**Week 4 Hands-on**

**Spring REST using Spring Boot 3**

**Spring-rest-handson:**

**Ex. 1 (Create a Spring Web Project using Maven)**

package com.example.spring\_rest\_handson;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ImportResource;

@SpringBootApplication

@ImportResource("classpath:beans.xml")

public class SpringRestHandsonApplication {

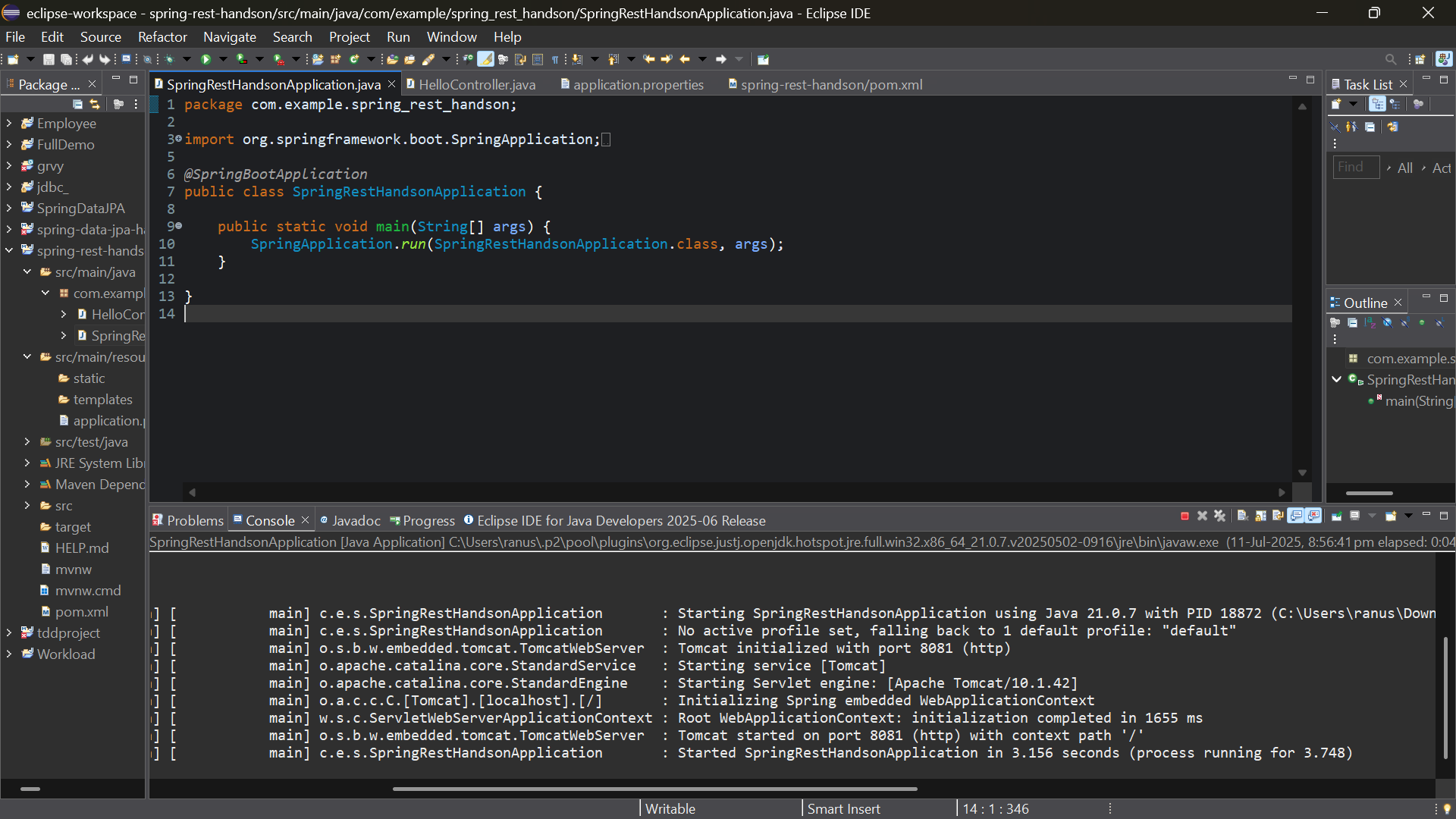
public static void main(String[] args) {

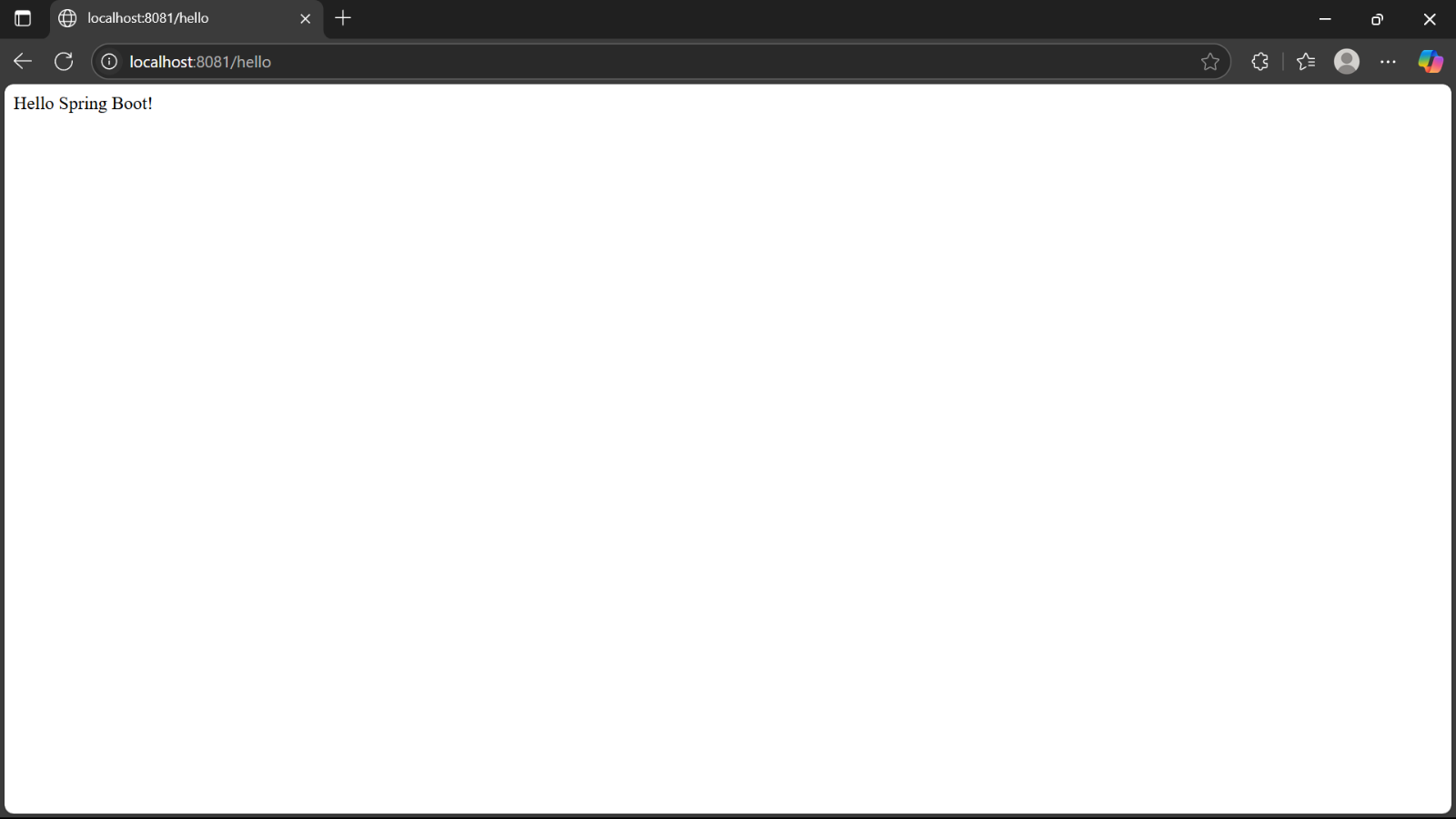
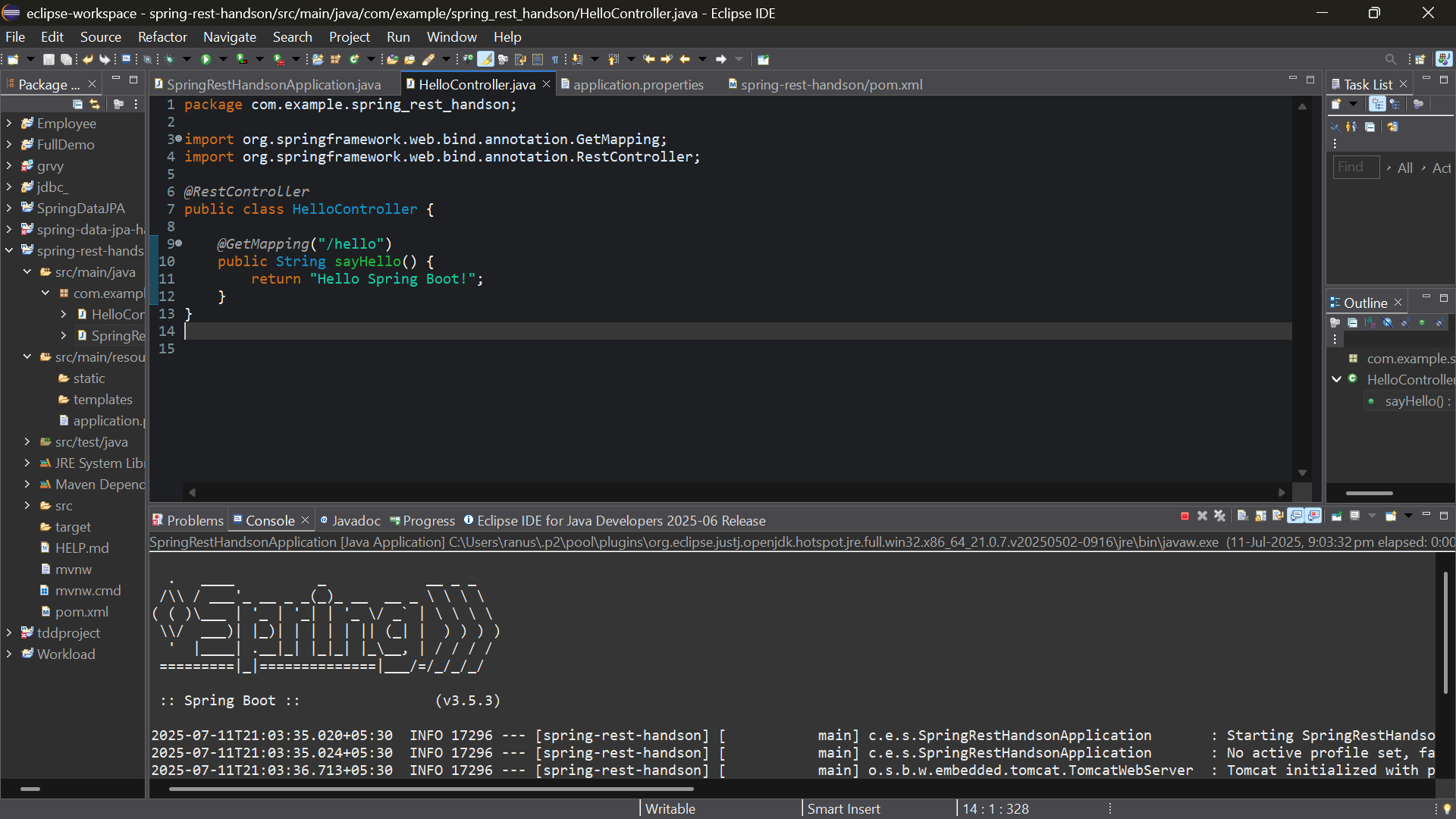
SpringApplication.run(SpringRestHandsonApplication.class, args);

}

}

**Output:**

****

****

**Ex. 2 (Spring Core – Load Country from Spring Configuration XML)**

package com.example.spring\_rest\_handson;

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; }

*@Override* public String toString() {

return code + " – " + name;

}

}

package com.example.spring\_rest\_handson;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ImportResource;

@SpringBootApplication

@ImportResource("classpath:beans.xml")

public class SpringRestHandsonApplication {

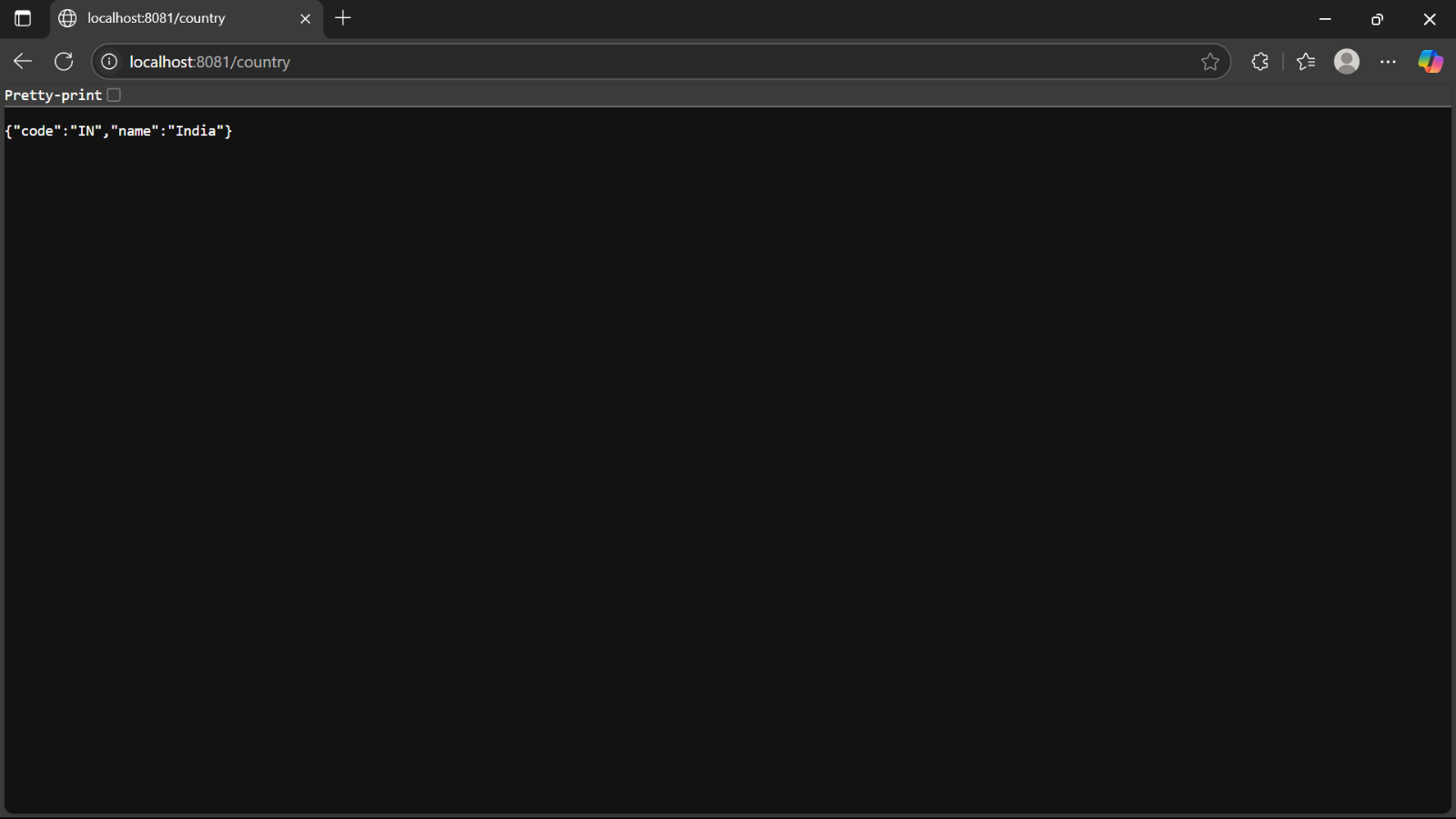
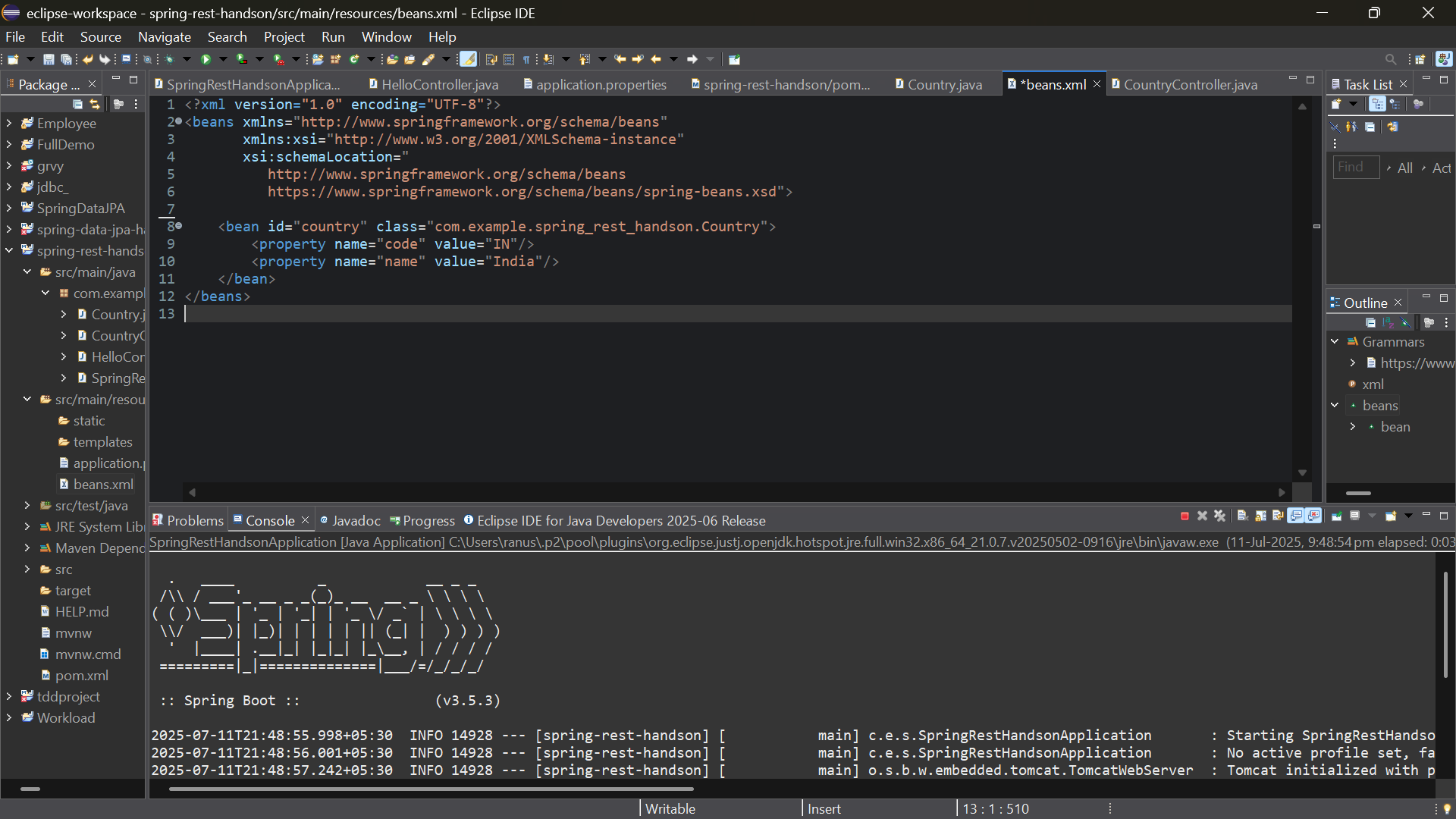
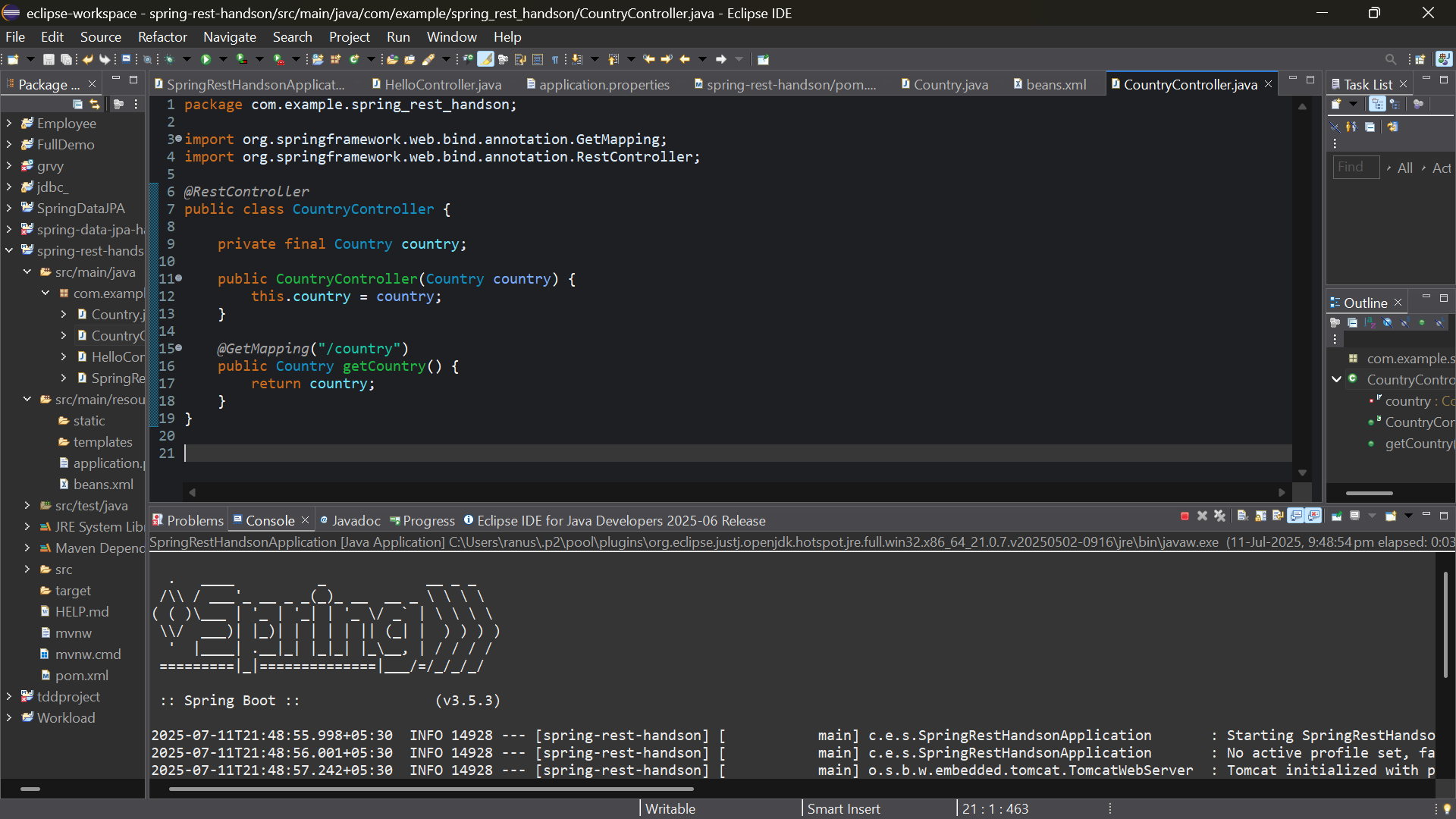
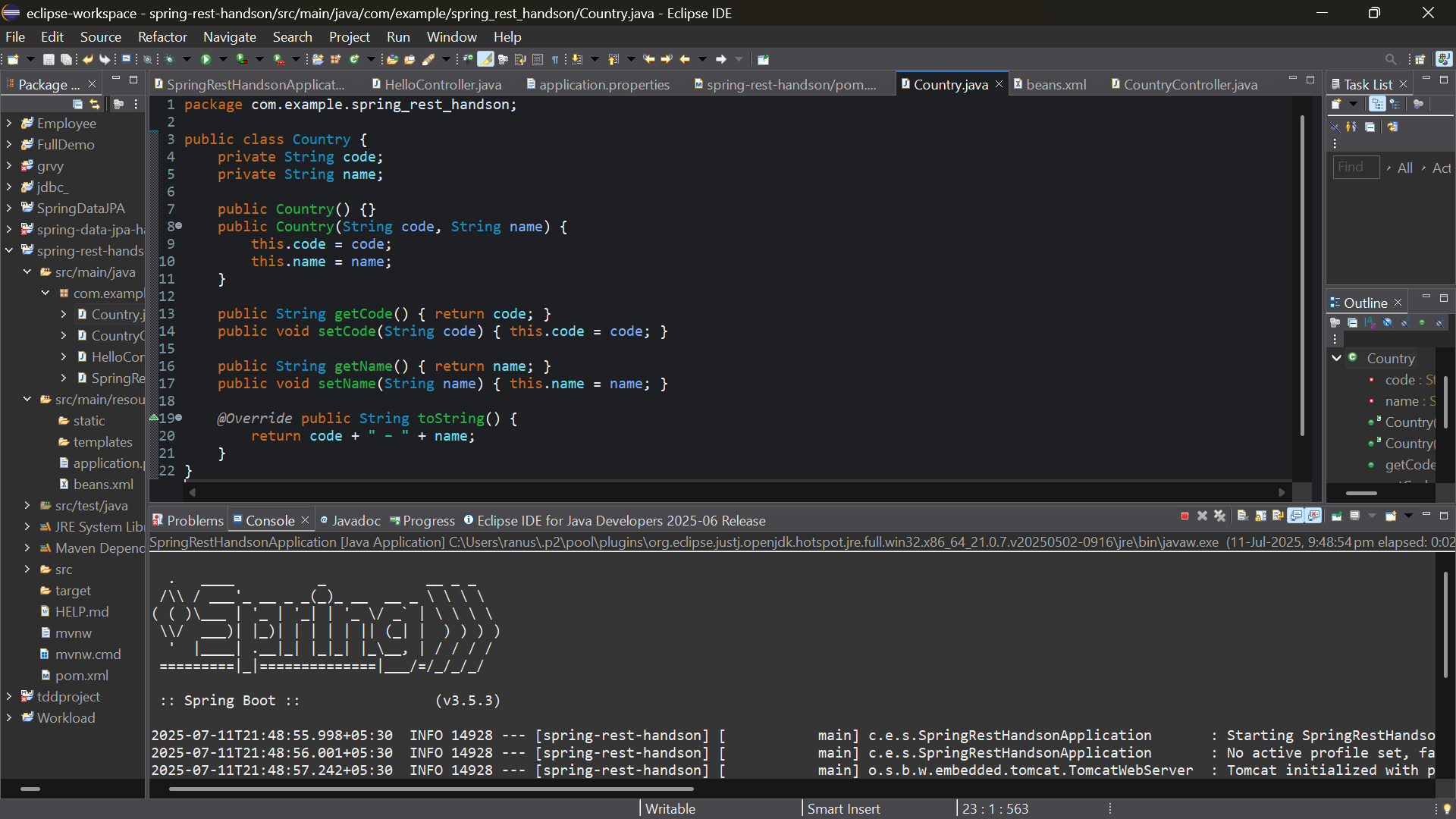
public static void main(String[] args) {

SpringApplication.run(SpringRestHandsonApplication.class, args);

}

}

**Output:**



**Ex. 3 (Hello World RESTful Web Service**)

package com.example.spring\_rest\_handson;

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

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

@RestController

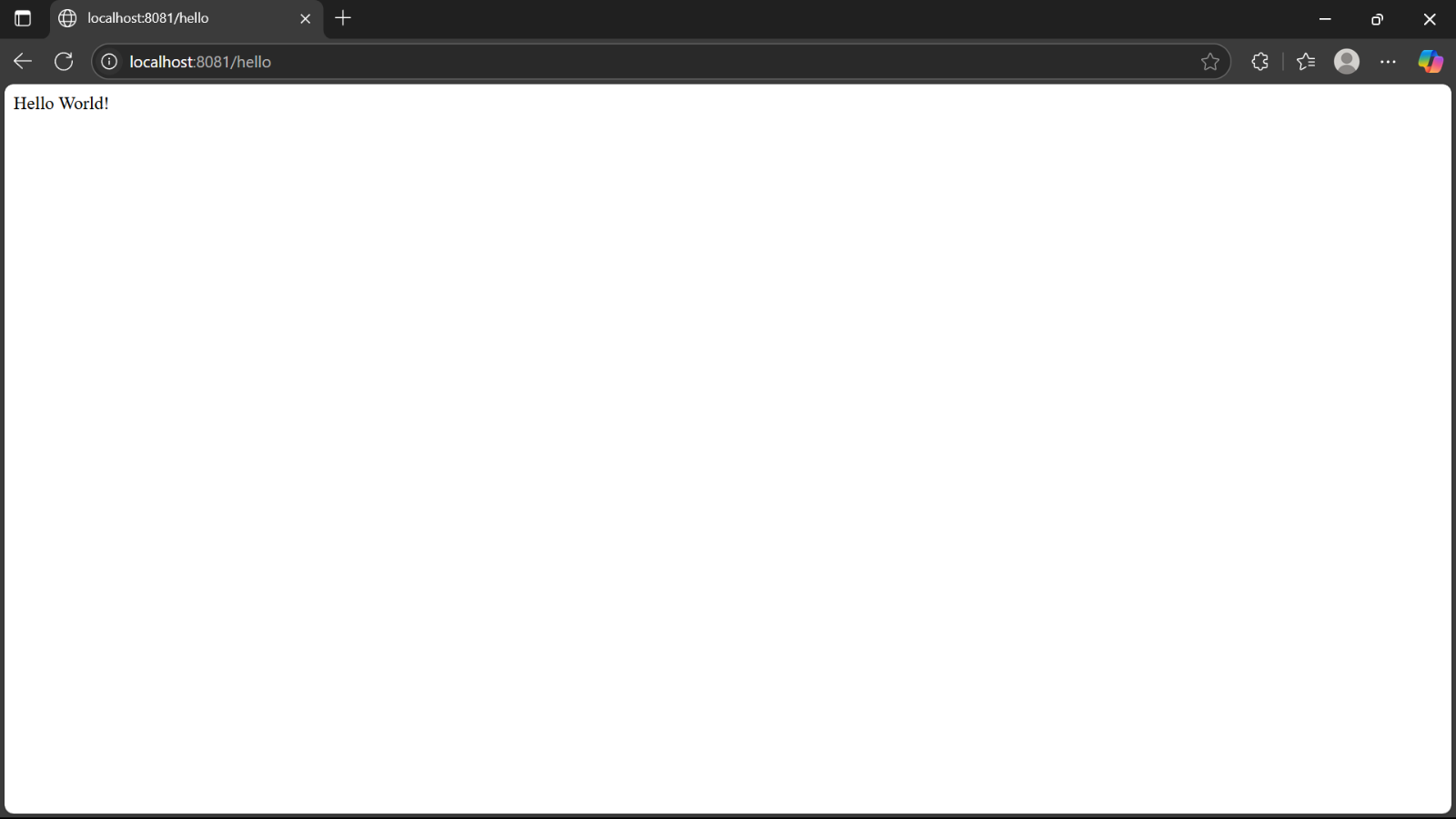
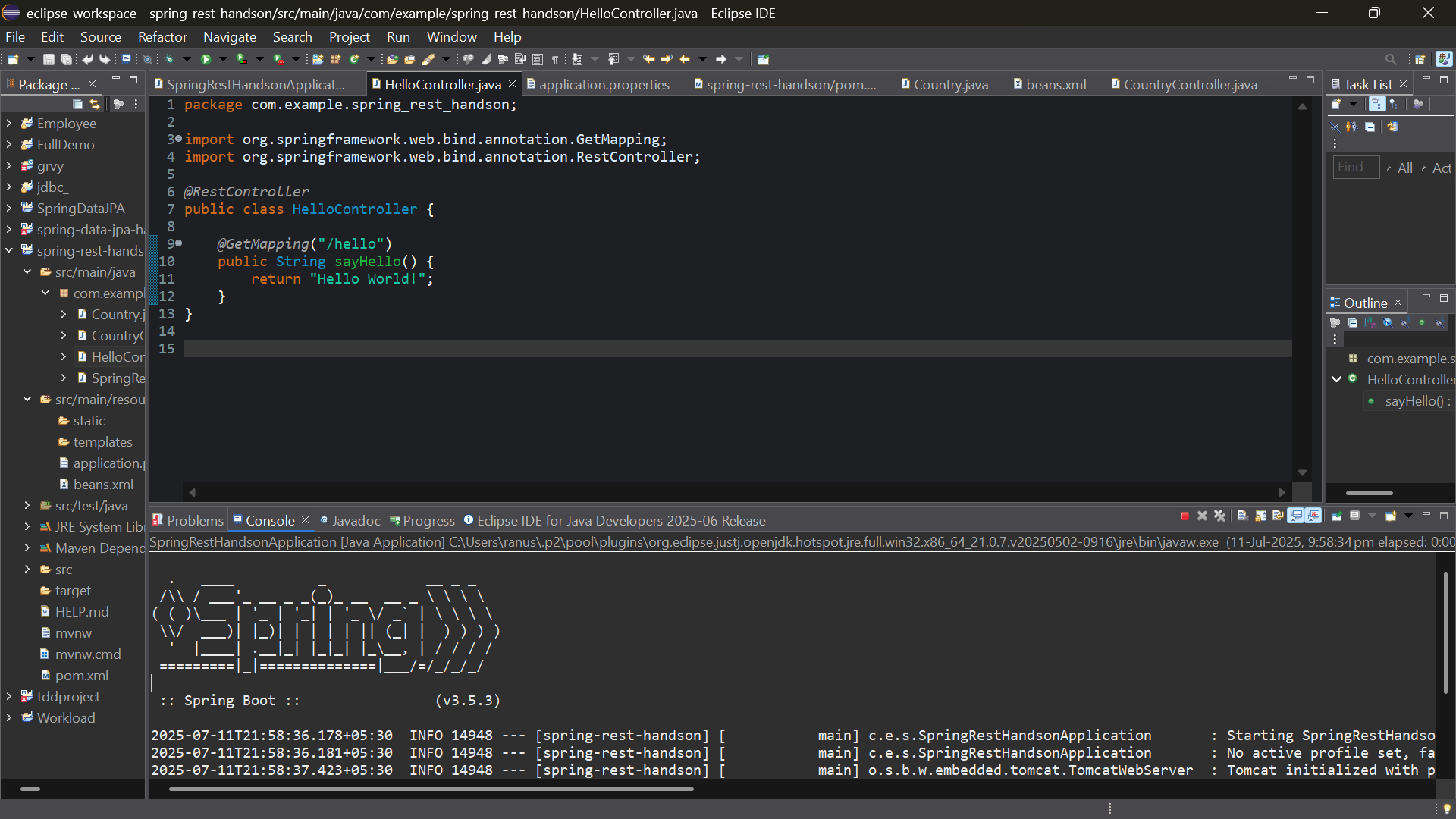
public class HelloController {

@GetMapping("/hello")

public String sayHello() {

return "Hello World!";

}

}**Output:**

**Ex. 4 (REST - Country Web Service**)

@RestController

public class CountryController {

@GetMapping("/country")

public Country getCountry() {

Country country = new Country();

country.setCode("IN");

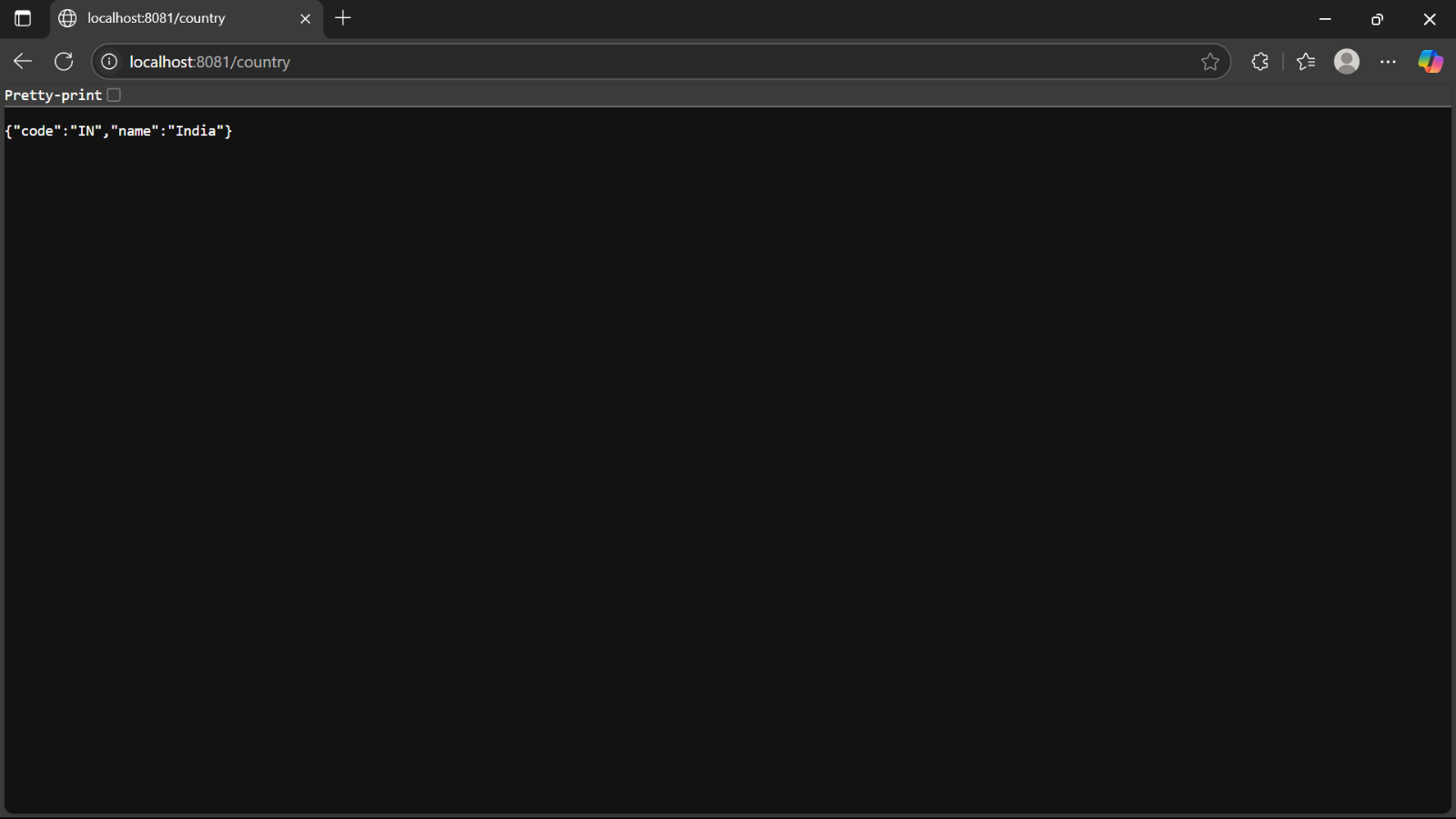
country.setName("India");

return country;

}

}

**Output:**

****

**Ex. 5 (REST - Get country based on country code**)

package com.example.spring\_rest\_handson;

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; }

*@Override* public String toString() {

return code + " – " + name;

}

}

package com.example.spring\_rest\_handson;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

import org.springframework.web.bind.annotation.PathVariable;

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

@RestController

public class CountryController {

private final CountryService service;

public CountryController(CountryService service) {

this.service = service;

}

@GetMapping("/country/{code}")

public ResponseEntity<Country> getCountryByCode(@PathVariable String code) {

Country c = service.getByCode(code);

if (c == null) {

return ResponseEntity.status(HttpStatus.NOT\_FOUND).build();

}

return ResponseEntity.ok(c);

}

}

package com.example.spring\_rest\_handson;

import org.springframework.stereotype.Service;

import java.util.Map;

@Service

public class CountryService {

private final Map<String, Country> countries = Map.of(

"IN", new Country("IN", "India"),

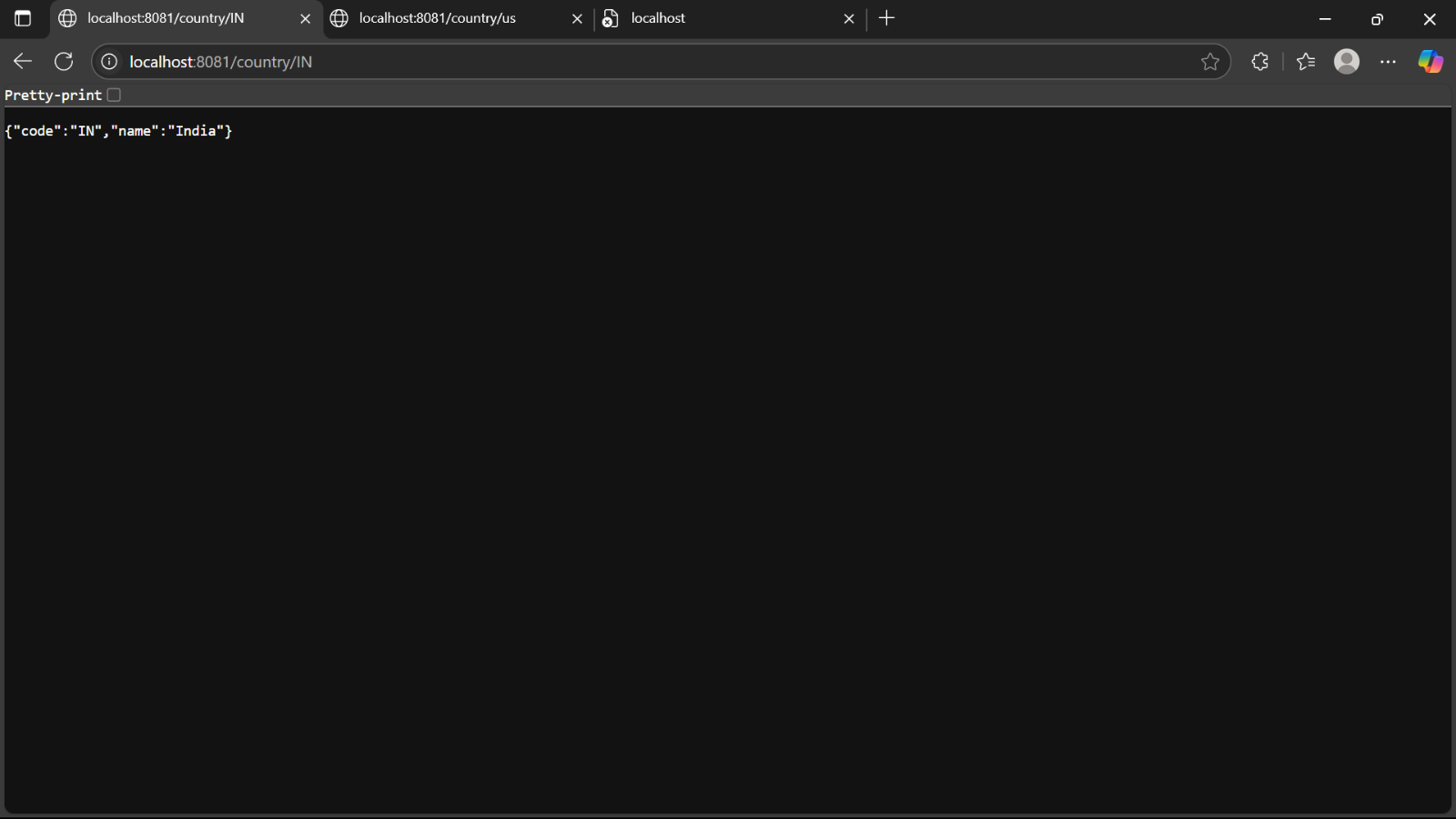
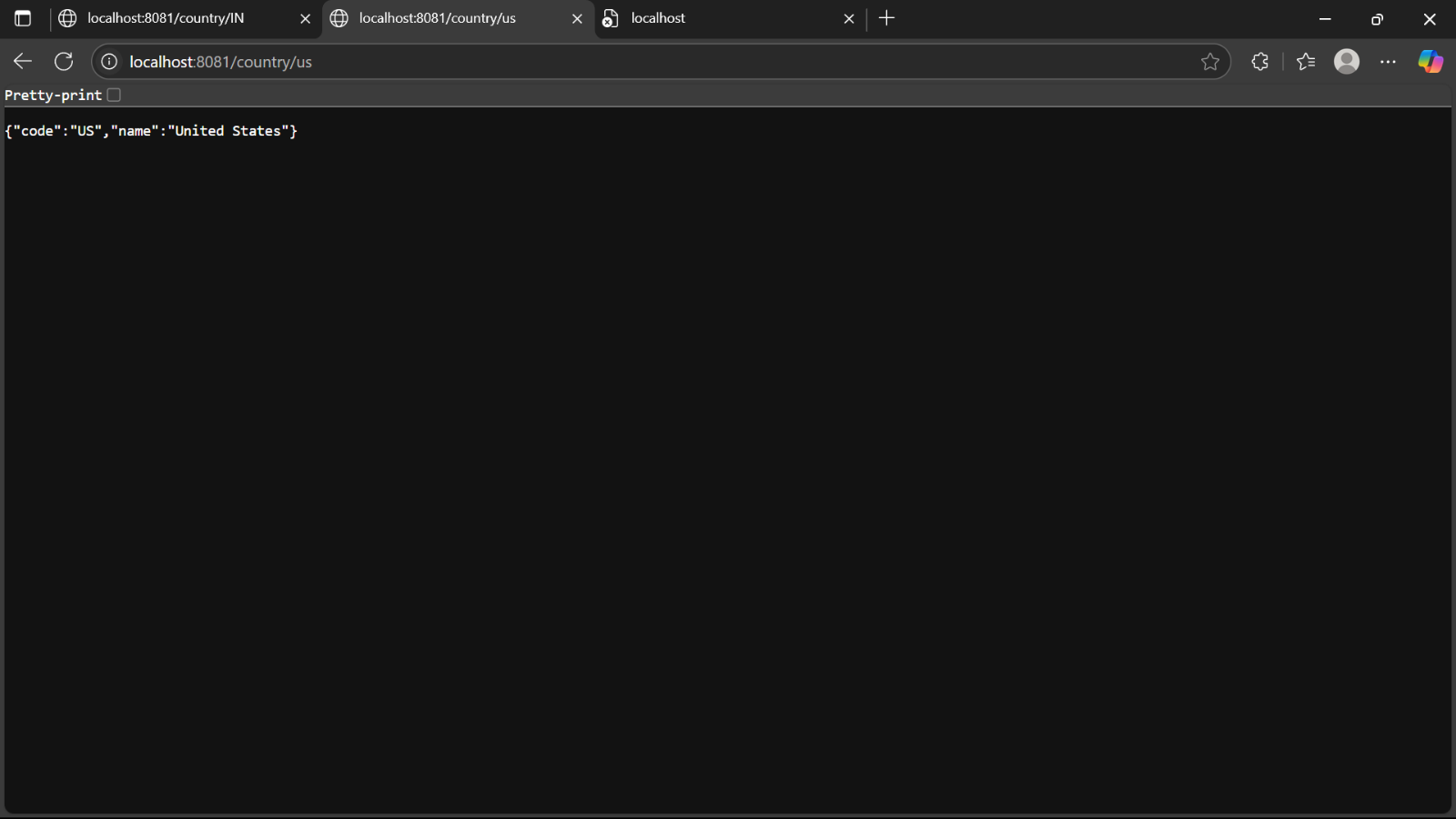
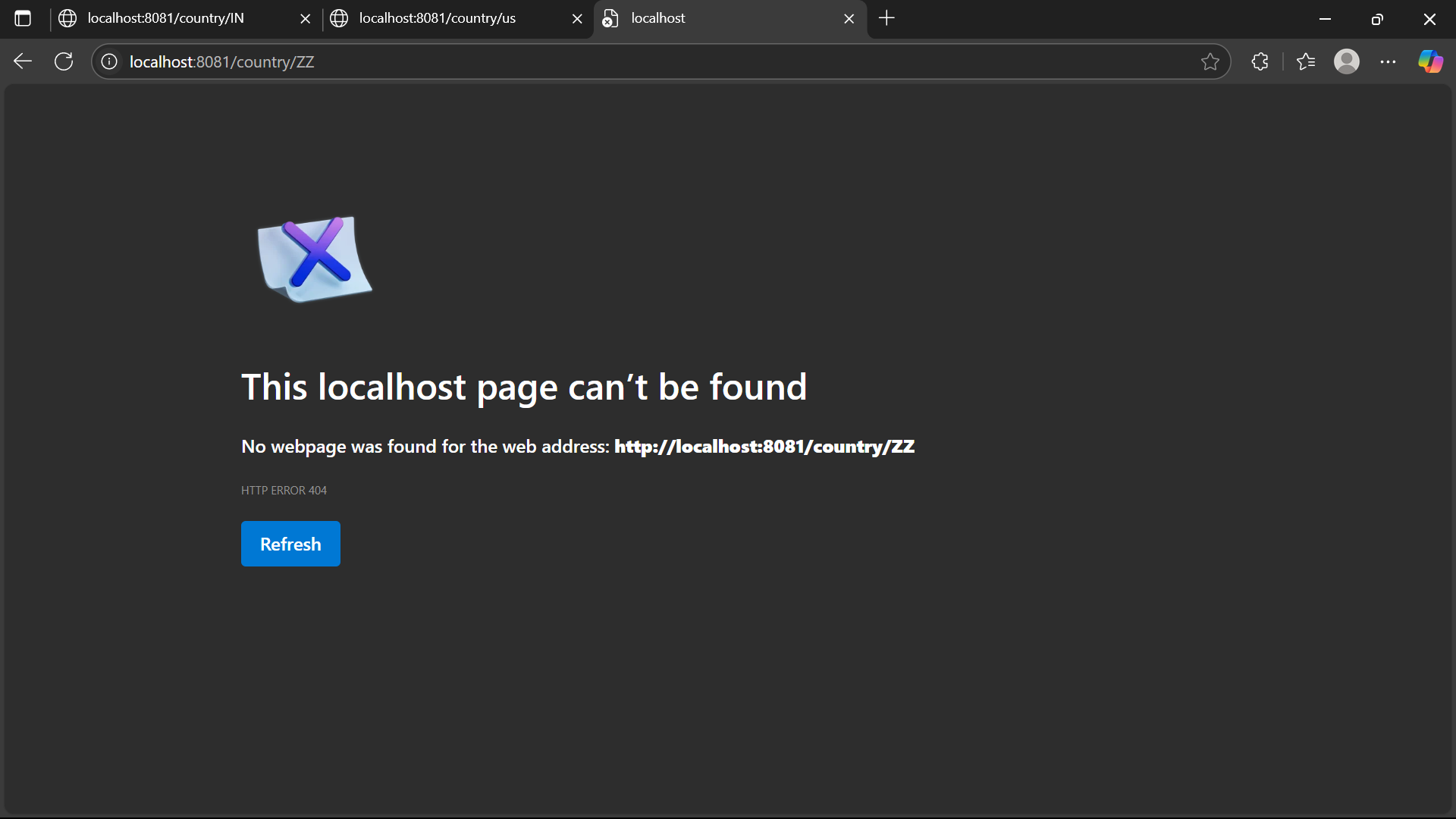
"US", new Country("US", "United States"),

"FR", new Country("FR", "France"));

public Country getByCode(String code) {

return countries.get(code.toUpperCase());

} }

**Output:**

**JWT-handson:**

**Ex. 6 (Create authentication service that returns JWT**)

package com.example.spring\_rest\_handson.JWT;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class Application {

public static void main(String[] args) {

SpringApplication.*run*(Application.class, args);

}

}

package com.example.spring\_rest\_handson.JWT;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

*@RestController*

public class AuthenticationController {

private final JwtUtil jwtUtil;

public AuthenticationController(JwtUtil jwtUtil) {

this.jwtUtil = jwtUtil;

}

*@PostMapping*("/authenticate")

public ResponseEntity<String> authenticate(*@RequestParam* String username,

*@RequestParam* String password) {

if ("admin".equals(username) && "password".equals(password)) {

String token = jwtUtil.generateToken(username);

return ResponseEntity.*ok*(token);

} else {

return ResponseEntity.*status*(401).body("Invalid credentials");

}

}

}

*@RestController*

class HelloController {

*@GetMapping*("/hello")

public String hello() {

return "Hello, secured world!";

}

}

package com.example.spring\_rest\_handson.JWT;

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.web.filter.OncePerRequestFilter;

import java.io.IOException;

import java.util.Collections;

public class JwtFilter extends OncePerRequestFilter {

private final JwtUtil jwtUtil;

public JwtFilter(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);

try {

String username = jwtUtil.validateTokenAndGetUsername(token);

UsernamePasswordAuthenticationToken authToken =

new UsernamePasswordAuthenticationToken(username, null, Collections.*emptyList*());

SecurityContextHolder.*getContext*().setAuthentication(authToken);

} catch (Exception e) {

response.setStatus(HttpServletResponse.***SC\_UNAUTHORIZED***);

return;

}

}

filterChain.doFilter(request, response);

}

}

package com.example.spring\_rest\_handson.JWT;

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 key = Keys.secretKeyFor(SignatureAlgorithm.HS256);

private final long expiration = 1000 \* 60 \* 60 \* 10;

public String generateToken(String username) {

return Jwts.builder()

.setSubject(username)

.setIssuer("demo-app")

.setIssuedAt(new Date())

.setExpiration(new Date(System.currentTimeMillis() + expiration))

.signWith(key)

.compact();

}

public boolean validateToken(String token) {

try {

Jwts.parserBuilder().setSigningKey(key).build().parseClaimsJws(token);

return true;

} catch (JwtException e) {

return false;

}

}

public String extractUsername(String token) {

return Jwts.parserBuilder()

.setSigningKey(key)

.build()

.parseClaimsJws(token)

.getBody()

.getSubject();

}

public String validateTokenAndGetUsername(String token) {

try {

Claims claims = Jwts.parserBuilder()

.setSigningKey(key)

.build()

.parseClaimsJws(token)

.getBody();

return claims.getSubject();

} catch (JwtException e) {

return null;

}

}

}

package com.example.spring\_rest\_handson.JWT;

import org.springframework.beans.factory.annotation.Autowired;

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.web.SecurityFilterChain;

import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;

*@Configuration*

public class SecurityConfig {

*@Autowired*

private JwtUtil jwtUtil;

*@Bean*

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http

.csrf(csrf -> csrf.disable())

.authorizeHttpRequests(auth -> auth

.requestMatchers("/authenticate").permitAll()

.anyRequest().authenticated()

)

.addFilterBefore(new JwtFilter(jwtUtil), UsernamePasswordAuthenticationFilter.class)

.httpBasic(httpBasic -> httpBasic.disable())

.formLogin(form -> form.disable());

return http.build(); }

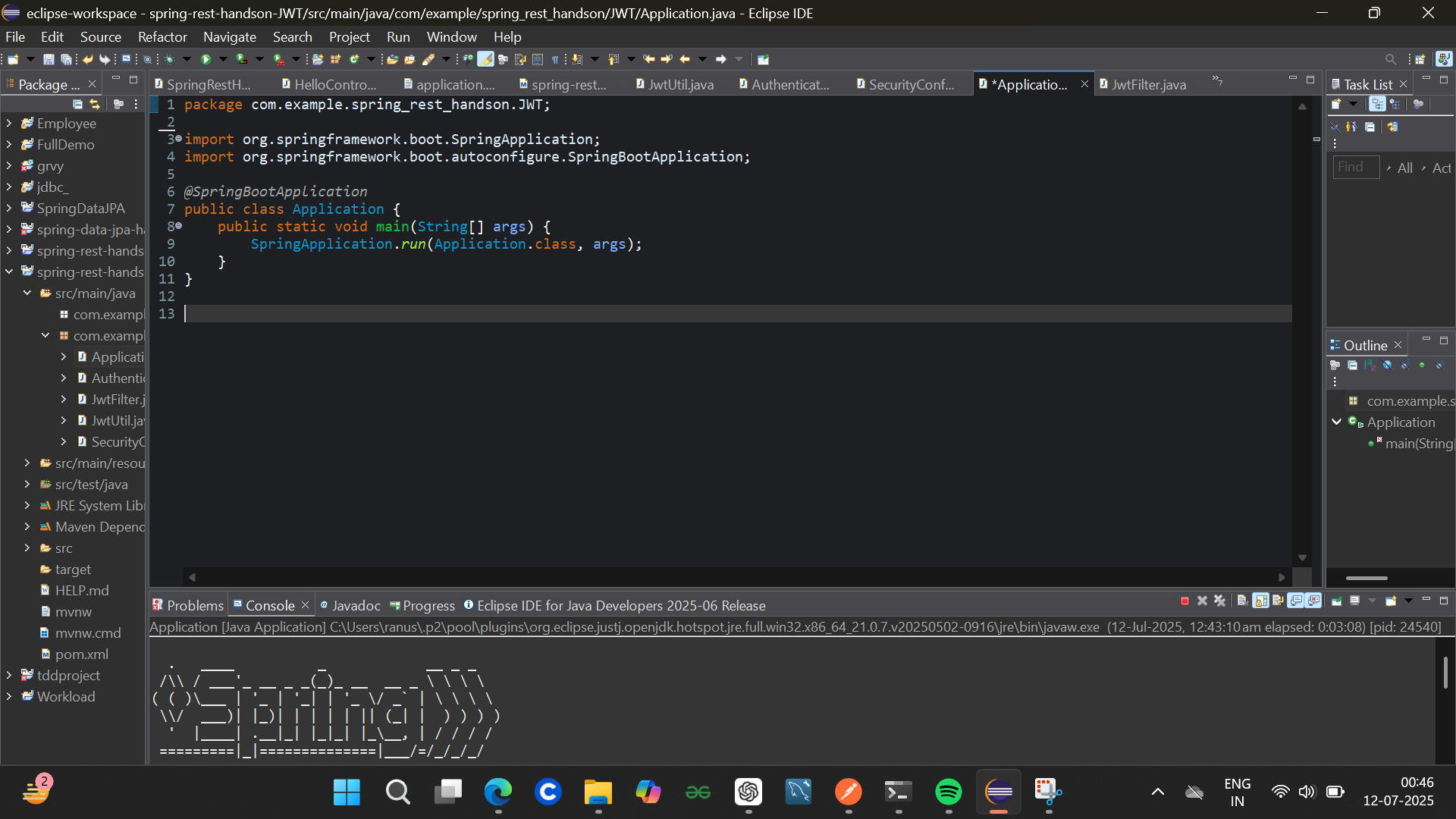
*@Bean*

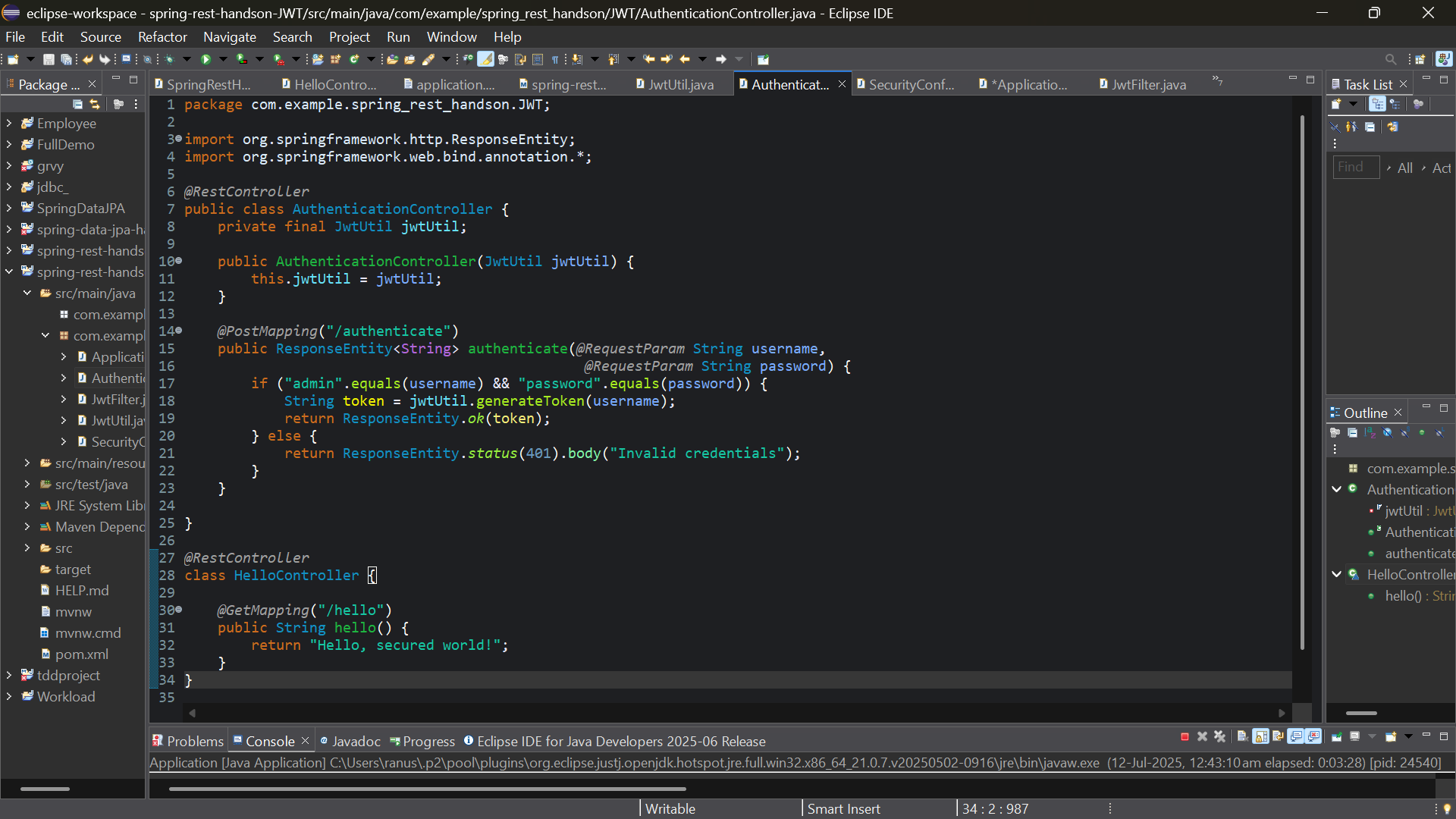
public AuthenticationManager authManager(AuthenticationConfiguration config) throws Exception {

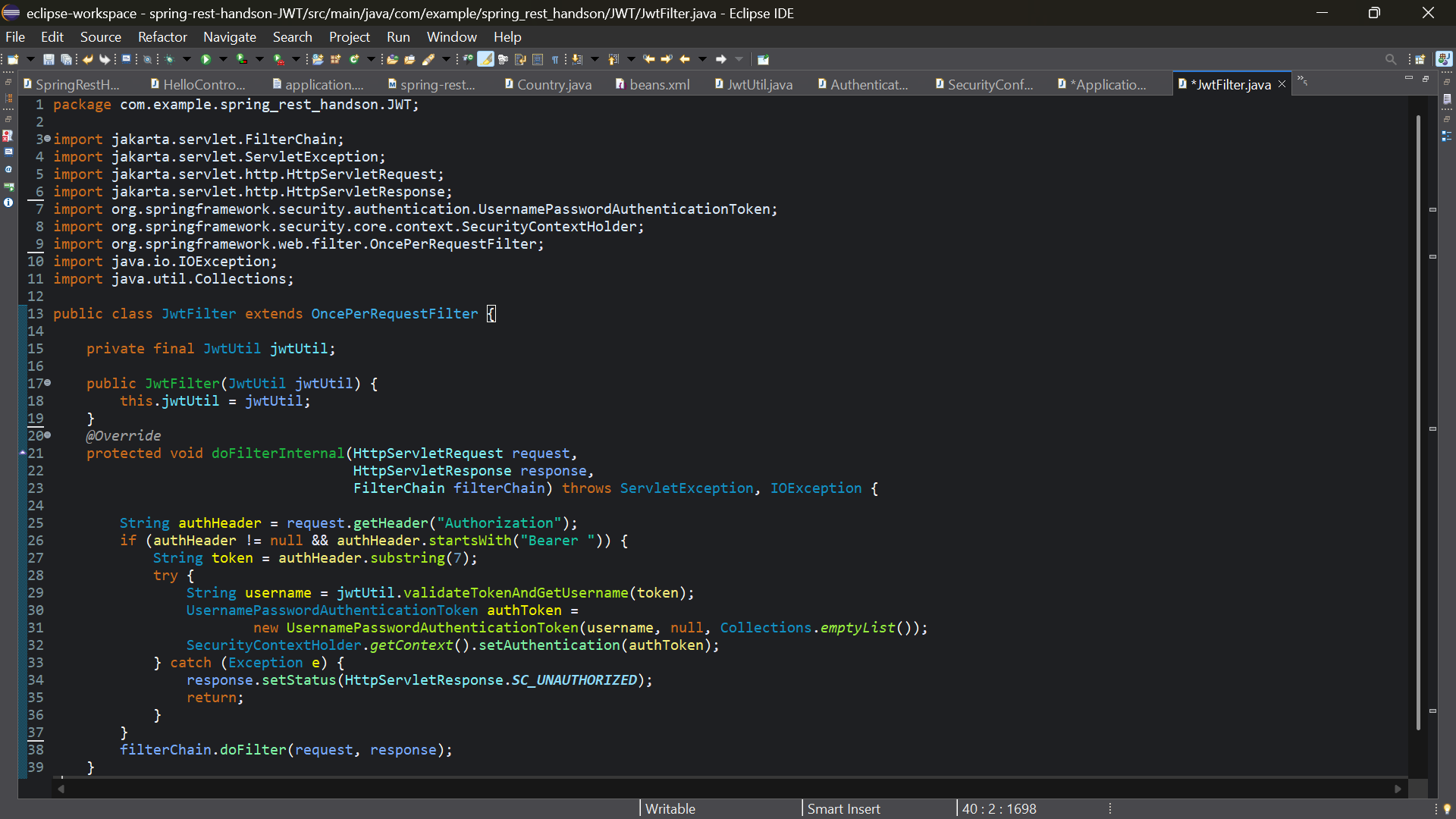
return config.getAuthenticationManager();

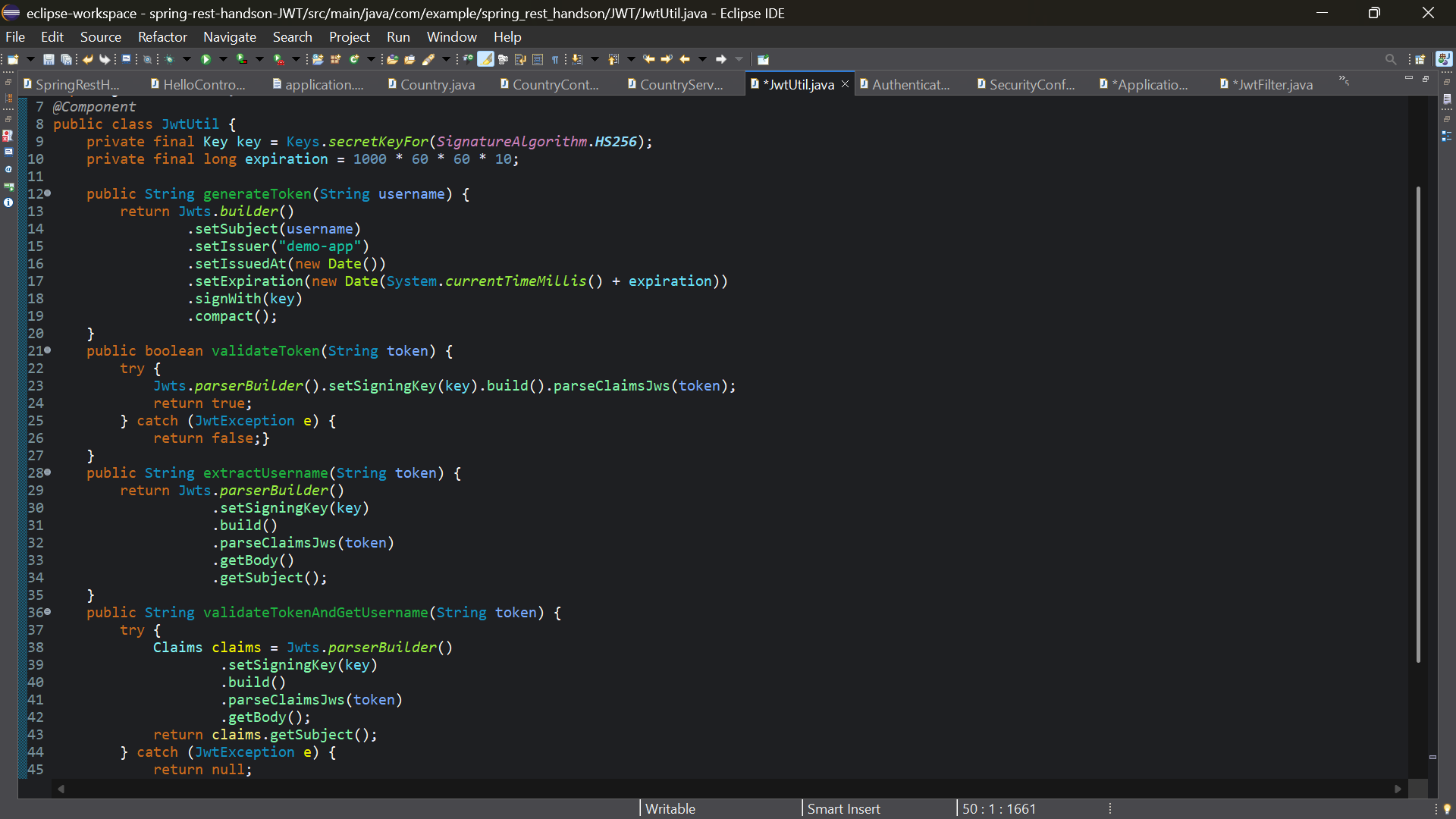
}

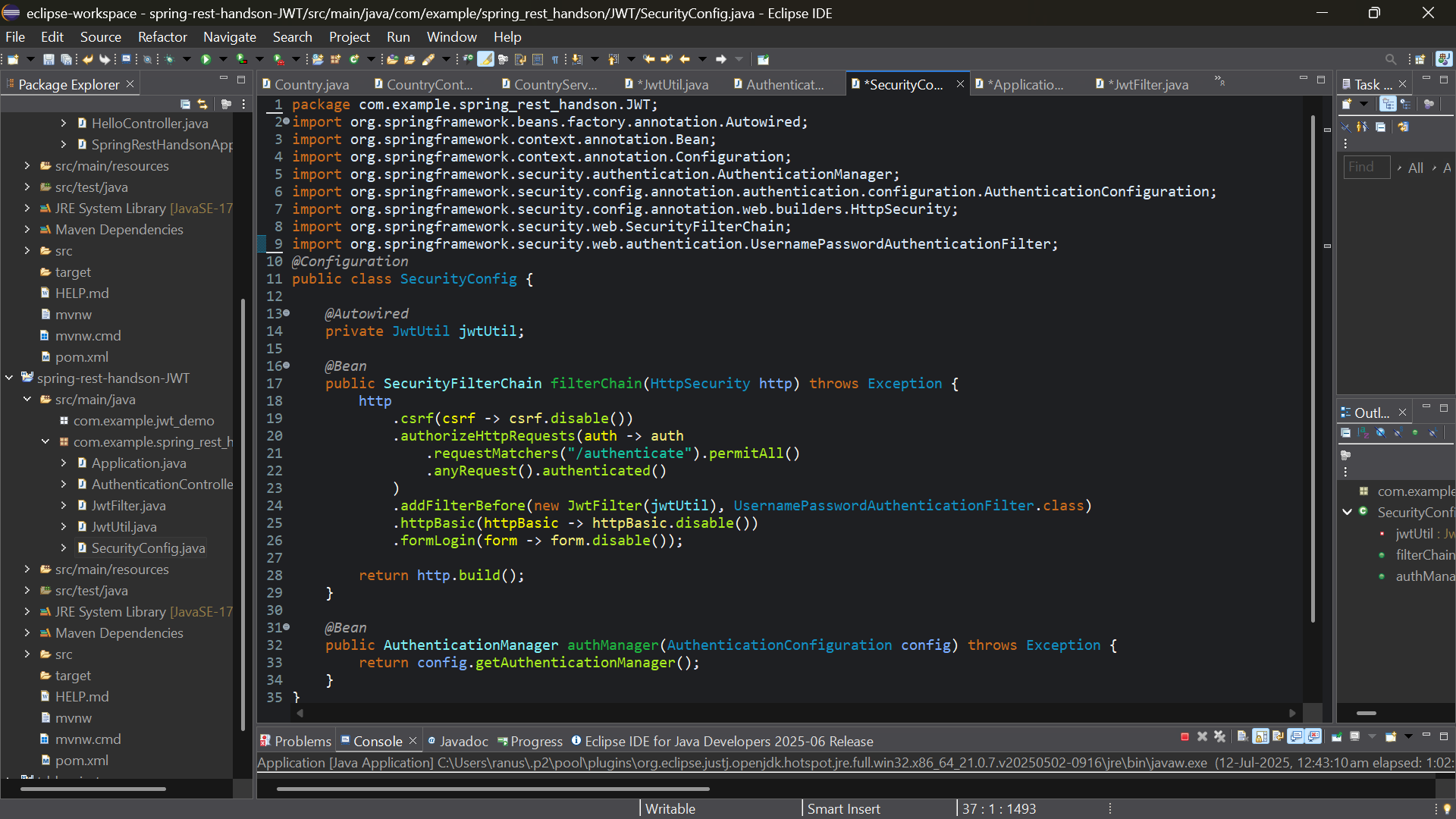
}

****

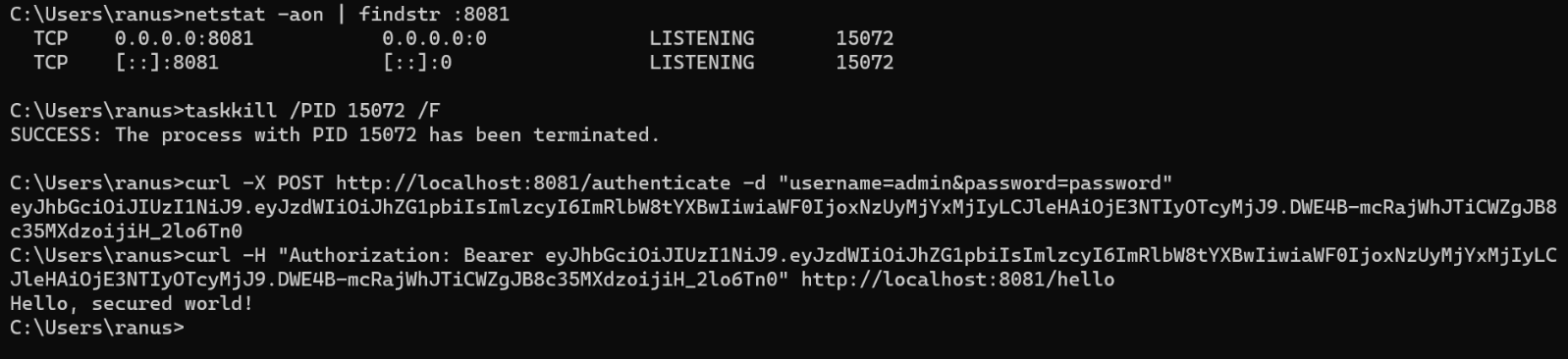
****

****





**Output:**

****