**<node, react 기초 강의>**

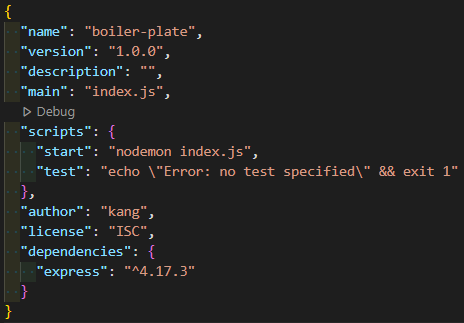
**1. express 개발 환경 세팅**

* express 설치
  + terminal에 npm install express --save 입력

**# index.js**



**# package.json**

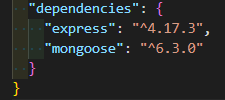


- npm run start 입력 시 "scripts"의 "start"에 입력된 명령문이 실행되어 index.js가 실행

**2. MongoDB와 연결**

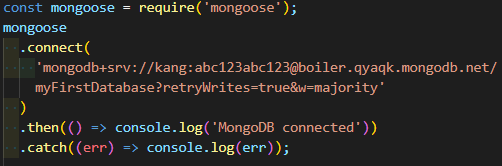
* Mogoose 다운로드
  + terminal에 npm install mongoose --save 입력

**# package.json**

****

- "dependencies"에 mogoose가 성공적으로 추가되었으면 성공적으로 설치된 것이다.

**# index.js**

****

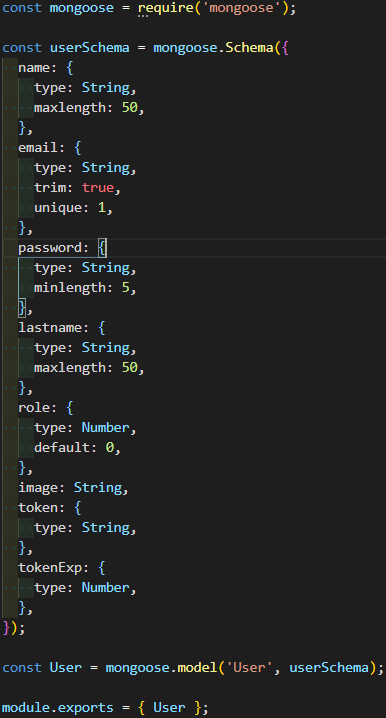


- 위와 같이 코드를 작성하여 MongoDB와 연결이 성공적으로 이루어지면 console창에 연결 성공 메시지 출력

**3. User Model 생성**

* Model VS Schema
  + 기본적으로 몽구스 모델은 몽구스 스키마를 감싸는 것.
  + Schema: 문서 구조, 기본값, 유효성 등을 정의
  + Model: DB 생성, 갱신, 쿼링(querying), 삭제 등을 위한 인터페이스를 제공

**# models/User.js**



- 위와 같이 모델과 스키마를 작성하고 외부에서도 활용하기 위해 module.exports를 입력하여 User model을 내보낸다.

**4. GIT**

GitHub server와 컴퓨터가 안전하게 통신할 수 있도록 ssh key 설정 필요

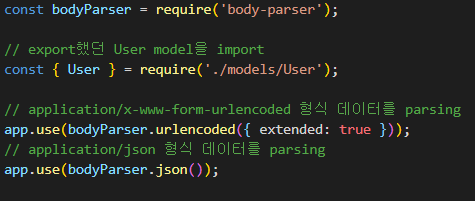
- 구글에 git ssh key 검색하여 차근차근 따라하기

**5. Body Parser & PostMan & 회원가입**

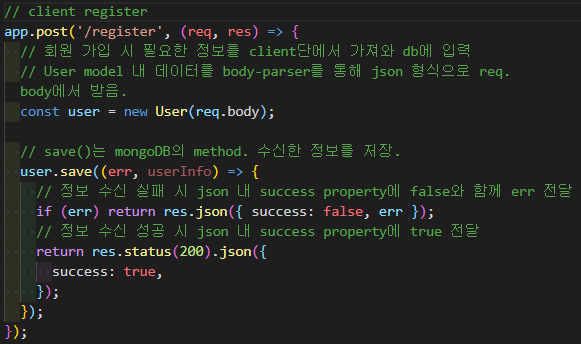
클라이언트에서 서버에 보낸 정보를 서버 측에서 받기 위해서는 bodyParser를 통해 정보들을 parsing해줘야 함.

- terminal에 npm install body-parser --save 입력하여 bodyParser를 설치

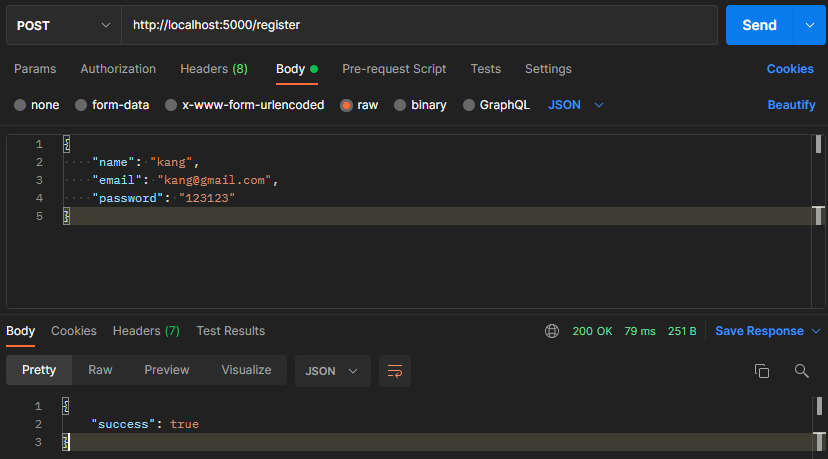
**# index.js**

****

- User model 및 body-parser import. body-parser는 두 가지 데이터 형식을 parsing함.



**testing on postman**

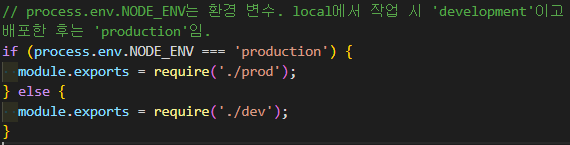
****

- json데이터의 프로퍼티 success에 true라고 출력되면 client단에서 입력된 데이터 서버로 성공적으로 전송한 것임.

**6. 정보 외부 노출 방지**

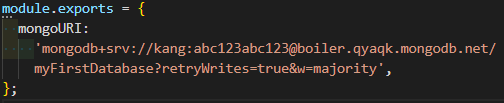
index.js 내에 mongoDB와 연결하기 위해 작성한 URI값을 그대로 gitHub에 commit하면 비밀번호가 노출됨. 따라서 암호화 작업을 해야 함.

**# config/key.js**

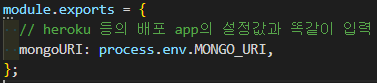
****

- 개발 환경에 따라 export하는 module 파일이 달라짐.

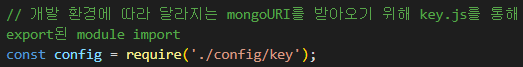
**# config/dev.js**

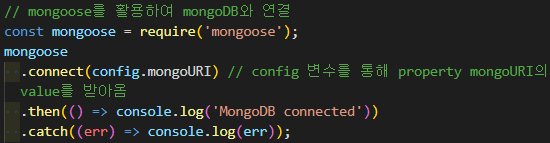


**# config/prod.js**



**# index.js**

****

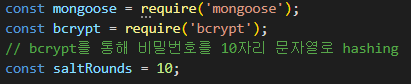
****

**7. bcrypt로 비밀번호 암호화하기**

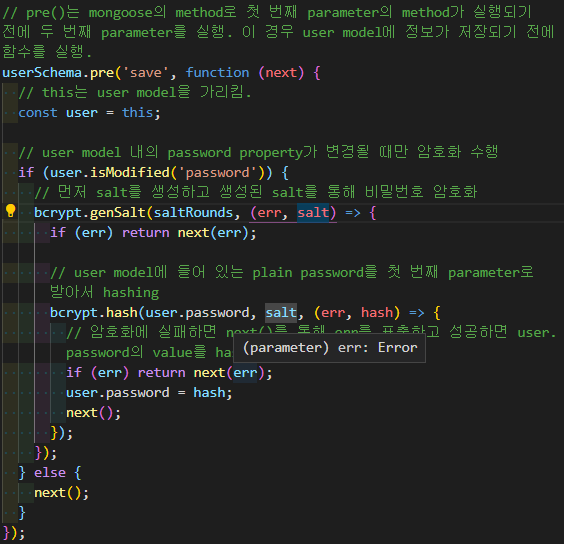
회원가입 시 입력한 비밀번호를 bcrypt로 암호화할 수 있음.

- terminal에 npm install body-parser --save 입력하여 설치

**# models/User.js**

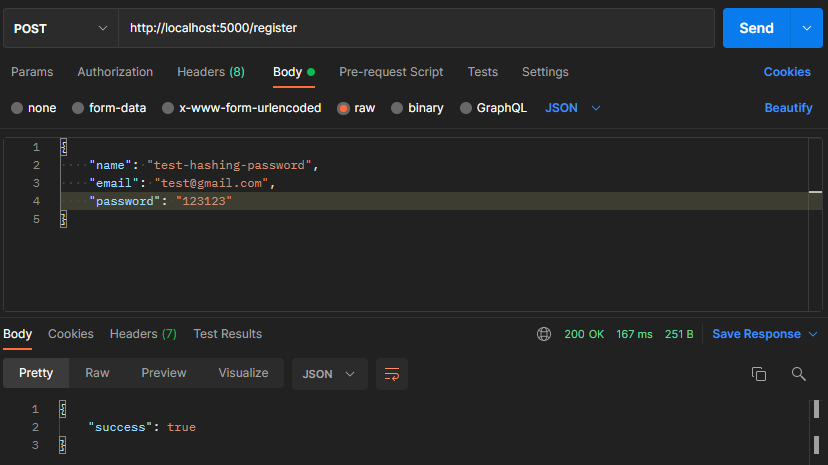
****

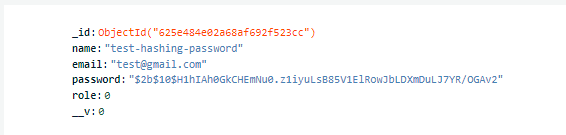
- 암호화를 위한 bcrypt 라이브러리 import



- 여기서 pre() 메서드의 두 번째 parameter에 화살표 함수를 넣으면 err가 발생한다. 왜냐하면 전역 범위에 설정된 userSchema를 갖고 와서 this로 지정해야 하는데 화살표 함수를 넣게 되면 해당 함수의 context가 this로 들어가기 때문에 err 발생.

**# testing on postman & check mongoDB**

****

****

**8. Login with Bcrypt**

**install libraries**

- jsonwebtoken: token 생성(npm install jsonwebtoken --save)

- cookie-parser: 생성된 token을 cookies에 저장(npm install cookie-parser --save)

**# models/User.js**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**# index.js**

**텍스트이(가) 표시된 사진

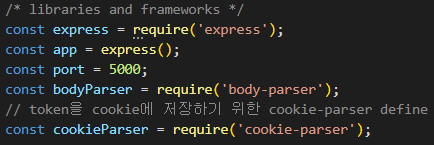
자동 생성된 설명**

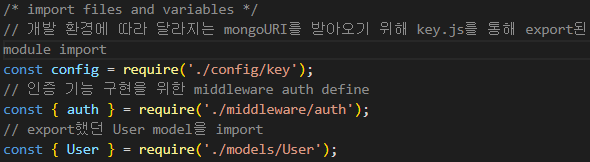
**9. Authentication**

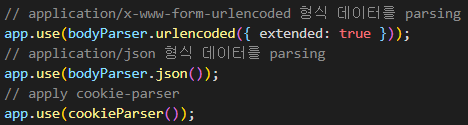
**install libraries**

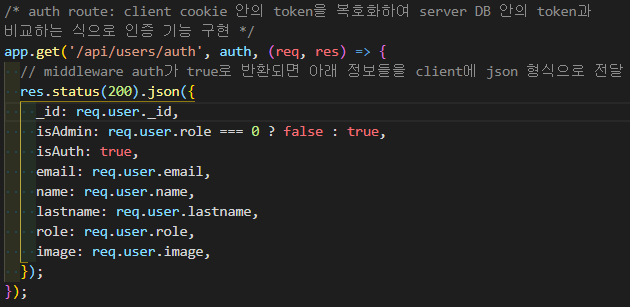
**-** cookie-parser: 생성된 token을 client cookie에 저장(npm install cookie-parser --save)

**# index.js**

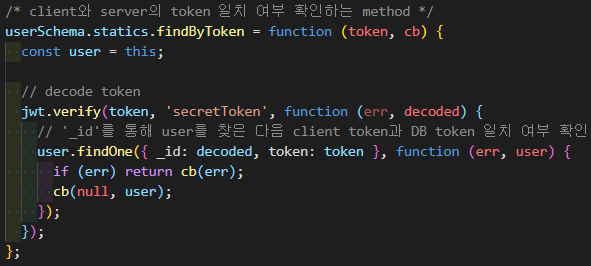
****

****

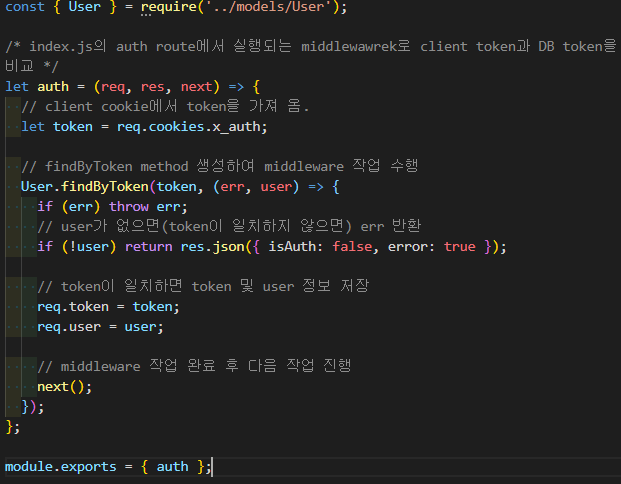
****

****

**# models/User.js**

****

**# middleware/auth.js**

****

**10. React**

**Real DOM vs Virtual DOM**

- real DOM: 전체 요소 중 하나가 update되면 전체 요소 모두를 reload해야 함

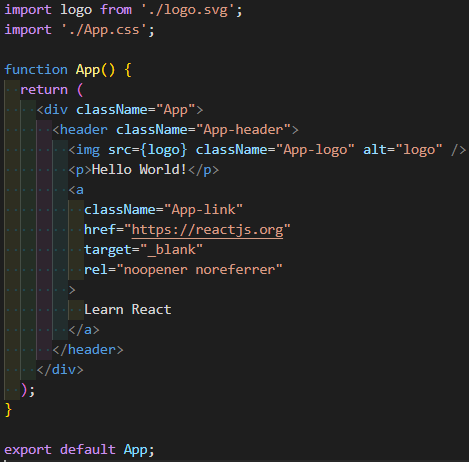
- virtual DOM(React의 기본 원리): 전체 요소 중 하나가 update되면 update된 요소만을 찾아 real DOM에서 그 요소만을 reload함.(diffing)

**11. get started React**

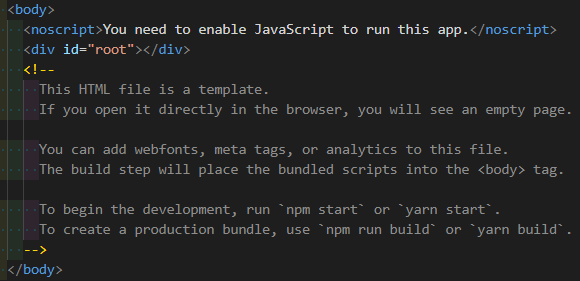
**installation**

- npx create-react-app .

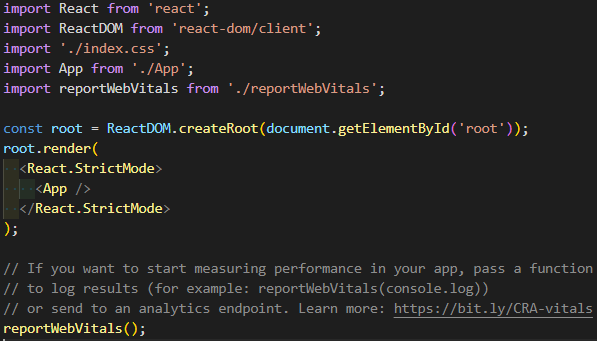
**# src/App.js**

****

**# public/index.html**

****

**# src/index.js**

****

- index.html: id='root' div 정의

- App.js: export function App

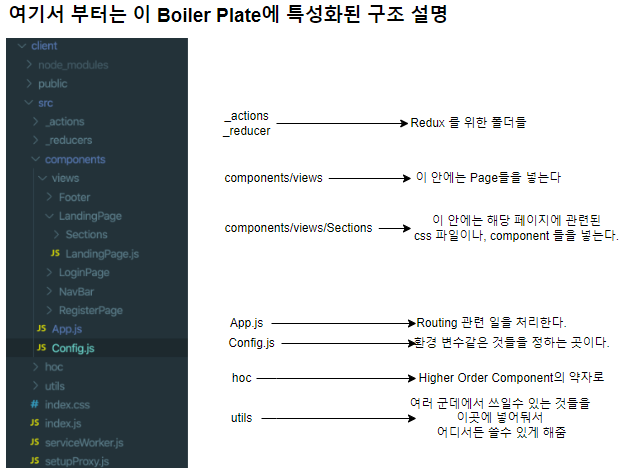
- index.js:

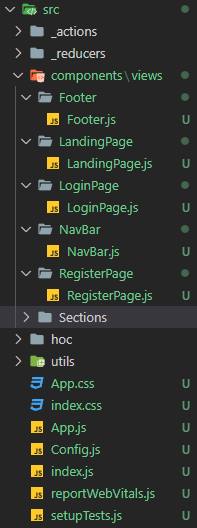
- variable root 생성(id='root' div)

- variable root에다 App import하여 기능 구현

**# directory structures**

****

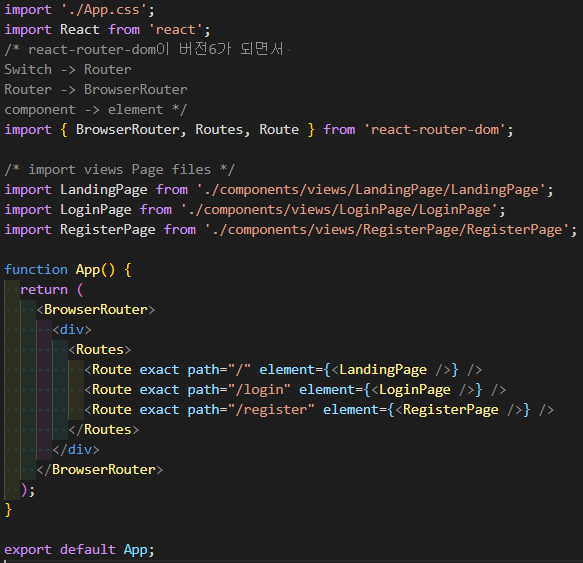
****



- boiler-plate 구현에 필요한 폴더 및 js파일들을 위와 같이 설정

- views 내 Page.js 생성 시 rfce 입력하면 폼 자동 완성됨.

**12. React router DOM**



**13. Flow & Axios, CORS 이슈, Proxy 설정**

**installation**

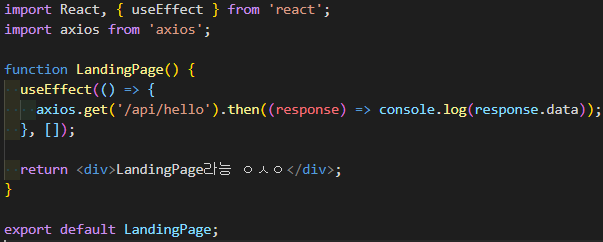
**-** axios:npm install axios --save at client

**-** proxy:npm install http-proxy-middleware --save at client

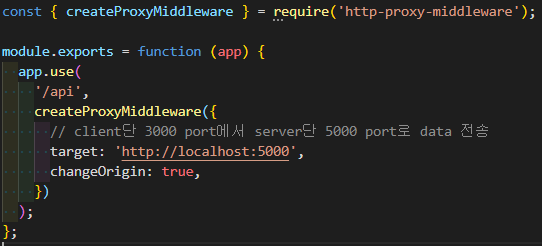
- concurrently: fe/be server 동시 구동(npm install http-proxy-middleware --save at server)

**client**

**# views/LandingPage/LandingPage.js**

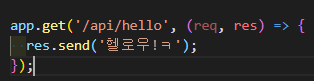
****

**# src/setupProxy.js**

****

**server**

**# index.js**

****

**# ./package.json: concurrently settings**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**14. CSS Frameworks**

- ant design: npm install antd --save at client

**# index.js**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**15. Redux**

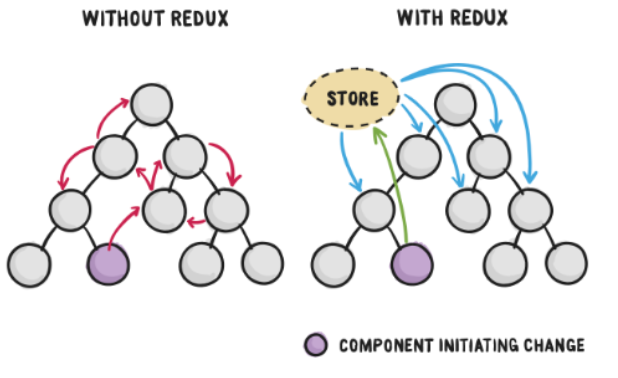
**installation**

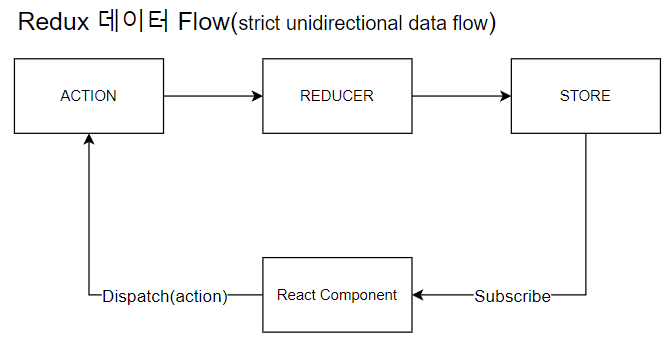
- npm install redux react-redux redux-promise redux-thunk --save at client

**Props VS State**

- Props: 부모 요소로부터 값을 물려 받음. 따라서 자녀 요소에서는 수정 불가능. 수정하려면 부모 요소를 수정해야 함.

- State: State는 Props처럼 부모 요소로부터 값을 물려 받는 게 아닌 해당 component 안에서 요소 수정 가능. Redux는 이러한 State를 관리하는 툴임.





텍스트이(가) 표시된 사진

자동 생성된 설명

**# index.js**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**# \_reducers/index.js**

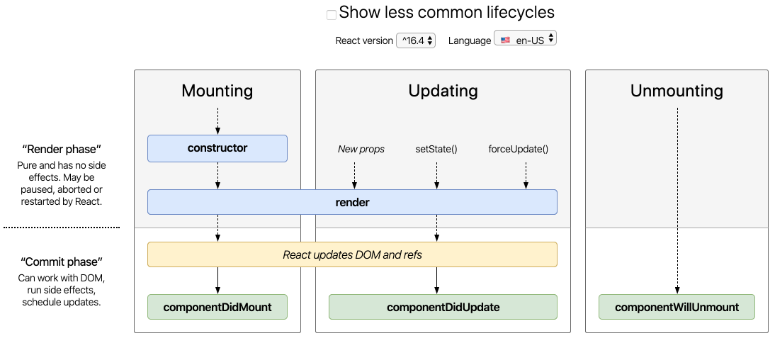
**텍스트이(가) 표시된 사진

자동 생성된 설명**

**16. React Hooks**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

****

**텍스트, 모니터, 스크린샷, 화면이(가) 표시된 사진

자동 생성된 설명**

**17. Login Page**

**# LoginPage/LoginPage.js**

**텍스트이(가) 표시된 사진

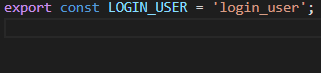
자동 생성된 설명**

**# \_actions/user\_action.js**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**# \_actions/types.js**

****

**# \_reducers/user\_reducer.js**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**# \_reducers/index.js**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

1. loginPage에서 userState를 통해 감지된 email 및 password 입력값을 body에 저장하여 dispatch 메서드의 인수로 입력
2. user\_action.js의 loginUser 함수를 통해 client에서 server로 전달된 data를 변수 request에 저장하여 redux로 전송
3. user\_action.js의 loginUser 함수의 return값 type이 'LOGIN\_USER'라면 user\_reducer.js를 거쳐 redux에 data 전송
4. 이때 전송되는 data는 server단을 거쳐 반환되는 값으로 redux dev tool에 다음과 같이 표시됨

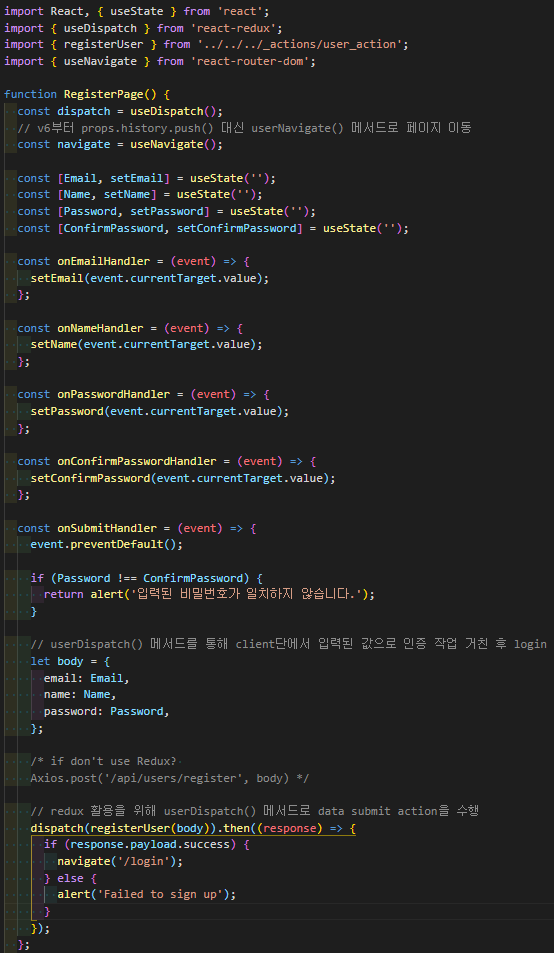
텍스트, 모니터, 스크린샷, 검은색이(가) 표시된 사진

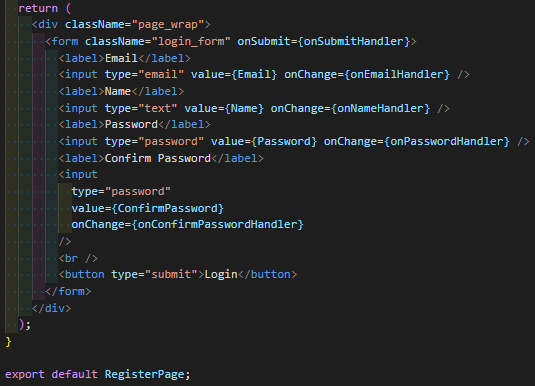
자동 생성된 설명

- login route 성공 반환값인 loginSuccess: true 및 userId가 잘 반환된 것을 볼 수 있다.

**18. Register Page**

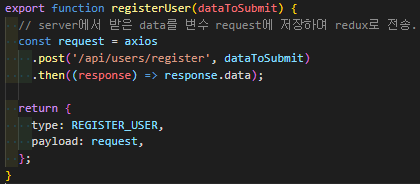
**# RegisterPage/RegisterPage.js**

****

****

**# \_actions/user\_action.js**

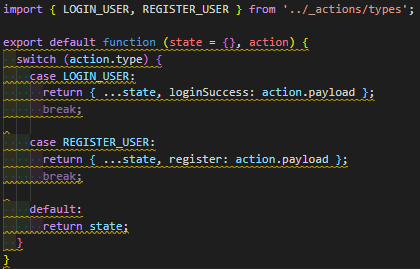
****

****

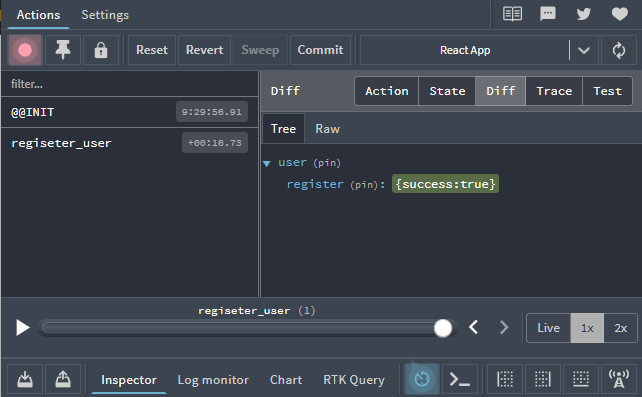
**# \_actions/types.js**

****

**# \_reducers/user\_reducer.js**

****

**if success to register?**

****

**19. Logout**

**# LandingPage/LandingPage.js**

****

**20. Authentication Check**