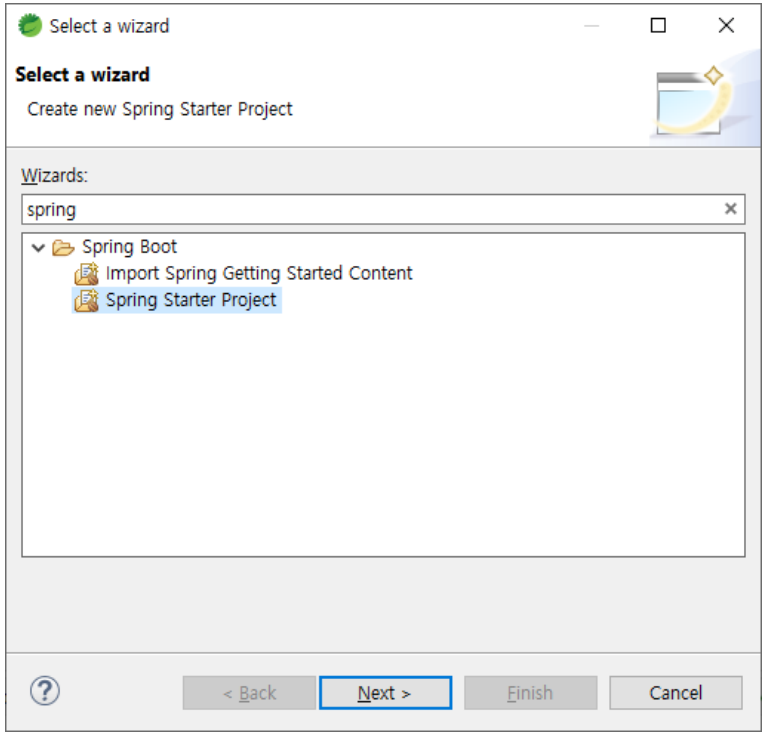
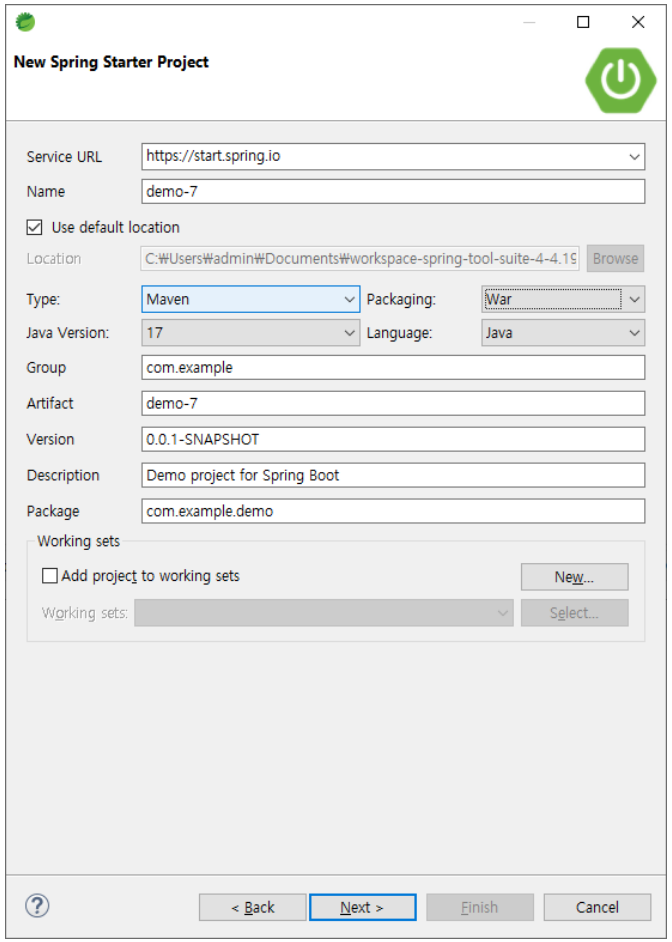
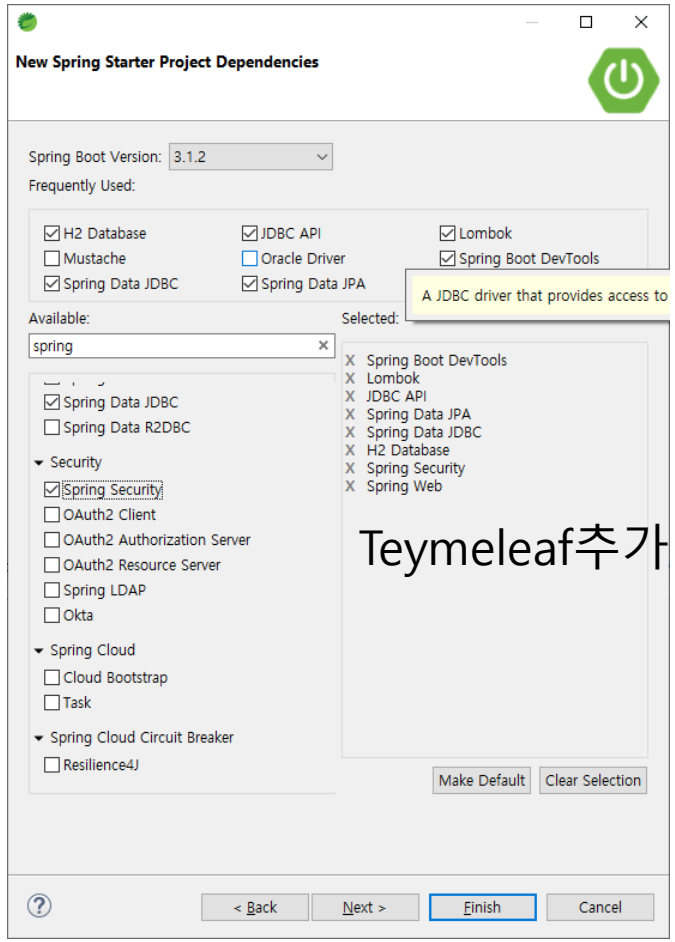
Spring boot 로그인


Select a wizard
Select a wizard
Create new Spring Starter Project
Wizards:
spring
Spring Boot
Import Spring Getting Started Content
Spring Starter Project
< Back
Next >
Finish
Cancel


New Spring Starter Project
Service URL
https://start.spring.io
Name
demo-7
Use default location
Location
C:₩Users₩admin₩Documents₩workspace-spring-tool-suite-4-4.19
Browse
Type:
Maven
Packaging:
War
Java Version:
17
Language:
Java
Group
com.example
Artifact
demo-7
Version
0.0.1-SNAPSHOT
Description
Demo project for Spring Boot
Package
com.example.demo
Working sets
Add project to working sets
New...
Working sets:
Select...
< Back
Next >
Finish
Cancel


New Spring Starter Project Dependencies
Spring Boot Version:
3.1.2
Frequently Used:
H2 Database
JDBC API
Lombok
Mustache
Oracle Driver
Spring Boot DevTools
Spring Data JDBC
Spring Data JPA
A JDBC driver that provides access to
Available:
Selected:
spring
Spring Data JDBC
Spring Data R2DBC
Security
Spring Security
OAuth2 Client
OAuth2 Authorization Server
OAuth2 Resource Server
Spring LDAP
Okta
Spring Cloud
Cloud Bootstrap
Task
Spring Cloud Circuit Breaker
Resilience4J
X Spring Boot DevTools
X Lombok
X JDBC API
X Spring Data JPA
X Spring Data JDBC
X H2 Database
X Spring Security
X Spring Web
Teymeleaf추가
Make Default
Clear Selection
< Back
Next >
Finish
Cancel

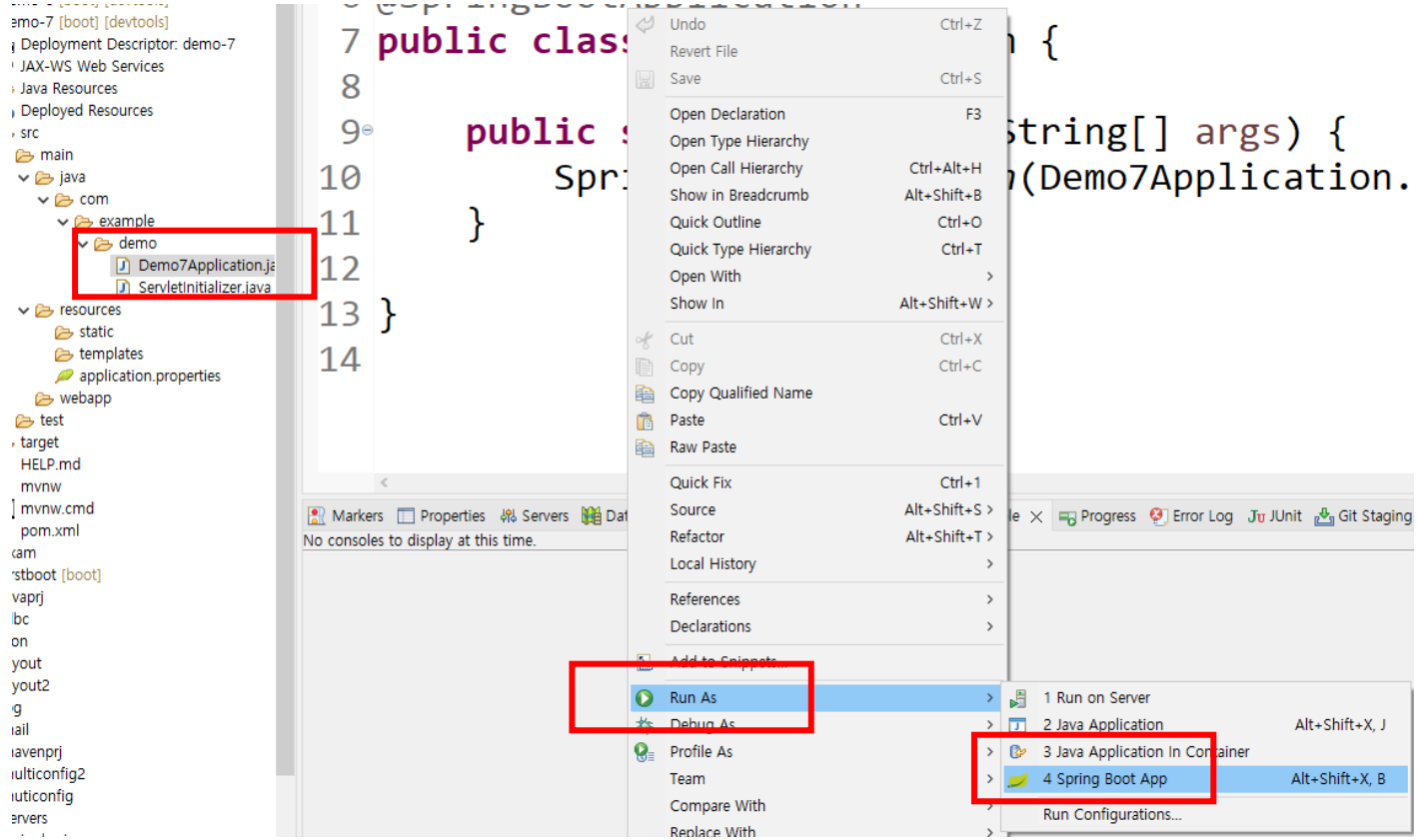
Pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<parent>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-parent</artifactId>
<version>3.1.2</version>
<relativePath/> <!-- lookup parent from repository -->
</parent>
<groupId>com.example</groupId>
<artifactId>demo-7</artifactId>
<version>0.0.1-SNAPSHOT</version>
<packaging>war</packaging>
<name>demo-7</name>
<description>Demo project for Spring Boot</description>
<properties>
<java.version>17</java.version>
</properties>
<dependencies>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-data-jdbc</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-jdbc</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
```

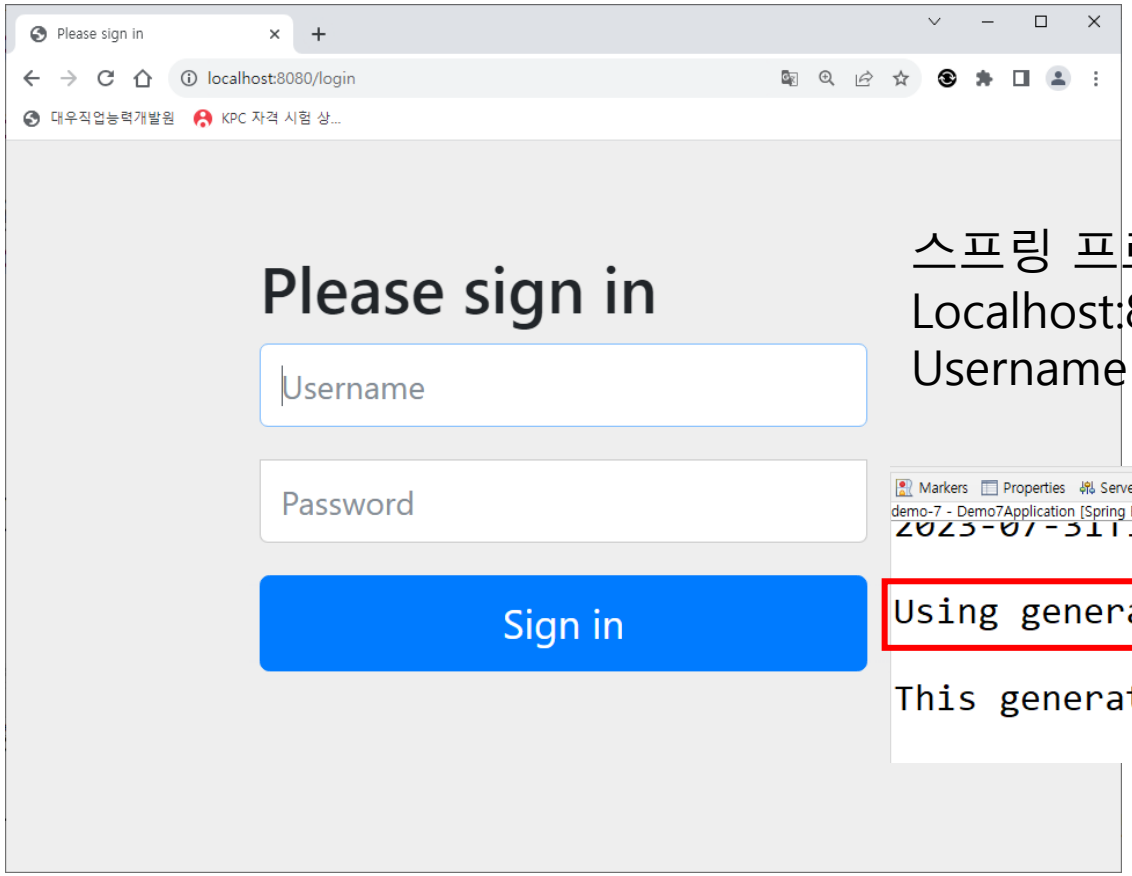
application.properties

```
spring.h2.console.enabled=true
spring.h2.console.path=/h2-console

spring.datasource.url=jdbc:h2:~/test;
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.jpa.properties.hibernate.hbm2ddl.auto=update
```

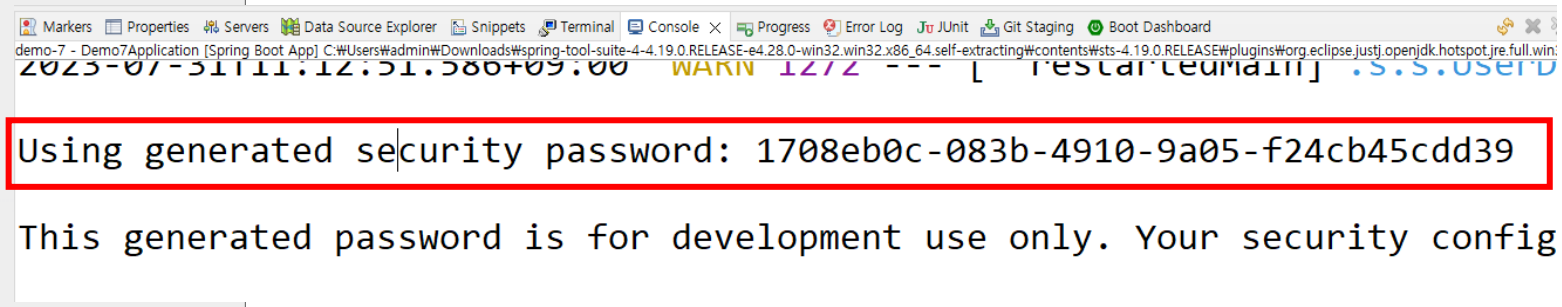
프로그램 실행

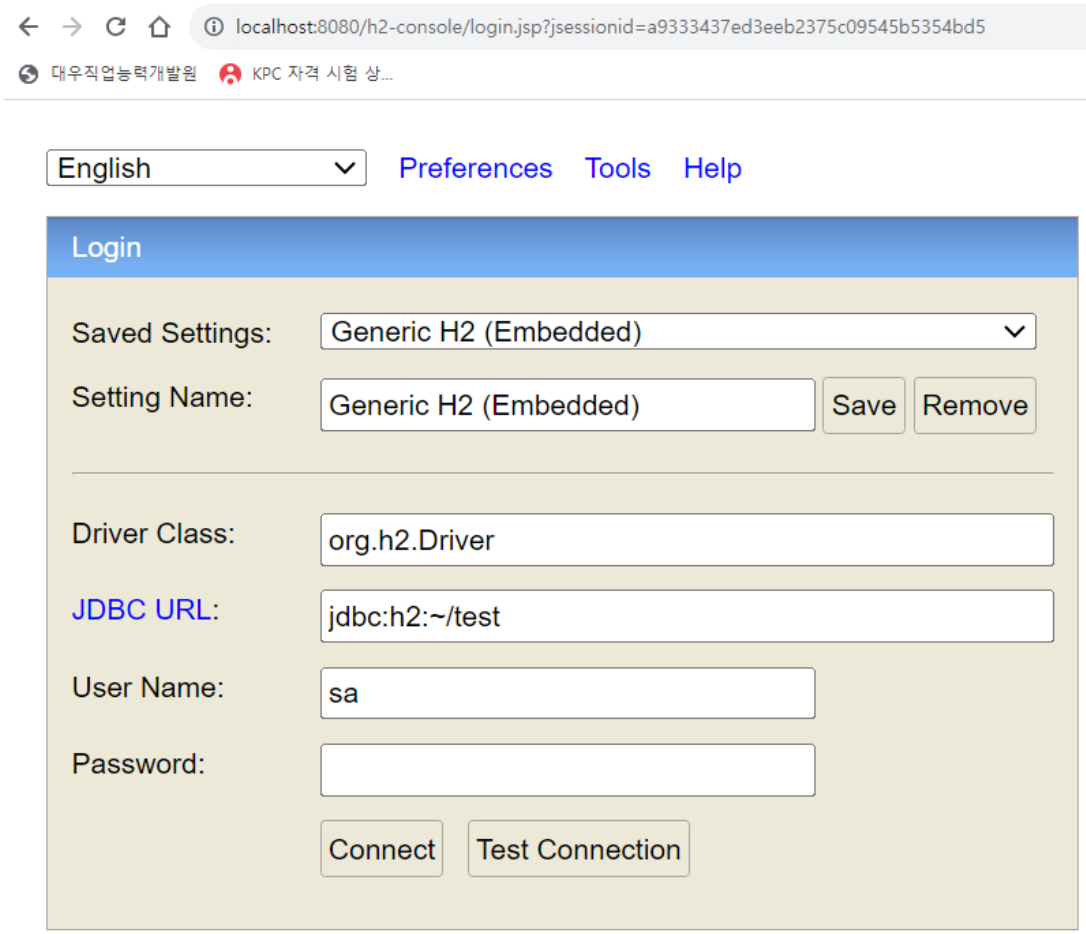


스프링 프로그램을 실행하면 console창에 패스워드가 생성



스프링 프로그램을 실행하면 console창에 패스워드가 생성
Localhost:8080접속하여
Username : user, password에는 콘솔창의 내용을 입력하면 1차인증





Localhost:8080/h2-console 를 접속하여 connect를 누르면 정상적으로 동작하지 않음
Security에서 보안접속 때문임
이를 해결하기 위해 config를 설정해야함.

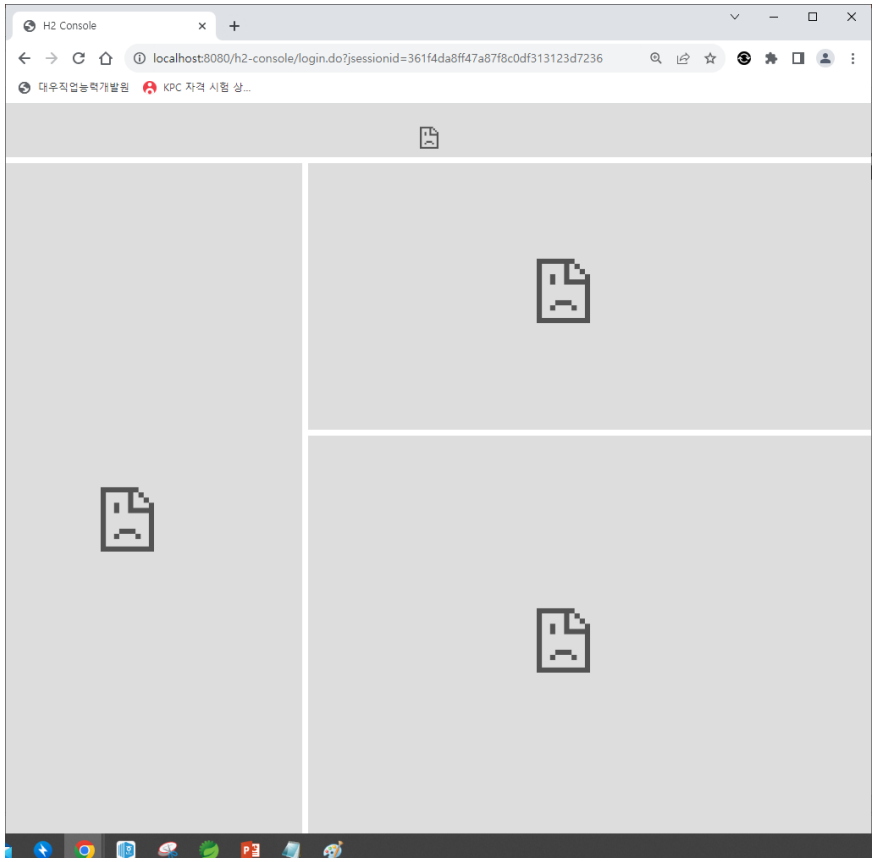
Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Mon Jul 31 11:19:41 KST 2023
There was an unexpected error (type=Forbidden, status=403).
Forbidden

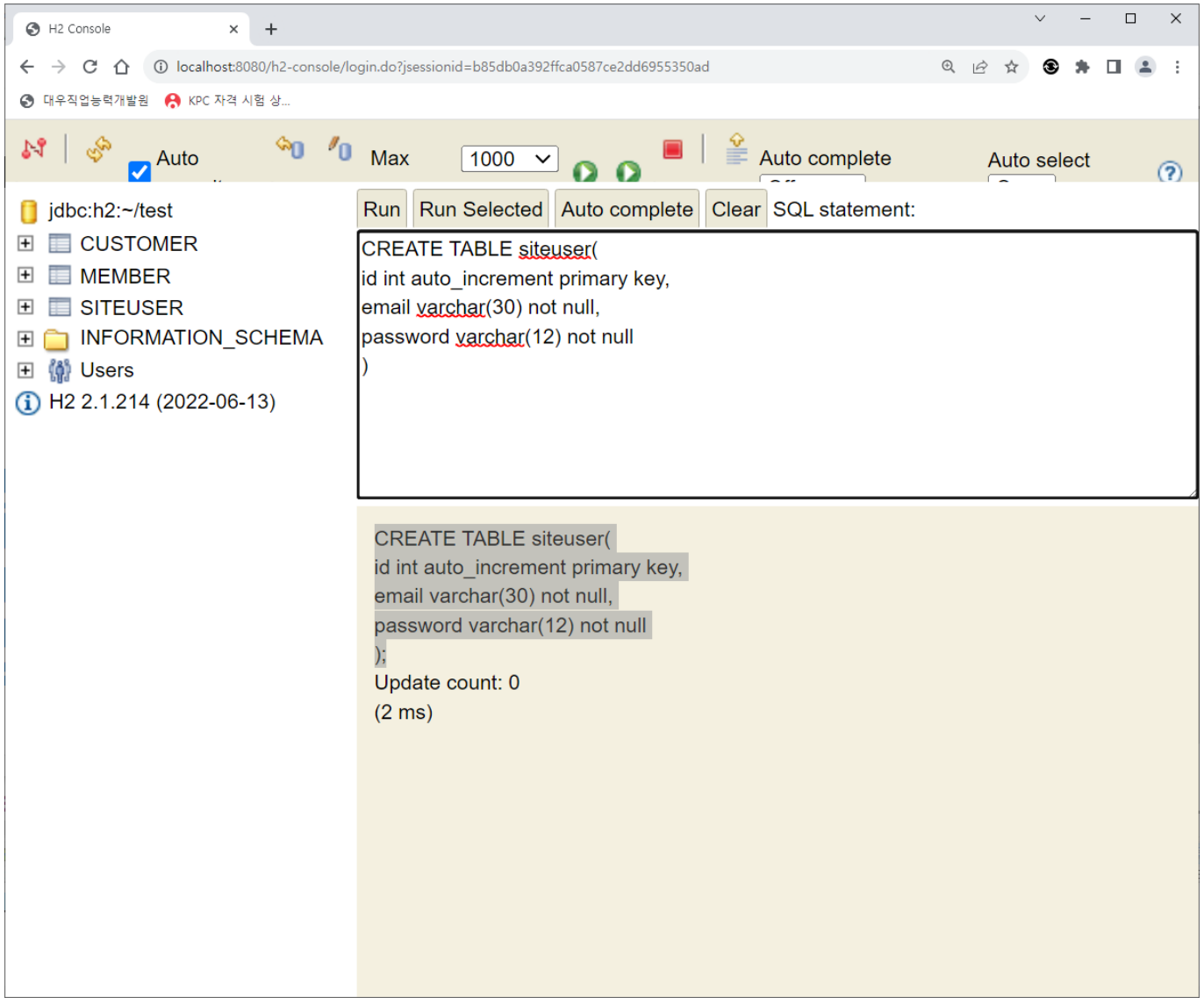
SecurityConfig 파일생성하여 filter 및 url처리

```
@Configuration
@EnableWebSecurity
public class SecurityConfig {


@Bean
SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
//사이트 접속시마다 접속하면 및 제한이 되어 있는 부분을 해제
// http.authorizeHttpRequests();
http.authorizeHttpRequests(
(authorizeHttpRequests)->
authorizeHttpRequests.requestMatchers(
new AntPathRequestMatcher("/**")).permitAll()
).csrf((csrf)->csrf.ignoringRequestMatchers(
new AntPathRequestMatcher("/h2-console/**")))
.headers((header)->header.addHeaderWriter(
new XFrameOptionsHeaderWriter(
XFrameOptionsHeaderWriter.XFrameOptionsMode.SAMEORIGIN)))
;
return http.build();
}
}
```



빨간색 부분까지는 모든 사이트의 접근허용, H2-CONSOLE접근 허용만 가능하고
오른쪽 그림과 같이 페이지가 적용되지 않는 문제가 발생
녹색 박스 코드를 입력하면 문제해결

H2 DB 사이트 사용자 테이블 생성

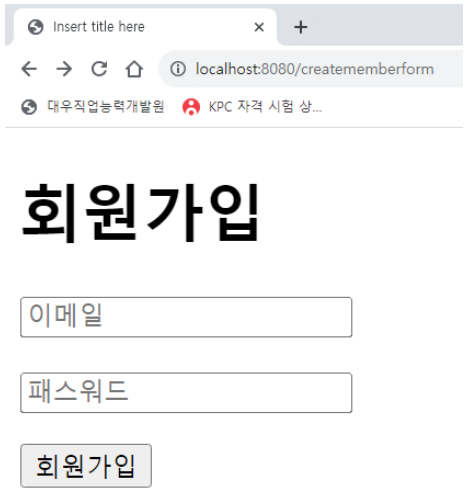



```
CREATE TABLE siteuser(
id int auto_increment primary key,
email varchar(30) not null,
password varchar(12) not null
);
```

회원가입

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<h1>회원가입</h1>
<form th:action="@{/creatememberProc}" method="post">
<input type="email" name="email" id="email" placeholder="이메일"><p>
<input type="password" name="password" id="password" placeholder="패스워드"><p>
<input type="submit" value="회원가입">
</form>
</body>
</html>
```

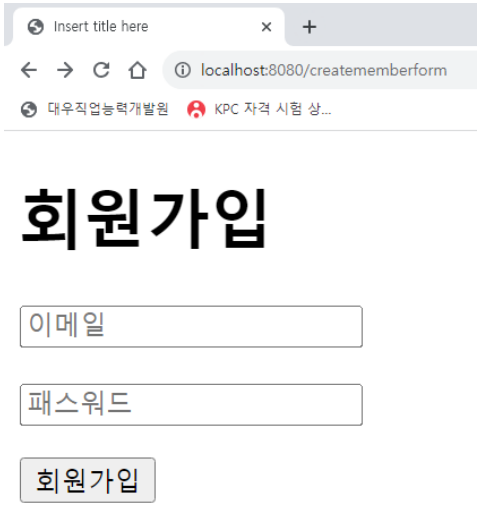
```
@SpringBootApplication
@Controller
public class Demo7Application {

//회원가입 폼
@RequestMapping("/creatememberform")
public String creatememberform() {
return "createmember";
}

//회원가입처리 th:action="@{/creatememberProc}"
@RequestMapping("/creatememberProc")
public String creatememberProc() {
return "";
}
```



주의 사항 :

1) thymeleaf라이브러리 확인
2) Name space확인
3) @로 처리된 사항까지 url처리가 있어야 정상적으로 작동함

회원가입처리1(데이터만 처리)




```java
@RequestMapping("/creatememberProc")
public String creatememberProc(@Validated SiteUser user) {
System.out.println(user.toString());
return "";
}
```

```java
import jakarta.persistence.Entity;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
import lombok.ToString;
```

```java
@Data
@NoArgsConstructor
@AllArgsConstructor
@ToString
//@Entity //만약에 객체명과 DB명이 일치하지 않을 경우 @Table(name="db테이블명")
public class SiteUser {
int id;
String email;
String password;
}
```

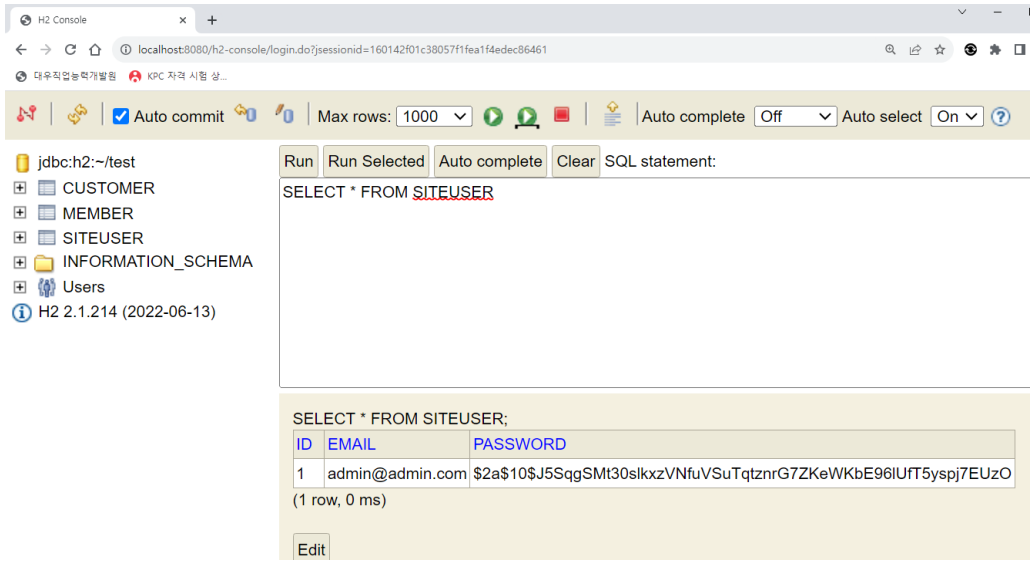
회원가입처리2(jpa활용)

Jpa 데이터베이스 연결

```
@Data
@NoArgsConstructor
@AllArgsConstructor
@ToString
@Entity //만약에 객체명과 DB명이 일치하지 않을 경우 @Table(name="db테이블명")
@Table(name="siteuser")

//객체명이 SiteUser즉 대문자가 이루어진 경우 자동으로 테이블이 데이터베이스에 생성되고 테이블명은 Site_User 자동생성
//만약 테이블이 기존에 있을 경우 @Table로 이름을 지정하여 처리도 가능
public class SiteUser {
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
@Column
int id;
@Column
String email;
@Column
String password;
}
```

```java
Jpa 인터페이스 상속을 받고 사용
import org.springframework.data.jpa.repository.JpaRepository;
public interface SiteUserResposity extends JpaRepository<SiteUser, Integer>{
}
```

```java
Application.java
@SpringBootApplication
@Controller
@RequiredArgsConstructor
public class Demo7Application {


//@Autowired 대신에 @RequiredArgsConstructor 사용하면 @Autowired가 필요없음
//final을 사용하여 객체 유지
final SiteUserResposity resp; //jpa구현

//회원가입처리 th:action="@{/creatememberProc}"
@RequestMapping("/creatememberProc")
public String creatememberProc(@Validated SiteUser user) {
//패스워드 암호화
BCryptPasswordEncoder passencoder=new BCryptPasswordEncoder();
user.setPassword(passencoder.encode(user.getPassword()));
System.out.println(user.toString());
//db에 암화화된 데이터를 입력
resp.save(user);
return "loginform";
}
```



로그인폼생성

```
//로그인 폼
@GetMapping("/loginform")
public String loginform() {
return "loginform";
}
```

loginform.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<h1>로그인</h1>
<form th:action="@{/loginform}" method="post">
<input type="email" name="email" id="email" placeholder="이메일"><p>
<input type="password" name="password" id="password" placeholder="패스
워드"><p>
<input type="submit" value="로그인">
</form>
</body>
</html>
```

Action은 자신의 폼과 같은 url로 해야한다.

SecurityConfig.java

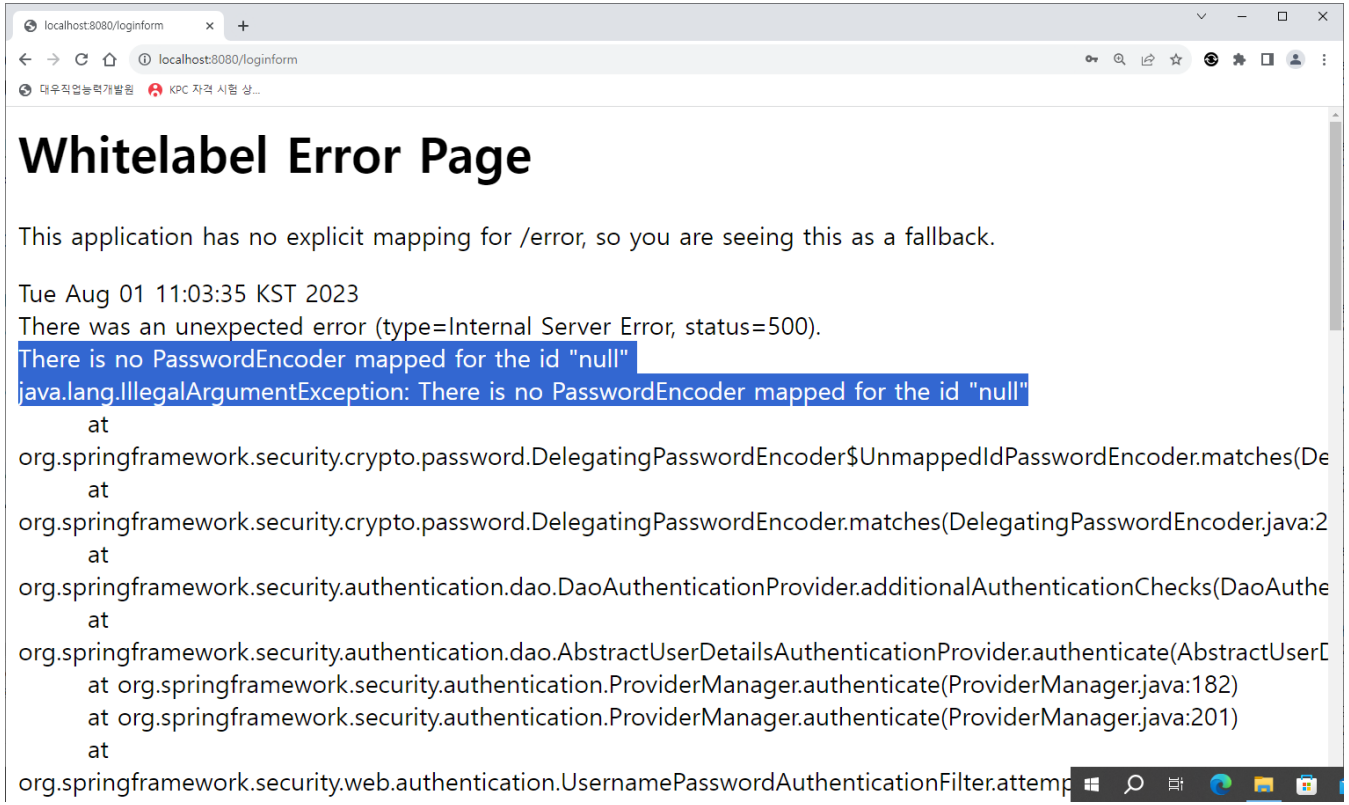
```
//로그인폼에서 로그인 버튼을 누르면 해당 코드가 처리된다.
.formLogin((formLogin) -> formLogin
.loginPage("/loginform")
.defaultSuccessUrl("/")
.usernameParameter("email") //이 값이 자동으로 UserSecurityService객체로 전달
.failureUrl("/loginform"))
;
```

```
//index
@GetMapping("/")
public String index() {
return "index";
}
```

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
index<br>
로그인 성공 페이지
</body>
</html>
```

.usernameParameter("email")

```
package com.example.demo;

import java.util.ArrayList;import java.util.List;import java.util.Optional;
import org.springframework.security.core.GrantedAuthority;import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.userdetails.User;import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Service;import lombok.RequiredArgsConstructor;

@Service
@RequiredArgsConstructor
public class UserSecurityService implements UserDetailsService {

    final SiteUserResposity resp;

    @Override
    public UserDetails loadUserByUsername(String email) throws UsernameNotFoundException {
        System.out.println("email:"+email);
        Optional<SiteUser> ouser = resp.findByEmail(email);

       if (ouser.isEmpty()) {
            throw new UsernameNotFoundException("사용자를 찾을수 없습니다.");
        }
        SiteUser user = ouser.get();

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

        return new User(user.getEmail(), user.getPassword(), authorities);
    }
}
```



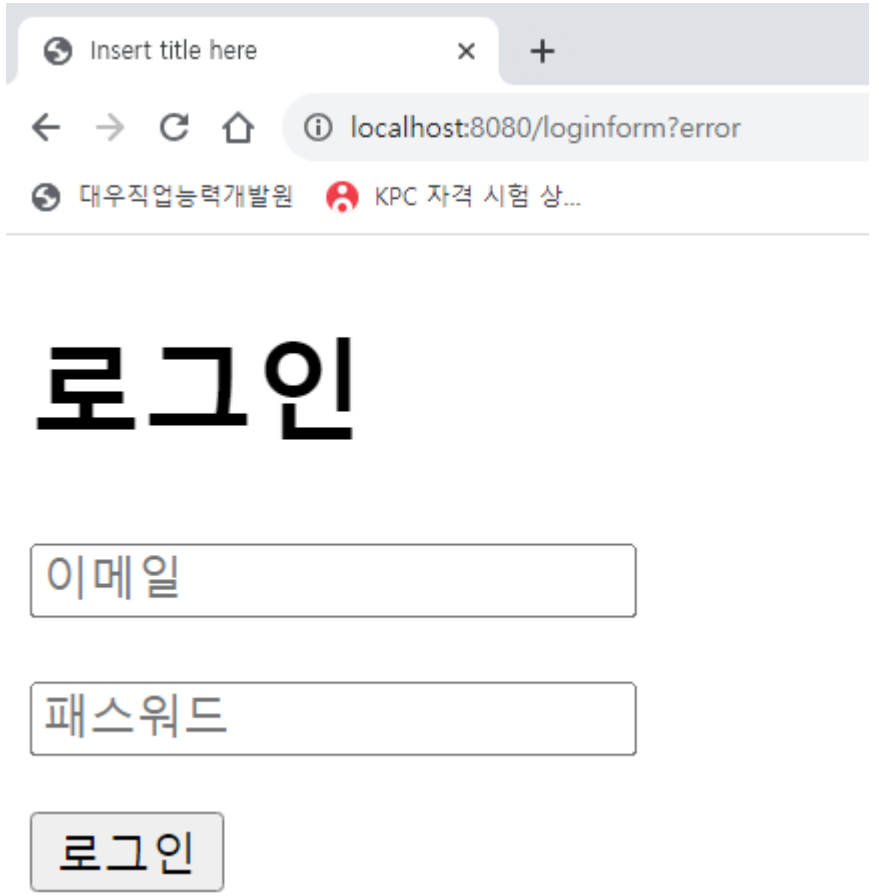
암호화된 코드를 복호화해야하므로 PasswordEncoder가 필요하다.
이 코드를 설정파일에서 bean을 생성해야 한다.

```
@Bean
PasswordEncoder passwordEncoder() {
return new BCryptPasswordEncoder();
}
```

```
.formLogin((formLogin) -> formLogin
.loginPage("/loginform")
.defaultSuccessUrl("/")
.usernameParameter("email") //이 값이 자동으로
UserSecurityService객체로 전달
//.failureUrl("/loginfail"))
);
```

.failureUrl함수를 주석처리하면
Url코드에 error이 붙는다.
Html페이지에 아래코드를 추가하면 로그인이 안될 경우
메시지가 나타나다.

```
<div th:if="${param.error}">
<div >사용자 ID 또는 비밀번호를 확인하세요.</div>
</div>
```



로그아웃처리

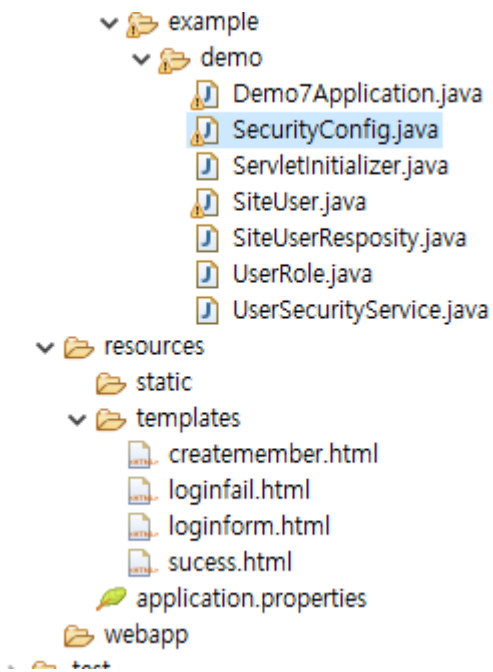
Sec name스페이스를 지원하기 위한 라이브러리

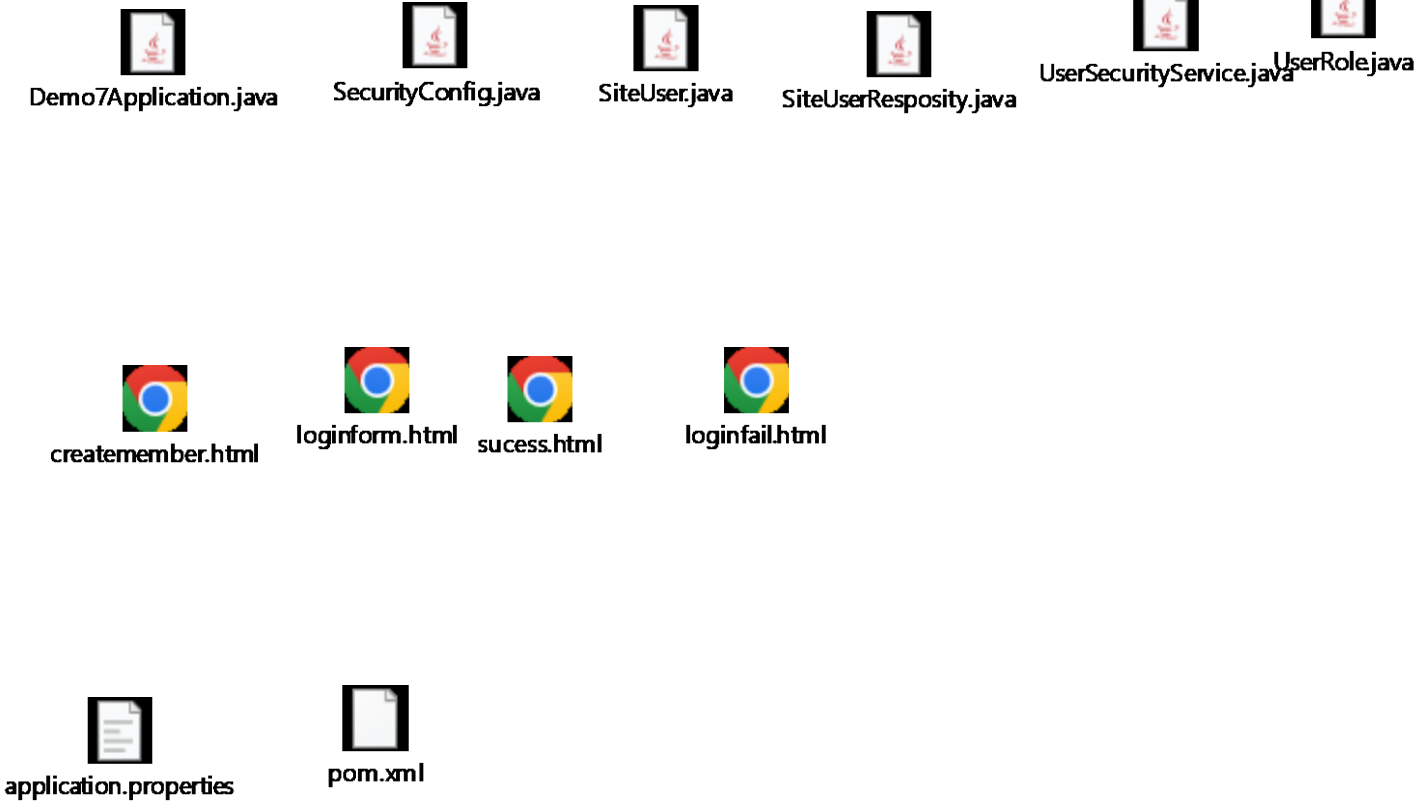
```
<dependency>
<groupId>org.thymeleaf.extras</groupId>
<artifactId>thymeleaf-extras-
springsecurity6</artifactId>
</dependency>
```

```
<html xmlns:th="http://www.thymeleaf.org"
xmlns:sec="http://www.thymeleaf.org/extras/spring-security">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
로그인 하지 않은 경우[<a sec:authorize="isAnonymous()" th:href="@{/loginform}">로그인</a>]<p>
로그인 성공한 경우[<a sec:authorize="isAuthenticated()" th:href="@{/logout}">로그아웃</a>]<p>
<span sec:authentication="name"></span>

</body>
</html>
```

컨트롤러에 /logout을 설정할 필요가 없으며
설정파일에 아래코드를 추가하면 설정파일이 실행되고(AntPathRequestMatcher())
로그아웃 되었을때 페이지 이동, 세션을 해제하게 된다.

```
.logout((logout)->logout
.logoutRequestMatcher(new
AntPathRequestMatcher("/logout"))
.logoutSuccessUrl("/loginform")
.invalidateHttpSession(true)
);
```


example
demo
Demo7Application.java
SecurityConfig.java
ServletInitializer.java
SiteUser.java
SiteUserResposity.java
UserRole.java
UserSecurityService.java
resources
static
templates
createmember.html
loginfail.html
loginform.html
sucess.html
application.properties
webapp


Demo7Application.java
SecurityConfig.java
SiteUser.java
SiteUserResposity.java
UserSecurityService.java
UserRole.java
createmember.html
loginform.html
sucess.html
loginfail.html
application.properties
pom.xml