EVERYTIME

Celeb Five

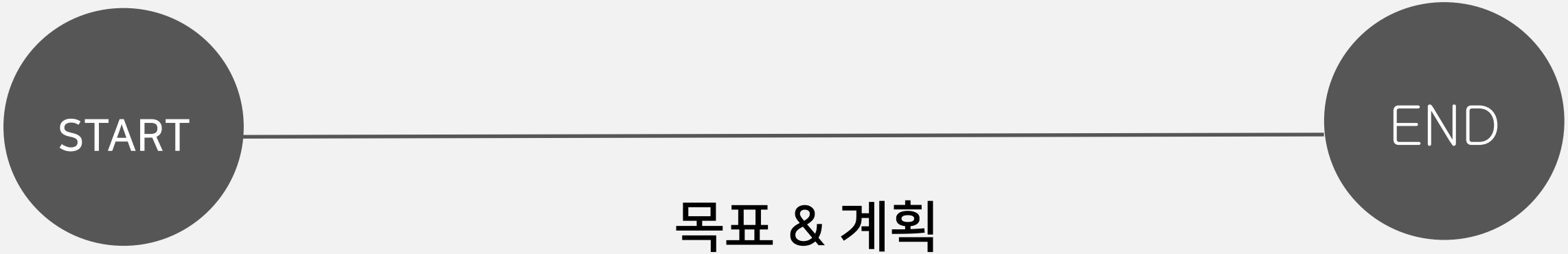
Log In

MEAN스택을 활용한 에브리타임 서비스

주소 : celebtime.herokuapp.com

05 정현성 김성일 반정원
이연재 최명진

Our Plan



- 목표 : 전국 최고의 대학교 커뮤니티 구축
- 계획 : 셀럽파이브 회사 설립 및 역할 분담

주제 선정

주제 선정 이유

- 에프리타임은 어플이다라는 인식
- 로그인이 더 간편한 방법은 없을까?

구상도(1/2)

구상도(2/2)

Client

클라이언트
인증서 요청

서버 인증서 발급

Server

전자서명 간편 로그인

Web Service

CELEBTIME

Celeb Five

Log Out

조원 편성

김성일	게시판 DB 연동 및 레지스터 담당 165 19	
이연재	백엔드 구현 및 PPT 제작 165 19	
반정원	프론트엔드 총 개발 담당 96 9	
최명진	시간표 및 메인페이지 제작 96 9	

Capston
Design
Project

1. 메인페이지 소개

.

2. 각 메뉴별 기능 소개

- 사용자 등록/로그인
- 공지사항(학식,셔틀버스)
- 시간표
- 게시판 DB 연동

3. 토큰, 인증서 발급

- 공개토큰, 비밀토큰 발급
- 인증서 발급
- 전자서명 간편 로그인

CELEBTIME

Home.component.html

Main Register Login

Log Out

```
<div class="jumbotron text-center">
  <img src="../../../assets/images/5.png" width="300"
  height="300" />
  <link
    href="https://fonts.googleapis.com/css?
    family=Stylish&display=swap&subset=korean"
    rel="stylesheet"
  />
  <link
    href="https://fonts.googleapis.com/css?
    family=Nanum+Pen+Script&display=swap"
    rel="stylesheet"
  />
  <br />
  <p class="lead">
    350만 대학생을 위한
  </p>
  <p class="row">
    국내 1위 대학생 서비스 셀럽파이브!
  </p>

  <br />
</div>
```

정적 컨텐츠 생성

CELEBTIME

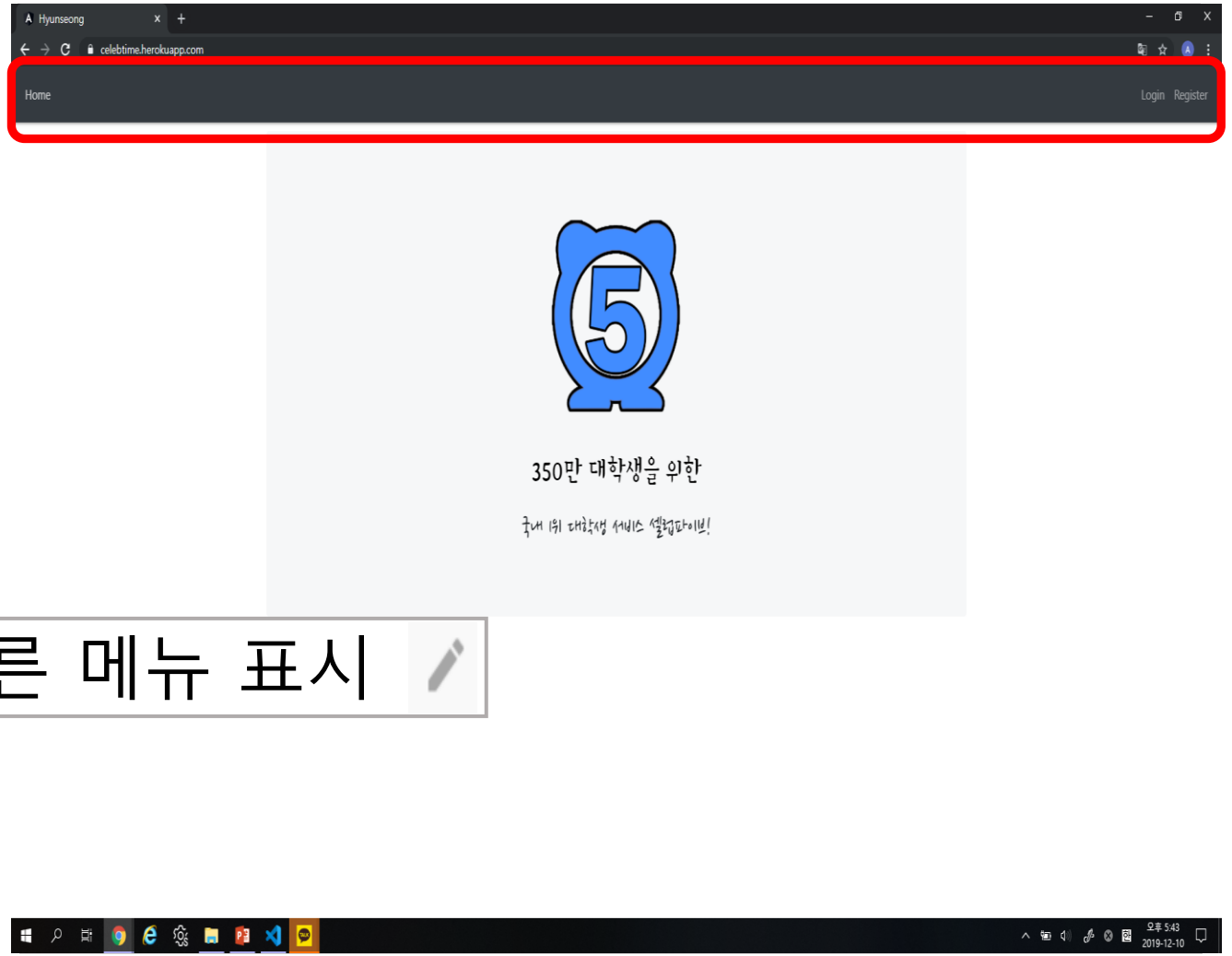
Angular.json

Main Register Login

Log Out

```
"test": {
  "builder": "@angular-devkit/
  build-angular:karma",
  "options": {
    "main": "src/test.ts",
    "polyfills": "src/polyfills.ts",
    "tsConfig": "tsconfig.spec.json",
    "karmaConfig": "karma.conf.js",
    "assets": [
      "src/favicon.ico",
      "src/assets"
    ],
```

```
  <li *ngIf="checkLoggedIn()" class="nav-item">
    <a class="nav-link" routerLink="/list">Board</a>
  </li>
  <li *ngIf="checkLoggedIn()" class="nav-item">
    <a class="nav-link" routerLink="/
    cert">인증서발급</a>
  </li>
</ul>
<ul class="navbar-nav ml-auto">
  <li *ngIf="!checkLoggedIn()" class="nav-item">
    <a class="nav-link" routerLink="/login">Login</a>
  </li>
  <li *ngIf="!checkLoggedIn()" class="nav-item">
    <a class="nav-link" routerLink="/
    register">Register</a>
  </li>
  <li *ngIf="checkLoggedIn
    <a class="nav-link" (c
    href="#">Logout</a>
  </li>
  <li *ngIf="checkLoggedIn()" class="nav-item">
    <a class="nav-link" (click)="onDeleteCertClick()
    " href="#"
      >인증서삭제</a
```



로그인 여부에 따른 메뉴 표시

```
export class RegisterComponent implements OnInit {
  name: string;
  email: string;
  username: string;
  password: string;
  phone: number;

  constructor(
    private validateService: ValidateService,
    private flashMessage: FlashMessagesService,
    private authService: AuthService,
    private router: Router
  ) {}

  ngOnInit() {}

  onRegisterSubmit() {
    // UI 에서 입력한 사용자 등록정보를 이용하여 user
    객체 생성
    const user = {
```

5개의 사용자 입력정보

```
<form
  (ngSubmit)="onRegisterSubmit()"
  style="padding-top:10px; padding-left:10px;
  padding-right:10px;
>
```

[(ngModel)] ← 2-way data-binding

Validate.service.ts

```
validateEmail(email) {
  var re = /^(([^<>()[\]\\.,;:\s@\"]+(\.[^<>()[\]\\.,;
  :\s@\"]+)*)|(\".+\"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.
  [0-9]{1,3}\.[0-9]{1,3}\]|((
  [a-zA-Z]{2,}))$/;
  return re.test(email);
}
```

```
validateRegister(user) {
  if (
    user.name == undefined ||
    user.email == undefined ||
    user.username == undefined ||
    user.password == undefined ||
    user.phone == undefined
  ) {
    return false;
  } else {
    return true;
  }
}
```

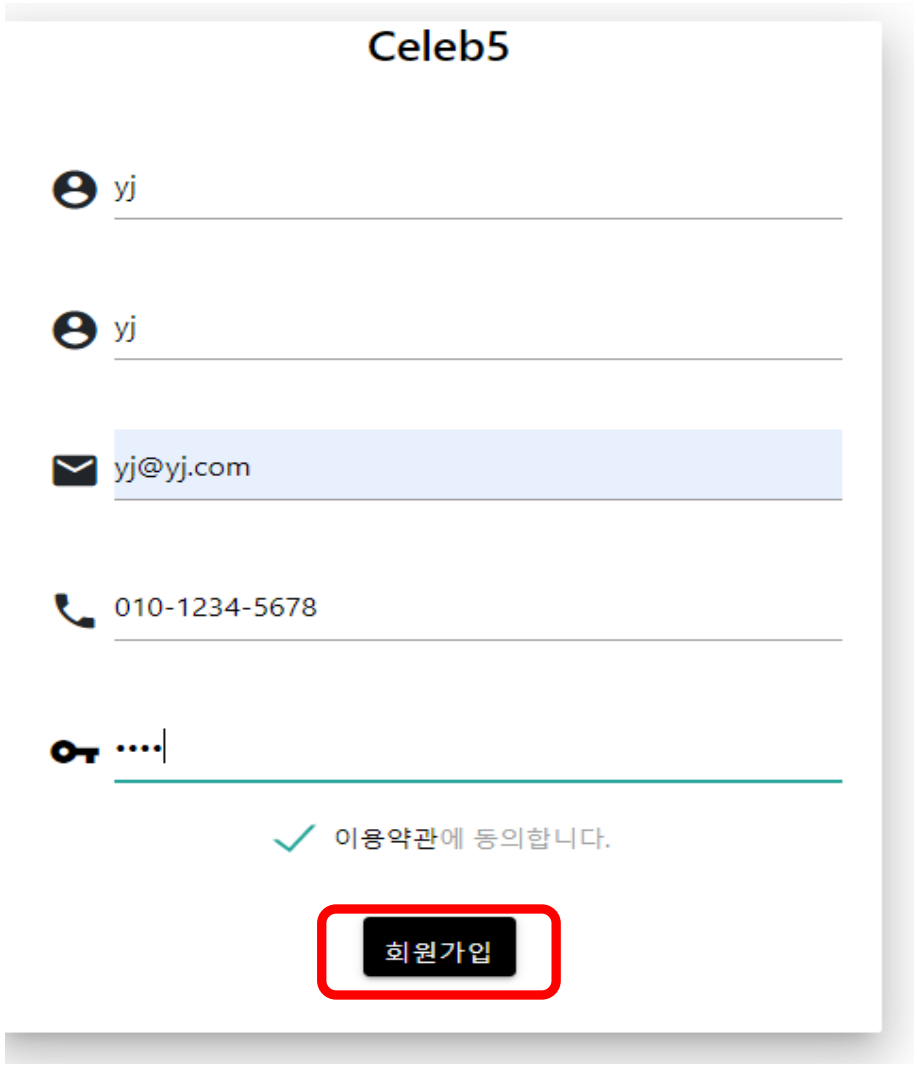
사용자 입력값 검증 추가

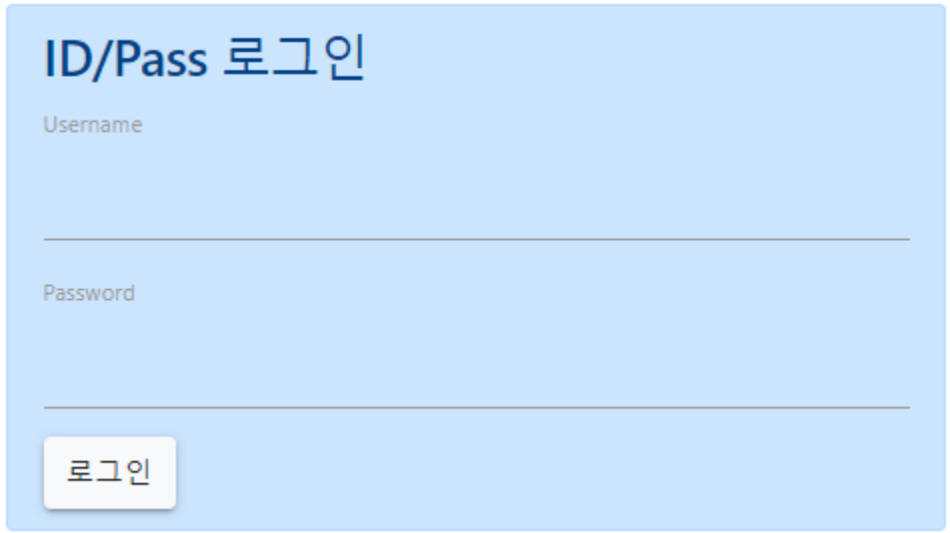
CELEBTIME

사용자 등록 성공

Log Out

Main Register Login





```
onLoginSubmit() {
  const login = {
    username: this.username,
    password: this.password
  };

  this.authService.authenticateUser(login).subscribe
  (data => {
    if (data.success) {
      // console.log(data);
      this.authService.storeUserData(data.ptoken,
      data.stoken, data.userNoPW);
      this.flashMessage.show("로그인 성공", {
        cssClass: "alert-success",
        timeout: 5000
      });
      this.router.navigate(["/"]);
    } else {
      // console.log(data);
      this.flashMessage.show(data.msg, {
        cssClass: "alert-danger",
        timeout: 5000
      });
```

```
storeUserData(ptoken, stoken, userNoPW) {
  this.ptoken = ptoken;
  this.stoken = stoken;
  this.userNoPW = userNoPW;
  localStorage.setItem("ptoken", ptoken);
  localStorage.setItem("stoken", stoken);
  localStorage.setItem("user", JSON.stringify(userNoPW)
  );
}
```

localStorage에 저장

JSON 형태 저장

로그인 성공

350만 대학생을 위한

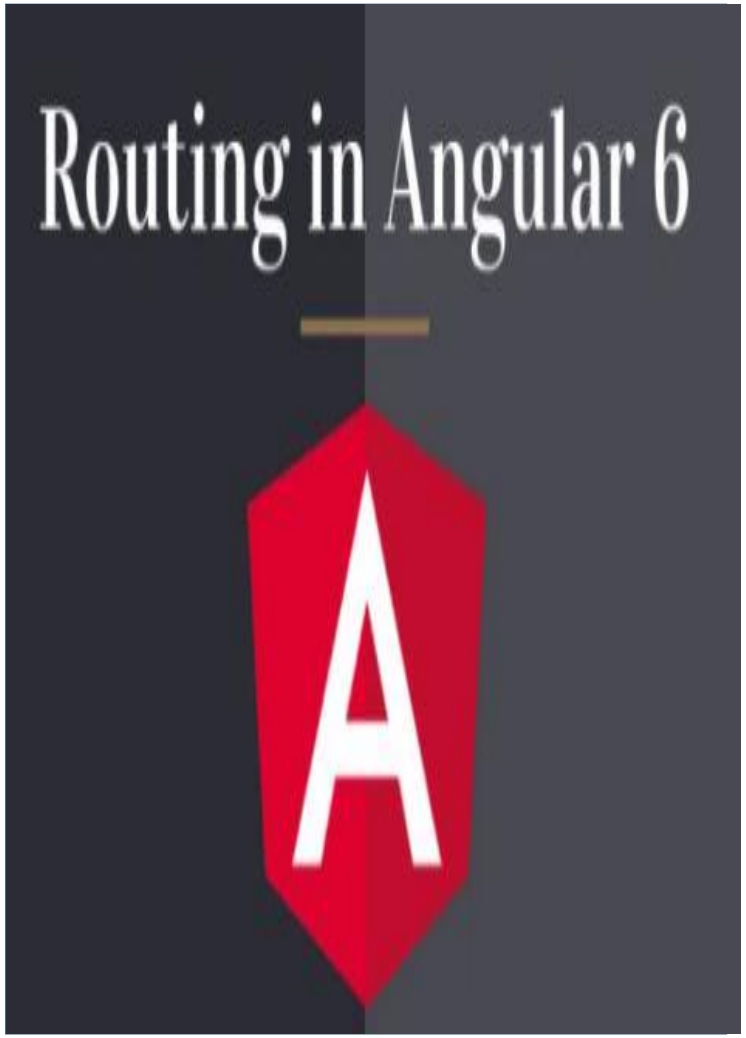
국내 1위 대학생 서비스 셀럽파이브!

Angular Routing?

페이지 이동에도 페이지 리프레쉬가 없고, url에 따라 다른 페이지 템플릿을 일정한 영역에 불러와 주는 역할

구현방법

1. app-routing.module.ts에 컴포넌트 등록
2. Angular Routing을 이용하기 위해 RouterModule 등록



CELEBTIME

공지사항

Log Out

Notice Timetable Board Dashboard

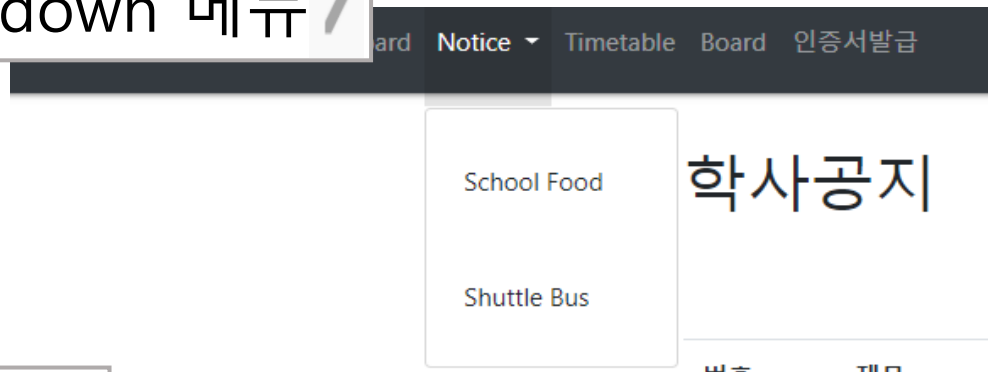
CelebFive

학사공지

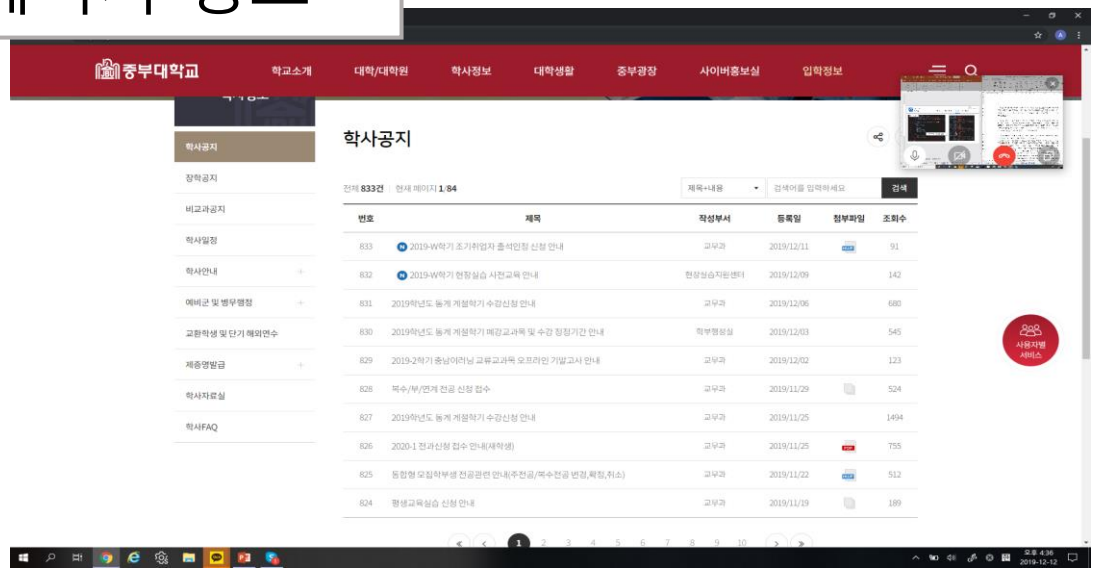
번호	제목	작성자	날짜	조회수
7	2019-2학기 캡스톤디자인 사전 설명회	현장실습지원센터	2019-09-25	768
6	2019-2 조기졸업 신청 접수 안내	교무과	2019-09-10	1182
5	2019-2학기 등록금 납부 안내(추가등록)	총무과(경리)	2019-09-09	1602
4	2019-2학기 폐강과목 안내	학부행정실	2019-08-30	1766
3	2019-2학기 현장실습 사전교육 안내	현장실습지원센터	2019-08-22	571
2	2019-2 추가개설 및 폐강 교과목 공지	교무과	2019-08-20	1467
1	2019-2학기 전과 승인 공지	교무과	2019-08-19	789

Previous 1 2 3 Next

Bootstrap dropdown 메뉴



학교홈페이지 링크



CELEBTIME
학식

Log Out

Notice Timetable Board Dashboard

고양캠퍼스 식단

아코디언을 활용한 식단

한식

일품1

일품2

고양캠퍼스 식단

한식


▶ 월요일


▶ 화요일


▶ 수요일


▶ 목요일


▶ 금요일

일품1

일품2

스쿨버스

경기/서울/충청캠퍼스 노선

◎ 등교

지 역	노선명	출발 시간		탑승금액	비고
경기/서울 => 고양 캠퍼스	부천/부평	7:00	부천역 남	₩2,000	사진보기
		7:20	인천 부평 북부역 11번출구 나와 80	₩2,000	사진보기
	파주	8:00	버거킹 금촌역점 건너편		
		8:20	내유동 내유초등학교정문 건너편 ㅂ		
		8:21	가장동 신일천막 앞 버스승강장 앞	₩2,000	
	김포	7:30	김포시청 정문 쌍용자동차 앞	₩2,000	사진보기
		7:35	사우동 내고향 숯불갈비 앞 버스승	₩2,000	

부천노선(부천역)왕복(소요 시간:1시간00분,38km) ×

* 부천 남부역2번출구 그레이스 쇼핑센타 안경마을 앞(경기 부천시 심곡본동 558)

Modal을 활용한 셔틀버스

Sample 불러오기 시간표 생성

Table을 활용한 시간표

Time	Monday	Tuesday	Wenduesday	Thursday	Friday
1				프로그래밍	
2		웹 보안		프로그래밍	해킹바이러스
3	토익 초급	웹 보안		프로그래밍	해킹바이러스
4	토익 초급				
5		해킹바이러스			웹 보안
6	암호학 응용	해킹바이러스			웹 보안
7	암호학 응용				정보보호관리
8	암호학 응용				정보보호관리
9					정보보호관리

board.js

```
d.js > boardSchema
1   const mongoose = require("mongoose");
2   const config = require("../config/database");
3
4   const boardSchema = mongoose.Schema({
5     number: {
6       type: Number
7     },
8     content: {
9       type: String,
10      required: true
11    },
12    date: {
13      type: String,
14      required: true
15    }
16  });
```

1. 필요한 스키마 생성

```
<input type="text"
class="form-control"
placeholder="내용"
aria-label="Recipient's username"
    aria-describedby="button-addon2"
    style="height: 50px;" [(ngModel)]
    ="content" name="name">
<input type="text"
class="form-control"
placeholder="순번"
aria-label="Recipient's username"
    aria-describedby="button-addon2"
    style="height: 50px;" [(ngModel)]
    ="number" name="name">
<input type="text"
class="form-control"
placeholder="날짜"
aria-label="Recipient's username"
    aria-describedby="button-addon2"
    style="height: 50px;" [(ngModel)]
    ="date" name="date">
```

```
if (!this.validateService.validateAddBoard(data))
{
  this.flashMessage.show
  ("모든 필드들을 채워주세요", {
    cssClass: "alert-danger",
    timeout: 3000
  });
  console.log("모든필드
  return false;
}

this.authService.AddtoBoard(data).subscribe(data
=> {
  if (data.success) {
    this.flashMessage.show("등록되었습니다.", {
      cssClass: "alert-success",
      timeout: 3000
    });
    this.router.navigate(["/"]);
```

2. 작성한 글을 검증

list.component.ts

```
router.post("/list/test1", (req, res, next) => {
  let board = new Board({
    content: req.body.content,
    date: req.body.date,
    number: req.body.number
  });
  Board.addBoard(board, (err, data) => {
    if (err) {
      res.json({ success: false, msg: "게시판 등록
    }
  });
});
```

```
Board1.addBoard = function(board, callback) {
  board.save(callback);
};
```

3. 게시판 DB 저장

```
ngOnInit() {
  this.authService.getBoard().subscribe(boards => {
    this.boards = boards;
  });
}
```

```
getBoard(): Observable<any> {
  const listUrl = this.prepEndpoint("users/list/
  test2");
  return this.http.get(listUrl, httpOptions);
}
```

```
router.post("/list/test1", (req, res, next) => {
  let board = new Board({
    content: req.body.content,
    date: req.body.date,
    number: req.body.number
  });
```

```
<div style="width: 1000px; margin: 0 auto; height: 100%;" id="content">
  <table class="table table-hover">
    <tbody>
      <tr *ngFor="let board of boards">
        <th>{{board.number}}</th>
        <td style="cursor:pointer;" onClick="location.href='#'">
          <strong>{{ board.content }}</strong>
          <br>
          <a class="small text-muted">37분 전</a>
          <a class="small text-muted">조회수 9</a>
        </td>
        <td>{{board.date}}</td>
        <button class="btn btn-primary" (click)="deleteMongoBoard(board)">삭제</button>
      </tr>
    </tbody>
  </table>
```

3. 게시판 DB 출력

생존신고

6

2019-12-12

등록

Hot Music

5	현성	2019-11-30	삭제

« 1 2 3 »

등록되었습니다.

게시물이 삭제되었습니다.

고양캠 자유게시판

내용

순번

날짜

등록

5	현성	2019-11-30	삭제
6	생존신고	2019-12-12	삭제

Key	Value
ptoken	JWT eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjp7Il9pZCI6IjV...
stoken	JWT eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjoiZXIKaGJHY...
user	{"name":"연재","username":"연재","email":"jhs451342@naver.com"}

미완성

공개토큰

JWT eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjp7Il9pZCI6IjVkZWUxMjJkNmQwZDQ5NWIwOGFlMzA5NyIsIm5hbWUiOiJ5aiIsImVtYWlsIjoieWpAeWouY29tIiwidXNlcm5hbWUiOiJ5aiIsInBhc3N3b3JkIjoiJDJhJDEwJFprRDZSc0l1WWZuN0xlQkRheEIaU3VyTHFaZzZIaFFGRmJuc3NiLzdWcmcxU2g4emQ3cWwyIiwicGhvbmUiOjMwLCJfX3YiOjB9LCJpYXQiOjE1NzYwOTU4ODgsImV4cCI6MTU3NjcwMDY4OH0.2fK55EkTItrX_SNeUbNA2Rlw_pdZDCEmjcaBXD6y62w

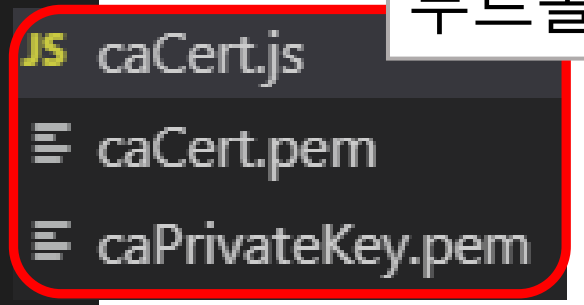
비밀토큰

JWT eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjoiZXIKaGJHY2IPaUpJVXpJMU5pSXNJbll1Y0NJNkIrcFhWQ0o5LmV5SmtZWFJoSWpwN0lsOXBaQ0k2SWpWa1pXVXhNakprTm1Rd1pEUTVOV0l3T0dGbE16QTVOeUlzSW01aGJXVWlPaUo1YWlJc0ltVnRZV2xzSWpvaWVXcEFIV291WTI5dElpd2lkWE5sY201aGJXVWlPaUo1YWlJc0luQmhjM04zYjNKa0lqb2IKREpoSkRFd0pGcHJSRFpTYzBJMVdXWnVOMHhsUWtSaGVFbGFVM1Z5VEhGYVp6WmxhRkZHUm1KdWM2NWkvZHVp6WmxhRkZHUm1KdWMzTmlMemRXY21eFUyZzRIbVEzY1d3eUIpd2IjR2h2Ym1VaU9qTXdMQ0pmWDNZaU9qQjlMQ0pwWVhRaU9qRTFOelkwTURVNE9EZ3NJbVY0Y0NJNk1UVTNOamN3TURZNE9IMC4yZks1NUVrVEI0clhfU05IVWJOQTJSbHdfcGRaRENFbWpjYUJYRDZ5NjJ3In0.B-N8BCYYq6OgV6K6wCbt3c-UtkUcAuxW_Uo4P8Aafk0

```
var forge = require("node-forge");
var fs = require("fs");
var pki = forge.pki; // 1. CA 인증서 생성
// generate a keypair and create an X.509v3 certificate
var caKeys = pki.rsa.generateKeyPair(2048);
var caCert = pki.createCertificate();
// CA 개인키 파일 저장
console.log(pki.privateKeyToPem(caKeys.privateKey));
fs.writeFileSync("caPrivateKey.pem", pki.privateKeyToPem
(caKeys.privateKey));
console.log("CA개인키 저장 - caPrivateKey.pem \n");
caCert.publicKey = caKeys.publicKey;
caCert.serialNumber = "01";
caCert.validity.notBefore = new Date();
caCert.validity.notAfter = new Date();
caCert.validity.notAfter.setFullYear(
  caCert.validity.notBefore.getFullYear() + 1
);
```

caCert.js

루트폴더 저장



```html
<h2 class="page-header">인증서 발급</h2>
<form (ngSubmit)="onCertRequest()">
  <div class="form-group">
    <label>
      Common Name (이름)
       - 이름은 username과 같아야 하며 자동 입력됩니다.
    </label>
    <input
      type="text"
      [(ngModel)]="common"
      name="common"
      class="form-control"
      disabled
    />
```

Cert.component.html

Cert.component.ts

```ts
onCertRequest() {
  const request = {
    country: this.country,
    state: this.state,
    locality: this.locality,
    organization: this.organization,
    orgUnit: this.orgUnit,
    common: this.common
  };
  this.authService.certRequest(request).subscribe(data
  => {
    if (data.success) {
      this.authService.storeCert(data.cert,
      data.caCert);
      console.log(data.cert);
      this.flashMessage.show
      ("인증서가 발급되었습니다.", {
        cssClass: "alert-success",
        timeout: 3000
      });
      this.router.navigate(["dashboard"])
    } else {
      this.flashMessage.show("인증서 발급 실패", {
        cssClass: "alert-danger"
```

인증서 저장, dashboard 이동

```
certRequest(request): Observable<any> {
  // 키쌍 생성
  let keyPair = pki.rsa.generateKeyPair(2048);
  let publicKey = keyPair.publicKey;
  let privateKey = keyPair.privateKey;
  let publicKeyPem = pki.publicKeyToPem(publicKey);
  let privateKeyPem = pki.privateKeyToPem(privateKey);
  // 개인키 저장
  localStorage.setItem("privateKey", privateKeyPem); //
   인증서 발급 요청 req 생성
  const req = {
    country: request.country,
    state: request.state,
    locality: request.locality,
    organization: request.organization,
    orgUnit: request.orgUnit,
    common: request.common,
    publicKey: publicKeyPem
  };
  const certUrl = this.prepEndpoint("users/cert");
  return this.http.post(certUrl, req, httpOptions);
}
storeCert(cert, caCert) {
  localStorage.setItem("cert", cert);

  localStorage.setItem("caCert", caCert);
}
```

Auth.service.ts

- 키쌍 생성
- Pem 형식 변환
- 로컬스토리지에 개인키 저장, req 생성
- users/cert에 post 형식으로 req를 전송, 서버의 응답
- 인증서를 로컬스토리지에 저장

```
const forge = require("node-forge");
const fs = require("fs");
const pki = forge.pki;
const caCertPem = fs.readFileSync("caCert.pem", "utf8");
const caPrivateKeyPem = fs.readFileSync
("caPrivateKey.pem", "utf8");
const caCert = pki.certificateFromPem(caCertPem);
const caPrivateKey = pki.privateKeyFromPem
(caPrivateKeyPem);
```

```
cert.sign(caPrivateKey);
return res.json({
  success: true,
  cert: pki.certificateToPem(cert),
  caCert: caCertPem
});
```

인증서 발급, JSON 형식 리턴

```
var userAttrs = [
  {
    shortName: "CN",
    value: req.body.common
  },
  {
    shortName: "C",
    value: req.body.country
  },
  {
    shortName: "ST",
    value: req.body.state
  },
  {
    shortName: "L",
    value: req.body.locality
  },
  {
    shortName: "O",
    value: req.body.organization
  },
  {
    shortName: "OU",
    value: req.body.orgUnit
  }
(userAttrs);
```

사용자 정보 설정

인증서 생성

```
var caAttrs = [
  {
    shortName: "CN",
    value: caCert.subject.getField("CN").value
  },
  {
    shortName: "C",
    value: caCert.subject.getField("C").value
  },
  {
    shortName: "ST",
    value: caCert.subject.getField("ST").value
  },
  {
    shortName: "L",
    value: caCert.subject.getField("L").value
  },
  {
    shortName: "O",
    value: caCert.subject.getField("O").value
  },
  {
    shortName: "OU",
    value: caCert.subject.getField("OU").value
  }
];
cert.setIssuer(caA
```

발급자 정보 설정

인증서 발급

Common Name (이름) - 이름은 username과 같

yj

Organizational Unit Name (부서)

Information Security

Organization Name (기관)

Joongbu Uni

Locality Name (도시)

Goyang-si

State or Province Name (지역)

Gyeong-gi

Country (국가)

KR

인증서 발급 요청

사용자 인증서

-----BEGIN CERTIFICATE-----
MIIEHzCCAwegAwIBAgIBATANBgkqhkiG9w0BAQUFADCBkTEYMBYGA1UEAxMPQnlv
dW5nY2hlb24gTGVlMQswCQYDVQQGEwJLUjEUMBIGA1UECBMLR3llb25nZ2ktZG8x
EjAQBgNVBAcTCUdveWFuZy1zaTEWMBQGA1UEChMNSm9vbmdidSBVbml2LjEmMCQG
A1UECxMdRGVwdC4gb2YgSW5mb3JtYXRpb24gU2VjdXJpdHkwHhcNMTkxMjExMTkz
NjQyWhcNMjAxMjExMTkzNjQyWjBeMQswCQYDVQQDEwJ5ajELMAkGA1UEBhMCS1Ix
DzANBgNVBAgTBmphbXNpbDEOMAwGA1UEBxMFc2VvdWwxDzANBgNVBAoTBmpvb2di
dTEQMA4GA1UECxMHam9vbmdidTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoC

서버 인증서

-----BEGIN CERTIFICATE-----
MIIEUzCCAzugAwIBAgIBATANBgkqhkiG9w0BAQUFADCBkTEYMBYGA1UEAxMPQnlv
dW5nY2hlb24gTGVlMQswCQYDVQQGEwJLUjEUMBIGA1UECBMLR3llb25nZ2ktZG8x
EjAQBgNVBAcTCUdveWFuZy1zaTEWMBQGA1UEChMNSm9vbmdidSBVbml2LjEmMCQG
A1UECxMdRGVwdC4gb2YgSW5mb3JtYXRpb24gU2VjdXJpdHkwHhcNMTkxMjA4MTUz
MjE1WhcNMjAxMjA4MTUzMjE1WjCBkTEYMBYGA1UEAxMPQnlvdW5nY2hlb24gTGVl
MQswCQYDVQQGEwJLUjEUMBIGA1UECBMLR3llb25nZ2ktZG8xEjAQBgNVBAcTCUdv
eWFuZy1zaTEWMBQGA1UEChMNSm9vbmdidSBVbml2LjEmMCQGA1UECxMdRGVwdC4g

CELEBTIME

인증서 삭제

Log Out

Notice Timetable Board Dashboard

Logout 인증서삭제

인증서 삭제 완료. 다시 로그인하세요.

사용자 인증서

서버 인증서

ID/Pass 로그인

Username

Password

로그인

전자서명 간편 로그인

전자서명 간편로그인은 서버로부터 인증서를 발급받은 경우에만 사용할 수 있습니다. 아이디, 패스워드를 입력할 필요 없이 간편하게 로그인 가능합니다.

전자서명 간편로그인

```
// AuthenticateSig 전자서명 관련 로그인
router.post("/authenticateSig", (req, res, next) => {
  const username = req.body.username;
  User.getUserByUsername(username, (err, user) => {
    if (err) throw err;
    if (!user) {
      return res.json({
        success: false,
        msg:
        "User not found! 등록된 사용자가 없습니다..."
      });
    }
    const currentTime = req.body.currentTime;
    const signatureHex = req.body.signatureHex;
    const certPem = req.body.certPem;
    const cert = pki.certificateFromPem(certPem);
    const publicKey = cert.publicKey;
    const signature = forge.util.hexToBytes(signatureHex)
    ;
    const common = cert.subject.getField("CN").value;
    const currentTime1 = new Date().getTime();
    const diffTime = currentTime1 - currentTime;
    let md = forge.md.sha1.create();
    md.update(username, "utf8");
    md.update(currentTime, "utf8");
    let verified1 = publicKey.verify(md.digest().byte
     signature);
    let verified2 = caCert.verify(cert);
    let verified3;
    if (diffTime < 1000000) verified3 = true;
    let verified4;
    if (username == common) verified4 = true;
    if (
      verified1 == true &&
      verified2 == true &&
      verified3 == true &&
      verified4 == true
    ) {
```

DB에서 정보 읽어옴

인증서 객체, 전자서명값 등 생성

공개/비밀토큰 생성

```
const ptoken =
  "JWT " +
  jwt.sign({ data: user }, config.secret, {
    expiresIn: 604800 // 1 week, 유효기간: 1주일
  });
const stoken =
  "JWT " +
  jwt.sign({ data: ptoken }, config.secret, {
    noTimestamp: true // 발급시점,
     유효기간 삭제. 토큰의 유효성은 ptoken에서만 검
    증 예정
  });
```

4가지 검증
1. 전자서명이 유효?
2. 사용자 인증서가 유효?
3. 시간차이 확인
4. 요청자가 인증서 소유자인지?

CELEBTIME

로그인 성공

Log Out

Main Register Login

전자서명 간편 로그인

전자서명 간편로그인은 서버로부터 인증서를 발급받은 경우에만 사용할 수 있습니다. 아이디, 패스워드를 입력할 필요 없이 간편하게 로그인 가능합니다.

전자서명 간편로그인

클릭하여 간편 로그인

아이디, 패스워드 필요X
전자서명 생성, 서버에 제시

전자서명으로 로그인

로그인 성공

닉네임: yj

UserInfo

{"name":"yj","username":"yj","email":"yj@yj.com"}

공개,비밀토큰 생성

공개토큰

JWT eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjp7Il9pZCI6IjVkZWUxMjJkNmQwZDQ5NWIwOGFlMzA5NyIsIm5hbWUiOiJ5aiIsImVtYWlsIjoieWpAeWouY29tIiwidXNlcm5hbWUiOiJ5aiIsInBhc3N3b3JkIjoiJDJhJDEwJFprRDZSc0I1WWZuN0xIQkRheElaU3VyTHFaZzZlaFFGRmJuc3NiLzdWcmcxU2g4emQ3cWwyIiwicGhvbmUiOjMwLCJfX3YiOjB9LCJpYXQiOjE1NzYxMzM4NzYsImV4cCI6MTU3NjczODY3Nn0.GoisXHHPj7pzqBHjQHVthTif-IzyIHjsCspLi36jmNg

비밀토큰

JWT eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjoiSldUwqBleUpoYkdjaU9pSklVekkxTmIJc0IuUjVjQ0k2SWtwWFZDSjkuZXIKa1IYUmhJanA3SWw5cFpDSTZJaIZrWIdVeE1qSmtObVF3WkRRNU5XSXdPR0ZsTXpBNU55SXNJbTVoYIdVaU9pSjVhaUlzSW1WdFIXbHNJam9pZVdwQWVXb3VZMjI0SWI3aWRYTmxjbTVoYIdVaU9pSjVhaUlzSW5CaGMzTjNiM0prSWpvaUpESmhKREV3SkZwclJEWINjMEkxV1dadU4weGxRa1JoZUVsYVUzVnlUSEZhWnpabGFGRkdSbUp1YzNOaUx6ZFdjbWN4VTJnNGVtUTNjV3d5SWI3aWNHaHZibVVpT2pNd0xDSmZYM1IpT2pCOUxDSnBZWFFpT2pFMU56WXhNek00TnpZc0ltVjRjQ0k2TVRVM05qY3pPRFkzTm4wLkdvaXNYSEhQajdwekJIalFIVnRoVGlmLUl6eUhqc0NwTGkzNmpOZw

사용자 인증서

-----BEGIN CERTIFICATE-----
MIIEODCCAyCgAwIBAgIBATANBgkqhkiG9w0BAQUFADCBkTEYMBYGA1UEAxMPQnlv
dW5nY2hlb24gTGVlMQswCQYDVQQGEwJLUjEUMBIGA1UECBMLR3Ilb25nZ2ktZG8x
EjAQBgNVBAcTCUdveWFuZy1zaTEWMBQGA1UEChMNSm9vbmdidSBVbmI2LjEmMCQG
b2YgSW5mb3JtYXRpb24gU2VjdXJpdHkwHhcNMTkxMjEyMDY1
MDY1MTQwWjB3MQswCQYDVQQDEwJ5ajELMAkGA1UEBhMCS1Ix
W9uZy1naTESMBAGA1UEBxMJR295YW5nLXNpMRQwEgYDVQQK
EwtKb2JUdMBsGA1UECxMUSW5mb3JtYXRpb24gU2VjdXJpdHkwggEi

완성 후 성능분석

학생들이 필요로 하는 정보를 알기 쉽게 배치하였습니다.

게시판 글의 등록 및 삭제를 구현하였습니다.

인증서 발급 기능을 통하여 전자서명 간편 로그인을 구현하였습니다.

향후 계획

난수화 토큰인증을 구현할 것입니다.

게시판 글의 수정을 연구할 계획입니다.

시간표 불러오기 기능을 통해 친구들의 시간표까지 확인할 수 있도록 할 것입니다.

Q&A

SECRET BOARD

Write your post here!

Anonymous 13 seconds ago

Hot Posts

More

Secrete Board 165 19

Secrete Board 96 9

FINISH
OUR
PRESENTATION
THANK YOU