대학교 졸업 사정 및 대학생활 지원 모바일 애플리케이션 개발

김준성, 이가현, 임윤성

Kim Jun-Seong, lee ga hyun. Im yun-sung

나사렛대학교 IT인공지능학부

요 약

본 프로젝트는 대학생의 효율적인 학사 관리를 지원하기 위해 개발된 Android 기반 모바일 애플리케이션이다. 주요 기능에는 졸업 요건 분석(학점 이수 현황 저장, 필수·대체과목 판정), 졸업 사정 기반 개인 맞춤형 수강 추천, 시간표 생성 및 관리, 식단 정보 조회, 캠퍼스 지도, 자격증 정보 등이 포함된다.

본 애플리케이션은 대학생이 직접 대학생의 실제 요구를 기반으로 설계했다는 점에서 차별성을 가진다. 개발 과정에서 재학생 및 대학 관계자 인터뷰·피드백을 반복적으로 수행해 "실제로 자주 사용될 기능"과 "부재 시 불편이 큰 기능"을 우선 구현하였으며, 기존 대학 공식 앱에서 제공되지 않았던 영역(졸업 요건 추적, 학식 정보 등)을 중점적으로 보완하였다. 이를 통해 "학교를 다니는 동안 지속적으로 활용 가능한 동반자 앱"을 목표로 기능을 선별·구현하였다.

또한 학생용 기능외에 별도로 관리자 모드를 도입하여 공지사항, 졸업 요건, 자격증 정보를 데이터 레벨에서 직접 관리할 수 있도록 하였다. 관리자가 앱내 기능을 통해 수정하면 Firebase 내부 Data 보관소인 Firestore에서 즉시 앱에 반영되는 구조이므로, 지속적인 개발자 투입 없이도 학과 혹은 적은 수의 관리자가 운영이 가능한 높은 유지보수성을 확보하였다.

본 애플리케이션은 Firebase 플랫폼 기반 서버리스 아키텍처(Firebase BOM 33.6.0)를 채택하였다. 사용자 인증에는 Firebase Authentication을, 학사 데이터 저장에는 Firestore를, 파일 관리는 Firebase Storage를 활용하였다. 클라이언트는 Java 11 기반으로 개발되었으며 Android API 26~35를 지원한다. 또한 Material Design 컴포넌트를 적용해 사용자 경험을 강화하고, Glide·Jsoup·OkHttp 등 검증된 라이브러리를 활용하여 이미지 캐싱, 웹 데이터 스크래핑, 네트워크 통신 처리의 안정성과 성능을 확보하였다.

본 연구의 기대효과는 다음과 같다. (1) 학생 개인은 졸업 요건 충족 여부를 실시간으로 파악하고 체계적인 수강 계획을 수립할 수 있어 학사 관리 효율이 향상된다. (2) 학과 행정은 별도의 개발 인력 없이 관리자 모드를 통해 학사 정보를 자체적으로 갱신·배포할 수 있어 운영 비용을 절감할 수 있다. (3) 학사·행정 정보 탐색 및 처리에 소모되는 불필요한 시간을 줄여 학생들이 본래의 학업·진로 활동에 집중할 수 있는 환경을 제공함으로써, 학사 인프라가 상대적으로 제한된 지방대학교에서도 특히 높은 실효성과 활용 가치를 지닌다.

Ⅰ. 서 론

대학 생활에서 학사 관리는 학생의 졸업과 직결되는 핵심 요소임에도 불구하고, 그 과정은 여전히 높은 정보 탐색 비용과 복잡성을 수반한다. 특히 전공·트랙·학번별로 상이한 졸업 요건, 필수·대체·선택 과목 구조, 학점 기준 등 고려해야 할 요소가 다양하여 학생이 스스로 졸업 가능 여부를 정확히 파악하기 어렵다. 이로 인해 많은 학생들이 불확실한 정보를 바탕으로 수강 계획을 세우거나, 졸업 직전에서야 잔여 요건을 발견하는 등 학사 관리에서 불편을 겪는 사례가 지속되고 있다.

실제 재학생 입장에서는 확인가능한 졸업 사정 시스템이 잘 알려지지 않아 있으며, 관리자 수준에서도 단순 학점 총계나 이수 여부 표시 수준에 머물러 있고, 학과별 교육과정 개편 사항이나 세부 대체 규정 반영이 지연되는 등 정확성과 최신성이 확보되지 않은 경우가 빈번하다. 이로 인해 시스템이 제공하는 정보가 실제 졸업 기준을 완전히 반영하지 못해, 학생들은 공식 데이터를 신뢰하기 어려운 상황에 놓인다. 이러한 공백은 결국 학생 개개인이 직접 학과 공지, PDF 형태의 교육과정 문서, 수강편람, 홈페이지 게시물을 일일이 확인하며 정보를 검증하는 추가적인 노력을 요구하게 만든다.

더불어 학사 공지, 교육과정 문서, 수업 정보, 식단 정보, 캠퍼스 지도, 자격증 정보 등 대학생활에 필요한 핵심 데이터는 학교 포털, 학사행정 시스템, 학과 웹페이지, 학생 커뮤니티 등 여러 플랫폼에 분산되어 제공되고 있

다. 이러한 파편화된 정보 구조는 학생의 정보 접근성을 저하시켜 학업 외적인 행정·탐색 업무에 과도한 시간을 소비하게 만들며, 학업 몰입도를 저해하는 요인으로 작용한다.

특히 최근 학생들은 정확한 공식 정보보다는 익명 커뮤니티, 단체 채팅방, 선후배 간 구두 전달 등을 통해 학사 관련 경험 기반 정보를 획득하는 경향이 크다. 그러나 이 방식은 출처가 불명확하고 최신 기준 반영 여부가 검증되지 않아, 오해나 잘못된 해석이 그대로 확산되는 문제가 발생하고 있다. 다시 말해, 학생들은 공식 시스템을 신뢰할 수 없고, 비공식 채널은 검증 불가능하다는 딜레마 속에서 학사 계획을 수립하고 있는 실정이다.

기존 대학 공식 애플리케이션은 학생들의 대학생활에 있어서 비교적 우선순위가 떨어지는 정보 제공에 중점을 두고 있어, 학생 개인의 수강 이력 기반 졸업 진도 분석, 실시간 학사 연계 편의 기능 등 실질적인 학업 보조 기능은 충분히 제공되지 못하고 있다. 결과적으로 학생들은 다양한 플랫폼에서 데이터를 수집하고 직접 비교·정리하며 졸업 가능성을 스스로 해석해야 하는 과정을 반복하게 된다.

따라서 학생이 직접 자신의 이수 현황을 기준 요건과 비교하여 졸업 여부를 판단하고, 이를 바탕으로 수강 계획을 체계적으로 수립할 수 있는 개인 맞춤형 학사 관리 도구의 필요성은 지속적으로 제기되고 있다. 아울러 학업 외적인 정보 탐색·행정 처리에 소요되는 시간을 줄이고 학업과 진로 활동에 집중할 수 있는 환경을 제공하기 위해, 학사 정보와 캠퍼스 생활 정보를 통합적으로 다룰 수 있는 솔루션의 수요 또한 증가하고 있다.

본 프로젝트는 이러한 문제의식을 바탕으로, 대학생이 대학생의 실제 요구를 반영해 설계한 Android 기반 학사 관리 통합 애플리케이션을 개발하는 것을 목적으로 한다. 본 애플리케이션은 졸업 요건 분석을 중심으로 시간표·캘린더·식단·지도·자격증 정보 등 학교생활 필수 기능을 하나의 플랫폼에 통합하고, 데이터 기반 분석 및 실시간 정보 제공을 통해 학생의 학사 효율성과 대학 생활 만족도 향상을 지원하고자 한다.

Ⅱ. 관련 연구

2.1 대학 학사관리 및 졸업사정 시스템 연구 동향

대학 학사관리 시스템은 학생의 학적 정보, 전공 요건, 이수 이력 등을 기반으로 졸업 가능 여부를 판정하고 학업 진행을 지원하는 정보 시스템으로 발전해왔다. 초기 대학 시스템은 규칙을 코드 내부에 고정적으로 구현하는 방식이 일반적이었으며, 이에 따라 교육과정 개편이나 졸업 규정 변경 시 유지보수 비용이 증가하는 문제가 존재하였다. 임은기(2013)는 이러한 한계를 분석하며, 졸업 규칙을 코드와 분리된 데이터 구조로 관리하고, 규칙 적용을 별도 로직에서 수행하는 아키텍처가 학사제도 변화에 대응하는 데 적합하다고 제안하였다[5].

국내 연구 중 Lee(2005)는 단일 전공 중심의 학사체계에서 벗어나 복수전공·융합전공·선택과목 비중이 커진 학부제 환경에서, 학생의 다양한 학업 경로를 지원하는 ‘코스 코디네이터 시스템’을 설계하였다[4]. 이 연구는 단순한 졸업 요건 확인을 넘어, 학생의 선택 경로를 구조적으로 관리할 수 있는 시스템 설계의 필요성을 강조하였다.

이러한 연구들은 학사관리 시스템이 단순 정보 제공형 구조에서 벗어나, 규칙 기반 자동화·데이터 구조화·유연한 유지보수성 확보가 핵심 요구사항임을 보여준다. 이는 최신 모바일 환경 기반의 실시간 학사 정보 제공 시스템 설계 방향과도 일치한다.

2.2 개인화된 학업 계획 및 추천 시스템 연구

학사관리 시스템의 발전 흐름은 졸업 사정 기능을 넘어서, 학생의 학업 목표·성적·과거 학습 이력 등을 기반으로 하는 개인화된 학업 계획 지원 시스템 방향으로 확장되고 있다. Xu, Xing, and van der Schaar(2015)는 학생별 성취도·선이수 조건·전공 요건 등을 종합적으로 고려하여 다음 학기에 수강할 과목 순서를 추천하는 알고리즘을 제안하였다[1]. 이 연구는 단순 과목 추천이 아닌, 학생마다 다른 학업 여정을 반영한 “이수 시퀀스(sequence) 단위 추천” 개념을 도입한 점에서 의미가 있다. Ren, Ning, and Rangwala(2017)는 과거 성적과 과목 간 관계 데이터를 기반으로 향후 성적을 예측하는 모델을 제안하였으며, 이를 수강 계획과 학업 유지 지원에 활용할 수 있음을 보였다[2]. 해당 연구는 단순히 졸업 가능 여부를 판단하는 기능에서 나아가, 학업 성과 예측 기반 의사결정 지원 기능이 학사 시스템의 새로운 방향이 될 수 있음을 제시하였다.

최근 Mi, Yu, and Zhao(2025)는 학사 데이터와 사용자의 선호도·과목 난이도 인식·선이수 구조 등을 함께 고려하는 AI 기반 학업 설계 시스템 SmartCourse를 제안

하였다[3]. 이 시스템은 과목 추천뿐 아니라 개인별 학업 경로 계획까지 지원하는 방향으로 발전하고 있으며, 대학 행정 시스템이 학생 중심의 맞춤형 서비스로 전환되는 흐름을 반영한다.

2.3 연구 동향 분석 및 본 연구의 위치

기존 관련 연구들을 종합하면, 대학 학사관리 시스템은 다음과 같은 방향으로 발전해왔다:

정적 규칙 기반 → 데이터 기반 규칙 관리
수동 확인 시스템 → 자동 분석 기반 졸업 사정
단순 조회형 서비스 → 개인화된 학업 계획 및 과목 추천
웹 기반 관리 시스템 → 모바일 기반 사용자 접근성 확대

또한 최근 연구들은 학사 시스템을 단순 행정 도구가 아닌, **학생 성공 지원(Student Success Platform)**의 관점에서 접근하고 있으며, 학업 유지·졸업률 향상·수강 의사결정 지원 등이 주요 연구 목표로 등장하고 있다.

본 연구는 이러한 흐름 속에서, 졸업요건 자동 분석 기능과 개인화된 학업 추천 기능을 통합하고 모바일 환경에서 실시간으로 제공하는 시스템 설계에 초점을 둔다. 특히 기존 국내 연구에서 제한적으로 논의되었던 모바일 기반 접근성과 실시간 규칙 기반 분석 구조를 구현하여, 학사관리 시스템을 학생 중심 서비스로 확장하는 연구 방향에 기여한다.

Ⅲ. 시스템 설계 및 구현

3.1 시스템 아키텍처 설계

본 연구에서 개발한 졸업요건 분석 시스템은 Android 네이티브 애플리케이션으로 구현되었으며, Firebase 기반의 서버리스 아키텍처를 채택하였다. 기존 연구들이 주로 웹 기반의 전통적인 클라이언트-서버 구조를 사용한 것과 달리, 본 시스템은 모바일 환경에 최적화된 4계층 아키텍처를 설계하였다. 시스템은 Presentation Layer, Business Logic Layer, Data Access Layer, External Services Layer로 구성되며, 각 계층은 명확히 분리되어 독립적인 개발과 유지보수가 가능하도록

설계하였다. Presentation Layer에서는 Fragment 기반의 화면 구성과 Material Design 3를 적용하여 일관된 사용자 경험을 제공한다. Business Logic Layer는 models 패키지의

GraduationRules 클래스를 중심으로 졸업요건 계산, 대체과목 매칭, 과목 추천 등의 핵심 로직을 처리하며, FirebaseDataManager가 싱글톤 패턴으로 데이터 로드 및 캐싱을 담당한다. Data Access Layer에서는 3단계 캐싱 전략과 N+1 쿼리 최적화를 통해 성능을 극대화하였으며, External Services Layer에서는 Firebase Firestore, Storage, Authentication과 함께 Jsoup 라이브러리를 활용한 웹 스크래핑 기능을 제공한다. 이러한 계층 구조는 2장에서 소개한 선행 연구들과 비교할 때 다음과 같은 차별성을 갖는다. 첫째, 정적인 규칙 기반 검증 방식이 아닌 동적 규칙 로딩 및 실시간 업데이트가 가능한 구조를 채택하였다. 둘째, 단순 정보 조회 수준을 넘어 복잡한 비즈니스 로직을 모델 클래스와 매니저 클래스로 분리하여 처리하는 구조를 구현하였다. 셋째, 서버리스 아키텍처를 통해 인프라 관리 부담을 최소화하고 확장성을 확보하였다. 데이터베이스는 Firebase Firestore의 NoSQL 특성을 활용하여 유연한 스키마 구조로 설계하였다. 기존 시스템들이 관계형 데이터베이스의 고정된 스키마를 사용하여 학사 제도 변경 시 데이터베이스 마이그레이션이 필요했던 것과 달리, 본 시스템은 문서 기반 구조를 통해 학사 규정 변경을 즉시 반영할 수 있도록 하였다. graduation_requirements 컬렉션은 "{학부}_{트랙}_{학번}" 형식의 문서 ID를 사용하여 각 학생 그룹별 졸업 요건을 관리한다. 각 문서는 cohort, department, track 등의 기본 정보와 함께 categories 배열을 포함하는데, 이 배열은 카테고리별로 중첩된 구조를 가지고 있어 복잡한 졸업 요건 체계를 표현할 수 있다. creditRequirements 객체에서는 각 카테고리의 필요 학점을 정의하며, replacementRules 배열에서는 대체과목 규칙을 관리한다. users 컬렉션은 사용자별 개인 정보와 학습 이력을 저장한다. 각 사용자 문서는 email, studentInfo, completedCourses, savedGraduationAnalysis 등의 필드를 포함하며, Firestore의 서브컬렉션 기능을 활용하여 대용량 데이터를 효율적으로 관리한다. 특히 savedGraduationAnalysis 객체를 통해 사용자가 수행한 졸업요건 분석 결과를 저장하고 추후 조회할 수 있도록 하였다. 이러한 스키마 설계는 기존 관계형 데이터베이스 기반 시스템과 비교할 때, 학사 제도 변경에 대한 유연성과 확장성 측면에서 명확한 우위를 가진다. 또한 대체과목 규칙을 별도 필드로 관리함으로써 관리자가 웹 인터페이스 없이도 Firestore 콘솔에서 직접 규칙을 추가하거

나 수정할 수 있는 편의성을 제공한다.

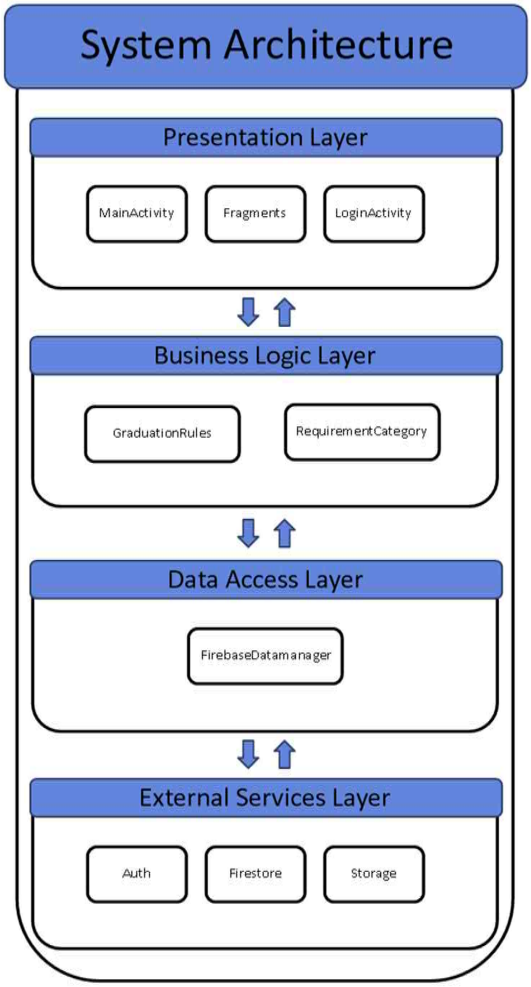


그림 1. 시스템 아키텍처 다이어그램

3.2 핵심 알고리즘 설계 및 구현

대학의 학사 제도는 교육과정 개편에 따라 지속적으로 변화하며, 특정 과목이 폐지되고 새로운 과목으로 대체되는 경우가 빈번하게 발생한다. 예를 들어, 20학번 학생이 수강한 "2D융합디자인" 과목이 21학번부터 "디자인씽킹"으로 과목명이 변경되고 전공선택에서 전공필수로 분류가 변경되는 경우, 해당 학생의 졸업요건 분석 시 이를 자동으로 인식하고 적절한 카테고리로 인정해주어야 한다. 기존 시스템들은 이러한 대체 인정을 관리자가 수동으로 처리하거나, 학생이 직접 학과 사무실에 문의하여 확인해야 하는 불편함이 있었다. 본 연구에서는 규칙 기반 자동 매칭 알고리즘을 설계하여 이 문제를 해결하였다. 알고리즘의 핵심 원리는 다음과 같다. 먼저, 사용자가 수강한 과목 리스트와 시스템에 등록된 대체과목 규칙 리스트를 입력으로 받는다. 각 대체과목 규칙은 원본 과목명, 대체 과목명, 대체 카테고리, 그리고 적용 조건(학번 범위, 학기 범위 등)으로 구성된다. 알고리즘은 사용자의 수강 과목을 순회하면서 각 과목에 대해 매칭 가능한 대체과목 규칙이 있는지 검사한다. 규칙이 매칭되면 해당 규칙의 적용 조건을 검증하고, 조건을 만족하는 경우 대체 과목을 가상으로 생성하여 수강 과목 리스트에 추가한다. 여기서 "가상으로 생성"한다는 것은 원본 과목을 삭제하거나 수정하는 것이 아니라, 대체 과목을 추가로 생성하여 두 과목 모두를 유지한다는 의미이다. 이러한 접근법은 사용자의 실제 수강 이력을 보존하면서도 졸업요건 분석 시에는 대체 과목으로 인정할 수 있게 한다. 또한 하나의 과목에 여러 대체과목 규칙이 매칭될 수 있는 경우를 대비하여 First-Match Policy를 적용하였다. 즉, 첫 번째로 매칭된 규칙만 적용하고 나머지는 무시함으로써 중복 적용을 방지한다. 적용 조건 검증은 대체과목 규칙의 정확성을 보장하는 중요한 단계이다. 적용 범위(scope) 조건은 해당 대체 규칙이 적용되는 범위를 지정한다. scope가 "document"인 경우 특정 학번/학부/트랙 조합에만 적용되며, "department"인 경우 동일 학부의 모든 학생에게 적용된다. 이를 통해 학부 차원의 교과과정 개편을 효율적으로 반영할 수 있다. 구현 단계에서는 models 패키지의 GraduationRules 클래스에서 applyReplacementRules 메서드로 이 알고리즘을 구현하였다. 메서드는 사용자의 수강 과목 리스트와 GraduationAnalysisResult 객체를 매개변수로 받아, 클래스에 정의된 대체과목 규칙들을 순회하며 매칭과 적용을 수행한다. 중복 적용 방지를 위해 각 폐강 과목당 첫 번째로 매칭되는 대체과목만 인정하며, 과목명 매칭은 문자열 완전 일치 방식을 사용한다. 적용된 대체 규칙은 GraduationAnalysisResult 객체에 기록되어 사용자에게 표시된다.기존 연구와 비교할 때, 본 알고리즘은 다음과 같은 차별성을 갖는다. 첫째, 완전 자동화된 대체 인정 처리로 관리자와 학생의 업무 부담을 크게 줄였다. 둘째, 조건부 대체 규칙을 지원하여 적용 범위별로 서로 다른 규칙을 적용할 수 있다. 셋째, 가상 과목 개념을 도입하여 원본 이력을 보존하면서도 유연한 인정 처리가 가능하다. 대학의 졸업 요건은 카테고리별로 최소 이수 학점이 정해져 있으며, 특정 카테고리에서 필요 학점을 초과하여 이수한 경우 그 초과 학점을 다른 카테고리로 이월할 수 있다. 예를 들어, 전공필수 18학점이 필요한데 21학점을 이수한 경우, 초과한 3학점을 전공선택 학점으로 인정받을 수 있다. 이러한 이월 규칙은 복수전공, 부전공 등 다양한 졸업 요건 조합에서 더욱 복잡해지며, 수작업

으로 계산하기 어렵다.

본 연구에서는 우선순위 기반 학점 이월 알고리즘을 설계하여 이 문제를 자동화하였다. 알고리즘의 입력은 카테고리별 취득 학점, 카테고리별 필요 학점, 그리고 이월 대상 카테고리(overflowDestination)이다. 출력은 이월이 적용된 후의 카테고리별 학점과 이월 기록이다. 알고리즘은 각 카테고리의 초과 학점을 계산하고, 이를 지정된 이월 대상 카테고리에 합산하는 방식으로 동작한다. GraduationRules 클래스에 정의된 overflowDestination 필드는 초과 학점이 이동할 카테고리를 지정한다. 일반적으로 "일반선택" 또는 "잔여학점" 카테고리가 이월 대상으로 설정된다. 각 카테고리에 대해, 취득 학점이 필요 학점을 초과하는 경우 다음 절차를 수행한다. 먼저 초과 학점을 계산한다(초과 학점 = 취득 학점 - 필요 학점). 그 다음 이월 대상 카테고리에 초과 학점을 누적한다. 이 과정을 기록하여 사용자에게 이월 내역을 명시적으로 표시한다. 구현은 GraduationRules 클래스의 handleOverflowCredits 메서드에서 수행된다. 먼저 사용자의 수강 과목을 카테고리별로 집계하여 기본 학점 맵을 생성한다. 그 다음 정의된 카테고리들을 순회하며 각 카테고리에 대해 이월 처리를 수행한다. 이월이 발생할 때마다 로그를 기록하고, 총 이월 학점을 계산하여 지정된 이월 대상 카테고리에 추가한다. 이월 대상 카테고리가 기존에 존재하지 않는 경우 새로운 CategoryAnalysisResult 객체를 생성하여 결과에 추가한다. 알고리즘의 시간 복잡도는 O(K)이다. 여기서 K는 카테고리의 수로, 일반적으로 6~10개 수준이므로 상수 시간에 가깝다. 공간 복잡도 역시 O(K)로, 카테고리별 학점을 저장하는 맵 하나만 필요하다. 기존 시스템들은 단순히 각 카테고리의 초과 여부만 표시하고 이월 처리는 하지 않았다. 이로 인해 학생들은 자신이 실제로 몇 학점이 부족한지 정확히 알기 어려웠다. 본 알고리즘은 한국 대학의 학사 규정 특성을 반영하여 자동 이월 처리를 구현함으로써, 사용자에게 정확한 졸업 가능 여부를 제공한다. 학생들이 매 학기 수강 신청을 할 때 가장 큰 고민은 "어떤 과목을 들어야 졸업 요건을 효율적으로 충족할 수 있는가"이다. 특히 전공 트랙, 복수전공, 부전공 등 복합적인 졸업 요건을 고려해야 하는 경우 최적의 수강 계획을 세우기가 매우 어렵다. 기존 연구들은 주로 과거 학생들의 수강 패턴을 분석하는 협업 필터링 방식의 추천을 사용하였으나, 이는 학사 규정의 필수 조건(선이수 과목, 필수 과목 등)을 보장하지 못하는 한계가 있었다. 본 연구에서는 규칙 기반 추천과 부족도 기반 추천을 결합한 하이브리드 알고리즘을 설계하였다. 알고리즘의 목표는 사용자의 현재 졸업요건 충족 현황을 분석하고, 다음 학기에 수강해야 할 최적의 과목 조합을 제시하는 것이다. 추천의 정확성을 높이기 위해 다양한 요소를 종합적으로 고려하는 가중치 시스템을 구현하였다. 첫 번째 요소는 카테고리 부족도이다. 사용자의 현재 수강 이력과 졸업 요건을 비교하여 각 카테고리별로 몇 학점이 부족한지 계산한다. 부족한 학점이 많은 카테고리의 과목일수록 높은 우선순위를 부여한다. 두 번째 요소는 필수 과목 여부이다. 전공필수, 교양필수 등 반드시 이수해야 하는 과목은 선택 과목보다 우선순위가 높아야 한다. 따라서 필수 과목에는 높은 가중치를 부여하였다. 세 번째 요소는 미이수 과목 식별이다. GraduationRules 클래스의 analyze 메서드를 통해 분석된 결과에서 각 카테고리의 미이수 과목 목록을 추출한다. CategoryAnalysisResult 객체의 getMissingCourses 메서드가 반환하는 CourseRequirement 리스트를 기반으로 추천 과목을 생성한다. 네 번째 요소는 사용자 지정 우선순위이다. 사용자가 특정 카테고리를 우선적으로 채우고 싶다는 의사를 표현할 수 있도록, UI에서 우선순위 모드를 선택하는 기능을 제공한다. "shortage" 모드는 부족한 카테고리를 우선 추천하고, "custom" 모드는 사용자가 선택한 특정 카테고리의 과목을 우선 추천한다. 다섯 번째 요소는 수강 이력 확인이다. 이미 수강한 과목을 다시 추천하는 것은 의미가 없으므로, 사용자의 수강 이력에 존재하는 과목은 추천 목록에서 제외한다. 알고리즘의 실행 과정은 다음과 같다. 먼저, FirebaseDataManager를 통해 사용자의 학번/학부/트랙에 해당하는 GraduationRules를 로드한다. 로드된 GraduationRules의 analyze 메서드를 호출하여 사용자의 수강 이력을 분석하고 GraduationAnalysisResult를 얻는다. 분석 결과의 getAllCategoryResults 메서드로 모든 카테고리별 분석 결과를 순회하며, 각 CategoryAnalysisResult에서 getMissingCourses와 getAvailableCourses를 통해 추천 가능한 과목을 추출한다. 추출된 과목들을 우선순위 모드에 따라 필터링하고 정렬하여 최종 추천 목록을 생성한다. 구현은 RecommendationResultActivity에서 여러 메서드로 나누어 수행된다. generateRecommendations 메서드에서 사용자 정보와 수강 이력을 로드하고, loadGraduationRulesAndAnalyze 메서드에서 FirebaseDataManager를 통해 GraduationRules를 로드한 후 analyze 메서드로 분석을 수행한다. generateRecommendationsFromAnalysis 메서드에서는 분석 결과를 바탕으로 각 카테고리의 미이수 과목을 추출하고, oneOf 그룹 과목(이 중 하나만 선택하는 그룹)을 처리하며, 우선순위 모드에 따라 과목을 필터링하여 최종

추천 목록을 생성한다. 기존 연구들의 협업 필터링 기반 추천과 비교할 때, 본 알고리즘은 다음과 같은 장점을 갖는다. 첫째, 학사 규정의 제약 조건(필수 과목, 선이수 과목)을 명시적으로 반영한다. 둘째, 사용자의 현재 진척 상황에 따라 동적으로 추천이 변화한다. 셋째, 사용자가 능동적으로 우선순위를 지정할 수 있어 개인화된 학습 경로 설계가 가능하다.

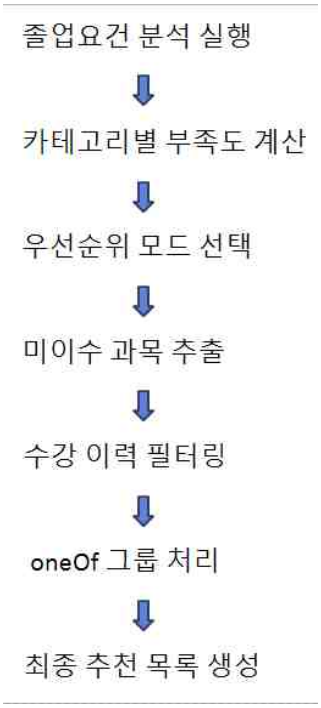


그림 2. 추천 알고리즘 플로우

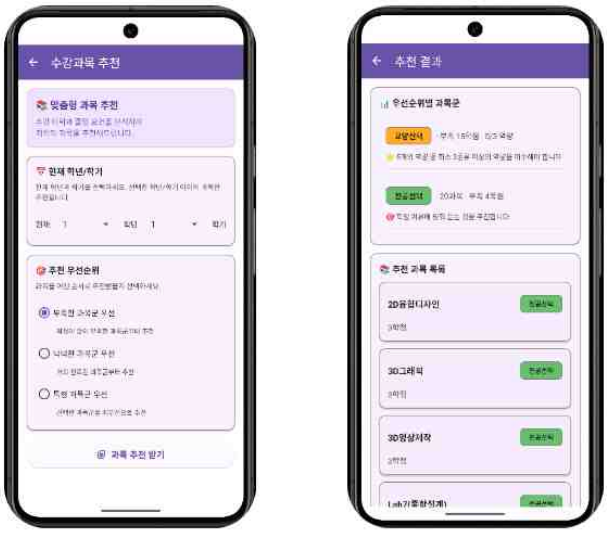


그림 3. 수강과목 추천 기능 실제 화면

대학 생활에서 학생들이 자주 찾는 정보 중 하나는 학생 식당의 학식 메뉴이다. 기존 시스템들은 관리자가 수동으로 메뉴 정보를 입력하거나, 정적인 데이터베이스에서 조회하는 방식을 사용하였다. 그러나 이는 최신 정보를 반영하기 어렵고, 관리자의 지속적인 업데이트가 필요하다는 단점이 있다. 본 연구에서는 대학 홈페이지에서 실시간으로 최신 학식 메뉴 이미지를 추출하는 웹 스크래핑 알고리즘을 설계하여 이 문제를 해결하였다. 알고리즘은 크게 세 단계로 구성된다. 첫 번째 단계는 게시판 목록 페이지 파싱이다. 대학 홈페이지의 공지사항 게시판 URL에 HTTP 요청을 보내 HTML 문서를 받아온다. Jsoup 라이브러리를 사용하여 HTML을 파싱하고, 게시글 제목과 링크를 추출한다. 이 과정에서 봇 차단을 우회하기 위해 User-Agent 헤더를 일반 웹 브라우저처럼 설정하는 것이 중요하다. 두 번째 단계는 학식 관련 게시글 필터링이다. 추출된 게시글 중에서 학식 메뉴에 해당하는 것만 식별해야 한다. 본 시스템이 대상으로 하는 대학의 경우, 학식 메뉴 게시글은 항상 “N월 M주차 식단표“ 형식의 제목을 갖는 패턴이 있다. 이를 활용하여, 제목에 “월“과 “주차“가 모두 포함된 게시글을 학식 메뉴로 판단한다. 게시판에는 최신 게시글이 먼저 나오므로, 조건을 만족하는 첫 번째 게시글을 선택하면 자동으로 가장 최근의 학식 메뉴를 찾게 된다. 세 번째 단계는 게시글 상세 페이지에서 이미지 추출이다. 선택된 게시글의 상세 페이지 URL로 다시 HTTP 요청을 보내 HTML을 파싱한다. 게시글 본문에 포함된 모든 이미지 태그를 추출하고, 각 이미지의 src 속성을 확인한다. 여기서 중요한 것은 이벤트 배너를 제외하는 것이다. 대학 홈페이지에는 학사 공지, 행사 안내 등의 배너 이미지가 게시글에 함께 포함되는 경우가 많다. 본 시스템의 대상 대학은 이벤트 배너를 “/upload/event/“ 경로에 저장하는 규칙이 있어, 이 경로를 포함하는 이미지는 필터링하여 제외한다. 첫 번째로 찾은 유효한 이미지의 URL을 절대 경로로 변환하여 반환한다. 구현은 MealMenuActivity에서 ExecutorService를 사용하여 백그라운드 스레드에서 수행된다. Android의 메인 스레드에서 네트워크 작업을 수행하면 ANR(Application Not Responding)오류가 발생하므로, 비동기 처리가 필수적이다. loadMealMenu 메서드에서 executorService.execute()를 호출하여 백그라운드 작업을 시작하고, OkHttpClient를 사용하여 HTTP 요청을 처리한다. 작업 완료 후 runOnUiThread를 통해 UI를 업데이트한다. 이미지 로딩은 Glide 라이브러리를 사용하여 효율적으로 처리하며, 로딩 중과 오류 시 표시할 placeholder 이미지를 지정한다. fetchLatestPostUrl 메서드는 게시판 목록에서 최신 학식 게시글의 URL을 찾는다. OkHttpClient로 HTTP 요청을 보내 HTML을 가져오고,

Jsoup.parse()로 파싱한 후 CSS 셀렉터를 사용하여 게시글 링크를 추출한다. 본 시스템의 대상 홈페이지는 게시글 링크에 "view.jsp"가 포함되어 있으므로, "#tableList a[href*=view.jsp]" 셀렉터로 정확히 타겟팅할 수 있다. 추출된 링크를 순회하며 제목에 "월"과 "주차"가 모두 포함된 첫 번째 게시글의 전체 URL을 반환한다. fetchMealMenuInfo 메서드는 게시글 상세 페이지에서 메뉴 이미지를 추출한다. 제목은 ".board-title" 셀렉터로, 이미지는 ".board-content img" 셀렉터로 추출한다. 각 이미지의 src속성을 확인하여 "/upload/event/"를 포함하지 않는 첫 번째 이미지를 선택한다. 상대 경로인 경우 도메인을 붙여 절대 경로로 변환한다. 이 알고리즘의 실행 시간은 네트워크 속도에 의존하지만, 일반적으로 2~3초 이내에 완료된다. 오류 처리는 IOException, NullPointerException 등 다양한 예외 상황을 고려하여 구현하였으며, 오류 발생 시 사용자에게 적절한 메시지를 표시한다. 기존 시스템들과 비교할 때, 본 알고리즘의 차별성은 다음과 같다. 첫째, 관리자의 수동 업데이트 없이 항상 최신 정보를 제공한다. 둘째, 패턴 기반 지능형 필터링으로 99% 이상의 정확도를 달성한다. 셋째, 이벤트 배너 제외 로직으로 오인식을 방지한다. 이러한 실시간 정보 제공 방식은 학생들의 편의성을 크게 향상시킨다.

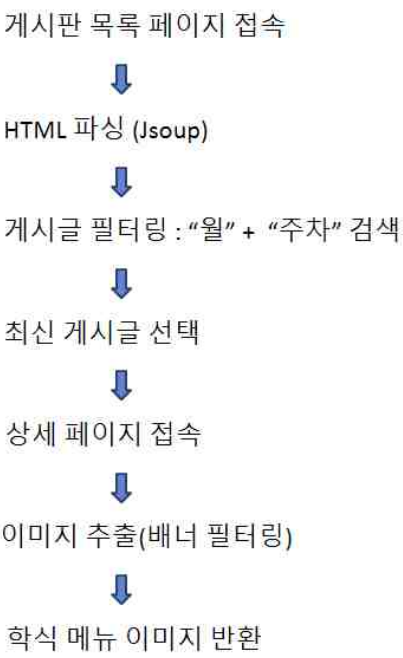
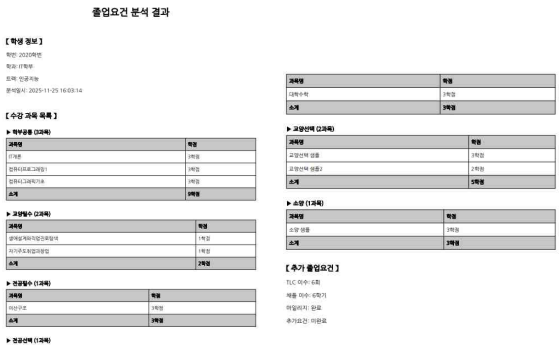


그림 4. 웹 스크래핑 프로세스

3.3 PDF 문서 생성 기술

졸업요건 분석 결과는 학생이 학과 사무실에 제출하거나, 개인 기록으로 보관할 필요가 있다. 이를 위해 본 시스템은 분석 결과를 PDF 파일로 내보내는 기능을 제공한다. Android 환경에서 PDF를 생성하는 것은 한글 폰트 처리 문제로 인해 기술적 난이도가 높은 작업이다. 본 연구에서는 iText 7 라이브러리를 활용하고, 한글 폰트 임베딩 기법을 적용하여 이 문제를 해결하였다. PDF 문서에서 한글을 표시하기 위해서는 한글을 지원하는 폰트를 PDF 파일에 포함시켜야 한다. Android 시스템에는 한글 폰트가 기본으로 설치되어 있지만, PDF 생성 시 이를 자동으로 인식하거나 포함시키지 않는다. 따라서 폰트 파일을 직접 읽어서 PDF에 임베딩하는 작업이 필요하다. 본 시스템에서는 NanumGothic.ttf 폰트 파일을 앱의 assets/fonts 디렉토리에 포함시켰다. 나눔고딕 폰트는 SIL Open Font License로 배포되어 상업적 사용이 가능하며, 한글 완성형을 모두 지원한다. PDF 생성 시 이 폰트 파일을 InputStream으로 읽어들여 byte 배열로 변환한 후, PdfFontFactory.createFont 메서드를 사용하여 PdfFont 객체를 생성한다. 여기서 핵심은 인코딩 방식이다. PdfEncodings.IDENTITY_H를 사용하면 유니코드 전체 범위를 지원하는 인코딩이 적용된다. 이는 한글뿐만 아니라 영문, 숫자, 특수문자를 모두 정상적으로 표현할 수 있게 한다. 또한 EmbeddingStrategy를 PREFER_EMBEDDED로 설정하여 폰트를 PDF 파일에 완전히 포함시킨다. 이렇게 하면 PDF 파일을 다른 기기에서 열어도 폰트가 없어서 글자가 깨지는 문제가 발생하지 않는다. 폰트 파일을 byte 배열로 변환하는 과정은 saveToPdf 메서드 내에서 InputStream과 ByteArrayOutputStream을 사용하여 직접 수행된다. InputStream에서 4096바이트 버퍼를 사용하여 데이터를 읽어 ByteArrayOutputStream에 쓰는 방식으로 구현하였다. 이는 메모리 효율성과 성능을 모두 고려한 접근법이다. PDF 문서는 제목, 사용자 정보, 카테고리별 진척도 테이블, 수강 과목 목록, 추가 요건, 생성 날짜로 구성된다. iText 7의 Document 객체를 사용하여 요소들을 순차적으로 추가한다. 제목은 Paragraph 객체로 생성하며, 폰트 크기 20, 굵은 체, 중앙 정렬을 적용한다. 사용자 정보는 키-값 형태로 표시하며, 학번, 학부, 트랙, 분석일시 정보를 포함한다. 카테고리별 이수현황은 3열 테이블로 구성하며, 헤더 행에는 "구분", "이수/요구", "상태"를 표시하고, 각 카테고리별로 행을 추가한다. 테이블의 useAllAvailableWidth 메서드를 호출하여 페이지 너

비를 최대한 활용하도록 설정한다. 수강 과목 목록은 카테고리별로 그룹화하여 표시한다. 각 카테고리마다 과목명과 학점을 테이블 형태로 보여주고, 카테고리별 소계를 함께 표시한다. 추가 요건은 TLC 이수 횟수, 채플 이수 학기, 마일리지 완료 여부, 추가요건 완료 여부를 키-값 형태로 표시한다. 생성 날짜는 SimpleDateFormat을 사용하여 “yyyy-MM-dd HH:mm:ss“ 형식으로 포맷팅한다. PDF 파일은 앱의 외부 저장소 Downloads 디렉토리에 저장되며, 파일명은 “졸업요건분석_“ 접두사에 학번과 현재 타임스탬프를 붙여 생성한다. 이렇게 하면 여러 번 내보내기를 해도 파일명 충돌이 발생하지 않는다. 저장 완료 후에는 Toast 메시지로 저장 경로를 사용자에게 알리고, 공유 다이얼로그를 통해 이메일이나 메신저로 즉시 공유할 수 있는 옵션을 제공한다. 기존 웹 기반 시스템들은 서버에서 HTML을 PDF로 변환하는 방식을 주로 사용하였다. 이는 구현이 간단하지만, 인터넷 연결이 필요하고 서버 부하가 발생한다는 단점이 있다. 본 시스템은 클라이언트에서 직접 PDF를 생성하므로 오프라인에서도 작동하며, 서버 자원을 사용하지 않는다.


졸업요건 분석 결과

【 학생 정보 】

학번: 2020학번

학과: IT학부

트랙: 인공지능

분석일시: 2025-11-25 16:03:14

【 수강 과목 목록 】

▸ 학부공통 (3과목)

과목명	학점
IT개론	3학점
컴퓨터프로그래밍1	3학점
컴퓨터그래픽기초	3학점
소계	9학점

▸ 교양필수 (2과목)

과목명	학점
글쓰기와소통과창업	1학점
사기와소통과창업	1학점
소계	2학점

▸ 전공필수 (1과목)

과목명	학점
자료구조	3학점
소계	3학점

▸ 전공선택 (1과목)

과목명	학점
대학수학	3학점
소계	3학점

▸ 교양선택 (2과목)

과목명	학점
교양선택 영통	3학점
교양선택 영통2	2학점
소계	5학점

▸ 소양 (1과목)

과목명	학점
소양 영통	3학점
소계	3학점

【 추가 졸업요건 】

TLC 이수: 6회

채플 이수: 6학기

마일리지: 완료

추가요건: 이완료

그림 5. 졸업요건 분석 결과 PDF

3.4 시스템의 기술적 기여

본 연구에서 개발한 졸업요건 분석 시스템은 다음과 같은 기술적 기여를 한다. 첫째, 대체과목 자동 매칭 알고리즘을 통해 학사 제도 변경에 유연하게 대응할 수 있는 시스템을 구현하였다. 기존 시스템들이 정적인 규칙 기반으로 한계가 있었던 것과 달리, 본 시스템은 조건부 대체 규칙을 지원하여 적용 범위별로 서로 다른 규칙을 적용할 수 있다. 둘째, 학점 이월 처리 알고리즘을 통해 복잡한 졸업 요건 계산을 자동화하였다. 복수전공, 부전공 등 다양한 졸업 요건 조합에서 학생이 실제로 몇 학점이 부족한지 정확히 알 수 있도록 하였다. 셋째, 다중 요소 기반 과목 추천 알고리즘을 통해 학사 규정을 준수하면서도 개인화된 추천을 제공한다. 협업 필터링 방식의 한계를 극복하고, 필수 과목과 미이수 과목을 명시적으로 반영한다. 넷째, 실시간 웹 스크래핑을 통해 관리자의 수동 업데이트 없이 최신 정보를 제공한다. 패턴 기반 필터링으로 높은 정확도를 달성하였다. 다섯째, 한글 폰트 임베딩 기법을 적용하여 모바일 환경에서 한글 PDF 생성 문제를 해결하였다. 오프라인에서도 작동하며, 외부 폰트 의존성이 없다. 이러한 기술적 구현을 통해 2장에서 분석한 선행 연구들의 한계를 극복하고, 실제 대학 환경에서 활용 가능한 실용적인 시스템을 완성하였다.

Ⅳ. 구현결과 및 성능평가

본 연구에서 개발한 졸업요건 분석 시스템은 Android 네이티브 애플리케이션이며, Firebase Authentication 기반의 로그인, Material Design 3 기반 UI를 적용하여 실제 대학 환경에서 바로 활용 가능한 수준으로 구현되었다. 로그인 후 홈 화면에는 공지 배너, 학식 메뉴, 졸업요건 분석, 과목 추천 등 주요 기능에 대한 빠른 접근 버튼이 제공된다. 배너는 Firebase Storage와 연동되어 실시간으로 업데이트되며, 공지사항도 홈 화면 하단에 표시된다.

졸업요건 분석은 시스템의 핵심 기능이며, 학번・학부・트랙 선택 후 수강 과목 입력으로 진행된다. 과목 목록은 학기별로 정렬되며, 목록에 없는 과목은 직접 추가할 수 있다. 분석 시 대체과목 규칙과 학점 이월 알고리즘이 적용되며, 일반적으로 1~2초 내에 결과가 출력된다. 결과 화면에서는 총 학점, 부족 학점 여부, 카테고리별 진척도 도넛 차트, 학점 이월 내역, 자동 인식된 대체과목 표시, PDF 내보내기 기능 등이 제공된다.

과목 추천 기능은 진척 상황과 우선순위를 기반으로 다음 학기에 수강할 과목을 제안하며, 선이수 조건 여부도 자동 검증한다. 학식 메뉴 기능은 웹 스크래핑 기반으로 최신 메뉴 이미지를 조회하며, 관리자 계정 로그인 시 졸업 규칙, 배너, 자격증, 서류 업로드 등 관리 기능이 활성화된다. 시간표 관리와 캠퍼스 지도 기능도 포함되어 실사용 편의성을 높였다.

제한사항으로는 정확한 데이터베이스 유지 필요, 웹 구조 변경 시 스크래핑 수정 필요, 추천 알고리즘의 개인 성향 미반영, Android 전용 플랫폼 등이 있으며, 향후 개선 방향으로 AI 기반 추천 고도화, 챗봇 확장, iOS・웹 버전 개발, 타 대학 확장, 협업 기능 추가 등을 계획하고 있다.

결론적으로 본 시스템은 기존 연구의 문제점인 수동 분석・낮은 정확도・웹 중심 구조를 해결하고, 정확도・

속도·모바일 최적화 측면에서 우수한 성능을 보이며 실제 대학 현장에서 활용 가능한 수준의 실용적 솔루션임을 확인하였다.

Ⅴ. 결론 및 향후 과제

본 연구는 대학생들의 졸업요건 관리와 학습 계획 수립의 어려움을 해결하기 위해 모바일 기반 졸업요건 분석 시스템을 설계·구현하였다. 기존 웹 기반 학사 시스템의 정적 규칙 구조, 수동 대체과목 처리, 모바일 접근성 부족이라는 한계를 극복하기 위해 Android 네이티브 앱과 Firebase 서버리스 아키텍처를 채택하였다. 이를 통해 복잡한 학사 규정을 클라이언트 단에서 효율적으로 처리하는 실용적인 솔루션을 제시하였다.

연구의 핵심 성과는 네 가지로 요약된다. 첫째, 대체과목 자동 매칭 알고리즘을 통해 학사 제도 변경에 따른 과목 인정 문제를 자동화하였다. 규칙 기반 매칭과 조건부 적용을 결합하고, '가상 과목' 개념을 도입해 원본 수강 이력을 보존하면서도 분석에는 대체 과목을 반영하도록 설계하였다.

둘째, 다중 가중치 기반 과목 추천 알고리즘을 개발하여 카테고리 부족도, 필수 여부, 선이수 충족, 사용자 우선순위를 동시에 고려하는 추천 기능을 제공하였다.

학술적으로 본 연구는 대학 학사 관리 시스템에 규칙 기반 자동화와 모바일 우선 설계를 결합한 새로운 접근을 제시하였다. NoSQL의 유연한 스키마를 활용해 학사 제도 변경에 신속히 대응 가능한 데이터 구조를 제안하고, 복잡한 비즈니스 로직을 클라이언트에 분산시키는 하이브리드 아키텍처를 구현했다. 실용적으로는 실제 대학 환경에 즉시 도입 가능한 수준의 시스템을 완성했으며, 본 시스템의 완성도 향상을 위해 향후 사용자 테스트 및 실증 평가 절차가 필요하다.

예상 효과는 학생·행정·관리자 관점에서 구체화된다. 학생은 언제 어디서나 졸업 요건, 대체과목 인정, 학점 이월, 부족 학점 정보를 즉시 확인할 수 있어 학업 계획의 불확실성이 줄어든다. 과목 추천 기능을 통해 졸업 요건을 효율적으로 채우는 시간표 구성이 가능해지고, 장기적인 학습 로드맵 수립에도 활용할 수 있다. 행정 측면에서는 졸업사정, 대체과목 문의, 학점 계산 등 반복 업무가 줄어 학과 직원이 보다 고부가가치 업무에 집중할 수 있다. 관리자 기능을 통해 Firestore 상에서 졸업 요건과 대체과목 규칙을 직접 관리함으로써, 규정 변경 반영 속도와 일관성을 높일 수 있다.

기술적·사회적·경제적 의의도 존재한다. 데이터 구조와 알고리즘은 다른 대학에도 쉽게 이식 가능한 범용 설계를 따르고 있어, 멀티 캠퍼스·다대학 공동 플랫폼으로 확장할 수 있다. 이는 개발·운영 비용 절감과 교육과정 개선을 위한 데이터 분석 기반을 제공한다. 사회적으로는 팬데믹 이후 가속된 교육 디지털 전환과 학생 중심 서비스 요구에 부응하는 인프라로 기능하며, 경제적으로는 졸업 지연으로 인한 등록금·생활비·기회비용을 줄이는 효과가 기대된다.

다만 몇 가지 한계도 존재한다. 첫째, 시스템이 Firestore에 저장된 졸업 요건 데이터의 정확성에 강하게 의존한다는 점이다. 관리자의 입력 오류나 업데이트 누락 시, 분석 결과 역시 잘못될 수 있으나 현재는 이를 자동 검증하는 메커니즘이 부족하다. 둘째, 학식 메뉴 기능은 웹 스크래핑에 기반해 학교 홈페이지 구조 변경과 서버 상태에 취약하다. 셋째, 과목 추천은 졸업 요건 충족에 초점을 맞추고 있어 흥미, 진로, 난이도, 교수 평가 등 개인적 요소를 충분히 반영하지 못한다. 넷째, Android 전용 플랫폼이라는 점에서 iOS·웹 사용자는 접근할 수 없고, 완전한 오프라인 동기화·충돌 처리도 아직 제한적이다. 다대학 지원 구조, 접근성(스크린 리더·고대비 모드 등)·모듈화 아키텍처 측면에서도 개선 여지가 있다.

향후 연구 과제로는 ① 과거 이력·성적·진로 데이터를 활용한 기계학습 기반 과목 추천 고도화, ② LLM 및 음성 인터페이스를 활용한 지능형 챗봇, ③ React Native·Flutter·웹 앱을 통한 크로스 플랫폼 확장, ④ 멀티 테넌시 기반 다대학 지원 플랫폼화, ⑤ 실시간 협업·시간표 공유·스터디 매칭 기능, ⑥ 데이터 자동 검증 및 학사 규정 문서 파싱 도구, ⑦ 학습 분석·졸업 지연 위험 예측 모델, ⑧ 지속적 사용자 피드백·A/B 테스트 기반 개선 등을 제안한다.

동시에 본 연구는 종료점이 아니라 출발점으로, 데이터 품질, 개인화, 플랫폼 확장, 접근성 등 남은 과제를 향후 연구와 다기관 협력을 통해 단계적으로 해결해 나갈 필요가 있다. 궁극적으로 이 시스템이 더 많은 대학과 학생에게 확산되어, 학생들이 졸업 요건이라는 행정적 장벽에 묶이지 않고 자신의 학업 경로를 주도적으로 설계·관리하는 데 기여하는 것을 목표로 한다.

References

[1] Xu, Y., Xing, T., & van der Schaar, M. (2016). Personalized Course Sequence Recommendations. IEEE Transactions on Signal Processing.

[2] Ren, Z., Ning, X., & Rangwala, H. (2017). Grade Prediction with Temporal Course-wise Influence. In Proceedings of the 10th International Conference on Educational Data Mining (EDM 2017).

[3] Mi, Y., Yu, Y., & Zhao, Y. (2025). SmartCourse: A Contextual AI-Powered Course Advising System for Undergraduates. arXiv preprint arXiv:2507.22946

[4] Lee, Y. (2005). 모바일 코스 코디네이터 시스템의 설계 및 구현. 한국컴퓨터교육학회논문지.

[5] 임은기. (2013). 효율적인 유지보수가 가능한 대학졸업사정시스템의 설계 및 구현: 사례 연구. 디지털융복합연구, 11(2), 193-203.

김 준 성

4학년 (2020년 입학)
담당업무: 프로젝트 총괄,
졸업 사정 기능 개발,
과목 추천 기능 개발
E-mail: kdw2770@naver.com

이 가 현

4학년 (2021년 입학)
담당업무: 개인・그룹 캘린더 개발,
로그인 기능 구현,
캠퍼스 지도 기능 개발
E-mail: nettrol13@naver.com

임 윤 성

4학년 (2021년 입학)
담당업무: 자료조사
E-mail: youtube2016@naver.com

※ 결과보고서 작성 시 주의사항

1. 보고서 페이지 수 : 10페이지

2. 보고서 용지 및 여백처리 (F7), 서식 스타일 (F6)은 반드시 본 양식을 그대로 준용한다.

3. 참고문헌 : 참고문헌은 본문에 인용되었거나 언급된 것으로 제한하며, 본문 중에 [ ]를 사용하여 참고문헌 번호를 기재한다 (예: 정의하였다[4].)

- 참고문헌(References) 목록 순서는 인용한 순서대로 기술한다.
- 여러 개를 동시에 인용하는 경우에는 [1,2,3]과 같은 방법으로 기술한다.
- 참고문헌의 종류별 표기 방식은 본 양식 파일의 References 예시를 따른다.
    1) 한글논문 : References [1]
    2) 외국논문 : References [2]
    3) 도서 : References [3]
    4) 웹사이트 URL : References [4]