

강찜콩(2조) 보고서

액체지향프로그래밍(캡스톤디자인)(001) 2조

고명주, 구나은, 김나린, 임수정, 전여원, 정서원, 최지은

목차

I.	제작 동기 및 목적, 목표	2p
II.	전체 프로그램 / Main	2-7p
III.	Database, Backend	8-14p
IV.	Frontend	15-21p
V.	프로그램 발전 방향	22-24p
VI.	제작 과정 & 느낀 점	25-27p
VII.	기대 효과	28p

I. 제작 동기 및 목적, 목표

1. 프로그램 제작 동기 및 목적

숙명 여자 대학교 학생들이 교내 강의실을 대여할 때, 기존의 예약하는 과정이 번거로워 해결하고자 본 프로그램을 제작하였다. 프로그램 “강찜콩”으로 학생들은 원하는 조건에 맞는 공간을 편리하게 예약하여 사용할 수 있다. 본 프로그램은 직관적인 인터페이스를 통해 간소화한 예약 과정으로 학생들의 편의성에 중점을 두고 제작했다.

2. 프로그램 제작 목표

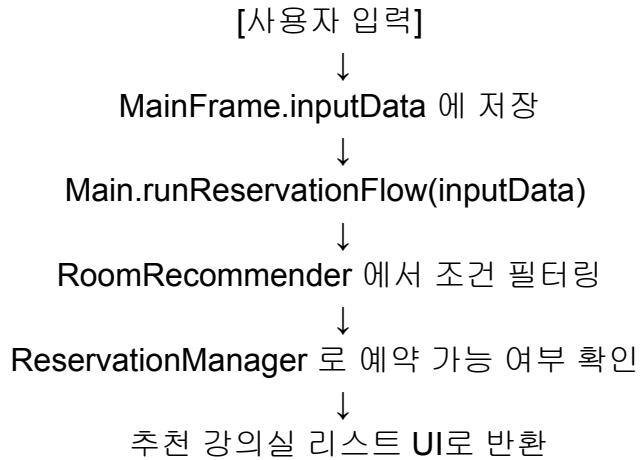
- 1) 실시간 조회: 사용자가 교내 건물(명신관, 순현관, 프라임관) 강의실의 예약 가능 여부를 실시간으로 조회할 수 있는 시스템 구축.
- 2) 맞춤형 검색: 세부 조건(사용 날짜, 시간, 인원, 필요 시설)에 따른 강의실 필터링 기능을 통해 사용자가 원하는 공간 제공.
- 3) 간편한 예약: 재학생(7자리 학번) 로그인을 기반으로 예약 시스템 구현.

II. 전체 프로그램

1. 프로그램 개요

사용자의 학번을 입력해서 로그인하고 사용자가 원하는 조건에 따라서 빈 강의실을 예약하고 사용할 수 있는 프로그램

2. 전체 알고리즘



3. Main

1) 개요

Main 클래스는 사용자 입력 흐름을 백엔드 필터링 로직과 연결하여, 조건에 맞는 강의실 추천 결과를 생성하고 UI에 반환되도록 하는 중심 역할을 수행한다.

2) 핵심 역할 및 흐름

- (1) 사용자가 로그인 후 입력한 학번, 건물, 날짜, 시간, 인원, 시설 등의 정보는 MainFrame 내부의 inputData 필드에 순차적으로 저장
(inputData: ReservationInputData 클래스의 인스턴스로, 모든 예약 조건을 하나의 객체로 통합하여 관리함)
- (2) Main.runReservationFlow(inputData) 메서드 호출
- (3) RoomRecommender가 inputData를 바탕으로 조건에 맞는 강의실 목록을 필터링
- (4) 추천 가능한 강의실 리스트 생성

3) 구현한 주요 연동 작업

- ① 사용자 입력값을 MainFrame.inputData에 저장되도록 각 패널과 연결
- ② runReservationFlow() 메서드를 통해 백엔드 필터링 흐름과 연결
- ③ 추천 강의실 리스트가 UI에서 바로 활용 가능하도록 리스트 형태로 반환되도록 설계

4) Class 설명

(1) Main

① 개요 : 사용자 입력 데이터를 바탕으로 강의실 필터링 및 추천 과정을 실행하는 핵심 흐름 제어 클래스. 프론트엔드와 백엔드를 연결하며, 추천 결과를 반환하는 메인 메서드.

② 로직

(가) inputData에 저장된 예약 조건(건물, 날짜, 시간, 인원, 시설 등)을 출력하여 확인

(나) RoomFileReader를 통해 모든 강의실 데이터를 불러옴

(다) 주말 여부 확인: 예약 날짜가 주말(SATURDAY, SUNDAY)일 경우 WeekendException 발생

(라) RoomRecommender.recommendRooms() 호출하여 조건에 맞는

강의실

필터링

(마) ReservationManager를 통해 시간대 중복 여부 확인

(바) 추천 가능한 강의실이 있다면 콘솔에 출력, 없으면 “추천 불가” 메시지 출력

(2) ReservationInputData

① 개요 : 강의실 추천에 필요한 사용자 입력 조건들을 하나의 객체로 묶어 전달하는

데이터 컨테이너 클래스. Main 클래스 및 추천 관련 클래스들에 공통으로 사용됨.

② 구성 변수

(가) studentId (String): 로그인한 학생의 학번

(나) date (LocalDate): 예약 희망 날짜

(다) start (LocalTime): 예약 시작 시간

(라) end (LocalTime): 예약 종료 시간

(마) building (String): 예약을 원하는 건물 이름

(바) minCapacity (int): 최소 인원 수

(사) requiredFacilities (Set<String>): 필수로 원하는 시설 목록 (ex. 마이크, 콘센트 등)

(아) isStepped (boolean): 계단형 강의실 여부

③ 특징

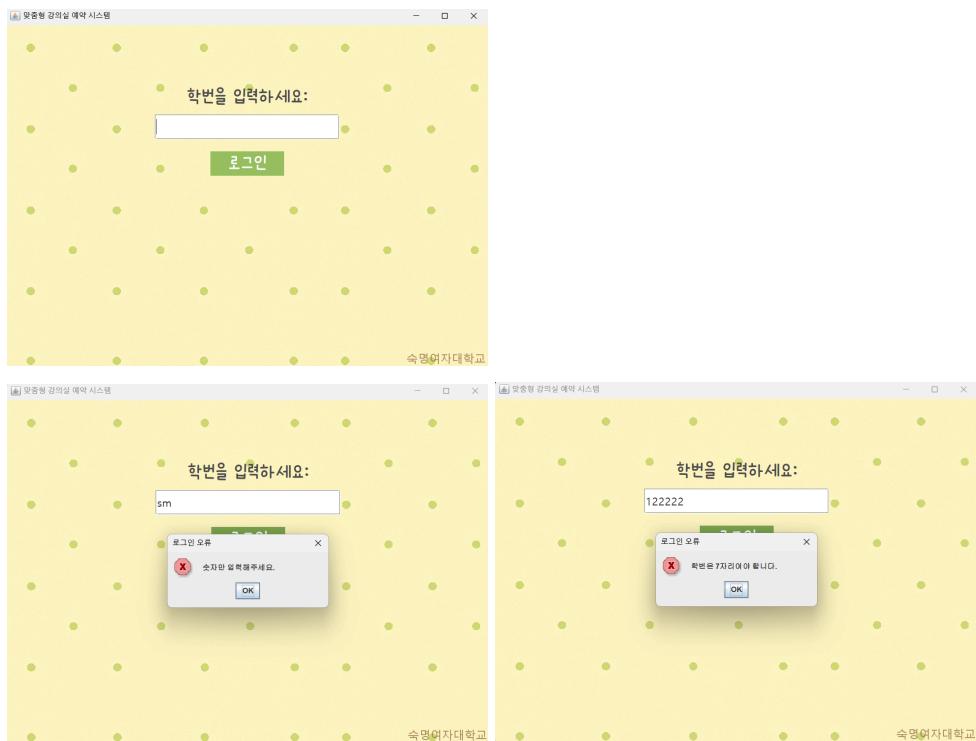
필드에 직접 접근하는 public 구조로 설계되어, 사용자가 선택한 조건이
MainFrame에서 이 객체에 곧바로 저장됨
이후 runReservationFlow() 메서드에서 필터링 및 예약 가능 여부 판단 시
0이
액체를 통째로 전달함으로써 흐름이 간결하게 연결됨

4. 실행 화면

1) 인트로



2) 학번 입력(로그인)

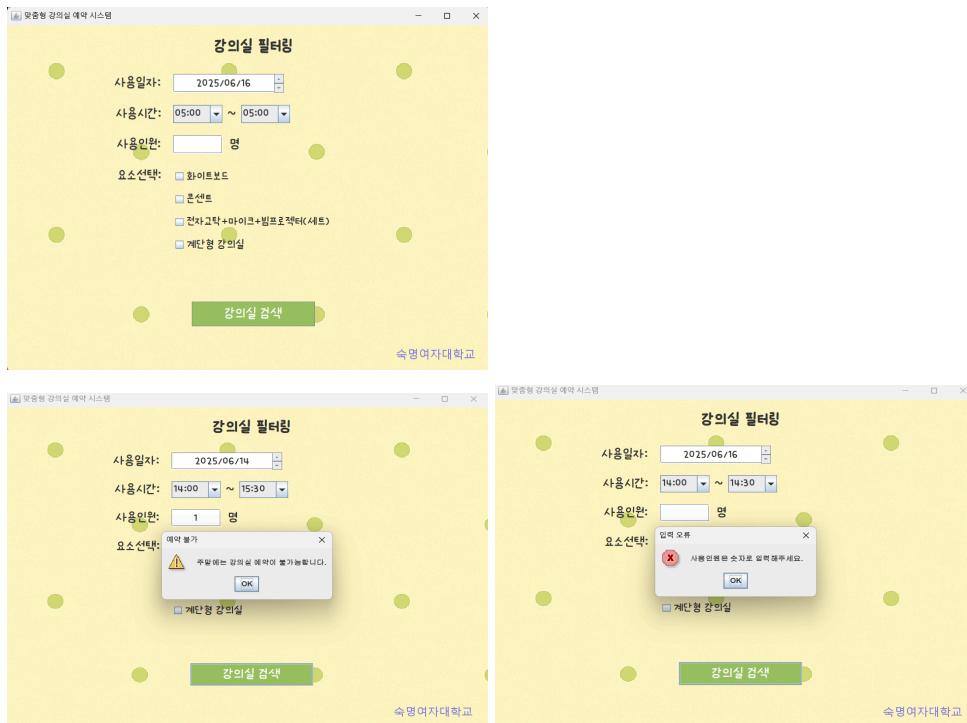


*잘못된 입력 시 팝업으로 표시

3) 건물 선택

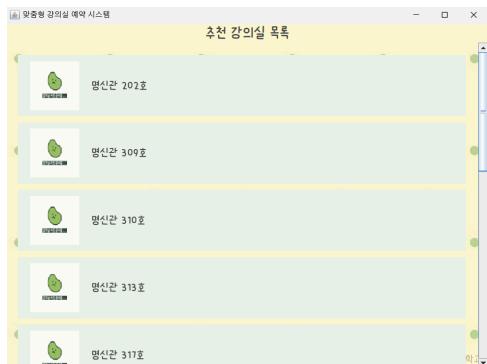


4) 희망 강의실 조건 선택



*잘못된 입력 시 팝업으로 표시

5) 추천 강의실 목록



6) 강의실 정보 확인



7) 예약 완료



5. 백엔드 관리자 화면

```
C:\ClassroomReservation>java -cp out frontend.App  
? [입력 데이터 확인]  
건물: 명신관  
날짜: 2025-06-16  
시간: 10:30 ~ 11:30  
인원: 2  
시설: [콘센트]  
계단형 여부: false  
학생 ID: 2410914  
추천 강의실 목록:  
- 210호  
- 517호  
- 518호  
사용 일자: 2025/06/16  
시작 시간: 10:30  
종료 시간: 11:30  
사용 인원: 2  
선택한 요소: [콘센트]
```

```
? [입력 데이터 확인]  
건물: 프라임관  
날짜: 2025-06-16  
시간: 13:00 ~ 15:00  
인원: 2  
시설: [콘센트]  
계단형 여부: false  
학생 ID: 2222222  
추천 강의실 목록:  
- 102호  
- 103호  
- 204호  
- 303호  
- 304호  
사용 일자: 2025/06/16  
시작 시간: 13:00  
종료 시간: 15:00  
사용 인원: 2  
선택한 요소: [콘센트]
```

III. Database, Backend

1. Database

1) Database 개요

본 프로젝트에서는 별도의 데이터베이스(DBMS)를 사용하지 않고, 강의실 정보와 시간대 정보를 담은 `roomtest.txt` 파일을 기반으로 데이터를 처리하였다.

(1) roomtest.txt

① 개요 : `roomtest.txt`는 강의실 추천 시스템의 핵심 데이터 소스로, 강의실의 기본 정보와 사용 가능 시간대를 함께 저장한 정형 텍스트 파일 데이터베이스를 대체하는 역할을 하며, 강의실 추천 기능 구현에 있어 기초 데이터(건물명, 호실, 수용인원, 시설 등)를 제공

② 로직 : `RoomFileReader` 클래스를 통해 `roomtest.txt` 파일을 읽고, 각 줄을 다음 방식으로 처리

(가) 파일을 한 줄씩 읽으며 반복 처리 (빈 줄은 건너뜀)

(나) 각 줄은 '!' 기호를 기준으로 기본 정보와 시간 정보로 분리됨

(다) 기본 정보 파싱

- ',' 기호로 건물명, 강의실명, 수용 인원, 계단식 여부, 시설 목록을 구분

- 시설 목록은 ';' 기호로 다시 나누어 `Set<String>` 형태로 저장

(라) 시간 정보 파싱

- 요일(mon, tue 등)과 괄호로 둘러싼 시간 범위들을 파싱

- 시간 범위는 '/'로 여러 개 구분되고, 각 범위는 '-' 기호로 시작·종료 시각을 구분

(마) 파싱된 정보는 `RoomSelection` 객체로 저장되고, 리스트 형태로 관리됨

[예시 데이터]

명신관,405호,50,false,빔:콘센트;마이크!mon(09:00-11:00/13:00-15:00)/tue(10:00-12:00)

2. Backend

1) Backend 개요

프로그램 “강찜콩”의 로그인, 내부 시설 선택 후 강의실 필터링, 예약 시스템을 작동하게 하는 class

2) Class 설명

(1) LoginValidator

① 개요 : 학번의 유효성을 검사 후 결과에 따라 로그인 성공 또는 오류 메시지 출력

② 로직

(가) login Method

- validateLoginId method → try-catch 구문으로 호출해 예외 처리 관리

(나) validateLoginId Method

- \d+ → 학번 only 숫자 구성 확인 / 아닐 경우 "숫자만 입력해주세요." 예외 메시지 출력 (InvalidLoginIdException 발생)
- 학번의 길이 7자리인지 확인 / 아닐 경우 "학번은 7자리여야 합니다." 예외 메시지 출력 (InvalidLoginIdException 발생)

③ 변수

(가) loginId (String): Method 파라미터, 사용자가 입력한 검증 대상 학번 문자열

(2) DateUtils

① 개요 : "yyyy-MM-dd" 형식의 문자열 LocalDate 객체로 변환, LocalDate 객체에서 요일을 한글로 변환하는 class

② 로직

(가) parseDate Method

- DateTimeFormatter로 "yyyy-MM-dd" 정의, 이에 맞춰 입력된 문자열을 LocalDate 객체로 파싱(변환)

(나) getKoreanDayOfWeek Method

- 입력된 LocalDate 객체의 getDayOfWeek() 결과 → switch 문으로 확인 등 해당하는 한글曜일 문자열 반환

③ 변수

(가) `input(String)`: `parseDate` Method 파라미터, "yyyy-MM-dd"

형식의 날짜 문자열

(나) `date(LocalDate)`: `getKoreanDayOfWeek` Method 파라미터,

요일을 얻고자 하는 `LocalDate` 객체

(3) ReservationInputData

① 개요 : 여러 변수에 있을 예약 관련 조건들을 하나의 객체로 둑어 관리 전달하는 데이터 필드 class

② 변수

(가) `studentId (String)`: 학생의 학번

(나) `date (LocalDate)`: 예약 희망 날짜

(다) `start (LocalTime)`: 예약 희망 시작 시간

(라) `end (LocalTime)`: 예약 희망 종료 시간

(마) `building (String)`: 희망 건물 이름

(바) `minCapacity (int)`: 최소 수용 인원

(사) `requiredFacilities (Set<String>)`: 필수 시설 목록

(아) `isStepped (boolean)`: 계단식 강의실 필요 여부

(4) RoomFileReader

① 개요 : 파일 I/O 통해 데이터 읽고 파싱해 `RoomSelection` 객체 리스트 생성하는 class

② 로직

(가) 파일 한 줄씩 읽으며 반복 처리.(빈 줄 건너뜀)

(나) 각 줄은 '!' 기준으로 기본 정보, 사용 가능 시간 정보로 분리

(다) 기본 정보 파싱

- ',' 기준으로 건물명, 강의실명, 수용인원, 계단식 여부, 시설 목록 파싱
- 시설 목록은 ';' 기준으로 분리

(라) 시간 정보 파싱

- 요일 (`mon, tue` 등)과 괄호 안의 시간 범위를 파싱
- 시간 범위는 '/'로 여러 개 구분, 각 범위는 '-'로 시작과 끝 시간 구분.
- 주말(토, 일) 데이터는 건너뜀.

(마) 파싱된 모든 정보 취합해 하나의 `RoomSelection` 객체 생성, 이 객체를 `rooms` 리스트에 추가.

(바) 파일의 모든 줄 처리 후, 완성된 rooms 리스트 반환.

③ 변수

(가) filename (String): 읽어올 강의실 정보 파일의 경로.

(나) rooms (List<RoomSelection>): 파일 바탕으로 생성된 RoomSelection 객체들을 저장하는 리스트.

(다) reader (BufferedReader): 파일을 한 줄씩 읽기 위한 객체.

(5) RoomSelection

① 개요 : 강의실의 이름, 위치, 수용 인원, 보유 시설, 사용 가능 시간 등 고유한 속성을 하나의 단위로 묶어 관리하는 class

② 로직

(가) qals, hashCode Method : 건물 이름과 강의실 이름이 모두 같으면 동일한 객체로 판단

③ 변수

(가) room (String): 강의실 이름.

(나) buildingName (String): 건물의 이름.

(다) capacity (int): 수용 가능 인원.

(라) facilities (Set<String>): 보유 시설 목록.

(마) isStepped (boolean): 계단식 강의실 여부.

(바) minCapacity (int): 최소 수용 인원

(사) availableTimes (Map<DayOfWeek, List<TimeRange>>): 요일별로 예약 가능한 시간대 목록.

(6) RoomRecommender

① 개요 : 전체 강의실 목록 중에서 사용자가 제시한 조건에 부합하는 강의실 필터링하여 목록으로 제공하는 class

② 로직

(가) recommendRooms Method : 자바의 Stream API 사용하여 모든 강의실(allRooms) 순회 roomMatches method로 각 강의실 검사

(나) roomMatches Method : 다음의 모든 조건이 참인지 확인, 하나라도 거짓이면 해당 강의실 추천 제외.

- 건물 이름이 일치하는가?
- 수용 인원이 최소 요구 인원 이상인가?
- 계단식 강의실 요구 시, 실제 계단식인가?
- 필요한 모든 시설을 갖추고 있는가?
- 요청한 시간이 강의실의 운영 시간에 포함되는가?

- 해당 날짜와 시간에 다른 예약이 없는가?
(`manager.isAvailable` 호출)

③ 변수

- (가) `recommendRooms` method에 전달되는 파라미터 통해 동작.
- (나) `allRooms` (`List<RoomSelection>`): 필터링의 대상이 되는 시스템의 모든 강의실 목록.
- (다) `buildingName`, `minCapacity`, `requiredFacilities`, `requireStepped`: 사용자가 요청한 필터링 조건.
- (라) `desiredTimeRanges`, `targetDate`: 예약하려는 시간과 날짜.
- (마) `manager` (`ReservationManager`): 특정 시간에 해당 강의실이 이미 예약되었는지 확인하기 위한 객체.

(7) `ReservationInputData`

① 개요 : 예약 관련 조건들을 하나의 객체로 묶어 관리하고 전달하는 데이터 필드 `class`

② 변수

- (가) `studentId` (`String`): 예약을 요청한 학생의 학번.
- (나) `date` (`LocalDate`): 예약 희망 날짜.
- (다) `start`, `end` (`LocalTime`): 예약 희망 시작 및 종료 시간.
- (라) `building` (`String`): 희망 건물 이름.
- (마) `minCapacity` (`int`): 최소 수용 인원.
- (바) `requiredFacilities` (`Set<String>`): 필수 시설 목록.
- (사) `isStepped` (`boolean`): 계단식 강의실 필요 여부.

(8) `ReservationManager`

① 개요 : 강의실 예약 가능 여부를 확인, 새로운 예약 처리, 특정 학생의 예약 내역 조회하는 기능 제공하는 `class`

② 로직

- (가) `isAvailable Method`: 특정 강의실이 요청된 날짜와 시간에 예약 가능한지 검사. 강의실 자체의 운영 시간과 `reservations` 변수에 기록된 다른 예약과의 시간 충복 여부 모두 확인
- (나) `reserve Method`: 새로운 예약 생성
 - 먼저 예약하려는 날짜가 주말(토/일)인지 확인, 주말일 경우 `WeekendException` 예외 발생

- `isAvailable` 호출 → 예약이 가능한지 최종 확인 후, `reservations`와 `userReservations` 두 변수에 모두 새로운 예약 정보 기록

(다) `getReservationsByStudentId` Method: `userReservations` 변수 사용해 특정 학번의 모든 예약 기록 찾아 리스트로 반환

③ 변수

(가) `reservations` (Map): 강의실 중심으로 모든 예약 정보를 저장하는 변수, [강의실 객체 -> [날짜 -> 예약된 시간 목록]] 형태의 중첩된 자료구조

(나) `userReservations` (Map): 사용자 중심으로 예약 정보를 저장하는 변수, [학번 -> 해당 학생의 예약 기록 목록] 형태

(9) ReservationRecord

① 개요 : 학생, 강의실, 날짜, 시간 등 확정된 예약의 구성 요소 하나의 객체로 둑어 관리하는 class

② 로직

(가) 데이터 보관이 주된 역할, 생성자 통해 모든 정보를 한 번에 입력받음

(나) `toString` Method 재정의 : 객체 출력할 때 "학번: 강의실명 (건물명), 날짜: YYYY-MM-DD, 시간: HH:MM - HH:MM" 형식으로 보기 쉽게 변환

③ 변수

(가) `studentId` (String): 예약을 완료한 학생의 학번.

(나) `room` (RoomSelection): 예약된 강의실 정보가 담긴 객체.

(다) `date` (LocalDate): 예약된 날짜.

(라) `timeRanges` (List<TimeRange>): 예약된 시간대 목록.

(10) TimeRange

① 개요 : 시간 범위를 객체로 정의하고, 시간대끼리 겹치는지 혹은 서로를 포함하는지 등을 계산하는 class

② 로직

(가) `overlaps(TimeRange other)` Method: 두 시간대가 조금이라도 겹치는지 확인

(나) `contains(TimeRange other)`: 하나의 시간대가 다른 시간대를 완전히 포함하는지 확인

(다) equals, hashCode Method : 시간대 객체를 컬렉션에서 비교하거나 검색할 때 동작

③ 변수

(가) start (LocalTime): 시간 범위의 시작 시간.

(나) end (LocalTime): 시간 범위의 종료 시간.

3) 예외 처리 설명

(1) InvalidLoginIdException

① 개요 : 학번 유효성 검사 실패라는 상황을 알리는 역할

② 발생 조건

(가) 학번이 숫자로만 구성되지 않았을 때.

(나) 학번이 정확히 7자리가 아닐 때.

③ 처리

(가) LoginValidator의 login method에서 이 예외를 catch하여, 로그인 오류 메세지를 사용자에게 출력

(2) WeekendException

① 개요 : 주말 예약 시도 위반 처리

② 발생 조건

(가) ReservationManager의 reserve method 내에서 예약 요청한 날짜의 요일을 확인 → 해당 요일이 토요일이나 일요일일 경우, 예외 throw

③ 처리

(가) reserve method를 호출한 쪽에서 이 예외를 catch하여 예외 메시지를 사용자에게 출력

IV. Frontend

프론트엔드 파트 1

[화면 전환 흐름]

1. IntroPanel

1) 초기 화면 동작

(1) 프로그램 실행 시 자동으로 나타나는 인트로 화면

① **gangjjim** 로고 이미지가 중앙에 크게 보여짐

② 8초간 유지되거나, 사용자가 이미지를 클릭하면 다음 화면으로 넘어감

③ 브랜드 이미지를 전달하는 역할을 함

2. StartPanel

1) 인트로 종료 후 자동 전환

(1) 중앙에 **gangjjim** 이미지와 안내 문구 배치

① 문구는 "강의실을 짐콩하시겠어요?"

(2) 사용자 상호작용

① "네" 버튼 클릭 시 **LoginPanel**로 이동

② **enter** 키 입력 시에도 전환 가능

3. LoginPanel

1) 사용자 로그인 기능

(1) 학번 입력 필드 및 로그인 버튼 구성

① 유효한 학번인지 확인: 공백 아님, 정수 변환 가능, 1~25 사이 숫자

(2) 조건 충족 시 **BuildingSelectionPanel**로 전환

4. BuildingSelectionPanel

1) 건물 선택 기능

(1) 순현관, 명신관, 프라임관 등의 이미지 버튼 제공

① 사용자가 건물 버튼 클릭 시 선택됨

(2) 다음 요소 선택 화면으로 이동되도록 구성됨

【패널별 코드 상세 설명】

1. IntroPanel.java

1) 패널 구조

(1) JPanel 상속 및 BoxLayout 수직 정렬

① gangjjim 이미지를 ImageIcon으로 로드하여 크기 조정

② JLabel로 중앙에 이미지 표시

(2) 이벤트 처리

① 이미지 클릭 시 MainFrame의 showPanel("start") 호출

② javax.swing.Timer로 8초 후 자동 전환 설정

(3) 그래픽 처리

① paintComponent(Graphics g) 오버라이딩

② 타일 형태의 배경 이미지 반복 출력

2. StartPanel.java

1) 레이아웃 및 구성

(1) BorderLayout 사용, 주요 컴포넌트는 중앙에 BoxLayout으로 배치

① 안내 문구는 ps.ttf 폰트를 적용한 JLabel

② gangjjim 이미지와 "네" 버튼 포함

(2) 이벤트 처리

① "네" 버튼 클릭 시 showPanel("login") 호출

② MouseAdapter를 통해 hover 시 배경색 변경 처리

(3) 배경 및 푸터

① paintComponent()에서 배경 이미지 반복 출력

② 하단에 "숙명여자대학교" 텍스트, RGB(120,80,40), alpha 150의 반투명 색상 적용

3. LoginPanel.java

1) 레이아웃 구성

(1) BorderLayout 중앙에 수직 정렬된 입력창, 안내문구, 버튼 배치

① 학번 입력은 JTextField, 폭은 300픽셀 제한

(2) 입력값 검증 및 전환

① 공백 검사, 정수형 검사, 범위 검사(1~25)

② 모든 조건 통과 시 showPanel("building") 호출

(3) 시각 효과

① 버튼 hover 시 색상 변화 구현

② paintComponent()에서 배경 이미지 반복 출력

4. BuildingSelectionPanel.java

1) 패널 구조

(1) BorderLayout 기반

① 중앙에 건물 선택 버튼들 수평 배치(FlowLayout)

② 하단에는 반투명 갈색 푸터 텍스트 배치

(2) 건물 항목 생성

【코드 상세 설명】

1. ElementSelectionPanel.java

1) UI 및 로직 통합 구성

(1) JPanel을 상속받고 배경 이미지 및 폰트를 커스터마이징

- ① loadBackgroundImage()에서 이미지 리소스를 불러와 패널에 반복 렌더링
- ② loadCustomFont()에서 사용자 정의 폰트(ps.ttf)를 불러와 전역 적용

(2) 주요 컴포넌트 배치

① 날짜 선택: JSpinner + SpinnerDateModel, 최소 날짜는 2025년 5월 1일

- ② 시간 선택: JComboBox 두 개(시작, 종료), 5시~21시까지 30분 단위로 생성

③ 인원 입력: JTextField, 정수로 변환하여 예외 처리

④ 요소 선택: JCheckBox 배열로 제공, 선택 시 HashSet에 저장

(3) JButton("강의실 검색") 클릭 시 다음 동작 수행

① 사용자의 입력값을 MainFrame의 inputData에 저장

② 날짜는 LocalDate, 시간은 LocalTime, 요소는 Set<String>으로 전달

③ 요소 중 "계단형 강의실" 선택 여부는 isStepped로 별도 추출

④ backend의 Main.runReservationFlow() 메서드 실행

(4) 예외 처리

① 숫자가 아닌 인원 입력 시: NumberFormatException → 경고창

- ② 날짜 파싱 오류, 주말 예약 시 → WeekendException, 기타 오류 → 경고창 + 콘솔 출력

프론트엔드 파트 3

【화면 전환 흐름】

1. RecommendPanel ("추천")

- 1) 사용자가 추천된 강의실 목록을 확인할 수 있는 패널
- 2) 특정 강의실을 클릭하면 해당 강의실의 상세 정보를 보여주는 DetailPanel ("상세")로 전환

2. DetailPanel ("상세")

- 1) 선택한 강의실에 대한 이미지 및 주요 정보를 표시

- 2) “뒤로” 버튼 클릭 시 → 다시 추천 화면으로 이동
- 3) “예약하기” 버튼 클릭 시 → CompletePanel(“완료”)로 전환

3. CompletePanel (“완료”)

- 1) 예약 완료 메시지 출력
- 2) “홈으로 돌아가기” 버튼 클릭 시 → 다시 추천 화면으로 이동

→ 화면 전환은 CardLayout을 사용하여 각 패널 간 자연스럽게 이동할 수 있도록 구현함.

【패널별 코드 상세 설명】

1. BackgroundPanel

- 1) 공통 배경 이미지를 처리하는 패널로, 모든 화면의 배경으로 상속되어 사용됨.
- 2) 생성자에서 이미지 경로를 받아 paintComponent에서 반복 타일 형식으로 그려줌.
- 3) 투명 컴포넌트 위에도 배경이 유지되도록 함.
- 4) 로그인 화면, 추천 목록, 상세 정보, 완료 화면 모두 이 클래스를 기반으로 함.
- 5) 코드 중복 없이 배경 처리를 일괄적으로 담당함.

2. RecommendPanel

- 1) 추천 강의실 리스트를 보여주는 화면으로, RoomSelection 리스트를 받아 버튼 목록을 생성함.
- 2) 각 버튼은 강의실 이름 + 아이콘 형식이며 클릭 시 상세 패널로 이동하는 액션이 연결됨.
- 3) 내부는 BoxLayout과 JScrollPane을 사용해 스크롤 가능한 수직 리스트 구조로 구성됨.
- 4) 이미지가 없을 경우 기본 이미지로 대체되며, 좌측 아이콘+우측 텍스트 배치.
- 5) setRoomList 메서드로 추천 강의실 목록 갱신 가능함.
- 6) 사용자 선택 → 상세 정보 흐름의 중간 허브 역할.

3. DetailPanel

- 1) 선택된 강의실의 정보를 큰 이미지와 함께 보여주는 상세 정보 화면.
- 2) Room0 객체를 받아 건물명, 계단형 여부, 보유 시설 등을 라벨로 표시함.
- 3) “예약하기” 버튼 클릭 시 완료 패널로 이동함.
(MainFrame1.showPanel("complete")).
- 4) 정보 간선은 setRoom(Room0 room) 메서드에서 수행됨.
- 5) 강의실 사진은 ImageIcon으로 크기 조절하여 표시됨.
- 6) 추천 목록 → 상세 확인 → 예약 흐름의 핵심 역할.

4. CompletePanel

- 1) 예약 완료 메시지를 보여주는 최종 화면으로, 텍스트와 “처음으로” 버튼으로 구성됨.
- 2) 버튼 클릭 시 MainFrame1.showPanel("login")으로 되돌아가 초기화면으로 복귀함.
- 3) 사용자의 예약 흐름을 종료하고 다시 시작할 수 있는 루프 형성.
- 4) 디자인은 다른 패널과 동일하게 배경 이미지 + 중앙 정렬 방식 유지.
- 5) 사용자 여정의 마지막 단계로 심플하게 구성됨.

5. RoundedButton

- 1) 곡선 테두리를 가진 사용자 정의 버튼 컴포넌트.
- 2) paintComponent와 paintBorder를 오버라이드하여 둥근 배경과 테두리를 그림.
- 3) 클릭 및 포커스 시 색상 변화 없이 시각적으로 일관된 디자인 제공.
- 4) “예약하기”, “처음으로” 버튼 등 주요 동작 버튼에 사용됨.
- 5) UI 통일성과 버튼 스타일링을 간편하게 해주는 도구 역할.

6. FontUtil

- 1) 프로젝트 전반에 사용할 공통 폰트를 반환하는 유틸리티 클래스.
- 2) `getPsFont(float size)` 메서드는 지정한 크기의 PyeongChangPeace 폰트를 리턴함.
- 3) `ClassLoader`를 통해 리소스 경로에서 TTF 파일을 불러오고 캐싱함.
- 4) 폰트 로드 실패 시 기본 폰트로 대체됨.
- 5) UI 텍스트 크기 통일성과 디자인 품질 향상을 지원함.

7. Room1

- 1) 상세 정보 패널에서 사용하는 강의실 데이터 객체로, 필수 정보만 추려서 구성됨.
- 2) 필드에는 이름, 이미지 경로, 화이트보드 여부, 콘센트 여부, 마이크 여부, 계단형 여부가 포함됨.
- 3) `RoomSelection`에서 선택된 정보를 기반으로 생성됨.
- 4) 상세 화면에서 표시될 정보를 구조화하고 전달하는 역할.
- 5) 가공된 강의실 모델로 UI에 최적화된 데이터 제공.

8. MainFrame1

- 1) 전체 애플리케이션의 메인 프레임으로, `CardLayout`으로 각 패널을 전환함.
- 2) 생성자에서 로그인, 추천, 상세, 완료 패널을 초기화하고 카드에 등록함.
- 3) `showPanel(String name)`으로 특정 화면을 전환하고, `changePanel("상세", room)`으로 상세 정보 전달도 수행함.
- 4) 사용자 흐름(로그인 → 추천 → 상세 → 완료 → 로그인)을 관리함.
- 5) 앱의 흐름 제어 중심이자 컨트롤 타워 역할을 맡음.

V. 프로그램 발전 방향

1. 인공지능 발전 방향

1) 실시간 시간대 기반 추천

- (1) 개발 목표 : 사용자가 접속한 시간을 기준으로 즉시 이용 가능한 강의실을 추천하여 탐색 편의성 극대화
- (2) 사용 알고리즘 : K-Means Clustering, Rule-Based 필터링
- (3) 세부 내용
 - ① 다수 사용자의 예약 시간대 패턴 학습 → '오전형/오후형/야간형' 사용자 그룹 분류
 - ② 접속 시간에 해당 그룹이 가장 선호하는 강의실 우선적으로 제안

2) 사용자 선호 시설 기반 맞춤 추천

- (1) 개발 목표 : 사용자별 시설 선택 기록 분석해 개인에게 최적화된 맞춤형 강의실 추천하는 시스템 구현
- (2) 사용 알고리즘 : 협업 필터링 (KNN 기반), Matrix Factorization, Rule-Based 초기 대응
- (3) 세부 내용
 - ① 사용자별 시설 선택 기록 바탕으로 '사용자-시설 행렬(User-Facility Matrix)' 생성
 - ② 행렬을 기반으로 비슷한 조건 선호하는 다른 사용자들이 만족했던 강의실 예측하여 추천

3) 모바일 추천 알림

- (1) 개발 목표 : 사용자의 예약 패턴을 학습해 다음 예약 시점 예측 후 푸시 알림을 통해 사용자 편의와 프로그램 사용량 증가
- (2) 사용 알고리즘 : Prophet (시계열 예측), LSTM (딥러닝 기반 시계열 예측), Rule-Based 반복 예약 트리거
- (3) 세부 내용
 - ① 사용자의 반복적인 예약 패턴(요일, 시간 등)
 - ② 예측된 다음 예약 시점에 맞춰 푸시 알림 발송
 - ③ 반복되는 예약 시점 기준으로 앱 내에 자동 추천 배너 생성

2. DB 관련

1) 사용자 예약 기록, 현황/취소 기능 (사용자 아이디 관리)

(1) 개발 목표 : 사용자 아이디별로 예약 저장하고, 확인하며, 취소하는 등 편리한 관리 기능 제공

(2) 세부 내용

- ① 어떤 사용자가 예약했는지 알 수 있도록 사용자 정보와 예약 정보 연결
- ② 사용자는 '마이페이지'에서 자신이 예약한 내역 관리
- ③ 예약을 취소해도 기록은 바로 사라지지 않고 '취소' 상태로 남겨, 나중에 예약 내역 다시 확인 가능

2) 예약된 내역 저장해 다른 사용자의 예약 방지(DB 사용)

(1) 개발 목표 : 현재 프로그램에서 사용하는 임시 예약 저장 시스템을 업그레이드해서 예약 내역을 프로그램의 DB에 저장해 한 강의실이 같은 시간에 중복으로 예약되는 것 막아 시스템의 신뢰도 상승

(2) 세부 내용

- ① 새로운 예약 저장하기 전에, 먼저 그 시간에 다른 예약이 있는지 시스템이 자동으로 확인
- ② 두 사람이 동시에 같은 자리를 예약하려고 할 때, 단 한 사람의 예약만 성공하도록 처리하여 오류, 혼선 방지

3. Thread 관련

1) 예약 종료 임박 알림

(1) 사용 Thread 기능 : 지연 실행(Delayed Execution)

(2) 작동 방식

- ① 사용자가 예약 완료 시, 시스템은 '예약 종료 10분 전' 특정 시간 계산
- ② 해당 시간이 되면 푸시 알림 보내는 작업(Task) 생성
- ③ 이 작업을 백그라운드 스레드에서 동작하는 스케줄러(Scheduler)에 등록
- ④ 스케줄러는 정해진 시간이 될 때까지 기다렸다가, 정확한 시점에 작업을 실행시켜 알림 전송

2) 강의실 상태 현황 실시간 자동 동기화

(3) 사용 Thread 기능 : 주기적 실행(Periodic Execution)

(4) 작동 방식

- ① 사용자가 강의실 목록 화면에 들어오면, 1분마다 서버에 최신 정보를 업데이트하라는 작업(Task) 생성
- ② 백그라운드 스레드의 스케줄러(Scheduler)에 등록하여 1분 간격으로 반복 실행하도록 설정
- ③ 새로운 정보를 받아오면, 사용자가 보고 있는 화면(UI) 업데이트
→ 사용자가 일일히 새로고침하지 않아도 프로그램이 주기적으로 스스로 업데이트해 자동화 가능

4. 추가 기능

1) 인기 강의실 랭킹 시스템

2) 강의실 리뷰 시스템

VI. 제작 과정 & 느낀 점

1. DB) 고명주(2412185)

이번 프로젝트에서는 별도의 DBMS 없이 텍스트 파일 기반의 데이터베이스를 구현하기 위해, 직접 수작업으로 강의실 정보와 시간표 데이터를 수집하였습니다. 총 약 90여개의 강의실 시간표를 하나하나 확인하면서, 수업 유무와 가능한 시간대를 일일이 기록했고, 각 강의실의 내부 사진 역시 직접 건물을 돌아다니며 촬영해 확보하였습니다. 처음에는 단순한 데이터 입력이라고 생각했지만, 건물마다 구조가 달랐고, 사용 가능 여부도 모두 달라 세심한 확인이 필요했고, 예상보다 많은 시간이 소요되었습니다. 하지만 단순히 프로젝트를 위한 데이터가 아니라, 실제로 우리 학교 학생들이 사용할 수 있는, 현실적으로 도움이 되는 시스템을 만들고 싶다는 마음이 있었기에 그 과정을 끝까지 버틸 수 있었습니다. 정형화된 데이터가 아니라 실제 학교 환경을 반영한 정보였기 때문에, 프로그램의 실용성과 신뢰도가 높아졌다는 점에서 뿌듯함을 느꼈습니다. 데이터 수집 과정이 비록 힘들었지만, 그 안에서 사용자 입장을 깊이 이해하고, 프로그램이 실제 사용성과 연결될 때의 의미를 다시 한 번 느낄 수 있었습니다. 이 경험을 통해 단순한 기능 구현을 넘어, 사용자 중심의 서비스란 무엇인지 고민하게 된 소중한 시간이었습니다.

2. Backend

이번 프로젝트에서 저희는 백엔드 개발을 담당하여, 사용자 조건에 따라 적절한 강의실을 추천하고, 해당 강의실의 예약 가능 여부를 검증하며, 최종 예약 결과를 기록하는 시스템을 설계했습니다. 작업은 크게 두 파트로 나누어 진행되었습니다.

Part-1) 구나은 (2416133)

초기에는 단순히 조건에 맞는 강의실을 리스트에 담는 정도로 기능을 구현할 수 있을 거라 생각했습니다. 그러나 프로젝트가 진행되면서 요구 조건은 점점 복잡해졌고 단순한 조건 비교를 넘어, 교시별 사용 가능 여부와 날짜 기반 예약 충돌 방지, 계단형 여부, 내부 시설 조건 등 다양한 요소를 동시에 고려해야 했습니다.

이러한 문제를 해결하기 위해 단순한 로직 중심 사고에서 벗어나, 데이터 중심의 구조 설계로 접근 방식을 전환했습니다. 그 결과, Map<Room, Map<LocalDate, Set<Integer>>>와 같은 중첩된 자료 구조를 도입하게 되었고, 이를 통해 날짜별, 시간별로 강의실 예약 현황을 효과적으로 관리할 수 있게 되었습니다.

특히, Room, ReservationManager, ReservationRecord 등의 클래스를 직접 설계하면서, 클래스 하나는 하나의 책임만을 가져야 한다는 객체지향의 원칙을 자연스럽게 체득하게 되었습니다. 초기에는 기능 구현 속도에 집중했다면, 점차 구조와 역할 분리를 더 중요하게 생각하게 되었고, 이는 프로그램의 유지보수성과 확장성을 크게 향상시켜 주었습니다.

Part-2) 김나린 (2416835)

저는 사용자 입력 처리, 로그인 검증, 날짜/시간 유효성 검사, 예외 처리 등을 담당했습니다. 처음에는 입력만 잘 받아오면 된다고 생각했지만, 실제 사용자 환경을 고려하니 다양한 예외 상황이 존재했습니다. 예를 드러, 잘못된 학번 입력, 주말 예약 시도, 비어있는 시간 입력 등은 단순 조건문으로 처리하기엔 불안정하다는 문제점이 있었습니다.

이를 해결하기 위해 InvalidLoginException, WeekendException 등의 예외 클래스를 별도로 정의하고, LoginValidator, DateUtils와 같은 유틸리티 클래스를 만들어 입력의 유효성을 체계적으로 검증하고 예외에 책임을 부여했습니다.

또한, 사용자와 직접 맞닿는 흐름을 구성하며, 클래스의 이름, 메서드의 반환형, 처리 흐름이 프론트엔드 개발자에게 직관적어야 한다는 점도 크게 배웠습니다. 이는 협업의 핵심이기도 했고, 결과적으로 코드를 더 병확하고 재사용 가능한 구조로 만드는데 기여했습니다.

이번 백엔드 개발 과정은 단순히 “작동하는 코드”를 넘어서, 유지보수 가능한 구조, 확장 가능한 설계, 그리고 다른 개발자와 협업할 수 있는 코드 작성이라는 개발자의 핵심 역량을 훈련할 수 있는 소중한 경험이었습니다. 무엇보다도 기능 중심 사고에서 구조 중심 사고로 전환하는 계기가 되었고, 팀 프로젝트에서의 역할 분담과 협력의 중요성을 몸소 느낄 수 있었습니다.

5. Frontend

Part-1) 전여원(2413399)

이번 작업에서는 IntroPanel, StartPanel, LoginPanel, BuildingSelectionPanel 총 네 개의 화면을 구현하였고, 각 패널 간 전환은 CardLayout을 통해 구성하였습니다. 또한, 배경 이미지 반복 출력, 사용자 지정 폰트 적용, 버튼 hover 효과 등 시각적 요소들도 직접 처리하였습니다. 구현 중 가장 어려웠던 점은 각 패널에서 공통으로 적용되는 디자인 요소들을 일관되게 유지하면서도, 기능적으로 충돌 없이 작동하게 만드는 것이었습니다. 특히 로그인 화면에서는 학번 유효성 검사를 포함한 예외 처리 부분에서 세심한 로직 설계가 필요했고, 이벤트 리스너를 활용한 사용자 반응 구현에도 많은 테스트가 필요했습니다. 이러한 과정을 통해 GUI 프로그램의 세부적인 사용자 흐름 제어와 UI 일관성 유지의 중요성을 깊이 있게 체감할 수 있었습니다.

Part-2) 최지은(2414320)

Ui/Ux에 관심이 생겨 프론트엔드를 맡아 팀 프로젝트를 진행하였습니다. 프론트엔드 역할을 맡아, 사용자가 선택한 조건에 따라 강의실이 필터링되는 강의실 필터링 화면을 구현했습니다. 처음에는 기본적인 초기 화면을 구성했으며, 이후 팀 내 프론트엔드 인원들과 화면을 디자인하고 그 디자인을 바탕으로 통합하는 작업을 진행했습니다. 프론트엔드에서 화면 디자인까지 맡아 구현했던 만큼, 단순한 기능 구현을 넘어 디자인 통일성과 사용자 경험(UI/UX)에 대한 고민도 함께 할 수 있었습니다.

얼추 화면이 완성된 후엔, event를 구현하기 위해 사용자가 입력한 값을 변수로 저장하는 과정을 진행했습니다. 단지 화면만 구성하면 안 되고, 백엔드와의 연결을 위한 변수 처리까지 해줘야 비로소 그게 프론트가 할 일이었습니다. 이 점을 프로젝트를 하면서 깊이 깨달았습니다.

또한 이번 프로젝트는 개발에 있어서 협업과 의사소통의 중요성을 체감할 수 있었던 경험이었습니다. 코드를 혼자 짜는 것과 달리, 여러 명이 함께 협업하며 코드를 작성하는 일은 생각보다 훨씬 어려웠습니다. 변수 이름이나 자료형 같은 기본적인 요소부터, 전체적인 코드 스타일까지도 혼자만의 기준이 아닌 팀원들과 충분한 소통을 통해 맞춰 나가야 했고, 제가 작성한 코드를 여러 차례 수정해야 하는 경우도 많았습니다. 이러한 과정을 거치며 실제 프로젝트에서 협업과 의사소통 능력이 얼마나 중요한지 절실히 깨달았습니다. 단순히 기능을 구현하는 데서 끝나는 것이 아니라, 서로의 코드를 이해하고 존중하며 함께 완성해나가는 과정이 필요했습니다.

또한 이번 팀 프로젝트는 단순한 과제가 아니라, 하나의 웹/앱이 개발되는 전반적인 과정을 직접 체험할 수 있었던 값진 경험이었습니다. 프로젝트 기획부터 업무 배분, 그리고 실제 코드를 짜고 최종 통합까지 하는 전체 과정에 대해 배울 수 있었습니다. 물론 쉽지만은 않았지만, 그만큼 많은 것을 배우고 성장할 수 있었기에 힘들면서도 매우 의미 있고 가치 있는 시간이었습니다.

Part-3)

6. Main) 임수정(2410786)

이번 프로젝트에서 저는 연동 역할을 하는 `main`을 담당했습니다. 사용자의 입력 값을 `ReservationInputData`에 순차 저장하고, 이 데이터를 기반으로 강의실 추천이 실제로 동작하도록 전체 로직을 설계·연결하는 역할이었습니다. `RoomRecommender`, `ReservationManager`, `RoomFileReader` 등 주요 백엔드 클래스들과의 연결 지점을 조율하면서, 단순한 실행 코드가 아닌 시스템 흐름의 중심축을 구축해야 했습니다. 특히 어려웠던 점은 팀원들이 각자 구현한 코드의 작성 방식이나 로직 구조가 조금씩 달랐기 때문에, 통일성 있는 연동을 위해 하나하나 세부 흐름을 파악하고 정리해야 했다는 점이었습니다. 효율적으로 수정 요청 사항을 전달하고, 소통에 오해가 없도록 하기 위해선 제 자신이 먼저 전체 흐름을 더 깊이 이해해야 했고, 이 과정이 생각보다 많은 시간과 집중력을 필요로 했습니다.

또한 익숙하지 않은 파일 구조 관리에서 경로 설정이나 클래스 참조 오류가 자주 발생했고, 여러 화면(UI)과 기능이 연결되는 구조 속에서 작은 실수가 전체 시스템 오류로 이어질 수 있기에 부담도 컸습니다. 특히 텍스트 기반의 강의실 데이터가 방대하고 구조화되지 않아, 이를 안정적으로 읽고 필터링하는 과정에서 어려움이 많았고, 그만큼 데이터 연동의 중요성을 다시금 체감했습니다.

무엇보다 이 작업을 하면서 팀워크와 협업의 중요성을 느꼈습니다. 제가 맡은 역할은 모든 화면과 기능이 자연스럽게 이어질 수 있도록 중심에서 조율하는 것이었기 때문에, 단순한 기술적 구현을 넘어 책임감이 중요하다는 것을 체감할 수 있었습니다. 개발자 간의 원활한 협력 없이는 아무리 기능이 좋아도 작동하지 않는다는 사실을 직접 겪으며, 협업의 기본기와 소통 역량의 중요성을 다시 생각해보는 계기가 되었습니다.

VII. 기대 효과

이 프로그램은 사용자가 입력한 조건에 따라 현재 사용할 수 있는 강의실을 빠르게 확인하고, 원하는 공간을 손쉽게 선택할 수 있도록 도와주는 시스템입니다. 각 강의실의 특성과 설비 요소를 시각적으로 직관적으로 확인할 수 있어 정보 접근성이 뛰어납니다. 로그인부터 건물 선택, 요소 필터링, 추천 결과 확인까지 일관된 UI 흐름을 제공하여 사용자는 복잡한 절차 없이 쉽게 예약 과정을 마칠 수 있습니다. 특히 버튼 **hover**, 클릭 등 사용자와의 인터랙션 요소가 적극 반영되어 있어 사용성 측면에서도 만족도가 높습니다. 학번 유효성 검사와 조건 필터링을 통해 잘못된 접근을 방지하며, 기본적인 데이터 신뢰성을 확보할 수 있습니다. 강의실 예약을 위한 시간과 노력을 절약할 수 있으며, 건물별 가용 강의실에 대한 정보를 한눈에 파악할 수 있어 교내 자원의 효율적 활용을 도모할 수 있습니다. 또한, 학내 구성원 간 강의실 이용 혼선이나 중복 사용을 예방할 수 있는 기반을 마련합니다. 향후 시스템을 학사 행정 시스템과 연동할 경우, 실사용 측면에서의 범용성과 확장성도 매우 높은 편입니다. 전체적으로 본 프로그램은 학생 중심의 실용성과 관리 편의성을 모두 갖춘 예약 기반 정보 시스템으로 기능할 수 있습니다.

이러한 기본 시스템 위에 향후 인공지능 기반 기능을 추가함으로써, 사용자 경험을 더욱 향상시킬 수 있습니다. 접속 시간과 예약 패턴을 분석하여 실시간으로 이용 가능한 강의실을 추천하고, 시설 선택 기록을 활용한 협업 필터링으로 개인화된 추천도 가능해집니다.

또한 반복적인 예약 시점을 예측하여 푸시 알림이나 자동 추천 배너를 제공함으로써 사용자의 편의성과 프로그램 활용도를 동시에 높일 수 있습니다.