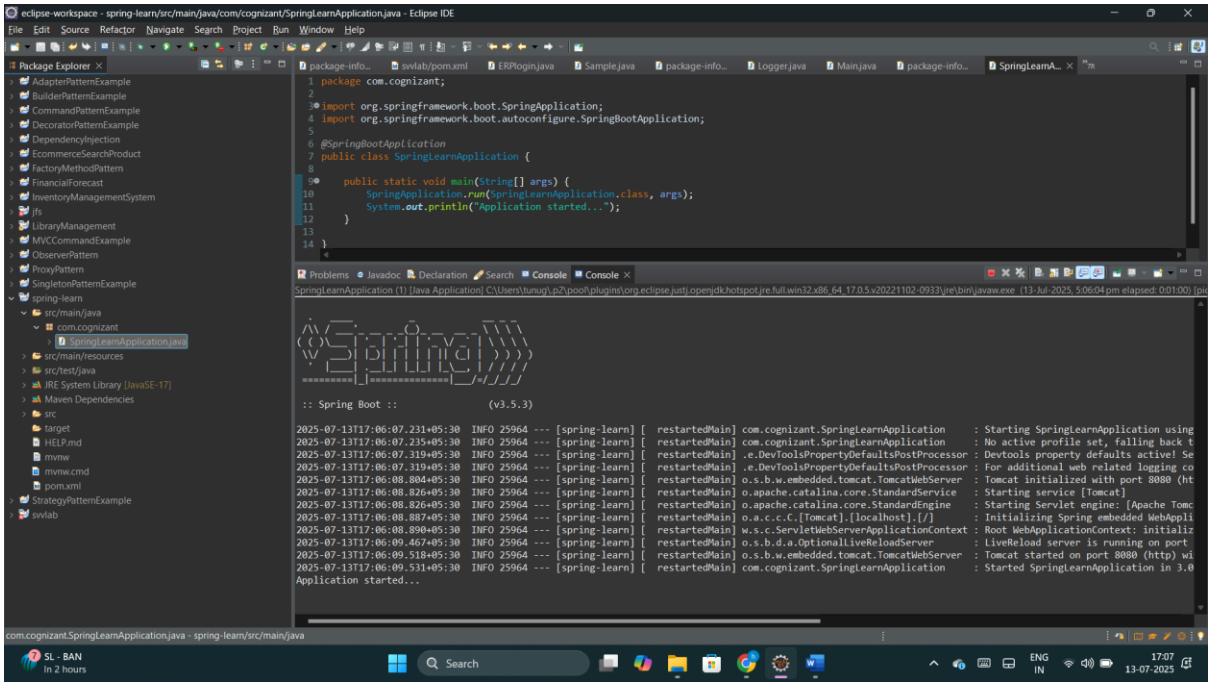
Name: Tunuguntla Harshitha
Superset Id: 4992580
Email : 2200033132cseh@gmail.com

Week 4 – 1. Spring-Rest-Handson:

1.Create spring web project using maven

Output:

Src/main/java and output

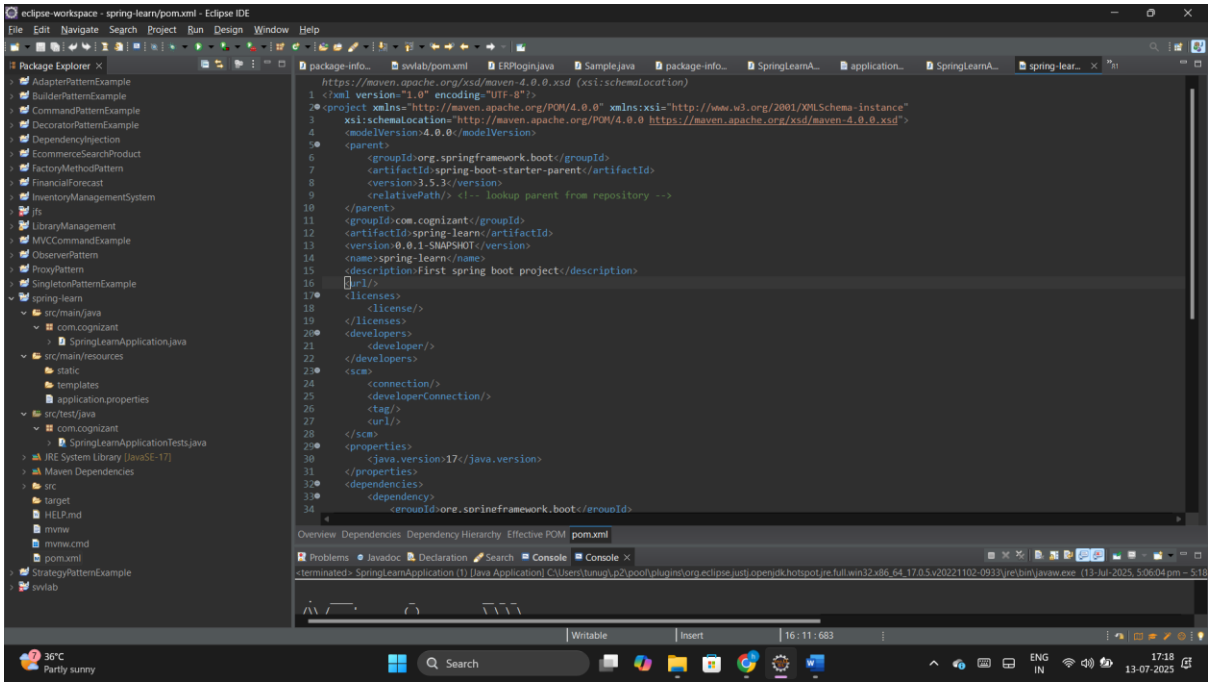


Walk through of the code:

@SpringBootApplication is a shortcut annotation for:

1. @Configuration – configuration class
2. @EnableAutoConfiguration – enables Spring Boot auto-configuration
3. @ComponentScan – scans com.cognizant package and sub-packages for Spring components.

Pom.xml:



Header and Meta data:
Defines this file as a Maven POM (Project Object Model)

XML namespaces used to validate and understand Maven structure.

Parent configuration:
Inherits settings from Spring Boot's parent POM

Provides default plugin configurations and dependency versions

Ensures consistency across Spring Boot projects

Project identity:
groupId: your company or domain name

artifactId: project name

version: version of the project (you can later change to 1.0.0 etc.)

Project description :
Used for project documentation or publishing to repositories

Java version –

Tells Spring Boot and Maven to compile using Java 17

Make sure your system has JDK 17 installed

Dependencies – webstarter , dev tools, unit testing :

Adds Spring MVC, embedded Tomcat, JSON support

Required for web/REST APIs,

Enables auto-restart and live reload during development

Only runs in dev mode (not in production builds),

Adds JUnit, Mockito, and other test libraries.

Dependency Hierarchy:
spring-boot-starter-web

└— spring-boot-starter

  ├— spring-boot

  ├— spring-core

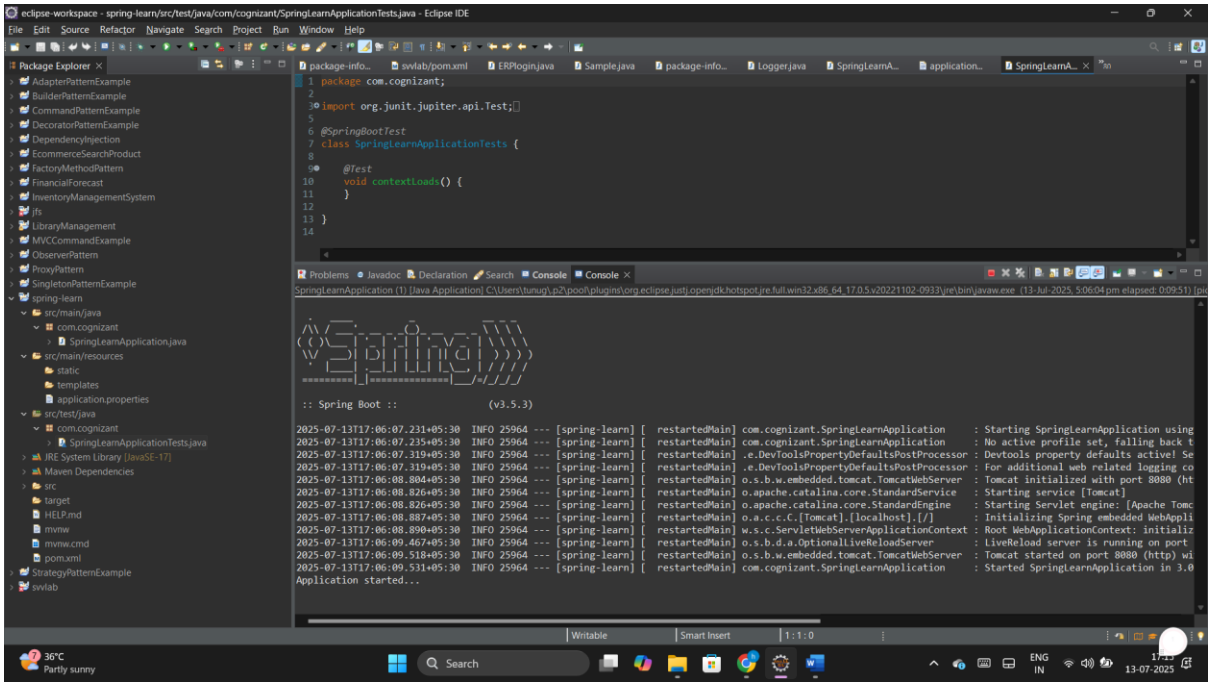
  └— ...

└— spring-web

└— spring-webmvc

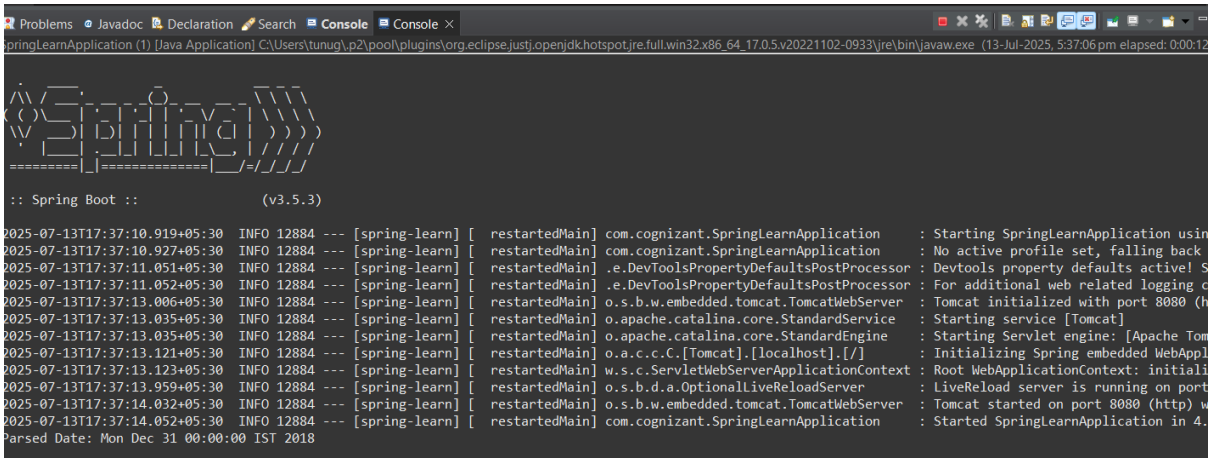
└— tomcat-embed-core

Sec/test/java:



2. **Load SimpleDateFormat from Spring Configuration XML**

Output:



2.SPRING-REST-HANDSON

1. Hello World RESTful Web Service :

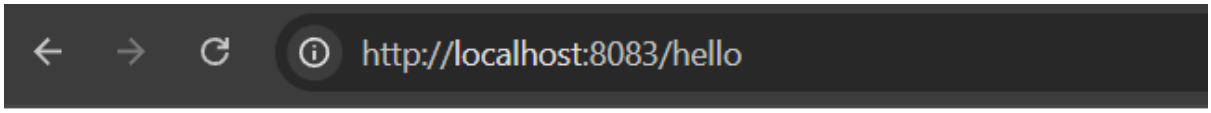
HelloController.java:

```java
package com.cognizant.springlearn.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 response = "Hello World!!";
        LOGGER.info("END - sayHello()");
        return response;
    }
}
```

Application.properties:

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

Output:



Hello World!!

2. **REST - Country Web Service**

Output:

1. What happens in the controller method?

When a GET request is made to /country, Spring maps it to getCountryIndia() because of the @RequestMapping("/country") annotation.

Inside the method:

A Spring XML application context is created and loads date-format.xml.

It retrieves the Country bean from XML (<bean id="country" ...>) — which contains code and name properties.

The Country object is returned.

2. How the bean is converted into JSON reponse?
   Spring Boot automatically uses **Jackson** (a JSON serialization library) to convert Java objects to JSON format.
   Behind the scenes:
   You returned a Country object.
   Spring uses HttpMessageConverter, specifically MappingJackson2HttpMessageConverter, to:
   Convert your Country object into JSON.
   Set the appropriate Content-Type header: application/json.

3. In network tab of developer tools show the HTTP header details received
   To view this in Chrome:

1. Open your browser.
2. Go to http://localhost:8083/country.
3. Right-click anywhere → Inspect → Go to Network tab.
4. Refresh the page if needed.
5. Click on the request named /country.

4. In postman click on "Headers" tab to view the HTTP header details received

Steps:

1. Open Postman.
2. Make a GET request to:

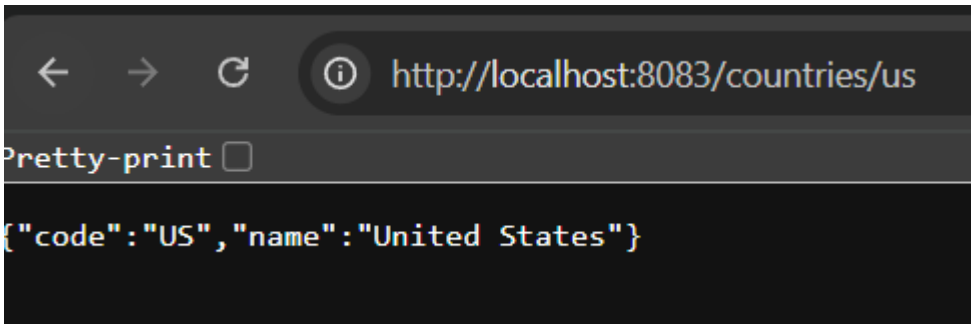
   bash

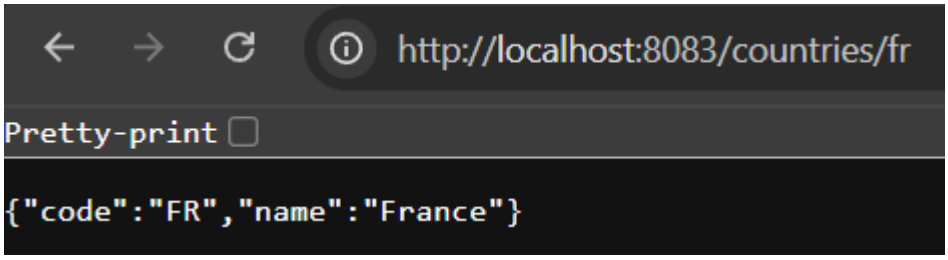
   CopyEdit

   http://localhost:8083/country
3. Click Send.
4. Click the "Headers" tab in the response area (below the body).

Get country based on country code

Output:





5. **JWT-HANDSON:**

Create authentication service that returns JWT **:**