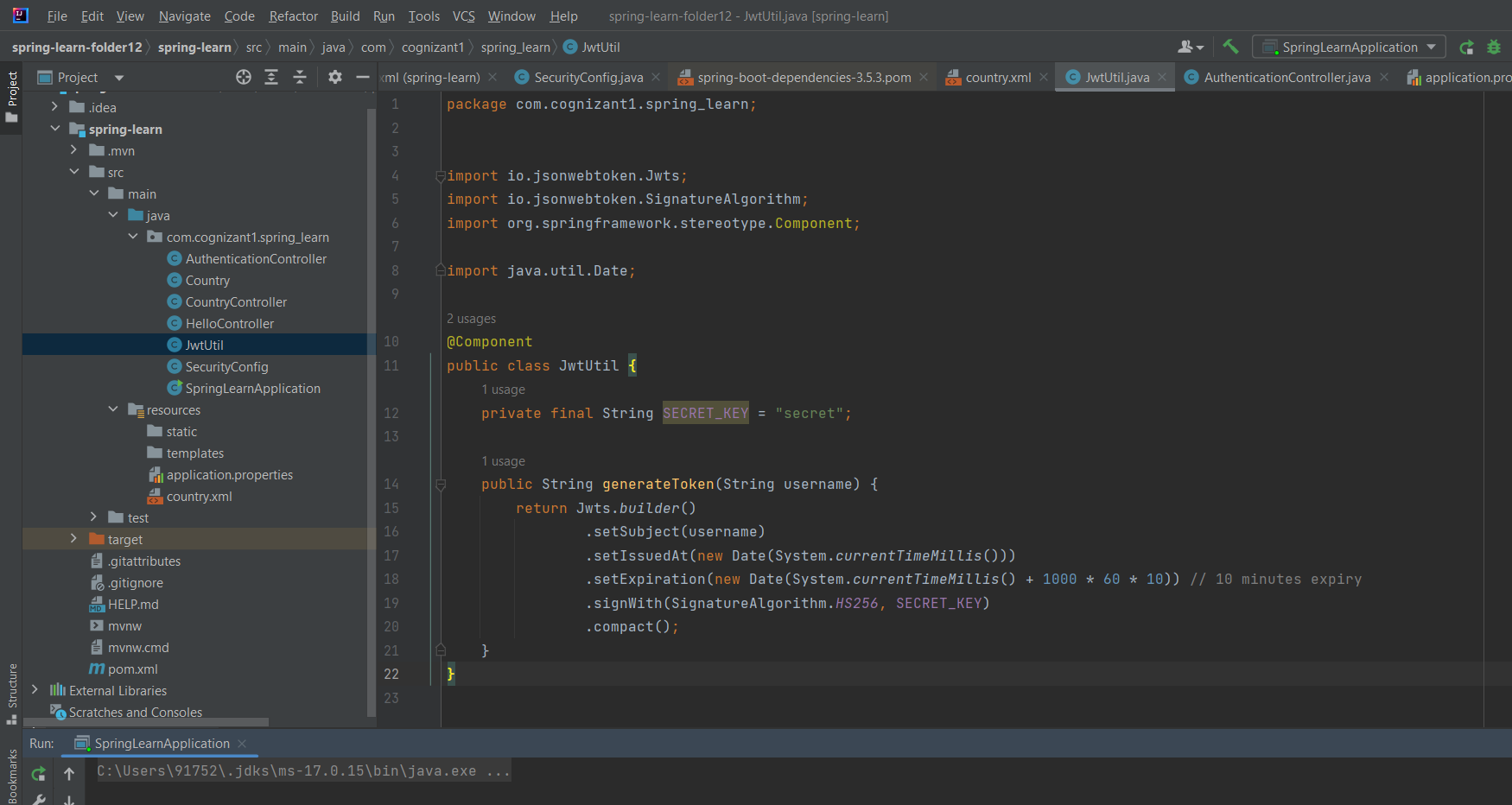
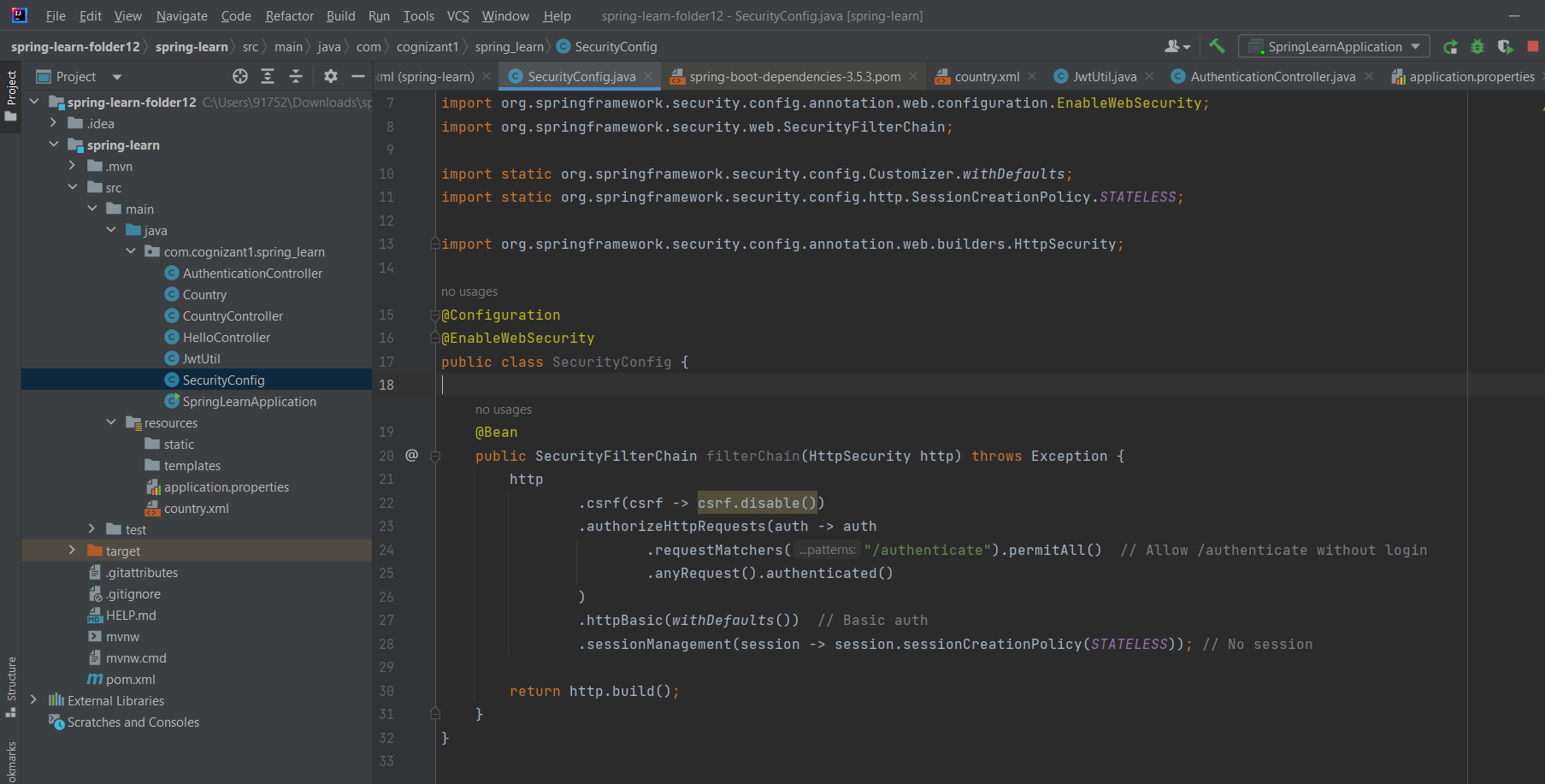
**Task 5- Create authentication service that returns JWT**

1. JwtUtilController.java
2. package com.cognizant1.spring\_learn;  
     
     
   import io.jsonwebtoken.Jwts;  
   import io.jsonwebtoken.SignatureAlgorithm;  
   import org.springframework.stereotype.Component;  
     
   import java.util.Date;  
     
   @Component  
   public class JwtUtil {  
    private final String SECRET\_KEY = "secret";  
     
    public String generateToken(String username) {  
    return Jwts.*builder*()  
    .setSubject(username)  
    .setIssuedAt(new Date(System.*currentTimeMillis*()))  
    .setExpiration(new Date(System.*currentTimeMillis*() + 1000 \* 60 \* 10)) // 10 minutes expiry  
    .signWith(SignatureAlgorithm.*HS256*, SECRET\_KEY)  
    .compact();  
    }  
   }



2-SecurityConfig

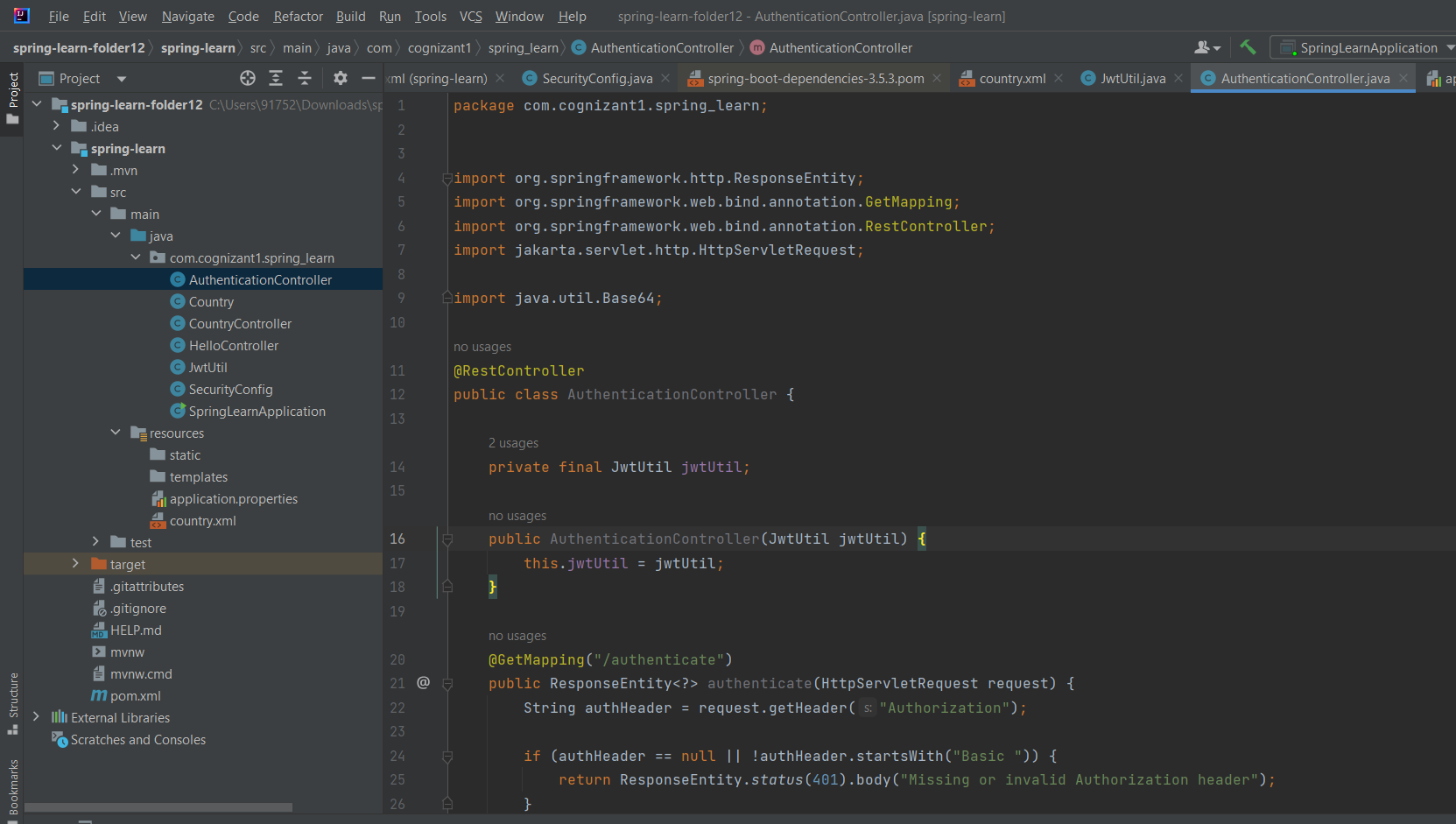
package com.cognizant1.spring\_learn;  
  
  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.security.config.Customizer;  
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;  
import org.springframework.security.web.SecurityFilterChain;  
  
import static org.springframework.security.config.Customizer.*withDefaults*;  
import static org.springframework.security.config.http.SessionCreationPolicy.*STATELESS*;  
  
import org.springframework.security.config.annotation.web.builders.HttpSecurity;  
  
@Configuration  
@EnableWebSecurity  
public class SecurityConfig {  
  
 @Bean  
 public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {  
 http  
 .csrf(csrf -> csrf.disable())  
 .authorizeHttpRequests(auth -> auth  
 .requestMatchers("/authenticate").permitAll() // Allow /authenticate without login  
 .anyRequest().authenticated()  
 )  
 .httpBasic(*withDefaults*()) // Basic auth  
 .sessionManagement(session -> session.sessionCreationPolicy(*STATELESS*)); // No session  
  
 return http.build();  
 }  
}



1. SpringLearnApplication
2. package com.cognizant1.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);  
    }  
   }

4-AuthenticationController

package com.cognizant1.spring\_learn;  
  
  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RestController;  
import jakarta.servlet.http.HttpServletRequest;  
  
import java.util.Base64;  
  
@RestController  
public class AuthenticationController {  
  
 private final JwtUtil jwtUtil;  
  
 public AuthenticationController(JwtUtil jwtUtil) {  
 this.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[] decodedCreds = Base64.*getDecoder*().decode(base64Credentials);  
 String credentials = new String(decodedCreds); // user:pwd  
  
 String[] userAndPass = credentials.split(":", 2);  
 String username = userAndPass[0];  
 String password = userAndPass[1];  
  
 // hardcoded user validation  
 if ("user".equals(username) && "pwd".equals(password)) {  
 String token = jwtUtil.generateToken(username);  
 return ResponseEntity.*ok*("{\"token\":\"" + token + "\"}");  
 } else {  
 return ResponseEntity.*status*(403).body("Invalid credentials");  
 }  
 }  
}



1. pom.xml
2. <?xml version="1.0" encoding="UTF-8"?>  
   <project xmlns="http://maven.apache.org/POM/4.0.0"  
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0  
    https://maven.apache.org/xsd/maven-4.0.0.xsd">  
     
    <modelVersion>4.0.0</modelVersion>  
     
    <parent>  
    <groupId>org.springframework.boot</groupId>  
    <artifactId>spring-boot-starter-parent</artifactId>  
    <version>3.5.3</version>  
    <relativePath/> <!-- lookup parent from repository -->  
    </parent>  
     
    <groupId>com.cognizant1</groupId>  
    <artifactId>spring-learn</artifactId>  
    <version>0.0.1-SNAPSHOT</version>  
    <name>spring-learn</name>  
    <description>Demo project for Spring Boot</description>  
     
    <properties>  
    <java.version>17</java.version>  
    </properties>  
     
    <dependencies>  
    <!-- Spring Web -->  
    <dependency>  
    <groupId>org.springframework.boot</groupId>  
    <artifactId>spring-boot-starter-web</artifactId>  
    </dependency>  
     
    <!-- Spring Security -->  
    <dependency>  
    <groupId>org.springframework.boot</groupId>  
    <artifactId>spring-boot-starter-security</artifactId>  
    </dependency>  
     
    <!-- JWT (JSON Web Token) -->  
    <dependency>  
    <groupId>io.jsonwebtoken</groupId>  
    <artifactId>jjwt</artifactId>  
    <version>0.9.1</version>  
    </dependency>  
     
    <!-- Jakarta Servlet API -->  
    <dependency>  
    <groupId>jakarta.servlet</groupId>  
    <artifactId>jakarta.servlet-api</artifactId>  
    <version>5.0.0</version>  
    <scope>provided</scope>  
    </dependency>  
    </dependencies>  
     
    <build>  
    <plugins>  
    <plugin>  
    <groupId>org.springframework.boot</groupId>  
    <artifactId>spring-boot-maven-plugin</artifactId>  
    </plugin>  
    </plugins>  
    </build>  
     
   </project>

6-Output

