Computer Science Central WEB APP

ANALYSIS AND DESIGN DOCUMENT

Team 8

Joseph Rodriguez

Louis Rivera

Rafel Salem

CONTENTS

INTRODUCTION

Purpose: The purpose of this document is to detail the design decisions involved with making the Computer Science Central Web App. This document will serve as a guide for developers during the construction of the app . Design decisions for the app are outlined in this document and should be followed and referenced during development and for any future update of the app. This document can also serve as a verification tool for any quality persons to ensure that the final product meets the goals set forth by this document.

Scope: The following will be included in the first iteration of the web application:

- Account creation page where users can create accounts for the website.
- Login section that allows user to login to previously created accounts.
- Profile view page that allows users to view and change their account information.
- Video library that will allow users to post and view video content.
- Algorithm knowledge base that will users to post and view coding files.
- Community forum that will be a forum where users can communicate via text posts.

An accompanying database will need to be developed for the site. The database will house account information, programming files and videos, and forum posts so that this information can be pulled up by users when it's needed.

The video library will contain videos created by admins. The videos will be relevant to common programming issues that students may face.There will be functionality that allows users to search for videos by name or topic covered / programming language the video is relevant to. User will also have the ability to comment on a video asking for further assistance. The user may edit or delete their own comments.

The algorithm knowledge base will allow users to ask questions and comment on a specific algorithms solution. These solutions will be posted and chosen by admin in order to select the most common algorithm problems. The user will be given the ability to edit or delete their own comments.

Lastly, The community forums will be able to handle user comments, edits to their own posts and rating of other’s comments. The user will not be able to edit or delete other user posts; this is a an admin functionality.

Overview: The remainder of this document is laid out in the follow manner. First is an overall description of the web application that will cover product prescriptive, product functions, user characteristics, constraints, assumptions and dependencies. Next major topics that will be covered are functional requirements, user requirements, system requirements, system models, and non-functional requirements, respectively. Lastly, the appendix and index will follow.

- Functional requirements will contain the services to be provided to the user.The functional requirements will utilize UML diagrams for each user interaction. User requirements will be cover in this section.The system requirements will go a little further is discussing functional and non-functional requirements with a more refined description of user requirements.Followed by the last functional requirement which is system model. This is where the document will display relationships between the system components, the system and it environment.

  - Within the non-functional portion of this document there will be the constraints, performance, security, usability, and maintainability requirements.

- *Definitions and Acronyms:*

  - The Site / The website: This is always referring to website that is being developed, ComputerScienceCentral.
  - Users: Anyone who is on the site and taking advantage of the site's functionality.
  - Admin / Administrator: A site user that has an account with administrative privileges that allow them to access additional site functionality. This functionality usually involves managing content that is added by users (i.e. deleting and modify content).
  - CS - Computer Science
  - UML - Unified modeling language
  - Algorithm - a well refined way to solve a computer problem (through programming).
  - IDE - stands for integrated development environment and is typically software that helps programmers write programs.
  - Programming language - is a language that is used to write instructions in for execution.
  - Source Code - Any collection of computer instructions, possibly with comments, written using a human-readable programming language, usually as plain text.
  - CRUD - Create, Read, Update, Delete as a function of the database.

SYSTEM ARCHITECTURE

SYSTEM OVERVIEW

This product will have a few parts, the client or the user interface will interact with the web application to request pages. The server will display the pages with the correct information from the database. As the client requests a certain page, the server will retrieve the necessary information for that particular page. It will do so by pulling the required information from the database using server side code and display everything requested to the client side view. This will create a completely dynamic web application.

This project is data-centric, therefore, the database will be used to store data. This data will contain the following: username, passwords, comments, forum posts, and video posts. The client and the server will both interact with database to obtain the correct information. The server will handle all passing of data to the user views. This is where the heart of the web application will live.

It is intended for this to be a combination of Stackoverflow with a Khanacademy twist. We would like to bring together the community aspects of Stackoverflow where users can ask questions and upvote their favorite answer. Khan Academy provides educational videos for students our video library should handle all of the educational videos with user feedback.

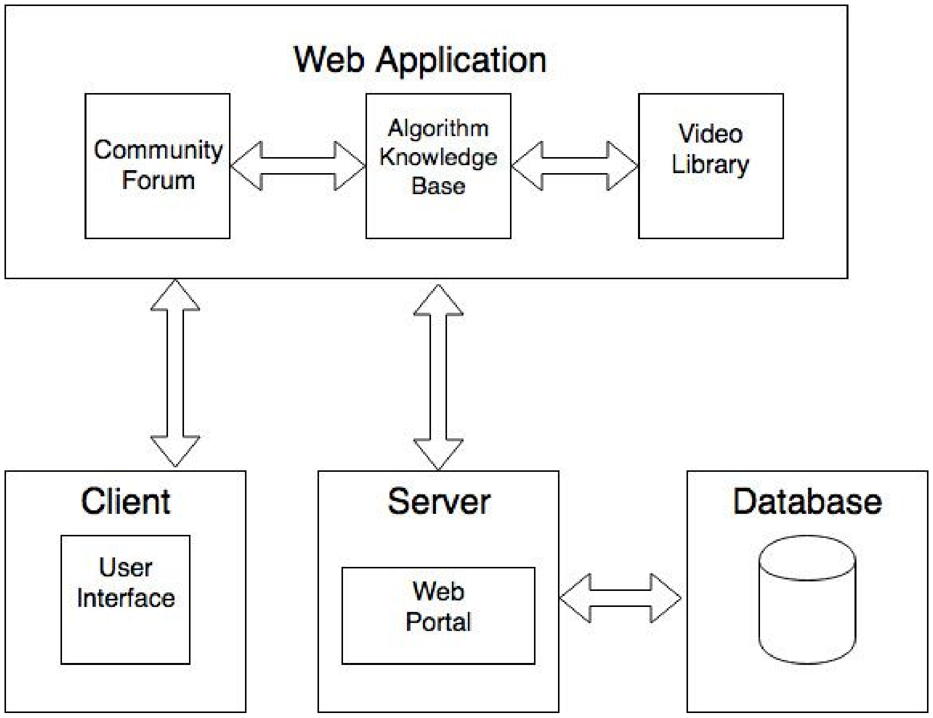
The web application will have a graphical user interface for the user to interact with. The user interface will be created with simplicity in mind. It will have consistent colors throughout the web application to maintain a professional look. The web application will contain buttons and textboxes for the user to interact with. The hardware interfaces will include the a mouse and a keyboard to obtain input from the user. The user can click on buttons and type content into textboxes.

ARCHITECTURAL DESIGN

Computer Science Central will be a web application that utilizes client server application. It will be a multilayered architecture and broken into 4 layers: Model, View, Controller, and Data. As depicted by the diagram below the client will interact with the view then passing data to the controller which determines which set of business logic or data that is required to be sent to the user.

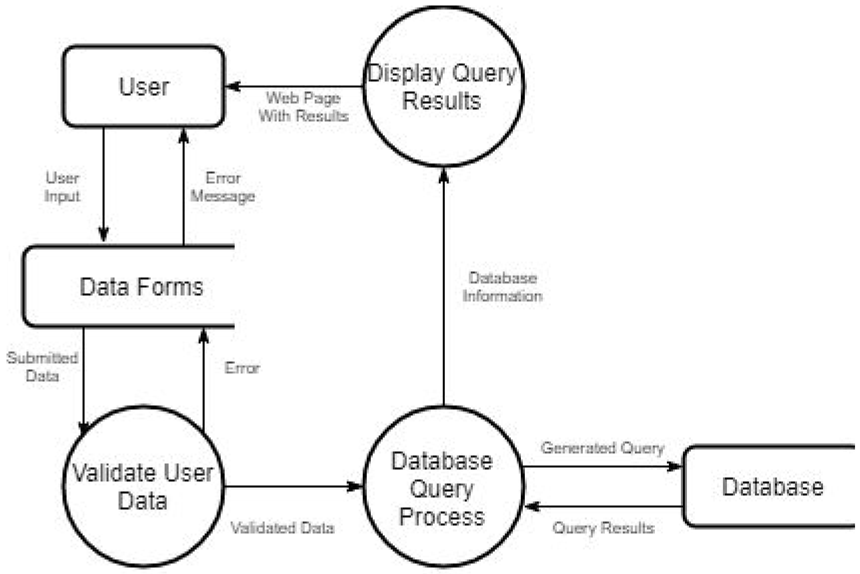
As noted above, the system will be divided into 4 subsystems. The diagram below gives a high level example of how the layers interact with each other.

- **Model:** The model layer will end up modeling the data from the database. The model layer will also contain all of the business logic held within the web application itself.
- **View:** The view is portion of the web application in which the user directly interacts with. This layer will utilize route or GET and POST actions to send and receive data from the controller.
- **Controller:** The controller will act as the messenger within the web application. Its sole responsibility is to handle communication between the user (View layer) and the Model.
- **Data:** The data layer will be where all of the data utilized within the application is stored.



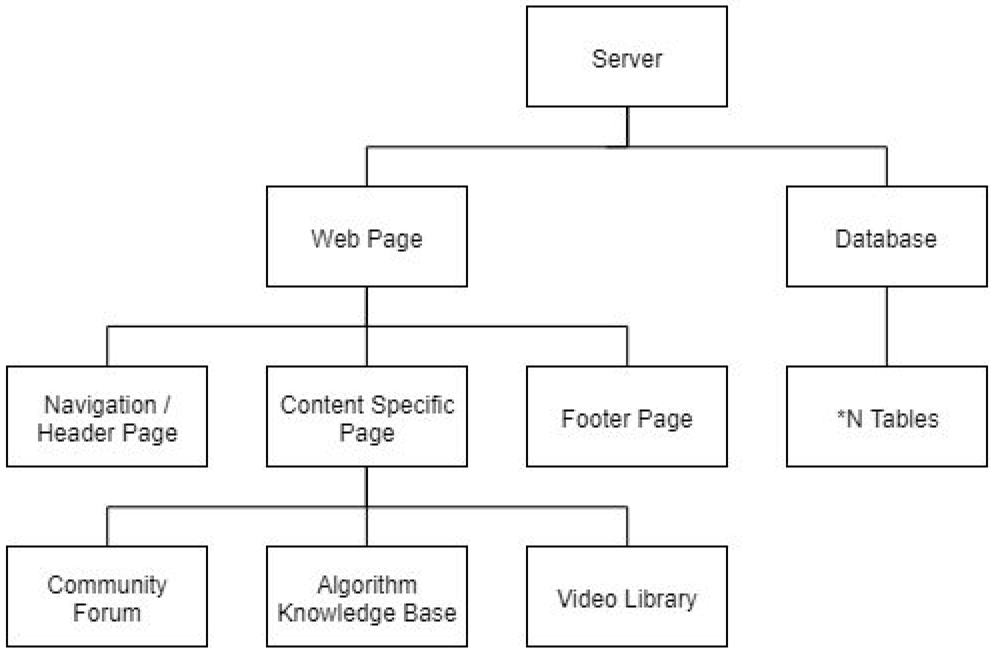
DECOMPOSITION DESCRIPTION

Data Flow Diagram:



This diagram represents the flow of data as it moves from the user through the rest of the site. The webpages will provide a user interface through which a user may submit data. The input will be entered into webpage forms and depending on the form the data is being entered into, a database query will be generated to process the data. The results of the query are displayed to the user.

Structural Decomposition Diagram:



This is a diagram represents a structural breakdown of how the site is going to be organized on the server. The web page portion of the diagram depicts how the site is going to combine components to build a complete webpage. The right half of the diagram is a representation of the database that will be installed on the server. The *N tables component represents the tables that will be created in the server's main database. Although there is only one subcomponent depicted in this diagram, there will be several tables that exist in the database which are outlined later on in the in the "Database Dictionary / Schema" section of this document.

DESIGN RATIONALE

The server is broken up into two main parts that drive the functionality of the site. This modular server architecture was chosen in order to ensure that the individual modules could be developed autonomously so that the features of one module don't impact other modules. However the web pages and the database are still hosted on the same server to ensure that the website is able to communicate with and send commands to the database portion of the site.

The website is broken up into three major sections: the header, the footer, and the content specific section. The header and the footer are specific to the site and not necessarily to any one feature. This modularity was chosen so that the user interface of the site remains consistent and easy to navigate. Another benefit is that if any feature needs to be added to the site in future updates, less work will need to be done to integrate the feature's content page into the site. Only a change to the navigation bar in the site's header and having the content pages on the server is required. This modularity also means that the each content specific module doesn't have to worry about managing its own separate connection to the database, the main site page loader will already have that functionality built in.

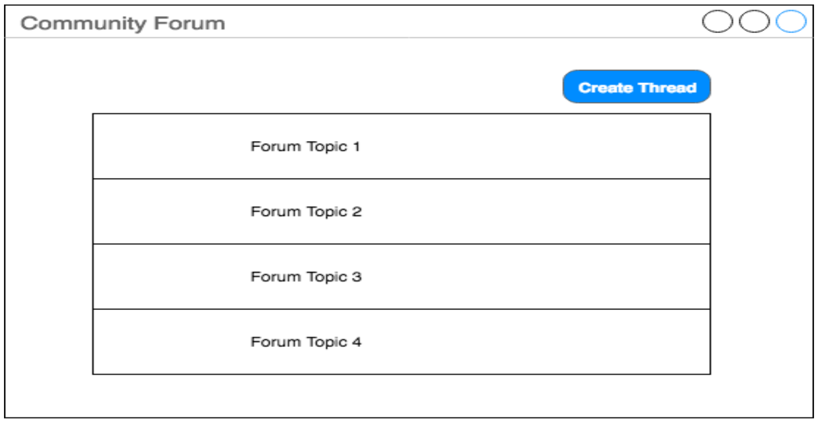
Despite this modularity, a lot of pages on the site are dependent on the main page and the code it contains for ensuring that commands get to the database properly and that all pages are loaded correctly. This means that the code for the main webpage has to be robust enough to accomodate all of the functionality any of the content specific pages might need.

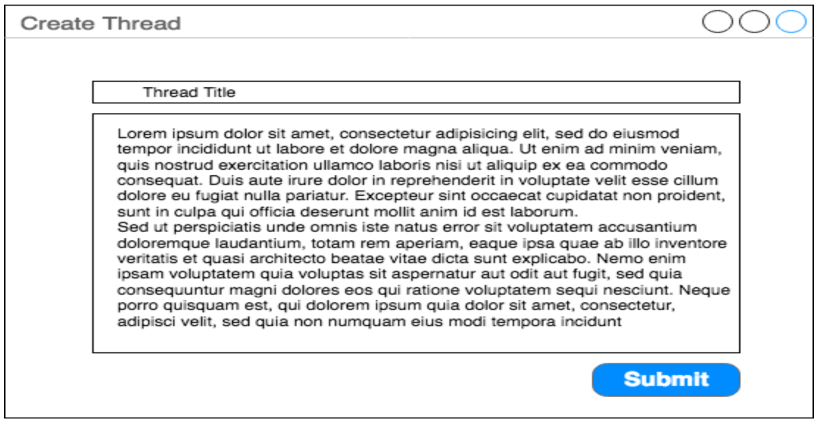
This server architecture was also chosen due to the fact that both PHP and mySQL both needed to be installed on the same system in order to achieve proper communication between the two. The only way PHP code will also only run through a server that is PHP compatible, making it so that the website has to be hosted on some server, as a lot of the advanced code for the site comes from PHP.

HUMAN INTERFACE DESIGN

Community forum feature (Joseph Rodriguez)

CREATE THREAD USE CASE





Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt

The create a thread page will allow users to create a thread within the community forum. In order to access the create thread page the user must already be in the community forum.

INPUT & OUTPUT

INPUT:

- **Thread Title:** String representing the title of the thread being created by user. Minimum of 3 and maximum of 50 characters.
- **Thread Body:** Text representing the body of the forum post. Min 10 and maximum 2000 characters.

OUTPUT:

Upon successful creation of the thread topic, the system will redirect the user to the community forum homepage and post their post in the topic of the community forum topics list. If the data entered is not valid, the system will show the user a list of errors that are to be corrected prior to a successful submission

Possible messages:

Failed: "Title length needs to be longer than 3 characters and shorter than 50."

Failed: "Thread body length needs to be longer than 10 characters and shorter than 2000."

ACTIONS

Upon submission of the thread post, the system will validate that the title and body have input, and that the title and body both meet their requirements for minimum characters and maximum characters.

If the validation is successful, the system will run a query to store the user's thread post within the database.

If the validation failed, the system will return an error message for each invalid field and no query will be made.

PRE AND POST CONDITIONS

Pre-conditions: The user should be already be a user within the system database and the user should also be logged in.

Post-conditions: User thread post is stored successfully. User is redirected to the community forum homepage.

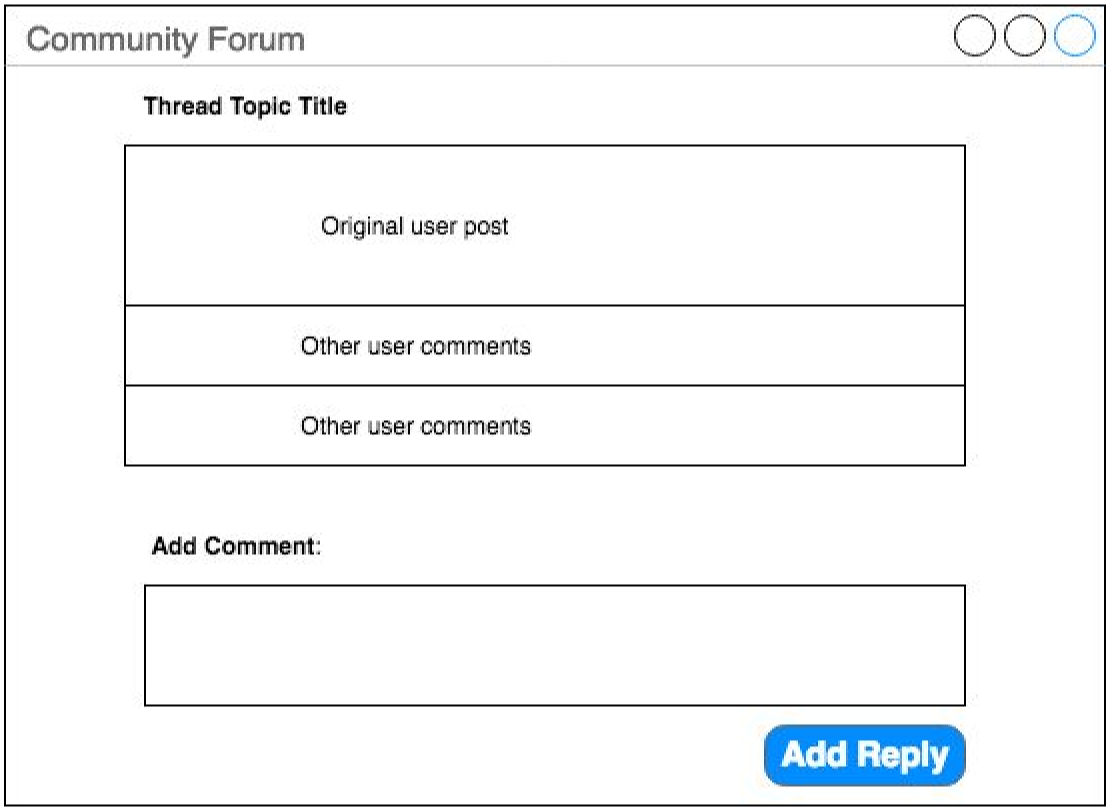
VALIDATION

The following validations will be performed on the submitted data:

- **Title:** field must be at least the minimum specified length and no more than the maximum length.

- **Body:** field must be at least the minimum specified length and no more than the maximum length.

REPLY USE CASE



The forum topic that is selected will allow the user to read thread the original post and all other user comments. It will also allow the user to add their own comment if necessary.

INPUT & OUTPUT

INPUT:

- **Comment:** text representing the user comment which has to be a minimum of 3 characters and a max of 2000 characters.

OUTPUT:

A comment is created within the thread post and the user page is refreshed so that the user can see the updated thread containing their comment.

Possible messages:

Failed: “Comment length needs to be longer than 10 characters and shorter than 2000.”

ACTIONS

Users will scroll through topics as they please. Once finding a topic that they can relate to or contribute to they will then click the reply button. This button will prompt a text body to appear. Once user has typed their comment they can then click the submit button and their post will be displayed.

PRE AND POST CONDITIONS

Pre-conditions: Only users that are registered and logged in may contribute to a thread by replying.

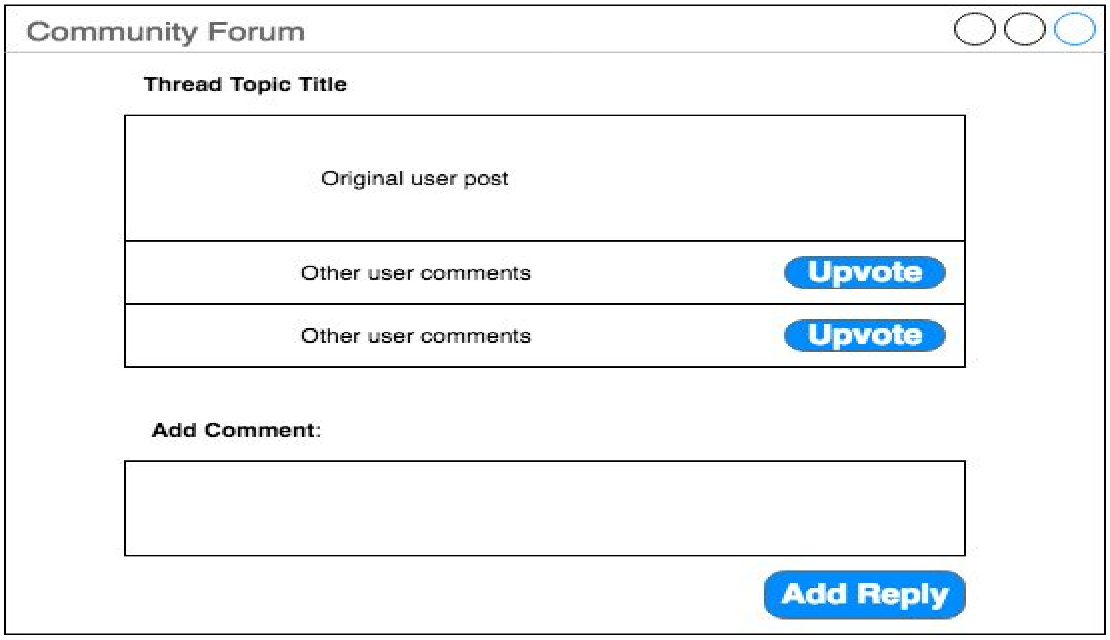
Post-conditions: The comment is stored in a database and is handled by server-side code to retrieve and display it to the view.

VALIDATION

The following validations will be performed on the submitted data:

Comment: field must be at least the minimum specified length and no more than the maximum length.

UPVOTE USE CASE



The user will be able to upvote by clicking the upvote button on comments that they believe contribute to the community.

INPUT & OUTPUT

INPUT:

- **Upvote click:** User click.

OUTPUT:

Upon user click of the upvote button an incrementation of the comments vote is made within the database and the comment vote count increments on the thread page.

Possible messages:

Failed: “You must be logged in in order to upvote a comment.”

ACTIONS

As users read through a thread and determine the best response they can upvote a comment by clicking a button. This will allow for comment to also move to the top of the thread (closer to the original post).

PRE AND POST CONDITIONS

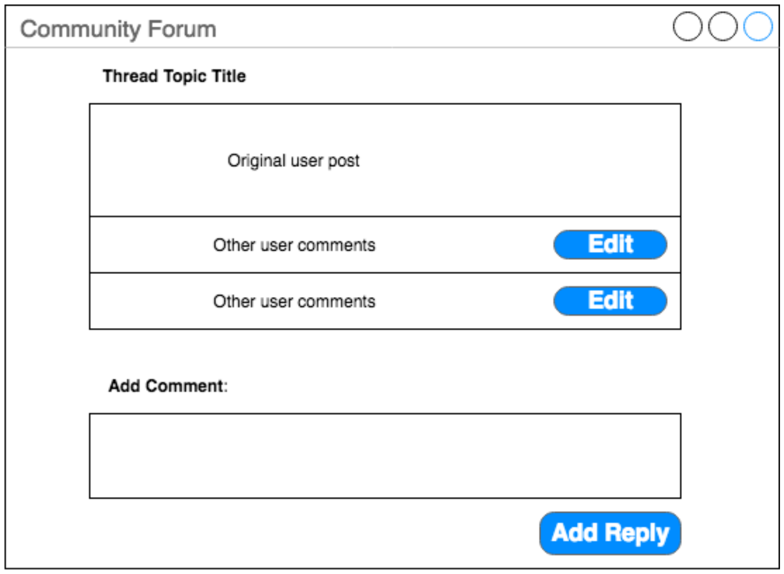
Pre-conditions: User must have not upvoted this comment before. User must also be logged in to upvote a comment.

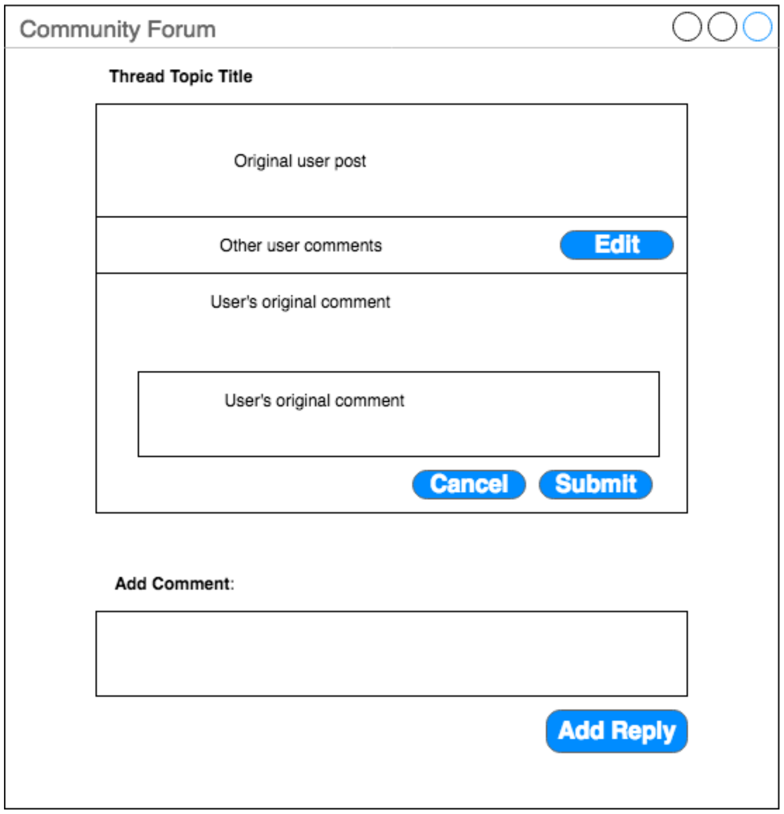
Post-conditions: The comment will be ranked amongst the current comments and their current weight. Comments will be displayed in order of their weight rank.

VALIDATION

The system must check if the user is logged in and has not voted for this comment before prior to incrementing the comment vote weight.

EDIT COMMENT USE CASE





The User will be able to edit their own comments in case of a mistake.

INPUT & OUTPUT

INPUT:

- **Edit button click:** User click.
- **Comment:** text representing the user’s edited comment between the minimum and maximum characters.

OUTPUT:

Upon submission the user’s new edited comment will be displayed.

Possible messages:

Failed: “Comment length needs to be longer than 10 characters and shorter than 2000.”

ACTIONS

User that have already posted a comment are able to go back and edit their comments by clicking an edit button that is laid out within the displayed comment. In addition, admins will also be able to edit any other user post to ensure that their are no offensive comments.

PRE AND POST CONDITIONS

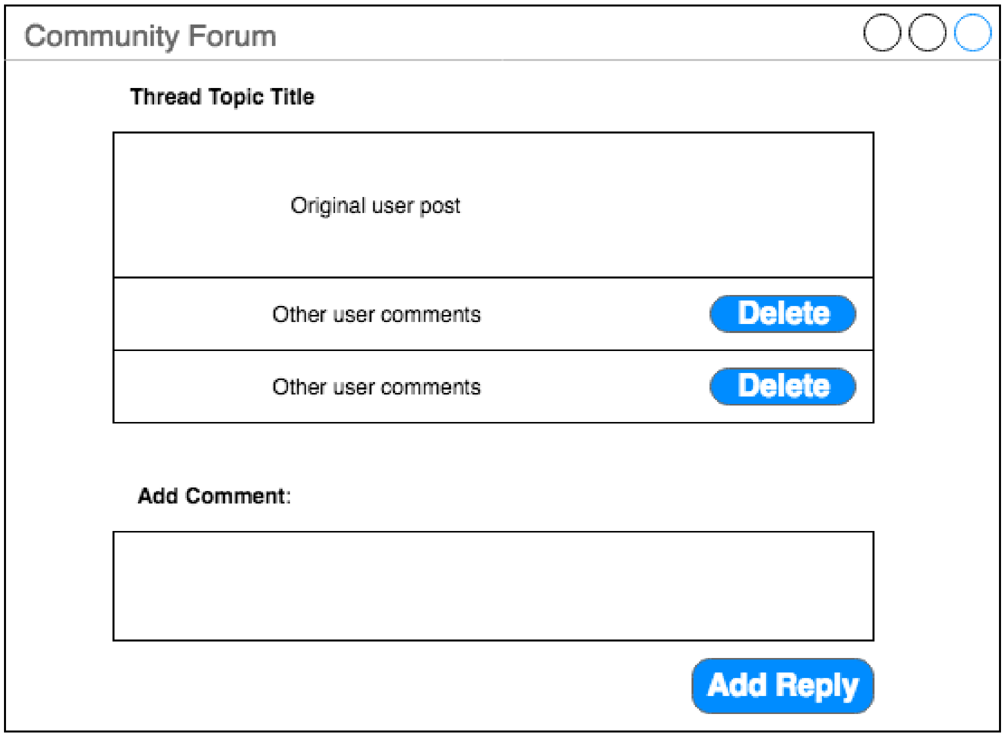
Pre-conditions: User must be logged in and have previously posted in order to edit a comment.

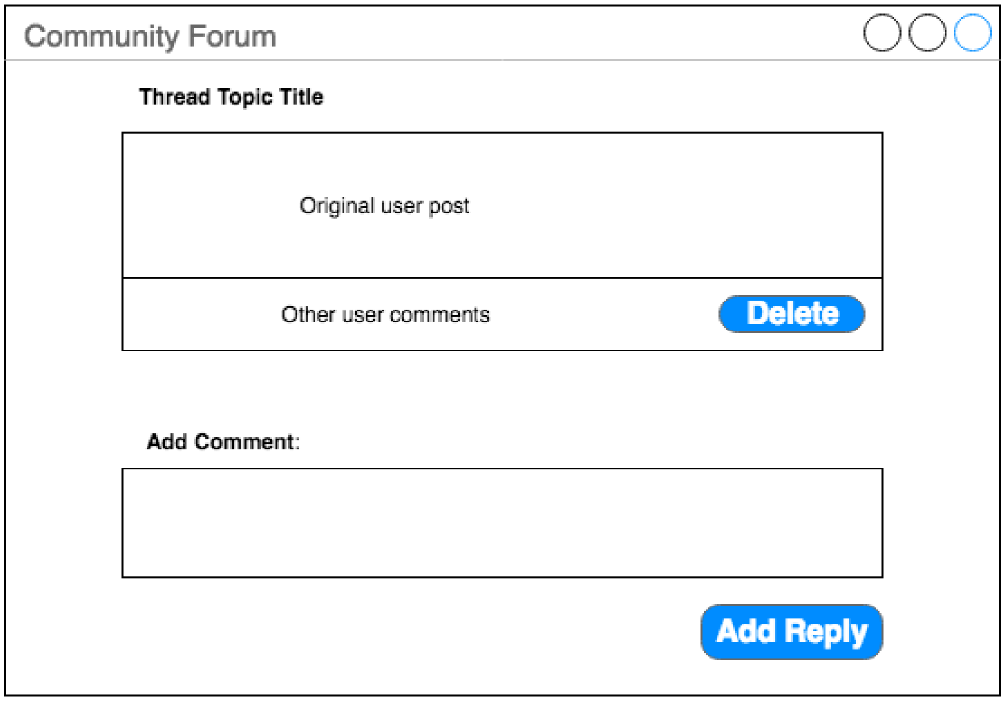
Post-conditions: After submit is clicked the page will display the edited comment. Necessary changes will be made to the database to display the correct information.

VALIDATION

The system must check if the user is logged in and has previously posted in order to all them to edit a post. In addition, if a user has admin permissions they will be allowed to edit any comment within the community forum.

DELETE COMMENT USE CASE





The User will be able to delete their own comments in case of a mistake.

INPUT & OUTPUT

INPUT:

- **Delete button click:** User click.

OUTPUT:

Upon user click the post will be deleted. The page will be refreshed with the deleted comment.

Possible messages:

Failed: "You must be a logged in in order to delete a comment."

ACTIONS

User that have already posted a comment are able to go back and delete their comments by clicking an delete button that is laid out within the displayed comment. In addition, admins will also be able to delete any other user post to ensure that their are no offensive comments.

PRE AND POST CONDITIONS

Pre-conditions: User must be logged in and have previously posted in order to delete a comment.

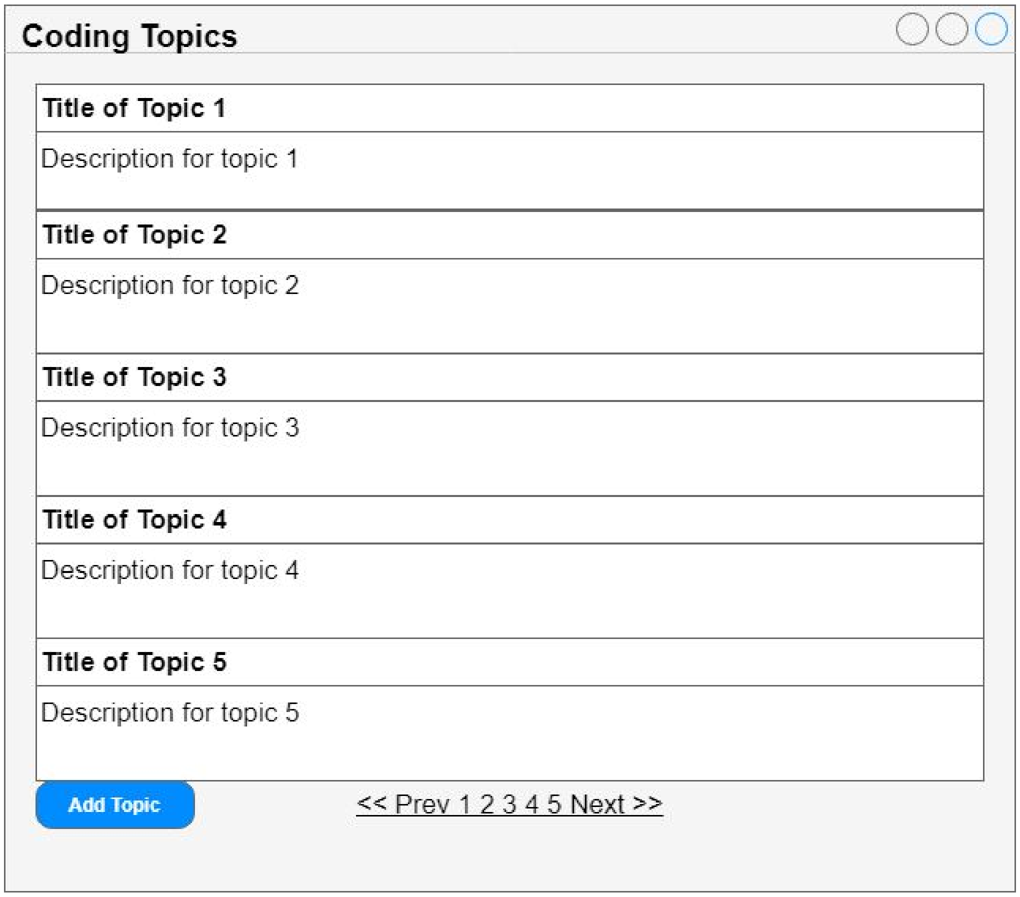
Post-conditions: After delete is clicked the page will display the current thread without the deleted comment. The necessary query will be made within the database to ensure that this displayed correctly.

VALIDATION

The system must check if the user is logged in and has previously posted in order to all them to delete a post. In addition, if a user has admin permissions they will be allowed to delete any comment within the community forum.

CODING TOPICS AND SOLUTIONS FEATURE (By Louis Rivera)

VIEW EXISTING CODING TOPIC USE CASE


Coding Topics

Title of Topic 1

Description for topic 1

Title of Topic 2

Description for topic 2

Title of Topic 3

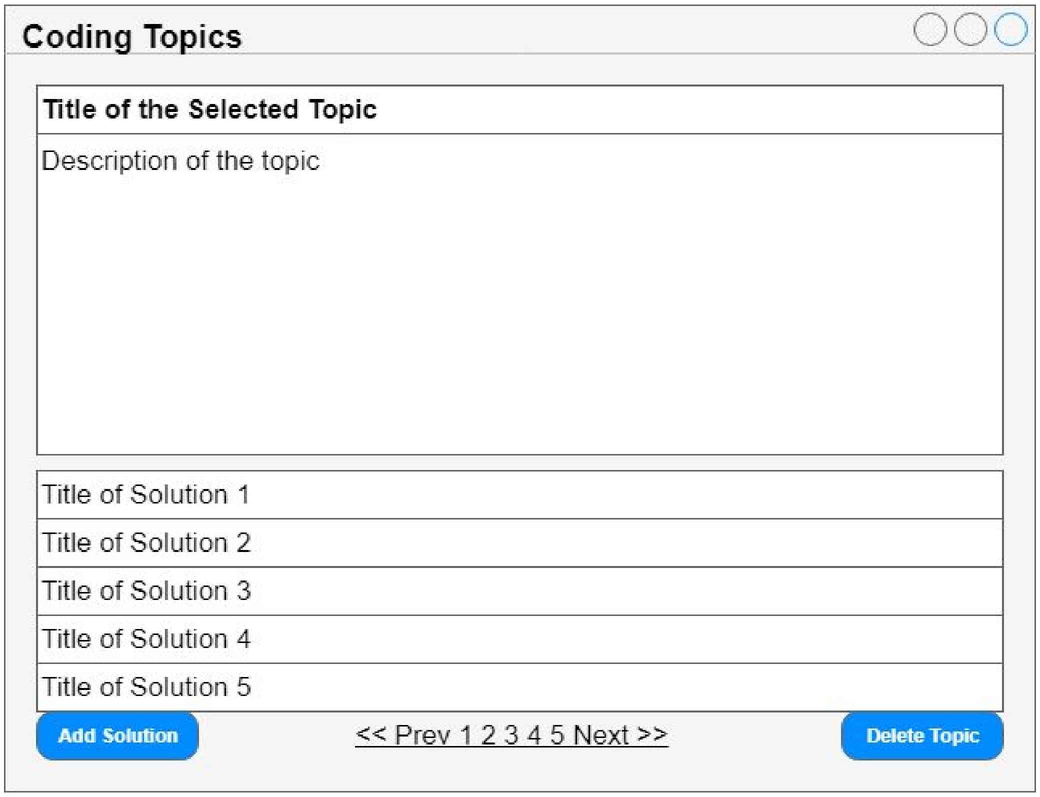
Description for topic 3

Title of Topic 4

Description for topic 4

Title of Topic 5

Description for topic 5



When first entering the coding topics page, the first page will be displayed listing all topics posted to the site so far as well as giving users the option to post more topics. Clicking on a topic's title will bring up the second page shown, which displays the information for that specific topic.

On the second page the user is also presented the options for selecting specific solutions, selecting any solution will take the user to that specific solution's page. Clicking on the add solution button will take the user to the add a solution page.

The delete topic button can also be found here but should be shown only if the user is logged in as an admin. The button takes the admin to the delete topic page.

The topics and solutions are listed 5 at a time. The navigation buttons at the bottom of the page are for cycling to the next 5 topics.

INPUT & OUTPUT

INPUT:

- **Selected Topic:** Each of the topics that is listed acts as a button that can be selected by the user
- **Topic Navigation:** The topics are listed in sets of 5 at a time. The navigation buttons at the bottom allow the user to cycle to the next set with "Next", the previous set with "Prev", or to a specific set with the numbers.
- **Selected Solution:** Each solution that is listed on a topic page acts as a button that can be selected by the user.

- **Solution Navigation:** The solutions are listed in sets of 5 at a time. The navigation buttons at the bottom allow the user to cycle to the next set with “Next”, the previous set with “Prev”, or to a specific set with the numbers.

OUTPUT:

The initial page seen will be a list of all available topics that are currently stored in the site’s database. The list of topics are displayed in sets of 5 at a time.

A specific topic being clicked on will cause the site to switch to a new page with information specific to that topic. The topic’s title and a description of the topic are displayed. A list of solutions that have been posted to the topic are also displayed. The solutions are displayed in sets of 5 at a time.

Important to note that when viewing a specific topic there is a Delete Topic button. This button will only be enabled / visible for administrators of the site.

ACTIONS

When Viewing all Topics:

- Clicking on a topic’s title will load the page containing that topic’s specific information.
- Selecting the “Add Topic” button will switch to the page that allows user to post new topics to the site.
- Clicking the navigation buttons will change the set of topics the user is currently viewing.

When Viewing a Specific Topic:

- Clicking on one of the displayed solutions will bring the user to a page specific to that solution.
- Selecting the “Add Solution” button will switch to a page that allows users to add solution to the currently viewed topic.
- Selecting the “Delete Topic” button will switch to a page that allows admins to delete the topic being viewed.
- Clicking on the navigation buttons will change the set of solutions the user is currently viewing.

PRE AND POST CONDITIONS

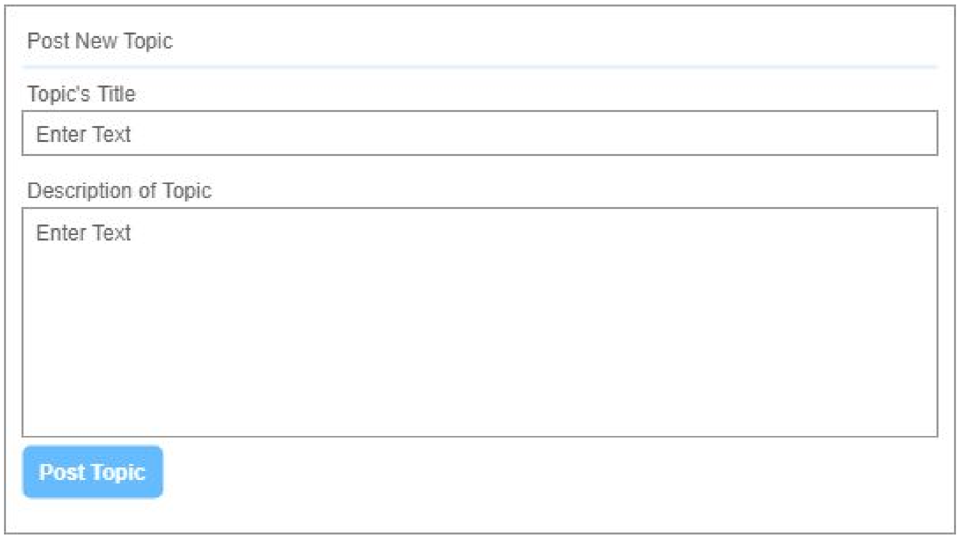
Pre-conditions: There are no restrictions on viewing the list of topics or on viewing a specific topic.

Post-conditions: If a user is on the page containing all topics, they are presented with a list of all topics in the site's database. If a user has selected a specific topic they, they are presented with a page that is specific to that topic.

VALIDATION

The only inputs taken from the user are in the form of selections made on the lists of data presented by the site. As a result no validation should be necessary on the selections the user is making as all selections that are possible will have been retrieved from the site’s database.

POST NEW CODING TOPIC USE CASE



The Post New Topic page will allow users who are logged into the site to add a new topic to the site that will be viewable by other users.

INPUT & OUTPUT

INPUT:

- **Topic's Title:** String. Will be used by users to identify topics, can be a minimum of 1 character to a maximum of 50 characters.
- **Description of Topic:** String. The description of the topic a user is attempting to post. Can be a minimum of 1 character to a maximum of 10,000 characters.

OUTPUT:

If the post was success, the user is shown a message informing them that their post has gone through. If the post was unsuccessful the user is shown an error message telling them which fields violated the character limit restrictions and requests that they correct the error and try again.

ACTIONS

When the form is submitted, the site validates that both fields have text supplied by the user and that the text doesn't violate the character limits.

If the validation is successful, the site will run a query to store the topic in the site's database.

If the validation is unsuccessful, the site presents the user with an error message detailing which of the fields failed the validation.

PRE AND POST CONDITIONS

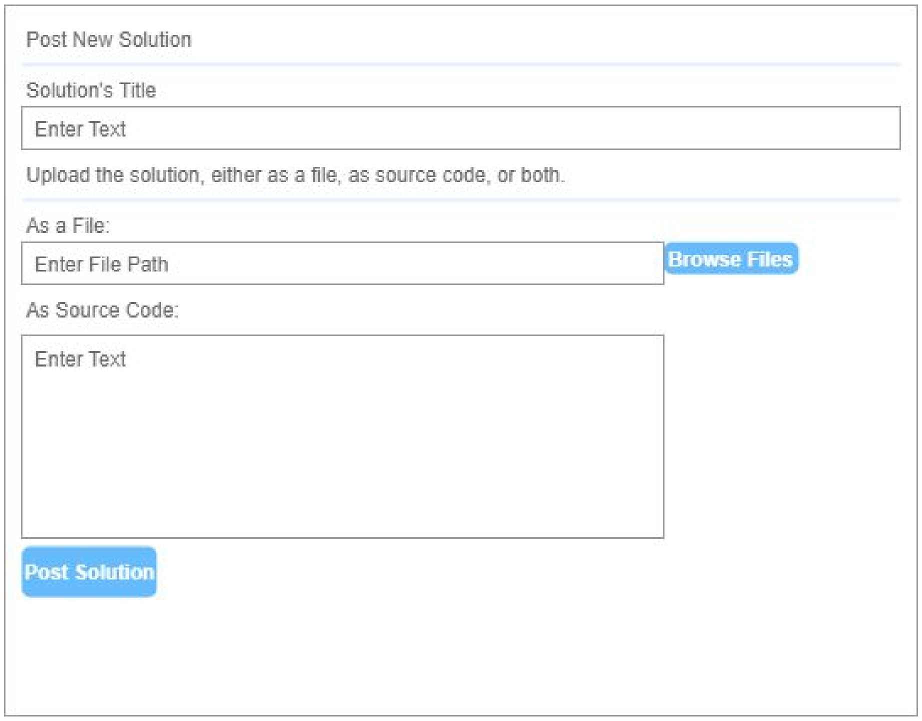
Pre-conditions: A user has to be logged into the site.

Post-conditions: A successful post will result in a new topic being created and stored in the site's database. Other users of the site should be able to view the new topic.

VALIDATION

The Topic's Title and Description fields will both be validated for ensure they are within the legal character limits.

ADD NEW CODING SOLUTION USE CASE



The Post New Solution will allow a user that is logged into the site and viewing a topic to add a new solution to that topic.

INPUT & OUTPUT

INPUT:

- **Solution's Title:** String. The name given to a specific solution. Must be a minimum of 1 character and a max of 50 characters.
- **Solution File:** File. The textbox will contain a file path that should point to a file on the user's computer containing the code for the user's solution. The file will have to be of a specific type, .txt, .java, or .cpp.
- **Solution Source Code:** String. The source code for the solution can be entered in as text. The text can be a minimum of 1 character and a maximum of 10,000 characters.

OUTPUT:

If the solution posting was successful, the user will be shown a message telling them that their solution was added to the topic they were viewing.

If the title or source code the user entered violated a character limit an error message will be shown that informs the user of the violation and asks them to correct the issue and try again.

If a user doesn't choose a file and doesn't enter in source code, then an error message is displayed to the user telling them that at least one of the two is required and asks the user to correct the issue and try again.

ACTIONS

When the user chooses to browse files, a window appears that allows the user to select a file on their computer that they want to be uploaded. Selecting a file will insert the file path to that file into the text box.

When the user selects to post the solution, the site verifies that the title doesn't violate character limits. If a violation occurs then the user is shown an error message informing the user of the violation.

The site also verifies that at least one of the solution options is utilized. Either source code has to be entered or a file has to be selected for uploading. If neither option is selected an error message is displayed to the user informing them that a solution is required.

If source code is enter the site verifies that it doesn't violate any character limits. If the field does violate a character limit then the user is shown an error message informing the user of the violation.

If a file is chosen to be uploaded the site verifies that the file attempting to be uploaded exists at the given location and that the file has an extension that is supported by the site. If one of these rules is broken an error message is displayed to the user informing them of the violation.

If no errors or violations occur as a result of the information the user is attempting to submit, a new solution is added to the topic the user is viewing. A query is run adding the solution title, source code and / or file to the site's database. The solution is stored so that it references the topic that it was posted to.

PRE AND POST CONDITIONS

Pre-conditions: A user is logged into the site and has either the source code for a solution or a file containing the code available to them. The user also has to be viewing a specific topic.

Post-conditions: Assuming a post is valid, the user's solution is added to the site's database. The solution can then be viewed by other users viewing the topic the solution was posted to.

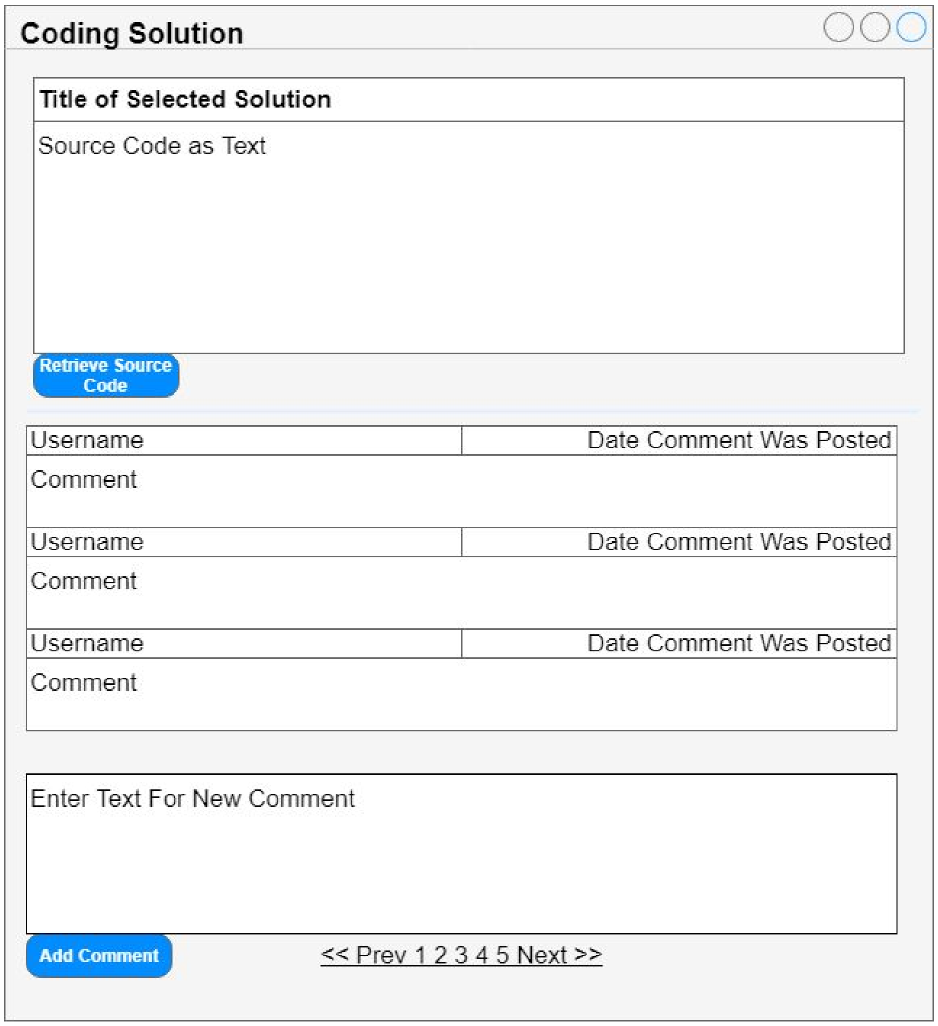
VALIDATION

Either a file or source code has to be provided in order for the transaction to go through.

The file path has to be checked to ensure the file exists and that the file chosen is of the proper extension: .txt, .java, .cpp.

Both the title and the source code text fields (assuming the source code field is utilized) have to be checked to ensure they fall within character limits.

COMMENT ON A CODING SOLUTION USE CASE



INPUT & OUTPUT

INPUT:

- **New Comment Text:** String. Used if a user wants to add a new comment to the currently viewed solution. The text for the comment must be a minimum of 1 character and a maximum of 10,000 characters.
- **Comment Navigation:** The comments are listed in sets of 3 at a time. The navigation buttons at the bottom allow the user to cycle to the next set with "Next", the previous set with "Prev", or to a specific set with the numbers.

OUTPUT:

Clicking on the navigation buttons at the buttons will cause the page to display a new set of comments.

Entering in a comment that doesn't violate any character limits will cause the comment to be added to the list of comments being displayed. If a comment's text does violate character limits then an error message is displayed to the user informing them of the violation.

ACTIONS

Clicking on the navigation causes the displayed list of comments to cycle to the next 3 available comments. All comments being displayed are ones that have been posted specifically to the solution being viewed.

If a user enters a comment and chooses to add it to the solution, the comment is verified to ensure that it doesn't violate any character limits. If it does the site shows the user an error. If no violation occurs then the site adds the comment to the site's database. A query is made adding the comment, along with the user who posted the comment and the time the comment was posted, to the database. The comment is stored so that it reference the solution that it was posted to.

PRE AND POST CONDITIONS

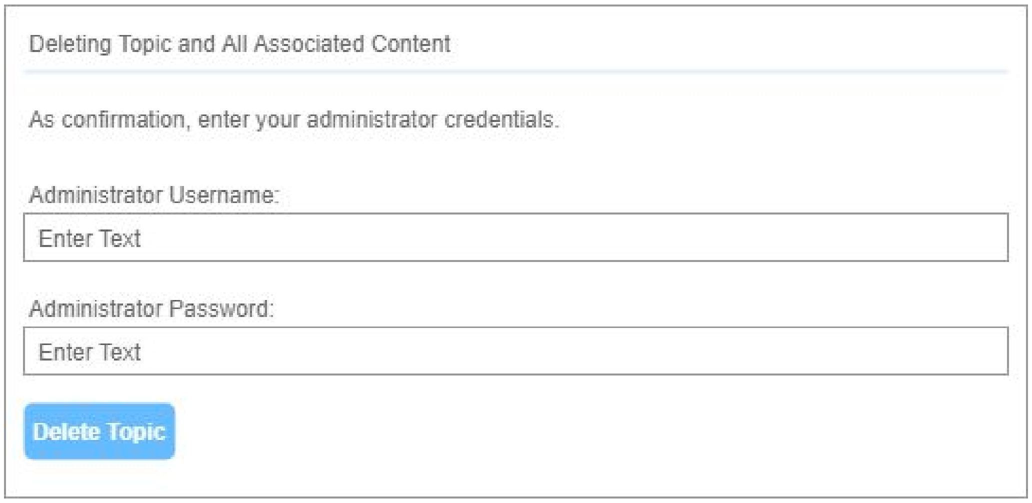
Pre-conditions: The user has selected a solution while viewing a topic. In order for a user to post a comment to that solution they also have to be logged into the site.

Post-conditions: While viewing the solution the user is shown that solutions information as well as all comments posted to that solution. Successfully posting a comment causes that comment to be displayed along with all other comments posted to that solution as well as being added to the sites database, along with information on the user that posted the comment and the time the comment was posted.

VALIDATION

A comment that is posted is validated to ensure that the text entered doesn't violate any character limits.

DELETE TOPIC USE CASE


Deleting Topic and All Associated Content

As confirmation, enter your administrator credentials.

Administrator Username:

Enter Text

Administrator Password:

Enter Text

This page is used by administrators who are trying to delete topics from the site. While a topic is being viewed and an admin is logged in on their amin account, the admin can choose to delete the viewed topic. They are then prompted with this page to confirm that they want the topic to be deleted.

INPUT & OUTPUT

INPUT:

- **Administrator Username:** String: Represents the administrators username for their site admin account. Can be a minimum of 6 characters and a maximum of 20 characters.

- **Administrator Password:** String. Represents the password used to log into the site with their admin account. Can be a minimum of 8 characters and a maximum of 20 characters. Must also contain one capital letter and one number.

OUTPUT:

If the text entered by the admin doesn't violate any character limits and the information entered matches a record for an account in the site's database that has administrator privileges, then the admin is shown a message informing them that the topic along with all related content on the site has been deleted.

If the text entered for the username or password violates a character limit, the admin is shown an error message informing them that the violation has occurred.

If the credential's entered by the admin can't be found in the site's database or if the credentials are found but the account they belong to isn't an administrator account, then the admin is shown an error message informing them that in order to confirm the deletion, correct credentials have to be entered provided.

ACTIONS

Once the the administrator enters their credentials in and choose to delete the topic the site validates the credentials. If the credentials don't match an existing admin's account credentials in the database, the request is denied and the admin is shown an error. If the credentials violate any character limits the admin is shown an error.

If the credentials are valid then a query is run on the database. The topic that was being viewed is removed from the database as well as all solutions posted to the topic as well as any comments posted to those solutions.

PRE AND POST CONDITIONS

Pre-conditions: A user is logged into an administrator account and viewing a topic. The delete topic button is selected.

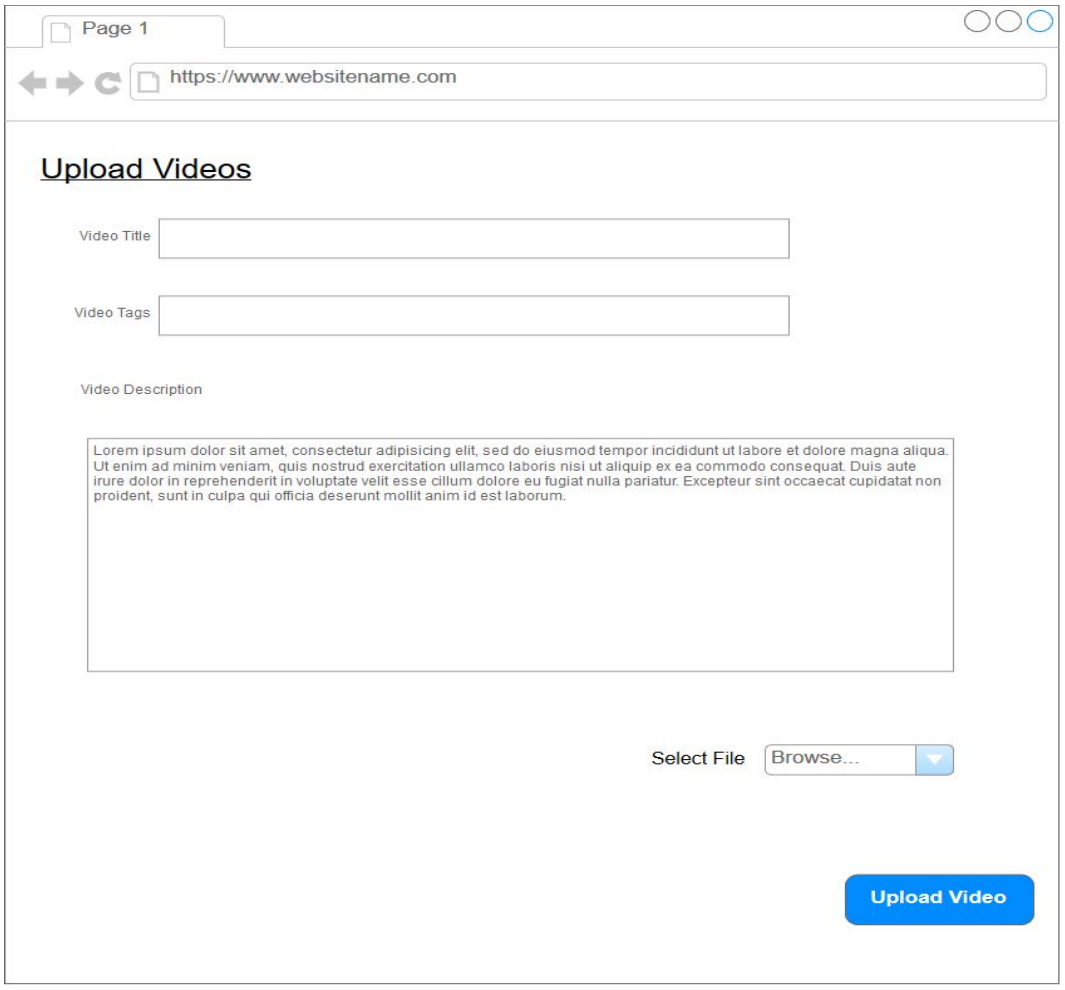
Post-conditions: The topic, all associated solutions and comments, are removed from the database and can no longer be displayed on the site.

VALIDATION

The administrator credentials are validated to ensure that they fall within the text field character limits. The credentials supplied are validated against the database entries for all users. An account entry must exist in the database that matches the supplied credentials, username and password supplied should match the ones in the database, and the account must be flagged as one that has administrator privileges.

Video Library Feature (by Rafel Salem)

Upload Videos Use Case



Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

The upload videos page will allow the site administrators to upload new videos for the users to watch. The administrators must fill all the fields and have a video file uploaded.

Input & Output

Input:

- **Video Title**: a string representing the video title that has a lower length limit of 1 and an upper length limit of 50 characters.
- **Video Tags**: a string of words that describe a video and are separated by commas with an upper limit of 50 characters.
- **Video Description**: a string representation of the description of the video with an upper limit of 300 characters.
- **Video File**: a file format of type video with size limit of 300 mb.

Output:

A message is displayed to the administrator to announce whether the video upload process was successful or not. Additionally, once the message is acknowledged, the administrator will be taken back to the appropriate page.

Actions

Once all of the fields are filled and the administrator clicks the submit button, the information and video will be filled out/created respectively in the database.

Pre and Post Conditions

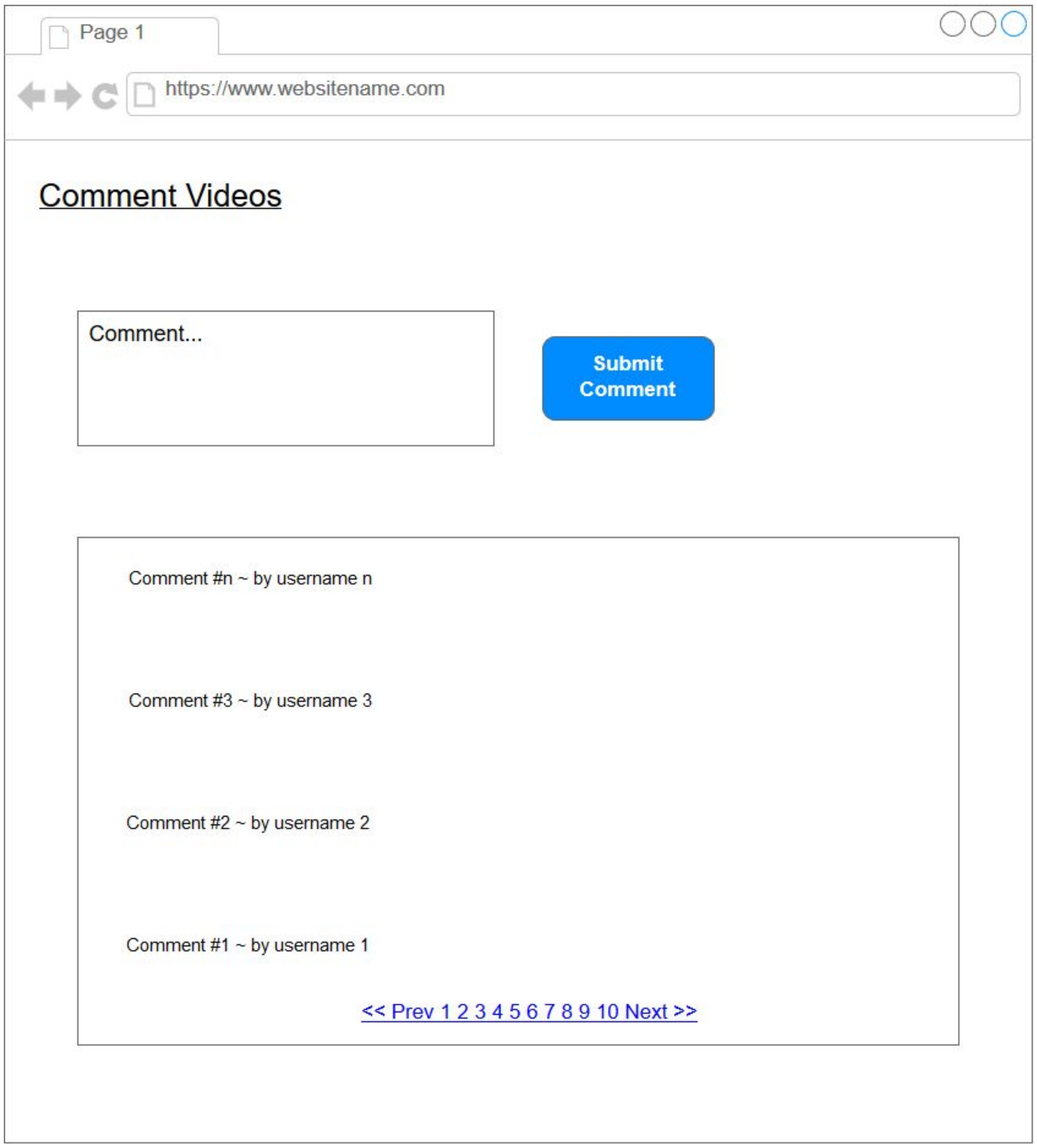
Pre-conditions: the fields that are filled out must adhere to the imposed restrictions.

Post-conditions: if the video was uploaded successfully there will be a message indicating the success of the process. If the video was not uploaded successfully a message will display the process was not successful.

Validation

All the fields will be checked to determine if they adhere to the imposed requirements. The video title will be checked for an upper and lower limit fit. The tags will be checked to see if it fit the upper limit. The description will be checked to see if it fit the upper limit. And the video will be checked to see if it fits the size limit and is in video format.

Comment Videos Use Case


Page 1
https://www.websitename.com
Comment Videos
Comment...
Submit Comment
Comment #n ~ by username n
Comment #3 ~ by username 3
Comment #2 ~ by username 2
Comment #1 ~ by username 1
<< Prev 1 2 3 4 5 6 7 8 9 10 Next >>

The comment videos page will allow users to write and submit comments on the selected video.

Input & Output

Input:

- **Comment**: a string that represents an opinion the user writes in response to the video. The comment will have a lower limit of 1 character and an upper limit of 300 characters.

Output:

The page will be refreshed and if the process was successful, the user will be able to see their comment in the comments section. If it is not successful, the user will be notified to check comment length.

Actions

The user writes a comment about the video they are watching or have watched. The user then submits their comment. The comment will be checked and an appropriate response will be returned to the user upon process success or failure. And if it was successful, it would be added to the database field holding the comment section.

Pre and Post Conditions

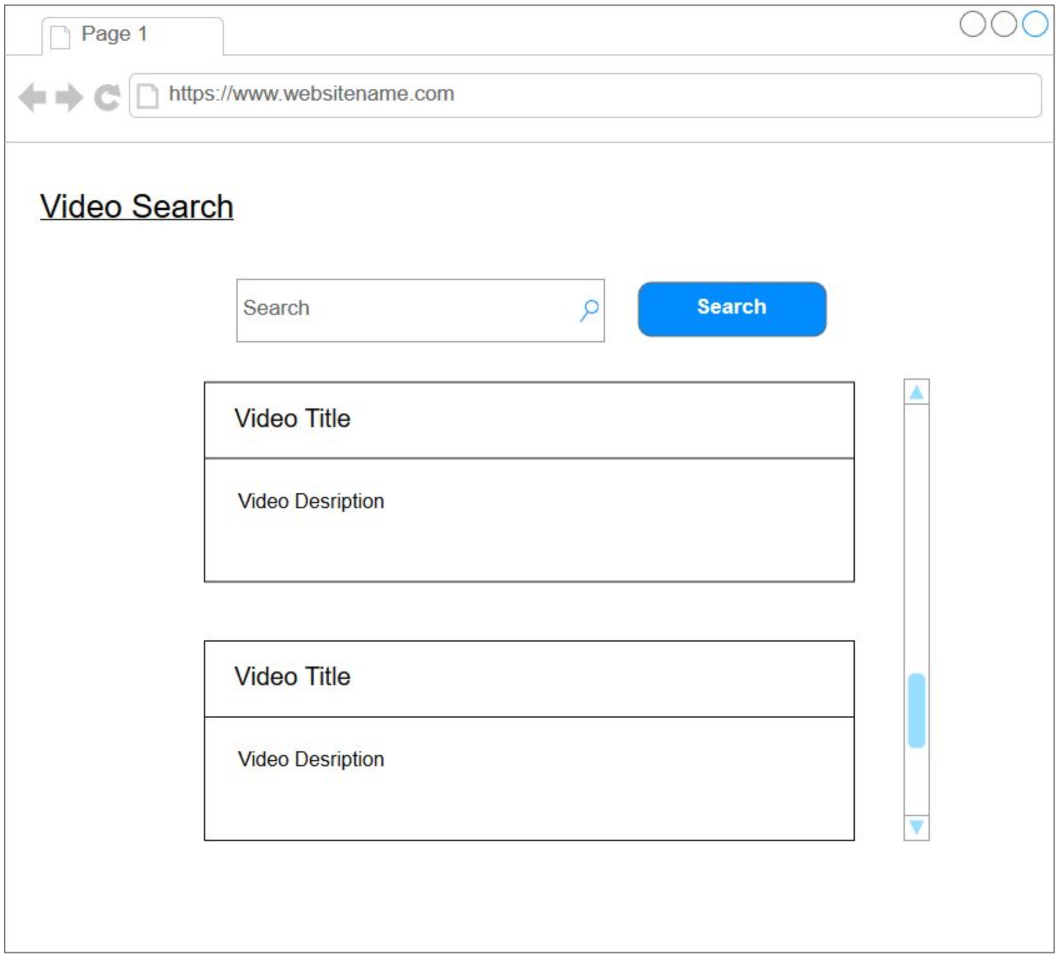
Pre-conditions: the comment must adhere to the upper limit of 300 characters that is imposed on all comments.

Post-conditions: successful submission of the comment will add the comment to the comment section. Otherwise the user will be given a message to warn them of the 300 characters limit.

Validation

The length of the comment will be checked to see if it is 300 characters or less.

Search Videos Use Case



The search videos page will allow the user to enter a string to filter a given list of videos/titles.

Input & Output

Input:

- **Video Title or Tags**: a string that represents a video title or tag with a lower limit of 1 character and an upper limit of 50 characters.

Output:

The video section will update and filter out the videos that do not fit the title or tag patterns.

Actions

After the user presses the search button, the web application will check to see if the string fits the length requirements. Then, it will try to display videos that match the provided string exactly. If none are found, then no videos are displayed.

Pre and Post Conditions

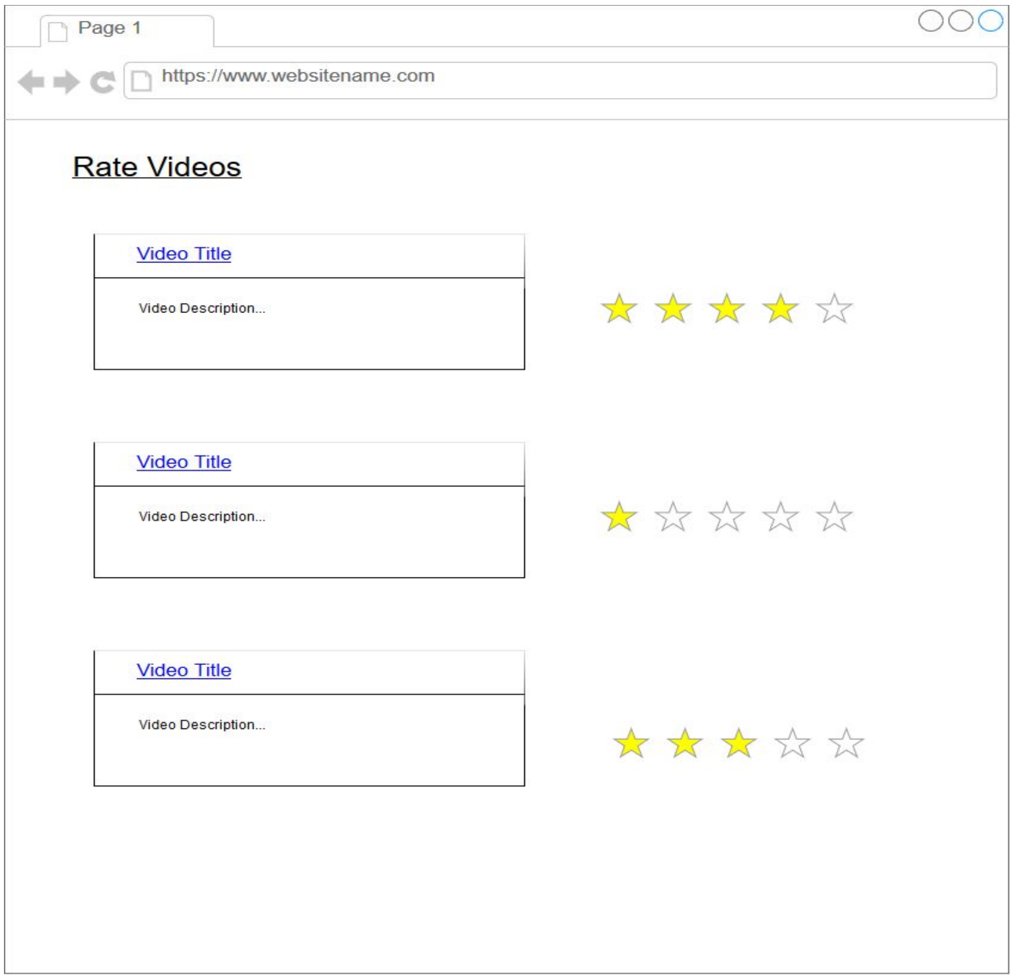
Pre-conditions: the string provided by the user must fit the length requirements. A video with a matching title must exist in the database.

Post-conditions: the video list will be filtered to show the video that matches the string.

Validation

The spring that is written by the user will be checked to see if it is within the range of 1 character to 50 characters.

Rate Videos Use Case



The rate videos page will allow the user to select a rating for a video and submit their choice for the selected video.

Input & Output

Input:

- **Rating**: the user will click on a rating button which ranges from 0 to 5 stars. The rating information will be passed to the database through Javascript.

Output:

The rating container will update the stars' colors according to the rating and display it to the user.

Actions

Upon picking a rating, the user's choice will be sent to the database for storage. The website will also update the rating container to display the rating chosen by the user.

Pre and Post Conditions

Pre-conditions: the user must not have already rated the video.

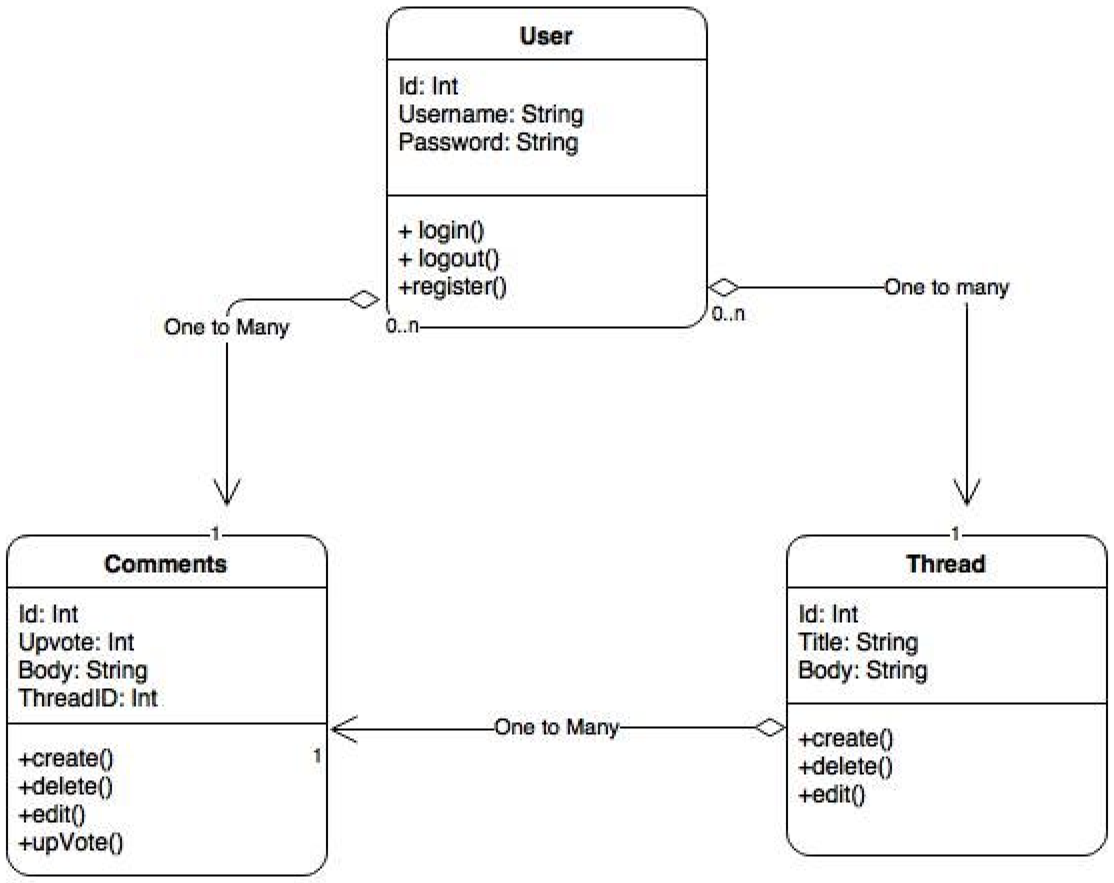
Post-conditions: the ratings will change to reflect the user's choice. The rating cannot be changed afterwards to protect integrity of data/opinion.

Validation

The web application will check if the video has been rated before by this user. If not, the web application will accept the user's rating. Else, the rating will not be modified.

COMPONENT DESIGN

COMMUNITY FORUM FEATURE



Entities

User: The user class will interact with other class (Thread, and Comments) so that each comment or thread topic created will belong to a user. Therefore, a user must be logged in. The method contained within the user class will have login, register, and logout functions.

User Attributes/Fields:

- ID: Unique identifier for a single User instance
- Username: The user name will be unique to a specific user and will act as the display name for the user.
- Password: A string created by the user to gain access to the application

Relationships:

- Thread: Each user will be able to create more than one thread therefore a user may have a one to many relationship with thread.
- Comments: A user may post more than one comment per thread therefore leading to a one to many relationship between user and comments.

Thread: The Thread class contains a many to one relationship with Users so one User may have many Thread topics. It is also contains a many to one relation with comments because for every one Thread there should multiple Comments. The functions contained within the Thread class will allow for CRUD operations to take place because on User permissions.

User Attributes/Fields:

- ID: Unique identifier for a single thread instance.
- Title: the title contains the topic for which the user chooses for their particular post.
- Body: is a string that contains all of the user's ideas or questions for a particular thread post.

Relationships:

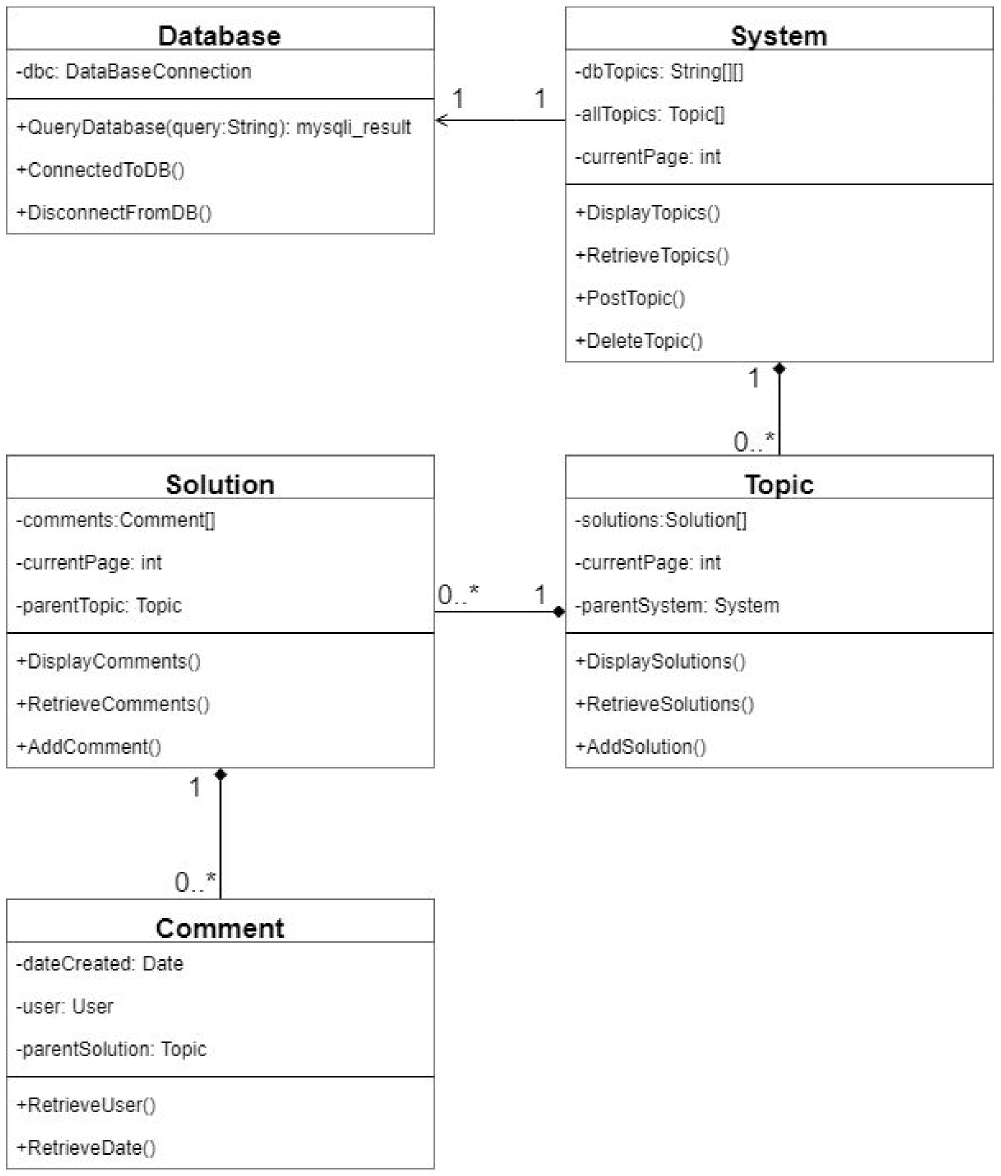
- Comments: A thread may contain more than one comment and therefore a one to many relationship exists.

Comments: The comment class will have a many to one relation with both Users and Thread. This will ensure that Thread can have one or more comments that belong to one or more different Users. The functions contained within the Comments class will allow users to utilize CRUD operations based on user permissions.

User Attributes/Fields:

- ID: Unique identifier for a single comment instance.
- Upvote: The upvote variable will contain the specific weight of the comment so that it may move up within the comment ranks.
- Body: is a string that contains all of the user's ideas or questions for a particular comment.
- Thread ID: will be identify which thread a comment belongs to. Each comment will belong strictly a specific thread.

CODING TOPICS AND SOLUTIONS FEATURE (By Louis Rivera)



ENTITIES

Database: This class is responsible for handling the connection to the database as well as sending queries to the database on the site's behalf. When querying the database an object of type mysqli_result is returned. This object contains the query results and needs other PHP functions to parse the results. There is only ever one database entity.

Database Attributes:

- dbc - When a connection is made to the site's database, a database connection object is returned and stored in this variable. Through this object, queries are sent to the database.

System: The system class keeps track of all topics available in the system and is responsible for displaying them. It also can create new topics as well as delete existing topics. There is only ever one system entity.

System Attributes:

- dbTopics - A query to the database will return a mysqli_result object that contains the results from the query. This 2D array contains the parsed rows. The rows of the arrays are the rows returned from the object and the columns are the columns from the database table.
- allTopics - This is an array of all topics that the system currently has inside of it.
- currentPage - Since topics are displayed in sets, this variable keeps track of what page is currently being viewed in order to make topic displaying easier.

Topic: Each topic entity keeps track of the solutions posted to itself and is responsible for displaying those solutions. This class is also responsible for creating new solution. Systems and Topics have a one to many relationship, respectively. Topics and Solutions have a one to many relationship, respectively.

Topic Attributes:

- solutions - This array stores all solution objects that have been added to a specific topic entity.
- currentPage - Since solutions are displayed in sets, this variable keeps track of what page is currently being viewed in order to make solutions displaying easier.
- parentSystem - This object is a reference to the system object. It'll be used when topics need to make system calls like modifying the database information.

Solution: The solution entity keeps track of and displays comments. It is also responsible for the addition of new comments that are posted to the solutions themselves. Topics and solutions have a one to many relationship, respectively. Solutions and comments have a one to many relationship, respectively.

Solution Attributes:

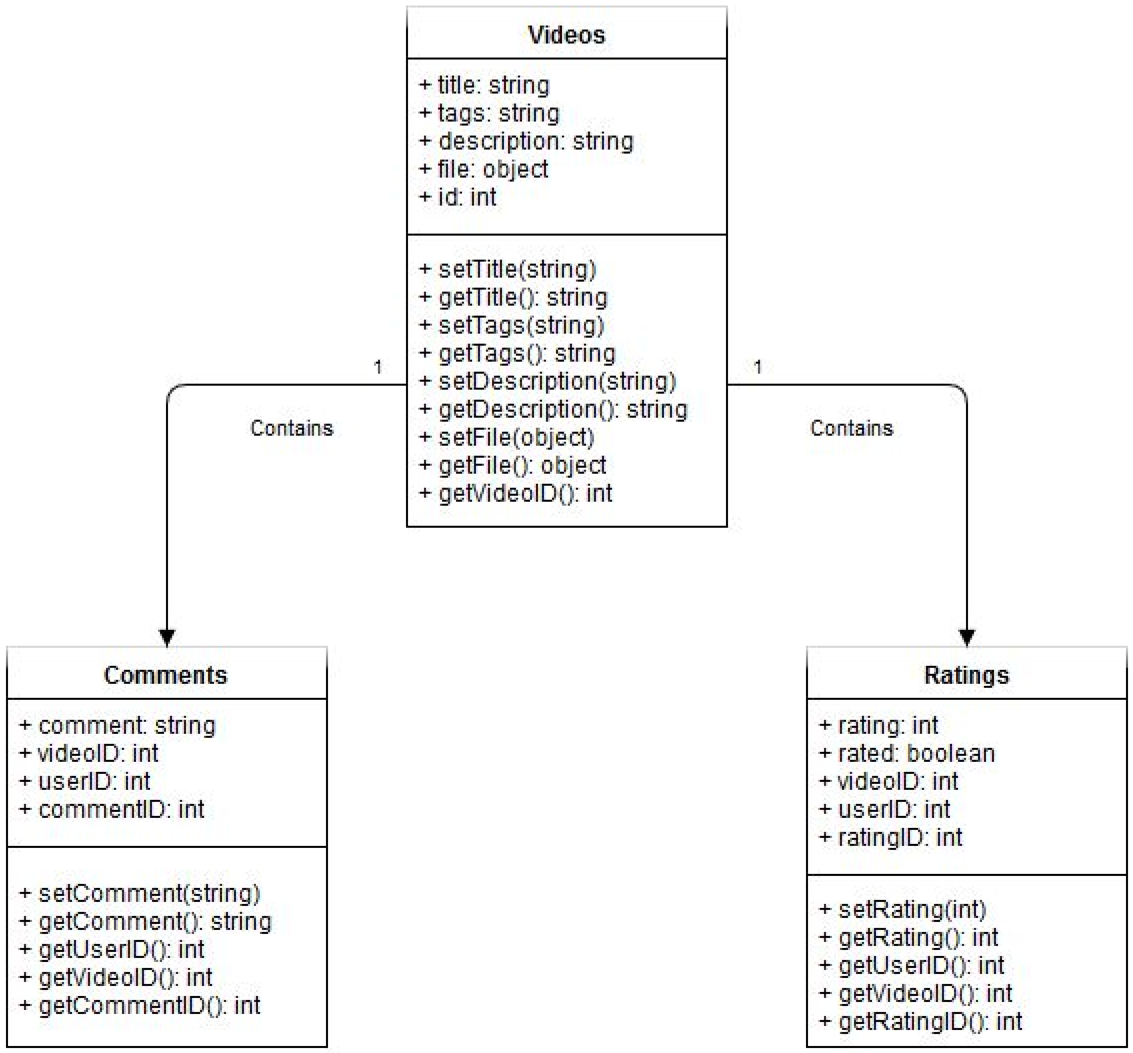
- comments - This array stores all comments that have been posted to a specific topic entity.
- currentPage - Since comments are displayed in sets, this variable keeps track of what page is currently being viewed in order to make solutions displaying easier.
- parentTopic - This object is a reference to the topic that a solution was posted under.

Comment: This object mostly keeps track of the information about the creation of the comment for displaying alongside the comment. Solutions and comments have a one to many relationship, respectively.

Comment Attibures:

- dateCreated - A date that indicating the date the comment was created.
- user - A reference to the user object that was used to create the comment
- parentSolution - A reference to the parentSolution that a comment was posted under.

Video Library Feature (by Rafel Salem)



Entities:

Videos: this object holds information about the videos uploaded by the administrators. The information can be set or retrieved.

Video attributes:

- title - the title of the video given by the uploader.
- tags - a combination of tags separated by commas.
- description - a description of the video.
- file - a reference to the object holding video file.
- id - an int value that uniquely identifies a video.

Relationships:

- The unique video id will tie a comment object to a video object.
- The unique video id will tie a rating object to a video object.

Comments: this object stores the comment the users make on the videos.

Comments attributes:

- comment - holds the comment written to describe a video.
- videoID - holds the unique video ID the comment belongs to.
- userID - holds the unique user ID the comment belongs to.
- commentID - stores a unique int value to identify the comment.

Ratings: this object stores the rating the user chose for the video. It also checks to see if the video was already rated by the user or not.

Ratings attributes:

- rating - holds the information about what the given rating of the video is by the user.
- rated - stores a boolean value to indicate whether the video has been rated by the logged in user before or not.
- videoID - holds the unique video ID the rating belongs to.
- userID - holds the unique user ID the rating belongs to.
- ratingID - holds a unique int value that identifies a comment.

DATA DESIGN

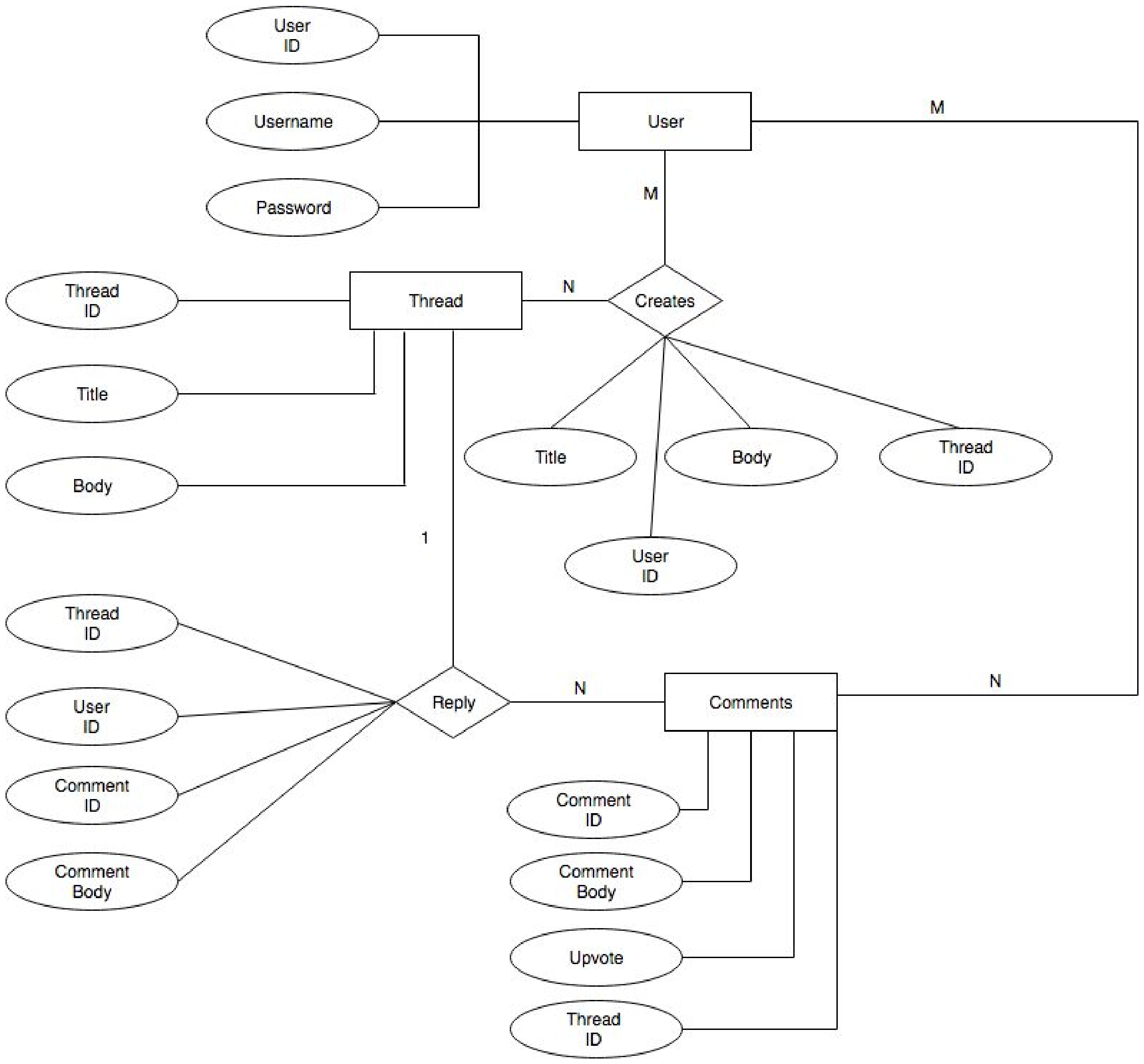
COMMUNITY FORUM FEATURE (JOSEPH RODRIGUEZ)

The class diagram above describes the classes that are necessary to develop the community forum and displays their database relationship.

The user class will interact with other class (Thread, and Comments) so that each comment or thread topic created will belong to a user. Therefore, a user must be logged in. The method contained within the user class will have login, register, and logout functions.

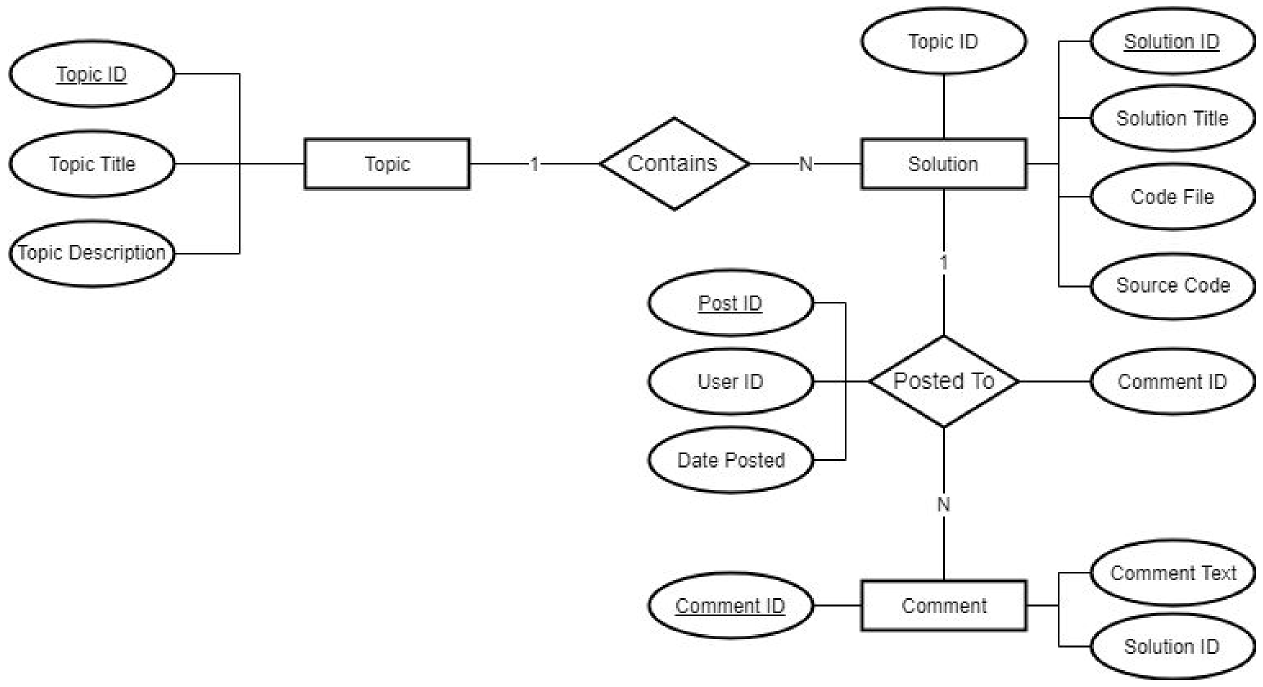
The comment class will have a many to many relation with both User and Thread. This will ensure that Thread can have one or more comments that belong to one or more different Users. The functions contained within the Comments class will allow users to utilize CRUD operations based on user permissions.

The Thread class contains a many to many relationship with Users so that many Users may have many Thread topics. It is also contains a one to many relation with comments because for every one Thread there should multiple Comments. The functions contained within the Thread class will allow for CRUD operations to take place because on User permissions.



CODING TOPICS AND SOLUTIONS FEATURE (By Louis Rivera)

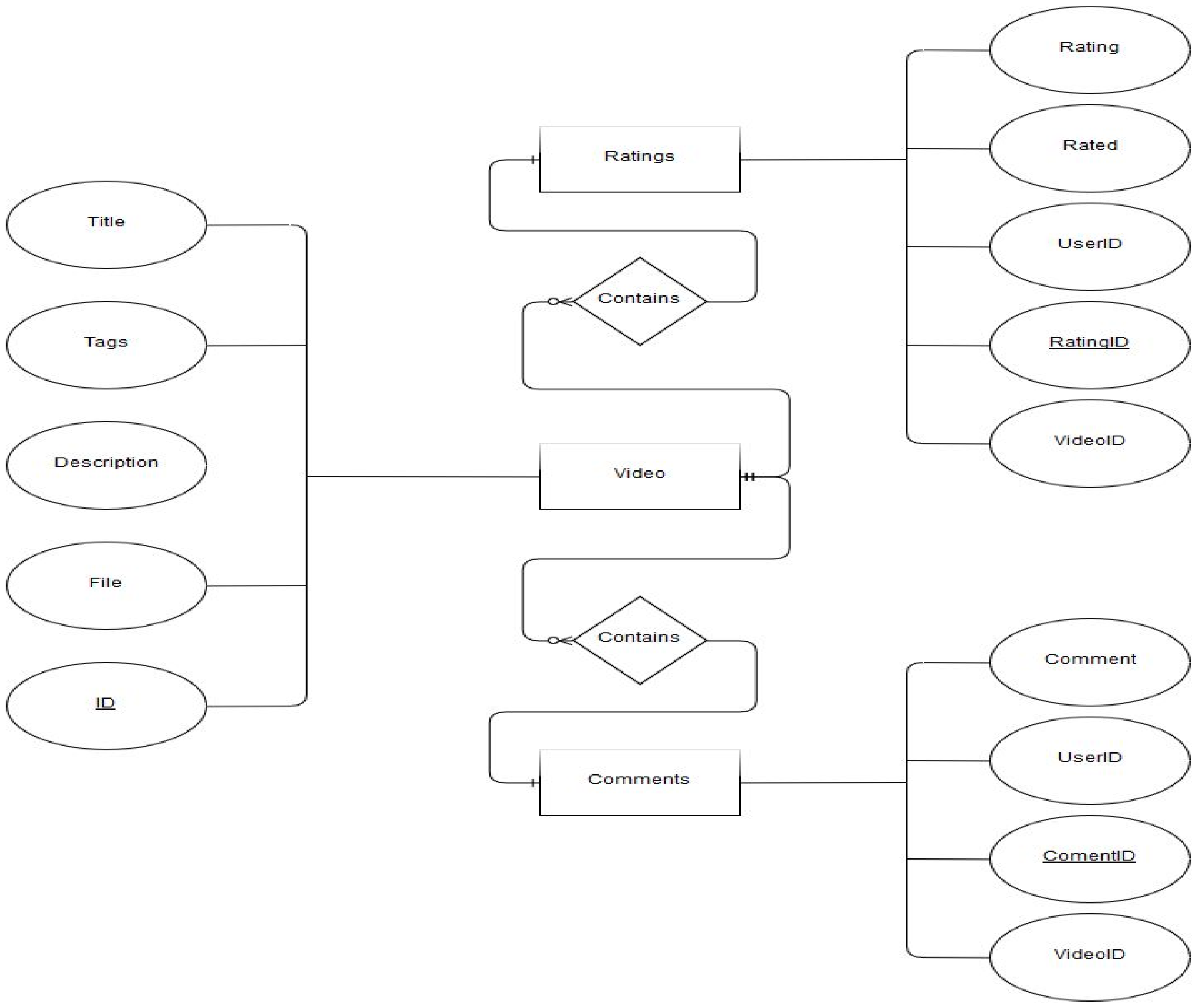
All relationships in the database will be one to many relationships. The relationship between topics and solutions is that topics contain solutions, but the relationship itself, “contains”, doesn’t have any attributes itself. The “posted to” relationship will keep track of the user and the date posted of any commented that gets posted to a solution.



Video Library Feature (by Rafel Salem)

There are three entities, two actions, and 14 attributes in total. The relationship between the video entity to the actions are one to many. And the relationships from actions to the other two entities are one. The actions are only composed of the action “contains”. This indicates the main entity “video” has two sub entities that are closely related to it. The primary keys for each entity is their name in addition to the abbreviation “id”.

The important thing to note is that the video entity has ratings and comments. The comments are the collection of comments the video contains and is not the comment the user holds in the video. The ratings are also a collection of ratings and not a single user’s ratings in a macro sense. On a micro sense, every user has their own unique comment and rating, and that is why there are IDs to identify both entities.



DATABASE DICTIONARY / SCHEMA

COMMUNITY FORUM FEATURE

The following tables are defined for the community forum module.

User Table			
Entity	**Type**	**Value**	**Description**
user_id	INT	Not Null	PK. Unique identifier of a user.
user_name	varchar(30)	Not Null	User’s unique display name
user_pw	varchar(50)	Not Null	User’s password

Thread Table			
Entity	**Type**	**Value**	**Description**
thread_id	INT	Not Null	PK. Unique identifier of a thread
thread_title	varchar(50)	Not Null	The name of topic used by user
thread_body	TEXT	Not Null	The body of the topic created by user.

Comment Table			
Entity	**Type**	**Value**	**Description**
comment_id	INT	Not Null	PK. Unique identifier of a comment.
thread_id	INT	Not Null	FK. Unique identifier of a thread
upvote	INT		Incrementing weighted score for a particular comment
comment_body	TEXT	Not Null	The comment body for which the user wants to express.

CODING TOPIC AND SOLUTIONS FEATURE (By Louis Rivera)

The following tables are defined for the "Coding Topics and Solution" module

Topic Table			
Entity	**Type**	**Value**	**Description**
topic_id	unsigned int	Not null	PK. Unique identifier of a topic.
topic_title	varchar(50)	Not null	Title of the topic.

topic_description	varchar(10,000)	Not null	A description of the topic being posted.

Solution Table			
Entity	**Type**	**Value**	**Description**
solution_id	unsigned int	Not Null	PK. Unique identifier of a solution.
topic_id	unsigned int	Not null	FK. Relationship key to the topic table.
solution_title	varchar(50)	Not null	Title of the solution.
source_code	varchar(10,000)		The solution posted in the form of text.
code_file	mediumblob		The solution posted in the form of a file.

Comment Table			
Entity	**Type**	**Value**	**Description**
comment_id	unsigned int	Not Null	PK. Unique identifier of a comment.
solution_id	unsigned int	Not null	FK. Relationship key to the solution table.
comment_text	varchar(10,000)		The text portion of the comment posted.

Posted_To Table			
Entity	**Type**	**Value**	**Description**
post_id	unsigned int	Not Null	PK. Unique identifier of a comment.
comment_id	unsigned int	Not null	FK. Relationship key to the comment table.
user_id	int	Not null	FK. Relationship key to the user table.
date_posted	Date	Not null	The date the comment was posted.

Video Library Feature (by Rafel Salem)

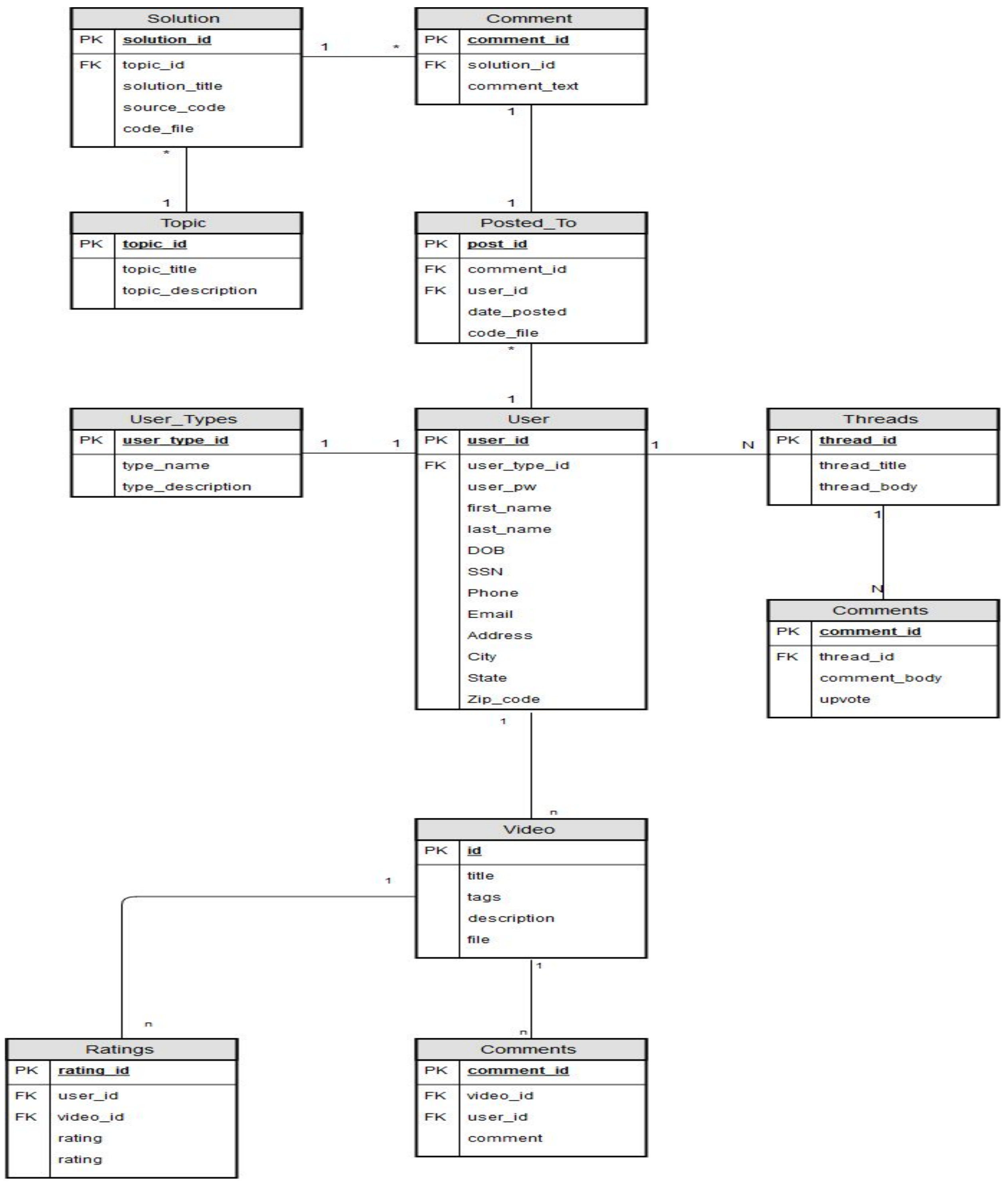
The following tables are defined for the Registration/Authentication module

Video			
Entity	**Type**	**Value**	**Description**
title	varchar(50)	Not Null	The title of the video.
tags	varchar(50)	Not Null	The tags that help identify the video easier.
description	varchar(300)	Not Null	A description of what the video is about.
file	DISTINCT	Not Null	A reference to the file object that holds the video.
id	INT	Not Null	PK. Unique identifier of a video.

Comment			
Entity	**Type**	**Value**	**Description**
rating	INT	Not Null	Stores the rating given by a user.
rated	BOOLEAN	Not Null	A value to tell if the user has rated the video or not.
user_id	INT	Not Null	FK. Ties a rating to a user.
rating_id	INT	Not Null	PK. A unique key that identifies user comments.
video_id	INT	Not Null	FK. Ties a rating to a video.

Rating			
Entity	**Type**	**Value**	**Description**
comment	varchar(300)	Not Null	A string that stores the comment written by a user.
user_id	INT	Not Null	FK. Ties a comment to a user.
comment_id	INT	Not Null	PK. A unique value that identifies the comment.
video_id	INT	Not Null	FK. Ties a comment to a video.

FULL DATABASE MODEL



REQUIREMENTS MATRIX

Module	*Use Case*	*Design Component*
Community Forum	*Create Thread Topic*	*User Interface: Create Thread* *ERD Diagram 1* *Component Model: Community Forum*
	Reply to Thread	*User Interface:Reply to Thread* *ERD Diagram 1* *Component Model: Community Forum*
	Upvote Comment	*User Interface: Upvote Comment* *ERD Diagram 1* *Component Model: Community Forum*
	Edit Comment	*User Interface: Edit Comment* *ERD Diagram 1* *Component Model: Community Forum*
	Delete Comment	*User Interface: Delete Comment* *ERD Diagram 1* *Component Model: Community Forum*
Coding Topics Module	*View Existing Code Topic*	*User Interface: View Existing Code Topic* *ERD Diagram 2* *Component Model: Coding Topics*

		Database Schema: Section 2
	Post New Coding Topic	*User Interface: Post New Coding Topic* *ERD Diagram 2* *Component Model: Coding Topics* *Database Schema: Section 2*
	Add New Coding Solution	*User Interface: Add New Coding Solution* *ERD Diagram 2* *Component Model: Coding Topics* *Database Schema: Section 2*
	Comment on a Coding Solution	*User Interface: Comment on a Coding Solution* *ERD Diagram 2* *Component Model: Coding Topics* *Database Schema: Section 2*
	Delete Topic	*User Interface: Delete Topic* *Component Model: Coding Topics*
Video Library	*Upload Video*	*User Interface: Upload Video* *ERD Diagram 3* *Component Model: Video Library* *DB Schema section3*
	Search Video	*User Interface: Search Video* *ERD Diagram 3*

		Component Model: Account Management *DB Schema section3*
	Comment Video	*User Interface: Comment Video* *ERD Diagram 3* *Component Model: Video Library* *DB Schema section3*
	Rate Video	*User Interface: Rate Video* *ERD Diagram 3* *Component Model: Video Library* *DB Schema section3*

APPENDICES

The web application provides different website functionality depending on the user type. For administrators, there's more control over the three features of the web application. Enabling them to delete topics, upload videos, and many other privileges.

The video library feature cannot provide a video streaming feature due to the server limitations. Therefore the user will have to download the video then view it. To remedy the lack of video streaming, the videos will have a description to help the user determine whether they want to download them or not for viewing.

The decision to provide unique functionality only to administrators is made to increase the security of the web application. Additionally, it allows uniform data manipulation as every functionality can be filtered by account type.