캡스톤 제안서(계획서)

프로젝트	제목	졸업가능?		
		졸업요건 조회 서비스 웹사이트		
팀장	팀명	졸업가능?		
	성명	배성규	학번	201758101
	연락처	010-4099-8259		
	E-MAIL	bae1374@hs.ac.kr		
구분	성명	학번	E-MAIL	연락처(H.P)
팀원 인적사항	박수빈	201858054	psb0320@hs.ac.kr	010-2473-9211
	김지윤	201858103	tkstpqpfldk9@hs.ac.kr	010-5498-3867
	김민석	201858013	zas7186@hs.ac.kr	010-3769-4312
지도교수				

본인과 팀원은 2021학년도 1학기 캡스톤1 과목의 설계 프로젝트에 대한 캡스톤 계획서를 다음과 같이 제출합니다.

2021년 03월 09일

팀 장 : 배성규 (서명)

한신대학교 컴퓨터공학부

목 차

1. 문제 및 목적 2

2. 관련 연구 2

3. 주요기능 및 기대효과 6

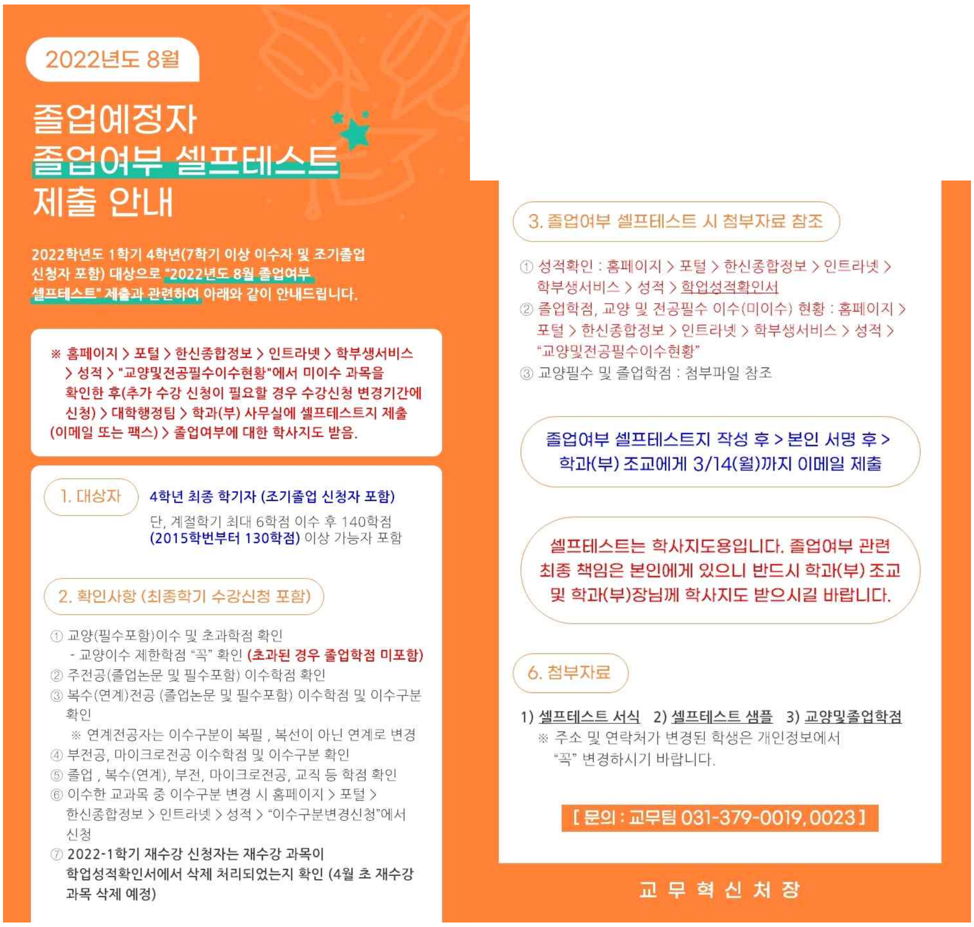
4. 개발 환경 8

5. 위험 요소 9

6. 일정 계획 9

7. 참고 문헌 10

1. 문제 및 목적



[한신대학교 졸업예정자 졸업여주 셀프테스트 제출 안내 (이미지 출처 : 한신대학교 학사공지)] [2]


졸업 예비자의 졸업사정 SELF TEST [샘플]

■ 개인별 기록사항

■ 졸업학점 이수

[졸업 예비자의 졸업사정 SELF TEST 샘플 (이미지 출처 : 한신대학교 학사공지)] [2]

현재 우리 학교는 졸업 예정자들을 대상으로 졸업사정 셀프 테스트를 시행하고 있다. 기존 진행되는 졸업사정 셀프 테스트는 졸업 예정자인 학우들만 결과를 확인할 수 있고, 직접 학과 사무실에 셀프 테스트 양식을 작성하여 제출해야 하는 불편함이 있다. 또한 졸업 예정자가 아닌 학우들은 학과/학번별로 졸업요건이 상이하기 때문에 졸업요건을 정확하게 파악하기 어렵다. 졸업사정 셀프 테스트의 결과도 조교가 직접 체크해야 하기 때문에 결과지를 받는 시간이 오래 걸린다.

이를 해결하기 위해 졸업요건 조회 서비스 웹 사이트를 구현하여 졸업요건에 충족했는지를 알 수 있도록 하고, 부족한 졸업요건이 있다면 어떤 부분인지를 알려주어 졸업요건을 만족할 수 있도록 돕는다.

졸업요건 조회 서비스의 목적으로는 학우들이 웹 사이트에서 학업성적확인서 PDF만 올려 빠르고 간편하게 본인의 졸업요건을 파악하고 부족한 부분을 보완해 주도록 도와준다. 또한 조교들은 졸업 요건을 확인해야 하는 업무를 하지 않아도 되어서 부담을 덜어주고 다른 업무에 더 집중할 수 있다.

2. 관련 연구

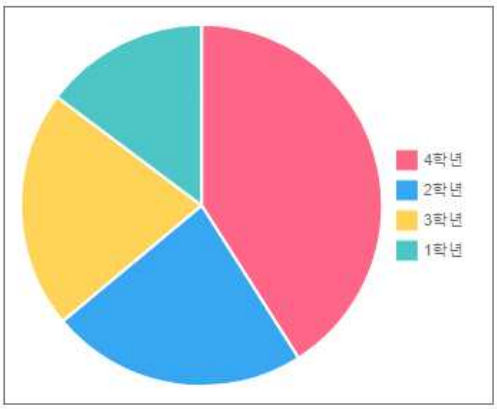
1) 졸업요건 서비스 서베이 : 기존 상황과 전망

2022년 3월 29일 ~ 4월 1일까지 4일간 한신대학교 학우들을 대상으로 설문조사를 진행하였고, 총 61명의 응답을 받았다.

61명 응답 응답 별 결과보기 > 필터

요약

1. 당신의 학년은?

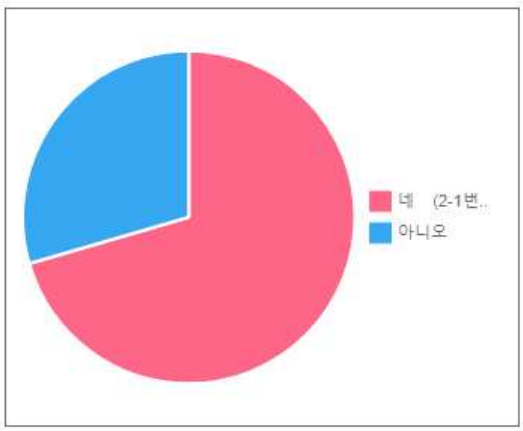


숨기기 취소 | 정렬 초기화 | 조합 | 차트 편집

응답	응답수	
4학년	25	41%
2학년	14	23%
3학년	13	21.3%
1학년	9	14.8%

학년 응답에서 4학년이 가장 많았고 뒤이어 2학년, 3학년, 1학년 순으로 높은 응답률을 보였다.

2. 당신의 학번/학과의 졸업요건 정확하게 알고 있습니까?

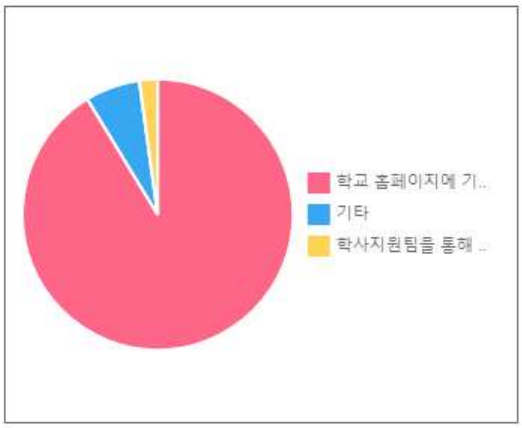


숨기기 취소 | 정렬 초기화 | 조합 | 차트 편집

응답	응답수	
네 (2-1번, 2-2번으로)	43	70.5%
아니오	18	29.5%

응답한 학우 중 29.5%의 학우들이 본인의 학번/학과의 졸업요건을 알지 못하고 있었다.

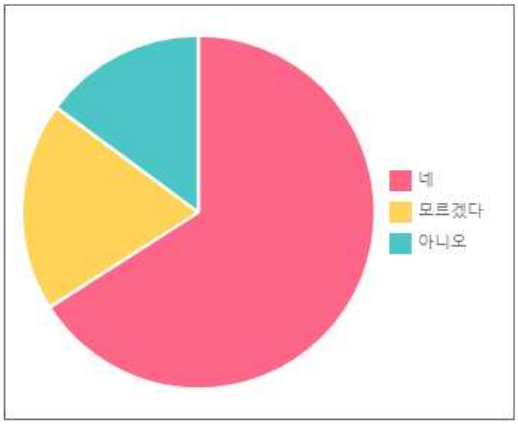
2-1. 어떤 방법을 통해 졸업요건을 파악하였습니까?



숨기기 취소 | 정렬 초기화 | 조합 | 차트 편집

응답	응답수	
학교 홈페이지에 기재된 정보를 바탕으로 직접 파악	42	68.9%
응답 없음	15	24.6%
기타	3	4.9%
학사지원팀을 통해 파악	1	1.6%

2-2. 졸업요건을 파악할 때 불편함이 있었습니까?



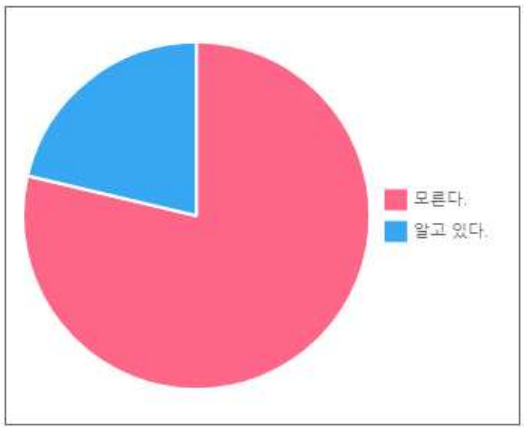
숨기기 취소 | 정렬 초기화 | 조합 | 차트 편집

응답	응답수	
네	31	50.8%
응답 없음	14	23%
모르겠다	9	14.8%
아니오	7	11.5%

이전 질문에서 본인의 학번과 학과에 맞는 졸업요건을 정확하게 파악하고 있는 학우들을 대상으로 두 가지의 설문을 추가로 진행했다. 따라서 '응답 없음' 항목을 무효화시켰다.

본인의 학번/학과에 맞는 졸업요건을 파악하고 있는 학우들의 대부분(68.9%)이 직접 학교 홈페이지를 통해 졸업 정보를 파악하였다. 그리고 50.8%의 학우들이 본인의 졸업요건을 파악하는 과정에서 불편함이 있었다고 응답했다.

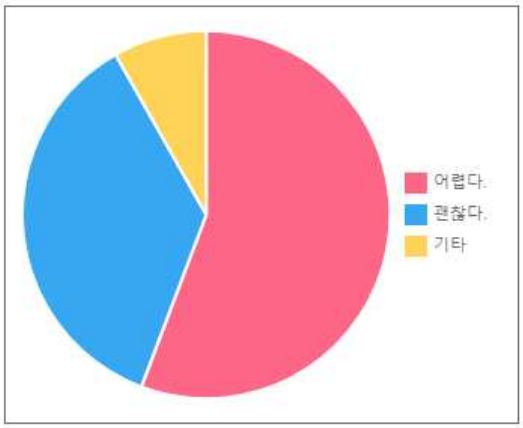
3. 교내 졸업예정자를 대상으로 '졸업여부 셀프테스트'가 진행되고 있는 것을 알고 있습니까?



숨기기 취소 | 정렬 초기화 | 조합 | 차트 편집

응답	응답수	
모른다.	48	78.7%
알고 있다.	13	21.3%

4. 졸업여부 셀프테스트 샘플입니다. 학생이 직접 샘플처럼 작성하여 학과 사무실에 이메일 또는 팩스로 방식입니다. 이 방법을 어떻게 생각하시나요?

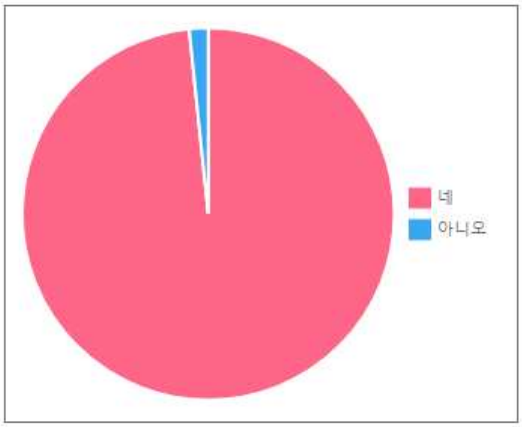


숨기기 취소 | 정렬 초기화 | 조합 | 차트 편집

응답	응답수	
어렵다.	34	55.7%
괜찮다.	22	36.1%
기타	5	8.2%

또 현재 우리 학교에서 졸업예정자를 대상으로 진행하는 졸업여부 셀프테스트를 알고 있는지에 대한 질문에 78.7%의 학우들이 모른다고 응답하였다. 그리고 현재 셀프테스트 샘플과 함께 셀프테스트에 대한 간략한 설명을 하였고, 55.7%의 학우들이 현재 셀프테스트 방식이 어렵게 느껴진다고 응답하였다.

5. 저희 '졸업가능'팀은 학업성적확인서 PDF를 업로드 하면 졸업요건 결과를 보여주는 서비스를 만들 계획입니다. 졸업요건 서비스가 있다면 이용할 의향이 있습니까?



숨기기 취소 | 정렬 초기화 | 조합 | 차트 편집

응답	응답수	
네	60	98.4%
아니오	1	1.6%

마지막으로 만들 졸업요건 서비스에 대한 간략한 설명과 함께 이용 의향을 조사한 결과 98.4%의 학우들이 이용할 예정이라고 응답했다.

2) 유사 시스템

- 타 대학교 시스템

구분		제1전공				
이수구분	영역	이수요구학점	이수학점	이수해야 할 학점	교과과정 이수요건	비고
기초교양		8	8	0		
핵심교양	인문I		3	-		
	인문II		0	3		
	소통		3	-		
	글로벌		0	3		
	창의		0	3		
	소계	15	6	9	교과과정	
자유교양		2	4	-2		
전공기초교양		6	6	0		
전공(전공선택,학부(과)기초)		48	12	36	교과과정	
일반선택		51	0	51	교과과정	일반선택 이수 인정학점 : 2학점
총계		130	36	94		

[국민대학교 이수구분별 취득학점 조회 표 (이미지 출처 : 국민대학교 국희 블로그)] [3]

취득학점현황

구분	기준	이수현황	비고
> 정규 등록 학기 수	8	6	현재학기 미포함
> 총 이수학점	130	97.0	
> 주전공	65	55.0	
> 기초교양	15	12.0	대학교필
> 교양필수	39	27.0	학부교필
> 교양선택	학과별 교육과정 참조	0.0	
> 기타		0.0	일반선택/교직 등
> 인하졸업인증		미취득(영)	

[인하대학교 취득학점현황 (이미지 출처 : 인하대 재학생 지인)]

예시로 가져온 두 대학교 모두 이수학점(전공/교양)은 표기되나 부족한 과목의 구체적인 정보를 알려주지 않아 학생이 직접 일일이 찾아야 하는 불편함이 있다.

3) 학사지원팀 인터뷰

졸업 업무를 담당하는 학사지원팀에 직접 전화와 메일을 통해 여러 차례 인터뷰를 진행하였다.

1. 졸업요건을 채우지 못해 졸업이 유예되거나 취소되는 경우가 전체 졸업예정자 중 몇 퍼센트 정도인지 궁금합니다.

2021학년도 전기(2022년 2월) 기준 졸업(예정)자 정보를 참조하시기 바랍니다.

2021학년도 전기(2022년 2월) 졸업(예정)자 정보			
졸업예정자	졸업생	졸업유예	졸업불가
1,374명	774명	91명	509명

졸업불가사유로는 1. 교양학점미달, 2. 전공최소학점미달, 3. 졸업학점미달, 4. 교양필수 미이수, 5. 졸업논문 미이수, 6. 전공필수 미이수 등이 있습니다.

2. 졸업요건에 관련된 문의 메일, 전화가 하루에 대략 몇 건 정도 있는지 궁금합니다.

관련 문의는 졸업시즌인 12월~2월(전기), 6월~8월(후기)에 집중되는 편입니다.

3. 만약 학업성적확인서를 통해 졸업 가능 여부를 확인하고 부족한 졸업요건을 알려주는 서비스가 있다면 학사지원팀에서는 이용할 의향이 있으십니까?
학사지원팀에서 서비스를 이용한다면 그렇게 생각하신 이유를 여쭤보고 싶습니다

ex) 있다. 문의 전 화량을 줄일 수 있을 것 같다.

있다. 우리 대학에서는 매학기 초 졸업예정자에 대한 "졸업여부 셀프테스트"를 시행하고 있습니다.

다만, 수기로 작성을 하고 있어 학생들이 본인의 졸업요건을 스스로 웹 상에서 체크하고 관리할 수 있다면 활용도가 높을 것 같습니다.

질문 1. 졸업요건을 채우지 못해 졸업이 유예되거나 취소되는 경우가 전체 졸업예정자 중 몇 퍼센트 정도인지 궁금합니다.

=> 전체 졸업예정자 중 37.05%의 학생들이 졸업 요건을 채우지 못해 졸업이 취소되었습니다.

질문 2. 졸업요건에 관련된 문의 메일, 전화가 하루에 대략 몇 건 정도 있는지 궁금합니다.

=> 관련 문의는 졸업시즌인 12월~2월(전기), 6월~8월(후기)에 집중되는 편입니다.

질문 3. 만약 학업성적확인서를 통해 졸업 가능 여부를 확인하고 부족한 졸업요건을 알려주는 서비스가 있다면 학사지원팀에서는 이용할 의향이 있으십니까?
학사지원팀에서 서비스를 이용한다면 그렇게 생각하신 이유를 여쭤보고 싶습니다.

=> " 있다." 우리 대학에서는 매 학기 초 졸업예정자에 대한 "졸업여부 셀프 테스트"를 시행하고 있습니다. 다만, 수기로 작성을 하고 있어 학생들이 본인의 졸업요건을 스스로 웹 상에서 체크하고 관리할 수 있다면 활용도가 높을 것 같습니다.

2) 시스템 구축 관련 기술

① Back-end

- Spring boot

스프링 부트는 JAVA를 기반으로 한 웹 어플리케이션 프레임워크이다. 스프링 부트의 특징으로 톰캣과 같은 내장 서버가 존재하기 때문에 설치와 버전 관리를 신경 쓸 필요가 없다. 또한, 기존 스프링은 객체 의존성을 관리해 줘야 하지만 스프링 부트에서는 자동으로 의존성 관리가 가능하다.

② Front-end

- React

페이스북에서 만든 자바스크립트 라이브러리로, 프론트엔드의 사용자 인터페이스와 동작을 정의하기 위해 사용된다. 싱글 페이지 애플리케이션(SPA)를 전제로 하고 있어 리렌더링이 잦은 환경에서 빠른 퍼포먼스를 내는 것이 가능하다. 또한, 기본적으로 모듈형 개발이기 때문에 생산성도 높은 라이브러리며 타 프레임워크에 간편하게 붙여서 사용하는 것이 가능하다. [4][5][6][7][8]

3. 주요기능 및 기대효과

[기능적 요구사항]

주요기능	화면	기능	상세
졸업요건 확인 기능	**학업성적확인서 PDF 업로드 페이지**	사용자는 PDF 파일 업로드 화면에서 업로드 버튼을 통해 사용자의 학업성적확인서 PDF 파일을 업로드할 수 있어야 한다.	- 학업성적확인서 PDF 파일 업로드 성공 시 결과 조회 페이지로 이동할 수 있어야 한다. - 실패 시(파일 업로드 실패 또는 PDF 파일이 아닌 파일의 업로드 또는 PDF 파일이지만 학업성적확인서가 아닌 파일의 업로드) 업로드가 실패되었음을 알 수 있어야 한다.
	결과 조회 페이지	졸업요건 만족 시 '졸업 가능!' 페이지 제공받을 수 있어야 한다	*아래 [졸업요건 검사 순서] 참고
		졸업요건 불만족 시 부족한 졸업요건 목록을 제공받을 수 있어야 한다.	
교양 추천 기능	**인기 교양 과목 조회**	사용자는 학생들이 많이 수강한 순서대로 교양 목록을 제공받을 수 있어야 한다.	
정보 공유 게시판	**게시글 조회**	사용자는 게시글 내용을 확인할 수 있어야 한다.	
		사용자는 게시글 작성자일 경우 게시글을 수정 버튼을 확인할 수 있어야 한다.	
		사용자는 게시글 작성자일 경우 게시글을 삭제 버튼을 확인할 수 있어야 한다.	
	게시글 작성	사용자는 별도의 로그인 없이 제목, 내용, 작성자, 비밀번호로만 게시글 작성할 수 있어야 한다. (비밀번호는 수정/삭제를 위함)	- 사용자는 게시글 작성 성공 시 작성한 게시글 페이지로 이동할 수 있어야 한다. - 사용자는 게시글 작성 실패 시 '비어 있는 항목이 있습니다.' 문구를 제공받을 수 있어야 하며, 이어서 작성할 수 있어야 한다.

정보 공유 게시판	**게시글 수정**	사용자는 게시글 작성 시 입력했던 비밀번호를 입력할 수 있어야 한다.	
		사용자는 수정을 위해 입력한 비밀번호가 일치하면 게시글 수정 페이지로 이동할 수 있어야 한다.	- 사용자는 수정 성공 시 수정한 게시글 페이지로 이동할 수 있어야 한다. - 사용자는 게시글 수정 실패 시 '비어 있는 항목이 있습니다.' 문구를 제공받을 수 있어야 하며, 이어서 작성할 수 있어야 한다.
		사용자는 수정을 위해 입력한 비밀번호가 일치하지 않으면 ' 비밀번호가 일치하지 않습니다.' 문구를 제공받을 수 있어야 하며, 게시글 페이지로 이동할 수 있어야 한다.	
	게시글 삭제	사용자는 게시글 작성 시 입력했던 비밀번호를 입력할 수 있어야 한다.	
		사용자는 삭제를 위해 입력한 비밀번호가 일치하여 게시글 삭제에 성공 시 게시글 목록 페이지로 이동할 수 있어야 한다.	
		사용자는 삭제를 위해 입력한 비밀번호가 일치하지 않으면 ' 비밀번호가 일치하지 않습니다.' 문구를 제공받은 후 게시글 페이지로 이동할 수 있어야 한다.	
	게시글 목록	사용자는 게시글 목록을 볼 수 있어야 한다.	
		사용자는 안내글을 확인할 수 있어야 한다.	

정보 공유 게시판	**게시글 검색**	사용자는 게시글 제목/내용으로 게시글을 검색할 수 있어야 한다.	
		검색어를 포함하는 제목/내용의 게시글이 있으면 게시글 목록에 제공받을 수 있어야 한다.	
		검색어를 포함하는 제목/내용의 게시글이 없으면 비어 있는 게시글 목록 제공받을 수 있어야 한다.	

1. 졸업요건 확인 기능

1-1. 학업성적확인서 PDF 업로드 페이지

1-1-1. 사용자는 PDF 파일 업로드 화면에서 업로드 버튼을 통해 사용자의 학업성적확인서 PDF 파일을 업로드할 수 있어야 한다.

1-1-1-1. 학업성적확인서 PDF 파일 업로드 성공 시 결과 조회 페이지로 이동할 수 있어야 한다.

1-1-1-2. 실패 시(파일 업로드 실패 또는 PDF 파일이 아닌 파일의 업로드 또는 PDF 파일이지만 학업성적확인서가

아닌 파일의 업로드) 업로드가 실패되었음을 알 수 있어야 한다.

1-2. 결과 조회 페이지

1-2-1. 졸업요건 만족 시 '졸업 가능!' 페이지 제공받을 수 있어야 한다.

1-2-2. 졸업요건 불만족 시 부족한 졸업요건 목록을 제공받을 수 있어야 한다.

[졸업요건 검사 순서]

학사지원팀으로부터 제공받은 졸업불가사유 6가지를 참고하여 졸업요건 검사 순서를 정하였다.

학사지원팀으로부터 제공받은 졸업불가사유는 아래와 같다.

1. 교양학점 미달
2. 전공 최소학점 미달
3. 졸업학점 미달
4. 교양필수 미이수
5. 졸업논문 미이수
6. 전공필수 미이수

졸업요건 검사는 아래의 순서로 이루어진다.

1. 파일 검사

1-1. 업로드된 파일이 한신대학교 학업성적확인서 PDF가 맞는지 확인한다.

```
/**
 * 파일 읽어오기 + [파일 검사]
 */
// 파일 읽어오기
String[] list = PdfRead(fileName);

// 한신대 학업성적확인서 확인
if (checkPDF(list) == 0) return 0;
```

```java
/**
 * pdf 읽어오기
 */
private static String[] PdfRead(String fileName) throws Exception {

    File source = new File(fileName);
    PDDocument pdfDoc = PDDocument.load(source);
    String text = new PDFTextStripper().getText(pdfDoc);
    String[] pdf = text.split( regex: "\n");

    return pdf;
}


/**
 * [파일 검사] 한신대학교 학업성적확인서 pdf 파일이 맞는지 검사
 */
private static Integer checkPDF(String[] list) {
    String text = Arrays.toString(list);

    if (text.contains("포털>한신종합정보>성적")) {
        return 1;
    }

    return 0;
}
```

2. 기본 정보 추출

학번, 학과, 총 취득학점, 교양 학점, 전공 학점, 비교과 이수 학기, 비교과 마일리지 등 기본적으로 필요한 정보를 PDF 파일로부터 추출한다.

```java
/**
 * 기본 정보 추출
 */
for (int i = 0; i < list.length; i++) {
    String line = list[i];

    // 학번 추출
    if (line.contains("학 번")) {
        studentId = Integer.parseInt(line.substring(4, 8));
    }

    // 학과 추출
    if (line.contains("부전공Ⅰ")) {
        String[] strings = list[i - 1].split( regex: " ");
        studentMajor = strings[2].substring(0, strings[2].length() - 1);
    }


    // 총 취득학점 추출
    if (line.contains("총 취득학점")) {
        totalCredit = Integer.parseInt(line.substring(7, line.length() - 1));
    }

    // 교양, 전공 이수학점 확인
    if (line.contains("교양: ") && line.contains("전공: ")) {
        kyCredit = Integer.parseInt(line.substring(4, 6));
        majorCredit = Integer.parseInt(line.substring(11, 13));
    }

    // 수강한 전필 과목 추출
    if (line.startsWith("전필") && !line.contains("F") && !line.contains("NP")) {
        String[] strings = line.split( regex: " ");
        requiredMajor.add(strings[2]);
    }
```

```
    // 수강한 교필 과목 추출
    if (line.startsWith("교필") && !line.contains("NP")) {
        String[] strings = line.split( regex: "\\s+");
        if (!(strings[4].contains("F"))) {
            requiredKy.add(strings[2]);
        }
    }

    // 비교과 이수 학기 카운트
    if (line.contains("학기 인정")) {
        nonSubject++;
    }

    // 마일리지 추출
    if (line.contains("마일리지")) {
        mileage = Integer.parseInt(line.substring(22, line.length() - 1));
    }

    // 영어인증자 추출
    if (line.contains("영어인증")) {
        engCertification = true;
    }

    // 모든 교양 과목 추출 (for 인재상 & 핵심역량 검사, 교양 카운트 증가)
    if ((line.startsWith("교선") || line.startsWith("교필")) && !line.contains("NP")) {
        String[] strings = line.split( regex: "\\s+");
        if (strings.length < 5) {
            allKy.add(strings[2]);
        } else if (!(strings[4].contains("F"))) {
            allKy.add(strings[2]);
        }
    }
}
```

3. 학점 검사

3-1. 총 취득학점이 졸업학점을 넘는지 검사한다.

3-1-1. 이수학점이 졸업학점을 넘었다면 통과한다.

3-1-2. 이수학점이 졸업학점을 넘지 않았다면 몇 학점을 더 들어야 하는지 제공한다.

3-2. 교양학점이 해당 학번의 교양 이수학점 범위 내에 있는지 검사한다.

3-2-1. 교양학점이 기준 범위 안에 든다면 통과한다.

3-2-2. 교양학점이 기준 범위 안에 들지 않았다면 몇 학점을 더 들어야 하는지 제공한다.

3-3. 전공학점이 해당 학과의 전공 최소이수학점을 넘는지 검사한다.

3-3-1. 전공학점이 해당 학과의 전공 최소이수학점을 넘었다면 통과한다.

3-3-2. 전공학점이 해당 학과의 전공 최소이수학점을 넘지 않았다면 몇 학점을 더 들어야 하는지 제공한다.

```
/**
 * [학점 검사]
 */
failure.append(checkCredit(studentId, studentMajor, totalCredit, kyCredit, majorCredit));
```

```
private static StringBuffer checkCredit(int studentId, String studentMajor, int totalCredit, int kyCredit, int majorCredit) {
    StringBuffer failure = new StringBuffer();

    // 해당 학과 졸업학점 가져오기
    int graduateCredit = DBConnection.getGraduateCredit(studentMajor);
    // 해당 학과 전공최소학점 가져오기
    int majorMinCredit = DBConnection.getMajorMinCredit(studentMajor);

    // (총 취득학점 - 초과한 교양 학점)
    if (studentId <= 2016) {
        // 16학번 이전 교양 학점 : 35~45학점
        if (kyCredit > 45) {
            totalCredit -= (kyCredit - 45);
            failure.append("교양학점 " + (kyCredit - 45) + "학점 초과되어 총 취득학점에서 " + (kyCredit - 45) + "학점 제외 (총 취득학점 : " + totalCredit + "학점)\n");
        }
    } else {
        // 17학번 이후 교양 학점 : 35~49학점
        if (kyCredit > 49) {
            totalCredit -= (kyCredit - 49);
            failure.append("교양학점 " + (kyCredit - 49) + "학점 초과되어 총 취득학점에서 " + (kyCredit - 49) + "학점 제외 (총 취득학점 : " + totalCredit + "학점)\n");
        }
    }

    // 1. 졸업학점 검사
    if (graduateCredit > totalCredit) {
        failure.append("졸업학점 " + (graduateCredit - totalCredit) + "학점 미달\n");
    }

    // 2. 교양학점 검사
    if (kyCredit < 35) {
        failure.append("교양학점 " + (35 - kyCredit) + "학점 미달\n");
    }

    // 3. 전공 최소이수학점 검사
    if (majorMinCredit > majorCredit) {
        failure.append("전공학점 " + (majorMinCredit - majorCredit) + "학점 미달\n");
    }

    return failure;
}
```

4. 전공필수 검사

4-1. 해당 학과의 전공필수 목록을 데이터베이스로부터 가져와서 전부 들었는지 검사한다.

4-1-1. 전부 들었다면 통과한다.

4-1-2. 전부 듣지 않았다면 들어야 할 과목들을 제공한다.

```java
/**
 * [전공필수 검사]
 */
failure.append(checkRequiredMajor(studentMajor, requiredMajor));
```

```java
/**
 * [전공필수 검사]
 */
private static StringBuffer checkRequiredMajor(String studentMajor, List<String> requiredMajor) {
    StringBuffer failure = new StringBuffer();

    // 해당 학과에서 전공필수 과목 가져오기
    List<String> requiredMajors = DBConnection.getRequiredMajor(studentMajor);

    // 들어야 할 전필과 학생이 들은 전필 비교
    Collection<String> std = requiredMajor;
    requiredMajors.removeAll(std);

    for (String str : requiredMajors) {
        failure.append("전공필수 '" + str + "' 미수강\n");
    }

    return failure;
}
```

5. 교양필수 & 비교과 검사

5-1. 해당 학번에 해당하는 교양필수를 전부 들었는지 검사한다.

5-1-1. 전부 들었다면 통과한다.

5-1-2. 전부 듣지 않았다면 들어야 할 과목들을 제공한다.

5-2. 해당 학번에 해당하는 비교과 프로그램을 이수했는지 검사한다.

5-2-1. 해당 학번에 해당하는 비교과 프로그램 기준을 넘었으면 통과한다.

5-2-2. 해당 학번에 해당하는 비교과 프로그램 기준을 넘지 않았다면 부족한 이수 학기 또는 마일리지 점수를 제공한다.

```java
/**
 * [교양필수 검사]
 */
failure.append(checkRequiredKy(studentId, studentMajor, nonSubject, mileage, engCertification, requiredKy));
```

```java
/**
 * [교양필수 검사]
 */
private static StringBuffer checkRequiredKy(int studentId, String studentMajor, int nonSubject, int mileage, boolean engCertification, List<String> requiredKy) {
    StringBuffer failure = new StringBuffer();

    int countCP = 0; // 채플 카운트
    int christian = 0; // 기독교 카운트
    int bible = 0; // 성서 카운트
    int collegeGuide = 0; // 대생길 카운트
    int socialGuide = 0; // 사생길 카운트
    int readDebate = 0; // 독토 카운트
    int writing = 0; // 글기 카운트
    int eng1 = 0; // 영어1
    int eng2 = 0; // 영어2
    int counseling = 0; // 진로와상담 카운트

    for (String line : requiredKy) {
        // 채플 검사
        if (line.equals("채플")) countCP++;

        // 기독교 과목 검사
        if (line.contains("기독교")) christian++;

        // 성서 과목 검사
        if (line.contains("성서")) bible++;

        // 대학생활길잡이 검사 (아노텐 '캠퍼스라이프' 대체)
        if (line.equals("대학생활길잡이") || line.equals("캠퍼스라이프")) collegeGuide++;

        // 사회생활길잡이 검사 (아노텐 '인문강단' 대체)
        if (line.equals("사회생활길잡이") || line.equals("인문강단")) socialGuide++;

        // 독서와토론 검사
        if (line.equals("독서와토론")) readDebate++;

        // 글쓰기의기초 검사 (19학번 이후 소프트웨어교과목으로 대체 가능)
        if (line.equals("글쓰기의기초") || line.contains("소프트웨어")) writing++;
```

```
        // 영어Ⅰ,Ⅱ 검사 (아노덴 'Speaking EnglishⅠ,Ⅱ' 대체 / 19학번 이후 영어인증자 면제)
        if (line.equals("영어Ⅰ") || line.equals("EnglishⅠ") || engCertification) eng1++;
        if (line.equals("영어Ⅱ") || line.equals("EnglishⅡ") || engCertification) eng2++;

        // 진로와상담 검사
        if (line.equals("진로와상담")) counseling++;
    }

    // 채플 검사
    if (countCP < 4) failure.append("교양필수 '채플' " + (4 - countCP) + "회 미수강\n");

    // 기독교 과목 검사
    if (christian < 1) failure.append("교양필수 '기독교 관련 과목' 미수강\n");

    // 성서 과목 검사
    if (bible < 1) failure.append("교양필수 '성서 관련 과목' 미수강\n");

    // 대학생활길잡이 검사 (아노덴 '캠퍼스라이프' 대체)
    if (collegeGuide < 1) {
        if (studentMajor.contains("아노덴")) {
            failure.append("교양필수 '캠퍼스라이프' 미수강\n");
        } else {
            failure.append("교양필수 '대학생활길잡이' 미수강\n");
        }
    }

    // 사회생활길잡이 검사 (아노덴 '인문강단' 대체)
    if (socialGuide < 1) {
        if (studentMajor.contains("아노덴")) {
            failure.append("교양필수 '인문강단' 미수강\n");
        } else {
            failure.append("교양필수 '사회생활길잡이' 미수강\n");
        }
    }
```

```
    // 독서와토론 검사
    if (readDebate < 1) failure.append("교양필수 '독서와토론' 미수강\n");

    // 글쓰기의기초 검사 (19학번 이후 소프트웨어 과목으로 대체 가능)
    if (writing < 1) {
        if (studentId >= 2019) failure.append("교양필수 '글쓰기의기초' 또는 '소프트웨어교과목' 미수강\n");
        else failure.append("교양필수 '글쓰기의기초' 미수강\n");
    }

    // 영어Ⅰ,Ⅱ 검사 (아노텐 'Speaking EnglishⅠ,Ⅱ' 대체 / 19학번 이후 영어인증자 면제)
    if (eng1 < 1) {
        if (studentMajor.contains("아노텐")) {
            failure.append("교양필수 'Speaking EnglishⅠ' 미수강");
        } else {
            failure.append("교양필수 '영어Ⅰ' 미수강");
        }
        if (studentId >= 2019) {
            failure.append(" 또는 영어인증 미인증\n");
        } else {
            failure.append("\n");
        }
    }
    if (eng2 < 1) {
        if (studentMajor.contains("아노텐")) {
            failure.append("교양필수 'Speaking EnglishⅡ' 미수강");
        } else {
            failure.append("교양필수 '영어Ⅱ' 미수강");
        }
        if (studentId >= 2019) {
            failure.append(" 또는 영어인증 미인증\n");
        } else {
            failure.append("\n");
        }
    }

    // 진로와상담 검사
    if (counseling < 4) failure.append("교양필수 '진로와상담' " + (4 - countCP) + "회 미수강\n");
```

```
    // 비교과 검사
    if (studentId >= 2020) {
        // 20학번 이후 마일리지 300점 이상
        if (mileage < 300) {
            failure.append("비교과 마일리지 " + (300 - mileage) + "점 미달\n");
        }
    } else if (studentId >= 2017) {
        // 17학번 이후 비교과 이수 학기 3학기 이상 또는 마일리지 300점 이상
        if (!(mileage >= 300 || nonSubject >= 3)) {
            failure.append("비교과 이수 학기 " + (3 - nonSubject) + "학기 미이수 또는 비교과 마일리지 " + (300 - mileage) + "점 미달\n");
        }
    }

    return failure;
}
```

6. 기타 교양 검사

6-1. 19학번은 교양을 인재상별로 5학점 이상 들었는지 검사한다.

6-1-1. 전부 들었다면 통과한다.

6-1-2. 전부 듣지 않았다면 인재상별 부족한 학점을 제공한다.

6-2. 20학번 이후는 교양을 핵심역량별로 1과목 이상 들었는지 검사한다.

6-2-1. 전부 들었다면 통과한다.

6-2-2. 전부 듣지 않았다면 핵심역량별 부족한 과목 수를 제공한다.

```
/**
 * [핵심역량 검사]
 */
failure.append(checkCoreKy(studentId, allKy));

/**
 * [인재상 검사]
 */
failure.append(checkTalent(studentId, allKy));
```

```java
private static StringBuffer checkTalent(int studentId, List<String> allKy) {
    StringBuffer failure = new StringBuffer();

    /**
     * 테스트해보고 싶으면 아래 if문 주석 처리하고 돌리기!!!
     */
    // 2019 학번 아니면 검사 필요없이 비어 있는 스트링버퍼 return
    //if (studentId != 2019) return failure;

    String[] talentType = {"소통하는지성인", "실천하는평화인", "도전하는창의인"};
    int creditSum = 0;

    for (String type : talentType) {
        creditSum = 0;

        List<String> talents = new ArrayList<>();
        talents.addAll(DBConnection.getTalent1(type));
        talents.addAll(DBConnection.getTalent2(type));

        talents.retainAll(allKy);

        // test
        System.out.println("\n===인재상 [" + type + "] 과목===");

        for (String str : talents) {
            int kyCredit = DBConnection.getKyCredit(str);

            // test
            System.out.println(str + " (" + kyCredit + "학점)");

            creditSum += kyCredit;
        }

        if (creditSum < 5) {
            failure.append("교양 인재상 '" + type + "' " + (5 - creditSum) + "학점 미이수\n");
        }
    }

    return failure;
}
```

```java
/**
 * [핵심역량 검사] (2020 학번 이상)
 */
private static StringBuffer checkCoreKy(int studentId, List<String> allKy) {
    StringBuffer failure = new StringBuffer();

    /**
     * 테스트해보고 싶으면 아래 if문 주석 처리하고 돌리기!!!
     */
    // 2020 미만 학번이면 검사 필요없이 비어 있는 스트링버퍼 return
    //if (studentId < 2020) return failure;

    String[] coreType = {"인문", "소통", "지식정보", "창의융합", "리더십", "글로벌"};

    for (String type : coreType) {
        List<String> cores = new ArrayList<>();
        cores.addAll(DBConnection.getCore1(type));
        cores.addAll(DBConnection.getCore2(type));

        cores.retainAll(allKy);

        // test
        System.out.println("\n===핵심역량 [" + type + "] 과목===");
        for (String str : cores) {
            System.out.println(str);
        }

        if (cores.size() < 1) {
            failure.append("교양 핵심역량 '" + type + "' 미수강\n");
        }
    }

    return failure;
}
```

2. 교양 추천 기능

2-1. 인기 교양 과목 조회

사용자는 학생들이 많이 수강한 순서대로 교양 목록을 제공받을 수 있어야 한다.

```
/**
 * 교양 카운트 증가
 */
failure.append(updateKyCount(allKy));
```

```java
/**
 * 교양 카운트 증가
 */
private static StringBuffer updateKyCount(List<String> allKy) {
    StringBuffer failure = new StringBuffer();

    for (String name : allKy) {
        if (DBConnection.updateKyCount(name) == 0) {
            failure.append("** 교양 [" + name + "] 카운트 실패!");
        }
    }

    return failure;
}
```

3. 정보공유 게시판

3-1. 게시글 조회

3-1-1. 사용자는 게시글 내용을 확인할 수 있어야 한다.

3-1-2. 사용자는 게시글 작성자일 경우 게시글을 수정 버튼을 확인할 수 있어야 한다.

3-1-3. 사용자는 게시글 작성자일 경우 게시글을 삭제 버튼을 확인할 수 있어야 한다.

3-2. 게시글 작성

3-2-1. 사용자는 별도의 로그인 없이 제목, 내용, 작성자, 비밀번호로만 게시글 작성할 수 있어야 한다. (비밀번호는 수정/삭제를 위함)

3-2-1-1. 사용자는 게시글 작성 성공 시 작성한 게시글 페이지로 이동할 수 있어야 한다.

3-2-1-2. 사용자는 게시글 작성 실패 시 '비어 있는 항목이 있습니다.' 문구를 제공받을 수 있어야 하며, 이어서 작성할 수 있어야 한다.

3-3. 게시글 수정

3-3-1. 사용자는 게시글 작성 시 입력했던 비밀번호를 입력할 수 있어야 한다.

3-3-2. 사용자는 수정을 위해 입력한 비밀번호가 일치하면 게시글 수정 페이지로 이동할 수 있어야 한다.

3-3-1-1. 사용자는 수정 성공 시 수정한 게시글 페이지로 이동할 수 있어야 한다.

3-3-1-2. 사용자는 게시글 수정 실패 시 '비어 있는 항목이 있습니다.' 문구를 제공받을 수 있어야 하며, 이어서 작성할 수 있어야 한다.

3-3-3. 사용자는 수정을 위해 입력한 비밀번호가 일치하지 않으면 '비밀번호가 일치하지 않습니다.' 문구를 제공받을 수 있어야 하며, 게시글 페이지로 이동할 수 있어야 한다.

3-4. 게시글 삭제

3-4-1. 사용자는 게시글 작성 시 입력했던 비밀번호를 입력할 수 있어야 한다.

3-4-2. 사용자는 삭제를 위해 입력한 비밀번호가 일치하여 게시글 삭제에 성공 시 게시글 목록 페이지로 이동할 수 있어야 한다.

3-4-3. 사용자는 삭제를 위해 입력한 비밀번호가 일치하지 않으면 '비밀번호가 일치하지 않습니다.' 문구를 제공받은 후 게시글 페이지로 이동할 수 있어야 한다.

3-5. 게시글 목록 및 검색

3-5-1. 게시글 목록

3-5-1-1. 사용자는 게시글 목록을 볼 수 있어야 한다.

3-5-1-2. 사용자는 안내글을 확인할 수 있어야 한다.

3-5-2. 게시글 검색

3-5-2-1. 사용자는 게시글 제목/내용으로 게시글을 검색할 수 있어야 한다.

3-5-2-2. 검색어를 포함하는 제목/내용의 게시글이 있으면 게시글 목록에 제공받을 수 있어야 한다.

3-5-2-3. 검색어를 포함하는 제목/내용의 게시글이 없으면 비어 있는 게시글 목록 제공받을 수 있어야 한다.

```xml
<!-- 게시글 목록 조회 -->
<select id="selectBoardList" resultType="BoardListDto">
    SELECT brd_key, brd_title, brd_writer, brd_wt_time, brd_lookup
    FROM board
    WHERE del_flag = 0
</select>

<!-- 게시글 key로 조회 -->
<select id="selectBoardByKey" resultType="BoardDto" parameterType="Integer">
    SELECT brd_title, brd_writer, brd_wt_time, brd_lookup, brd_content
    FROM board
    WHERE brd_key = #{key}
</select>

<!-- 조회수 증가 -->
<update id="updateLookup" parameterType="Integer">
    UPDATE board
    SET brd_lookup = brd_lookup + 1
    WHERE brd_key = #{key}
</update>

<!-- 게시글 작성 -->
<insert id="insertBoard" parameterType="BoardInsertDto">
    INSERT INTO board (
        brd_writer,
        brd_password,
        brd_title,
        brd_content
    ) VALUES (
        #{brdWriter},
        #{brdPassword},
        #{brdTitle},
        #{brdContent}
    )
</insert>
```

(게시판 쿼리문1)

```
<!-- 게시글 수정 -->
<update id="updateBoard" parameterType="BoardUpdateDto">
    UPDATE board
    SET brd_writer = #{brdWriter},
        brd_password = #{brdPassword},
        brd_title = #{brdTitle},
        brd_content = #{brdContent},
        brd_md_time = CURRENT_TIMESTAMP()
    WHERE brd_key = #{brdKey}
</update>

<!-- 게시글 삭제 -->
<update id="deleteBoard" parameterType="Integer">
    UPDATE board
    SET del_flag = 1
    WHERE brd_key = #{key}
</update>
```

(게시판 쿼리문2)

BoardDto.java

```java
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import java.sql.Timestamp;

@Getter
@Setter
@NoArgsConstructor
public class BoardDto {
    private String brdWriter;
    private String brdTitle;
    private String brdContent;
    private Timestamp brdWtTime;
    private Integer brdLookup;
}
```

BoardInsertDto.java

```java
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

@Getter
@Setter
@NoArgsConstructor
public class BoardInsertDto {
    private String brdWriter;
    private String brdPassword;
    private String brdTitle;
    private String brdContent;
}
```

BoardListDto.java

```java
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import java.sql.Timestamp;

@Getter
@Setter
@NoArgsConstructor
public class BoardListDto {
    private Integer brdKey;
    private String brdWriter;
    private String brdTitle;
    private Timestamp brdWtTime;
    private Integer brdLookup;
}
```

BoardUpdateDto.java

```java
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

@Getter
@Setter
@NoArgsConstructor
public class BoardUpdateDto {
    private Integer brdKey;
    private String brdWriter;
    private String brdPassword;
    private String brdTitle;
    private String brdContent;
}
```

(게시판 dto)

```java
@Mapper
public interface BoardMapper {

    // 게시글 목록 조회
    List<BoardListDto> selectBoardList();

    // 게시글 조회 + 조회수 증가
    List<BoardDto> selectBoardByKey(Integer key);
    void updateLookup(Integer key);

    // 게시글 작성
    void insertBoard(BoardInsertDto boardInsertDto);

    // 게시글 수정
    void updateBoard(BoardUpdateDto boardUpdateDto);

    // 게시글 삭제
    void deleteBoard(Integer key);

}
```

(게시판 mapper)

```java
@Service
@RequiredArgsConstructor
public class BoardServiceImpl implements BoardService {
    private final BoardMapper boardMapper;

    /**
     * 게시글 목록 조회
     * @return
     */
    @Override
    public List<BoardListDto> selectBoardList() { return boardMapper.selectBoardList(); }

    /**
     * 게시글 조회
     * @param key
     * @return
     */
    @Override
    public List<BoardDto> selectBoardByKey(Integer key) { return boardMapper.selectBoardByKey(key); }

    /**
     * 조회수 증가
     * @param key
     */
    @Override
    public void updateLookup(Integer key) { boardMapper.updateLookup(key); }

    /**
     * 게시글 작성
     * @param boardInsertDto
     */
    @Override
    public void insertBoard(BoardInsertDto boardInsertDto) { boardMapper.insertBoard(boardInsertDto); }
```

```java
    /**
     * 게시글 수정
     * @param boardUpdateDto
     */
    @Override
    public void updateBoard(BoardUpdateDto boardUpdateDto) { boardMapper.updateBoard(boardUpdateDto); }

    /**
     * 게시글 삭제
     * @param key
     */
    @Override
    public void deleteBoard(Integer key) { boardMapper.deleteBoard(key); }
```

(게시판 service)

```java
@RestController
@RequiredArgsConstructor
@RequestMapping("/board")
public class BoardRestController {
    private final BoardService boardService;

    /**
     * 게시글 목록 조회
     * @return
     */
    @GetMapping
    public List<BoardListDto> selectBoardList() { return boardService.selectBoardList(); }

    /**
     * 게시글 조회 + 조회수 증가
     * @param key
     * @return
     */
    @GetMapping("/{key}")
    public List<BoardDto> selectBardByKey(@PathVariable("key") Integer key) {
        boardService.updateLookup(key);
        return boardService.selectBoardByKey(key);
    }

    /**
     * 게시글 작성
     * @param boardInsertDto
     */
    @PostMapping
    public void insertBoard(BoardInsertDto boardInsertDto) { boardService.insertBoard(boardInsertDto); }
```

```java
    /**
     * 게시글 수정
     * @param key
     * @param boardUpdateDto
     */
    @PutMapping("/{key}")
    public void updateBoard(@PathVariable("key") Integer key, BoardUpdateDto boardUpdateDto) {
        boardUpdateDto.setBrdKey(key);
        boardService.updateBoard(boardUpdateDto);
    }

    /**
     * 게시글 삭제
     * @param key
     */
    @DeleteMapping("/{key}")
    public void deleteBoard(@PathVariable("key") Integer key) { boardService.deleteBoard(key); }
```

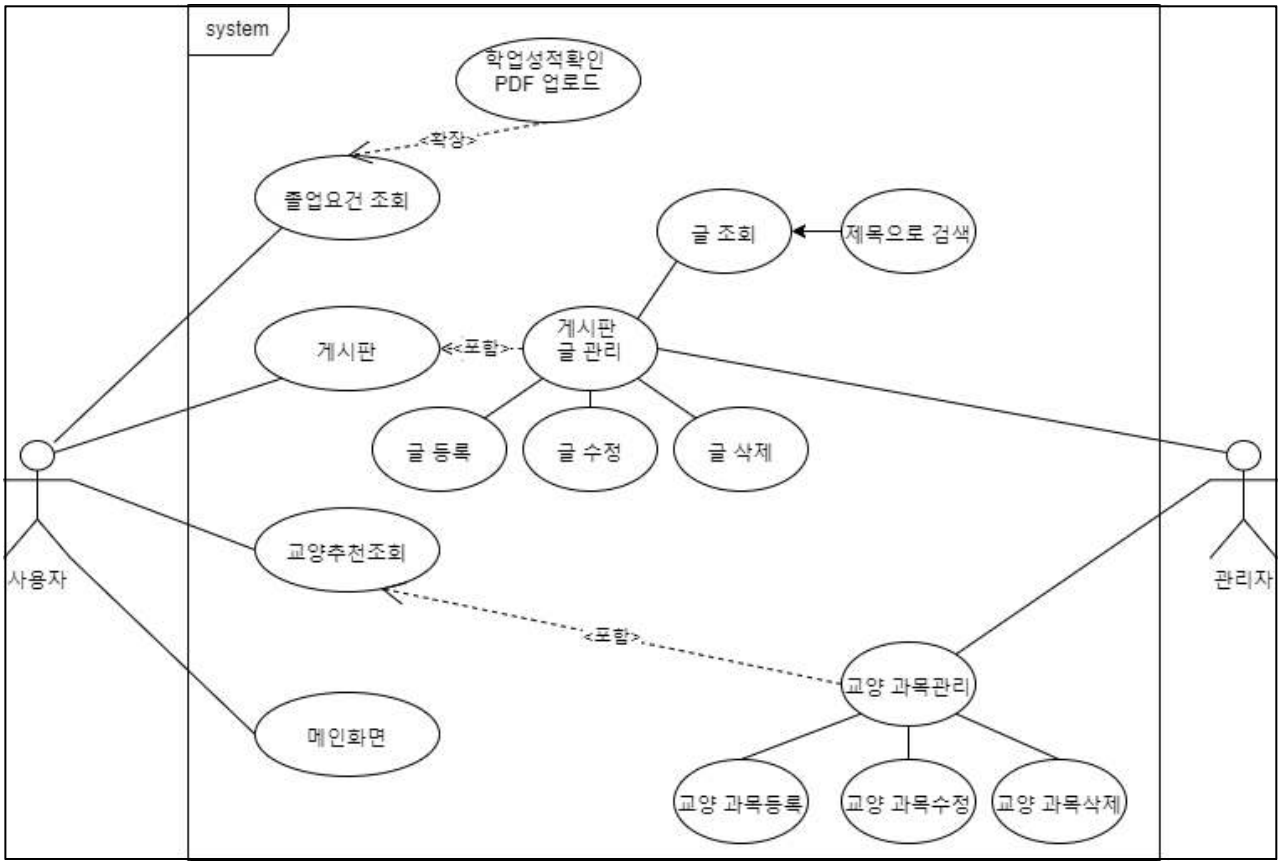
(게시판 controller)

[비기능적 요구사항]

성능	학업성적확인서를 업로드한 다음, 결과가 나오기까지 5초를 넘기면 안 된다.
	게시판 업로드는 3초를 넘기면 안 된다.
신뢰성	학업성적확인서를 올렸을 때 실패는 5% 미만이어야 한다.
사용 용이성	업로드 버튼을 누르고 올리면 바로 결과 창이 나온다.
유지보수성	새롭게 추가된 과목들의 추가가 가능해야 한다.
효율성	PDF 업로드 시 학업성적 확인서 하나만 업로드 해야 한다.

1. 학업성적확인서를 업로드한 다음, 결과가 나오기까지 5초를 넘기면 안 된다. (성능)
2. 게시판 업로드는 3초를 넘기면 안 된다. (성능)
3. 학업성적확인서를 올렸을 때 실패는 5% 미만이어야 한다. (신뢰성)
4. 업로드 버튼을 누르고 올리면 바로 결과창이 나온다. (사용용이성)
5. 새롭게 추가된 과목들의 추가가 가능해야 한다. (유지보수성)
6. PDF 업로드 시 학업성적확인서 하나만 업로드 해야 한다. (효율성)

[유스케이스 다이어그램]



[기대효과]

1. 시스템 자동화

사용자가 하나하나 조건을 찾지 않고 학업성적확인서 PDF를 업로드하면 결과 조회 페이지를 보여준다. 업로드를 실패하면 다시 업로드 페이지로 이동하여 업로드를 성공할 수 있도록 한다. 사용자의 졸업요건이 미달된 경우 부족한 졸업요건 리스트를 제공함으로써 사용자가 졸업 요건을 수기로 확인하는 것과 비교하여 정확성을 높이고 시간 절약을 할 수 있다.

2. 모든 학생이 사용

사용자는 학번, 학년에 상관없이 학업성적확인서 PDF를 업로드하

면 부족한 학점과 본인 학번에 맞는 졸업요건 리스트가 제공되고 인기 있는 교양 추천을 받을 수 있으며 정보 공유 게시판을 이용할 수 있다.

3. 주기적인 방문 효과

사용자가 졸업 시즌에만 단발성으로 끝나는 것이 아닌 정보 공유 게시판 기능을 통해 다양한 학사 정보와 수강신청, 졸업요건 대체 조건 등의 정보를 공유하고 주기적으로 사이트에 방문하게 한다.

4. 직관적인 UI

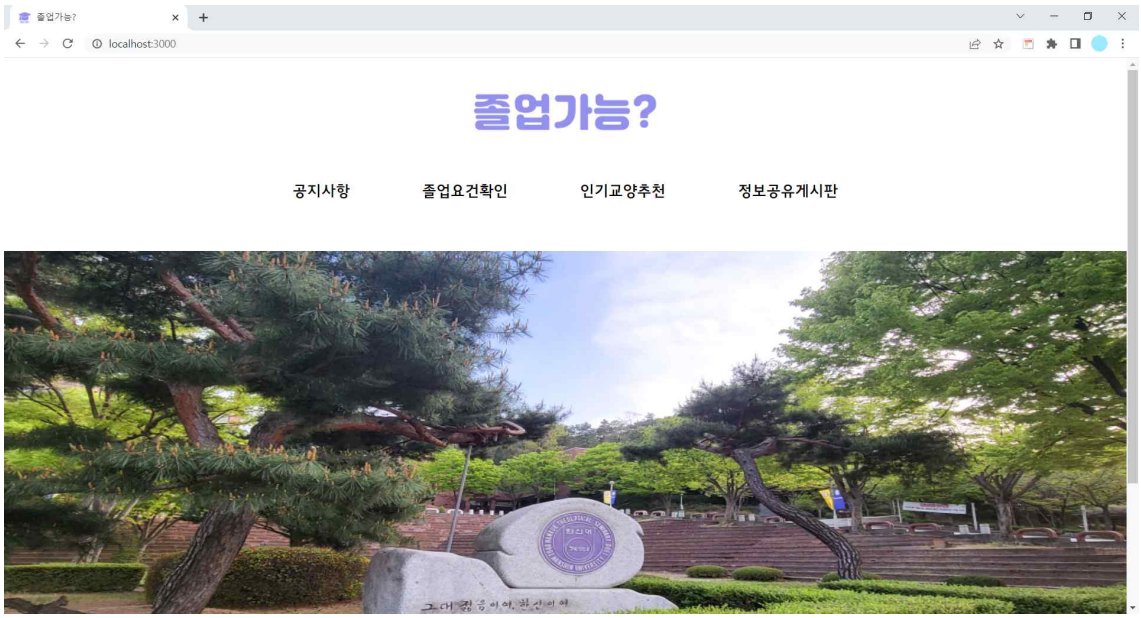
사용자가 학업성적확인서 PDF를 업로드를 성공한다면 다음 페이지인 결과 조회 페이지의 UI로 사용자의 졸업요건 확인 여부를 확인할 수 있도록 한다.

사용자가 파악하기 쉽도록 보기 좋은 구조를 통해 필요한 졸업요건이나 채우지 못한 필수 과목들의 조회 결과를 한눈에 파악할 수 있게 해준다.

5. 교무팀

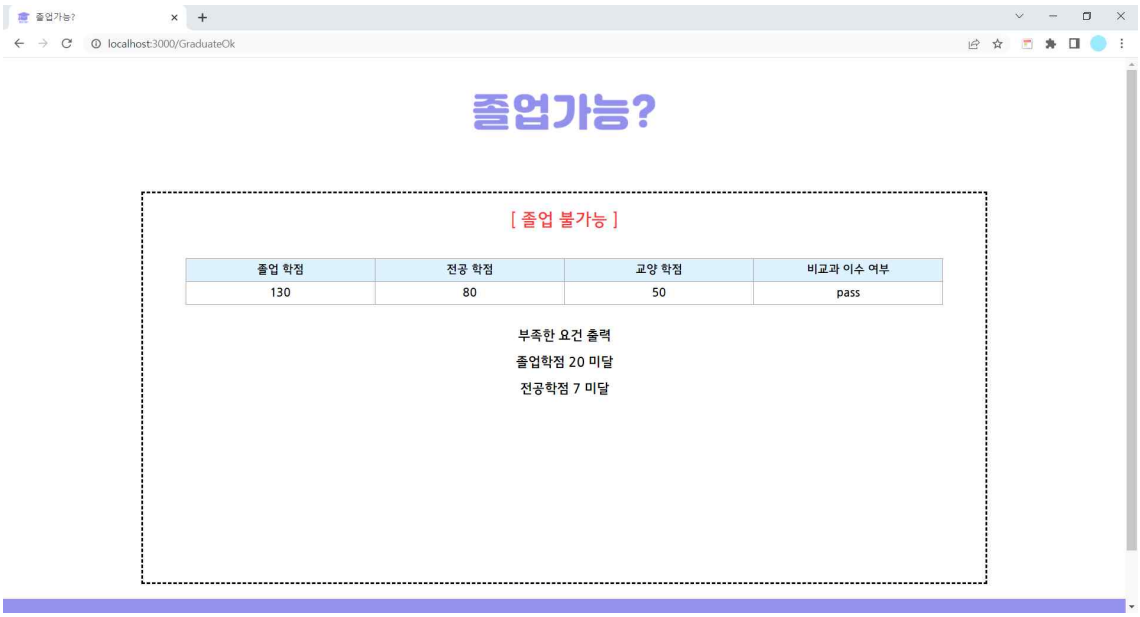
교무팀과 인터뷰를 진행한 결과, 졸업 시즌에 문의가 집중적으로 발생하고 (2022년 2월 기준) 전체 졸업예정자 중 37.05%의 학생들이 졸업요건을 채우지 못하고 졸업이 취소되었다. 우리가 만드는 졸업요건 확인 시스템을 교무팀에서 사용할 의향이 있다는 답변을 받았고 수기로 작성하여 확인하는 전과 달리 위 시스템이 도입되면 학생들이 본인의 졸업요건을 자동으로 확인할 수 있고 관리할 수 있어서 교무팀의 부담을 덜어내는 기대 효과가 있다.

[UI 프로토타입]

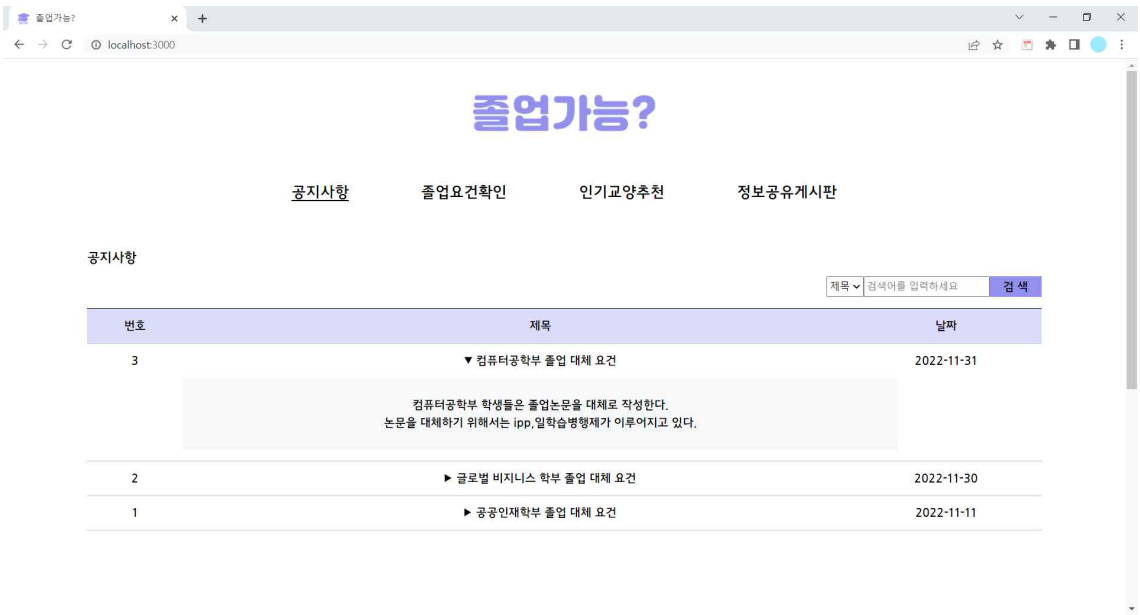


메인 페이지

PDF 업로드 페이지



졸업결과 페이지



공지사항 페이지

게시판 페이지

인기 교양 추천 페이지

[DB 테이블]


학과
MAJOR
학과코드
MAJOR_NO
VARCHAR(30)
학과이름
MAJOR_NAME
VARCHAR(500)
전공최소이수학점
MAJOR_MIN_CREDIT
INT
졸업이수학점
GRADUATE_CREDIT
INT
전공필수
MAJOR_REQUIRED_COURSE
전공필수 키
MRQ_KEY
INT
학과번호
MAJOR_CODE
VARCHAR(30)
학번
MRQ_STD_ID
INT
전필이름
MRQ_NAME
VARCHAR(500)
전필학점
MRQ_CREDIT
INT
삭제여부
DEL_FLAG
INT
교양
KY_COURSE
교양키
KY_KEY
INT
교양이름1
KY_NAME1
VARCHAR(500)
교양이름2
KY_NAME2
VARCHAR(500)
교양학점
KY_CREDIT
FLOAT
교양필수여부
KY_REQUIRED
INT
교양인재상
KY_TYPE
VARCHAR(200)
교양핵심역량
KY_CORE
VARCHAR(200)
교양수강횟수
KY_COUNT
INT
삭제여부
DEL_FLAG
INT
게시판
BOARD
게시판키
BRD_KEY
INT
작성자
BRD_WRITER
VARCHAR(500)
비밀번호
BRD_PASSWORD
VARCHAR(500)
제목
BRD_TITLE
VARCHAR(2000)
내용
BRD_CONTENT
TEXT
작성시간
BRD_WT_TIME
TIMESTAMP
수정시간
BRD_MD_TIME
TIMESTAMP
조회수
BRD_LOOKUP
INT
삭제여부
DEL_FLAG
INT
공지사항
NOTICE
공지키
NOTI_KEY
INT
카테고리
NOTI_CATEGORY
VARCHAR(200)
제목
NOTI_TITLE
VARCHAR(2000)
내용
NOTI_CONTENT
TEXT
작성시간
NOTI_WT_TIME
TIMESTAMP
삭제여부
DEL_FLAG
INT

4. 개발 환경

- IDE : Intelli J & VS Code
- DB : Maria DB
- 개발 언어 : JAVA, Java Script, HTML, CSS
- 라이브러리 : React
- 프레임워크 : Springboot
- 형상관리 : GIT

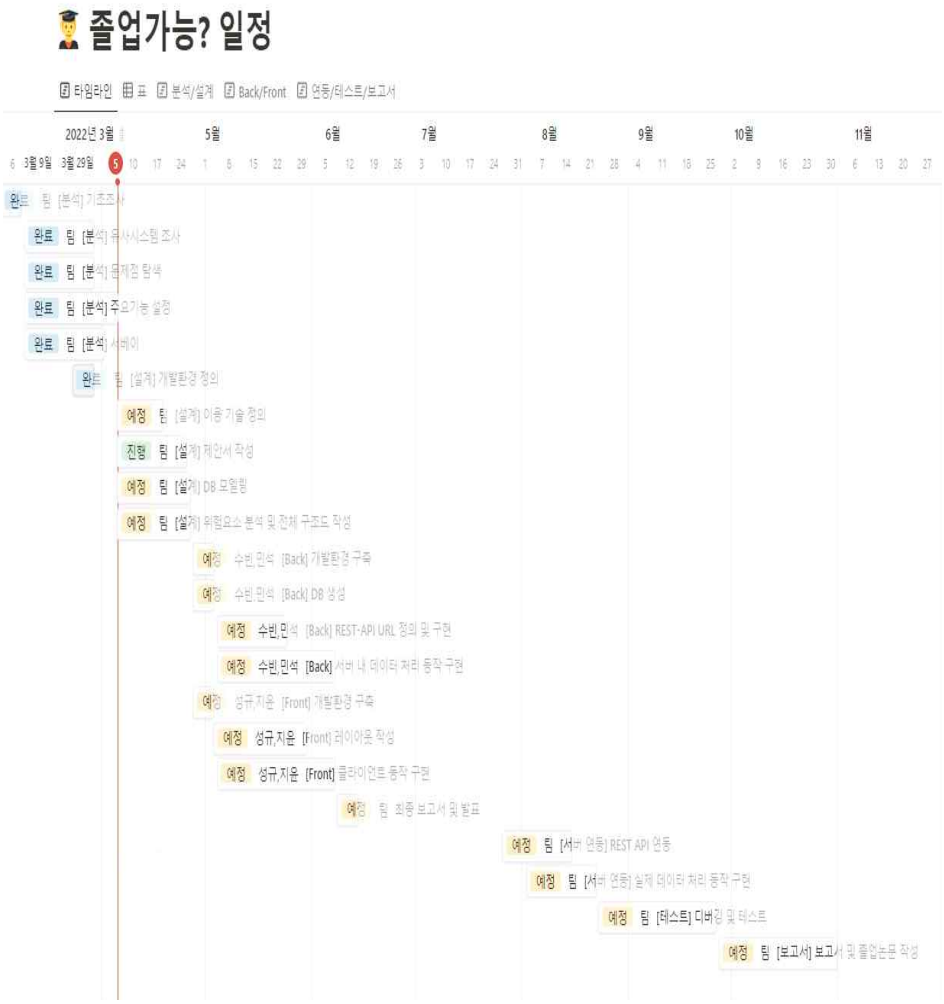
5. 위험 요소

- Spring boot 라이브러리를 사용할 때 적절한 라이브러리를 찾지 못 할 수 있다.
- 교양 추천 기능에서 교양 count가 제대로 안 될 가능성이 있다.
- 교양을 데이터베이스에 저장하는 과정에서 일부 누락되는 과목이 생길 수 있다.
- 졸업요건이 학과/학번별로 다양하여 정보가 누락될 가능성이 있다.

6. 일정 계획

(링크)

https://cyclic-pleasure-1d8.notion.site/cfc00b3dc86c48ffb1f7b8daa0fb2cc4?v=474ffe3bea07479fb3975b883077f168



👨‍🎓 졸업가능? 일정

타임라인 | 표 | 분석/설계 | Back/Front | 연동/테스트/보고서

# 번호	담당	구분	Aa 내용	날짜	진행상황
1	팀	분석	[분석] 기초조사	2022년 3월 2일 → 2022년 3월 8일	완료
2	팀	분석	[분석] 유사시스템 조사	2022년 3월 9일 → 2022년 3월 29일	완료
3	팀	분석	[분석] 문제점 탐색	2022년 3월 9일 → 2022년 3월 29일	완료
4	팀	분석	[분석] 주요기능 설정	2022년 3월 9일 → 2022년 4월 5일	완료
5	팀	분석	[분석] 서베이	2022년 3월 9일 → 2022년 4월 1일	완료
6	팀	설계	[설계] 개발환경 정의	2022년 3월 23일 → 2022년 3월 29일	완료
7	팀	설계	[설계] 이용 기술 정의	2022년 4월 5일 → 2022년 4월 18일	예정
8	팀	설계	[설계] 제안서 작성	2022년 4월 5일 → 2022년 4월 25일	진행
9	팀	설계	[설계] DB 모델링	2022년 4월 5일 → 2022년 4월 26일	예정
10	팀	설계	[설계] 위험요소 분석 및 전체 구조도 작성	2022년 4월 5일 → 2022년 4월 26일	예정
11	수빈,민석	Back	[Back] 개발환경 구축	2022년 4월 27일 → 2022년 5월 3일	예정
12	수빈,민석	Back	[Back] DB 생성	2022년 4월 27일 → 2022년 5월 3일	예정
13	수빈,민석	Back	[Back] REST-API URL 정의 및 구현	2022년 5월 4일 → 2022년 5월 24일	예정
14	수빈,민석	Back	[Back] 서버 내 데이터 처리 동작 구현	2022년 5월 4일 → 2022년 6월 7일	예정
15	성규,지윤	Front	[Front] 개발환경 구축	2022년 4월 27일 → 2022년 5월 3일	예정
16	성규,지윤	Front	[Front] 레이아웃 작성	2022년 5월 3일 → 2022년 5월 30일	예정
17	성규,지윤	Front	[Front] 클라이언트 동작 구현	2022년 5월 4일 → 2022년 6월 7일	예정
18	팀	보고서	최종 보고서 및 발표	2022년 6월 8일 → 2022년 6월 14일	예정
19	팀	서버 연동	[서버 연동] REST API 연동	2022년 7월 26일 → 2022년 8월 15일	예정
20	팀	서버 연동	[서버 연동] 실제 데이터 처리 동작 구현	2022년 8월 2일 → 2022년 8월 22일	예정
21	팀	테스트	[테스트] 디버깅 및 테스트	2022년 8월 23일 → 2022년 9월 26일	예정
22	팀	보고서	[보고서] 보고서 및 졸업논문 작성	2022년 9월 27일 → 2022년 10월 31일	예정

[졸업가능? 일정 노션 페이지] [9]

7. 참고 문헌

[1] 이성구, "소프트웨어 공학 기본 원리", 홍릉과학출판사, 2020.

[2] 한신대학교 학사공지 - [졸업] 2022년도 8월 졸업예정자 "졸업여부 셀프테스트" 제출 안내

[3] 국민대학교 소식 - [졸업요건]졸업요건과 이수학점 확인하는 방법!

[4] [React.JS] 강좌 12편 axios 모듈을 통한 웹서버와의 통신 (AJAX) 알아보기 | VELOPERT.LOG

[5] 구글 애널리틱스 API를 사용한 Flask 앱을 uWSGI와 nginx로 배포한 과정 | blog.rhostem.com

[6] 리액트 공식홈페이지 docs

[7] 리액트와 스프링 연동을 위한 과정

[8] 리액트 (웹 프레임워크) - 위키백과, 우리 모두의 백과사전 (wikipedia.org)

[9] 졸업가능? 일정 노션 페이지