Ruby on Rails 기반
스터디 관리 웹 서비스 설계 및 구현

Ruby on Rails-based study management web service design and implementation

본 보고서를 Expert Project 과제의 **최종보고서로 제출합니다.**

〈 요 약 문 〉

<table>
<tr><td>설계과제
목표</td><td>- Ruby on Rails 기반 스터디 관리 웹 서비스 설계 및 구현

- 스터디 파이가 어떤 서비스인지 어떻게 만들어졌는지 확인

- rails 프레임워크의 장점과 왜 사용하는지 확인

- 웹사이트를 구현하기 위한 메뉴구조도, 주요 화면 설계, DB설계 하기</td></tr>
<tr><td>내용</td><td>- 스터디 파이는 온라인으로 혼자 공부 해 보고 싶고 수많은 자료 중 어떤 자료로 공부해야 하는지 고민 하고 있는 사람들을 위한 온라인 기반 스터디 교육 플랫폼

- 스터디 파이 서비스 주요기능
• 사용자 관리(회원가입 / 로그인/ 회원정보변경/ 회원탈퇴)
• 스터디 개설(진행자 제안 및 스터디 승인)
• 스터디 신청
• 스터디 관리(Q&A, 과제 업로딩, 출석체크, 자료 업로딩)
• 스터디 참여(Q&A, 과제 제출, 스터디 후기)

- Scaffold를 통해 index, show, new, create, update, destroy 메소드를 생성

- gemfile 수정 및 devise 설치를 통해 로그인/회원가입 구현

- Controller 생성, Routes 업데이트, link_to를 활용하여 메인페이지를 만들어 원하는 페이지로 이동할 수 있도록 구현

- 스터디 개설을 위하여 경로를 추가함

- 스터디 신청을 위하여 파라미터 값으로 필요한 id를 가져옴

- 사용자가 신청한 스터디 목록을 보여주는 페이지를 생성</td></tr>
<tr><td>기대효과</td><td>- 오프라인이 아닌 온라인 스터디 서비스이므로 웹사이트를 통해 쉽게 이용 가능

- 구현 한 스터디 웹사이트에 들어가게 되면 메인화면에 다양한 주제의 스터디를 모집하고 있는 스터디 카테고리가 있어 신청을 원하는 사람은 쉽게 신청 가능

- Ruby on Rails 기반으로 웹 서비스를 구현하여 화면의 이동이 자유롭게 이루어 짐</td></tr>
<tr><td>Keywords</td><td>스터디 파이 | Ruby on Rails | Scaffold
사용자 관리 | 스터디 개설 | 스터디 신청</td></tr>
</table>

목 차

I. 서론

1. 배경

100만 명의 MOOC 참가자를 대상으로 한 펜실베니아 대학의 연구에 따르면 14%가 과정을 완전히 이수하였으며, 평균적으로 4%만이 효과가 있는 것으로 나타났다[1]. 이 연구 결과는 공부하고자 하는 열망은 가득하지만 혼자 끝까지 공부하는 것은 어렵다는 것을 의미한다.

그렇다면 사람들은 왜 공부하고자 하는 것일까?

초/중/고/대학교 약 16년간 공부를 했지만, 실무에서 사용할만한 지식/지혜에 대한 학습은 매우 부족하다. 또한, 이직 주기, 직업이 없어지고 생겨나는 주기가 빨라지고, 평생직장의 개념이 사라지면서 사람들은 막연히 미래에 대해 불안감을 가지게 되었다. 이러한 변화로 인해 인생의 전반적인 계획과 목표를 이루기 위해서는 한가지만을 추구하는 것이 아닌 새로운 역량을 쌓아 나가야 한다.

본 과제는 교육을 통해 기업이 원하는 인재상과 실제 구직자들의 역량이 일치하지 않는 불균형을 없애고 공부를 끝까지 하게 도와줌으로써 실적인 실력향상이 이뤄지도록 초점을 둔 '스터디파이'를 모티브로 Ruby on Rails 기반 스터디 관리 웹 서비스를 설계 및 구현하였다.

2. 스터디파이

스터디파이는 온라인으로 혼자 공부해보고 싶고 온라인에 수많은 자료 중 어떤 자료로 공부해야 하는지 고민하고 있는 사람들을 위한 온라인기반 스터디 교육 플랫폼이다. 스터디파이는 온라인, 오프라인 교육의 장점만 살릴 수는 없을까?라는 생각으로, 온라인에서 오프라인과 유사한 시스템(정해진 시간, 체계적인 커리큘럼, 주간 과제, 다른 사람들과 함께 학습, 내가 모르는 부분을 알려줄 수 있는 강사/멘토, 스터디를 완주했을 때 주어지는 보상)을 이용해서, 사람들이 끝까지 학습하도록 도와주고 있다. 이 방식을 통해, 평균 55%의 사람들이 스터디파이와 함께 끝까지 학습하고 있다[2].

그림 1. 스터디파이

3. Ruby on Rails

Ruby on Rails(이하 Rails)는 Ruby 환경에서 사용할 수 있는 웹 애플리케이션 프레임워크이다. Rails는 개발자가 웹 애플리케이션을 개발하기 시작할 때 필요할 것으로 생각되는 작업이나 리소스를 가정하고 미리 준비하여 웹 애플리케이션을 쉽게 작성할 수 있도록 설계되어 있다. Rails의 철학에는 중요한 이념이 두 가지 있다[3].

- Don't Repeat Yourself : DRY
- Convention Over Configuration

DRY는 소프트웨어 개발 원칙 중 하나며, '시스템을 구성하는 자식의 모든 컴포넌트는 항상 하나여야 하며, 명확하고, 신뢰할 수 있는 형태로 표현하지 않으면 안된다'라는 의미이다. 반복적인 코드를 철저하게 피하는 것으로, 코드를 유지보수하기 쉽게 하고, 간단히 확장할 수 있게 되며, 버그를 줄일 수 있다.

Rails에서는 웹 애플리케이션에서 실행될 다양한 기능들을 실현하기 위한 최선의 방법을 명확히 구상하고 있으며, 웹 애플리케이션의 각종 설정에 대해서도 기존의 경험이나 관습에 기초해, 각 설정들의 기본값을 정해두고 있다. 독단적으로 결정된 기본값 덕분에, 개발자가 설정파일을 설정할 필요가 없다.

Rails는 MVC 패턴(Model-View-Controller)이라 불리는 아키텍처를 사용한다. MVC 패턴이란, 애플리케이션을 모델(Model, 비즈니스 로직), 뷰(View, 사용자 인터페이스), 컨트롤러(Controller, 모델과 뷰 제어)와 같이 수행하는 역할로 명확히 구분해서 구성하는 것을 말한다. [그림 2]는 Rails에서 MVC 패턴 흐름을 나타낸 것이다[4].

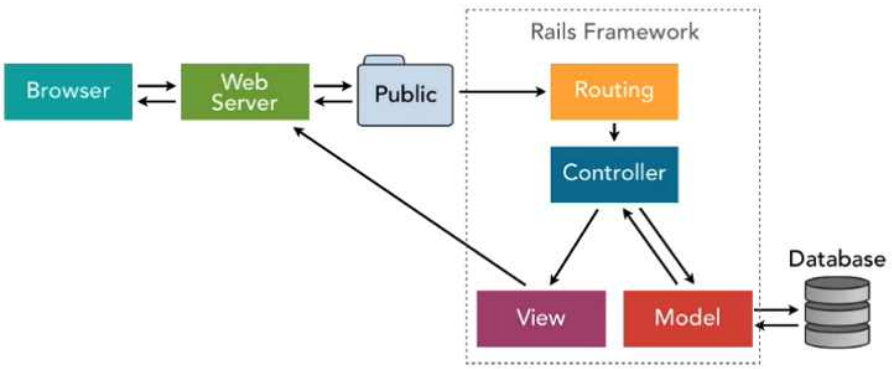


그림 2. Rails 아키텍처

또한, Rails는 RESTful 인터페이스를 지원한다. RESTful 인터페이스는 리소스의 위치를 URI로 나타내고 리소스에 CRUD(Create, Read, Update, Delete)을 할지 HTTP 메소드(GET, POST, PATCH, DELETE)로 나타내는 방법을 의미한다. RESTful 인터페이스를 사용하면 통일감 있고 의미가 명확한 URL 설계가 가능하다.

II. 과제 수행 내용

1. 서비스 설계

1.1 서비스 주요 기능 정의

서비스 주요 기능은 다음과 같다.

- ❍ 사용자 관리(회원가입 / 로그인 / 회원정보변경 / 회원탈퇴)
- ❍ 스터디 개설(진행자 제안 및 스터디 승인)
- ❍ 스터디 신청
- ❍ 스터디 관리(Q&A, 과제 업로딩, 출석체크, 자료 업로딩)
- ❍ 스터디 참여(Q&A, 과제 제출, 스터디 후기(설문조사, 랭킹))

사용자 관리에 대해서는 세부적으로 회원가입, 로그인, 비밀번호 찾기, 회원정보변경, 로그아웃, 회원탈퇴의 기능을 포함하고 있으며, 회원 가입 시에는 이메일 형식으로 가입이 가능하다. 이후 로그인을 통해 스터디 제안 및 신청이 가능하다. 로그인 정보는 세션을 통해 저장이 되어, 동일한 사람이 요청했는지 확인이 가능하다.

스터디 개설은 모든 사용자가 개설 신청을 할 수 있으며, 관리자에 의해 스터디 승인이 이루어진다. 스터디 개설 승인이 수락되면 관리자와 제안한 사용자를 제외하여 스터디 신청을 할 수 있다.

스터디 운영자. 즉, 스터디를 제안한 사용자는 출석체크, 자료 업로딩, 과제 업로딩, Q&A 서비스 등을 할 수 있으며, 스터디 참여자는 스터디 운영자가 올린 자료를 확인하여 스터디에 참가할 수 있다. 스터디가 종료되면 모든 사용자(운영자, 참여자) 후기를 등록하여 서로 평가할 수 있다. 평가 데이터에 기반하여 랭킹 및 추천 시스템을 구축할 수 있다.

1.2 DB 설계

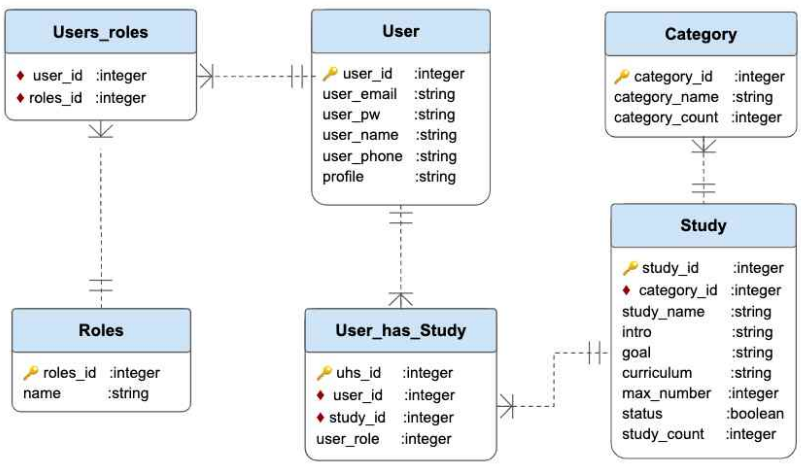


그림 3. ERD Diagram

[그림 3]은 ERD Diagram을 나타낸 것이다. User, Study 테이블을 중심으로 Study 테이블과 Category 테이블은 일대다 관계로 category_id가 Study 테이블에 외래키로 컬럼이 추가되어있다.

User 테이블과 Study 테이블, User 테이블과 Roles 테이블은 다대다 관계로 매핑 테이블이 각자 추가되어 있다. Roles 테이블의 name 컬럼을 통해 일반 사용자인지 관리자인지 구분이 되며, Users_roles 테이블을 통해 각 User에 대한 권한이 부여된다.

User_has_study 테이블을 통해 일반 사용자와 스터디에 대한 연결 관계가 생성이 되며, user_role 컬럼을 통해 스터디 제안자인지 참여자인지 구분이 된다.

1.3 메뉴 구조도

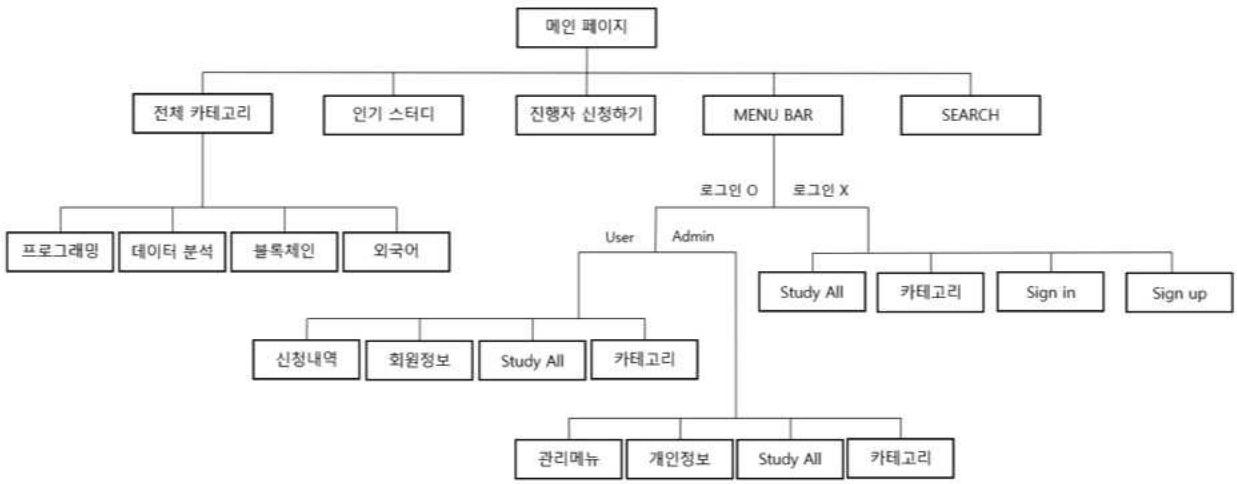


그림 4. 메뉴 구조도

[그림 4]는 서비스 메뉴 구조도이다. Flat한 설계를 지향하기 위하여 최대 Depth를 3을 기준으로 설계하고, 메인 페이지를 중심으로 메뉴 구조도를 설계하였다. 메인 페이지 하위 메뉴로 전체 카테고리, 인기 스터디, 진행자 신청하기, 메뉴바, 검색 메뉴가 존재하며, 전체 카테고리는 하위에 프로그래밍, 데이터 분석, 블록체인, 외국어 등 각종 카테고리가 존재한다. 메뉴바는 Study All, 각종 카테고리는 공통 메뉴로 하위에 존재한다. 비회원일 경우, Sign In, Sign Up 메뉴가 존재한다. 회원일 경우 로그인한 계정이 일반 사용자인지 관리자인지에 따라 메뉴 구조가 변하며, 일반 사용자일 경우 신청/제안내역, 회원정보가, 관리자는 모든 회원과 스터디를 관리하기 위한 관리메뉴와 정보수정메뉴가 존재한다.

1.4 주요 화면 설계

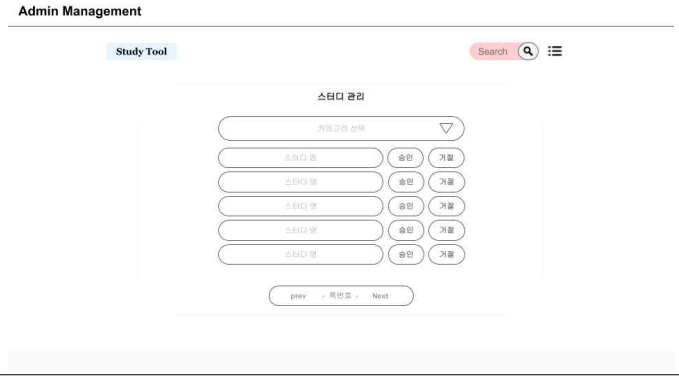
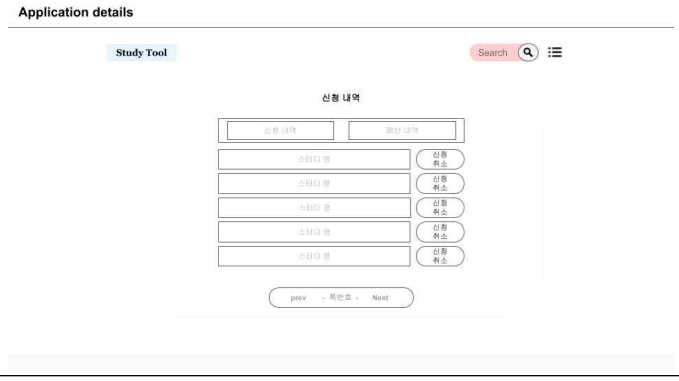
[표 1], [표 2]는 주요 화면을 설계한 것이다. 메인 페이지로 이동할 수 있는 로고와 스터디 검색 버튼, 메뉴바 버튼은 모든 페이지에서 구성하도록 설계하였다. 메인 페이지는 전체 카테고리와 전체 스터디를 실시간으로 확인할 수 있으며, 인기도에 따른 랭킹을 확인 가능하도록 설계하였다. 스터디 상세 페이지는 스터디에 대한 전반적인 정보와 신청하기 버튼으로 페이지를 구성하였다.

[표 1] 메인 페이지 및 스터디 상세 페이지 설계

메인	
스터디 상세 페이지	

스터디 관리 페이지는 일반 사용자가 스토리를 제안하면 각 카테고리에 맞는 스터디를 확인할 수 있도록 화면을 구성하였다. 각 스터디마다 수락 버튼과 거절 버튼을 구성하여 관리의 편의성을 높였다. 스터디 신청 / 제안 내역 페이지는 일반 사용자가 자신이 신청하고 제안한 스터디를 확인할 수 있는 페이지이다. 참여하고 있는 스터디를 삭제할 수 있도록 스터디 삭제 버튼을 추가하여 페이지를 설계하였다.

[표 2] 스터디 관리 페이지 및 스터디 신청 / 제안 내역 페이지 설계

스터디 관리 페이지	
신청 / 제안 내역 페이지	

1.5 사용자 Flow-Chart

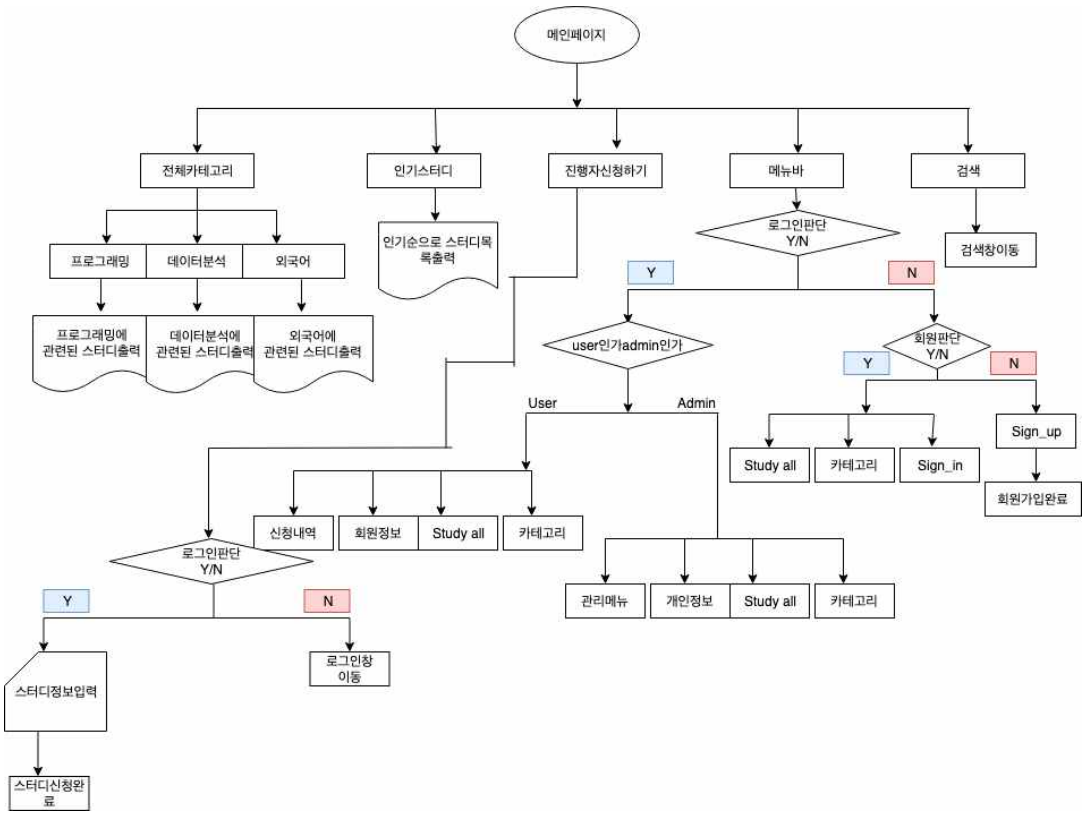


그림 5. 사용자 Flow Chart

[그림 5]는 페이지 관점에서 작성한 사용자 Flow Chart이다. [그림 4]의 메뉴구조도와 흡사하게 사용자는 서비스를 이용할 수 있으며, 각 카테고리 마다 스터디들이 나열되고 상세 페이지로 이동할 수 있다. 로그인 유무, 사용자 권한에 따라 페이지 이동이 다르며, 동일한 페이지라도 권한에 따라 구성요소에 변화를 주었다.

2. 서비스 구현

2.1 시퀀스 다이어그램

서비스 실행 과정에서 다양한 Task 수행을 가시적으로 나타내기 위해 MVC 간의 메시지 전달을 시간적 흐름으로 분석하는 시퀀스 다이어그램을 이용하였다.

시퀀스 흐름은 사용자 관점에서 View를 기준으로 작성하였다. [그림 6], [그림 7], [그림 8]은 핵심 Task를 회원가입, 스터디 제안, 스터디 승인 및 신청으로 잡고 세 가지 Task에 대한 흐름을 나타내었다.

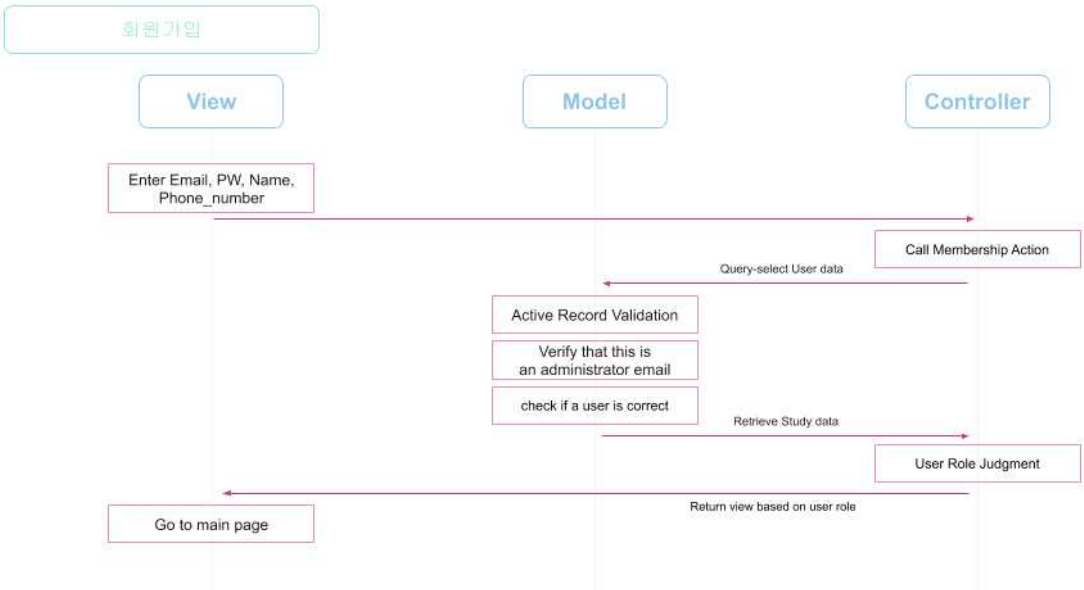


그림 6. 회원가입 시퀀스 다이어그램

[그림 6]은 회원가입에 대한 시퀀스 다이어그램이다. Devise Gem을 이용하여 회원가입을 구현하였으며, Devise의 흐름을 나타낸 것이다. 회원 가입 시 이메일, 비밀번호, 이름, 핸드폰 번호를 필수로 입력해야 하며, 입력된 데이터는 Users 컨트롤러의 회원 가입에 해당하는 액션을 호출한다. 액션이 호출되며 데이터는 모델로 전달이 되며 모델에서 데이터 유효성, 데이터 중복 등을 확인을 하여 컨트롤러로 알맞은 정보라는 것을 알려준다. 컨트롤러는 사용자의 권한이 일반 사용자인지, 관리자인지 판단하여 그에 맞는 정보를 뷰에게 요청한다.

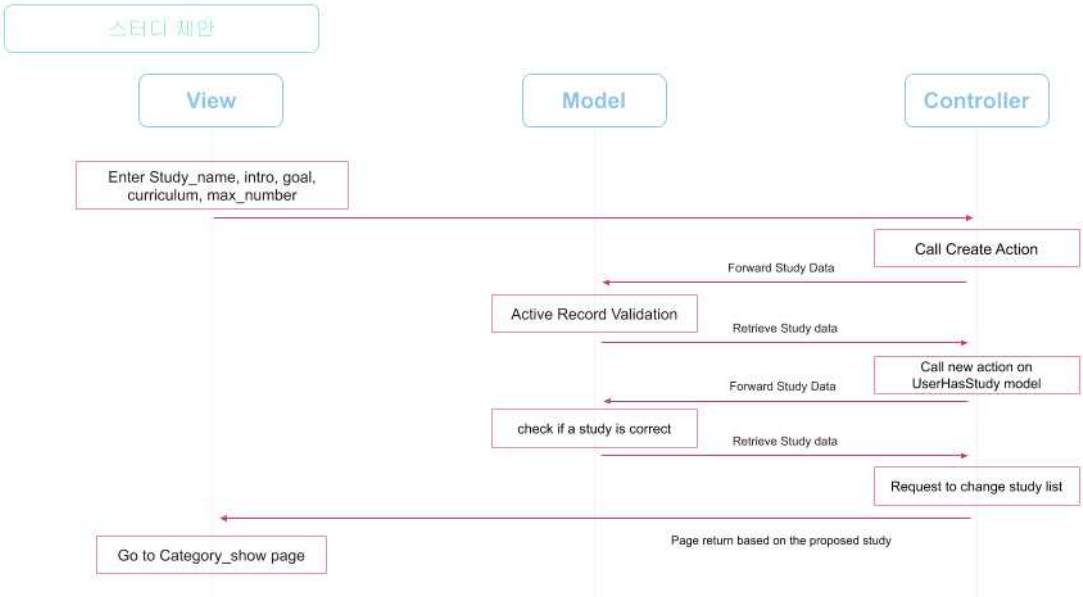


그림 7. 스터디 제안 시퀀스 다이어그램

[그림 7]은 스터디 제안에 대한 시퀀스 다이어그램이다. 사용자는 스터디를 제안하기 위해서 스터디명, 개요, 목표, 커리큘럼, 정원을 모두 입력해주어야 한다. 데이터를 입력하였다면 studies 컨트롤러에 create 액션을 호출하며 입력된 데이터를 Study 모델에 전달한다. 모델에서는 데이터 검증을 하여 모든 데이터가 유효한다면 UserHasStudy 테이블에 user_id와 study_id 값을 외래키로 주어 다대다 관계를 성립하게 한다. 데이터가 완전히 삽입되었을 시 컨트롤러는 스터디 리스트를 갱신하고 카테고리 show 뷰에게 요청한다.

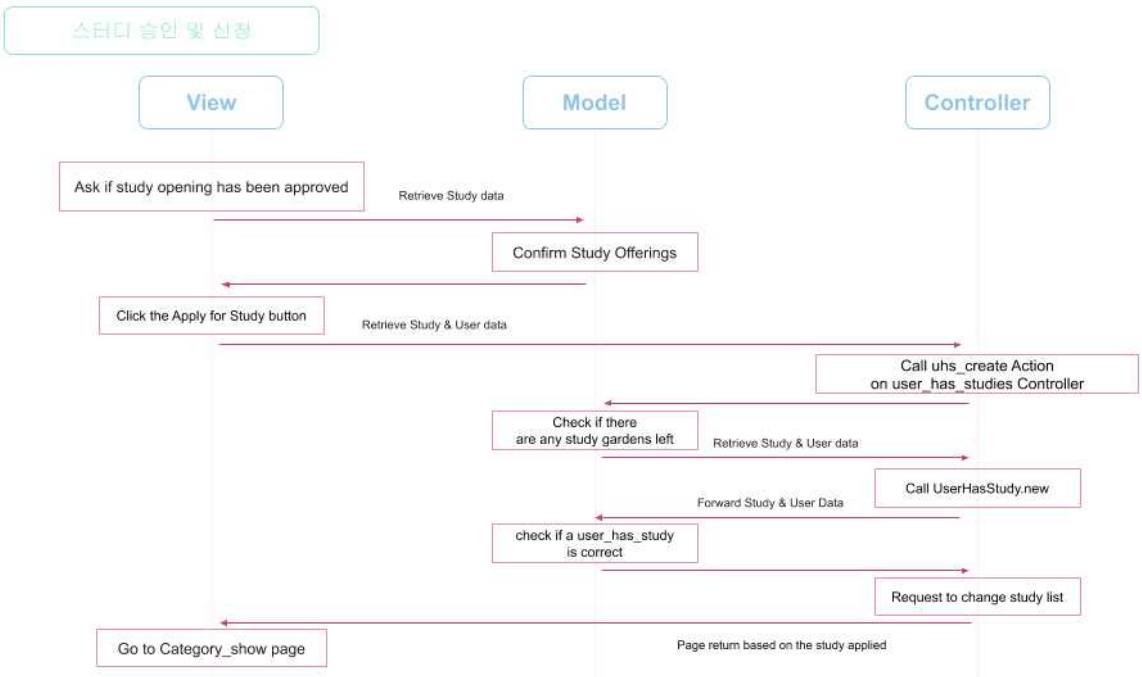


그림 8. 스터디 승인 및 신청 시퀀스 다이어그램

[그림 8]은 스터디 승인 및 신청에 관한 시퀀스 다이어그램이다. 사용자가 스터디를 제안하면, 관리자는 스터디 승인 결정을 내릴 수 있다. 이 때, 관리자가 스터디 개설을 수락한다면 일반 사용자는 신청하기 버튼이 출력된다. 일반 사용자가 스터디를 신청하기 위해서는 신청 버튼을 클릭해야 되고, 버튼 클릭 시 UserHasStudy 테이블에 study_id와 user_id 값을 외래키로 받아 파라미터를 통해 컨트롤러에 전해주고, 뷰에서 받는다. 스터디 신청 할 때 인원 초과인지 확인하기 위해 Model에 Custom Method를 구현하여 확인한다. 인원을 초과하였을 때는 인원 초과 메시지를 출력하고, 아닐 때는 신청 완료 메시지를 출력하고, 카테고리 Show 페이지를 보여준다.

2.2 서비스 구현 및 결과

2.2.1 관리자 권한 부여

```ruby
# user.rb
def assign_default_role
    emails = ['admin01@gmail.com', 'admin02@gmail.com', 'admin03@gmail.com']
    if emails.include? self.email
      add_role(:admin)
    else
      add_role(:user)
    end
  end
```

그림 9. 사용자 권한 부여 코드

[그림 9]는 사용자 권한을 부여하는 코드이다. 특정 이메일만 관리자 권한을 가질 수 있도록 미리 배열에 저장하여 가입 시 함수를 호출해 판단하도록 구현하였다.

2.2.2 사용자 권한 범위

[표 3] 관리자 및 일반 사용자 권한 코드

Admin	User
can :manage, :all	can[:index, :show], :all can[:new, :create], Study

[표 3]은 관리자 및 일반 권한에 관한 코드이다. 관리자는 모든 컨트롤러에서 모든 액션이 가능하고, 일반 사용자는 모든 컨트롤러에서 index, show 액션이 가능하다. 일반 사용자는 Study 컨트롤러에서 new, create 액션은 가능하지만, edit, update, destroy 액션은 불가능하다.

2.2.3 관리 페이지

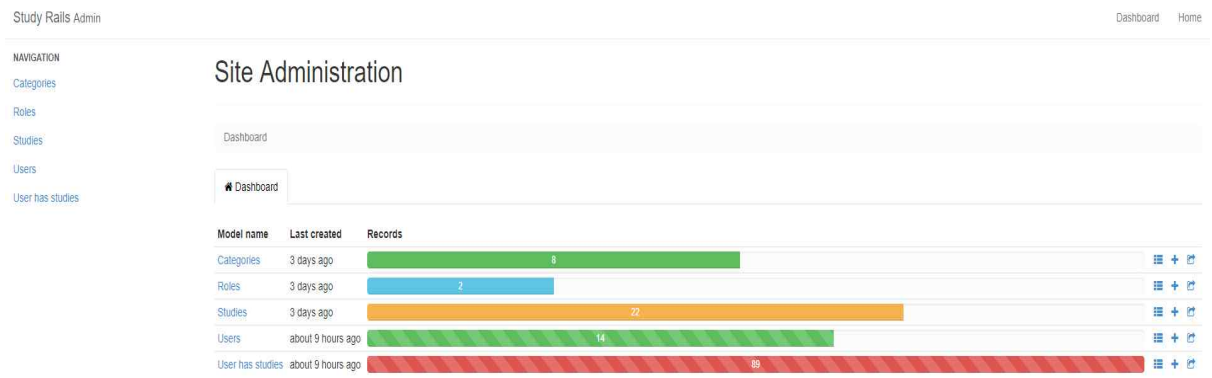


그림 10. 사이트 관리 페이지 대시보드

[그림 10]은 사이트 관리 페이지에 대한 그림이다. 관리자가 사이트를 관리하기 편리하도록 gem admin을 통해 관리 인터페이스 도구를 사용하였다. 관리 페이지에서는 추가, 수정, 삭제 등 모든 기능을 관리할 수 있으며, 데이터 또한 관리할 수 있다.

2.2.3 스터디 신청 페이지

그림 11. 카테고리 show(a), 스터디 show(b)

[그림 11]은 스터디 신청 페이지를 나타낸 그림이다. (a)는 카테고리 show를 출력한 결과이며, (b)는 스터디 show를 출력한 결과이다. 기본적으로 가나다순으로 스터디를 정렬하였고, 개설이 승인된 스터디만 신청하기 버튼이 출력된다. 스터디 마감이 임박하면 '곧 마감됩니다'라는 메시지가, 마감이면 '마감되었습니다'라는 메시지가 출력된다. 마감된 스터디는 정원 초과라는 메시지를 출력하며, 스터디 신청이 되지 않도록 구현하였다.

2.2.4 마이 페이지 – 스터디 신청 / 제안 리스트

[그림 12] 마이페이지 중 스터디 신청 / 제안 리스트 화면

[그림 12]는 마이 페이지 중 스터디 신청 및 제안 리스트 페이지 화면이다. 자신과 연관된 스터디만 출력이 되다. 역할과 스터디명이 출력되며, 삭제 버튼을 누르면 신청하거나 제안한 것을 삭제를 함으로 취소할 수 있도록 구현하였다.

2.3 설계 제한요소 및 구성 요소

[표 4] 설계 제한요소

설계 제한요소	내용
미학	- 메인에는 카테고리 및 카테고리에 해당하는 스터디를 확인할 수 있도록 화면 구성 - 스터디를 신청하고 신청한 스터디 목록을 확인 수 있도록 화면 구성 - user와 admin에 따라 구조와 페이지 이동을 다르게 구성
내구성	- 코드 변경을 하지 않을 시 계속 사용 가능 - 코드 변경 시 변경한 기능으로 구성이 바뀜

[표 4]는 설계 제한 요소를 나타낸 표이다. 설계 제한 요소는 크게 미학과 내구성을 기준으로 제한을 주었다. 미학은 각 페이지 당 UI 요소에 해당하는 것으로 페이지가 변하면 구성 요소도 변하며, 로그인을 한 계정의 권한에 따라 구성 요소가 변한다는 것에서 착안한 것이다. 내구성은 Rails 원칙에 의거하여 최소한의 코드로 기능을 구현하고자 제한요소로 잡았다. 중복 코딩을 줄이며 유지보수를 보다 쉽게 하자는 취지에서 착안하였다.

[표 5] 설계 구성요소

설계 구성요소	내용
합성	- Ruby on Rails 기반 스터디 관리 웹 서비스 설계 및 구현 - 스터디 파이가 어떤 서비스인지 어떻게 만들어졌는지 확인 - Rails 프레임워크의 장점과 왜 사용하는지 확인 - 웹사이트를 구현하기 위한 메뉴구조도, 주요 화면 설계, DB설계
분석	- 온라인으로 혼자 공부해보고 싶고 온라인에 수많은 자료 중 어떠한 자료로 공부해야 하는지 고민하는 사람을 위한 온라인기반 스터디 교육 플랫폼 - 메인 : 카테고리, 인기스터디, 진행자 신청, 메뉴바, 검색 메뉴가 있음 - 주요기능 : 사용자 관리, 스터디 신청, 스터디 관리, 스터디 참여가 있음 - 웹사이트를 Ruby on Rails를 기반으로 쉽고 빠르게 만들 수 있음
제작	- Scaffold를 통해 index, show, new, create, update, destroy 메소드를 생성 - gemfile 수정 및 devise 설치를 통해 로그인/회원가입 구현 - Controller 생성, Routes 업데이트, link_to를 활용하여 메인페이지를 만들어 원하는 페이지로 이동할 수 있도록 구현 - 스터디 개설을 위하여 경로를 추가함 - 스터디 신청을 위하여 파라미터 값으로 필요한 id를 가져옴 - 사용자가 신청한 스터디 목록을 보여주는 페이지를 생성
시험	- 메인페이지 화면이 제대로 뜨는지 확인 - 로그인 계정이 일반 사용자인지 관리자인지에 따라 메뉴 구조가 다른지 확인 - 사용자가 스터디를 신청 했을 때 제대로 신청되었는지 확인 - 스터디 신청과 인원초과 확인 - 사용자가 신청한 스터디 목록을 마이페이지에서 확인
평가	- 각 카테고리마다 스터디목록이 나열되어 있고 상세페이지로 이동함 - 로그인 유무, 사용자 권한에 따라 페이지 이동이 다르며, 동일한 페이지라도 권한에 따라 다르게 나타남 - 스터디를 신청할 때 중복 신청이 되어 중복신청이 안 되도록 수정해야 함. - user/admin 권한이 주어지지 않은 경우 권한을 주도록 함 - 각각 구현한 코드에서 부족한 부분과 추가해야 할 부분을 보완해야 함

[표 5]는 설계 구성 요소를 나타낸 표이다. 크게 합성, 분석, 제작, 시험, 평가로 구분하여 설계를 하였다.

III. 결론

[표 6] 설계 목표의 중요도 및 달성도

목표	중요도(%)	달성도(%)	수행내용
DB생성 및 DB간의 관계 설정	40%	40%	user테이블,study테이블,study_list테이블생성 user:study n:m study_list:user 1:n study_list:study 1:n
DB 간 데이터통신	40%	30%	study생성 시 user의 name,id를 user테이블에서 받아올수있음 study_list테이블에 study테이블의 name,max,content를 받아올수있음 study테이블의 정원값을보고 스터디신청이가능한지 아닌지알수있음 admin페이지에서user테이블의값과 study테이블의값을 받아와 데이터삭제수정가능
관리자, 스터디 참가자, 스터디 진행자 권한설정	20%	15%	rails_admin gem을 사용하여 관리자페이지생성 admin페이지에서 user,study관리 삭제 수정가능 user테이블의 role 칼럼에 주최자와 참가자를 boolean값인 0과1로 한 뒤 cancancan gem을 이용 게시물의 읽기쓰기삭제권한설정
합계	100%	85%	

본 과제에서는 Ruby 기반 웹 프레임워크인 Ruby on Rails를 기반하여 스터디 관리 서비스를 설계 및 구현하였다. Rails의 scaffolding 기능을 이용하여 MVC 모델을 쉽게 구현하였으며, DB와의 데이터 통신을 편리하게 구현할 수 있었다.

기존 스터디파이 홈페이지는 모바일 친화적인 디자인을 가지고 스크롤 형식의 웹사이트였다. 기존 사이트와 차이점이라 하면 모바일보다는 데스크탑에서도 사용성을 높이는 방향으로 UI를 설계하였고, 부가 서비스없이 핵심 기능을 위주로 구현했다는 점을 들 수 있다.

하지만 본 과제에서 기능 설계에 초점을 맞추다보니 Rails 원칙을 지키지 못했다는 점이 개선해야 될 부분이라 할 수 있다. Rails 철칙 중 하나인 코드 중복을 최소화하기 위하여 모델에 메소드화하는 작업이 필요하며, 코드의 명명 규칙이 일정하지 못하다는 점에서 유지보수에 취약하다는 점을 개선해야 될 부분이다.

추후 앞서 제시한 부분뿐만 아니라 홈페이지에 접속하면 자동으로 비회원 계정이 생성되어 세션으로 저장하여 비회원도 index, show 액션이 가능하게 할 필요가 있을 것으로 보인다.

[참고문헌]

[1] 펜셀베이아 대학, "The Life Cycle of a Million MOOC Users", 2019.06.10, https://www.gse.upenn.edu/pdf/ahead/perna_ruby_boruch_moocs_dec2013.pdf

[2] TaeWoo Kim, "스터디파이는 왜 이 일을 하는가?", 2019-05-20, "https://brunch.co.kr/@taewookim/12"

[3] Ruby on Rails Guides, "https://guides.rubyonrails.org/"

[4] "Ruby on Rails의 기본 원리 이해하기 : HTTP, MVC, Routes", 2019-06-10", "https://www.freecodecamp.org/news/understanding-the-basics-of-ruby-on-rails-http-mvc-and-routes-359b8d809c7a/"