

# 멘토링 이후 변경 사항 및 설문 분석

졸업 프로젝트 주제: SW 자율학습 강의 Docker+웹 서비스 개발

종합설계1\_02분반

야간자율학습

고태완  
김지혜  
최수연

# 멘토링 그 이후

- 현재 메인 시스템이 3개로 규모가 너무 큼 (웹기반 동영상강의플랫폼, 퀴즈-채점, 도커를 이용한 실습 환경)
- 도커를 이용한 실습 환경에 초점을 맞추기로 결정(선택과 집중)
- 이미 시장에서 잘 사용되고 있는 이론 강의에 대한 서비스 및 퀴즈는 최종 프로젝트에서 삭제
- 실습 환경을 이미지로 저장하여 수강생들 개개인에게 하나의 컨테이너로 제공
  - => OS에 대한 접근이 필요한 실습에서 수강생은 root 권한으로 자유롭게 실습 진행 가능, 단 1회용 컨테이너, 실습이 끝난 후에 삭제됨
  - => Shell과 Code Editor를 웹페이지에서 제공.
    - 단 Shell을 사용하지 않아도 되는 경우(OS에 접근하지 않아도 되는 경우) Shell을 제공하지 않음
    - Code Editor를 사용할 경우에 채점을 하지 않고 컴파일 된 결과만 보여줌.
  - => 기본적으로 사용자가 업로드한 실습 강의에 대해서 저장된 실습의 난이도를 이용해 사용자에게 수준별 학습을 제공할 수 있도록 Curation한다.
  - => 특정 강사가 특정 사용자들에 대해서만 실습을 가능하게 할 것을 원하는 경우, 이 실습은 private 상태가 되며 특정 강사가 저장한 환경으로 사용 권한을 가진 수강생 들에 한해서 해당 환경으로 바로 실습을 진행 할 수 있도록 한다.
- Docker와 Kubernetes를 이용한 Orchestration 방식의 유연한 서버 운영
- CI(Continuous Integration) 기반의 소스 코드 에디터 사용. 소스코드를 올리면 빌드 후 에러 포함한 결과 리포트를 사용자에게 보여줌.

위와 같은 이유로 웹기반 동영상강의플랫폼, 퀴즈-채점은 제거  
도커를 이용한 실습 환경 중점적으로 구현 => SE문서에 좀 더 상세하게 기록

# 멘토링 그 이후

- 현재 메인 시스템이 3개로 규모가 너무 큼 (웹기반 동영상강의플랫폼, 퀴즈-채점, 도커를 이용한 실습 환경)
- 도커를 이용한 실습 환경에 초점을 맞추기로 결정(선택과 집중)
- 이미 시장에서 잘 사용되고 있는 이론 강의에 대한 서비스 및 퀴즈는 최종 프로젝트에서 삭제
- 실습 환경을 이미지로 저장하여 수강생들 개개인에게 하나의 컨테이너로 제공
  - => OS에 대한 접근이 필요한 실습에서 수강생은 root 권한으로 자유롭게 실습 진행 가능, 단 1회용 컨테이너, 실습이 끝난 후에 삭제됨
  - => Shell과 Code Editor를 웹페이지에서 제공.
    - 단 Shell을 사용하지 않아도 되는 경우(OS에 접근하지 않아도 되는 경우) Shell을 제공하지 않음
    - Code Editor를 사용할 경우에 채점을 하지 않고 컴파일 된 결과만 보여줌.
  - => 기본적으로 사용자가 업로드한 실습 강의에 대해서 저장된 실습의 난이도를 이용해 사용자에게 수준별 학습을 제공할 수 있도록 Curation한다.
  - => 특정 강사가 특정 사용자들에 대해서만 실습을 가능하게 할 것을 원하는 경우, 이 실습은 private 상태가 되며 특정 강사가 저장한 환경으로 사용 권한을 가진 수강생 들에 한해서 해당 환경으로 바로 실습을 진행 할 수 있도록 한다.
- Docker와 Kubernetes를 이용한 Orchestration 방식의 유연한 서버 운영
- CI(Continuous Integration) 기반의 소스 코드 에디터 사용. 소스코드를 올리면 빌드 후 에러 포함한 결과 리포트를 사용자에게 보여줌.

위와 같은 이유로 웹기반 동영상강의플랫폼, 퀴즈-채점은 제거  
도커를 이용한 실습 환경 중점적으로 구현 => SE문서에 좀 더 상세하게 기록



문서 재작성 및 프로토타입 재제작

# 문서

---

1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스다이어그램
5. 시퀀스다이어그램

1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스다이아그램
5. 시퀀스다이아그램

```

    usecaseDiagram
        actor Client
        actor Server
        usecase UC1[회원가입]
        usecase UC2[로그인]
        usecase UC3[아이디 찾기]
        usecase UC4[비밀번호 찾기]
        Client --> UC1
        Client --> UC2
        Client --> UC3
        Client --> UC4
        Server --> UC2
        Server --> UC3
        Server --> UC4
    
```

```
graph LR
    Client[Client] --> UC1((강좌 등록))
    subgraph System
        UC1 --> UC2((테스트 생성))
        UC2 -.->|확장| UC1
    end
    UC1 --> Server[Server]
```

The diagram illustrates the system's components and interactions. A stick figure labeled 'Client' is connected to a use case labeled '강좌 등록' (Course Registration). This use case is part of a larger system boundary. Inside the system boundary, there is another use case labeled '테스트 생성' (Test Generation). A solid arrow points from '강좌 등록' to '테스트 생성', and a dashed arrow labeled '확장' (Extension) points back from '테스트 생성' to '강좌 등록'. Finally, an arrow points from '강좌 등록' to a rectangular box labeled 'Server'.

1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스다이아그램
5. 시퀀스다이아그램

```

    usecaseDiagram
        actor Client
        actor Server
        participant System as 사용자 식별
        usecase UC1 as 회원가입
        usecase UC2 as 로그인
        usecase UC3 as 아이디 찾기
        usecase UC4 as 비밀번호 찾기
        Client --> UC1
        Client --> UC2
        Client --> UC3
        Client --> UC4
        Server --> UC1
        Server --> UC2
        Server --> UC3
        Server --> UC4
    
```

The diagram illustrates the interactions between a Client and a Server with a central system boundary labeled '사용자 식별' (User Identification). The Client and Server are represented by stick figures. The system boundary contains four use cases: '회원가입' (Sign Up), '로그인' (Login), '아이디 찾기' (Find ID), and '비밀번호 찾기' (Find Password). Arrows indicate that both the Client and the Server can interact with all four use cases.

```

    usecaseDiagram
        title 강의 수강 System
        actor Client
        actor Server
        usecase LevelTest as 레벨 테스트
        usecase LectureRecommend as 강의 추천
        usecase Learning as 학습
        usecase ReviewTest as 복습 테스트
        usecase LectureEvaluation as 강의 평가
        usecase PracticeLearning as 실습 학습
        usecase TheoryLearning as 이론 학습
        usecase EnvSetting as 환경 설정
        usecase MandatoryGuideline as 강제 가이드라인
        usecase AutoEnvSetting as 자동 환경설정

        Client --> LevelTest
        Client --> LectureRecommend
        Client --> Learning
        Client --> ReviewTest
        Client --> LectureEvaluation

        LevelTest --> LectureRecommend : 수강
        Learning <|-- PracticeLearning
        Learning <|-- TheoryLearning
        PracticeLearning --> EnvSetting : 수강
        EnvSetting --> MandatoryGuideline
        EnvSetting --> AutoEnvSetting
        TheoryLearning --> LectureEvaluation : 수강
        ReviewTest --> LectureEvaluation : 수강
  
```

```

    usecaseDiagram
        actor Client
        participant PasswordReset as 강좌 비밀번호
        participant PasswordGenerator as 테스트 생성
        participant Server

        PasswordGenerator -.-> PasswordReset : 확장
        Client --> PasswordReset
        PasswordReset --> Server
    
```

The diagram illustrates the password reset process. A stick figure labeled 'Client' is on the left. A large rectangle labeled '강좌 앱로드' (App Load) contains two use cases: '테스트 생성' (Test Generation) and '강좌 비밀번호' (Password Reset). A dashed arrow labeled '확장' (Extension) points from '테스트 생성' to '강좌 비밀번호'. An arrow points from the 'Client' to the '강좌 비밀번호' use case. Another arrow points from the '강좌 비밀번호' use case to a rectangle labeled 'Server' on the right.

# 주와

```

sequenceDiagram
    participant Client
    participant Serv
    Client->>Bar
    Bar->>G1_1[장고 이어서 수강]
    Bar->>G1_2[장고 프로젝트 확인]
    Bar->>G1_3[장고 질문]
    Bar->>G1_4[장고 질문 답변]
    Bar->>G1_5[장고 질문 확인]
    Bar->>G1_6[장고 질문 답변 확인]
    Bar->>G2_1[장고 질문 답변 확인]
    Bar->>G2_2[장고 질문 답변 확인]
    Bar->>G2_3[장고 질문 답변 확인]
    Bar->>G2_4[장고 질문 답변 확인]
    Bar->>G2_5[장고 질문 답변 확인]
    Bar->>G2_6[장고 질문 답변 확인]
    Bar->>G3_1[장고 질문 답변 확인]
    Bar->>G3_2[장고 질문 답변 확인]
    Bar->>G3_3[장고 질문 답변 확인]
    Bar->>G3_4[장고 질문 답변 확인]
    Bar->>G3_5[장고 질문 답변 확인]
    Bar->>G3_6[장고 질문 답변 확인]
    Bar->>Serv
  
```

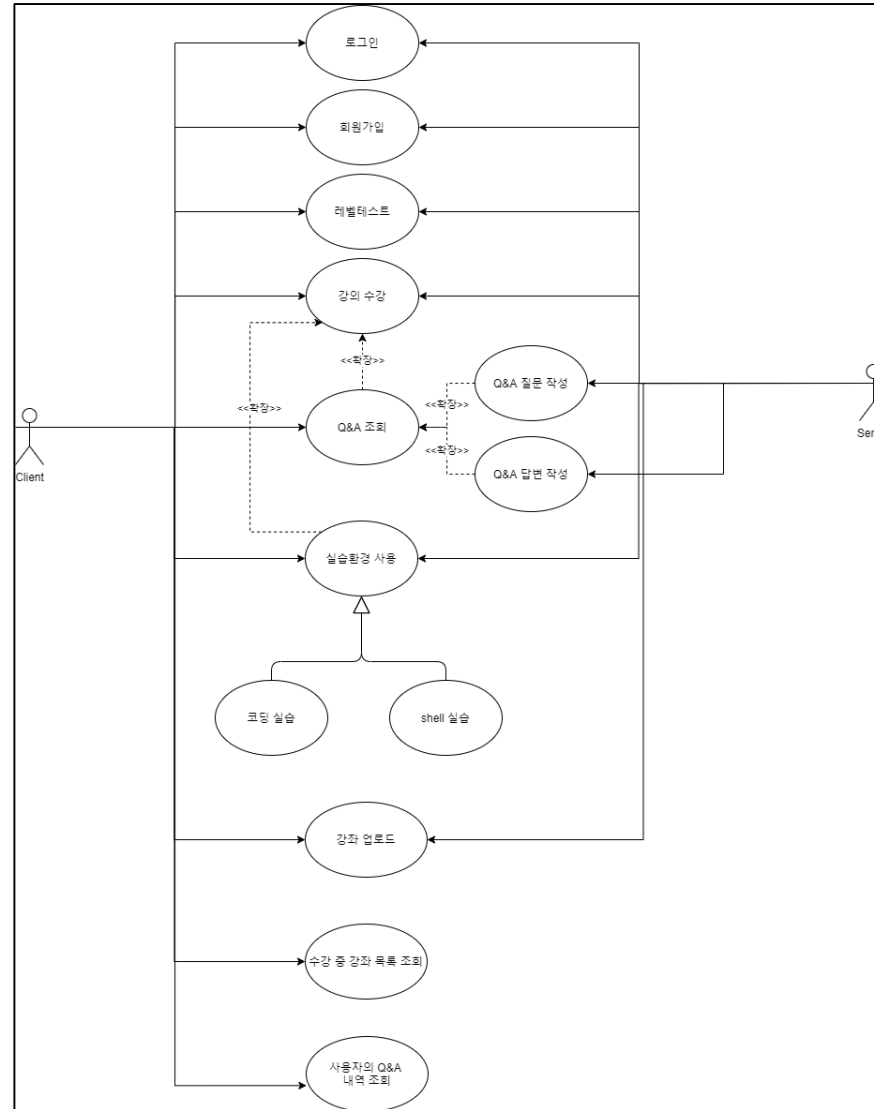
3.

# 간소화

# 문서

1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스다이어그램
5. 시퀀스다이어그램

하나의 시스템  
Sub System X



# 문서

1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스 다이어그램
5. 시퀀스 다이어그램

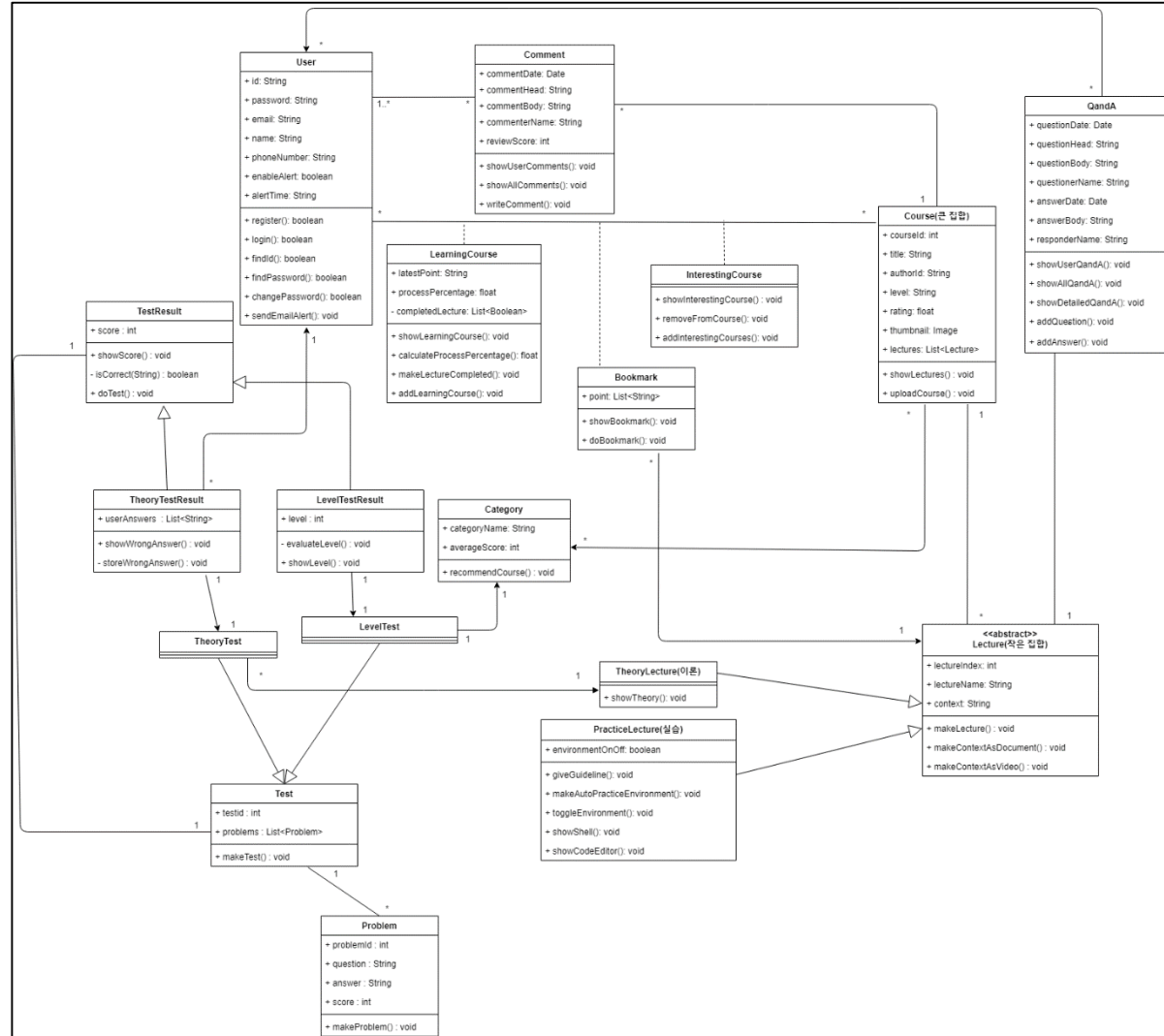
28개 -> 12개의 Specification, 핵심(실습 환경 구축)에 집중

3. Usecase Specification.....	6
3.1. 회원 가입 .....	6
3.2. 로그인 .....	7
3.3. 레벨 테스트.....	8
3.4. 강의 수강 .....	9
3.5. 코딩 실습 .....	10
3.6. Shell 실습 .....	11
3.7. Q&A 조회.....	12
3.8. Q&A 질문 작성 .....	13
3.9. Q&A 답변 작성 .....	14
3.10. 강좌 업로드 .....	15
3.11. 수강 중 강좌 목록 조회.....	17
3.12. 사용자의 Q&A 내역 조회.....	18



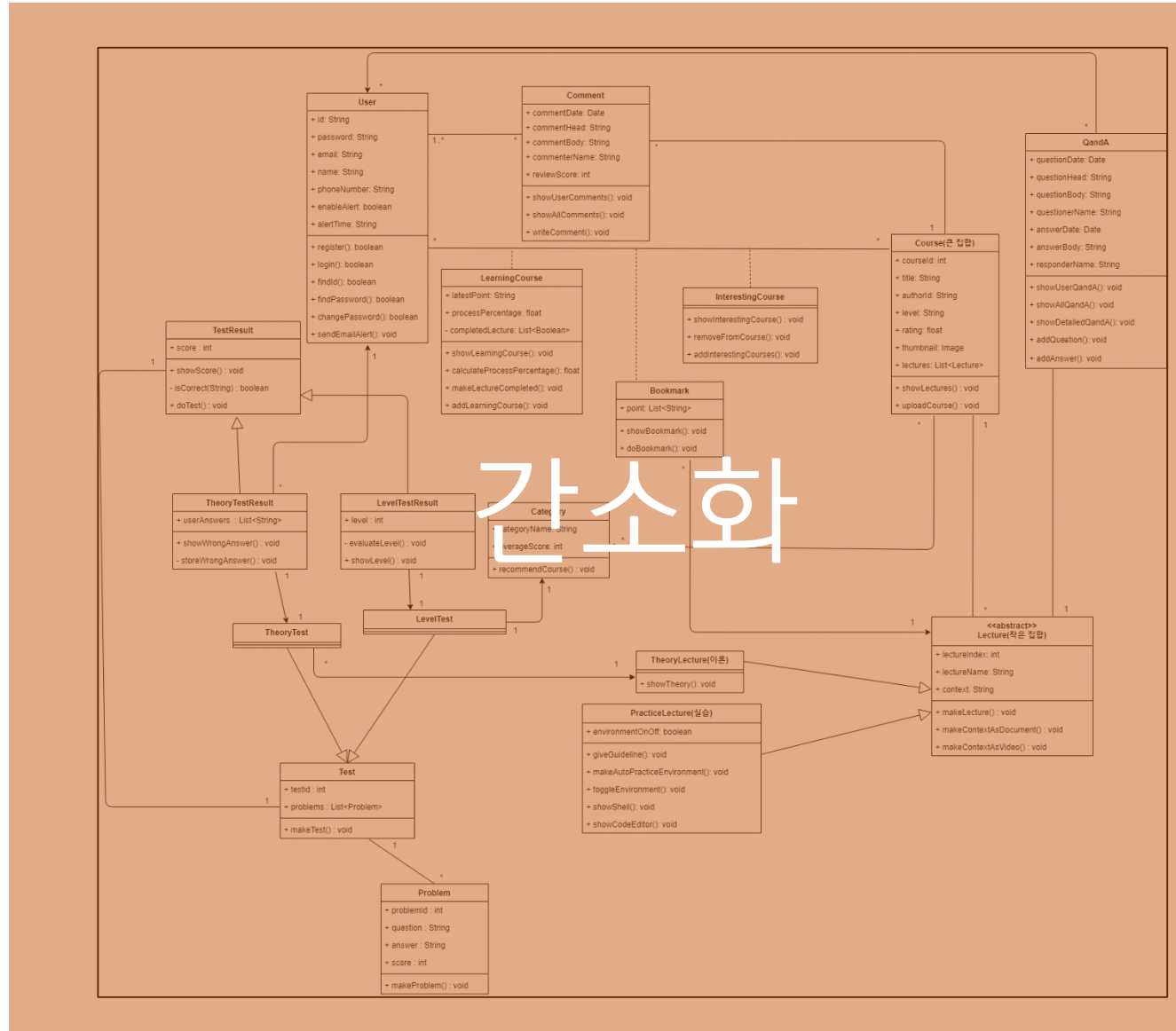
# 문서

1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스다이어그램
5. 시퀀스다이어그램



# 문서

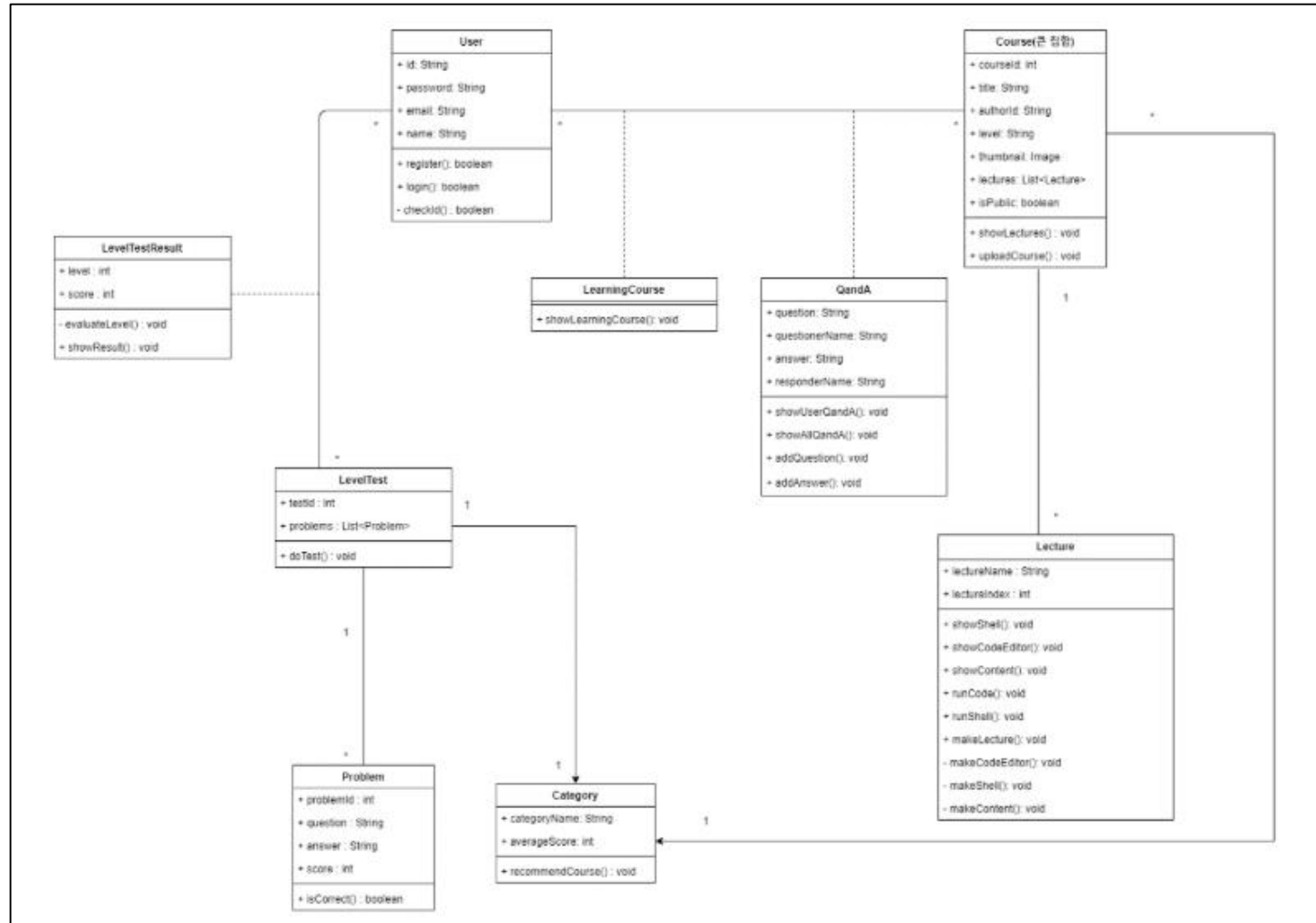
1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스 다이어그램
5. 시퀀스 다이어그램



# 문서

1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스 다이어그램
5. 시퀀스 다이어그램

18개 -> 9개의 Specification, 핵심(실습 환경 구축)에 집중



# 문서

1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스다이어그램
5. 시퀀스다이어그램

## Table of Contents

### 목차

1. INTRODUCTION .....	5
1.1. OBJECTIVE .....	5
2. USE CASE DIAGRAM .....	6
3. SEQUENCE DIAGRAM .....	9
3.1. 회원가입 .....	9
3.2. 로그인 .....	11
3.3. 아이디 찾기 .....	12
3.4. 비밀번호 찾기 .....	13
3.5. 레벨 테스트 .....	14
3.6. 강의 추천 .....	15
3.7. 이론 학습 .....	16
3.8. 복습 테스트 .....	17
3.9. 강의 평가 .....	18
3.10. 실습 학습 .....	19
3.11. 환경 가이드라인 .....	21
3.12. 자동 환경설정 .....	22
3.13. 강좌 업로드 .....	23
3.14. 테스트 생성 .....	26
3.15. 수강 중 강좌 목록 조회 .....	28
3.16. 강좌 이어서 수강 .....	29
3.17. 오답노트 확인 .....	30
3.18. 찜한 강좌 조회 .....	31
3.19. 찜한강좌와 수강 .....	32
3.20. 찜한 강좌 삭제 .....	33
3.21. 강좌 재검토 조회 .....	34
3.22. 강의 재검토 사용 .....	35
3.23. Q&A 내역 조회 .....	36
3.24. Q&A 내역 상세 조회 .....	37
3.25. 비밀번호 변경 .....	38

종합설계 1

3.26. 강좌 알림 관리 .....	40
3.27. 이메일 수신 관리 .....	41

# 문서

1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스다이어그램
5. 시퀀스다이어그램

## Table of Contents

### 목차

1. INTRODUCTION .....	5
1.1. OBJECTIVE .....	5
2. USE CASE DIAGRAM .....	6
3. SEQUENCE DIAGRAM .....	9
3.1. 회원가입 .....	9
3.2. 로그인 .....	11
3.3. 아이디 찾기 .....	12
3.4. 비밀번호 찾기 .....	14
3.5. 레벨 테스트 .....	14
3.6. 강의 추천 .....	16
3.7. 이론 학습 .....	16
3.8. 복습 테스트 .....	17
3.9. 강의 평가 .....	18
3.10. 실습 학습 .....	19
3.11. 환경 가이드라인 .....	21
3.12. 자동 환경설정 .....	22
3.13. 강좌 업로드 .....	23
3.14. 테스트 생성 .....	26
3.15. 수강 중 강좌 목록 조회 .....	28
3.16. 강좌 이어서 수강 .....	29
3.17. 오답노트 확인 .....	30
3.18. 찜한 강좌 조회 .....	31
3.19. 찜한강좌와 수강 .....	32
3.20. 찜한 강좌 삭제 .....	33
3.21. 강좌 책갈피 조회 .....	34
3.22. 강의 책갈피 사용 .....	35
3.23. Q&A 내역 조회 .....	36
3.24. Q&A 내역 상세 조회 .....	37
3.25. 비밀번호 변경 .....	38

종합설계 1

3.26. 강좌 알림 관리 .....	40
3.27. 이메일 수신 관리 .....	41

간소화

# 문서

1. 문제정의서
2. 요구사항명세서
3. 유스케이스
4. 클래스다이어그램
5. 시퀀스다이어그램

1.1. OBJECTIVE.....	4
<b>2. USE CASE DIAGRAM.....</b>	<b>5</b>
<b>3. SEQUENCE DIAGRAM .....</b>	<b>6</b>
3.1. 회원가입 .....	6
3.2. 로그인 .....	8
3.3. 레벨 테스트.....	9
3.4. 강의 수강.....	11
3.5. 코딩 실습.....	12
3.6. SHELL 실습 .....	13
3.7. Q&A 조회.....	14
3.8. Q&A 질문 작성 .....	15
3.9. Q&A 답변 작성 .....	16
3.10. 강좌 업로드.....	17
3.11. 수강 중 강좌 목록 조회.....	20
3.12. 사용자의 Q&A 내역 조회.....	21

# 프로토타입 V2

---

프로토타입 Version 2.

Level Test기능을 남긴 채 실습 환경에 관한 기능 위주로 구현  
Code Editor, Shell, Result Windows 로 실습 환경 지원

강좌를 업로드할 때 실습 환경을 저장하여 수강생들에게 링크로 배포할 수 있음  
(Public, Private 등 공개 여부 설정 가능)

그 외 자잘한 편의 기능 삭제 => 추후 구현 가능

# 프로토타입 V2

---

Demo: <https://youtu.be/rGqttuH5E8s>





# 설문조사

SW와 관련이 있는 지인들에게 요청, 6명이 응답

## Code-At-Once 프로토타입 설문조사

안녕하세요. 종합설계 02분반 야간자율학습 팀입니다.

저희는 "SW자율학습 웹서비스+ 실습 환경 제공"이라는 주제로 졸업프로젝트를 진행하고 있습니다.

첨부된 영상은 4분 15초로, 한번 시청하신 후 설문해주시면 감사하겠습니다.

code-at-once는 codacademy, codesandbox, 백준 사이트처럼 실습 환경 및 코드 에디터를 제공합니다. 사용자는 환경설정에 대한 시간을 투자하지 않고 온전히 실습에 집중할 수 있습니다.

위 서비스들은 코드 에디터만 제공하지만, code-at-once는 shell을 제공합니다. 따라서 진짜 OS를 사용하는 것처럼 직접 Library를 설치할 수 있고 환경 변수, 커널, OS단에서 설정할 수 있는 값들 또한 수정할 수 있습니다. 각각의 shell은 유저 개개인에게 독립적으로 동작합니다.

코드 에디터에서는 내가 입력한 코드를 가상의 환경에서 컴파일하여 결과 값을 웹에서 보여줍니다.

code-at-once는 특정 유저가 환경을 설정하여 배포할 수 있습니다. 이를 Private으로 설정하여 강좌를 생성할 경우 공유 가능한 링크를 통해 배포된 환경이 복제되어 링크를 이용하는 사용자가 사용할 수 있습니다.

\* 필수항목

# 설문조사

SW와 관련이 있는 지인들에게 요청, 6명이 응답

## Code-At-Once

안녕하세요. 종합설계 02분반 야간자  
저희는 "SW자율학습 웹서비스+ 실습  
입니다.

첨부된 영상은 4분 15초로, 한번 시청

code-at-once는 codacademy, code  
공합니다. 사용자는 환경설정에 대한

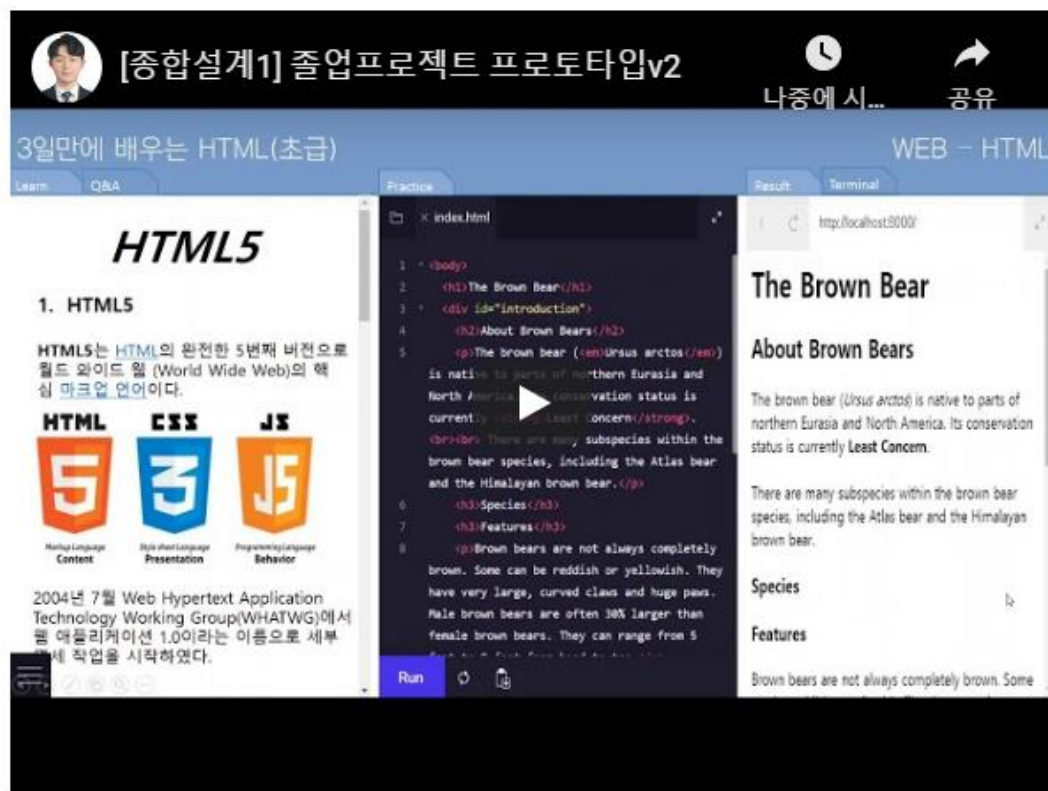
위 서비스들은 코드 에디터만 제공하  
를 사용하는 것처럼 직접 Library를  
값들 또한 수정할 수 있습니다. 각각

코드 에디터에서는 내가 입력한 코드가  
니다.

code-at-once는 특정 유저가 환경을  
강좌를 생성할 경우 공유 가능한 링  
가 사용할 수 있습니다.

\* 필수항목

## 프로토타입 영상



# 설문조사

---

1. 만약 환경설정의 번거로움을 해소하여 바로 코딩할 수 있는 서비스가 있다면 사용하시겠습니까?
2. codecademy, codesandbox.io, 백준과 같은 SW 학습 서비스를 사용해본 적이 있습니까?
  - 2-1. 위와 같은 SW 학습 서비스를 사용하면서 불편한 점이 있다면 적어주세요.
3. 영상에서 본 프로토타입 중 학습 과정에 대해서 추가되면 좋겠다, 혹은 없어도 되겠다라고 생각하는 기능이 있습니까?
4. 본 서비스가 출시된다면 사용할 의향이 있습니까?

# 설문조사

## 설문 결과



## 결론

사용자들은 기존의 다양한 기능을 제공할 때 보다 부족한 점을 느끼지 못하였고, 좀 더 전문화된 실습 환경 서비스 구축으로 프로젝트가 전환이 되었다.

**야간 자율학습**