**소프트웨어 UML Diagrams Document**

**수강신청 도우미**

**시대 플랜메이트**

2019920026 서웅진

2019920011 김민서

2020920018 류재욱

2021910033 정명훈

2021920017 김영신

**변 경 이 력**

|  |  |  |  |
| --- | --- | --- | --- |
| **버전** | **일자** | **변경 내역** | **작 성 자** |
| B\_001 | 2024-11-10 | UML Diagrams Document | 서웅진 |
| B\_002 | 2024-11-16 | (sequence diagram, architecture diagram) 추가 | 서웅진 |
| B\_003 | 2024-11-17 | Static diagram 추가 | 서웅진 |
|  |  |  |  |
|  |  |  |  |

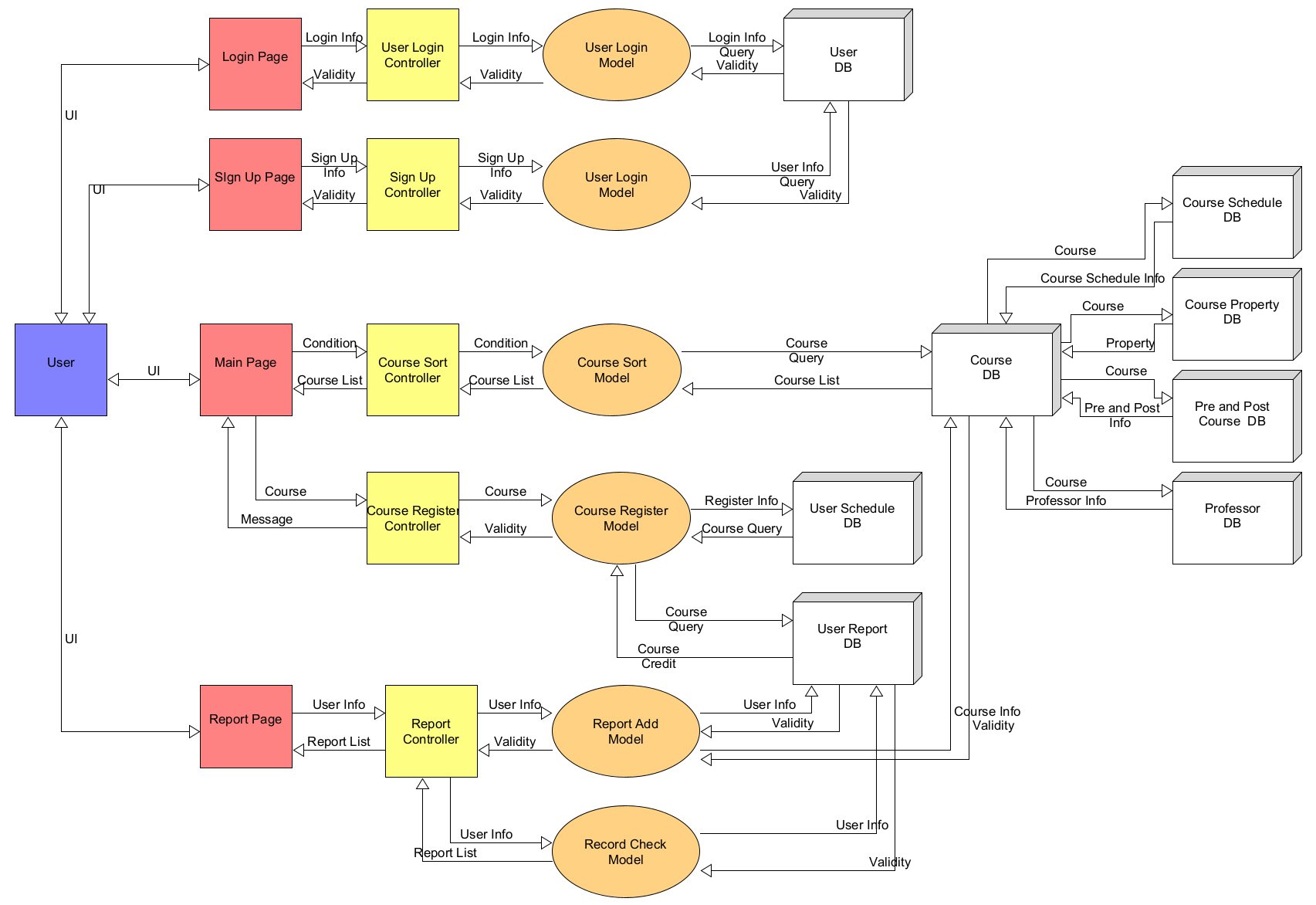
**Table of Contents**

**1. High-level Architecture Diagram 소개 및 설명**

**2. Class Diagrams for static view 소개 및 설명**

**3. Sequence Diagrams for dynamic view 소개 및 설명**

**1. High-level Architecture Diagram 소개 및 설명**



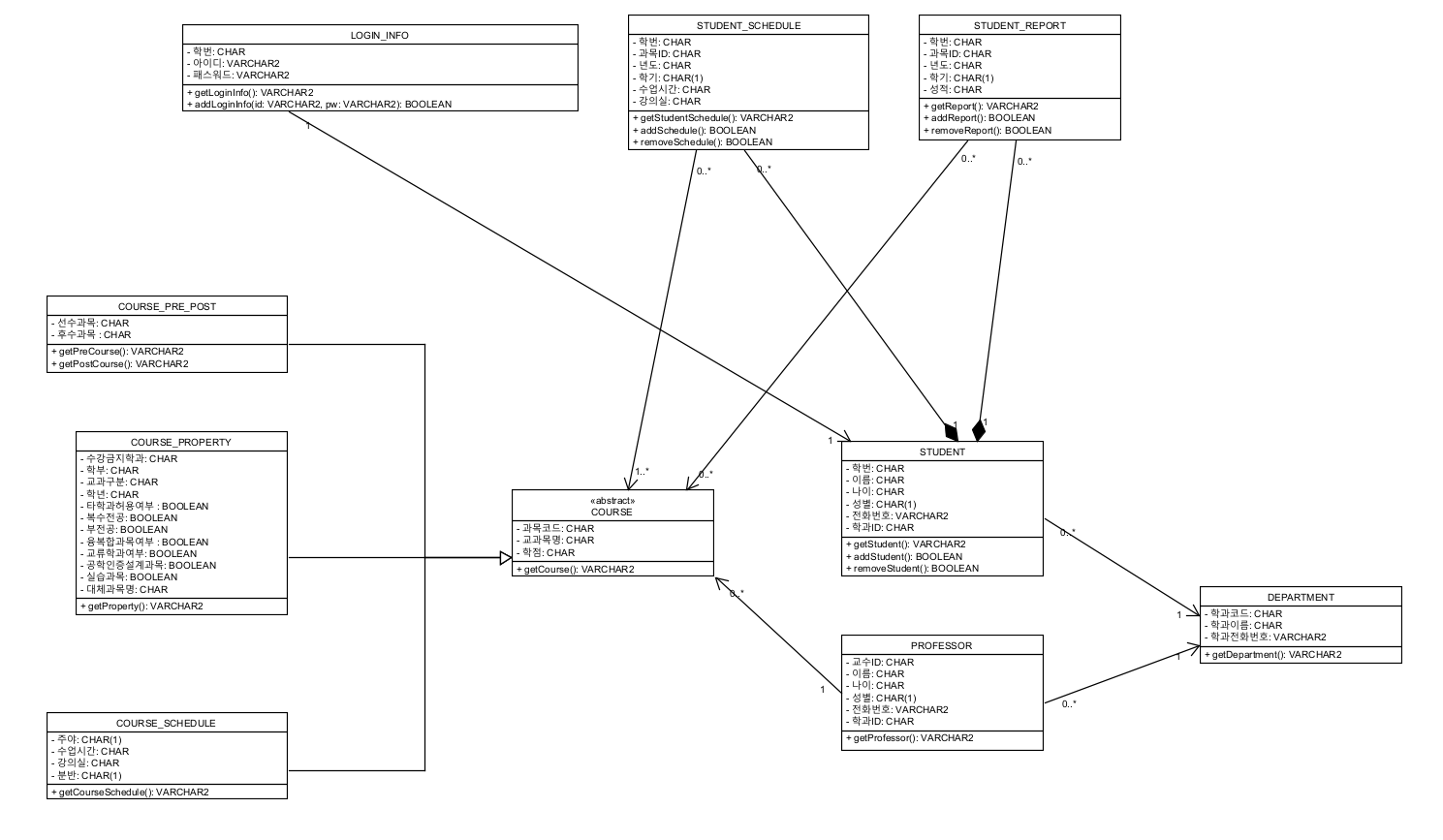
MVC Architecture Diagram

저희 조는 웹 개발이 목적이기에 MVC Style을 Architecture로 선정하였습니다.

이전에 선정한 Pipe & Filter Style의 경우 강력한 재사용성을 기반으로 선정하려 했으나, User와의 상호작용이 주를 이루는 저희 웹개발의 특성상 Pipe & Filter Style의 경우는 부적절하다고 판단하였습니다. 따라서 이러한 상호작용과 시스템 확장성이 용이한 Event-based Style 기반의 MVC Style을 선정하였습니다.

User-View-Controller-Model-DB의 구성으로 Architecture을 만들었습니다. MVC Style의 특성으로 인해 View와 Controller, Model이 서로 연결되어 있는 부분 간의 상호작용만 존재하기에 화면을 출력하는 View와 데이터를 처리하는 Model이 독립적으로 운영됩니다. 이러한 독립성으로 인해 코드 간의 종속성이 줄어들고, 구성요소 간의 역할이 명확합니다. 이러한 특징은 코드 분리가 쉽고 협업을 용이하게 할 수 있습니다.

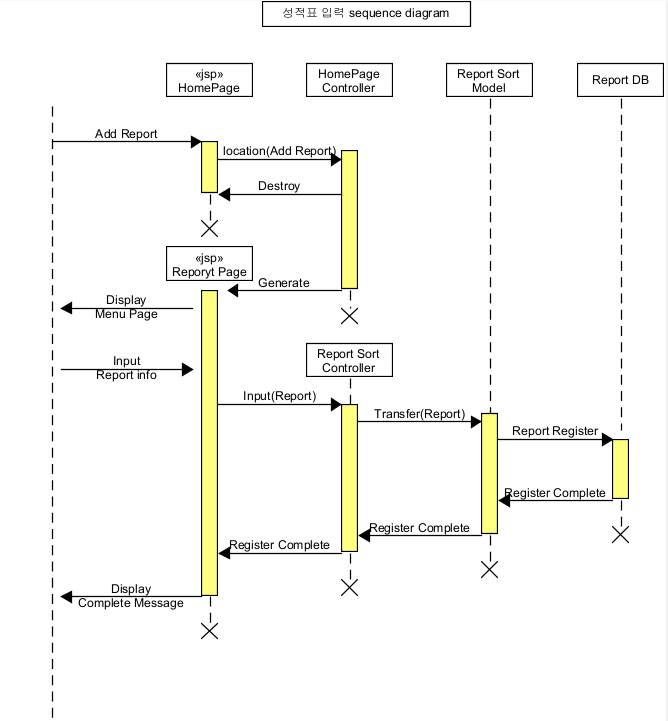
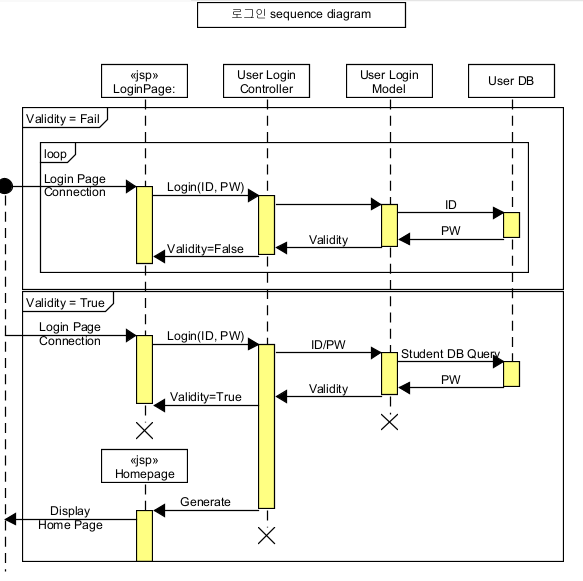
**2. Class Diagrams for static view 소개 및 설명**



위 다이어그램에서 STUDENT\_SCHEDULE과 STUDENT\_REPORT는 각각 시간표와 성적표를 담당합니다. 그리고 COURSE의 경우 반복적으로 QUERY문으로 사용되는 속성들은 COURSE\_PROPERTY로 테이블을 따로 만들어 관리하고 COURSE\_SCHEDULE도 마찬가지로 분리하였습니다. COURSE\_PRE\_POST의 경우 과목이 폐지될 수도 있는 COURSE의 특성상 자기 참조를 피하기 위해 따로 테이블로 분리하였습니다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **클래스** | | **관계 유형형** | **설명** | **Multiplicity** |
| Course | CoursePrePost | 일반화(Generalization) | Course는 추상 클래스이며, CoursePrePost, CourseProperty, CourseSchedule 클래스가 이를 상속받음. |  |
| CourseProperty | 일반화(Generalization) |  |
| CourseSchedule | 일반화(Generalization) |  |
| Student | Department | 연관(Association) | 학생은 하나의 학과에 속하며, 하나의 학과에는 여러 학생이 있을 수 있음음 | • Student → Department: 1  • Department → Student: 0..\* |
| Professor | Department | 연관(Association) | 교수는 하나의 학과에 속하며, 하나의 학과에는 여러 교수가 있을 수 있음 | • Professor → Department: 1  • Department → Professor: 0..\* |
| Student | StudentReport | 구성(Composition) | 성적표는 학생의 일부로, 학생이 없으면 존재 불가. 한 명의 학생은 여러 성적표를 가질 수 있지만, 각 성적표는 반드시 한 명의 학생에게 속 | • Student → StudentReport: 0..\*  • StudentReport → Student: 1 |
| StudentReport | Course | 연관(Association) | 각 성적표는 여러 과목의 성적을 포함할 수 있고, 하나의 과목은 여러 학생의 성적표에 기록될 수 있음 | • StudentReport → Course: 0..\*  • Course → StudentReport: 0..\* |
| Student | StudentSchedule | 구성(Composition) | 시간표는 학생의 일부로, 학생이 없으면 존재 불가. 한 명의 학생은 여러 시간표를 가질 수 있지만, 각 시간표는 특정 학생에게만 속함 | • Student → StudentSchedule: 0..\*  • StudentSchedule → Student: 1 |
| StudentSchedule | Course | 연관(Association) | 학생은 여러 과목을 시간표에 추가할 수 있고, 하나의 과목은 여러 학생의 시간표에 추가될 수 있음 | • StudentSchedule → Course: 1..\*  • Course → StudentSchedule: 0..\* |
| Professor | Course | 연관(Association) | 한 명의 교수는 여러 과목을 가르칠 수 있으며, 각 과목마다 담당 교수가 있음 | • Professor → Course: 0..\*  • Course → Professor: 1 |
| LogIn\_Info | Student | 연관(Association) | 각 로그인 정보는 특정 학생 한명에게 속하며, 각 학생은 고유한 로그인 정보를 갖게됨 | • LogIn\_Info → Student: 1  • Student → LogIn\_Info: 1 |

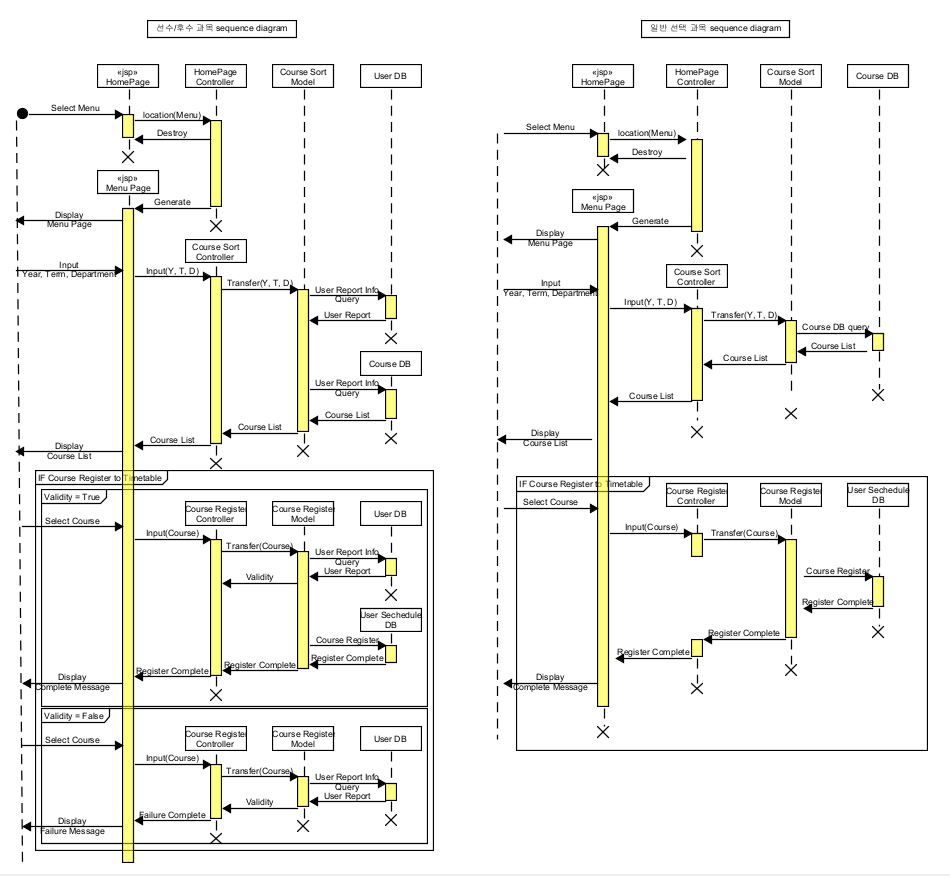
**3. Sequence Diagrams for dynamic view 소개 및 설명**

****

로그인 Sequence Diagram 성적표 Sequence Diagram

로그인 event는 로그인 페이지에서 ID, PW를 입력 받아 USER DB에서 비교 후 실패 시 로그인 페이지를 반복하고 성공 시 Home Page로 이동하도록 설계하였습니다.

성적표 입력 event는 Home Page에서 성적표 입력을 클릭하여 Report Page 이동 후 입력 한 정보를 Report DB에 Insert 하도록 설계하였습니다.



선수/후수 과목 Sequence Diagram 일반 선택 과목 Sequence Diagram

6개의 usecase는 이벤트의 흐름이 학생의 성적표/시간표를 요구하는지에 따라 크게 2개로 나뉩니다. 따라서 위와 같이 2개의 대표 sequence diagram을 설계하였습니다.

학생의 성적표/시간표를 요구하지 않는 일반 선택 과목 정렬, 수강 금지 과목 필터링, 실습 과목 및 공학인증설계 과목 정렬 usecase의 sequence는 다음과 같습니다.

학생이 원하는 usecase를 선택하여 해당 페이지로 이동 후 년도, 학기, 과 정보를 입력하여 query된 list를 확인합니다. 여기서 학생이 시간표 추가를 원할 시 과목을 선택하여 시간표DB에 insert 후 완료 message를 display합니다.

다음으로 학생의 성적표/시간표를 요구하는 선수/후수 과목 정렬, 재수강 과목조회, 시간별 과목 필터링 usecase의 sequence는 아래와 같습니다.

학생이 원하는 usecase를 선택하여 해당 페이지로 이동 후 년도, 학기, 과 정보를 입력하여 query된 list를 확인합니다. 여기서 학생이 시간표 추가를 원할 시 학생의 성적표DB에서 성적표를 query하여 수강 불가능 시 실패 message를 display하고 가능 시 선택된 과목을 시간표DB에 insert 후 완료 message를 display합니다.