**High-level Architecture**

**&**

**UML Diagrams**

시대평

팀명:SoGoing

팀원:2020920032 석우진

2020920039 이강민

2020920055정구홍

2020920059조종빈

2021120009김고은

2022920027박상윤

변 경 이 력

|  |  |  |  |
| --- | --- | --- | --- |
| 버전 | 일자 | 변경 내역 | 작 성 자 |
| 1.0 | 2024.11.16 | Class Diagram, Activity Diagram작성 | 박상윤 |
| 1.1 | 2024.11.16 | Class Diagram 수정 | 박상윤, 이강민 |
| 1.2 | 2024.11.16 | Activity Diagram 수정 | 박상윤, 이강민 |
| 2.0 | 2024.11.17 | Sequence Diagram 작성 | 이강민 |
| 3.0 | 2024.12.8 | Class Diagram 수정 | 박상윤 |
| 3.1 | 2024.12.8 | Sequence Diagram 수정 | 박상윤 |
| 3.2 | 2024.12.8 | UC-010 Sequence Diagram 추가 | 박상윤 |

Table of Contents

[1. High Level Architecture](#_Toc178012895)

[1.1 Monolithic Architecture](#_Toc178012896)

[1.2 Microservices Architecture](#_Toc178012897)

[1.3 Trade-off](#_Toc178012898)

[1.4 Our Architecture: N-layer Architecture](#_Toc178012899)

[2. Class Diagram for static view](#_Toc178012900)

[2.1 Class Diagram Overview](#_Toc178012898)

[2.2 Class Description](#_Toc178012898)

[3. Sequence Diagrams for dynamic view](#_Toc178012901)

[3.1 Usecase Table](#_Toc178012898)

[3.2 Sequence Diagrams](#_Toc178012898)

[4. Activity Diagrams for dynamic view](#_Toc178012901)

[4.1 User Activity Flow](#_Toc178012898)

[4.1 Student Activity Flow](#_Toc178012898)

[4.2 Staff Activity Flow](#_Toc178012898)

# High Level Architecture

## Monolithic Architecture

* 정의: 모든 기능이 하나의 애플리케이션 내에서 통합되어 작동하는 전통적인 아키텍처
* 구성요소:
  + Fronted: 웹 브라우저에서 사용 가능한 UI 제공
  + Backend: Spring을 이용한 API 서버와 데이터 처리 로직
  + Database: PostgreSQL을 통해 데이터 관리
* 장점
  + 개발 및 배포가 단순
  + 통합 테스트가 용이
  + 낮은 초기 개발 비용
* 단점
  + 확장성 제한: 애플리케이션이 커지면 변경이 복잡해지고 배포 시 모든 기능을 재배포해야 함
  + 성능 한계: 각 컴포넌트 간의 높은 결합으로 인해 특정 기능에서 과부하 발생 시 전체 성능에 영향을 미침

텍스트, 스크린샷, 양초이(가) 표시된 사진

자동 생성된 설명

## Microservices Architecture

* 정의: 독립적으로 배포 가능한 여러 서비스로 시스템을 분할하여 기능 별로 나눠진 아키텍처
* 구성요소:
  + Frontend: 웹 UI는 동일하지만, API를 여러 서비스에서 제공
  + Backend: 각 기능 별 독립적인 서비스로 분할
  + Database: 각 서비스마다 별도의 DB를 사용하여 독립성 유지
* 장점
  + 확장성: 서비스 별로 독립적인 배포가 가능해, 각 서비스의 성능에 따라 별도로 확장 가능
  + 유지 보수 용이성: 각 기능이 독립적이므로, 한 기능의 수정이 다른 서비스에 미치는 영향이 최소화됨
* 단점
  + 복잡한 배포 및 관리: 여러 개의 서비스로 분할되어, 배포가 복잡하고 통합 테스트가 필요
  + 추가적인 인프라 비용: 각 서비스에 대한 별도 관리 및 서비스 간 통신으로 인한 비용

텍스트, 스크린샷, 디자인이(가) 표시된 사진

자동 생성된 설명

## Trade-off

텍스트, 스크린샷, 라인, 도표이(가) 표시된 사진

자동 생성된 설명

* 복잡성 vs 유연성
  + Monolithic: 낮은 복잡성, 제한된 유연성
  + Microservices: 높은 복잡성, 뛰어난 유연성
* 개발 속도 vs 확장성
  + Monolithic: 초기 개발이 빠르지만, 확장이 어려움
  + Microservices: 초기 개발이 느리지만, 확장이 쉬움

## Our Architecture: N-layer Architecture

* 정의: 애플리케이션의 기능들을 논리적인 계층으로 분리하여, 각 계층이 특정 역할과 책임을 담당하도록 구성된 아키텍처
* 계층
  + Presentation Layer: 사용자 인터페이스 및 사용자 상호작용 처리
  + Business Logic Layer: 비즈니스 규칙 및 핵심 로직 구현
  + Data Access Layer: 데이터베이스 조작 및 데이터 접근 로직
* 장점
  + 관심사의 분리: 각 계층이 독립적인 책임을 가짐
  + 확장성: 각 계층을 독립적으로 확장 가능
  + 테스트 용이성: 각 계층을 독립적으로 테스트 가능
  + 유지 보수 용이성: 각 계층이 독립적인 책임을 가지므로, 유지 보수가 용이
  + 보안성: 중요 비즈니스 로직 보호
* 선택 이유
  + 명확한 관심사 분리
  + 유지보수성 향상
  + 코드 재사용성 증가
  + 테스트 용이성

텍스트, 폰트, 스크린샷, 번호이(가) 표시된 사진

자동 생성된 설명

# Class Diagram for Static View

## 2.1 Class Diagram Overview

도표, 텍스트, 평면도, 기술 도면이(가) 표시된 사진

자동 생성된 설명

## 

## Class Description

1. **Actor Class**

* 시스템을 실제로 사용하는 사용자 유형을 나타내는 클래스

**User**

* 시스템의 사용자 정보를 가지고 있는 추상 클래스
* 속성
  + 식별 정보
  + 계정 정보
* 주요 기능
  + 로그인/로그아웃
  + 사용자 정보 업데이트
  + 회원 탈퇴

**Student**

* User 클래스를 상속받는 학생 클래스
* 속성
  + 학생 정보
* 주요 기능
  + 사용자 정보 업데이트
  + 부서에 지원하기
  + 부서 찜하기
  + 부서에 후기 남기기

**Staff**

* User 클래스를 상속받는 부서 관리자 클래스
* 속성
  + 부서 관리자 정보
* 주요 기능
  + 부서 관리
  + 지원서 관리
  + 부서 관리

1. **Entity Class**

* 시스템에서 데이터로 관리되는 핵심 객체들을 나타내는 클래스
* 데이터베이스의 테이블로 매핑

**Department**

* 부서 정보

**Resume**

* 지원서 정보

**Feedback**

* 부서 후기 정보

**Dibs**

* 사용자가 찜한 부서

# Sequence Diagram for Dynamic View

## Usecase Table

|  |  |
| --- | --- |
| 유스케이스 ID | 유스케이스 |
| UC-001 | 회원가입 |
| UC-002 | 로그인 |
| UC-003 | 부서 목록 열람 |
| UC-004 | 부서 페이지 열람 |
| UC-005 | 부서 후기 작성 |
| UC-006 | 지원서 작성 |
| UC-007 | 부서 페이지 수정 |
| UC-008 | 지원자 명단 확인 |
| UC-009 | 프로필 수정 |
| UC-010 | 부서 찜하기 |

## Sequence Diagrams

**1. 회원가입**

**텍스트, 도표, 스크린샷, 번호이(가) 표시된 사진

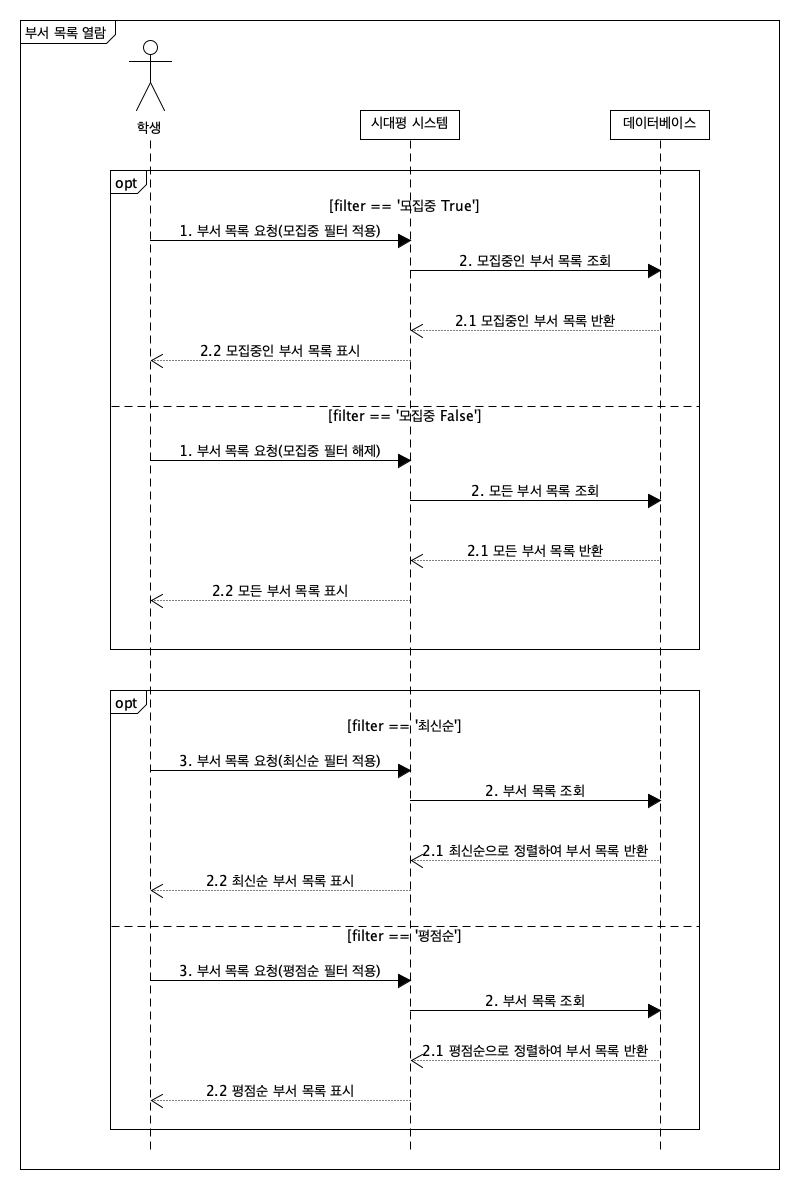
자동 생성된 설명**

**2. 로그인**

텍스트, 도표, 스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명

**3. 부서 목록 열람**

****

**4. 부서 페이지 열람**

**텍스트, 도표, 라인, 평행이(가) 표시된 사진

자동 생성된 설명**

**5. 부서 후기 작성**

텍스트, 도표, 스크린샷, 번호이(가) 표시된 사진

자동 생성된 설명

**6. 지원서 작성**

텍스트, 도표, 평행, 번호이(가) 표시된 사진

자동 생성된 설명

**7. 부서 페이지 수정**

텍스트, 도표, 스크린샷, 번호이(가) 표시된 사진

자동 생성된 설명

**8. 지원자 명단 확인**

텍스트, 도표, 평행, 번호이(가) 표시된 사진

자동 생성된 설명

**9. 프로필 수정**

**텍스트, 도표, 번호, 스크린샷이(가) 표시된 사진

자동 생성된 설명**

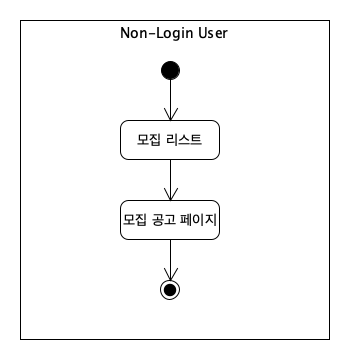
**10. 부서 찜하기**

**텍스트, 스크린샷, 도표, 번호이(가) 표시된 사진

자동 생성된 설명**

# Activity Diagram for Dynamic View

## User Activity Flow



## Student Activity Flow

텍스트, 도표, 평행, 평면도이(가) 표시된 사진

자동 생성된 설명

## Staff Activity Flow

텍스트, 도표, 평행, 라인이(가) 표시된 사진

자동 생성된 설명