

Create authentication service that returns JWT From File 5.JWT-handson

SecurityConfig.java From Package com.cognizant.spring_learn.config Code:

```
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.config.annotation.web.configuration.EnableWebS
ecurity;
import org.springframework.security.web.SecurityFilterChain;

@Configuration
@EnableWebSecurity

public class SecurityConfig {

    @Bean

    public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

        http.csrf().disable()

            .authorizeHttpRequests(auth -> auth

                .requestMatchers("/authenticate").permitAll()

                .anyRequest().authenticated()

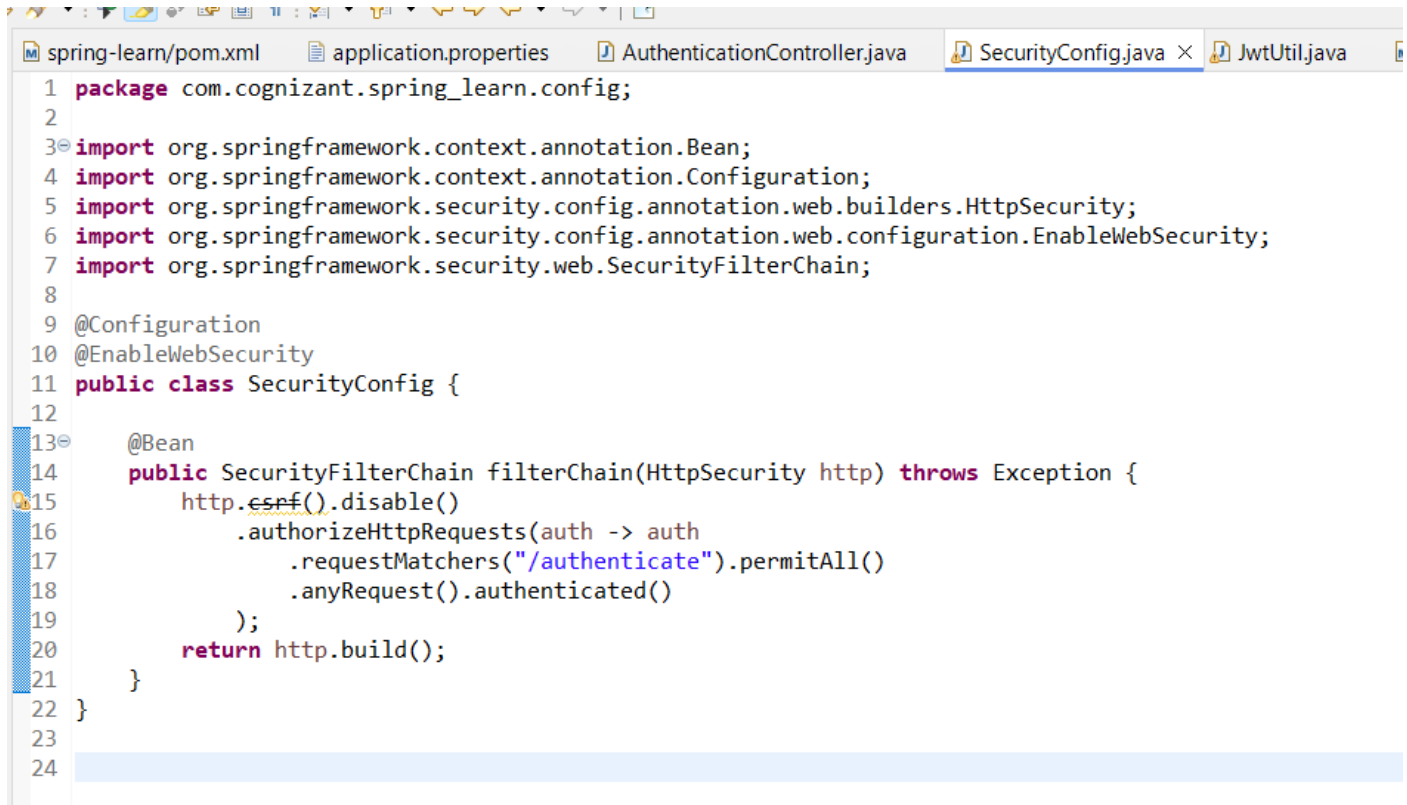
            );

        return http.build();

    }

}
```

Pic From Eclipse IDE:

A screenshot of the Eclipse IDE interface. The top toolbar shows various icons for file operations. Below the toolbar, the 'Project Explorer' on the left shows a project named 'spring-learn'. The 'Package Explorer' on the right shows the package structure. The main editor window displays the 'SecurityConfig.java' file. The code is as follows:

```
1 package com.cognizant.spring_learn.config;
2
3 import org.springframework.context.annotation.Bean;
4 import org.springframework.context.annotation.Configuration;
5 import org.springframework.security.config.annotation.web.builders.HttpSecurity;
6 import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
7 import org.springframework.security.web.SecurityFilterChain;
8
9 @Configuration
10 @EnableWebSecurity
11 public class SecurityConfig {
12
13     @Bean
14     public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
15         http.csrf().disable()
16             .authorizeHttpRequests(auth -> auth
17                 .requestMatchers("/authenticate").permitAll()
18                 .anyRequest().authenticated()
19             );
20         return http.build();
21     }
22 }
23
24
```

AuthenticationController.java From Package com.cognizant.spring_learn.controller:

Code:

```
package com.cognizant.spring_learn.controller;

import java.util.Map;

import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestHeader;
import org.springframework.web.bind.annotation.RestController;
import com.cognizant.spring_learn.util.JwtUtil;
import java.util.Base64;
import org.springframework.beans.factory.annotation.Autowired;

@RestController

public class AuthenticationController {
```

```
@Autowired

private JwtUtil jwtUtil;

@GetMapping("/authenticate")

public ResponseEntity<?> authenticate(@RequestHeader("Authorization")
String authHeader) {

    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 decodedString = new String(decodedBytes);
    String[] credentials = decodedString.split(":", 2);
    String username = credentials[0];
    String password = credentials[1];

    // Optional: Validate username and password against in-memory values
    if (!username.equals("user") || !password.equals("pwd")) {

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

    }

    String token = jwtUtil.generateToken(username);
    return ResponseEntity.ok(Map.of("token", token));

}

}
```

Pic From Eclipse IDE:

spring_learn/controller/AuthenticationController.java - Eclipse IDE

```
ect Run Window Help
spring-learn/pom.xml application.properties AuthenticationController.java SecurityConfig.java JwtUtil.java spring-learn/por

2
3 import java.util.Map;
4
5 import org.springframework.http.ResponseEntity;
6 import org.springframework.web.bind.annotation.GetMapping;
7 import org.springframework.web.bind.annotation.RequestHeader;
8 import org.springframework.web.bind.annotation.RestController;
9
10 import com.cognizant.spring_learn.util.JwtUtil;
11
12 import java.util.Base64;
13
14 import org.springframework.beans.factory.annotation.Autowired;
15
16 @RestController
17 public class AuthenticationController {
18
19     @Autowired
20     private JwtUtil jwtUtil;
21
22     @GetMapping("/authenticate")
23     public ResponseEntity<?> authenticate(@RequestHeader("Authorization") String authHeader) {
24         if (authHeader == null || !authHeader.startsWith("Basic ")) {
25             return ResponseEntity.status(401).body("Missing or invalid Authorization header");
26         }
27
28         String base64Credentials = authHeader.substring("Basic ".length());
29         byte[] decodedBytes = Base64.getDecoder().decode(base64Credentials);
30         String decodedString = new String(decodedBytes);
31         String[] credentials = decodedString.split(":", 2);
32         String username = credentials[0];
33         String password = credentials[1];
34
35         // Optional: Validate username and password against in-memory values
36         if (!username.equals("user") || !password.equals("pwd")) {
37             return ResponseEntity.status(401).body("Invalid credentials");
38         }
39
40         String token = jwtUtil.generateToken(username);
41         return ResponseEntity.ok(Map.of("token", token));
42     }
43 }
44
45
46
```

JwtUtil.java From Package com.cognizant.spring_learn.util:

Code:

```
package com.cognizant.spring_learn.util;
```

```
import io.jsonwebtoken.Jwts;
```

```
import io.jsonwebtoken.SignatureAlgorithm;
```

```
import org.springframework.stereotype.Component;
```

```
import java.util.Date;
```

@Component

```
public class JwtUtil {
```

```
    private final String SECRET_KEY = "my-secret-key"; // keep it secure in real apps
```

```
    private final long EXPIRATION_TIME = 1000 * 60 * 60; // 1 hour in milliseconds
```

```
    public String generateToken(String username) {
```

```
        return Jwts.builder()
```

```
            .setSubject(username)
```

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

```
            .setExpiration(new Date(System.currentTimeMillis() + EXPIRATION_TIME))
```

```
            .signWith(SignatureAlgorithm.HS256, SECRET_KEY)
```

```
            .compact();
```

```
    }
```

```
}
```

Pic From Eclipse IDE:

The image is a screenshot of the Eclipse IDE interface. The top menu bar shows 'Project', 'Run', 'Window', and 'Help'. Below the menu bar is a toolbar with various icons. The project explorer on the left shows a project named 'spring-learn' with files 'pom.xml', 'application.properties', 'AuthenticationController.java', 'SecurityConfig.java', 'JwtUtil.java', and 'spring-learn/pom.xml'. The 'JwtUtil.java' file is selected and its code is displayed in the main editor. The code is as follows:

```
1 package com.cognizant.spring_learn.util;
2
3 import io.jsonwebtoken.Jwts;
4 import io.jsonwebtoken.SignatureAlgorithm;
5 import org.springframework.stereotype.Component;
6
7 import java.util.Date;
8
9 @Component
10 public class JwtUtil {
11
12     private final String SECRET_KEY = "my-secret-key"; // keep it secure in real apps
13     private final long EXPIRATION_TIME = 1000 * 60 * 60; // 1 hour in milliseconds
14
15     public String generateToken(String username) {
16         return Jwts.builder()
17             .setSubject(username)
18             .setIssuedAt(new Date(System.currentTimeMillis()))
19             .setExpiration(new Date(System.currentTimeMillis() + EXPIRATION_TIME))
20             .signWith(SignatureAlgorithm.HS256, SECRET_KEY)
21             .compact();
22     }
23 }
24
25
```

Pom.xml: Dependencies

```
<dependencies>
```

```
  <!-- Spring Web Starter -->
```

```
  <dependency>
```

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

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

```
  </dependency>
```

```
  <!-- Spring DevTools -->
```

```
  <dependency>
```

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

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

```
    <scope>runtime</scope>
```

```
    <optional>true</optional>
```

```
  </dependency>
```

```
  <!-- Spring Web -->
```

```
  <dependency>
```

```
    <groupId>org.springframework</groupId>
```

```
    <artifactId>spring-web</artifactId>
```

```
  </dependency>
```

```
  <!-- Spring Web MVC -->
```

```
  <dependency>
```

```
    <groupId>org.springframework</groupId>
```

```
    <artifactId>spring-webmvc</artifactId>
```

```
  </dependency>
```

<!-- Servlet API -->

<dependency>

<groupId>javax.servlet</groupId>

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

<version>4.0.1</version>

<scope>provided</scope>

</dependency>

<!-- Spring Security -->

<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> <!-- or jjwt-gson -->

<version>0.11.5</version>

<scope>runtime</scope>

</dependency>

<!-- JAXB API (required for Java 11+) -->

<dependency>

<groupId>jakarta.xml.bind</groupId>

<artifactId>jakarta.xml.bind-api</artifactId>

<version>3.0.1</version>

</dependency>

<!-- JAXB Runtime -->

<dependency>

<groupId>org.glassfish.jaxb</groupId>

<artifactId>jaxb-runtime</artifactId>

<version>3.0.2</version>

</dependency>

<!-- Testing -->

<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

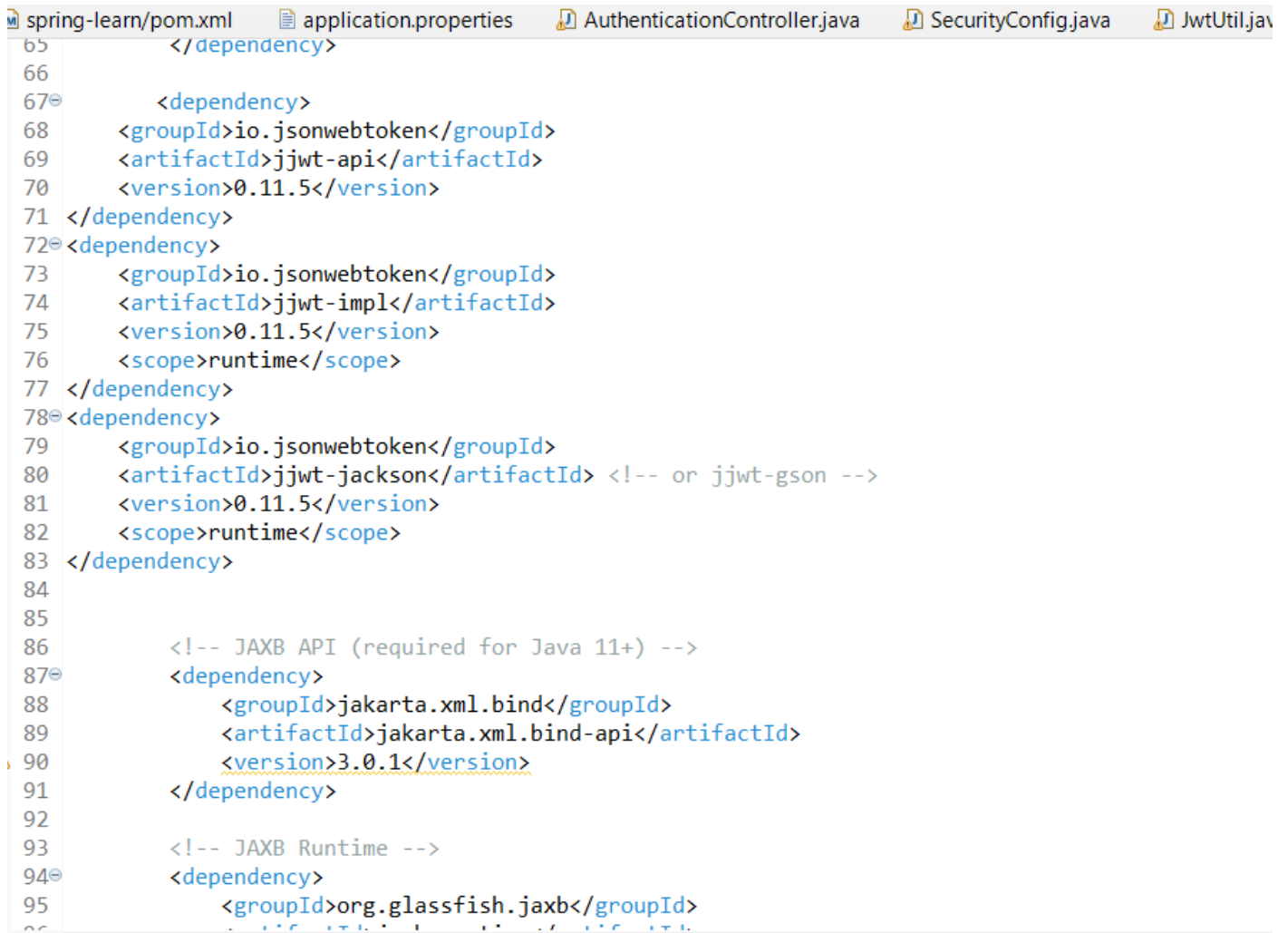
<artifactId>jjwt</artifactId>


```
<version>0.9.1</version>
```

```
</dependency>
```

```
</dependencies>
```

Pic From Eclipse IDE:



```
spring-learn/pom.xml application.properties AuthenticationController.java SecurityConfig.java JwtUtil.java
65     </dependency>
66
67     <dependency>
68         <groupId>io.jsonwebtoken</groupId>
69         <artifactId>jjwt-api</artifactId>
70         <version>0.11.5</version>
71     </dependency>
72     <dependency>
73         <groupId>io.jsonwebtoken</groupId>
74         <artifactId>jjwt-impl</artifactId>
75         <version>0.11.5</version>
76         <scope>runtime</scope>
77     </dependency>
78     <dependency>
79         <groupId>io.jsonwebtoken</groupId>
80         <artifactId>jjwt-jackson</artifactId> <!-- or jjwt-gson -->
81         <version>0.11.5</version>
82         <scope>runtime</scope>
83     </dependency>
84
85
86     <!-- JAXB API (required for Java 11+) -->
87     <dependency>
88         <groupId>jakarta.xml.bind</groupId>
89         <artifactId>jakarta.xml.bind-api</artifactId>
90         <version>3.0.1</version>
91     </dependency>
92
93     <!-- JAXB Runtime -->
94     <dependency>
95         <groupId>org.glassfish.jaxb</groupId>
```

SpringLearnApplication.java class

code:

```
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) {

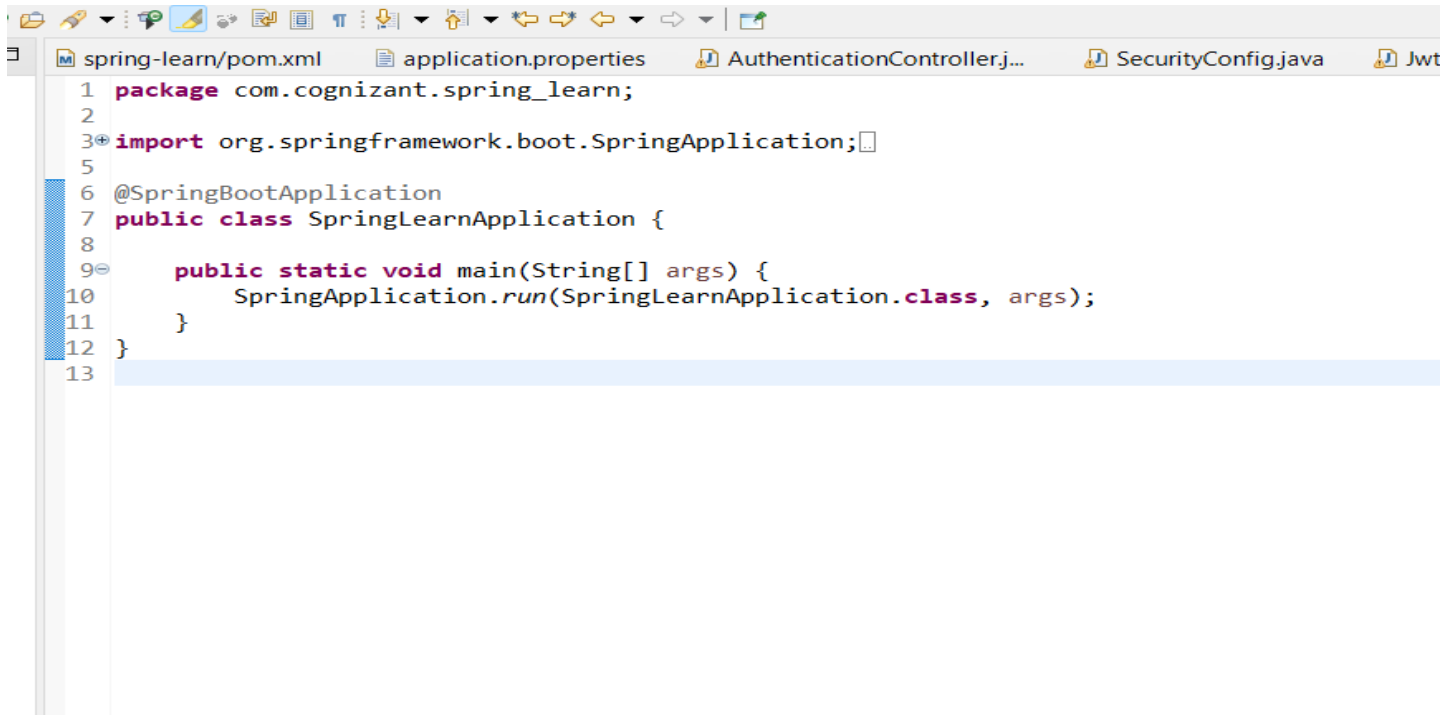
    SpringApplication.run(SpringLearnApplication.class, args);

}

}

```

Pic From Eclipse IDE:



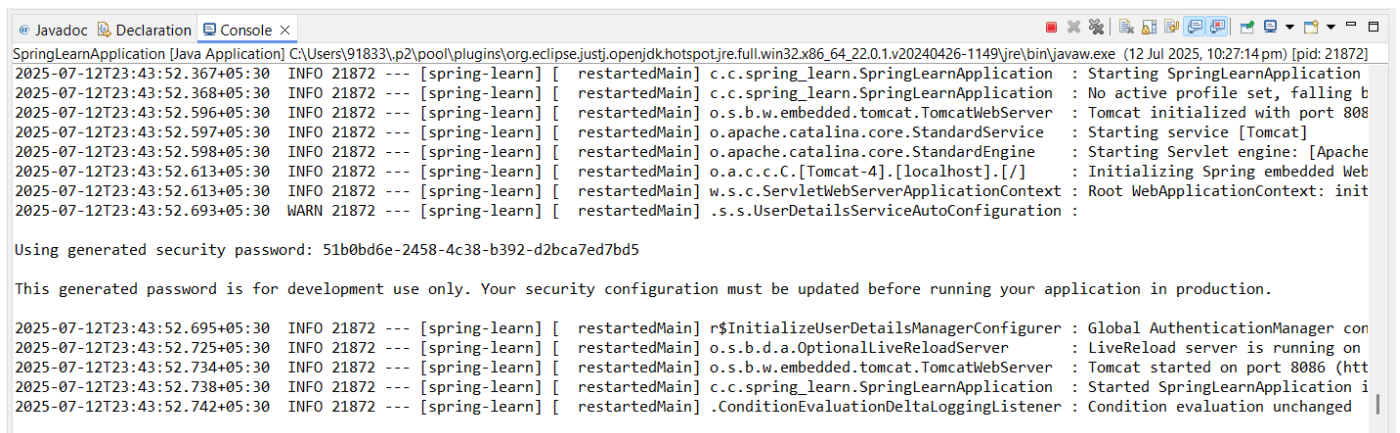
```

1 package com.cognizant.spring_learn;
2
3 import org.springframework.boot.SpringApplication;
4
5
6 @SpringBootApplication
7 public class SpringLearnApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(SpringLearnApplication.class, args);
11     }
12 }
13

```

After Running This Class

OUTPUT From Eclipse :



```

SpringLearnApplication [Java Application] C:\Users\91833\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64.22.0.1.v20240426-1149\jre\bin\javaw.exe (12 Jul 2025, 10:27:14 pm) [pid: 21872]
2025-07-12T23:43:52.367+05:30 INFO 21872 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : Starting SpringLearnApplication
2025-07-12T23:43:52.368+05:30 INFO 21872 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : No active profile set, falling b
2025-07-12T23:43:52.596+05:30 INFO 21872 --- [spring-learn] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 808
2025-07-12T23:43:52.597+05:30 INFO 21872 --- [spring-learn] [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2025-07-12T23:43:52.598+05:30 INFO 21872 --- [spring-learn] [ restartedMain] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache
2025-07-12T23:43:52.613+05:30 INFO 21872 --- [spring-learn] [ restartedMain] o.a.c.c.C.[Tomcat-4].[localhost].[/] : Initializing Spring embedded Web
2025-07-12T23:43:52.613+05:30 INFO 21872 --- [spring-learn] [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: init
2025-07-12T23:43:52.693+05:30 WARN 21872 --- [spring-learn] [ restartedMain] .s.s.UserDetailsServiceAutoConfiguration :

Using generated security password: 51b0bd6e-2458-4c38-b392-d2bca7ed7bd5

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

2025-07-12T23:43:52.695+05:30 INFO 21872 --- [spring-learn] [ restartedMain] r$IinitializeUserDetailsServiceConfigurer : Global AuthenticationManager con
2025-07-12T23:43:52.725+05:30 INFO 21872 --- [spring-learn] [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on
2025-07-12T23:43:52.734+05:30 INFO 21872 --- [spring-learn] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8086 (htt
2025-07-12T23:43:52.738+05:30 INFO 21872 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : Started SpringLearnApplication i
2025-07-12T23:43:52.742+05:30 INFO 21872 --- [spring-learn] [ restartedMain] .ConditionEvaluationDeltaLoggingListener : Condition evaluation unchanged

```

Output From Postman:

GET http://localhost:8086/auth

+

...

HTTP

http://localhost:8086/authenticate

Save

</>

view more actions

GET

http://localhost:8086/authenticate

Send

▼

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

● none

● form-data

● x-www-form-urlencoded

● raw

● binary

JSON

▼

1

{

2

"username": "user",

3

"password": "pwd"

4

}

5

Body

Cookies (1)

Headers (11)

Test Results

🌐

Status: 200 OK

Time: 27 ms

Size: 474 B

Save Response

▼

Pretty

Raw

Preview

Visualize

⌵

"token": "t3LRv1CVhwKfoqZY1aVQqEUiB1oWcWn0ft3tgv0dL0.eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiaWF0IjoxNjU3MDM3OTQ3LCJleHAiOjE2NTcwMzg3NjR9"

⌵