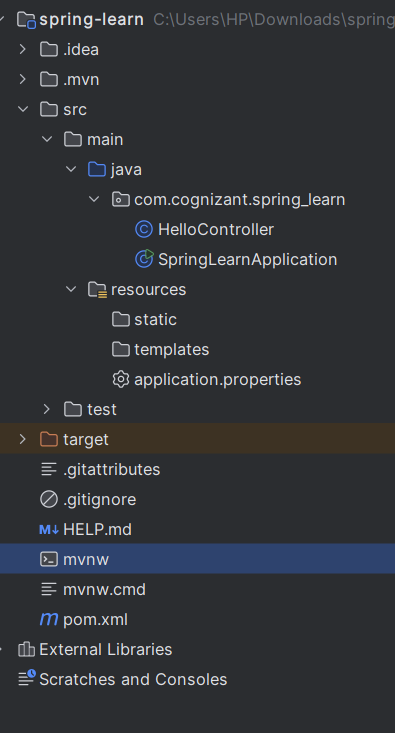
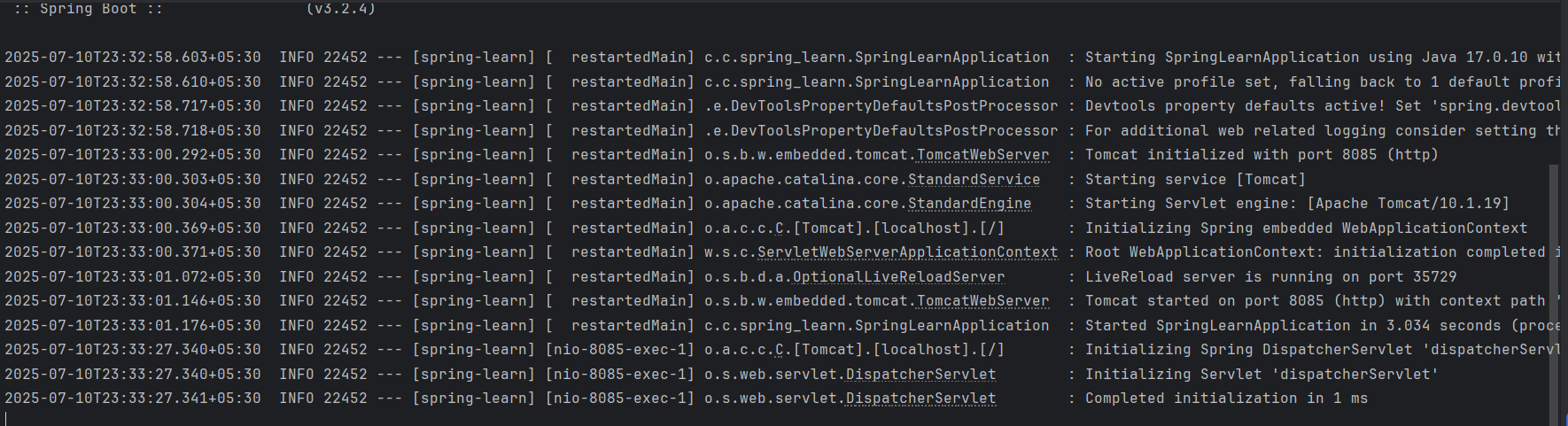
**Spring REST using Spring Boot 3**

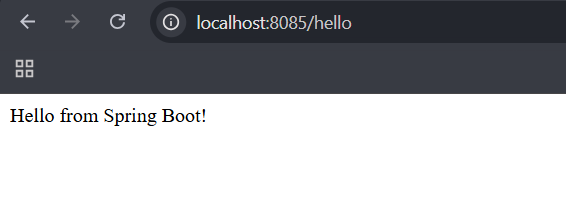
**Create a Spring Web Project using Maven**

****

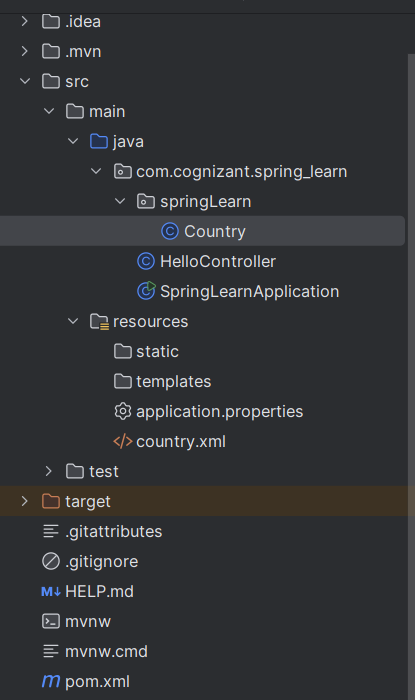
|  |
| --- |
| **SpringLearnApplication.java**  **package com.cognizant.spring\_learn;  import org.springframework.boot.SpringApplication; import org.springframework.boot.autoconfigure.SpringBootApplication;  @SpringBootApplication public class SpringLearnApplication {   public static void main(String[] args) {  System.*out*.println("SpringLearnApplication started... ✅");  SpringApplication.*run*(SpringLearnApplication.class, args);  } }**  **HelloController.java**  **package com.cognizant.spring\_learn;  import org.springframework.web.bind.annotation.GetMapping; import org.springframework.web.bind.annotation.RestController;  @RestController public class HelloController {   @GetMapping("/hello")  public String hello() {  return "Hello from Spring Boot!";  } }** |

**OUTPUT-:**

****

****

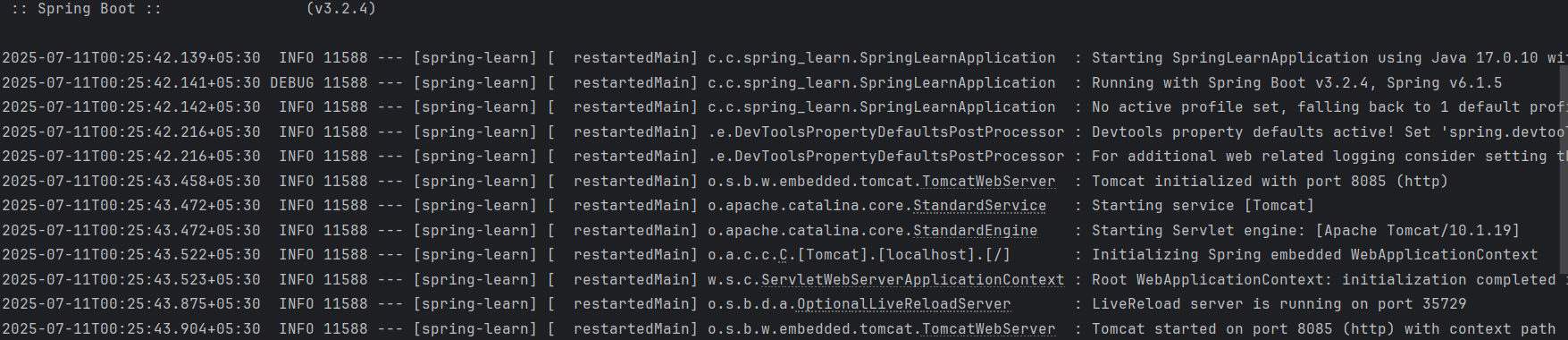
**Spring Core – Load Country from Spring Configuration XML**

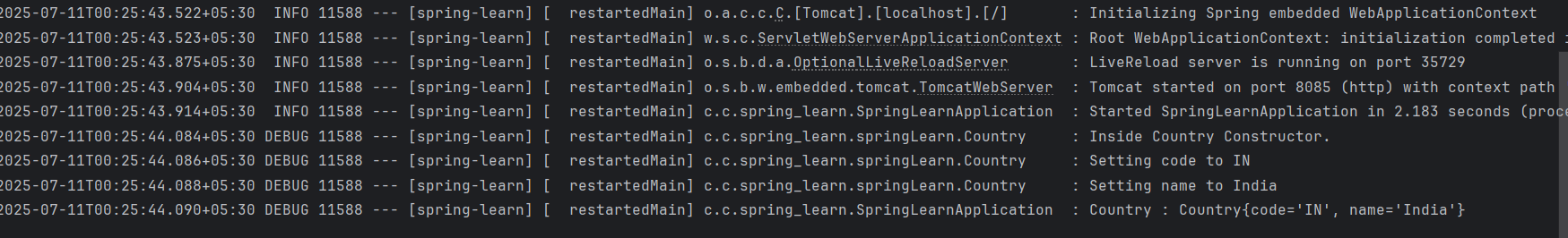
****

|  |
| --- |
| **Country.java**  **package com.cognizant.spring\_learn.springLearn;  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("Getting code");  return code;  }   public void setCode(String code) {  *LOGGER*.debug("Setting code to {}", code);  this.code = code;  }   public String getName() {  *LOGGER*.debug("Getting name");  return name;  }   public void setName(String name) {  *LOGGER*.debug("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 com.cognizant.spring\_learn.springLearn.Country; 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.support.ClassPathXmlApplicationContext;  @SpringBootApplication public class SpringLearnApplication {   private static final Logger *LOGGER* = LoggerFactory.*getLogger*(SpringLearnApplication.class);   public static void main(String[] args) {  SpringApplication.*run*(SpringLearnApplication.class, args);  *displayCountry*();  }   public static void displayCountry() {  ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  Country country = (Country) context.getBean("country", Country.class);  *LOGGER*.debug("Country : {}", country.toString());  } }**  **Hello.java**  **<?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.springLearn.Country">  <property name="code" value="IN" />  <property name="name" value="India" />  </bean>  </beans>** |

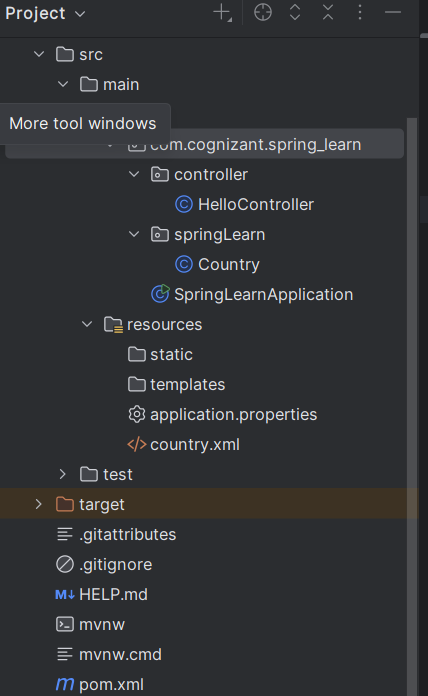
**OUTPUT-:**

****

****

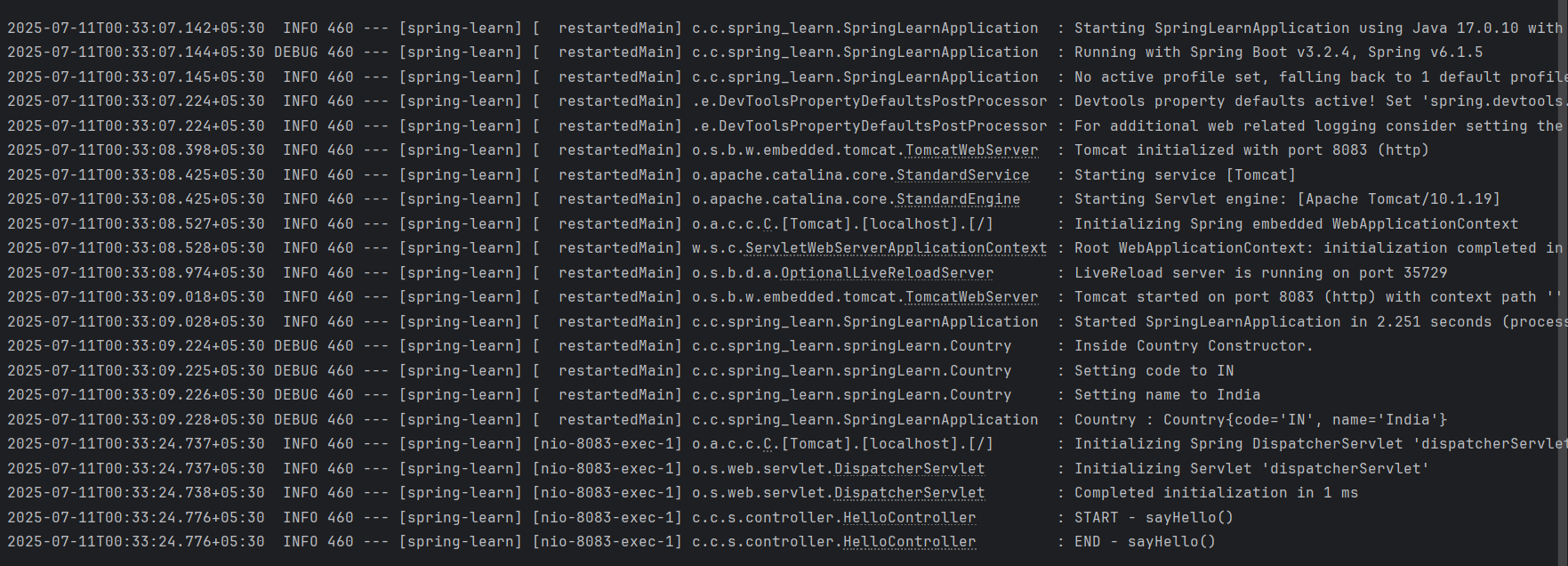
****

**Hello World RESTful Web Service**

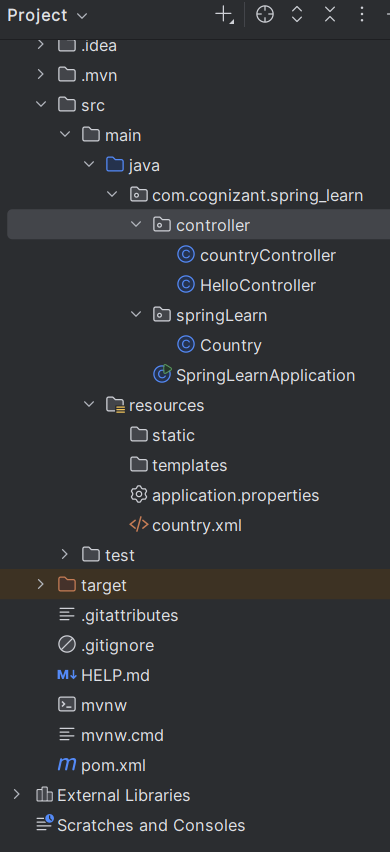
****

|  |
| --- |
| **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;**  **}**  **}**  **SpringLearnApplication.java**  **package com.cognizant.spring\_learn;  import com.cognizant.spring\_learn.springLearn.Country; 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.support.ClassPathXmlApplicationContext;  @SpringBootApplication public class SpringLearnApplication {   private static final Logger *LOGGER* = LoggerFactory.*getLogger*(SpringLearnApplication.class);   public static void main(String[] args) {  SpringApplication.*run*(SpringLearnApplication.class, args);  *displayCountry*();  }   public static void displayCountry() {  ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  Country country = (Country) context.getBean("country", Country.class);  *LOGGER*.debug("Country : {}", country.toString());  } }**  **Country.java**  **package com.cognizant.spring\_learn.springLearn;  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("Getting code");  return code;  }   public void setCode(String code) {  *LOGGER*.debug("Setting code to {}", code);  this.code = code;  }   public String getName() {  *LOGGER*.debug("Getting name");  return name;  }   public void setName(String name) {  *LOGGER*.debug("Setting name to {}", name);  this.name = name;  }   @Override  public String toString() {  return "Country{" +  "code='" + code + '\'' +  ", name='" + name + '\'' +  '}';  } }** |

**OUTPUT-:**

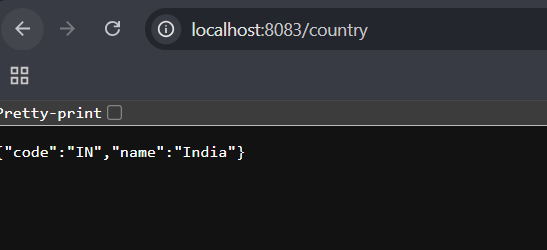
****

**REST - Country Web Service**

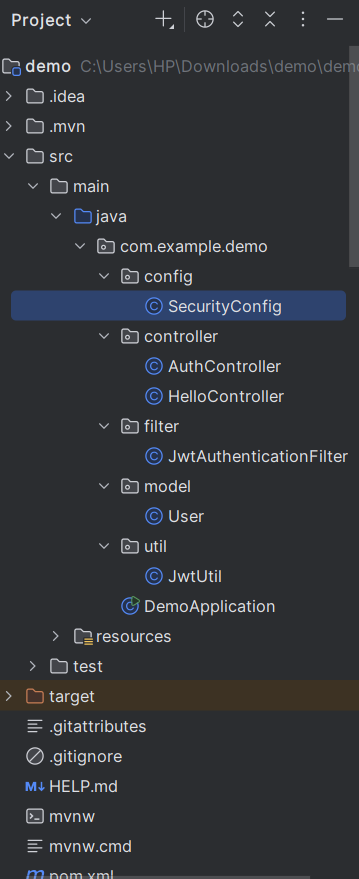
****

|  |
| --- |
| **countryController.java**  **package com.cognizant.spring\_learn.controller;  import com.cognizant.spring\_learn.springLearn.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.spring\_learn;  import com.cognizant.spring\_learn.springLearn.Country; 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.support.ClassPathXmlApplicationContext;  @SpringBootApplication public class SpringLearnApplication {   private static final Logger *LOGGER* = LoggerFactory.*getLogger*(SpringLearnApplication.class);   public static void main(String[] args) {  SpringApplication.*run*(SpringLearnApplication.class, args);  *displayCountry*();  }   public static void displayCountry() {  ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  Country country = (Country) context.getBean("country", Country.class);  *LOGGER*.debug("Country : {}", country.toString());  } }** |

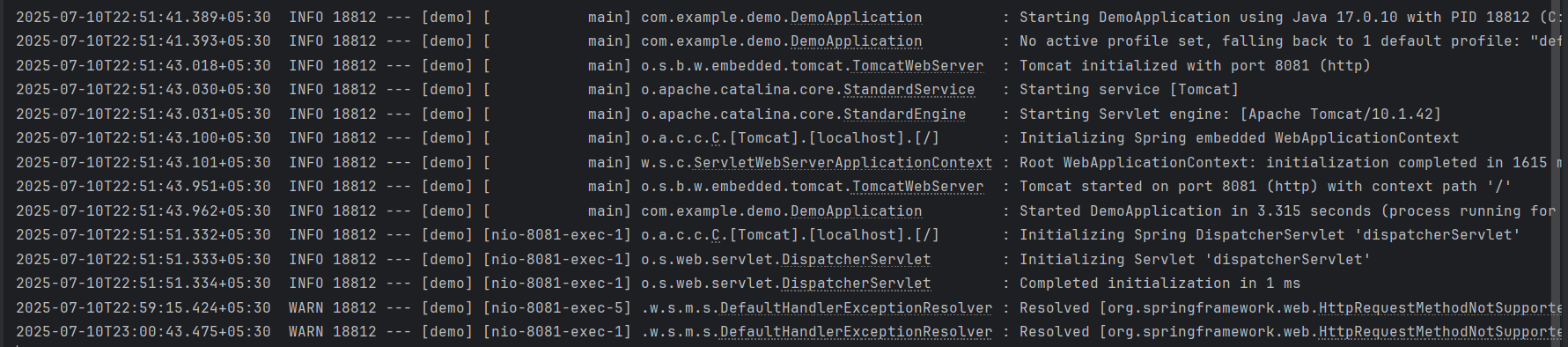
**OUTPUT-:**

****

**Create authentication service that returns JWT**



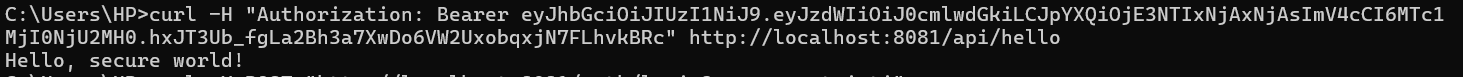
|  |
| --- |
| SecurityConfig.java  package com.example.demo.config;  import com.example.demo.filter.JwtAuthenticationFilter; import org.springframework.context.annotation.Bean; import org.springframework.context.annotation.Configuration; import org.springframework.security.authentication.AuthenticationManager; import org.springframework.security.config.annotation.authentication.configuration.AuthenticationConfiguration; import org.springframework.security.config.annotation.web.builders.HttpSecurity; import org.springframework.security.config.annotation.web.configurers.AbstractHttpConfigurer; import org.springframework.security.config.http.SessionCreationPolicy; import org.springframework.security.web.SecurityFilterChain; import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;  @Configuration public class SecurityConfig {   private final JwtAuthenticationFilter jwtAuthFilter;   public SecurityConfig(JwtAuthenticationFilter jwtAuthFilter) {  this.jwtAuthFilter = jwtAuthFilter;  }   @Bean  public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {  http.csrf(AbstractHttpConfigurer::disable)  .authorizeHttpRequests(auth -> auth  .requestMatchers("/auth/\*\*").permitAll() // allow login without auth  .anyRequest().authenticated() // secure everything else  )  .sessionManagement(session -> session  .sessionCreationPolicy(SessionCreationPolicy.*STATELESS*)  )  .addFilterBefore(jwtAuthFilter, UsernamePasswordAuthenticationFilter.class);   return http.build();  }    @Bean  public AuthenticationManager authenticationManager(AuthenticationConfiguration config) throws Exception {  return config.getAuthenticationManager();  } }  AuthController.java  package com.example.demo.controller;  import com.example.demo.util.JwtUtil; import org.springframework.web.bind.annotation.\*;  @RestController @RequestMapping("/auth") public class AuthController {   private final JwtUtil jwtUtil;   public AuthController(JwtUtil jwtUtil) {  this.jwtUtil = jwtUtil;  }   @GetMapping("/login")  public String login(@RequestParam String username) {  return jwtUtil.generateToken(username);  } }  JwtAuthenticationFilter.java  package com.example.demo.filter;  import com.example.demo.util.JwtUtil; import jakarta.servlet.FilterChain; import jakarta.servlet.ServletException; import jakarta.servlet.http.HttpServletRequest; import jakarta.servlet.http.HttpServletResponse; import org.springframework.security.authentication.UsernamePasswordAuthenticationToken; import org.springframework.security.core.context.SecurityContextHolder; import org.springframework.security.web.authentication.WebAuthenticationDetailsSource; import org.springframework.stereotype.Component; import org.springframework.web.filter.OncePerRequestFilter;  import java.io.IOException; import java.util.Collections;  @Component public class JwtAuthenticationFilter extends OncePerRequestFilter {   private final JwtUtil jwtUtil;   public JwtAuthenticationFilter(JwtUtil jwtUtil) {  this.jwtUtil = jwtUtil;  }   @Override  protected void doFilterInternal(HttpServletRequest request,  HttpServletResponse response,  FilterChain filterChain)  throws ServletException, IOException {  String authHeader = request.getHeader("Authorization");   if (authHeader != null && authHeader.startsWith("Bearer ")) {  String token = authHeader.substring(7);  if (jwtUtil.validateToken(token)) {  String username = jwtUtil.extractUsername(token);  UsernamePasswordAuthenticationToken authentication =  new UsernamePasswordAuthenticationToken(username, null, Collections.*emptyList*());  authentication.setDetails(new WebAuthenticationDetailsSource().buildDetails(request));  SecurityContextHolder.*getContext*().setAuthentication(authentication);  }  }  filterChain.doFilter(request, response);  } }  JwtUtil.java  package com.example.demo.util;  import io.jsonwebtoken.\*; import io.jsonwebtoken.security.Keys; import org.springframework.stereotype.Component;  import java.security.Key; import java.util.Date;  @Component public class JwtUtil {   private final Key secretKey = Keys.*secretKeyFor*(SignatureAlgorithm.*HS256*);  private final long expirationTime = 86400000; // 24 hours   public String generateToken(String username) {  return Jwts.*builder*()  .setSubject(username)  .setIssuedAt(new Date())  .setExpiration(new Date(System.*currentTimeMillis*() + expirationTime))  .signWith(secretKey)  .compact();  }   public String extractUsername(String token) {  return Jwts.*parserBuilder*()  .setSigningKey(secretKey)  .build()  .parseClaimsJws(token)  .getBody()  .getSubject();  }   public boolean validateToken(String token) {  try {  Jwts.*parserBuilder*().setSigningKey(secretKey).build().parseClaimsJws(token);  return true;  } catch (JwtException e) {  return false;  }  } }  HelloController.java  package com.example.demo.controller;  import org.springframework.web.bind.annotation.GetMapping; import org.springframework.web.bind.annotation.RestController;  @RestController public class HelloController {   @GetMapping("/api/hello")  public String hello() {  return "Hello, secure world!";  } } |



**OUTPUT-:**

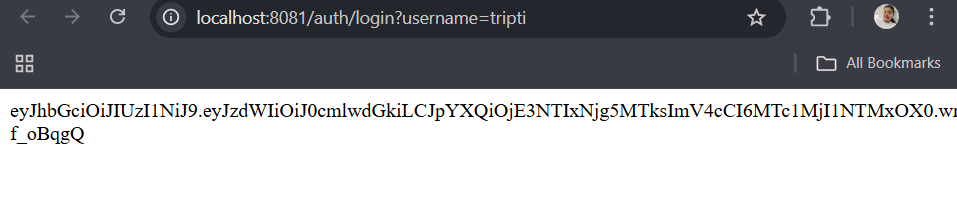
**JWT token generated**

eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ0cmlwdGkiLCJpYXQiOjE3NTIxNjAxNjAsImV4cCI6MTc1MjI0NjU2MH0.hxJT3Ub\_fgLa2Bh3a7XwDo6VW2UxobqxjN7FLhvkBRc



**Finding the JWT token -:**





Confirming it is a **valid JWT** with the correct claims

