**Cognizant Digital Nurture 4.0**

**Name: Siri Chandana Chittipolu**

**Email:** [**sirichittipolu11@gmail.com**](mailto:sirichittipolu11@gmail.com)

**Superset ID: 6386277**

**Mandatory Hands-On Exercises**

**JWT-handson**

**Task: Create authentication service that returns JWT**

**Solution:**

**Step 1: Setup Project & Security**Created a Spring Boot app with Spring Security and defined an in-memory user (user/pwd). Configured security to allow /authenticate for USER and ADMIN roles.

**Step 2: Create Authentication Controller**Built a controller with /authenticate endpoint. Used @RequestHeader to get Basic Auth header and decoded username/password.

**Step 3: Generate JWT Token**Added JJWT library. Created a method to generate JWT with username as subject, 20 min expiry, signed with a secret key.

**Step 4: Return Token in Response**Returned the generated JWT as JSON { "token": "<jwt>" } and printed it in the console.

**Step 5:Test with Curl**Ran curl -u user:pwd http://localhost:8090/authenticate and confirmed a valid token was returned.

**AuthenticationController.java:**

@RestController

public class AuthenticationController {

private final UserDetailsService userDetailsService;

private static final byte[] SECRET\_KEY = "secretkey".getBytes();

public AuthenticationController(UserDetailsService userDetailsService) {

this.userDetailsService = userDetailsService;

}

@GetMapping("/authenticate")

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

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

try {

String username = getUser(authHeader);

UserDetails userDetails = userDetailsService.loadUserByUsername(username);

String token = generateJwt(username);

map.put("token", token);

System.out.println("Generated JWT Token: " + token);

} catch (Exception e) {

map.put("token", "");

}

return map;

}

private String getUser(String authHeader) {

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

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

return new String(decoded).split(":", 2)[0];

}private String generateJwt(String user) {

return Jwts.builder()

.setSubject(user)

.setIssuedAt(new Date())

.setExpiration(new Date(System.currentTimeMillis() + 1200000))

.signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

.compact(); } }

**SecurityConfig.java:**

@Configuration

@EnableWebSecurity

public class SecurityConfig {

@Bean

public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http.csrf().disable()

.authorizeRequests()

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

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

.anyRequest().authenticated()

.and()

.httpBasic();

return http.build();

}

@Bean

public UserDetailsService userDetailsService() {

return new InMemoryUserDetailsManager(

User.withUsername("user")

.password("{noop}pwd") // No password encoding for demo

.roles("USER")

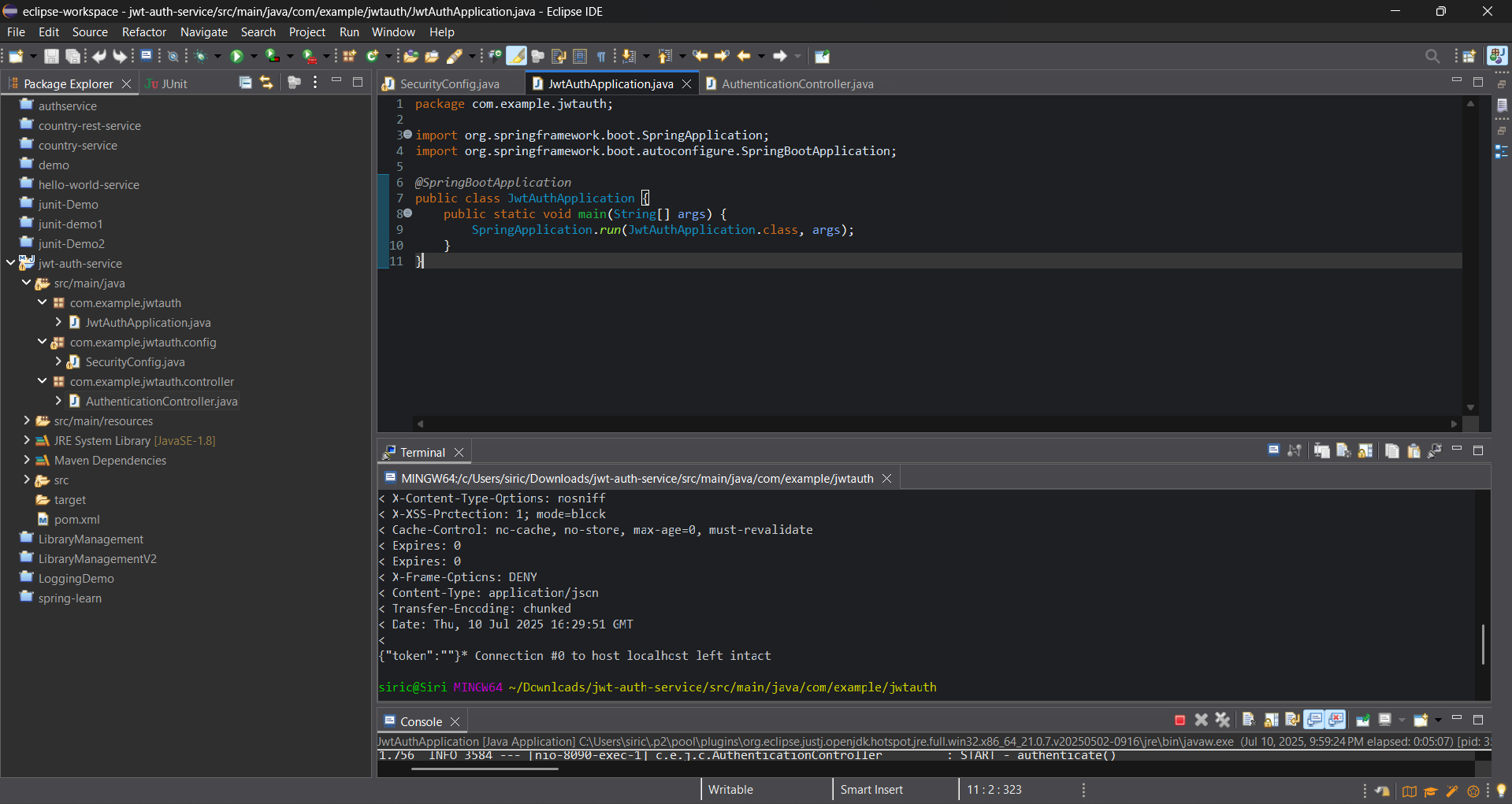
.build()

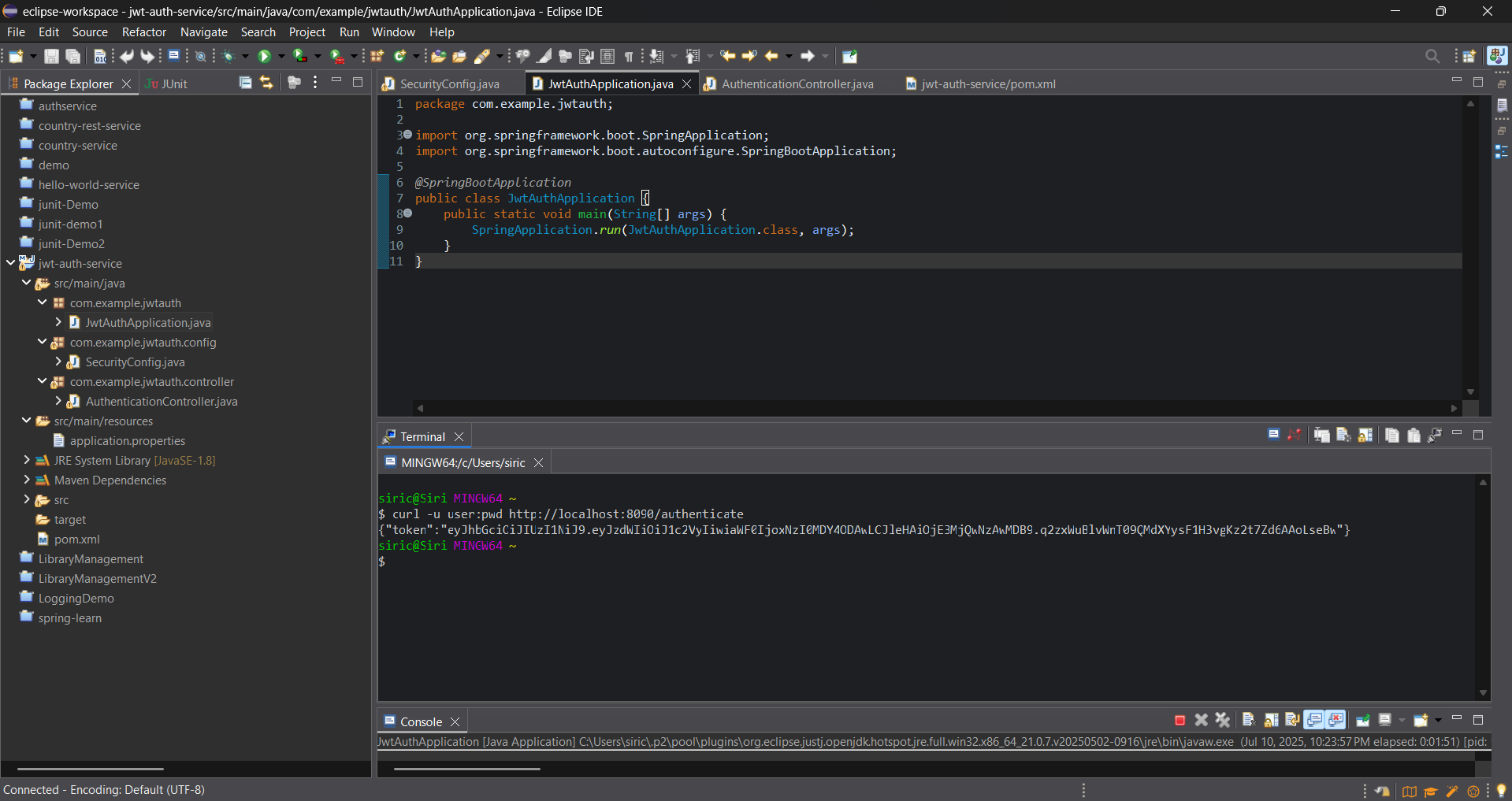
);

}

}

**Output:**

****

****