Project Meeting Log

CRAISELION : Handong Team Meeting Archiving and Exchange Web Platform

Version:	1.0
Date:	2024-04-08
Client:	Student
Confidential:	No
developer:	Seokjae Ma, Donggyu Kim, Sechang Jang, Junhyeok Choi, Minseo Lee

Team

Name	Email	ID	Role
Seokjae Ma	21800239@handong.ac.kr	21800239	Project Manager
Donggyu Kim	22000063@handong.ac.kr	22000063	Scrum Master
Sechang Jang	21900628@handong.ac.kr	21900628	Documenter
Junhyeok Choi	21900764@handong.ac.kr	21900764	Developer
MinSeo Lee	22100503@handong.ac.kr	22100503	Developer

Version

Version	Date	Description
1.0		

Table of Contents

1 Introduction

1.1 Project background

Handong University hosts a variety of Residential Colleges (RCs), within which numerous team meetings are regularly held. These meetings play a crucial role in fostering community formation and interaction among students. However, the current system faces several significant challenges.

First, there are difficulties in exchanging information between teams within the same RC, and even more so between different RCs. This lack of interaction within and between RCs hinders the development of a community-centered culture at Handong, which the university aims to promote.

Second, there is a scarcity of guidance provided to team leaders, leading to challenges in leadership and team management. This situation diminishes the efficiency and effectiveness of team activities, negatively affecting student engagement and enthusiasm.

Third, the preservation of materials and information generated during team meetings throughout the year is not adequately addressed. Due to the absence of a proper archiving system, important documents are often lost, posing significant challenges to long-term project management and the reuse of materials.

To resolve these issues, we propose a web platform that facilitates communication and exchange among teams or RCs and preserves information and materials related to team meetings, thereby aiding in the development of the entire community at Handong University. This platform will serve as an essential resource for the student support team and the students (team executives), who are the main stakeholders, in enhancing the team culture and leadership development within Handong University.

2 Whole Team Meeting

2.1 3/28 Meeting Minutes

Write a **Brief Proposal** to send a JC in an email

Handong Team Meeting Archiving and Communication Web Platform

Problem Definition:

Currently, team meetings held in each RC (Residential College) at Handong University face difficulties in interacting with other RCs and even with other teams within the same RC. This situation limits the growth of a community-centered culture at Handong University. Additionally, there is a lack of guidance for team leaders, and materials generated from team meetings are often not preserved and lost.

Stakeholders:

- Student Support Team
- Students (Team Executives)

Solution:

To address these issues, we propose a web platform that facilitates communication and exchange between teams or RCs, preserves information and materials related to team meetings, and aids in the development of all communities at Handong University.

Detail Meeting Log

Problem Definition

- **Communication Issues**
    - Team meetings within each RC (Residential College) find it difficult to interact with other RCs.
    - Even within the same RC, teams struggle to communicate with one another.
        - These two issues hinder the growth of a community-centered culture at Handong University.
    - Lack of guidance for new team leaders/family heads on how to lead effectively.
    - Materials and information generated during team meetings throughout the year tend to be lost and not preserved.
        - Due to the loss and lack of preservation of these materials, students are reluctant to take on leadership roles in team meetings.
    - **Consider Issues from the Professors' Perspective as Well!**

Solution

- A web platform that archives and preserves all information and materials related to team meetings, aiding the development of all communities at Handong University.

TODO

2.2 4/3 Meeting Minutes

Meeting With JC

Request → Student Affairs Office (receive requirements from there) → Direct Contact

The topic is acceptable.

Is it a topic that can really be maintained? - Design and consideration.

HSF?

Professor's Opinion:

- Understanding the requirements of the Student Affairs Office and the executives is more important than the professor's own needs.
- Consideration of privacy issues is necessary.
- The reason why the student guidebook is not often sought after: the system is centered around students.
- Feed feature → likes.
- Appointment of team leaders → Student Affairs Office.
- If it overlaps ambiguously with school administration, it could become complicated. Develop it in an unofficial but useful and flexible manner. Avoid being too official (useful and free).

Stakeholders - Students, Student Affairs Office (professors are not the main clients).

Can the Student Affairs Office use it continuously?

Inconveniences from the Professor's Perspective:

- From the professor's perspective, there were no significant inconveniences.

TODO

Having other meetings considering this topic with JC.

2.3 4/4 Meeting Minutes

Requirement Analysis

Handong University hosts a variety of Residential Colleges (RCs), within which numerous team meetings are regularly held. These meetings play a crucial role in fostering community formation and interaction among students. However, the current system faces several significant challenges.

To resolve these issues, we propose a web platform that facilitates communication and exchange among teams or RCs and preserves information and materials related to team meetings, thereby aiding in the development of the entire community at Handong University. This platform will serve as an essential resource for the student support team and the students (team executives), who are the main stakeholders, in enhancing the team culture and leadership development within Handong University.

Interview & Answers From Students

All Interviewees: 15 Handong students

If you have been a team leader, what aspects did you find challenging? / If you have not been a team leader, which parts do you think might be challenging?

💡 *Social interaction and cultural adaptation, Repetitive Team Contents, Difficulty of finding team meeting requirements,* Managing a large team without good information and creating a good atmosphere, Motivating disengaged members and lacking proper recognition and support for team leaders, Dealing with disengagement and lack of motivation in team members

What are your thoughts on the current team meetings? (e.g., well-conducted, difficult, etc.)

💡 well conducted overall

What content in the team meetings did you find beneficial?

💡 Outside activities, Team barbeque, Stock (Virtual Investing) Games, movie posters imitating, In-campus BattleGround games, etc.

What value does the team meeting time hold for you?

💡 Building community and valuing sacrifices; feels it's becoming more of a class requirement. It is a community united in God, where each other's sacrifices are valued. However, its significance is diminishing and it's increasingly viewed merely as part of the coursework.

What differences do you think exist between the roles of senior students and freshmen in team meetings?

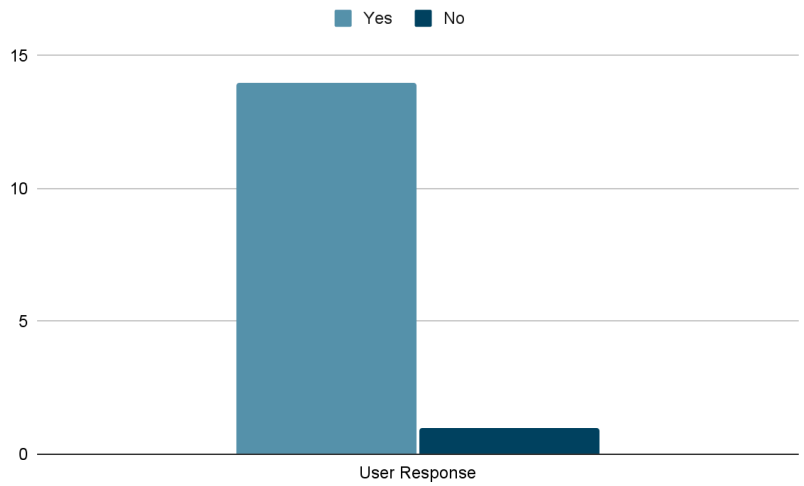
💡Freshmen are expected to bring vitality to the team, while seniors should actively work on improving and maintaining a good team atmosphere, taking some responsibility outside of meetings to foster better relationships with team members and freshmen.

Would you be interested in using a school service that records and shares team meetings?

💡 I might use it if available, it could be a great substitute for team SNS. However, features such as a points (ranking) system, archiving team meeting content, and storing memories should be well managed.

Interviews showed positive results about the Team Meeting Archiving and Exchange Web Platform.

This graph shows the number of students who reacted positively to using the new web service platform.



Their requirements features are the following:

- Post uploads and shares through the team
- Share team meeting contents
- Team meeting matching feature
- Team photo gallery
- Scrap and archiving feature
- RC / Team-based ranking system

MVP (Minimum Viable Product)

- **File Upload**
  - Function to upload and store all materials generated during team meetings.
- **Login System**
  - Google login for easy access.
  - Three levels of permissions: system administrators, professors, and students.
- **Bulletin Board Functionality**
  - Instagram feed-style bulletin board for information sharing.
  - Team meeting matching service to be further developed.
- **Information Delivery Page**
  - A page to efficiently convey important information related to team meetings.
- **Tag-Based Document Retrieval**
  - Ability to search and display documents by tags.

Additional Considerations:

- **Security**
  - Ensure a robust system for user information and document security.
- **UI/UX Improvement**
  - Develop an intuitive and user-friendly interface for ease of use.
- **Scalability**
  - Design the system with scalability in mind for easy addition of future features.

TODO

Find References
Research and analyze the technologies and methodologies needed to implement the MVP

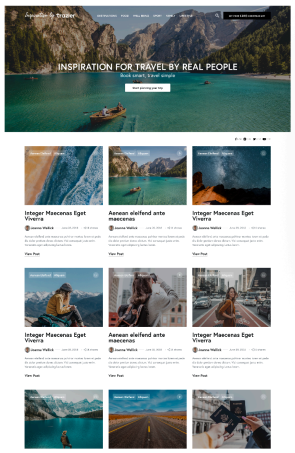
2.4 4/11 Meeting Minutes

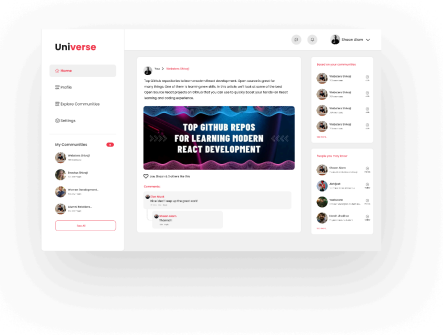
Make Prototype

Design Reference Sharing and Agreement

- **Data Archiving** - Search by categories (by institution/name)
- **FAQ Page**
- **Handong Key Colors**
    - RC page header section
- **Main Page**
    - Symbolism of team meetings
    - Use of animation
        - Similar to the main page of LikeLion
- **Matching Page**
    - Key keywords
- **Feed Example**
    - Instagram
    - Disquiet
    - Feed preview (drag format, photos only)
        - Team recall function - separately
- **Reservation Confirmation Page**
    - Reservation system for team meeting activities
    - Use posts to reserve team meeting locations
- **Scrap Feature**
- **Dormitory-Specific Login System**
- **Ranking System**
    - Scores between teams
    - Events
        - Top team on the main page
- **Team Follow Function**
    - Similar to YouTube subscriptions
- **Clean UI**
    - Grid
    - Communication

References





깔끔한 UI 가 좋습니다!

Home Page

Figure 1

Figure 2

My info Page

Figure 3

Team meeting matching board Page

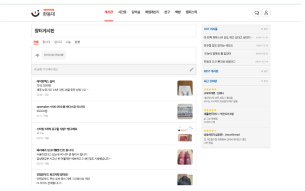


Figure 4

Team meeting feed Page

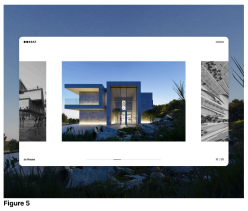
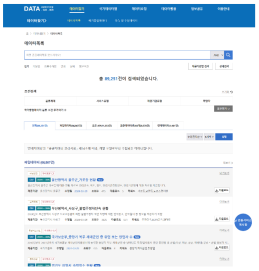
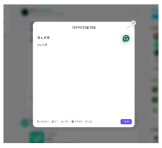


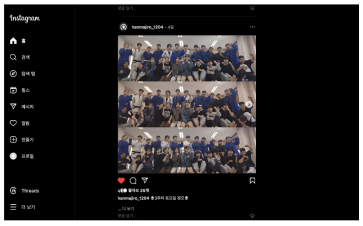
Figure 5

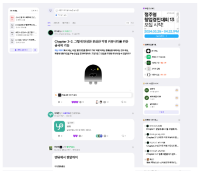


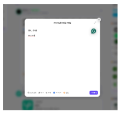


공간중심의 예약 서비스

필터 기능도 추가해서 하트 많은순으로도 볼수 있으면 좋겠습니다.







TODO

Requirements Specification - Can be postponed due to time constraints

IA Documentation

UT Plan Establishment

2.5 4/18 Meeting Minutes

Technical Stack and Framework

1. Frontend Development: The user interface of the platform will be built using React, renowned for its scalability and flexibility. React's component-based architecture will allow for a responsive, user-friendly experience across various devices, catering to the needs of students and team leaders alike.

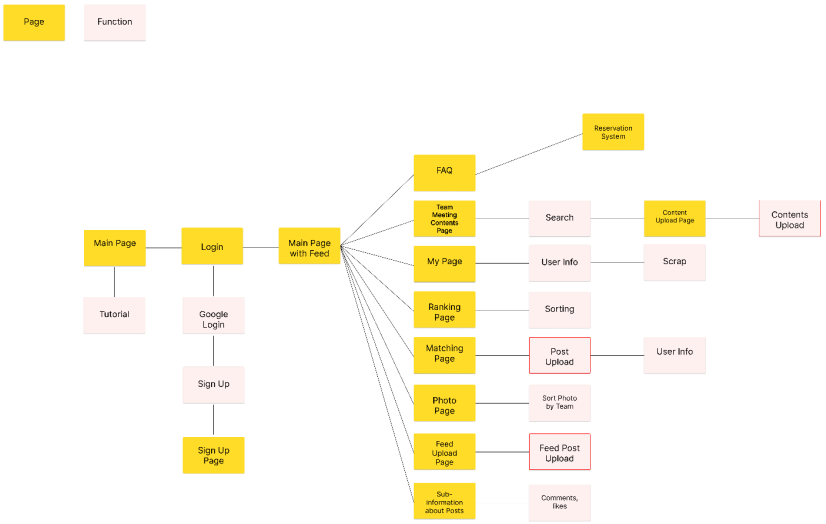
2. Backend Services: The backend of the platform will be developed using Spring Boot, chosen for its rapid development capabilities and its extensive suite of features for building robust web services. Spring Boot will facilitate easy integration with other systems and databases at Handong University.

3. Database Management: MySQL will serve as the primary database management system due to its reliability and efficiency in handling complex data structures necessary for storing user data, meeting records, and archival materials.

4. Security and Compliance: To ensure the security of data transmission and user authentication, Spring Security will be implemented. It will provide comprehensive security configurations to safeguard personal and operational data against potential threats.

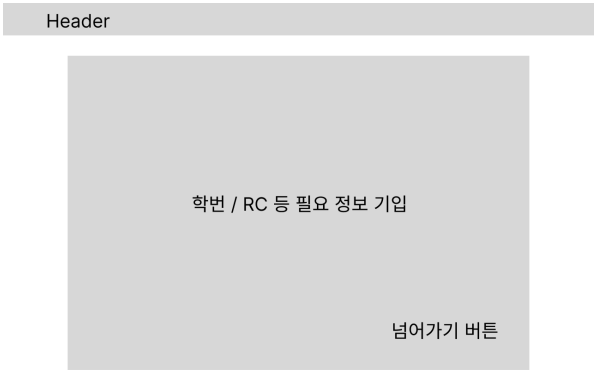
5. Document Storage and Archiving: For managing and archiving digital assets and meeting documents effectively, the platform will integrate a document management system that supports tagging, categorization, and easy retrieval of archived materials.

Prototype

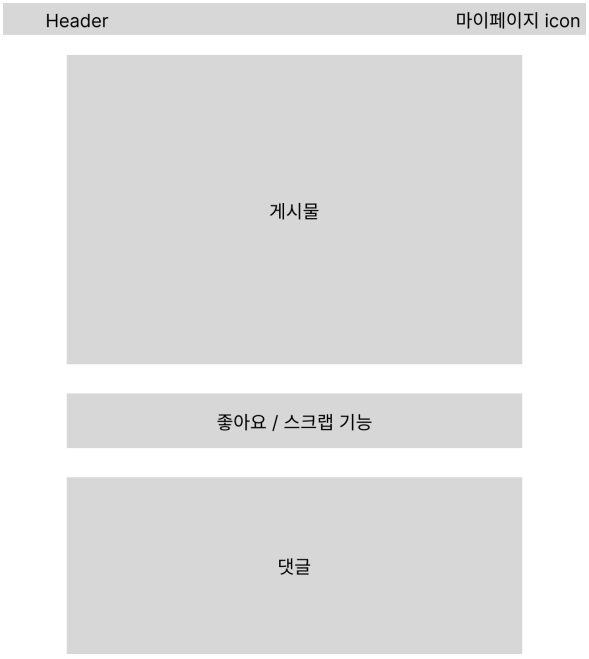


System Information ArchitectureIA (Information Architecture) involves structuring and organizing information to enable users to easily understand and access it. The IA component of our system aims to enhance user experience, facilitate efficient information retrieval, and assist users in quickly finding the information they need.

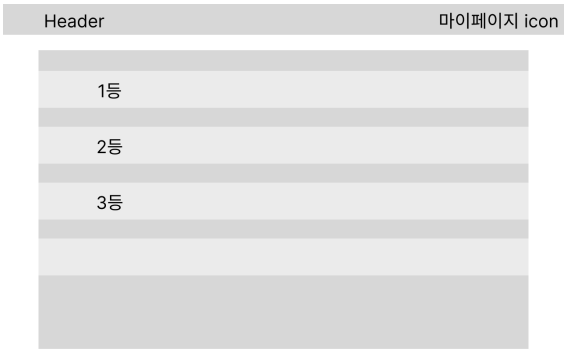
Through our system's IA, we aim to ensure that users can easily find and comprehend information. To achieve this, we consider user groups and profiles when organizing information, provide clear and consistent content classification, and optimize user access paths. Additionally, we support efficient system development by considering system architecture and user experience design and provide a secure information environment, including considerations for security and permissions.
Through this IA approach, our system can offer users an intuitive and convenient experience while effectively managing and utilizing information.


Header
학번 / RC 등 필요 정보 기입
넘어가기 버튼

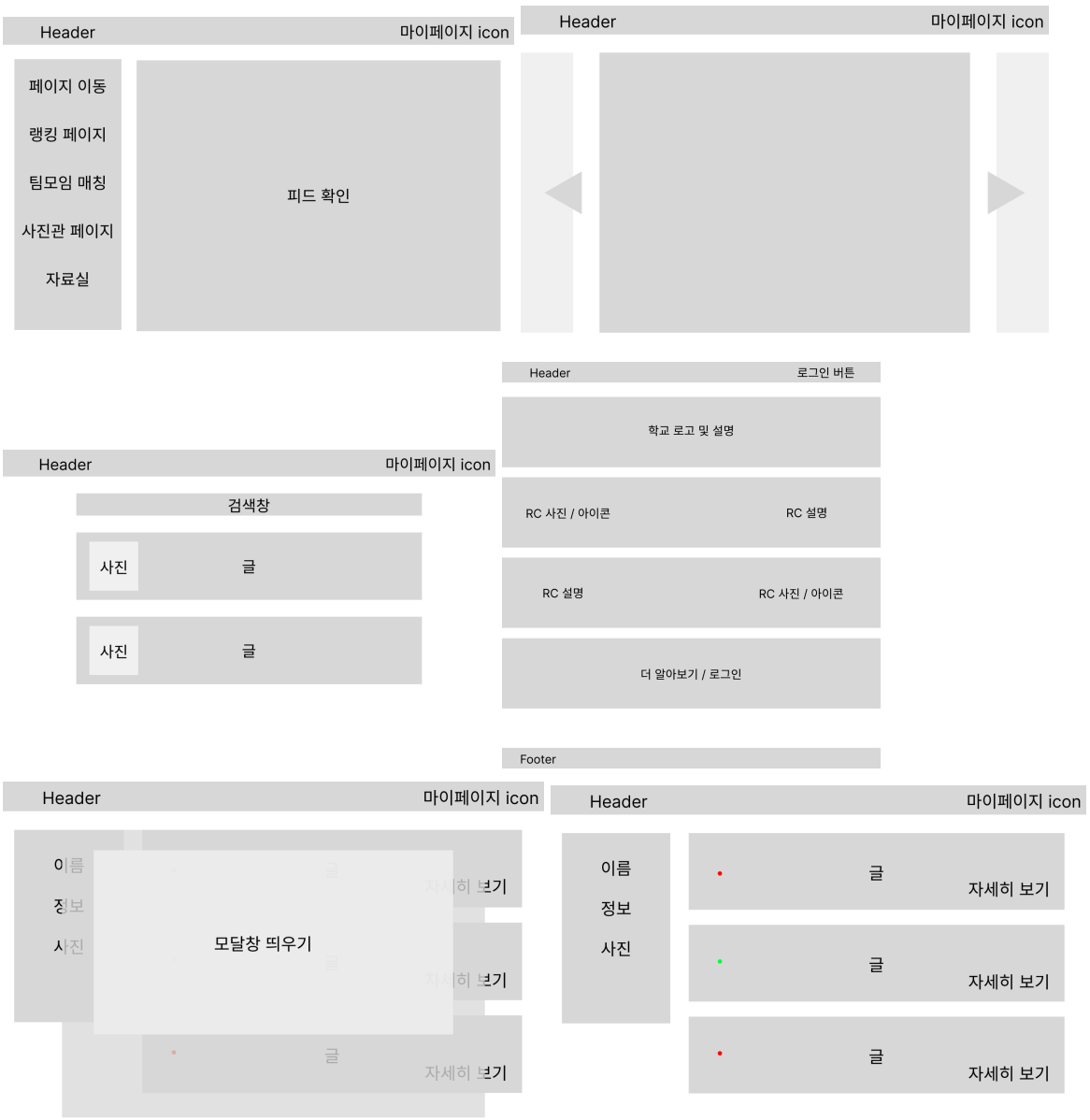

Header
마이페이지 icon
게시물


Header
마이페이지 icon
게시물
좋아요 / 스크랩 기능
댓글


Header
마이페이지 icon
사진
제목
본문
완료


Header
마이페이지 icon
1등
2등
3등


Header
마이페이지 icon
이름
정보
사진
내가 쓴 글
우리 팀이 쓴 글
내가 스크랩 한 글



TODO

Prepare Requirement Analysis Documents

2.6 5/2 Meeting Minutes

Meeting Method now on

Tentative Meeting - Weekly

- **Not necessarily offline, but avoid Zoom.**
- **Use Zoom only in emergencies.**

Progress Report

- **Follow the Scrum methodology.**

Server

1. **Priority: CRA server**
2. **Secondary: Desktop**
3. **Tertiary: Naver Cloud**

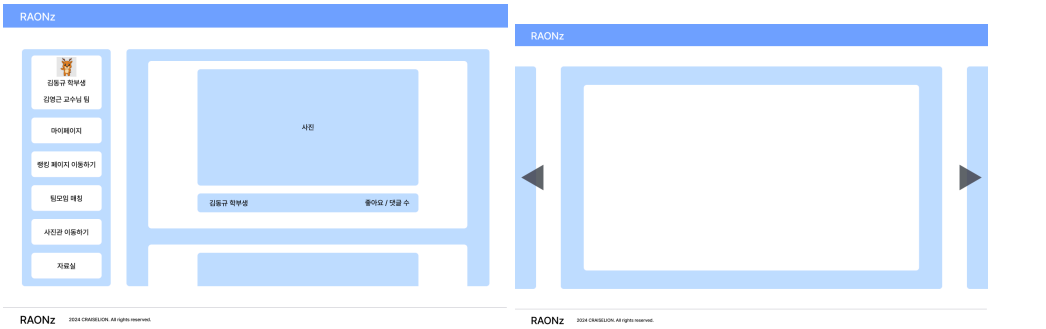
Communication

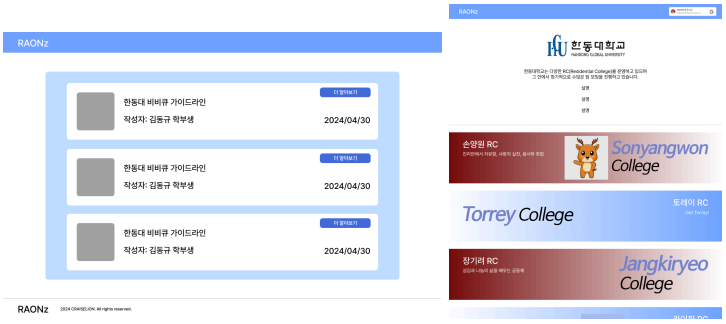
- **Communicate via the group chat on KakaoTalk.**

2.7 5/9 Meeting Minutes

Design Feedback

P



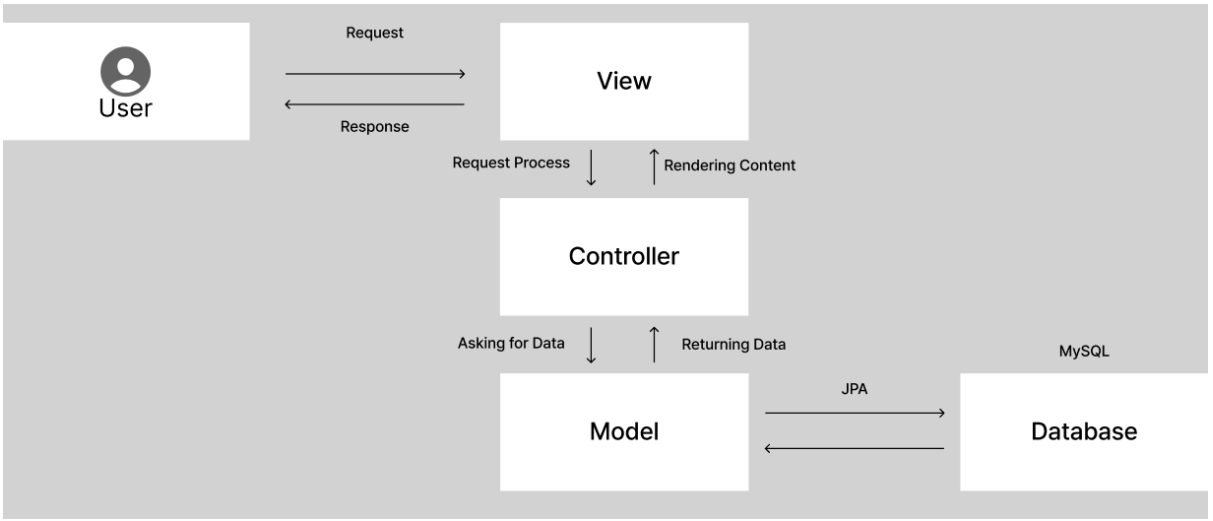


System Modeling

3.2 MVC Pattern

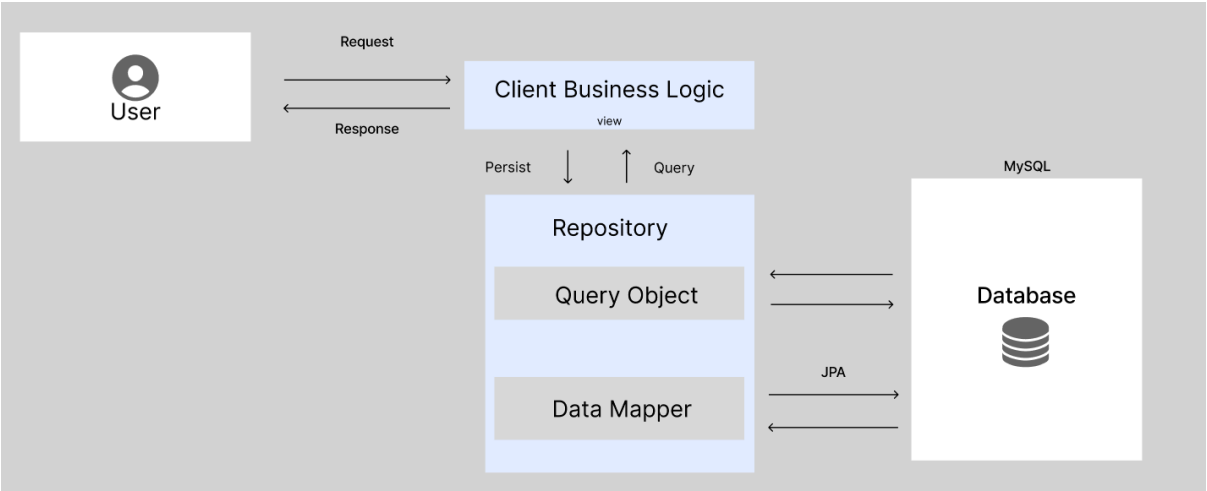
The Model-View-Controller (MVC) is a design pattern used in software engineering. It separates an application into three interconnected components:

- **Model:** This handles data and business logic. It interacts with the database, as shown in your image. The model retrieves data from the database using MySQL and JPA and returns the data to the controller.
- View: This presents data to the user. It's responsible for the user interface (UI) components. In the context of the previous flowchart, the view would be responsible for displaying the various pages like "Photo Page", "Feed Page", etc.
- Controller: This acts as an intermediary that handles input from the user and updates the Model and View accordingly. In your image, the controller is shown processing requests and rendering content.



In the context of the previous question, the MVC pattern could be used to structure the website. The flowchart could represent the user's journey through the View, while the Controller handles the logic of what page to display next, and the Model manages the data related to the user and the content of the pages.

3.3 Repository Pattern



The Repository Pattern is a design pattern that mediates between the domain and data mapping layers using a collection-like interface for accessing domain objects.

In the diagram you provided, the Repository Pattern is illustrated as follows:

- User: The user interacts with the application, sending requests and receiving responses.
- Client Business Logic: This layer handles the business logic of the application. It receives user requests and interacts with the repository to retrieve or persist data.
- Repository: This is the core of the Repository Pattern. It communicates with the data layer to perform CRUD operations (Create, Read, Update, Delete). Inside the repository, there are several components:
- Query Object: This object handles specific queries to the database.
- Data Mapper: This component maps the data from the database to objects that the application can use, and vice versa.
- JPA (Java Persistence API): This is a standard interface for accessing databases in Java. It's used to interact with the database.
- Database: This is where the data is stored. The database in the diagram is a MySQL database

Using GitHub

https://github.com/2024-SE-Project/.github/blob/main/profile/README.md

1. Creating an Issue

✷ **How to Use Git Issues**

- Create an issue for the feature you will work on.
- Standardize the issue title as [Type] - Description (e.g., [Style] - Add text style).
- Tag the person responsible for the task in Assignees.
- Tag the relevant type in Labels.

- In the description field, include details about the task, any related issue numbers, and any references.
- Once the issue is created, it will be assigned an issue number #N.

2. Creating a Branch

✹ **How to Use Git Branches**

- Work only on the branch you have created. (Refer to How to Use Issues for branch creation.)
- The branch naming convention is <yourname_type/#issuenumber> (e.g., ehdrb01_feat/#1).

3. Changing Branch with Checkout

git checkout ehdrb01/#12

4. Working

- Commit after completing each task.
- Merge the branch you worked on into the develop branch via a pull request.
- Merge after code review.

- **Type**: Short description (in Korean)
  - feat: 로그인
  - e.g., style: 텍스트 디자인 시스템 구축

Type	Description	Example
feat	Add or implement a new feature	feat: 로그인 기능 구현
edit	Fix a simple typo	edit: 로그인 캐시 처리 방식 수정
style	Add or modify UI/Style-related files	style: 폰트 등록
add	Add asset files (images, icons, etc.)	add: 위젯 이미지 추가
chore	Move or rename files/paths	chore: feet -> feat 이름 변경
delete	Delete dump files	delete: Empty.md 파일 삭제
merge	Merge branches	merge: pull request #3 from ehdrb01_style/#1
fix	Fix bugs	fix: Color 버그 수정
docs	Work on documentation	docs: Readme 작성
refactor	Refactor code	refactor: 변수명 수정
model	Work on the database (model)	model: 데이터 모델 생성
init	Initialize the project	init: 프로젝트 생성
test	Create test cases	test: 테스트 케이스 생성
build	Rebuild	build: 동일 버전 재빌드(x.xx)
version	Update version	version: 버전(2.0.0) 업데이트

✹ **How to Write Pull Request Titles**

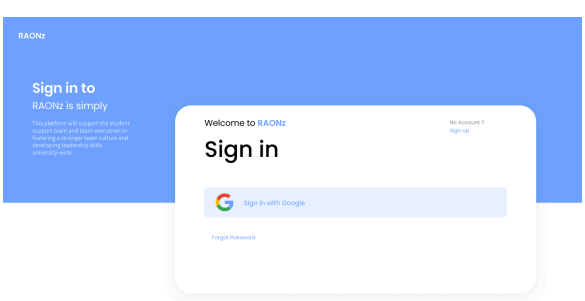
- Format: name_type/#issuenumber -> target branch (e.g., ehdrb01_style/#1 -> dev)

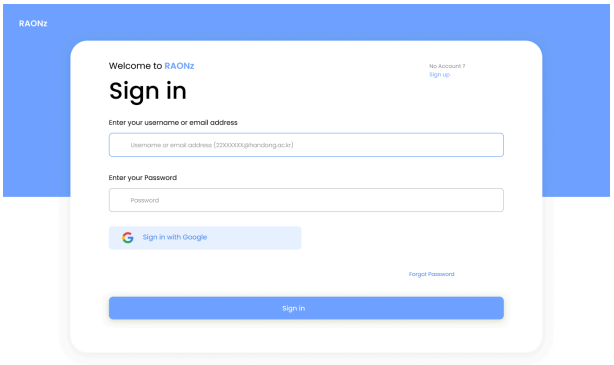
TODO

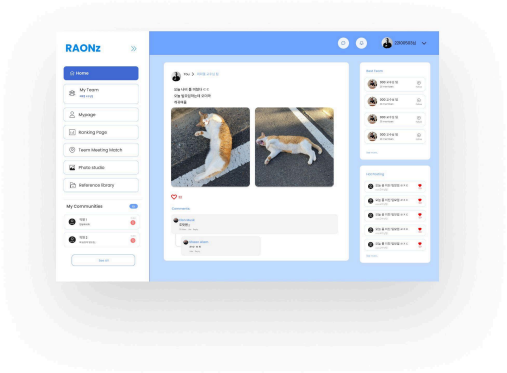
There is feedback that the design is less intuitive.
Preparing System Modeling Documentations

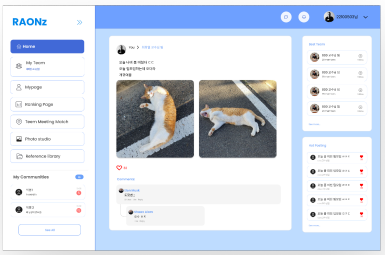
2.8 5/16 Meeting Minutes

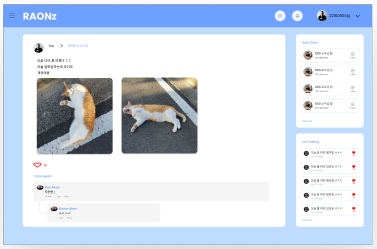
New Design System

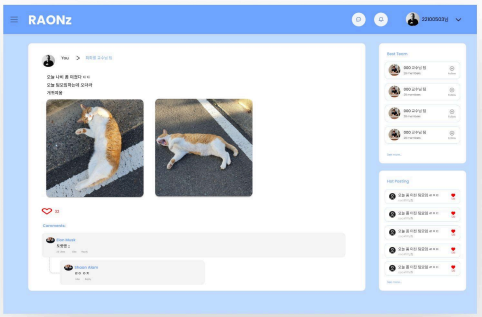


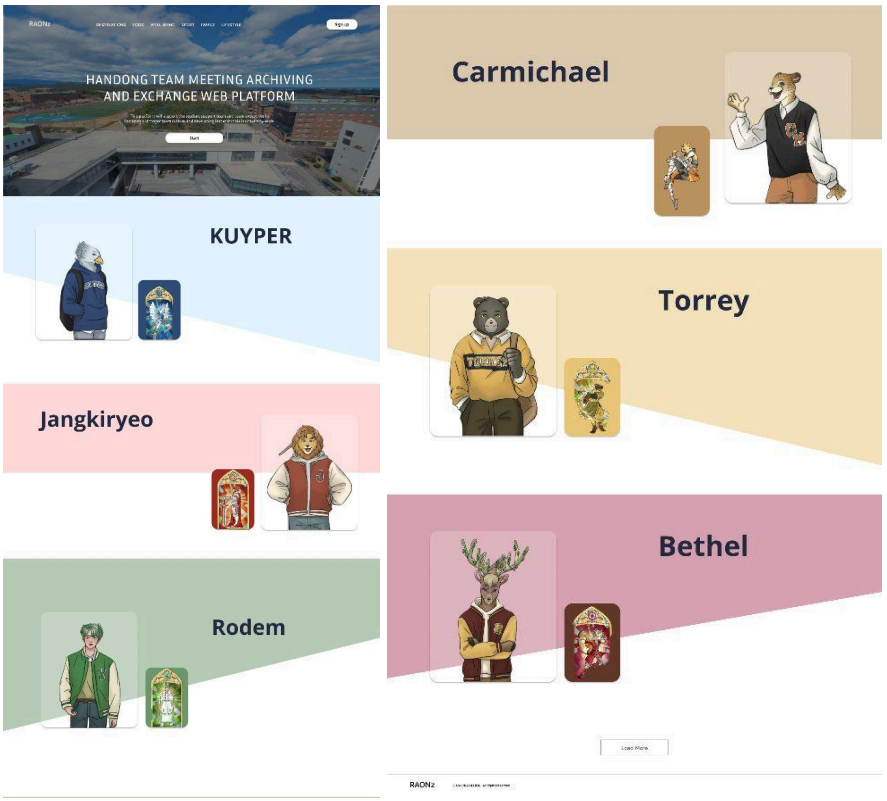












System Modeling

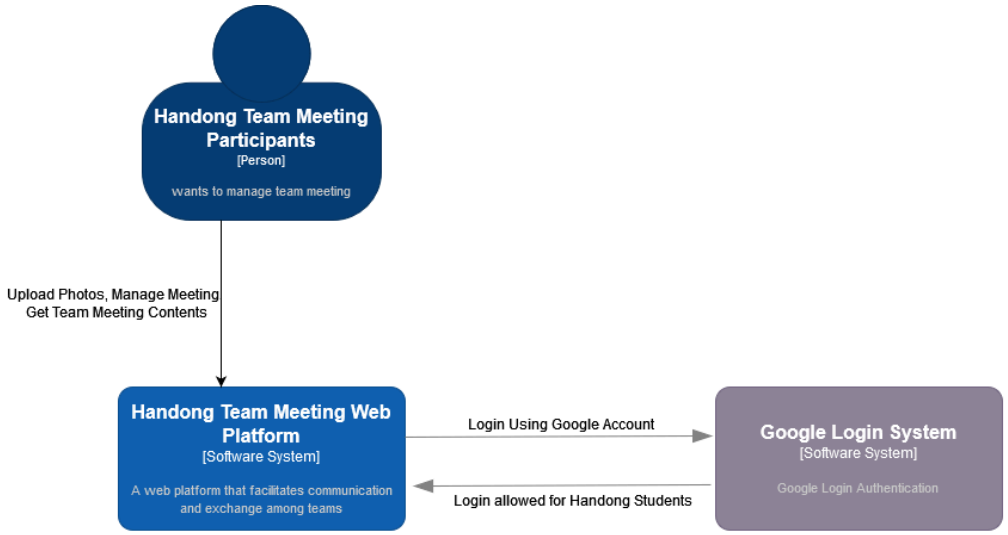
Paperwork is ongoing.

C4 model

The detailed technical design is constructed using the C4 model. This slowly zooms in on the different parts of the application. This model is divided into four layers, which are context, containers, components, and code. We are going to draw for Context, Container, and Component for system modeling and design.

Context Diagram

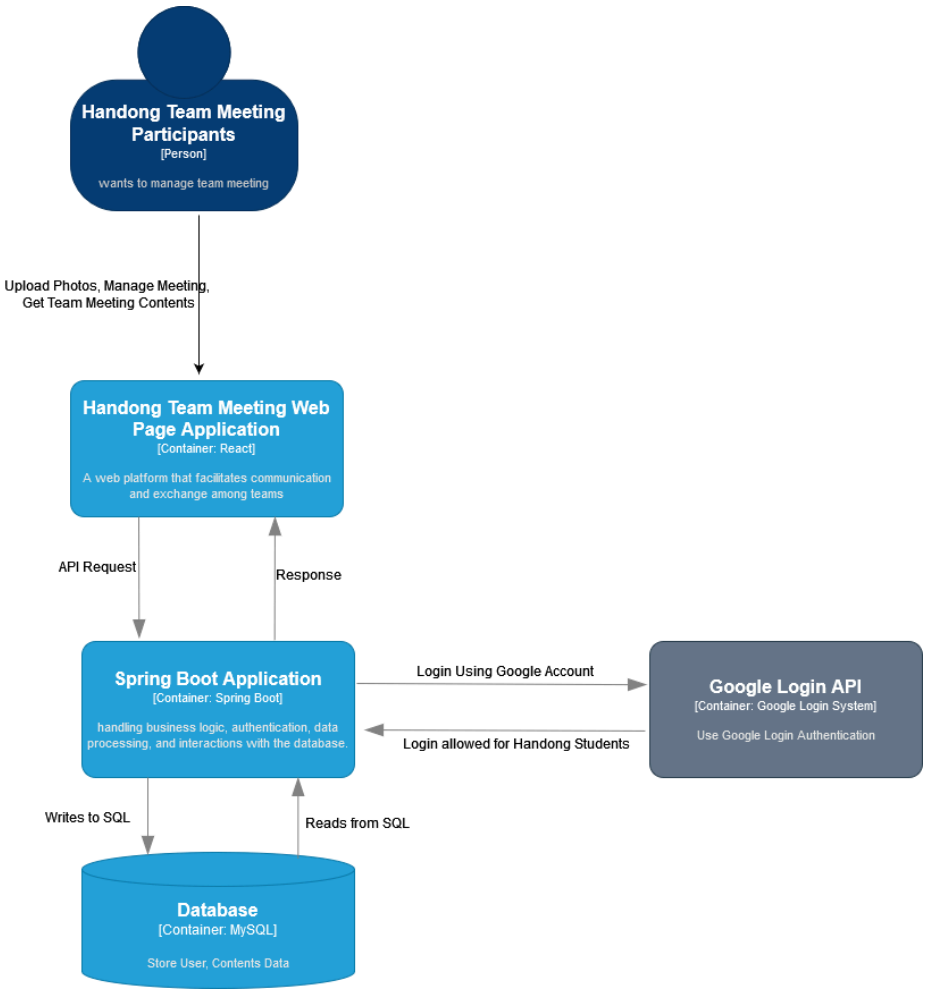
- **System:** Handong University Team Collaboration Platform
- **Purpose:** Facilitate communication and collaboration within and across Residential Colleges (RCs), improve leadership guidance, and archive essential documents.
- **Primary Users:** Students (team executives) and Handong Students



Level 1. Context Diagram

Web Application:

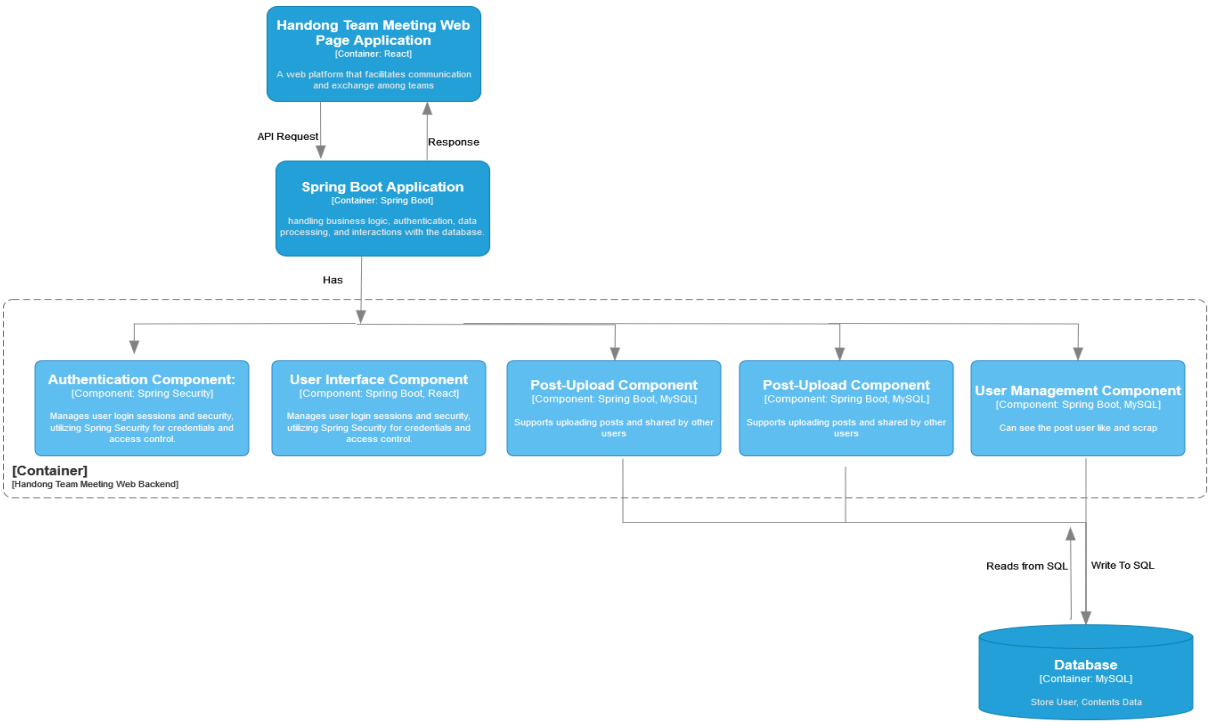
- **Frontend:** Developed with React, serves the dynamic content and interfaces directly with the users through their web browsers.
- **Backend:** Spring Boot application, handling business logic, authentication, data processing, and interactions with the database.
- **Database:** MySQL database, stores user data, meeting records, and archived documents.



4.1.2 Component Diagram on C4

Brief Web Application Components:

- **User Interface Component:** Handles the rendering of the user interface and user interactions.
- **Authentication Component:** Manages user login sessions and security, utilizing Spring Security for credentials and access control.
- **Meeting Management Component:** Supports scheduling, updating, and tracking of inter-team meetings.
- **Document Storage Component:** Interfaces with the Document Management System to facilitate document upload, categorization, and retrieval.
- **Post-Upload Component:** Supports uploading posts and sharing by other users
- **User Management Component:** Can see the post user like and scrap


Handong Team Meeting Web Page Application
[Container: React]
A web platform that facilitates communication and exchange among teams
API Request
Response
Spring Boot Application
[Container: Spring Boot]
handling business logic, authentication, data processing, and interactions with the database.
Has
Authentication Component:
[Component: Spring Security]
Manages user login sessions and security, utilizing Spring Security for credentials and access control.
User Interface Component
[Component: Spring Boot, React]
Manages user login sessions and security, utilizing Spring Security for credentials and access control.
Post-Upload Component
[Component: Spring Boot, MySQL]
Supports uploading posts and shared by other users
Post-Upload Component
[Component: Spring Boot, MySQL]
Supports uploading posts and shared by other users
User Management Component
[Component: Spring Boot, MySQL]
Can see the post user like and scrap
[Container]
[Handong Team Meeting Web Backend]
Reads from SQL
Write To SQL
Database
[Container: MySQL]
Store User, Contents Data

2.9 5/23 Meeting Minutes

Implementation

Backend progress report

- Done with Google authentication
- Done with the file upload
- Done with setting the DB
- Done with Post
- Done with Scrap
- Doe with Likes
- Done with My Page

Frontend progress report

- Done with the main page
- Done with the animation

https://youtu.be/y5CFzX0Zjfw

TODO

complete at least all MVP

MVP (Minimum Viable Product)

- **File Upload**
  - Function to upload and store all materials generated during team meetings.
- **Login System**
  - Google login for easy access.
  - Three levels of permissions: system administrators, professors, and students.
- **Bulletin Board Functionality**
  - Instagram feed-style bulletin board for information sharing.
  - Team meeting matching service to be further developed.
- **Information Delivery Page**
  - A page to efficiently convey important information related to team meetings.
- **Tag-Based Document Retrieval**
  - Ability to search and display documents by tags.

2.10 5/30 Meeting Minutes

Frontend Progress report

https://youtu.be/URvlpgSalxA

When you log in from the home page, it moves to the main page where you can register content. The sidebar is set up so that you can go back to the home page from the profile and ranking pages. Once the database is connected, the content will be listed on the main page as registered.

Backend Progress report: Done with every function

- **File Upload**
  - Function to upload and store all materials generated during team meetings.
- **Login System**
  - Google login for easy access.
  - Three levels of permissions: system administrators, professors, and students.
- **Bulletin Board Functionality**
  - Instagram feed-style bulletin board for information sharing.
  - Team meeting matching service to be further developed.
- **Information Delivery Page**
  - A page to efficiently convey important information related to team meetings.
- **Tag-Based Document Retrieval**
  - Ability to search and display documents by tags.
- **Done with Springboot security**

TODO

- Start integration of frontend and backend

2.11 6/3 Meeting Minutes

Start Integration of the front-end and back-end

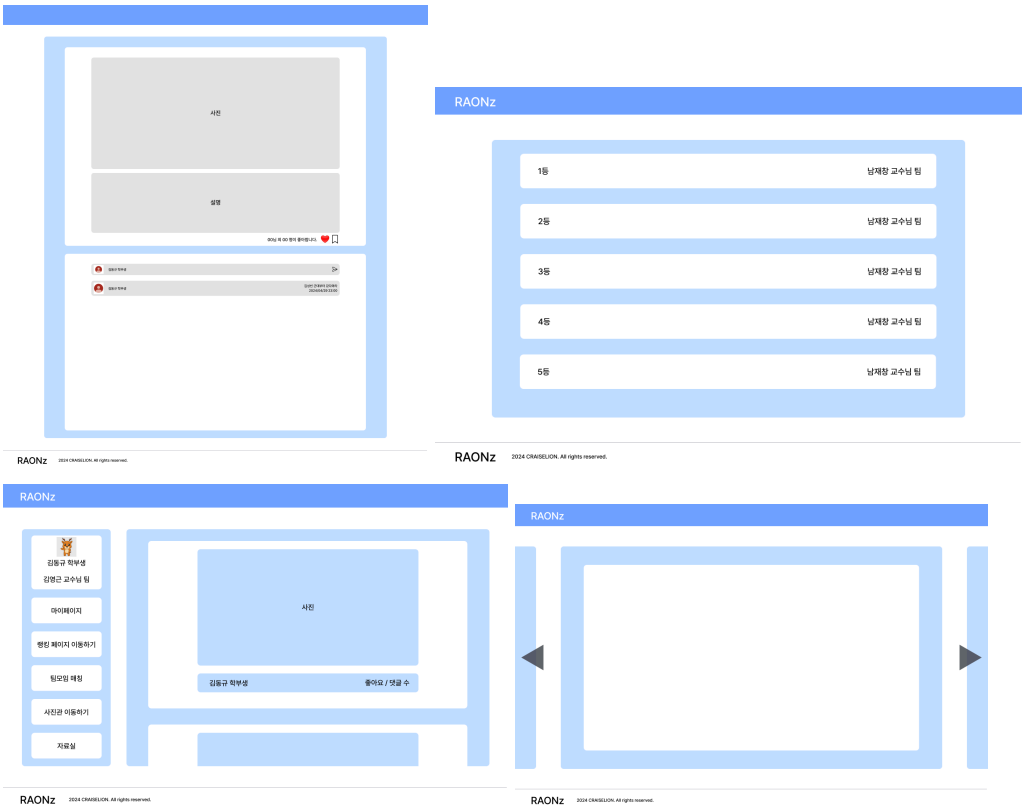
3. FrontEnd Team Meeting Log

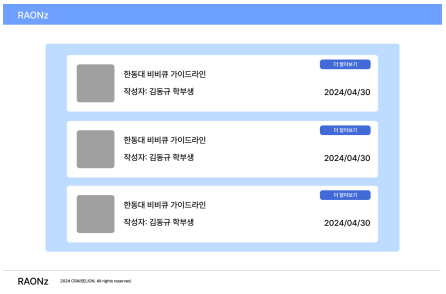
3.1 5/2 Meeting Minutes

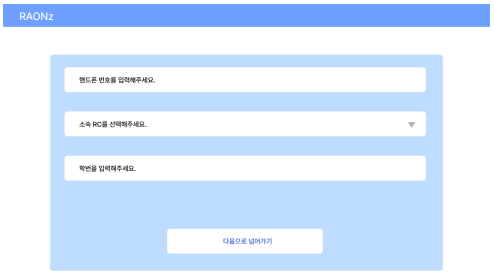
Tasks TODO:

- **Google Login Token**
  - Implement the handling of Google login tokens.
- **Path Configuration and Navigation**
  - Set up the navigation paths:
    - Main Page (`mainPage`)
    - Login Page (`./loginPage`)
    - Ranking Page (`RankingPage`)
- **Finish Design**
  - Complete the design work.

Current Design had a negative feedback.





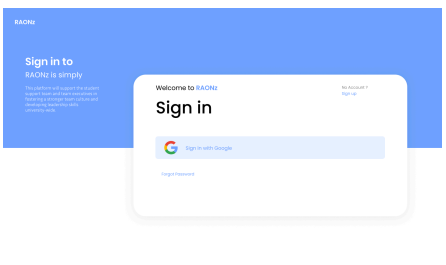


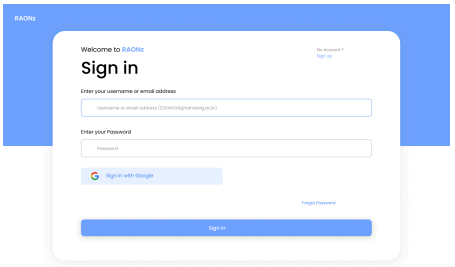
3.2 5/10 Meeting Minutes

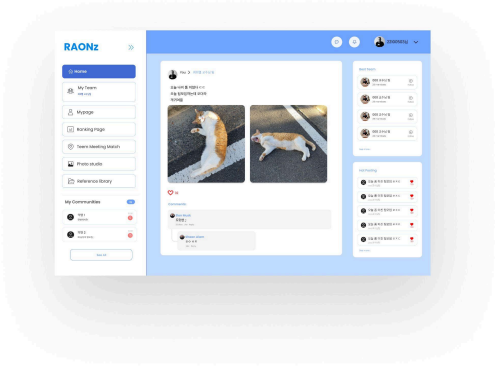
Tasks Done:

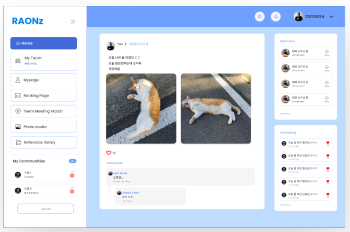
Over the past few days, significant progress has been made on the project. Initially, the router system was set up, and the home and login pages were created on May 5, 2024. Following this, the home page design was updated on May 7, 2024, incorporating a new background and logo, along with an update to the RAONz logo. On May 8, 2024, the home page received another update with the addition of an RC image. The following day, on May 9, 2024, further modifications were made, including updates to the home page image and improvements to the login page. Additionally, navigation paths were set for key pages, ensuring a seamless user experience.

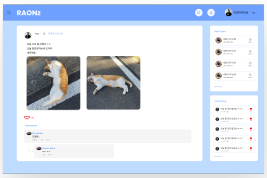
New Design

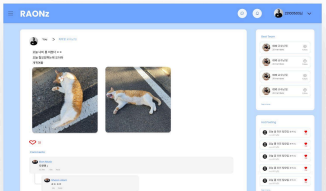






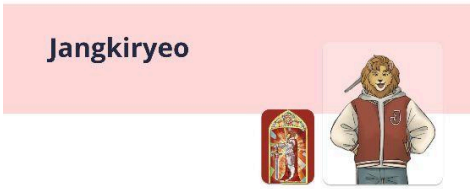






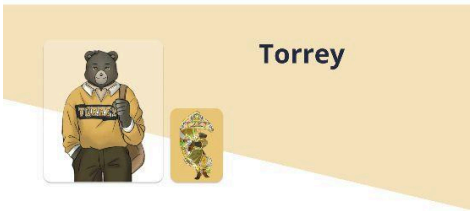

RAONz
HANDONG TEAM MEETING ARCHIVING
AND EXCHANGE WEB PLATFORM

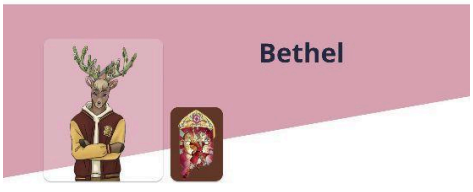

KUYPER

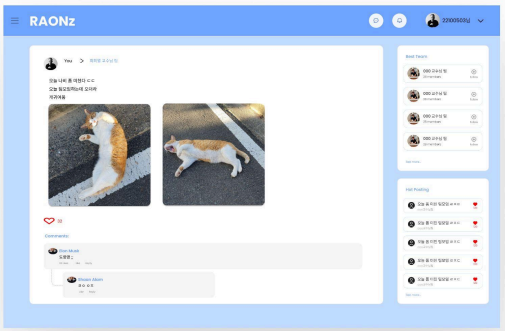

Jangkiryeo

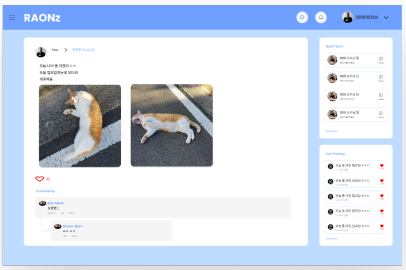

Rodem

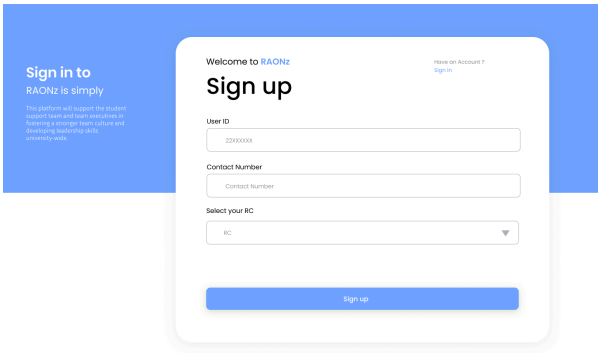

Carmichael

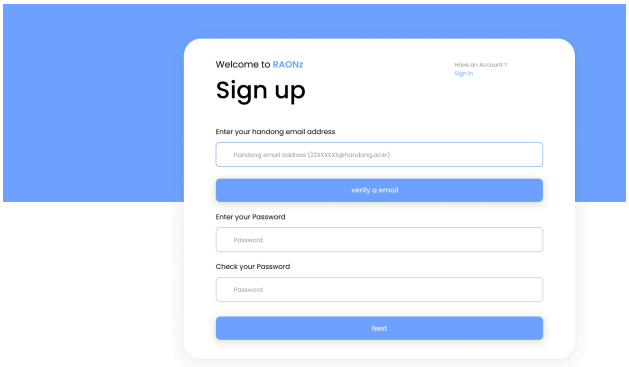

Torrey


Bethel


RAONz


RAONz


Sign in to
RAONz is simply
This platform will support the student support team and team executives in fostering a stronger team culture and developing leadership skills university-wide.
Welcome to RAONz
Sign up
Have an Account ?
Sign in
User ID
22XXXXXX
Contact Number
Contact Number
Select your RC
RC
Sign up


Welcome to RAONz
Sign up
Have an Account ?
Sign in
Enter your handong email address
handong email address (22XXXXXX@handong.ac.kr)
verify a email
Enter your Password
Password
Check your Password
Password
Next

3.3 5/15 Meeting Minutes

- Done with the main page
- Done with the animation

https://youtu.be/y5CFzX0Zjfw

TODO

complete at least all MVP

3.4 5/26 Meeting Minutes

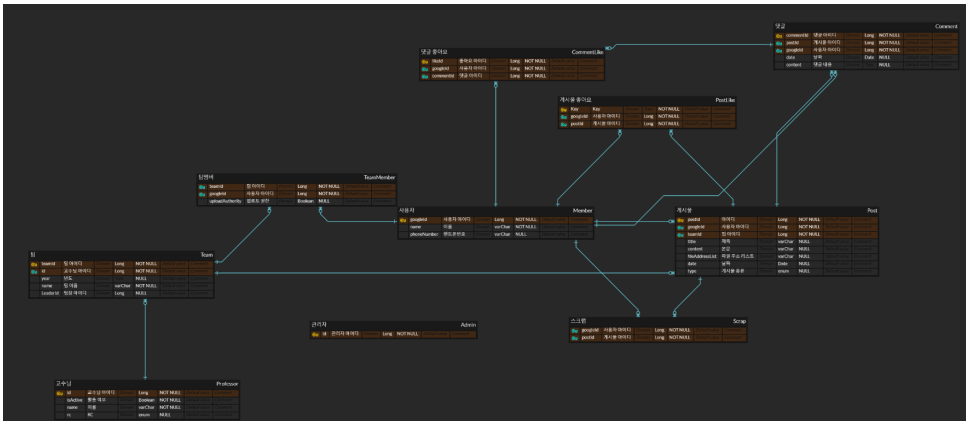
https://youtu.be/URvlpgSalxA

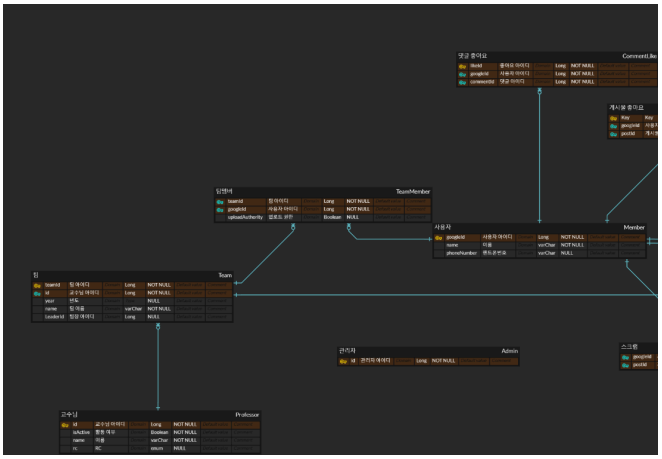
When you log in from the home page, it moves to the main page where you can register content. The sidebar is set up so that you can go back to the home page from the profile and ranking pages. Once the database is connected, the content will be listed on the main page as registered.

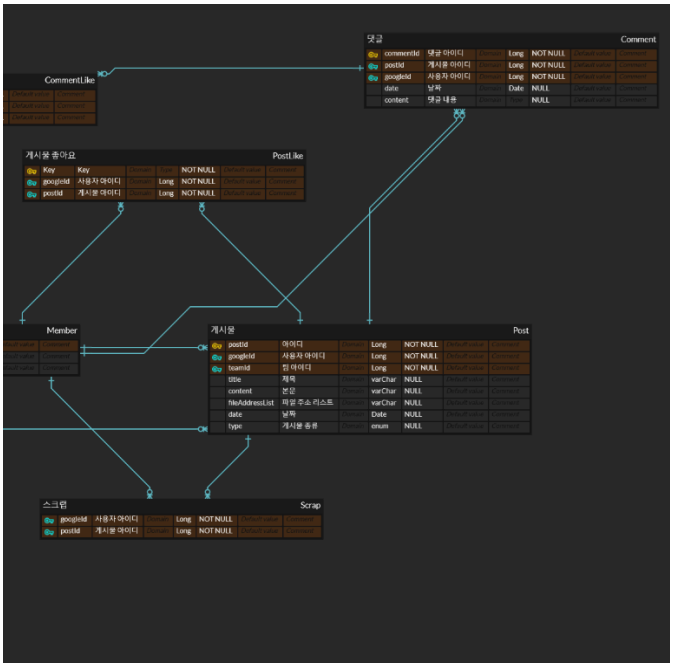
3. BackEnd Team Meeting Log

3.1 5/2 Meeting Minutes

Done with the ER diagram







3.2 5/11 Meeting Minutes

Done with API Details

Request URL	Request Type	Authorization	Description	Return Value
code/google/{registerId}	POST	None	Google login endpoint. Retrieves user info from Google server for signup/login	Spring Security token
post/add	POST	Only users with manager permissions	Only users with manager permissions can create posts. The token must be sent to the bearer.	Created postId or null if failed
post/delete/{id}	DELETE	Manager permissions + same team, Admin can delete directly	Request Param for post ID required, along with the token.	Deleted postId or null if failed
post/get/all/{index}	GET	No restrictions	Provides information about all posts, paginated by 10s.	PostResponseList
post/get/{id}	GET	No restrictions	Provides information about a single post.	PostResponse
post/update/{id}	PATCH	Manager permissions + same team, Admin can update directly	Request Param for postId required, the token used to determine user permissions for the update.	Updated postId or null if failed
team	GET	No restrictions	Token required.	PictureResponseList or null if failed
comment/add/{postId}	POST	No restrictions, Token required	Successfully adds a comment and returns comment ID.	Comment id or null if failed
comment/delete/{commentId}	DELETE	Only comment creator and admin	Token required.	Deleted commentId or null if failed
comment/update/{commentId}	PATCH	Only comment creator	Token required.	Updated commentId or null if failed
material/get/all/{index}	GET	No restrictions	Provides information about all posts, paginated by 10s.	PostResponseList
material/get/{id}	GET	No restrictions	Provides information about a single post.	PostResponse
material/add	POST	Only users with manager permissions	Only users with manager permissions can create posts. The token must be sent in the bearer.	Created postId or null if failed
material/delete/{id}	DELETE	Manager permissions + same team, Admin can delete directly	Request Param for post Id required, along with token.	Deleted postId or null if failed

material/update/{id}	PATCH	Manager permissions + same team, Admin can update directly	Request Param for postId required, token used to determine user permissions for the update.	Updated postId or null if failed
faq/get/all/{index}	GET	No restrictions	Provides information about all posts, paginated by 10s.	PostResponseList
meet/get/all/{index}	GET	No restrictions	Provides information about all meeting posts, paginated by 10s.	PostResponseList
meet/get/{id}	GET	No restrictions	Provides detailed information about a specific meeting post.	PostResponse
meet/add	POST	Admin and manager permissions	Only users with admin or manager permissions can create meeting posts. The token must be sent in the bearer.	Created postId or null if failed
meet/delete/{id}	DELETE	Manager + same team, Admin can delete directly	Request Param for post Id required, along with the token.	Deleted postId or null if failed
meet/update/{id}	PATCH	Admin and manager permissions, same team possible	Request Param for postId required, the token used to determine user permissions for the update.	Updated postId or null if failed
like/add/{postId}	POST	No restrictions request param, and tokens required	Adds like to the specified post and returns the postId.	PostId or null if failed
like/delete/{postId}	DELETE	No restrictions request param, and tokens required	Removes like from the specified post and returns the postId.	PostId or null if failed
like/get	GET	No restrictions, token required	Returns a list of posts the user has liked.	PostLikeList or null if failed
star/add/{postId}	POST	No restrictions request param, and tokens required	Adds a star to the specified post and returns the postId.	PostId or null if failed
star/delete/{postId}	DELETE	No restrictions request param, and tokens required	Removes a star from the specified post and returns the postId.	PostId or null if failed
star/get	GET	No restrictions, token required	Returns a list of posts the user has starred.	StarList or null if failed
team/add	POST	No restrictions, token required	Adds a new team with the specified members. The token must be provided.	TeamId or null if failed

3.3 5/17 Meeting Minutes

- Completed Google login implementation and database integration for user registration and login.
- Implemented JWT token issuance and exception handling for expired tokens.
- Added features for material and FAQ sections.
- Developed functionalities to fetch, update, and delete posts.
- Merged several branches to the main branch.
- Implemented features to add, delete, and view scrap items.
- Updated post and user entity primary keys to resolve conflicts.
- Modified post update and deletion to verify UserId.
- Merged multiple pull requests, resolving conflicts.
- Implemented features to add, delete, and view scrap items.
- Updated post and user entity primary keys to resolve conflicts.
- Modified post update and deletion to verify UserId.
- Merged multiple pull requests, resolving conflicts.
- Completed Google login implementation and connected it to the database for user registration and login.
- Added JWT token issuance and exception handling for token expiration.
- Implemented features for material and FAQ sections.
- Developed functionalities to fetch, update, and delete posts.

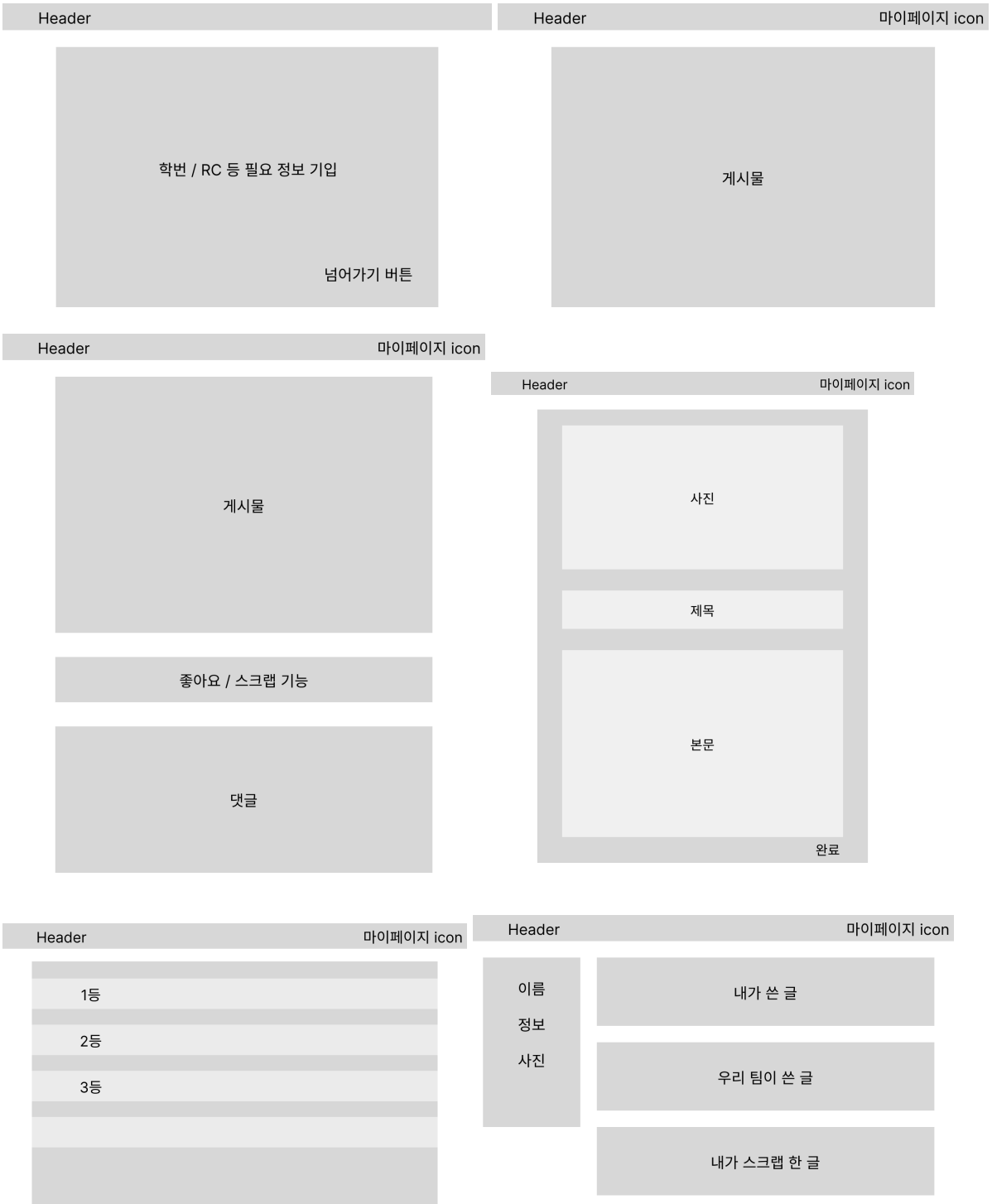
3.4 5/24 Meeting Minutes

- Implemented get and delete functionalities for PostFile.
- Merged pull request #28.
- Added functionality to retrieve PostFile.
- Implemented file list upload functionality for Post.
- Added key JSON file.
- Fixed errors in Post response.
- Merged several pull requests and handled changes.
- Updated Post response to fix errors.
- Added implementation for Post file upload.
- Merged pull request #27.
- Implemented functionalities related to Team creation and adding Team members.
- Added functionality to return like and scrap status when retrieving a page.
- Merged several pull requests (#24, #23, #22).
- Fixed Page Get functionality to return userDto instead of user.
- Implemented functionality to retrieve scrap list and post like list in MyPage.
- Merged pull request #21.
- Implemented adding and deleting PostLike functionality.
- Implemented retrieving and updating MyPage functionality.
- Implemented Comment functionalities: create, update, delete.
- Implemented Comment like functionality.
- Merged several pull requests (#19, #18, #17, #16, #15).

4. Design Team Meeting Log

4.1 4/15 Meeting Minutes

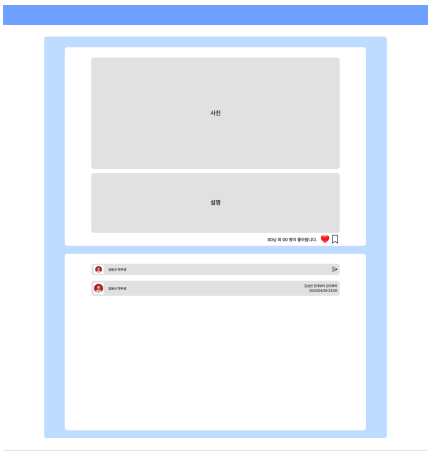
Prototype Confirm

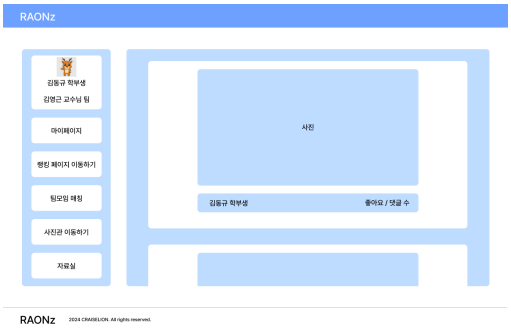


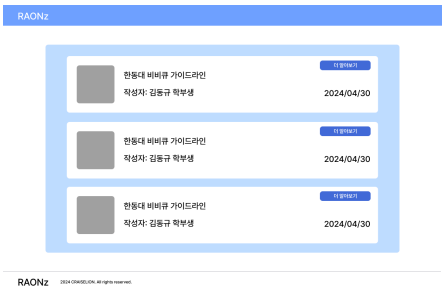

Header
마이페이지 icon
페이지 이동
랭킹 페이지
팀모임 매칭
사진관 페이지
자료실
피드 확인
Header
마이페이지 icon
Header
로그인 버튼
학교 로고 및 설명
RC 사진 / 아이콘
RC 설명
RC 설명
RC 사진 / 아이콘
더 알아보기 / 로그인
Footer
Header
마이페이지 icon
검색창
사진
글
사진
글
Header
마이페이지 icon
이름
정보
사진
모달창 띄우기
글
자세히 보기
글
자세히 보기
글
자세히 보기
Header
마이페이지 icon
이름
정보
사진
글
자세히 보기
글
자세히 보기
글
자세히 보기

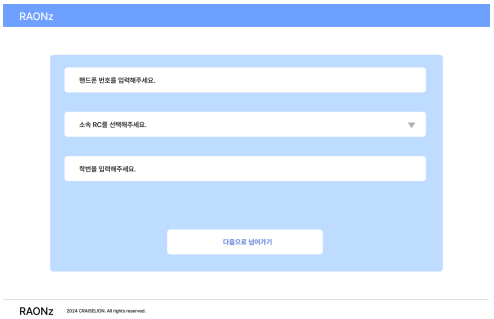
4.2 4/22 Meeting Minutes

2nd Design.







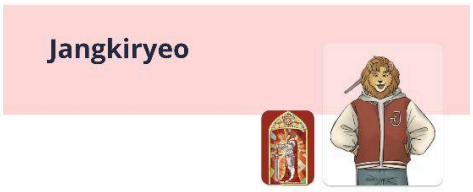


4.3 5/2 Meeting Minutes

Design feedback

- less intuitive
- not user friendly
- need stronger expression on the first page

Main Page Design Change?

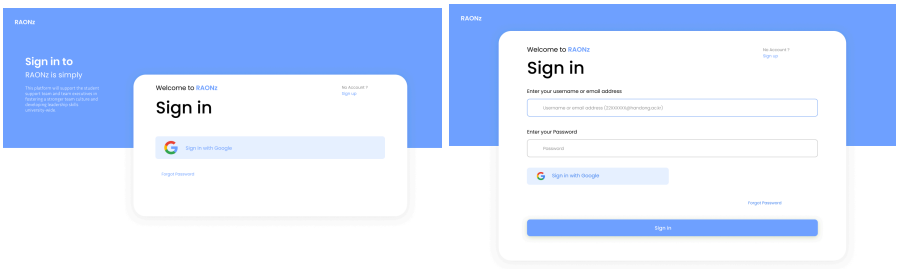


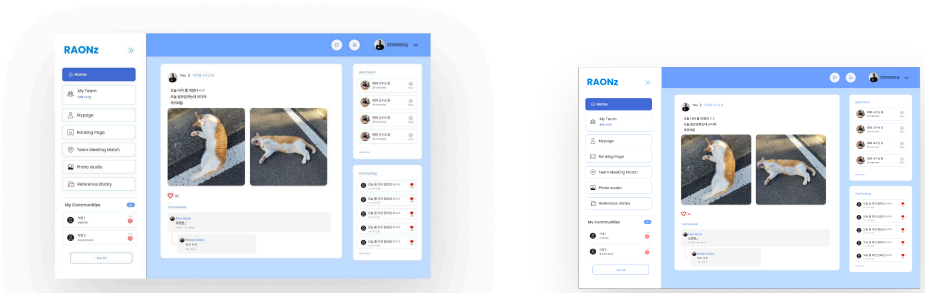
4.4 5/8 Meeting Minutes

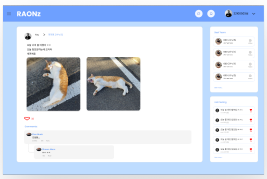
Design Change

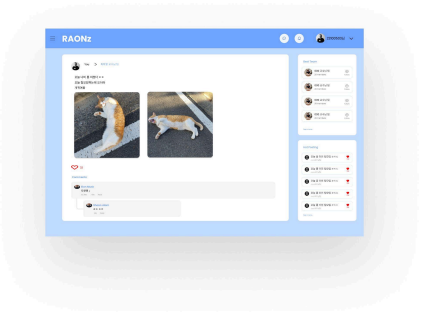
New Design according to the feedback

New Design



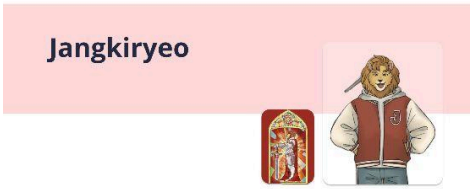






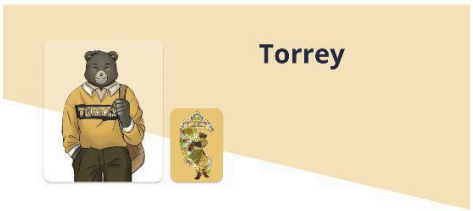

RAONz
HANDONG TEAM MEETING ARCHIVING
AND EXCHANGE WEB PLATFORM

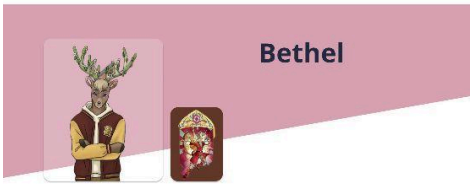

KUYPER

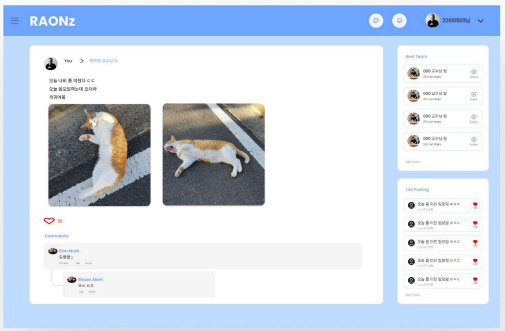

Jangkiryeo

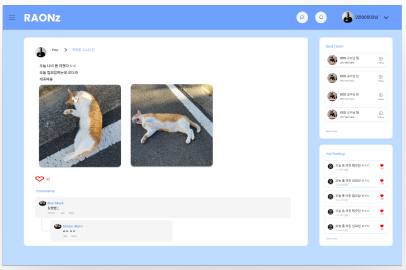

Rodem

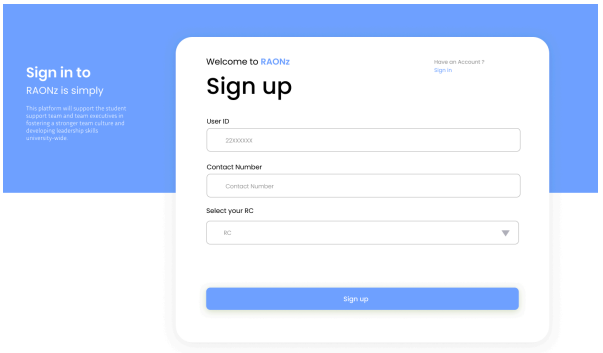

Carmichael

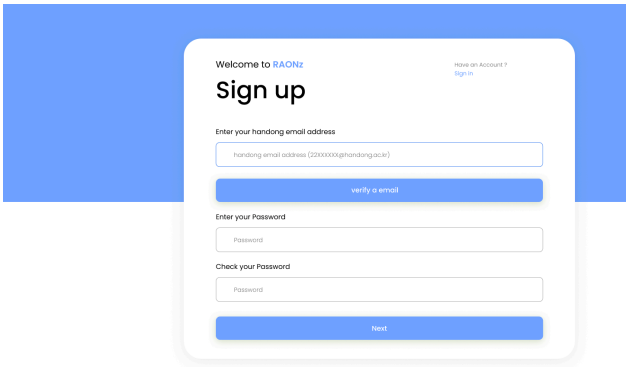

Torrey


Bethel


RAONz

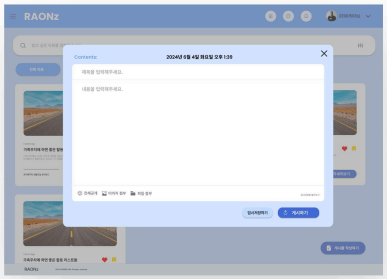

RAONz

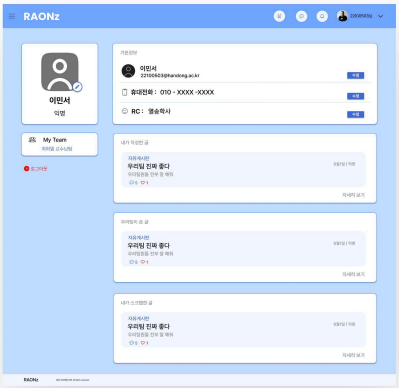

Sign in to
RAONz is simply
This platform will support the student support team and team executives in fostering a stronger team culture and developing leadership skills university-wide.
Welcome to RAONz
Sign up
Have an Account ?
Sign in
User ID
22XXXXXX
Contact Number
Contact Number
Select your RC
RC
Sign up

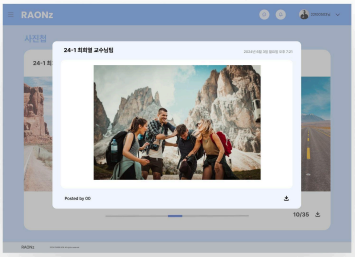

Welcome to RAONz
Sign up
Have an Account ?
Sign in
Enter your handong email address
handong email address (22XXXXXX@handong.ac.kr)
verify a email
Enter your Password
Password
Check your Password
Password
Next

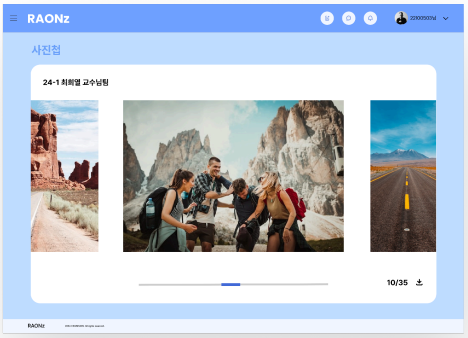
4.5 5/14 Meeting Minutes

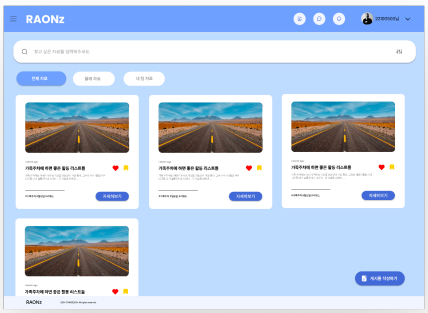
Design Completed

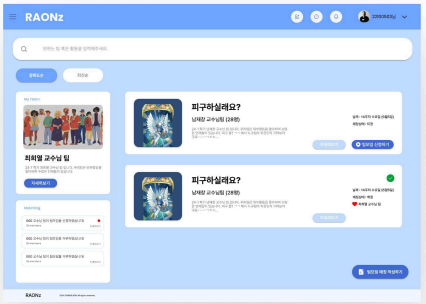


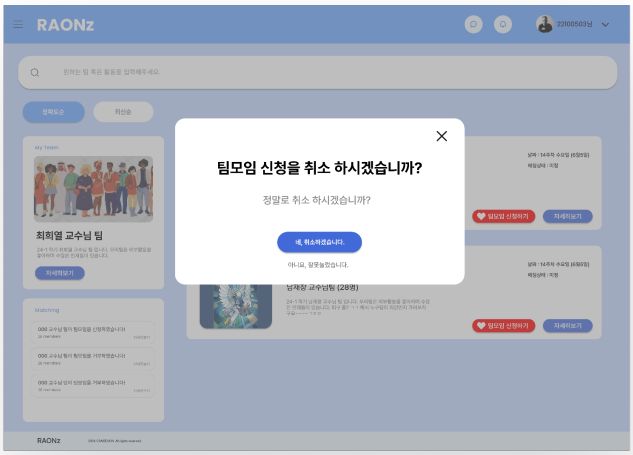







RAONz


RAONz
최희열 교수님 팀
피구하실래요?
피구하실래요?


RAONz
팀모임 신청을 취소 하시겠습니까?
정말로 취소 하시겠습니까?
네, 취소하겠습니다.
아니요, 잘못눌렀습니다.
최희열 교수님 팀


RAONz
My Team
대표사진
최희열 교수님 팀
RC : 로뎀
학기: 24-1
멤버 수: 24명
대표 한 산탈

5. Documentation Team Meeting Log

5.1 4/20 Meeting Minutes

Requirement Engineering Document

https://www.techtarget.com/searchsoftwarequality/definition/MoSCoW-method
Refer to an example that JC gave us
Analyze conducted interviews

Feedback

- Need an explanation about IA

5.2 4/29 Meeting Minutes

System Modeling and Design

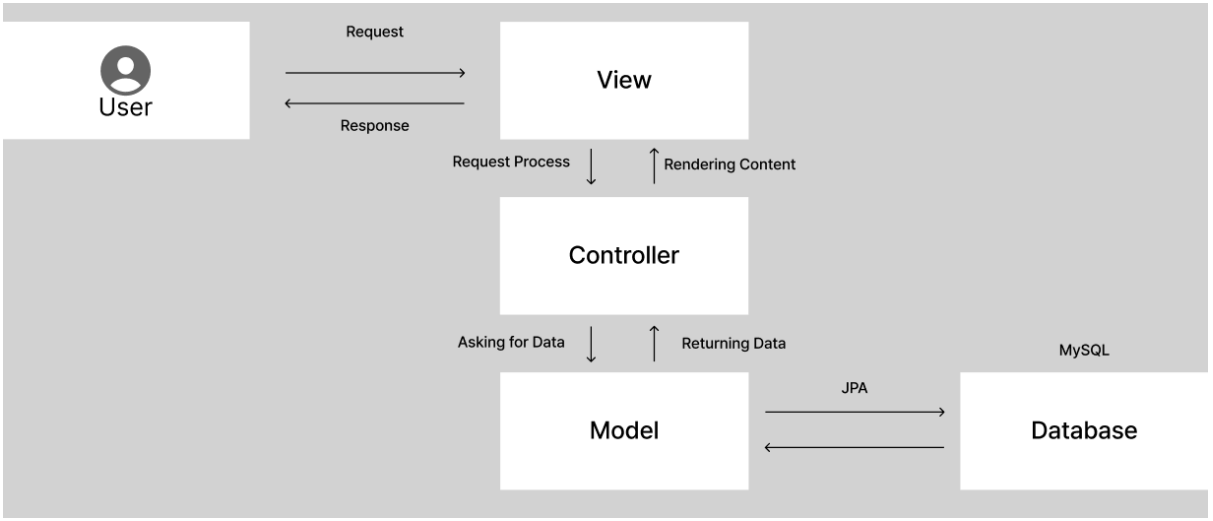
Refer to the C4 model

Patterns

MVC Pattern

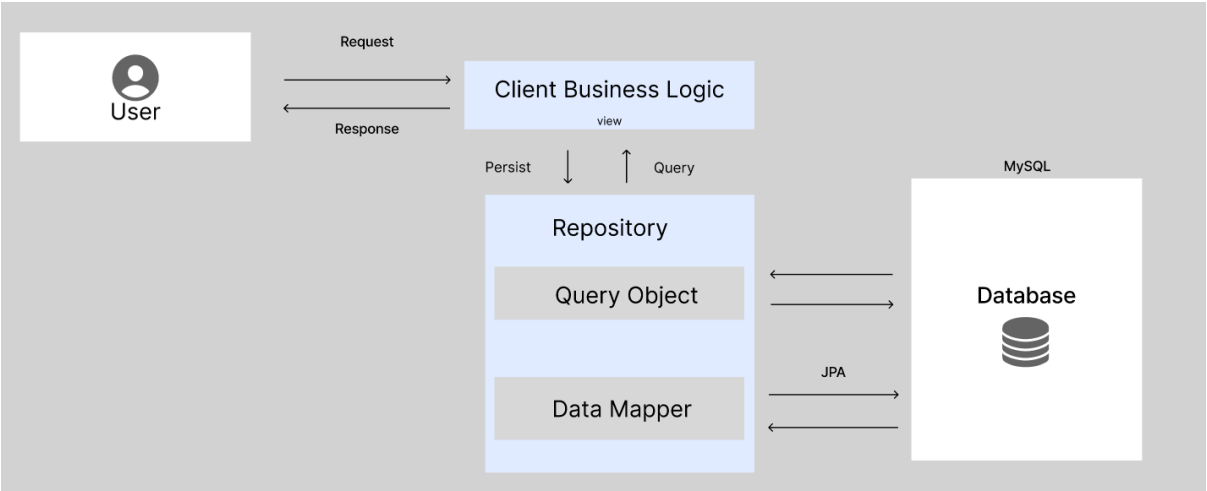
The Model-View-Controller (MVC) is a design pattern used in software engineering. It separates an application into three interconnected components:

- **Model:** This handles data and business logic. It interacts with the database, as shown in your image. The model retrieves data from the database using MySQL and JPA and returns the data to the controller.
- View: This presents data to the user. It's responsible for the user interface (UI) components. In the context of the previous flowchart, the view would be responsible for displaying the various pages like "Photo Page", "Feed Page", etc.
- Controller: This acts as an intermediary that handles input from the user and updates the Model and View accordingly. In your image, the controller is shown processing requests and rendering content.



In the context of the previous question, the MVC pattern could be used to structure the website. The flowchart could represent the user's journey through the View, while the Controller handles the logic of what page to display next, and the Model manages the data related to the user and the content of the pages.

Repository Pattern



The Repository Pattern is a design pattern that mediates between the domain and data mapping layers using a collection-like interface for accessing domain objects.

In the diagram you provided, the Repository Pattern is illustrated as follows:

- User: The user interacts with the application, sending requests and receiving responses.
- Client Business Logic: This layer handles the business logic of the application. It receives user requests and interacts with the repository to retrieve or persist data.
- Repository: This is the core of the Repository Pattern. It communicates with the data layer to perform CRUD operations (Create, Read, Update, Delete). Inside the repository, there are several components:
- Query Object: This object handles specific queries to the database.
- Data Mapper: This component maps the data from the database to objects that the application can use, and vice versa.
- JPA (Java Persistence API): This is a standard interface for accessing databases in Java. It's used to interact with the database.
- Database: This is where the data is stored. The database in the diagram is a MySQL database

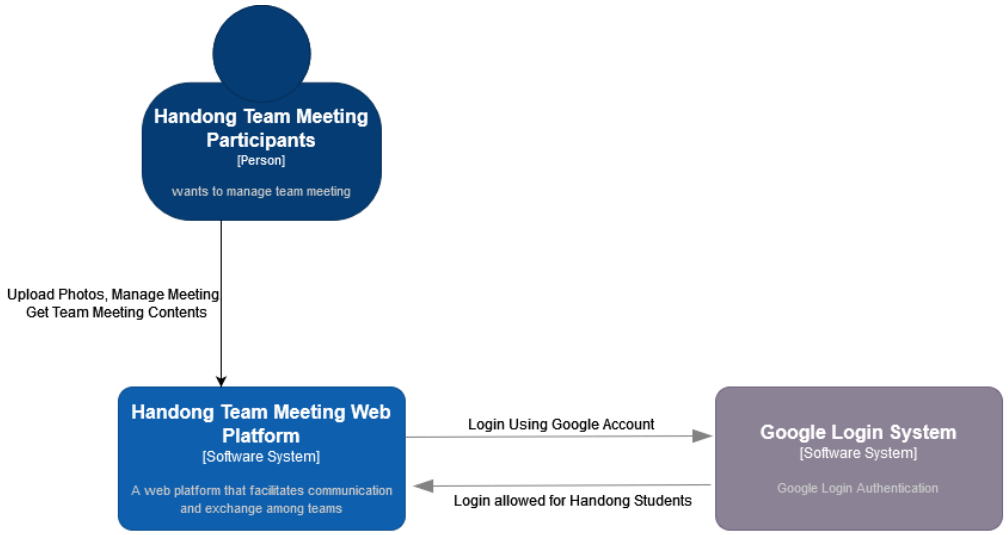
5.2 5/4 Meeting Minutes

C4 model

The detailed technical design is constructed using the C4 model. This slowly zooms in on the different parts of the application. This model is divided into four layers, which are context, containers, components, and code. We are going to draw for Context, Container, and Component for system modeling and design.

Context Diagram

- **System:** Handong University Team Collaboration Platform
- **Purpose:** Facilitate communication and collaboration within and across Residential Colleges (RCs), improve leadership guidance, and archive essential documents.
- **Primary Users:** Students (team executives) and Handong Students

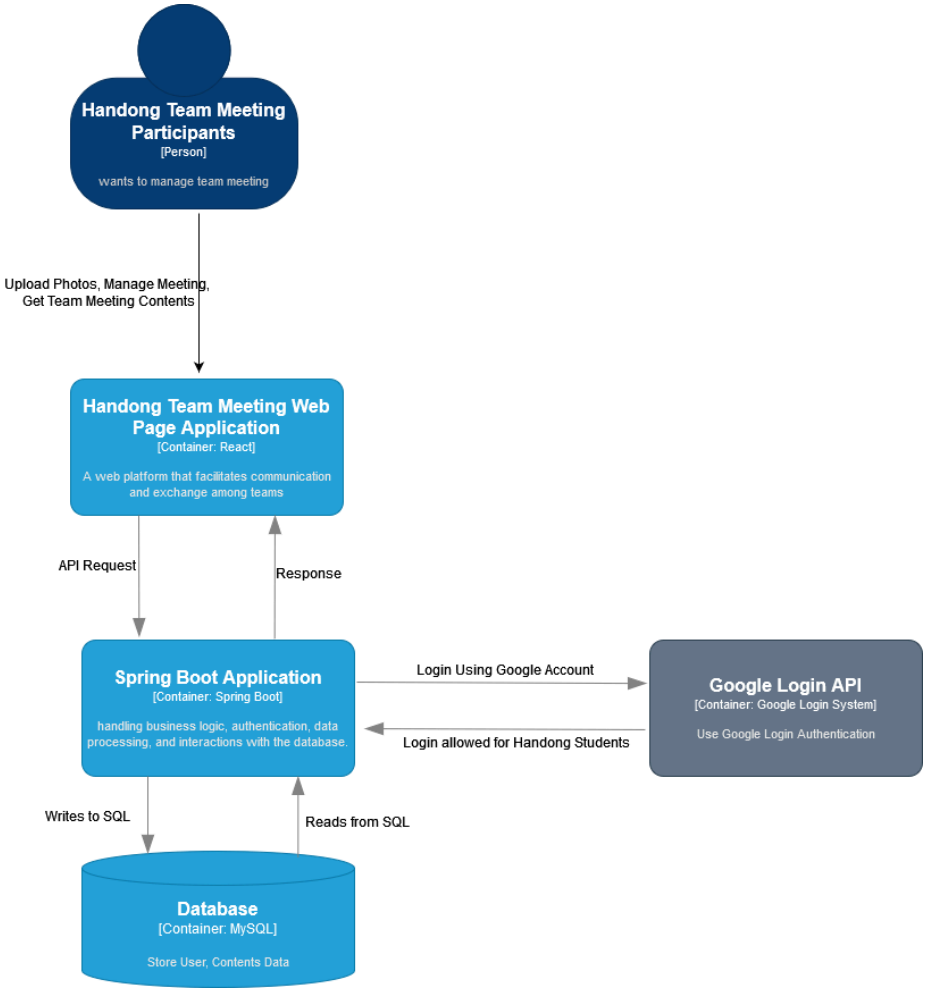


Level 1. Context Diagram

Web Application:

- **Frontend:** Developed with React, serves the dynamic content and interfaces directly with the users through their web browsers.
- **Backend:** Spring Boot application, handling business logic, authentication, data processing, and interactions with the database.

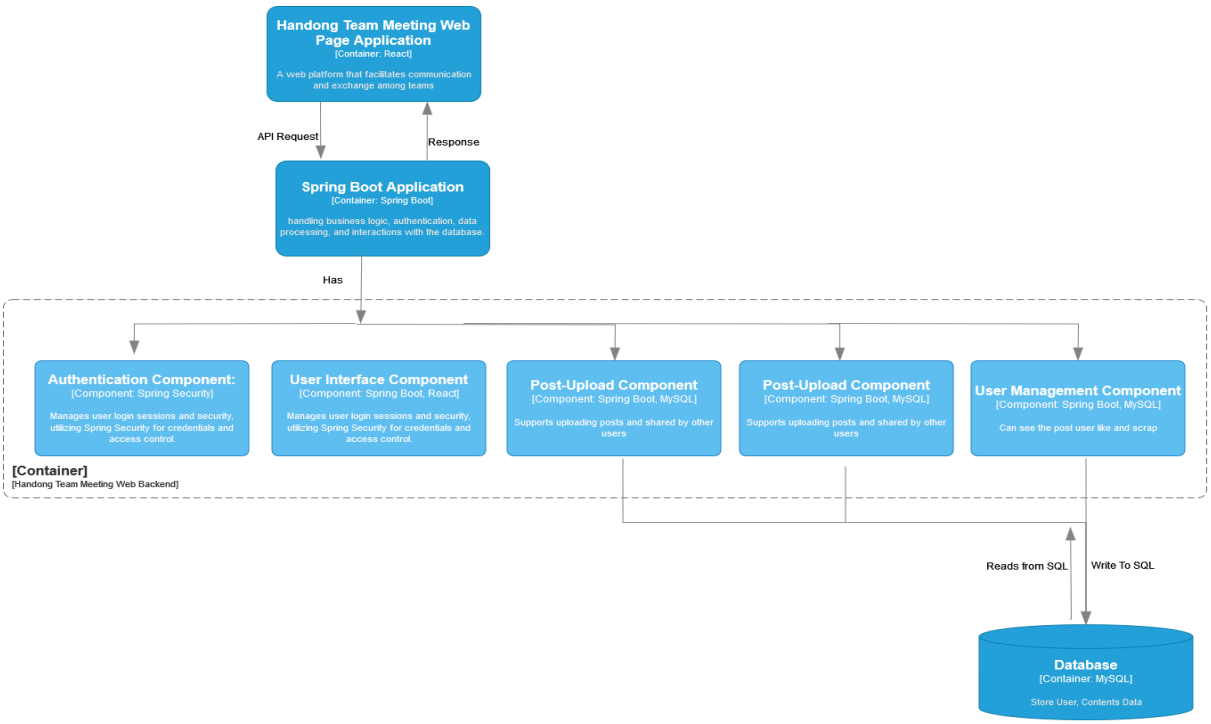
- **Database:** MySQL database, stores user data, meeting records, and archived documents.



4.1.2 Component Diagram on C4

Brief Web Application Components:

- **User Interface Component:** Handles the rendering of the user interface and user interactions.
- **Authentication Component:** Manages user login sessions and security, utilizing Spring Security for credentials and access control.
- **Meeting Management Component:** Supports scheduling, updating, and tracking of inter-team meetings.
- **Document Storage Component:** Interfaces with the Document Management System to facilitate document upload, categorization, and retrieval.
- **Post-Upload Component:** Supports uploading posts and sharing by other users
- **User Management Component:** Can see the post user like and scrap


Handong Team Meeting Web Page Application
[Container: React]
A web platform that facilitates communication and exchange among teams
API Request
Response
Spring Boot Application
[Container: Spring Boot]
handling business logic, authentication, data processing, and interactions with the database.
Has
Authentication Component:
[Component: Spring Security]
Manages user login sessions and security, utilizing Spring Security for credentials and access control.
User Interface Component
[Component: Spring Boot, React]
Manages user login sessions and security, utilizing Spring Security for credentials and access control.
Post-Upload Component
[Component: Spring Boot, MySQL]
Supports uploading posts and shared by other users
Post-Upload Component
[Component: Spring Boot, MySQL]
Supports uploading posts and shared by other users
User Management Component
[Component: Spring Boot, MySQL]
Can see the post user like and scrap
[Container]
[Handong Team Meeting Web Backend]
Reads from SQL
Write To SQL
Database
[Container: MySQL]
Store User, Contents Data