

Software Requirements Specification

StudyBoard

Team Members

Pyungkang Hong	UI/UX Design	pyungkang.hong@stonybrook.edu
Hawon Park	PM	hawon.park@stonybrook.edu
Jeong Ho Shin	LP	jeongho.shin@stonybrook.edu

Last Modified: December 6th, 2021

Table of Contents

1. Introduction	3
1.1 Product Description	3
1.2 Scope	3
1.3 Users	4
1.4 User Feedback	4
1.5 Existing Alternatives	4
1.6 Definition	5
1.7 References	5
2. Requirements	6
2.1 Functional Requirements	6
2.2 Use Cases	8
2.3 Non-functional Requirements	11
3. System Architecture	12
3.1 Overview	12
3.2 Data Design	14
3.3 UML Sequence Diagrams	19
3.4 API Design	21
3.5 Deployment	21
3.6 Code Conventions	22
4. Schedule	22
5. Contributions	23

1. Introduction

1.1 Product description

People often have questions that they need answering. There might be a specific term that they don't understand or a topic that they are trying to review. More often than not, these people turn to the internet to try and find an answer. They might be able to find what they are looking for in five minutes, or they might also end up spending hours without much success. Even if they can find answers to their questions, these answers might be vague, factually incorrect, or too poorly explained to understand.

StudyBoard aims to prevent such problems by providing a web platform where anyone with a question in mind can post their thoughts and receive clear and instructional answers. Users are able to interact with one another by leaving replies and sharing posts. Users can also like content that they see, be it question or reply. Users can also flag any inappropriate or incorrect questions and replies. Such posts or their replies are reviewed by moderators and removed if needed. This ensures that no one leaves the platform with the wrong information in hand and that all content is of high quality. Having a unified search bar also helps users peruse through any subject of their choosing.

These features, when combined, facilitates active discussion and improved learning by enabling each person to leave with the exact information that they were looking for.

1.2 Scope

StudyBoard is a web platform that aims to help users get replies to their questions. This application allows users to post, reply to, and share questions. This platform also allows users to vouch for specific responses by liking the replies. Users can increase in rank via the number of likes that their replies have.

When a piece of content seems inappropriate, users may flag the content to be reviewed by the moderators. While StudyBoard does not automatically verify the factuality of a reply, the user interactions of liking and flagging content would sift out inaccurate replies over time. Moderators would review any reported content and decide whether to accept or reject the report as valid.

StudyBoard is not limited to any specific group of people but is open to anyone with a question in mind. StudyBoard would be optimized to be a responsive, cross-platform web application supported by any modern web browser.

1.3 Users

StudyBoard's primary target group is anyone with a question that they would like answered. While there is no specific limitation in terms of educational level or experience in who can ask a question, StudyBoard assumes that the user has moderate experience in browsing the modern internet to search for information on platforms such as Stack Overflow or Quora.

StudyBoard has three types of users. First, there are the normal users who can post questions, reply to questions, and interact with one another. Then, there are users who can be elevated to an endorsed status where their word has more weight than normal users. Users are upgraded based on a combination of the number of likes they have and the number of flags that they have received. Last but not least are the moderators whose job is to manage flagged content to keep the platform safe from incorrect or inappropriate content.

1.4 User feedback

StudyBoard has received preliminary feedback from three users of varying technological expertise who were shown the SRS document and the most recent UI mock-ups. They brought up the issue of how to deal with questions with duplicate or similar content, which we decided to allow. They also asked if users would be able to have a list of tags to which the platform would filter newly generated questions for a more personalized question feed. We decided that this was a missing key feature that we wanted to add to our platform.

StudyBoard plans to receive further feedback at least two more times. At the very least, we want to receive feedback after the alpha and beta deployments for StudyBoard. We plan to get feedback from a wide variety of users so that we can optimize the experience for as many users as possible.

1.5 Existing alternatives

There are a few existing alternatives to this problem of finding an answer on the internet. A list of these platforms includes Stack Overflow, Naver Knowledge iN, and Quora. All three of these platforms mentioned above provide a space where anyone can ask a question and receive a response. One key strength shared by all three platforms is that they all have a relatively large active user base that steadily asks and answers questions online. Another key strength that these platforms share is that they all implement their respective versions of a ranking/rating system for users and the content they post.

However, there are some weaknesses to these platforms that StudyBoard aims to overcome. Naver Knowledge iN is a strictly Korean website, limiting usage to just Korean users. Quora requires users to be logged in to search for content. Stack Overflow is highly specialized in the field of Computer Science.

StudyBoard aims to adapt the best features found in these platforms to create a streamlined, comprehensive platform. Specifically, StudyBoard takes inspiration from Stack Overflow's question and reply model and Naver Knowledge iN's ranking system to create an easily accessible platform with a wider content scope. Furthermore, StudyBoard would allow users to freely search for content without having to log in which is contrary to Quora's model. Users would only be required to log in to submit a reply and access related features.

1.6 Definitions

Moderator	A moderator is someone who is in charge of deleting inappropriate content from the platform.
Event	An event is when there is an update to the website (newly posted reply, newly posted question, like counts going up, etc).
UI	User Interface.
Nginx	Nginx is a web server that can also be used as a reverse proxy, load balancer, mail proxy, and HTTP cache.
I.E.	Internet Explorer; Stock web browser provided with the Windows operating system.
Instance	An instance is a singular web server that contains the platform's source code.
UX	User Experience.
PM	Project Manager.
LP	Lead Programmer.
CDN	Content Delivery Network.
Google OAuth	Google OAuth is an authentication service used to implement Google Sign-In functionality in web and mobile applications.
SDD	Software Design Specification.
Pagination	Pagination is a web design technique in which a document is divided into pages to fit the content in the browser.
Thread	A thread is a series of sub-posts attached to a single post.

1.7 References

Stack Overflow	https://stackoverflow.com/
Naver Knowledge iN	https://kin.naver.com/
Quora	https://www.quora.com/

2.1 Functional Requirements

Stretch Goals are preceded by an asterisk (*)

Login Page

- Users can log into the website using Auth0.

Post Question

- Users can post questions with optional media attached (Images).
- Users can edit and delete their questions.
- Users can add tags to their posts to further describe what kind of question they are asking.
- * Users can format their questions within a text editor.

Question Feed

- Users can view the question feed in pagination style.
- Users can also navigate to a question's individual page with all the respective replies
- Users can sort the feed by most recently posted or the highest number of user interactions (endorsed likes, likes, favorites, replies).
- Users can filter the question feed with their preferred tags.

Question Page

- Users can add replies to a question which are displayed thread style underneath the question.
- Users can add replies to other replies which are also displayed thread style underneath the reply.
- Users can like a question or reply
 - System upgrades users who have enough likes to 'endorsed' status.
- Users can flag a question or reply as inappropriate.
 - * Endorsed users that have received enough reports are demoted to normal status.
- Users can get the link to a specific question by clicking on the share button.
- Users can add a specific question to their favorites list
- Replies can be most recently posted or by the number of user interactions (endorsed likes, likes, favorites, replies).

Unified Search Bar

- Users can search for existing questions using the title, content, or tags.
- * Users can see search result previews

Rank System

- System displays a special mark next to the names of endorsed users.
- Users can see the highest-ranking users in a chart.

Profile Page

- Users can view and edit their nicknames.
- Users can see their list of favorite questions.
- Users can see their uploaded questions.
- Users can see and edit their list of preferred tags.

Moderator User

- Moderators can see a list of flagged content that they can review.
- Moderators can delete a post that they think is inappropriate.
- Moderators can delete a reply that they think is inappropriate.

- Moderators can mark a post as appropriate.
- Moderators can mark a reply as appropriate.
- Moderators can blacklist a user from accessing the platform.

Notifications

- Users are notified when their question or reply is liked.
- Users are notified when their question or reply has a new reply.
- Users are notified when the user gains the endorsed status
- Users are notified when the user loses the endorsed status
- Users are notified when the question or reply that they reported has been deleted.
- * Users are notified when their question or reply is deleted by a moderator.

2.2 Use cases

Use Case 1: User selects Tags for Personalized Question Feed After First Login

Primary actor:	User
Goal in context:	The user selects the tags to personalize their question feed.
Preconditions:	User has logged into the platform using Google OAuth.
Scenario:	<ol style="list-style-type: none"> 1. User clicks Sign-In with Google. 2. System displays the 'Sign Up' page. 3. User clicks/enters at least one tag to follow (required). 4. User clicks/enters as many additional tags as the user wants for a personalized feed. 5. User clicks on the 'Sign In' to finalize the Sign-In process. 6. System verifies the data and directs the user to the 'Question Feed' page.
Extensions:	5a. User clicks on the 'Cancel' button. 5a.1 System deletes user information.
Priority:	Expected

Use Case 2: User Posts a Question

Primary actor:	User
Goal in context:	The user wants to ask a question that they need answering.
Preconditions:	User has successfully logged into the platform.
Scenario:	<ol style="list-style-type: none"> 1. User clicks on the 'Post Question' textbox to type out their question. 2. User submits the question by clicking on the 'Post' button. 3. System updates the feed with the new post at the top of the list.
Extensions:	1a. User wants to add an image to the question. 1a.1 User clicks on the 'Add Image' button. 1a.2 System shows the file selector window. 1a.3 User selects the image to add and clicks on the 'ok' button. 1b. User wants to add tags to the question. 1b.1 User types '#' and a word to add tags. 2a.1 User wants to cancel the posting process. 2a.1 User clicks on the 'Cancel' button. 2a.2 System displays an alert asking for confirmation. 2a.3 User clicks on the 'ok' button. 2a.3 The process is cancelled.
Priority:	Essential

Use Case 3: User replies to a question

Primary actor:	User
Goal in context:	User replies to a question posted by another user.
Preconditions:	The question exists in the database.
Scenario:	<ol style="list-style-type: none"> 1. User clicks on the 'Reply' button. 2. System prompts the user to type the response. 3. User clicks on the 'Submit' button to post their response to a question. 4. System updates the post with the added reply.
Extensions:	3a. User clicks on the 'Cancel' button. 3a.1 Text input by the user is deleted and the process is cancelled.
Priority:	Essential

Use Case 4: User adds a question to their favourites list

Primary actor:	User
Goal in context:	User adds a question so that it appears in the user's profile page.
Preconditions:	There exists a question in the database
Scenario:	<ol style="list-style-type: none"> 1. User clicks on the Favourite button. 2. System appends the question to the user's list of Favourite questions. 3. Systems informs that the current question was added to their favourites list
Extensions:	1a. User clicks on the Favourite button again to remove the question from their favourites list. 1a.1 System removes the current question from the user's list of favourite questions.
Priority:	Essential

Use Case 5: User searches for content using Unified Search Bar

Primary actor:	User
Goal in context:	User wants to search for a question or a person.
Preconditions:	The user has opened the StudyBoard website.
Scenario:	<ol style="list-style-type: none"> 1. User clicks on the search bar. 2. User input the content that they want to see. 3. User clicks on the search button. 4. System displays the list of posts or people that matched the search keyword.
Extensions:	4a. There is no content with the given keyword. 4a.1 The system displays a text stating that nothing was found for the

	current search request.
Priority:	Essential

Use Case 6: User likes a reply

Primary actor:	User
Goal in context:	User thinks a reply is helpful and wants to promote the reply.
Preconditions:	The question has received one or more replies and the user is on the question page.
Scenario:	<ol style="list-style-type: none"> 1. User clicks on the like button. 2. System updates the 'like' count for the respective reply.
Extensions:	<ol style="list-style-type: none"> 1a. User has already 'liked' the reply. <ol style="list-style-type: none"> 1a.1 System negates the number of 'likes' for the respective reply. 2a. User is an endorsed user. <ol style="list-style-type: none"> 1a.1 System increments two to the like count.
Priority:	Essential

Use Case 7: User flags a reply as inappropriate

Primary actor:	User
Goal in context:	User finds a reply inappropriate and demands that the moderator removes it.
Preconditions:	User is viewing a specific thread.
Scenario:	<ol style="list-style-type: none"> 1. User clicks on the 'flag' button. 2. The system displays a pop-up window prompting the user to type in the reason for the flag. 3. User clicks on the 'Ok' button. 4. System informs the user that their flag was received via system alert. 5. System updates the list of flagged posts on the moderator's page.
Extensions:	<ol style="list-style-type: none"> 3a. User cancels the flagging process. <ol style="list-style-type: none"> 3a.1 User clicks the 'Cancel' button. 3a.2 The system cancels the process.
Priority:	Essential

Use Case 8: Moderator deletes a flagged reply

Primary actor:	Moderator
Goal in context:	Moderator deletes a flagged response in order to keep the platform safe.
Preconditions:	Moderator is logged in and is on the 'flagged replies page.
Scenario:	<ol style="list-style-type: none"> 1. Moderator clicks on his/her own profile page. 2. Moderator clicks on the flagged replies column. 3. System displays the list of flagged replies.

	4. Moderator reviews a specific reply and clicks delete. 5. System adds a notification to the writer of the deleted post.
Extensions:	3a. There are no replies to review 3a.1 System displays that there is no reply flagged as inappropriate.
Priority:	Essential

2.3 Non-functional Requirements

Performance Requirements

- The system should be able to handle an average of 100 concurrent users with a per-page response time under 2 seconds.
- The number of corresponding replies, likes, and flags should be updated across all elements in the UI as an event occurs.

Operating Constraints

- The platform should run on any modern browser (I.E. may be an exception).

Modifiability

- It should take less than five minutes to get an updated, error-free source code running in the web instance.

Reliability

- In case of any server failure, the average downtime should be less than five minutes.
- The server should be able to automatically restart the latest backup version of the web instance via NGINX upon failure.

Security

- Personal data should only be accessible by that specific user.

Usability

- Users with moderate experience with social networking services such as Facebook and Twitter should take less than 10 minutes to learn how to use the web platform comfortably.