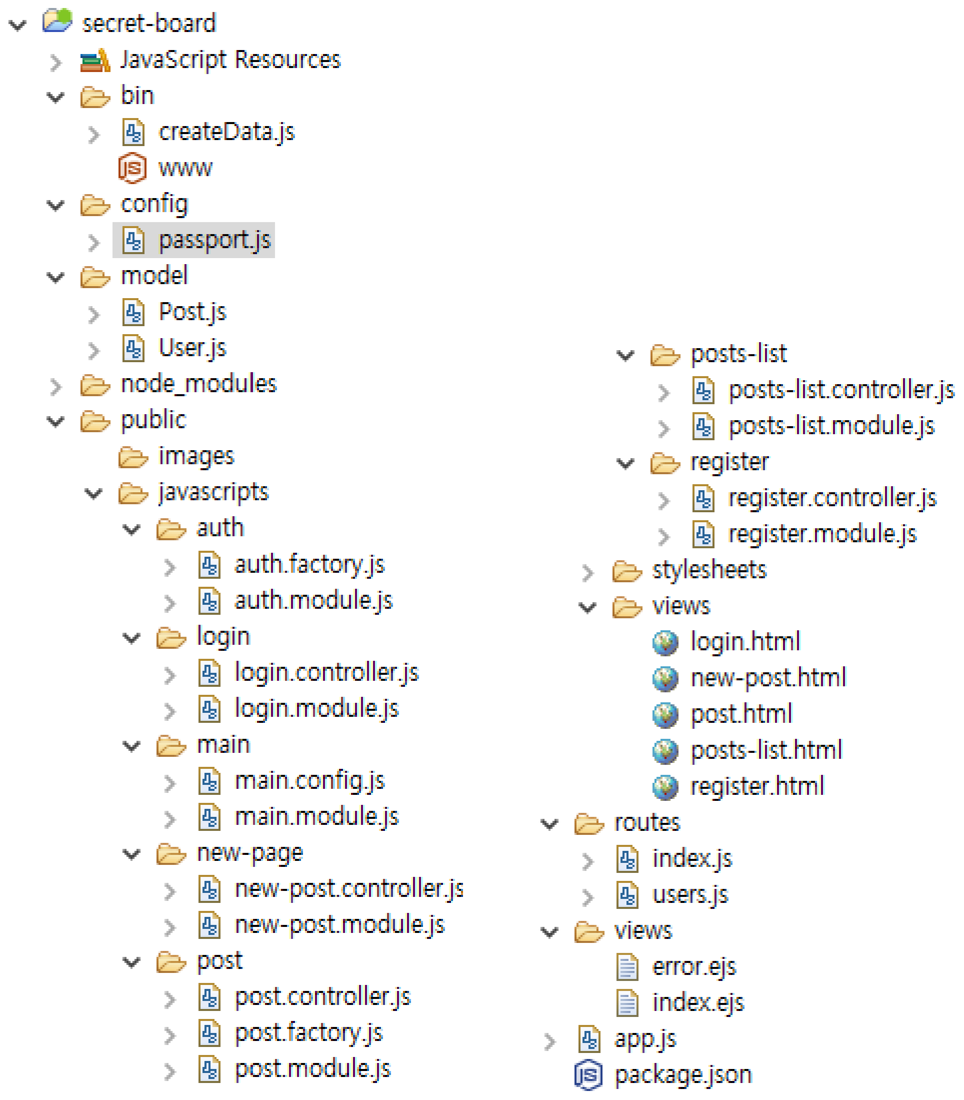
- MEAN stack을 이용한 웹서버 및 개인용 게시판

1. 기능

- 회원가입을 통해 ID를 생성하고, 개인이 마음대로 사용할 수 있는 게시판을 제공

2. 서버 디렉토리 구조

- 기본틀로 express-generator를 이용하여 디렉토리 구조를 구성함

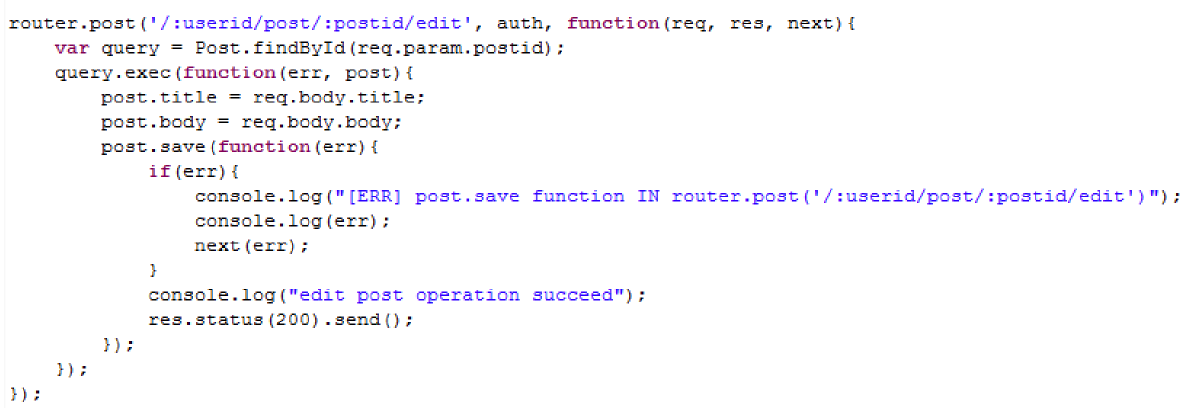


```
secret-board
  JavaScript Resources
  bin
    createData.js
    www
  config
    passport.js
  model
    Post.js
    User.js
  node_modules
  public
    images
    javascripts
      auth
        auth.factory.js
        auth.module.js
      login
        login.controller.js
        login.module.js
      main
        main.config.js
        main.module.js
      new-page
        new-post.controller.js
        new-post.module.js
      post
        post.controller.js
        post.factory.js
        post.module.js
      posts-list
        posts-list.controller.js
        posts-list.module.js
      register
        register.controller.js
        register.module.js
    stylesheets
    views
      login.html
      new-post.html
      post.html
      posts-list.html
      register.html
  routes
    index.js
    users.js
  views
    error.ejs
    index.ejs
  app.js
  package.json
```

- 확장성과 관리의 편의성을 위해서 기능별로 모듈을 나누고, angularjs의 $stateprovider 서비스를 이용함

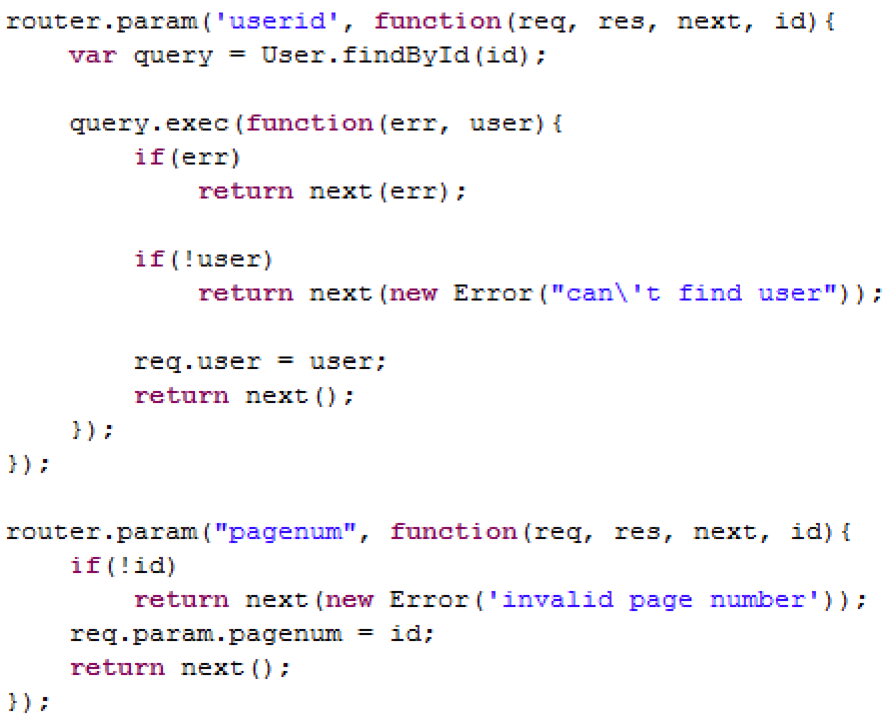
3. 핵심 구현 요소

- 사용자의 요청을 처리하는 서버측 javascript 코드(express의 Router 이용)



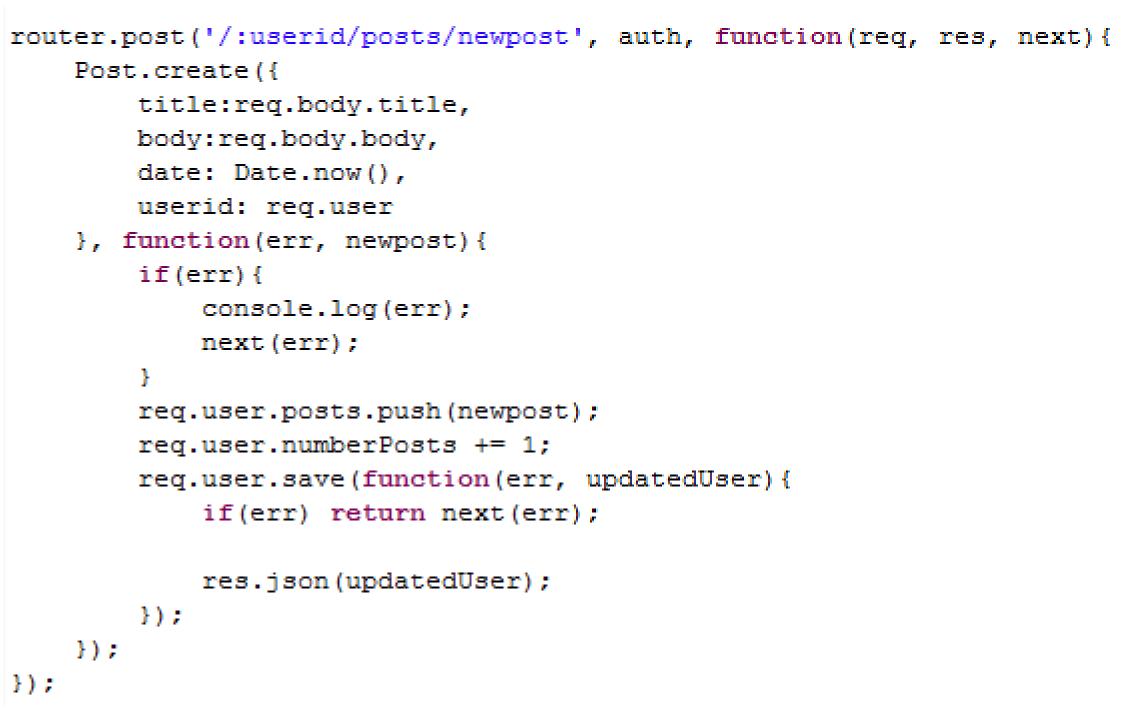
```
router.post('/:userid/post/:postid/edit', auth, function(req, res, next){
    var query = Post.findById(req.param.postid);
    query.exec(function(err, post){
        post.title = req.body.title;
        post.body = req.body.body;
        post.save(function(err){
            if(err){
                console.log("[ERR] post.save function IN router.post('/:userid/post/:postid/edit')");
                console.log(err);
                next(err);
            }
            console.log("edit post operation succeed");
            res.status(200).send();
        });
    });
});
```

<여러 method와 url에 관한 핸들러 중 하나>



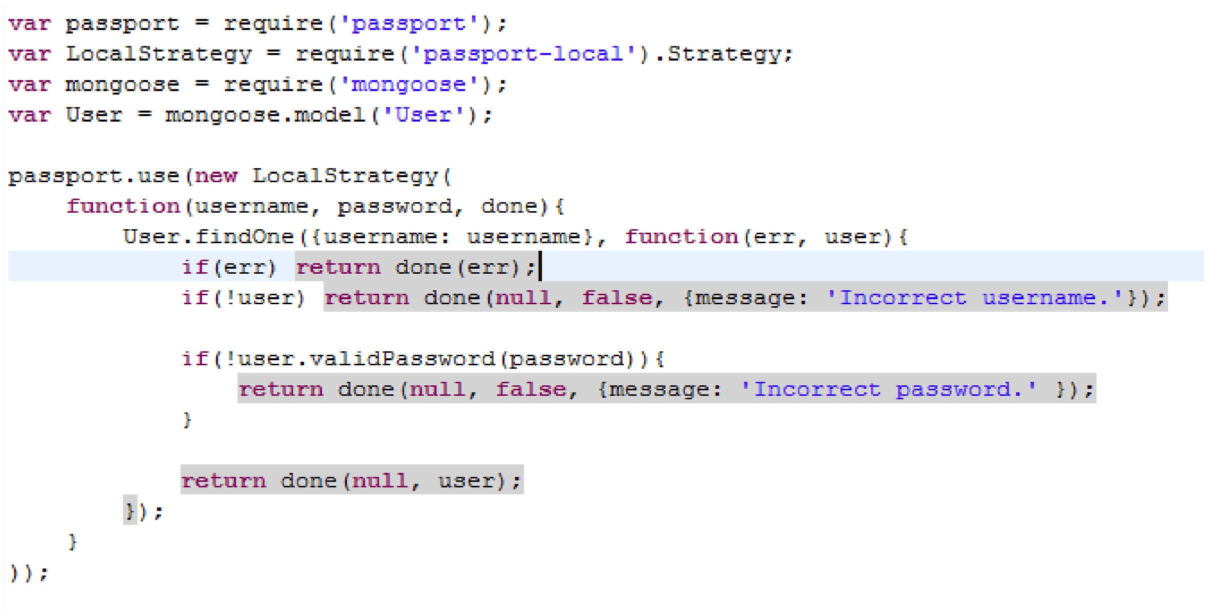
```
router.param('userid', function(req, res, next, id){
    var query = User.findById(id);

    query.exec(function(err, user){
        if(err)
            return next(err);

        if(!user)
            return next(new Error("can\'t find user"));

        req.user = user;
        return next();
    });
});

router.param("pagenum", function(req, res, next, id){
    if(!id)
        return next(new Error('invalid page number'));
    req.param.pagenum = id;
    return next();
});
```

<url에 있는 parameter 추출하는 param 함수들 중 일부 -
이후 request에 대한 데이터 처리 후 response를 보내는데 이용>

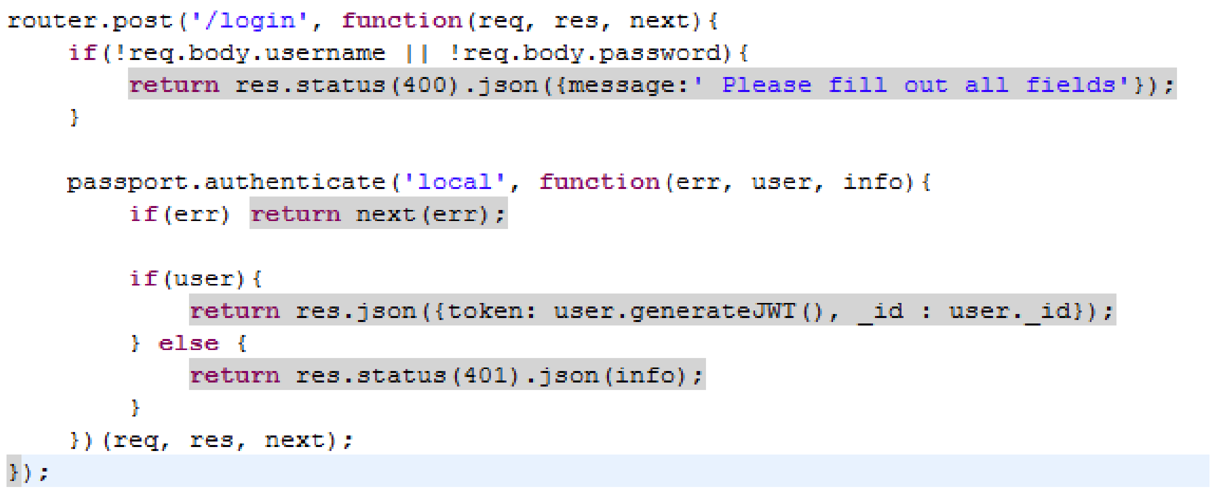


```
router.post('/:userid/posts/newpost', auth, function(req, res, next){
    Post.create({
        title:req.body.title,
        body:req.body.body,
        date: Date.now(),
        userid: req.user
    }, function(err, newpost){
        if(err){
            console.log(err);
            next(err);
        }
        req.user.posts.push(newpost);
        req.user.numberPosts += 1;
        req.user.save(function(err, updatedUser){
            if(err) return next(err);

            res.json(updatedUser);
        });
    });
});
```

<mongoose를 이용한 MongoDB 데이터 수정 또는 삽입>



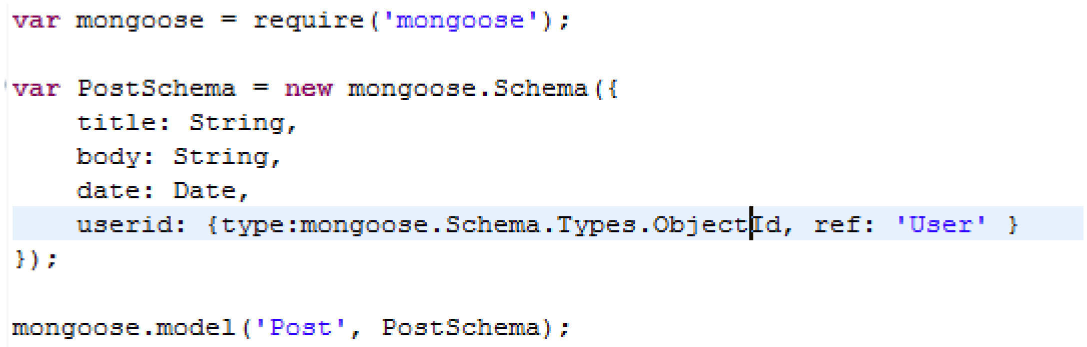
```
var passport = require('passport');
var LocalStrategy = require('passport-local').Strategy;
var mongoose = require('mongoose');
var User = mongoose.model('User');

passport.use(new LocalStrategy(
    function(username, password, done){
        User.findOne({username: username}, function(err, user){
            if(err) return done(err);
            if(!user) return done(null, false, {message: 'Incorrect username.'});

            if(!user.validPassword(password)){
                return done(null, false, {message: 'Incorrect password.' });
            }

            return done(null, user);
        });
    }
));
```



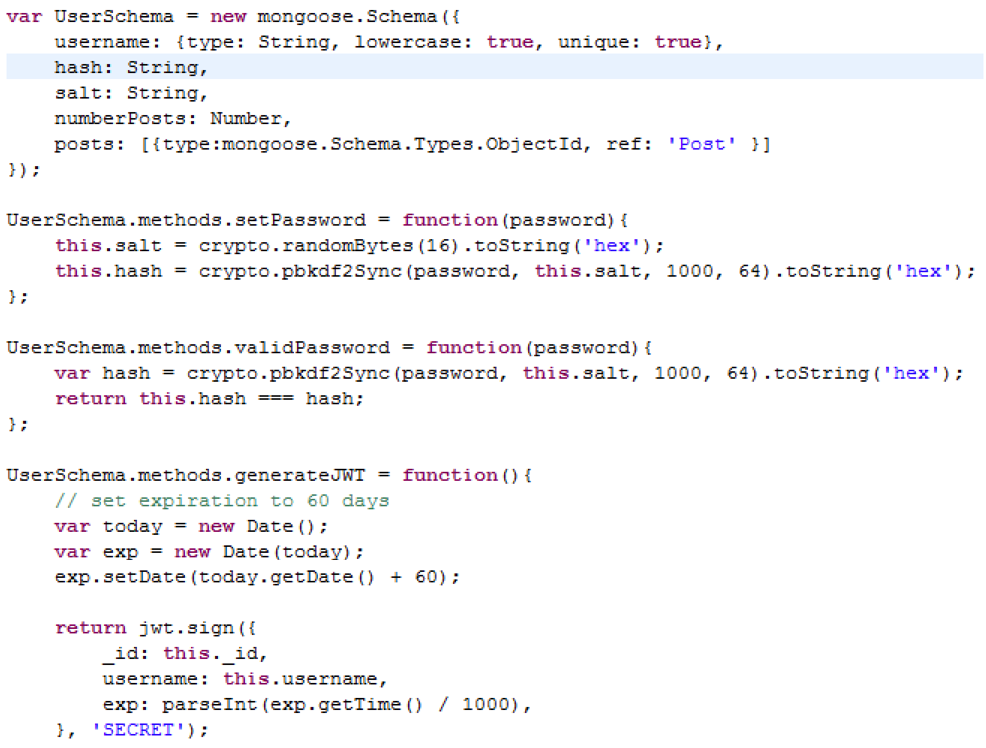
```
router.post('/login', function(req, res, next){
    if(!req.body.username || !req.body.password){
        return res.status(400).json({message:' Please fill out all fields'});
    }

    passport.authenticate('local', function(err, user, info){
        if(err) return next(err);

        if(user){
            return res.json({token: user.generateJWT(), _id : user._id});
        } else {
            return res.status(401).json(info);
        }
    })(req, res, next);
});
```

<passport의 local-strategy를 이용한 사용자 인증>

- Mongoose Schema 정의

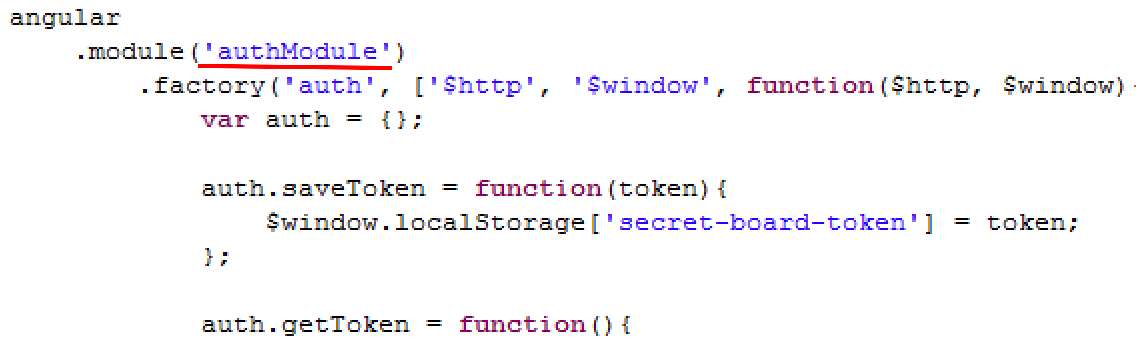


```
var mongoose = require('mongoose');

var PostSchema = new mongoose.Schema({
    title: String,
    body: String,
    date: Date,
    userid: {type:mongoose.Schema.Types.ObjectId, ref: 'User' }
});

mongoose.model('Post', PostSchema);
```



```
var UserSchema = new mongoose.Schema({
    username: {type: String, lowercase: true, unique: true},
    hash: String,
    salt: String,
    numberPosts: Number,
    posts: [{type:mongoose.Schema.Types.ObjectId, ref: 'Post' }]
});

UserSchema.methods.setPassword = function(password){
    this.salt = crypto.randomBytes(16).toString('hex');
    this.hash = crypto.pbkdf2Sync(password, this.salt, 1000, 64).toString('hex');
};

UserSchema.methods.validPassword = function(password){
    var hash = crypto.pbkdf2Sync(password, this.salt, 1000, 64).toString('hex');
    return this.hash === hash;
};

UserSchema.methods.generateJWT = function(){
    // set expiration to 60 days
    var today = new Date();
    var exp = new Date(today);
    exp.setDate(today.getDate() + 60);

    return jwt.sign({
        _id: this._id,
        username: this.username,
        exp: parseInt(exp.getTime() / 1000),
    }, 'SECRET');
```

- AngularJS에서 자주 사용하는 기능은 서비스로 등록하여, 필요한 모듈마다 의존성 주입을 통해 서비스 이용

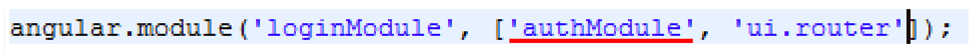


```
angular
    .module('authModule')
        .factory('auth', ['$http', '$window', function($http, $window)
            var auth = {};

            auth.saveToken = function(token){
                $window.localStorage['secret-board-token'] = token;
            };

            auth.getToken = function(){
```

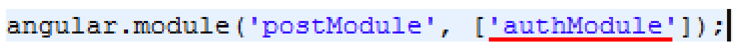
.

.

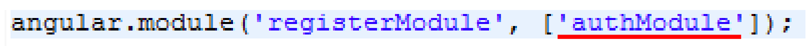
.



```
angular.module('loginModule', ['authModule', 'ui.router']);
```



```
angular.module('postModule', ['authModule']);
```

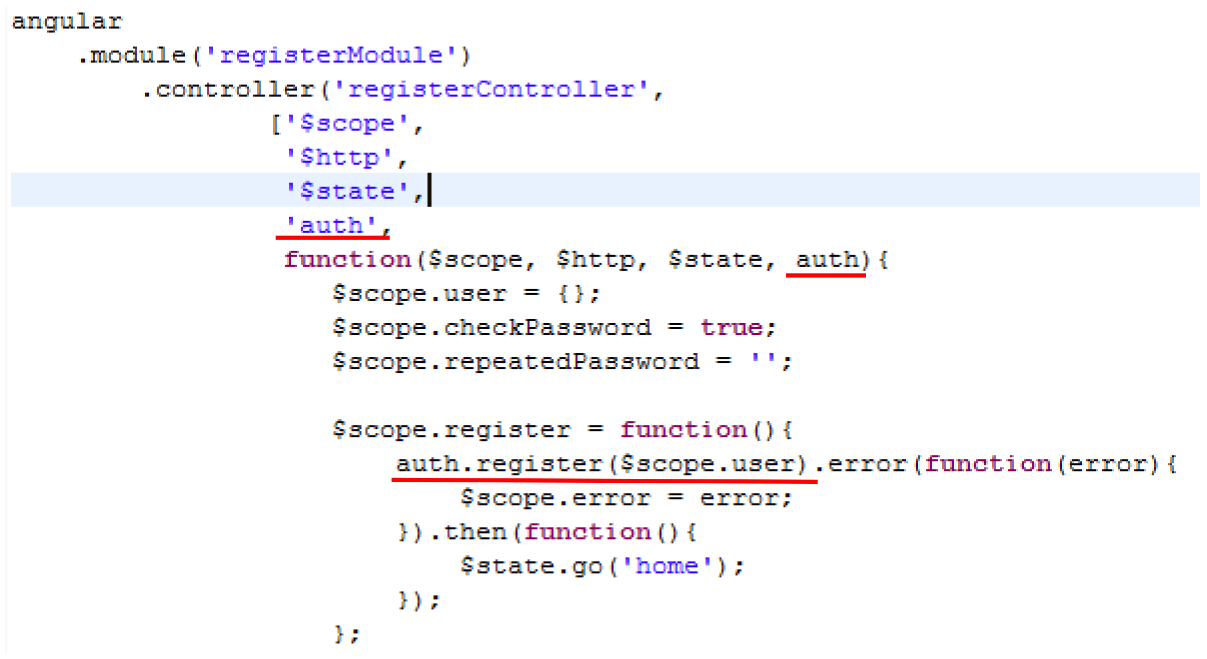


```
angular.module('registerModule', ['authModule']);
```

.

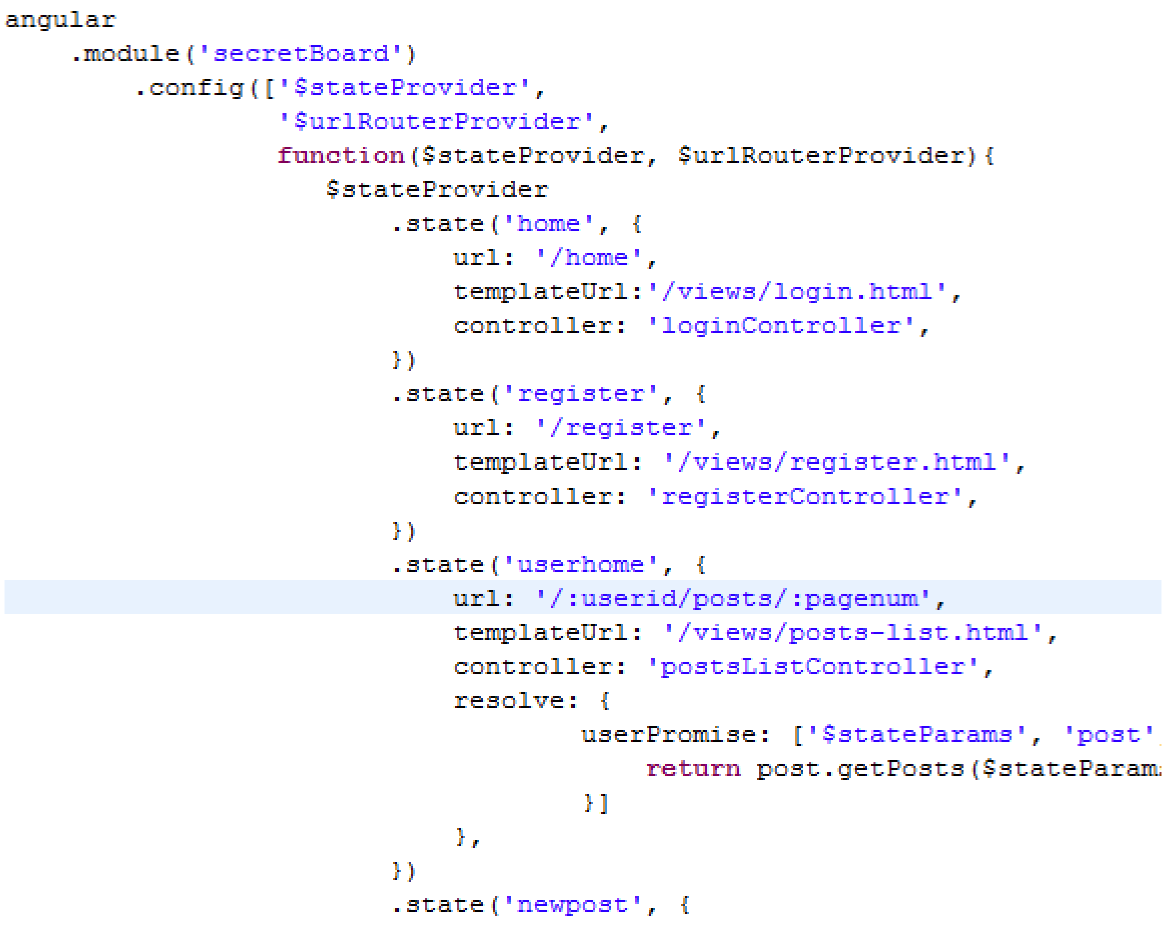
.

.



```
angular
    .module('registerModule')
        .controller('registerController',
               ['$scope',
                '$http',
                '$state',
                'auth',
                function($scope, $http, $state, auth){
                   $scope.user = {};
                   $scope.checkPassword = true;
                   $scope.repeatedPassword = '';

                   $scope.register = function(){
                       auth.register($scope.user).error(function(error){
                           $scope.error = error;
                       }).then(function(){
                           $scope.go('home');
                       });
                   };
```

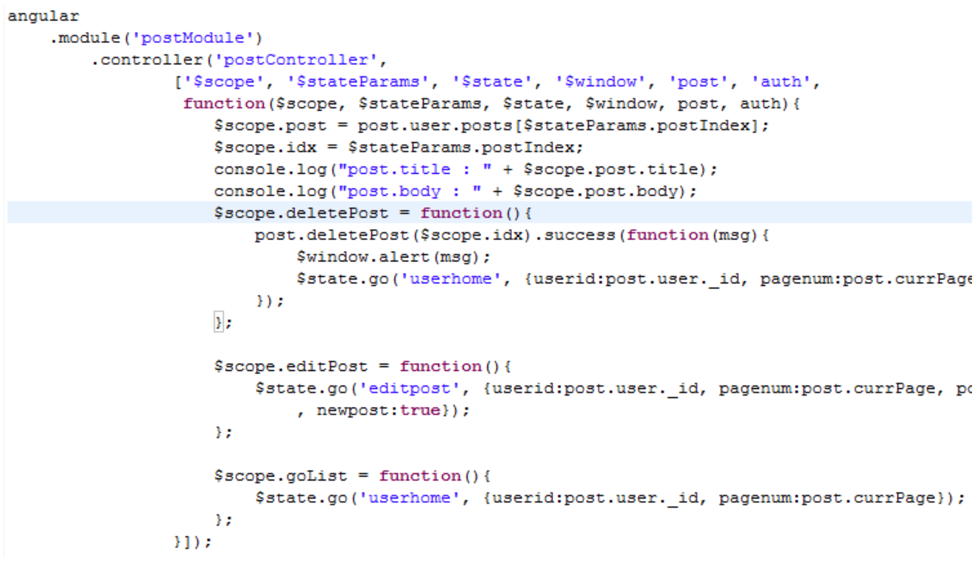
- 각 url에 관한 template을 한 곳에서 관리($stateprovider 서비스 이용)



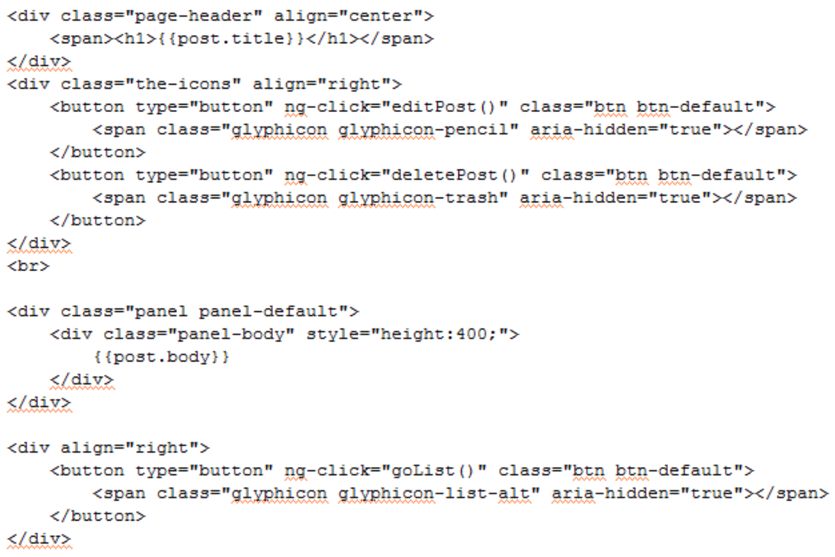
```
angular
    .module('secretBoard')
        .config(['$stateProvider',
                 '$urlRouterProvider',
                 function($stateProvider, $urlRouterProvider){
                    $stateProvider
                        .state('home', {
                            url: '/home',
                            templateUrl:'/views/login.html',
                            controller: 'loginController',
                        })
                        .state('register', {
                            url: '/register',
                            templateUrl: '/views/register.html',
                            controller: 'registerController',
                        })
                        .state('userhome', {
                            url: '/:userid/posts/:pagenum',
                            templateUrl: '/views/posts-list.html',
                            controller: 'postsListController',
                            resolve: {
                                    userPromise: ['$stateParams', 'post'
                                        return post.getPosts($stateParam
                                    }]
                            },
                        })
                        .state('newpost', {
```

<이하 생략>

-> 각 템플릿은 $stateprovider 서비스에 의해 정의된 controller에 연결됨

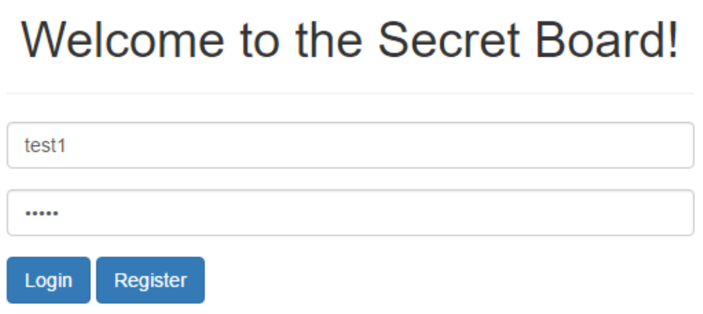


```
angular
    .module('postModule')
        .controller('postController',
                ['$scope', '$stateParams', '$state', '$window', 'post', 'auth',
                 function($scope, $stateParams, $state, $window, post, auth){
                    $scope.post = post.user.posts[$stateParams.postIndex];
                    $scope.idx = $stateParams.postIndex;
                    console.log("post.title : " + $scope.post.title);
                    console.log("post.body : " + $scope.post.body);
                    $scope.deletePost = function(){
                        post.deletePost($scope.idx).success(function(msg){
                            $window.alert(msg);
                            $state.go('userhome', {userid:post.user._id, pagenum:post.currPag
                        });
                    };

                    $scope.editPost = function(){
                        $state.go('editpost', {userid:post.user._id, pagenum:post.currPage, pc
                            , newpost:true});
                    };

                    $scope.goList = function(){
                        $state.go('userhome', {userid:post.user._id, pagenum:post.currPage});
                    };
                }]);
```



```
<div class="page-header" align="center">
    <span><h1>{{post.title}}</h1></span>
</div>
<div class="the-icons" align="right">
    <button type="button" ng-click="editPost()" class="btn btn-default">
        <span class="glyphicon glyphicon-pencil" aria-hidden="true"></span>
    </button>
    <button type="button" ng-click="deletePost()" class="btn btn-default">
        <span class="glyphicon glyphicon-trash" aria-hidden="true"></span>
    </button>
</div>
<br>

<div class="panel panel-default">
    <div class="panel-body" style="height:400;">
        {{post.body}}
    </div>
</div>

<div align="right">
    <button type="button" ng-click="goList()" class="btn btn-default">
        <span class="glyphicon glyphicon-list-alt" aria-hidden="true"></span>
    </button>
</div>
```

4. 홈페이지 실제 모습

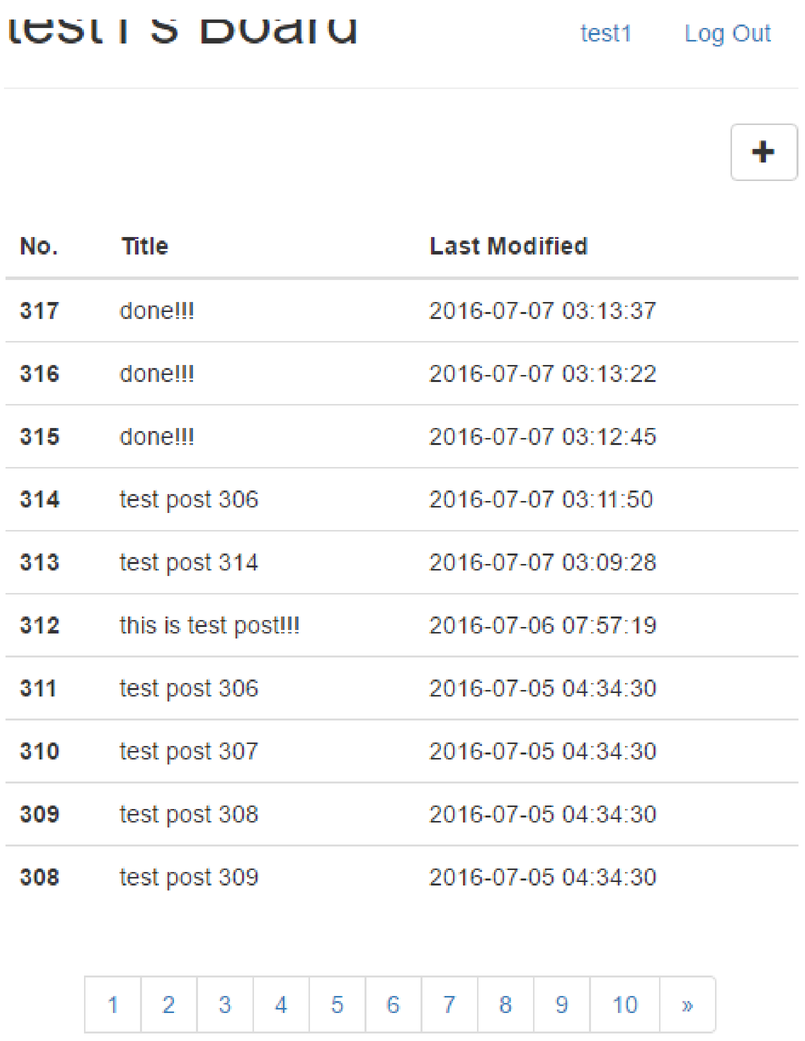


Welcome to the Secret Board!

test1

•••••

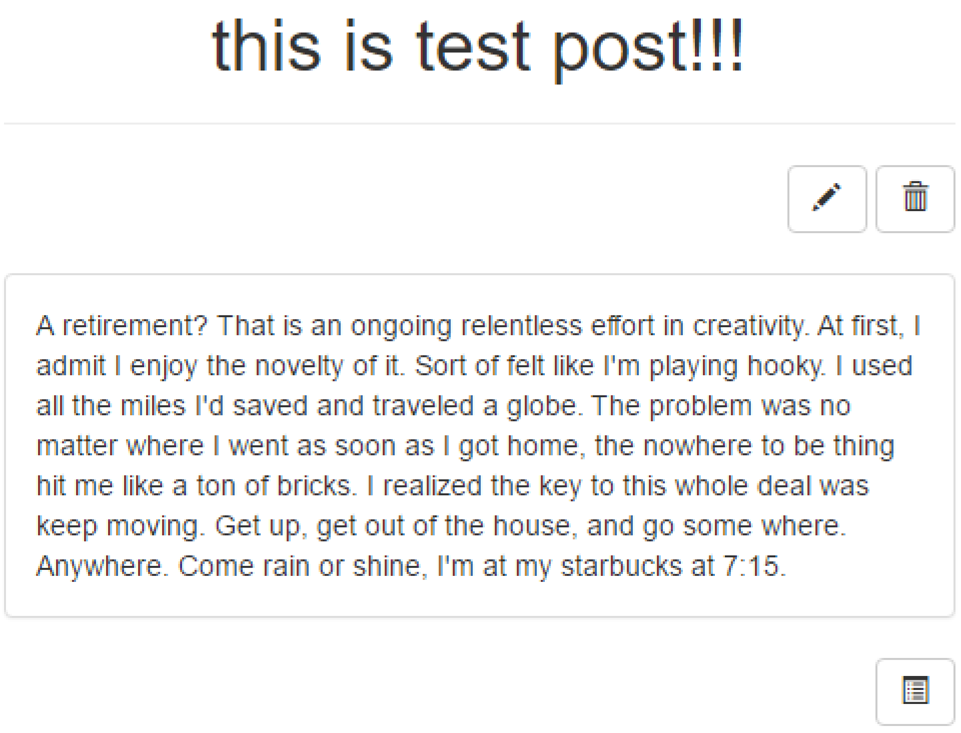
<login 화면>



test1 's Board

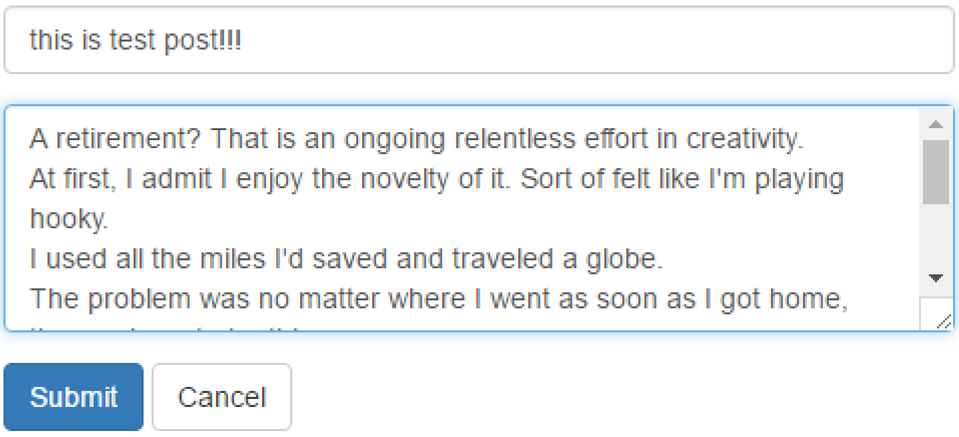
No.	Title	Last Modified
317	done!!!	2016-07-07 03:13:37
316	done!!!	2016-07-07 03:13:22
315	done!!!	2016-07-07 03:12:45
314	test post 306	2016-07-07 03:11:50
313	test post 314	2016-07-07 03:09:28
312	this is test post!!!	2016-07-06 07:57:19
311	test post 306	2016-07-05 04:34:30
310	test post 307	2016-07-05 04:34:30
309	test post 308	2016-07-05 04:34:30
308	test post 309	2016-07-05 04:34:30

<게시글의 리스트>


this is test post!!!

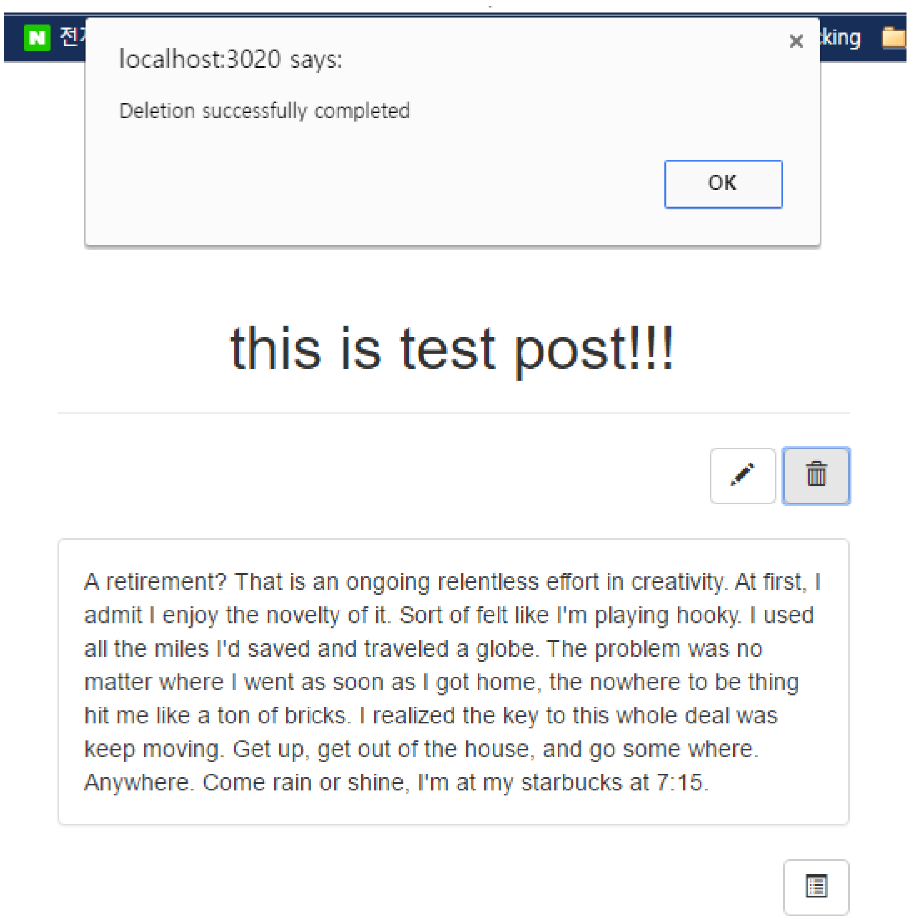
A retirement? That is an ongoing relentless effort in creativity. At first, I admit I enjoy the novelty of it. Sort of felt like I'm playing hooky. I used all the miles I'd saved and traveled a globe. The problem was no matter where I went as soon as I got home, the nowhere to be thing hit me like a ton of bricks. I realized the key to this whole deal was keep moving. Get up, get out of the house, and go some where. Anywhere. Come rain or shine, I'm at my starbucks at 7:15.

<포스트 상세 내용>


this is test post!!!

A retirement? That is an ongoing relentless effort in creativity.
At first, I admit I enjoy the novelty of it. Sort of felt like I'm playing hooky.
I used all the miles I'd saved and traveled a globe.
The problem was no matter where I went as soon as I got home,

<글 수정 화면>



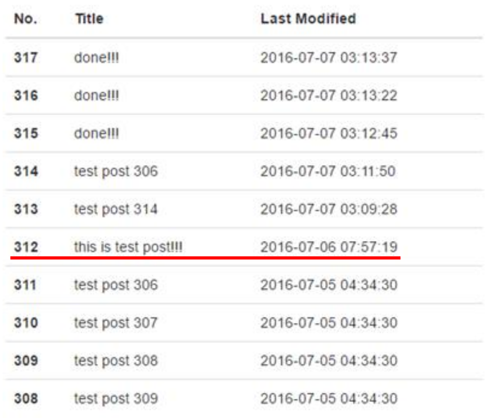
localhost:3020 says:

Deletion successfully completed

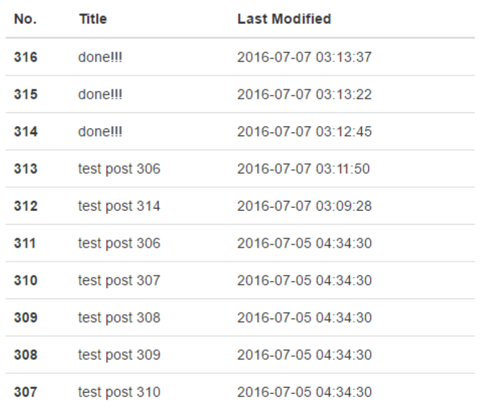
this is test post!!!

A retirement? That is an ongoing relentless effort in creativity. At first, I admit I enjoy the novelty of it. Sort of felt like I'm playing hooky. I used all the miles I'd saved and traveled a globe. The problem was no matter where I went as soon as I got home, the nowhere to be thing hit me like a ton of bricks. I realized the key to this whole deal was keep moving. Get up, get out of the house, and go some where. Anywhere. Come rain or shine, I'm at my starbucks at 7:15.

<삭제 성공 화면>



No.	Title	Last Modified
317	done!!!	2016-07-07 03:13:37
316	done!!!	2016-07-07 03:13:22
315	done!!!	2016-07-07 03:12:45
314	test post 306	2016-07-07 03:11:50
313	test post 314	2016-07-07 03:09:28
312	this is test post!!!	2016-07-06 07:57:19
311	test post 306	2016-07-05 04:34:30
310	test post 307	2016-07-05 04:34:30
309	test post 308	2016-07-05 04:34:30
308	test post 309	2016-07-05 04:34:30



No.	Title	Last Modified
316	done!!!	2016-07-07 03:13:37
315	done!!!	2016-07-07 03:13:22
314	done!!!	2016-07-07 03:12:45
313	test post 306	2016-07-07 03:11:50
312	test post 314	2016-07-07 03:09:28
311	test post 306	2016-07-05 04:34:30
310	test post 307	2016-07-05 04:34:30
309	test post 308	2016-07-05 04:34:30
308	test post 309	2016-07-05 04:34:30
307	test post 310	2016-07-05 04:34:30

* 전제 소스 코드 : https://github.com/shharn/mysources/tree/master/Secret-Board/src