



# 3중 학사관리 시스템

파이썬 프로그램 만들기 중간 발표

Team와해

김재영 나지수 송제용 임지후 장현준 정종빈 최혁



# 목차

프로젝트 소개

팀 구성 및 역할 분담

모듈 구성 (1)

모듈 구성 (2)

현재 진행 상황

향후 계획



# 프로젝트 소개

## 개발 목적

- 학생, 교수, 관리자를 위한  
통합 학사관리 시스템 개발
- 파이썬 프로그래밍 역량 강화
- 모듈화된 설계를 통한 유지보수성 향상
- 사용자별 맞춤형 기능 제공
- 효율적인 학사 정보 관리 실현

## 프로젝트 개요

- 3중 사용자(학생, 교수, 관리자)  
지원 시스템
- 모듈화된 구조로 개발
- 각 사용자별 권한 관리 시스템 구현
- 파이썬 클래스와 상속 개념 활용
- 팀 협업을 통한 개발 진행

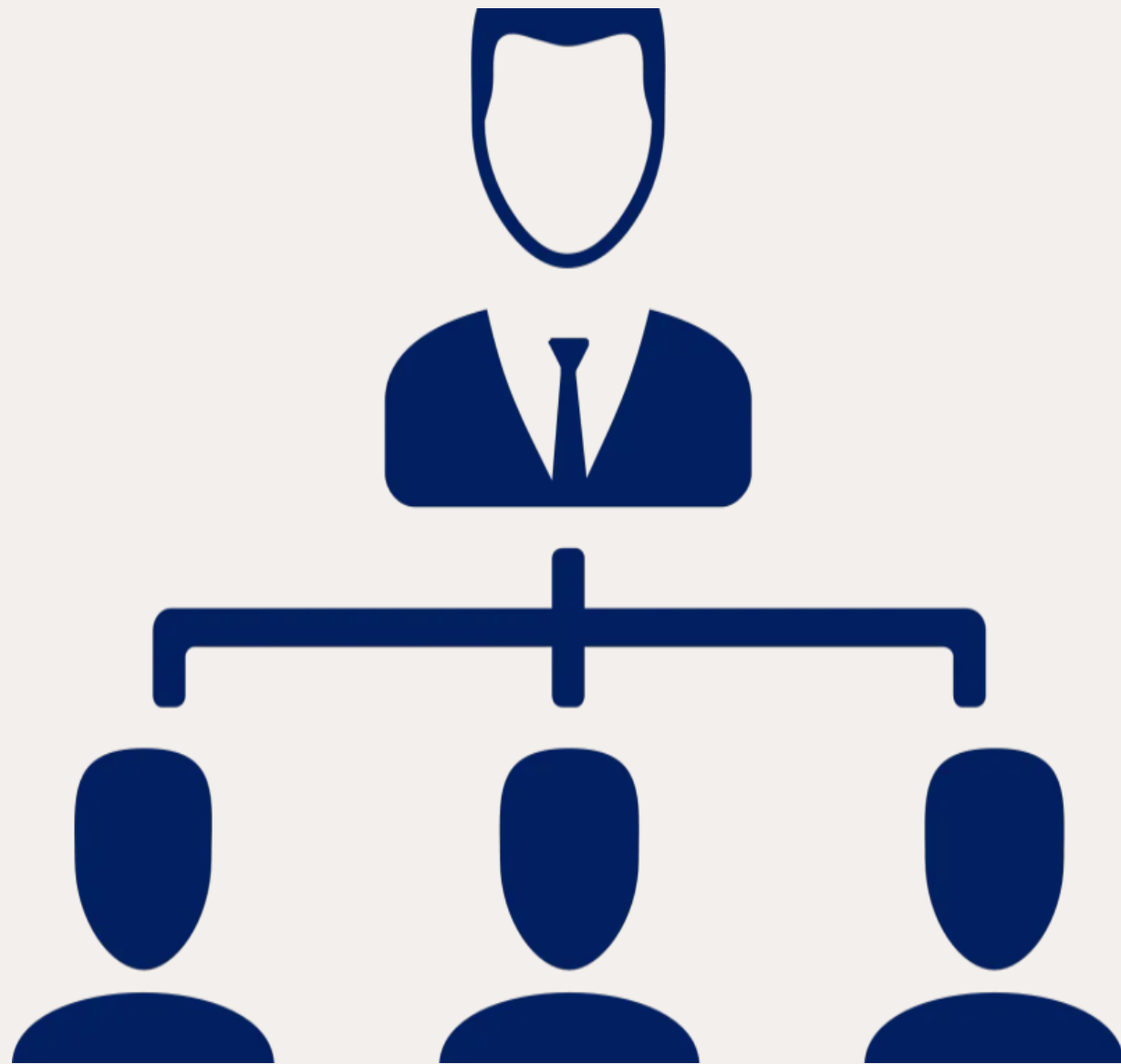
## 시스템 주요 기능

- 학생: 수강신청, 성적 확인, 개인정보 관리
- 교수: 강의 관리, 성적 입력, 학생 정보 조회
- 관리자: 전체 시스템 관리,  
사용자 계정 관리
- 공통: 로그인/로그아웃,  
데이터 저장 및 불러오기



# 팀 구성 및 역할 분담

○ 본 프로젝트는 6명의 팀원이 각자 모듈을 담당하여 개발하는 방식으로 진행되고 있습니다.  
각 모듈은 독립적으로 개발되며 공통 인터페이스를 통해 연결됩니다.



## 팀원 소개 및 담당 모듈

- 김재영: common\_module.py 담당
- 나지수, 임지후: main.py 담당
- 송제용: admin\_class.py 담당
- 장현준: professor\_class.py 담당
- 정종빈: student\_class.py 만들기
- 최혁: data.py 만들기

## 협업 방식

- 주 1회 정기 미팅을 통한 진행상황 공유
- 깃허브를 활용한 코드 버전 관리
- 모듈 간 인터페이스 사전 정의
- 코드 리뷰를 통한 품질 향상
- 문제 발생 시 즉시 공유 및 해결 방안 모색
- 정기적인 피드백 교환



# 모듈 구성 (1)

```
# common_module.py
import os
import json
import data
from datetime import datetime

GRADE_TO_SCORE = {
    'A+': 4.5, 'A0': 4.0, 'B+': 3.5, 'B0': 3.0,
    'C+': 2.5, 'C0': 2.0, 'D+': 1.5, 'D0': 1.0}
```

## common\_module.py

- 담당: 김재영
- 모든 클래스에서 공통으로 사용하는 함수 구현
- 사용자 인증 및 권한 검증 기능
- 데이터 입출력 표준화
- 예외 처리 및 로깅 기능

```
# main.py
import common_module as common
from student_class import Student
from professor_class import Professor
from admin_class import Admin

def main():
    """메인 실행 함수"""
```

## main.py

- 담당: 나지수, 임지후
- 프로그램의 시작점
- 사용자 인터페이스 구현
- 메뉴 시스템 및 네비게이션
- 각 모듈 통합 및 연결
- 세션 관리 기능

```
# admin_class.py
import data
import common_module as common
from datetime import datetime

class Admin:
    def __init__(self, user_id):
        self.user_id = user_id
        self.info = data.admins[user_id]
```

## admin\_class.py

- 담당: 송제용
- 관리자 권한 기능 구현
- 사용자 계정 관리
- 시스템 설정 관리
- 데이터베이스 관리
- 로그 및 통계 기능

common\_module.py

data.py

main.py

professor\_class.py

student\_class.py

## 모듈 간 연계

- 모듈 간 표준 인터페이스 정의
- 공통 데이터 구조 활용
- 이벤트 기반 통신 구현
- 의존성 최소화 설계
- 확장성을 고려한 구조



## 모듈 구성 (2)

○ 각 모듈은 객체지향 설계 원칙에 따라 개발되며, 상속과 다형성을 활용하여 코드 재사용성을 높이고 있습니다.

### professor\_class.py

- 담당: 장현준
- 교수 사용자 클래스 구현
- 강의 개설 및 관리 기능
- 학생 성적 입력 및 수정
- 강의 계획서 관리
- 출석 관리 시스템
- 학생 정보 조회 기능
- 공지사항 등록 기능

### student\_class.py

- 담당: 정종빈
- 학생 사용자 클래스 구현
- 수강신청 및 취소 기능
- 성적 조회 시스템
- 개인정보 관리 기능
- 강의 평가 시스템
- 학사 일정 조회
- 공지사항 확인 기능
- 질의응답 시스템

### data.py

- 담당: 최혁
- 데이터 저장 및 관리 모듈
- 파일 입출력 기능 구현
- 데이터 백업 및 복구 기능
- 데이터 무결성 검증
- 사용자 데이터 관리
- 강의 정보 데이터 관리
- 성적 데이터 관리
- 시스템 로그 관리



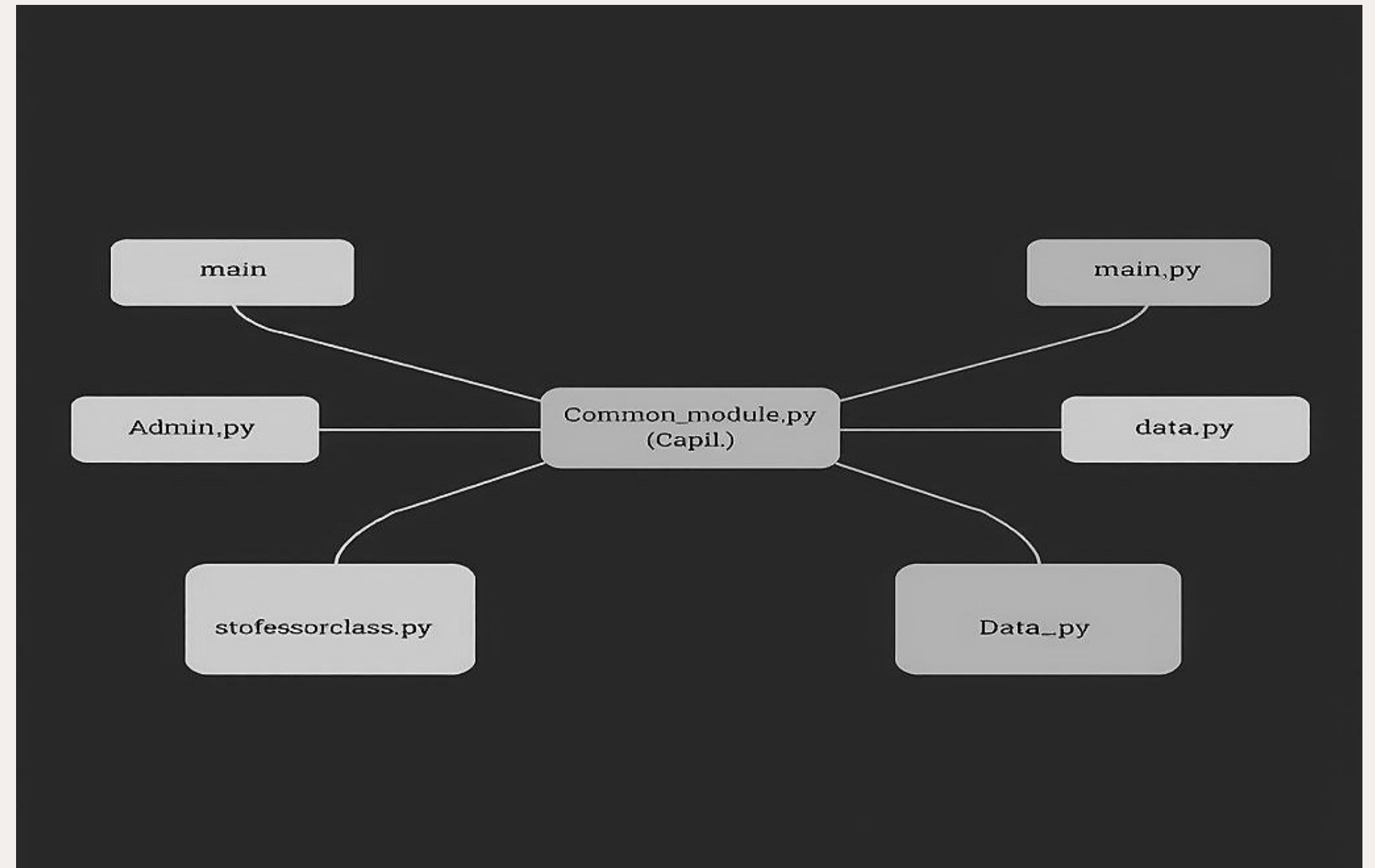
# 현재 진행 상황

## 모듈별 개발 완료 현황

- 현재 모든 팀원이 각자 담당한 모듈 개발이 완성 단계에 있습니다.

## 코드 최적화 작업

- 코드 리뷰 세션을 통한 피드백 교환
- 중복 코드 제거 및 함수 최적화
- 모듈 간 인터페이스 표준화
- 예외 처리 로직 보강





# 향후 계획: 핵심 모듈 개발 및 테스트

○ 남은 핵심 기능 개발과 테스트를 통해 시스템 안정성을 확보하겠습니다.

## 기능 개발

- 사용자 인증 기능
- 성적 처리 시스템
- 수강 신청 모듈
- 강의 평가 기능
- 학사 일정 관리
- 알림 시스템
- 권한 관리 기능

## 통합 테스트

- 모듈 간 연동 테스트
- 시나리오 기반 테스트
- 사용자 역할별 테스트
- 데이터 흐름 검증
- 전체 시스템 성능 측정
- 부하 테스트

## 버그 수정

- 오류 로깅 시스템 구축
- 버그 추적 관리
- 코드 리팩토링
- 기능 개선 사항 반영
- 사용자 피드백 기반 수정
- 성능 최적화





# 향후 계획: 통합 테스트 및 시범 운영

## ○ 통합 테스트 일정

- 11월 2주: 모듈 간 연동 테스트
- 11월 3주: 전체 시스템 테스트
- 11월 4주: 성능 및 부하 테스트

## ○ 사용자 피드백 수집

- 온라인 설문조사 시스템 구축
- 사용자 인터뷰 진행
- 실시간 피드백 채널 운영
- 오류 신고 시스템 구현

## ○ 유지 보수 계획

- 정기 업데이트 일정 수립
- 기술 지원 체계 구축
- 사용자 매뉴얼 작성
- 지속적인 기능 개선 계획

## ○ 시스템 안정화 전략

- 단계적 기능 릴리즈
- 백업 및 복구 시스템 구축
- 모니터링 시스템 도입
- 부하 분산 전략 수립