

Spring Security with Authentication

Installation Guide:

1. GitHub link: <https://github.com/baljeet-singh97/JAVA-Projects/tree/main/Phase%203/SpringSecurityAuth>
2. Download the entire project as Zip in local system.
3. import the project in Eclipse IDE

CustomAuthenticationProvider.java

we use a custom AuthenticationProvider with to two dummy users

```
List<User> dummyUsers = new ArrayList<>();

public CustomAuthenticationProvider(){
    dummyUsers.add(new User("john", "secret", "ROLE_USER"));
    dummyUsers.add(new User("admin", "supersecret", "ROLE_ADMIN"));
}
```

In the authenticate method we check if the passed credentials match

```
@Override
public Authentication authenticate(Authentication authentication) throws AuthenticationException {
    String name = authentication.getName();
    String password = authentication.getCredentials().toString();
```

This method returns an Authentication object, that contains username, password and a list of authorities, we pass the user's role as SimpleGrantedAuthority

At that point, the user is authenticated super.setAuthenticated(true);

```
Optional<User> authenticatedUser = dummyUsers.stream().filter(
    user -> user.getName().equals(name) && user.getPassword().equals(password)
).findFirst();

if(!authenticatedUser.isPresent()){
    throw new BadCredentialsException("Some Text");
```

```

    }

    List<GrantedAuthority> authorities = new ArrayList<>();

    authorities.add(new SimpleGrantedAuthority(authenticatedUser.get().getRole()));

    Authentication auth = new UsernamePasswordAuthenticationToken(name, password, authorities);

    return auth;
}

```

MainController.java

In given controller, we have three API methods.

```

@RequestMapping("/")

public String hello(){

    return "Hello World";

}

@RequestMapping("/protected")

public String protectedHello(){

    return "Hello World, i was protected";

}

@RequestMapping("/admin")

public String admin(){

    return "Hello World from admin";

}

```

SpringSecurityConfig.java

AuthenticationProvider provides two methods authenticate and supports

In Line 1 to 6 are all urls public except /protected and /admin, for the urls below we force Basic authentication:

The /protected url is protected by the USER role

The /admin url is protected by the ADMIN role

We pass a custom AuthenticationProvider to AuthenticationManagerBuilder

```
protected void configure(HttpSecurity http) throws Exception {
```

```

http.httpBasic().and().authorizeRequests()

    .antMatchers("/").permitAll()

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

    .antMatchers("/admin").hasRole("ADMIN");
}

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

    auth.authenticationProvider(new CustomAuthenticationProvider());

}

```

User.java

Created a pom class to create table in the database

```

package com.simplilearn.demo;

public class User {

    private String name;

    private String password;

    private String role;

    public User(String name, String password, String role) {

        this.name = name;

        this.password = password;

        this.role = role;

    }

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;
    }
}

```

```
}
```

```
public String getPassword() {
```

```
    return password;
```

```
}
```

```
public void setPassword(String password) {
```

```
    this.password = password;
```

```
}
```

```
public String getRole() {
```

```
    return role;
```

```
}
```

```
public void setRole(String role) {
```

```
    this.role = role;
```

```
}
```

```
}
```