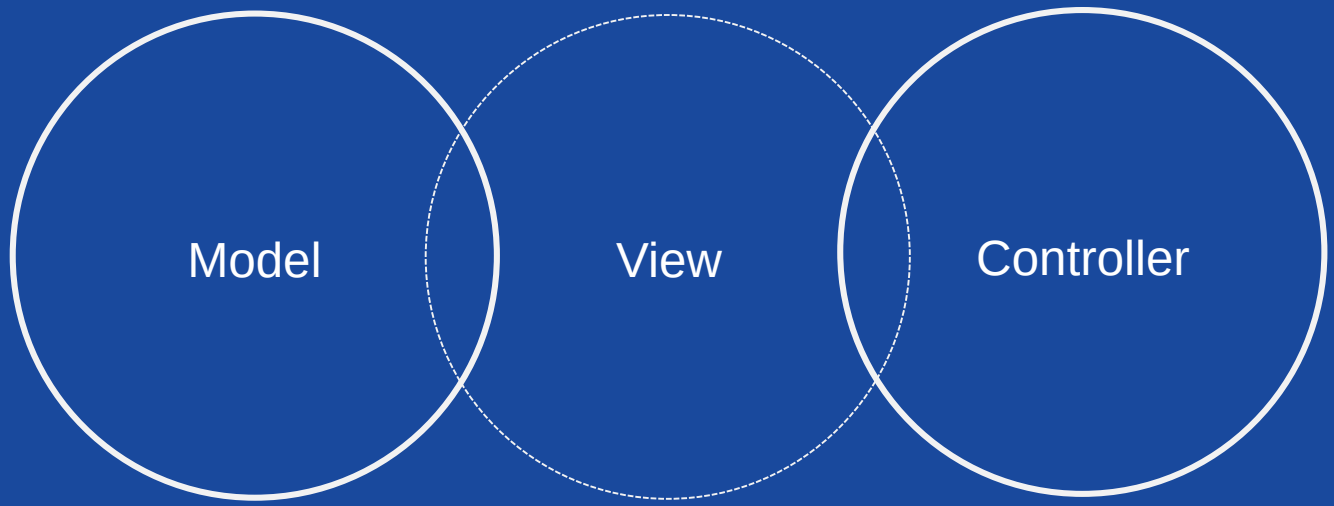
WEBSTAGRAM

Nodejs web project

social network service
using infinity scroll

1. 팀소개



Hyunsu Nam	Paul Kim	Eunbin Son
모델 설계 및 관리, 사용자 기능 설계 및 구현	뷰 설계 및 관리, 구현, 프론트 로직 관리	컨트롤러 설계 및 구현, 인증 관리 및 구현

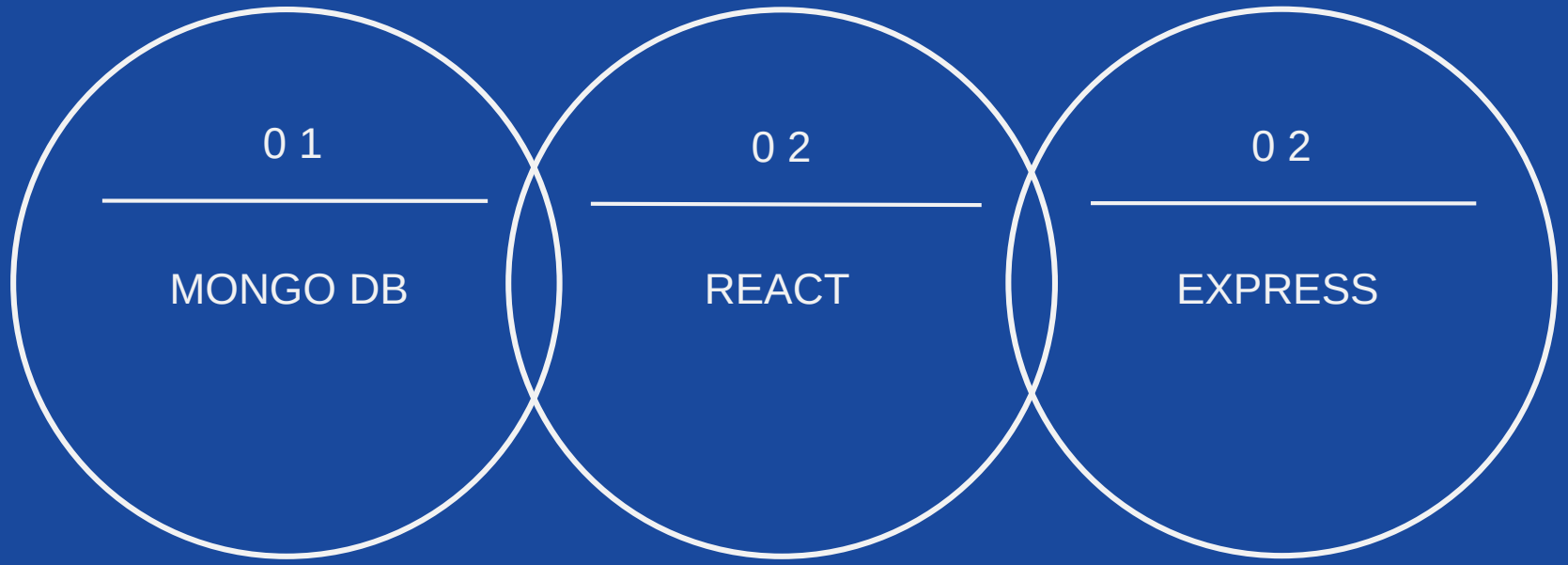
2. 프로젝트 소개

React, express를 사용한 social networking service입니다.

배운 기술을 활용하여 sns서비스를 만드는 것이 이번
프로젝트의 주제였습니다. nodejs를 기반으로 구현하였습니다.
최신 javascript문법과 redux, redux hook등의
최신 기능을 적용하였습니다.

사용자 인증 기능 및, 게시물 업로드와 코멘트 작성, 프로필 변경
등의 기능을 제공합니다. 또한, 제한없는 social networking service의
제공을 위하여 무한 스크롤을 기능을 제공합니다.

3. 개발환경



Mongodb 사용법을 학습하여 적용하였습니다.

프런트 스택으로 리액트 및 리덕스를 적용하였습니다.

배운 기술을 토대로 백엔드를 구성하였습니다.

4. API명세

USER AUTH

User	/api/users	GET	profile	사용자 프로필 변경
Auth	/api/auth	GET	auth	인증
	/api/auth	GET	logout	로그아웃
	/api/auth	POST	login	사용자 로그인
	/api/auth	POST	signup	사용자 회원가입

4. API명세

POST COMMENT

Post	/api/posts	GET	get	게시물 요청
	/api/posts	POST	img	이미지 업로드
	/api/posts	POST	create	게시물 생성
	/api/posts	POST	like	좋아요 등록
Comment	/api/comments	POST	get	댓글 요청
	/api/comments	POST	create	댓글 생성

USER

	API	Method	Role	Parameters	
1	/api/users/profile	post	사용자 정보 요청 (프로필)	userID	사용자 id
	/api/users/profileUpdate		사용자 정보 업로드 (프로필)	userID	사용자 id
				location	거주지역 정보
				birthday	사용자 생일
				job	사용자 직업
				profileImg	프로필 사진
	/api/users/following		팔로우 관계 생성	followingID	팔로우 하는 사용자의 ID
	/api/users/getFollowing		팔로잉 친구 정보 요청	followerID	팔로워 정보를 얻으려는 사용자 ID

POST

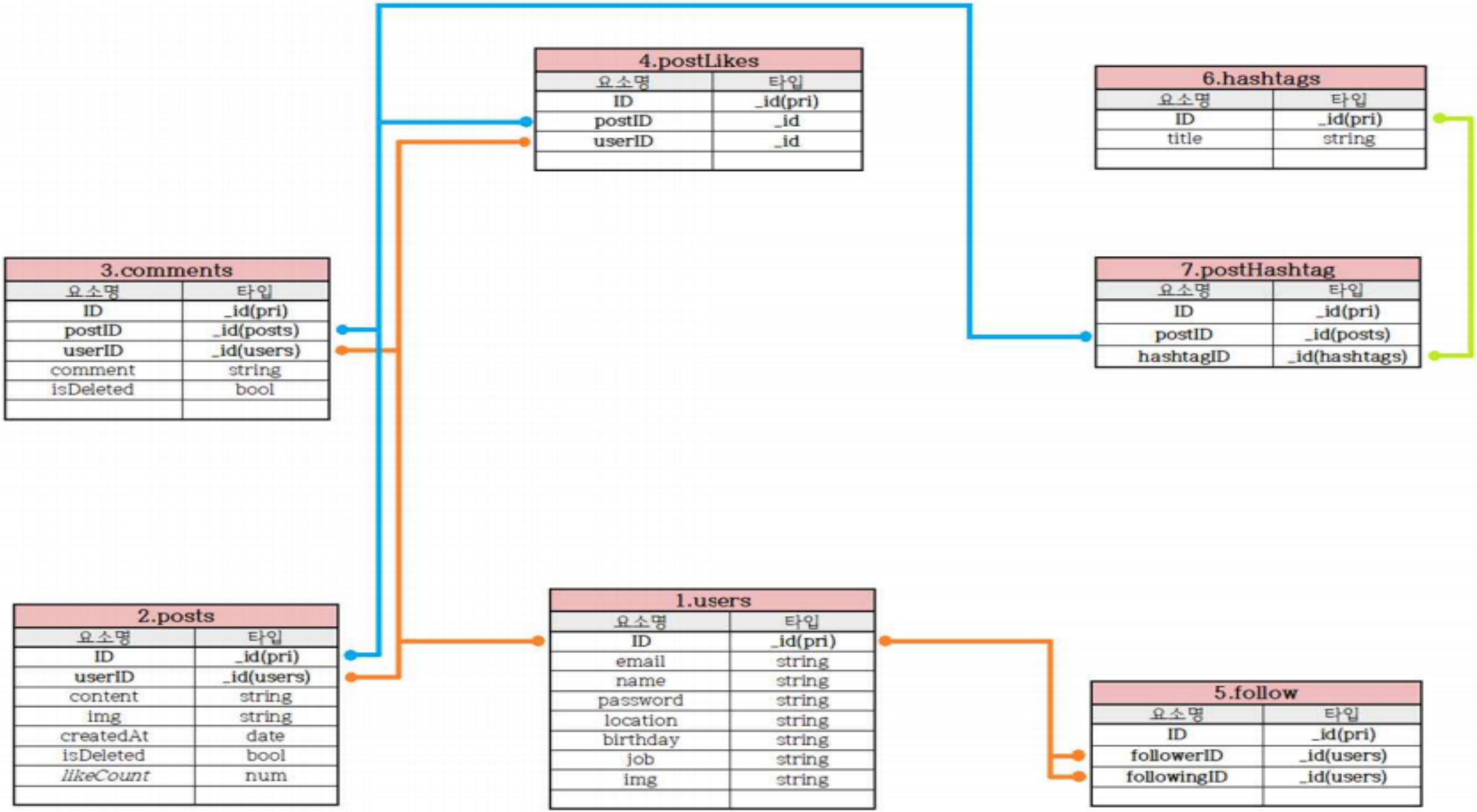
	API	Method	Role	Parameters	
1	/api/posts/get	GET	게시물 정보 요청	-	-
2	/api/posts/img	POST	이미지 업로드	img	이미지
				url	이미지 주소
3	/api/posts/create		게시물 생성 요청	userID	게시글 작성자
				content	게시글 내용
				postImg	게시글 이미지
4	/api/posts/like		좋아요 정보 생성 요청	postID	해당 글 id
				userID	사용자 id

AUTH

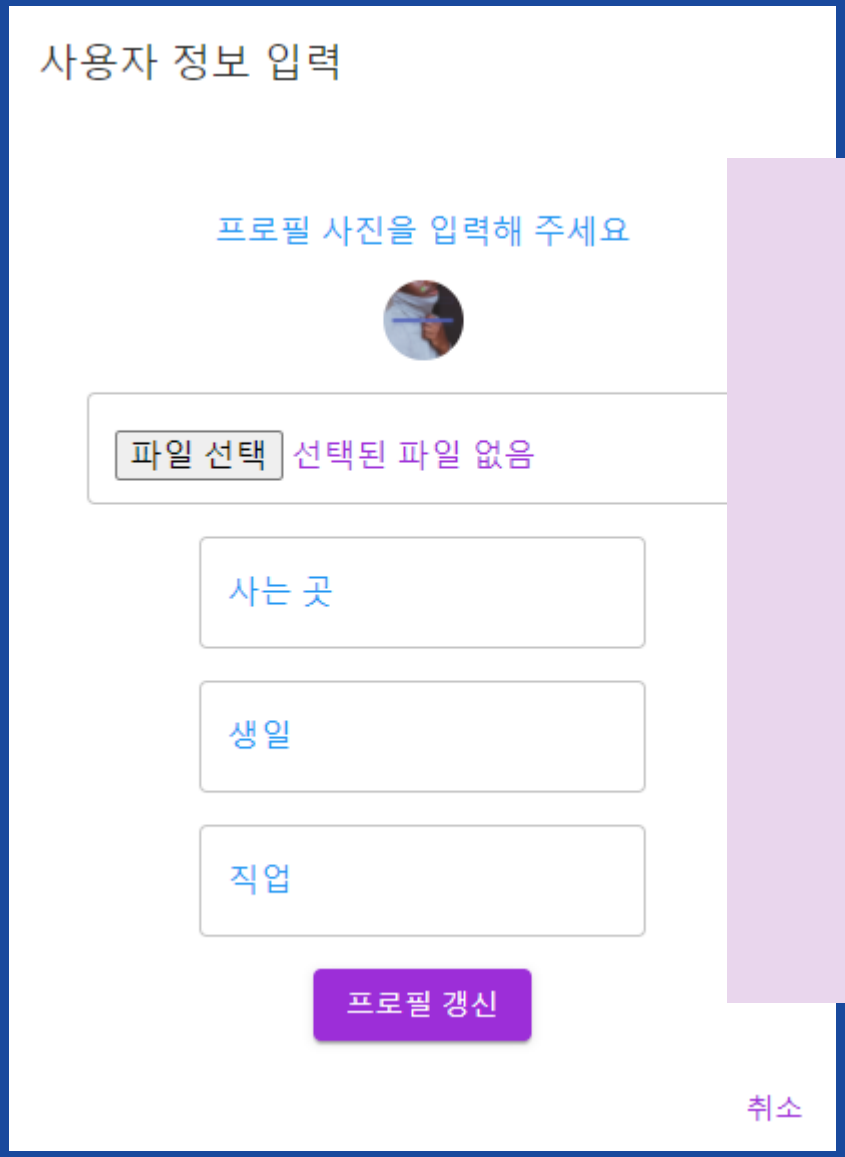
	API	Method	Role	Parameters	
1	/api/auth/auth	GET	사용자 정보 인증	_id	사용자 id
				isAdmin	관라자 여부
				isAuth	인증 여부
				email	사용자 이메일
				name	사용자 이름
				location	거주 지역
				birthday	생일
				job	직업
				profileImg	프로필 이미지
				following	팔로잉
	/api/auth/logout		사용자 로그아웃	isAuth	인증 여부
				_id	사용자 id
2	/api/auth/login	POST	사용자 로그인/ 회원가입	email	이메일
				password	비밀번호
	/api/auth/signup			email	이메일
				name	이름
				passward	비밀번호

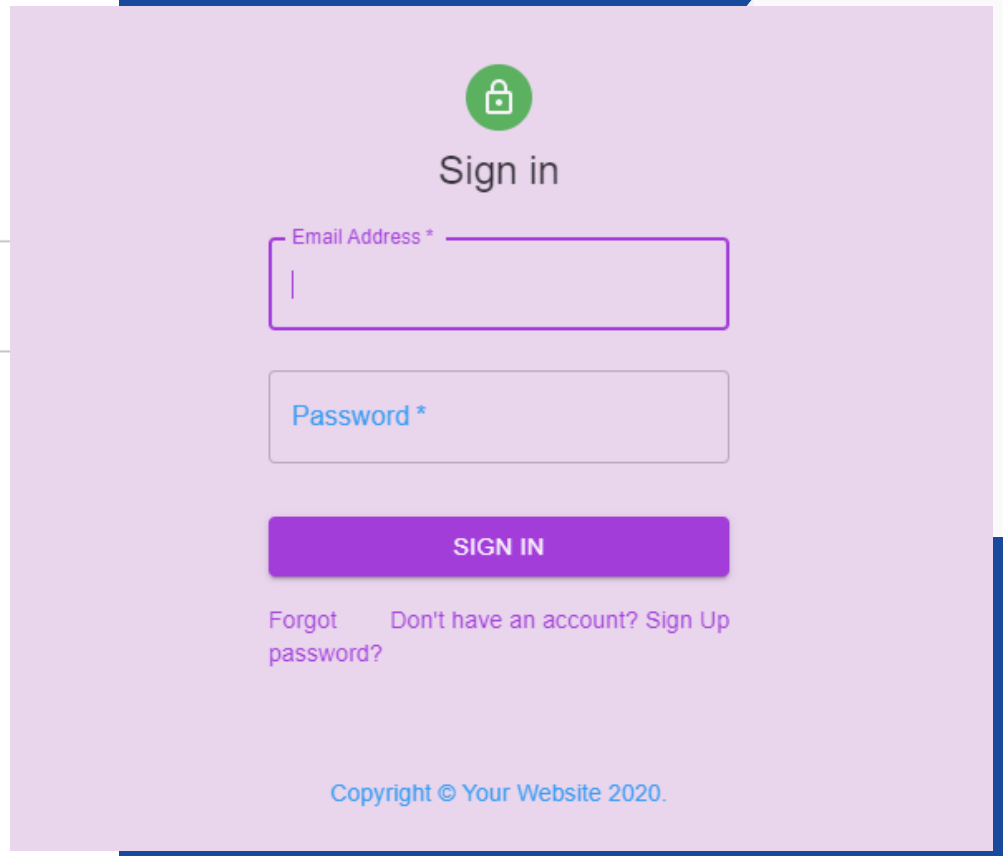
COMMENT

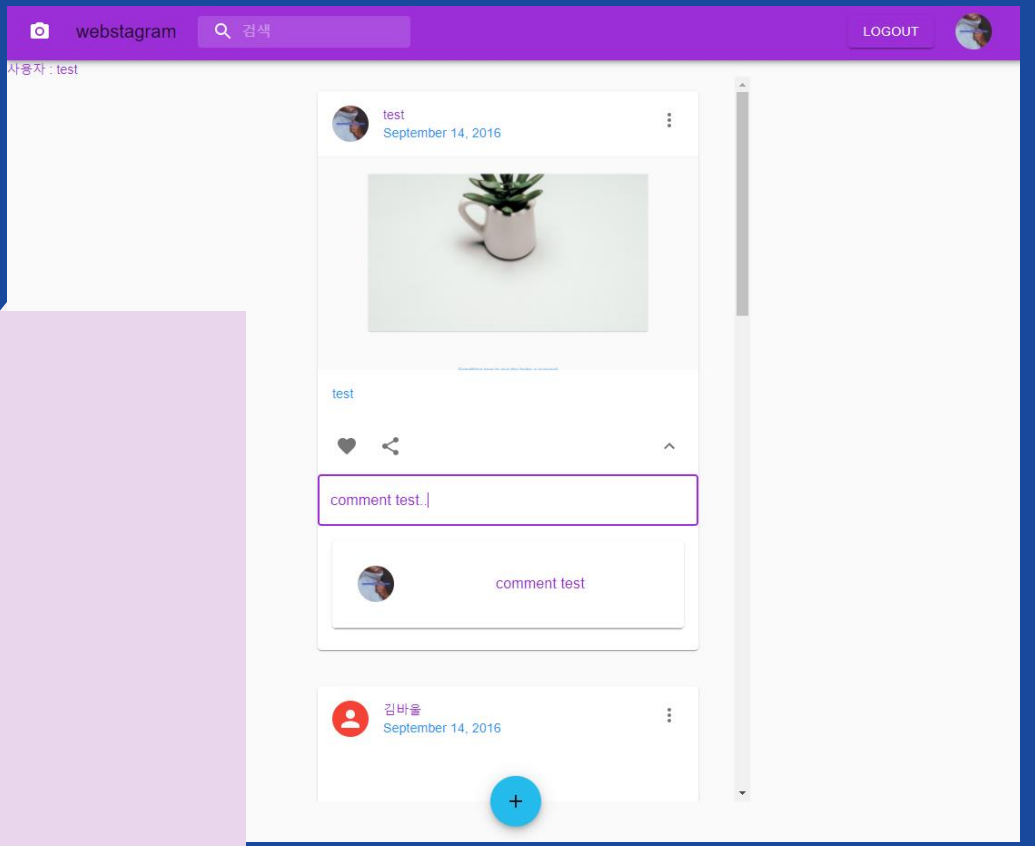
	API	Method	Role	Parameters	
1	/api/comments/create	POST	댓글 생성 요청	userID	작성자 id
				postID	해당글 id
				comment	댓글 내용
2	/api/comments/get		댓글 정보 요청	postID	해당글 id


4.postLikes
요소명 타입
ID _id(pri)
postID _id
userID _id
6.hashtags
요소명 타입
ID _id(pri)
title string
3.comments
요소명 타입
ID _id(pri)
postID _id(posts)
userID _id(users)
comment string
isDeleted bool
7.postHashtag
요소명 타입
ID _id(pri)
postID _id(posts)
hashtagID _id(hashtags)
2.posts
요소명 타입
ID _id(pri)
userID _id(users)
content string
img string
createdAt date
isDeleted bool
likeCount num
1.users
요소명 타입
ID _id(pri)
email string
name string
password string
location string
birthday string
job string
img string
5.follow
요소명 타입
ID _id(pri)
followerID _id(users)
followingID _id(users)

5. 프로젝트 화면


사용자 정보 입력
프로필 사진을 입력해 주세요
파일 선택
선택된 파일 없음
사는 곳
생일
직업
프로필 갱신
취소


Sign in
Email Address *
Password *
SIGN IN
Forgot password?
Don't have an account? Sign Up
Copyright © Your Website 2020.


webstagram
검색
LOGOUT
사용자 : test
test
September 14, 2016
test
comment test..
comment test
김바울
September 14, 2016

6. 향후 개발 방향과 소스코드

프런트/리액트리덕스

```
function Login(props) {
  const dispatch = useDispatch();
  const [Email, setEmail] = useState("")
  const [Password, setPassword] = useState("")
  const onEmailHandler = (event) => {
    setEmail(event.currentTarget.value)

  const onPasswordHandler = (event) => {
    setPassword(event.currentTarget.value)

  const onSubmitHandler = (event) => {
    event.preventDefault(); //페이지가 리프레시 되는 것을 막는다.
    console.log('Email', Email);
    console.log('Password', Password);
    let body = {
      email: Email,
      password: Password
    }
    dispatch(loginUser(body)) //action을 취하기 위한 메서드이다.
      .then(response => {
        if (response.payload.loginSuccess) {
          props.history.push('/') //리액트에서 페이지를 이동하는 방법
        } else {
          alert('Error')
        }
      })
  }
  const oAuthLoginHandler = (resData) => {
    console.log(resData);
    const {id} = resData.profile;
    const {email} = resData.profile.kakao_account;
    let body = {
      oAuthId: id,
      email: ema ,il
    }
    dispatch(loginUser(body))
  }
```

개발 계획 내용
1. 게시물 삭제 및 수정
2. 좋아요 추가 및 취소
3. 팔로잉 및 언팔로우
4. 동영상 업로드
5. 게시물 검색기능
6. 해시태그 등록 및 검색
7. 카카오, 페이스북 등 소셜로그인

```
try {
  fs.readdirSync('client/public/uploads');  //
} catch (error) { // 디렉토리를 못 찾았을 경우
  console.error('uploads 폴더가 없어 uploads 폴더
  fs.mkdirSync('client/public/uploads');  // 아
}


const upload = multer({
  storage: multer.diskStorage({
    destination(req, file, cb) {
      cb(null, 'client/public/uploads');
    },
    filename(req, file, cb) {
      const ext = path.extname(file.originalname);
      cb(null, path.basename(file.originalname, ext) + Date.now() + ext);  // 파일을 내려받을 때 아마,
    },
  }),
  limits: { fileSize: 5 * 1024 * 1024 }, // 올릴 수 있는 파일 사이즈?
});

  router.post('/img', upload.single('postImg'), (req, res, next) => {  // '/img' 요청이 온다면, 로그인
    const url = req.file.filename;
    console.log(url);
    res.json({ url: url });  // 응답을 json 파일로 하는 듯,
  });
```

백/이미지 업로드