**고급 웹 프로그래밍 과제**

2021113490 이현서

**목차**

1. 과제 배경 및 목표
2. 개발 내용 및 구조
3. 개발 후 소감
4. 외부 소스코드 참조

**과제 목표**

이번에 저는 STUIK ( Study In Kert )라는 웹 앱을 기획하였습니다. 저는 Kert라는 학술 동아리의 부회장으로서, 회장을 대신해서 모든 행사를 기획하고 관리 하는 역할을 해야 했습니다. 그리고 스터디를 관리하는 명부나, 관련 정보를 관리하기가 힘듦을 느낄 수 있었습니다.

그래서 이번기회에 간단히 MVP을 만들고, 나중에 방학때에 살을 입히고자 했습니다. 예를 들자면 회원인증 DB를 따로 구축하여 이메일 인증을 한다 거나, 스터디 활동 내용을 매번 기록해서 모든 사람이 스터디 내용을 공유할 수 있게 하는 기능을 구현하고자 합니다. 하지만 이번 고급 웹 프로그래밍 과제에서는 꼭 필요한 기능을 구현하는데 목표를 두고 진행하였습니다.

이번 프로젝트의 결과로 KERT내의 부원들이 조금 더 나은 환경에서 스터디를 진행할 수 있었으면 좋겠다는 마음이 큽니다.

**개발 내용 및 구조**

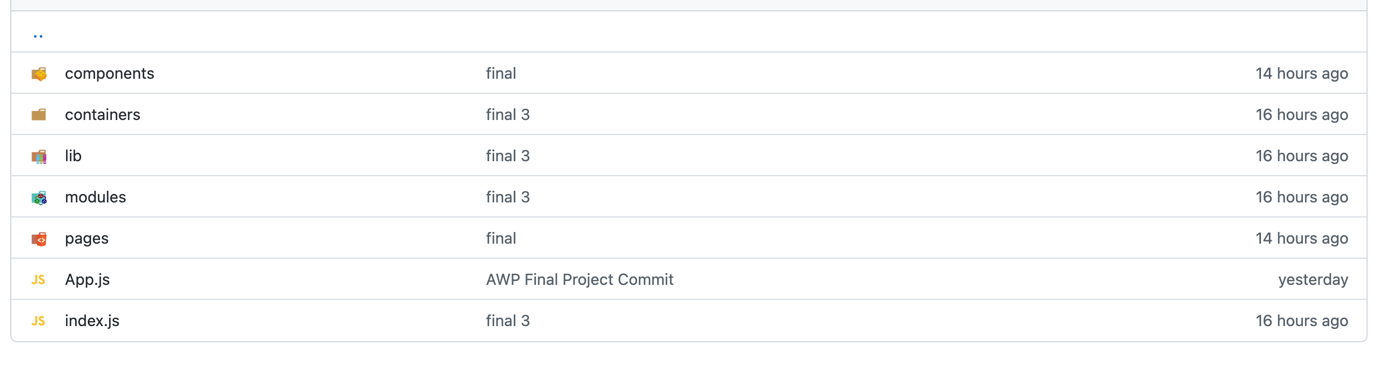
https://github.com/LE123123/KertStudyMatchingBack

https://github.com/LE123123/KeryStudyMatching

우선 깃에 올려 둔 소스 코드를 기반으로 개발 내용과 구조를 설명하도록 하겠습니다. 전자의 레포지토리에 개발환경과, 대략적인 사용방법을 README에 작성해 두었습니다. 참고하면서 보고서 내용을 봐주시면 감사하겠습니다.

**FE ( Front End )** https://github.com/LE123123/KeryStudyMatching

먼저 FE는 React Framework를 사용하였고 CRA( Create React App)을 활용하여 프로젝트를 생성하였습니다. 또한 FE코드는 BE =>M ( FE => VC ) 패턴을 준수하였습니다.



src폴더의 대략적인 형태는 위와 같습니다. 필수 기능의 필요성에 따라서 폴더를 구분했습니다. **components**는 **container**에서 주입된 **Redux-saga**의 전역 상태 또는, 이벤트 핸들러, 뮤테이션등을 받는 역할을 합니다.

그리고 **pages**는 이러한 components의 요소들을 이용하여 component들을 페이지에 렌더링 하는 역할을 합니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

다음은 **lib**입니다. 여기에는 나중에 설명할 BE에서 작성된 API Endpoint 를 axios로 관리하는 **api**폴더를 가지고 있고, **styles**에는 전역 스타일, 색, 전역 테마를 관리하는 파일들이 존재합니다. 그리고 **createRequestSaga**에는 redux-saga의 saga-function을 loading 상태와 쉽게 결합하기 위한 util-function을 작성했습니다..

텍스트이(가) 표시된 사진

자동 생성된 설명

그 다음으로는 **modules**입니다. modules에는 redux-saga의 전역 상태를 부분적으로 저장해 두었습니다. Auth, loading, post, posts, profile, user, write등등.. 다 페이지의 기능마다 필요한 전역 상태입니다. 그리고 **index.js**에서는 모든 reducer( redux의 일부분 )를 합쳤습니다.

그리고 **App.js**에서 react-router의 코드를 짜고, index.js에서 이를 렌더링 시켜주었습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

그리고 src외에도 다양한 폴더가 있습니다. 우선 **public**에서는 favicon과 **SEO**를 위한 meta태그들을 수정했습니다.

또한, **production**과 **development**의 환경변수를 달리하기 위해, **.env.development**와 .env.production을 달리 두어서 개발, 배포를 유용하게 하였습니다.

마지막으로 $npm run build를 통해서 **build**파일을 만들었습니다. 이 build파일은 BE에서 정적 파일 서빙을 할 것이므로 BE로 옮겨주었습니다.

**BE ( Back End )**  https://github.com/LE123123/KertStudyMatchingBack

BE는 Express팀에서 개발한 Express보다 경량화 된, async문법을 정식으로 지원하는 Koa-Framework을 사용했습니다.

텍스트이(가) 표시된 사진

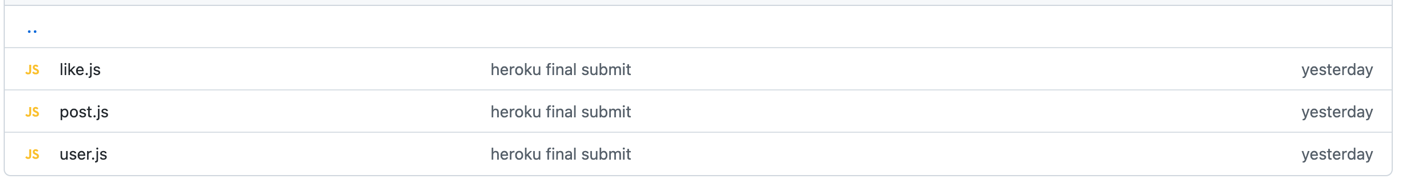
자동 생성된 설명

BE의 구조입니다. 여기서는 FE에서 VC를 구현했다면, **M (Model)**을 구현할 차례입니다. 우선 src폴더를 보겠습니다. (**MongoDB**활용)

텍스트이(가) 표시된 사진

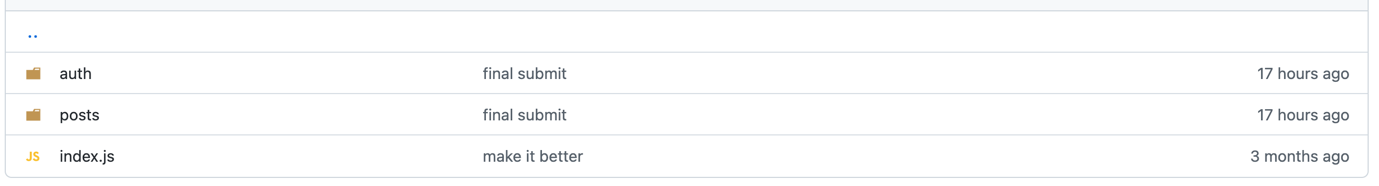
자동 생성된 설명

우선 **model**을 보겠습니다.



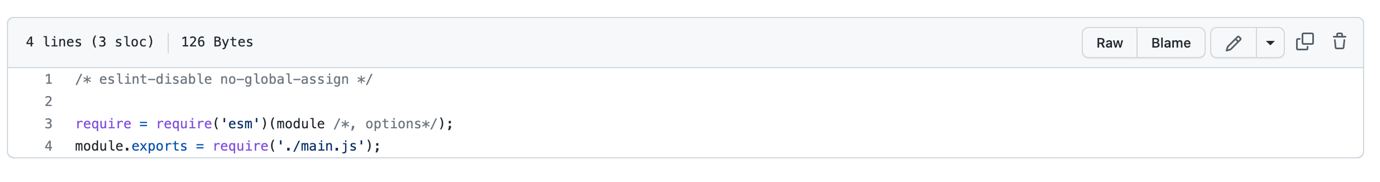
회원정보를 위한 **user스키마**를 정의하였고, 스터디 정보를 위한 **post스키마**를 정의하였습니다. 그리고 해당 스터디의 탈퇴와, 참여를 위해 **like스키마**를 정의하였습니다. 이들 간의 ref도 설정하였습니다 ( 나중에 **populate**를 사용해서 관련 모델을 다 가져오기 위함 )

다음으로는 이들을 사용하기 위한 **API Endpoint**를 작성해 주어야 합니다. Koa도 Express처럼 router기능을 지원합니다. 이를 위해 **api**폴더를 봅시다.



이 api폴더에는 크게 **/api/auth/[~~]**을 위한 **auth**폴더가 존재하고, **/api/posts/[~~]**을 위한 **posts**폴더가 존재하게 됩니다. 그리고 /api/auth는 **/api/auth/:id/[~~]**와 **/api/auth/[~~]**을 분리해서 작성하였습니다. 각 auth, posts을 합쳐 주는 것이 index.js파일이고, 각 auth와 posts파일은 index, [post or auth].ctrl.js로 이루어져 있습니다. ctrl에는 index에 주입할 구문이 들어있는 것입니다.

또한 위 Model은 모두 ES6문법 ( import, export )문법을 활용하였는데, 이는 정식 지원이 아니므로, 이를 가능하기 위해서 **esm**모듈을 설치해서 index.js에 다음과 같은 코드를 작성하였습니다.

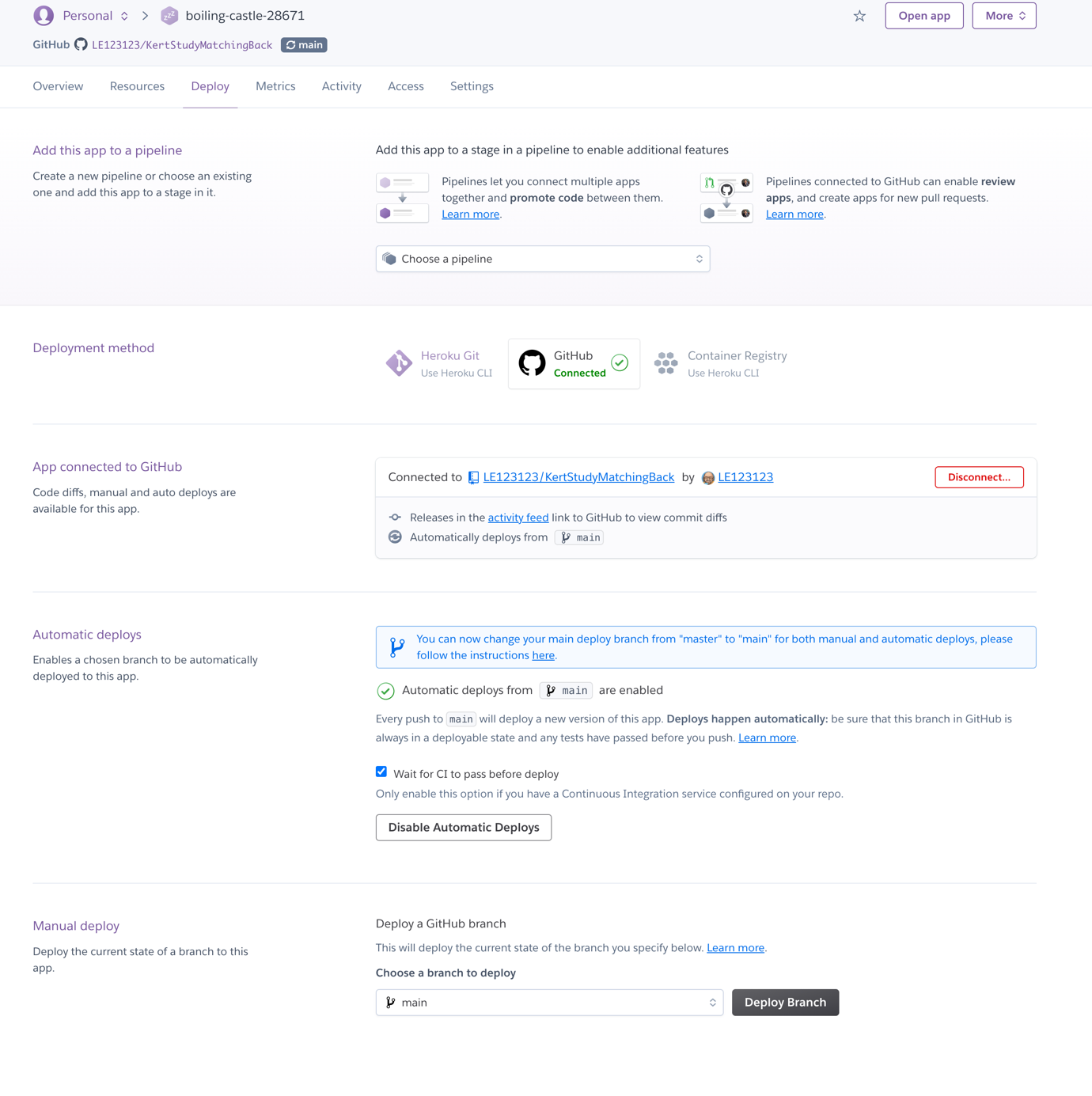


여기에 main.js을 주입하면 $node -r esm src 이렇게 -r esm 옵션을 주어 index.js을 실행하게 되면 우리가 하던 데로 main에서 코드를 작성해 주면 끝입니다.

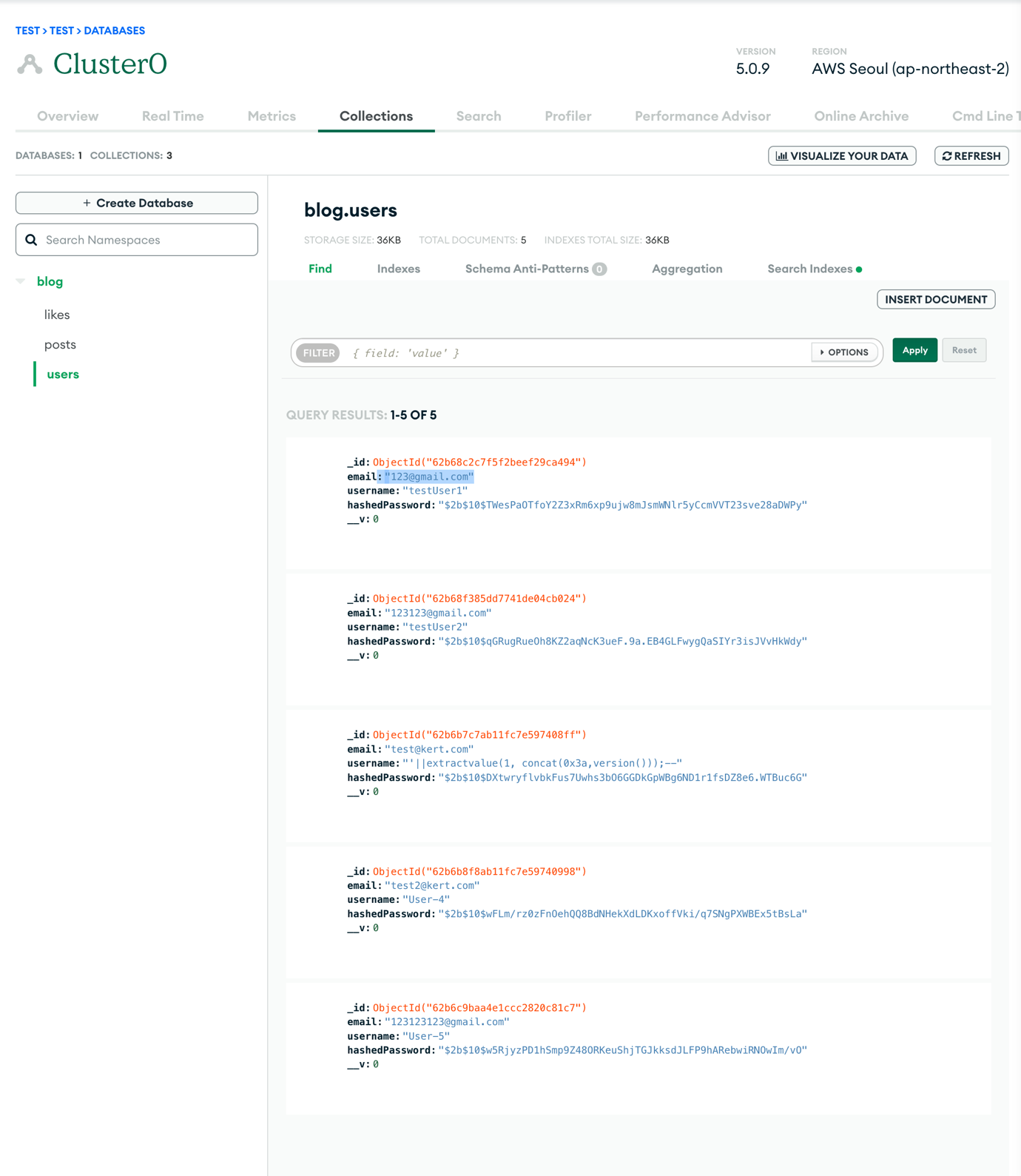
다음은 **main.js**입니다. main은 BE의 핵심입니다. 우선 저는 위에서 만든 라우터를 주입하였습니다. 그리고 스터디 글에 이미지 삽입을 위해서, **multer** 코드를 작성했습니다. 그리고 환경변수를 통해 mongodb atlas에 접속하는 코드를 작성하였고, **jwt** 회원인증을 위해서, 이와 관련된 코드를 작성하였습니다. 그리고 개발 환경에서 **CORS**설정을 진행해 주었고, 정적 서빙을 위한 **가상 경로** 설정을 해주었고, **/api/upload** Endpoint를 추가로 작성해 주었습니다. 그리고 이를 환경변수에 있는 포트로 열어 주었습니다.

나중에 **TDD**를 위한 test파일도 만들어 두었지만 사용하지 않았고, nginx을 위해 ssl폴더도 만들었지만 아직 사용하진 않았습니다. 그리고 ReadMe작성을 위한 **ReadMe**폴더를 작성하였습니다.

**배포**



일단 배포는 수업시간에 활용한 Heroku를 git 레포지토리와 연결해서 자동 배포기능을 사용하였습니다. 그리고 mongodb atlas에 클러스터를 하나 생성하고 배포하였습니다.



이로서 배포된 도메인은 다음과 같습니다.

<https://boiling-castle-28671.herokuapp.com/>

**개발 후 소감**

텍스트이(가) 표시된 사진

자동 생성된 설명

프로젝트를 완료하고, 버그바운티도 하고, KERT활동방에 이를 공지하는 과정을 거쳤습니다. 아직 MVP단계이기 때문에 기능이 많이 부족합니다. 하지만 주니어 개발자인 저로서는 너무 뿌듯했습니다.

프로젝트를 진행하기 위해서 많은 자료를 조사하였고, 많은 시간을 투자하였습니다. 3~4주라는 시간동안 혼자 개발하면서, 개발은 역시 혼자 하는게 아니라는 것을 깨 닳았고, 나중에 추가 기능을 구현하는 활동을 할 때는 관심있는 부원과 협업해서 진행해야 해야겠다 느꼈습니다.

이후에는 모든 과정을 블로그에 기록하고, 회고록을 작성하면서 부족한점과, 더욱 성장하기 위해 공부해야 할 것을 파악해야 겠다고 생각했습니다. 아무쪼록 이러한 활동을 하면서 부원들이 좋아하고, 실제로 내가 만든 서비스를 사용해주고, 동아리에 조금이라도 기여했다는 것이 너무 뿌듯했습니다.

**외부 소스코드 참조**

기초적인 redux-saga와 컴포넌트, Koa Framework을 활용하는 틀은 “리액트를 다루는 기술” (velopert 저자)를 참고하였습니다.

그리고 MUI, 등 스타일링 규칙이나 공개되어 있는 오픈소스 등을 활용하여 스타일링 하였습니다.