WEEK 4 HANDS ON

JWT HANDS ON

**Create authentication service that returns JWT**   
  
<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

        http://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>2.7.18</version>

    <relativePath/>

</parent>

<groupId>com.cognizant</groupId>

<artifactId>spring-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<dependencies>

    <dependency>

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

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

    </dependency>

    <dependency>

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

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

    </dependency>

    <dependency>

    <groupId>io.jsonwebtoken</groupId>

    <artifactId>jjwt</artifactId>

    <version>0.9.0</version>

    </dependency>

    <dependency>

    <groupId>org.slf4j</groupId>

    <artifactId>slf4j-api</artifactId>

    </dependency>

    <dependency>

    <groupId>ch.qos.logback</groupId>

    <artifactId>logback-classic</artifactId>

    </dependency>

</dependencies>

<build>

    <plugins>

    <plugin>

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

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

        <version>2.7.18</version>

    </plugin>

    </plugins>

</build>

</project>

package com.cognizant.spring\_learn.controller;

import io.jsonwebtoken.JwtBuilder;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestHeader;

import org.springframework.web.bind.annotation.RestController;

import java.util.Base64;

import java.util.Date;

import java.util.HashMap;

import java.util.Map;

@RestController

public class AuthenticationController {

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

    @GetMapping("/authenticate")

    public Map<String, String> authenticate(@RequestHeader("Authorization") String authHeader) {

        LOGGER.debug("START: authenticate");

        LOGGER.debug("Authorization Header: {}", authHeader);

        String user = getUser(authHeader);

        String token = generateJwt(user);

        Map<String, String> map = new HashMap<>();

        map.put("token", token);

        LOGGER.debug("END: authenticate");

        return map;

    }

    private String getUser(String authHeader) {

        LOGGER.debug("START: getUser");

        // Remove "Basic " prefix

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

        byte[] decodedBytes = Base64.getDecoder().decode(encodedCredentials);

        String decodedString = new String(decodedBytes);

        String user = decodedString.split(":")[0];

        LOGGER.debug("Decoded user: {}", user);

        LOGGER.debug("END: getUser");

        return user;

    }

    private String generateJwt(String user) {

        LOGGER.debug("START: generateJwt");

        JwtBuilder builder = Jwts.builder();

        builder.setSubject(user);

        builder.setIssuedAt(new Date());

        builder.setExpiration(new Date(System.currentTimeMillis() + 1200000)); // 20 mins

        builder.signWith(SignatureAlgorithm.HS256, "secretkey");

        String token = builder.compact();

        LOGGER.debug("Generated Token: {}", token);

        LOGGER.debug("END: generateJwt");

        return token;

    }

}

package com.cognizant.spring\_learn.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;

@Configuration

public class SecurityConfig {

    @Bean

    SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

        http

            .csrf().disable() // For test only. Not for prod.

            .authorizeRequests()

                .antMatchers("/countries").hasRole("USER")

                .antMatchers("/authenticate").hasAnyRole("USER", "ADMIN")

                .anyRequest().authenticated()

            .and()

            .httpBasic();

        return http.build();

    }

}

package com.cognizant.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);

    }

}

Output

