System Tests

NOTE: Sprint 1 was basically just design so none of those acceptance criteria are used for the acceptance tests. Also, Sprint 2's user stories were not stories and they were moved to Spring 3.

Test scenario: Validation of registering a new account
Pre-condition:

1. The application frontend and backend should be running

steps to reproduce:

1. Launch the application
2. Click on New User
3. Enter the username
4. Enter the password
5. Click Register
6. Log out after each successful registration

User stories covered: sprint 3 user story 1, sprint 4 user story 4

scenario 1:
Test description: Enter a valid username and password
Test data: {username: "john", password: "123"}
Expected behavior: successful registration and logged in automatically and redirected to chatpage
Actual behavior: successful registration and logged in automatically and redirected to chatpage
Result: Pass
Executed by: Brad

scenario 2:
Test description: Enter a invalid username and valid password
Test data: {username: "john_&%", password: "123"}
Expected behavior: unsuccessful registration, stay on register page
Actual behavior: unsuccessful registration, stay on register page
Result: Pass
Executed by: Brad

scenario 3:
Test description: Enter an invalid username and valid password
Test data: {username: "aaabbbcccdddeeefffggghhhiiijjjkkklllmmmnnnoooppp", password: "123"}
Expected behavior: unsuccessful registration, stay on register page
Actual behavior: unsuccessful registration, stay on register page
Result: Pass
Executed by: Brad

scenario 4:
Test description: Enter a valid username and invalid password
Test data: {username: “john”, password:
“1234567890123456789012345678901234567890123456789012345678901234567890”}
Expected behavior: unsuccessful registration, stay on register page
Actual behavior: unsuccessful registration, stay on register page
Result: Pass
Executed by: Brad

scenario 5:
Test description: Enter a invalid username and invalid password
Test data: {username: “aaabbbcccdddeeefffggghhhiiijjjkkklllmmmnnnoooppp”, password:
“1234567890123456789012345678901234567890123456789012345678901234567890”}
Expected behavior: unsuccessful registration, stay on register page
Actual behavior: unsuccessful registration, stay on register page
Result: Pass
Executed by: Brad

scenario 6:
Test description: Enter a valid username and password
Test data: {username: “ken”, password: “12345”}
Expected behavior: successful registration and logged in automatically and redirected to chatpage
Actual behavior: successful registration and logged in automatically and redirected to chatpage
Result: Pass
Executed by: Brad

Test scenario: Logging in to an account
Pre-condition:

1. The application frontend and backend should be running
2. An account already exists with credentials {username: “john”, password: “123”}

Steps to reproduce:

1. Launch the application
2. Go to the /login page (should be redirected here by default)
3. Enter the username
4. Enter the password
5. Click login button
6. Logout after each successful login

User stories covered: Sprint 3 story 1

Scenario 1:
Test Description: Enter correct username, correct password

Test Data: {username: “john”, password: “123”}
Expected Behavior: Log in succeeds, you are redirected to /chatpage
Actual Behavior: Log in succeeds, you are redirected to /chatpage
Result: Pass
Executed by: Nithin

Scenario 2:
Test Description: Enter correct username, incorrect password
Test Data: {username: “john”, password: “456”}
Expected Behavior: Log in fails, you stay on the login page and an error alert is displayed
Actual Behavior: Log in fails, you stay on the login page and an error alert is displayed
Result: Pass
Executed by: Nithin

Scenario 3:
Test Description: Enter incorrect username, correct password
Test Data: {username: “abcdefg”, password: “123”}
Expected Behavior: Log in fails, you stay on the login page and an error alert is displayed
Actual Behavior: Log in fails, you stay on the login page and an error alert is displayed
Result: Pass
Executed by: Nithin

Scenario 4:
Test Description: Enter incorrect username, incorrect password
Test Data: {username: “blahblahblah”, password: “@#$%^&*”}
Expected Behavior: Log in fails, you stay on the login page and an error alert is displayed
Actual Behavior: Log in fails, you stay on the login page and an error alert is displayed
Result: Pass
Executed by: Nithin

Test scenario: Create a new chat room
Pre-condition:
1. The application frontend and backend should be running

steps to reproduce:
1. Launch the application
2. Go to the /login page (should be redirected here by default)
3. Enter a username “john” and password “123”
4. Click login button
5. Click create chat button
6. Enter chat name
7. select users
8. Click submit button

9. For scenario 4 only, logout
10. Repeat steps 2-4 but with username “ken” and password “12345”

User stories covered: sprint 3 user story 2, sprint 4 user story 4

scenario 1:
Test description: Enter a valid chatname and add a user
Test data: {chatname: “my chat room”, users: [“ken”]}
Expected behavior: successful chat creation, new chat added to chat list with users john and ken
Actual behavior: successful chat creation, new chat added to chat list with users john and ken
Result: Pass
Executed by: Brad

scenario 2:
Test description: Enter a valid chatname and add no user
Test data: {chatname: “alone”, users:[]}
Expected behavior: successful chat creation, new chat added to chat list with just user john
Actual behavior: successful chat creation, new chat added to chat list with just user john

scenario 3:
Test description: Enter an invalid chatname and add no user
Test data: {chatname: “i can’t do this ”, users:[“ken”]}
Expected behavior: unsuccessful chat creation, stay on chat creation dialog
Actual behavior: unsuccessful chat creation, stay on chat creation dialog

scenario 4:
Test description: Enter a valid chatname and add no user
Test data: {chatname: “alone2”, users:[]}
Expected behavior: successful chat creation, new chat added to chat list with just user john, user ken cannot see chat room “alone2”
Actual behavior: successful chat creation, new chat added to chat list with just user john, user ken cannot see chat room “alone2”

Test scenario: Selecting a chat room
Pre-condition:

1. The application frontend and backend should be running
2. These two accounts have been registered: {username: “john”, password: “123”} and {username: “ken”, password: “12345”}
3. The following chatroom has been created by “john” (besides “general”): {chatname: “my chat room”, users: [“ken”]}

steps to reproduce:

1. Launch the application
2. In window 1, log in as john to get redirected to /chatpage
3. In window 2, log in as ken to get redirected to /chatpage
4. Select chat on the left, type message in the bottom textbox

User stories covered: sprint 3 user story 2, sprint 4 user story 1

Scenario 1:
Test description: Send message from john (window 1) to general chat
Test data: {username: “john”, chatname: “general”, message: “message1”}
Expected behavior: “message1” text bubble appears at bottom of conversation panel when “general” is selected. When “my chat room” is selected, it disappears.
Actual behavior: Same

Scenario 2:
Test description: Send message from ken (window 2) to general chat
Test data: {username: “ken”, chatname: “general”, message: “message2”}
Expected behavior: “message2” text bubble appears at bottom of conversation panel when “general” is selected (below “message1”). When “my chat room” is selected, it disappears.
Actual behavior: Same

Scenario 3:
Test description: Send message from john (window 1) to “my chat room” chat
Test data: {username: “john”, chatname: “my chat room”, message: “message3”}
Expected behavior: “message3” text bubble appears at bottom of conversation panel when “my chat room” is selected. When “general” is selected, the two old messages “message1” and “message2” appear instead.
Actual behavior: Same

Scenario 4:
Test description: Send message from ken (window 2) to “my chat room” chat
Test data: {username: “ken”, chatname: “my chat room”, message: “message4”}
Expected behavior: “message4” text bubble appears at bottom of conversation panel when “my chat room” is selected (below “message3”). When “general” is selected, the two old messages “message1” and “message2” appear instead.
Actual behavior: Same

Unit Tests

Frontend

Module 1: Login

- Equivalence classes: valid credentials and invalid credentials
    - One test checks that valid credentials (those that result in success response from backend) result in a successful login
    - Another checks that invalid credentials (those that result in a failure response from the backend) result in an unsuccessful login
- Tested by Nithin

Module 2: CreateChat
- Equivalence classes: valid chat name and creation
    - Test checks that valid chat name creates the chat successfully
    - Test checks that invalid chat name does not create the chat
- Tested by Brad

Module 3: ConversationPanel
- Equivalence classes: valid list from chatname room endpoints
    - Test checks that list contains valid chatroom names
- Tested by Brad and Nithin

Backend

All backend unit tests written by Jesse

Module 1: GET /user
- Equivalence classes: authenticated, not authenticated
    - If it is not authenticated, check it responds with error code 401
    - It it is authenticated, check it responds with users data and 200 code

Module 2: POST /user
- Equivalence classes: authenticated, not authenticated
    - If it is not authenticated, check it responds with error code 401
    - It it is authenticated, check it responds with 201 code

Module 3: POST /login
- Equivalence classes: valid credentials, invalid credentials
    - If credentials are valid, check it responds with token
    - It it is authenticated, check it responds with 401 code

Module 4: POST /chat
- Equivalence classes: authenticated, not authenticated
    - If it is not authenticated, check it responds with error code 401
    - It it is authenticated, check it responds with 201 code

Module 5: GET /chat/<id>

- Equivalence classes: authenticated and chat exists, authenticated and chat doesn’t exist, not authenticated
    - If it is not authenticated, check it responds with error code 401
    - It it is authenticated and chat exists, check it responds with 200 code