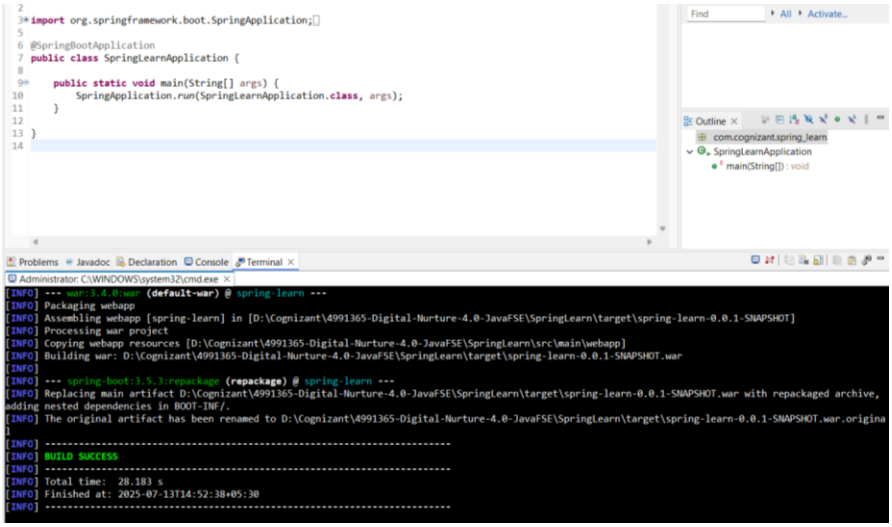
WEEK 4

spring-rest-handson, JWT-handson

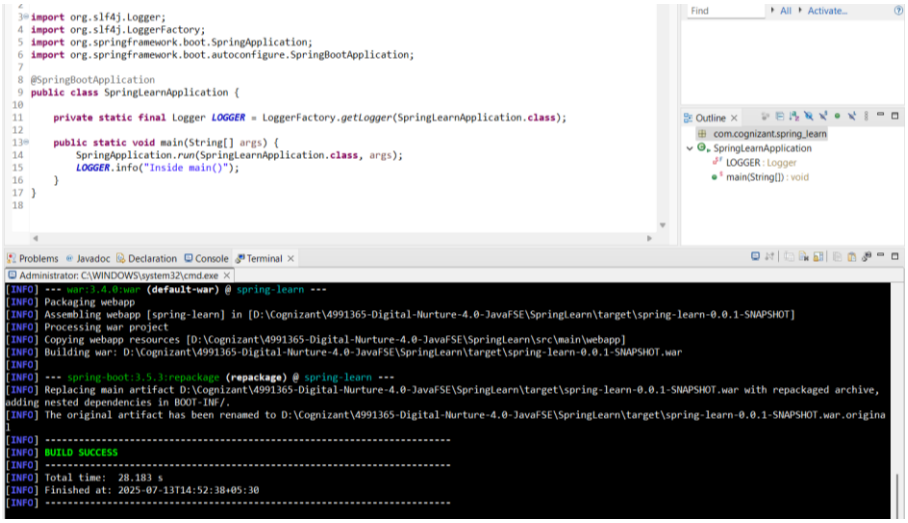
Spring-rest-handson:

1. Create a Spring Web Project using Maven:

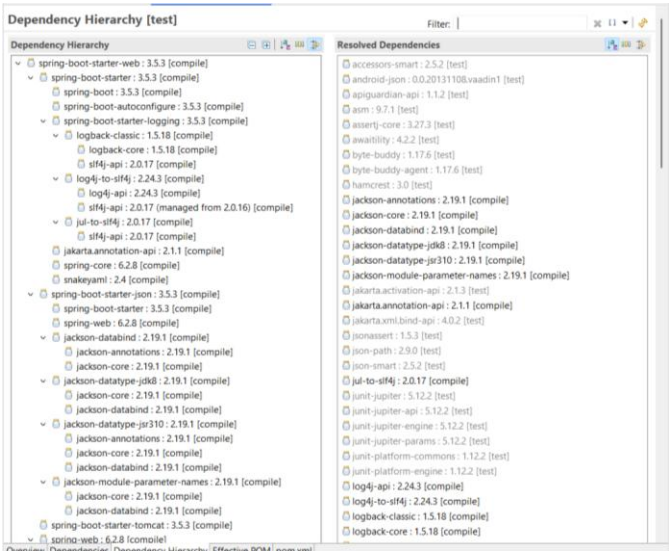
Create a SpringLearn Application and build it:



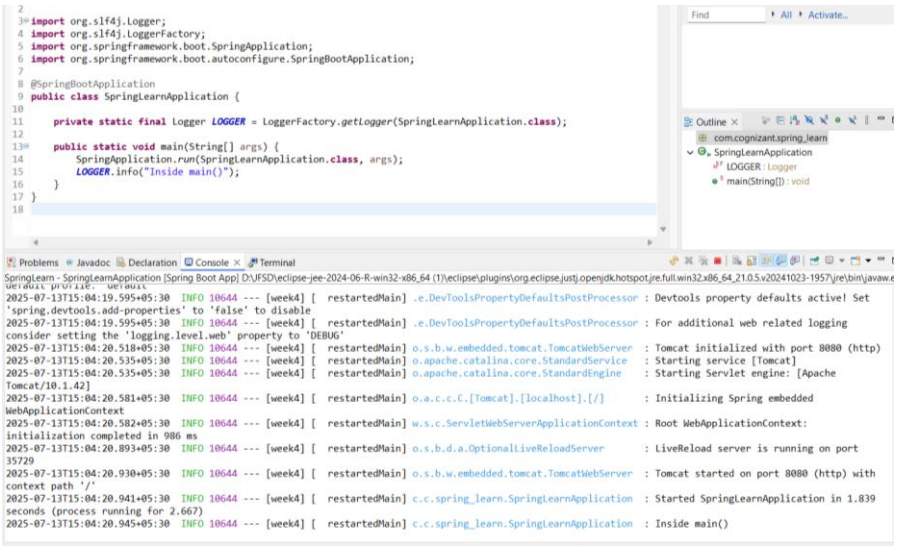
Add Logging to the Application:



Dependency Hierarchy:



Output:



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

Create Country.java in src/main/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("Inside getCode()");
    return code;
  }


  public void setCode(String code) {
    LOGGER.debug("Inside setCode()");
    this.code = code;
  }
```

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

    public void setName(String name) {
        LOGGER.debug("Inside setName()");
        this.name = name;
    }

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

Create a 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
       http://www.springframework.org/schema/beans/spring-beans.xsd">

    <bean id="country" class="com.cognizant.springlearn.Country">
        <property name="code" value="IN"/>
```

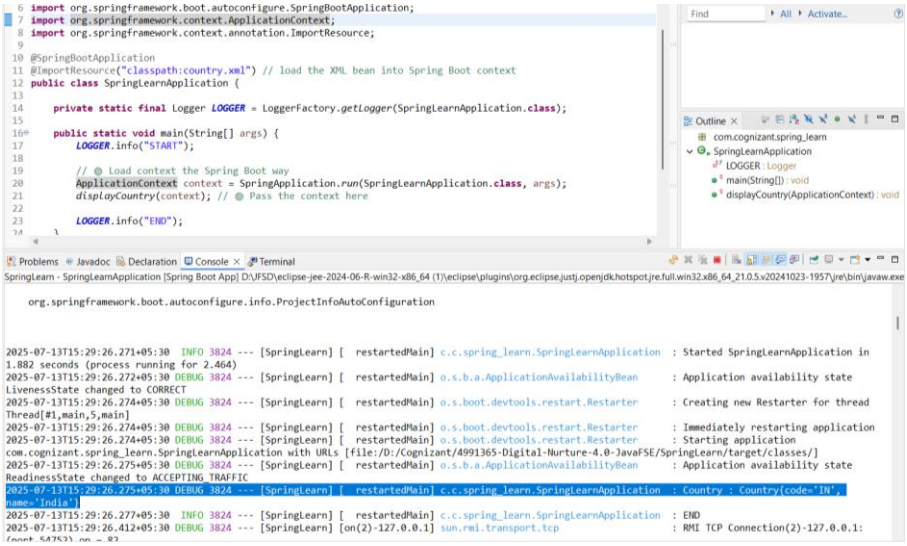
```
    <property name="name" value="India"/>
  </bean>

</beans>
```

Change SpringLearnApplication code:

```
package com.cognizant.spring_learn;

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.annotation.ImportResource;

@SpringBootApplication
@ImportResource("classpath:country.xml") // load the XML bean into Spring Boot context
public class SpringLearnApplication {

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

  public static void main(String[] args) {
    LOGGER.info("START");


    ApplicationContext context = SpringApplication.run(SpringLearnApplication.class, args);
```

```
        displayCountry(context);

        LOGGER.info("END");
    }

    public static void displayCountry(ApplicationContext context) {
        Country country = context.getBean("country", Country.class);
        LOGGER.debug("Country : {}", country.toString());
    }
}
```

Output:



3. Hello World RESTful Web Service:

Create a com.cognizant.spring_learn.controller package

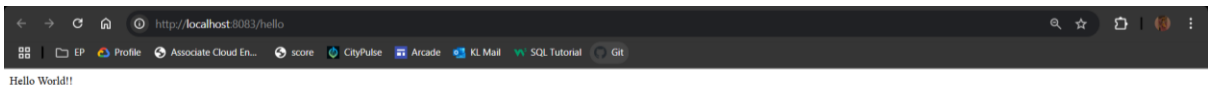
Create a HelloController.java :

```
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()");
    LOGGER.info("END - sayHello()");
    return "Hello World!!";
  }
}
```

Change port to 8083 in application.properties using :

```
server.port=8083
```

Output:

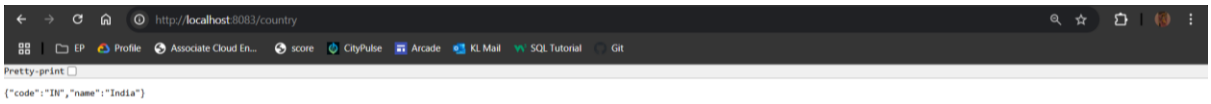


4. REST - Country Web Service:

Create CountryController.java:

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


import com.cognizant.spring_learn.Country;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.*;
import org.springframework.context.ApplicationContext;
import org.springframework.web.bind.annotation.*;


@RestController
public class CountryController {


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


  @Autowired
  private ApplicationContext context;


  @RequestMapping("/country")
  public Country getCountryIndia() {
    LOGGER.info("START - getCountryIndia()");
    Country country = context.getBean("country", Country.class);
    LOGGER.info("END - getCountryIndia()");
    return country;
  }
}
```

Output:



5. REST - Get country based on country code:

Update country.xml with a list of countries:

```
<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.spring_learn.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.Country">
          <property name="code" value="IN"/>
          <property name="name" value="India"/>
        </bean>
        <bean class="com.cognizant.spring_learn.Country">
          <property name="code" value="US"/>
          <property name="name" value="United States"/>
        </bean>
        <bean class="com.cognizant.spring_learn.Country">
          <property name="code" value="DE"/>
          <property name="name" value="Germany"/>
        </bean>
        <bean class="com.cognizant.spring_learn.Country">
          <property name="code" value="JP"/>
          <property name="name" value="Japan"/>
        </bean>
      </list>
    </constructor-arg>
  </bean>

</beans>
```

Create a CountryService in a service package:

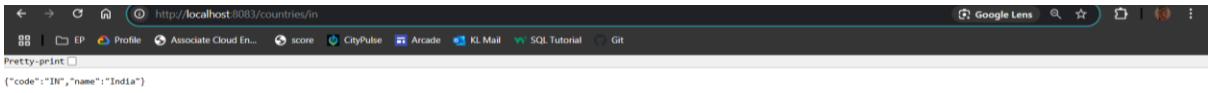
```
package com.cognizant.spring_learn.service;

import com.cognizant.spring_learn.Country;
import org.slf4j.Logger;
```

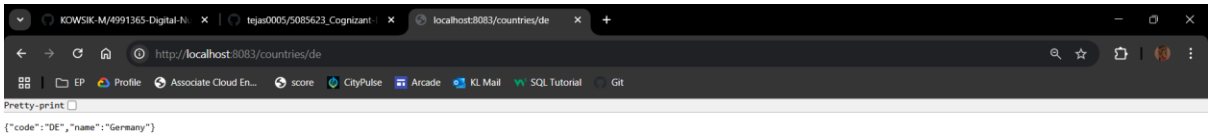
```
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationContext;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class CountryService {

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

   @Autowired
   private ApplicationContext context;

   public Country getCountry(String code) {
      LOGGER.info("START - getCountry()");
      List<Country> countryList = (List<Country>) context.getBean("countryList");

      // Lambda-based search (case-insensitive)
      Country match = countryList.stream()
            .filter(c -> c.getCode().equalsIgnoreCase(code))
            .findFirst()
            .orElse(null);

      LOGGER.info("END - getCountry()");
      return match;
   }
}
```

Update CountryController to get Country by Country Code:

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

import com.cognizant.spring_learn.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;
  }
}
```

Output:





JWT-handson:

6. Create authentication service that returns JWT:

Add Spring Security dependency:

<dependency>

```
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-security</artifactId>
        </dependency>
```

Create AuthenticationController.java:

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

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.web.bind.annotation.*;

import java.util.HashMap;
import java.util.Map;

@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 - authenticate()");
    LOGGER.debug("Authorization Header: {}", authHeader);

    Map<String, String> map = new HashMap<>();
    map.put("token", ""); // JWT will be filled in later

    LOGGER.info("END - authenticate()");
```

```
        return map;
    }
}
```

Create a com.cognizant.spring_learn.config and in that SecurityConfig.java:

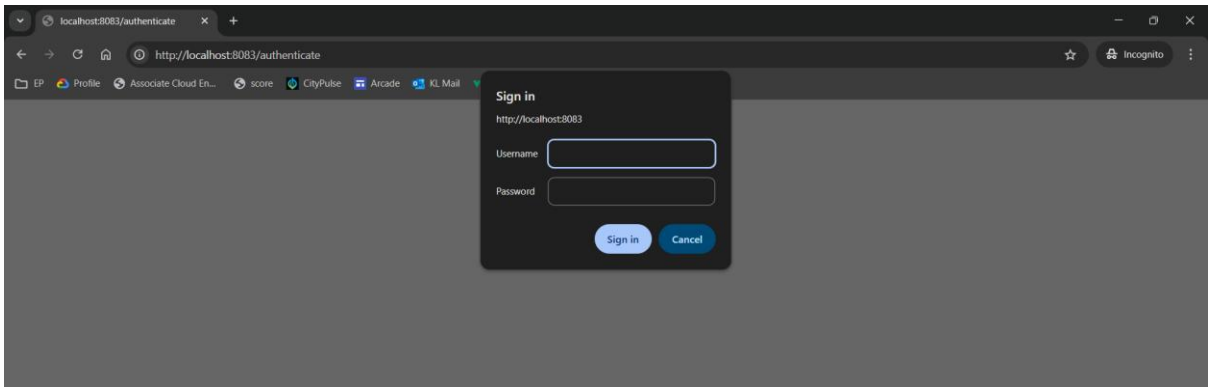
```
package com.cognizant.spring_learn.config;

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(authz -> authz
                .requestMatchers("/countries").hasRole("USER")
                .requestMatchers("/authenticate").hasAnyRole("USER", "ADMIN")
                .anyRequest().authenticated()
            )
            .httpBasic();

        return http.build();
    }
}
```

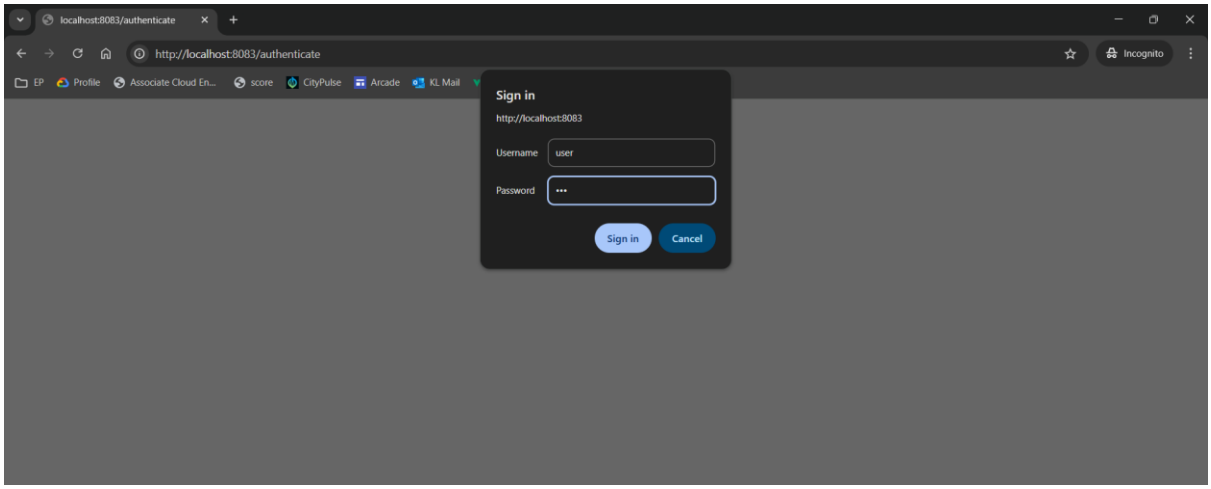
Create a MyUserDetailsConfig:

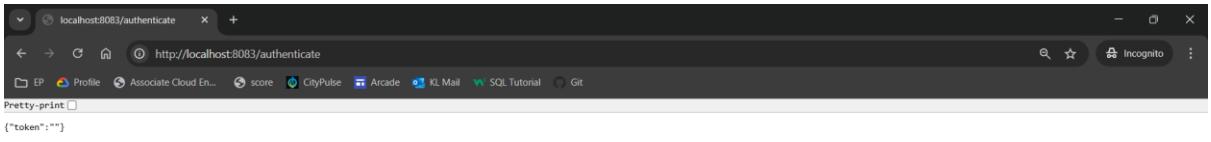
```
package com.cognizant.spring_learn.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.core.userdetails.*;
import org.springframework.security.provisioning.InMemoryUserDetailsManager;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;

@Configuration
public class MyUserDetailsConfig {

  @Bean
  public InMemoryUserDetailsManager userDetailsService(PasswordEncoder passwordEncoder) {
    UserDetails user = User.builder()
        .username("user")
        .password(passwordEncoder.encode("pwd"))
        .roles("USER")
        .build();

    UserDetails admin = User.builder()
        .username("admin")
        .password(passwordEncoder.encode("admin"))
        .roles("ADMIN")
        .build();

    return new InMemoryUserDetailsManager(user, admin);
  }
```

```
    @Bean
    public PasswordEncoder passwordEncoder() {
        return new BCryptPasswordEncoder();
    }
}
```

Output:






localhost:8083/authenticate
http://localhost:8083/authenticate
Incognito
EP
Profile
Associate Cloud En...
score
CityPulse
Arcade
KL Mail
SQL Tutorial
Git
Pretty-print
{"token":""}