



캡스톤 디자인(1)

최종발표

머신러닝을 이용한 영화 탐색 및 추천 웹

목차

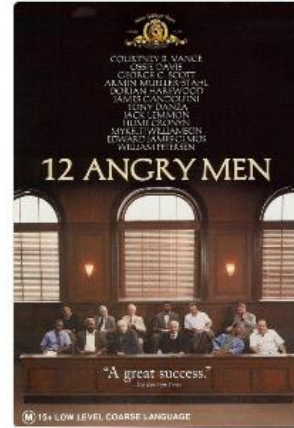
1. 개요 및 목적
2. 요구명세
3. 설계 및 구현
4. 실행 결과
5. 개선 사항 및 향후 계획

개요 및 목적



영화 검색

예상별점이 높은 작품



12인의 노한 사람들
예상 ★4.5

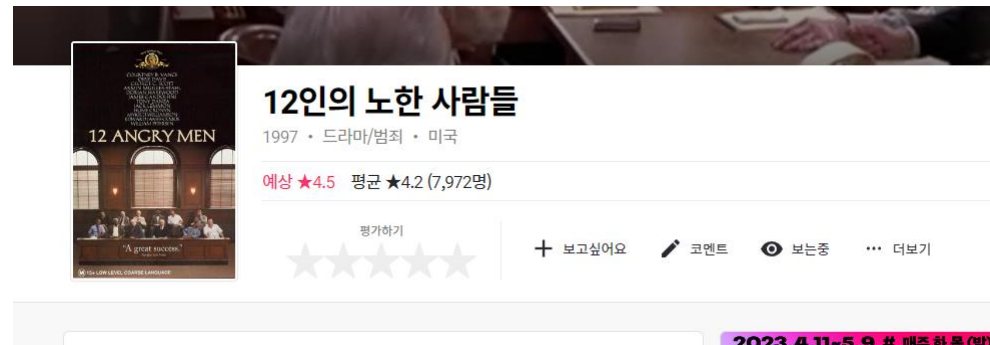


피어나
예상 ★4.1



천국보다 아름다운
예상 ★4.2

예상 별점이 높은 작품 추천



영화 평가, 코멘트 남기기, 커뮤니티

요구 명세

User Requirement

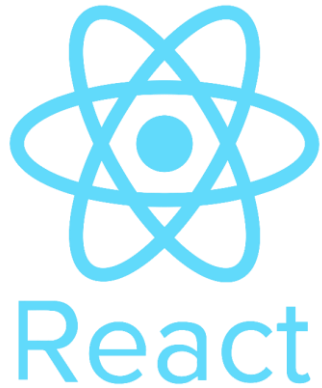
1. 사용자가 원하는 영화를 검색하여 찾을 수 있다.
2. 로그인 이 가능하고 리뷰와 평점을 작성하여 저장 할 수 있다.
3. 작성한 평점을 바탕으로 사용자에게 맞는 영화를 추천해줄 수 있다.

System Requirement

1. URL에 알맞은 페이지를 전송한다.
2. 영화 정보 Api에서 요구한 영화의 상세정보 등을 올바르게 불러온다.
3. 로그인, 리뷰 등의 정보를 DB에 올바르게 작성하고 불러 올 수 있다.
4. 목적에 맞는 머신러닝 Api 를 작성하고 활용 할 수 있다.

소프트웨어 아키텍처

Client - server



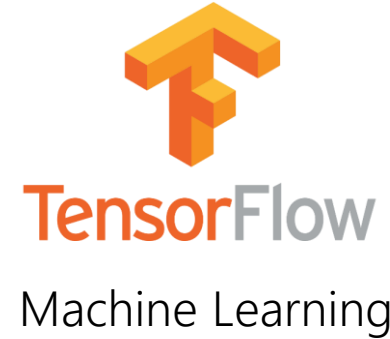
Client



Framework



Server



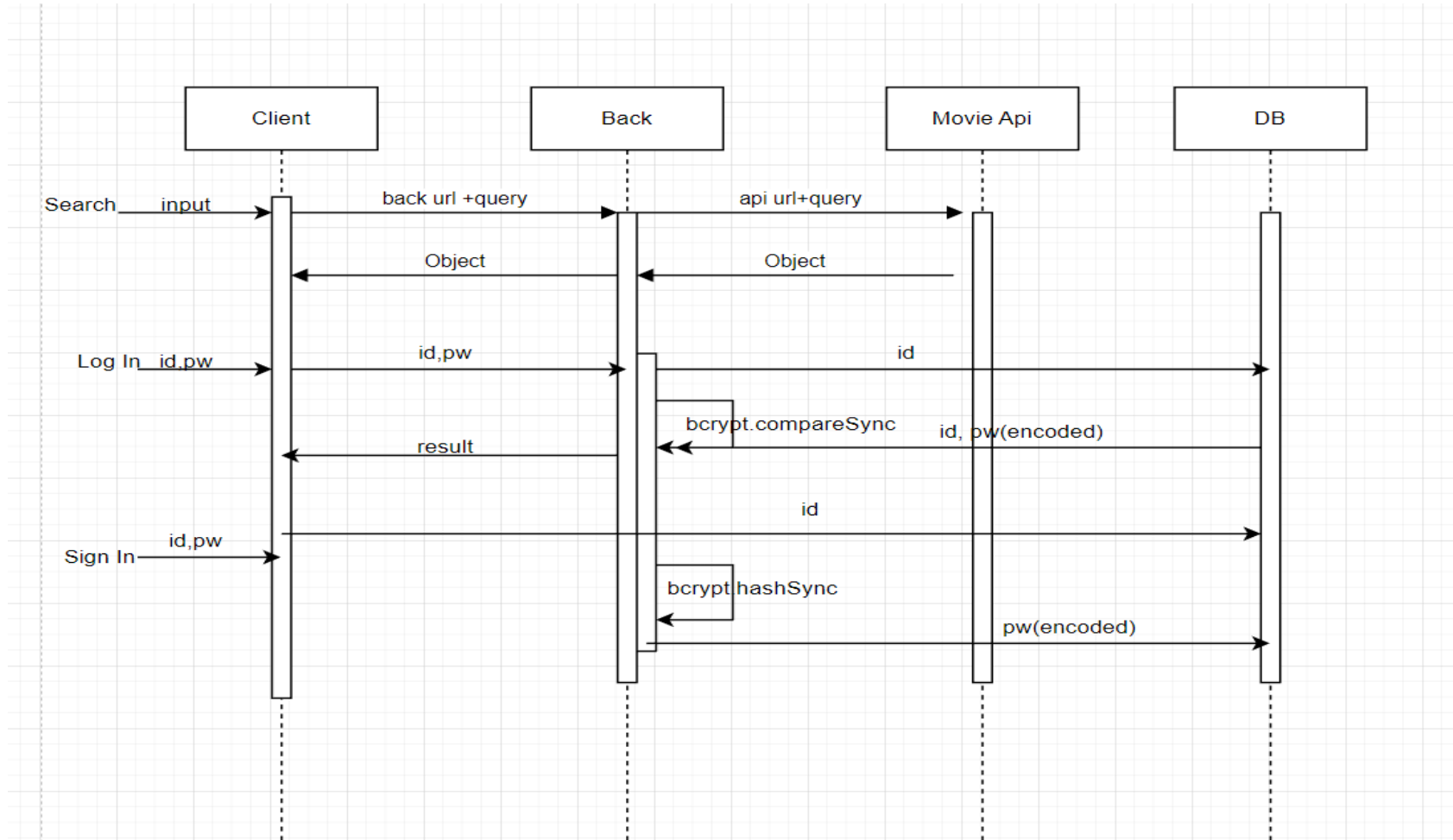
Data Base



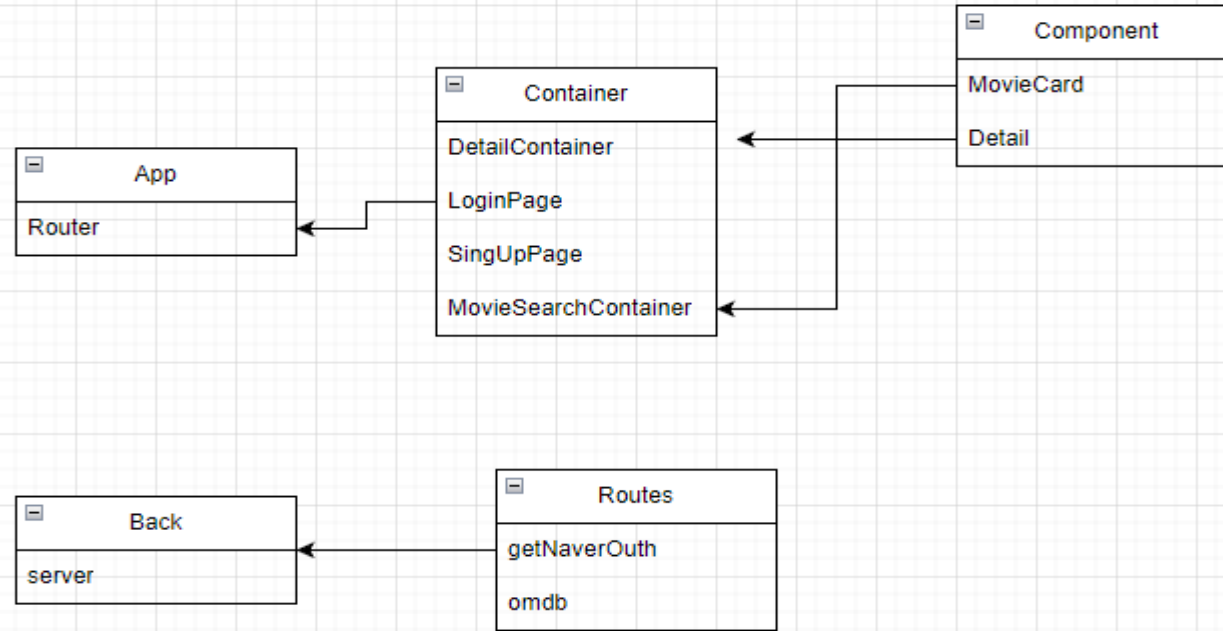
OAuth

OMDB API

Sequence Diagram



Class Diagram



실행 결과

Search

존 워 4

2023년도

평점

8.00/ 10

존 워

2014년도

평점

7.40/ 10

존 워 3:파라벨
룸

2019년도

평점

7.40/ 10

존 워:리로드

2017년도

평점

7.30/ 10

실행 결과

```
/* GET home page. */
router.get('/getOmdbMovie', async function(req,res){
  let query = req.query.query;
  res.setHeader('Access-Control-Allow-Origin', '*');
  res.setHeader('Access-Control-Allow-Credentials', 'true'); // 쿠키 주고 받기
  try{
    let movieRes= await Axios.get(
      `http://www.omdbapi.com/?i=tt3896198&apikey=79033114&s=${query}`
    );
    return res.json(movieRes.data);
  } catch (e) {
    return res.json({
      status : 400,
      message : e
    });
  }
});

module.exports = router;
```

```
const handleButton = async () => {
  try {
    const res = await axios.get("http://localhost:3001/naver/getNaverMovie", {
      params: {
        query: query,
      },
    });
    if (res && res.status === 200) {
      const { data } = res;
      console.log(data);
      setItems(data.items);
    }
  } catch (e) {
    console.log("error ", e);
  }
};


return (
  <Fragment>
    <div style={{display: 'flex', justifyContent: 'center', padding: '2rem'}}>
      <Search
        placeholder="영화를 검색해 보세요!"
        onSearch={(value) => console.log(value)}
        onChange={handleQuery}
        onClick={handleButton}
        style={{ width: 200 }}
      />
    </div>
    <div>
      <Row>
        {jsonData.items.map((item) => {
          return (
            <Col xs={24} sm={12} md={6} lg={4} xl={4}>
              <MovieCard item={item}></MovieCard>;
            </Col>
          );
        })}
      </Row>
    </div>
  </Fragment>
);
```

Restart

실행 결과

Detail

[Movie](#) [Home](#) [Search](#) [login](#)



존 윅 4

2023/04/12 · 액션, 스릴러, 범죄 · 2h50m

개요

죽을 위기에서 살아난 존 윅은 최고 회의를 쓰러트릴 방법을 찾아낸다. 비로소 완전한 자유의 희망을 보지만, 빌런 그라몽 후작과 전 세계의 최강 연합은 존 윅의 오랜 친구까지 적으로 만들어 버리고, 새로운 위기에 놓인 존 윅은 최후의 반격을 준비하는데...

리뷰 남기기!

평점 : ☆ ☆ ☆ ☆ ☆

리뷰 작성

실행 결과

DB

```
_id: 19  
comment: "재밌어요!!"  
rating: "8"  
userId: "1020james"
```

```
_id: 20  
comment: "재미없어요"  
rating: "4"  
userId: "1020james"
```

최후의 반격을 준비하는데...

리뷰 남기기!

평점 : ☆ ☆ ☆ ☆ ☆

리뷰 작성

1020james

★ ★ ★ ★ ☆

재밌어요!!

1020james2

★ ★ ☆ ☆ ☆

재미없어요 ㅜㅜ

실행 결과

Sign Up

```
_id: ObjectId('646f670e2f96a3fd4f089c8c')  
id: "1020james"  
pw: "t43u9f4j3ff903jt43t4rhdsgfds94qnyq9aks93ajj83rea84u8385483u4543m5g943f..."  
salt: "w8ngy40mq7379638a9wb78830g5321f4m32m89md3421d29mf853g68waf57c32e3f5435..."
```

[Movie](#) [Home](#) [Search](#) [Login](#) [Sign up](#)

실행 결과

Login

발행 해놓은 고유키를 이용한 네이버 로그인 url 전달

```
/* GET home page. */
router.get('/getNaverOuth', async function(req,res){
  let id = req.query.id;

  res.setHeader('Access-Control-Allow-Origin', '*');
  res.setHeader('Access-Control-Allow-Credentials', 'true'); //쿠키 주고 받기 허용
  try{
    let naverOuth= await Axios.get(
      ('https://nid.naver.com/oauth2.0/authorize?response_type=code&client_id=' + process.env.REACT_APP_NAVER_CLIENT_ID + '&redirect_uri=' +
        encodeURIComponent(process.env.REACT_APP_NAVER_CALLBACK_URI) + '&state=' + Math.random().toString(36).substr(3, 14))
    );
    return res.json(naverOuth);
  } catch (e) {
    return res.json({
      status : 400,
      message : e
    });
  }
}
```

Movie Home Search Login Sign up

로그인



TALK



실행 결과

bcrypt 라이브러리 활용

```
const createSalt = async () => {  
  const buf = await randomBytesPromise(64);  
  
  return buf.toString("base64");  
};
```

검증에 필요한 Salt값 생성

```
export const createHashedPassword = async (password) => {  
  const salt = await createSalt();  
  const key = await pbkdf2Promise(password, salt, 104906, 64, "sha512");  
  const hashedPassword = key.toString("base64");  
  
  return { hashedPassword, salt };  
};
```

해싱할 비밀번호, Salt, 횟수, 길이, 알고리즘 을 인자로 가지는
모듈을 통해 암호화된 비밀번호 생성

```
export const verifyPassword = async (password, userSalt, userPassword) => {  
  const key = await pbkdf2Promise(password, userSalt, 99999, 64, "sha512");  
  const hashedPassword = key.toString("base64");  
  
  if (hashedPassword === userPassword) return true;  
  return false;  
};  
module.exports = app;
```

회원이가입시 발행한 password와 salt값을 통해 비밀번호 검증

개선 사항

1. 프론트 전환 애니메이션 필요
2. 관리자 에게만 작성 평점을 관리할 수 있는 component 가 보이는 기능 필요.
3. naver 뿐만 아닌 kakao google 소셜 로그인 지원
4. 작성 글 등을 관리 할 수 있는 마이페이지 구현
5. 영화 정보 api를 omdb가 아닌 tmdb로 변경

향후 계획

6월 - 1학기 목표에 미흡했던 개선사항들 개선

7월,8월 - data set 확보 및 모델 학습

9월~ - 완성된 모델을 Api화 하여 사용 및 영화추천 기능 개발



감사합니다
