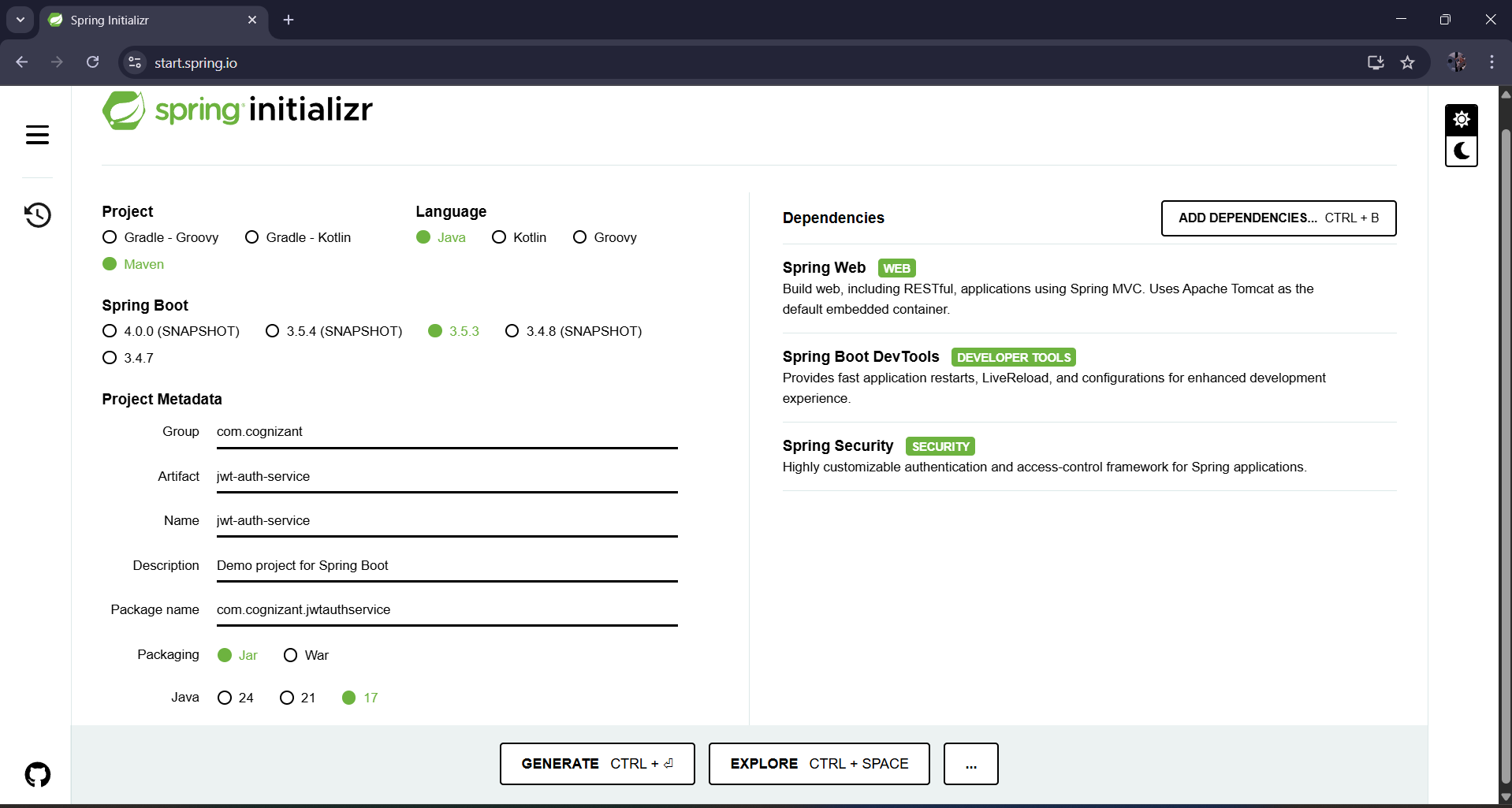
**Create authentication service that returns JWT**

* **Step 1: Start spring initializr**



* Do the same steps and then generate it .
* Unzip the folder .
* Open it in itntelliJ.
* **Step 2: Structure Your JWT Auth Classes Inside com.cognizant.jwtauthservice**
* **It should look like this:**

A screenshot of a computer program

AI-generated content may be incorrect.

* **com.cognizant.jwtauthservice.controller.AuthenticationController**

package com.cognizant.jwtauthservice.controller;  
  
import com.cognizant.jwtauthservice.util.JwtUtil;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RestController;  
  
import jakarta.servlet.http.HttpServletRequest;  
import java.util.Base64;  
import java.util.HashMap;  
import java.util.Map;  
  
@RestController  
public class AuthenticationController {  
  
 @GetMapping("/authenticate")  
 public Map<String, String> authenticate(HttpServletRequest request) {  
 String authHeader = request.getHeader("Authorization");  
  
 if (authHeader == null || !authHeader.startsWith("Basic ")) {  
 throw new RuntimeException("Missing or invalid Authorization header.");  
 }  
  
 String base64Credentials = authHeader.substring("Basic ".length()).trim();  
 byte[] decoded = Base64.*getDecoder*().decode(base64Credentials);  
 String[] credentials = new String(decoded).split(":", 2);  
  
 String username = credentials[0];  
 String password = credentials[1];  
  
 if (!username.equals("user") || !password.equals("pwd")) {  
 throw new RuntimeException("Invalid credentials");  
 }  
  
 String token = JwtUtil.*generateToken*(username); // You'll define JwtUtil  
 Map<String, String> response = new HashMap<>();  
 response.put("token", token);  
 return response;  
 }  
}

* **com.cognizant.jwtauthservice.config.SecurityConfig**

package com.cognizant.jwtauthservice.config;  
  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.security.config.annotation.web.builders.HttpSecurity;  
import org.springframework.security.core.userdetails.User;  
import org.springframework.security.core.userdetails.UserDetails;  
import org.springframework.security.core.userdetails.UserDetailsService;  
import org.springframework.security.provisioning.InMemoryUserDetailsManager;  
import org.springframework.security.web.SecurityFilterChain;  
import org.springframework.security.config.Customizer;  
  
@Configuration  
public class SecurityConfig {  
  
 @Bean  
 public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {  
 http  
 .authorizeHttpRequests(auth -> auth  
 .requestMatchers("/authenticate").permitAll()  
 .anyRequest().authenticated()  
 )  
 .httpBasic(Customizer.*withDefaults*()); // ✅ Updated  
  
 return http.build();  
 }  
  
 @Bean  
 public UserDetailsService users() {  
 UserDetails user = User  
 .*withUsername*("user")  
 .password("{noop}pwd")  
 .roles("USER")  
 .build();  
 return new InMemoryUserDetailsManager(user);  
 }  
}

* **com.cognizant.jwtauthservice.util.JwtUtil**

package com.cognizant.jwtauthservice.util;  
  
import io.jsonwebtoken.Jwts;  
import io.jsonwebtoken.SignatureAlgorithm;  
  
import java.util.Date;  
  
public class JwtUtil {  
  
 private static final String *SECRET\_KEY* = "mySecretKey";  
  
 public static String generateToken(String username) {  
 return Jwts.*builder*()  
 .setSubject(username)  
 .setIssuedAt(new Date())  
 .setExpiration(new Date(System.*currentTimeMillis*() + 60 \* 60 \* 1000)) // 1 hour  
 .signWith(SignatureAlgorithm.*HS256*, *SECRET\_KEY*)  
 .compact();  
 }  
}

**OUTPUT OF POSTMAN:**

**Test your JWT auth endpoint like this:**

* **GET http://localhost:8083/authenticate**
* **In Postman:**
  + **Auth type: Basic Auth**
  + **Username: user**
  + **Password: pwd**
* **Click Send**

