

5. JWT-handson

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/>

  </parent>

  <groupId>com.cognizant</groupId>

  <artifactId>spring-learn</artifactId>

  <version>0.0.1-SNAPSHOT</version>

  <name>jwt-auth-service</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.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>
    <dependency>
        <groupId>io.jsonwebtoken</groupId>
        <artifactId>jjwt</artifactId>
        <version>0.9.1</version>
    </dependency>
</dependencies>
<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
        </plugin>
    </plugins>
</build>
</project>

```

application.properties

```

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

```

AuthenticationController.java

```

package com.example;

import io.jsonwebtoken.Jwts;
import io.jsonwebtoken.SignatureAlgorithm;
import jakarta.servlet.http.HttpServletRequest;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import java.util.Base64;
import java.util.Date;

```

@RestController

```
public class AuthenticationController {

    private static final String SECRET_KEY = "secretkey123";

    @RequestMapping(value = "/authenticate", method = RequestMethod.GET)

    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[] decodedBytes = Base64.getDecoder().decode(base64Credentials);

        String decoded = new String(decodedBytes);

        String[] parts = decoded.split(":", 2);

        if (parts.length != 2) {

            return ResponseEntity.status(400).body("Invalid credentials format");

        }

        String username = parts[0];

        String password = parts[1];

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

            String token = Jwts.builder()

                .setSubject(username)

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

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

                .signWith(SignatureAlgorithm.HS256, SECRET_KEY)

                .compact();

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

        } else {

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

        }

    }

}
```

SecurityConfig.java

```
package com.example;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.web.SecurityFilterChain;
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()

            .anyRequest().authenticated()


        );

        return http.build();

    }

}
```

Output :



```
=====
:: Spring Boot ::                (v3.5.3)

2025-07-13T22:15:52.653+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] com.example.JwtAuthServiceApplication : Starting JwtAuthServiceApplication using Java
2025-07-13T22:15:52.655+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] com.example.JwtAuthServiceApplication : No active profile set, falling back to 1 defa
2025-07-13T22:15:52.697+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active! Set 'spring
2025-07-13T22:15:52.698+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging consider s
2025-07-13T22:15:53.314+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 8090 (http)
2025-07-13T22:15:53.322+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2025-07-13T22:15:53.323+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.
2025-07-13T22:15:53.347+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationCo
2025-07-13T22:15:53.348+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization co
2025-07-13T22:15:53.508+05:30 WARN 9600 --- [jwt-auth-service] [ restartedMain] .s.s.UserDetailsServiceAutoConfiguration :

Using generated security password: faa1caf0-e9e5-4329-8cdc-365165eb4c98

This generated password is for development use only. Your security configuration must be updated before running your application in production.

2025-07-13T22:15:53.513+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] r$InitializeUserDetailsServiceConfigurer : Global AuthenticationManager configured with
2025-07-13T22:15:53.647+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2025-07-13T22:15:53.670+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8090 (http) with conte
2025-07-13T22:15:53.677+05:30 INFO 9600 --- [jwt-auth-service] [ restartedMain] com.example.JwtAuthServiceApplication : Started JwtAuthServiceApplication in 1.312 se
```

Command :

curl -s -u user:pwd <http://localhost:8090/authenticate>

Response :

```
{"token":"eyJhbGciOiJIUzI1NiJ9..."}
```