

고급웹프로그래밍

중간 시험 – Part2

김지현

60181628

< 목차 >

1. 웹사이트의 주제 & 주제 선정 이유
2. 각 소스코드의 이름 & 기능 설명
3. 실행방법 & 실행 과정별 결과

1. 웹사이트의 주제 & 주제 선정 이유

- 웹사이트 주제

학생 정보와 조 인원 수를 입력하면 자동으로 조를 나누어 주는 웹사이트

- 주제 선정 이유

이번에 캡스톤디자인1을 수강하는데,

OT를 진행할 때 조를 나누는 시간과 과정이 길다고 느꼈습니다.

그래서 학생들을 무작위로 추천하여 조를 나누는 방식이

효율적일 것 같아 이 주제를 선정하게 되었습니다.

2. 각 소스코드의 이름 & 기능 설명

전체적인 구조

```
> node_modules
└─ public
   └─ img
      ├── logout.png
      ├── meeting.png
      └── professor.jpg
      # draw.css
      <> draw.html
      JS draw.js
      # user.css
      <> user.html
      JS user.js
   └─ views
      └─ img
         ├── meeting.png
         # login.css
         login.pug
      .env
      JS app.js
      {} package-lock.json
      {} package.json
```

app.js

```
JS app.js X
JS app.js > ...
1  const dotenv = require('dotenv');
2  dotenv.config();
3
4  const express = require('express');
5  const morgan = require('morgan');
6  const cookieParser = require('cookie-parser');
7  const session = require('express-session');
8  var pug = require('pug');
9
10 const path = require('path');
11 const app = express();
12 // process.env.PORT가 없으면 3000으로 port 지정
13 app.set('port', process.env.PORT || 3000);
14
15 // Pug 사용을 위한 설정
16 app.set('views', path.join(__dirname, 'views'));
17 app.set('view engine', pug );
18
19 // morgan 사용을 위한 설정
20 app.use(morgan('dev'));
21 // static 사용을 위한 설정
22 app.use('/', express.static(path.join(__dirname, 'public')));
23 // body-parser 사용을 위한 설정
24 app.use(express.json());
25 // body-parser 사용을 위한 설정
26 app.use(express.urlencoded({ extended: false }));
27 // cookie-parser 사용을 위한 설정
28 app.use(cookieParser(process.env.COOKIE_SECRET));
29 // express-session 사용을 위한 설정
30 app.use(session({
31   resave: false,
32   saveUninitialized: true,
33   secret: process.env.COOKIE_SECRET,
34   cookie: {
35     httpOnly: true,
36     secure: false,
37     maxAge: 6000000,
```

서버 구동의 핵심이 되는 파일

1. 서버가 실행될 포트 지정

2. PATH와 METHOD에 따른 동작 지정

3. Middleware 지정

(에러처리, morgan, cookie-parser, express-session, body-parser 등)

4. 지정된 포트에서 서버 실행

package.json

현재 프로젝트에 대한 정보와
사용중인 패키지에 대한 정보

{ } package.json X

{ } package.json > { } dependencies

```
1  {
2    "name": "web_midterm_exam",
3    "version": "0.0.1",
4    "description": "",
5    "main": "app.js",
6    "scripts": {
7      "start": "nodemon app.js",
8      "test": "echo \"Error: no test specified\" && exit 1"
9    },
10   "keywords": [],
11   "author": "KJH",
12   "license": "MIT",
13   "dependencies": {
14     "cookie-parser": "^1.4.5",
15     "dotenv": "^8.2.0",
16     "express": "^4.17.1",
17     "express-session": "^1.17.1",
18     "morgan": "^1.10.0",
19     "pug": "^3.0.2"
20   }
21 }
22
```

package-lock.json

패키지 간 의존 관계에 대한 정보

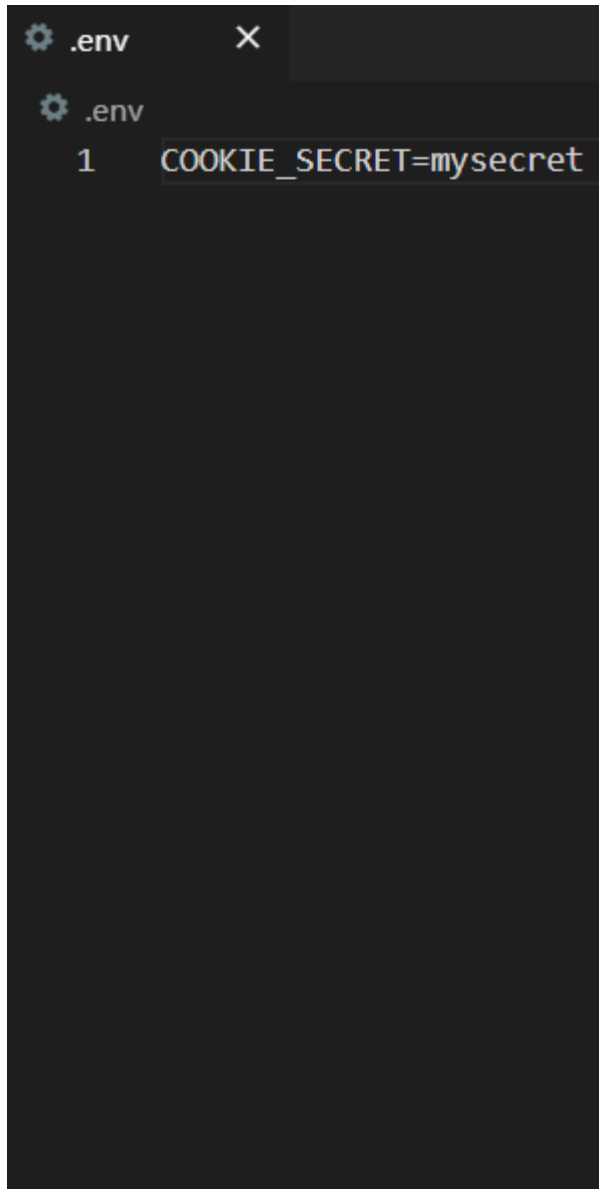
{ } package-lock.json X

{ } package-lock.json > { } dependencies > { } @babel/helper-validator-identifier

```
1  {
2    "name": "web_midterm_exam",
3    "version": "0.0.1",
4    "lockfileVersion": 1,
5    "requires": true,
6    "dependencies": {
7      "@babel/helper-validator-identifier": {
8        "version": "7.12.11",
9        "resolved": "https://registry.npmjs.org/@babel/helper-validator-identifier/-/helper-validator-identifier-7.12.11.tgz",
10       "integrity": "sha512-np/L63uARFybkoHokJUmf1QfEvRVCPbmQeUQpKow5cQ3xWV9i3rUHodKDJQfTVX61qKi+UdYk8kik84n7X0w=="
11     },
12     "@babel/parser": {
13       "version": "7.13.16",
14       "resolved": "https://registry.npmjs.org/@babel/parser/-/parser-7.13.16.tgz",
15       "integrity": "sha512-6bAg36mCwuqL00hbR+z7PHuqWiCeP7Dzg73OpQwsAB1Eb8HnGEz5xYBzCfbu+YjoaJsJs+qheDxVAuqbt3ILEw=="
16     },
17     "@babel/types": {
18       "version": "7.13.17",
19       "resolved": "https://registry.npmjs.org/@babel/types/-/types-7.13.17.tgz",
20       "integrity": "sha512-RawydLgxbOPDlTLJNtoIypwmdmAY//uQIz1Kt2+iBiJaRlVuI6QLuXVayWGNf0zp8YU4L4lIacoCyTntpb4wiA==",
21       "requires": {
22         "@babel/helper-validator-identifier": "^7.12.11",
23         "to-fast-properties": "^2.0.0"
24       }
25     },
26     "accepts": {
27       "version": "1.3.7",
28       "resolved": "https://registry.npmjs.org/accepts/-/accepts-1.3.7.tgz",
29       "integrity": "sha512-3eNHSbkYhZLJ8i9G6BpJJYdV81JJG0UJBB0Esz0HQclwKev1eW2C1N+0s1a5W8wVr4n7PZkNtdIG1d2jA==",
30       "requires": {
31         "mime-types": "~2.1.24",
32         "negotiator": "0.6.2"
33       }
34     },
35     "acorn": {
36       "version": "7.4.1",
37       "resolved": "https://registry.npmjs.org/acorn/-/acorn-7.4.1.tgz",
```


.env

process.env 관리 파일

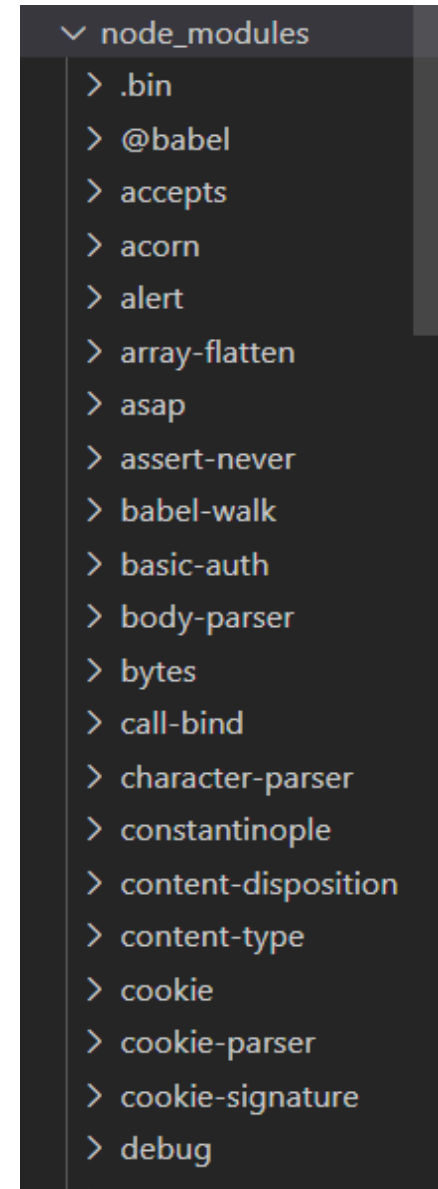


A screenshot of a code editor with a dark theme. The top of the editor shows a tab for '.env' with a gear icon on the left and a close 'X' icon on the right. Below the tab, the file content is displayed. Line 1 contains the text 'COOKIE_SECRET=mysecret'.

```
.env
1  COOKIE_SECRET=mysecret
```

node_modules 폴더

express, express와 의존 관계가 있는 패키지



A screenshot of a file explorer with a dark theme. The 'node_modules' directory is expanded, showing a list of subdirectories. The list includes: .bin, @babel, accepts, acorn, alert, array-flatten, asap, assert-never, babel-walk, basic-auth, body-parser, bytes, call-bind, character-parser, constantinople, content-disposition, content-type, cookie, cookie-parser, cookie-signature, and debug.

- > node_modules
 - > .bin
 - > @babel
 - > accepts
 - > acorn
 - > alert
 - > array-flatten
 - > asap
 - > assert-never
 - > babel-walk
 - > basic-auth
 - > body-parser
 - > bytes
 - > call-bind
 - > character-parser
 - > constantinople
 - > content-disposition
 - > content-type
 - > cookie
 - > cookie-parser
 - > cookie-signature
 - > debug

login

로그인 기능 (app.js에서 설정한 id와 pw가 맞으면 user.html로 넘어감)

```
login.pug X
views > login.pug
1  doctype html
2  head
3    meta(charset='utf-8')
4    title 로그인
5    style
6    ||| include login.css
7  #main
8    #main_top
9    #main_left
10     img#logo(src='./img/meeting.png')
11     span#logo_text1 조
12     span#logo_text2 추천
13     span#logo_text3 시스템
14     form#form(action='/admit', method='post')
15       span#id_text 아이디
16       input#id_input(type='text', name='login')
17       br
18       span#pwd_text 비밀번호
19       input#pwd_input(type='password', name='password')
20       br
21       button#login_btn(type='submit') 로그인
22
```

login.pug

```
# login.css X
views > # login.css > *
1  * {
2    margin: 0;
3    padding: 0;
4    box-sizing: border-box;
5    border: none;
6  }
7  #main {
8    position: absolute;
9    width: 1536px;
10   height: 720px;
11   background-color: rgba(255,255,255,1);
12   overflow: hidden;
13 }
14 #main_top {
15   overflow: visible;
16   position: absolute;
17   width: 1215px;
18   height: 100px;
19   left: 320px;
20   top: 0px;
21   background-color: rgba(126,158,233,1);
22   text-align: center;
23 }
24 #main_left {
25   overflow: visible;
26   position: absolute;
27   width: 320px;
28   height: 1080px;
29   left: 0px;
30   top: 0px;
31   background-color: rgba(126,158,233,1);
32 }
33 #logo {
34   position: relative;
35   width: 250px;
36   height: 250px;
37   top: 130px;
38   overflow: visible;
```

login.css

user

학생 사용자의 정보(이름, 학번)을 등록하고 조 인원 수를 등록하는 기능 / 다시 로그인 페이지로 돌아갈 수도 있음

```
user.html
1 <!DOCTYPE html>
2 <html lang="ko">
3
4 <head>
5   <meta charset="utf-8">
6   <title>조 추천</title>
7   <link rel="stylesheet" href="/user.css" />
8 </head>
9
10 <body>
11   <div id="main">
12     <div id="main_top">
13       <span id="main_top_text">조 추천</span>
14       <a href="/login" id="logout"></a>
15     </div>
16     <div id="main_left">
17       
18       <span id="text_pro_name">김명민 교수님</span>
19       <span id="text_course_name">캡스톤디자인1</span>
20       <span id="text_course_num">HED01401</span>
21     </div>
22     <div id="column_std_info">
23       <span id="text_std_info">학생 정보</span>
24     </div>
25     <div id="column_std_list">
26       <span id="text_std_list">학생 명단</span>
27     </div>
28     <div id="std_info">
29       <form id="student_form">
30         <span id="name_text">이름</span>
31         <input type="text" id="name_input" name="name"><br>
32         <span id="num_text">학번</span>
33         <input type="text" id="num_input" name="stNum"><br>
34         <button type="submit" id="btn_std_info">추가</button>
35       </form>
36     </div>
37     <div id="std_list">
38   </div>
```

user.html

```
user.css
1 * {
2   margin: 0;
3   padding: 0;
4   box-sizing: border-box;
5   border: none;
6 }
7 #main {
8   position: absolute;
9   width: 1536px;
10  height: 720px;
11  background-color: #d3d3d3;
12  overflow: hidden;
13 }
14 #main_top {
15   overflow: visible;
16   position: absolute;
17   width: 1215px;
18   height: 100px;
19   left: 320px;
20   top: 0px;
21   background-color: #808080;
22   text-align: center;
23 }
24 #main_top_text {
25   position: relative;
26   overflow: visible;
27   top: 20px;
28   font-family: Yu Gothic;
29   font-style: normal;
30   font-weight: bold;
31   font-size: 45px;
32   color: #d3d3d3;
33 }
34 #logout {
35   position: absolute;
36   overflow: visible;
37   top: 5px;
38   left: 1100px;
```

user.css

```
user.js
1 //페이지 로딩 시 학생 사용자 정보를 가져오는 함수
2 async function User() {
3   try {
4     // 학생들의 정보를 가지고 올
5     const res = await axios.get('/users');
6     const users = res.data;
7
8     // id가 std_list인 것을 std_list로 설정
9     const std_list = document.getElementById('std_list');
10    std_list.innerHTML = '';
11
12    // 학생들을 학생명단에 반복적으로 화면 표시
13    Object.keys(users).map(function (key) {
14      // div를 userDiv로 생성
15      const userDiv = document.createElement('div');
16      // span을 span으로 생성
17      const span = document.createElement('span');
18      // 학생 사용자의 이름과 학번을 span으로 지정함
19      span.textContent = users[key].name + ' (' + users[key].stNum + ')';
20      // span의 속성을 지정 (style을 바꿈 - 폰트)
21      span.setAttribute("style", "font-family: Yu Gothic;font-style: normal");
22      // button을 edit로 생성
23      const edit = document.createElement('button');
24      // edit를 '수정'으로 지정함
25      edit.textContent = '수정';
26      // edit의 속성을 지정 (style을 바꿈 - 폰트, 색상, 테두리)
27      edit.setAttribute("style", "margin-left: 10px ; margin-right: 3px");
28      // edit에 event를 등록 시킴 click시 실행
29      edit.addEventListener('click', async () => {
30        // name, stNum prompt를 띄워 수정 값을 받는다
31        const name = prompt('수정할 이름을 입력하세요');
32        const stNum = prompt('수정할 학번을 입력하세요');
33        // name, stNum 값을 prompt에 입력하지 않았을 경우
34        if (!name || !stNum) {
35          return alert('수정할 이름과 학번을 반드시 입력하여야합니다.');
```

user.js

draw

user에서 입력 받은 학생들의 정보를 조 인원 수만큼 추출하는 기능 / 학생들을 무작위로 추출함 / user로 돌아가서 재설정할 수 있음

```
draw.html X
public > draw.html > ...
1  <!DOCTYPE html>
2  <html lang="ko">
3
4  <head>
5    <meta charset="utf-8">
6    <title>조 추출 결과</title>
7    <link rel="stylesheet" href="./draw.css" />
8  </head>
9
10 <body>
11   <div id="main">
12     <div id="main_top">
13       <span id="main_top_text">조 추출 결과</span>
14     </div>
15     <div id="main_left">
16       <button id="replay" type="button" onclick="location.href='user.html'">다시 하기</button>
17       
18     </div>
19     <div id="main_center">
20       <div id="result"></div>
21       <span id="span"></span>
22     </div>
23   </div>
24   <script src="https://unpkg.com/axios/dist/axios.min.js"></script>
25   <script src="./draw.js"></script>
26
27 </body>
28
29 </html>
```

draw.html

```
# draw.css X
public > # draw.css > *
1  * {
2    margin: 0;
3    padding: 0;
4    box-sizing: border-box;
5    border: none;
6  }
7  #main {
8    position: absolute;
9    width: 1536px;
10   height: 720px;
11   background-color: #000000;
12   overflow: hidden;
13 }
14 #main_top {
15   overflow: visible;
16   position: absolute;
17   width: 1215px;
18   height: 100px;
19   left: 320px;
20   top: 0px;
21   background-color: #000000;
22   text-align: center;
23 }
24 #main_top_text {
25   position: relative;
26   overflow: visible;
27   top: 20px;
28   font-family: Yu Gothic;
29   font-style: normal;
30   font-weight: bold;
31   font-size: 45px;
32   color: #000000;
33 }
34 #main_left {
35   overflow: visible;
36   position: absolute;
```

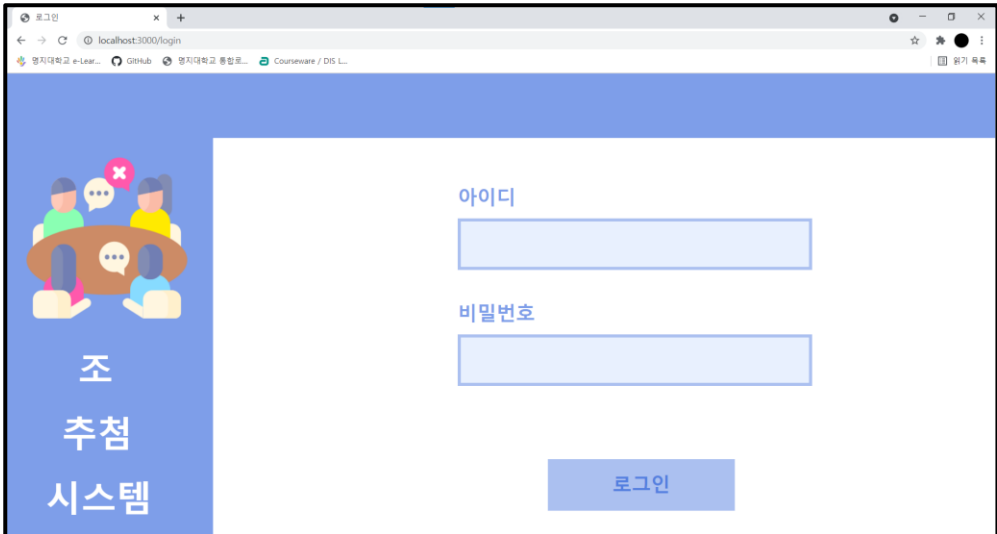
draw.css

```
JS draw.js X
public > JS draw.js > Drawer
1  // 페이지 로딩 시 조 추출 결과를 가져오는 함수
2  async function Drawer() {
3    try {
4      // 학생들의 정보를 가지고 올
5      const res = await axios.get('/users');
6      // 조 추출 인원 수 정보를 가지고 올
7      const res2 = await axios.get('/user_num');
8
9      const users = res.data;
10     const user_num = res2.data;
11
12     // 조 추출 인원 수를 num으로 설정
13     const num = Number(user_num[0].number);
14
15     // id가 result인 것을 result로 설정
16     const result = document.getElementById('result');
17     result.innerHTML = '';
18
19     // div를 resultDiv으로 생성
20     const resultDiv = document.createElement('div');
21     // id가 span인 것을 result로 설정
22     const span = document.getElementById('span');
23
24     // 학생들의 key를 받기 위해 선언
25     const users_key = [];
26
27     // 학생들의 key들을 선언된 users_key에 key를 삽입
28     Object.keys(users).map(function (key) {
29       users_key.push(key);
30     });
31
32     // 학생들을 무작위로 추출하기 위해 함수 실행
33     const random_users = Random_Users(users_key)
34
35     // 배열의 0번째부터 추출하기 위해 user를 0으로 선언
36     var user = 0;
37     // 1조부터 시작하기 위해 team_num을 1로 선언
```

draw.js

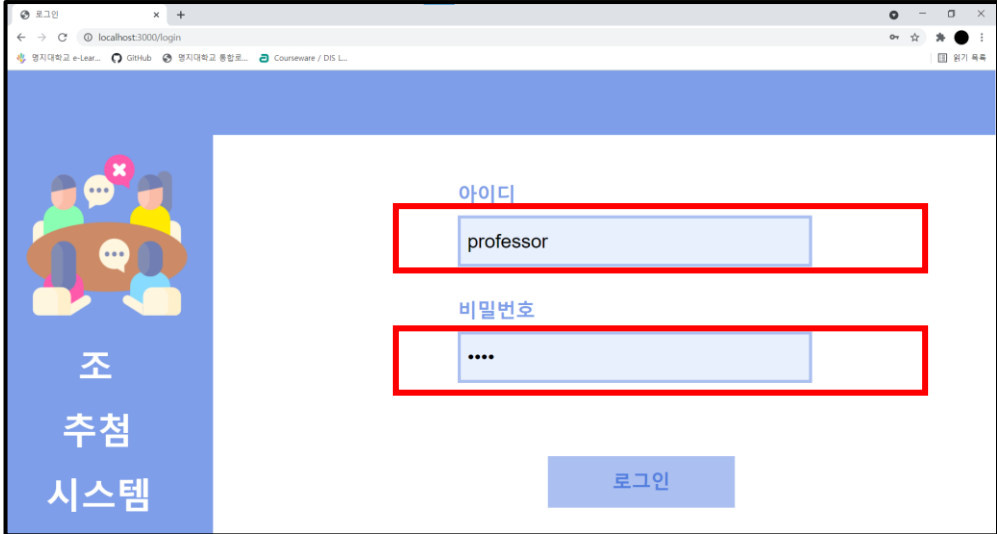
3. 실행방법 & 실행 과정별 결과

login 화면



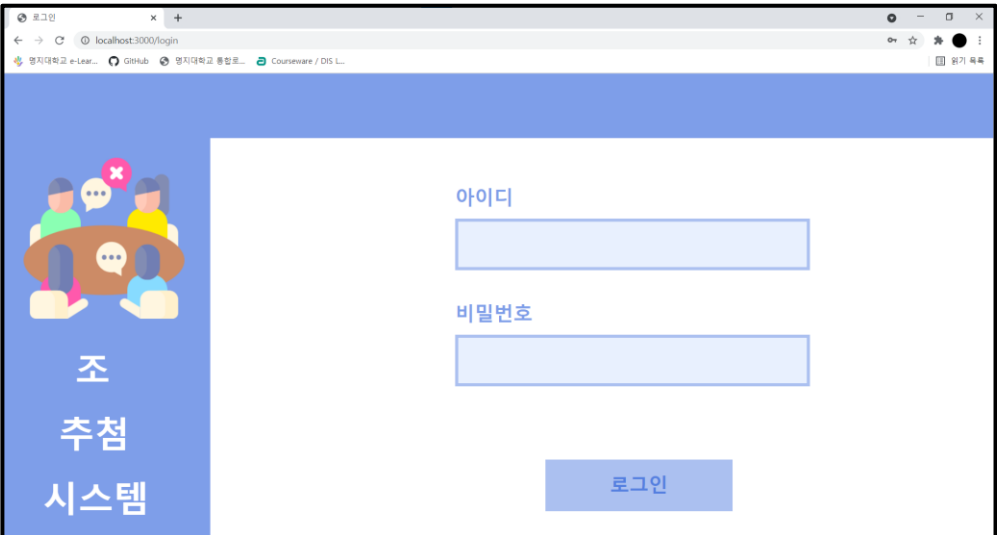
The login screen features a blue header with the text '로그인' (Login). On the left, there is a blue sidebar with a white icon of four people and the text '조 추천 시스템' (Group Recommendation System). The main area is white and contains two input fields: '아이디' (ID) and '비밀번호' (Password). A blue '로그인' (Login) button is at the bottom right.

메인 화면



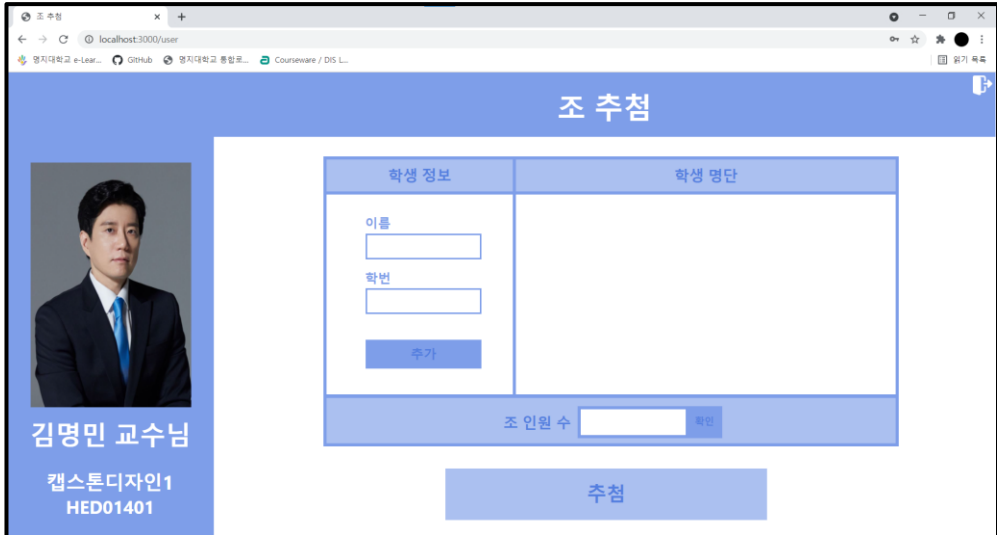
The login screen is the same as the main screen, but the '아이디' (ID) field is filled with 'professor' and the '비밀번호' (Password) field is filled with '....'. Both input fields are highlighted with a red border.

로그인 정보 입력시



The login screen is the same as the main screen, but the '로그인' (Login) button is now disabled and greyed out. The text '로그인 실패시' (Login Failed) is displayed below the button.

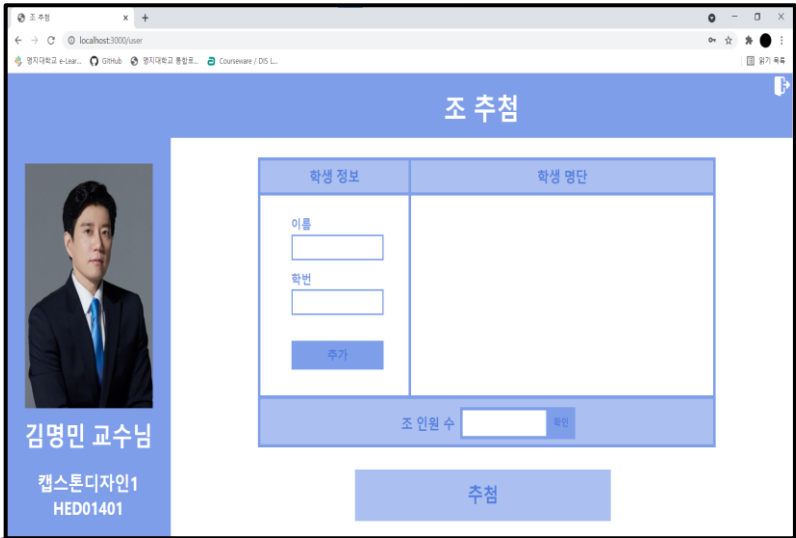
로그인 실패시



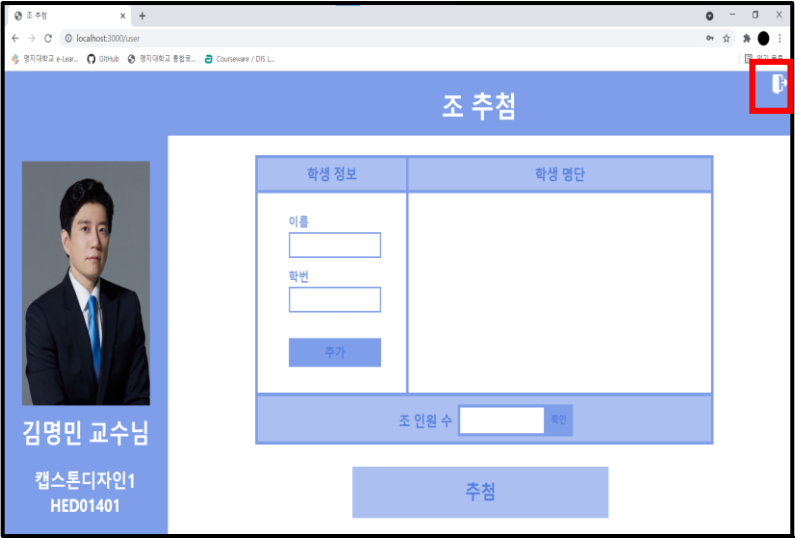
The login screen is the same as the main screen, but the '로그인' (Login) button is now enabled and blue. The text '로그인 성공시' (Login Successful) is displayed below the button. The main area is now white and contains a blue sidebar with a white icon of a person and the text '조 추천 시스템' (Group Recommendation System). The main area is divided into two sections: '학생 정보' (Student Information) and '학생 명단' (Student List). The '학생 정보' section contains a profile picture of a man, the name '김명민 교수님' (Professor Kim Myung-min), and the ID '캡스톤디자인1 HED01401'. The '학생 명단' section contains a table with columns '이름' (Name) and '학번' (Student ID Number). The table has one row with the name '김명민' and the student ID number 'HED01401'. A blue '추가' (Add) button is at the bottom right of the table. A blue '추천' (Recommend) button is at the bottom right of the page.

로그인 성공시

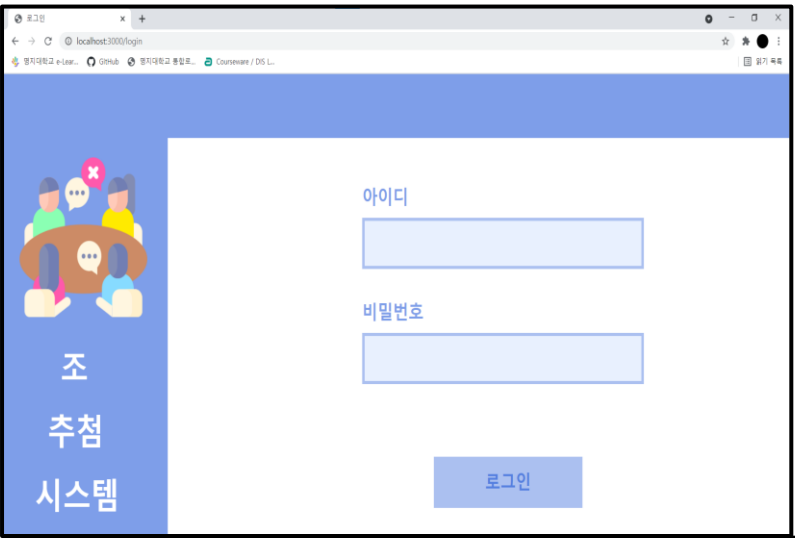
user 화면 - 1



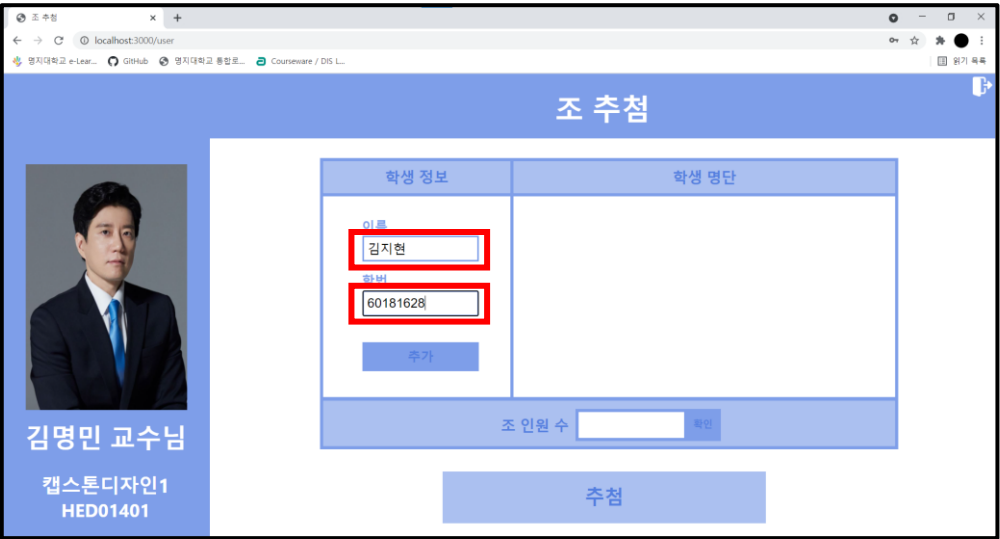
메인 화면



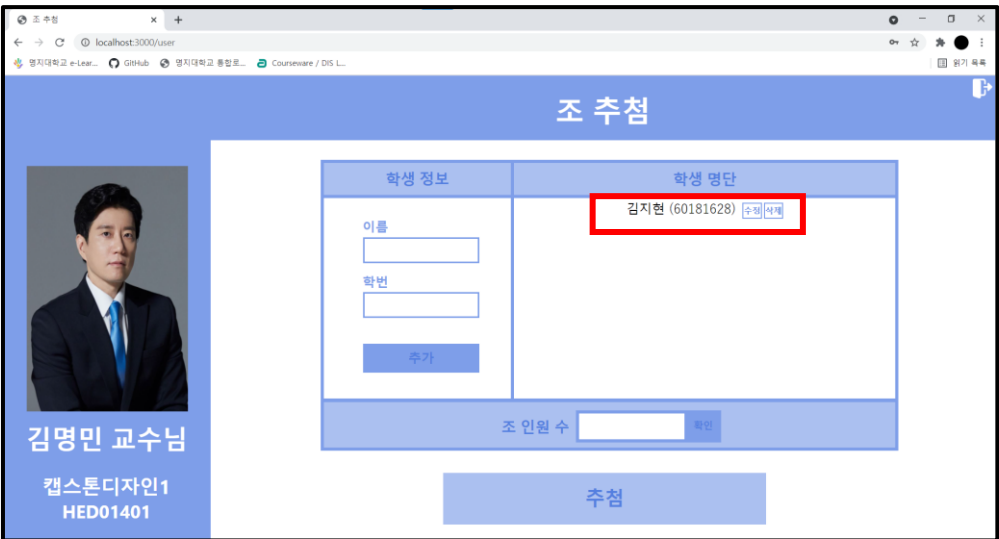
버튼 클릭시



버튼 클릭시 login 화면으로 전환




학생 정보 입력



학생 정보 입력 후

user 화면 - 2

조추첨



김명민 교수님
캡스톤디자인1
HED01401

학생 정보

이름
김지현
학번
60181628
추가


학생 명단

조 인원 수 확인

추첨

학생 정보 입력

조추첨



김명민 교수님
캡스톤디자인1
HED01401

학생 정보

이름

학번

추가


학생 명단
김지현 (60181628) 등록 삭제

조 인원 수 확인

추첨

학생 정보 성공적으로 입력시

조추첨



김명민 교수님
캡스톤디자인1
HED01401

학생 정보

이름

학번
60181628
추가


학생 명단

조 인원 수 확인

추첨

이름 미입력시

조추첨



김명민 교수님
캡스톤디자인1
HED01401

학생 정보

이름
김지현
학번

추가


학생 명단

조 인원 수 확인

추첨

학번 미입력시

조추첨



김명민 교수님
캡스톤디자인1
HED01401

학생 정보

이름

학번

추가

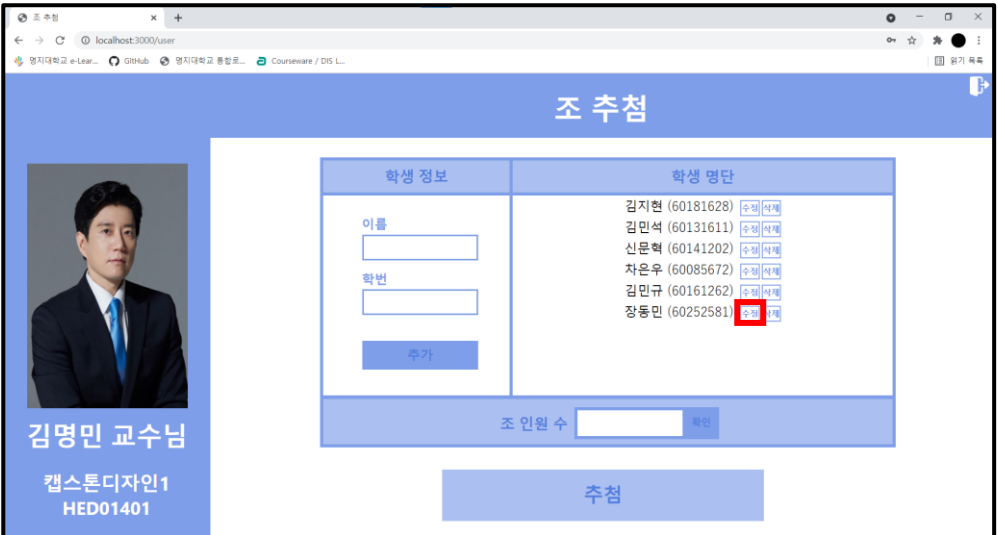
학생 명단

조 인원 수 확인

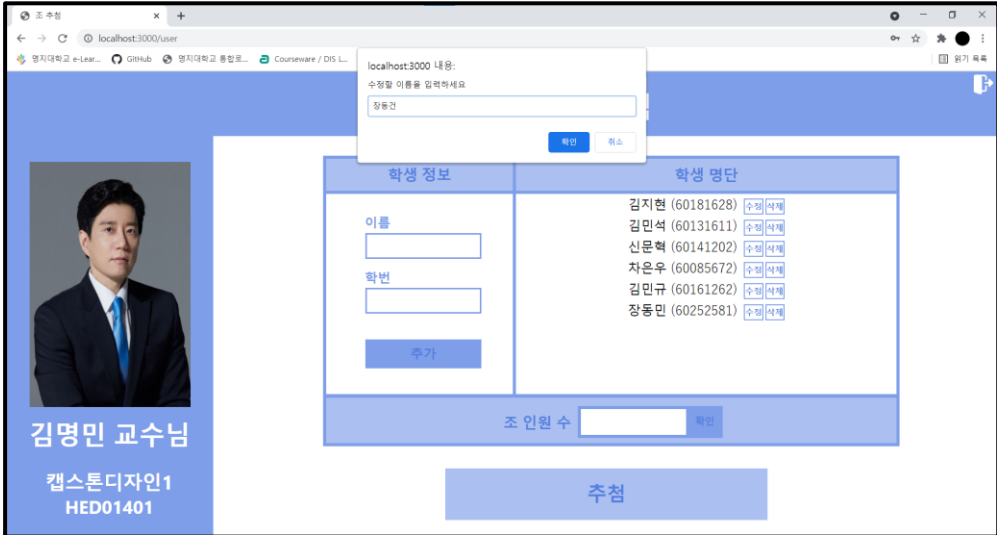
추첨

이름 & 학번 미입력시

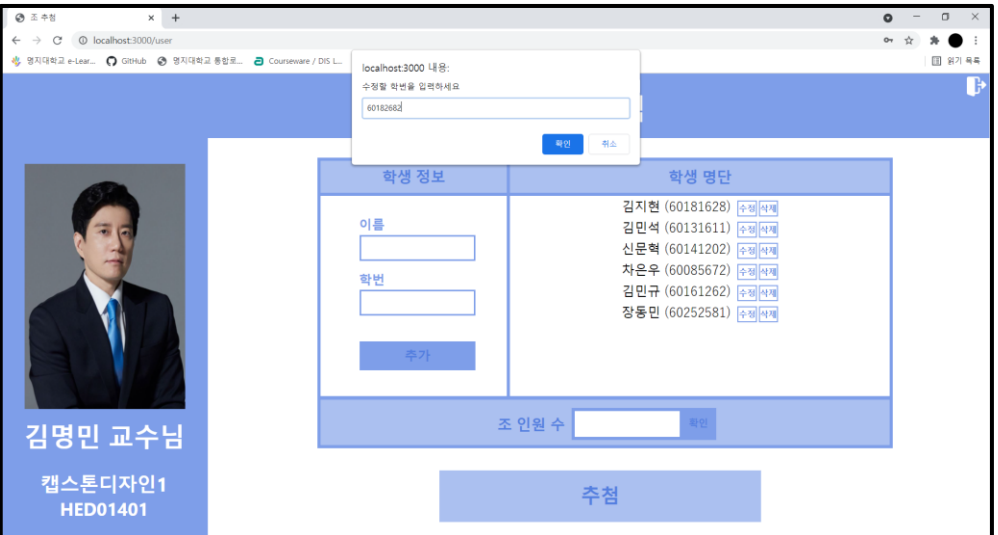
user 화면 - 3



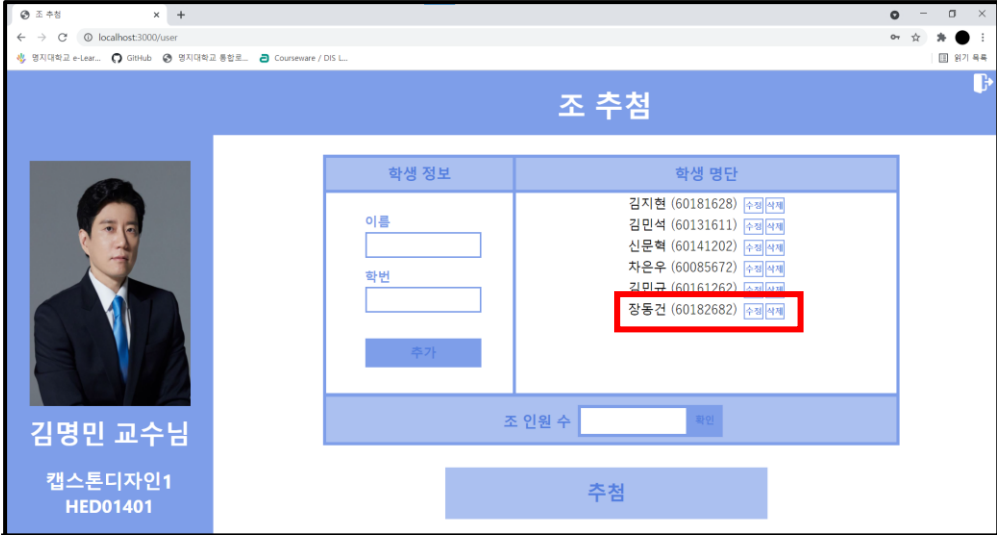
수정 버튼 클릭시



수정할 이름을 입력하는 prompt창이 뜬

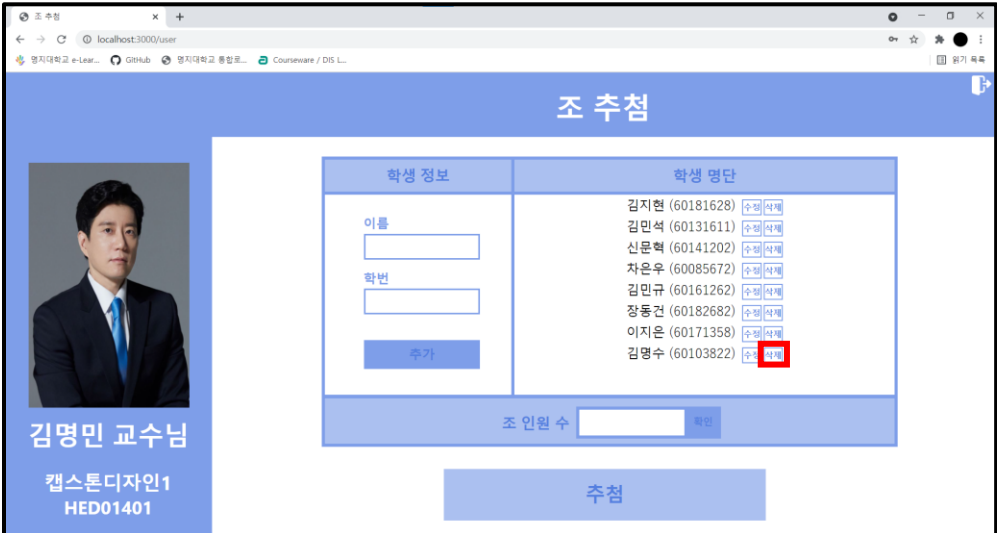


수정할 학번을 입력하는 prompt창이 뜬

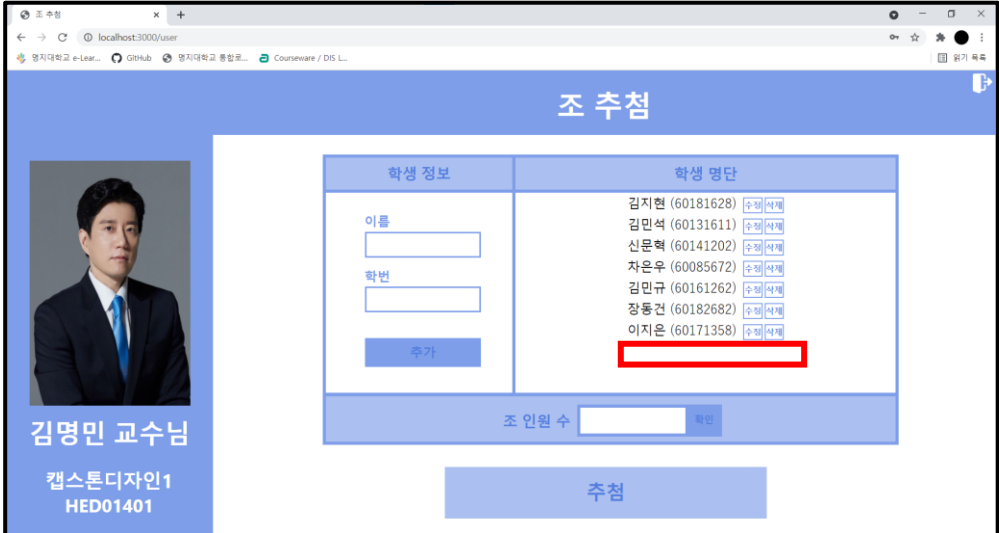


수정 완료

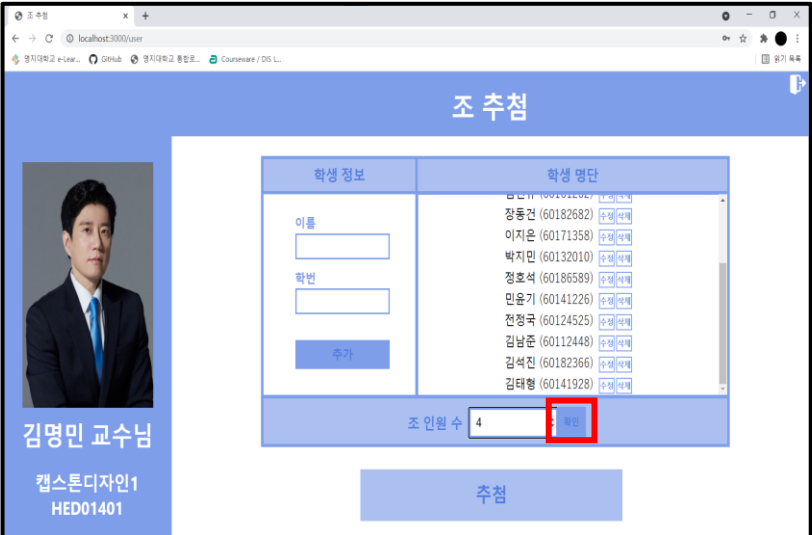
user 화면 - 4



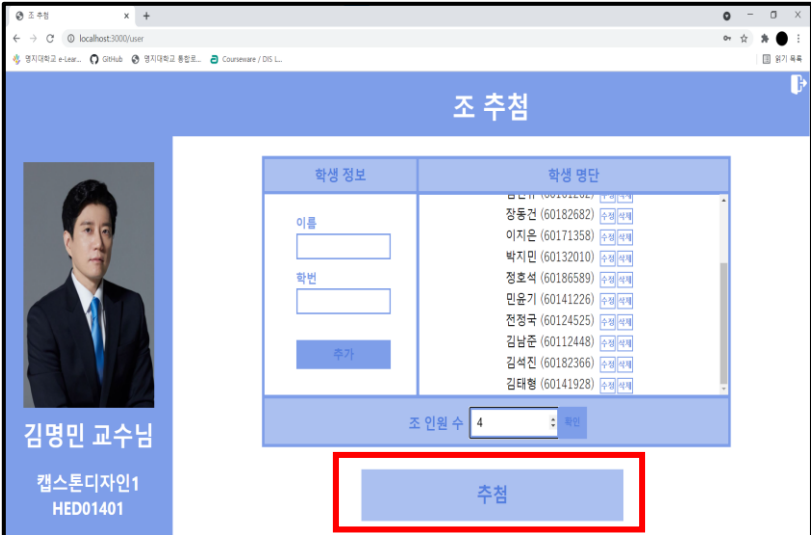
삭제 버튼 클릭시



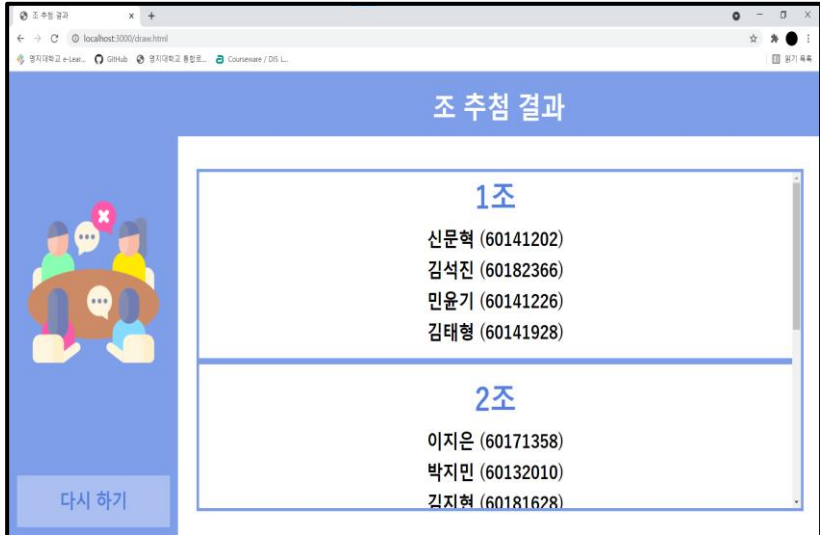
삭제 완료



조 인원 수 입력 후 확인버튼 클릭

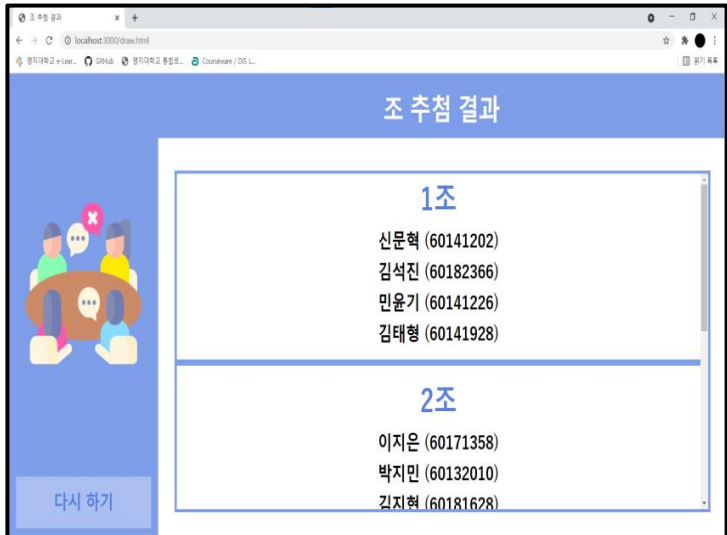


추천버튼 클릭시

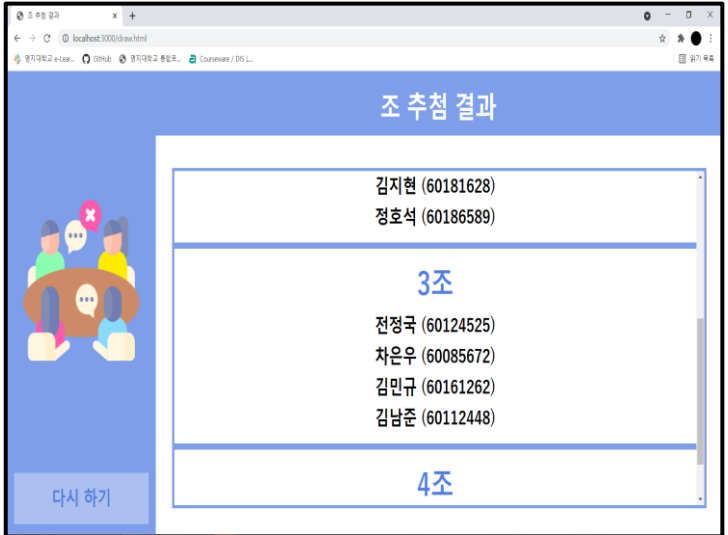


추천버튼 클릭시 draw 화면으로 전환

draw 화면



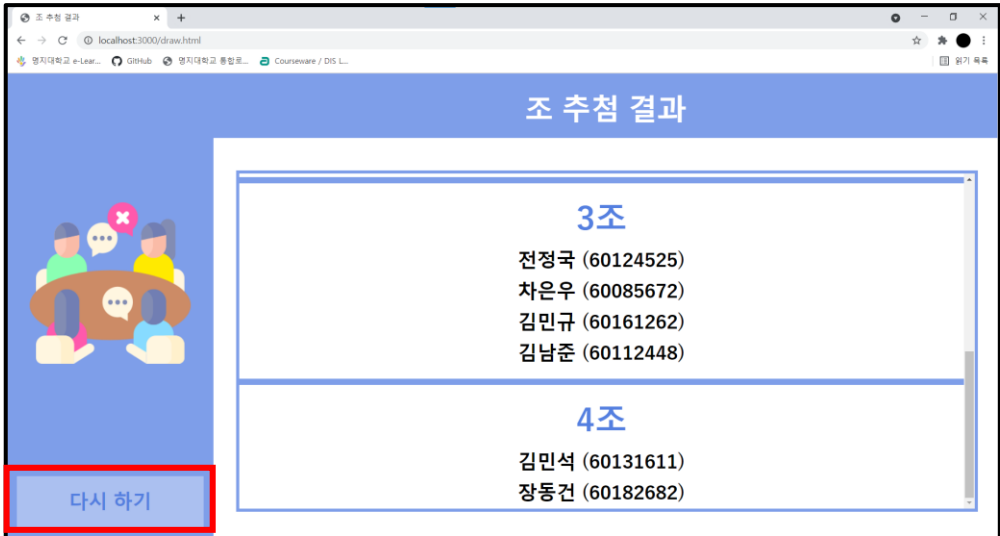
결과화면 1



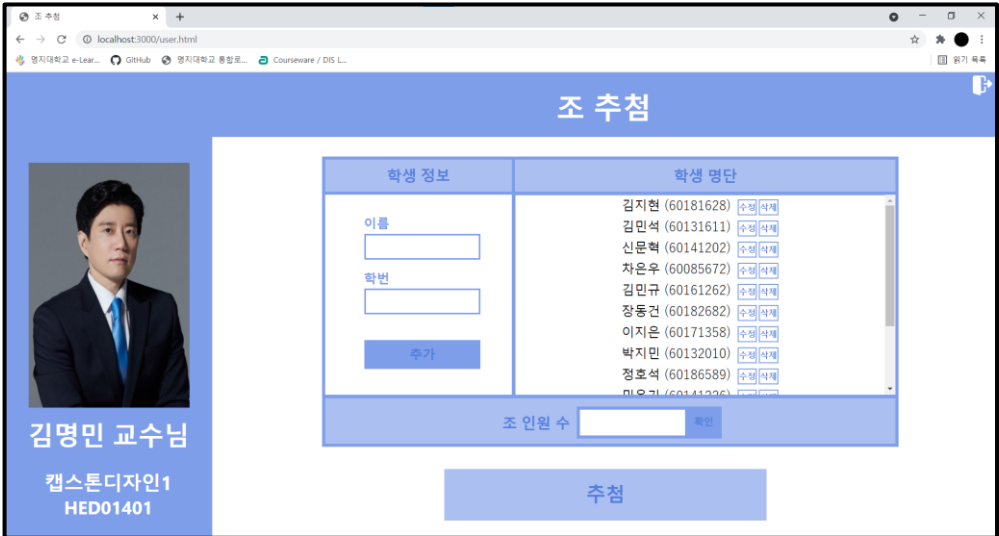
결과화면 2



결과화면 3



다시 하기 버튼 클릭시



다시 하기 버튼 클릭시 user 화면으로 전환

user 화면 - draw 화면 (다시하기를 클릭했을 때)

user 화면

조 추천

김명민 교수님
캡스톤디자인1
HED01401

학생 정보

이름
학번

추가

학생 명단

신문혁 (60141202)	[수정] [삭제]
차은우 (60085672)	[수정] [삭제]
김민규 (60161262)	[수정] [삭제]
이지은 (60171358)	[수정] [삭제]
정호석 (60186589)	[수정] [삭제]
민윤기 (60141226)	[수정] [삭제]
김남준 (60112448)	[수정] [삭제]
김석진 (60182366)	[수정] [삭제]
김태형 (60141928)	[수정] [삭제]
윤정환 (60172642)	[수정] [삭제]

조 인원 수 5

확인

추천

학생 추가, 삭제, 수정을 통해 총 10명인 상태에서 조 인원 수를 5로 설정

user 화면

조 추천

김명민 교수님
캡스톤디자인1
HED01401

학생 정보

이름
학번

추가

학생 명단

신문혁 (60141202)	[수정] [삭제]
차은우 (60085672)	[수정] [삭제]
김민규 (60161262)	[수정] [삭제]
이지은 (60171358)	[수정] [삭제]
정호석 (60186589)	[수정] [삭제]
민윤기 (60141226)	[수정] [삭제]
김남준 (60112448)	[수정] [삭제]
김석진 (60182366)	[수정] [삭제]
김태형 (60141928)	[수정] [삭제]
윤정환 (60172642)	[수정] [삭제]

조 인원 수 5

확인

추천

추천 버튼 클릭

draw 화면

조 추천 결과

1조

차은우 (60085672)
신문혁 (60141202)
이지은 (60171358)
김민규 (60161262)
민윤기 (60141226)

2조

김태형 (60141928)
윤정환 (60172642)

다시 하기

draw 화면으로 전환

draw 화면

조 추천 결과

1조

차은우 (60085672)
신문혁 (60141202)
이지은 (60171358)
김민규 (60161262)
민윤기 (60141226)

2조

김태형 (60141928)
윤정환 (60172642)

다시 하기

결과 화면 1

draw 화면

조 추천 결과

1조

차은우 (60085672)
신문혁 (60141202)
이지은 (60171358)
김민규 (60161262)
민윤기 (60141226)

2조

김태형 (60141928)
윤정환 (60172642)
김석진 (60182366)
김남준 (60112448)
정호석 (60186589)

다시 하기

결과 화면 2