package com.example.jwt\_auth;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
@SpringBootApplication  
public class JwtAuthApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(JwtAuthApplication.class, args);  
 }  
}

<?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.example</groupId>  
 <artifactId>jwt-auth</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>jwt-auth</name>  
 <description>Demo project for Spring Boot</description>  
 <url/>  
 <licenses>  
 <license/>  
 </licenses>  
 <developers>  
 <developer/>  
 </developers>  
 <scm>  
 <connection/>  
 <developerConnection/>  
 <tag/>  
 <url/>  
 </scm>  
 <properties>  
 <java.version>17</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-security</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>io.jsonwebtoken</groupId>  
 <artifactId>jjwt-api</artifactId>  
 <version>0.11.5</version>  
 </dependency>  
 <dependency>  
 <groupId>io.jsonwebtoken</groupId>  
 <artifactId>jjwt-impl</artifactId>  
 <version>0.11.5</version>  
 <scope>runtime</scope>  
 </dependency>  
 <dependency>  
 <groupId>io.jsonwebtoken</groupId>  
 <artifactId>jjwt-jackson</artifactId>  
 <version>0.11.5</version>  
 <scope>runtime</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.security</groupId>  
 <artifactId>spring-security-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
</project>

package com.example.jwt\_auth.controller;  
  
import com.example.jwt\_auth.util.JwtUtil;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.\*;  
import java.nio.charset.StandardCharsets;  
import java.util.Base64;  
import java.util.Map;  
@RestController  
public class AuthenticationController {  
 @GetMapping("/authenticate")  
 public ResponseEntity<Map<String, String>> authenticate(@RequestHeader("Authorization") String authHeader) {  
 if (authHeader != null && authHeader.startsWith("Basic ")) {  
 String base64Credentials = authHeader.substring("Basic ".length());  
 byte[] decodedBytes = Base64.*getDecoder*().decode(base64Credentials);  
 String credentials = new String(decodedBytes, StandardCharsets.*UTF\_8*);  
 String[] values = credentials.split(":", 2);  
 String username = values[0];  
 String password = values[1];  
 if ("user".equals(username) && "pwd".equals(password)) {  
 String token = JwtUtil.*generateToken*(username);  
 return ResponseEntity.*ok*(Map.*of*("token", token));  
 }  
 }  
 return ResponseEntity.*status*(401).build();  
 }  
}

package com.example.jwt\_auth.util;  
import io.jsonwebtoken.Jwts;  
import io.jsonwebtoken.SignatureAlgorithm;  
import io.jsonwebtoken.security.Keys  
import java.security.Key;  
import java.util.Date;  
public class JwtUtil {  
 private static final Key *SECRET\_KEY* = Keys.*secretKeyFor*(SignatureAlgorithm.*HS256*); // Auto-generates key  
 public static String generateToken(String username) {  
 long now = System.*currentTimeMillis*();  
 return Jwts.*builder*()  
 .setSubject(username)  
 .setIssuedAt(new Date(now))  
 .setExpiration(new Date(now + 1000 \* 60 \* 10)) // 10 minutes  
 .signWith(*SECRET\_KEY*)  
 .compact();  
 }  
}

spring.application.name=jwt-auth  
server.port=8090

package com.example.jwt\_auth.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.web.SecurityFilterChain;  
import org.springframework.security.core.userdetails.User;  
import org.springframework.security.core.userdetails.UserDetails;  
import org.springframework.security.provisioning.InMemoryUserDetailsManager;  
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;  
import org.springframework.security.crypto.password.PasswordEncoder;  
@Configuration  
public class SecurityConfig {  
 @Bean  
 public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {  
 http  
 .csrf().disable()  
 .authorizeHttpRequests()  
 .requestMatchers("/authenticate").permitAll()  
 .anyRequest().authenticated()  
 .and()  
 .httpBasic(); // Enables Basic Auth (user:pwd)  
 return http.build();  
 }  
 @Bean  
 public InMemoryUserDetailsManager userDetailsService() {  
 UserDetails user = User  
 .*withUsername*("user")  
 .password(passwordEncoder().encode("pwd"))  
 .roles("USER")  
 .build();  
 return new InMemoryUserDetailsManager(user);  
 }  
 @Bean  
 public PasswordEncoder passwordEncoder() {  
 return new BCryptPasswordEncoder();  
 }  
}

OUTPUT:





