Synchronous

Team 10

Sprint 3 Test Cases Document

User Story 1:

1. Multiple Users can create a Whiteboard App
   a. Input: Users click on a button to create a whiteboard app
   b. Output: Whiteboard app is created and appears on screen in the workspace
2. Multiple users can create multiple whiteboard apps.
   a. Input: More than one user should create a whiteboard app
   b. Output: Multiple whiteboards are created and appear on screen in workspace, each created by a different person
3. Multiple users can see whiteboard
   a. Input: A Created whiteboard, and logging on as multiple users each with their own session
   b. Output: Each user is able to see the whiteboard
4. Multiple Users can create whiteboard content
   a. Input: Each user should try to draw on a newly created whiteboard
   b. Output: Each user's input to appear on the whiteboard
5. Multiple users can change whiteboard content
   a. Input: Each user should try to alter the whiteboard content, by drawing, erasing, or changing the content in some way.
   b. Output: The changes that the users make should appear on the whiteboard
6. Whiteboard content changes persist upon exit
   a. Input: Inputting changes to the whiteboard content
   b. Output: Changes should be reflected and should persist after exit.
7. Multiple Users can see whiteboard changes in real time
   a. Input: One user should see the whiteboard while another user is providing input
   b. Output: The user should be able to see the other user inputting data on the whiteboard in real time.
8. Multiple users can see whiteboard changes after refresh
   a. Input: Input changes to the whiteboard
   b. Output: Changes should be reflected and should persist after refresh.

User Story 2:

1. Multiple users can draw on whiteboard with any color
   a. Input: Multiple users should draw on the whiteboard, while choosing a random color to draw.
   b. Output: The different output should be reflected on the whiteboard, with the ink reflecting the chosen colors.
2. Colors persist upon exit
   a. Input: Input ink on whiteboard with a chosen color.
   b. Output: Ink with color should appear and persist even after workspace exit.
3. Colors persist upon refresh.

    a. Input: Input ink on whiteboard with a chosen color.
    b. Output: Ink with color should appear and persist even after workspace refresh.
4. Multiple users can change color of ink on whiteboard
    a. Input: Multiple users should draw on the whiteboard, and then try and change the color of the ink already on the whiteboard
    b. Output: The ink should reflect the changes and appear with the new chosen color.
5. Multiple users can change the color of the pen.
    a. Input: Multiple users should try and change the color of their pen and then draw on the whiteboard
    b. Output: The ink on the whiteboard should appear with the changed color.
6. Color change to pen persists after change
    a. Input: Change the color of the pen
    b. Output: The color change should persist even after input or tool switch
7. Color change to ink on whiteboard persists after change
    a. Input: Change the color of ink already on the screen
    b. Output: The color change should persist.
8. Colors persist across all user screens.
    a. Input: Multiple users should try to view the whiteboard with ink of different colors
    b. Output: The users should be able to see all the ink with all the colors.

User Story 3:

1. Change the marker size
    a. Input: User should try and change the size of the marker and draw on whiteboard
    b. Output: The ink should be the size of the newly requested marker size.
2. Change the ink size
    a. Input: Change the size of ink already on the whiteboard
    b. Output: The ink should be the newly requested size
3. Multiple users can change size of marker or ink
    a. Input: Multiple users should try and change the size of the marker and draw, and change the size of the ink already on the screen
    b. Output: The markers and the ink should be the newly requested size.
4. Multiple users can see the size change of the ink
    a. Input: After changing the ink size, each user should open the workspace and observe the screen
    b. Output: The users should be able to see the change in ink size.
5. Multiple users can draw with the newly changed marker
    a. Input: after marker size change, users should be able to draw with the marker.
    b. Output: newly resized ink should appear on the screen.
6. The drawing post size change stays the same size
    a. Input: User should draw after changing the size.
    b. Output: The size should persist after change.
7. The ink persists post refresh

    a. Input: Refresh the page with some ink on the whiteboard
    b. Output: the ink should persist after refresh
8. The ink persists upon exit
    a. Input: Exit the workspace and reenter the workspace with some ink on the whiteboard
    b. Output: The ink should persist after exit and reentry.

User Story 4:

1. Multiple users can erase strokes
    a. Input: Multiple users should try and erase ink on the whiteboard
    b. Output: The ink should disappear.
2. Strokes do not persist upon refresh or exit
    a. Input: After erasing, refresh the page or exit the workspace and reenter.
    b. Output: The ink should still be erased.
3. Changes are reflected across all screens
    a. Input: Erase ink on the screen
    b. Output: The ink should disappear from all user instances of the workspace.
4. Post erase, users can add ink in the same spot.
    a. Input: Erase ink and then draw in the same spot.
    b. Output: The new ink should appear.

User Story 5:

1. Users can see who is currently drawing
    a. Input: One user should watch the screen while another user draws
    b. Output: The user who is drawing should be identified and highlighted for the other user to see.
2. Users can differentiate between other current users
    a. Input: One user should watch the screen while multiple users draw on screen
    b. Output: The viewing user should be able to see each person drawing and be able to identify which user is which
3. Users can see other users' pointers while drawing
    a. Input: One user should watch the screen while another user draws
    b. Output: The drawing user's pointer should appear on the viewing user's screen, and should be identifiable by user
4. Users can see other users' pointers while not drawing.
    a. Input: One user should watch the screen while another user moves the pointer
    b. Output: The moving user's pointer should should appear on the viewing user's screen, and should be identifiable by user

User Story 6:

1. Users can upload an image to the whiteboard
    a. Input: User should try and upload an image to the whiteboard

    b. Output: Image should appear as a background for the whiteboard
2. Users can draw on the image
    a. Input: Users should draw on the image
    b. Output: The ink should persist on the image
3. Users changes persist
    a. Input: Users should modify the drawing
    b. Output: The changes should persist, even after a refresh or an exit.
4. Uploaded drawing persists
    a. Input: A user should upload an image.
    b. Output: The image should persist even after refresh or an exit

User Story 7:

1. Users can download the contents of the whiteboard
    a. Input: User should try and download the contents of the whiteboard
    b. Output: The whiteboard contents are downloaded and saved on to the local system.
2. Users can download the contents of the whiteboard as a pdf
    a. Input: Users should specify the file type as a pdf and download the contents as a pdf
    b. Output:The whiteboard contents are downloaded as a pdf and saved on to the local system.
3. Users can download the contents of the whiteboard as a jpg
    a. Input: Users should specify the file type as a jpg and download the contents as a jpg
    b. Output: The whiteboard contents are downloaded as a jpg and saved on to the local system.
4. Users can download the contents of the whiteboard as a png
    a. Input: Users should specify the file type as a png and download the contents as a pdf
    b. Output:The whiteboard contents are downloaded as a png and saved on to the local system.

User Story 8:

1. Users can add shapes
    a. Input: User should try and add a shape to the workspace
    b. Output: The shape should appear in the workspace
2. Users can add tables
    a. Input: User should try and add a table to the workspace
    b. Output: The table should appear in the workspace
3. Users can add data to tables
    a. Input: User should add data to the tables
    b. Output: The data appears in the table and persists
4. Users can change the characteristics of shapes

    a. Input: User should try and change some characteristics of the shape, like color, size etc.
    b. Output: the changes should appear and persist.

User Story 9:

1. Users can download an entire workspace
    a. Input: Users should try and download the entire workspace
    b. Output: The workspace should be downloaded to the local system.
2. Users can download it as a zip
    a. Input: users should specify downloading the workspace as a zip file
    b. Output: the workspace should be compressed as a zip file and then downloaded to the local system
3. Users can download all the apps at once
    a. Input: Users should try and download all the apps contents at once
    b. Output: The contents should be downloaded to the local system
4. Users can use a UI based option to download the workspace.
    a. Input: The user should have a UI option like a menu or a button to download the workspace and should click on that option to download the workspace
    b. Output: the workspace is downloaded to system.

User Story 10:

1. Users can upload a previously downloaded workspace
    a. Input: Users should try and upload a previous workspace
    b. Output: The workspace should appear on the screen
2. All apps reflect the contents of the uploaded workspace
    a. Input: Users should check the contents of the uploaded workspace and compare with the actual contents of the previously downloaded workspace
    b. Output: The content should match
3. The data in the workspace is of the newly uploaded workspace and not the previous one.
    a. Input: Compare the output of the new workspace with the content of the old workspace
    b. Output: The new workspace content should not have the content of the old workspace.
4. All users can see the uploaded workspace
    a. Input: multiple users should log on and access the workspace with the uploaded content
    b. Output: Each user should be able to see the workspace and its contents.

User Story 11:

1. Users can split the workspace into multiple workspaces

    a. Input: Click on a tab to create a new workspace within the overall workspace
    b. Output: A new workspace should be created with its own sidebar and buttons
2. Each tab has its own contents
    a. Input: add apps in multiple workspaces
    b. Output: The workspaces should have their own apps and data.
3. Each tab's data persists
    a. Input: Enter data in each workspace and switch between workspaces
    b. Output: The data in each workspace should persist.
4. All users can see each workspace
    a. Input: Multiple users should log onto the workspace and navigate between tabs
    b. Output: They should be able to see each and every workspace with data reflected in each tab.

User Story 12:

1. Each user can edit data after lost connection
    a. Input: Disconnect the system from the internet and try to enter data
    b. Output: data should change with each new entry
2. Each user can see the changes synced after resumed connection
    a. Input: Reconnect the system and let the workspace resync
    b. Output: The workspace should show all the data, including the new data added before resync
3. Changes persist after resync
    a. Input: After resync try to add more changes
    b. Output: The changes added should be added to workspace and changes should be updated
4. No loss of data post resync.
    a. Input: Post resync try to refresh the page or exit and reenter the workspace.
    b. Output: The data should still be in the workspace.