

# 웹 / 앱 게시판 프로젝트

기간

2017.08 ~ 2017.10

이유란

 **GitHub** <https://github.com/leeyoulan/project>

# 목차

## 1. 웹 게시판

- 1) 화면구성 ( 회원가입 )
- 2) 화면구성 ( 게시글 상세 )
- 3) 화면구성 ( 파일업로드 / 파일다운로드 )
- 4) 경로 정의
- 5) DB 구성 ( 웹 )

## 2. 앱 게시판

- 1) 화면구성 ( 로그인 )
- 2) 화면구성 ( 게시글 목록 )
- 3) API 정의 , Status Code 정의
- 4) BODY ( IN\_JSON / OUT\_JSON ) 정의
- 5) DB 구성 ( 앱 )

# 웹 게시판 (개인 프로젝트)

- 개발툴 : Node.js (Express 프레임워크) ( v. 6. 11. 2 ) / NPM ( v. 3. 10. 10 )  
/ MySQL ( v. 5. 6. 36 )  
/ HTML(ejs 템플릿 엔진), CSS / Github, SourceTree

- 프로젝트 설명

: 회원가입과 로그인 후 게시글과 댓글의 등록, 수정, 삭제가 가능합니다.

passport 모듈을 사용하여 회원가입, 로그인, 로그아웃 기능을 구현하였고,

bcrypt-nodejs 모듈을 사용하여 비밀번호 암호화를 하였으며,

Github, SourceTree를 통해 프로젝트 관리를 하였습니다.

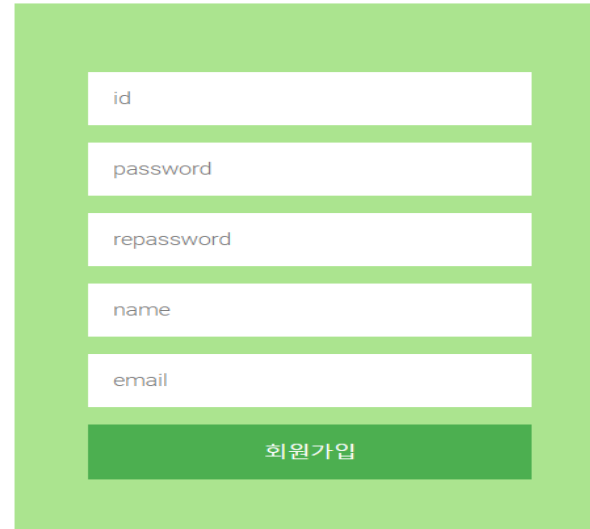
지속적인 업데이트를 통해 대댓글 기능과 채팅기능을 추가할 예정입니다.

## 화면구성 ( 회원가입 )

### 웹 화면

#### 회원가입

회원가입 화면



A registration form with a light green background. It contains six white input fields stacked vertically, labeled 'id', 'password', 'repassword', 'name', and 'email'. Below the fields is a green button with the text '회원가입'.

#### 회원가입

아이디 또는 비밀번호를 입력하지 않은 경우



The registration form with a red error message at the top: '아이디와 비밀번호를 입력해주세요'. Below the message are the same six input fields and the green '회원가입' button.

#### 회원가입

비밀번호와 비밀번호 재입력이 일치하지 않은 경우



The registration form with a red error message at the top: '비밀번호가 일치하지 않습니다.'. Below the message are the same six input fields and the green '회원가입' button.

#### 회원가입

이름을 입력하지 않은 경우



The registration form with a red error message at the top: '이름을 입력해주세요.'. Below the message are the same six input fields and the green '회원가입' button.

#### 회원가입

이메일을 입력하지 않은 경우



The registration form with a red error message at the top: '이메일을 입력해주세요.'. Below the message are the same six input fields and the green '회원가입' button.

## 화면구성 ( 회원가입 )

```
passport.use('local-join',new LocalStrategy({
  usernameField: 'id',
  passwordField: 'password',
  passReqToCallback : true
}, function(req,id,password,done){
  var repassword = req.body.repassword;
  var name = req.body.name;
  var email = req.body.email;

  var query = connection.query('select * from member where id=?',[id],function(err,rows){
    if(err) return done(err);

    if(repassword==''){
      return done(null,false, {message : '비밀번호를 재입력해주세요.'})
    }

    if(name==''){
      return done(null,false, {message : '이름을 입력해주세요.'})
    }

    if(email==''){
      return done(null,false, {message : '이메일을 입력해주세요.'})
    }

    if(password!==repassword){
      return done(null,false, {message : '비밀번호가 일치하지 않습니다.'})
    }

    if(rows.length){
      console.log('existed member');
      return done(null,false, {message : '사용중인 아이디입니다.'})//message:flash 사용
    } else {
      bcrypt.hash(password, null, null, function(err, hash) {
        var sql = {'id':id,'password':hash, 'name':name,'email':email};
        var query = connection.query('insert into member set ?',sql, function(err, rows) {
          if(err) throw err;
          return done(null, {'id': id});
        });
      });
    }
  });
});
```

회원이 가입 성공시 로그인 화면으로 이동,  
실패시 회원가입 화면으로 이동

```
router.post('/',passport.authenticate('local-join',{
  successRedirect : '/login',
  failureRedirect : '/join',
  failureFlash : true
})))
```

회원이 가입 실패시 에러 메시지와 함께  
회원가입 화면으로 이동

```
router.get('/',function(req,res){
  var msg;
  var errMsg = req.flash('error');
  if(errMsg) msg = errMsg;
  if(errMsg=='Missing credentials'){
    msg='아이디와 비밀번호를 입력해주세요';
  }
  res.render('join.ejs',{message: msg});
});
```

비밀번호 재입력, 이름, 이메일, 비밀번호 일치를 확인하여 이상이 있는 경우  
회원가입 화면으로 돌아가 에러 메시지를 같이 출력함

쿼리문에서 아이디 확인 후  
동일한 아이디가 존재하는 경우 회원가입 화면으로 돌아가 에러 메시지 출력,  
동일한 아이디가 존재하지 않는 경우 비밀번호 암호화하여 입력정보를 저장.

웹 화면

글상세

글수정 글삭제 리스트로 돌아가기

테스트 4 테스트 4 테스트 4 테스트 4

작성자 dldbiks 작성일 2017/9/29 조회수 107

테스트 4 테스트 4 테스트 4 테스트 4

댓글[3]

이유란 2017/9/30  
테스트 1

수정 삭제

이유란 2017/9/30  
테스트 2

수정 삭제

이유란 2017/9/30  
이유란

수정 삭제

등록

글쓰기 로그아웃

번호	제목	작성자	작성일	조회수	댓글
1	테스트 1	유란	2017/9/26	30	3
2	테스트 2 테스트 2	유란	2017/9/27	26	1
4번 게시글 클릭했을 경우		dldbiks	2017/9/28	36	2
4	테스트 4 테스트 4 테스트 4 테스트 4	dldbiks	2017/9/29	78	3
5	테스트 5 테스트 5 테스트 5	이유란	2017/9/30	22	1
6	게시판 테스트	유란	2017/10/7	21	0

## 화면구성 ( 게시글 상세 )

```
router.get('/:post_num', function(req,res){
  var post_num= req.params.post_num;

  var sql = 'update board set post_hit = post_hit+1 where post_num = ?';
  connection.query(sql, post_num, function(err,result){
    if (err) {
      res.status(500).send('Something broke!');
      console.log(err.code);
    }
    else {
      var sql = 'select post_num, post_id, post_title, post_content, date_format(post_time,"%Y/%c/%e") post_time, post_hit, originalFileName,savedFileName\
        ,comm_num, comm_id, comm_content, date_format(comm_time,"%Y/%c/%e") comm_time, (select count(comm_num) from comment where post_num=?) comm_count\
        from board join comment using(post_num) where post_num=?';
      connection.query(sql, [post_num, post_num], function(err,rows){
        if (err) {
          res.status(500).send('Something broke!');
          console.log(err.code);
        }else if(rows[0]){
          res.render('boards/board_view.ejs',{rows:rows,'message':' '});
        }else if(rows[0]==undefined){
          var sql = 'select post_num, post_id, post_title, post_content, date_format(post_time,"%Y/%c/%e") post_time, post_hit, originalFileName, savedFileName\
            from board where post_num=?';
          connection.query(sql, post_num, function(err,rows){
            if (err) {
              res.status(500).send('Something broke!');
              console.log(err.code);
            }else{
              res.render('boards/board_view.ejs',{rows:rows,'message':' '});
            }
          });
        }
      });
    }
  });
});
```

← 게시글 클릭시 가장 먼저 조회수 1 증가시킴

← 게시글에 댓글이 존재하는지 쿼리문으로 확인 후 rows[0]이 true 일 경우 게시글 정보와 댓글 정보를 받아서 게시글 상세 화면으로 이동

← rows[0]이 undefined 일 경우 쿼리문으로 게시글 정보만 받아서 게시글 상세 화면으로 이동

## 화면구성 ( 파일업로드/ 파일다운로드 )

### 웹 화면

글쓰기

파일업로드

글쓰기

리스트로 돌아가기

제목	<input type="text"/>
내용	<div></div>

파일 업로드 버튼 클릭했을 경우

글쓰기

글쓰기

리스트로 돌아가기



### 글상세

글수정

글삭제

리스트로 돌아가기

#### 파일업로드 테스트

작성자 이유란33 작성일 2017/10/13 조회수 6

첨부파일 클릭시 파일 다운로드

[첨부파일] [이유란\\_게시판\\_포트폴리오.pdf](#)

파일업로드 테스트 파일업로드 테스트

댓글[1]

이유란33 2017/10/13

파일업로드 테스트입니다.

수정 삭제

이동



# 화면구성 ( 파일업로드/ 파일다운로드 )

## board\_write.ejs

```
<script language="javascript">
```

```
function popupOpen(){
```

```
  var popUrl = "upload";
```

```
  var popOption = "width=370, height=360, resizable=no, scrollbars=no, status=no";
```

```
  window.open(popUrl, "", popOption);
```

```
}
```

```
</script>
```

```
</head>
```

```
<body>
```

```
<h1>글쓰기</h1>
```

```
<div class="upbutton"><button onclick="popupOpen()">파일업로드</button></div>
```

```
<form action="/board_write" method="post">
```

```
  <input type="hidden" name="originalFileName" id="originalFileName" value="originalFileName"/>
```

```
  <input type="hidden" name="savedFileName" id="savedFileName" value="savedFileName"/>
```

```
  <input type="hidden" name="fileSize" id="fileSize" value="fileSize"/>
```

```
  <input type="hidden" name="filePath" id="filePath" value="filePath"/>
```

1. 파일업로드 버튼 클릭시 팝업창 열림

5. 팝업창에서 정보를 받아 value에 저장.

## upload.js

```
router.get('/', function(req, res){
```

```
  res.render('upload');
```

```
});
```

```
router.post('/', upload.single('userfile'),function(req, res){
```

```
  if (req.file!=undefined) {
```

```
    var files = req.file;
```

```
    var originalFileName = files.originalname;
```

```
    var savedFileName = files.filename;
```

```
    var fileSize = files.size;
```

```
    var filePath = path.join('C:\\Users\\youlan\\Desktop\\node_project\\web',files.path);
```

```
    var fileDetail2 = {'originalFileName':originalFileName,'savedFileName':savedFileName,'fileSize':fileSize,'filePath':filePath};
```

```
    var fileDetail = JSON.stringify(fileDetail2);
```

```
    res.send(fileDetail);
```

```
  }
```

```
});
```

3. 파일정보를 받아서 수정한 후 다시 보냄

## upload.ejs

```
<head>
```

```
  <meta charset="utf-8">
```

```
  <title>파일업로드</title>
```

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
```

```
<script type="text/javascript" language="javascript">
```

```
$(function(){
```

```
  $("#uploadbutton").click(function(){
```

```
    var formData = new FormData();
```

```
    formData.append("userfile",$("input[name=userfile]")[0].files[0]);
```

```
    $.ajax({
```

```
      url: '/upload',
```

```
      data: formData,
```

```
      processData: false,
```

```
      contentType: false,
```

```
      type: 'POST',
```

```
      error:function(){
```

```
        window.location.reload();
```

```
      },success:function(fileDetail){
```

```
        var filejson = JSON.parse(fileDetail);
```

```
        window.opener.document.getElementById("originalFileName").value = filejson.originalFileName;
```

```
        window.opener.document.getElementById("savedFileName").value = filejson.savedFileName;
```

```
        window.opener.document.getElementById("fileSize").value = filejson.fileSize;
```

```
        window.opener.document.getElementById("filePath").value = filejson.filePath;
```

```
        window.close();
```

```
      }
```

2. 팝업창에서 확인버튼 클릭시 파일 정보를 **post** /upload로 보냄

4. 성공 시 받은 파일 정보를 board\_write.ejs로 보낸 후 팝업창 닫음

# 경로정의

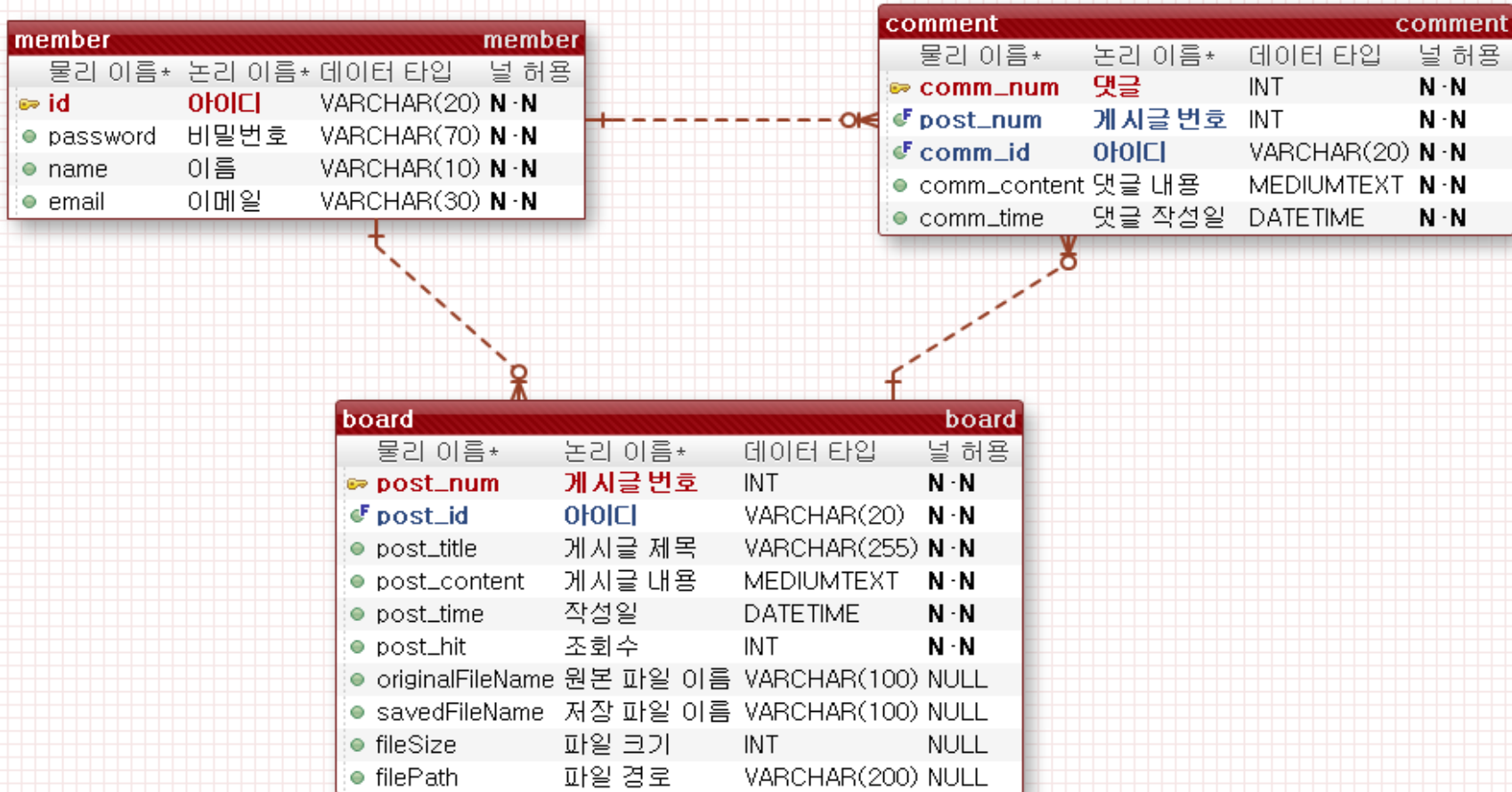
<b>/join</b>	get	/join	회원가입 화면
	post	/join	회원가입
<b>/login</b>	get	/login	로그인 화면
	post	/login	로그인
<b>/logout</b>	get	/logout	로그아웃 화면, 로그아웃
<b>/boards</b>	get	/board_list	게시글 목록
	get	/board_view/:post_num	게시글 상세
	post	/board_view	게시글 상세(댓글쓰기)
	get	/board_write	글 쓰기 화면
	post	/board_write	글 쓰기
	get	/board_update/:post_num	글 수정 화면
	post	/board_update	글 수정
	get	/board_delete/:post_num	글 삭제 화면
<b>/comments</b>	post	/board_delete	글 삭제
	get	/comm_update/:post_num/:comm_num	댓글 수정 화면
	post	/comm_update	댓글 수정
	get	/comm_delete/:post_num/:comm_num	댓글 삭제 화면
<b>/find</b>	post	/comm_delete	댓글 삭제
	get	/find_id	아이디 찾기 화면
	post	/find_id	아이디 찾기
	get	/find_pw	비밀번호 찾기 화면
	post	/find_pw	비밀번호 찾기
	get	/re_pw	비밀번호 재설정 화면
	post	/re_pw	비밀번호 재설정

<b>/upload</b>	get	/upload	업로드 화면
	post	/upload	업로드
<b>/download</b>	get	/download/:savedFileName	다운로드

# DB 구성 ( 웹 )

ERD

eXERD으로 작성



# 앱 게시판 (팀 프로젝트)

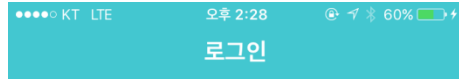
- 개발툴 : Node.js(Express 프레임워크) / MySQL/ Github, SourceTree
- 역할 : 백엔드, DB설계
- 프로젝트 설명

: 회원가입과 로그인 후 게시글과 댓글의 등록, 수정, 삭제가 가능합니다.

express-mysql-session 모듈을 사용하여 로그인, 로그아웃 기능을 구현하였고,  
웹과 동일하게 bcrypt-nodejs 모듈을 사용하여 비밀번호 암호화를 하였으며,  
Github, SourceTree를 통해 프로젝트 관리를 하였습니다.  
또한, Postman을 통하여 API 테스트를 하였습니다.

# 화면구성 ( 로그인 )

## 앱 화면



아이디

비밀번호

- ☐ 아이디 저장
- ☒ 자동 로그인

로그인

[아이디 찾기](#) [비밀번호 찾기](#) [회원가입](#)

## API 테스트 화면

[ id : leeyoulan , password : leeyoulan ] 으로 회원가입 되어있는 경우

Id와 password 정보가 같은 경우  
-> {code : 200} response 함  
= 요청성공

POST http://localhost:55555/members/login

Authorization Headers (1) Body Pre-request Script

form-data x-www-form-urlencoded raw binary

```
1 { "id": "leeyoulan",  
2   "password": "leeyoulan"  
3 }
```

Body Cookies Headers (6) Test Results

Pretty Raw Preview

```
{ "code": "200" }
```

Id 정보가 존재하지 않는 경우  
-> {code : 001} response 함  
= 존재하지 않는 사용자

POST http://localhost:55555/members/login

Authorization Headers (1) Body Pre-request Script

form-data x-www-form-urlencoded raw binary

```
1 { "id": "leeyoulan22",  
2   "password": "leeyoulan"  
3 }
```

Body Cookies Headers (6) Test Results

Pretty Raw Preview

```
{ "code": "001" }
```

Id와 password 정보가 다른 경우  
-> {code : 002} response 함  
= 비밀번호 불일치

POST http://localhost:55555/members/login

Authorization Headers (1) Body Pre-request Script

form-data x-www-form-urlencoded raw binary

```
1 { "id": "leeyoulan",  
2   "password": "youlan"  
3 }
```

Body Cookies Headers (6) Test Results

Pretty Raw Preview

```
{ "code": "002" }
```

## 화면구성 ( 로그인 )

```
router.post('/login',function(req,res){
  var id = req.body.id;
  var password = req.body.password;
  pool.getConnection(function(err,connection){
    if(err) {
      res.json({'code':'500'});
      console.log(err.code);
    }

    var sql = 'select * from member where id=?';

    connection.query(sql,[id],function(err,rows){
      if (err) {
        res.json({'code':'500'});
        console.log(err.code);
        connection.release();
      }

      if(rows.length){
        bcrypt.compare(password, rows[0].password, function(err, result) {
          if (result) {
            req.session.id = id;
            res.json({'code':'200'});
            console.log('200');
            connection.release();
          } else {
            res.json({'code':'002'});
            console.log('002');
            connection.release();
          }
        })
      } else {
        res.json({'code':'001'});
        console.log('001');
        connection.release();
      }
    })
  })
}
```

회원가입시 암호화 되어 저장된 비밀번호와  
사용자가 입력한 비밀번호 비교

result 값이 true이면 id를 session에 저장 후  
{code : 200} response 함  
false이면 {code : 002} response 함

## 회원가입 ( 비밀번호 암호화 부분 )

```
bcrypt.hash(password, null, null, function(err, hash) {
  var post = {'id':id,'password':hash,'name':name,'email':email};
  var sql = 'insert into member set ?';
```

# 화면구성 ( 게시글 목록 )

## 앱 화면

게시글 목록		
1	게시글 테스트입니다.	이유란
	조회: 12   댓글: 0	2017.09.03
2	게시글 123!@#	조성표
	조회: 6   댓글: 3	2017.09.05
3	게시글 목록 테스트	이유란
	조회: 6   댓글: 5	2017.09.09
4	좌측 상단은 메뉴 우측 상단은 글쓰기	이유란
	조회: 10   댓글: 0	2017.09.12
5	게시글 테스트입니다.	이유란
	조회: 12   댓글: 4	2017.09.16

## API 테스트 화면

```
GET http://localhost:55555/posts

Body Cookies Headers (6) Tests

Pretty Raw Preview JSON

1 - [
2 - {
3 -   "post_num": 1,
4 -   "post_id": "이유란",
5 -   "post_title": "게시글 테스트입니다.",
6 -   "post_time": "2017/9/3",
7 -   "post_hit": 12,
8 -   "comm_count": 0
9 - },
10 - {
11 -   "post_num": 2,
12 -   "post_id": "조성표",
13 -   "post_title": "게시글 123!@#",
14 -   "post_time": "2017/9/5",
15 -   "post_hit": 6,
16 -   "comm_count": 3
17 - },
18 - {
19 -   "post_num": 3,
20 -   "post_id": "이유란",
21 -   "post_title": "게시글 목록 테스트",
22 -   "post_time": "2017/9/9",
23 -   "post_hit": 6,
24 -   "comm_count": 0
25 - },
26 - {
27 -   "post_num": 4,
28 -   "post_id": "이유란",
29 -   "post_title": "좌측 상단은 메뉴 우측 상단은 글쓰기",
30 -   "post_time": "2017/10/12",
31 -   "post_hit": 0,
32 -   "comm_count": 0
33 - }
34 - ]
```

get http://localhost:55555/posts로  
접속했을 때 response 되는  
json 형태의 데이터

## 화면구성 ( 게시글 목록 )

```
router.get('/',function(req,res){

pool.getConnection(function(err,connection){
  if (err) {
    res.json({'code':'500'});
    console.log(err.code);
  }

  var sql = 'select post_num, post_id, post_title, post_content, date_format(post_time,"%Y/%c/%e") post_time, post_hit,\
    (select count(comm_num) from comment c where b.post_num=c.post_num) comm_count from board b;';
  connection.query(sql,function(err,result){
    if (err) {
      res.json({'code':'500'});
      console.log(err.code);
      connection.release();
    } else {

      var result_sql = []; ← 쿼리문 결과를 넣어줄 result_sql 배열 생성

      for (var i = 0; i < result.length; i++) {
        result_sql.push({'post_num' : result[i].post_num,
          'post_id' : result[i].post_id,
          'post_title' : result[i].post_title,
          'post_time' : result[i].post_time,
          'post_hit' : result[i].post_hit,
          'comm_count' : result[i].comm_count
        }); ← sql 쿼리문을 통해 나온 값을 push를 통해
        result_sql 배열에 넣어줌

      }
      res.json(result_sql); ← 결과값이 담겨있는 result_sql 배열을
      console.log('200');    json 형태로 response 함
      connection.release();
    }
  });
}
```



## API 정의

### //member

member 리소스	/members		
member URL	post	/members	회원가입
		/members/login	로그인
	get	/members/logout	로그아웃
		/members/id	아이디 찾기
		/members/pw	비밀번호 찾기
	put	/members/pw	비밀번호 재설정

### //post

post 리소스	/posts		
post URL	get	/posts	게시글 목록
		/posts/details	게시글 상세
		/posts/comments	댓글 상세
	post	/posts	글 쓰기
		/posts/comments	댓글 쓰기
	put	/posts	글 수정
		/posts/comments	댓글 수정
	delete	/posts	글 삭제
		/posts/comments	댓글 삭제

## Status Code 정의

category	코드	설명
전체	200	요청 성공
	500	서버 오류
로그인	001	존재하지 않는 사용자
	002	비밀번호 불일치
회원가입	003	이미 가입된 아이디
아이디 찾기	004	일치하는 아이디가 없음
비밀번호 찾기	005	일치하는 비밀번호가 없음

## BODY ( IN\_JSON / OUT\_JSON ) 정의

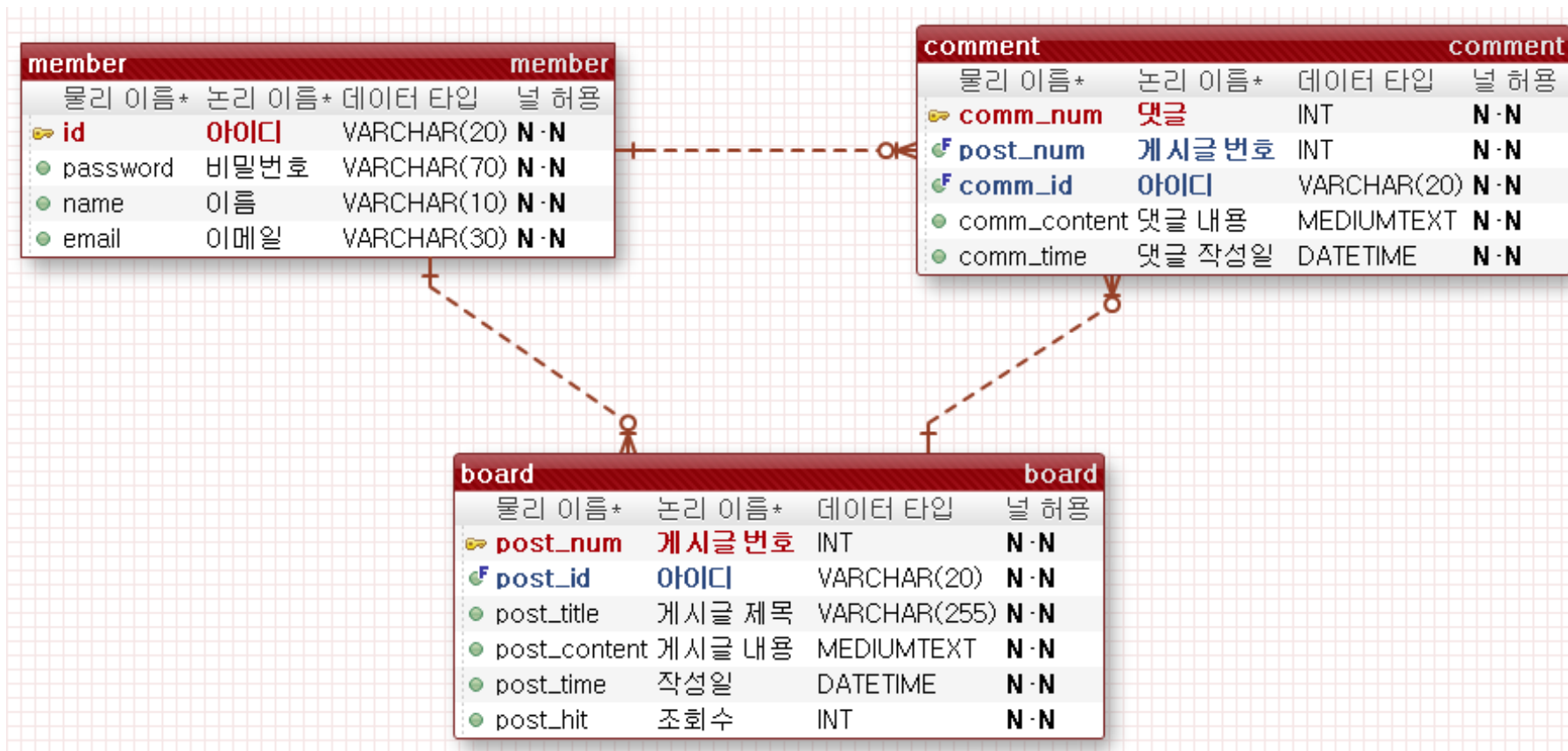
Key	상황	Req/Res	Field명	Field Desc
body	로그인 <b>post</b> /members/login	Req	id	아이디
			password	비밀번호
		Res	Status_code	상태코드 참조
	로그아웃 <b>get</b> /members/logout	Res	Status_code	상태코드 참조
	회원가입 <b>post</b> /members	Req	id	아이디
			password	비밀번호
			name	이름
			email	이메일
		Res	Status_code	상태코드 참조
	아이디 찾기 <b>get</b> /members/id	Req	name	이름
			email	이메일
		Res	id	아이디
		Res	Status_code	상태코드 참조
	비밀번호 찾기 <b>get</b> /members/pw	Req	id	아이디
			name	이름
			email	이메일
		Res	Status_code	상태코드 참조
	비밀번호 재설정 <b>put</b> /members/pw	Req	id	아이디
			password	비밀번호
		Res	Status_code	상태코드 참조
	게시글 목록 <b>get</b> /posts	Res	post_num	게시글 번호
			post_id	아이디
			post_title	게시글 제목
			post_time	작성일
			post_hit	조회수
			comm_count	댓글 수 (=count(comm_num))

body	게시글 상세 <b>get</b> /posts/details	Req Res	post_num	게시글 번호
			post_id	(게시글)아이디
			post_title	게시글 제목
			post_content	게시글 내용
			post_time	작성일
			post_hit	조회수
	댓글 상세 <b>get</b> /posts/comments	Req Res	post_num	게시글 번호
			comm_num	댓글 번호
			comm_count	댓글 수
			comm_id	(댓글)아이디
			comm_content	댓글 내용
			comm_time	댓글 작성일
	글 쓰기 <b>post</b> /posts	Req Res	post id	아이디
			post title	게시글 제목
			post content	게시글 내용
			post time	작성일
			Status_code	상태코드 참조
	(답)댓글 쓰기 <b>post</b> /posts/comments	Req Res	post_num	게시글 번호
			comm_id	(댓글)아이디
			comm_content	댓글 내용
			comm_time	댓글 작성일
			Status_code	상태코드 참조
	글 수정 <b>put</b> /posts	Req Res	post_num	게시글 번호
			post title	게시글 제목
			post_content	게시글 내용
			Status_code	상태코드 참조
	(답)댓글 수정 <b>put</b> /posts/comments	Req Res	post_num	게시글 번호
			comm_num	댓글 번호
			comm_content	댓글 내용
			Status_code	상태코드 참조
	글 삭제 <b>post</b> /posts	Req Res	post_num	게시글 번호
			Status_code	상태코드 참조
	(답)댓글 삭제 <b>delete</b> /posts/comments	Req Res	post_num	게시글 번호
			comm_num	댓글 번호
			Status_code	상태코드 참조

# DB 구성 ( 앱 )

ERD

eXERD으로 작성



**감사합니다.**