Ringkasan:

Pada tutorial ini kita belajar mengenai web security. Dalam mengembangkan web kita perlu melakukan verifikasi terhadap pengguna sistem kita. Tujuan dari verifikasi ini ialah agar tidak sembarang orang dapat masuk ke sistem. Setelah mem verifikasi/otentikasi terhadap siapa saja pengguna yang berhak untuk masuk kedalam sistem, langkah selanjutnya adalah melakukan otorisasi. Otorisasi merupakan pengaturan hak pengguna yang sebelumnya telah di otentikasi. Pengguna dibedakan berdasarkan levelnya masing – masing agar tujuan dari sistem tercapai.

Untuk melakukan otorisasi pada aplikasi spring bot. Kita perlu membuat sebuah kelas WebSecurityConfig yang di turunkan (inheritance) melalui kelas WebSecurityConfigureAdapter. Kita akan menimpa (override) method configure yang memiliki parameter HttpSecurity.

```
@Configuration
@EnableWebSecurity
public class WebSecurityConfig extends WebSecurityConfigurerAdapter {
	@Override
	protected void configure(HttpSecurity http) throws Exception
	{
	  http
	    .authorizeRequests()
	    .antMatchers("/").permitAll()
	    .anyRequest().authenticated()
	    .and()
	    .formLogin()
	    .loginPage("/login")
	    .permitAll()
	    .and()
	    .logout()
	    .permitAll();
	}

	@Autowired
	public void configureGlobal (AuthenticationManagerBuilder auth) throws Exception
	{
	  auth.inMemoryAuthentication()
	      .withUser("admin").password("admin")
	      .roles("ADMIN");
	}
}
```

Pada method configureGlobal kita menggunakan user admin, password admin lalu roles nya adalah admin.

Latihan

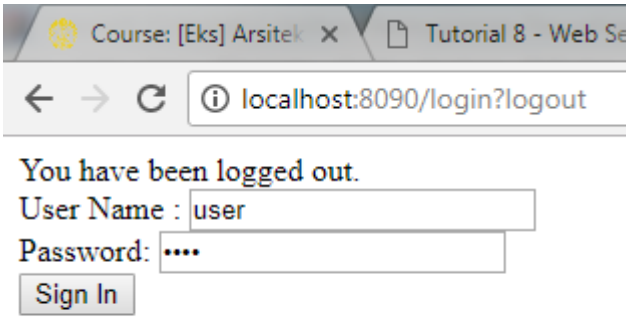
1. Untuk menambahkan signout di halaman student/view/npm cukup tambahkan kode yang sama pada view.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
    <head>
        <title>View Student by NPM</title>
    </head>
    <body>
    <h2 th:text=" 'Login as ' + ${#httpServletRequest.remoteUser}">Login as</h2>
    <form th:action="@{/logout}" method="post">
        <input type="submit" value="Sign Out"/>
    </form>
        <h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
        <h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
        <h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
    </body>
</html>
```

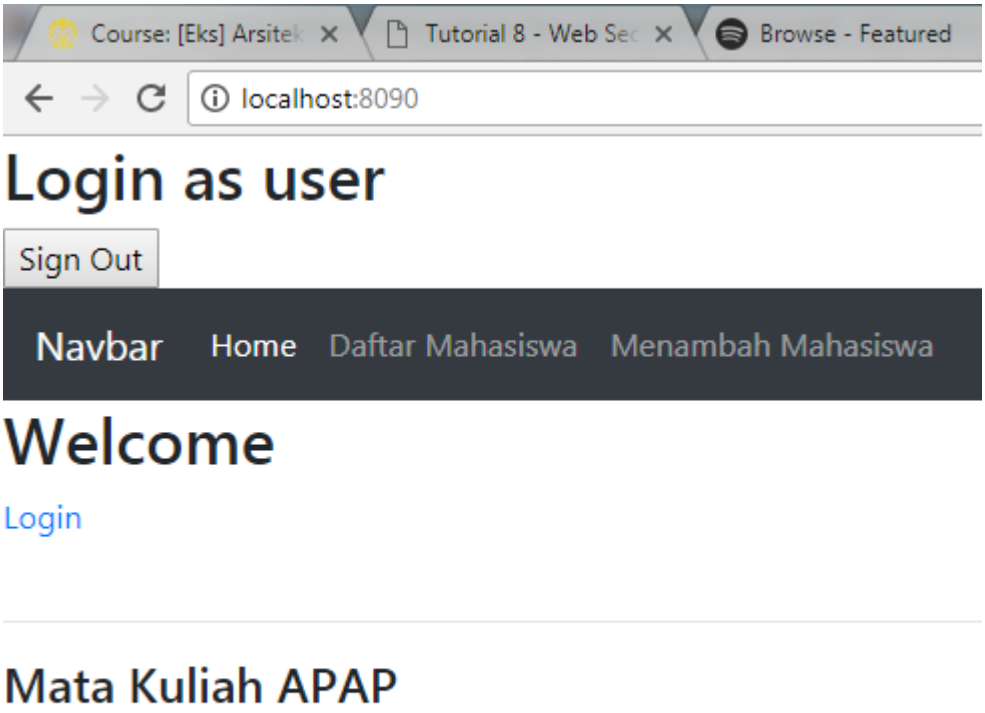
2. Membuat user dengan username **user**, dan password **user** dengan role **user**. Tambahkan pada bagian method configureGlobal.

```
@Autowired
public void configureGlobal (AuthenticationManagerBuilder auth) throws Exception
{
    auth.inMemoryAuthentication()
            .withUser( username: "admin").password("admin")
            .roles("ADMIN");
    auth.inMemoryAuthentication()
            .withUser( username: "user").password("user")
            .roles("USER");
}
```

Cobalah melakukan login dengan menggunakan username **user.**



Apabila login berhasil maka akan tampil informasi seperti berikut ini.



3. untuk membuat form pencarian, tulislah kode program seperti berikut ini.

```
<div th:if="${#httpServletRequest.isUserInRole('USER')}">
    <form th:action="@{/student/view/(npm=${npm})}" method="get">
        <div class="row">
            <div class="col-md-2">
                <label for="npm">NPM</label>
            </div>
            <div class="col-md-4">
                <input type="text" id="npm" name="npm" class="form-control"/>
            </div>
        </div>
        <div class="row">
            <button type="submit" value="save" class="btn btn-primary ">Lihat</button>
        </div>
    </form>
</div>
```

Pada form tersebut kita mengarahkan action nya pada /student/view/npm dengan method nya adalah get. Selain itu pada bagian th:if="${#httpServletRequest.isUserInRole('USER')}" berfungsi untuuk melakukan otorisasi, agar yang dapat melihat form ini hanya pengguna yang berperan sebagai user.

4. Sebelumnya user dengan role **admin** sudah dapat melihat semua data mahasiswa. Untuk menambahkan otorisasi terhadap role **admin** agar ia dapat melihat **view student by npm,** Maka perlu ditambahkan pada bagian WebSecurityConfig.java seperti berikut ini.

```
http
    .authorizeRequests()
    .antMatchers( ...antPatterns: "/").permitAll()
    .antMatchers( ...antPatterns: "/student/viewall").hasRole("ADMIN")
    .antMatchers( ...antPatterns: "/student/view/**").hasRole("USER")
    .antMatchers( ...antPatterns: "/student/view/**").hasRole("ADMIN")
    .anyRequest().authenticated()
```