**팀 EAMS 프로젝트 설계 계획서**

**주제: 개선된 학사 관리 시스템**

**2조**

**학년/반: 3-A**

**팀명: EAMS(Enhanced Academic Management System)**

**프로젝트 참여자: 2106117 김태연, 2306007 김수혁, 2306008 김명훈**

**제출일자: 2025년 6월 9일**

**목차**

|  |
| --- |
| **1 개요**   * 1. **프로젝트 개요**   2. **프로젝트 선정 동기**   3. **프로젝트로 인한 효과**   4. **프로젝트의 산출물**   **2 업무 분류 차트**  **3 표준 및 개발 절차**  **4. 개발 환경**  **5 조직 구성 및 인력 배치**  **5.1 조직 구성**  **5.2 직무 기술**  **6 일정 예측**  **7 스터티자료 및 참고자료**  **8 위험관리**  **8.1 변경관리**  **8.2 위험관리**  **8.3 진행관리**  **8.4 문제점 해결 방안**  **9 검토회의**  **9.1 검토회의 일정**  **9.2 검토회의 진행방법**  **9.3 검토회의 후속조치**  **10 유지보수**  **11 문서화**  **12 성능 테스트 방법(임시)**  **13 메인 인터페이스 설계**  **14 기능 설계**  **15 db테이블 설계**  **16 클래스 설계** |

**1. 개요**

**1.1 프로젝트 개요**

**해당 프로젝트는 기존의 학사 관리 시스템에서 발생하는 문제점들을 개선하여 새로운 학사 관리 시스템을 개발하는 팀 프로젝트이다.**

**1.2 프로젝트 선정 동기**

**현재 사용되는 학사관리 시스템의 기본적으로 두가지 문제점이 존재하며 첫**

**번째 문제점은 기능들이 어디에 위치해 있는지 찾는 것이 어렵다는 문제이**

**존재한다.**

**두 번째 문제점은 메신저의 부재로 학생들이나 교수들이 학사 관리 시스템안**

**에서 소통할 창고인 메신저가 굉장히 빈약하다는 문제점이 존재한다.**

**이러한 두가지 문제점은 충분히 개선할 수 있는 문제점이라고 봤기에 해당 프로젝트를 계획했다.**

**1.3 프로젝트로 인한 효과**

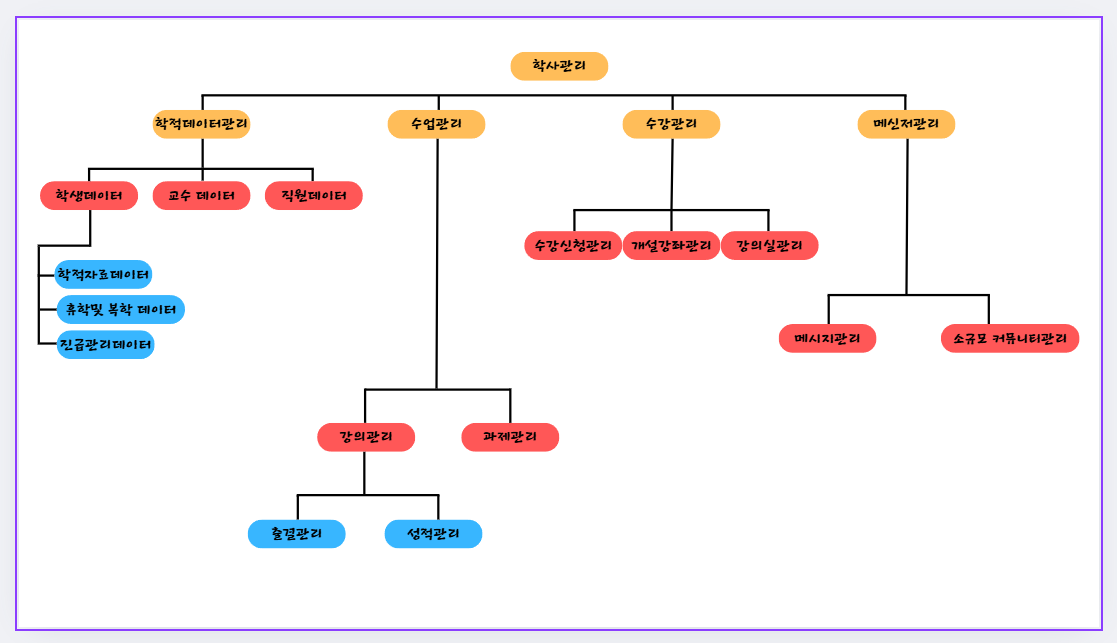
**첫 번째 문제점은 디자인 적인 부분에서 많은 개선이 필요한 문제로, 타 학교 포털 사이트와 여러 웹사이트의 디자인을 참고하여 기존 디자인보다 기능들의 위치를 더 직관적으로 배치하며, 또한 검색 기능에 연관 검색어 기능을 추가하여 사용자가 입력한 단어와 관련된 기능을 쉽게 확인할 수 있도록 함으로 원하는 기능의 위치를 보다 쉽게 찾을 수 있을 것을 기대한다.**

**현재 학사 관리 시스템에는 메신저 기능이 존재하지만, 사실상 거의 사용되지 않는 상황이다. 이에 따라, 단체 방 기능과 파일 업로드 기능 등 다양한 기능을 갖춘 새로운 메신저를 개발하고, 이를 학생들과 교수들이 자주 방문하는 포털 사이트의 메인 페이지에 배치한다면, 메신저를 더 자주 접하게 되어 사용 빈도가 증가할 것으로 기대된다.**

**1.4 프로젝트의 산출물**

**이 프로젝트는 웹사이트 기반의 프로젝트로 이 프로젝트 산출물은 웹사이트로만 작동하는 학사 관리 시스템과 그 학사 관리 시스템의 기능들이 어떤 식으로 동작하게 할 지 결정한 회의록이 본 프로젝트의 산출물로 해당 산출물들은 깃허브에 공개된다.**

**2 업무 분류 차트**

****

**우선 wbs차트를 보면 학사관리를 네 가지 작업으로 세분화 시켰다.**

**학적 데이터관리는 실제로 데이터가 들어가는 작업들을 나타낸다.**

**실제로 관리할 데이터들은 세가지로 학생 데이터, 교수 데이터, 직원 데이터 세가지이기에 학생 데이터 관리, 교수 데이터 관리, 직원 데이터 관리 이렇게 세가지의 기능으로 분리한다.**

**학생데이터 관리에는 학생 인적 사항 그리고 학적 기록이 관리할 수 있도록 하며**

**복학 및 휴학 처리 기능과 특정 조건이 되면 자동으로 진급처리 되는 기능 이렇게 3가지 기능이 존재한다.**

**수업관리는 성적입력 출석관리, 과제 관리 등 교수들이 수업들을 관리할 때 사용하는 기능들을 모아 놓은 것이다.**

**수강 관리는 수강 신청과 관련된 작업들이 들어간다. 수강관련 작업에는 교수들이 강좌를 개설하는 것과 학생들이 개설된 강좌들을 신청하는 수강 신청이 존재한다.**

**메신저 관리는 직접적인 메시지를 보내는 메시지 기능과 자유 게시판과 같은 소규모 커뮤니티 개설 관리하는 소규모 커뮤니티 관리 두가지로 이루어져 있다.**

**3 표준 및 개발 절차**

**이 프로젝트는 전통적인 폭포수(Waterfall) 모델을 기반으로 하되, 설계 단계에서 프로토타입 개발을 병행하여 설계가 적절한지 판단해 설계의 완성도를 높인다.**

**이는 이미 계획단계에서 요구사항이 어느정도 구체적으로 뽑혔기에 분석과정에서 프로토타입을 진행하여 요구사항을 구체화하는 것 보다 설계단계에서**

**포로토타입 모델을 진행하여 요구사항을 분석하여서 나온 기능들이 정확한지**

**검증함과 동시에 프로토타입은 디자인 설계하는 과정에서도 많은 도움이 되기에 설계단계에서 프로토타이핑을 진행하여 전체적인 완성도를 높인다.**

**또한 소스코드를 작성할 때 표준적인 절차는 클래스명은 파스칼 스타일로 작성하며 함수(또는 메서드)와 변수명은 캐멀 스타일로 작성한다.**

**if문과 for문을 사용할 때는 가능한 주석을 쓰며 클래스와 메서드를 선언할 시 무슨 클래스와 메서드인지 설명하는 주석을 반드시 작성해야 한다.**

**4 개발 환경**

**HTML및 CSS와 자바 스크립트, next.js 프레임워크를 통해 진행한다.**

**IDE는 비주얼 스튜디오 코드로 하여 소스코드를 작성한다.**

**Next.js프레임워크를 통해 프론트엔드 및 백엔드를 전부 개발하고자 한다.**

**Next.js는 틱톡, 나이키 홈페이지, twitch 등 많은 기업들에서 사용하고 있는 프레임워크로 자바 스크립트를 통해 많은 작업들이 가능하며 간단하게**

**웹사이트 개발이 가능하기에 next.js를 통해 개발한다.**

**마지막으로 DB는 MariaDB를 사용하여서 개발하고자 한다.**

**5 조직 구성 및 인력 배치**

**5.1 조직 구성**

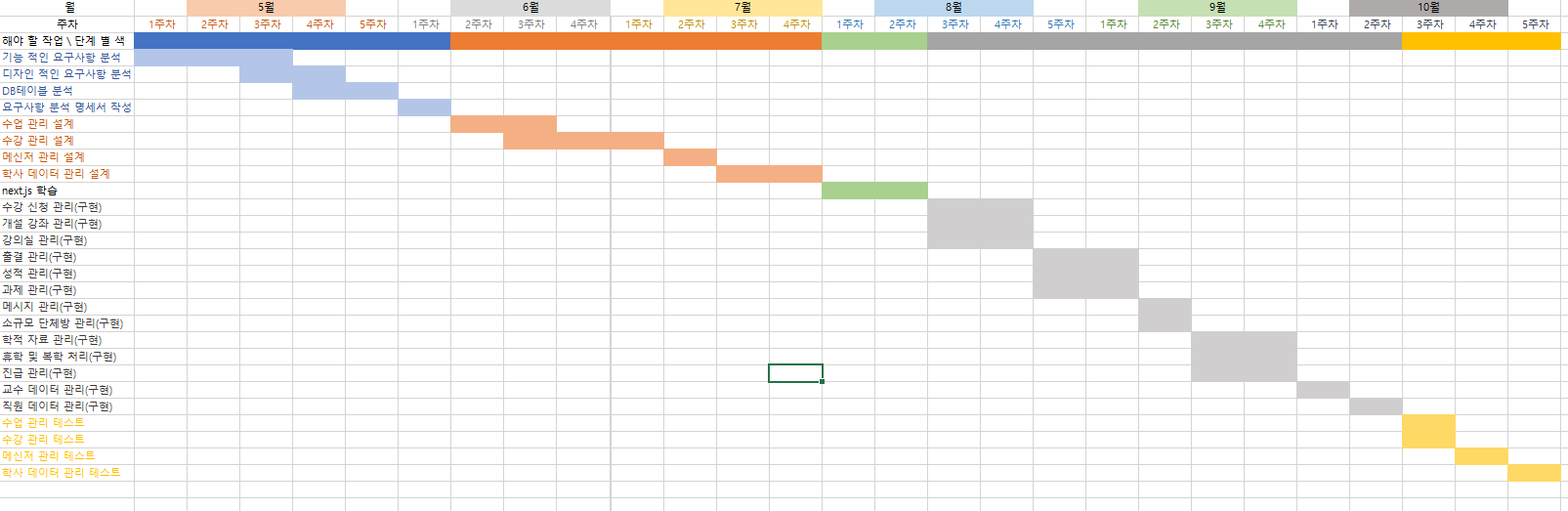
**조직 구성 3명의 팀원으로 구성되며 모두가 다 같이 모든 파트를 전부 참여하기에 구분이 거의 없지만 각 파트의 파트장은 존재한다. 백 엔드에 경우 김명훈씨가 파트장이며, 프론트 엔드에 경우 김수혁씨가 파트장이며, 디자인파트의 경우 김태연씨가 파트장이다. 파트장들은 각자 파트에서는 발언권이 높으며 해당 파트는 대체로 파트장에 의견에 따라 구현하게 된다.**

**5.2 개발의 필요한 기술**

* **DBSQL문 관련 지식**
* **정규화**
* **뷰와 인덱스 객체 관련 지식**
* **ERD 설계**
* **테이블 관계 설정 관련 지식**
* **자바 스크립트 이벤트 처리**
* **DBSQL문 관련 지식**
* **정규화**
* **뷰와 인덱스 객체 관련 지식**
* **ERD 설계**
* **테이블 관계 설정 관련 지식**
* **자바 스크립트 이벤트 처리**
* **자바 스크립트 기본 문법**
* **객체 지향 개념**
* **HTML 및 CSS 지식**
* **깃 관련 지식**
* **NEXT.JS 프레임워크 관련지식**
* **React.js라이브러리 지식**
* **Prisma 라이브러리 지식**

**현재 모든 팀원들은 NEXT.JS 프레임워크 관련 지식과 깃 관련 지식, React.js라이브러리 지식을 제외한 모든 기술들을 거의 다 습득하였다.**

**6. 일정 예측**

****

**5,6월은 프로젝트 분석 및 설계를 진행할 것이며 6월 기말고사 후에는**

**Next.js에 대해서 스터디가 진행될 예정이다.**

**Next.js에 대한 스터디가 마무리되는 7월 중순부터 구현 단계를 진행한다.**

**구현 단계는 수업관리, 수강관리, 메신저 관리, 학사 데이터관리 순서로 진행될 예정이다. 이는 DB와 관련 작업이 많은 들어가는 학사 데이터 관리를 가장 나중에 작업하여서 학사 데이터 관리 작업이 진행될 때 DB테이블을 확정시키 위합이다.**

**구현단계가 끝나는 10월 후반에는 테스트 작업이 진행된다.**

**테스트 작업은 주변 지인들에게 해당 소프트웨어를 공유하여서 개선 사항이 존재하는지를 물어보며 개선사항이 존재할 경우 해당 개선사항을 구현하기 위해 많은 시간이 투자될 경우가 아닐 경우에는 개선사항을 구현한다.**

**7. 스터티 자료 및 참고 자료**

**스터티의 사용할 책은 처음부터 시작하는 Next.js / React 개발 입문책으로**

**해당 책을 통해 NEXT.JS에 대해 학습을 진행한다.**

**만일 자바스크립트를 구현중에 에러 사항이 생길 경우 혼자 공부하는 자바 스크립트책과 깃허브, 유튜브를 참고한다.**

**DB 경우 오민근 교수님에 자료들을 참고하여서 DB를 제작한다.**

**8 기술관리 방법**

**8.1 변경 관리**

**만일 설계대로 구현이 불가능하다고 판단될 시 검토회에서 설계를 검토한다, 이때 변경된 설계에서 해당 요구사항을 구현하기 위해 추가적인 기능이 많이 요구될 시 사용자의 편의를 위한 기능들이 존재할 경우 구현의 필수적인 기능들이 아닌 편의를 위한 기능들을 전부 삭제를 하며 모든 기능이 요구사항을 구현하기 위한 필수적인 기능일 시 몇 개의 기능들로 대체할 수 있는지 검토해본다, 만일 대체가 불가능 할 시 추가적으로 필요한 기능들을 추가하여 구현한다.**

**8.2 위험 관리**

**파일 삭제: 깃허브에 저장하여 팀원 컴퓨터의 파일이 없어지게 되도 깃허브에 미리 파일들을 보관하여 사라져도 불러올 수 있게 한다.**

**소통부재: 팀원 간에 소통이 안되어 긴밀하게 연결되는 파트에서 제대로 작동이 안되는 문제가 발생할 수 있다. 이를 방지하고자 기능을 구현했 을시 어떤 식으로 구현했는지 정규 회의에서 말하는 시간을 갖는다**

**8.3 비용 및 진도관리**

**일정이 미뤄졌을 경우, 검토 회의에서 지연된 이유를 파악하고, 필요하면 업무를 다시 재분배해서 새로운 일정을 짠다.**

**새로운 요구사항이 추가되었을 때, 해당 요구사항을 구현하는 데 큰 비용 이 들지 않는다면(기능 점수 8이하), 가능한 한 추가한다.**

**하지만 너무나 많은 기능이 추가되면 안되기에 최대 기능점수는 260으로 잡는다.**

**만일 해당 최대 기능점수를 넘는 기능을 추가해야 할 경우 기존의 기능 중에 삭제할 기능을 검토한다.**

**8.4 문제점 해결 방안**

**지출: 금전적인 문제가 발생하여서 지출이 필요할 경우 각자 1/n 지불하여**

**문제를 해결한다.**

**장시간 팀원의 부재: 인원이 부재 될 경우 따로 회의를 현재 인원의 맞게**

**파트를 재분배하여 일정을 조정한다.**

**9 검토 회의**

**9.1 검토회의 일정**

**매주 수요일마다 검토회의를 진행한다.**

**매일 월요일 문서화회의를 진행하여 현재 진행사항들에 대해 저장한다.**

**9.2 검토회의 진행 방법**

**메인 회의 때 결정한 작업들이 잘 진행되었나 확인하며 추가로 안건들에**

**대해서도 회의한다.**

**9.3 검토회의 후속 조치**

**만일 검토회를 통해 진행하여 결정된 사항은 특별한 경우가 아니면 무조**

**건 이행한다.**

**10 유지보수**

**건의사항 포럼을 만들어서 해당 건의사항을 바로 작성할 수 있게 한다.**

**신속하게 처리할 수 있는 건의사항(ex: 포털 메뉴 위치 변경)은 빠르게 처리한다.**

**신속하게 처리할 수 없는 건의사항(ex: 도서관 홉페이지 같은 신기능 추가)의 경우 비용 산정한 후 많은 비용이 들지 않을 경우에 추가를 진행하며 만일 비용이 많은 드는 기능일 경우 다른 기능으로 대체 가능한지 찾는다.**

**만일 직접 문의 원할 경우 개발자와 직접적인 1대 1 문의를 원할 시 이메일로**

**문의한다.**

**11 문서화**

**회의기록 및 결정사항들은 모두 워드파일로 저장하여 나중에 확인할 수 있도록 한다.**

**발표의 사용된 ppt와 각 단계에서 산출한 문서들은 word,excel 파일로 작성되어 확인 가능하게 한다.**

**프로젝트 개발에 사용된 모든 소스코드 파일들을 깃허브에 업로드하여 관리한다.**

**12 성능 시험 방법**

**반응시간 테스트: 모든 인터페이스의 요소들을 클릭하였을 때 반응시간이 3초**

**내 안으로 되어야 한다. ->수강, 수업, 메신저 인터페이스의 핵심 요소들을(성적**

**입력, 수강신청, 게시판 글 등록, 메시지 보내기) 수행하였 때 반응하는 시간을**

**측정한다.**

**트래픽 수용도 측정: 실제 웹서버 구축하여 프로그램을 통해 3대의 컴퓨터에서**

**동시에 10개의 브라우저를 띄워서 수강신청 사이트에 동시에 접속하게 하여 한번**

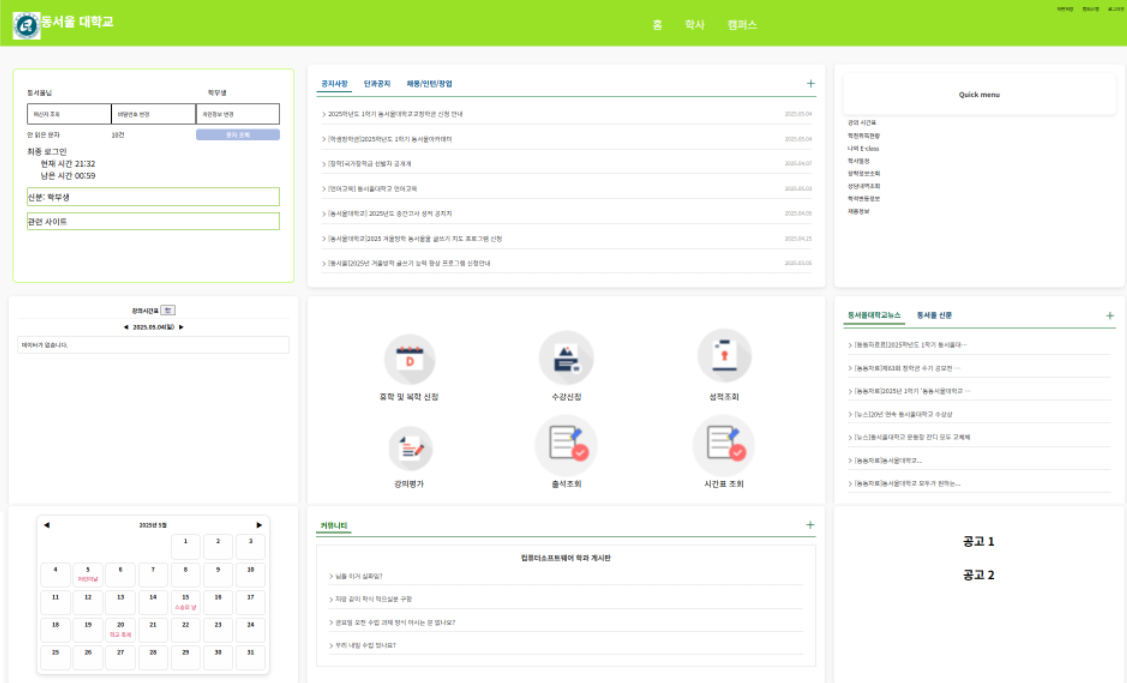
**에 몇 명에 사용자를 견디는 지 테스트를 진행한다.**

**처리량 측정: 10초에 얼마나 많은 작업을 처리하는 지 테스트를 측정해야 한다. 따라서 처리량이 가장 많은 메신저 인터페이스에서 10초 동안 메신저를 1000개를 보내었을 때 어느정도 처리하는 지 테스트를 진행한다.**

**안정성: 서버를 긴 시간동안 작동시키며 2시간마다 각 기능들을 테스트를 진행하여 서버가 긴 시간동안 작동하였을 때 이상여부를 테스트를 한다.**

**데이터 입력: 타 팀에게 부탁하여 실제 데이터를 입력해보게 하여 db상에 데이터가 제대로 저장되는지 테스트를 진행한다.**

**13. 메인 인터페이스 설계**

****

**해더 구역**

* **로그인: 일반적으로는 메인 사이트에 메인 구역의 내용들이 보이지 않게 하며 로그인 버튼을 클릭 후에 나오는 로그인 창에서 로그인이 완료되어야 메인 구역의 내용들이 보이도록 한다.**
* **로그인이 완료되며 로그인이 로그아웃으로 변경되며 남은 시간이 0이 될 시 자동으로 로그아웃 된다.**
* **학사 버튼: 클릭시 동서울 대학교 학사홈페이지로 이동하게 된다.**
* **캠퍼스 버튼: 클릭시 동서울 대학교 캠퍼스맵으로 이동하게 된다.**
* **검색 기능: 메인 인터페이스의 모든 기능 및 학사 시스템내 모든 기능을 검색할 시 아래의 띄게 하여 클릭 시 해당 기능으로 이동한다.**
* **마이 페이지 버튼: 클릭 시 학사 관리내에 개인정보 조회 구역으로 이동하게 된다.**

**메인 구역**

* **프로필 구역: 자신의 이름, 신분, 메신저 조회, 로그인 시간, 등이 표시되는 구역**
* **공지사항 구역: 직원들이 작성한 공지사항들이 화면에 출력되는 구역으로 클릭할 시 게시판의 공지사항 영역으로 이동된다.**
* **퀵 메뉴 구역: 학사 일정 채용 정보들과 같은 다른 사이트로 이동할 수 있는 창으로 클릭할 시 해당 웹사이트로 이동한다.**
* **강의 시간표 구역: 해당 학생 및 교수가 오늘 진행되는 수업에 대한 일정이 표시되는 구역**
* **메인 기능 구역: 휴학 및 복한 신청, 수강 신청, 성적조회, 강의평가 등 학생들이 가장 많이 사용하는 기능들로 클릭할 시 해당 기능이 있는 웹이지로 이동한다.**
* **뉴스 구역: 직원 및 교수들이 게시판에 올려놓은 최신 뉴스들이 표시되는 구역으로 클릭할 시 게시판의 뉴스영역으로 이동한다.**
* **학사 일정 구역: 학사일정의 데이터에서 해당 달의 일정이 표시되는 구역이다.**
* **커뮤니티 구역: 해당 학과에서 개설된 자유게시판의 최신 글 목록이 표시되며 클릭할 시 자유게시판으로 이동한다.**
* **공고 구역: 각종 소식과 공모전, 채용 공고 등 학생들에게 필요한 공고들이 표시되는 구역이다.**

**14. 기능 설계**

**1.41 수업 관리 설계**

**학생 명단 조회: 등록 테이블에서 해당 수업을 신청한 학생들의 목록들을 반환받아 이를 화면 상에 출력한다.**

**출석 (입력/수정): 사용자가 교수일 경우에만 가능하다. 수업 인터페이스에서 출석 창에 들어가 현재 날짜를 클릭 학생목록을 출력되며 학생목록 옆에 결석시간이 표시되며 결석시간을 입력을 등록 버튼을 클릭 시 출석테이블의 각 학생들의 해당 수업의 출석데이터가 삽입된다. 만일 수정하고자 할 경우 수정 버튼을 클릭한 후 결석시간들을 수정하면 된다.**

**출석 조회: 신청 수업 목록 또는 담당 수업 목록에서 해당 수업을 클릭 후**

**출석 탭에서 각 날짜의 출석데이터를 조회할 수 있다.**

**성적 (입력/수정/조회): 학생은 메인 사이트에서 성적 조회를 클릭하여 자신이**

**신청한 수업의 성적을 조회할 수 있으며 교수 및 조교는 수업관리에서 성적 창**

**에서 각 학생들의 성적을 조회할 수 있으며 사용자의 신분이 담당교수일 경우**

**에만 입력 및 수정 가능하며 성적 탭에서 출력된 학생들의 성적을 입력을 할**

**수 있으며 입력한 성적은 성적테이블의 저장된다.**

**과제 (입력/수정/삭제): 수업 창에서 과제 탭 클릭한 뒤에 과제 생성을 클릭할 시 과제생성 사이트로 이동하여 과제 기한, 내용 등을 작성한 후 등록을 클릭하여 과제를 생성을 할 수 있으며 수정에 경우에는 해당 과제로 들어가 수정을 클릭하여 해당 과제 글을 수정할 수 있다. 과제등록이 완료될 시 과제테이블의 레코드가 등록되며 수정 버튼을 클릭할 시 get방식으로 과제레코드의 id를 과제 사이트로 전달하며 과제 사이트로 id를 통해 레코드의 정보들을 불러와 이를 과제 생성창에 출력하여 내용을 수정하여 등록 버튼을 클릭하실 레코드가 update 되게 하여 구현한다.**

**과제 제출(입력/수정/삭제): 학생은 과제창에서 교수가 만든 과제를 제출할 수 있으며 이때 파일 입력이 가능하다. 과제를 제출할 시 과제 제출 테이블에서 칼럼이 만들어지며 제출시 등록한 파일의 경로는 과제 제출 테이블의 file\_path멤버의 저장된다.**

**14.2 수강 신청 설계**

**신청 가능 수업 목록(조회): 수업 테이블에서 현재 신청가능한(수업 학기가 현재학기)수업들을 반환 받아서 화면상에 출력한다. 학생이 수강 신청사이트에 들어올 경우 현재 수업테이블에서 현재 학기에 진행되는 수업(2025 1학기 일경우 where=2025.03.04)때 수업레코드를 반환 받고 수업 클래스로 받아 이를 출력하는 방식으로 구현한다.**

**수강 신청(입력/수정/조회): 학생들은 메인 인터페이스에서 수강신청사이트로 이동할 수 있으며 수강신청사이트에서 신청가능 한 수업들을 보고 신청 버튼을 클릭하여 수업을 신청할 수 있으며 이때 같은 시간의 수업들은 신청할 수 없다. 신청 버튼을 클릭 시 신청 클래스안에 수업 클래스가 들어가게 하여 장바구니 버튼 클릭 시 현재 자신이 신청 클래스안에 수업들이 출력되게 한다. 신청 완료 버튼클릭 시 신청 클래스에 내용이 등록테이블에 레코드로 저장되게 되며 수정 버튼 클릭 시 등록 테이블 상에 레코드를 현재 신청 클래스에 내용으로 update하게 한다.**

**수강 정원 추첨(삭제): 수강 신청 날짜가 끝나실 신청버튼 및 수정버튼을 비활성화하며 정원이 넘어간 수업들에 학생들을 추첨해서 랜덤으로 정원을 뽑는다.**

**시간표(조회): 자신이 수강사이트에서 현재 시간 표 조회 클릭 시 현재 신청 목록**

**옆에 현재 신청한 수업들로 구성된 시간표를 출력한다. 사용자가 수업을 신청할경우 이를 현재 신청클래스에 저장하며 시간표 조회 버튼을 클릭 시 신청 클래스의 수업들을 월,화,수,목,금으로 되어 있는 시간표로 출력한다.**

**수업 검색: 수강 신청 사이트에서 검색 창에 원하는 수업을 입력할 시 입력한 문자열과 비슷한 수업들에 목록을 DB에서 반환 받아 보여주며 검색할 때 전공, 교양, 학점 등 범위를 지정해서 검색할 수 있게 한다.**

**수업 개설(입력/수정/삭제): 교수와 강사들은 수강사이트에서 강좌 개설 메뉴가 존재하며 해당 창에서 강의 계획서, 수업시간, 정원 등을 작성하며 작성이 완료될 시 등록 버튼을 클릭하여 수업을 개설되며 개설된 수업은 수강사이트에서 수정을 통해 수업 내용들을 수정할 수 있다. 수업 등록버튼 클릭 시 작성한 내용(수업제목, 학과, 강의 계획서, 수업시간, 정원)들을 수업 테이블에 레코드로 생성한다.**

**14.3 메신저 설계**

**메시지 전송 기능(입력/조회/삭제): 사용자들은 메인 인터페이스 상에 팝업으로 작게 형성되어 있는 메신저 창에서 원하는 사용자를 클릭하여 개인 채팅방을 개설하여서 해당 채팅방에서 메세자를 보낼 수 있다. 이때 보낸 메시지는 메시지 테이블에 저장되게 되게 되며 채팅방에서 보낸 메시지는 시간 순서 대로 정렬되어 보여 지게 된다. 삭제 클릭 시 메시지 테이블에서 해당 레코드가 삭제되며 해당 채팅방을 다시 불러오게 하여 삭제된 메세지가 보여 지지 않게 한다.**

**메시지 수신 기능(조회): 자신이 속한 채팅방에서 새로운 메시지가 송신되었을 경우 메신저에서 새로운 메시지가 왔다는 것을 알람으로 보여줘야 한다. 사용자가 송신했을 때 해당 채팅방에 소속된 모든 사용자의 채팅방을 초기화 시키고 새로운 메시지가 존재할 경우 색상을 다르게 하여 새로운 메시지가 왔다는 알린다.**

**14.4 데이터 관리 설계**

**자신 정보 정보(수정/조회): 사용자들은 프로필 구역에 개인정보 조회를 클릭하여 자신의 개인정보 및 인적사항(생년월일, 이름, 학과, 연락처, 지도교수, 계좌번호)을 조회할 수 있으며 수정버튼을 클릭 시 연락처, 주소, 계좌번호를 수정할 수 있다. 사용자의 신분을 조회하며 학생일 경우 인적사항의 학년, 반, 재학상태가 표시되며 교수, 조교일 경우 담당 반, 강의실 등이 표시된다. 개인정보 조회 사이트로 들어갈 시 메인 사이트에서 신분과 학번을 get방식으로 전달하며 전달받은 신분에 맞는 테이블(학생, 교수, 직원, 조교)에서 테이블 해당 학번에 레코드를 반환받아서 해당 레코드에 내용을 출력한다. 수정 버튼을 클릭 시 연락처, 주소, 계좌번호가 input태크로 변환되며 수정 완료 버튼 클릭 시 레코드를 업데이트를 진행한다.**

**휴학 및 재학 처리(수정): 개인 정보 조회 사이트에 휴학 및 재학 처리 탭에서 휴학 신청을 클릭 시 휴학을 신청했습니다. 팝업창이 나온다. 이후 직원 사용자가 휴학 및 재학처리 탭에서 이를 수락하실 학생 레코드의 재학상태가 휴학으로 업데이트 된다.**

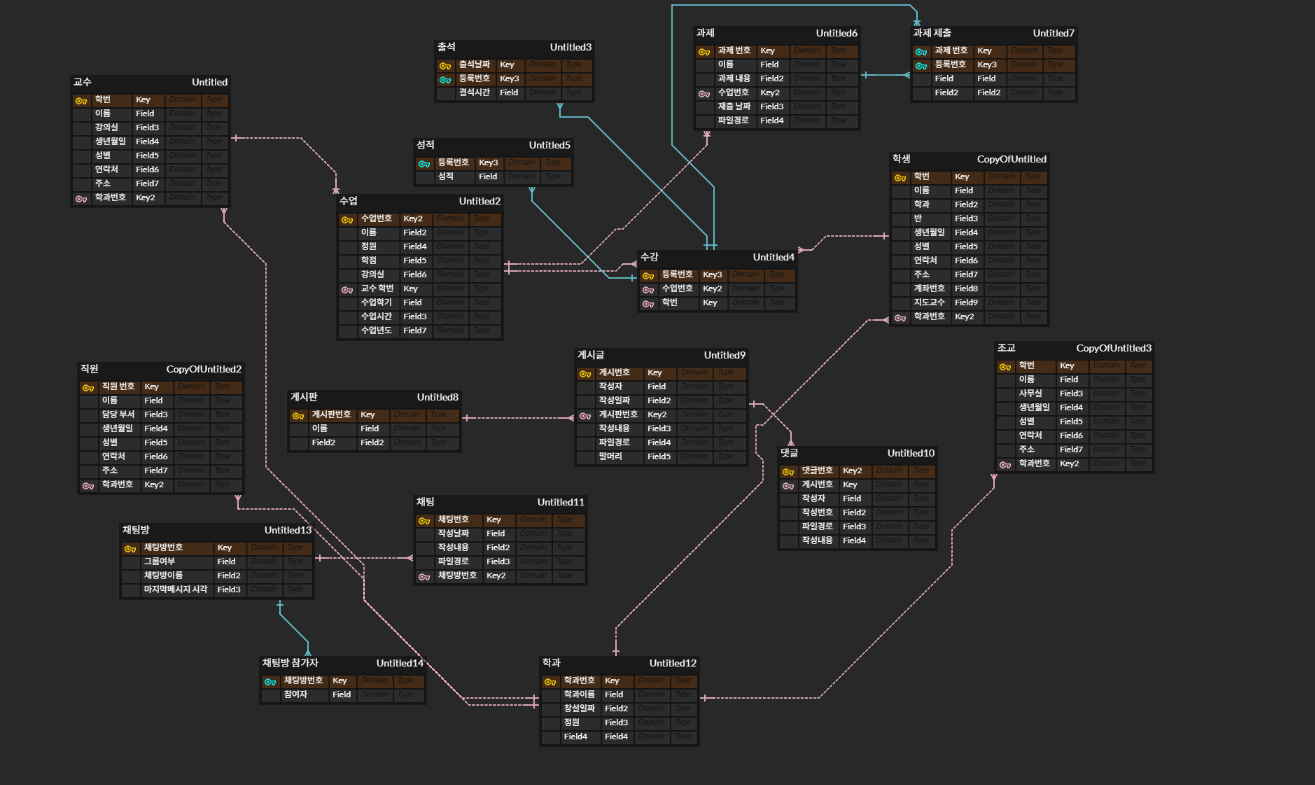
**14.5 커뮤니티 설계**

**게시판 개설(입력/수정/삭제): 직원 및 조교가 게시판 개설을 할 수 있으며 커뮤니티 사이트에서 게시판 생성을 통해 게시판을 생성할 수 있다. 게시판이 생성되면 게시판 테이블의 칼럼이 생성되게 된다.**

**게시글 조회: 커뮤니티 사이트에서 작성된 게시글 목록에서 특정 게시글을 클릭시 해당 게시글로 이동하며 작성된 글을 볼 수 있게 한다. 게시글 테이블에서 해당 게시판(fk)같은 칼럼들의 제목, 작성자, 작성일자를 반환 받아서 칼럼 20개마다 한페이지를 구성하게 하며 각 페이지의 글을 클릭 시 get방식을 통해 게시글 id를 전달하여 게시글 조회사이트로 이동 get으로 받은 id와 동일한 칼럼을 게시글 테이블에서 반환 받아 게시글에 제목, 작성내용, 첨부 파일이 출력 되게 하여 구현한다.**

**글 작성(입력/수정/삭제): 사용자가 커뮤니티 사이트에서 클릭 게시글 작성 사이트로 넘어가며 등록 버튼을 클릭 시 게시글이 작성되며 사진 버튼을 클릭 시 파일을 업로드를 하여 이를 출력하게 한다. 등록 버튼 클릭 시 게시글의 칼럼이 생성되게 하며 파일 업로드는 input 태그의 file타입으로 해서 받아 파일자체는 내부 board\_file폴더 내부에 저장하며 파일의 경로를 게시글 테이블의 파일 경로 칼럼에 저장되어 db에서 조회가 가능하게 한다. 글 삭제 버튼을 클릭 시 게시글의 파일경로를 이용해 board\_file폴더에 파일을 삭제하며 파일 삭제가 완료될 시 테이블레코드를 삭제하고 사용자를 게시글 조회사이트로 이동시킨다.**

**15. DB설계(초기안)**

****

**교수 테이블 설계**

**칼럼구조**

**학번(PK, 정수형) 제약조건 IDENTITY check 학번 between 0 and 99999로 최대 5자리까지 가능하게 해야 한다.**

**이름(문자형) 제약 조건 NOT NULL varchar(50)로 최대 50자리까지 가능하게 한다.**

**학과번호(fk, 정수형) 학과테이블의 fk이다. 제약조건 NOT NULL**

**생년월윌(날짜형) 형식은 0000년00월00일 제약조건 NOT NULL**

**연락처(문자형) 형식은 000-0000-0000 제약조건 NOT NULL**

**주소(문자형) 제약조건 NULL 허용 varchar(50)로 최대 50자리까지 가능하게 한다.**

**성별(문자형) 제약조건 NOT NULL varchar(1)로 최대 1자리까지 가능하게 한다.**

**연구실(문자형) 자신의 랩실 또는 사무실 번호를 저장한다. 제약조건 NOT NULL**

**계좌번호(문자형) 형식은 0000-0000-0000-0000 제약조건 NULL 허용 char(43)로 43자리를 입력해야 한다.**

**역할 및 사용기능 조회 횟수**

**교수 테이블은 학교의 존재하는 교수의 정보를 저장하는 역할의 테이블이다.**

**사용기능은 과제관리, 출석관리 수강신청 등 다양한 기능에서 사용 될 예정이다.**

**출석, 성적, 과제등 다양한 기능에서 사용될 것으로 예상되기에 조회량이 많은 테이블 일것으로 보인다.**

**학과 테이블 설계**

**칼럼구조**

**학과번호(PK,정수형)**

**학과이름(문자형) 최대 100글자까지 가능하다. 제악조건 unique NOT NULL**

**창설일자(날자형) 학과가 개설된 날짜를 저장한다. 형식은 0000-00-00이다. 제약조건 NOT NULL**

**정원(정수형) 해당 학과 현재 재적 중인 학생수를 저장하는 칼럼이다. 학생 테이블에서 count를 통해 계산할 예정 제약조건 NOT NULL**

**역할 및 사용 기능, 조회 회수**

**학과 테이블은 현재 대학내에 존재하는 학과들에 정보를 저장하는 테이블로 현재 DB상**

**교수·학생·조교테이블이 학과에 소속된 정보를 참조할 때 모두 이 테이블을 FK로 하여 사용한다.**

**사용 기능은 수강신청, 개인정보관리 등에서 사용될 예정이다.**

**학과 정보는 교수 조회, 학생 조회, 수업 편성, 학사 행정(장학, 수강 인원 집계**

**등) 다수의 기능이 공통으로 참조하므로 조회 빈도가 매우 높을 것으로 보인다.**

**조교 테이블**

**칼럼구조**

**학번(pk, 정수형) 제약조건 IDENTITY check 학번 between 0 and 99999로 최대 5자리까지 가능하게 해야 한다.**

**이름(문자형) 제약 조건 NOT NULL varchar(50)로 최대 50자리까지 가능하게 한다.**

**사무실(문자형) 자신이 소속된 사무실의 번호를 저장한다. 제약 조건 NOT NULL varchar(50)로 최대 50자리까지 가능하게 한다.**

**생년월일(날짜형) 형식은 0000년00월00일 제약 조건 NOT NULL**

**연락처(문자형) 형식은 000-0000-0000이 제약 조건 NOT NULL**

**주소(문자형) 제약 조건 NULL 허용**

**성별(문자형) 제약조건 NOT NULL varchar(1)로 최대 1자리까지 가능하게 한다**

**학과번호(fk) 학과테이블의 fk이다. 제약 조건 NOT NULL**

**계좌번호(문자형) 형식은 0000-0000-0000-0000 제약조건 NULL허용 char(43)로 43자리를 입력해야 한다.**

**역할 및 사용 기능, 조회 횟수**

**조교 테이블은 학교의 소속된 조교의 정보를 저장하는 테이블이다.**

**사용 기능 개인 정보관리, 수업 시간 변경 등에서 사용될 예정이다.**

**조교가 게시판에 글을 올리거나 댓글을 작성할 때, 자신정보 조회 등 조교 ID를 기준으로 테이블 조회하지만 현재 대학내 조교 수가 그리 많지 않기에 조회량이 많지 않을 것으로 보인다.**

**직원 테이블**

**칼럼 구조**

**직원번호(fk,정수형) 제약조건 IDENTITY check 학번 between 0 and 99999로 최대 5자리까지 가능하게 해야 한다.**

**이름(문자형) 제약 조건 NOT NULL varchar(50)로 최대 50자리까지 가능하게 한다.**

**담당부서(문자형) 직원 현재 부서를 저장한다. 제약 조건 NOT NULL**

**생년월일(날짜형) 형식은 0000년00월00일 제약 조건 NOT NULL**

**성별(문자형) 제약조건 NOT NULL varchar(1)로 최대 1자리까지 가능하게 한다**

**연락처(문자형) 형식은 000-0000-0000 제약 조건 NOT NULL**

**주소(문자형) 제약 조건 NULL 허용**

**계좌번호(문자형) 형식은 0000-0000-0000-0000 제약조건 NULL허용 char(43)로 43자리를 입력해야 한다.**

**역할 및 사용 기능, 조회횟수**

**직원 테이블은 학교의 근무하는 직원의 정보를 저장하는 테이블이다.**

**사용기능 개인정보 조회, 휴학 처리 등에서 사용될 예정이다.**

**직원이 자신 정보 조회, 채팅방, 게시판 작성 등에서 직원 테이블을 조회하기에 조회량이 많을 것으로 보인다.**

**학생테이블**

**칼럼 구조**

**학번(fk,정수형) 제약조건 IDENTITY check 학번 between 0 and 99999로 최대 5자리까지 가능하게 해야 한다..**

**이름(문자형) 제약 조건 NOT NULL varchar(50)로 최대 50자리까지 가능하게 한다.**

**학년/반(문자형) 현재 해당 학생의 학년과 반에 대한 정보를 저장하는 칼럼으로**

**형식 3/A이다. 제약 조건 NOT NULL char(3)로 3자리를 입력해야 한다.**

**생년월일(날짜형) 형식은 0000년00월00일 제약 조건 NOT NULL**

**성별(문자형) 제약조건 NOT NULL varchar(1)로 최대 1자리까지 가능하게 한다**

**연락처(문자형) 형식은 000-0000-0000 제약조건 NOT NULL**

**주소(문자형) NULL 허용**

**계좌번호(문자형) 형식은 0000-0000-0000-0000 제약 조건 NULL 허용 char(43)로 43자리를 입력해야 한다**

**지도교수(FK) 교수 테이블의 FK이다. 제약조건 NOT NULL**

**학과 번호(FK) 학과테이블의 FK이다. 제약조건 NOT NULL**

**재학 여부(문자형) 재학, 휴학 재적 3개만 입력 가능하다. 제약조건 NOT NULL char(43)로 2자리를 입력해야 한다.**

**역할 및 조회 횟수**

**학생테이블은 현재 대학 내 학생의 정보를 저장하는 테이블이다.**

**사용 기능은 수강신청,출석관리,성적관리 등 다양한 기능에서 활용될 예정이다.**

**학생 테이블은 수강 신청, 채팅방, 게시판, 성적조회, 자신정보 조회 등 여러 기능에서 사용되며 학생 또한 많기에 가장 조회 횟수 높을 것이라고 보는 테이블이다.**

**수업테이블**

**칼럼 구조**

**수업번호(pk, 정수형) 제약조건 IDENTITY**

**수업이름(문자형) 제약조건 NULL varchar(50)로 최대 50자리까지 가능하게 한다.**

**수강인원(정수형) 제약 조건 NOT NULL**

**학점(정수형) 제약 조건 NOT NULL check 학점 between 0 and 9로 최대 1자리 입력 가능하다.**

**강의실(문자형) 제약 조건 NOT NULL**

**교수학번(fk) 교수테이블의 fk이다. 제약 조건 NOT NULL**

**수업년도(문자형) 수업이 진행된 년도룰 저장한다. 형식은 0000년 제약 조건 NOT NULL**

**수업학기(정수형) 수업인 진행된 학기를 저장하는 칼럼으로 1학기일시 1로 저장한다. 제약조건 최대 1자리가능하다. 제약 조건 NOT NULL**

**수업시간(정수형) 수업이 진행되는 시간으로 만일 3교시동안 진행될 시 3으로 저장한다. 제약조건 최소, 최대 1자리 가능하다. 제약 조건 NOT NULL**

**역할 및 사용 기능, 조회 횟수**

**교수 또는 강사가 수업진행 하는 수업정보를 저장하는 테이블이다.**

**사용기능으로는 수강신청기능,성적입력, 출석입력,과제관리 등에 사용될 예정이다.**

**여러 기능들에 사용될 예정인 테이블로 조회량이 많을 것으로 보인다.**

**수강테이블**

**칼럼구조**

**등록번호(pk,정수형) 학번,수업번호의 대리키로 제약조건 IDENTITY**

**수업번호(fk) 수업테이블의 fk이다. 제약 조건 NOT NULL**

**학번(fk) 학생 테이블의 fk이다. 제약 조건 NOT NULL**

**역할 및 사용 기능, 조회 횟수**

**학생 테이블과 수업테이블의 교차 테이블로 학생이 어떤 과목을 수강하는 지에 대한 정보를 저장하는 테이블이다.**

**사용 기능으로는 수강 신청 기능, 성적입력 및 조회, 출석입력 및 조회, 과제 제출 등에서 사용될 예정인 테이블이다.**

**학기 조, 학기 말에 조회 횟수가 많을 것으로 보인다.**

**성적테이블**

**칼럼구조**

**등록번호(PK,FK) 등록테이블의 FK이자 성적테이블의 PK이다.**

**성적(문자형) 제약조건 varchar로 최대2글자까지 입력 가능하다. A, C-, B+ NOT NULL**

**역할 및 사용 기능, 조회 횟수**

**등록 테이블에 서브타입으로 학생이 수강했던 수업들의 성적을 저장하는 테이블이다.**

**사용기능으로 성적입력 및 조회에 사용될 예정이다.**

**성적기간에만 사용될 테이블이기에 특정기간에 조회 횟수가 집중적으로 높아질 것으로 보인다.**

**출석테이블**

**칼럼구조**

**출석날짜(PK, 날짜형)**

**등록번호(PK,FK) 등록테이블의 FK이자 성적테이블의 PK이다.**

**결석시간(정수형) 해당 일자 수업에서 결석한 시간을 저장하는 칼럼이다. 제약조건 check -1<결석시간 AND 결석시간<99, NOT NULL**

**출석일자 와 등록번호를 묶어 복합 PK로 지정한다. 이를 통해 같은 학생이 같은 날짜에 출석 레코드를 두 개 이상 생성하는 것을 방지할 수 있다.**

**역할 및 사용 기능, 조회 횟수**

**등록 테이블에 서브타입으로 학생이 수강했던 수업들의 출석시간을 저장하는 테이블이다.**

**사용기능으로 출석 입력 및 조회에 사용될 예정이다.**

**매 수업마다 출석을 입력하기에 조회 횟수는 전채적으로 높은편은 아니지만 지속적인 조회가 있을것으로 예상된다.**

**과제 테이블**

**칼럼구조**

**과제번호(PK, 정수형) 제약조건 IDENTITY**

**과제이름(문자형) 제약조건 varchar(100)으로 해서 최대 글자 100자리로 제한한다. Not null**

**과제 내용(문자형)** **과제 등록 시 과제 설명, 요구사항 등에 내용을 저장하는 칼럼이다. 제약조건 not null로 만약 입력안할 시 백엔드에서 “”으로 저장한다.**

**수업 번호(FK, 정수형) 수업테이블의 FK이다. 제약 조건 Not null**

**등록일자(날짜형) 형식은 0000-00-00이다.**

**파일경로(문자형) 과제등록 할 때 입력한 파일의 경로를 저장하는 칼럼이다. 제약조건 null 허용 제약조건 varchar(300)으로 해서 최대 글자 300자리로 제한한다**

**역할 및 사용 기능, 조회 횟수**

**수업에서 과제가 만들어질 시 이를 등록하기 위한 테이블이다.**

**사용기능은 과제 등록 및 수정 기능에 사용될 예정이다.**

**입력은 그리 많지 않을 것으로 예상하여 학생들이 주기적으로 조회하기에 조회 횟수 또한 그리 높지 않을 것으로 예상한다.**

**과제제출 테이블**

**과제번호(PK,FK,정수형) 과제테이블의 FK이다.**

**등록번호(PK,FK,정수형) 등록테이블의 FK이다.**

**제출이름(문자형) 제약조건 NOT NULL varchar(100)으로 해서 최대 글자 100자리로 제한한다.**

**파일경로(문자형) 과제등록 할 때 입력한 파일의 경로를 저장하는 칼럼이다. 제약조건 null 허용 제약조건 varchar(300)으로 해서 최대 글자 300자리로 제한한다.**

**제출내용(문자형) 과제를 제출할 때 설명, 절할 말 등 내용을 저장하는 칼럼이다.**

**제약 조건 not null로 입력안할 시 백엔드에서 “”으로 저장한다**

**과제번호 와 등록번호를 묶어 복합 PK로 지정한다. 이를 통해 같은 학생이 과제를 두 개 이상 제출하는 것을 방지할 수 있다.**

**역할 및 사용 기능, 조회 횟수**

**과제 제출 테이블은 어떤 학생이 어떤 과제를 제출했는지에 관한 정보를 테이블이다.**

**사용기능은 과제 제출 및 제출 수정 기능에 사용될 예정이다.**

**학생들이 “제출 여부 확인”을 위해 가끔 조회하고 교수·가 “제출 목록 확인”을 주기적으로 조회하지만 상대적으로 조회량이 많은 편은 아닐 것으로 예상한다.**

**게시판 테이블**

**칼럼구조**

**게시판 번호(pk,정수형) 제약조건 IDENTITY**

**게시판 이름(문자형) 제약조건 NOT NULL varchar(100)으로 최대 100자리 가능하게 한다.**

**게시글 수(정수형) 현재 게시판에 등록된 게시글의 개수를 저장하는 칼럼이다.**

**제약조건 NOT NULL**

**역할 및 사용 기능, 조회 횟수**

**게시판 테이블은 게시글을 저장하는 게시판에 정보를 저장하는 테이블이다.**

**사용기능으로 게시글 등록, 게시글 수정, 게시글 삭제 등을 진행할 때 게시글 테이블과 크로스 조인할 예정이다.**

**게시판 목록을 보여줄 때마다 조회되므로 조회 횟수가 높을 것으로 예상한다.**

**게시글 테이블**

**칼럼 구조**

**게시번호(PK,정수형) 제약조건 IDENTITY**

**작성자(문자형) 제약조건 NOT NULL varchar(50)로 최대 50자리까지 가능하게 한다.**

**작성일자(날짜형) 게시글이 작성된 날짜를 저장하는 칼럼이다. 제약조건 NOT NULL 형식은 0000년00월00일이다.**

**게시판 번호(FK) 게시판 테이블의 FK로 현재 게시글이 작성된 게시판을 조회하는 용도이다.**

**게시글제목(문자형) 제약조건 NOTNULL varchar(100)로 최대 100자리까지 가능하게 한다.**

**게시글내용(문자형) 제약조건 NOTNULL TEXT로 최대 글자 수 제한이 없게 한다.**

**파일경로(문자형) 입력한 파일의 경로를 저장하는 칼럼이다. 제약조건 null 허용 제약조건 varchar(300)으로 해서 최대 글자 300자리로 제한한다**

**댓글허용 여부(문자형) 게시판에 게시글의 댓글을 작성여부를 저장하는 칼럼으로**

**제약조건 NOT NULL varchar(1)로 최대 1자리까지 가능하며 t,f두개만 입력 가능하다.**

**역할 및 사용 기능, 조회 횟수**

**게시글 테이블은 게시판에서 등록한 게시글을 저장하는 테이블이다.**

**사용기능은 게시글 작성, 게시글 수정, 게시글 삭제, 게시글 조회 등에 사용될 예정이다.**

**사용자들이 게시글을 읽는 순간마다 지속적으로 조회가 발생하기 때문에 조회 빈도가 높을 것으로 예상한다.**

**댓글 테이블**

**칼럼 구조**

**댓글 번호(PK,정수형) 제약조건 IDENTITY**

**작성자(문자형) 제약조건 NOT NULL varchar(50)로 최대 50자리까지 가능하게 한다.**

**파일경로(문자형) 입력한 파일의 경로를 저장하는 칼럼이다. 제약조건 null 허용 제약조건 varchar(300)으로 해서 최대 글자 300자리로 제한한다**

**작성내용(문자형) 제약조건 NOT NULL varchar(100)로 최대 100자리까지 가능하게 한다.**

**상위댓글(FK,정수형) 자신 테이블을 참조하는 FK이다. 제약조건 NULL 허용**

**상위 댓글을 자신 테이블을 참조하는 FK로 이는 댓글에 댓글을 달 경우 상위댓글을 조회하기 위한 칼럼이다.**

**역할 및 사용 기능, 조회 횟수**

**댓글은 각 게시글(Post)에 달린 댓글(및 대댓글) 정보를 저장하는 테이블입니다.**

**사용기능은 댓글 작성, 게시글 조회에서 사용될 예정이다.**

**댓글 테이블 또한 사용자들이 게시글을 읽는 순간마다 지속적으로 조회가 발생하기 때문에 조회 빈도가 높을 것으로 예상한다.**

**채팅방테이블**

**칼럼구조**

**채팅방번호(PK,정수형) 제약조건 IDENTITY**

**채팅방 이름(문자형) 제약조건 NOT NULL varchar(30)로 최대 30글자까지 하게 한다.**

**그룹여부(문자형) 제약조건 NOT NULL varchar(1)로 최대 1글자까지 가능하게 하며 t,f 두개만 입력 가능하다.**

**마지막 메시지시각(날짜형) 제약조건 NULL 허용 형식은 0000년 00월 00일**

**역할 및 사용기능 조회횟수**

**채팅방 테이블은 메신저에서 현재 사용자들이 존재하는 채팅방 정보를 저장하는 테이블이다.**

**사용 기능으로 채팅방 개설, 메시지 조회, 메시지 관리 등에서 사용될 예정이다.**

**메신저가 사용횟수가 높다고 보기에 조회량이 많을 것으로 예상한다.**

**채팅방 참여자테이블**

**칼럼구조**

**채팅방 번호(PK,FK,정수형) 채팅방 테이블의 FK이자 본 테이블에 PK이다.**

**없게 한다.**

**참여자(문자형) 해당 메시저를 보낸 작성자가 저장되는 칼럼이다. 제약조건 NOT NULL varchar(50)로 최대 50글자까지 가능하다.**

**역할 및 사용기능 조회 횟수**

**채팅방참여자 테이블은 채팅방(ChatRoom)에 속한 참여자목록을 관리한다.**

**사용 기능으로 채팅방 개설,채팅방 관리, 메시지 조회, 메시지 관리 등에서 채팅방테이블에서 크로스 조인으로 사용될 예정이다.**

**메신저가 사용횟수가 높다고 보기에 조회량이 많을 것으로 예상한다.**

**메시지 테이블**

**칼럼구조**

**메시지번호(PK,정수형) 제약조건 IDENTITY**

**채팅방번호(FK) 채팅방테이블의 FK이다.**

**작성날짜(날짜형) 제약조건 NOT NULL 형식은 0000년00월00일**

**작성내용(문자형) 제약조건 NOT NULL TEXT로 최대 글자 수 제한이 없게 한다.**

**파일경로(문자형) 입력한 파일의 경로를 저장하는 칼럼이다. 제약조건 null 허용 제약조건 varchar(300)으로 해서 최대 글자 300자리로 제한한다**

**작성자(문자형) 메시지를 작성한 작성자를 저장하는 칼럼이다 제약조건은 NOT NULL**

**역할 및 사용기능 조회 횟수**

**메시지 테이블은 각 채팅방(ChatRoom)에서 주고받는 모든 메시지를 저장하는 테이블이다.**

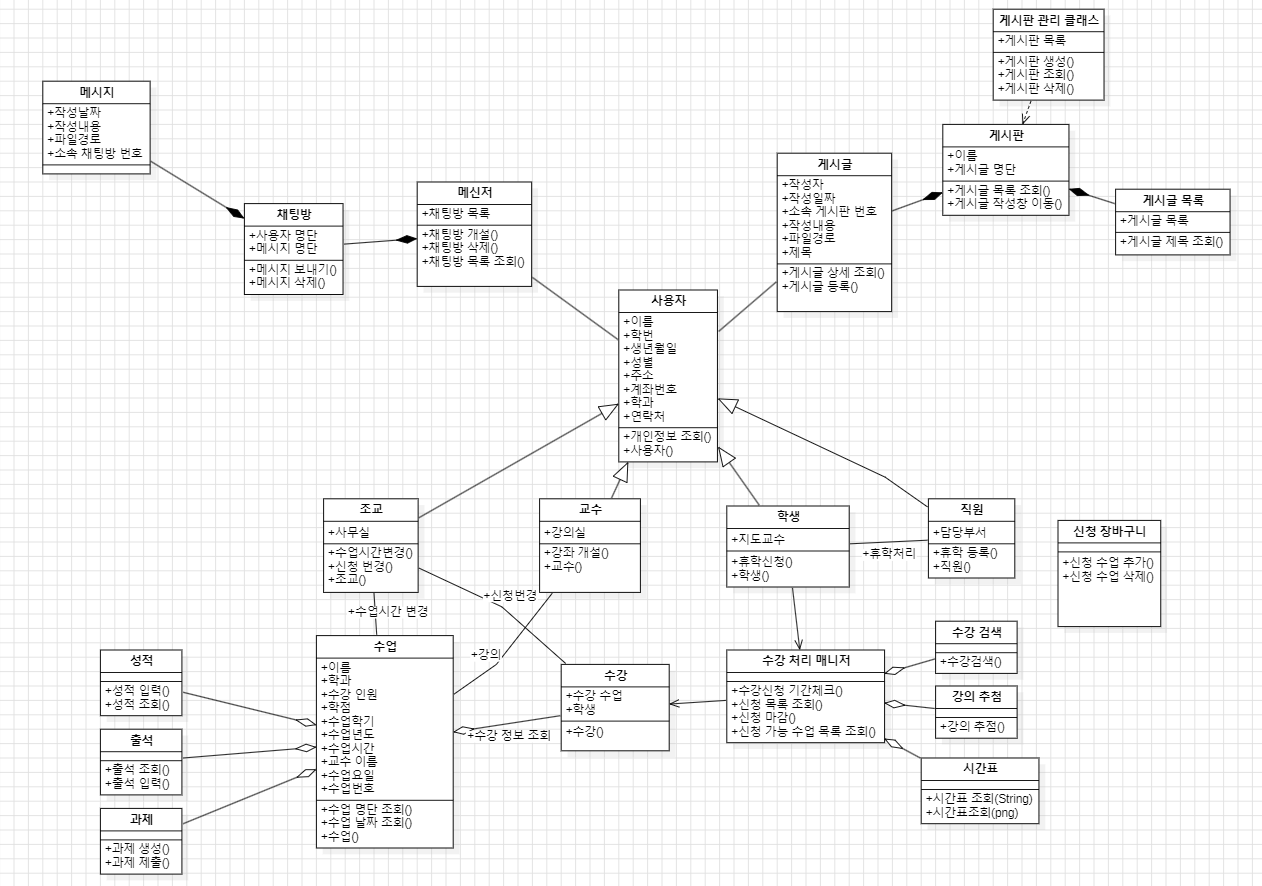
**사용 기능으로 메시지 조회, 메시지 관리 등에서 사용될 예정이다.**

**메신저가 사용횟수가 높다고 보기에 조회량이 많을 것으로 예상한다.**

**인덱스 설계**

**18. 클래스 설계**

**클래스 다이어그램**

****

**사용자 클래스**

**필드 멤버**

**이름(String)**

**학번(int)**

**생년월일(String)**

**성별(String) 형식은 “남성”, “여성”**

**주소(String)**

**계좌 번호(String) 형식은 0000-0000-0000-0000**

**학과(String)**

**연락처(String) 형식은 000-0000-0000**

**메서드**

**생성자(학번):  
주어진 학번을 기반으로 사용자에 신분에 맞는 테이블테이블에서 관련 정보를 조회하여 멤버 변수를 초기화한다.**

**개인정보조회():   
현재 사용자 객체 자신을 반환하는 메서드로, 호출자가 사용자 정보를 확인할 때 사용한다.**

**역할 로직 구성**

**사용자 클래스는 현재 클라이언트의 정보를 저장하는 기본 클래스이며, 이 클래스를 상속받아 교수, 학생, 직원 등 여러 하위 클래스가 존재한다.**

**로그인 시 입력된 학번과 비밀번호가 일치하면, 사용자의 신분에 맞는 포털 사이트로 이동하며 이때 POST 방식으로 학번을 전달한다.**

**포털 사이트로 이동 시, 학번을 바탕으로 사용자 객체가 생성되며, 이를 기반으로 세션을 생성한다.**

**포털 사이트 내부에서는 사용자 객체의 정보를 특정 구역(예: 구역1)에 출력하여 사용자 정보를 표시한다.**

**교수 클래스**

**필드 멤버**

**강의실(string)**

**직급(string)**

**메서드**

**생성자(학번):  
학번을 기반으로 데이터베이스에서 테이블 정보를 조회하여 멤버 변수를 초기화한다.**

**강의개설(수업이름(string), 정원(int),학점(int),강의실(string), 교수학번(int),수업학기(string),수업 시간(int) 수업 년도(string)):   
강의개설 메서드 실행 시 매개변수 바탕으로 수업클래스를 생성하고 이 수업 클래스의 속성 값으로 수업테이블의 칼럼을 생성한다.**

**역할 및 로직 구성**

**교수 클래스는 사용자 클래스의 하위 클래스로, 교수 신분을 나타낸다.**

**교수는 수강신청사이트에서 강좌개설 버튼을 클릭 시, 폼 창으로 이동한다.**

**해당 폼창에서 수업 관련 정보를 입력할 시, 백엔드에서 교수 클래스의 강의개설 메서드를 실행한다.**

**학생 클래스**

**필드(멤버 변수):**

**지도교수 (String)**

**학년 (int)**

**반 (String)**

**메서드:**

**생성자(학번):   
학번을 기반으로 데이터베이스에서 학생 정보를 조회하여 멤버 변수를 초기화한다.**

**휴학신청(휴학 사유(string)):   
휴학신청 사유와 학번, 이름을 직원에게 전송하며 휴학 테이블에 해당 정보를 등록한다.**

**역할 및 로직 제어:**

**학생 클래스는 사용자 클래스의 하위 클래스로, 학생 신분을 나타낸다.**

**수강신청 사이트에서 GET 방식으로 학번을 받아 학생 객체를 생성하며, 이를 바탕으로 학생이 신청할 수 있는 수업 정보를 수강신청 사이트에 출력한다.**

**학생은 휴학 신청 버튼 클릭 시 휴학 신청 사유 입력 창으로 이동하며, 등록 버튼 클릭 시 입력된 사유와 학번이 휴학 테이블에 저장된다.**

**(보류중)**

**조교 클래스**

**필드 멤버**

**사무실(string)**

**메서드**

**수강 변경(학생 학번(int), 수업번호(int),)**

**직원 클래스**

**필드 멤버**

**담당부서(string)**

**진급(string)**

**메서드**

**휴학등록(학생 학번(int), 승인 여부(boolean)):   
매개변수로 받은 값을 기반으로 해당 학생의 휴학테이블의 레코드에 승인 여부를 칼럼을 업데이트한다.**

**역할 및 로직 처리**

**직원 클래스는 사용자 클래스의 하위 클래스로, 직원 신분을 나타낸다.**

**직원이 게시판으로 이동할 경우, 공지사항 게시판에 등록, 수정, 삭제 버튼이 표시된다.  
다른 신분의 사용자에게는 해당 버튼들이 표시되지 않도록 한다.**

**직원은 휴학 등록 페이지에서 학생들이 신청한 휴학 내역(승인 여부칼럼이 미결 상태인 레코드)을 확인할 수 있다.  
각 신청 내역 옆에는 승인과 불허 버튼이 표시되며, 버튼 클릭 시 휴학등록 메서드가 호출되어 해당 신청의 승인 여부를 처리한다.**

**수업 클래스**

**필드(멤버 변수):**

**이름 (String)**

**학과 (String)**

**수강 인원 (String)**

**수업 년도 (String, 형식: “0000년”)**

**수업 시작일 (String)**

**수업 학기 (String)**

**수업 시간 (int)**

**교수 이름 (String)**

**수업 요일 (String)**

**수업 번호 (int, DB 테이블의 수업 번호 칼럼과 대응)**

**메서드:**

**생성자(수업 번호):  
수업 번호를 기반으로 수업테이블에서 수업 정보를 조회하여 이를 기반으로 객체를 초기화한다.**

**학생 명단 조회():  
수강 테이블에서 수업 번호를 기준으로 해당 수업을 수강하는 학생 명단을 조회하여 반환한다.**

**수업 시간 조회():   
수업 시작일을 기준으로 16주간의 수업 날짜를 배열 형태로 반환한다.**

**역할 및 로직 제어:**

**수업 클래스는 수업 테이블에 저장된 수업 정보를 관리하기 위한 클래스이다.**

**교수가 수업 항목을 클릭하면, 교수 학번을 기준으로 수업 테이블에서 해당 수업 정보를 조회하여 수업 클래스를 생성한다.**

**출석 클래스**

**메서드:**

**출석 입력(수강객체(Enrollment), 결석시간(int), 출석날짜(string)):  
주어진 매개변수를 기반으로 출석 테이블에 레코드를 생성하거나 기존 레코드를 수정한다.**

**출석 조회(학번, 수업, 출석 날짜):  
해당 출석 날짜를 기준으로 출석 테이블에서 출석 정보를 조회하여 반환한다.**

**역할 및 로직 제어:**

**출석 클래스는 출석 관련 로직 처리를 담당한다.**

**교수가 “출석” 버튼을 클릭하면, 수업 객체의 수업 시간 조회() 메서드를 실행하여 각 수업 날짜를 화면 왼쪽 상단에 출력한다.**

**이후 수업 객체의 학생 명단 조회() 메서드를 실행하여 학생 클래스 배열을 반환받고, 이를 화면에 출력한다.**

**“출석 입력” 버튼 클릭 시 수강 클래스(학생, 수업) 배열, 결석 시간, 수업 날짜를 POST 방식으로 백엔드에 전달한다.**

**백엔드에서는 이를 받아 출석 입력 메서드를 통해 각각 출석 테이블에 레코드를 입력 또는 수정한다.**

**학생이 “출석 조회” 버튼을 클릭하면, 학생 번호를 기반으로 수업 클래스를 생성하여 현재 수강 중인 수업 목록을 출력한다.**

**수업 선택 시 수업 클래스의 날짜 조회 메서드를 통해 출석 일자 목록을 출력한다.**

**원하는 출석 일자를 클릭하면, 해당 일자를 기준으로 출석 조회 메서드가 실행되어 출석 정보를 화면에 출력한다.**

**과제 클래스**

**메서드**

**과제 생성(수업번호, 교수이름, 과제 제목, 과제 내용, 파일이름):  
매개변수를 기반으로 과제 테이블에 레코드를 생성한다.**

**과제 제출(학번, 수업번호, 제출제목, 제출내용, 파일경로):  
매개변수를 기반으로 과제 제출 테이블에 레코드를 생성한다.**

**역할 및 로직 처리**

**과제 클래스는 과제 기능 관련 로직 처리를 담당하는 클래스이다.**

**교수가 “과제 생성” 버튼을 클릭하면 과제 게시판으로 이동하며, “생성” 버튼 클릭 시 과제 생성 폼으로 이동한다.**

**“등록” 버튼 클릭 시 입력된 내용을 인수로 하여 과제 생성() 메서드를 실행하고, 과제 테이블에 저장한다.**

**학생이 과제 제출 버튼을 클릭하면 과제 제출 창으로 이동하며, 제출 완료 시 입력한 정보와 학생 및 수업 정보를 인수로 하여 과제 제출() 메서드를 실행하여, 과제 제출 테이블에 저장한다.**

**성적 클래스**

**메서드**

**성적 입력(수강객체(Enrollment), 성적(string)):  
매개변수를 기반으로 성적 테이블에 레코드를 생성하거나 기존 레코드를 수정한다.**

**성적 조회(학번, 수업):**

**매개변수를 기반으로 성적 테이블에서 성적 정보를 조회하여 반환한다.**

**역할 및 로직 제어**

**성적 클래스는 성적 관련 로직 처리를 담당하는 클래스이다.**

**교수가 “성적” 버튼을 클릭하면, 수업 객체의 학생 명단 조회() 메서드를 실행하여 학생 클래스 배열을 반환받고 이를 화면에 출력한다.**

**이후 “성적 입력” 버튼 클릭 시 해당 학생 및 수업 정보를 바탕으로 성적 입력 메서드를 실행하여 성적 테이블에 저장하거나 수정한다.**

**학생이 “성적 조회” 버튼을 클릭하면, 학번을 기반으로 수업 클래스를 생성하여 현재 수강 중인 수업 목록을 출력한다.  
수업을 선택하면 학번과 수업 정보를 이용해 성적 조회 메서드를 실행하고, 반환받은 성적 정보를 화면에 출력한다.**

**수강 클래스**

**필드(멤버 변수)**

**학생번호 (int): 학생의 학번을 저장한다.**

**수업번호 (int): 해당 학생이 수강한 수업의 번호를 저장한다.**

**메서드**

**생성자(수업번호: int, 학생번호: int):  
전달받은 매개변수를 기반으로 수강 객체를 초기화한다.**

**역할 및 로직 처리**

**수강 클래스는 학생의 수강 정보를 저장하는 클래스로, 성적, 과제, 출석, 수강신청 등 다양한 로직에서 처리 단위로 사용된다.**

**수강 처리 매니저 클래스**

**필드 멤버**

**마감날짜(String): 신청 마감일자. 형식은 "0000년-00월-00일".**

**메서드**

**생성자(수업년도: String, 수업학기: int):  
주어진 수업년도와 수업학기를 기반으로 마감테이블에서 마감날짜를 조회하여 초기화한다.**

**신청가능수업목록조회(수업년도: String, 수업학기: int):  
현재 년도와 학기를 기반으로 수업테이블을 검색하여, 신청 가능한 수업 클래스 배열을 반환한다.**

**마감설정(마감날짜: String):  
관리자만 호출 가능한 메서드로, 신청 마감일자를 설정하고 마감테이블에 해당 값을 저장한다.**

**신청마감():  
현재 마감일자를 반환한다.**

**역할 및 로직 처리**

**수강 처리 매니저 클래스는 수강신청 관련 주요 로직을 담당하는 클래스이다.**

**학생이 포털사이트에서 "수강신청"을 클릭하면 수강 처리 매니저 클래스가 생성된다.**

**이후 수강 처리 매니저 클래스의 신청가능 수업목록 조회 메서드를 실행하여, 신청 가능한 수업 목록을 출력한다.**