeader.startsWith("Basic ")) {
           String base64Credentials = authHeader.substring("Basic ".length());
           byte[] credDecoded = Base64.getDecoder().decode(base64Credentials);
```

```
            String credentials = new String(credDecoded);
            String[] values = credentials.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 + "\"}");
            } else {
                return ResponseEntity.status(401).body("Invalid Credentials");
            }
        } else {
            return ResponseEntity.badRequest().body("Missing Authorization Header");
        }
    }
}
```

SecurityConfig.java:

```
package com.cognizant.spring_learn.config;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.web.SecurityFilterChain;
import org.springframework.context.annotation.Bean;
@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()
            );
        return http.build();
    }
}
```

Output:

```
C:\Users\HP\Desktop\DSEnurture\Week-4\springlearn>curl -s -u user:pwd http://localhost:8080/authenticate
{"token":"eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiaWF0IjoxNzUyMzIyNTc3LCJleHAiOjE3NTIzMjM3Nzd9.DFc8dgPkxxeRc_89bYQaDJ0Ht2SGhkEFJmVD7VWB1XA"
}
```