Class 230303

Spring (Spring Security) - 4일차

인증과 인가

- 인증 (authentication)
  - 회원 로그인, 회원/비회원, 멤버쉽
- 인가 (anthorization)
  - 1급 기밀, 접근 권한, 무료회원/유료회원

환경설정

1. **pom.xml**

   ▼ spring-security-web, spring-security-config 의존성 라이브러리 추가

```
<!-- spring-security-web -->
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-web</artifactId>
    <version>5.4.6</version>
</dependency>

<!-- spring-security-config -->
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-config</artifactId>
    <version>5.4.6</version>
</dependency>
```

2. **web.xml**

   ▼ filter 추가 : jsp 페이지를 유저에게 보내기 전에 필터링

```
<!-- security -->
<filter>
    <filter-name>springSecurityFilterChain</filter-name>
    <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
</filter>
<filter-mapping>
    <filter-name>springSecurityFilterChain</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

   ▼ param 추가 : security-context.xml 설정 (권한 / 인증 설정)

```
<!-- The definition of the Root Spring Container shared
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>
        /WEB-INF/spring/root-context.xml
        /WEB-INF/spring/security-context.xml
    </param-value>
</context-param>
```

3. **security-context.xml 작성**
    - 위치 : WEB-INF/spring 하위에 작성

security-context.xml 설정

▼ 기본 프레임

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans
    xmlns="http://www.springframework.org/schema/security"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
     http://www.springframework.org/schema/beans/spring-beans.xsd
     http://www.springframework.org/schema/security
     http://www.springframework.org/schema/security/spring-security.xsd">

    <http auto-config="true" use-expressions="true">

    <!-- intercept-url, login 등 세팅 -->

    </http>

    <authentication-manager>
        <authentication-provider>
            <user-service>

            <!-- 유저, 권한 세팅 -->

            </user-service>
        </authentication-provider>
    </authentication-manager>
</beans:beans>
```

▼ 유저별 권한 설정

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans
    xmlns="http://www.springframework.org/schema/security"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
     http://www.springframework.org/schema/beans/spring-beans.xsd
     http://www.springframework.org/schema/security
     http://www.springframework.org/schema/security/spring-security.xsd">
```

```
    <http auto-config="true" use-expressions="true">

      <intercept-url pattern="/cars/add" access="hasAuthority('ROLE_ADMIN')" />
      <intercept-url pattern="/manager" access="hasRole('ROLE_MANAGER')" />
      <intercept-url pattern="/member" access=" ()" />
      <intercept-url pattern="/**" access="permitAll" />

    </http>

    <authentication-manager>
        <authentication-provider>
            <user-service>

              <user name="admin" password="{noop}admin" authorities="ROLE_ADMIN, ROLE_USER"/>
              <user name="manager" password="{noop}manager" authorities="ROLE_MANAGER, ROLE_USER"/>
              <user name="guest" password="{noop}guest" authorities="ROLE_USER"/>

            </user-service>
        </authentication-provider>
    </authentication-manager>
</beans:beans>
```

- **표현식**
  - **authenticated**() ; 인증된 사용자의 접근을 허용
  - **fullyAuthenticated**(): 인증된 사용자의 접근을 허용, rememberMe인증 제외
  - **permitAll**(): 무조건 허용
  - **denyAll**(): 무조건 차단
  - **anonymous**(): 익명사용자 허용
  - **rememberMe**(): rememberMe 인증 사용자 접근 허용
  - **access**(String): 주어진 **SpEL표현식**의 평가 결과가 true 이면 접근허용
  - **hasRole**(String): 사용자가 주어진 역할이 있다면 접근을 허용
  - **hasAuthority**(String): 사용자가 주어진 권한이 있다면 허용
  - **hasAnyRole**(String...): 사용자가 주어진 어떤권한이라도 있으면 허용
  - **hasAnyAuthority**(String...): 사용자가 주어진 권한중 어떤 것이라도 있다면 허용
  - **hasIpAddress**(String): 주어진 IP로 부터 요청이 왔다면 접근을 허용

▼ 로그인 / 로그아웃

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans
    xmlns="http://www.springframework.org/schema/security"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
   http://www.springframework.org/schema/beans/spring-beans.xsd
   http://www.springframework.org/schema/security
   http://www.springframework.org/schema/security/spring-security.xsd">

    <!-- 기본 로그인 창 사용 시 -->
    <!-- <http auto-config="true" use-expressions="true"> -->
    <!-- 커스텀 로그인 창 사용 시 -->
    <http use-expressions="true">

      <intercept-url pattern="/cars/add" access="hasAuthority('ROLE_ADMIN')" />
```

```
      <intercept-url pattern="/admin" access="hasAuthority('ROLE_ADMIN')" />
      <intercept-url pattern="/manager" access="hasRole('ROLE_MANAGER')" />
      <intercept-url pattern="/member" access="isAuthenticated()" />
      <intercept-url pattern="/**" access="permitAll" />

    <form-login
      login-page="/cars/login"
      default-target-url="/"
      authentication-failure-url="/cars/loginFailed"
      username-parameter="username"
      password-parameter="password"/>

    <csrf/> <!-- 보안 관련 -->
    <logout logout-success-url="/cars/logout"/>

    </http>

    <authentication-manager>
        <authentication-provider>
            <user-service>

              <user name="admin" password="{noop}admin" authorities="ROLE_ADMIN, ROLE_MANAGER, ROLE_USER"/>
              <user name="manager" password="{noop}manager" authorities="ROLE_MANAGER, ROLE_USER"/>
              <user name="guest" password="{noop}guest" authorities="ROLE_USER"/>

            </user-service>
        </authentication-provider>
    </authentication-manager>
</beans:beans>
```

- `form-login`
- `csrf`
- `logout`

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>로그인</title>
<%@ include file="include/header.jsp" %>
</head>
<style>
html,
body {
  height: 100%;
}

body {
  align-items: center;
  padding-bottom: 40px;
}

.form-signin {
  width: 100%;
  max-width: 330px;
  padding: 15px;
  margin: auto;
}

.form-signin .checkbox {
  font-weight: 400;
}
```

```
.form-signin .form-floating:focus-within {
  z-index: 2;
}

.form-signin input[type="email"] {
  margin-bottom: -1px;
  border-bottom-right-radius: 0;
  border-bottom-left-radius: 0;
}

.form-signin input[type="password"] {
  margin-bottom: 10px;
  border-top-left-radius: 0;
  border-top-right-radius: 0;
}
@media (min-width: 768px) {
  .bd-placeholder-img-lg {
    font-size: 3.5rem;
  }
}
</style>
<body>
<%@ include file="include/navbar.jsp" %>
<div class="container">
  <div class="form-signin text-center">

    <form action="/login" method="post" id="loginFrm">
      <h1 class="h3 py-5 fw-normal">로그인</h1>

      <div class="form-floating">
        <input type="text" class="form-control" id="username" name="username" placeholder="ID">
        <label for="floatingInput">이메일</label>
      </div>
      <div class="form-floating">
        <input type="password" class="form-control" id="password" name="password" placeholder="Password">
        <label for="floatingPassword">비밀번호</label>
      </div>

      <button class="w-100 btn btn-lg btn-secondary mb-3" id="loginBtn">로그인</button>
      <input type="hidden" name="${_csrf.parameterName}" value="${_csrf.token}" />
    </form>

  </div>
</div>

<%@ include file="include/footer.jsp" %>
<script>

$('#password').on("keypress", function(){
  if(event.keyCode == 13) {
    $('#loginFrm').suvmit();
  }
})
</script>
</body>
</html>
```

- `<form action="/login" method="post">`
- input
  - `name = username`
  - `name = password`

Python Adv - Crawling 4일차

▼ 모듈

```
import requests                          # 웹 문서를 text형태로 가져오기 위해 필요
from bs4 import BeautifulSoup as bs      # 가져온 text형태의 문서의 태그에 접근할 수 있게 정제
import pandas as pd                      # 데이터 분석
from selenium import webdriver           # 웹 브라우저 컨트롤 (막힌 사이트 스크래핑 용)
from urllib.request import urlretrieve   # 파일 다운로드 관련
import re                                # 정규식 사용
import time
from selenium.webdriver.common.by import By     # 브라우저 조작 관련
from selenium.webdriver.common.keys import Keys # 브라우저 key 입력 관련

import folium                            # 데이터 분석
```

▼ 스타벅스 매장 정보 전체

```
'''
스타벅스
'''

driver = webdriver.Chrome('chromedriver.exe')                    # 크롬을 이용해 불러옴
driver.get('https://www.starbucks.co.kr/store/store_map.do')

time.sleep(2)                                                    # 페이지 로드 전에 하위 코드 실행 방지

area_btn = driver.find_element(By.CSS_SELECTOR, '.loca_search h3 a')
area_btn.click()
time.sleep(1)

seoul_btn = driver.find_element(By.CSS_SELECTOR, '.set_sido_cd_btn')
seoul_btn.click()
time.sleep(1)

all_btn = driver.find_element(By.CSS_SELECTOR, '.set_gugun_cd_btn')
all_btn.click()
time.sleep(2)

html = driver.page_source
bsObj = bs(html)

li = bsObj.select('.quickSearchResultBoxSidoGugun .quickResultLstCon')
ap = bsObj.select('.quickSearchResultBoxSidoGugun .quickResultLstCon > .result_details')
stype = bsObj.select('.quickSearchResultBoxSidoGugun .quickResultLstCon > i')

print(len(li))
print(len(stype))
```

```
storeData = []
for i in range(len(li)):
    apData = str(ap[i]).split('<br/>')
    storeData.append([
        li[i]['data-lat'],
        li[i]['data-long'],
        li[i]['data-name'],
        bs(apData[0]).text,
        bs(apData[1]).text,
        stype[i].text
        ])
df = pd.DataFrame(storeData, columns=['위도', '경도', '매장명', '주소', '전화번호', '매장 종류'])
df
```

▼ `folium` 데이터 분석 모듈 기본 사용

```
map = folium.Map(
    location=[37.5666805, 126.9784147],      # 중심 위도, 경도
    zoom_start=13,                           # zoom 설정 (낮을수록 많이 보임)
    tiles='Stamen Terrain'                   # 지도 모양 설정
)

# 서울 마커 찍기
folium.CircleMarker(
    location=[37.5666805, 126.9784147],
    fill=True
).add_to(map)

# 스타벅스 마커 찍기
for store in storeData:
    folium.Marker(
        location=[store[0], store[1]],
        tooltip=store[2],
        popup=store[3],
    ).add_to(map)
map
```

- 타일 종류
  - OpenStreetMap (기본값)
  - Stamen Terrain, Stamen Toner, Stamen Watercolor
  - CartoDB positron, CartoDB dark_matter