Software Requirements Specification (SRS)
Discussion Board

Team: Group 5

Authors: Alex Scheufele, Tyler Lordan, Aidan Scribner, Cameron Gerossie, Bryan Tait
Customer: Educational Institutions
Instructor: James Daly

1 Introduction

This software requirements specifications(SRS) document outlines and documents the necessary details for the development of the proposed message board overhaul for Blackboard.

In this document, there will be detailed subsections on the following:

1.1 Purpose, 1.2 Scope, 1.3 Abbreviations, 1.4 Organization

2 Overall Description, 2.1 Product Perspective, 2.2 Product Functions, 2.3 User Characteristics, 2.4 Constraints, 2.5 Dependencies, 2.6 Apportioning Requirements

3 Specific Requirements

4 Modeling Requirements, 4.1 Use-Case Diagram, 4.2 Class Diagram, $.3 sequence Diagram, 4.4 State Diagram

5 Prototype, 6 References, 7 Contract

1.1Purpose

The purpose of this document is to establish a list of basic requirements between clients, users, stakeholders, and developers of our proposed message board. This document is for all the aforemention to review the project parameters and hereby follow the established outline for the creation, documentation and deployment of the product.

1.2Scope

The software product being developed is a message board for Blackboard. Blackboard is a Learning Management System(LMS) used to access and manage academic course content by both student and instructor alike. While Blackboard already has a discussion board feature from classroom feedback Blackboard was determined to be deficient in many areas, including its current implemented message board.

The benefits of the message board is for students to have the ability to network amongst each other. Students are allowed to create topics to lead discussions, and enable student collaboration. There will be several interactive forum feature, students can rate posts/threads they found helpful and instructors can endorse answers. This allows greater inactivity amongst students, and their respective instructors.

The objective is to eliminate the need for 3rd party websites like Piazza for student and instructor collaboration. This includes will allow students to seek help from each other for subjects covered in class and assignments. By removing the need for 3rd party sites the instructor can easily moderate the student discourse and endorse answers or questions.

The application domain of our product is a web server.

The message board will enable student and instructor to create posts which can be sorted by instructor created tags. Students can rate posts allowing the most pressing questions to be shown at the top. The instructor can also endorse an answer or question for it to be featured, so all accessing that thread will notice it.

1.3Definitions, acronyms, and abbreviations

Comment: A reply to a thread.

Discussion Thread: A topic that a student or instructor intends to discuss. Has a starting message to start the thread.

Dark mode: Dark colors for the background and light colors for the foreground i.e. white text on black background.

Endorse: A property that an instructor can give to a post. A post with this property will be given a higher priority than any post without it when a user views the discussion board.

HTTPS: Hypertext Transfer Protocol Secure, is the protocol used for secure communication over a computer network.

HTML: Hyper Text Markup Language, the standard markup language for documents to be displayed in a browser.

JSON: JavaScript Object Notation, is a human-readable data interchange format, to store and transmit object data.

LMS: Learning management system that hosts course content.

Netvotes: A counter that displays the total sum of user votes. Votes against or downvotes decrement the sum, while upvotes or votes in favor increment the sum.

Rating: A numerical property of posts, quantifying user approval or disapproval.

Post: A way to describe threads or comments.

Tag: A label describing a thread's content that is displayed next to a thread. Tags are used in sorting threads.

Reply: Another way of saying a Comment on a thread.

Forum: A feature in the current LMS we will replace. A top level container for threads. Forums have a title describing the content of contained threads.

1.4Organization

- **Section 2:** Overall description of project
    - 2.1 Product Perspective
    - 2.2 Product Functions
    - 2.3 User Characteristics
    - 2.4 Constraints
    - 2.5 Assumptions and Dependencies
    - 2.6 Apportioning of Requirements

- **<u>Section 3</u>:** Specific requirements for the project
- **<u>Section 4</u>:** Modeling requirements for the project
- **<u>Section 5</u>:** Prototype of the project
    - 5.1 How to Run Prototype
    - 5.2 Sample Scenarios
- **<u>Section 6</u>:** References used for the project
- **<u>Section 7</u>:** Point of Contact for the instructor

2 Overall Description

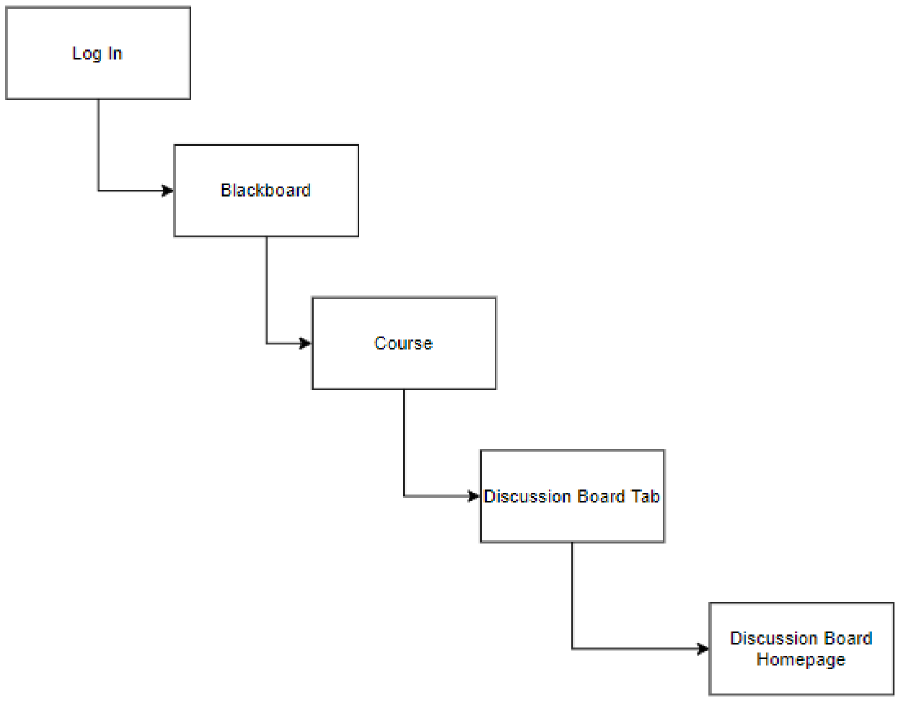
Section 2 will cover the context of the message board software and where it would be added in Blackboard. This section will also describe implementation details on how it will perform its functions. The expected user, constraints, assumptions and apportioning of requirements will also be listed.

2.1 Product Perspective

The discussion board is one subsection of a course in the LMS. The discussion board is a hub for communication between students and instructors. BlackBoard's current discussion board implementation suffers from poor organization and an unnatural layout. In specific, threads are only grouped by "Forum" with no ability to sort the forum list. Furthermore, the discussion board lacks the ability for students and instructors to provide feedback to a post without cluttering the thread with additional comments.

In developing a tool for students and instructors, the user interface must facilitate ease of use. This is the first interface constraint we face. Interacting with the discussion board must be intuitive for all users. From a technical viewpoint, an additional constraint is determined by the library we use to develop the front-end. Our ability to create a natural user experience will be constrained by this decision. The available storage of our web server will constrain the amount of data we can store, potentially limiting the amount of posts on the site. Storing data will be constrained to a JSON file storing the trees representing thread and reply structure. User operations are constrained to interacting with threads including but not limited to: creating posts, deleting posts, upvoting posts, downvoting posts, endorsing posts, replying to posts. Site operation is constrained by the necessity to present the user with an accurate, up to date, representation of the post data stored server side in the discussion board JSON file. A limitation of JSON is the inefficiency that arises when one single file becomes very large. Reading from and writing to the JSON file becomes timely, and issues of mutual exclusion can arise if many users are attempting operations concurrently. To address this a database may replace the JSON approach in a future release.

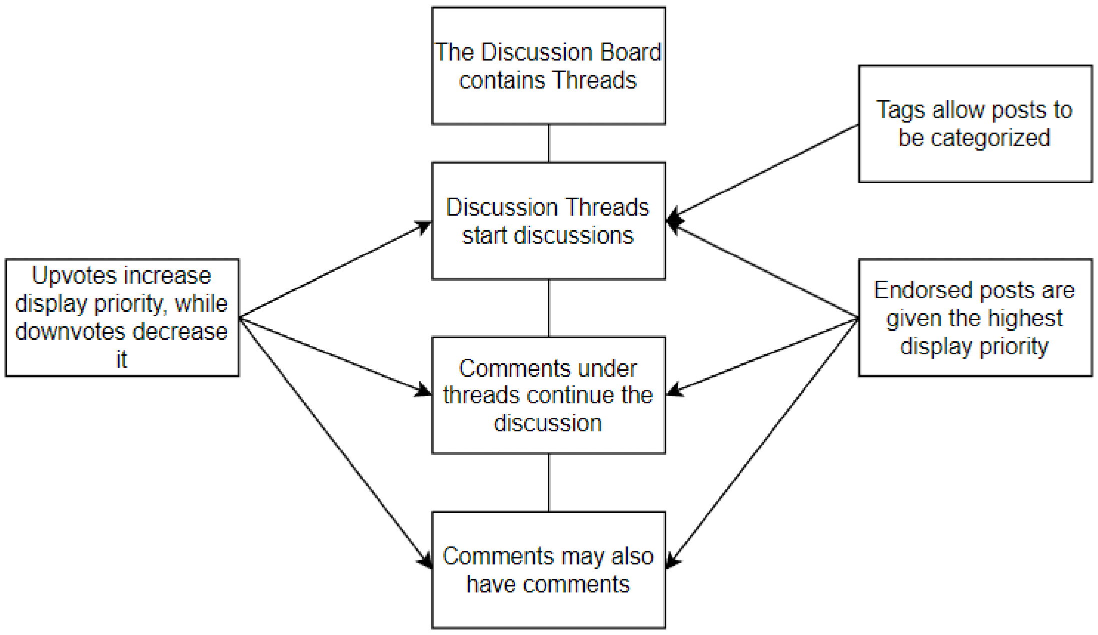
Our discussion board prototype will be accessible through any web browser supporting modern Javascript and HTML. Users will interact with the discussion board through the user interface using their computers display, keyboard, and mouse. The user will interface with the software through fillable text fields, buttons, and dropdown menus. The prototype will communicate with the user's web browser through the internet using the HTTPS protocol.


Log In
Blackboard
Course
Discussion Board Tab
Discussion Board Homepage

2.2Product Functions

Users will be able to access the discussion board of their class. On the discussion board, users can create new threads that contain user-specified text. The threads will have a title section and a body section. When creating a thread, Users are prompted to input text for both the title and the body, they can also apply a tag that is selected from a list of pre-defined tags. After inputting valid text and applying a tag, the user may confirm the creation of the new thread (they may also cancel the creation of the thread at any point).

After the thread is created, it will be visible on the discussion board, any user may interact with the thread by either upvoting, downvoting, or replying to it. If a user chooses to reply to the thread, they will be prompted for input and after they've entered it, they may confirm to create a new comment on the thread (They can also cancel the creation, similarly to threads). Upvoting will increase the rating of the thread, while downvoting will decrease it. Comments under threads may also receive upvotes, downvotes, or replies. Users may edit or delete posts that they've created. Instructors may also endorse posts. Instructors have the ability to edit or delete posts that have been created by any user.


The Discussion Board contains Threads
Tags allow posts to be categorized
Discussion Threads start discussions
Upvotes increase display priority, while downvotes decrease it
Endorsed posts are given the highest display priority
Comments under threads continue the discussion
Comments may also have comments

2.3User Characteristics

The expected user is an academic student or instructor. Proficient with navigating software interfaces and utilization of computers. Users are expected to create posts that are relevant to the material on the discussion boards for that course.

Users are expected to be students or instructors of an academic course. Users are proficient in using a web browser to access the internet and interact with websites.

2.4Constraints

The amount of content in the discussion board will be constrained by the decision to use JSON to store post content. Memory inefficiency increases as the file grows. Reading from and writing to the JSON file will take more time and resources the larger it is. If all user interaction triggers reading and writing to the same server side JSON file, issues of mutual exclusion may introduce threats to the integrity of the data.

Another restrictions will be expected forum/message board features may need to cut to fit within the few month development time.

The message board is not safety critical as no danger to life will be caused by program failure. However, there are failsafe features built into the product. If post modification or creation fails cancel post creation and do not modify the JSON to preserve the data integrity on file. As a precaution the inputs to the system will be sanitize to ensure proper performance. If cookie identifying an user is missing, reprompt the user to login.

2.5 Assumptions and Dependencies

It is assumed that the user is running a browser that is capable of running HTML5 and modern Javascript. In order to run the browser, it is also assumed that the user is on a computer that is capable of running the browser, and that they have an internet connection. In order to view and interact with the website, an internet connection, keyboard, mouse, and monitor are required.

2.6 Apportioning of Requirements

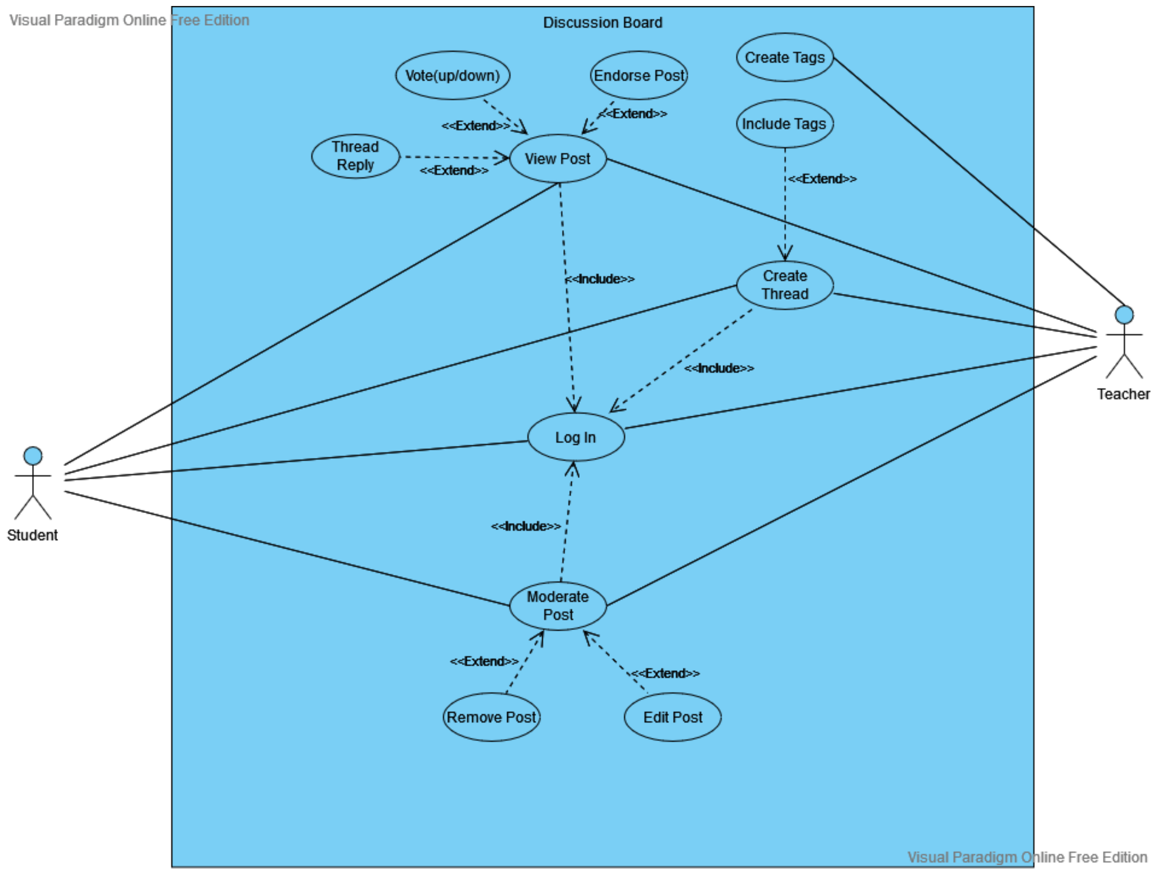
Three features will not be included initially, but are considered for future releases. Spelling and grammar correction for post content is the first. This feature was determined to be outside the scope of necessity and will not be included in the first release. Additionally the use of a database for content storage will be considered for future versions. The constraints of a JSON approach described in sections 2.1 and 2.4 make a good case for switching to a database approach. However, for the purposes of the initial release, JSON will suffice. Input sanitization is an important feature to filter profanity and inappropriate content. Though an important feature, it falls beyond the scope of necessity for the initial release and may be included in future releases

3 Specific Requirements

1. User-generated discussion board messages.
    1.1. Able to view posts on the discussion board.
    1.2. Able to post comments on discussion threads.
    1.3. Able to post images, files, and links on discussion threads.
    1.4. Original poster tag on thread creator.
2. Posts are ranked based on user interactions.
    2.1. Students can rate to posts
    2.2. Instructors can rate and endorse posts
        2.2.1. Endorsed messages are displayed under the title message
3. User thread management capabilities.
    3.1. Instructors can moderate posts.
        3.1.1. Instructors can edit any posts.
        3.1.2. Instructors can remove any posts.
    3.2. Users can create threads.
    3.3. The instructor creates tags to category-sort threads.
    3.4. Students can select relevant tags on thread creation.
4. Aesthetic requirements for specifying the appearance of the message board.
    4.1. The default display mode is dark mode.
    4.2. Tags are color-coded to differentiate them.
    4.3. Instructor tag to identify a user as an instructor.
    4.4. Student tag to identify a user as a student.
    4.5. Student name and image is displayed above their posts.
5. How user information will be obtained, stored, and used.
    5.1. Home page for users to login into the discussion board.
    5.2. Cookies to store login information of users.

4 Modeling Requirements

Use-Case Diagram



Use Case Name:	Log In
Actors:	Student and Instructor
Description:	The user is prompt to login to gain access to the discussion board for the specific course
Type:	Primary
Includes:	None
Extends:	None
Cross-refs:	Requirement 5.1
Uses cases:	N/A

Use Case Name:	Create Thread
Actors:	Student and Instructor
Description:	The user creates a new thread to discuss a topic related to something in the class
Type:	Primary
Includes:	Log In
Extends:	None
Cross-refs:	Requirement 3.2
Uses cases:	User must have logged into the system

Use Case Name:	Include tags
Actors:	Student and Instructor
Description:	Attaches a tag to the thread to allow for better organization of the threads
Type:	Secondary
Includes:	None
Extends:	Create Thread
Cross-refs:	Requirement 3.4
Uses cases:	User must have logged into the system and selected to create a thread

Use Case Name:	Create Tags
Actors:	Instructor
Description:	The user can create a new tag that can be used to be attached to threads
Type:	Secondary
Includes:	None
Extends:	None
Cross-refs:	Requirement 3.3
Uses cases:	User must have logged into the system as an instructor

Use Case Name:	View Post
Actors:	Student and Instructor
Description:	Allows the user to view a thread and all of the comments associated with it
Type:	Primary
Includes:	Log In
Extends:	None
Cross-refs:	Requirement 1.1
Uses cases:	User must have logged into the system

Use Case Name:	Thread Reply
Actors:	Student and Instructor
Description:	A user can reply to a thread with a comment, allowing users to communicate to each other about a thread topic
Type:	Secondary
Includes:	None
Extends:	View Post
Cross-refs:	Requirement 1.2
Uses cases:	User must have logged into the system and selected to view a post

Use Case Name:	Vote(up/down)
Actors:	Student and Instructor
Description:	Allows users to vote a thread or comment in a thread up or down depending on if they found it helpful or not
Type:	Secondary
Includes:	None
Extends:	View Post
Cross-refs:	Requirement 2.1
Uses cases:	User must have logged into the system and selected to view a post

Use Case Name:	Endorse Post
Actors:	Instructor
Description:	Allows for the Instructor to endorse a post that they find is useful/helpful for the students to read
Type:	Secondary
Includes:	None
Extends:	View Post
Cross-refs:	Requirement 2.2
Uses cases:	User must have logged into system as an instructor and selected to view a post

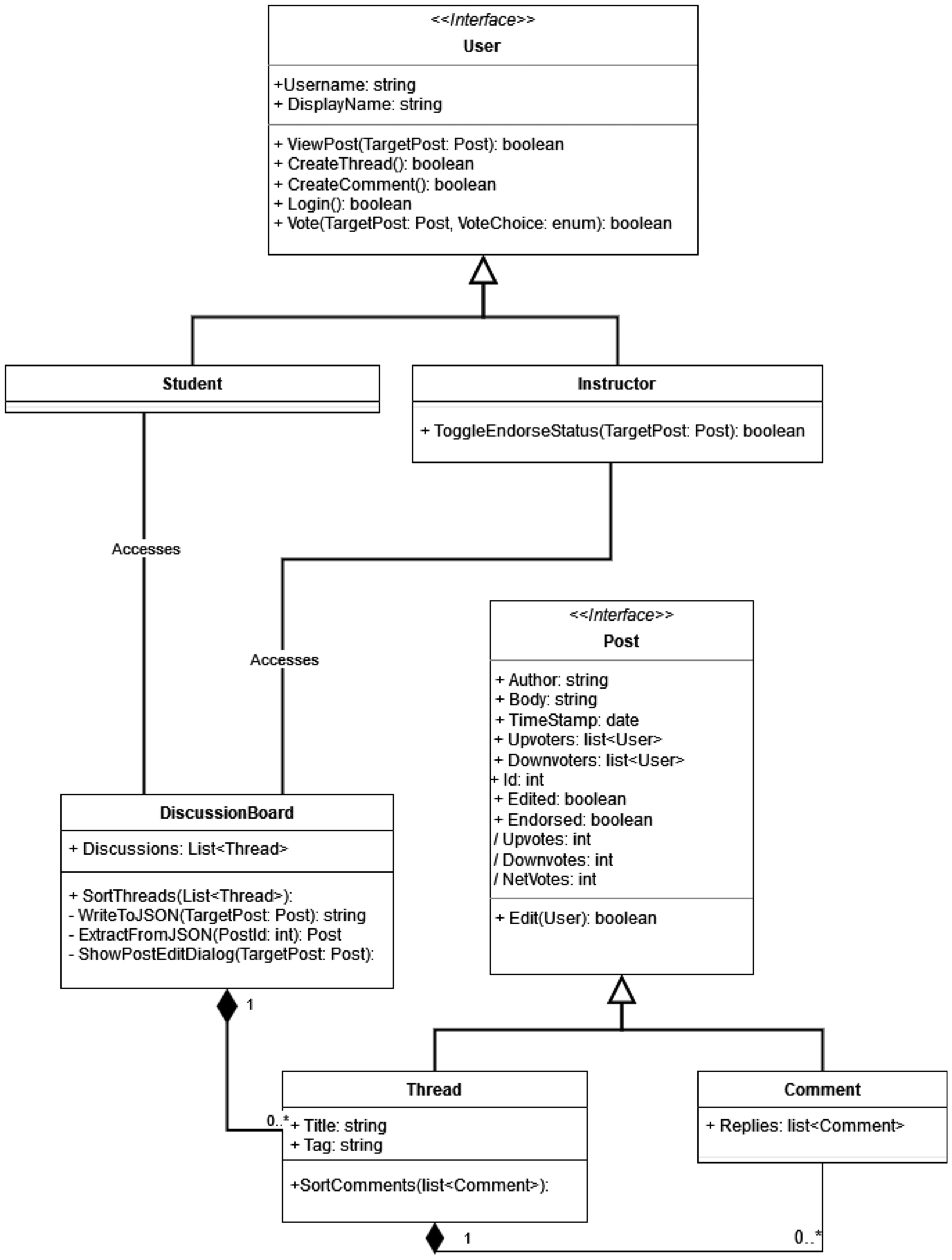
Use Case Name:	Moderate Post
Actors:	Student and Instructor
Description:	Allows for a student to change their own posts and allows for a instructor to change any of the posts
Type:	Secondary
Includes:	Log In
Extends:	None
Cross-refs:	Requirement 3.1
Uses cases:	User must have logged into system as an instructor and selected to view a post

Use Case Name:	Remove Post
Actors:	Student and Instructor
Description:	Allows for students to remove their posts and for instructors to remove any post
Type:	Secondary
Includes:	None
Extends:	Moderate Post

Cross-refs:	Requirement 3.1.2
Uses cases:	User must have logged into system as an instructor and selected to view a post

Use Case Name:	Edit Post
Actors:	Student and Instructor
Description:	Allows for a student to edit their own post and a instructor can edit any of the posts
Type:	Secondary
Includes:	None
Extends:	Moderate Post
Cross-refs:	Requirement 3.1.1
Uses cases:	User must have logged into system as an instructor and selected to view a post

Class Diagram



Data Dictionary:

Element Name		**Description**
User		This interface represents a user of the discussion board. It defines the attributes and methods of Students and Instructors
Attributes	Username: String	This is the information for the user to use to login to the system
	DisplayName: String	This is the name that other students will see when the user make a post on the discussion board
Methods	ViewThread(Thread): boolean	This method allows for the user to view a specific thread
	Vote(TragetPost: Post): boolean	Allows for the user to upvote or downvote a post on the discussion board
	CreateComment(): boolean	This method allows for users to create a comment on a thread.
	Login(): boolean	This method allows the user to login.
	CreateThread(): boolean	This method allows for the user to create a new thread on the discussion board
Relationships		The interface is implemented by the Student and Instructors classes

Element Name	**Description**
Student	This class represents a Student user that

	will use the discussion board to create, view, and vote posts
Relationships	This class implements the User interfaces attributes and methods. It also accesses the discussion board

Element Name		**Description**
Instructor		This class represents a instructor user that will use the discussion board to create, view, vote, endorse, and moderate any posts
Methods	ToggleEndorseStatus(TargetPost: Post): boolean	Flips the current endorse status of a post.
Relationships		This class implements the User interfaces attributes and methods. It also accesses the discussion board

Element Name		**Description**
DiscussionBoard		This class represents the discussion board system and contains all of the threads organized in a certain order
Attributes	Discussions: List<Thread>	This attribute contains the list of all of the threads in the discussion boards
Methods	SortThreads(List<Thread>)	This method allows for the discussion board to organize the list of threads in a certain way
	WriteToJSON(TargetPost: Post):string	Storing Post information in the JSON file as a string.
	ExtractFromJSON(	Building the post object from the JSON

	postId: int): Post	file information, using an unique postId to identify it.
	ShowPostEditDialog(TargetPost: Post):	During post creation or editing show an edit dialog box to edit the text fields of the post
Relationships		This class is accessed by the users and it contains a list of the threads inside of it and organizes them.

Element Name		**Description**
Post		This interface represents a post on the discussion board. It defines the attributes and methods for the Thread and Comment classes
Attributes	Author: String	This attribute stores the name of the user who made the post
	Body: String	This attribute holds the body text of the post
	TimeStamp: date	This attribute holds the time that the post was made
	Upvoters: list<User>	This attribute holds the list of people who upvoted the post
	Downvoters: list<User>	This attribute holds the list of people who downvoted the post
	Edited: boolean	This attribute holds state of whether the post has been edited after being posted
	Endorsed: boolean	This attribute holds state of whether the post has been endorsed or not by an instructor
	Id: int	A unique identifier for the post

	Upvotes: int	This attribute holds the number of upvotes the post has received
	Downvotes: int	This attribute holds the number of downvotes the post has received
	NetVotes: int	This attribute holds the total number of votes the post has received
Methods	Edit(User): boolean	This method takes in a user and checks whether they can edit it or not and then returns true if the post is successfully edited
Relationships		This interface is implemented by the Thread and comment classes

Element Name		**Description**
Thread		This class represents a thread in the discussion board and it holds the comments and organizes them
Attributes	Title: String	This attribute holds the title of the thread that will be displayed on the discussion board to users
Methods	SortComments(list<Comment>)	This method sorts the comments associated with this thread
Relationships		This class implements the post interfaces attributes and methods. It also contains multiple comments that serve as replies to the thread.

Element Name	**Description**
Comment	This class represents a reply to a thread

		and it holds sub-comments inside of it.
Attributes	Replies: list<Comment>	Contains the replies to this comment in the form of a list of sub comments
Relationships		This class implements the post interfaces attributes and methods. It contains subcomments that serve as replies to itself

Sequence Diagrams:

Diagram 1: User creates a thread on the discussion board

The user logs into the discussion board for their class. They create a new thread which calls the CreateThread() method in the DiscussionBoard object. This then displays a dialog box to edit the body of the thread by calling the method ShowPostEditDialog() which returns the Contents. Depending on what Contents returned as the system operates differently. If Contents is null then the system cancels out of creating a thread by returning CreateThread() as false. If Contents isn't null then the system stores the Contents in a new Thread object and returns the new set up of Thread object to the DiscussionBoard. The DiscussionBoard then converts the Thread to be stored in a json and stores it, it does this by calling the WriteToJson() method. Finally the DiscussionBoard returns true from the CreateThread() function to signify the Thread has been posted to the system. The Thread object is destroyed.

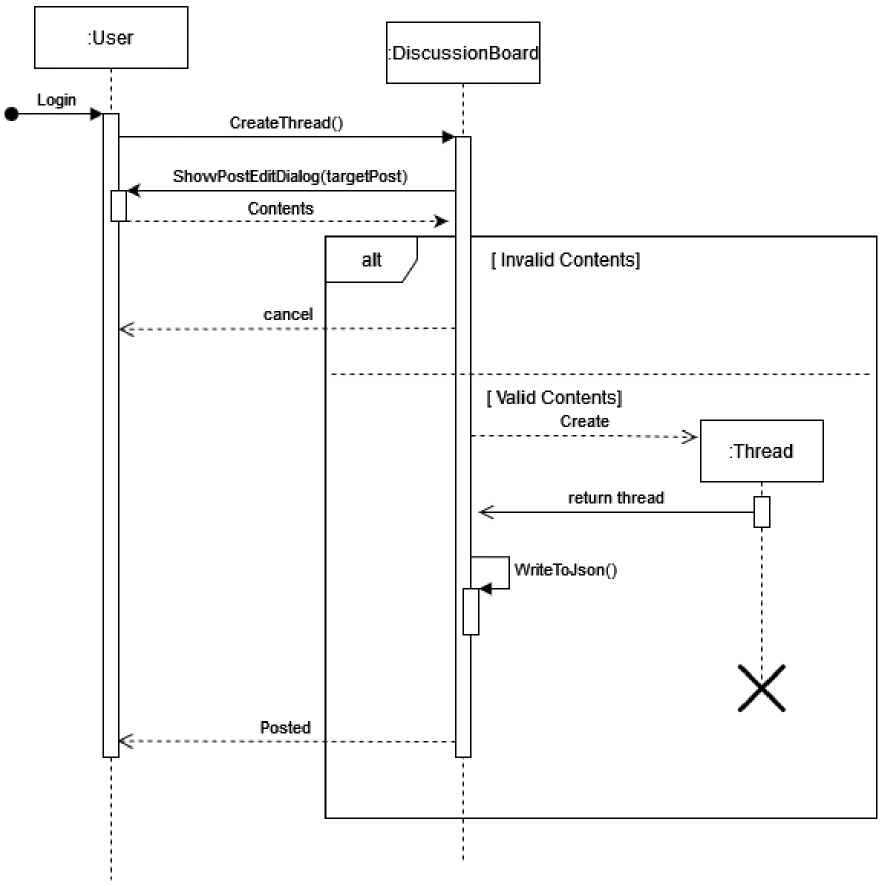
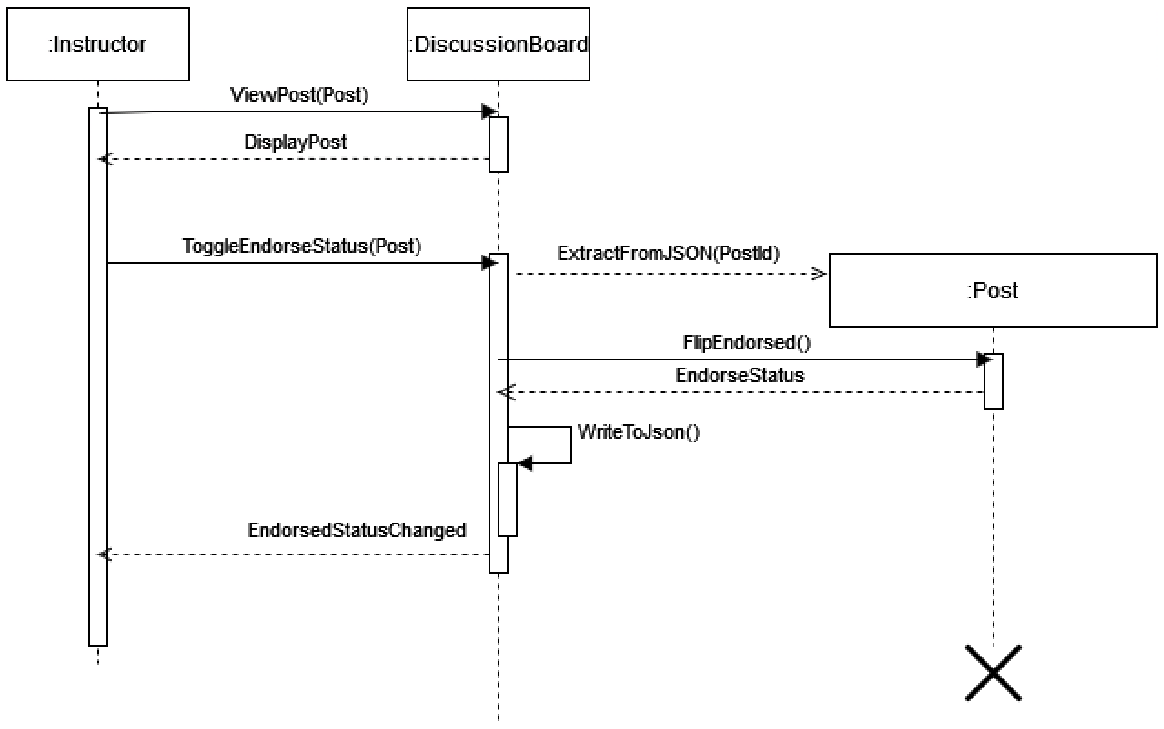
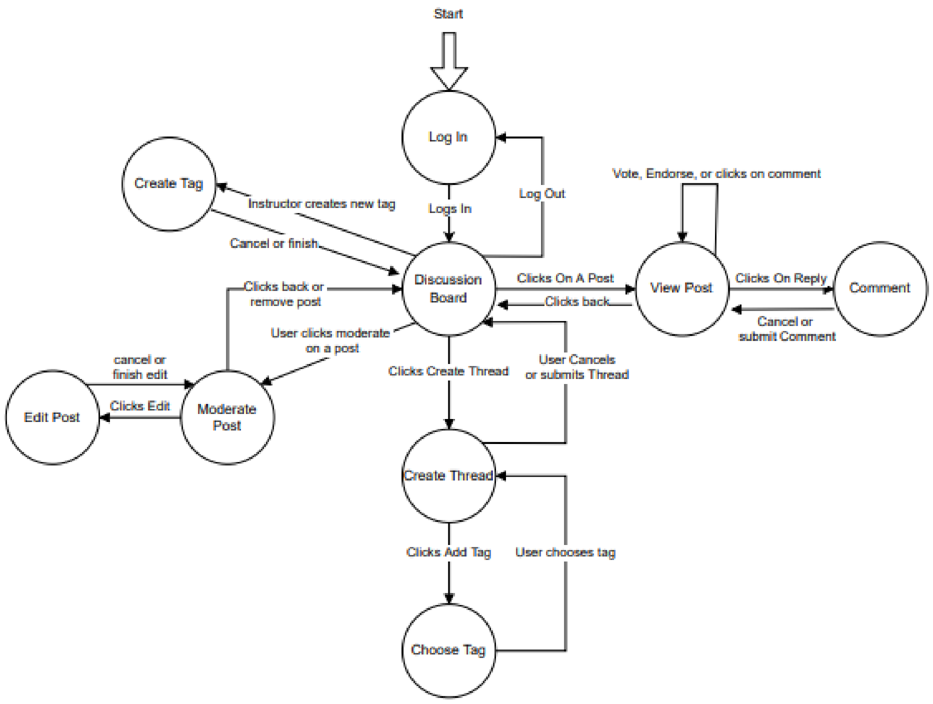


Diagram 2: An instructor endorses/unendorses a post on the discussion board

Once the instructor has logged into the system, they click on a post to view it, calling the ViewPost() method which then returns true if the post displays to the screen. Once on the post the instructor then endorses or unendorses a post by clicking on the endorse button, this calls the ToggleEndorseStatus() method. Then the ExtractFromJSON() method is called which grabs the post to be changed from the JSON file. It then calls the FlipEndorsed() method to flip the endorsed status of the post to the opposite state and returns the state. Finally the method WriteToJson() is called to update the post in the JSON file, and true is returned back to the instructor from the ToggleEndorseStatus() method to signify the state was changed. The post object is destroyed.


:Instructor
:DiscussionBoard
ViewPost(Post)
DisplayPost
ToggleEndorseStatus(Post)
ExtractFromJSON(PostId)
:Post
FlipEndorsed()
EndorseStatus
WriteToJson()
EndorsedStatusChanged

State Diagram



5 Prototype

The prototype will show a working message board. The board will include several clickable boxes, that either adjust displayed numerical values on posts or allow users to enter information in a text field for post creation. During thread creation there will be a menu to select relevant tags to categorize the thread. Users can edit their own posts after creation, while instructors can edit any post.

5.1How to Run Prototype

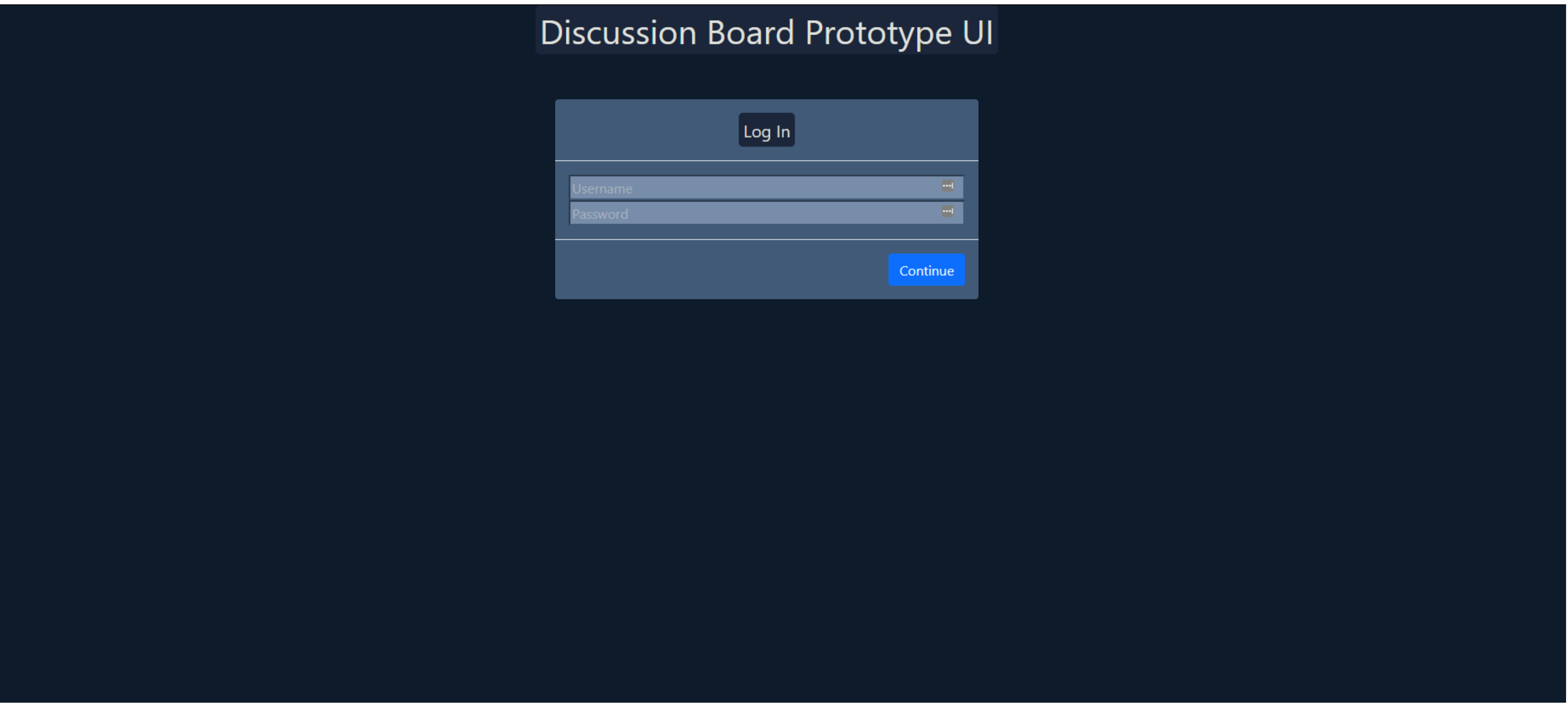
System requirements:

- Internet connection
- HTML5 and Javascript capable browser
- Keyboard and mouse

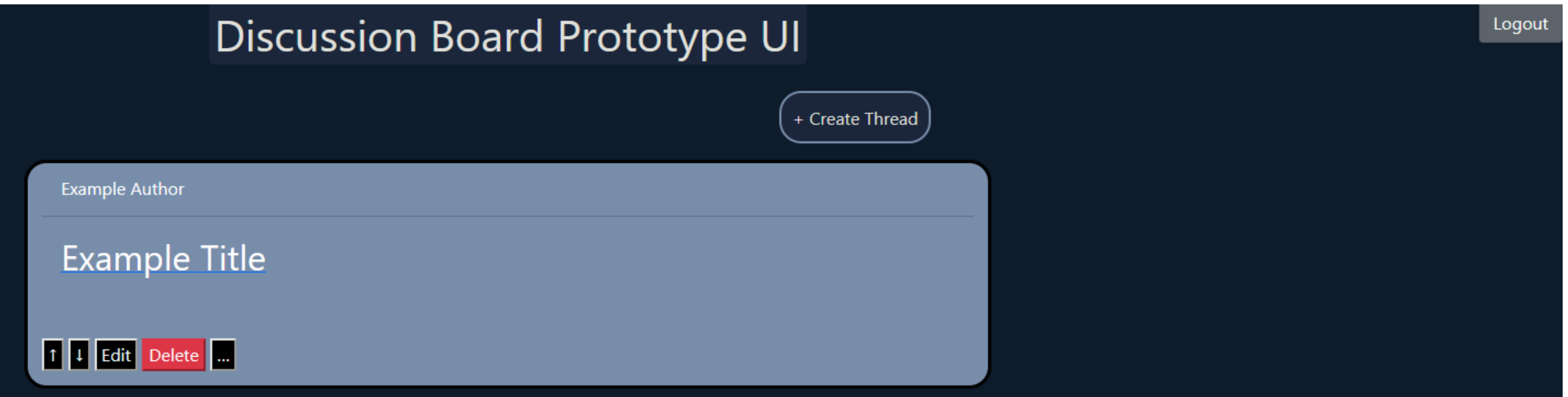
The prototype is a three html pages with accompanying css files. To run the prototype simply go to it directly through the link:
https://bryan0x05.github.io/software_eng_I_project/PrototypeUI/startpage.html
or the prototype can be accessed from the team website[2] by clicking the “prototypes” link towards the bottom of the page.
https://bryan0x05.github.io/software_eng_I_project/

5.2Sample Scenarios

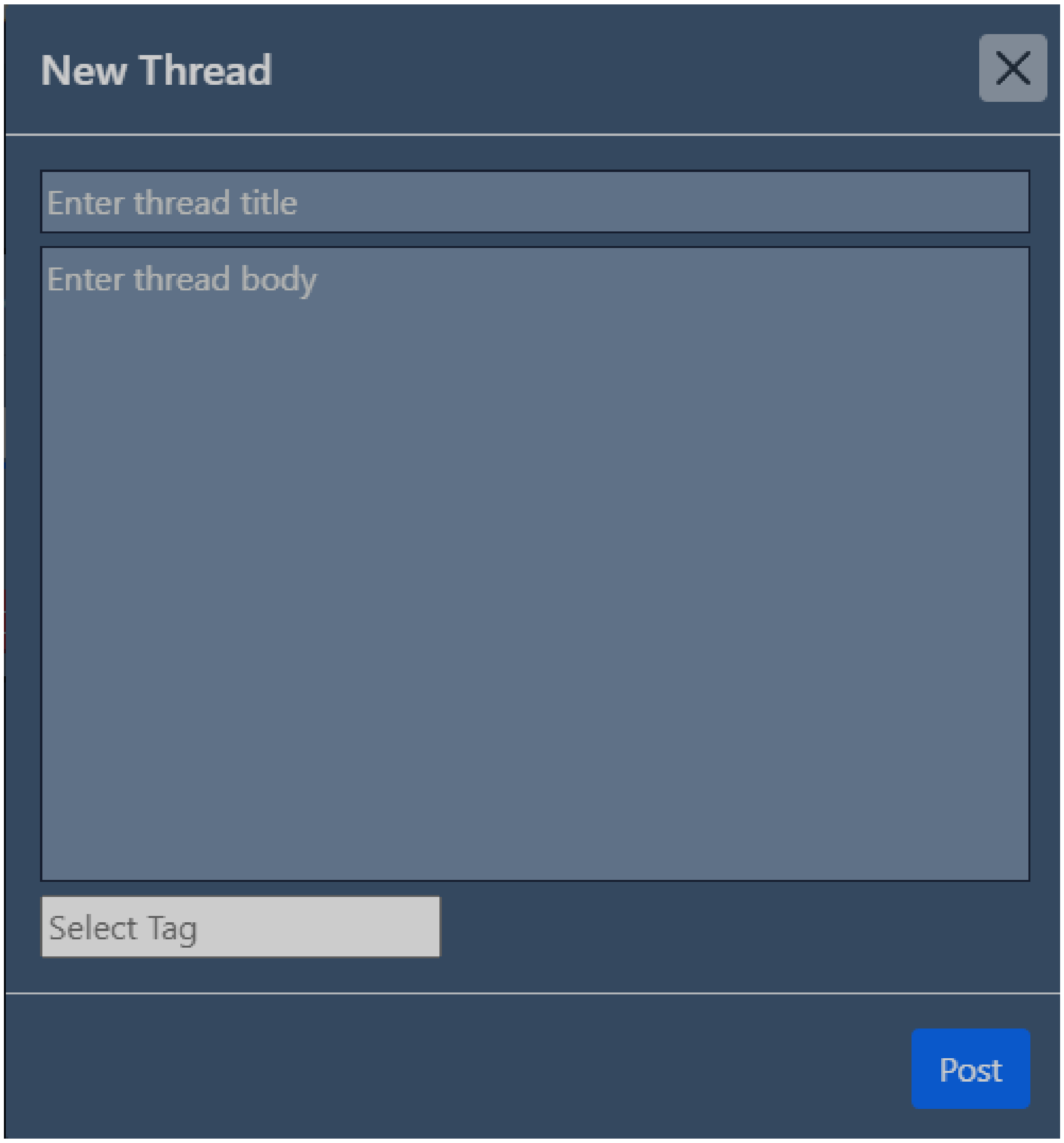
The user can log into the message board through the login page. For the sake of the prototype UI hitting continue will immediately bring the user to the discussion board.

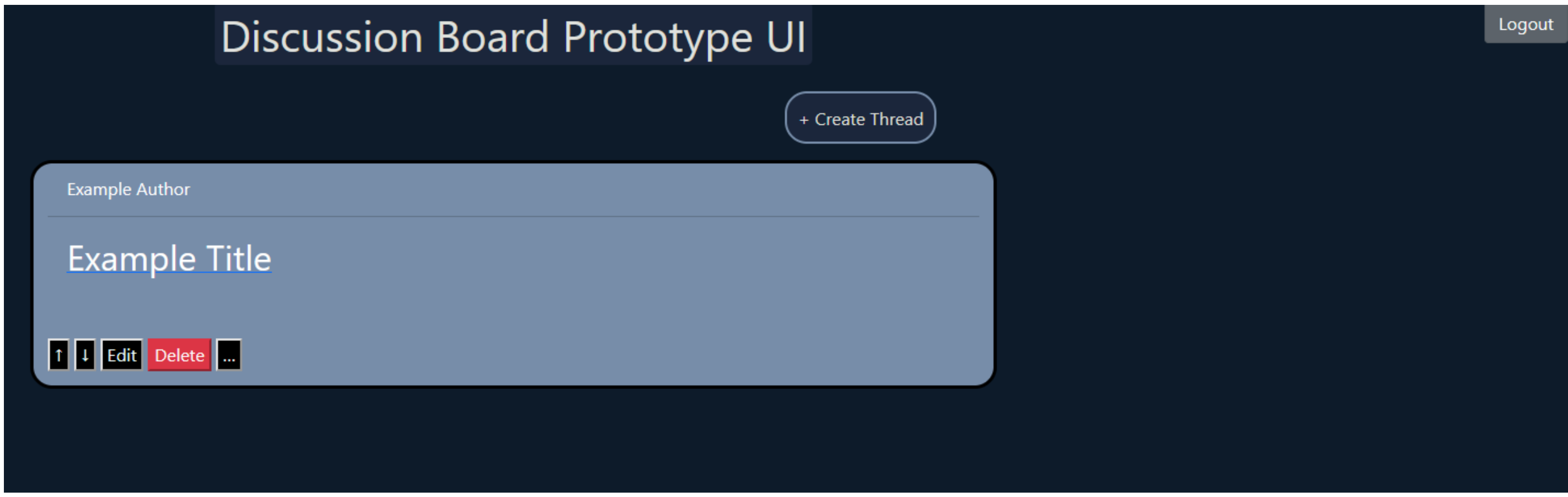


On the main page, the User can create a new thread by clicking the create thread button

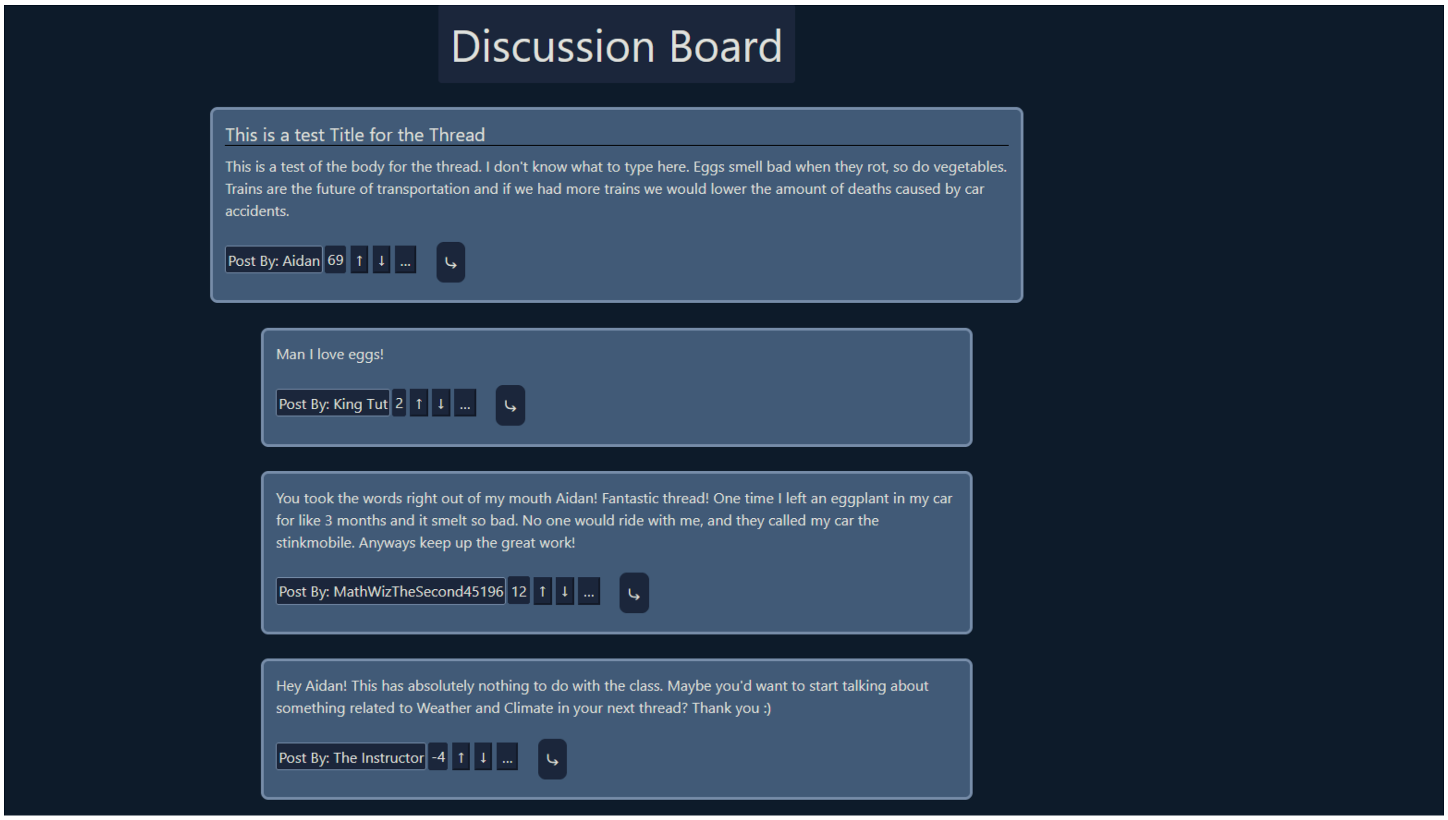


After clicking the create thread button, the thread creation modal is displayed and the user can input text into the text areas, they can also select a tag. When the user is done entering text, they can click post to post the content (not implemented in prototype_v1). If the user wants to cancel the creation, they can click the X button at the top, which brings them back to the main page. The main page will only display thread titles and their respective authors.The User can click on the thread title to view it the thread and it's replies.





Upon clicking "Example Title" in this scenario the user is greeted with this thread. Here the user can see the thread body, and it's replies. Going left to right the author's name is visible, netvotes counter, up/down vote, moderate(instructor only unless it is the user's own post) and reply buttons. In this example the user would like to make a reply, and click the rightmost button.



This is a test Title for the Thread

This is a test of the body for the thread. I don't know what to type here. Eggs smell bad when they rot, so do vegetables. Trains are the future of transportation and if we had more trains we would lower the amount of deaths caused by car accidents.

Post By: Aidan 69 ↑ ↓ ...

Man I love eggs!

Post By: King Tut 2 ↑ ↓ ...

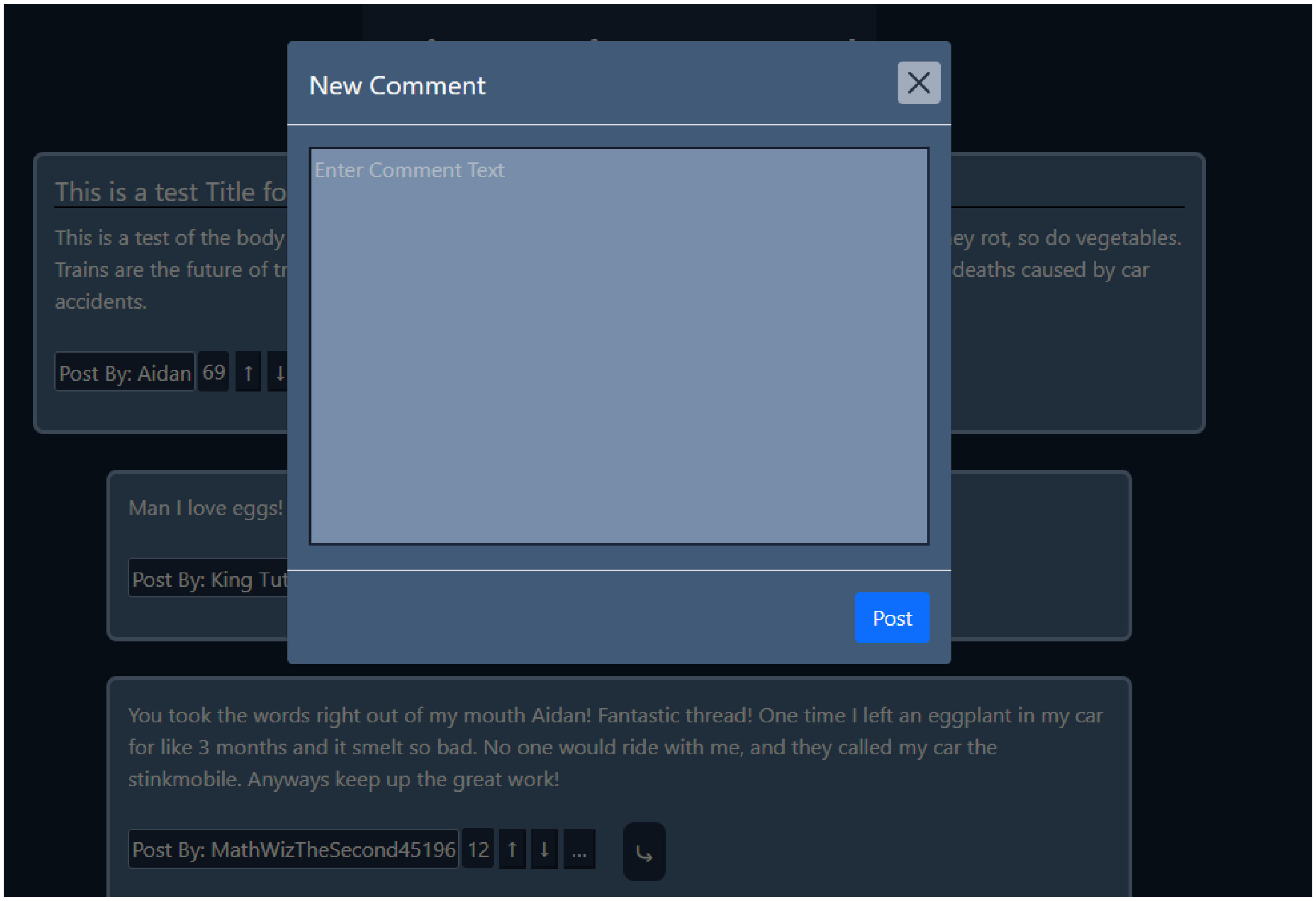
You took the words right out of my mouth Aidan! Fantastic thread! One time I left an eggplant in my car for like 3 months and it smelt so bad. No one would ride with me, and they called my car the stinkmobile. Anyways keep up the great work!

Post By: MathWizTheSecond45196 12 ↑ ↓ ...

Hey Aidan! This has absolutely nothing to do with the class. Maybe you'd want to start talking about something related to Weather and Climate in your next thread? Thank you :)

Post By: The Instructor -4 ↑ ↓ ...

Upon clicking the reply button this is a comment creation modal where the user can put in the text to be displayed.



This is a test Title fo

This is a test of the body ... ey rot, so do vegetables. Trains are the future of tr ... deaths caused by car accidents.

Post By: Aidan 69 ↑ ↓

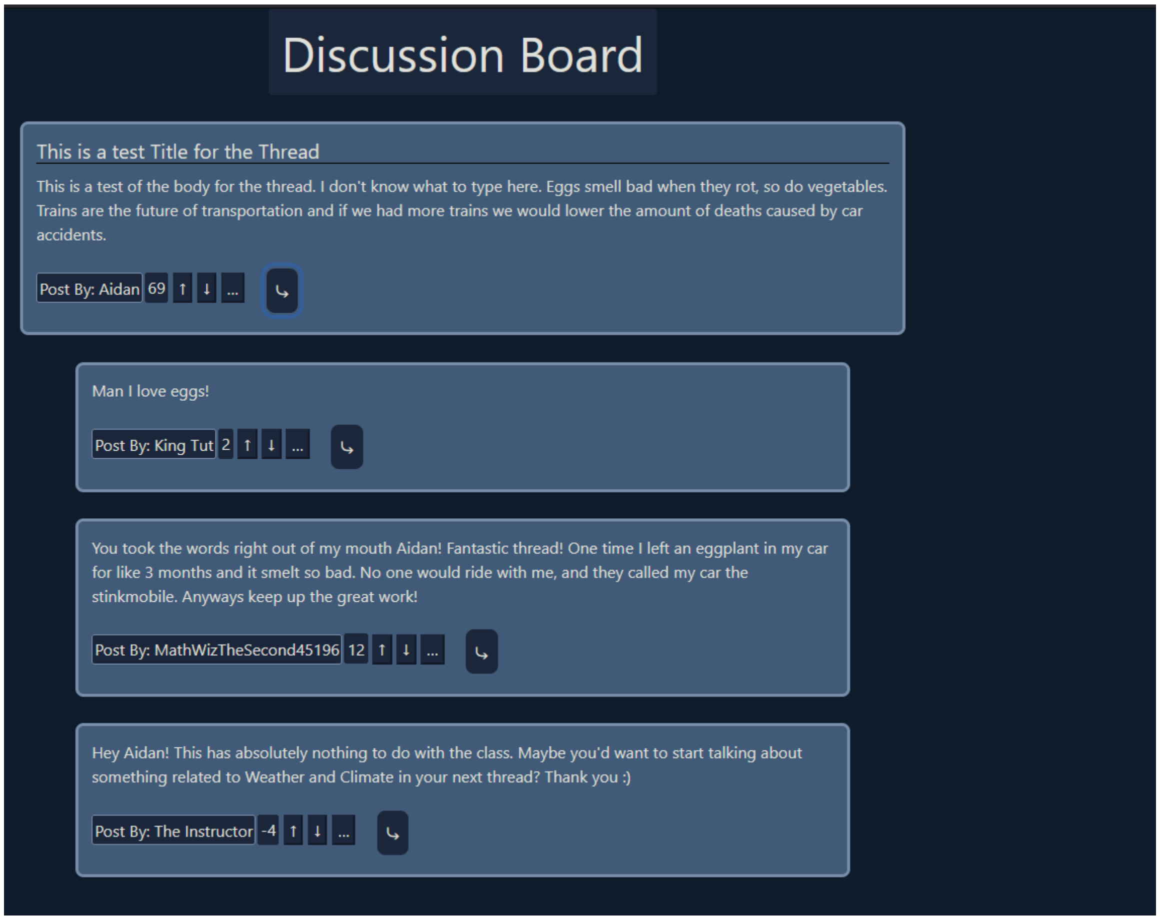
Man I love eggs!

Post By: King Tut

You took the words right out of my mouth Aidan! Fantastic thread! One time I left an eggplant in my car for like 3 months and it smelt so bad. No one would ride with me, and they called my car the stinkmobile. Anyways keep up the great work!

Post By: MathWizTheSecond45196 12 ↑ ↓ ...

When the the user is done, they can click on the big text “Discussion Board” to return to the main discussion board page.



This is a test Title for the Thread

This is a test of the body for the thread. I don't know what to type here. Eggs smell bad when they rot, so do vegetables. Trains are the future of transportation and if we had more trains we would lower the amount of deaths caused by car accidents.

Post By: Aidan 69 ↑ ↓ ...

Man I love eggs!

Post By: King Tut 2 ↑ ↓ ...

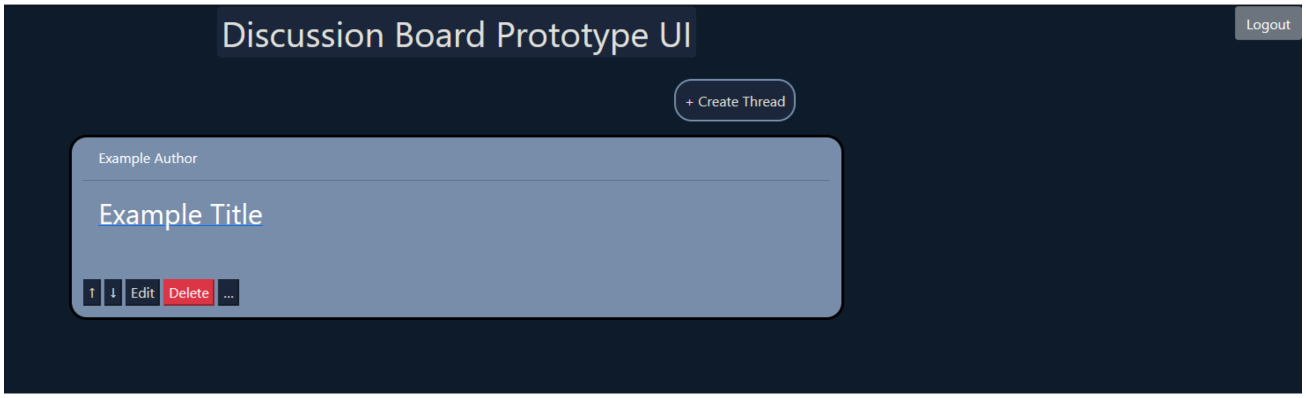
You took the words right out of my mouth Aidan! Fantastic thread! One time I left an eggplant in my car for like 3 months and it smelt so bad. No one would ride with me, and they called my car the stinkmobile. Anyways keep up the great work!

Post By: MathWizTheSecond45196 12 ↑ ↓ ...

Hey Aidan! This has absolutely nothing to do with the class. Maybe you'd want to start talking about something related to Weather and Climate in your next thread? Thank you :)

Post By: The Instructor -4 ↑ ↓ ...

In this example the user is done interacting with the discussion board. So the user can click the logout button in the upper right to leave the discussion board and logout.



6 References

Start of your text.

[1] D. Thakore and S. Biswas, “Routing with Persistent Link Modeling in Intermittently Connected Wireless Networks,” Proceedings of IEEE Military Communication, Atlantic City, October 2005.

[2] Scribner, Aidan, et al. “Discussion Board.” *Discussion Board Project*, 1.0, Team 5, 9 Nov. 2022, https://bryan0x05.github.io/software_eng_I_project/. Accessed 11 Nov. 2022.

[3] J. Thornton and M. Otto, “Get started with bootstrap,” *Bootstrap v5.2*. [Online]. Available: https://getbootstrap.com/docs/5.2/getting-started/introduction/. [Accessed: 16-Nov-2022].

7 Point of Contact

For further information regarding this document and project, please contact **Prof. Daly** at University of Massachusetts Lowell (james_daly at uml.edu). All materials in this document have been sanitized for proprietary data. The students and the instructor gratefully acknowledge the participation of our industrial collaborators.