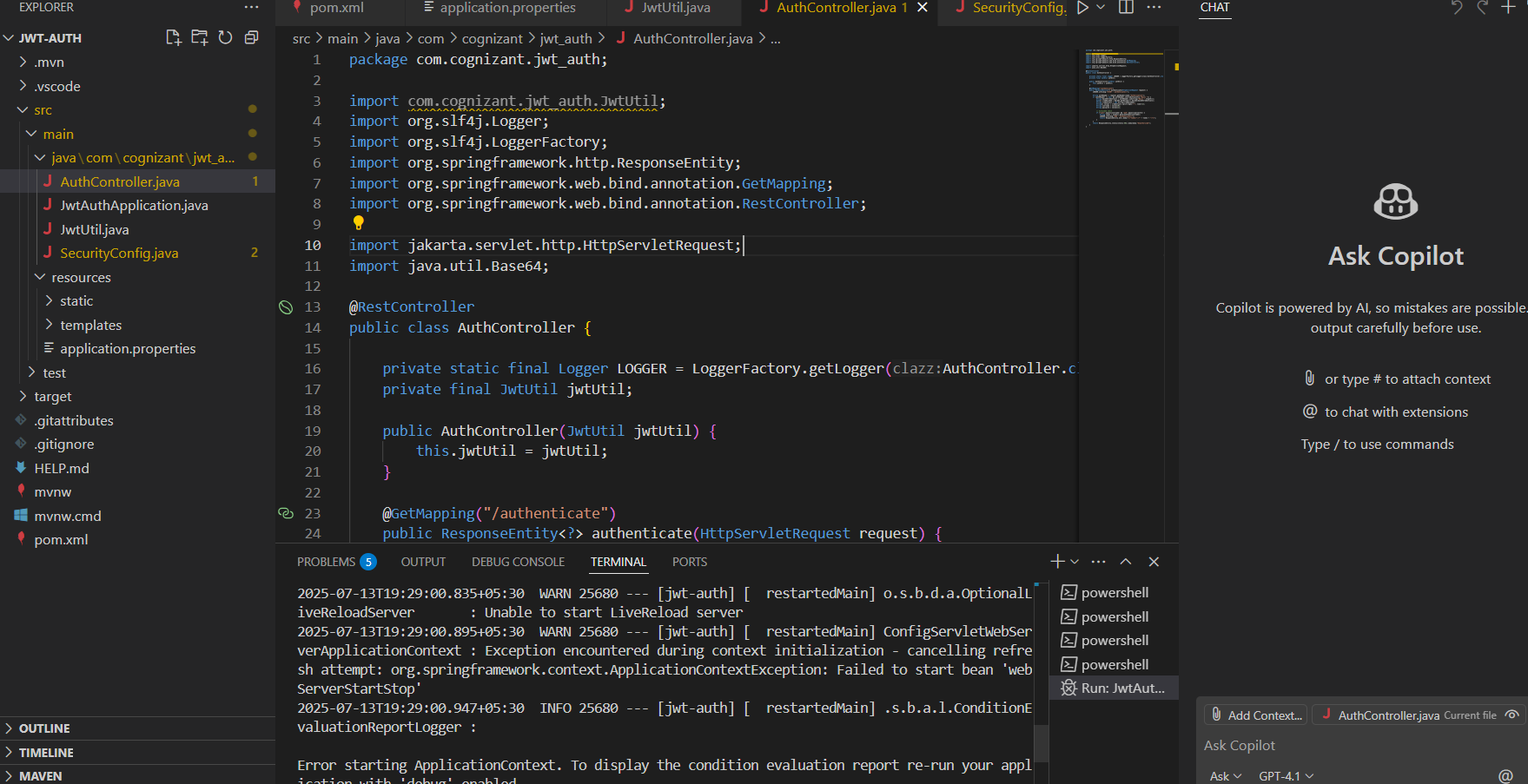
**Create authentication service that returns JWT**



pom.xml:

<?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.cognizant</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>org.springframework.boot</groupId>

            <artifactId>spring-boot-starter-web</artifactId>

        </dependency>

        <dependency>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-devtools</artifactId>

            <scope>runtime</scope>

            <optional>true</optional>

        </dependency>

        <dependency>

            <groupId>org.projectlombok</groupId>

            <artifactId>lombok</artifactId>

            <optional>true</optional>

        </dependency>

        <dependency>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-starter-test</artifactId>

            <scope>test</scope>

        </dependency>

        <dependency>

    <groupId>jakarta.servlet</groupId>

    <artifactId>jakarta.servlet-api</artifactId>

    <version>6.0.0</version>

    <scope>provided</scope>

</dependency>

        <dependency>

            <groupId>org.springframework.security</groupId>

            <artifactId>spring-security-test</artifactId>

            <scope>test</scope>

        </dependency>

        <dependency>

    <groupId>io.jsonwebtoken</groupId>

    <artifactId>jjwt</artifactId>

    <version>0.9.1</version>

</dependency>

    </dependencies>

    <build>

        <plugins>

            <plugin>

                <groupId>org.apache.maven.plugins</groupId>

                <artifactId>maven-compiler-plugin</artifactId>

                <configuration>

                    <annotationProcessorPaths>

                        <path>

                            <groupId>org.projectlombok</groupId>

                            <artifactId>lombok</artifactId>

                        </path>

                    </annotationProcessorPaths>

                </configuration>

            </plugin>

            <plugin>

                <groupId>org.springframework.boot</groupId>

                <artifactId>spring-boot-maven-plugin</artifactId>

                <configuration>

                    <excludes>

                        <exclude>

                            <groupId>org.projectlombok</groupId>

                            <artifactId>lombok</artifactId>

                        </exclude>

                    </excludes>

                </configuration>

            </plugin>

        </plugins>

    </build>

</project>

AuthController.java:

package com.cognizant.jwt\_auth;

import com.cognizant.jwt\_auth.JwtUtil;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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 AuthController {

    private static final Logger LOGGER = LoggerFactory.getLogger(AuthController.class);

    private final JwtUtil jwtUtil;

    public AuthController(JwtUtil jwtUtil) {

        this.jwtUtil = jwtUtil;

    }

    @GetMapping("/authenticate")

    public ResponseEntity<?> authenticate(HttpServletRequest request) {

        LOGGER.info("START - authenticate()");

        String authHeader = request.getHeader("Authorization");

        if (authHeader != null && authHeader.startsWith("Basic ")) {

            String base64Credentials = authHeader.substring("Basic ".length());

            byte[] credDecoded = Base64.getDecoder().decode(base64Credentials);

            String credentials = new String(credDecoded);

            String[] values = credentials.split(":", 2);

            String username = values[0];

            String password = values[1];

            // hardcoded check

            if ("user".equals(username) && "pwd".equals(password)) {

                String token = jwtUtil.generateToken(username);

                LOGGER.info("END - authenticate()");

                return ResponseEntity.ok().body("{\"token\":\"" + token + "\"}");

            }

        }

        return ResponseEntity.status(401).body("Unauthorized");

    }

}

JwtAuthApplication.java:

package com.cognizant.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);

    }

}

JwtUtil.java:

package com.cognizant.jwt\_auth;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.stereotype.Component;

import java.util.Date;

@Component

public class JwtUtil {

    private String secret = "mySecretKey";

    public String generateToken(String username) {

        return Jwts.builder()

                .setSubject(username)

                .setIssuedAt(new Date(System.currentTimeMillis()))

                .setExpiration(new Date(System.currentTimeMillis() + 1000 \* 60 \* 60 \* 10)) // 10 hours

                .signWith(SignatureAlgorithm.HS256, secret)

                .compact();

    }

}

SecurityConfig.java:

package com.cognizant.jwt\_auth;

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;

@Configuration

public class SecurityConfig {

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeHttpRequests()

.anyRequest().permitAll();

return http.build();

}

}