**개발 일지**

**Frontend : svelte**

**Backend : spring boot, PostgreSQL, Docker**

**DB : local**

**Editor : Intellilj, VSCode**

1. 보통 서버를 띄우기 위해 Tomcat을 사용하였으나, 웹만 따로 분리하여 개발을 하게 되는 경우에는 웹서버를 따로 띄어줘야 한다. 따라서 백엔드단에서는 spring boot에 내장된 Tomcat Server로 켜지는 거고 프론트엔드 단에서는 nodejs와 같이 웹서버로 띄어주는 것이다.
2. Svelte port 변경하기
   1. Package.json에서 "start": "sirv public --no-clear --port 5050"처럼 port옵션 붙여주고 설정해주면 된다.  
      (<https://stackoverflow.com/questions/57578775/how-to-change-the-default-port-5000-in-svelte>)
3. Sveltkit port 변경하기
   1. Package.json에서 “dev”구간에 “--port 5050” 옵션을 추가해주면 된다.
   2. Vite.config.js에서도 가능하다. (<https://rodneylab.com/sveltekit-port/>)
4. Svelte와 Sveltkit의 차이는 무엇일까?  
   ( <https://dev.to/danawoodman/when-to-use-svelte-vs-sveltetkit-vs-sapper-4o6a>)  
   현재는 Sveltkit도 1.0 정식 버전이 배포된 상태이다.
5. COSR설정을 해주기 위해서는 백엔드에서 처리를 해줘야한다. 우선 해당 링크를 통해 힌트를 얻을 수 있었다.  
   (<https://asecurity.dev/entry/Node-Svelte-CORS-%ED%95%B4%EA%B2%B0%EB%B0%A9%EC%95%88>)
6. CORS를 백엔드에서 처리해주는 방법이다. 우리의 경우 spring boot에서 설정해준다.  
   (<https://dev-pengun.tistory.com/entry/Spring-Boot-CORS-%EC%84%A4%EC%A0%95%ED%95%98%EA%B8%B0>)
7. Svelte의 경우 ssr과 csr이 자동 적용되어 있어 csr=false옵션을 넣어줘야 막을 수 있다.
8. onMount()메서드는 csr이 block된 상태에서는 사용할 수 없으므로 Post혹은 Get메서드를 이용하여 fetch를 통해 가져오도록 한다. (async/await으로 동기식으로 처리 필요)  
   (<https://eternaldev.com/blog/how-to-use-svelte-onmount-lifecycle-method/>)
9. SSR로 처리할 경우 await fetch()가 지나고 나서 화면을 뿌리게 되고 데이터를 가져오지 못한다. 또한 버튼 클릭 이벤트와 같이 한번 화면을 뿌리고 나서 동작이 안되는 경우도 존재한다. 어떻게 처리해야될까…
   1. CSR과 SSR을 조합해서 사용해야된다.   
      (<https://www.dioneuty.net/blog/view?idx=31>)
10. Svelte에서는 CSR로 실행할지 SSR로 실행할지에 대해서 +page.server.js로 하냐 +page.client.js로 하냐에 따라 다르다. page.js에서 사용할 경우 export const csr = false 옵션을 넣어 사용해야한다.
    1. SSR 데이터 뽑아와서 UI에 출력한 방법(<https://www.youtube.com/watch?v=jXtzWMhdI2U>)
11. onMount에 관해   
    The one you'll use most frequently is onMount, which runs after the component is first rendered to the DOM.  
    It's recommended to put the fetch in onMount rather than at the top level of the <script> because of server-side rendering (SSR). With the exception of onDestroy, lifecycle functions don't run during SSR, which means we can avoid fetching data that should be loaded lazily once the component has been mounted in the DOM.  
    => 요약하자면 onMount내부에 fetch를 사용하라는 것.  
    그렇지 않을 경우 SSR로 동작하여 라이프사이클이 돌지 않게 된다.  
    SSR의 경우 서버에서 랜더링을 거쳐 주기 때문에 느린 동작을 막을 수 있다.  
    마운트란 컴포넌트가 화면에 출력되는 것을 말한다.
12. CSR과 SSR의 차이(<https://proglish.tistory.com/216>)
    1. SEO는 검색 최적화 엔진으로 구글에 최상위 상단에 띄어주는 광고 같은 존재이다.
    2. 만일 CSR처럼 프론트로부터 빈화면을 받아 SEO를 실행해야 될 경우 시간이 상당히 오래걸리고 비효율적이다.
    3. 하지만 SSR처럼 프론트로부터 모든 화면을 받은 상태에서 SEO를 실행하는게 더 효율적이라 볼 수 있지 않을까 싶다.
    4. CSR vs SSR 선택은?  
       (<https://velog.io/@kysung95/SSR-vs-CSR-%EB%8B%B9%EC%8B%A0%EC%9D%98-%EC%84%A0%ED%83%9D%EC%9D%80-feat.-ssg>)
    5. 그렇다면 CSR과 SPA는 같은 걸까?  
       => CSR과 SPA를 보면 둘다 빈 html문서를 받아 클라이언트가 비동기적으로 처리한다는 점에서 비슷하다.  
       (<https://velog.io/@hlna0308/SPA%EC%99%80-CSR%EC%9D%80-%EA%B0%99%EC%9D%84%EA%B9%8C>)
13. svelte 라이프사이클에 대해서 (<https://freeseamew.gitbook.io/svelte/4./lifecycle)>)
    1. tick에 대하여 (https://learn.svelte.dev/tutorial/tick)  
       해당 예시와 7번에 설명한 글을 참고하면 어떤 특정 메서드에서 동작하고 컴포넌트가 Dom으로부터 라이프 사이클이 지나고 나면 특정 메서드에 호출된 명령어는 묻힐 수가 있다.  
       따라서 컴포넌트가 update된 후에 특정 메서드에 있는 특정 구간을 마지막으로 동작할 수 있도록 해주는 것이 tick이라 보면 된다.
    2. Tick과 afterUpdate의 사용 용도의 차이라면 tick은 하나의 메서드 내부에서 내가 원하는 위치, 특정 구간에 대해 지정을 할 수 있고 afterUpdate는 하나의 컴포넌트단위로 동작할 수 있도록 한다.
14. Stores => 서로 관계없는 컴포넌트끼리 같은 데이터를 접근해야 할 경우 사용  
    (<https://beomy.github.io/tech/svelte/store/>)
    1. Writable subscribe만으로는 구독취소가 되지 않는 미묘한 버그가 존재한다.
    2. 따라서 구독을 해지하는 방법으로는 unsubscribe를 받아 onDestroy로 해재하는 방법도 존재하지만, $로 바로 해지할 수 있는 방법이 존재하며 주의사항이 따른다. 위 링크를 참조하자)
    3. Derived를 사용하면, 존재하는 store를 이용하여 새로운 store생성이 가능하다. 즉, 기존에 존재하는 값을 내 입맛에 맞게 가공하여 사용할 수 있다.
15. Import (컴포넌트 이름) from ‘컴포넌트경로’ (import의 사용법)
16. SvelteKit이란?
    1. Svelte ui framework에서 SEO를 위한 SSR처리등 더 다양한 기능을 얹힌 프레임워크로 React로 따지면 Nextjs와 비슷하며, Vue로 따지면 Nuxtjs와 비슷하다.
    2. Hydrate에 대해서 rendering된 페이지를 받고나서 한번 더 렌더링을 통해 자리를 찾아가는 과정이라 보면된다. ( <https://helloinyong.tistory.com/315>)
17. SSR로 로딩 시점에 데이터 가져와 UI에 출력하는 법
    1. js파일에서 load메서드를 사용한다.
    2. SSR로만 한정 짓고 싶으면 +page.server.js파일로 하여 사용하거나, +page.js파일에서 export const csr =false로 둔다.
    3. 만약 +page.server.js에 있는 메서드를 svelte파일에서 메서드 호출 시 SSR코드를 CSR코드로 import할 수 없다는 에러가 발생한다. 따라서 CSR원리로 메서드를 호출하고 싶으면 +page.js에 메서드를 만들어 사용하거나 +page.client.js로 만들어 사용해도 된다.
18. Svelte의 Routind은 데릭터리 경로에 있는 +page.svelte에 의해서 구분된다.  
    (ex. /routes/abou ㅌt/+page.svelte파일을 만들면 url패턴은 about이 되는 것)
19. +layout.js는 Sibling과 Child관계에 대해서만 적용이 되고 부모 레이아웃은 적용되지 않는다.
    1. Directory 경로에 따라 라우팅이 설정되는데 blog디렉터리를 만들고 하위 디렉터리명이 [slug]일 경우 blog/xxx 일 때 “xxx”에 어떤 값이 들어가도 blog/[slug]에 있는 +page.svelte에 대한 레이아웃이 뜬다.
20. Routing, Loading, Rendering 관계
    1. Routing: 들어오는 요청에 맞는 경로를 파악하여 UI를 띄운다.
    2. Loading: 라우트에 필요한 데이터를 가져온다.
    3. Rendering: HTML을 만들거나 DOM을 업데이트(브라우저 안에서)한다.
21. Loading
    1. Load()는 페이지가 열릴 때 자동으로 실행되는 메서드이다. 따라서 load()안에 load({params})와 같이 넣어두면 params에 해당 디렉터리명(상위빼고)만 들어간다.  
       (<https://learn.svelte.dev/tutorial/page-data>)
    2. load메서드에 ({fetch})로 들어가면 동작 원리가 어떻게 될까..
    3. layout.server.js에서 load()를 통해 데이터를 반환하고 이걸 하위 디렉터리에서 data를 쓰고싶으면 하위 디렉터리에서 layout.svelte를 만들어 export let data로 받는다. 하위 디렉터리에 page.server.js에서 반환하는 data에는 layout의 값을 포함하여 갖고 layout.svelte에서는 상위에서 layout.server.js 반환한 값만 가지고 있게 된다.
    4. 텍스트이(가) 표시된 사진

       자동 생성된 설명
    5. layout에서 page.svelte의 data를 가져다 쓰고 싶을 때는 아래와 같이 사용  
       텍스트이(가) 표시된 사진

       자동 생성된 설명
    6. Default actions cannot coexist with named actions.  
       => action 메서드에 default로 선언할 경우 delete나 create와 같은 다른 명을 적을 수가 없으므로 주의하자
    7. SvelteKit takes us to an unfriendly-looking error page. On the server, we see a 'todos must be unique' error, but SvelteKit hides unexpected error messages from users because they often contain sensitive data.  
       => sveltekit은 예상치 못한 에러메세지를 숨기는데 중요한 정보가 있을 수 있기 때문이다.
    8. 텍스트이(가) 표시된 사진

       자동 생성된 설명위와 같이 정상 출력도 보낼 수 있다.  
       텍스트이(가) 표시된 사진

       자동 생성된 설명
22. Svelte Error처리(<https://learn.svelte.dev/tutorial/error-basics>)
    1. Throw new Error(“msg”);처리를 하면 500에러가 뜨면서 html에 보이는 에러내용은 “Internal Error”로 발생한다. 문서를 읽어보면 500으로 발생하는 에러의 경우 민감한 정보 출력으로 인해 막아뒀다고 한다.
    2. Throw error(420, ‘Enhance your calm’)에러의 경우 420으로 발생하면 에러 메시지가 html에 출력된다.
    3. +error.svelte를 활용하여 error처리를 커스텀 할 수 있으며 동작방식은 layout과 동일하다.
    4. Static error page라 하여 최상위 루트 레이아웃에서 데이터를 로딩하는 시점이나 에러페이지 렌더링하는 과정에서 발생하는데, src/error.html파일로 커스텀 할 수 있다.
    5. redirect기능 => throw redirect(307, ‘경로’); (<https://learn.svelte.dev/tutorial/redirects>)
23. Page Option
    1. Window.innerWidth와 같은 브라우저의 화면 크기를 구하는 상황(CSR)이 필요한 경우에는 SSR을 막아주는 옵션을 넣어줘야 한다. Svelte는 ssr이 default값으로 들어가기 때문이다.  
       => csr이 true로 되어있어도 위 명령어는 html단에서 자동적으로 실행되는 이유에선지 ssr을 막아줘야만 돌아간다.
    2. 브라우저에서 클릭과 같이 상호작용이 필요한 동작들에 대해 클라이언트로부터 js동작이 가능하게 해야하지만 CSR을 false로 둬버리면 위와 같은 상호작용이 더는 불가능해진다.
    3. Pre-rendering(SSG) : Static Site Generator  
       => 정적 사이트로 누가 접속하든 항상 동일한 내용을 보여주는 웹사이트를 만드는 곳에서 최적화된 방법이라 생각한다.  
       SSG는 웹페이지를 미리 만들어둔 상태에서 요청이 들어면 그대로 준다.  
         
       얼핏 보면 SSR과 동일하게 느껴지지만 SSG는 빌드 시점에 모든 SSG웹사이트 전체를 미리 만들어 둔다는 것이고 SSR은 사용자 요청이 들어왔을 때만 해당 웹사이트를 만들어서 준다고 생각하면 된다.
    4. SSR이 어째서 수시로 업데이트 되는 곳에서 유리하다는 건지 도저히 이해가 안간다…(<https://callmedevmomo.medium.com/%EC%9B%B9-%EB%A0%8C%EB%8D%94%EB%A7%81%EC%9D%98-2-%EA%B0%80%EC%A7%80-%EC%A0%91%EA%B7%BC-csr-ssr-%EB%B6%80%EC%88%98%EA%B8%B0-ba9e2b9c0d6b>) 해당 링크를 통해 약간의 이해는 되긴했다…SSR이 빠르게 화면에 적용되기 때문이고 CSR은 말그대로 사용자와 서버 간의 상호작용을 고려한 거에 대해서 장점이 있는 거고….
    5. (<https://www.daleseo.com/spa-ssg-ssr/>)
    6. (<https://miracleground.tistory.com/entry/SSR%EC%84%9C%EB%B2%84%EC%82%AC%EC%9D%B4%EB%93%9C-%EB%A0%8C%EB%8D%94%EB%A7%81%EA%B3%BC-CSR%ED%81%B4%EB%9D%BC%EC%9D%B4%EC%96%B8%ED%8A%B8-%EC%82%AC%EC%9D%B4%EB%93%9C-%EB%A0%8C%EB%8D%94%EB%A7%81>)
24. MPA와 SPA에 대해
    1. MPA는 특정 요청을 post로 쐈을 때 변경된 데이터만 바꿔서 적용하면 되는데 바뀐 데이터를 가지고 페이지를 다시 통째로 만들어 응답을 주는 것이다.
    2. SPA는 위에서 바뀐 부분에 대해서만 응답을 주고 해당 부분 데이터만 변경된다.
    3. (<https://velog.io/@yejine2/SPASingle-Page-Application-VS-MPA>)
    4. (<https://callmedevmomo.medium.com/%EC%9B%B9-%EB%A0%8C%EB%8D%94%EB%A7%81%EC%9D%98-2-%EA%B0%80%EC%A7%80-%EC%A0%91%EA%B7%BC-csr-ssr-%EB%B6%80%EC%88%98%EA%B8%B0-ba9e2b9c0d6b>)
    5. trailingSlash => url 끝에 ‘/’가 있는 것  
       export const trailingSlash = ‘always’, ‘ignore’, ‘never’ 3가지의 옵션이 있는데 always는 뒤에 항상 /가 붙는 것이고 ignore은 그냥 있으면 있고 없으면 없고,,, never는 항상 /를 빼준다.
25. SvelteKit에서 BootStrap설치하는 법
    1. App.html에   
       <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">
    2. 해당 링크는 좀 색다르게 사용하긴하는데 불편할듯해서 위 방법이 편하다.  
       (<https://frontendshape.com/post/how-to-install-bootstrap-5-in-svelte>)
26. SvelteKit의 tutorial로 개발하는 과정에 같은 코드임에도 에러가 발생한다면 jsconfig.json에서 checkJS를 false로 바꿔준다.(TypeScript를 사용하지 않겠다는 의미이다.)
27. Layout을 구성할 때 바뀌는 부분에 대해서는 <slot/>으로 처리할 경우 라우터를 통해 가르키는 페이지로 뜬다.(ex. href태그로 불러들인 링크)
28. Footer를 하단에 고정시키는 법  
    (<https://jineecode.tistory.com/103>)
29. Docker-compose로 postgreSQL설치 (<https://afsdzvcx123.tistory.com/entry/Docker-PostgreSQL-Docker-Compose-%ED%8C%8C%EC%9D%BC-%EC%9E%91%EC%84%B1>)  
    (<https://xeppetto.github.io/%EC%86%8C%ED%94%84%ED%8A%B8%EC%9B%A8%EC%96%B4/WSL-and-Docker/15-Docker-PostGreSQL/>)
30. PostgreSQL의 경우 Docker에서 설치하고 나면 port가 계속 5432로 디폴트로 되어있고 내가 docker-compose에서 설정한 포트로 처리가 되어있지 않다.  
    해당 이유는 PostgreSQL의 포트번호가 conf파일에 적용된 값으로 되어있어 해당 파일에서 포트를 바꿔줘야 한다. 나의 conf파일의 경우 data폴더 안에 있었다.  
    (<https://bumday.tistory.com/106>)
    1. 일반적으로 도커로 생성할 때 외부포트는 변경해주는 건 상관없다만 내부에서 사용하는 포트는 그대로 두자… 아래와 같이 괜한 기본 시스템 설정만 번거로워진다.
31. 포트를 바꾸고 나서 psql명령어가 안먹힐 텐데 psql -U root -p 포트번호 이런식으로 명령어를 입력하면 된다. ( <https://copycoding.tistory.com/412>)
32. Dbeaver에서 ER-DIAGRAM을 다뤄보기(<https://codingdog.tistory.com/entry/dbeaver-erd-diagram-%EC%9D%84-%EC%86%90%EC%89%BD%EA%B2%8C-%EC%83%9D%EC%84%B1%ED%95%98%EB%8A%94-%EB%B0%A9%EB%B2%95%EC%9D%84-%EC%95%8C%EC%95%84%EB%B4%85%EC%8B%9C%EB%8B%A4>)
33. Spring boot jpa 사용 방안(<https://wakestand.tistory.com/668>)  
    (<https://congsong.tistory.com/51>)
34. Jpa build gradle 설정(<https://blue-mina.tistory.com/99>)
35. Jpa application.properties 설정(<https://velog.io/@_koiil/Spring-boot-JPA-Hibernate-Postgres-%EC%97%B0%EB%8F%99%ED%95%98%EA%B8%B0>)
36. Hikari를 통해 커넥션 풀 설정해주기
    1. (<https://minkwon4.tistory.com/208>), (<https://velog.io/@miot2j/Spring-DB%EC%BB%A4%EB%84%A5%EC%85%98%ED%92%80%EA%B3%BC-Hikari-CP-%EC%95%8C%EC%95%84%EB%B3%B4%EA%B8%B0>)
    2. Hikari 디버그 설정  
       (<https://velog.io/@aram/Spring-Boot-HikariCP-log-%EC%B6%9C%EB%A0%A5-%EC%84%A4%EC%A0%95%ED%95%98%EA%B8%B0>)
37. Svelte-spa-router는 sveltekit에서 사용되지 않는다.  
    ( <https://stackoverflow.com/questions/70529698/svelte-spa-router-invalid-component-object-error>)
    1. 참고로 그냥 Svelte에서 사용되는 spa사용법 중 찾아본 링크들  
       (<https://svelte.dev/repl/6e7546119cff41fabe35df3ef999ff7e?version=3.29.4>)  
       (<https://asecurity.dev/entry/Svelte-SPA-routes-%EC%84%A4%EC%A0%95>)  
       (<https://velog.io/@jude-ui/svelte-spa-router>)  
       (<https://doppelgoer.tistory.com/entry/Svelte-%ED%8E%98%EC%9D%B4%EC%A7%80-%EB%9D%BC%EC%9A%B0%ED%8C%85>)
38. sveltekit에서 네비bar 위에 마우스를 hover만 해도 fetch가 되는 경우는 prefetching으로 a태그에 data-sveltekit-prefetch를 넣으면 마우스 hover시에 fetch로 데이터를 가져온다.
    1. app.html body태그에 preload-data옵션이 들어가 있어 기본적으로 동작할 것이다..
    2. 만일 이대로 사용하기 싫다면 레이아웃에는 csr=false옵션을 넣어주고 특정 네비 버튼에 대해서는 풀어주고 싶다면 해당 page에서 csr=true로 풀어주면 된다.
    3. 크롬의 JavaScript를 끄게 될 경우 Navbar공지사항으로 마우스 올릴 시 fetch를 보내지 않는다. 오로지 SSR로만 동작하게 된다.
    4. 또한 CSR형태로 두면 navbar버튼을 입력 시 화면 전체를 랜더링 하지않고 바뀐 부분에 대해서만 적용되므로 SPA의 역할을 하게된다. 하지만 SSR로 하게 될 경우 전체 페이지를 긁어오는 것을 확인 할 수 있다.
39. Page.server.js와 page.js의 차이를 살펴보았는데
    1. load메서드를 두었을 때 page.js에서 ssr=false로 두었을 경우 csr로 동작하지만 page.server.js는 동작하지 않는다.
    2. Page.js에서는 CSR SSR 모두 유동적으로 동작하며 기본적으로 CSR로 돌아가는듯 싶다.
    3. 참고로 CSR에서는 브라우저에 로그가 찍히고 SSR에서는 브라우저에 로그가 찍히지 않는다.
    4. load내부는 waterfall형태로 동작하여 첫번째 api데이터를 가져오고 나서 두번째 데이터를 가져온다. 하지만 각각 load안에 async로 가져오면 병렬로 가져온다.
40. ESlint, prettier 설정 (<https://veggie-garden.tistory.com/13>)
41. 드롭다운 메뉴 목록 위치 조절  
    (<https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=pjh445&logNo=221159751165>)
42. 드롭다운 메뉴 사이즈 조절  
    (<https://stackoverflow.com/questions/23042182/increase-bootstrap-dropdown-menu-width>)
43. Sveltekit redirect
    1. 301, 302에 관하여 (<https://nsinc.tistory.com/168>)
    2. Redirect 방법(<https://kit.svelte.dev/docs/load#redirects>)  
       (<https://stackoverflow.com/questions/74805197/how-to-redirect-to-page-in-sveltekit>)
44. PostgreSQL 테이블 목록 조회 명령어  
    SELECT \* FROM PG\_STAT\_USER\_TABLES;
45. DB테이블에서 실선과 점선에 대한 차이  
    (<https://sddev.tistory.com/81>)
46. dbeaver에서 foreign key 설정에 대한 링크  
    (<https://codingdog.tistory.com/entry/dbeaver%EC%97%90%EC%84%9C-foreign-key%EB%A5%BC-%EC%83%9D%EC%84%B1%ED%95%98%EB%8A%94-%EB%B0%A9%EB%B2%95%EC%9D%84-%EC%95%8C%EC%95%84%EB%B4%85%EC%8B%9C%EB%8B%A4>)
47. DB명명 규칙(<https://velog.io/@jkijki12/Database-%EC%9D%B4%EB%A6%84-%EC%A7%93%EA%B8%B0-%EC%96%B4%EB%A0%A4%EC%9A%B0%EC%84%B8%EC%9A%94>)
48. GET, PUT, DELETE, POST를 구분지어서 잘 사용할 것  
    (<https://velog.io/@yh20studio/CS-Http-Method-%EB%9E%80-GET-POST-PUT-DELETE>)
49. Actual and format argument lists differ in length에러가 발생하게 될 경우  
    (<https://smoh.tistory.com/458>)
50. Entity설정 과정에서 테이블명을 대문자로 했음에도 소문자로 테이블을 새로 생성해버리게 되는 문제가 발생하였다.  
    (<https://velog.io/@gillog/JPA-Spring-Boot-JPA-Entity-Table-%EB%8C%80-%EC%86%8C%EB%AC%B8%EC%9E%90-%EA%B5%AC%EB%B6%84-%EB%AA%BB%ED%95%98%EB%8A%94-%EA%B2%BD%EC%9A%B0-%ED%95%B4%EA%B2%B0>)
    1. 여기서 PostgreSQL의 경우 대문자 구분을 위해 \”USERS\_TB\”형태로 항상 \문자로 쌍따옴표를 붙인 형태로 따라다녀야 대문자라는 것으로 인식한다.
51. join이후에 백엔드에서 url을 redirect시켜야 하는 경우 (<https://velog.io/@albaneo0724/Spring-RestController-%ED%81%B4%EB%9E%98%EC%8A%A4%EC%97%90%EC%84%9C-%EB%A6%AC%EB%8B%A4%EC%9D%B4%EB%A0%89%ED%8A%B8-%EC%8B%9C%ED%82%A4%EB%8A%94-%EB%B0%A9%EB%B2%95>)
52. 날짜 데이터 처리  
    (<https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=tyboss&logNo=70034913643>)
53. LastLoginDate는 주로 TimeStamp형태로 저장한다.
54. Servlet.service() for servlet [dispatcherServlet] in context with path [] threw exception [Request processing failed 에러 발생시 Autowired생성자에 내가 쓸 datahandler를 주입시켰는지 확인해본다.
55. 1:1관계를 위해 userDto, userEntity, roleDto, roleEntity작업을 했으며 Entity에서 설정했던 방법이다.  
    ( <https://steady-hello.tistory.com/104>)
56. OAuth  
    (<https://gdtbgl93.tistory.com/180>)
57. hasRole로 설정하게 될 경우 “ROLE\_”이라는 접두사가 붙어 비교를 하지만 hasAuthority를 사용하면 접두사 없이 권한 체크를 한다.  
    (<https://codecrafting.tistory.com/9>)
58. Credentials이란 쿠키, 인증헤더, TLS client certificates를 말한다. Credentials이 있는 CORS요청은 Client와 Server 둘 다 Credentials를 사용하겠다는 속성을 설정해줘야 통신이 가능하다. => 즉 client와 server가 쿠키 값을 공유하겠다는 의미  
    (<https://basemenks.tistory.com/246>)
59. 프론트와 백엔드 분리 이후 쿠키 값이 공유가 되지 않을 때
    1. 동일한 Site에서 동작하지만 Origin이 다르므로 쿠키공유가 안된다. (CORS)  
       클라이언트와 서버에서 모두 Credentials 관련된 옵션을 설정해줄 것
    2. 클라이언트와 서버의 Site와 Origin이 모두 같을 때는 서버에서 응답 헤더의 set-cookie에 쿠키를 담아 전송하면 브라우저에서 자동으로 쿠키가 저장이 된다.
    3. 텍스트이(가) 표시된 사진

       자동 생성된 설명
    4. 나의 경우 <http://localhost:5170> (FrontEnd)와 <http://localhost:5052/api/v1/api-user/login>으로 동일한 사이트 내에서 동작하지만 Cross origin이다.
    5. (<https://velog.io/@youjung/%ED%94%84%EB%A1%A0%ED%8A%B8%EC%99%80-%EB%B0%B1%EC%97%94%EB%93%9C-%EB%B6%84%EB%A6%AC%ED%96%88%EC%9D%84-%EB%95%8C-%EC%BF%A0%ED%82%A4-%EA%B3%B5%EC%9C%A0-%EC%95%88%EB%90%A0-%EB%95%8C>)
60. Logout시에 쿠키 및 세션 삭제 방법  
    (<https://baejangho.com/entry/Spring-Security-Logout>)
61. Spring Boot Spring-Security 설정 방법 따라 해본 곳  
    (<https://dev-coco.tistory.com/120>)
62. 쿠키와 세션 토큰의 차이점은 알아두자  
    (<https://velog.io/@jung5318/%EC%BF%A0%ED%82%A4-%EC%84%B8%EC%85%98-%ED%86%A0%ED%81%B0%EC%9D%98-%EC%B0%A8%EC%9D%B4%EC%A0%90>)
63. 프론트와 백엔드 분리했을 때 쿠키 값이 공유가 되지 않는 경우  
    (<https://velog.io/@youjung/%ED%94%84%EB%A1%A0%ED%8A%B8%EC%99%80-%EB%B0%B1%EC%97%94%EB%93%9C-%EB%B6%84%EB%A6%AC%ED%96%88%EC%9D%84-%EB%95%8C-%EC%BF%A0%ED%82%A4-%EA%B3%B5%EC%9C%A0-%EC%95%88%EB%90%A0-%EB%95%8C>)  
    (<https://tech-monster.tistory.com/95>)  
    (<https://tecoble.techcourse.co.kr/post/2021-09-10-ssr/>)

추가적으로 현 개발일지 디렉터리에 따로 문서도 만들어 놨으니 참고 바란다.

1. Spring security에서 CorsConfigurationSource메서드를 설정해두고 WebConfigurer에 addCorsMappings를 설정해둔 상태에서 spring-security에 따르는 엔드포인트의 경우에는CorsConfigurationSource에 따른다.

Spring Framework에서 CORS(크로스 오리진 리소스 공유)는 웹 페이지 또는 웹 응용 프로그램이 다른 도메인 또는 오리진의 리소스에 액세스하는 것을 제한하는 보안 기능입니다. CORS를 사용하면 웹 응용 프로그램이 다른 도메인의 리소스에 액세스하기 위해 교차 오리진 HTTP 요청을 만들 수 있습니다.  
"addCorsMappings" 방법은 응용 프로그램의 모든 컨트롤러에 대해 전체적으로 CORS를 구성하는 데 사용됩니다. 일반적으로 Spring MVC의 구성을 사용자 지정하는 데 사용되는 WebMvcConfigurer 인터페이스에 사용됩니다.  
"addCorsMappings" 메서드는 경로 패턴과 CORS 구성 간의 매핑을 등록하는 데 사용되는 CorsRegistry 유형의 매개 변수를 사용합니다. 예를 들어, 이 방법을 사용하여 적절한 경로 패턴을 지정하고 모든 오리진을 허용하여 모든 오리진이 응용 프로그램의 특정 끝점에 액세스할 수 있도록 할 수 있습니다.  
반면, "corsConfigurationSource" 방법은 응용 프로그램의 특정 엔드포인트 또는 엔드포인트 집합에 대한 CORS를 구성하는 데 사용됩니다. 일반적으로 응용 프로그램의 Spring Security 구성을 정의하는 @Configuration 클래스에서 사용됩니다.  
"corsConfigurationSource" 메서드는 특정 엔드포인트에 대한 CORS 정책을 구성하는 데 사용되는 CorsConfigurationSource 유형의 매개 변수를 사용합니다. 이 방법을 사용하여 허용되는 오리진, 메서드, 헤더 및 기타 매개 변수를 지정하여 특정 엔드포인트에 대한 CORS 정책을 사용자 정의할 수 있습니다.  
요약하면, "addCorsMappings"는 응용 프로그램의 모든 컨트롤러에 대해 CORS를 글로벌하게 구성하는 데 사용되는 반면, "corsConfigurationSource"는 응용 프로그램의 특정 엔드포인트 또는 엔드포인트 집합에 대한 CORS를 구성하는 데 사용됩니다.  
  
In Spring Framework, CORS (Cross-Origin Resource Sharing) is a security feature that restricts web pages or web applications from accessing resources on different domains or origins. CORS allows web applications to make cross-origin HTTP requests to access resources from a different domain.

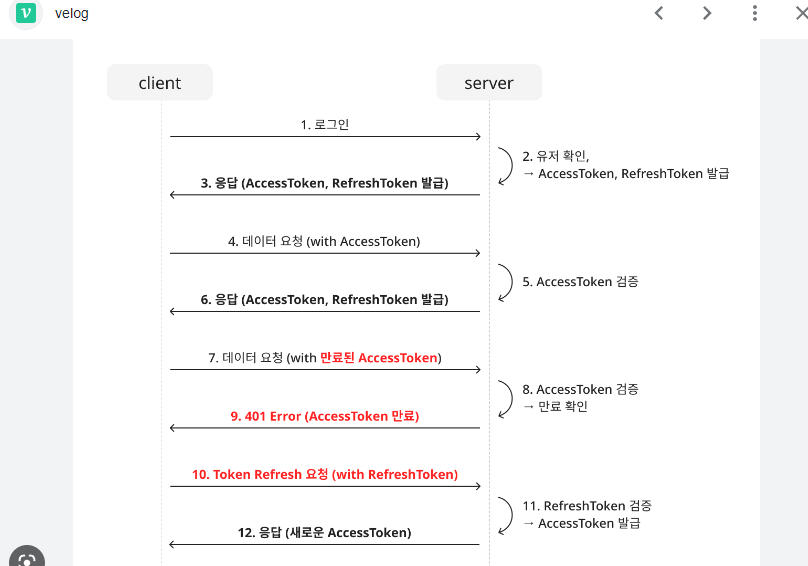
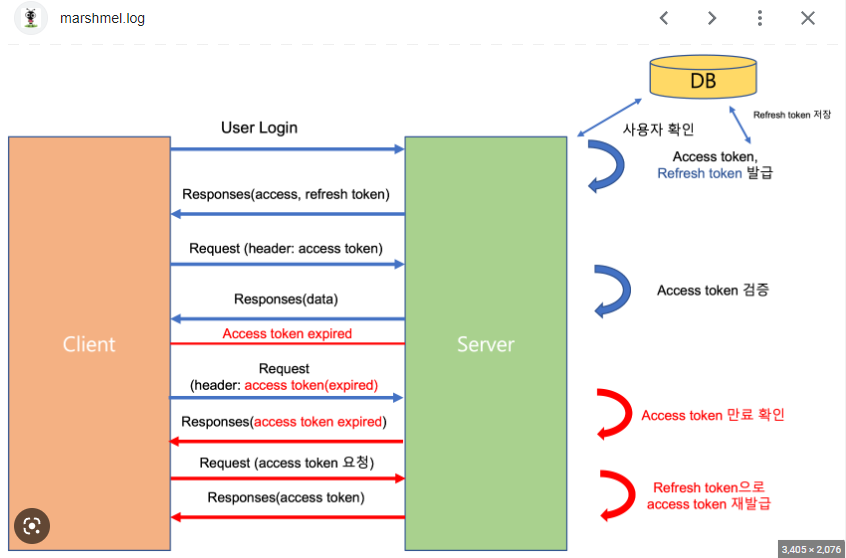
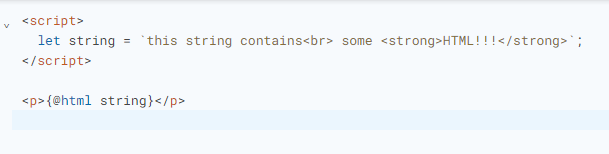
The "addCorsMappings" method is used to configure CORS globally for all controllers in an application. It is typically used in the WebMvcConfigurer interface, which is used to customize the configuration of Spring MVC.

The "addCorsMappings" method accepts a parameter of type CorsRegistry, which is used to register mappings between path patterns and CORS configuration. For example, you can use this method to allow all origins to access a specific endpoint in your application by specifying the appropriate path pattern and allowing all origins.

On the other hand, the "corsConfigurationSource" method is used to configure CORS for a specific endpoint or set of endpoints in an application. It is typically used in the @Configuration class that defines the Spring Security configuration for an application.

The "corsConfigurationSource" method accepts a parameter of type CorsConfigurationSource, which is used to configure the CORS policy for a specific endpoint. You can use this method to customize the CORS policy for a specific endpoint by specifying allowed origins, methods, headers, and other parameters.

In summary, "addCorsMappings" is used to configure CORS globally for all controllers in an application, while "corsConfigurationSource" is used to configure CORS for a specific endpoint or set of endpoints in an application.

1. Builder, AllArgsConstructor, NoArgsConstructor, Data에 대한 어노테이션 설명  
   (<https://velog.io/@dydaks7878/Builder-NoArgsConstructor-AllArgsConstructor-RequireArgsConstructor>)
2. JWT로 토큰 생성하여 포스트맨으로 테스트 해보기  
   (<https://gksdudrb922.tistory.com/217#JwtTokenProvider>)  
   위 상황에서 주의해야 될 점은 CustomuserDeatilsService클래스에서 createUserDetails메서드에 Rolename을 넣을 때 .roles안에 들어갈 인자는 “ROLE\_”로 시작하면 안된다. 자동으로 넣어서 객체를 생성해주기 때문이다.
3. corsConfiguration메서드를 configure에 Bean으로 등록하여 설정해줬음에도 Response Header에 적용이 되지 않는 이유는 filterChain메서드에서 HttpSecurity에 다음의 명령어로 설정을 적용시켜주지 않았기 때문.  
   http.cors().configurationSource(corsConfigurationSource())
4. Refresh Token을 세션에 저장해서 사용하게 될 경우 세션관리가 필요해지면서 credentials옵션 등 세션으로 로그인, 로그아웃 처리에 대한 기능 위에 얹히는 기분이 들고 jwt의 목적과 거리가 조금 멀다고 생각된다….
5. repository에서 Update와 Delete를 사용할때는 @Modify @Transactional 어노테이션을 꼭 붙여주도록 하자  
   (<https://velog.io/@cksdnr066/ERROR-Not-supported-for-DML-operations>)
6. Refresh Token을 어디에 저장해주는 게 JWT토큰의 목적과 맞을까  
   (<https://velog.io/@jisu2281l/TIL-Refresh-Token-%EC%A0%80%EC%9E%A5-%EC%9C%84%EC%B9%98%EB%8A%94-%EC%96%B4%EB%94%94>)
7. 
8. 
9. 도커 Redis로 설치할 때 docker-compose방법 이용  
   (<https://bskyvision.com/entry/docker-docker-compose%EB%A1%9C-Redis-%EC%BB%A8%ED%85%8C%EC%9D%B4%EB%84%88-%EC%8B%A4%ED%96%89%ED%95%98%EA%B8%B0>)
10. 문자열로 입력하고 html태그에 동적으로 문자를 입력하는데 html 형태로 저장하고 싶은 경우 {@html text}로 문자열에 <br> 입력해도 되고 style옵션으로 white-space:pre옵션을 주고 /n과 같은 명령어를 입력해도 된다.  
      
      
    텍스트이(가) 표시된 사진

    자동 생성된 설명
11. JWT와 Spring Security를 같이 사용하게 될 경우 에러처리가 발생할 때 에러를 처리해줄 수 있는 예외처리 필터를 각각 커스텀 해줘야한다. 그중 JWT는 세션관련 문서에 방법을 올려두었고 spring security의 경우 아래의 방법으로 진행하여 사용자 로그인 실패를 200 OK로 응답하고 내가 원하는 권한 실패의 내용을 body에 담아 보낼 수 있도록 하였다.  
    (<https://devjem.tistory.com/72#%EA%B-%B-%EB%B-%B-%--%EC%-D%B-%EC%A-%-D%--%EC%--%--%EC%--%B-%--%EC%B-%--%EB%A-%AC%--%EB%B-%A-%EB%B-%-->)
12. Npm 삭제 관리(지울 때 뒤에 ‘@3.1.1’이런식의 골뱅이서부터는 빼고 지우고 -g 옵션도 빼도 된다.  
    (<https://velog.io/@han-byul-yang/npm-%EC%84%A4%EC%B9%98-%EA%B2%80%EC%83%89-%EC%97%85%EB%8D%B0%EC%9D%B4%ED%8A%B8-%EC%82%AD%EC%A0%9C-%EC%A0%95%EB%A6%AC>)

Map에서 addLayer했음에도 맵타일 밑으로 가려지면서 우선순위가 낮은 경우   
// Add a new symbol layer before an existing layer

map.addLayer({

id: 'states',

// References a source that's already been defined

source: 'state-data',

type: 'symbol',

layout: {

// Set the label content to the

// feature's `name` property

'text-field': ['get', 'name']

}

// Add the layer before the existing `cities` layer

}, 'cities');

위와 같이 ‘cities’의 위치가 뒤에 존재할 경우 cities 레이어 뒤에 생겨버린다.

1. 추후 배포 후 cors 사라질 시 블로그의 cra파트 알아볼 것 (<https://xiubindev.tistory.com/115>)
2. Vite프록시 로그 찍는법  
   (<https://github.com/vitejs/vite/discussions/6469>)
3. Postgresql에서 serial의 값이 0으로 setval이 되지 않을 경우 (<https://stackoverflow.com/questions/16485708/reset-sequence-value-as-1()>  
   alter sequence mytable\_id\_seq minvalue 0 start with 1; 위 사이트에서 나와있다 싶이 요거 쓰면 해결 가능
4. JPA 네이밍 전략에 대해서 알고있자(<https://velog.io/@devduhan/Spring%ED%94%84%EB%A1%9C%EC%A0%9D%ED%8A%B8-JPA-Naming-%EC%A0%84%EB%9E%B5>)  
   (<https://namocom.tistory.com/999>)  
   camel-case를 언더스코어로 변경해주는 것이 기본 설정  
   (<https://mycup.tistory.com/237>)  
   (<https://jobc.tistory.com/127)\>
5. JPA 로깅 관련 설정  
   (<https://zzang9ha.tistory.com/399>)
6. Jasypt로 yml파일에 중요사항 암호화 방법  
   (<https://bamdule.tistory.com/251>)
7. Sveltekit에서 jquery 사용법  
   (<https://stackoverflow.com/questions/71017937/importing-jquery-in-sveltekit>)
8. @Data  
   @Builder => requiredAragsConstructor생성(기본생성자가 없어진다.)  
   @NoArgsConstructor => (기본 생성자를 따로 생성해줘야한다.)  
   @AllArgsConstructor => (기본 생성자를 생성해줌에 따라 All도 따로 해줘야 한다. 만약 기본 생성자가 없었다면 생성해줄 필요없이 알아서 all을 만들어 사용한다.)  
   (<https://resilient-923.tistory.com/418>)
9. lombok에서 빌더란 대충…(<https://blog.naver.com/seek316/222360574871>)
10. lombok에서 자주 사용되는 것들(<https://www.daleseo.com/lombok-popular-annotations/>)
11. Table selector 라이브러리(<https://resilient-923.tistory.com/418>)
12. Treejs 라이브러리(<https://github.com/daweilv/treejs>)
13. Springboot log4j2 적용(<https://velog.io/@mooh2jj/Log4j-%EB%9D%BC%EC%9D%B4%EB%B8%8C%EB%9F%AC%EB%A6%AC-%EC%84%A4%EC%A0%95>)
14. JPA영속성 공부  
    (<https://velog.io/@gillog/JPA-%EA%B8%B0%EB%B3%B8-%ED%82%A4-%EC%83%9D%EC%84%B1-%EC%A0%84%EB%9E%B5IDENTITY-SEQUENCE-TABLE>)  
    (<https://ict-nroo.tistory.com/130>)  
    (<https://ttl-blog.tistory.com/123>)
15. Spring-batch JobLauncher는 내부적으로 비동기 설정이 되어있다.  
    (<https://www.inflearn.com/questions/396844/%EB%B9%84%EB%8F%99%EA%B8%B0%EC%8B%9D%EC%9C%BC%EB%A1%9C-joblauncher-%EC%8B%A4%ED%96%89%EC%8B%9C%ED%82%A4%EB%8A%94-%EA%B2%83-%EA%B4%80%EB%A0%A8-%EC%A7%88%EB%AC%B8-%EB%93%9C%EB%A6%BD%EB%8B%88%EB%8B%A4>)  
    참고로 그냥 컨트롤러 메서드 여러 개에다가 post를 쐈을 때도 비동기식으로 동작했다.  
    Spring MVC 화성시에서는 따로 async를 설정해줬어야 했던게 xml 빈파일을 등록한 jobLauncher를 실행하기 때문에 async를 별도로 설정해줘야 한다.
16. 멀티스케줄러 참고 사이트  
    (<https://devsejong.tistory.com/60?category=482589>)  
    (<https://jaksimsamil.tistory.com/44>)  
    (<https://wooncloud.com/75>)
17. Annotation에 대한 정상 동작하지 않았을 때는 enable annotation processing을 체크해주도록 하자  
    (<https://woodeekim.tistory.com/33>)
18. Mybatis log4j2로그찍기  
    (<https://powernote.tistory.com/57>) 여기 링크 로그레벨 설정 안먹히거나 추가 귀찮을 시 그냥 yml파일에서…(<https://velog.io/@dkatlf900/MyBatislog4jdbc>)
19. 각 쿼리 로그 레벨 설정 설명  
    (<https://velog.io/@harpuria/Springboot-MyBatis-%EC%BF%BC%EB%A6%AC-%EB%A1%9C%EA%B7%B8-%EC%84%A4%EC%A0%95>)
20. sveltekit 윈도우 -> 맥북 환경 테스트 중 vite command not found 에러로 인해 npm i로 해결
21. 텍스트 드래그 방지하는 법  
    (<https://zzznara2.tistory.com/813>)’
22. 마우스 클릭 막는법  
    (<https://7network.tistory.com/entry/CSS-%EB%A7%88%EC%9A%B0%EC%8A%A4-%ED%81%B4%EB%A6%AD-%EA%B8%88%EC%A7%80-%EC%95%88%EB%90%98%EA%B2%8C-%EB%A7%89%EA%B8%B0>)
23. Spring Boot CORS 설정(<https://cocococo.tistory.com/entry/Spring-Boot-COR-%EC%84%A4%EC%A0%95-%EB%B0%A9%EB%B2%95>)
24. 부모 자식 컴포넌트간의 데이터 동적 적용 방법  
    (<https://developer-talk.tistory.com/600>)
25. 자바스크립트단 엑셀 파일 불러오기  
    (<http://www.gisdeveloper.co.kr/?p=8987>)
26. 좌표변환 각각 소스코드  
    (<https://www.osgeo.kr/17>)
27. 자바스크립트단에서 ()=>(화살표함수)와 없이 함수명을 쓰는 차이는 무엇일까  
    The difference between using `() =>` and not using it is related to how functions are invoked in JavaScript.

When you use `() =>` syntax, you are creating an anonymous arrow function that will be invoked when the event occurs. This allows you to pass arguments to the function or perform additional operations before invoking it.

For example, in the context of your code:

```html

<a href="#" on:click={() => actionIdxChk(index)}>...</a>

```

Here, the arrow function `() => actionIdxChk(index)` is created and assigned as the event handler for the `click` event. When the `click` event occurs, the arrow function is invoked, which then calls the `actionIdxChk` function with the `index` argument.

On the other hand, if you omit the `() =>` syntax and directly provide the function reference, like this:

```html

<a href="#" on:click={actionIdxChk(index)}>...</a>

```

In this case, the `actionIdxChk` function is immediately invoked when the template is rendered, rather than waiting for the `click` event. This is not the behavior you want for event handlers. Instead, you want the function to be invoked only when the `click` event occurs.

So, to ensure that the function is invoked correctly as an event handler, you need to wrap it inside an anonymous function or use the `() =>` syntax to create a new function that will be invoked on the event.

(<https://webclub.tistory.com/649>)

1. QGIS 사용법  
   (<https://m.blog.naver.com/ecolab210/222126153573>)
2. Sveltekit 한글 docs(<https://luvstudy.tistory.com/192>)
3. IndexedDB & localstorage & websql 차이에 관하여  
   (<https://nolanlawson.com/2015/09/29/indexeddb-websql-localstorage-what-blocks-the-dom/>)
4. NoSQL 종류 4가지중 key-value, column 기반 데이터, document 저장에 관해  
   (<https://jaemunbro.medium.com/nosql-%EB%8D%B0%EC%9D%B4%ED%84%B0%EB%B2%A0%EC%9D%B4%EC%8A%A4-%ED%8A%B9%EC%84%B1-%EB%B9%84%EA%B5%90-c9abe1b2838c>)
5. WebSQL 사용법 (<https://iamawebdeveloper.tistory.com/98>)
6. require안먹혀서 약간 가라식으로 되게 한거같다…  
   <script> window.require = () => {} const global = {} </script>  
   (<https://github.com/vitejs/vite/issues/3409>)
7. SQLite + sveltekit 이거 맞나…  
   (<https://www.google.com/search?q=sveltekit+javascript+sqlite&oq=sveltekit+javascript+sqlite&gs_lcrp=EgZjaHJvbWUyBggAEEUYOdIBCDY5NDNqMGoxqAIAsAIA&sourceid=chrome&ie=UTF-8#fpstate=ive&vld=cid:b49bcfd1,vid:BTI5oZM3buo>)
8. SQLJS 사용법하고 코드는 svelteket-web에서 참고  
   (<https://github.com/sql-js/sql.js/>)
9. JOIN 설명 굿잡  
   (<https://lcs1245.tistory.com/entry/SQL-JOIN-INNER-JOIN-OUTER-JOIN-LEFT-JOIN-RIGHT-JOIN-SELF-JOIN>)
10. sqlJS 챗GPT에서 사용법 설명(여기에 locateFile 꼭 설정 추가해줘야된다.)  
    (<https://chat.openai.com/c/1a88ccdd-6e55-4e29-bcae-cf0fd1b751fc>)
11. Jsoup으로 로그인 뚫기  
    1. 로그인페이지에 들어갔을 때 로그인 form태그 안에 input태그로 있는 변수들을 체크한다.  
    2. 해당 변수들을 Map에 넣어줌  
    3. 로그인 action url주소가 만약 form태그에 있으면 해당 url로 jsoup.~.execute()를 날리면 된다. 하지만 action url주소가 명시되어 있지 않다면 개발자도구 소스코드에서 form 버튼 입력 후 쏘게 되는 url주소를 찾아서 넣어야 된다.  
    4. 받아온 쿠키 값을 가지고 사용하면 끝난다.  
    (<http://www.joshi.co.kr/index.php?mid=SPRING&document_srl=305783>)  
    (<https://headfirst.tistory.com/9>)
12. 위처럼 로그인을 시도하여 개발자 도구 네트워크에서 찾아보려 하는데 redirect때문에 바로 다른 페이지로 넘어가는 경우 로그인 시에 송수신 되는 네트워크를 찾기가 어려울 수 있다. 그럴 때는 $('#skin > form')[0].addEventListener('submit', function(e) {e.preventDefault()})  
    위 명령어에서 필요부분만 바꿔서 진행하면 된다.   
    저게 고유의 동작으로 페이지를 이동시키고 난 후 동작을 멈추게 하는 방법이다.  
    (<https://studyingych.tistory.com/27>)
13. JAVA단 RSA 암복호화에 대해서는 아래의 링크를 통해 개발하였다.  
    (<https://lion-king.tistory.com/entry/JavaRSA-Java-RSA-%EC%95%94%ED%98%B8%ED%99%94%EB%B3%B5%ED%98%B8%ED%99%94-%EA%B5%AC%ED%98%84>)

자바스크립트 단 RSA 암복호화 개발할 때 확인한 사이트  
(<https://chulkang.tistory.com/81>)

1. SelectQuery 꺽쇠(O)와 스페이스(X) 차이는 바로 아래 자식만 부르냐의 차이  
   (<https://raisonde.tistory.com/entry/jQuery-Seletor%EC%97%90%EC%84%9C-%EC%8A%A4%ED%8E%98%EC%9D%B4%EC%8A%A4-%EC%99%80-%EA%BA%BD%EC%87%A0-%EC%9D%98-%EC%B0%A8%EC%9D%B4>)
2. HttpURLConnection에 쿠키를 품는 방법  
   (<https://nine01223.tistory.com/328>)
3. 엑셀 poi 4점대 사용하기  
   (<https://dailylifecoding.tistory.com/entry/apache-POI-%EA%B0%84%EB%8B%A8-%EC%82%AC%EC%9A%A9%EB%B2%95-1>)
4. Gradle 환경에서 Geotools 설치가 안될 때  
   (<https://goateedev.tistory.com/130>)  
   gradle 라이브러리 명령어  
   (<https://mvnrepository.com/artifact/org.geotools/gt-shapefile/19.0>)  
   dbf파일 읽어들이기  
   (<https://spatiumwdev.tistory.com/10>)
5. Jackson 라이브러리에서 ObjectMapper를 사용할 때 Dto의 변수들이 Camel케이스이고 json의 키값들이 snake타입일 때 어떻게 하면 될까  
   (<https://jessyt.tistory.com/31>)  
   Jackson의 ObjectMapper설정타입을 바꿔주면 된다.