

**Testing Document**

**USC** CSCI 201

**1. Tests**

**1.1 Whiteboard and Documents**

**Test Case**

*Description*: Tests whether performing a draw event on a course whiteboard propagates the event to all connected users.

*Expected result*: All connected users receive a draw event spawned from another user.

*Steps*:

• Log in as a user *A* on one browser instance and log in as user *B* on another browser instance.

• Open a course whiteboard *W* on both user *A* and user *B*.

• As user *A*, draw a curved line of approximately 100px in length using the pen tool with a specific color, *C*.

• As user *B*, check that the drawn line of color *C* appears on the whiteboard instance.

**Test Case**

*Description*: Tests whether performing a draw event on a course whiteboard persists across page loads.

*Expected result*: Drawn objects by a user should persist and load for future visits by all users.

*Steps*:

• Log in as a user *A* on one browser instance and log in as user *B* on another browser instance.

• Open a course whiteboard *W* on both user *A* and user *B*.

• As user *A*, draw a curved line of approximately 100px in length using the pen tool with a specific color, *C*.

• As user *A* and *B*, reload the page and check that the drawn line of color *C* appears on the whiteboard instance.

**Test Case**

*Description*: Tests whether changing the selected draw tool on the whiteboard sends the correct type of draw action.

*Expected result*: Changing the selected draw tool (e.g. from pen to ellipse) should the sent draw action for all connected clients.

• Log in as a user *A* on one browser instance and log in as user *B* on another browser instance.

• Open a course whiteboard *W* on both user *A* and user *B*.

• As user *A*, draw a curved line of approximately 100px in length using the pen tool with a specific color, *C*.

• As user *A*, switch to the ellipse tool and draw an ellipse of approximately 100px by 50px with the color *C*.

• As user *B*, check that the drawn line of color *C* appears on the whiteboard instance.

• As user *B*, check that the drawn ellipse of color *C* appears on the whiteboard instance.

**Test Case**

*Description*: Tests whether changing the selected color on the whiteboard sends the correct color for future draw events.

*Expected result*: Switching colors should cause future draw events to take on the new color.

• Log in as a user *A* on one browser instance.

• Open a course whiteboard *W* on user *A*.

• As user *A*, draw a curved line of approximately 100px in length using the pen tool with a specific color, *C*.

• As user *A*, change the selected color to *C2*.

• As user *A*, draw a curved line of approximately 100px in length using the pen tool and check that the new line has color *C2*.

**Test Case**

*Description*: Tests whether a user can add a new whiteboard to a course.

*Expected result*: Pressing the button to add a new whiteboard to a course adds and opens the new whiteboard.

• Log in as a user *A* on one browser instance and log in as user *B* on another browser instance.

• Create a new whiteboard as user *A* on course *K*.

• As user *A*, check that the page now shows the newly created whiteboard *W*.

• As user *B*, open the link to *W* and check that the whiteboard *W* appears as expected.

**Test Case**

*Description*: Tests whether a user can change the view permissions of a whiteboard.

*Expected result*: Selecting a new whiteboard permission should allow or disallow others to view the whiteboard depending on the setting.

• Log in as a user *A* on one browser instance and log in as user *B* on another browser instance.

• Create a new whiteboard as user *A* on course *K*.

• As user *A*, check that the page now shows the newly created whiteboard *W*.

• As user *A*, set the whiteboard access permissions to “Only Me”.

• As user *B*, open the private link to *W* and check access to *W* is not permitted.

• As user *A*, set the whiteboard access permissions to “Private”.

• As user *B*, open the link to *W* and check that the whiteboard *W* appears as expected.

**Test Case**

*Description*: Tests whether uploading a document to a course shows the document in the course documents list.

*Expected result*: Uploading a document should update the course documents list with the same metadata as the uploaded file.

• Log in as a user *A* on one browser instance.

• As user *A*, Upload a document *D* to a course *K*.

• As user *A*, check that the document *D* appears in *K*’s document list with the correct metadata.

**Test Case**

*Description*: Tests whether an uploaded document can be downloaded.

*Expected result*: Uploading a document as one user should allow other users to download the same document.

• Log in as a user *A* on one browser instance and log in as user *B* on another browser instance.

• As user *A*, upload a document *D* to a course *K*.

• As user *B*, check that the document *D* appears in *K*’s document list with the correct metadata.

• As user *B*, download the document *D* under *K* and verify that the SHA-256 matches the original document uploaded by user *A*.

**Test Case**

*Description*: Tests whether documents can be searched by name.

*Expected result*: Uploading a document with a filename and file title should allow the document to be searched by other users.

• Log in as a user *A* on one browser instance and log in as user *B* on another browser instance.

• As user *A*, upload a document *D* to a course *K*.

• As user *B*, check that the document *D* appears in *K*’s document list with the correct metadata.

• As user *B*, enter the document filename of document *D* through the search bar module and check that the document *D* appears in the filtered results as expected.

• As user *B*, enter the document file title of document *D* through the search bar module and check that the document *D* appears in the filtered results as expected.

**Test Case**

*Description*: Tests whether documents can be sorted by date added, filename, or file size.

*Expected result*: Selecting a sort order for a course documents list should order the documents listed.

• Log in as a user *A* on one browser instance.

• As user *A*, navigate to a course *K* with a minimum of three documents.

• As user *A*, select the date added sort filter and check that the documents listed are sorted by date.

• As user *A*, select the filename sort filter and check that the documents listed are sorted by filename.

• As user *A*, select the file size sort filter and check that the documents listed are sorted by file size.

**1.2 Forum and Chat**

**Test Case**

*Description*: Tests whether a new post can be created after clicking the “plus icon” on the forum page.

*Expected result*:A new post is created.

*Steps*:

• Navigate to the forum page and click the “plus icon”.

• The user should be prompted to create a new post.

• Fill in the appropriate fields for the new post and submit.

• The forum page should now display this new post.

**Test Case**

*Description*: Tests whether upvoting a post correctly updates the upvote count.

*Expected result*:Upvote counter increments by 1.

*Steps*:

• Find an existing post on the forum page (If none exists, create a new post).

• Click the orange upvote button.

• The expected change in the upvote counter should be observed.

**Test Case**

*Description*: Tests whether downvoting a post correctly updates the upvote count.

*Expected result*:Upvote counter decrements by 1.

*Steps*:

• Find an existing post on the forum page (If none exists, create a new post).

• Click the blue downvote button.

• The expected change in the upvote counter should be observed.

**Test Case**

*Description*: Tests whether title and body of a post is properly displayed upon the creation of a new post.

*Expected result*:The title and body of a post is displayed as a snapshot on the forum page.

*Steps*:

• Find an existing post on the forum page (If none exists, create a new post).

• The associated title and body for that post should be displayed as a snapshot.

**Test Case**

*Description*: Tests whether a post on the forum page is clickable and navigates to the associated post page.

*Expected result*:Clicking on a post navigates to the post’s page; the page should be correctly formatted.

*Steps*:

• Find an existing post on the forum page (If none exists, create a new post).

• Click on this post.

• The user should be taken to the post page associated with the post they clicked on.

• The post’s title and body should be displayed at the top of this page.

• The comments of the post should be displayed on this page below the body of the post.

**Test Case**

*Description*: Tests whether the comments are sorted in descending order based on their number of upvotes.

*Expected result*:Comments are sorted from top to bottom by number of votes .

*Steps*:

• Find an existing post on the forum page (If none exist, create a new post).

• Click on this post.

• The user should be taken to the post page associated with the post they clicked on.

• Ensure that there are at least two comments on this post of different ratings (If none exist, add two comments and only upvote one of them).

• The comments should be sorted in descending order based on their number of upvotes.

**Test Case**

*Description*: Tests whether a user can add a new comment to a post on the forum page.

*Expected result*:Clicking on the “plus” button should prompt a user to add a new comment, and the post page should be updated accordingly.

*Steps*:

• Find an existing post on the forum page (If none exists, create a new post).

• Click on this post.

• The user should be taken to the post page associated with the post they clicked on.

• Click on the “plus” button to add a new comment.

• Fill in the appropriate fields then submit.

• The user should be navigated back to the course page and observe updates to the page which include this new comment.

**Test Case**

*Description*: Tests whether a user can reply to a comment on the post page.

*Expected result*:Clicking on the “reply” button should expand a text input below the item in which the user can enter their reply. The reply should appear under the comment.

*Steps*:

• Find an existing post on the forum page (If none exists, create a new post).

• Click on this post.

• The user should be taken to the post page associated with the post they clicked on.

• Find an existing comment on the forum page (If none exist, create a new comment).

• Click the “reply” button.

• A text input field should be displayed for a reply.

• Type a reply then submit it.

• This new reply should appear under the associated comment

**Test Case**

*Description*: Tests search for forum posts.

*Expected result*: The posts whose titles contain the keyword appear in a list, their titles should be clickable and navigate to the associated post.

• Search for a title of a forum post which has been previously created.

• The post(s) title(s) should appear in a list.

• Click a post in the list.

• The user should be redirected to that post’s page with the appropriate information of the post displayed.

**Test Case**

*Description*: Tests whether the course channel chat persists on all tabs of the course.

*Expected result*: The course channel chat should persist on all tabs of the course as a floating chat box.

• Navigate to each tab within a course.

• Ensure that the course channel chat persists on each tab.

**Test Case**

*Description*: Tests whether a user can send a message to the course channel chat.

*Expected result*: After sending a message, the user’s message should appear within the course channel chat.

• Type a message in the chat box and send it.

• The message should appear within the chat log of the chat box.

**Test Case**

*Description*: Tests whether course channel chat lists chat in chronological order with the newest messages at the bottom of the chat log.

*Expected result*: Chat log lists messages in chronological order

• Type a message in the chat box and send it.

• The message should appear within the chat log of the chat box.

• Type another message in the chat box and send it.

• The message should appear within the chat log of the chat box and appear below the previous message.

**Test Case**

*Description*: Tests whether course channel chat is minimizable.

*Expected result*: Clicking on the title of the chat box minimizes it.

• Click on the title of the chat box.

• The box should be minimized.

**Test Case**

*Description*: Tests whether a user’s course chat is properly updated with new chat messages from other users.

*Expected result*: A user’s course chat is updated when another user sends a message.

• Log into two different user accounts who share a course.

• Navigate to this shared course page.

• In User A, send a message over the course chat.

• Ensure this message appears in User B’s chat log.

**1.3 Login, Registration, and User Settings**

**Test Case**

*Description:* Tests whether the user needs the correct credentials to access the site.

*Expected result:* Redirects user to their feed page

*Steps:*

• Try to login with nonexistent username

A notification that the username does not exist and to try again should be observed

•Try to login with existing username but incorrect password

A notification that the password does not match username and to try again should be observed

**Test Case**

*Description:* Tests whether the user is signing up with valid credentials

*Expected result:* Redirects user to their feed page (which should be empty initially)

*Steps:*

• Try to register for a new account with an already existing username

A notification that the username is already taken should be observed

• Try to register with a unique username but with a password of insufficient length

A notification that the password is not long enough should be observed

**Test Case**

*Description:* Tests whether a request to change the user’s password is valid

*Expected result:* The user’s password is changed to a new, requested one

*Steps:*

• Navigate to the user’s settings from the profile page, and from here change the password

• The user will be queried for their current password; if incorrect, they will continue to be queried until they either exit or provide the correct one

• The user will be queried for a new password; try to enter the current password and/or a password of insufficient length

A notification that the password is not long enough or is unchanged should be observed

**Test Case**

*Description:* Tests whether a request to delete the user’s account goes through

*Expected result:* The user’s account is deleted and they are redirected to the site landing page

*Steps:*

• Navigate to the user’s settings from the profile page, and from here delete your account

• The user will be queried for their current password; if incorrect, they will continue to be queried until they either exit or provide the correct one

• The user will be asked if they are sure they would like to delete their account, and that this action is final and irreversible

• If they choose to proceed, the account should be deleted and they should be redirected

**Test Case**

*Description:* Tests whether a request to change the profile picture goes through

*Expected result:* The user’s profile picture is changed sitewide

*Steps:*

• Navigate to the user’s settings from the profile page, and from here change your profile picture

• The user will be asked to upload a picture from their client of type .png or .jpg

• If the file type is not one of those two, the user will be requested to provide a valid file

• Upon successful retrieval of the file, the user’s profile picture/icon should be updated sitewide

**1.4 Profile and Feed**

**Test Case**

*Description*: Tests that the correct information is loaded on the user’s profile page.

*Expected result*: Profile page displays the information of the logged in user.

*Steps:*

• Click on the profile button.

The profile image, name, and other information should be that of the logged in user. The page should not have an “Add Friend” button.

**Test Case**

*Description*: Tests that the page’s class list navigates to the correct classes.

*Expected result*: The page of the clicked class is shown.

*Steps:*

• Click on a class from the class list.

That class’s class page is displayed with the correct information.

**Test Case**

*Description*: Tests that the page’s friend list links to that user’s profile, and that the friend’s profile indicates that they are friends.

*Expected result*: Profile page displays the information of the user whose username was clicked.

*Steps:*

• Click on a friend’s name in the friend list.

Redirect to that person’s profile page. Information should now be that of the clicked user. The “Add Friend” button should be greyed out and say “Friends”.

**Test Case**

*Description*: Tests search for users.

*Expected result*: The users whose username contains the keyword appear in a list, their names are clickable and navigate to their profiles.

• Search a random username.

User’s name and image should appear in a list.

• Click user’s name.

Redirect to that user’s profile. Appropriate information is shown and the “Add Friend” button displays the correct text and behavior based on the logged in user’s relationship with that user.

**Test Case**

*Description*: Tests that a user who the logged in user is not friends with is able to be added from their profile page.

*Expected result*: Add friend button adds the user to the friend list and changes the button accordingly

• Navigate to a user’s page who is not friends with the logged in user.

The “Add Friend” button should be clickable and read “Add Friend”.

• Click the “Add Friend” button.

Button should now read “Friends” and the user should be added to the friends list of the logged in user.

**Test Case**

*Description*: Feed displays correct information.

*Expected result*: Feed displays forum posts and document uploads of the logged in user’s friends. Also displays any instructor actions from the user’s classes.

• Navigate to the home page.

Feed displays actions of friends of the user and Instructor actions are displayed. Actions are sorted chronologically.

**Test Case**

*Description*: Feed usernames are clickable.

*Expected result*: Navigate to the profile page of the clicked user.

• Click a username in the feed.

Profile page displays the information of the clicked user.

**Test Case**

*Description*: Feed actions are clickable.

*Expected result*: Navigate to the post or document of the action.

• Click an action in the feed.

The correct document or forum post is navigated to.

**1.5 User Hierarchy**

**Test Case**

*Description*: Tests the read-only permissions of the guest user.

*Expected result*: Guest user should be able to view all content, but not modify or add new content.

• On the login page, click continue as guest.

• Feed is empty, but displays a prompt to search for a user or class.

• Search for a class and click on the result.

Class chat, forum, whiteboard, and documents should be viewable.

User is not able to add chat messages, forum posts, documents, or whiteboard items.

**Test Case**

*Description*: Tests the ability of a user to join a new class

*Expected result*: The course should be accessible from the Homepage and the user’s Profile Page

• Search for a class using the search bar at the top of any page and navigate to the desired class

• Choose to join the class on the class’ page, if not already joined

• The user should now be a part of the course and should see notifications from it in their feed

**Test Case**

*Description*: Tests the ability of the TA to add students to their classes

*Expected result*: Added student is now in the class.

• Login as a user with TA privileges and navigate to the class that those privileges are for.

• On the class page click “Add Student”.

Student now appears in the list of registered students for the class.

**Test Case**

*Description*: Tests the ability of the TA to remove students from their classes

*Expected result*: Removed student is no longer a member of the class

• Login as a user with TA privileges and navigate to the class that those privileges are for.

• On the class page click the “Remove” button next a student’s name in the list of registered students.

Student no longer appears in the list.

**Test Case**

*Description*: Tests the ability of the TA to delete content from their classes

*Expected result*: Content is deleted from the class

• Login as a user with TA privileges and navigate to the class that those privileges are for.

• On the a forum post or documents or chat message click the “Remove” button

The removed content is no longer visible.

**Test Case**

*Description*: Tests the ability of the Instructor, Moderator, or Admin to add a TA to a class *Expected result*: A student in the class is elevated to TA status for that class

• Login as a user with TA/Moderator/Admin privileges and navigate to the class that you want to add a TA to

• Navigate to the list of students that have joined the class, and select the student you would like to make a TA

The selected user should now have TA privileges for that class

**Test Case**

*Description*: Tests the ability of the moderator to suspend a user

*Expected result*: User has read-only permissions for the given amount of time.

• Login as a user with moderator privileges.

• On a user’s page, click the suspend button and enter the specified amount of time.

That user’s account is not able to create or modify content for the specified amount of time.

**Test Case**

*Description*: Tests the ability of a moderator to create an Instructor

*Expected result*: User now has the requested permissions.

• Login as a user with moderator privileges.

• On a user’s page, click the “Add as Instructor” button

• You will be queried for the classID of the course you want to add the Instructor to

The user now has Instructor privileges for that course.

**Test Case**

*Description*: Tests the ability of an admin to create a Moderator or an Admin

*Expected result*: User now has the requested permissions.

• Login as a user with admin privileges.

• On a user’s page, click the “Mod” or “Admin” button

The user now has moderator or admin privileges based on the button pressed.

**Test Case**

*Description*: Tests the ability of an admin to create a class

*Expected result*: New class is created.

• Login as a user with admin privileges.

• On the Home page click “Create Class” and enter the class name.

New class is created and joinable by students.

**Test Case**

*Description*: Tests the ability of an admin to delete a class

*Expected result*: The class is deleted.

• Login as a user with admin privileges.

• On a class page click “Delete Class” and click “Confirm”.

The class is deleted.