# Software Requirements Specification

# PRJ566 – Fall 2023

# PRJ566 – Team No. 7

# Name of Project:   Seneca Meet-Up Web Application

# Project Leader: Hyunjin Shin

**Last updated: Jan 24, 2024**

**Team Members:**

**1. Hyunjin Shin**

**2. Juhan Kim**

**3. Ke An Lo**

# TABLE OF CONTENTS

1. **Introduction/Overview - Document Information**
   1. **Document Authors**
   2. **Revision History**
   3. **Document Conventions**
   4. **Document Purpose**
   5. **Intended Audience**
   6. **Group Agreement**
2. **Project Overview**
   1. **Project Proposal**
   2. **Stakeholders and Users**
   3. **Functional Requirements**
   4. **Nonfunctional Requirements**
   5. **Project Scope**
   6. **System risks**
   7. **Operating Environment**
   8. **UI/UXD Interface Mockups**
3. **Process & Data Modeling**
   1. **UML Modeling: DFDs & Activity Diagrams**
   2. **Use Case Specification** 
      1. **Business Rules**
      2. **System Use Case Diagrams**
      3. **Use Case Description Tables**
4. **Domain Class Diagram**
5. **Database (Select either 5.1 or 5.2)**
   1. **RDBMS Artifacts**
      1. Scripts to create, populate, delete tables
      2. Data Dictionary
   2. **NoSQL Artifacts**
6. **Work breakdown Structure (WBS)**
7. **Milestones & Acceptance Criteria**
8. **Implementation Schedule (Agile/Waterfall)**
9. **Client / Faculty Sign-off**

# 1 - Introduction/Overview - Document Information

## 1.1 Document Authors

Hyunjin Shin

Juhan Kim

Ke An Lo

## 1.2 Revision History

|  |  |
| --- | --- |
| Week 03 | 1. Introduction/Overview  1.1 Document Authors  1.2 Revision History  1.3 Document Conventions  1.4 Document Purpose  1.5 Intended Audience  1.6 Group Agreement  2. Project Overview  2.1 Project Proposal |
| Week 04 | 1.6 Group Agreement  2.2 Stakeholders and Users  2.3 Functional Requirements  2.4 Nonfunctional Requirements |
| Week 05 | 2.5- Project Scope 2.6- System Risks 2.7- Operating Environment 2.8- Prototype: UI/UXD - Wireframes/Mock-ups  2.3- Functional requirements has been updated based on the feedback from week 4  1.3- Document Conventions has been updated for better readability and distinction to prevent confusion  2.1- in Product Vision and problem statement have been updated to clarify that instructors can also be users and benefit from this application. |
| Week 06 | 3.1 UML/DFD Modeling and Data Modeling |
| Week 07 | 3.2.1 Business Rules  3.2.2 System Use Case Diagrams  3.2.3 Use Case Description  2.3 Functional Requirements |
| Week 08 | Mock-up and Video Development  2.5 Project Scope  3.2.3 Use Case Description |
| Week 09 | 4. Domain Class Diagram  Mock-up and Video Development |
| Week 10 | 5.1.1 ERD  5.1.2 Data Dictionary |
| Week 11 |  |
| Final |  |

## 1.3 Document Conventions

Any text in red indicates changes recently made that are important to check.

Any text in blue is in-progress.

Any text in green has been updated based on the feedback

Any text highlighted in red indicates an exception or error.

Any text highlighted in yellow is an important point.

Any text *italicized* represents definitions.

Any text with ~~strike-through~~ is deleted.

## 1.4 Document Purpose

**A Software Requirements Specification is a comprehensive document that outlines the functional and non-functional requirements of a software system. The main purpose of an SRS is to serve as a blueprint for the development team, providing a clear and detailed description of what the software is supposed to do and how it is expected to perform.**

## 1.5 Intended Audience

**The Software Requirements Specification (SRS) document is designed to cater to a variety of audiences involved in the software development process.**

* **Project Managers**
* **Development Team**
* **Quality Assurance Team**
* **UI/UX Designers**
* **Sponsors**
* **Maintenance/Support Team**

## 1.6 Group Agreement

**TEAM AGREEMENT**

**Team #: 7**

**Project Title:** Seneca Meet-Up Web Application

**Project Time Frame: Start: 8th January 2024 – End: 23rd August 2024**

**Team Members: Hyunjin Shin, Juhan Kim, Ke An Lo**

**Team Leadership: - Team leadership will be rotated monthly among team members.**   
**Leadership Responsibility: The month's team leader will coordinate and oversee the team's activities. Responsibilities include setting agendas for meetings, facilitating communication, and ensuring tasks are progressing according to the project timeline.**

**Team Functions:**

* *We will primarily use MS Teams for real-time communication and updates. Will also make use of GitHub, Google Docs, etc.*
* *Attend team online meetings on a regular basis and participate in discussions.*
* *Complete assigned tasks, progress reports, and other documents needed.*
* *Let other team members know if he/she cannot attend the meeting or work on the task and provide reasons for individuals inability to attend.*
* *Let other team members know if he/she has any difficulties with working on an assigned task and ask for advice and help if needed.*
* *The team will review the work that is done and the information to be added to the final document on Saturday from 10:00 PM to 12:00 AM.*
* *If a group member would like to modify the final document after team review, they will notify other team members, and update the final document before Sunday at 11:00 PM and the team leader will upload the document on Sunday before 11:30 PM*

**Team Meetings: Every Wednesday at 5:15 PM online via MS Teams. Additional team meetings might be scheduled if needed.**

**Team Problems: - In the event of conflicts, team members will openly communicate their concerns, and efforts will be made to find a resolution through discussion.**  
**We will discuss as a group how to handle the situation**  
**Weighing the pros and cons of the individual’s contributions while considering the reasons the individual is not contributing as much as he/she should.**  
**Ultimately, voting on whether the individual should remain as part of the group and conveying the result to the professor if the individual is voted off**

**Team Commitment**

**The undersigned members (Hyunjin Shin, Juhan Kim, Ke An Lo) agree to work together on the project until the end of the PRJ666 next Semester. They recognize that as a team and individually they are responsible for the quality of all deliverables.**

**Name Date**

|  |  |
| --- | --- |
| Hyunjin Shin | Jan 24, 2024 |
| Juhan Kim | Jan 24, 2024 |
| Ke An Lo | Jan 24, 2024 |
|  |  |

ShapeShapeShapeShapeShapeShapeShapeShape

# 2 - Project Overview

## 2.1 Project Proposal

Project Background

Seneca College lacks a dedicated platform for student interaction and information exchange. The absence of a space for students to freely share opinions, ask questions, and organize activities has inspired us to initiate this project.

**Problem Statement**

|  |  |
| --- | --- |
| The Problem of: | Lack of a dedicated platform for student interaction and information exchange. |
| Affects: | Seneca students; Seneca faculty; members of Seneca Student Life; Seneca Employees. |
| The impact of which is: | student dissatisfaction; possible decrease of academic achievement due to lack of peer support; students' failure to receive event information; increased interaction and understanding between students and instructors |
| A successful solution would: | Provide a virtual platform that enables students to exchange information, pose questions, access event details, and express their opinions |

**Product Vision**

|  |  |
| --- | --- |
| For | Seneca Students |
| Who | are looking for enhanced opportunities to connect with their peers and instructors in order to share experiences and build a sense of community within the Seneca environment. |
| The Product Name | Seneca Social Is a web application. |
| That | Features user authentication, streamlined organization of group activities, effortless creation of posts and comments, and post management through a robust database system. |
| Unlike | MS Teams |
| Our product | Ensures a Seneca-specific environment, user-friendly comment and posting features, efficient and easy organization of group activities, and seamless mobile device accessibility |

## 2.2 Stakeholders and Users

|  |  |
| --- | --- |
| Stakeholder Name/Identifier | Category |
| CEO (Chief Executive Officer) | Administration, Sponsor |
| School Principal | Sponsor |
| Construction Manager and Scheduler | Administration, User  Needs accurate up to date information for costing and scheduling of project details |
| IT Administrator | Administration, User |
| Student Affairs Coordinator | Student Services |
| Students | User |
| Social Media Coordinator | Marketing |
| Event Coordinator | Events |
| Project Leader | Developers |
| Developers | Developers |
| UI/UX Designers | Design |
| QA Team | Quality Assurance |

## 2.3 Functional Requirements

|  |  |
| --- | --- |
| Requirement Type | Description |
| User Authentication and Authorization | * Seneca students and employees to create user accounts. * Authentication mechanisms to verify the identity of users during the login process. * Authorization roles will be defined, allowing various levels of access for regular users, event organizers, and authorized personnel like members of Seneca Student Life (SSF). |
| Event Notice Management | * Users with appropriate roles (e.g., SSF members) can create, update, and delete school event announcements. * Students can participate in events by following the instructions, enhancing their engagement with official college activities. * Event announcements will include details such as date, time, location, and instructions for participation. * Users will be able to view a list of upcoming events and filter them based on categories or preferences. |
| Activity Management | * Any user will be able to create and manage activities, specifying participant limits based on factors like program, semester, or gender. * Activities will have details such as date, time, location, and a brief description. * Users will be able to leave comments on activity pages and qualified individuals can apply to participate. |
| Freeboard | * Users should be able to create, update, delete, and read posts. * User should be able to leave comments on posts * Users should be able to insert images, video, or audio files in the post. * Comments should only be texts. |
| Q&A | * Allows users to submit tickets regarding any questions related to, but not limited to: System, Events, Account, Policy, etc. * Allows users to receive replies through notifications and emails. * Potential development of chat with live agents. |
| User Profile | * Allows users to post and modify personal information including Name, Photos, Program, Current year, Age, etc. * Option to hide certain information in terms of privacy (Age, Photo), while keeping transparency to a level (Cannot hide Name, Program). * Allows users to view other profiles and send a “friend request” * Allows users to create, modify, and delete their status. * Users can check the notifications for activities in this page (like dashboard) |
| Notifications | * The system will send notifications to users for important events, activities, or responses to their posts. * Users will have the option to customize their notification preferences. |
| Search and Filter | * The application will include search functionality to allow users to find specific events, activities, or posts. * Filters will be available to refine search results based on different criteria, such as date, category, or user type. |

## 2.4 Nonfunctional Requirements

|  |  |
| --- | --- |
| Requirement Type | Description |
| Operational | * The system should be accessible 24/7 with planned downtime not exceeding 1 day per month. * The system should be compatible with Google Chrome, Safari, Firefox, and Microsoft edge. * The UI/UX should be mobile responsive. |
| Performance | * The system should be able to handle more than 5,000 simultaneous connections. * The database should be able to store more than 50,000 user accounts. |
| Scalability | * The system architecture should be scalable for the possible future expansion of functionality * The authentication system should be able to be implemented using a third-party API. |
| Security | * User passwords must adhere to a minimum complexity standard (e.g., length, special characters). * User passwords must be stored securely using encryption algorithms * To edit and access user’s personal information, they are required to authenticate once more. * User’s personal information can only be accessed by the user and the system administrator. |
| Cultural & Political | * The system must comply with accessibility standards (WCAG) to ensure inclusivity |

## 2.5 Project Scope

1. Project Objectives:

The primary objective of the project is to establish a user-friendly web application, titled the Seneca Social, where Seneca students can seamlessly access school event information, share valuable insights, organize activities, and build connections with their peers.

2. Deliverables:

* Seneca Social
* Business Case Proposal
* Software Requirements Specification
* Implementation Schedule
* Video for Presentation
* Team Presentation

3. Project Boundaries:

The project scope includes the development of the mobile application and backend server. It does not include marketing activities or ongoing maintenance post-launch.

4. Project Constraints:

* Time constraints: The project analysis, planning and design must be completed before **22nd April 2024**
* Resource limitations: Limited availability of developers and designers

5. Project Assumptions:

* Our project has no budget constraints
* Seneca College and Seneca Student Life will be cooperative
* Seneca Students will agree on privacy policy and other policies required for using the web application.

6. Key Stakeholders:

* Seneca Members: Principal of Seneca College, Student Affairs Coordinator, Seneca Student Life Administrator, Seneca Event Coordinator, Seneca Students, Seneca Faculties, Project Leader
* IT Teams: IT Administrator, Developers, UI/UX Designers, QA Team, DevOps

7. Project Timeline:

* Start Date: Jan 15, 2024
* End Date: 23rd August 2024
* Milestones:
* Completion of Project Analysis
* Completion of Project Planning and Design
* Completion of Project Implementation
* Completion of Project Final Testing
* Completion of Deployment

8. Project Risks:

* One of the Project Members may be no longer able to participate in the project or drop out of the course due to personal reasons such as a VISA issue or financial issue
* Competition from existing similar applications
* Seneca Students may not be interested in the application.

9. Resource Requirements:

* Developers skilled in Web Application
* Backend developers
* Designers for UI/UX
* DevOps for hosting the backend server

10. Quality Standards:

* User interface must be intuitive and easy to navigate.
* App must be mobile responsive and compatible with various browsers such as Chrome, Edge, Safari, Firefox, etc.
* App must be stable and free from major bugs.

11. Approval Criteria:

App must pass testing phase with no critical issues.

12. Communication Plan:

Weekly Meeting will be conducted on MS Teams, and Milestones and Issues for the porject will be managed via GitHub.

13. Dependencies:

Availability of third-party APIs for certain functionalities (e.g., AWS Cognito authentication, or AWS S3)

14. Exit Criteria:

* App successfully deployed on Cloud.
* Handover of all documentation and assets to Seneca College.
* Final project review meeting with stakeholders.

## 2.6 System Risks

|  |  |
| --- | --- |
| **Risk** | **Response** |
| The project scope may expand beyond the initial requirements, i.e. scope creep, leading to delays, budget overruns, and resource constraints. | Establish clear project scope and requirements upfront. Implement change management processes to evaluate and prioritize requested changes. |
| Ineffective communication among team members may lead to misunderstandings, delays, and rework. | Establish clear communication channels and protocols. Conduct regular meetings, status updates, and progress reports to ensure everyone stays informed and aligned. |
| The app collects and stores sensitive information about students and faculty, raising concerns about data privacy and security breaches. | Implement data encryption techniques, strict access controls, and regular security audits. Comply with relevant data protection regulations and educate users in advance. |
| The platform may become a breeding ground for cyberbullying or the spread of inappropriate content among students. | Implement content moderation tools and mechanisms for reporting and removing inappropriate content. Educate users about responsible online behavior and provide resources for handling cyberbullying incidents. |
| Technical glitches, server downtime, or app crashes may disrupt access to important information and communication channels. | Conduct thorough testing during development phases, including stress testing and load testing, to identify and address potential performance issues. Implement redundancy and failover mechanisms to minimize downtime. Provide timely updates and support to address any technical issues promptly. |

## 2.7 Operating Environment

1. **Hardware Requirements:**

* The application will be hosted on a web server capable of handling concurrent user requests and database transactions efficiently handling concurrent user requests and database transactions.
* The server hardware should have sufficient processing power, memory, and storage to support the expected user load and data storage requirements.
* Clients accessing the application will require standard computing devices such as desktop computers, laptops, tablets, or smartphones with modern web browsers.

1. **Software Requirements:**

* Web Server: The application will be deployed on a web server software stack capable of serving web pages, handling HTTP requests, and executing server-side code.
* Database Management System (DBMS): The application will utilize a relational database management system (RDBMS) for storing user data, event information, and other application data. Options include MySQL, PostgreSQL, or MongoDB.
* Programming Languages and Frameworks: The application will be developed using web development technologies such as HTML, CSS, JavaScript, and a server-side scripting language like Python, or Node.js. Frameworks such as Express may be used to streamline development.
* Authentication and Authorization: The application will implement secure authentication mechanisms using industry-standard protocols such as JWT (JSON Web Tokens). User passwords will be securely hashed and stored using encryption techniques.
* Version Control: Source code management will be facilitated using version control systems such as Git, with repositories hosted on GitHub.
* Development and Testing Tools: Development will be supported by integrated development environments (IDEs) such as Visual Studio Code. Automated testing frameworks like Cypress may be used for unit testing, integration testing, and end-to-end testing.

1. **Network Infrastructure:**

* The application will require internet connectivity to allow users to access it from various locations.
* Secure communication protocols such as HTTPS will be used to encrypt data transmitted between clients and the server.
* Adequate bandwidth and network resources will be necessary to support the expected volume of concurrent users and data transfer.

1. **Deployment Environment:**

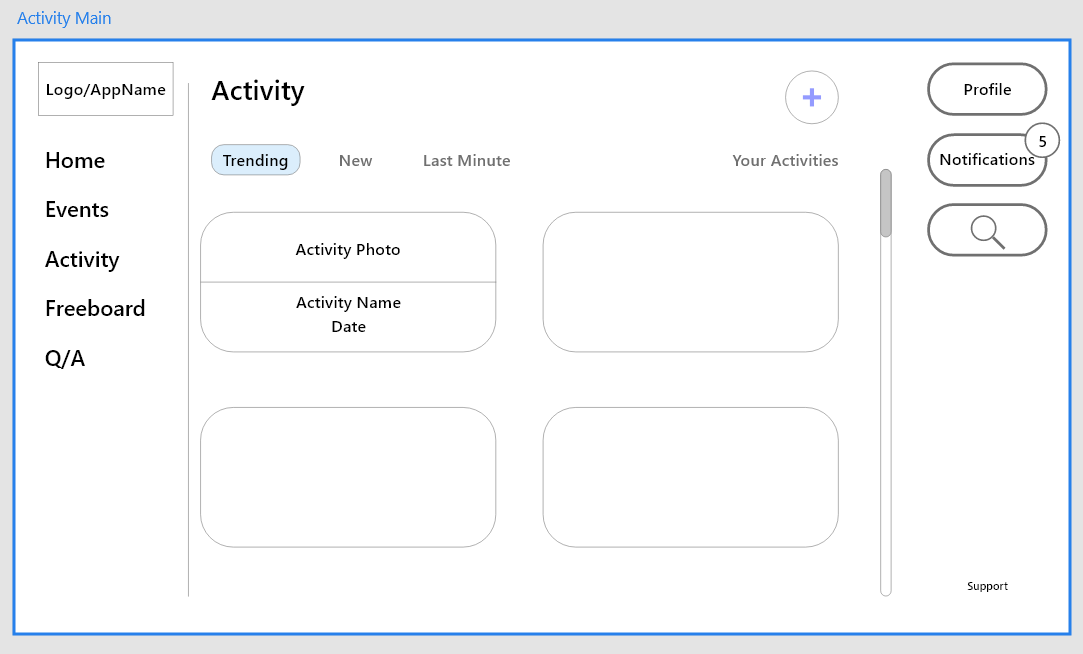
* The application will be deployed to a production environment hosted on a reliable cloud platform such as Vercel.
* Continuous integration and continuous deployment (CI/CD) pipelines will be established to automate the deployment process and ensure reliable and efficient updates to the application.
* Monitoring and logging tools will be implemented to track application performance, identify issues, and facilitate troubleshooting in real-time.

## 2.8 UI/UXD Interface Mock-ups 2, 2, 3

1. Home (one page) - Ke An Lo

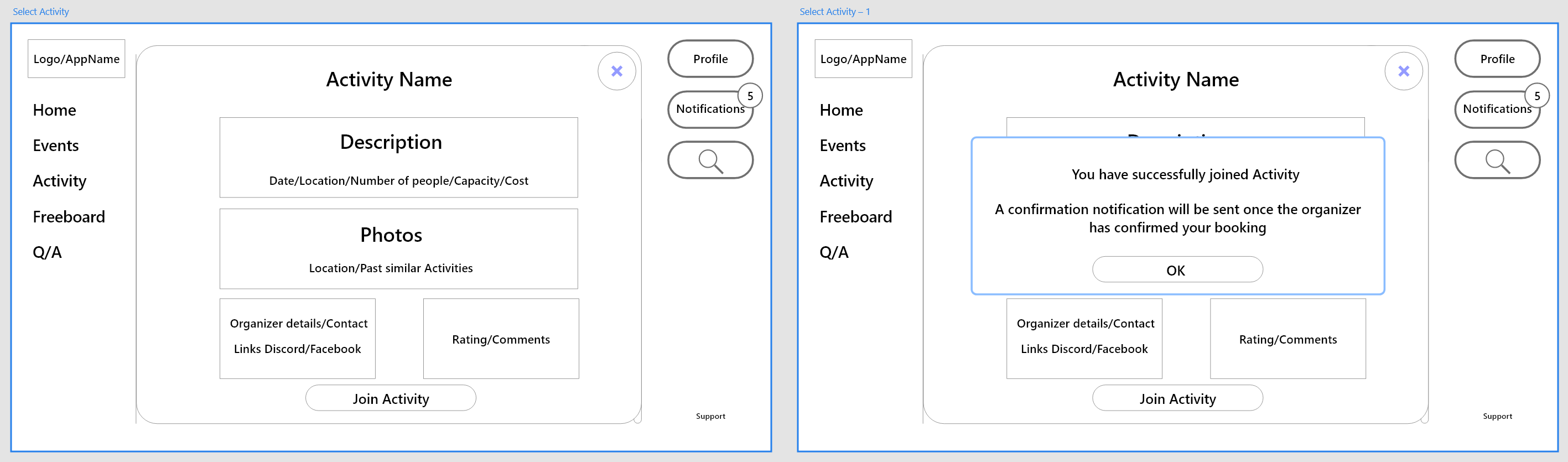
2. Activity Management – Juhan

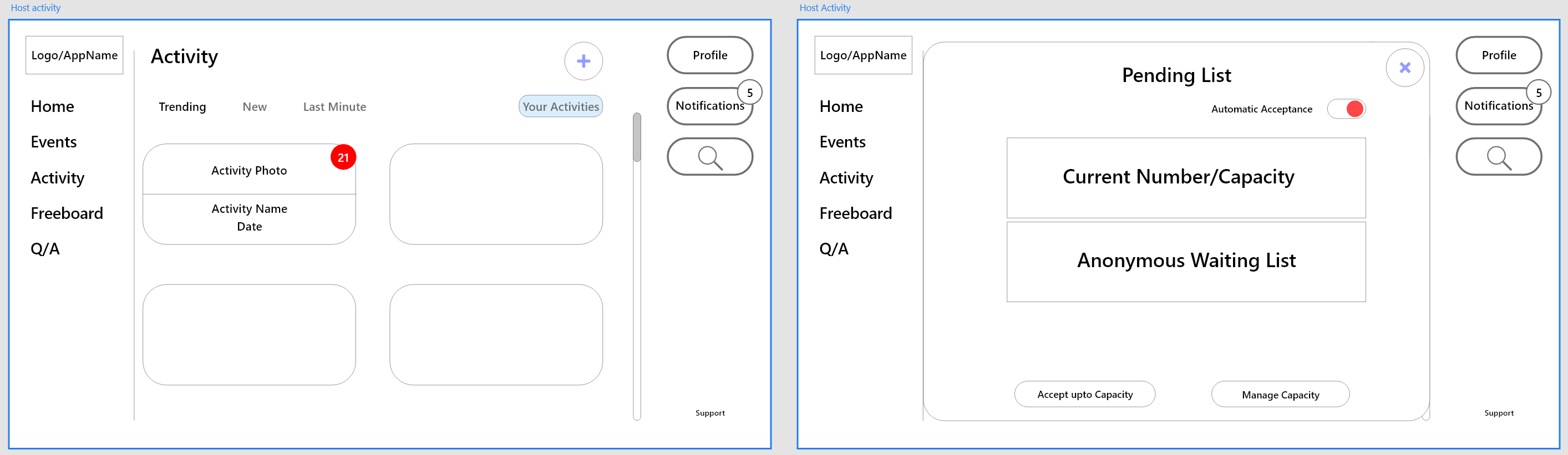
1) Activity Main



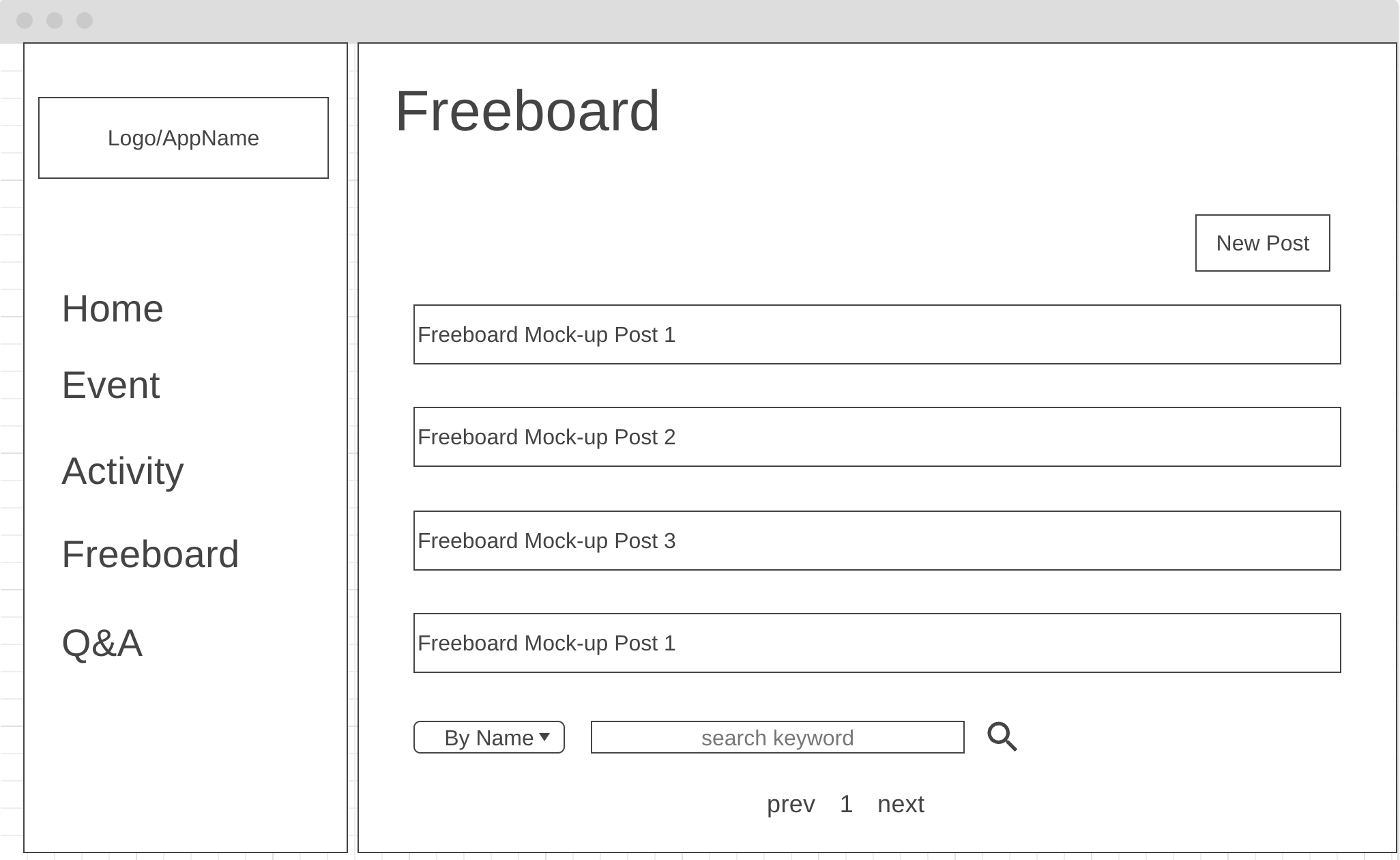
2) Create Activity

 3) Select Activity

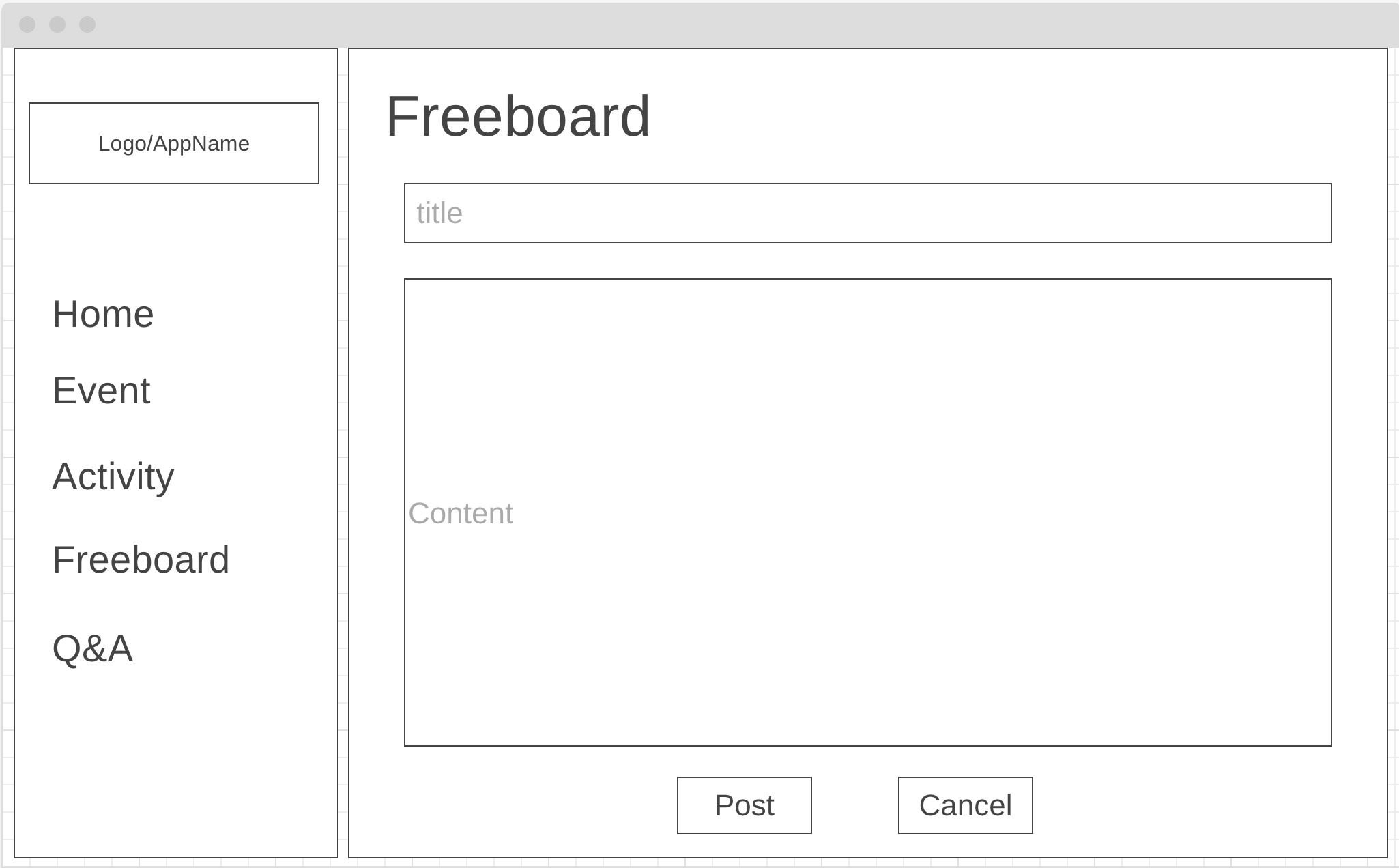


4) My Activity View 3. Freeboard (Assumption: user is already signed in) – Hyunjin Shin

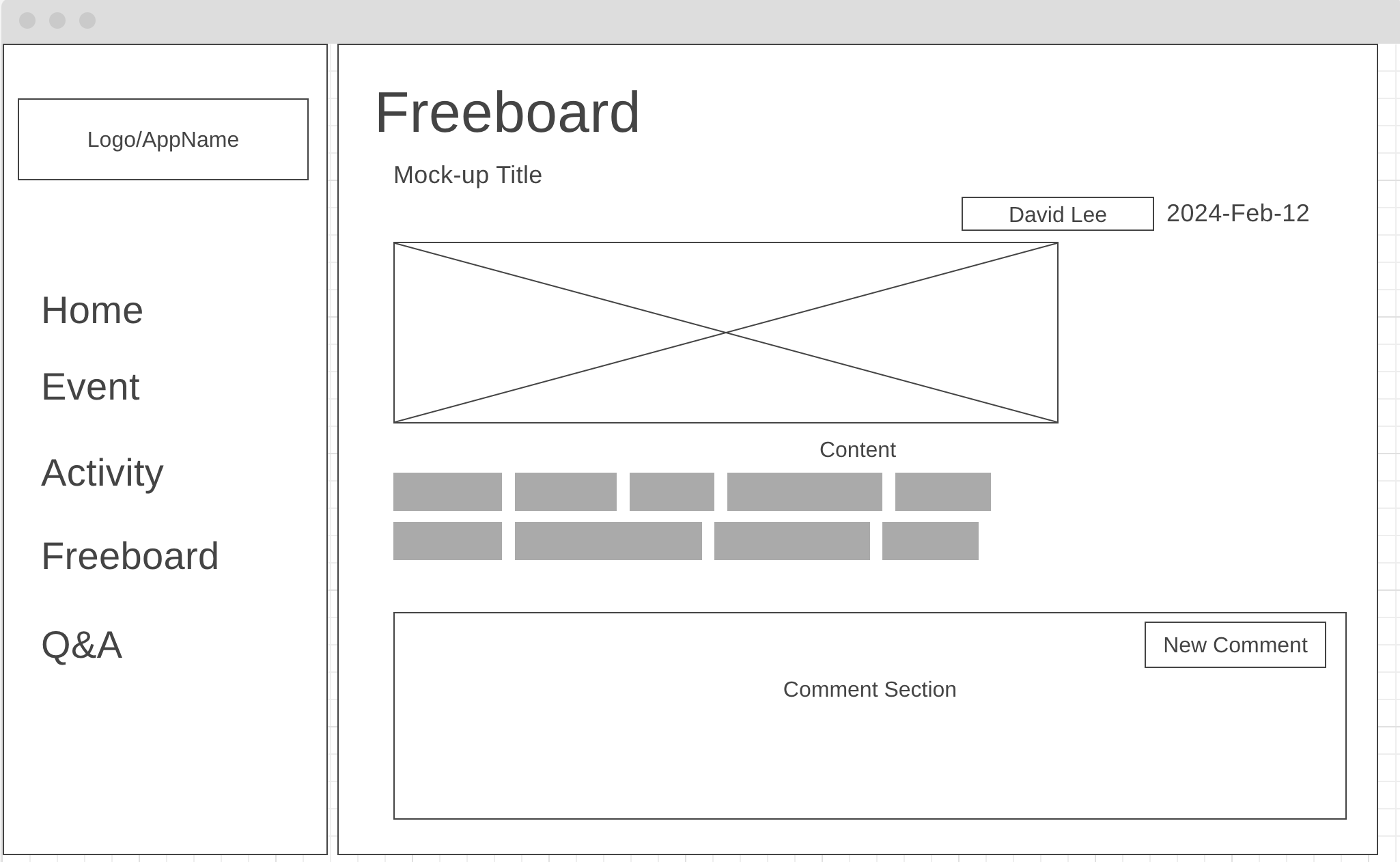
1) Visit Freeboard Main Page (Query)



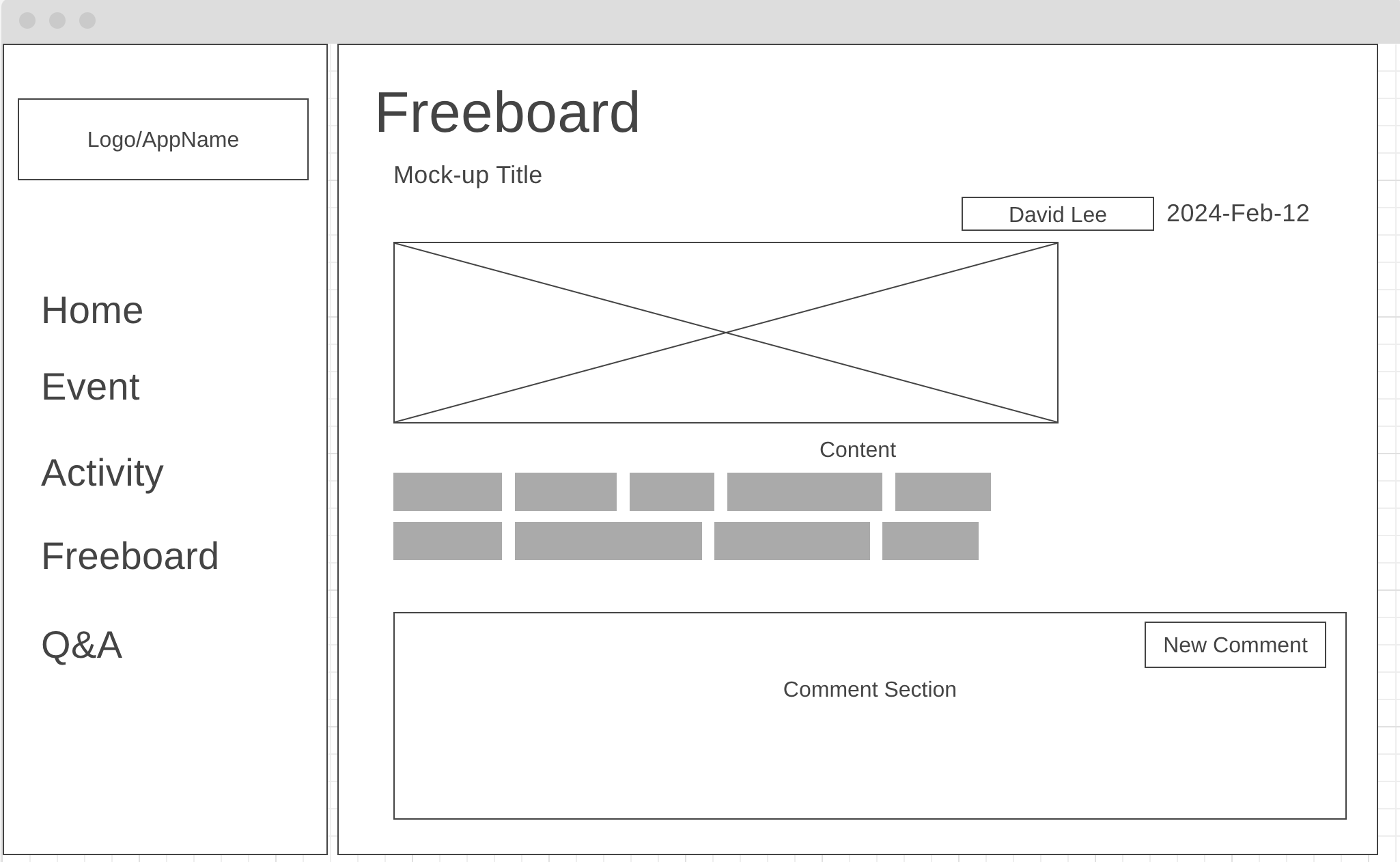
2) Freeboard – Create new post



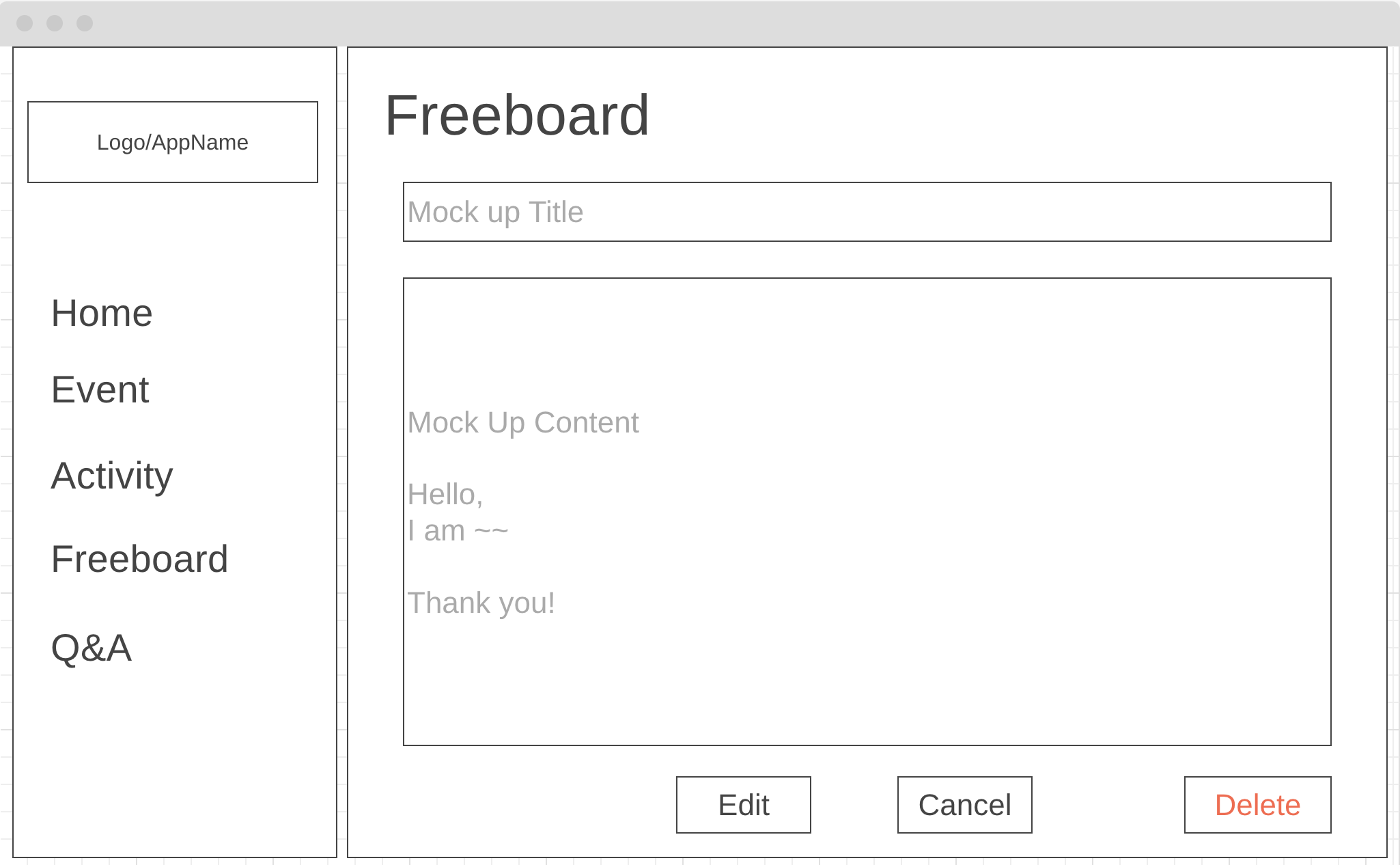
2) Ater post is created, the user will go to the post.



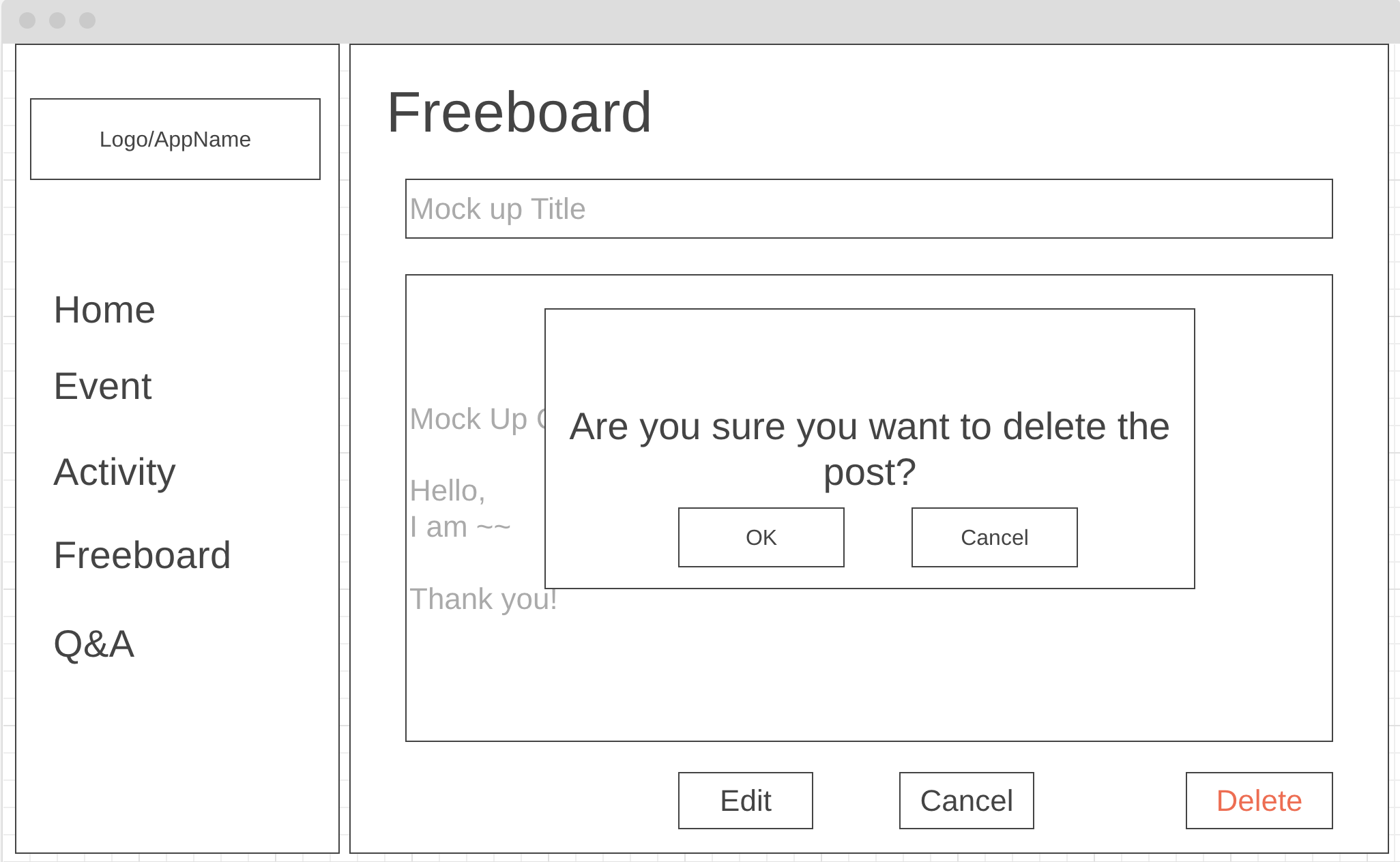
3) Freeboard – Read a post



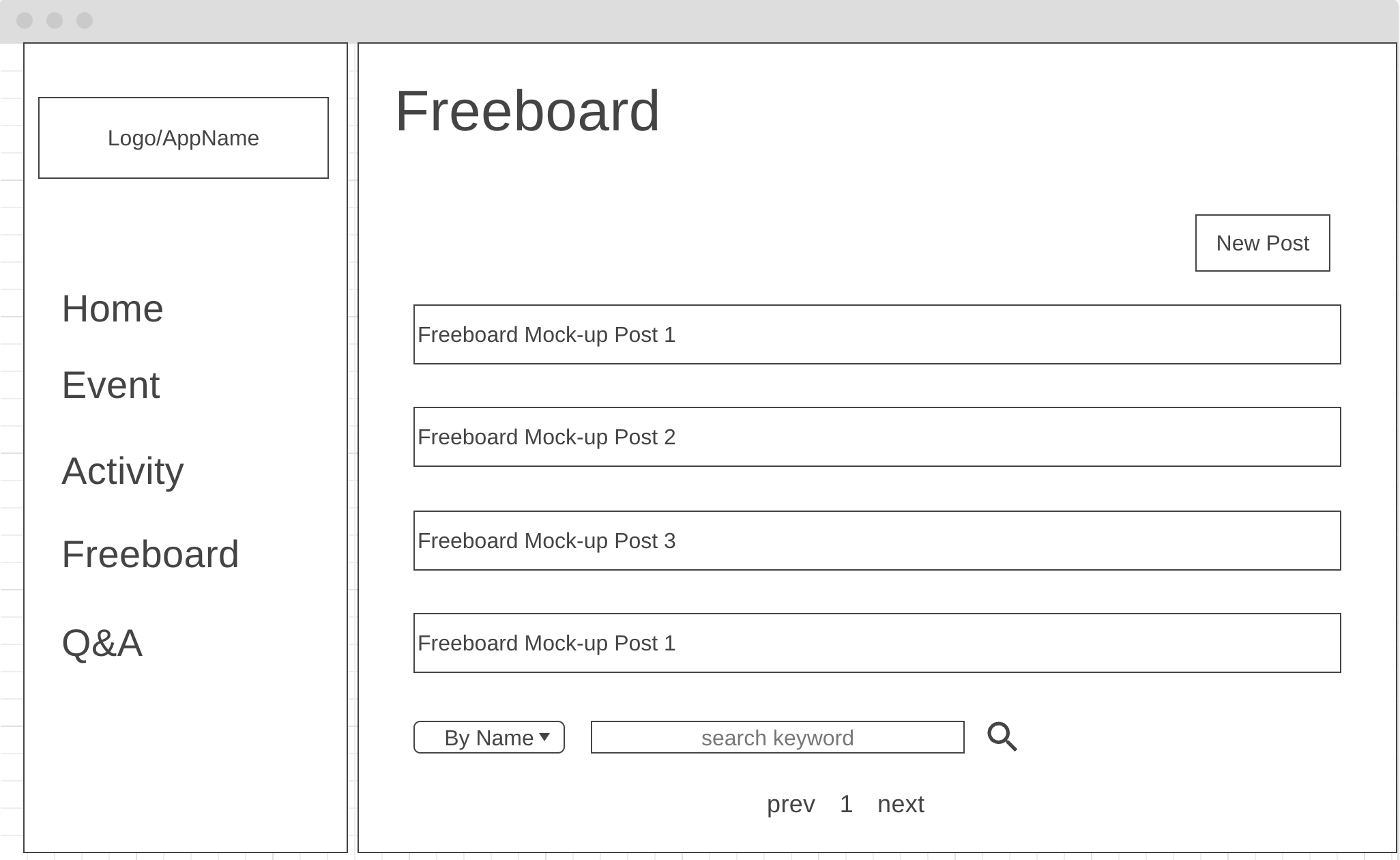
4) Freeboard – Update and Delete a post



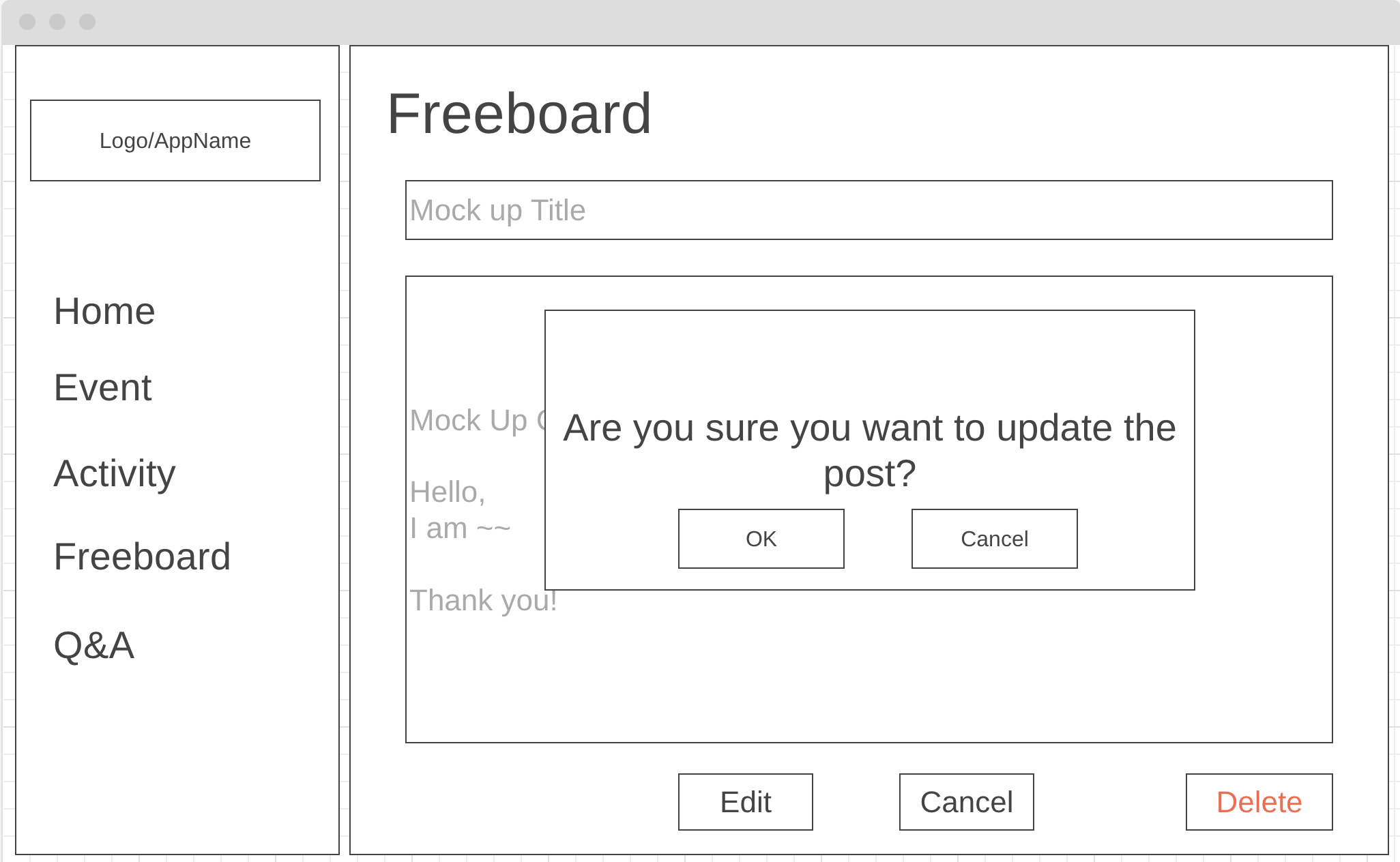
4) Freeboard – A) When user click the Delete button



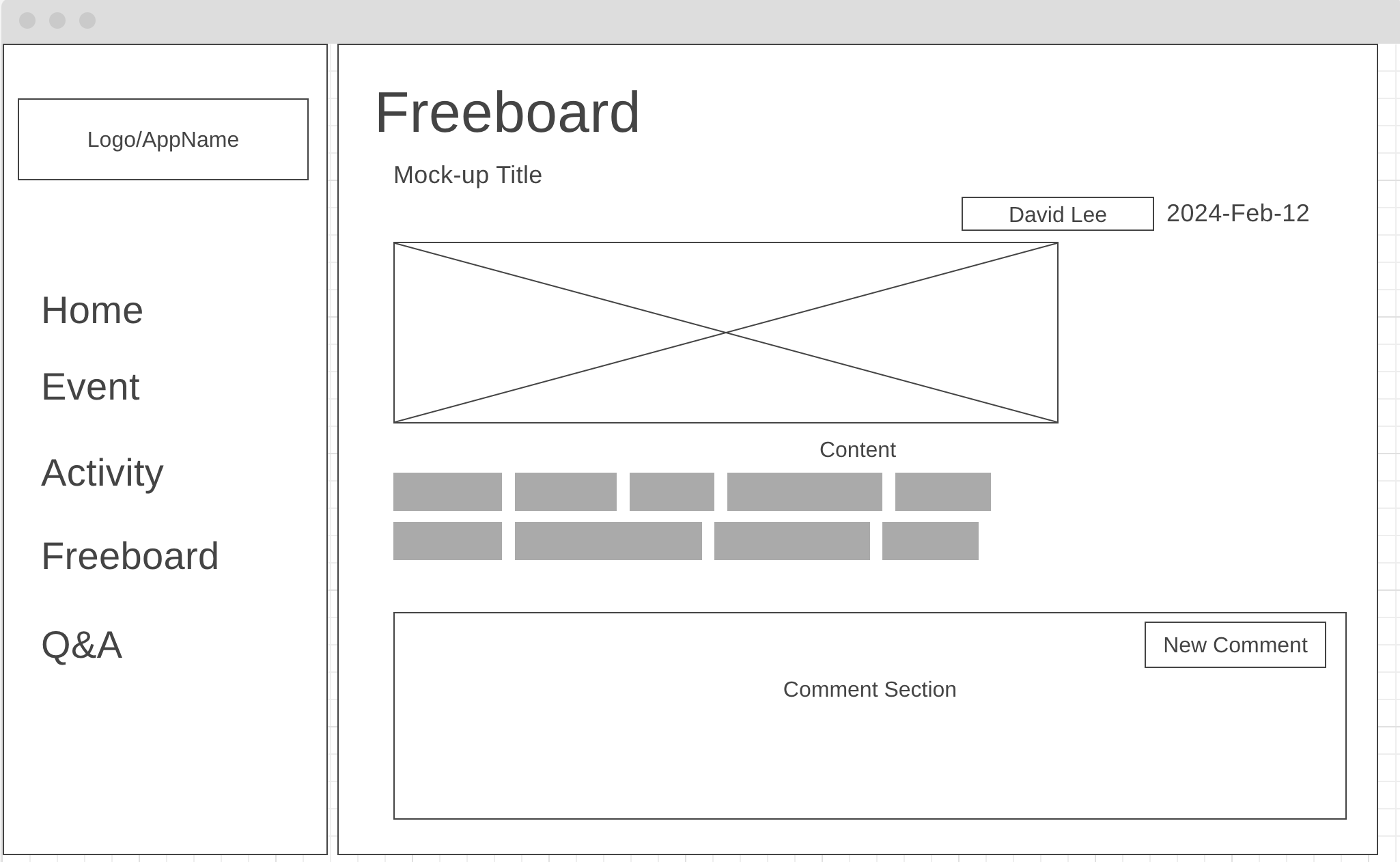
4) Freeboard – A) after deletion, go back to the landing page



4) Freeboard – B) When User click the Edit button

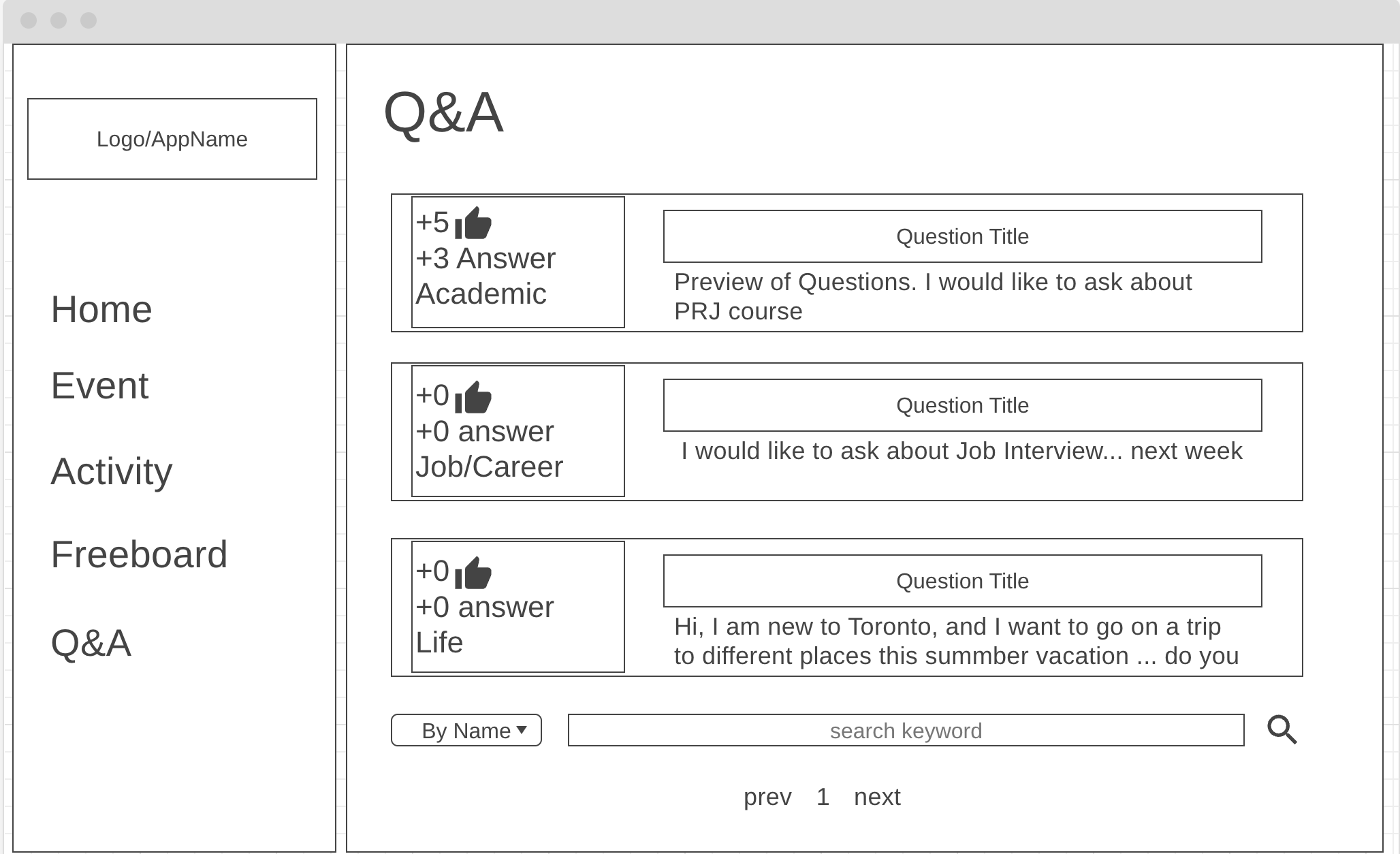


5) Freeboard – B) dislays the edited post

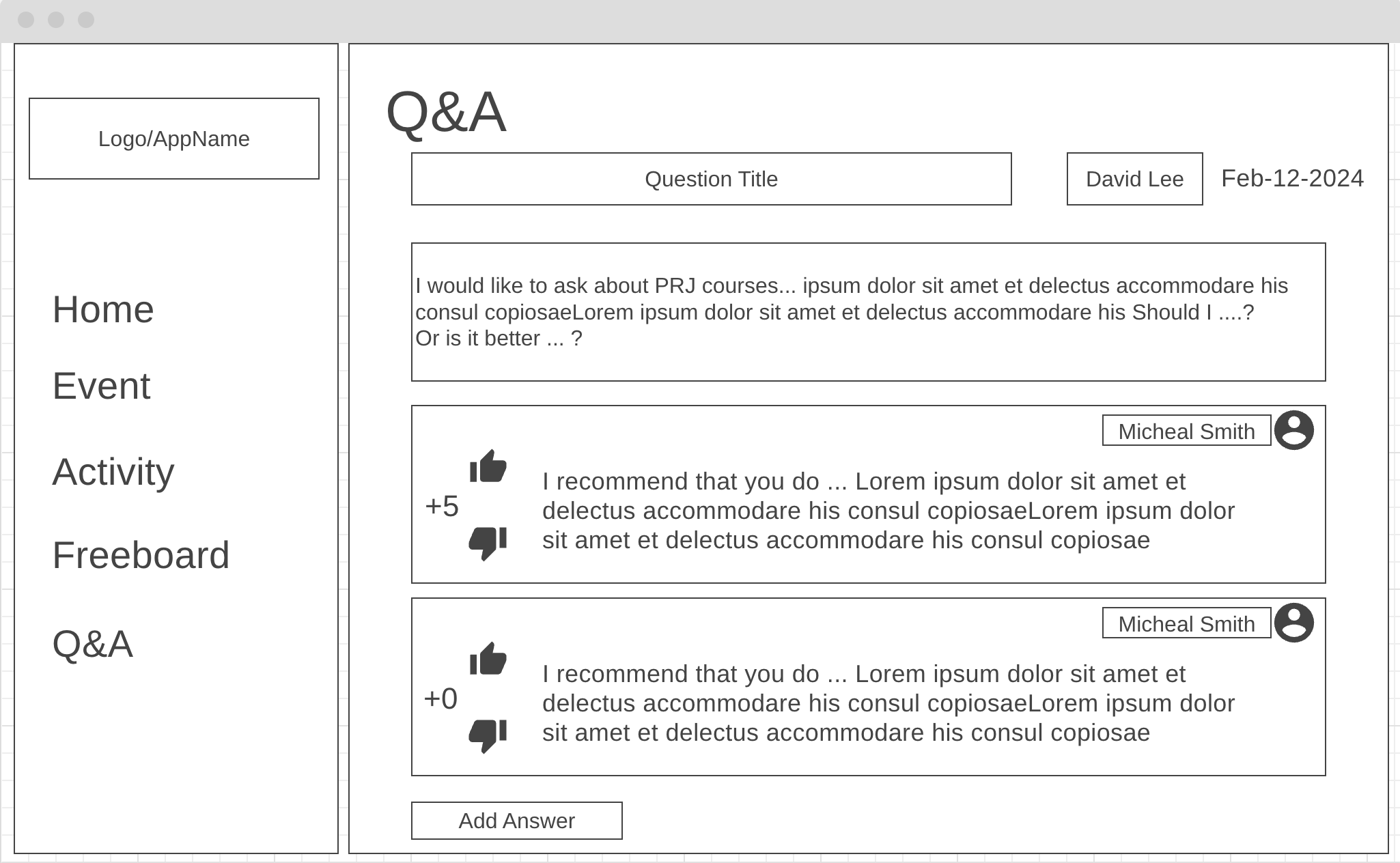


4. Q&A Hyunjin-Shin

1) Visit Q&A Main page (Assumption: user is already signed in) - Query Q&As



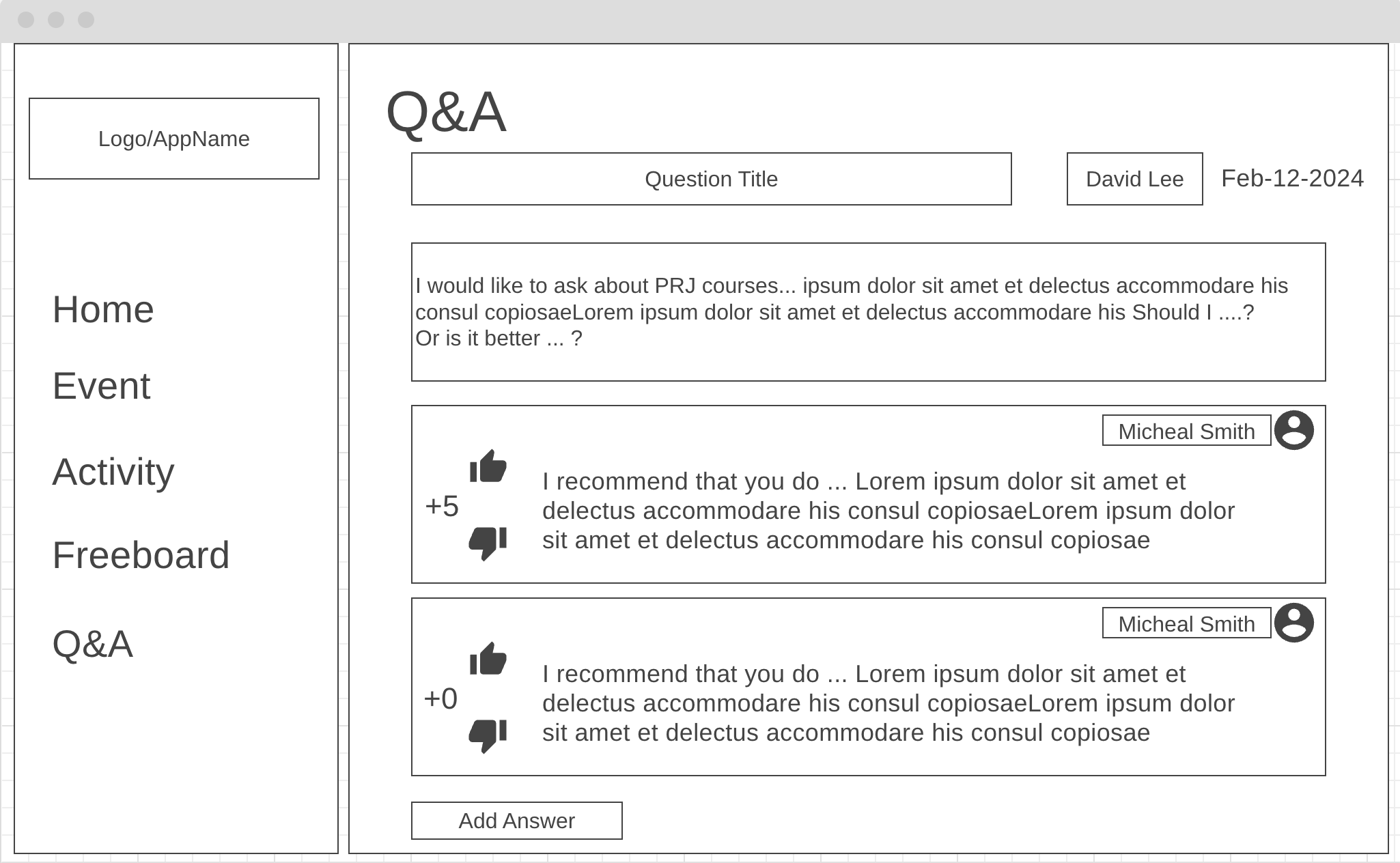
2) Q&A Read a question



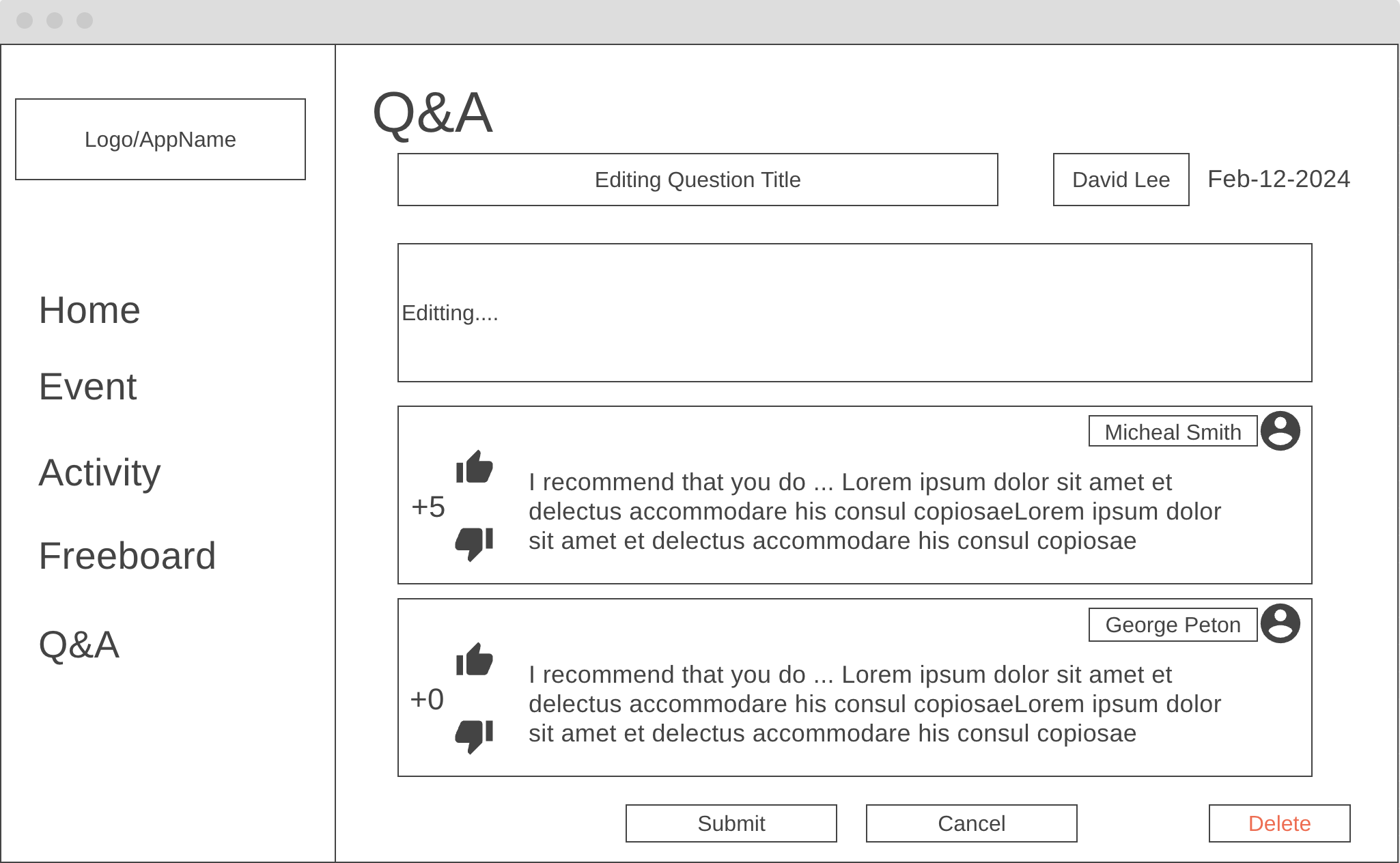
3) Q&A Add Answer



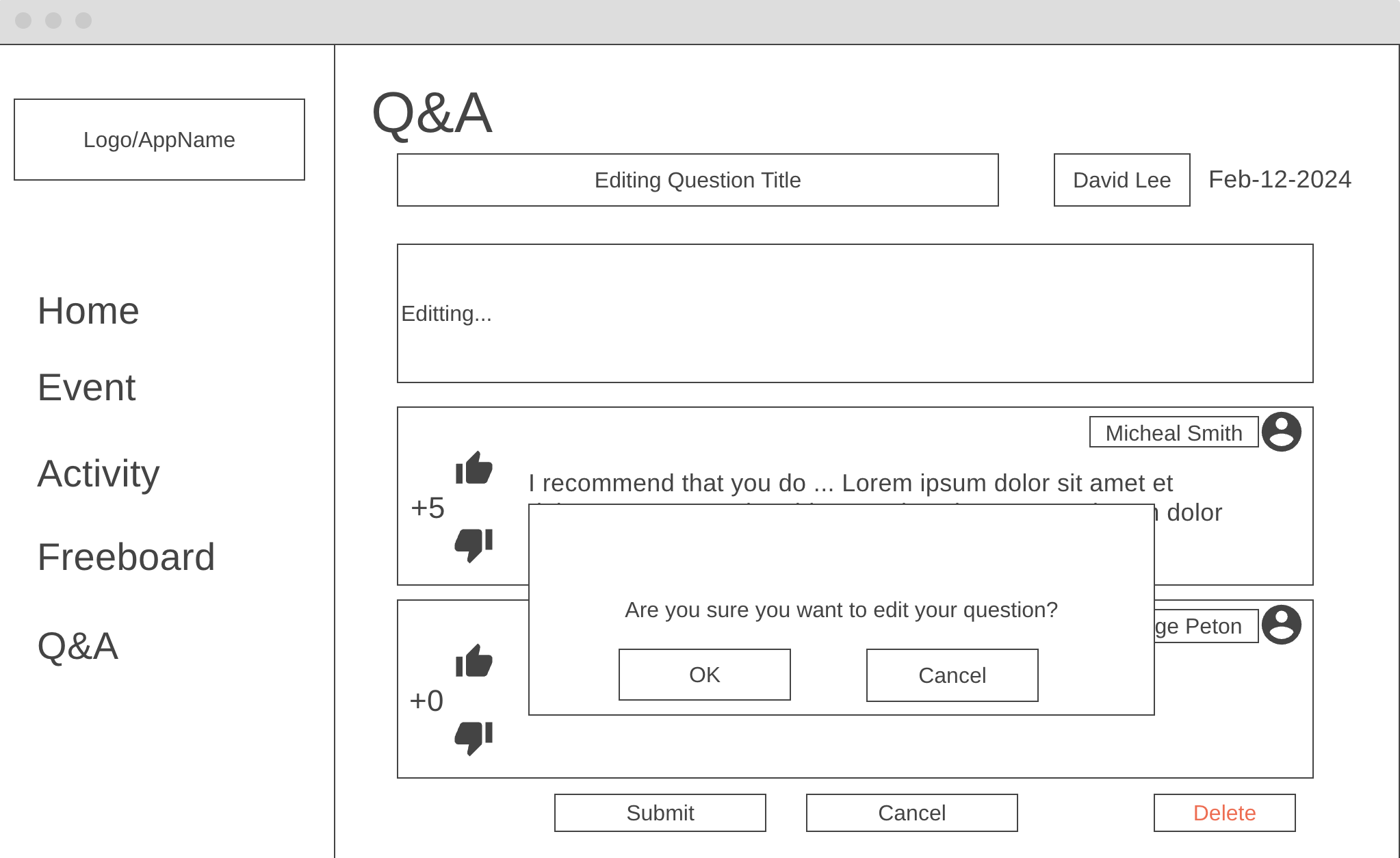
3) Q&A Add Answer – after adding answer, the added answer is displayed



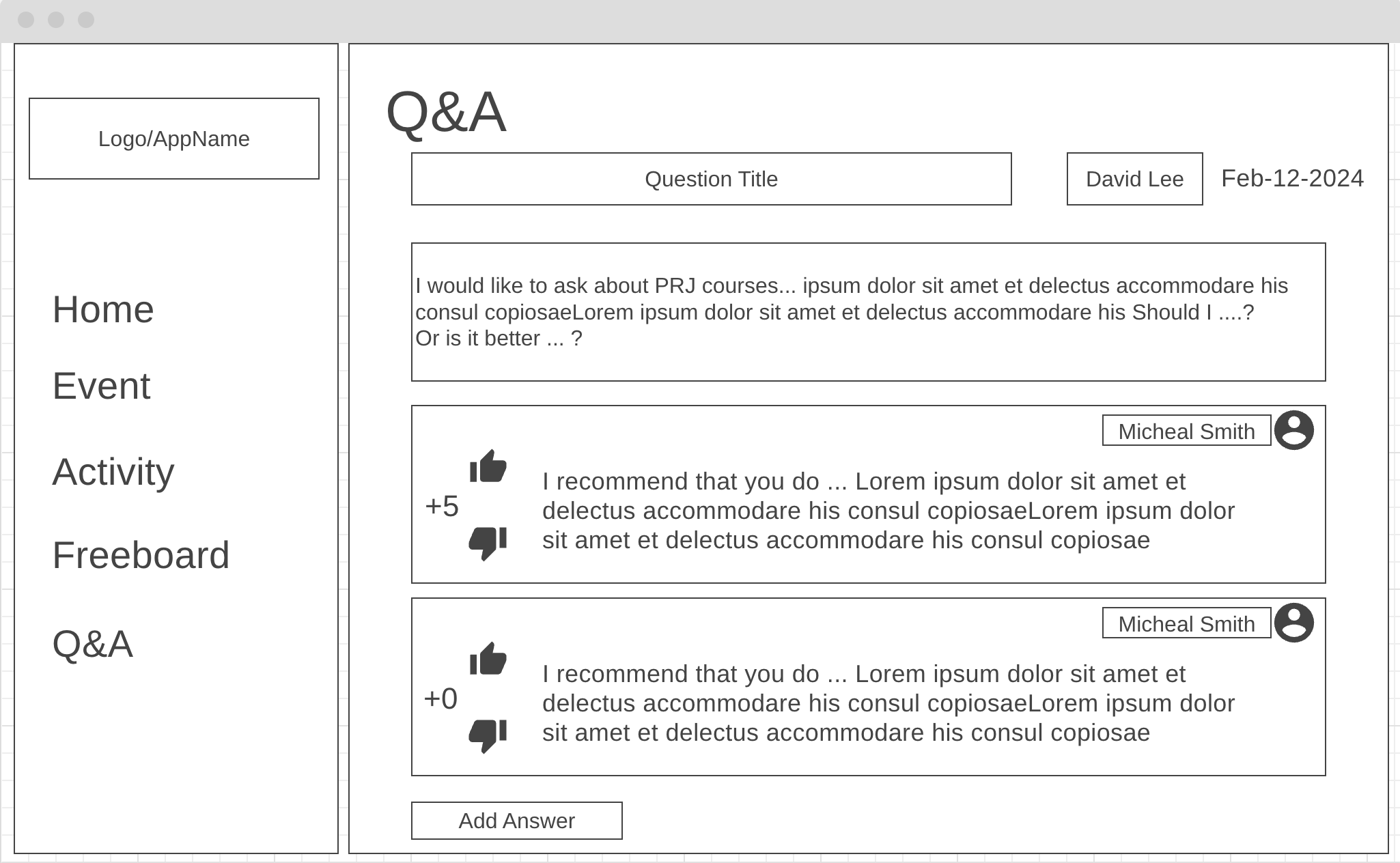
4) Q&A Edit Question – when user choose to edit the question.



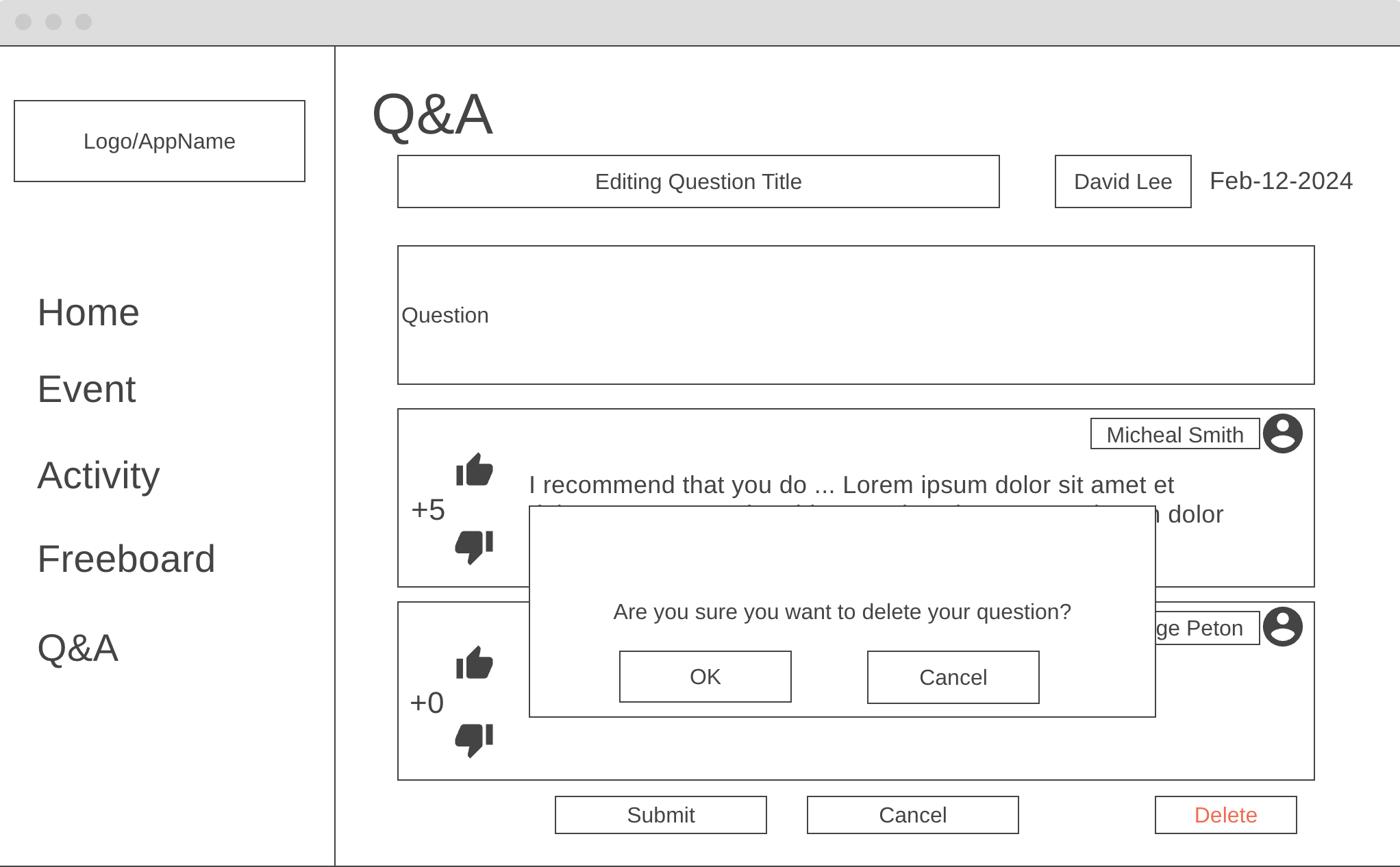
4) Q&A Edit Question –A) if user click submit button, then dialog box will pop up for confirmation.



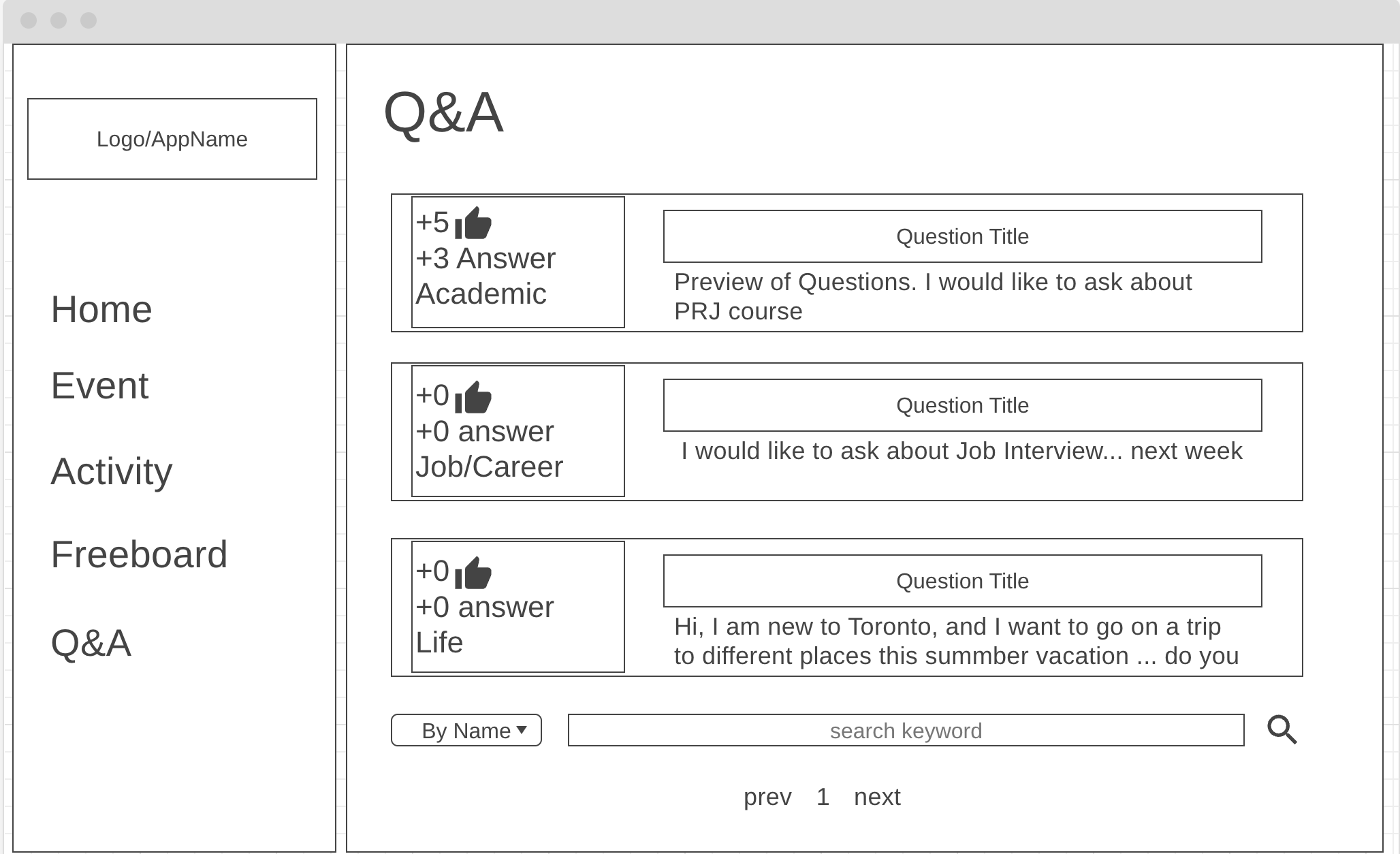
4) Q&A Edit Question –A) After it is updated, the edited Q&A will be displayed



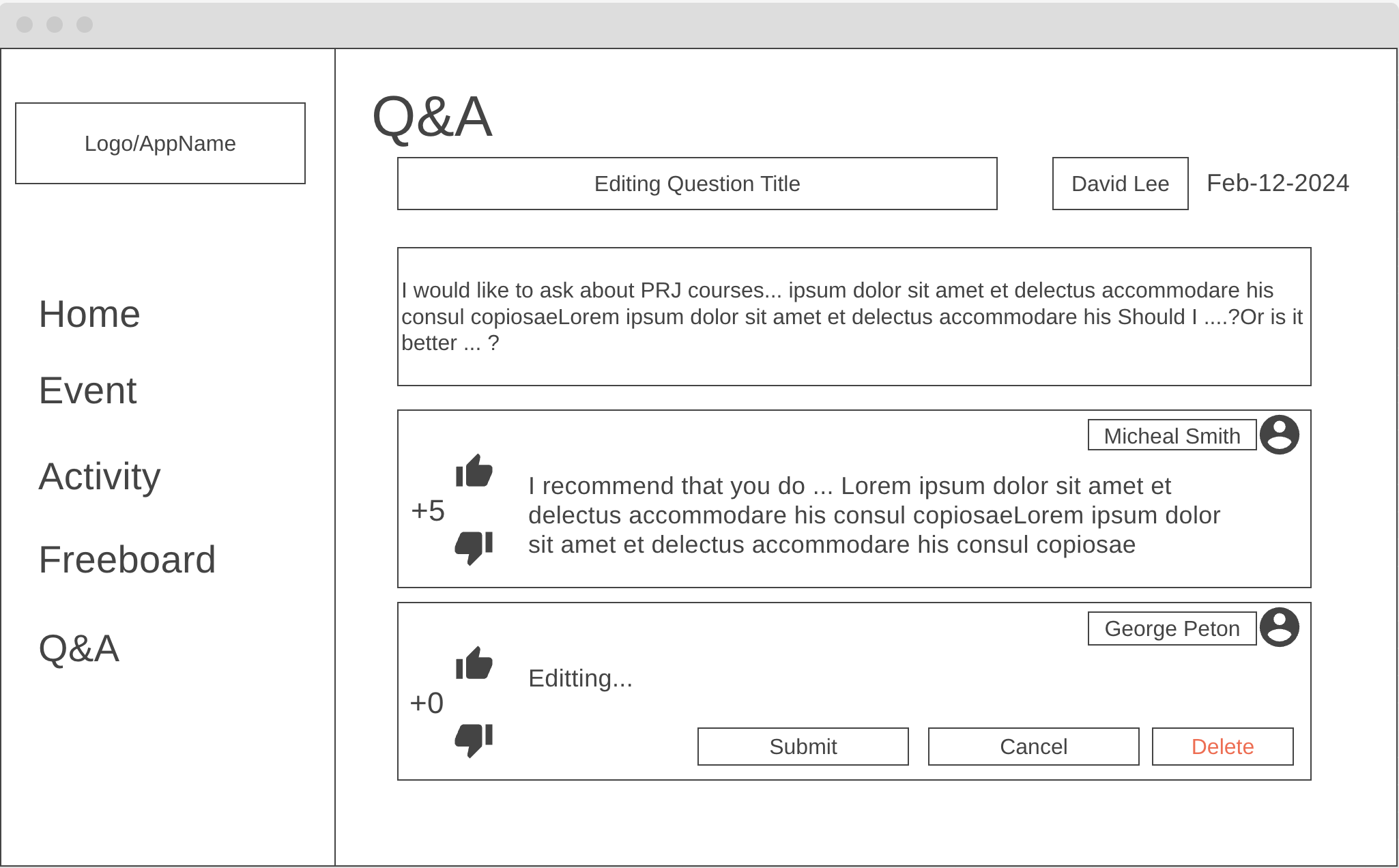
4) Q&A Edit Question –B) If user click delete button, it will show alert box.



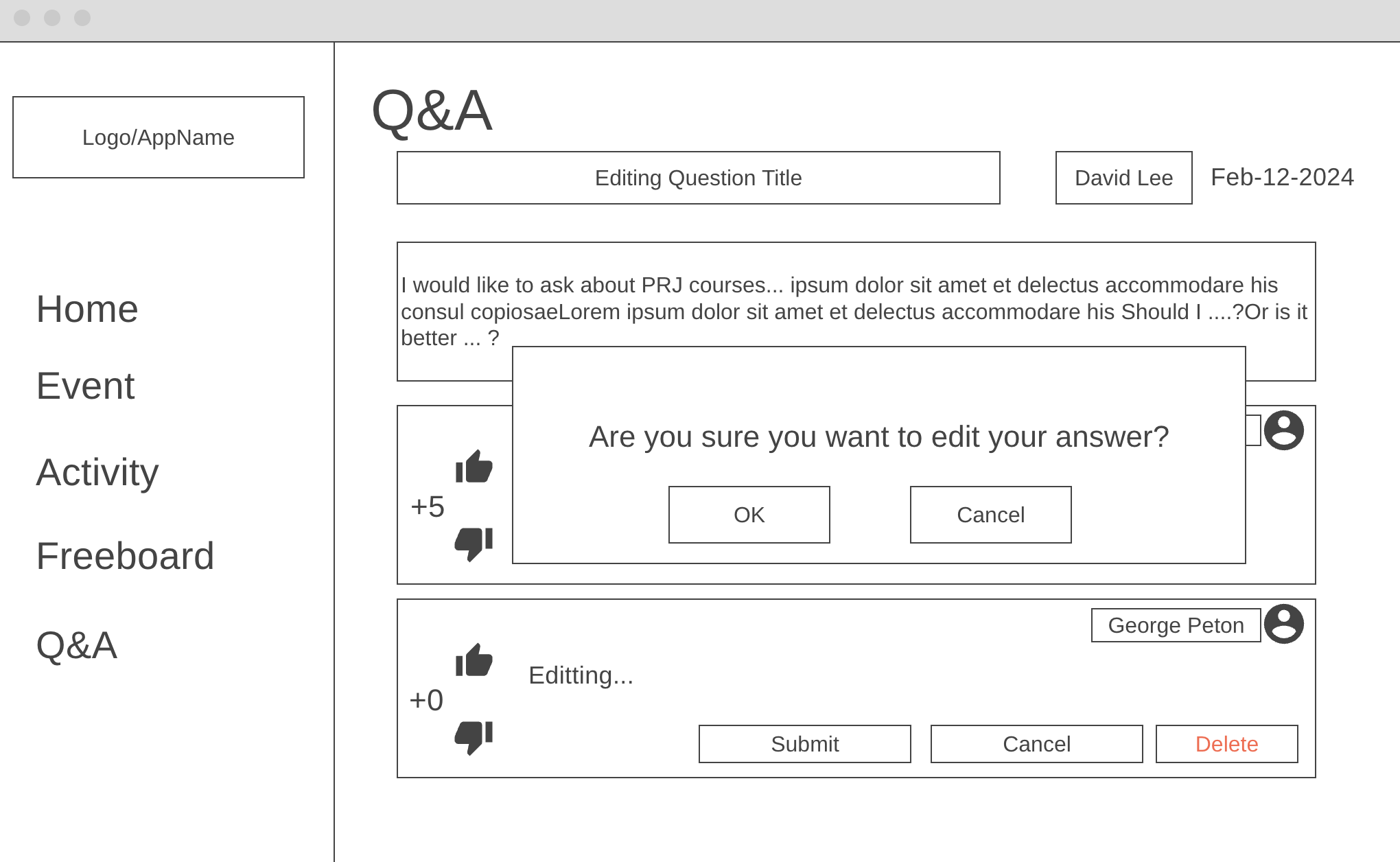
4) Q&A Edit Question –B) After deleted, it will go back to the landing page



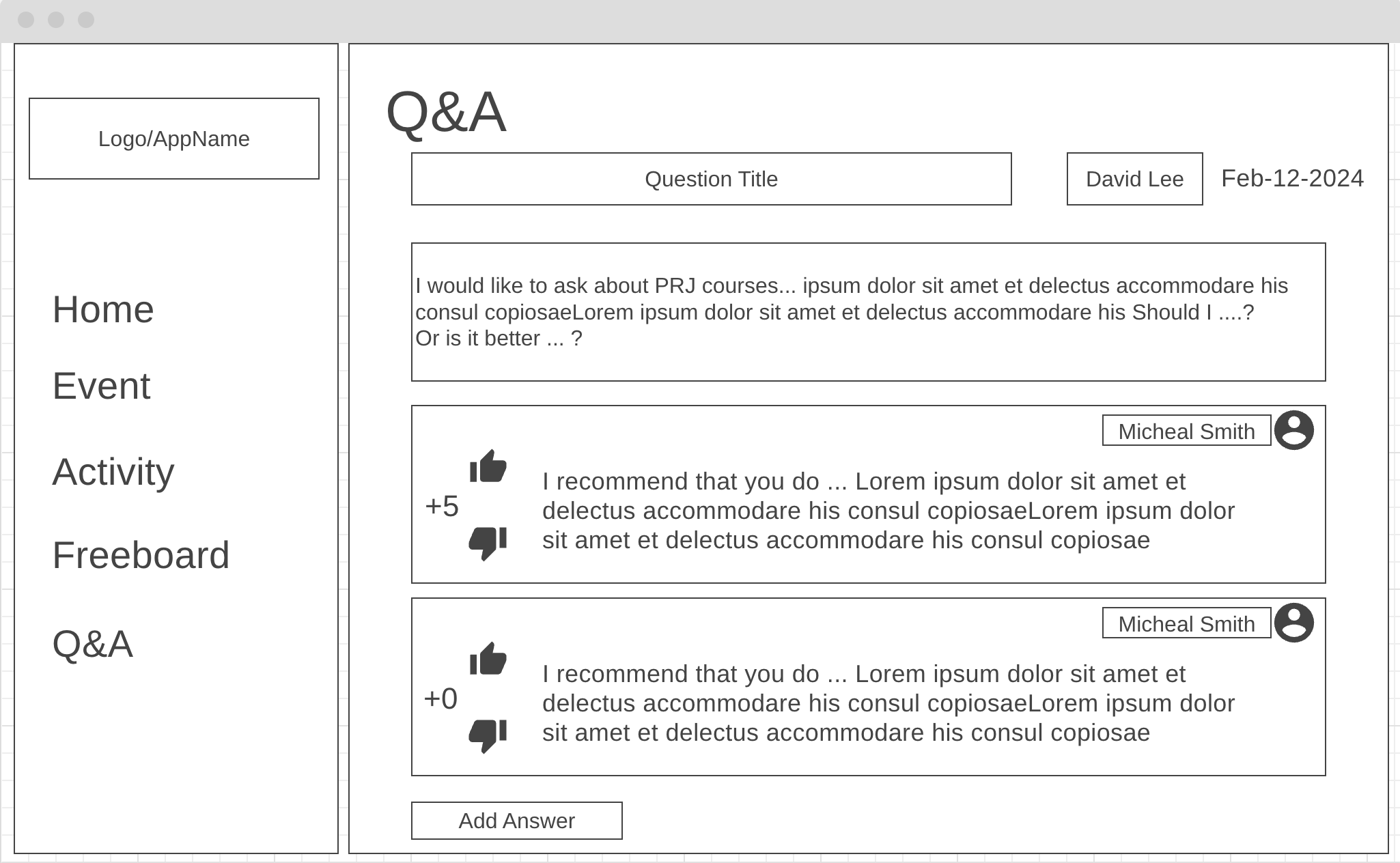
5) Q&A Edit Answer



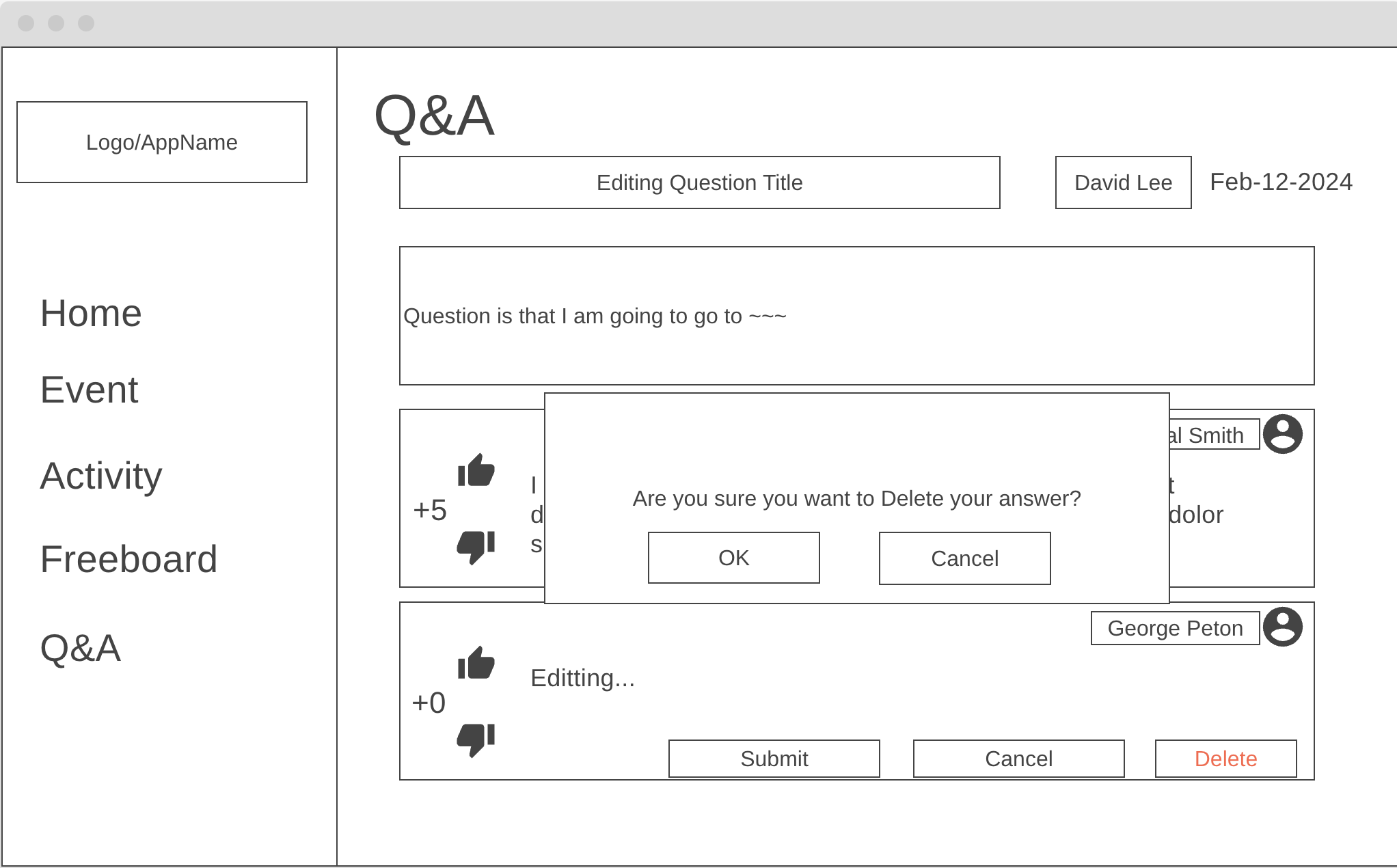
5) Q&A Edit Answer – A) when user click the submit button.



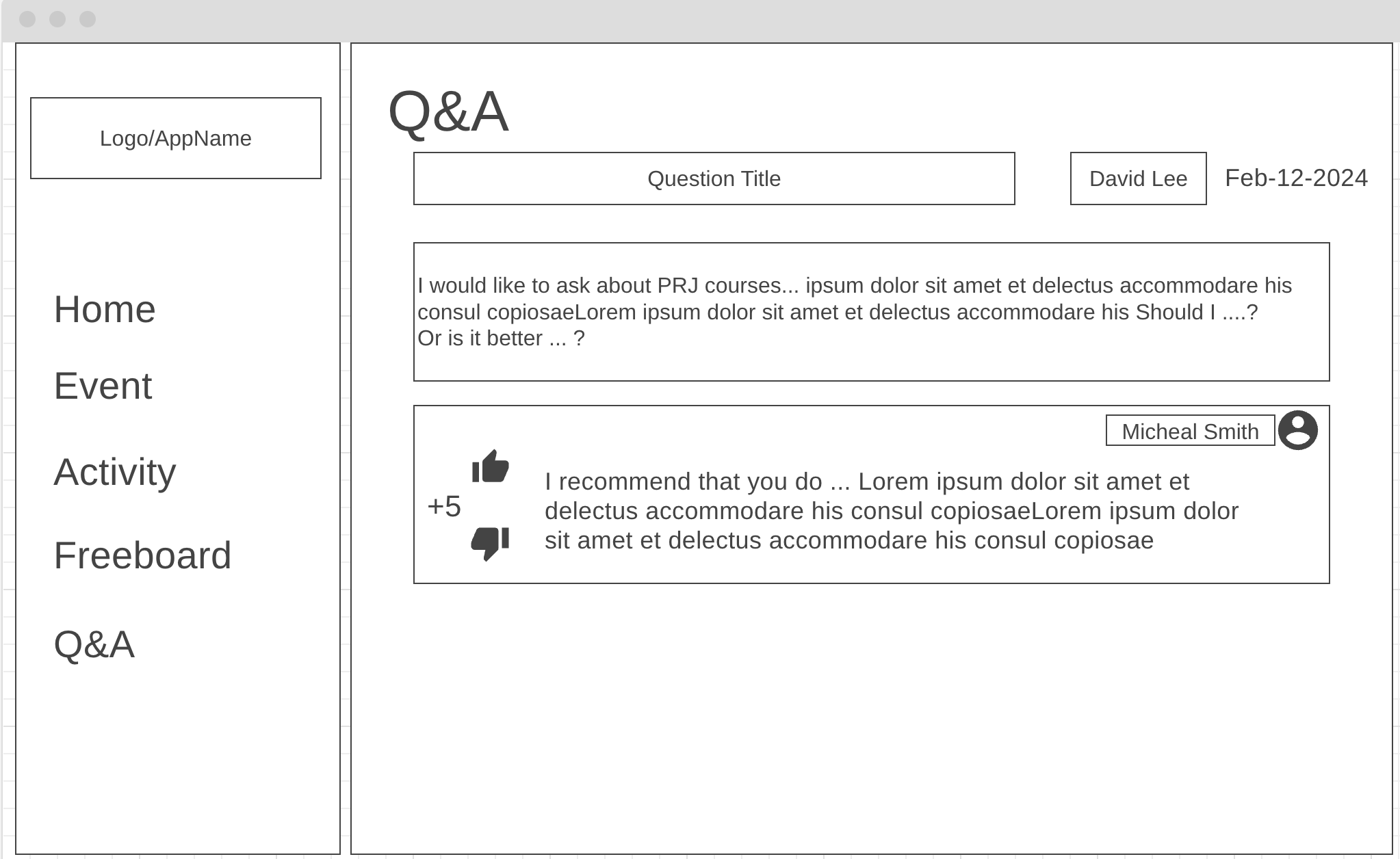
5) Q&A Edit Answer – A) if user click ok button, then it will show the Q&A post with edited answer.



5) Q&A Edit Answer – B) When user clicks the delete button, it will show a dialog box for confirmation.

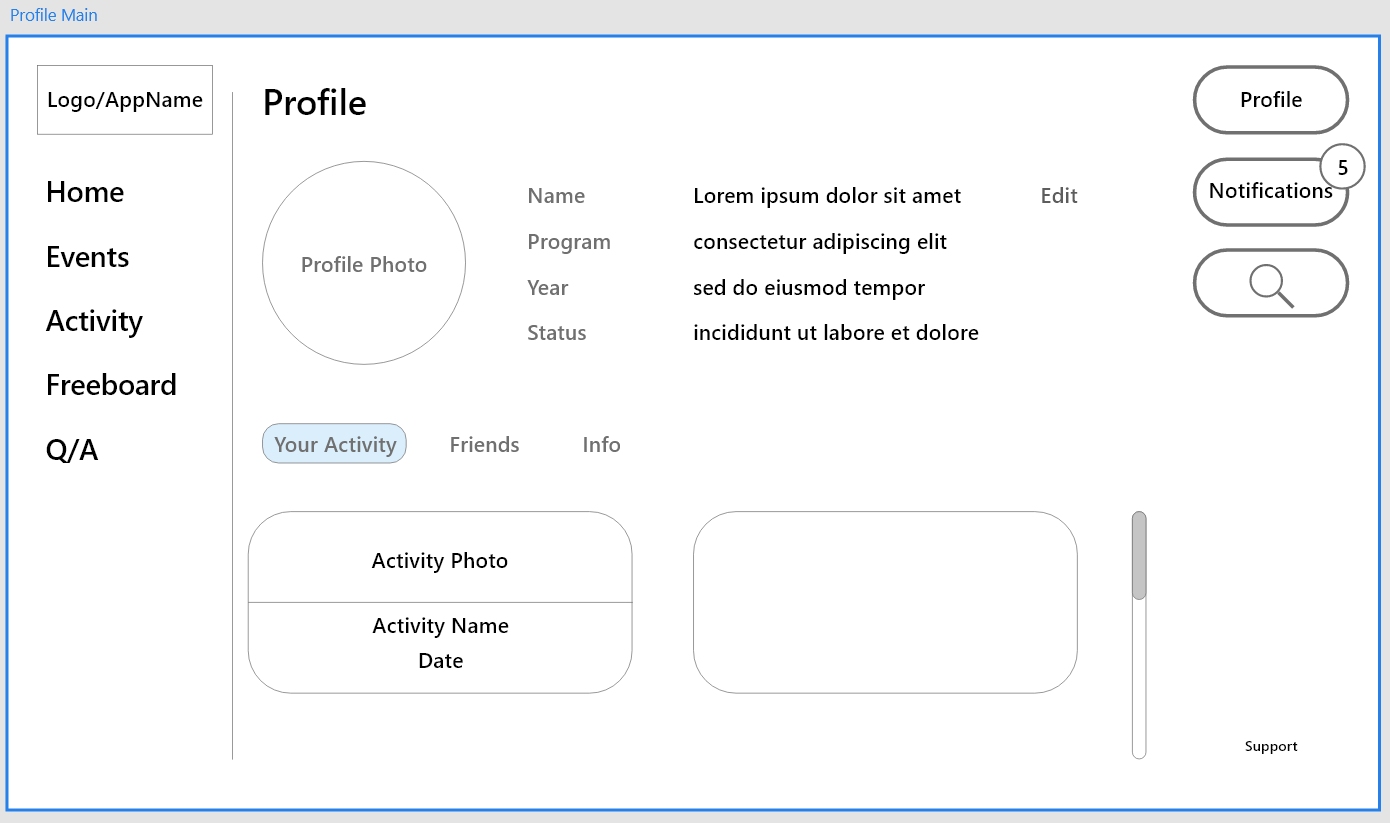


5) Q&A Edit Answer – B) After the answer is deleted, it will show the Q&A post without the deleted answer.

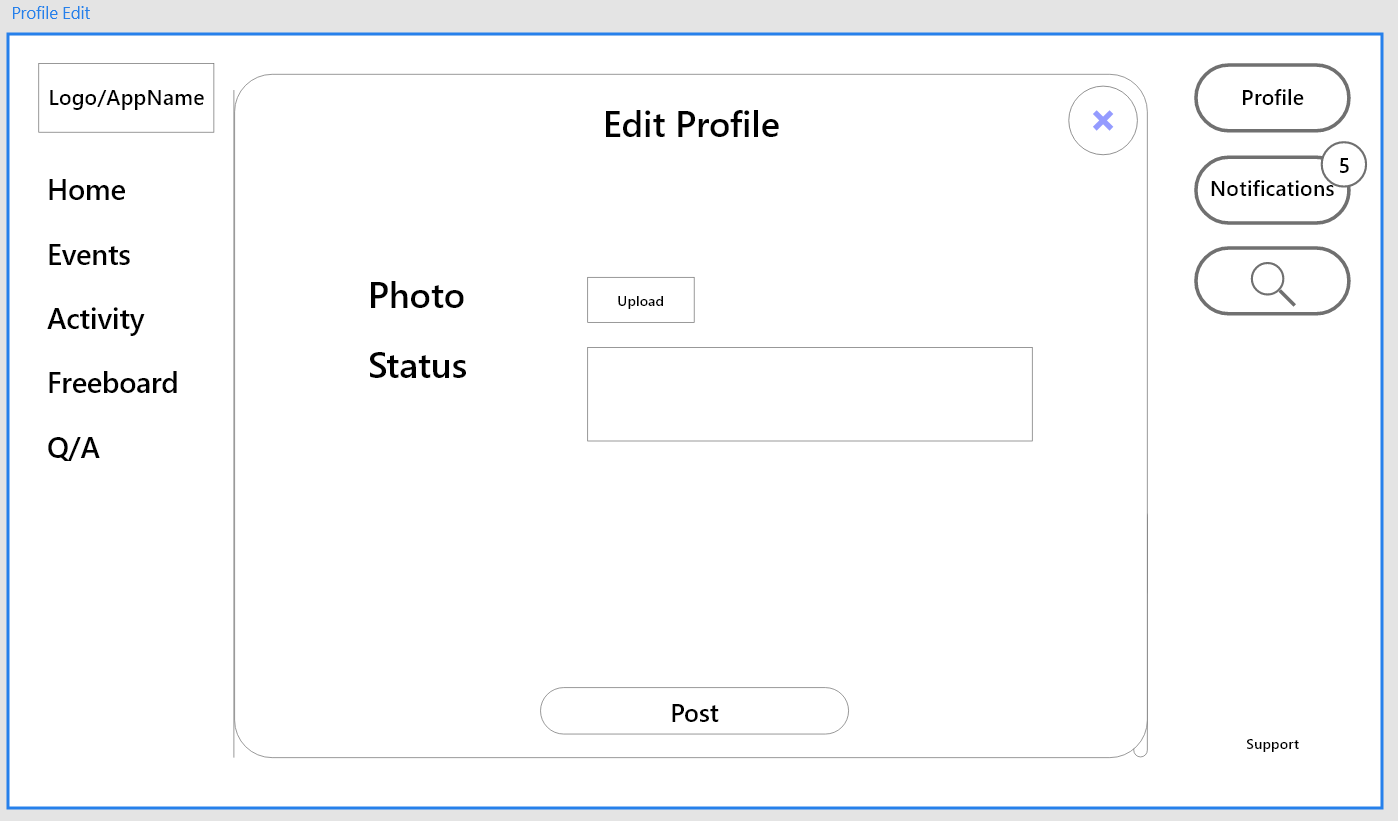


5. User Profile (personal) -> 5.5 Notifications ( a bit a lot) - Juhan

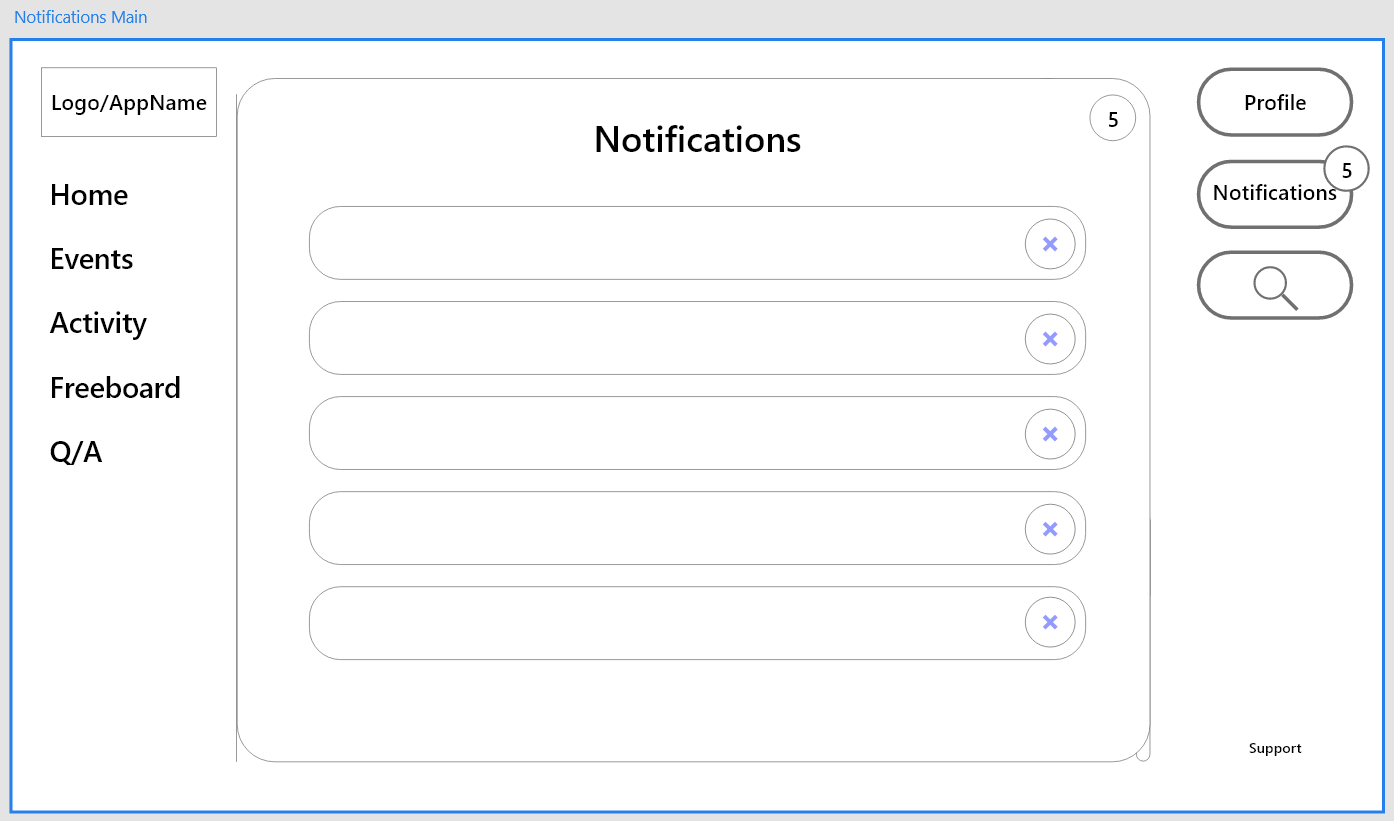
1) Profile Main



2) Profile Edit



3) Notification Main



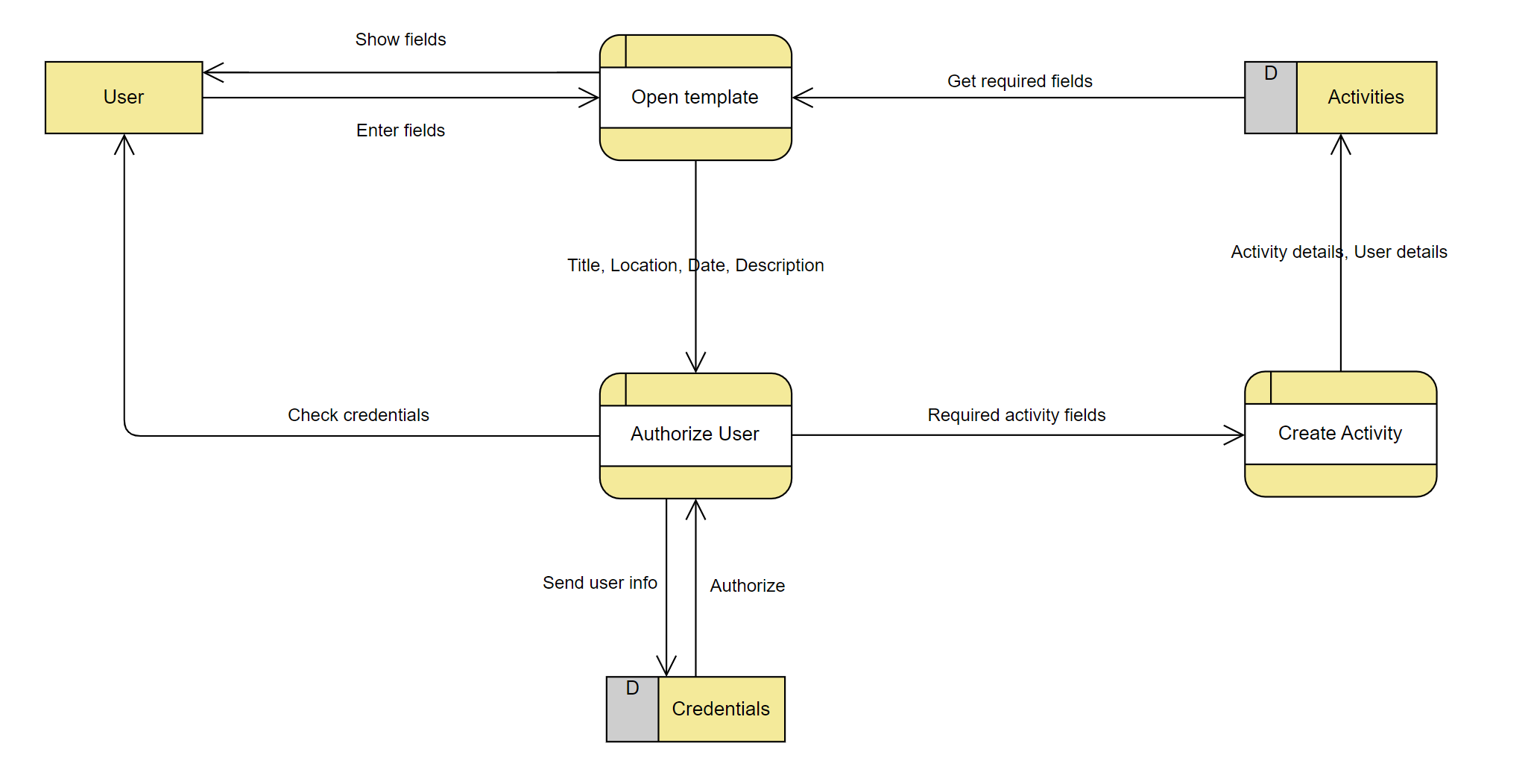
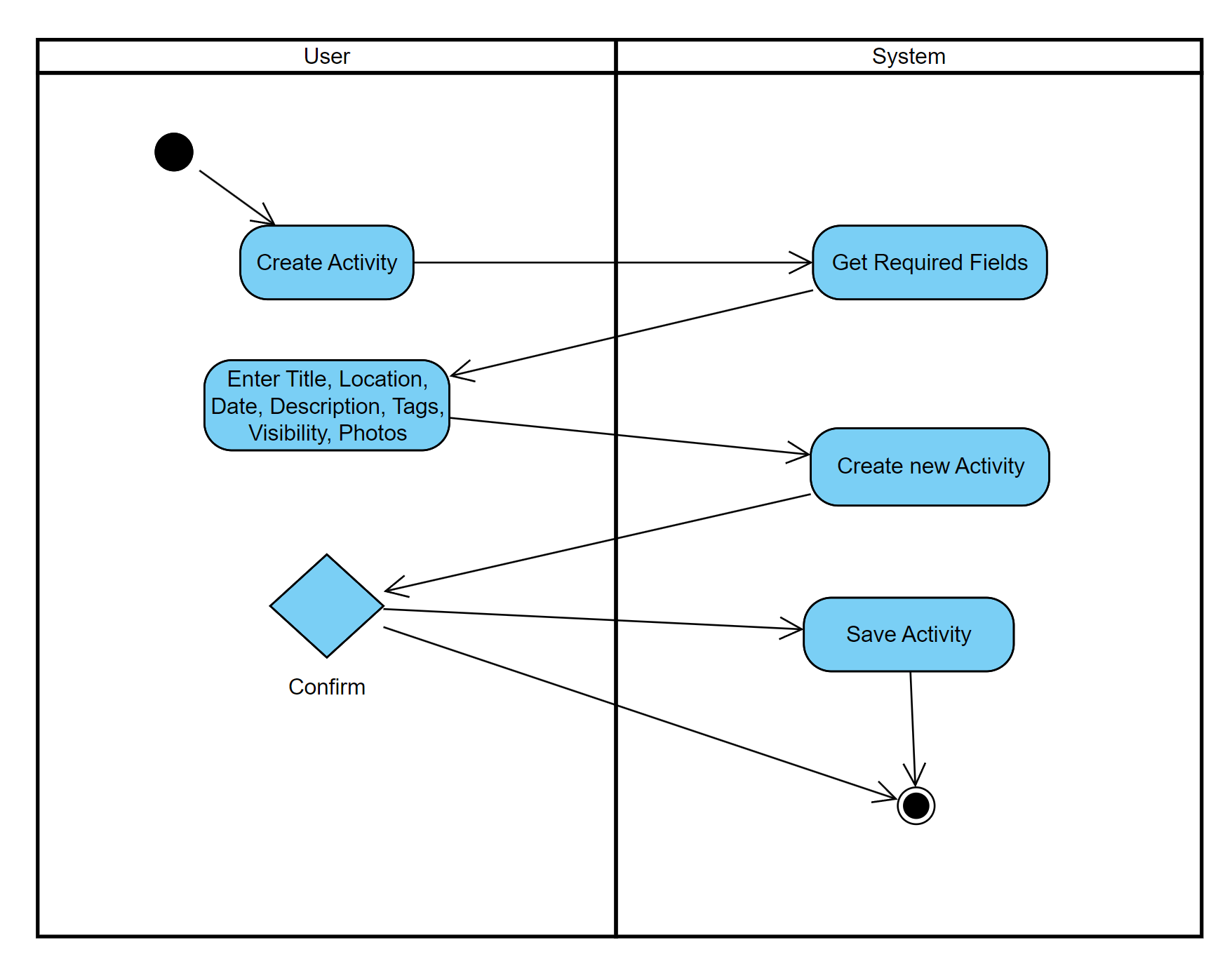
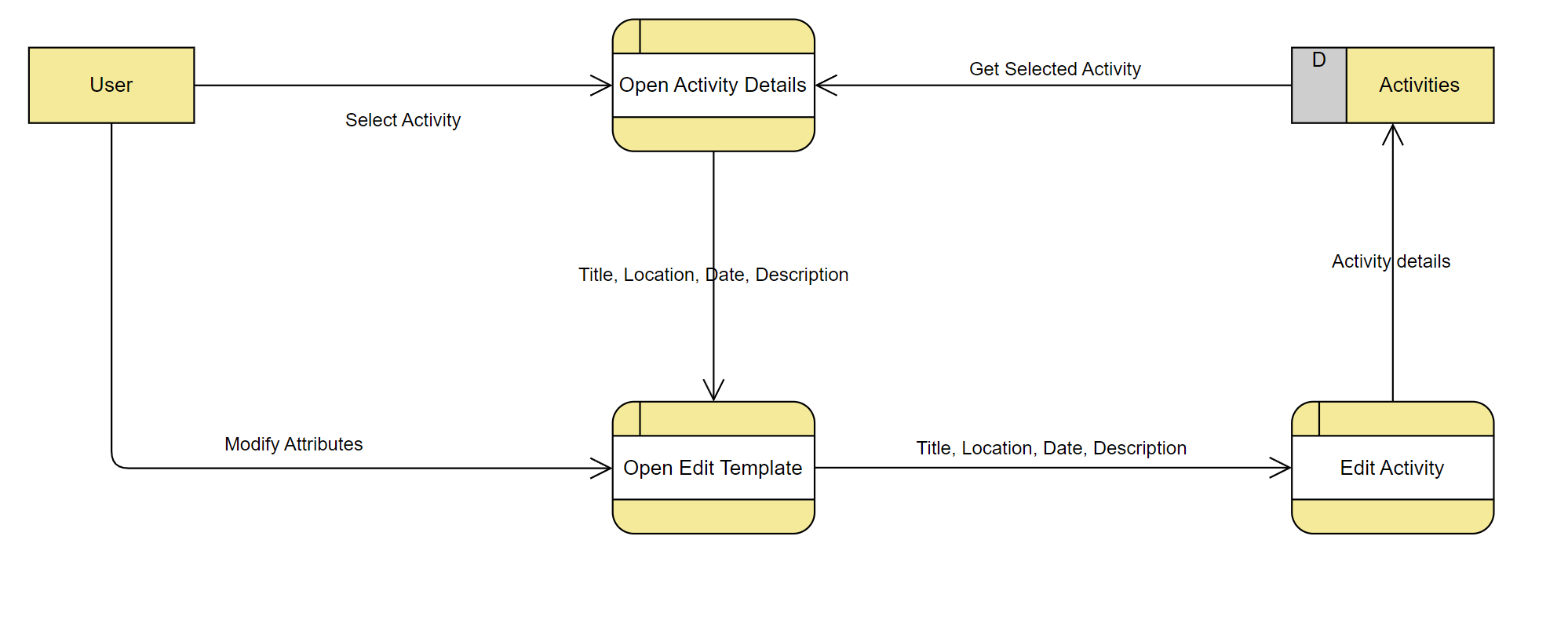
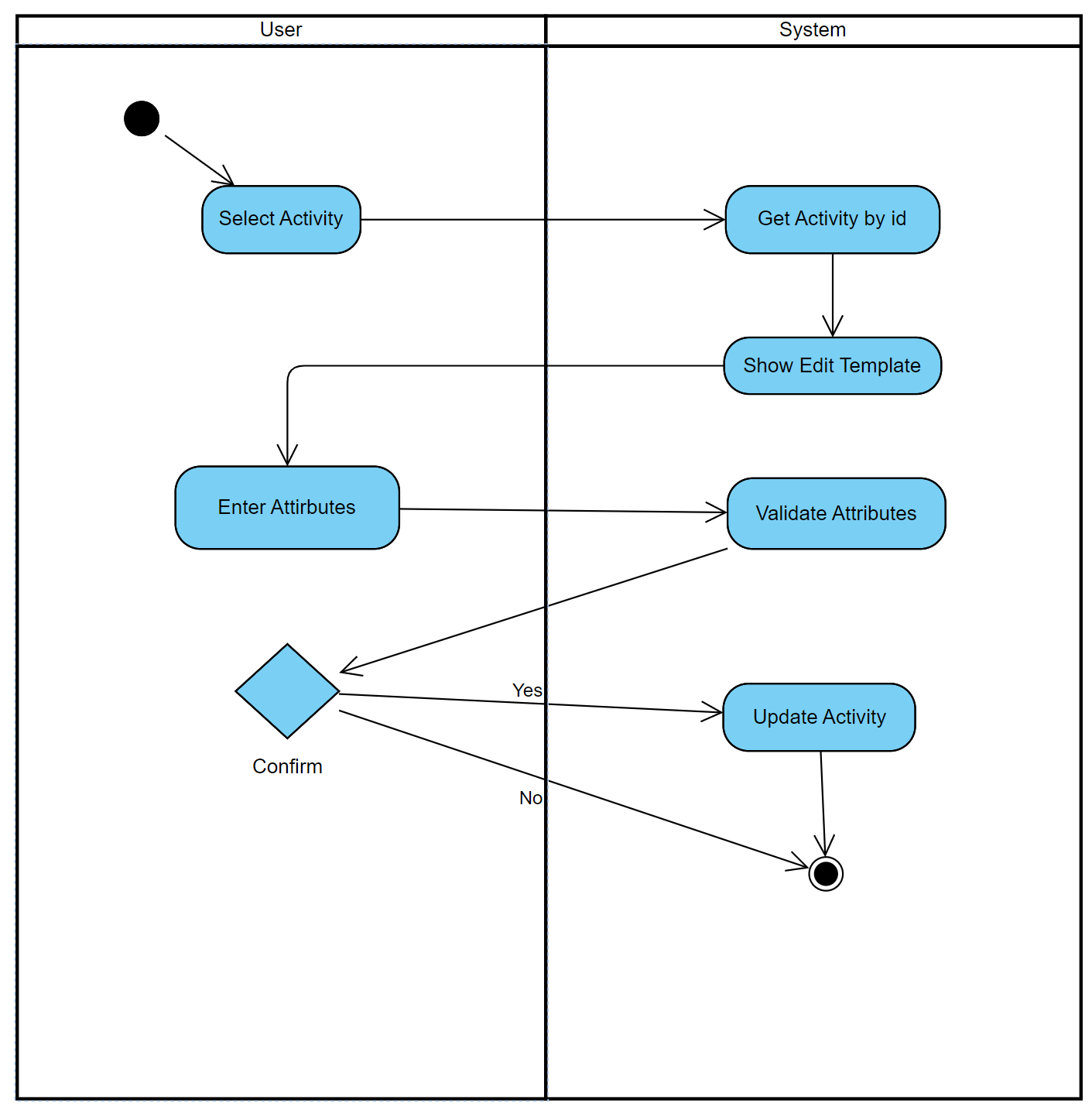
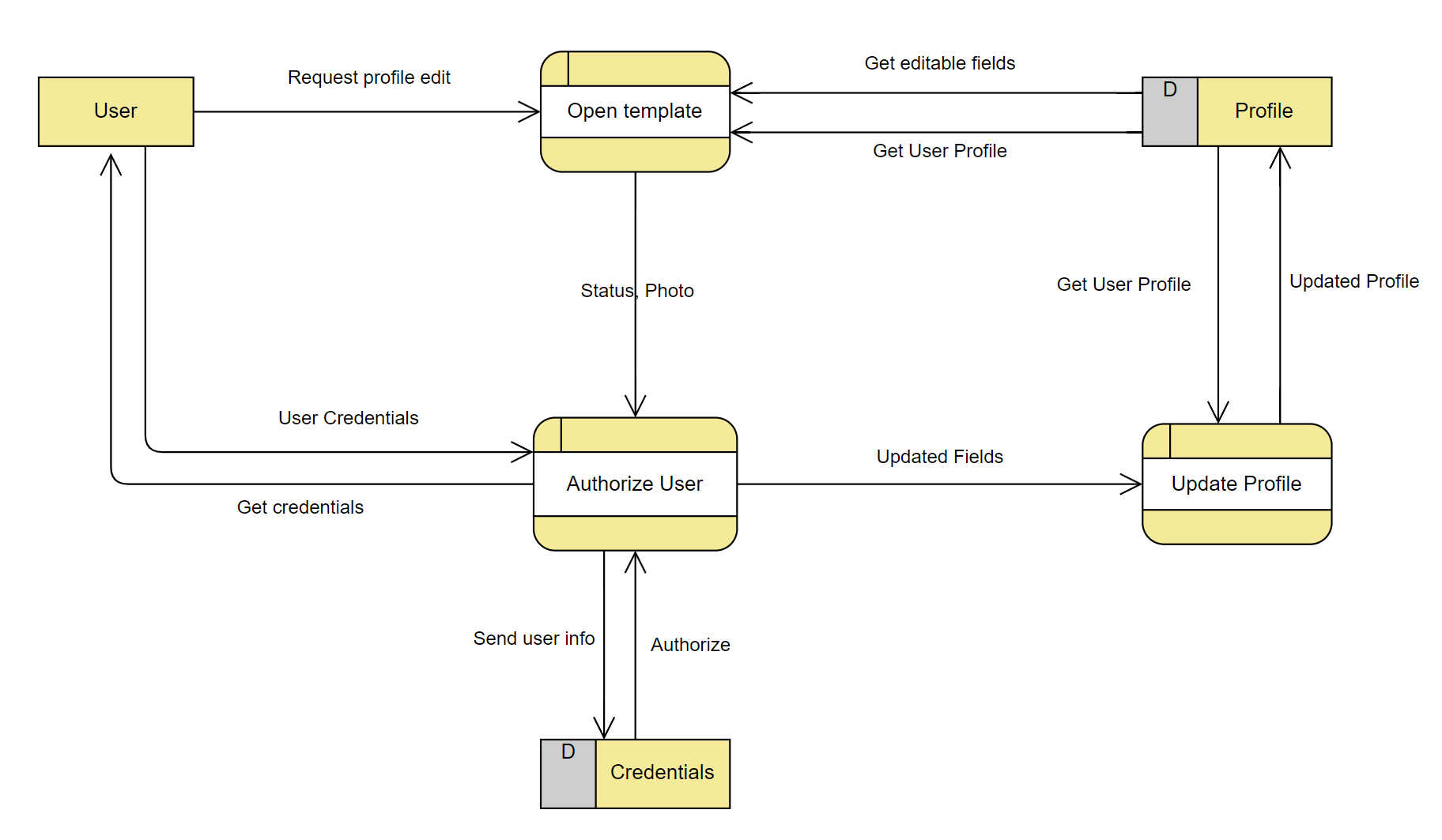
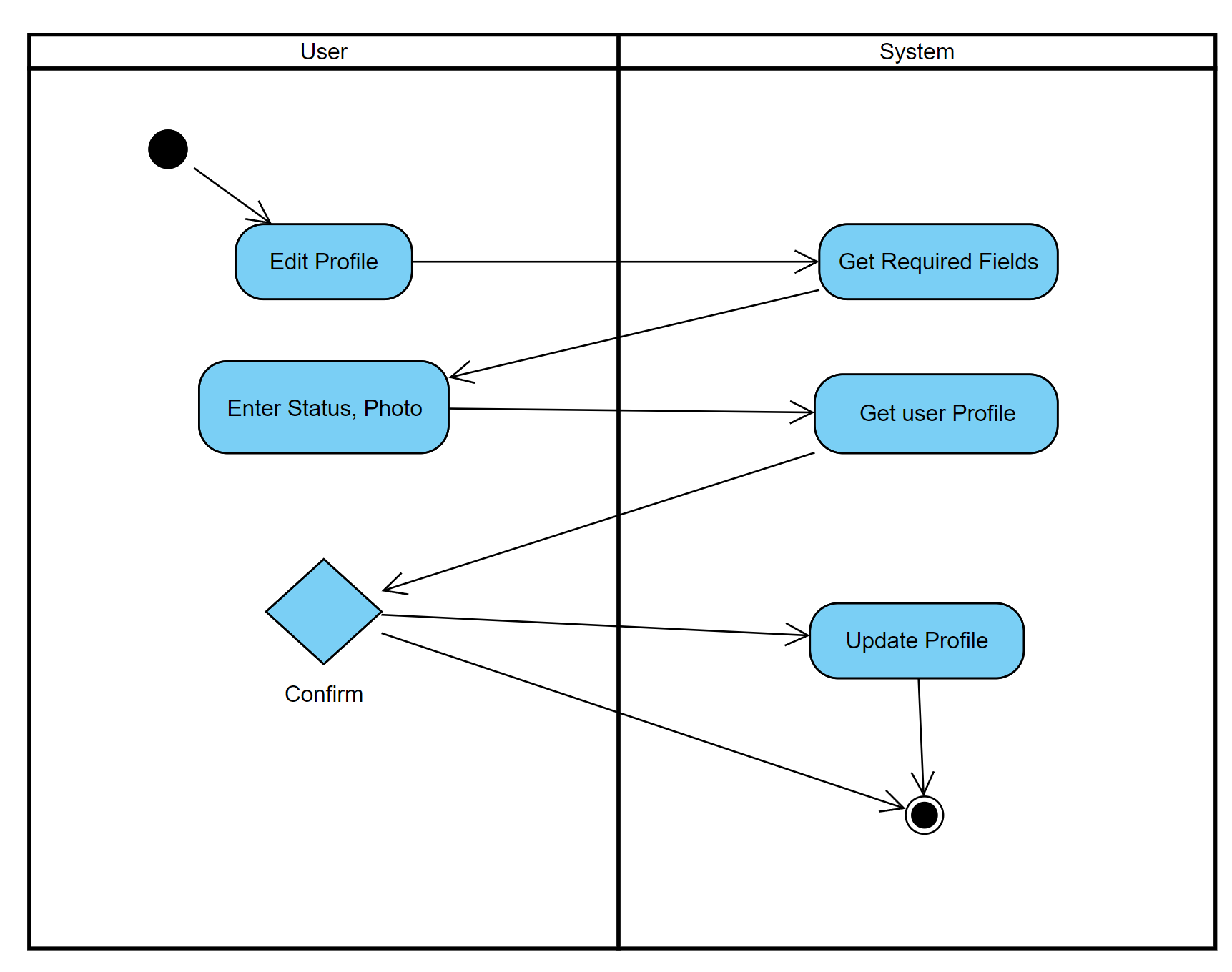
6. Event Notice (like announcement) - Ke An Lo

7. User Authentication and Authorization (Sign in, Sing- up) - Ke An Lo

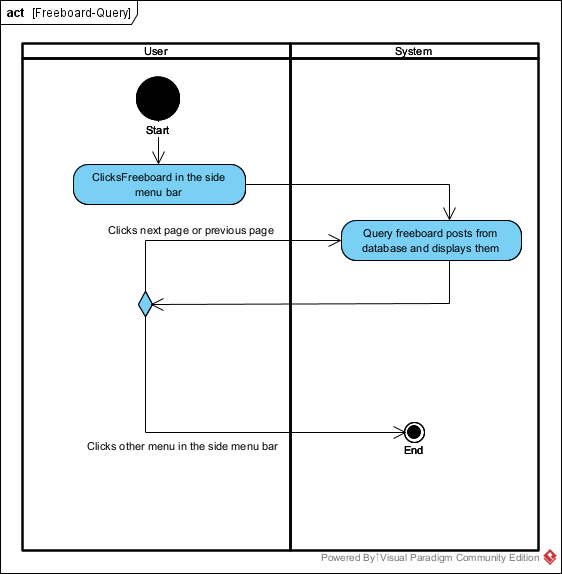
# Process and Data Modeling

## **3.1 UML/DFD Modeling and Data Modeling**

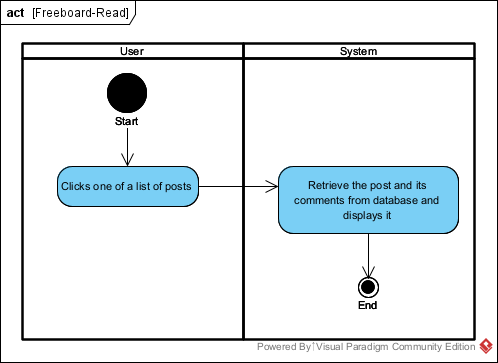
### Activity Diagrams and Data Flow diagram

* Activity Management
  + Create Activity
    - Data Flow Diagram
    - 
    - Activity Diagram
    - 
  + Modify Activity
    - Data Flow Diagram
    - 
    - Activity Diagram
    - 
* Profile
  + Edit Profile
    - Data Flow Diagram
    - 
    - Activity Diagram
    - 
* Freeboard (Hyunjin Shin)

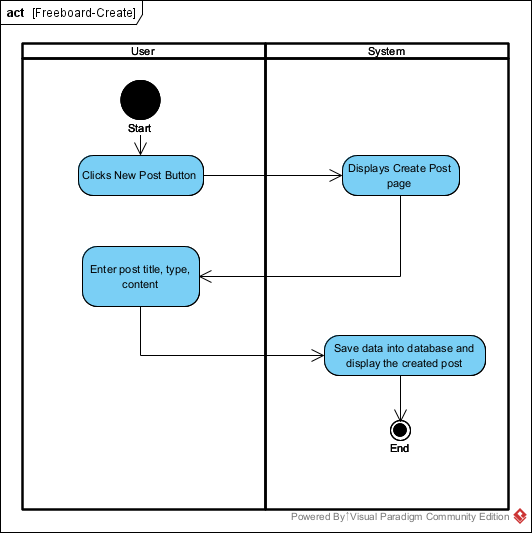
1. Query for landing page (Freeboard)



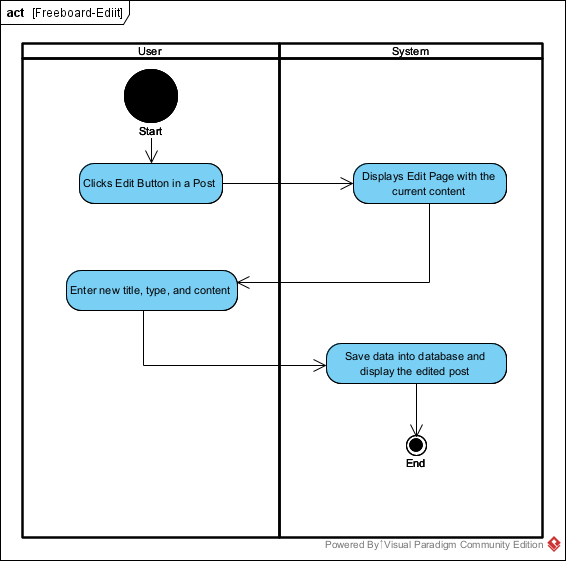
1. Read Post (Freeboard)



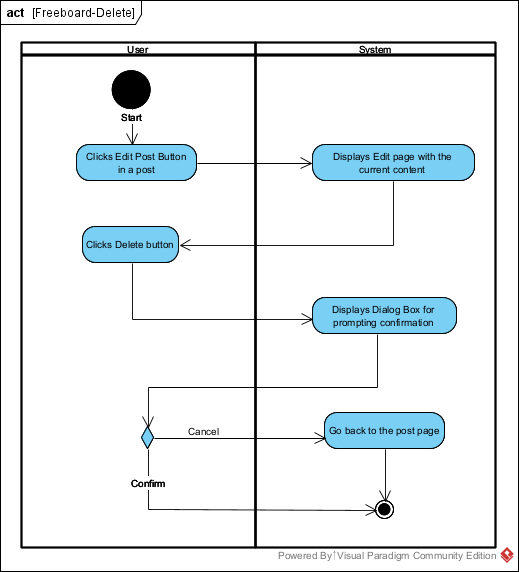
1. Create Post (Freeboard)



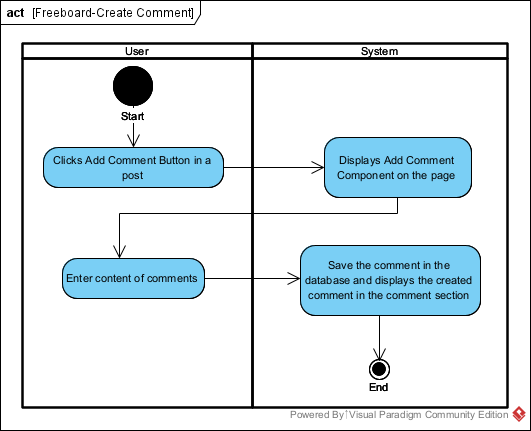
1. Edit Post (Freeboard)



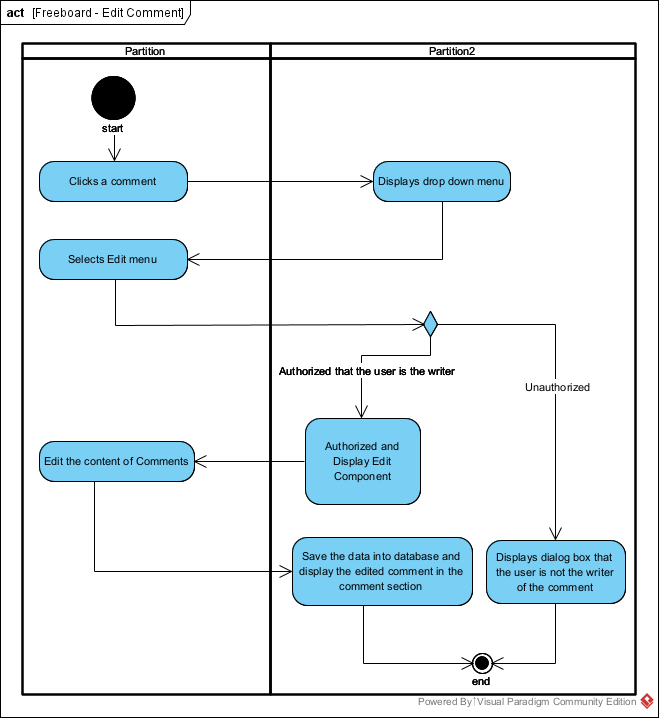
1. Delete Post (Freeboard)



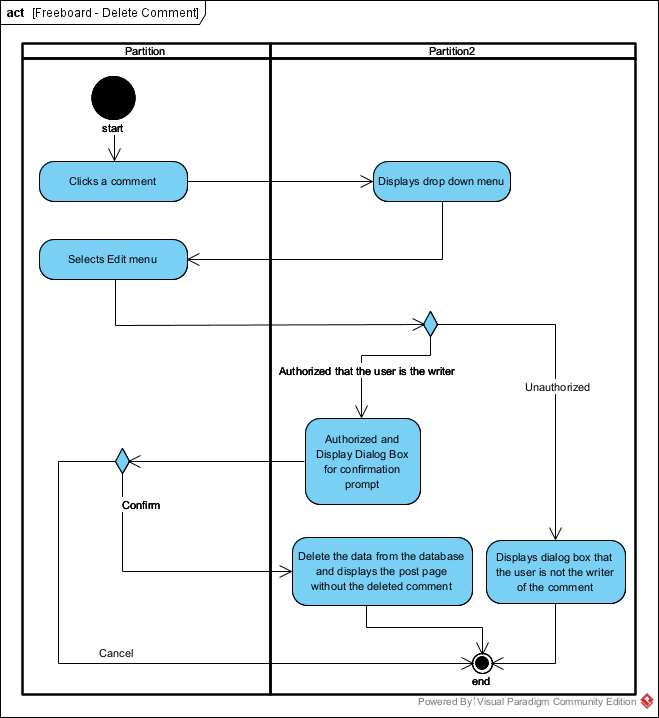
1. Create Comment (Freeboard)



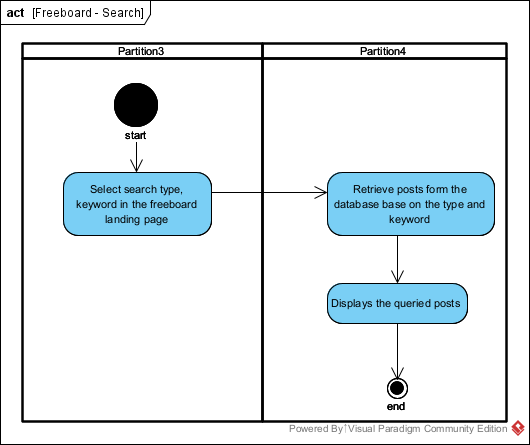
1. Edit Comment



1. Delete Comment (Freeboard)

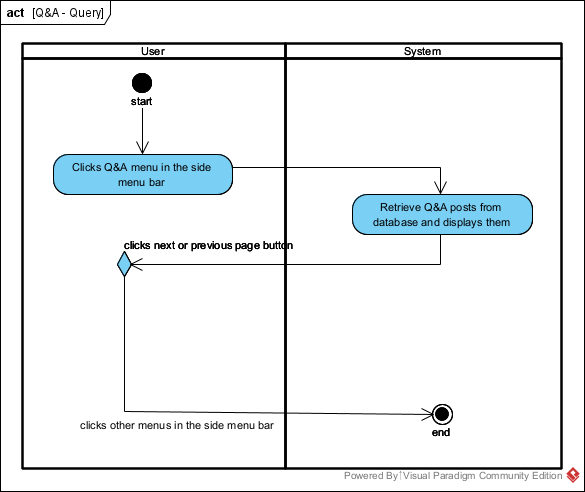


1. Search (Freeboard)

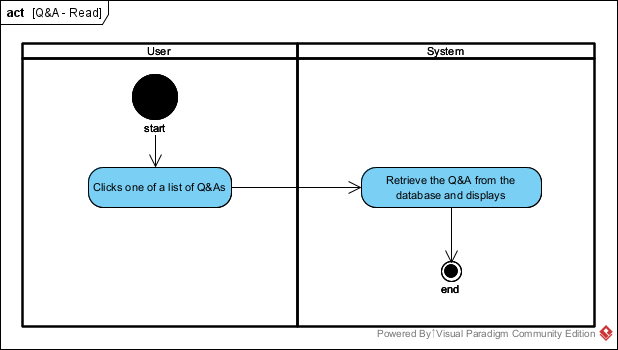


Q&A (Hyunjin Shin)

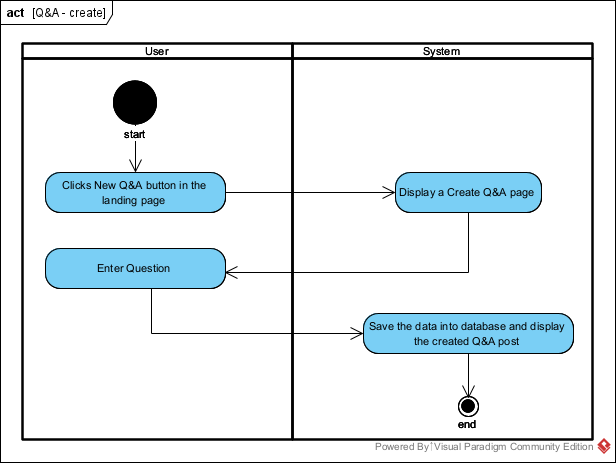
1. Query for landing page (Q&A)



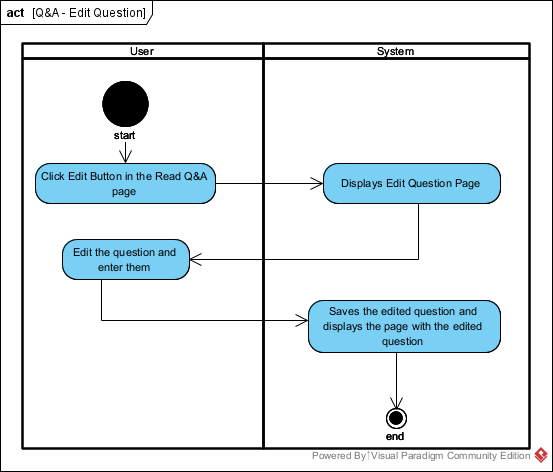
1. Read Q&A post



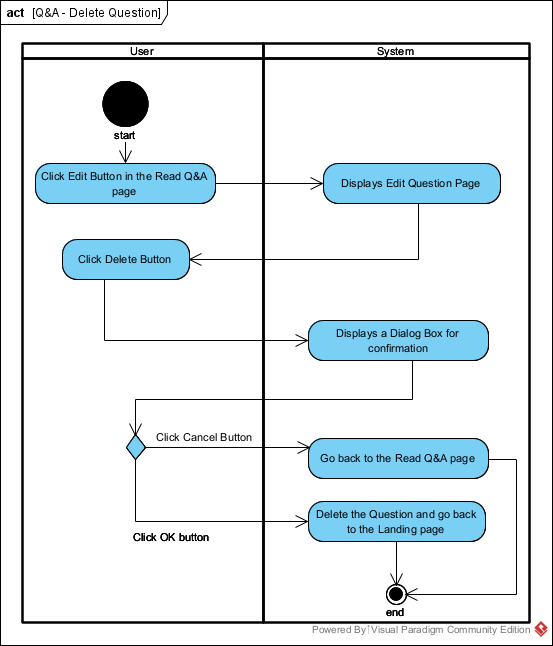
1. Create Q&A post



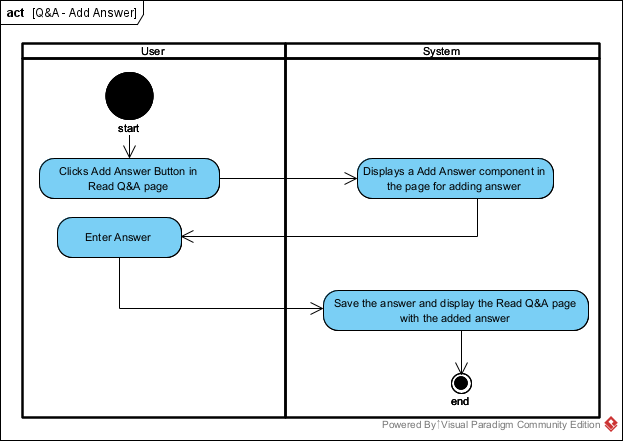
1. Edit Q&A post



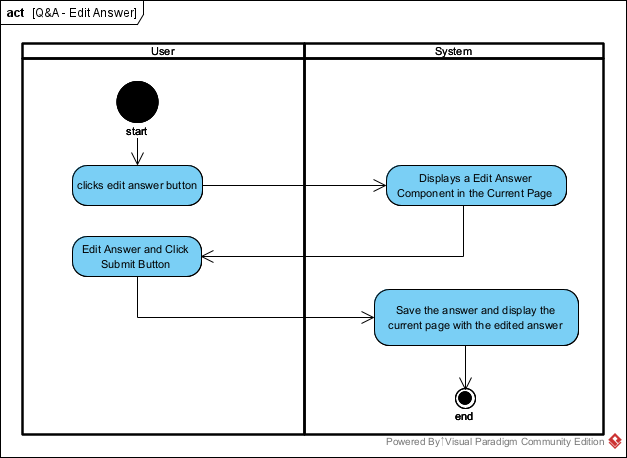
1. Delete Q&A Post



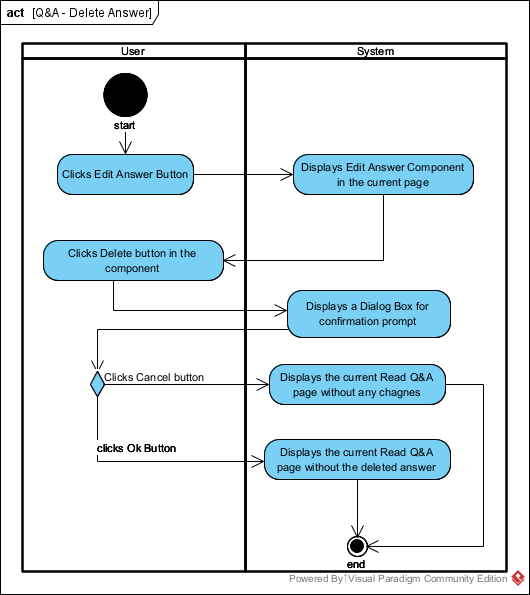
1. Add Answer (Q&A)



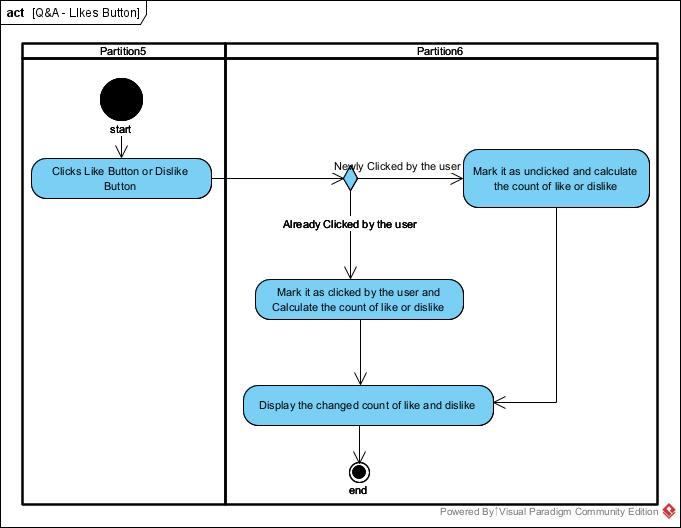
1. Edit Answer (Q&A)



1. Delete Answer (Q&A)



1. Likes Button (Q&A)

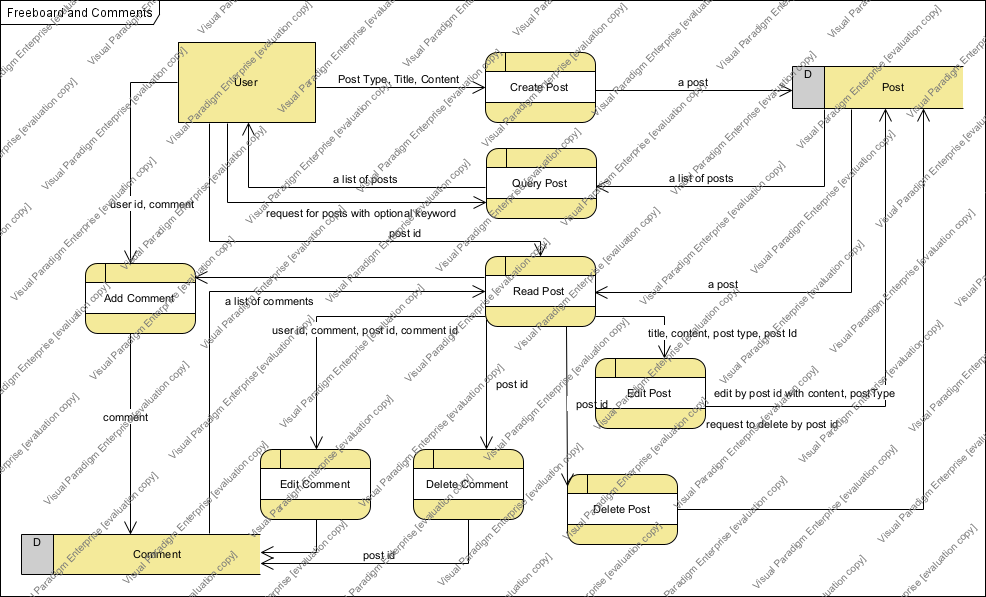


1. Search in the Landing page (Q&A)

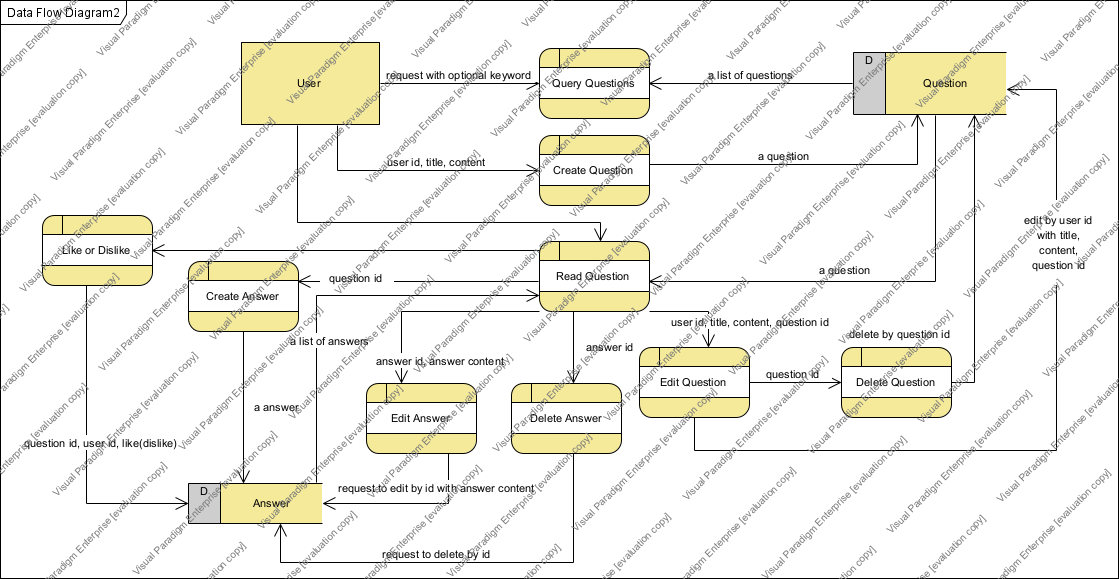


Data Flow Diagram

1. Freeboard and Comments



1. Q&A and Answer



## **3.2 Business Rules**

|  |  |  |
| --- | --- | --- |
| Business Rule Number | Business Rule Description | Related UC |
| BR01 | User must not post inappropriate post such as violent, sexual, or discriminative contents | All Use Cases |
| BR02 | User must be signed-in | UC01, 02, 03, 04 |
| BR03 | User can only update and delete the freeboard posts they created | UC01 |
| BR04 | User can only update and delete the freeboard comments they created | UC02 |
| BR05 | Comments can only be text contents and the length is maximum 400 characters. | UC02 |
| BR06 | User can only update and delete the Q&A posts they created | UC03 |
| BR07 | User can only update and delete the Q&A answers they created | UC03 |
| BR08 | Q&A questions can only be text and image. | UC04 |
| BR09 | Q&A answers can only be text and image. | UC04 |
| BR10 | User can only have one Profile | All Use Cases |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

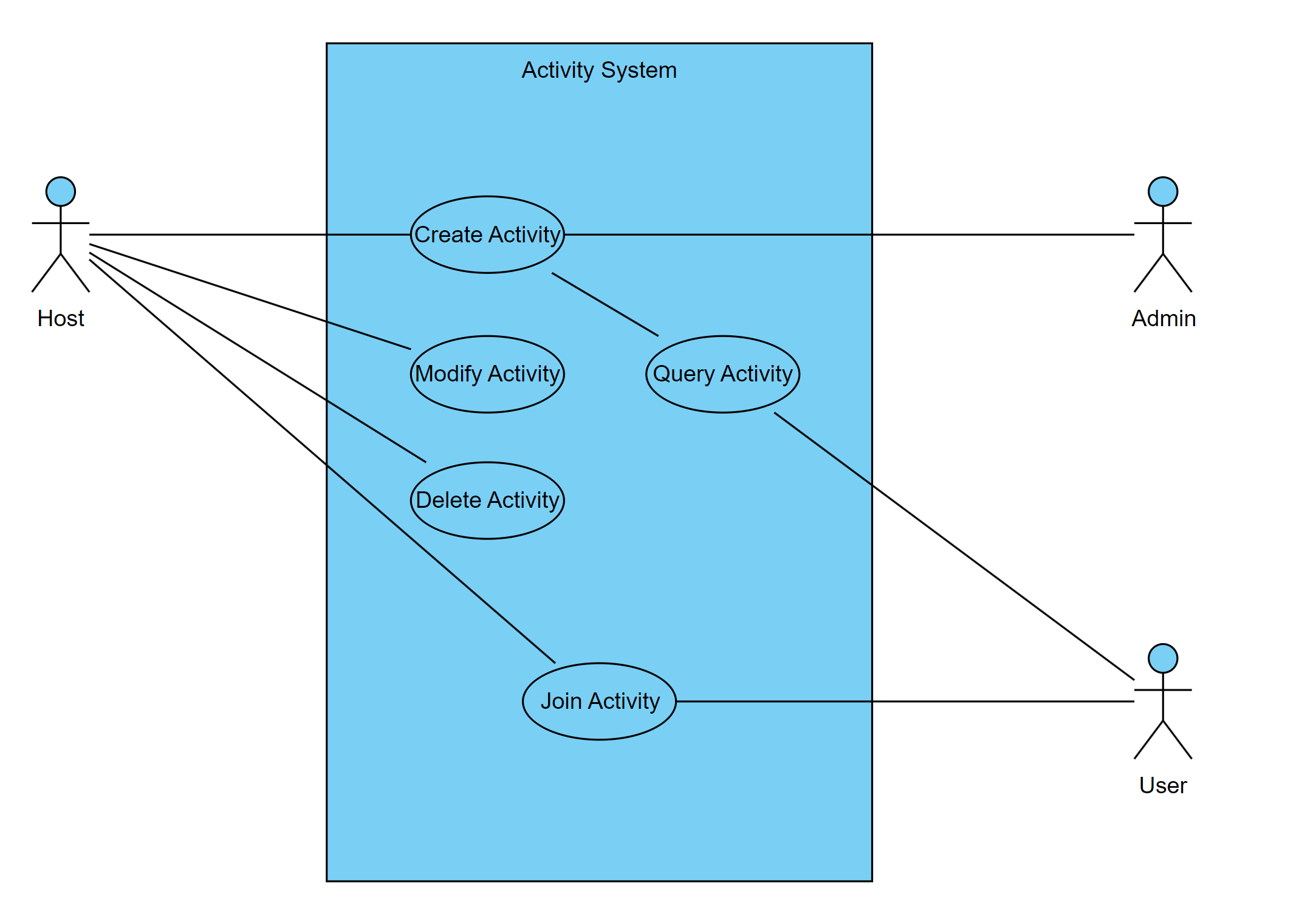
## **3.3 Use Case Specifications with corresponding interface mockups:**

## **Each use case needs to have the following:**

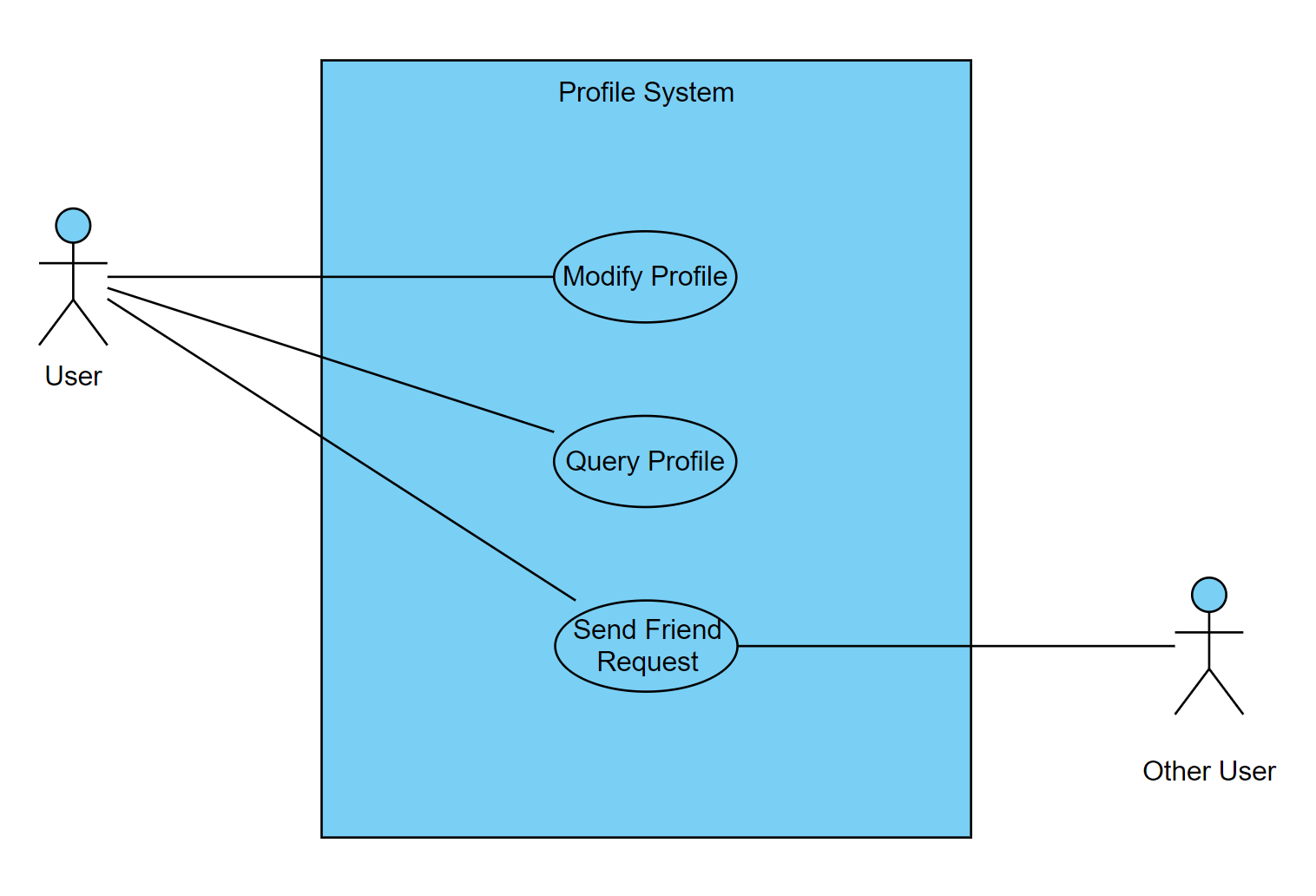
**1- Business Rules.**

**2- System Use Case Diagrams.**

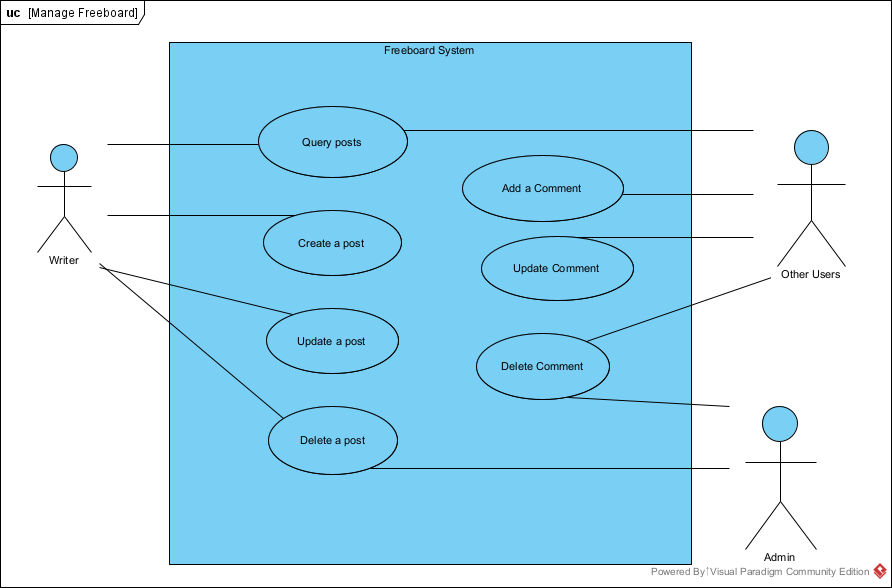
**Activity System**



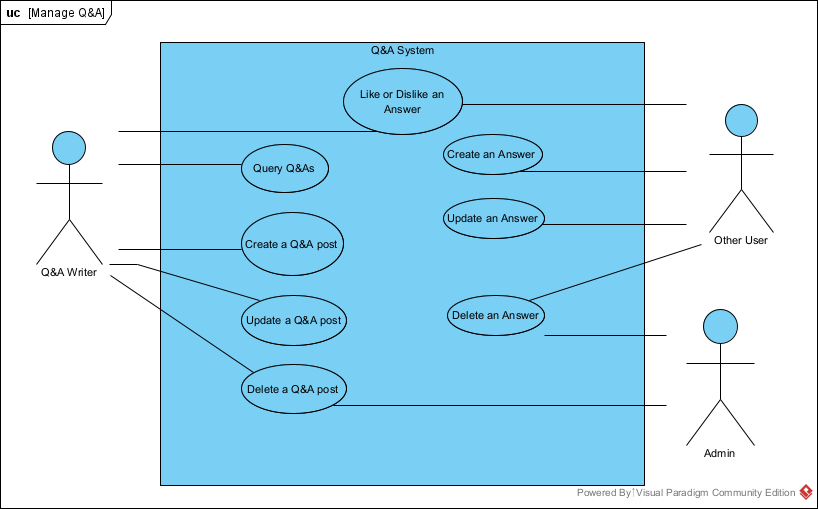
**Profile System**



**Freeboard System**



**Q&A System**



**3- Use Case Descriptions.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Use Case Name: Managing Freeboard | | | | | ID: UC01 |
| Description | | | This use case describes how user write on Seneca Social a post that is for sharing their opinion or stories and Add comments to a post | | |
| Trigger | | | User Signs-in to Seneca Social and writes a new post | | |
| Actor | | | Seneca Students and Employees | | |
| Normal Flow of Events | | | | | |
| Assumption | | | | User has Seneca Social Account | |
| 1. | User enters Seneca Social with their device (Desktop or Mobile) | | | | |
| 2. | User signs-in with their account | | | | |
| 3. | User clicks freeboard menu; then the system queries most recent freeboard posts | | | | |
| 4. | User clicks `Add New Post` button | | | | |
| 5. | User enters contents and save the post | | | | |
| Sub Flow of Events | | | | | |
| Assumption | | | | User has already created a post in freeboard | |
| S-1 | | Update Post | | | |
|  | | 1. | | User clicks the post they created | |
|  | | 2. | | User clicks Edit Button; Then the system will display Editor page | |
|  | | 3. | | User changes the content | |
|  | | 4. | | User saves the change | |
| S-2 | | Delete Post | | | |
|  | | 1. | | User clicks the post they created | |
|  | | 2. | | User clicks Edit Button; Then the system will display Editor page | |
|  | | 3. | | User clicks Delete Button; Then the system will display Alert Dialog Box | |
|  | | 4. | | User confirms | |
| S-3 | | Search Posts | | | |
|  | | 1. | | User enters keywords in search box and click the search button | |
|  | | 2. | | The System will display posts that matches the keyword | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Use Case Name: Managing Comments of Freeboard | | | | ID: UC02 |
| Description | | | This use case describes how user write comments in Freeboard Posts on Seneca Web App | |
| Trigger | | | User goes into a post and write a comment | |
| Actor | | | Seneca Students and Employees | |
| Normal Flow of Events | | | | |
| Assumption | | | User has Seneca Social Account | |
| 1. | User enters Seneca Social with their device (Desktop or Mobile) | | | |
| 2. | User signs-in with their account | | | |
| 3. | User clicks freeboard menu; then the system queries most recent freeboard posts | | | |
| 4. | User clicks a post | | | |
| 5. | User clicks Add Comment Button; then the System displays comment editor box | | | |
| 6. | User enters a comment and saves it. | | | |
| Sub Flow of Events | | | | |
| Assumption | | | User has already created a comment in a freeboard post | |
| S-1 | Update Comment | | | |
|  | 1. | User clicks a post | | |
|  | 2. | User clicks Edit Comment Button; Then the system will display Comment Editor page | | |
|  | 3. | User changes their comment | | |
|  | 4. | User saves the change | | |
| S-2 | Delete Comment | | | |
|  | 1. | User clicks the post they created | | |
|  | 2. | User clicks Edit Comment Button; Then the system will display Comment Editor page | | |
|  | 3. | User clicks Delete Button; Then the system will display Alert Dialog Box | | |
|  | 4. | User confirms the alert | | |
| Use Case Name: Managing Q&A | | | | ID: UC03 |
| Description | | | This use case describes how user write on Seneca Seneca Social Q&A post | |
| Trigger | | | User Signs-in to Seneca Meet-Up App and writes Q&A post | |
| Actor | | | Seneca Students and Employees | |
| Normal Flow of Events | | | | |
| Assumption | | | User has Seneca Social Account | |
| 1. | User enters Seneca Social with their device (Desktop or Mobile) | | | |
| 2. | User signs-in with their account | | | |
| 3. | User clicks Q&A menu; then the system queries most recent Q&As | | | |
| 4. | User clicks `Add New Q&A` button | | | |
| 5. | User enters a question and save the Q&A | | | |
| Sub Flow of Events | | | | |
| Assumption | | | User has already created a Q&A post | |
| S-1 | | Update Q&A | | |
|  | | 1. | User clicks the Q&A they created | |
|  | | 2. | User clicks Edit Button; Then the system will display Editor page | |
|  | | 3. | User changes the question | |
|  | | 4. | User saves the change | |
| S-2 | | Delete Post | | |
|  | | 1. | User clicks the Q&A they created | |
|  | | 2. | User clicks Edit Button; Then the system will display Editor page | |
|  | | 3. | User clicks Delete Button; Then the system will display Alert Dialog Box | |
|  | | 4. | User confirms | |
| S-3 | | Search Q&As | | |
|  | | 1. | User enters keywords in search box and click the search button | |
|  | | 2. | The System will display Q&As that matches the keyword | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Use Case Name: Managing answers in Q&A | | | | ID: UC04 |
| Description | | | This use case describes how user write comments in Freeboard Posts on Seneca Web App | |
| Trigger | | | User goes into a post and write a comment | |
| Actor | | | Seneca Students and Employees | |
| Normal Flow of Events | | | | |
| Assumption | | | User has Seneca Social Account | |
| 1. | User enters Seneca Seneca Social with their device (Desktop or Mobile) | | | |
| 2. | User signs-in with their account | | | |
| 3. | User clicks Q&A menu; then the system queries most recent freeboard posts | | | |
| 4. | User clicks a Q&A | | | |
| 5. | User clicks Add Answer Button; then the System displays answer editor box | | | |
| 6. | User enters Answer and saves it. | | | |
| Sub Flow of Events | | | | |
| Assumption | | | User has already created an Answer in a Q&A post | |
| S-1 | Update Answer | | | |
|  | 1. | User clicks a Q&A | | |
|  | 2. | User clicks Edit Answer Button; Then the system will display Answer Editor page | | |
|  | 3. | User changes their Answer | | |
|  | 4. | User saves the change | | |
| S-2 | Delete Answer | | | |
|  | 1. | User clicks a Q&A they created | | |
|  | 2. | User clicks Edit Answer Button; Then the system will display Answer Editor page | | |
|  | 3. | User clicks Delete Button; Then the system will display Alert Dialog Box | | |
|  | 4. | User confirms the alert | | |
| S-3 | Likes and Dislikes | | | |
|  | 1. | User clicks a Q&A | | |
|  | 2. | User clicks Like or Dislike button of an Answer in a Q&A post | | |
|  | 3. | The number count of Like for the Answer will be updated | | |
| Exceptional | | | S-3: When user clicks likes button twice, the like will be canceled; and it is the same for dislike button. | |

**Activity System**

|  |
| --- |
| Use Case Name: Create Activity |
| Primary Actor: User |
| Stakeholders and Interests:  User – wants to create a gathering of students with similar interests  Seneca Meetup Service – provides a service that allows users with common interests to gather with each other |
| Brief Description: This use case describes how activities are managed and maintained |
| Trigger: User wants to add a new activity to the listing |
| Normal Flow of Events:   1. Capture the date, location, title of the event 2. Capture the detailed description of the event 3. Capture the photos of the event 4. Add the Activity to the listing |

|  |
| --- |
| Use Case Name: Modify Activity |
| Primary Actor: User |
| Stakeholders and Interests:  User – wants to modify an existing activity that they’ve created previously  Seneca Social – provides a service that allows users with common interests to gather with each other |
| Brief Description: This use case describes how activities are managed and maintained |
| Trigger: User wants to modify an existing activity |
| Normal Flow of Events:   1. Capture the Activity Identifier 2. Capture new attributes of the Activity 3. Update the existing Activity |

|  |
| --- |
| Use Case Name: Delete Activity |
| Primary Actor: User |
| Stakeholders and Interests:  User – wants to delete an existing activity that they’ve created previously  Seneca Social – provides a service that allows users with common interests to gather with each other |
| Brief Description: This use case describes how activities are managed and maintained |
| Trigger: User wants to delete an existing activity |
| Normal Flow of Events:   1. Capture the Activity Identifier 2. Delete the Activity from the listing |

|  |
| --- |
| Use Case Name: Query Activity |
| Primary Actor: User |
| Stakeholders and Interests:  User – wants to query all Activities on the listing  Seneca Social – provides a service that allows users with common interests to gather with each other |
| Brief Description: This use case describes how activities are managed and maintained |
| Trigger: User wants to query all Activities |
| Normal Flow of Events:   1. Retrieve all available Activities from the listing 2. Display Activities and its attributes |

|  |
| --- |
| Use Case Name: Join Activity |
| Primary Actor: User |
| Stakeholders and Interests:  User – wants to join an Activity from the listing  Seneca Social – provides a service that allows users with common interests to gather with each other |
| Brief Description: This use case describes how activities are managed and maintained |
| Trigger: User wants to join an Activity from the listing |
| Normal Flow of Events:   1. Capture Activity Identifier 2. Capture User Identifier 3. Add User to Activity’s User list array 4. Send User confirmation notification |

**Profile System**

|  |
| --- |
| Use Case Name: Modify Profile |
| Primary Actor: User |
| Stakeholders and Interests:  User – wants to manage their profile page  Seneca Social – provides a service that allows users to manage their own profile page to represent themselves on the application |
| Brief Description: This use case describes how profile pages are managed and maintained |
| Trigger: User wants to edit profile page |
| Normal Flow of Events:   1. Capture new attributes 2. Delete old attributes 3. Update Profile attributes |

|  |
| --- |
| Use Case Name: Search Profile |
| Primary Actor: User |
| Stakeholders and Interests:  User – wants to manage their profile page  Seneca Social – provides a service that allows users to manage their own profile page to represent themselves on the application |
| Brief Description: This use case describes how profile pages are managed and maintained |
| Trigger: User wants to search a profile page |
| Normal Flow of Events:   1. Capture “Name” attribute 2. Query all public profile pages 3. Display profile pages that match “Name” |

|  |
| --- |
| Use Case Name: Send Friend Request |
| Primary Actor: User |
| Stakeholders and Interests:  User – wants to manage their profile page and query others  Seneca Social – provides a service that allows users to manage their own profile page to represent themselves on the application |
| Brief Description: This use case describes how profile pages are managed and maintained |
| Trigger: User wants to send a friend request on another profile page |
| Normal Flow of Events:   1. Capture Profile Identifier 2. Capture User Identifier 3. Send request notification to current profile identifier |

**4- Corresponding Mockups**

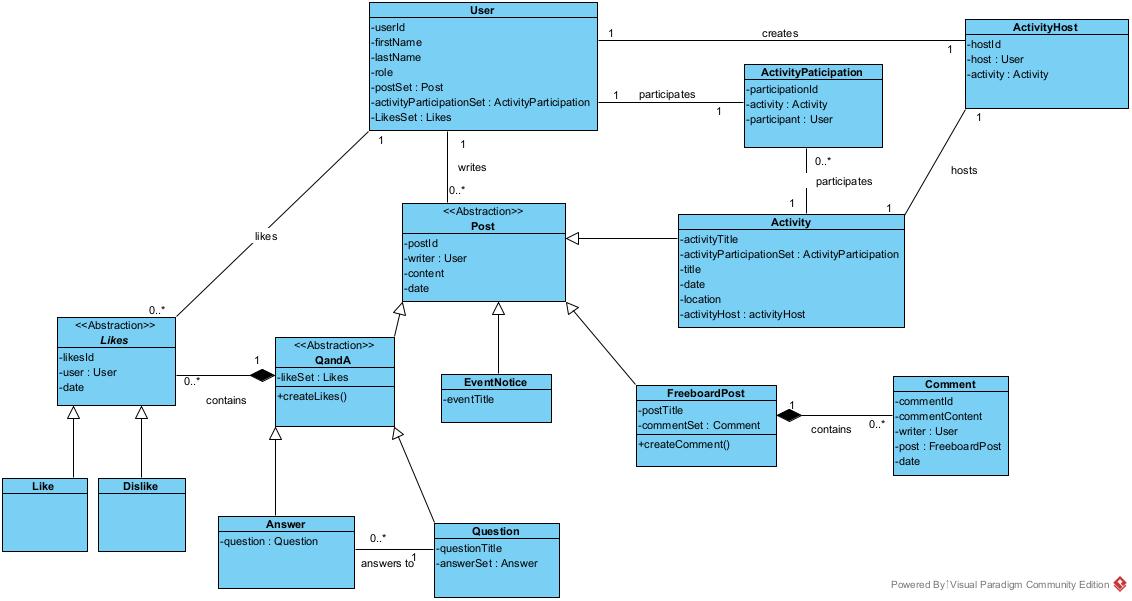
**Refer to Section 2.8 for Mockups and Video Development**

**Mockup Video Link:**

<https://youtu.be/9126SWYo6mA>

[](https://youtu.be/9126SWYo6mA)

# Domain Class Diagram



# Database

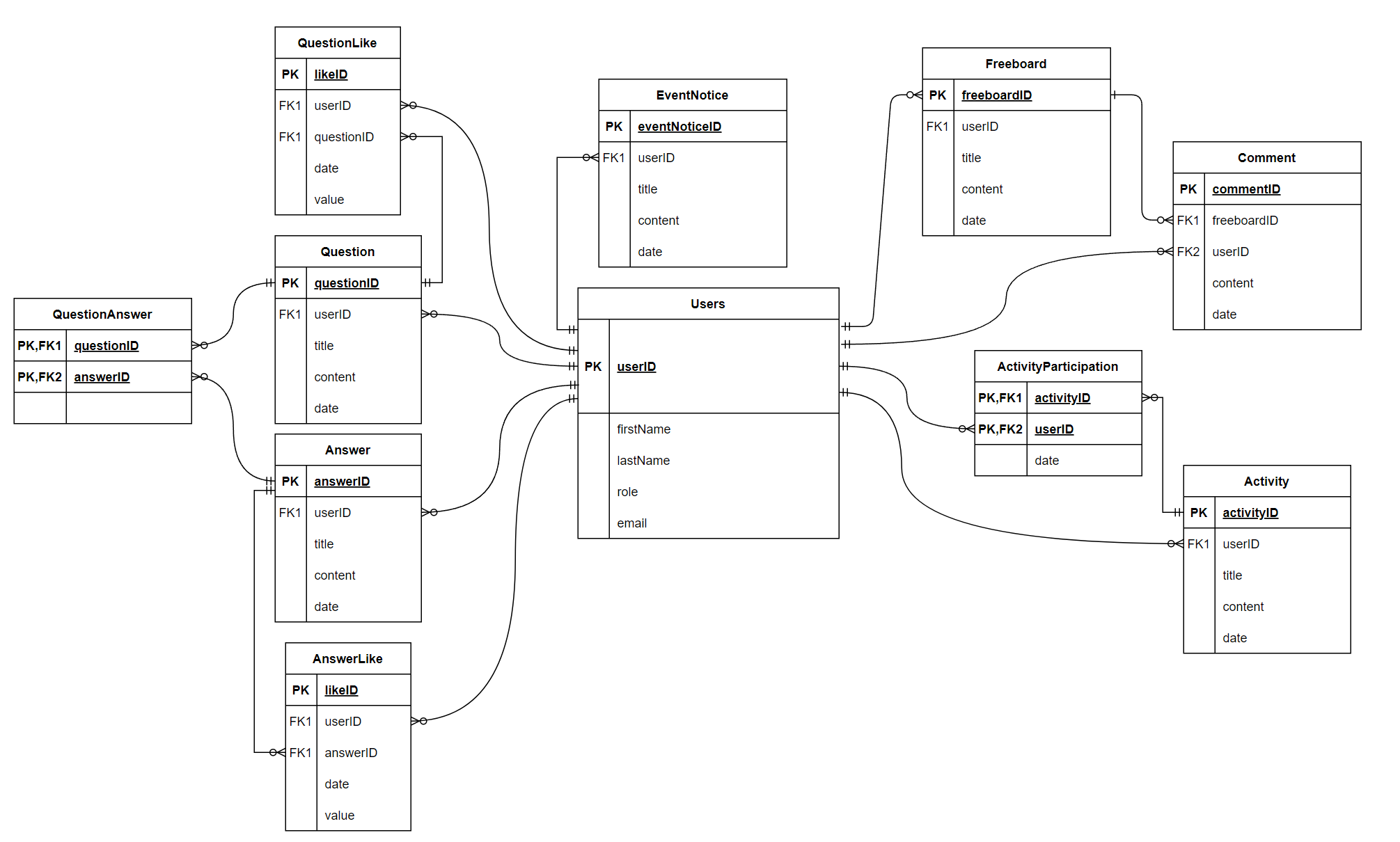


TABLE: **Users**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Size, Precision** | **PK/FK** | **Required** | **Range** | **Sample Data** | **Notes** |
| userID | String | 20 | PK | Y |  | “hshin41” | User’s chosen ID |
| firstName | String | 25 |  | Y |  | “John” |  |
| lastName | String | 25 |  | Y |  | “Smith” |  |
| role | String | 20 |  | Y |  | "User”, “Admin” |  |
| email | String | 100 |  | Y |  | “hshin41@myseneca.ca” | Seneca Email Address |

TABLE: **Freeboard**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Size, Precision** | **PK/FK** | **Required** | **Range** | **Sample Data** | **Notes** |
| freeboardID | Number | 10 | PK | Y | 0-999999 | 10001 | Automatically generated by system |
| userId | String | 25 | FK | Y |  | “hshin41” | Post Writer’s ID |
| title | String | 100 |  | Y |  | “I went to Ottawa Last Week” | Title of the freeboard post |
| content | String | 5000 |  | Y |  | "That was Amazing ...” | The content of freeboard |
| date | Date | 100 |  | Y |  | 2023/05/11-13:05:07 | Posted Date, if the post is updated, it will be replaced with the updated date |

TABLE: **Comment**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Size, Precision** | **PK/FK** | **Required** | **Range** | **Sample Data** | **Notes** |
| commentID | Number | 10 | PK | Y | 0-999999 | 100001 | Automatically generated by system |
| userId | String | 25 | FK | Y |  | “hshin41” | Comment Writer’s ID |
| content | String | 250 |  | Y |  | "I really envy you I wish I could ...” | The content of the comment |
| date | Date | 100 |  | Y |  | 2023/05/11-13:05:07 | Posted Date, if it is updated, it will be replaced with the updated date |

TABLE: **Question**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Size, Precision** | **PK/FK** | **Required** | **Range** | **Sample Data** | **Notes** |
| questionID | Number | 10 | PK | Y | 0-999999 | 101 | Automatically generated by system |
| userId | String | 25 | FK | Y |  | “hshin41” | Question Writer’s ID |
| title | String | 100 |  | Y |  | “I have a question about how to...” | Title of the question |
| content | String | 2000 |  | Y |  | "I tried to apply for the course ...” | The content of the question |
| date | Date | 100 |  | Y |  | 2023/05/11-13:05:07 | Posted Date, if the question is updated, it will be replaced with the updated date |

TABLE: **Answer**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Size, Precision** | **PK/FK** | **Required** | **Range** | **Sample Data** | **Notes** |
| answerID | Number | 10 | PK | Y | 0-999999 | 101 | Automatically generated by system |
| userId | String | 25 | FK | Y |  | “hshin41” | Question Writer’s ID |
| content | String | 2000 |  | Y |  | "I think the problem is ...” | The content of the answer |
| date | Date | 100 |  | Y |  | 2023/05/11-13:05:07 | Posted Date, if the question is updated, it will be replaced with the updated date |

TABLE: **QuestionAnswer**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Size, Precision** | **PK/FK** | **Required** | **Range** | **Sample Data** | **Notes** |
| questionID | Number | 10 | PK,FK1 | Y | 0-999999 | 101 |  |
| answerID | Number | 10 | PK,FK2 | Y |  | “hshin41” |  |

TABLE: **QuestionLike**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Size, Precision** | **PK/FK** | **Required** | **Range** | **Sample Data** | **Notes** |
| likeID | Number | 10 | PK | Y | 0-999999999 | 1001 | It will be generated by system |
| questionID | Number | 10 | FK | Y | 0-999999 | 101 |  |
| userId | String | 25 | FK | Y |  | “hshin41” | ID of the user who pressed the like button |
| date | Date | 100 |  | Y |  | 2023/05/11-13:05:07 | Posted Date, if the question is updated, it will be replaced with the updated date |
| value | Number | 1 |  | Y | -1 or 1 | 1 | 1 will be like and –1 will be dislike |

TABLE: **AnswerLike**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Size, Precision** | **PK/FK** | **Required** | **Range** | **Sample Data** | **Notes** |
| likeID | Number | 10 | PK | Y | 0-999999999 | 1001 | It will be generated by system |
| answerID | Number | 10 | FK | Y | 0-999999 | 101 |  |
| userId | String | 25 | FK | Y |  | “hshin41” | ID of the user who pressed the like button |
| date | Date | 100 |  | Y |  | 2023/05/11-13:05:07 | Posted Date, if the question is updated, it will be replaced with the updated date |
| value | Number | 1 |  | Y | -1 or 1 | 1 | 1 will be like and –1 will be dislike |

TABLE: **Activity**

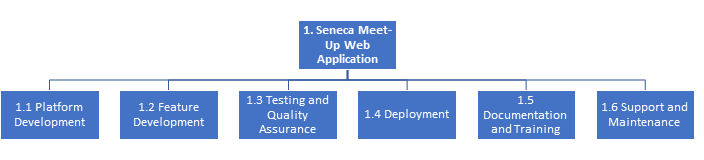
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Size, Precision** | **PK/FK** | **Required** | **Range** | **Sample Data** | **Notes** |
| activityID | Number | 5 | PK | Y | 0-99999 | 011 | System-generated |
| UserID | String | 25 | FK | Y |  | “hshin41” | A User who organized the activity |
| title | String | 50 |  |  |  | “Dance Party” | Title of the activity |
| content | String | 2000 |  | Y |  | meeting date, dress code, duration of the activity, etc | Detailed information about activity |
| date | Date | 100 |  |  |  | 2023/05/11-13:05:07 |  |
| location | String | 128 |  |  |  | “1750 Finch Ave East Toronto, On M2J 5G3” | The location of activity taking place |

TABLE: **ActivityParticipation**

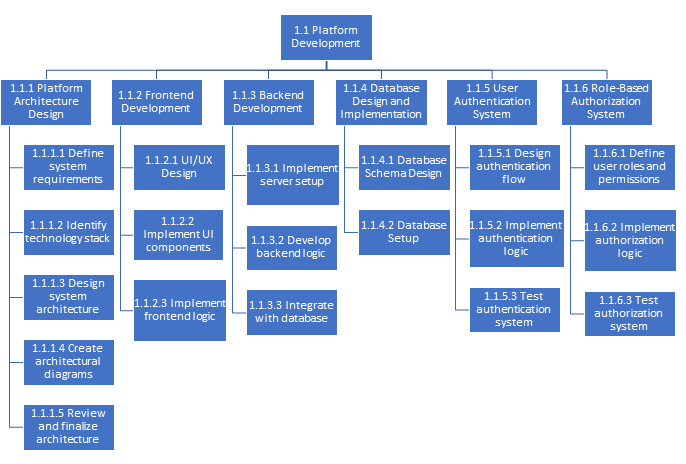
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Size, Precision** | **PK/FK** | **Required** | **Range** | **Sample Data** | **Notes** |
| userID | String | 25 | PK | Y |  | “hshin41” | User who participates in the activity |
| participationID | Number | 5 | FK | Y | 0-99999 | 011 | Activity ID |

# Work Breakdown Structure (WBS)

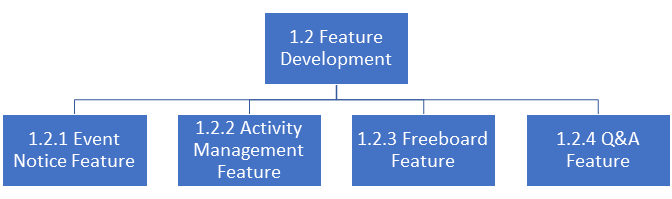
## Work Breakdown Structure



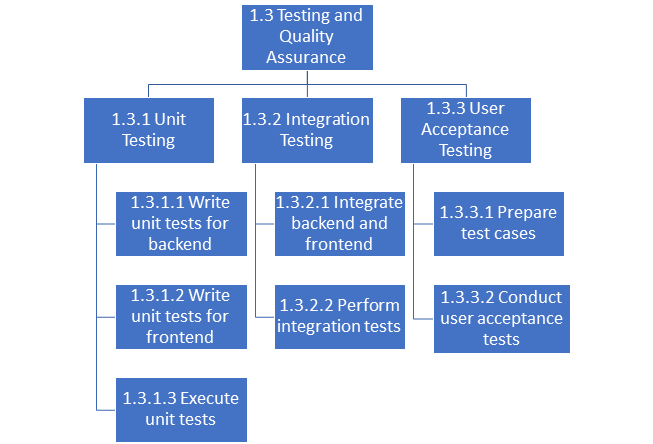
*Figure 1*



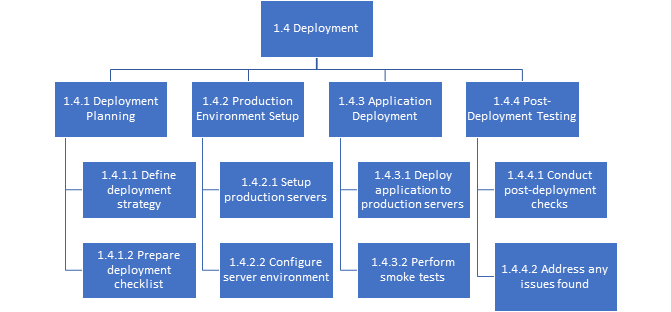
*Figure 2*



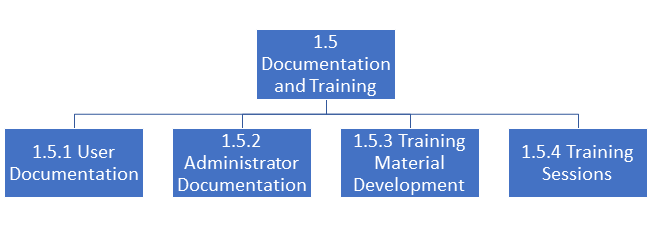
*Figure 3*



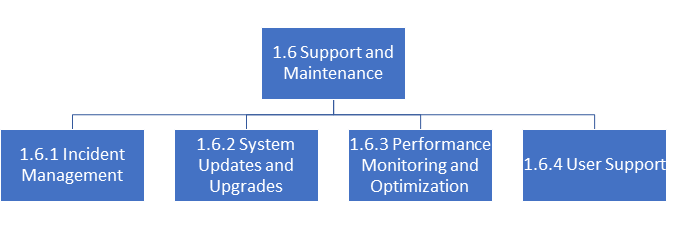
*Figure 4*



*Figure 5*



*Figure 6*



*Figure 7*

# Milestones and Acceptance Criteria

Milestone 1: Finalization of Design and Architectures

Expected Time Period: 2 weeks

Definition:

1. Create fully developed wireframes and mockups.
2. Develop UI/UX designs.
3. Define system architecture and components.
4. Review and iterate on design concepts.

Acceptance Criteria:

1. Fully developed Wireframes and mockups created and reviewed by stakeholders.
2. UI/UX designs developed and validated for usability and consistency.
3. System architecture documented and reviewed by technical experts.
4. Design concepts refined based on feedback and approved by stakeholders.

Milestone 2: Core Feature Development

Expected Time Period: 6 weeks

Definition

1. Implement Freeboard feature
2. Implement Activity feature
3. Implement Q&A feature
4. Implement Authentication
5. Implement Event Notice feature

Acceptance Criteria:

1. Unit testing to ensure each core functionality works as expected.
2. Unit tests pass successfully for each core functionality
3. Core functionalities implemented and tested individually.
4. Core features provide the functionality that they are supposed to do

Milestone 3: Database Implementation

Expected Time Period: 2 weeks

Definition

1. Database system selected based on project requirements
2. Database schema designed and finalized.
3. Physical database structures created and configured.
4. Perform integration testing to ensure compatibility with application systems

Acceptance Criteria:

1. Database tested for functionality, performance, and reliability
2. Integration tests confirm seamless interaction with application systems
3. Backup copies of the database stored securely and verified for integrity

Milestone 4: Testing

Expected Time Period: 1 week

Definition

1. Conduct testing for all the unit tests again for all the implemented codes
2. Conduct Integration testing
3. Conduct User Acceptance testing.

Acceptance Criteria

1. Testing should be automated for future maintenance.
2. All the tests must be passed without failing any test.

Milestone 5: Deployment

Expected Time Period: 2 weeks

Definition

1. Discuss the final version of the product to stakeholders for approval
2. Develop and finalize the deployment plan (decide how and where to deploy the product)
3. Execute the Deployment
4. Conduct End-User testing
5. Monitor the deployed product for any issues or performance concerns

Acceptance Criteria

1. Deployment plan documented and approved
2. Deployment completed successfully without major issues or disruptions
3. End-users feel comfortable with using the deployed product
4. While Monitoring, systems work and function as expected

**Milestone 6: Security Review**

Expected Time Period: 2 weeks

**Definition**

1. Conduct a thorough review of the application's security posture to identify potential vulnerabilities
2. Implement necessary security measures to mitigate risks and ensure data protection
3. Perform security testing, including penetration testing and vulnerability scanning, to validate the effectiveness of security controls

**Acceptance Criteria**

1. Security review conducted, and potential vulnerabilities identified and addressed
2. Security measures implemented to mitigate identified risks effectively
3. Security testing performed, and application passes penetration testing and vulnerability scanning

**Milestone 7: Project Closure and Handover**

Expected Time Period: 1 week

**Definition**

1. Complete all project closure activities
2. Hand over the project to the client

**Acceptance Criteria**

1. Project closure activities completed, including documentation and final reports
2. Hand over the project to the designated team or client following an established transition plan

**Milestone 8: Maintenance Planning**

Expected Time Period: N/A

**Definition**

1. Develop a maintenance plan outlining procedures for ongoing support, updates, and bug fixes post-deployment
2. Establish communication channels and support mechanisms for addressing user-reported issues and implementing changes

**Acceptance Criteria**

1. Maintenance plan developed and documented, detailing procedures for support, updates, and bug fixes
2. Communication channels and support mechanisms established for addressing user-reported issues

# Implementation Schedule

Product Backlogs (Agile-Scrum)

1. Freeboard

As a User, I want to have freeboard system in order to share my story and opinion with other users.

Description

A freeboard system will provide the user with create, read, update, and delete (CRUD) features for a post. A post consists of title, content, and comment sections. The writer of the post, and the date will be automatically generated by the server and saved together. The post will be saved into database and can be managed by the user. A freeboard system also provides management of comments in the freeboard. Users will be able to CRUD comments on a Post. Comments will be saved in database. Once a post is deleted, the comments in the post will also be deleted together.

Acceptance Criteria

* Users can create, read, update, delete a post
* Posts should be able to be stored in database
* Users can create, read, update, delete a comment
* Comments should be able to be stored in database
* When a post is loaded, the comments of the post should be loaded together
* Supports user-friendly UI.

Testing Criteria

* Created posts and comments must persist in Database
* Post cannot exceed 5000 characters
* Comments of a post must be able to be queried with the post.
* Comments cannot exceed 250 characters

1. Q&A

As a User, I want to ask questions in order to get information about Seneca Life and academic improvement

Description

Q&A system will enable the user to create, read, update, and delete a question. A question consists of title, and content. Other users can leave an answer to the question. Users can also rate the question or the answer by clicking like or dislike button. All the questions, answers, likes will be saved into database. Once a question is deleted, the likes and answer data are also deleted together.

Acceptance Criteria

* Users should be able create, read, update, delete a question
* A question should persist in database
* Users should be able to create, read, update, delete an answer for a question
* An answer should persist in database and related to the question
* Users can rate the answer and comments by likes.
* Likes should persist in database.
* User-friendly UI.

Test Criteria

* One user can click a like or dislike only once for a question or an answer.
* User can only leave one answer for one question
* Question cannot exceed 2000 characters.
* Answer cannot exceed 2000 characters.

1. Profile
   1. As a User, I want to have a Profile page on the school's Social Networking Service (SNS) platform, which displays my joined activities along with basic personal information such as name, current studying program, year, and status.
   2. Description
      1. The Profile page will showcase the user's engagement with various activities within the school community. It will list the activities the user has joined, along with personal details. The page will provide a snapshot of the user's involvement and interests within the school environment.
   3. Acceptance Criteria
      1. Users should be able to view their Profile page.
      2. Profile page should display the user's name, current studying program, year, and status.
      3. Joined activities should be listed on the Profile page.
      4. Personal information displayed should be accurate and up to date. Testing Criteria
   4. Test Criteria
      1. Ensure that all displayed personal information is correct.
      2. Verify that joined activities are accurately listed.
      3. Confirm that the Profile page layout is visually appealing and user-friendly.
2. Activities
   1. As a User, I want to create and join various events within the school community, such as music festivals, amusement park outings, city tours, beer nights, and more.
   2. Description
      1. The Activities feature enables users to create and participate in diverse events organized within the school community. Users can create events, set details such as date, time, location, and description, for others to join.
   3. Acceptance Criteria
      1. Users should be able to create new activities/events.
      2. Activities should include details such as date, time, location, and description.
      3. Users should be able to join existing activities.
   4. Test Criteria
      1. Create a new activity and verify that it appears in the list of available events.
      2. Join an existing activity and ensure successful participation.
      3. Verify that event details such as date, time, and location are accurately displayed.

**TASKS**

1. Database
2. Test
   1. Functionality Validation
      1. Ensure all features work as intended.
      2. Test CRUD operations for both posts and comments.
      3. Validate user authentication and authorization processes.
   2. Data Integrity Check
      1. Verify that data entered by users is correctly stored in the database.
      2. Ensure data retrieval functions return accurate results.
   3. Security Assessment
      1. Conduct security testing to identify vulnerabilities.
      2. Validate input validation and protection against common attacks.
   4. Performance Evaluation
      1. Test system performance under typical usage scenarios.
      2. Identify potential bottlenecks and optimize system components.
   5. Compatibility Testing
      1. Confirm compatibility with various browsers and devices.
      2. Ensure consistent user experience across different platforms.
   6. Error Handling Review
      1. Evaluate error messages and logging mechanisms.
      2. Ensure graceful handling of unexpected situations.
   7. Documentation Review
      1. Verify completeness and accuracy of deployment documentation.
      2. Ensure clear instructions for deployment and configuration.
3. Deployment

Back-end

1. The back-end server code is committed in GitHub
2. The source code is dockerized and uploaded in AWS Elastic Container Registry
3. The docker file runs on AWS Elastic Container Service, connecting to the AWS Oracle Database

Front-end

1. The Front-end code is committed in GitHub
2. The front-end is deployed using Cyclic.

Acceptance Criteria

* + Back-end server should be running on AWS ECS.
  + Front-end should be able to be accessed via various web browsers such as Chrome, Edge, and Firefox.
  + Front-end server should be able to send a HTTP request to the Back-end server and receive a HTTP response.
  + Back-end server should be able to save data into database and query from the database.

Agile Scrum

Sprint 1

Sprint backlogs:

* Authentication System
* Freeboard System

Sprint Date: May 6, 2024 – Jun 3, 2024

Sprint 2

Sprint backlogs:

* Q&A system
* Activity system

Sprint Date: Jun 3, 2024 – July 1, 2024

Sprint 3

Sprint backlogs:

* Profile System
* Event Notice System

Sprint Date: July 1, 2024 – Jul 29, 2024

Sprint 4

Sprint backlogs:

* Database Implementation
* Final Test
* Deployment

Sprint Date: Jun Jul 30, 2024 – Aug 27, 2024

# Client / Faculty Sign-off

**Date: \_\_\_\_\_\_\_April 14, 2024\_\_\_\_\_\_\_\_\_**

X Hyunjin Shin

Juhan Kim

Ke An Lo

Name of Client/Rep/Professor