WEEK - 4 SPRING REST USING SPRING BOOT 3

1. Create a Spring Web Project using Maven

//SpringLearnApplication.java

```
package com.cognizant.spring_learn;

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);
        System.out.println("SpringLearnApplication Started Successfully!");
    }
}
```

//hello.java

```
package com.cognizant.spring_learn;

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

@RestController
public class hello {
```

```
    @GetMapping("/hello")
    public String hello() {
        return "Welcome to Spring Boot!";
    }
}
```

OUTPUT:

2. Spring Core – Load SimpleDateFormat from Spring Configuration XML

//SpringLearnApplication.java

```
package com.cognizant.springlearn;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;

@SpringBootApplication
public class SpringLearnApplication {
```

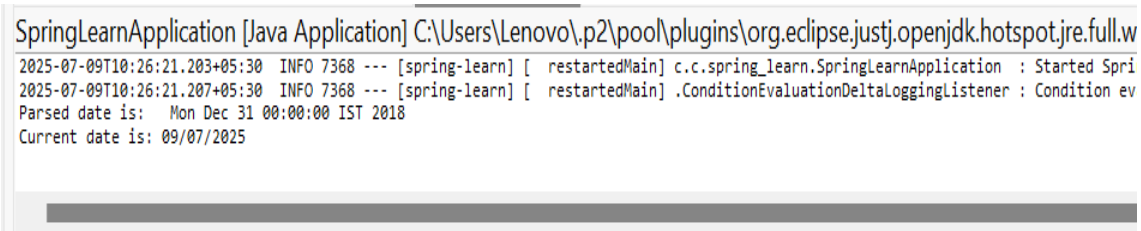
```
    public static void main(String[] args) {
        SpringApplication.run(SpringLearnApplication.class, args);

        displayDate();         // parses a fixed date
        displayCurrentDate();  // formats and prints “today”
    }

    private static void displayDate() {
        ApplicationContext ctx = new ClassPathXmlApplicationContext("date-
format.xml");
        SimpleDateFormat format = ctx.getBean("dateFormat",
SimpleDateFormat.class);

        try {
            Date date = format.parse("31/12/2018");
            System.out.println("Parsed date is:   " + date);
        } catch (ParseException e) {
            System.err.println("Parse error:     " + e.getMessage());
        }
    }

    private static void displayCurrentDate() {
        ApplicationContext ctx = new ClassPathXmlApplicationContext("date-
format.xml");
        SimpleDateFormat format = ctx.getBean("dateFormat",
SimpleDateFormat.class);

        Date now = new Date();
        String formattedNow = format.format(now);
```

```
        System.out.println("Current date is: " + formattedNow);
    }
}
```

//date-format.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">

    <!-- define a shared SimpleDateFormat bean -->
    <bean id="dateFormat" class="java.text.SimpleDateFormat">
        <constructor-arg value="dd/MM/yyyy"/>
    </bean>

</beans>
```

OUTPUT:



3. Hello World RESTful Web Service

//SpringLearnApplication.java

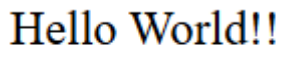
```
package com.cognizant.spring_learn;

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);
    }
}
```

//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(">> sayHello() start");
        LOGGER.info("<< sayHello() end");
        return "Hello World!!";
    }
}
```

OUTPUT:



4. REST - Country Web Service

//Country.java

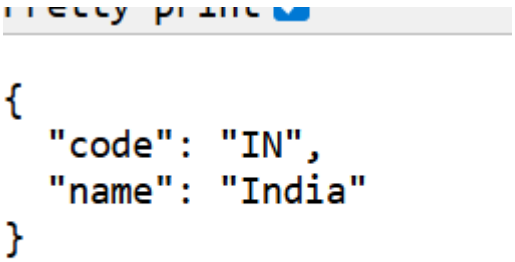
```
package com.cognizant.spring_learn.model;

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

    public Country() {}

    public Country(String code, String name) {
        this.code = code;
        this.name = name;
    }
```

```
    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;
    }
}

//CountryController.java
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(">> getCountryIndia() called");

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

      LOGGER.info("<< getCountryIndia() returning: {}", country.getName());
      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">

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

```
</bean>

</beans>
```

OUTPUT:



5. REST - Get country based on country code

//CountryService.java

```
package com.cognizant.springlearn.service;

import com.cognizant.springlearn.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);

        // Case-insensitive match using lambda
        return countries.stream()
                .filter(c -> c.getCode().equalsIgnoreCase(code))
                .findFirst()
                .orElse(null);
    }
}
```

//CountryController.java

```
package com.cognizant.springlearn.controller;

import com.cognizant.springlearn.model.Country;
import com.cognizant.springlearn.service.CountryService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

@RestController
public class CountryController {

    @Autowired
    private CountryService countryService;

    @GetMapping("/countries/{code}")
```

```
    public Country getCountry(@PathVariable String code) {
        return countryService.getCountry(code);
    }
}
```

//Country.java

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

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

    public Country() {}

    public Country(String code, String name) {
        this.code = code;
        this.name = name;
    }

    public String getCode() {
        return code;
    }

    public void setCode(String code) {
        this.code = code;
    }
```

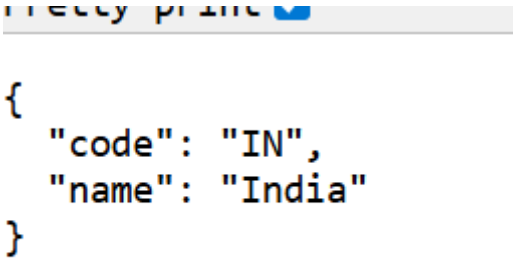
```
    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}
```

//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="countryList" class="java.util.ArrayList">
        <constructor-arg>
            <list>
                <bean class="com.cognizant.springlearn.model.Country">
                    <property name="code" value="IN" />
                    <property name="name" value="India" />
                </bean>
                <bean class="com.cognizant.springlearn.model.Country">
                    <property name="code" value="US" />
                    <property name="name" value="United States" />
```

```
        </bean>
        <bean class="com.cognizant.springlearn.model.Country">
            <property name="code" value="CN" />
            <property name="name" value="China" />
        </bean>
    </list>
  </constructor-arg>
 </bean>
</beans>
```

OUTPUT:



6. Create authentication service that returns JWT

//SpringLearnApplication.java

```
package com.cognizant.spring_learn;

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);
    }
}
```

//AuthController.java

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

import com.cognizant.spring_learn.util.JwtUtil;
import jakarta.servlet.http.HttpServletRequest;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.Base64;

@RestController
public class AuthController {

    @Autowired
    private JwtUtil jwtUtil;

    @GetMapping("/authenticate")
    public ResponseEntity<?> authenticate(HttpServletRequest request) {
        String authHeader = request.getHeader("Authorization");

        if (authHeader == null || !authHeader.startsWith("Basic ")) {
```

```
            return ResponseEntity.status(401).body("Missing or invalid
Authorization header");
        }

        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 (username.equals("user") && password.equals("pwd")) {
            String token = jwtUtil.generateToken(username);
            return ResponseEntity.ok().body("{\"token\":\"" + token + "\"}");
        } else {
            return ResponseEntity.status(403).body("Invalid credentials");
        }
    }
}
```

// 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 secretKey = "mySecretKey";

    public String generateToken(String username) {
        return Jwts.builder()
                .setSubject(username)
                .setIssuedAt(new Date())
                .setExpiration(new Date(System.currentTimeMillis() +
3600000)) // 1 hour
                .signWith(SignatureAlgorithm.HS256, secretKey)
                .compact();
    }
}
```

//SecurityConfig.java

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

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.web.SecurityFilterChain;
import
org.springframework.security.config.annotation.web.builders.HttpSecurity;
```

```
@Configuration
public class SecurityConfig {


    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws
Exception {
        http
            .csrf(csrf -> csrf.disable())
            .authorizeHttpRequests(auth -> auth
                .requestMatchers("/authenticate").permitAll()
                .anyRequest().authenticated()
            )
            .httpBasic();


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

OUTPUT: