

#3. INFO .

NAME : CRUD 게시판

URL : <https://react-jun.herokuapp.com/>

SPEC :      

RESPONSIVE : PC, TABLET



Hello !

최근 등록 글

EURA
포트폴리오 사이트에 놀러오세요.
저의 모든 작업물이 들어있는 포트폴리오입니다.
<https://eurako.github.io/eurafolio/>

2022.05.31.

EURA
안녕하세요.
노드프로젝트에 오신 것을 환영합니다.

2022.05.31.

EURA
회원가입을 하면 글쓰기가 가능합니다.
하단의 JOIN버튼을 클릭하여 간단 회원가입을 하면 글쓰기가 가능합니다.

2022.05.30.

EURA
What is Lorem Ipsum?
Lorem Ipsum is simply dummy text of the printing and typesetting industry.

2022.05.27.



#3_1. CONCEPT

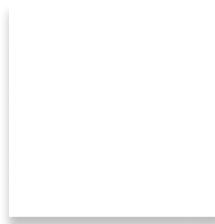
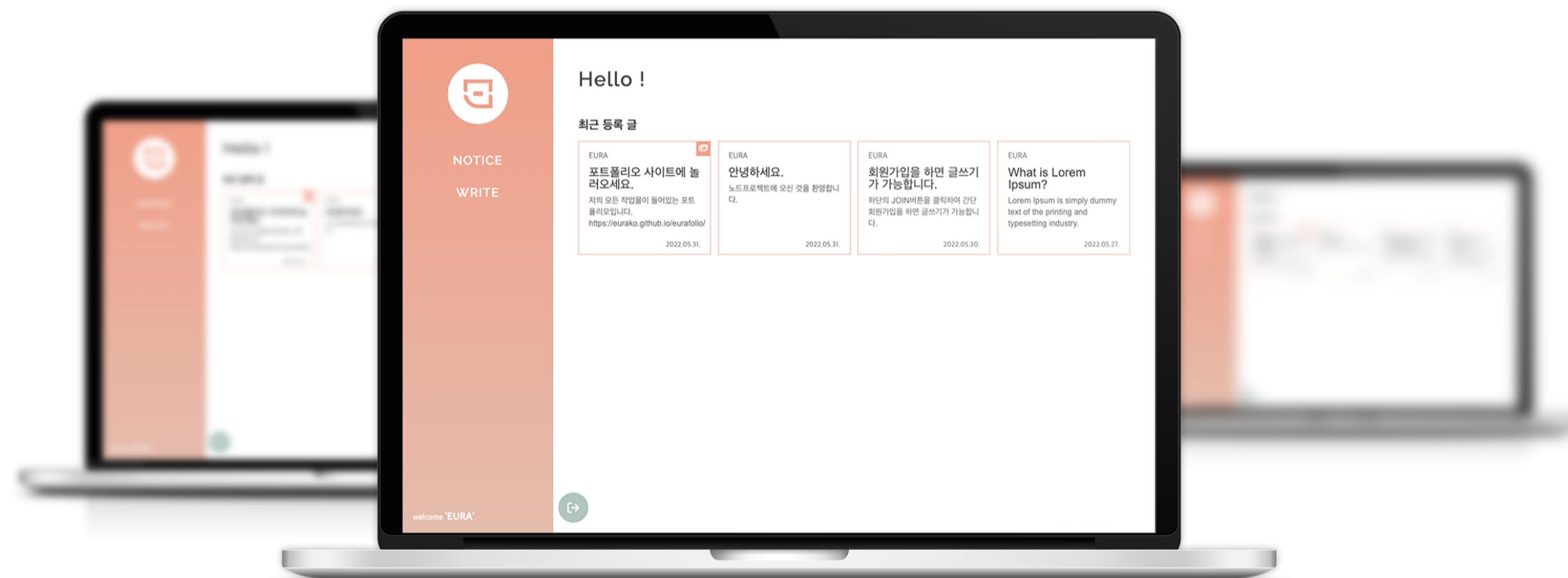
SERVER : node JS

DB : mongo DB

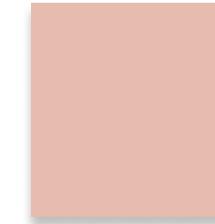
회원인증 : fire base

배포 : hefoku

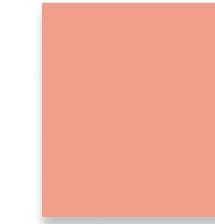
이미지 외부 저장소 : naver cloud



#FFFFFF



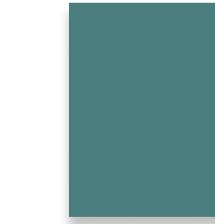
#E5BCAD



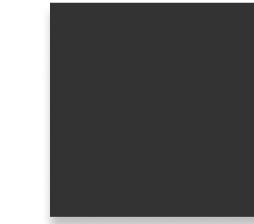
#F19F88



#AFC5BD



#4A7D7D



#333333

#3_2. NODE STRUCTURE

- node 서버와 리액트 폴더를 따로 분리해서 관리



APP



img

node서버에 등록될 이미지 폴더



react

view 폴더



server

node js 폴더



cloud

heroku에서 서버가 꺼지면 등록된 이미지가 날아가는 현상이 생겨 네이버클라우드에서 이미지를 등록



config

노출되면 안되는 정보를 담는 배포와 개발 시 다른 값을 전달



model

모델 스키마 폴더



router

클라이언트의 요청을 전달할 router



index.js

최종 node의 명령을 수행할 파일



package.json



Procfile

heroku 설정 파일

#3_3. JOIN

- 유효성 인증이 통과하면 firebase를 통해 인증완료 처리 후 redux를 통해 정보를 저장
- client에서 bodyParser로 전달받은 유저정보를 router에서 처리

react/compoments/Join.js

```
let createdUser = await firebase
  .auth()
  .createUserWithEmailAndPassword(email, pw1);

await createdUser.user.updateProfile({
  displayName: name,
});

const body = {
  email: createdUser.user.multiFactor.user.email,
  displayName: createdUser.user.multiFactor.user.displayName,
  uid: createdUser.user.multiFactor.user.uid,
};

axios.post('/api/user/join', body).then((res) => {
  if (res.data.success) {
    navigate('/login');
  } else {
    return alert('회원가입에 실패했습니다.');
  }
});
```

server/router/user.js

```
router.post('/join', (req, res) => {
  const temp = req.body;
  Counter.findOne({ name: 'counter' }).then((doc) => {
    temp.userNum = doc.userNum;

    const userData = new User(temp);
    userData
      .save()
      .then(() => {
        Counter.updateOne({ name: 'counter' }, { $inc: { userNum: 1 } })
      })
      .then(() => {
        res.status(200).json({ success: true });
      })
      .catch((err) => {
        console.log(err);
        res.status(400).json({ success: false });
      });
  });
});
```

#3_4. LOGIN

- 로그인 인증이 되지 않은 경우 에러별 메세지 출력

react/compoments/Login.js

```
const handleLogin = async () => {
  if (!(email && pw)) return alert('모든값을 입력하세요');

  try {
    await firebase.auth().signInWithEmailAndPassword(email, pw);
    navigate('/');
  } catch (err) {
    console.dir(err.code);
    if (err.code === 'auth/user-not-found') {
      setErrMsg('존재하지 않는 메일입니다.');
    } else if (err.code === 'auth/wrong-password') {
      setErrMsg('비밀번호가 일치하지 않습니다.');
    } else if (err.code === 'auth/invalid-email') {
      setErrMsg('이메일 형식이 아닙니다.');
    } else {
      setErrMsg('로그인에 실패했습니다.');
    }
  }
};
```

The image shows two side-by-side screenshots of a mobile-style login form. Both screenshots feature a large orange circular logo with a white stylized letter 'E' at the top. Below the logo are two input fields: 'E-mail' and 'Password'. In the first screenshot, the 'E-mail' field contains 'email@naver.com' and the 'Password' field contains '.....'. Below the inputs are two buttons: 'login' (orange) and 'join' (light blue). A red error message '존재하지 않는 메일입니다.' (Email does not exist) is displayed below the 'login' button. In the second screenshot, the 'E-mail' field contains 'aa@naver.com' and the 'Password' field contains '.....'. The same two buttons are present, and a red error message '비밀번호가 일치하지 않습니다.' (Password does not match) is displayed below the 'login' button.

#3_5. WRITE & EDIT

- 컨텐츠 작성 및 수정
- haroku에서 서버가 닫히면 등록된 이미지가 삭제되는 이슈가 있어 naver cloud를 통해 이미지 업로드
- 이미지 등록시 미리보기 및 파일명 노출

<code> react/compoments/Post.js

```
const onSubmit = () => {
  if (title === "" || content === "") {
    return alert('모든항목을 입력하세요');
  }
  const body = {
    title: title,
    content: content,
    img: img,
    imgName: imgName,
    uid: user.uid,
  };

  axios
    .post('/api/post/submit', body)
    .then((res) => {
      if (res.data.success) {
        alert('글작성이 완료되었습니다.');
        navigate('/list');
      } else {
        alert('글작성에 실패했어요');
      }
    })
    .catch((err) => {
      console.log(err);
    });
};</code>
```



#3_6. LIST

- 작성된 글 리스트 화면에 출력
- 이미지가 있는 경우 아이콘 표시
- 검색 시 바디파서로 해당 내용 전달해 리스트 재출력

[react/components/post/List.js](#)

```
const getList = () => {
  let num = 0;
  if (props.count) num = props.count;
  const body = {
    sort: sort,
    search: search,
    count: num,
  };
  axios
    .post('/api/post/list', body)
    .then((res) => {
      // console.log(res);
      if (res.data.success) {
        // console.log(res.data);
        setList(res.data.postList);
        setLoaded(true);
      }
    })
    .catch((err) => {
      console.log(err);
    });
};
```

[server/router/post.js](#)

```
router.post('/list', (req, res) => {
  const sort = {};
  if (req.body.sort === 'new') {
    sort.createdAt = -1;
  }
  if (req.body.sort === 'old') {
    sort.createdAt = 1;
  }

  Post.find({
    $or: [
      { title: { $regex: req.body.search } },
      { content: { $regex: req.body.search } },
    ],
  })
    .populate('writer')
    .sort(sort)
    .limit(req.body.count)
    .exec()
    .then((doc) => {
      res.status(200).json({ success: true, postList: doc });
    })
    .catch((err) => {
      console.log(err);
      res.status(400).json({ success: false });
    });
});
```



NOTICE

최신순 게시순

SEARCH

EURA 포트폴리오 사이트에 놀러오세요. 저의 모든 작업물이 들어있는 포트폴리오입니다. https://eurako.github.io/eurafolio/ 2022.05.31.	EURA 안녕하세요. 노드프로젝트에 오신 것을 환영합니다.
EURA 회원가입을 하면 글쓰기가 가능합니다. 하단의 JOIN버튼을 클릭하여 간단 회원가입을 하면 글쓰기가 가능합니다.	EURA What is Lorem Ipsum? Lorem Ipsum is simply dummy text of the printing and typesetting industry.
EURA 프로젝트의 개발 스펙 해당 프로젝트의 view는 REACT를 사용했고, server는 mongoDB를 사용했습니다.	EURA 유라의 노드프로젝트입니다. 안녕하세요. 유라의 노드프로젝트입니다. 해당 이미지 저를 나타내는 심볼입니다.
2022.05.30.	2022.05.27.

#3_7. DETAIL

- 리스트 클릭 시 상세 페이지 노출
- 작성일과 최종수정일 따로 표시
- 작성자에게만 수정, 삭제버튼 노출

[react/components/post/Detail.js](#)

```
useEffect(() => {
  const body = {
    postNum: params.postNum,
  };

  axios
    .post('/api/post/detail', body)
    .then((res) => {
      if (res.data.success) {
        setDetail(res.data.post);
        console.log(res.data.post);
        setLoaded(true);
      }
    })
    .catch((err) => {
      console.log(err);
    });
}, []);

```

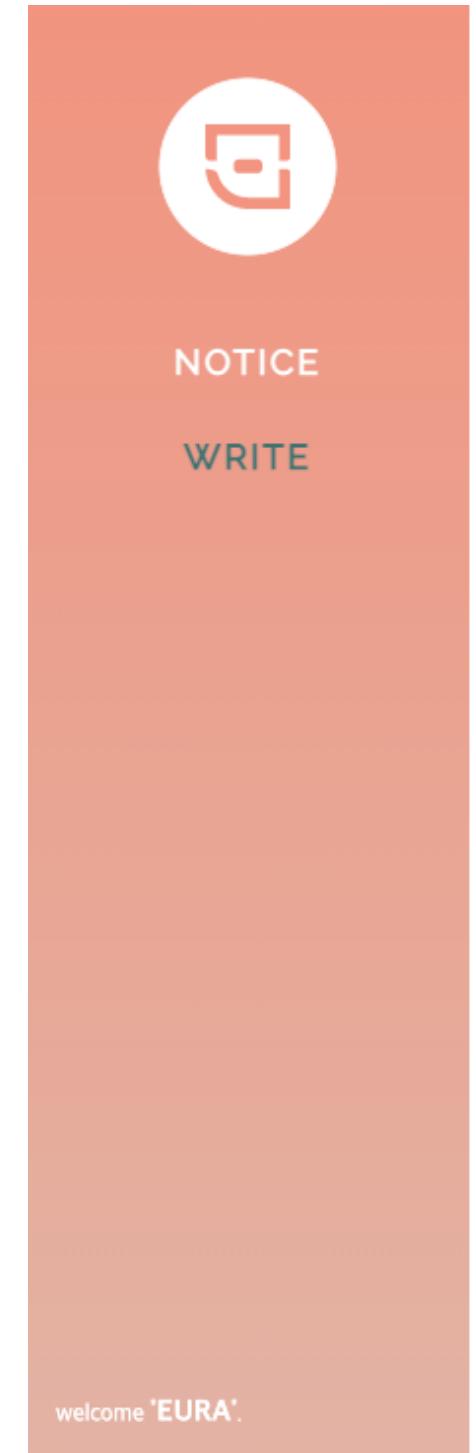
⋮

```
<p>{moment(detail.createdAt).format('L')}</p>
```

```
<p>{moment(detail.createdAt).format('a hh:mm:ss')}</p>
```

[server/router/post.js](#)

```
router.post('/detail', (req, res) => {
  Post.findOne({ postNum:
    Number(req.body.postNum) })
    .populate('writer')
    .exec()
    .then((doc) => {
      res.status(200).json({ success:
        true, post: doc });
    })
    .catch((err) => {
      console.log(err);
    });
});
```



View Detail

작성일	최종 수정일
2022.05.26. 오후 11:23:15	2022.05.30. 오후 02:48:50