**OAuth란?**

OAuth(Open Authorization)는 토큰 기반의 인증 및 권한을 위한 표준 프로토콜이다. OAuth와 같은 인증 프로토콜을 통해 유저의 정보를 페이스북, 구글, 카카오 등의 서비스에서 제공받을 수 있고 이 정보를 기반으로 어플리케이션 사용자에게 로그인이나 다른 여러 기능들을 손쉽게 제공할 수 있다.

자세한 내용은 [여기](https://opentutorials.org/course/3405)를 참조한다.

**스프링 부트로 OAuth2를 통한 로그인 기능**

스프링 부트로 OAuth2를 통한 로그인 기능을 제공할 수 있다.

여기서는 페이스북, 구글, 카카오, 네이버에서 제공하는 정보를 통해 사용자가 웹페이지에 손쉽게 로그인하는 기능을 구현할 것이다. 스프링부트로 해당 기능을 구현하기 전에 먼저 위 4가지 서비스의 개발자 사이트에 들어가서 OAuth 인증을 위한 설정을 해야한다.

[구글 디벨로퍼 콘솔](https://console.cloud.google.com/apis/dashboard?pli=1)

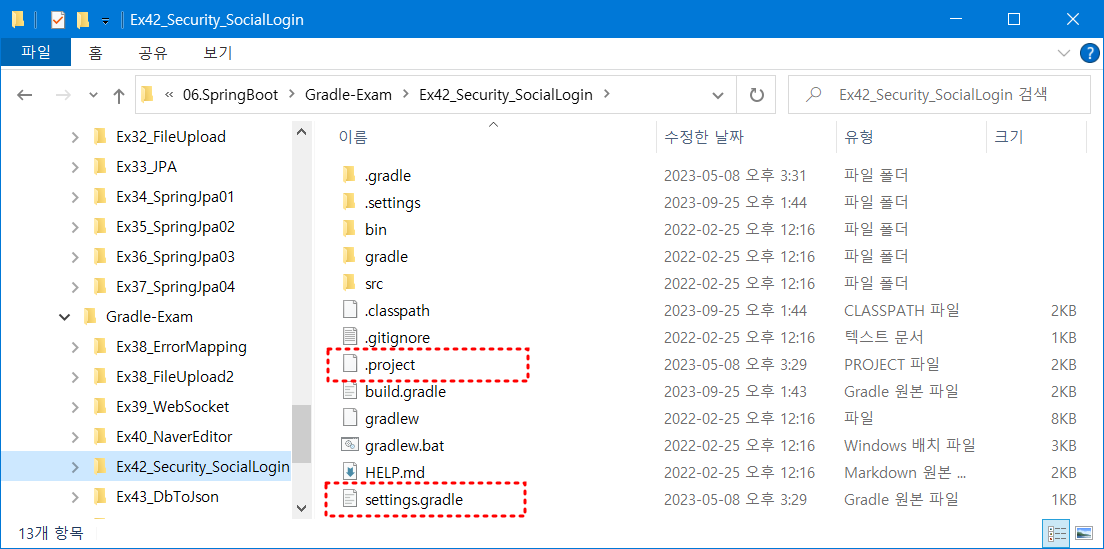
[카카오 개발자 페이지](https://developers.kakao.com/)

[네이버 개발자 센터](https://developers.naver.com/main/)

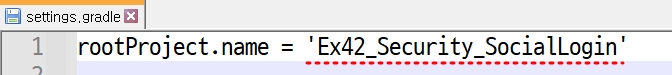
**프로젝트 구성**

Ex29\_Security\_tablibs 프로젝트를 복사하여 이름을 Ex42\_Security\_SocialLogin 으로 수정한다.

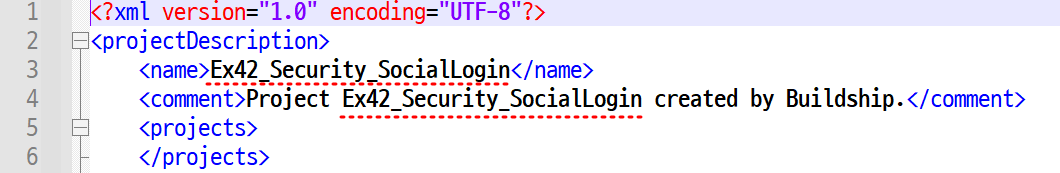
폴더 이름 변경



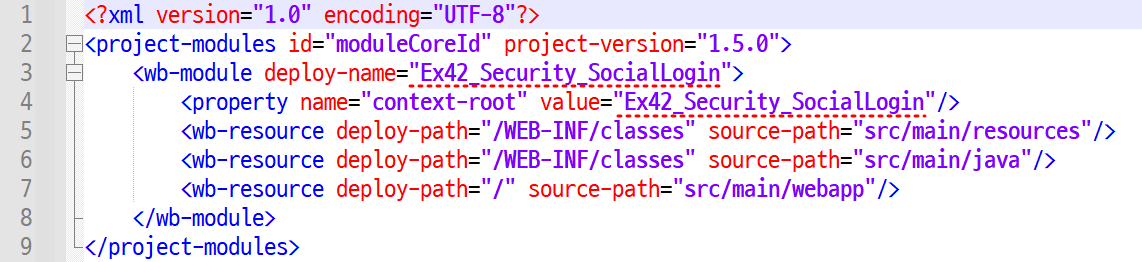
settings.gradle 파일안의 내용 변경



.project 파일안의 내용 변경

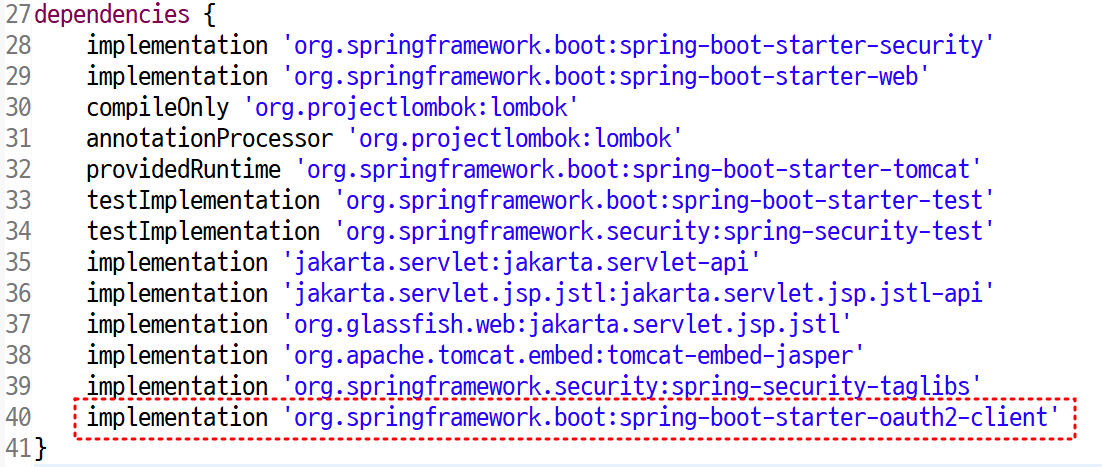


.settings 폴더의 org.eclipse.wst.common.component 파일을 열어서 다음처럼 내용 수정

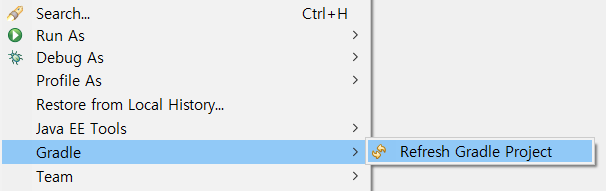


build.gradle 수정

* 다음 내용과 비교하여 빠진 부분이 있다면 입력하여 추가하도록 한다.

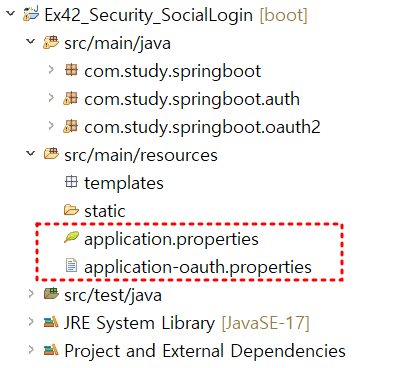


다음과 같이 메뉴를 선택하여 수정한 내용이 적용되도록 한다.



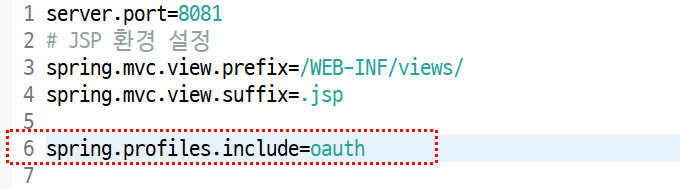
속성 파일 추가

* application.properties 를 복사하여 붙여넣고 이름을 application-oauth.properties 로 수정한다.



application.properties 내용을 다음과 같이 수정한다.

* application-oauth.properties 파일을 속성 파일로 추가한다고 지정



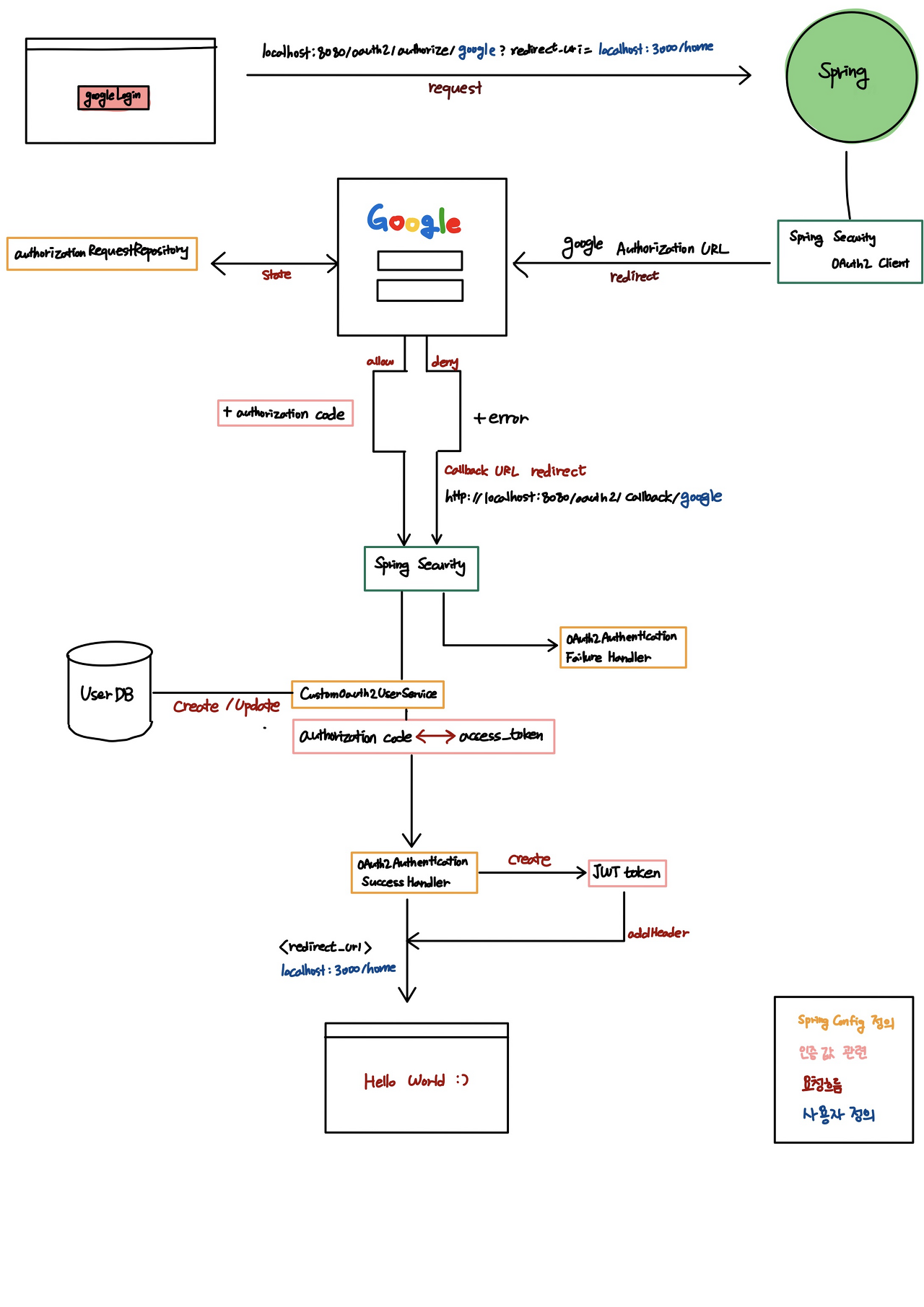
application-oauth.properties 내용 작성

다음의 내용을 복사하여 붙여 넣기 한다.

* 각 계정별 client-id, clietnt-secret 는 본인 계정의 것으로 입력한다.
* Google, Facebook 은 스프링에 기능이 내장되어 있기 때문에 이정도의 정보만 주면 된다.
* Facebook의 경우 provide로 추가 정보를 지정해야 사진을 가져올 수 있다.
* Kakao, Naver 는 기본으로 지원이 되지 않기에 모든 항목에 대한 정보를 제공해야 한다.
  + 관련 값들은 개발자 센터에서 모든 정보가 제공되고 있다.
  + 이런 식으로 어떤 사이트라도 OAuth 로그인을 구현할 수 있다.

| # Google  spring.security.oauth2.client.registration.google.client-id=본인클라이언트아이디  spring.security.oauth2.client.registration.google.client-secret=본인시크릿키  spring.security.oauth2.client.registration.google.scope=profile,email  # Facebook  spring.security.oauth2.client.registration.facebook.client-id=본인클라이언트아이디  spring.security.oauth2.client.registration.facebook.client-secret=본인시크릿키  spring.security.oauth2.client.provider.facebook.user-info-uri=https://graph.facebook.com/me?fields=id,name,email,picture  # Kakao  spring.security.oauth2.client.registration.kakao.authorization-grant-type=authorization\_code  spring.security.oauth2.client.registration.kakao.client-id=본인REST API 키  spring.security.oauth2.client.registration.kakao.redirect-uri={baseUrl}/login/oauth2/code/{registrationId}  spring.security.oauth2.client.registration.kakao.scope=profile,account\_email  spring.security.oauth2.client.registration.kakao.client-authentication-method=POST  spring.security.oauth2.client.registration.kakao.client-name=Kakao  spring.security.oauth2.client.provider.kakao.authorization-uri=https://kauth.kakao.com/oauth/authorize  spring.security.oauth2.client.provider.kakao.token-uri=https://kauth.kakao.com/oauth/token  spring.security.oauth2.client.provider.kakao.user-info-uri=https://kapi.kakao.com/v2/user/me  spring.security.oauth2.client.provider.kakao.user-name-attribute=kakao\_account  # Naver  spring.security.oauth2.client.registration.naver.client-id=본인클라이언트아이디  spring.security.oauth2.client.registration.naver.client-secret=본인시크릿키  spring.security.oauth2.client.registration.naver.redirect-uri={baseUrl}/login/oauth2/code/{registrationId}  spring.security.oauth2.client.registration.naver.authorization-grant-type=authorization\_code  spring.security.oauth2.client.registration.naver.scope=name,email,profile\_image  spring.security.oauth2.client.registration.naver.client-name=Naver  spring.security.oauth2.client.provider.naver.authorization-uri=https://nid.naver.com/oauth2.0/authorize  spring.security.oauth2.client.provider.naver.token-uri=https://nid.naver.com/oauth2.0/token  spring.security.oauth2.client.provider.naver.user-info-uri=https://openapi.naver.com/v1/nid/me  spring.security.oauth2.client.provider.naver.user-name-attribute=response |
| --- |

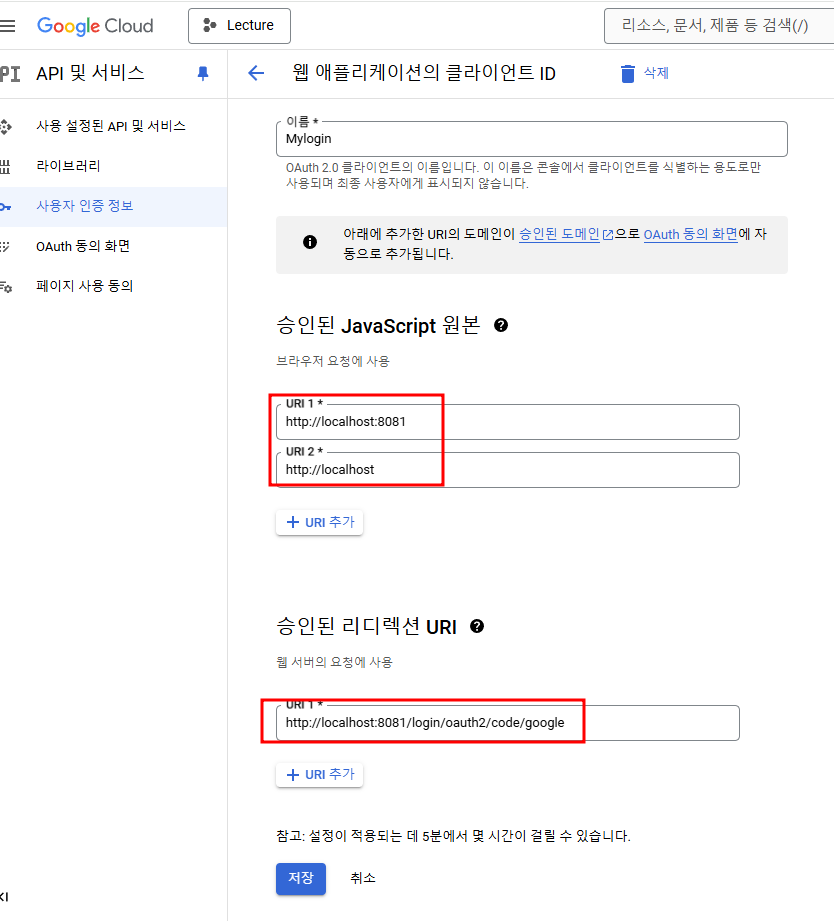
앞에서 작성한 정보들은 다음 플로우에서 사용된다. (참고)



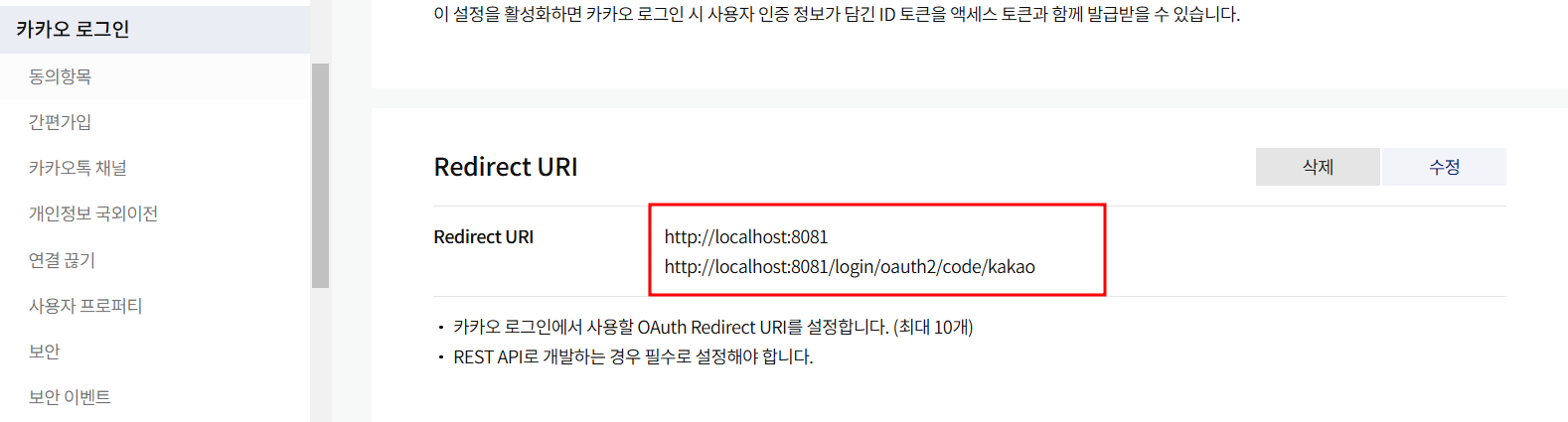
이제 OAuth 로그인을 지원하기 위해 다음과 같이 패키지를 추가하고 클래스들을 추가한다.

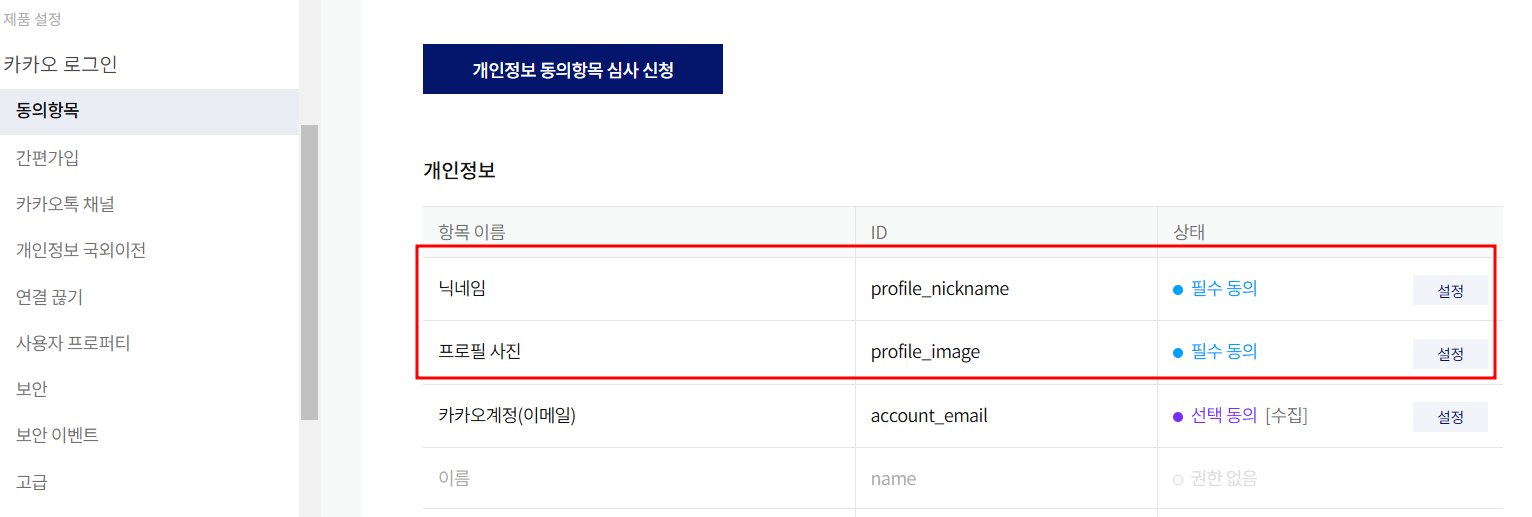
* 한 번 만들면 그대로 사용하는 것들이기 때문에 수업용 자료실의 자료를 이용한다.
* 이 부분은 Social Login 이 사이트가 추가될 때 수정이 필요하다. 그 외에는 수정없이 사용한다.

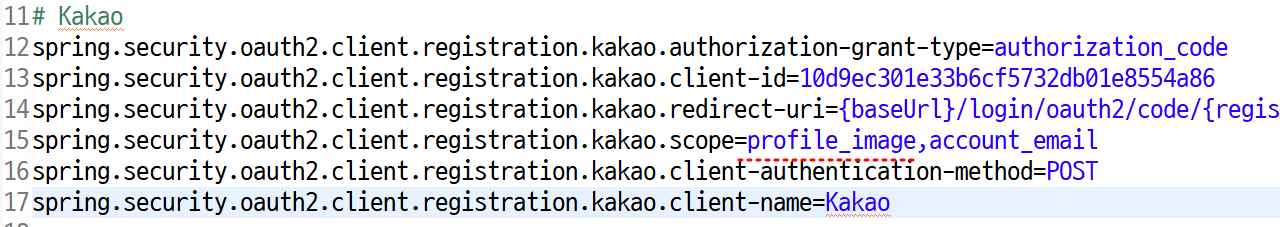
구글 설정에서는 다음을 추가한다.



카카오 설정에서는 다음을 확인한다.







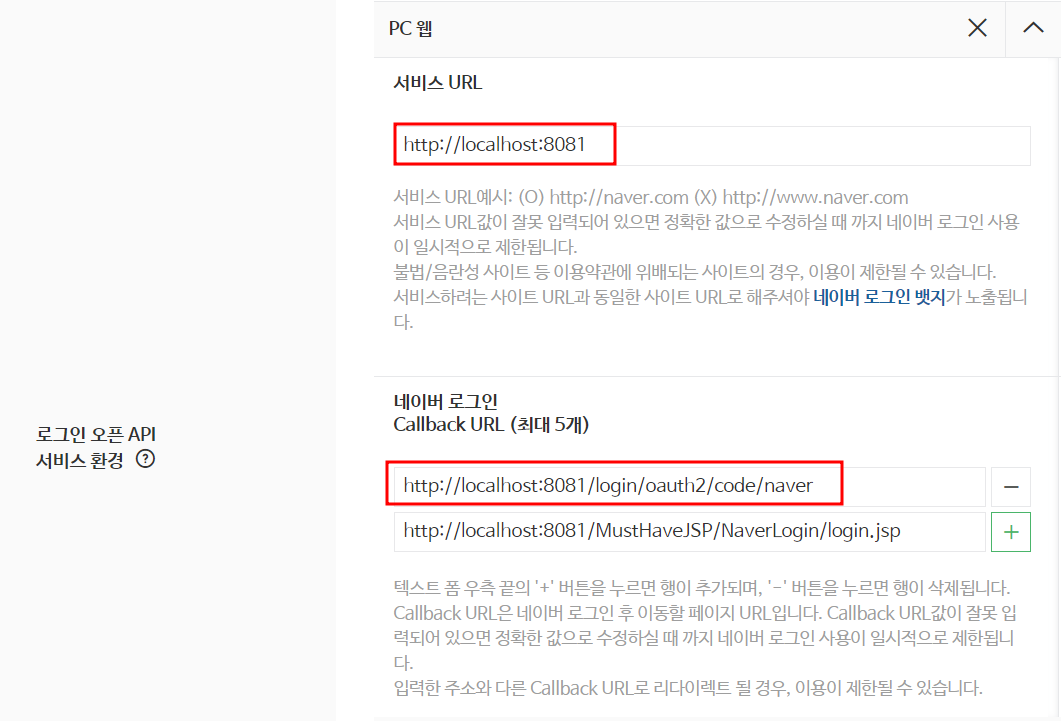
Naver 세팅에서 다음을 확인한다.

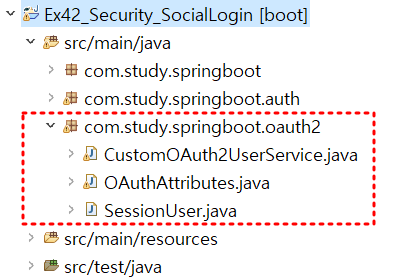


중간 부분에 [ 프로필 사진 ] 체크 확인 …



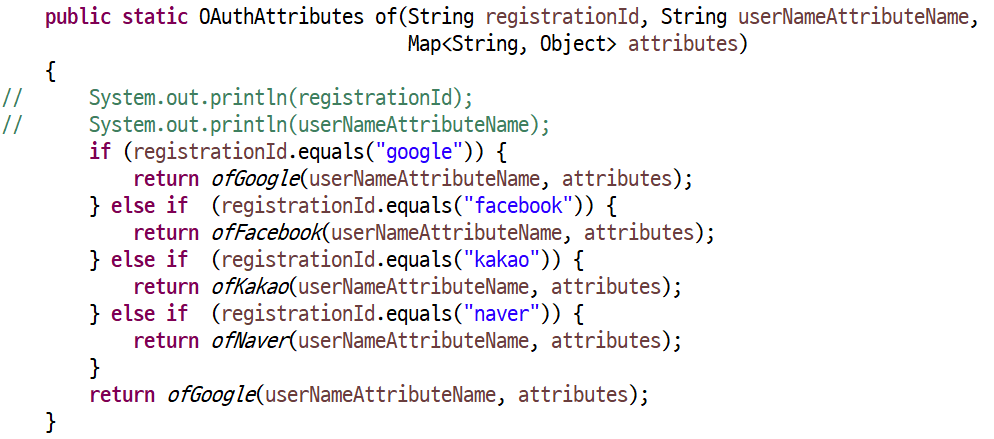
하단 부분에 다음을 확인



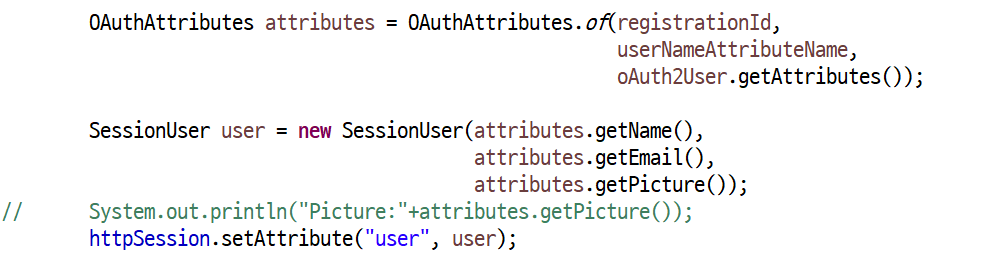


| SessionUser.java | 세션 정보를 객체로 저장하기 위한 bean |
| --- | --- |
| OAuthAttributes | 각 SocialLogin 의 응답 데이터를 파싱하여 공통 요소로 만들어 반환 |
| CustomOAuth2UserService | 실제 SocialLogin 을 수행하는 클래스  로그인에 성공하면 OAuthAttributes로 부터 받은 값으로 우리 프로그램상에서 사용할 데이터를 세션에 추가한다. |

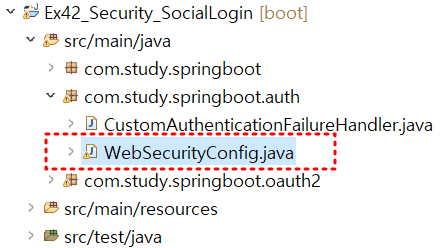
OAuthAttributes 에서 각 Social Login 성공으로 온 데이터 파싱하여 정리

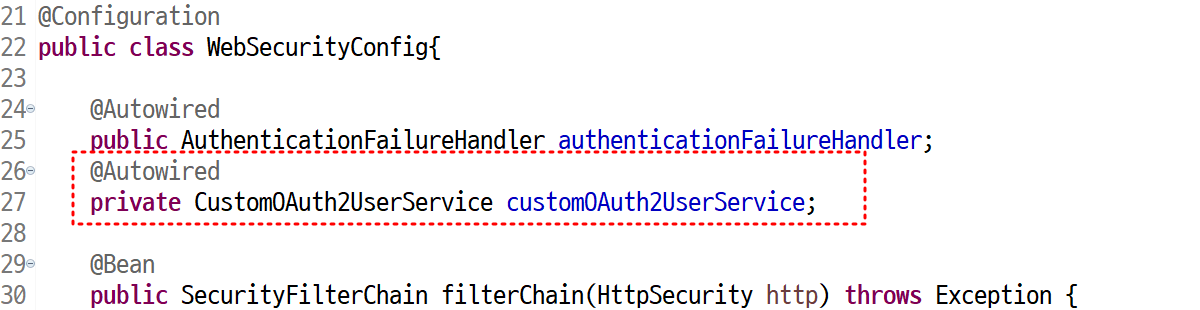


CustomOAuth2UserService 에서 세션에 필요한 데이터 세팅

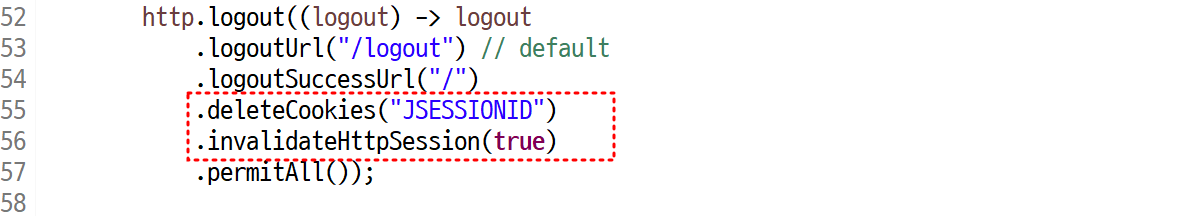


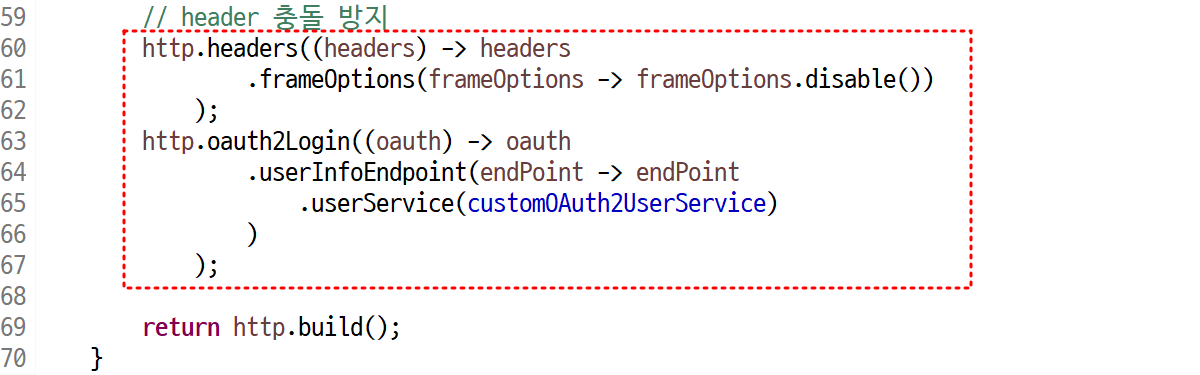
기존 WebSecurityConfig 에 OAuth 로그인을 사용하기 위한 설정 추가





…





loginForm.jsp 수정

Social Login 을 위한 링크를 다음과 같이 추가한다.



application-oauth.properties 에서

spring.security.oauth2.client.registration.google.xxx 같이 등록한 내용에서 google 이 부분을

<a href=”/oauth2/authorization/google> 로 지정하면 된디ㅏ.

welcome2.jsp 수정

페이스북의 경우는 세션 악용을 보호하기 위해 redirection url 에 몇 가지 데이터를 추가하는데 다음과 같이 자바스크립트를 이용하여 제거할 수 있다.



