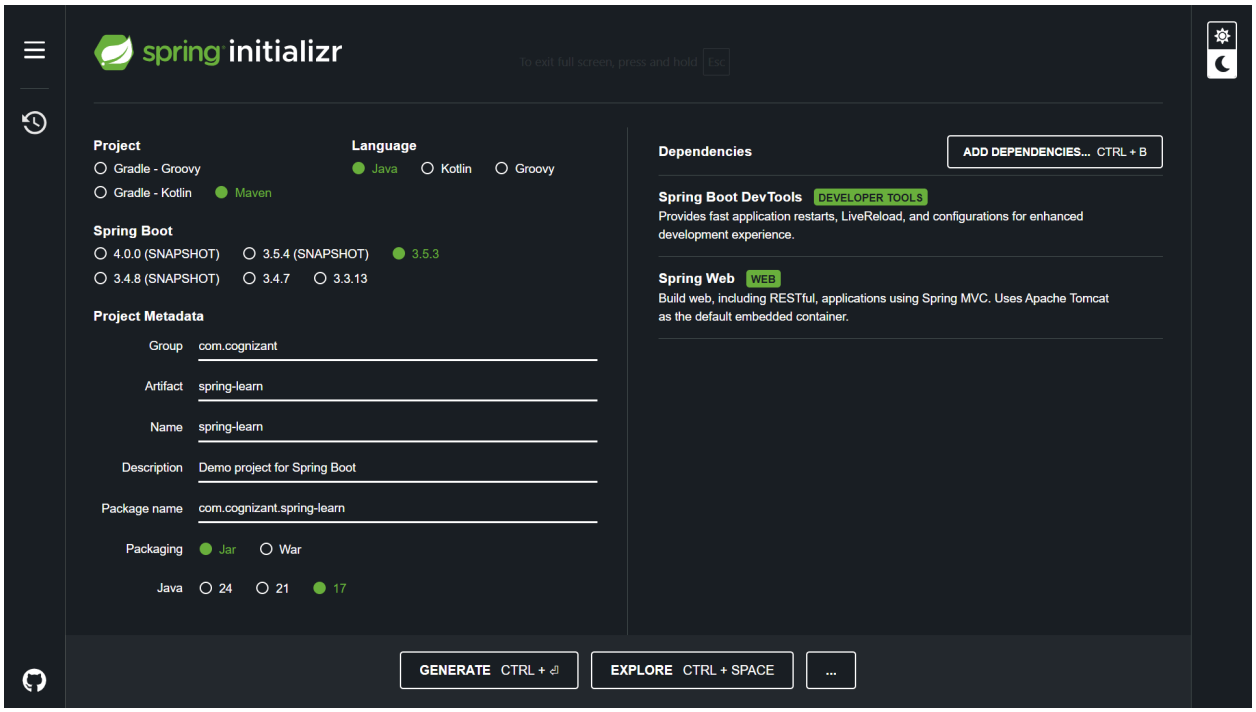
WEEK-4 HANDS ON

Exercise 1 - Create a Spring Web Project using Maven

- Initialized a spring project called “spring-learn” with “Spring Boot Dev tools” and “Spring Web” dependencies using spring initializer



pom.xml

```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>3.5.3</version>
                <relativePath/> <!-- lookup parent from repository -->
        </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>
<url/>
<licenses>
	<license/>
</licenses>
<developers>
	<developer/>
</developers>
<scm>
	<connection/>
	<developerConnection/>
	<tag/>
	<url/>
</scm>
<properties>
	<java.version>17</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>
```

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

- Created a file named country.xml in src/main/resources folder.
- Created a class called “Country.java” to represent country object with corresponding getter and setter methods.
- Included Logger in the SpringLearnApplication.java
- Added a function called “displayCountry” in the same file to fetch from xml file and log it in the logger.

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

Country.java

```
package com.cognizant.spring_learn;

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("getCode() called. Returning: {}", code);
        return code;
    }

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

    public String getName() {
        logger.debug("getName() called. Returning: {}", name);
        return name;
    }

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

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

}
```

SpringLearnApplication.java

```
package com.cognizant.spring_learn;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.cognizant.spring_learn.Country;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

@SpringBootApplication
```

```java
public class SpringLearnApplication {

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

	public static void displayCountry() {
		ClassPathXmlApplicationContext context = new
ClassPathXmlApplicationContext("country.xml");
		Country country = context.getBean("country", Country.class);
		logger.info("Country : {}", country.toString());
	}

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

}
```

Output:

The logs of the application are as follows:

```
Problems  Servers  Terminal  Data Source Explorer  Properties  Console ×  Eclipse IDE for Enterprise Java and Web Developers 2025-09

SpringLearnApplication [Java Application] C:\Users\gopih\Downloads\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.10.v20240120-1143\jr
LearnApplication  : Starting SpringLearnApplication using Java 17.0.10 with PID 21068 (C:\Users\gopi
LearnApplication  : No active profile set, falling back to 1 default profile: "default"
ultsPostProcessor : Devtools property defaults active! Set 'spring.devtools.add-properties' to 'fals
ultsPostProcessor : For additional web related logging consider setting the 'logging.level.web' prop
.TomcatWebServer  : Tomcat initialized with port 8080 (http)
StandardService   : Starting service [Tomcat]
StandardEngine    : Starting Servlet engine: [Apache Tomcat/10.1.42]
alhost].[/]       : Initializing Spring embedded WebApplicationContext
pplicationContext : Root WebApplicationContext: initialization completed in 1872 ms
eloadServer       : LiveReload server is running on port 35729
.TomcatWebServer  : Tomcat started on port 8080 (http) with context path '/'
LearnApplication  : Started SpringLearnApplication in 3.949 seconds (process running for 4.816)
LearnApplication  : Country : Country { code = 'IN', name = 'India' }
```

Exercise 3 - Hello World RESTful Web Service

- Created a called “HelloController.java” and mapped to /hello url in the folder src/main/java/com/cognizant/spring_learn/controller/
- Added the server port 8083 in application.properties file.
- Started the application and opened http://localhost:8083/hello in the browser.

HelloController.java

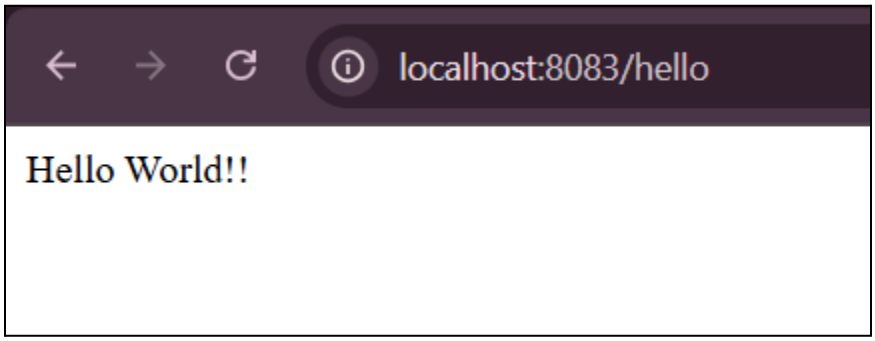
```java
package com.cognizant.spring_learn.controller;
```

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

  }
}
```

application.properties

```
spring.application.name=spring-learn
server.port=8083
```

Output

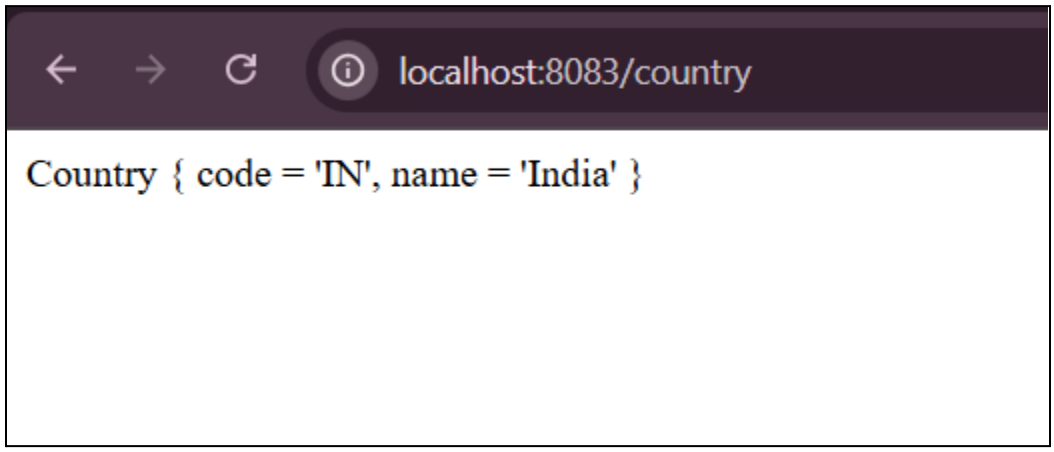


Exercise 4 - REST Country Web Service

CountryController.java

```java
package com.cognizant.spring_learn.controller;

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

import com.cognizant.spring_learn.Country;

@RestController
public class CountryController {
	private static Logger logger = LoggerFactory.getLogger(CountryController.class);

	@GetMapping("/country")
	public String getCountryIndia() {
		logger.info("START - getCountryIndia()");

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

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

Output:



Exercise 5 - REST - Get country based on country code

CountryService.java

```java
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> countryList = (List<Country>) context.getBean("countryList");
    return countryList.stream()
        .filter(c -> c.getCode().equalsIgnoreCase(code))
        .findFirst()
        .orElse(null);
  }
}
```

CountryController.java

```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) {
```

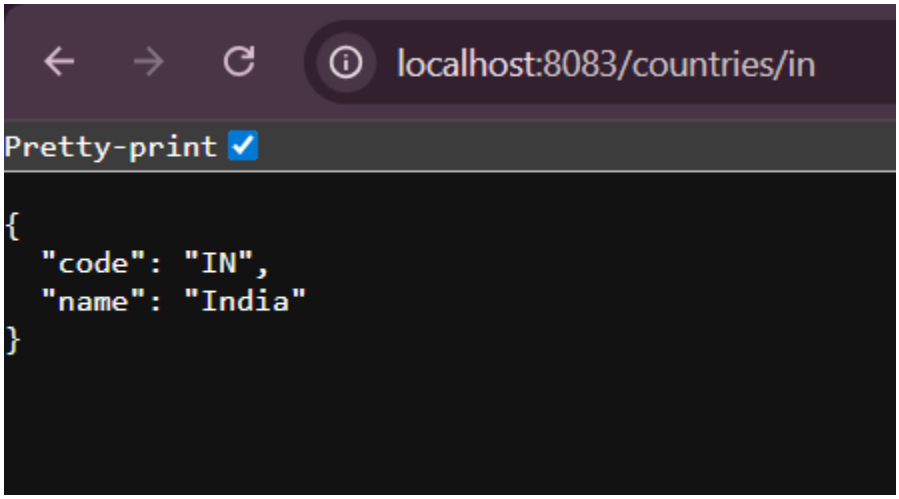
```
        logger.info("START - getCountry({})", code);
        Country country = countryService.getCountry(code);
        logger.info("END - getCountry()");
        return country;

    }
}
```

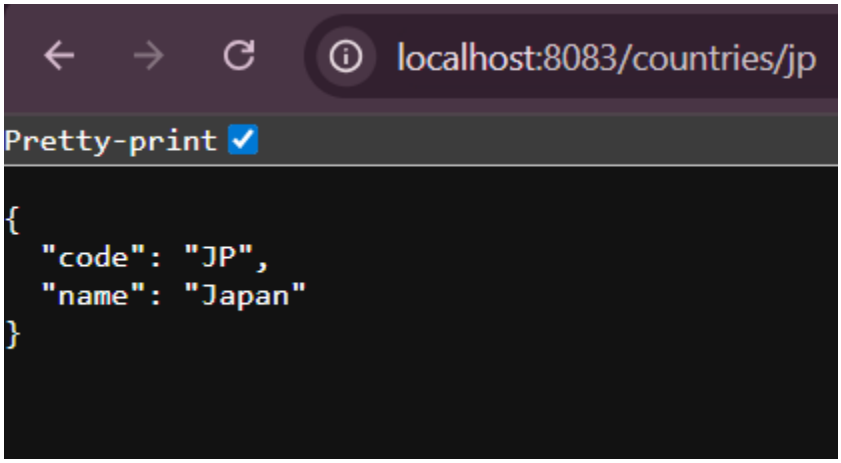
country.xml

```
<?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">

    <!-- Country Beans -->
    <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>

    <bean id="jp" class="com.cognizant.spring_learn.model.Country">
        <property name="code" value="JP"/>
        <property name="name" value="Japan"/>
    </bean>

    <!-- List of Countries -->
    <bean id="countryList" class="java.util.ArrayList">
        <constructor-arg>
            <list>
                <ref bean="in"/>
                <ref bean="us"/>
                <ref bean="jp"/>
            </list>
```

```
    </constructor-arg>
  </bean>

</beans>
```

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;
```

```java
@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:

```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"
}
```