박성호(3417)의 결과 보고서

ver. 2015.7 학습동아리 발표대회

무엇을 했는가

node.js 공부

마피아 게임 구현

나는 이것을 했다

1. node.js 공부

1. 왜 node.js 인가

PHP를 대체하는 웹 언어가 등장할 예정이고 Java를 이용해서 웹 서버를 구축하는 것은 상당한 난이도가 요구된다고 한다. 웹 서버를 구축하기에 적당한 언어를 물색하던 중 node.js를 알게 되었다. 소규모 서버에서 사용하면 좋은 성능을 보인다고 하고, 쿠키런, 아마존, Paypal 등 이름 있는 곳들에서 node.js를 차용했다는 소식을 듣고 node.js를 사용하기로 결정했다.

2. 어떻게 공부했는가

처음에는 책을 통해 공부했다. 하지만 빠르게 변화하는 node.js의 특성 상 책은 변화하는 node.js를 공부하는데 낡은 정보만을 제공해주었다. 그래서 대부분의 지식을 node.js 공식 웹 사이트와 Stack Overflow, GitHub 등 해외 커뮤니티 그리고 각 모듈들의 공식 웹 사이트를 통해 공부했다.

★ io.js

node.js 진영은 초기에는 활발하게 사용자들과 커뮤니티를 형성하고 개발자들의 수요에 맞춰서 변화하는 등의 좋은 모습을 보여줬다. 하지만 node.js의 스폰서가 교체되고 개발 팀의 리더가 바뀌면서 폐쇄적이고 사용자들의 수요가 반영되지 않고 변하는 고사하고 1년이 지나도 0.12 버전에서 1.0 버전이 나오지 않는 등 좋지 않고 비민주적인 모습들을 보여줬다. 이에 대응하여 node.js의 고문팀 격인 Node Forum에서 io.js라는 node.js를 fork한 프로젝트를 시작했다. (fork: 오픈 소스, 자유 소프트웨어 프로그램의 코드를 그대로 가져와 사용하는 것).

io.js는 지속적으로 업데이트가 이루어지는 자바스크립트 엔진인 V8의 변화와 자바스크립트 자체의 변화에 빠르게 대응하고 사용자들의 수요를 즉각적으로 반영하는 등 기존 node.js가 가진 문제점을 해결했다. 또한 node.js 진영의 최대 장점인, 기하급수적으로 많은 모듈들을 그대로 io.js에서 사용할 수 있도록 하여서 기존 node.js 개발자들은 부담 없이 io.js로 넘어 올 수 있었다. 현재 2015년 7월 기준으로 2.3.3 버전이 나왔을 만큼 업데이트도 꾸준히 이루어지고 있다.

io.js 진영의 목적은 node.js와의 경쟁이 아닌 공생이라고 말한 만큼 node.js와 많은 호환성을 가지지만 앞으로 어떻게 될지는 모르는 일이다. 하지만 최근 io.js 개발 팀이 node.js 개발 팀과 합치는 등 서로 하나가 되고자 하는 경향을 보이고 있어서 io.js 프로젝트의 창설 목적대로 '공생'이 이루어지는 듯 하다.

io.js Binary를 다운 받아서 실행할 때 iojs로도 실행이 되지만 node로도 실행이 된다. 그래서 본 보고서에서도 굳이 iojs가 아닌 node로 io.js를 표기하겠다.

3. 이런 것을 공부했다.

- 모듈(서버, 클라이언트 코드에서 공부한 내용이 표출되므로 공부한 내용은 설명하지 않음)
  - ✓ fs
  - ✓ os
  - ✓ process
  - ✓ url

- ✓ querystring
- ✓ socket.io

- Event Driven Programming Model의 개념(활동 일지에서 자세히 설명함)

2. 마피아 게임 구현

1. 왜 이것을 만드는가

친구들과 이야기하는 도중 우연찮게 마피아 게임 이야기가 나왔고 마침 node.js로 채팅 서버를 구현했던 나는 node.js로 마피아 게임을 만들 수 있겠구나 하는 생각을 했다. 채팅 서버를 기반으로 마피아 게임 서버를 만들기로 했다.

2. 어떻게 만들었고 만들것인가

- 개발 환경: node.js(후에 io.js로 변경)
- 게임에 대한 계획을 세운 후 계획에 맞춰서 개발하기로 함.

3. 마피아 게임 계획

- 마피아, 경찰, 시민, 의사가 기본적인 직업이다. (후에 많은 직업들을 추가할 예정)
- 회원가입 통해 사용자 정보를 저장하는 DB 를 구현한다. 이를 통해 사용자를 구분한다.
- 게임을 하기 전 사용자들이 머무는 로비를 구현한다.
- 게임이 이루어지는 공간인 '방'을 구현한다.
- 게임은 '방'에서 채팅으로 진행되며 게임이 끝나는 조건은 다음과 같다.
  - ✓ 마피아가 모두 죽는다. → 시민 승리
  - ✓ 시민이 모두 죽는다. → 마피아 승리
  - ✓ 두 명이 남았는데 이들은 마피아와 마피아가 아닌 사람이다. → 마피아 승리
- 그많데 SNS 인 Silver Fish 와 회원 정보를 연동한다. Silver Fish 가입자는 마피아 게임에 로그인이 가능하며 자신의 정보를 그대로 마피아 게임 서버가 사용할 수 있게 한다.
- 게임은 '도시'라는 공간에서 진행된다. '도시'는 그래픽을 통해 구현되며 사용자들은 '도시'를 자유롭게 돌아다닐 수 있다. 그래픽은 게임의 모든 요소가 개발된 후에 추가하기로 했다.
- '뉴스'가 사용자들에게 보이도록 한다. '뉴스'는 '도시'에서 이루어지는 사건, 사소한 일들을 보도해준다. 보도되는 내용은 쓸모 있는 내용과 쓸모 없는 내용으로 구분한다. 사용자들은 뉴스의 쓸모 있는 내용들에서 유용한 정보를 추리할 수 있게 하며 이를 통해 마피아의 정체와 같은 게임의 요소들을 추리할 수 있게 한다.

4. 마피아 게임 서버 코드 작성

서버의 코드는 다음과 같이 구성했다. 코드의 주석은 "//", "/* … */"의 형태이고 본 보고서에서 달은 주석은 "#"를 앞에 붙인다. 함수는 중요한 부분만 집어 넣었다.

```
var express = require('express');
var path = require('path');
var favicon = require('serve-favicon');
var logger = require('morgan');
var cookieParser = require('cookie-parser');
var bodyParser = require('body-parser');

var routes = require('./routes/index');
var users = require('./routes/users');

var game = require('./game');
var random = require('random-js')();

var app = express();


//socket.io server
var server = require('http').Server(app);
var io = require('socket.io')(server);
```

#서버에 필요한 모듈들을 불러오고 객체화한다.

```
//view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');

app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());

app.use(express.static(path.join(__dirname, 'public')));
app.use('/', routes);
app.use('/robby', routes);
app.use('/madeby', routes);
app.use('/ingame', routes);

app.use('/users', users);

//server on
server.listen(8080, function(){
       console.log('Mafia Server Start !');
});
```

#서버 라우팅 설정 및 가동

```
//newUserCon 이벤트를 통해 자동으로 생성시킨 닉네임을 현재 연결된 클라이언트 소켓에 전달.
var new_nickname = 'GUEST-' + count;
socket.emit('newUserCon', {nickname : new_nickname});

count++;

//닉네임을 자동으로 생성해서 접속한 클라이언트의 소켓 객체에 저장.
socket_ids[new_nickname] = socket.id;

socket.nickname = new_nickname;
surviveList[socket.nickname] = true;
ingameList[socket.nickname] = true;
```

```
live[socket.nickname] = true;

console.log(new_nickname + ' is connected');

io.sockets.emit('refreshList', Object.keys(socket_ids));
```

#사용자가 서버에 접속할 때 이루어지는 일들 정의.

일어나는 일들

1. 닉네임 생성
2. 소켓 아이디 저장
3. 소켓 닉네임 저장
4. 게임 제어에 필요한 변수들에 적당한 값을 저장
5. 클라이언트에게 사용자 재갱신 이벤트 발송

```
socket.on('changeName', function(data){
	//전에 쓰던 닉네임은 삭제
	if(socket.nickname != undefined){
		delete socket_ids[socket.nickname];
		delete ingameList[socket.nickname];
		delete surviveList[socket.nickname];
		delete live[socket.nickname];
	}
	if(socket.nickname == manager)
		manager = data.nickname;

	if(data.nickname == 'admin')
		manager = 'admin';

	/**
	* 닉네임을 현재 접속한 클라이언트의 소켓 객체에 저장.
	* 소켓 아이디-닉네임 배열에 닉네임을 key로 해서 현재 소켓 아이디 저장.
	* userlist 이벤트를 통해 현재 접속한 사용자들의 닉네임을 모든 클라이언트에 전송.
	**/

	socket_ids[data.nickname] = socket.id;
	ingameList[data.nickname] = true;
	surviveList[data.nickname] = true;
	live[data.nickname] = true;

	if(statusList[socket.nickname] != undefined){
		switch(statusList[socket.nickname]){
			case 1:
				delete citizenId[socket.nickname];
				citizenId[data.nickname] = socket.id;
				break;
			case 2:
				delete mafiaId[socket.nickname];
				mafiaId[data.nickname] = socket.id;
				break;
			case 3:
				delete policeId[socket.nickname];
				policeId[data.nickname] = socket.id;
				break;
			case 4:
				delete doctor[socket.nickname];
				doctorId[data.nickname] = socket.id;
				break;
		}
	}
```

```
        socket.nickname = data.nickname;
        io.sockets.emit('refreshList', Object.keys(socket_ids));
});
```

사용자가 닉네임을 바꿨을 때 서버에서 기존 변수들의 값을 새로운 닉네임에 맞게 변경한다. 변경 후에는 클라이언트에 사용자 재갱신 이벤트 발송.

```
//메시지 전달
socket.on('send_msg', function(data){
        var toSendNickname = data.toSend;
        //data.msg = toSendNickname + ' : ' + data.msg;

        //전체에게 메시지 전달
        if(data.toSend == 'ALL')
                io.sockets.emit('get_msg_all', data);

        //특정 사용자에게만 메시지 전달
        else{
                var toSendId = socket_ids[data.toSend];
                if(toSendId != undefined){
                        io.to(toSendId).emit('get_msg', data);
                        socket.emit('get_msg', data);
                }
        }
});
```

서버에서 채팅을 발송한 클라이언트의 메시지를 받고 메시지의 종류에 따라서 클라이언트들에게 메시지 발송.

```
//신분 결정
function make_status(){
        var status = 0;
        var def_id;

        //socket_ids에서 닉네임을 뽑아옴
        for(var def_nickname in socket_ids){
                //모든 신분 중 한 개의 신분이라도 0명이라면
                if(citizen == 0 || mafia == 0 || doctor == 0 || police == 0){
                        while(1){
                                status = random.integer(1, 4);

                                //status에 해당하는 신분의 수가 0명이라면
                                if( (citizen == 0 && status == 1) || (mafia == 0 && status ==
2) || (police == 0 && status == 3) || (doctor == 0 && status == 4) ){
                                        def_id = socket_ids[def_nickname];
                                        io.to(def_id).emit('def_status',  {t_status  :  status,
msg : '당신은 ' + game.status_list[status] + '입니다.'});
                                        regStatus(status, def_id, def_nickname);
                                        break;
                                }
                        }
                }

                //모든 신분이 1명 씩 있을 때 부터는 무작위
                else {
                        while(1) {
                                status = random.integer(1, 4);
                                def_id = socket_ids[def_nickname];

                                if(status   !=   1   &&   (Object.keys(citizenId).length   >=
Object.keys(socket_ids).length / 2) ){
                                        io.to(def_id).emit('def_status',  {t_status  :  status,
msg : '당신은 ' + game.status_list[status] + '입니다.'});
                                        regStatus(status, def_id, def_nickname);
```

```
                                        break;
                                }
                                else if(status == 1){
                                        io.to(def_id).emit('def_status', {t_status : status,
msg : '당신은 ' + game.status_list[status] + '입니다.'});
                                        regStatus(status, def_id, def_nickname);
                                        break;
                                }
                        }
                }
        }

        //game.showAll(citizenId, mafiaId, doctorId, policeId);
}


//투표 시작
socket.on('startVote', function(data){
        if(start == true && time == 1){
                voteStart++;

                io.sockets.emit('sys_msg', {msg : data.from + '님이 투표를 시작을 요청했습니다.',
color : color});
                if (voteStart == Object.keys(socket_ids).length - killed){
                        voteStart = 0;
                        //전체투표
                        voteToKill();
                }
        }
        else if(time == 2)
                io.to(socket_ids[data.from]).emit('sys_msg', {msg : '밤에는 직업별 투표만 가능합니
다.', color : color});
        else if(start == false && (citizen == 0 || mafia == 0))
                io.to(socket_ids[data.from]).emit('sys_msg', {msg : '이미 게임이 끝났습니다.',
color : color});
        else
                io.to(socket_ids[data.from]).emit('sys_msg', {msg : '투표는 게임이 시작한 후에 시작해
야 합니다.', color : color});
});
```

게임이 시작할 때 신분을 결정하는 함수. 정말 머리 깨지는 줄 알았다. 게임에 참가하는 인원은 무조건 4명 이상이고 각 직업별 비율은 적절한 수준을 유지해야 하므로 4명과 4명 이상일 때의 경우를 나눠서 직업을 배분했다. 사람이 4명일 때의 직업 배분이 정말 어려웠다. 알고리즘과 조건식 구성은 간단하나 문자형과 숫자형 변수가 구분되지 않았던 문제점이 있었다. 이 함수의 조건식에서 사용하는 모든 함수를 숫자형으로 바꿔 문제를 해결했다.

```
//투표 결과 처리
socket.on('citizenResult', function(selected){
        treatVote();
});

socket.on('mafiaResult', function(selected){
        if(surviveList[selected] == true && ingameList[selected] == true){
                surviveList[selected] = false;
                killed++;
        }
        treatVote();
});

socket.on('policeResult', function(data){
        var tmp = statusList[data.selected];

        if(tmp == 2)
                io.to(socket_ids[data.nick]).emit('sys_msg', {msg : data.selected + '는 마피아입
```

```
니다.', color : color});
	else
		io.to(socket_ids[data.nick]).emit('sys_msg', {msg : data.selected + '는 마피아가 아닙니다.', color : color});

	treatVote();
});

socket.on('doctorResult', function(selected){
	if(surviveList[selected] == false && ingameList[selected] == true){
		killed--;
		surviveList[selected] = true;
	}
	treatVote();
});
```

낮에 이루어지는 투표 결과를 각 직업별로 서버에서 수신한다. 수신한 결과를 직업별로 처리한 후 투표 결과를 처리하는 함수에 처리한 정보를 넘긴다.

```
//투표 결과 처리
socket.on('treatMVote', function(selected){
	voteCount++;

	if(voteResult[selected] == undefined)
		voteResult[selected] = 1;
	else
		voteResult[selected]++;

	//모두 투표했을 때
	if (voteCount == Object.keys(live).length) {
		var elected = [];
		var most = selected;

		io.sockets.emit('sys_msg', {msg : '투표가 끝났습니다.', color : color});

		//init
		elected[most] = voteResult[selected];

		//가장 많이 뽑힌 사람 탐색
		for(var target in voteResult){
			if (elected[most] < voteResult[target]){
				elected[target] = voteResult[target];
				delete elected[most];
				most = target;
			}
			else if (elected[most] == voteResult[target]) {
				elected[target] = voteResult[target];
			}
		}

		//가장 많이 뽑힌 사람이 1명이 아닐 때
		if (Object.keys(elected).length != 1) {
			_elected = Object.keys(elected);
			var _msg = "";

			for(var i=0; i<_elected.length; i++){
				if (i == _elected.length -1 ) {
					_msg += _elected[i] + '님 을 죽이자고 ' + voteResult[most] + '명이 투표했습니다.';
				}
				else
					_msg += _elected[i] + '님 과';
			}
			io.sockets.emit('sys_msg', {msg : _msg, color : color});
```

```
                        io.sockets.emit('sys_msg', {msg : '재투표를 하겠습니다.', color : color});

                        voteCount = 0;
                        voteToKill();
                        return;
                }

                //사망 처리
                killPlayer(most);

                io.sockets.emit('sys_msg', {msg : '투표로 죽은 사람은 ' + most + '님 입니다.', color :
color});
                voteCount = 0;

                for(i in voteResult)
                        delete voteResult[i];

                //밤이 되고 직업 별 투표
                voteStatus();
        }
});
```

서버에서 처리한 투표 관련 정보를 처리하는 함수를 정의한다. 가장 많이 투표가 된 사람이 죽으며 투표 인원 수가 같을 경우 재투표를 진행한다. 가장 많이 투표가 된 사람을 죽인 후 게임을 밤으로 진행한다.

```
function voteStatus(){
        time = 2;
        io.sockets.emit('sys_msg', {msg : '-----------------------------------------------
-------', color : color});
       io.sockets.emit('sys_msg', {msg : '밤이 되었습니다. 투표창에서 투표를 해주세요.', color : color});

        var t;
        //시민에게
        for(t in citizenId){
                if(ingameList[t] == true)
                        io.to(socket_ids[t]).emit('citizenVote', {list : Object.keys(live)});
        }
        //마피아에게
        for(t in mafiaId){
                if(ingameList[t] == true)
                        io.to(socket_ids[t]).emit('mafiaVote', {list : Object.keys(live)});
        }
        //경찰에게
        for(t in policeId){
                if(ingameList[t] == true)
                        io.to(socket_ids[t]).emit('policeVote', {list : Object.keys(live)});
        }
        //의사에게
        for(t in doctorId){
                if(ingameList[t] == true)
                        io.to(socket_ids[t]).emit('doctorVote', {list : Object.keys(live)});
        }
}
```

밤의 투표를 진행한다. 각 직업별로 투표 이벤트를 발생시킨다.

```
//직업별 투표 처리
function treatVote(){
        console.log('treatVote count' + voteCount);
        voteCount++;

        //모두 투표했을 때
        if (voteCount == Object.keys(live).length) {
```

```
                var killCount = 0;
                var targetSc;

                time = 1;
                io.sockets.emit('sys_msg', {msg : '----------------------------------------
---------------', color : color});
                io.sockets.emit('sys_msg', {msg : '낮이 되었습니다.', color : color});

                //사망자 처리
                for(var target in socket_ids){
                        //죽은 놈이면 게임에서 제외
                        if (surviveList[target] == false) {
                                killPlayer(target);
                                io.sockets.emit('sys_msg', {msg : target + '님이 죽었습니다.',
color : color});
                                killCount++;
                        }
                }

                if (killCount == 0)
                        io.sockets.emit('sys_msg', {msg : '아무도 죽지 않았습니다.', color : color});
                else
                        io.sockets.emit('sys_msg', {msg : '죽은 사람은 ' + killed + '명 입니다.',
color : color});

                io.sockets.emit('sys_msg', {msg : '남은 사람은 ' + Object.keys(live) + ' 입니다.',
color : color});

                voteCount = 0;

                switch(whoWin()){
                        case 0:
                                break;
                        case 1:
                                io.sockets.emit('sys_msg', {msg : '마피아가 모두 죽어 시민이 승리했습니
다!', color : color});
                                start = false;
                                showAll(1);
                                io.sockets.emit('refreshList', {user : Object.keys(socket_ids),
status : Object.values(statusList)} );
                                break;
                        case 2:
                                io.sockets.emit('sys_msg', {msg : '시민이 모두 죽어 마피아가 승리했습니
다!', color : color});
                                start = false;
                                showAll(2);
                                io.sockets.emit('refreshList', {user : Object.keys(socket_ids),
status : Object.values(statusList)} );
                                break;
                }
        }
}
```

밤의 투표 결과를 직업별 클라이언트로 수신해 결과를 처리하는 함수다. 이 함수에서 게임의 승리 조건을 검사해 게임을 계속 진행할지 아니면 누군가의 승리로 게임을 끝낼지 결정한다.

5. 마피아 클라이언트 코드 작성

```
<meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">

        <script src="/socket.io/socket.io.js"></script>
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
```

```
        <script src="js/bootstrap.min.js"></script>

        <link href="css/bootstrap.min.css" rel="stylesheet">

        <link href="select-box/dist/css/bootstrap-select.css" rel="stylesheet" >
        <script src="select-box/dist/js/bootstrap-select.js"></script>


        <style>
                @import url(http://fonts.googleapis.com/earlyaccess/naunmgothic.css);

                body {
                        font-family: "Helvetica Neue", Helvetica, Arial, 'Nanum Gothic',  sans-
serif;
                }

                table {
                        border-collapse: separate;
                        border-spacing: 10px 5px;
                }
        </style>
```

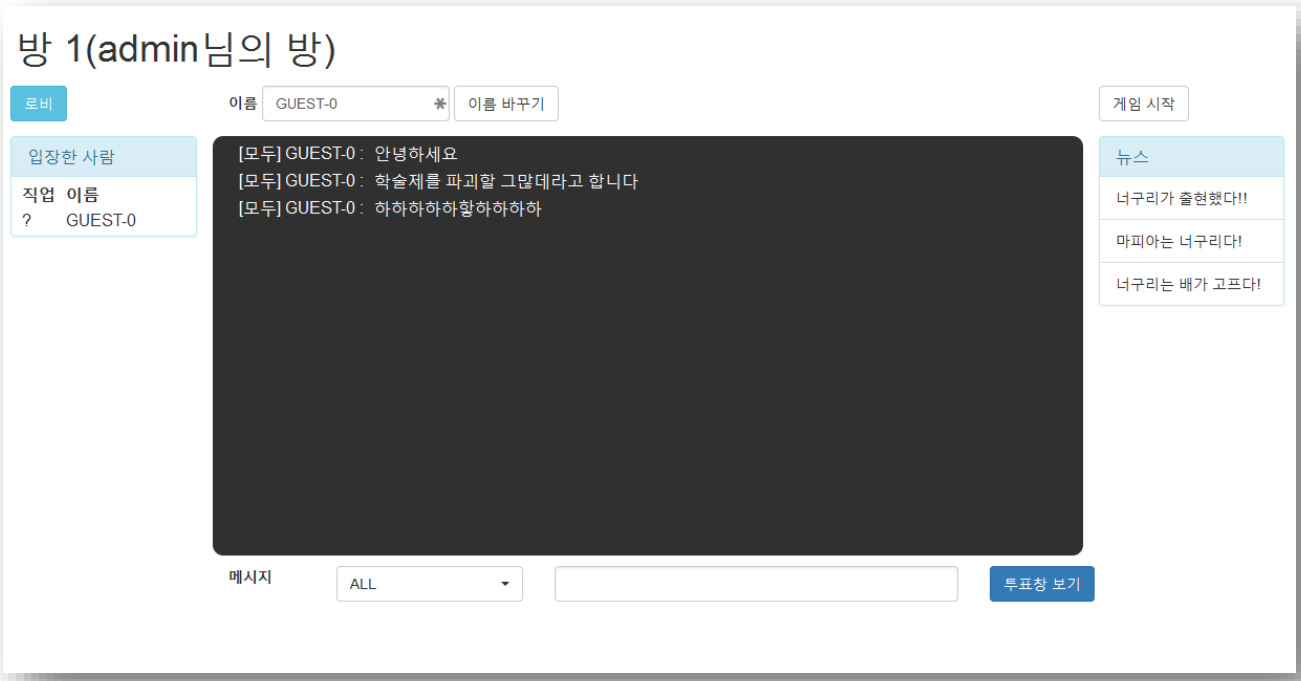
Bootstrap, jQuery, Google Font를 위한 클라이언트의 코드다.

```
<div class="container-fluid">
                <div class="col-md-12" style="border-radius: 10px; padding: 5px;">
                        <h1>방 1(admin님의 방)</h1>
                </div>

                <div class="row">
                        <div class="col-md-2">
                                <input type="button" class="btn btn-info" id="goLobby" value="
로비"/>
                        </div>

                        <div class="col-md-6">
                                <form role="form" class="form-inline">
                                        <label for="name">이름</label>
                                        <div class="form-group">
                                                <input     type="text"     class="form-control"
id="nickname"/>
                                                <input  type="button"  class="btn  btn-default"
id="changeNameBtn" value="이름 바꾸기"/>
                                        </div>
                                </form>
                        </div>


                        <div class="col-md-2 col-md-offset-2">
                                <input type="button" class="btn btn-default" id="startGameBtn"
value="게임 시작"/>
                        </div>
                </div>

                <div class="row" style="padding-bottom:10px;">
                        <div class="col-md-2">
                                <div class="panel panel-info">
                                        <div class="panel-heading">
                                                <h3 class="panel-title">입장한 사람</h3>
                                        </div>

                                        <table id="userInfo">
                                                <thead>
                                                        <tr>
                                                                <th>직업</th>
                                                                <th>이름</th>
```

```
                                        </tr>
                                    </thead>
                                </table>
                            </div>
                        </div>

                        <div    class="col-md-8"    id="div_table"    style="border-radius:10px;
background-color:#303030; height:400px; overflow:auto;">
                            <table             id="msgs"             style="font-size:12pt;
color:#ffffff;"></table>
                        </div>

                        <div class="col-md-2">
                            <div class="panel panel-info">
                                <div class="panel-heading">
                                    <h3 class="panel-title">뉴스</h3>
                                </div>
                                <ul class="list-group">
                                    <li class="list-group-item">너구리가 출현했다!!</li>
                                    <li class="list-group-item">마피아는 너구리다!</li>
                                    <li    class="list-group-item">너구리는    배가    고프
다!</li>
                                </ul>
                            </div>
                        </div>
                </div>

                <div class="row">
                    <label for="basic" class="col-lg-1 col-lg-offset-2 control-label">메시지
</label>

                    <div class="col-lg-2">
                        <select   id="userList"   class="selectpicker   show-tick   form-
control" data-live-search="true">
                            <option value="ALL" data-subtext="모두에게">ALL</option>
                        </select>
                    </div>

                    <div class="col-lg-4">
                        <input type="text" class="form-control" id="msgbox"/>
                    </div>
                    <div class="col-lg-2">
                        <button   class="btn   btn-primary"   data-toggle="modal"   data-
target="#voteModal">
                            투표창 보기
                        </button>
                    </div>
                </div>

                <!-- Modal -->
                <div  class="modal  fade"  id="voteModal"  tabindex="-1"  role="dialog"  aria-
labelledby="myModalLabel" aria-hidden="true">
                  <div class="modal-dialog">
                    <div class="modal-content">
                      <div class="modal-header">
                        <button type="button" class="close" data-dismiss="modal"><span
aria-hidden="true">&times;</span><span class="sr-only">Close</span></button>
                        <h4 class="modal-title" id="voteQst"></h4>
                      </div>
                      <div id="voteBox" class="modal-body">

                      </div>
                      <div class="modal-footer">
                        <button    type="button"    class="btn    btn-default"    data-
dismiss="modal">닫기</button>
                        <div id="voteBtn"></div>
                      </div>
```

```
                    </div> <!-- 모달 콘텐츠 -->
                </div> <!-- 모달 다이얼로그 -->
            </div> <!-- 모달 전체 윈도우 -->
```

Bootstrap을 이용해 방을 구현했다. 스크린 샷을 통해 확인하자.




```
<script type="text/javascript">
        var socket = io.connect();
        var myNickname;

        var myStatus = 0;

        var startTime;

        var chatOn = true;
        var chatCount = 0;

        var end = false;

        var status = { 1 : '시민', 2 : '마피아', 3 : '경찰', 4 : '의사'};

        <!--게임 시작-->
        $('#startGameBtn').click(function(){
                socket.emit('ready', {from : myNickname});
        });

        <!--닉네임 변경-->
        $('#changeNameBtn').click(function(){
                if(document.getElementById("nickname").value.length <= 10){
                        socket.emit('changeName', {nickname : $('#nickname').val()});
                        myNickname = $('#nickname').val();
                }
                else
                        alert('닉네임은 10글자 이하의 문자열만이 등록 가능합니다.');
                });

                <!--메시지 전송-->
                $('#msgbox').keyup(function(event){
                        if(event.which == 13){
                                if(chatOn == true){
                                        if(startTime != undefined && ( (new Date()).getTime() -
```

```
startTime) <= 1000){
                                        chatCount++;

                                        if(chatCount >= 10){
                                                chatCount = 0;

                                                var                area                =
document.getElementById("div_table");
                                                $('#msgs').append('<tr><td  valign="top"
style="padding:1px;   color:#F46690;">   [System]      :      </td><td   style="padding:1px;
color:#F46690;">' + '채팅 작작해라' + '\n' + '</td></tr>');
                                                area.scrollTop = area.scrollHeight;

                                                chatOn = false;
                                                setTimeout(function(){      chatOn      =
true; },10000);
                                        }
                                }

                                startTime = new Date().getTime();

                                var text = $('#msgbox').val();

                                if( text.substring(0, 3) == '/투표' || text.substring(0,
3) == '/ㅌㅍ'){
                                socket.emit('startVote', {from : myNickname});
                                        $('#msgbox').val('');
                                }
                                else   if(   text.substring(0,   4)   ==   '/도움말'   ||
text.substring(0, 1) == '/h' || text.substring(0, 5) == '/help'){
                                        socket.emit('help', myNickname);
                                        $('#msgbox').val('');
                                }
                                else {
                                        socket.emit('send_msg',         {toSend         :
$('#userList').val(), from : myNickname, msg : text});
                                        $('#msgbox').val('');
                                }
                        }
                        else {
                                var area = document.getElementById("div_table");
                                $('#msgs').append('<tr><td                  valign="top"
style="padding:1px;   color:#F46690;">   [System]      :      </td><td   style="padding:1px;
color:#F46690;">' + '너는 채팅 못함' + '\n' + '</td></tr>');
                                area.scrollTop = area.scrollHeight;
                        }
                }
        });


        function submit(kVote){
                var voting = document.getElementsByName('vote');
                var selected;

                for(var i=0; i<voting.length; i++)
                        if (voting[i].checked == true)
                                selected = voting[i].value;

                if(selected == undefined)
                  return;

                $('#voteBox').empty();
                $('#voteBtn').empty();

                if (kVote == 1) {
                        socket.emit('treatMVote', selected);
        $("#voteModal").modal('close');
                }
```

```
                        else if (kVote == 2) {
                                console.log('status ' + myStatus);
                                switch(myStatus){
                                        case 1:
                                                console.log('citizenVote');
                                                socket.emit('citizenResult', selected);
                                                $("#voteModal").modal('close');
                                                break;
                                        case 2:
                                                console.log('mafiaVote');
                                                socket.emit('mafiaResult', selected);
                                                $("#voteModal").modal('close');
                                                break;
                                        case 3:
                                                console.log('policeVote');
                                               socket.emit('policeResult', {selected : selected,
nick : myNickname});
                                                $("#voteModal").modal('close');
                                                break;
                                        case 4:
                                                console.log('doctorVote');
                                                socket.emit('doctorResult', selected);
                                $("#voteModal").modal('close');
                                                break;
                                }
                        }
                }

                socket.on('end', function(){
                                end = true;
                        });

                        socket.on('hide', function(){
                                $('#changeNameBtn').hide();
                        });

                        <!--닉네임 입력 창 갱신-->
                        socket.on('newUserCon', function(data){
                                //console.log('newUserCon : ' + data.nickname);
                                $('#nickname').val(data.nickname);
                                myNickname = data.nickname;
                        });

                        <!--닉네임 목록 갱신-->
                        socket.on('refreshList', function(data){
                                $('#userList').empty().append('<option    value="ALL"    data-
subtext="모두에게">ALL</option>');

                                for(var i=0; i<data.length; i++)
                                        $('#userList').append('<option value=' + data[i] + '>' +
data[i] + "</option>").selectpicker('refresh');

                                //인원 정보 갱신
                                $('#userInfo').empty().append('<thead><tr><th>직업</th><th>이름
</th></tr></thead>');

                                var _status;

                                for(var i=0; i<data.length; i++){
                                        if(end == true)
                                                $('#userInfo').append('<tr><td>'                +
status[data.status[i]] + '</td><td>' + data.user[i] + "</td></tr>");
                                        else
                                                $('#userInfo').append('<tr><td>?</td><td>'      +
data[i] + "</td></tr>");
                                }
                        });
```

```
                        <!--신분 표시-->
                        socket.on('def_status', function(data){
                                var area = document.getElementById("div_table");
                                $('#msgs').append('<tr><td valign="top"> [System]  :  </td><td
style="padding:1px; color:#FF4A68;">' + data.msg + '\n' + '</td></tr>');
                                area.scrollTop = area.scrollHeight;

                                myStatus = data.t_status;
                        });

                        <!--받은 메시지 표시-->

                        socket.on('sys_msg', function(data){
                                var area = document.getElementById("div_table");
                                $('#msgs').append('<tr><td  valign="top"  style="color:#'  +
data.color + ';"> [System]  :  </td><td style="padding:1px; color:#' + data.color + '">' +
data.msg + '\n' + '</td></tr>');
                                area.scrollTop = area.scrollHeight;
                        });

                        socket.on('get_msg', function(data){
                                var area = document.getElementById("div_table");
                                $('#msgs').append('<tr><td valign="top" style="color:#1975D1;">
[귓속말] ' + data.from + ' : </td><td>' + data.msg + '\n' + '</td></tr>');
                                area.scrollTop = area.scrollHeight;
                        });

                        socket.on('get_msg_all', function(data){
                                var area = document.getElementById("div_table");
                                $('#msgs').append('<tr><td valign="top"> [모두] ' + data.from +
' : </td><td>' + data.msg + '\n' + '</td></tr>');
                                area.scrollTop = area.scrollHeight;
                        });


                        <!--메인 투표-->
                        socket.on('mainVote', function(data){
                                $("#voteModal").modal('show');
                                $('#voteQst').append('낮의 투표로 죽일 사람은?');
                                for(var i=0; i<data.list.length; i++)
                                        $('#voteBox').append('<input  type="radio"  name="vote"
value="' + data.list[i] + '"/>' + data.list[i] + '<br>');
                                $('#voteBtn').append('<button  type="button"  class="btn  btn-
primary" data-dismiss="modal" onclick="submit(1)">선택</button></div>');
                        });

                        <!--직업 별 투표-->

                        socket.on('citizenVote', function(data){
                                $("#voteModal").modal('show');
                                $('#voteBox').append('대건고에서 가장 나이가 많은 사람은? <br>');
                                for(var i=0; i<data.list.length; i++)
                                        $('#voteBox').append('<input  type="radio"  name="vote"
value="' + data.list[i] + '"/>' + data.list[i] + '<br>');
                                $('#voteBtn').append('<button  type="button"  class="btn  btn-
primary" data-dismiss="modal" onclick="submit(2)">선택</button></div>');
                        });

                        socket.on('mafiaVote', function(data){
                                $("#voteModal").modal('show');
                                $('#voteBox').append('마피아는 죽일 사람을 선택하세요. <br>');
                                for(var i=0; i<data.list.length; i++){
                                        if(data.list[i] != myNickname)
                                                $('#voteBox').append('<input        type="radio"
name="vote" value="' + data.list[i] + '"/>' + data.list[i] + '<br>');
                                }
```

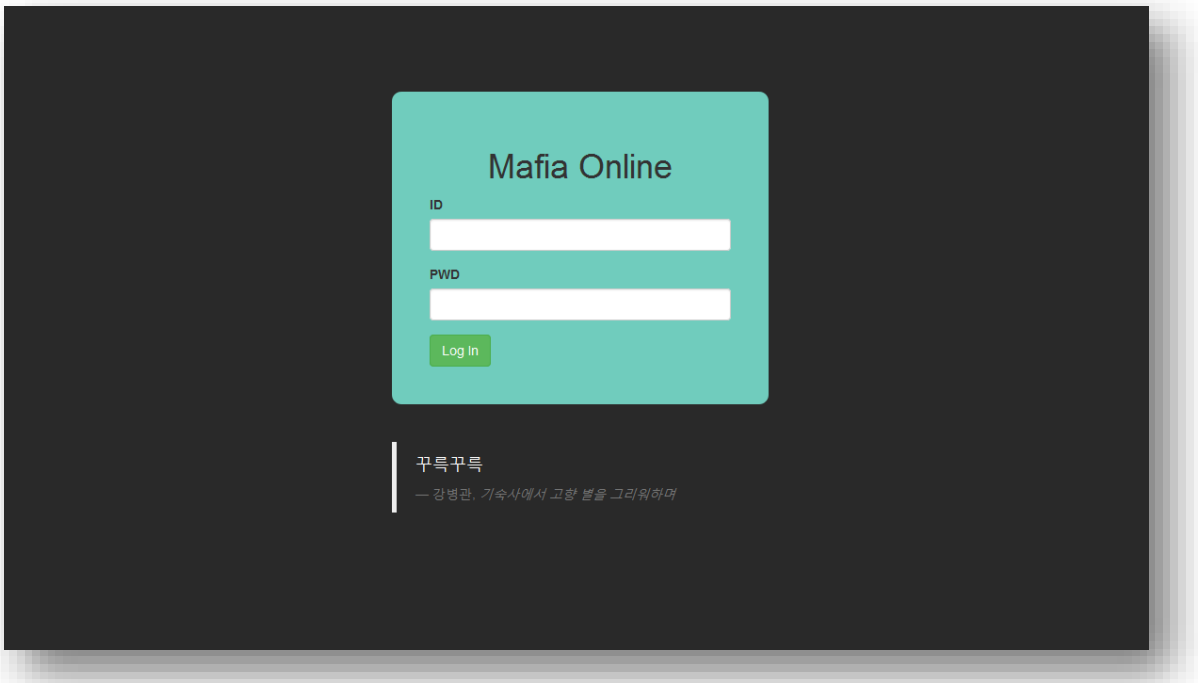
```
                        $('#voteBtn').append('<button  type="button"  class="btn  btn-
primary" data-dismiss="modal" onclick="submit(2)">선택</button></div>');
                });

                socket.on('policeVote', function(data){
                        $("#voteModal").modal('show');
                        $('#voteBox').append('경찰은 직업을 알고싶은 사람을 선택하세요. <br>');
                        for(var i=0; i<data.list.length; i++){
                                if(data.list[i] != myNickname)
                                        $('#voteBox').append('<input        type="radio"
name="vote" value="' + data.list[i] + '"/>' + data.list[i] + '<br>');
                        }
                        $('#voteBtn').append('<button  type="button"  class="btn  btn-
primary" data-dismiss="modal" onclick="submit(2)">선택</button></div>');
                });

                socket.on('doctorVote', function(data){
                        $("#voteModal").modal('show');
                        $('#voteBox').append('의사는 살리고 싶은 사람을 선택하세요. <br>');
                        for(var i=0; i<data.list.length; i++){
                                if(data.list[i] != myNickname)
                                $('#voteBox').append('<input  type="radio"  name="vote"
value="' + data.list[i] + '"/>' + data.list[i] + '<br>');
                        }
                        $('#voteBtn').append('<button  type="button"  class="btn  btn-
primary" data-dismiss="modal" onclick="submit(2)">선택</button></div>');
                });
        </script>
```

게임이 이루어지는 방의 클라이언트 코드이다. 대부분 서버에서 이벤트가 발생하면 이에 대한 Callback 함수의 형태로 이벤트를 처리한다.

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">

   <title>그많데가 약 빨고 만든 마피아 게임</title>

   <script src="/socket.io/socket.io.js"></script>
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
        <script src="js/bootstrap.min.js"></script>

        <link href="css/bootstrap.min.css" rel="stylesheet">

        <link href="select-box/dist/css/bootstrap-select.css" rel="stylesheet" >
        <script src="select-box/dist/js/bootstrap-select.js"></script>


        <style>
                @import url(http://fonts.googleapis.com/earlyaccess/naunmgothic.css);

                body {
                        font-family: "Helvetica Neue", Helvetica, Arial, 'Nanum Gothic',  sans-
serif;
          background: #292929;
      }

      form {
         background: #70CCBD;
         border-radius: 10px;
         padding: 40px;
         width: 400px;
         margin-top: 10%;
         margin-left: auto;
```

```
         margin-right: auto;
      }

      blockquote {
         width: 400px;
         margin-top: 40px;
         margin-left: auto;
         margin-right: auto;
      }
       </style>
</head>

<body>
       <div class="container">

         <form role="form">
            <h1 class="text-center">Mafia Online</h1>
            <div class="form-group">
               <label for="id">ID</label>
               <input type="text" class="form-control" id="userId"/>
            </div>
            <div class="form-group">
               <label for="pwd">PWD</label>
               <input type="text" class="form-control" id="userPwd"/>
            </div>
            <input type="button" class="btn btn-success" value="Log In"/>
         </form>

      <blockquote>
         <p style="color: #FFFFFF;">꾸륵꾸륵</p>
         <footer>강병관, <cite title="Source Title">기숙사에서 고향 별을 그리워하며</cite></footer>
      </blockquote>

       </div>

       <script type="text/javascript">
              function goReplace(str) {
                     location.replace(str);
              }
       </script>
</body>
</html>
```

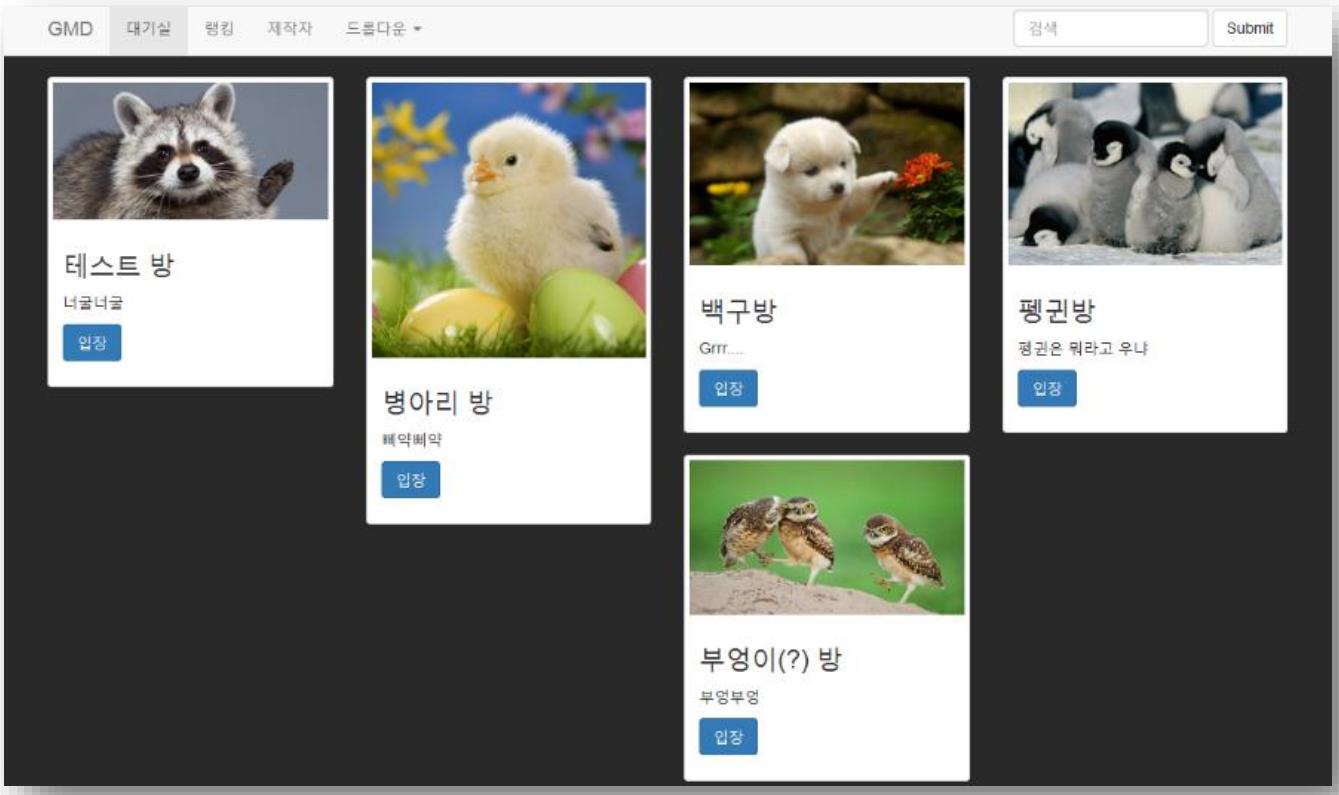
마피아 게임의 로그인 화면의 코드이다. 추후에 회원가입 페이지를 생성 후 라우팅 할 예정이다.



```
<body>
   <div class="container">

   <nav class="navbar navbar-default navbar-fixed-top" role="navigation">
      <!-- Brand and toggle get grouped for better mobile display -->
      <div class="container">
         <div class="navbar-header">
            <button   type="button"   class="navbar-toggle"   data-toggle="collapse"   data-
target=".navbar-ex1-collapse">
               <span class="sr-only">Toggle navigation</span>
               <span class="icon-bar"></span>
               <span class="icon-bar"></span>
               <span class="icon-bar"></span>
            </button>
            <a class="navbar-brand" href="#">GMD </a>
         </div>

         <!-- Collect the nav links, forms, and other content for toggling -->
         <div class="collapse navbar-collapse navbar-ex1-collapse">
            <ul class="nav navbar-nav">
               <li class="active"><a href="/robby">대기실</a></li>
               <li><a href="#">랭킹</a></li>
               <li><a href="/madeby">제작자</a></li>
               <li class="dropdown">
                  <a   href="#"   class="dropdown-toggle"   data-toggle="dropdown">드롭다운   <b
class="caret"></b></a>
                  <ul class="dropdown-menu">
                     <li><a href="#">서브메뉴 1</a></li>
                     <li><a href="#">서브메뉴 2</a></li>
                     <li><a href="#">서브메뉴 3</a></li>
                  </ul>
               </li>
         </ul>

         <form class="navbar-form navbar-right" role="search">
            <div class="form-group">
               <input type="text" class="form-control" placeholder="검색">
            </div>
            <button type="submit" class="btn btn-default">Submit</button>
         </form>

       </div><!-- /.navbar-collapse -->
    </div>

   </nav>

   <div class="row">
      <div class="col-sm-6 col-md-3">
         <div class="thumbnail">
            <img src="a.jpg" alt="...">
            <div class="caption">
               <h3>테스트 방</h3>
               <p>너굴너굴</p>
               <p><a href="/ingame" class="btn btn-primary" role="button">입장</a></p>
            </div>
         </div>
      </div>
      <div class="col-sm-6 col-md-3">
         <div class="thumbnail">
            <img src="b.jpg" alt="...">
            <div class="caption">
               <h3>병아리 방</h3>
               <p>삐약삐약</p>
```

```
                <p><a href="#" class="btn btn-primary" role="button">입장</a></p>
            </div>
        </div>
    </div>
    <div class="col-sm-6 col-md-3">
        <div class="thumbnail">
            <img src="c.jpg" alt="...">
            <div class="caption">
                <h3>백구방</h3>
                <p>Grrr....</p>
                <p><a href="#" class="btn btn-primary" role="button">입장</a></p>
            </div>
        </div>
    </div>
    <div class="col-sm-6 col-md-3">
        <div class="thumbnail">
            <img src="d.jpg" alt="...">
            <div class="caption">
                <h3>펭귄방</h3>
                <p>펭귄은 뭐라고 우냐</p>
                <p><a href="#" class="btn btn-primary" role="button">입장</a></p>
            </div>
        </div>
    </div>
    <div class="col-sm-6 col-md-3">
        <div class="thumbnail">
            <img src="e.jpg" alt="...">
            <div class="caption">
                <h3>부엉이(?) 방</h3>
                <p>부엉부엉</p>
                <p><a href="#" class="btn btn-primary" role="button">입장</a></p>
            </div>
        </div>
    </div>
</div>

<span class="label label-default">Default</span>
<span class="label label-primary">Primary</span>
<span class="label label-success">Success</span>
<span class="label label-info">Info</span>
<span class="label label-warning">Warning</span>
<span class="label label-danger">Danger</span>

</div>
</body>
```

마피아 게임의 로비의 코드이다. 방을 만든 사용자의 그림이 방의 썸네일 그림이 될 것이다. 현재 DB가 구현되어 있지 않기 때문에 임시로 Test 방을 하나 만들었다.



마지막 결과보고서를 작성하며

마지막 학습동아리 발표 대회를 준비하면서, 대건고에서의 공식적인 동아리 활동을 마치면서 많은 여운이 남습니다. 그 많데에서 재흠이와 제령이가 1년 넘게 진행한 게임 프로젝트도 끝이 보이고 정현이와 석훈이가 개발한 SNS도 끝이 났습니다. 찬란했던 그많데 1기는 앞으로 무엇을 할까요? 저희도 궁금하네요. 하하