CUChat: Testing Document

Department of Computer Science and Engineering

The Chinese University of Hong Kong

May 4, 2024

Group E4

An Keyu, 1155174008

Wang Zixin, 1155173944

Huang Yanwen, 1155157280

Yuan Lin, 1155141399

Yao Shihurong, 1155157155

Table of Contents

1. Test Plan

1.1 Test Approach

We would adopt module testing and bottom-up integration to test our software as CUChat is a complex content-based social media software. Both black-box and white-box strategies will be implemented in the testing. White-box testing will be mainly used for isolated and simple modules and black-box testing will be used to validate the overall functionality specified in the requirement.

1.2 Testing Strategy

We will adopt both black-box and white-box strategies in our testing. For black-box testing, we will check the software for the expected behaviour outlined in our software requirement documentation. We will analyse the requirements to identify test scenarios, therefore designing test cases that cover aspects of the software's functionality. We consider both valid and invalid inputs, boundary conditions and exceptional cases. For white-box testing, we will guarantee statement and edge coverage and try to go through some critical condition cases due to the difficulty of fulfilling edge and condition coverage due to the large software size. By ensuring statement and edge coverage, we can identify any potential errors or unreachable codes in the program, so that we can be confident about the correctness and functionality of the software. By covering all possible edges, we can identify potential issues related to boundary conditions, loop iterations, and branching logic within the software. We will also try testing some critical and boundary condition cases that have a higher likelihood of introducing bugs for compound condition nodes.

1.3 Testing Environment

The testing environment should fulfil the necessary hardware and software requirements, including Node.js v21.7.1, MongoDB v1.42.5, React v17.0.2, Chrome v124.0.6367.119, and all other necessary softwares specified in package-lock.json.

2. Test Sets

2.1 Set-1 Authentication

2.1.1 Purpose

The purpose of this testing is to verify the functionality and reliability of the "Sign up and Sign in" feature on the website. This testing will focus on ensuring that users can successfully create new accounts, log in, and access the system's features without encountering errors or security.

2.1.2 Sign up

2.1.2.1 Inputs

Input format in each test case: {username, password}

Test case	Input (explanation)
1	{“”, “”} (create with empty username and password)
2	{“winnie”, “3100”} (create with normal username and password)

2.1.2.2 Expected Outputs & Pass/Fail Criteria

Test case	Expected Output	Pass/Fail Criteria

1	Error Message	Pass Criteria: error message Fail Criteria: do not show error message
2	Successfully create a user with username “winnie” and password “3100”	Pass Criteria: Successfully create a user with username “winnie” and password “3100” Fail Criteria: Fail to create a user with username “winnie” and password “3100”

2.1.3 Sign in

2.1.3.1 Inputs

Input format: {username, password}

Test case	Input (explanation)
1	{“”, “”} (create with empty username and password)
2	{“winnie”, “3100”} (create with normal username and password)

2.1.3.2 Expected Outputs & Pass/Fail Criteria

Test case	Expected Output	Pass/Fail Criteria
1	Error Message	Pass Criteria: error message Fail Criteria: do not show error message
2	Successfully login a user with username “winnie” and password “3100”	Pass Criteria: Successfully login with username “winnie” and password “3100” Fail Criteria: Fail to login with username “winnie” and password “3100”

2.2 Set-2 Admin

2.2.1 Purpose

To test whether the admin can create/update/delete a user and update the specified information to the user database.

2.2.2 Inputs

Input format in each test case: {**create**, userID, username, password, introduction}, {**update**, userID, newusername, newpassword, newintroduction}, {**delete**, userID}

Test Case	Input (explanation)
1	{create, “”, “”, “”, “”} (create with empty input)
2	{create, “0”, “test”, “test”, “test”} (create a new user who has a userID same as admin)
3	{create, “1”, “test”, “test”, “test”} (create a new user who has a userID same as an existing user)

4	{create, “3100”, “Michael”, “CSCI3100”, “I would like to offer you a coupon”} (create a user with valid inputs in all fields)
5	{update, “3100”, “”, “”, “”} (update with empty input)
6	{update, “57825”, “test”, “test”, “test”} (update an unexisting user)
7	{update, “3100”, “Winnie”, “IloveCS”, “IloveCS”} (update with valid inputs in all fields)
8	{delete, “”} (delete with empty input)
9	{delete, “75425782”} (delete an unexisting user)
10	{delete, “0”} (delete admin)
11	{delete, “3100”} (delete an existing user)

2.2.3 Expected Outputs & Pass/Fail Criteria

#	Expected Output	Pass/Fail Criteria
1	Display an error message and do not create a user.	Pass: Display an error message and no user is created. Fail: Create an user with empty fields as userID, username, password and introduction; or, lead to a system crash.
2	Display an error message and do not create a user.	Pass: Display an error message and no user is created. Fail: Create an user with empty fields as userID, username, password and introduction; or, lead to a system crash.
3	Display an error message and do not create a user.	Pass: Display an error message and no user is created. Fail: Create an user with empty fields as userID, username, password and introduction; or, lead to a system crash.
4	Create a user with userID “3100”, username “Michael”, password “CSCI3100” and introduction “I would like to offer you a coupon”.	Pass: Create a user with userID “3100”, username “Michael”, password “CSCI3100” and introduction “I would like to offer you a coupon” and store it in the database. Fail: Cannot create the user as specified above.
5	Display an error message and do not update a user.	Pass: Display an error message and no user is updated.

		Fail: Update the user with userID “3100” to make its username, password and introduction become empty; or, lead to a system crash.
6	Display an error message and do not update a user.	Pass: Display an error message and no user is updated. Fail: Update a random user or lead to a system crash.
7	Update the user with userID “3100”, update its username to “Winnie”, password to “IloveCS”, and introduction to “IloveCS”	Pass: Update the user with userID “3100”, update its username to “Winnie”, password to “IloveCS”, and introduction to “IloveCS” and store it in the database. Fail: Cannot update the user as specified above.
8	Display an error message and do not delete a user.	Pass: Display an error message and no user is deleted. Fail: Delete a random user or lead to a system crash.
9	Display an error message and do not delete a user.	Pass: Display an error message and no user is deleted. Fail: Delete a random user or lead to a system crash.
10	Display an error message and do not delete admin.	Pass: Display an error message and no user is deleted. Fail: Delete the admin or lead to a system crash.
11	Delete the user with userID “3100”	Pass: Delete the user with userID “3100” successfully. Fail: Cannot delete the specified user.

2.3 Set-3 Create Post

2.3.1 Purpose

To test the posting process and ensure that users can successfully create posts. In addition, we would like to test the following controls.

(1) Whether the posting process takes input with no text or media as invalid input.

(2) Whether the posting process takes input with text but without media as valid input.

(3) Whether the posting process takes input with media (image) but without text as valid input.

(4) Whether the posting process takes input with media (video) but without text as valid input.

(5) Whether the posting process takes input with text and with media (image) as valid input.

(6) Whether the posting process takes input with text and with media (video) as valid input.

2.3.2 Inputs

Assuming the initial database contains no user data.

Input format in each test case: {text, media}, where the data type of the text input is string and the data type of the media input is image or video.

Test case	Input (explanations)
1	{“”, “”} (test with empty text and empty media)
2	{“Hello world!”, “”} (test with non-empty text and empty media)
3	{“”, “image.jpg”} (test with empty text and non-empty media(image))
4	{“”, “video.mp4”} (test with empty text and non-empty media(video))
5	{“Hello world!”, “image.jpg”} (test with non-empty text and non-empty media(image))
6	{“Hello world!”, “video.mp4”} (test with non-empty text and non-empty media(video))

2.3.3 Expected Output & Pass/Fail Criteria

Test case	Expected Output:	Pass/Fail Criteria:
1	The web page will pop up a message box showing text and media input cannot be both empty.	Pass criteria: The message box with error message is popped up, and no new post is created. Fail criteria: A post with empty input and empty media is created.
2	A new post with the user's input is created.	Pass criteria: A new post with input text only is created and displayed to the users. Fail criteria: No new post is created or post with wrong content is created.
3	A new post with the user's input is created.	Pass criteria: A new post with input image only is created and displayed to the users. Fail criteria: No new post is created or post with wrong content is created.
4	A new post with the user's input is created.	Pass criteria: A new post with input video only is created and displayed to the users. Fail criteria: No new post is created or post with wrong content is created.
5	A new post with the user's input is created.	Pass criteria: A new post with input text and input image is created and displayed to the users.

		Fail criteria: No new post is created or post with wrong content is created.
6	A new post with the user's input is created.	Pass criteria: A new post with input text and input video is created and displayed to the users. Fail criteria: No new post is created or post with wrong content is created.

2.4 Set-4 Like/Dislike Post

2.4.1 Purpose

To test whether users can successfully like/dislike posts. In addition, we would like to test the following controls.

(1) Whether the user can like the post when the post has not been liked by himself / herself.

(2) Whether the user can dislike the post when the post has been liked by himself / herself.

2.4.2 Inputs

Assuming the original post is created and is valid. The input is a button click.

Test case	Situation Before Input	Input (explanations)
1	The post has not been liked by the user	Like/Dislike button click (the Button click should automatically identify whether it is a like or dislike request)
2	The post has been liked by the user	Like/Dislike button click (the Button click should automatically identify whether it is a like or dislike request)

2.4.3 Expected Output & Pass/Fail Criteria

Test case	Expected Output	Pass/Fail Criteria
1	The “Like/Dislike button” is turned on (Default: Off). The number of likes has increased by 1.	Pass Criteria: The “Like/Dislike button” is turned on. The number of likes has increased by 1. Fail Criteria: The “Like/Dislike button” is not turned on or the number of likes is not increased by 1.
2	The “Like/Dislike button” is turned off (Default: Off). The number of likes has decreased by 1.	Pass Criteria: The “Like/Dislike button” is turned off. The number of likes has decreased by 1. Fail Criteria: The “Like/Dislike button” is not turned off or the number of likes is not decreased by 1.

2.5 Set-5 Retweet Post

2.5.1 Purpose

To test whether users can successfully retweet posts. In addition, we would like to test the following controls by using black-box testing.

(1) Whether we can retweet posts that contain text only.
(2) Whether we can retweet posts that contain media(image) only.
(3) Whether we can retweet posts that contain media(video) only.
(4) Whether we can retweet posts that contain text and media(image).
(5) Whether we can retweet posts that contain text and media(video).

2.5.2 Inputs

Assuming the original post is created and is valid. The input is a button click.

Test Case	Situation Before Input	Input (explanations)
1	The original post contains text only.	Retweet button click
2	The original post contains media(image) only.	Retweet button click
3	The original post contains media(video) only.	Retweet button click
4	The original post contains text and media(image).	Retweet button click
5	The original post contains text and media(video).	Retweet button click

2.5.3 Expected Output & Pass/Fail Criteria

Test Case	Expected Output	Pass/Fail Criteria
1	A window for creating a post pops up where the text in the text box is the username of the user of the original post + the text in the original post and with no media attached.	Pass Criteria: The window for creating posts with right content attached has popped up. Fail Criteria: The window is not popped up or with wrong content attached.
2	A window for creating a post pops up where the text in the text box is the username of the user of the original post and with the media (image) in the original post attached.	Pass Criteria: The window for creating posts with right content attached has popped up. Fail Criteria: The window is not popped up or with wrong content attached.
3	A window for creating a post pops up where the text in the text box is the username of the user of the original post and with the media (video) in the original post attached.	Pass Criteria: The window for creating posts with right content attached has popped up. Fail Criteria: The window is not popped up or with wrong content attached.

4	A window for creating a post pops up where the text in the text box is the username of the user of the original post + the text in the original post and with the media (image) in the original post attached.	Pass Criteria: The window for creating posts with right content attached has popped up. Fail Criteria: The window is not popped up or with wrong content attached.
5	A window for creating a post pops up where the text in the text box is the username of the user of the original post + the text in the original post and with the media (video) in the original post attached.	Pass Criteria: The window for creating posts with right content attached has popped up. Fail Criteria: The window is not popped up or with wrong content attached.

2.6 Set-6 Comment

2.6.1 Purpose

The purpose of this testing is to verify the functionality and reliability of the "Comment" feature on the website. This testing will focus on ensuring that users can interact with content by posting comments accurately.

2.6.2 Inputs

Input format in each test case: {text}

Test Case	Input (explanations)
1	“” (empty comment)
2	“This is a good post” (normal comment)

2.6.3 Expected Outputs & Pass/Fail Criteria

Test Case	Expected Output	Pass/Fail Criteria
1	The “comment” button do not react	Pass Criteria: The “comment” button do not react
2	The corresponding post will have a new comment “this is good post”	Pass Criteria: The corresponding post will have a new comment “this is good post” Fail Criteria: The corresponding post do not have a new comment “this is good post”

2.7 Set-7 Chat

2.7.1 Purpose

The purpose of this testing is to verify the functionality and reliability of the "Chat" feature on the website. This testing will focus on ensuring that users can communicate with each other in real-time through the chat interface, and that the system handles chat messages appropriately in various scenarios.

2.7.2 Inputs

Input format in each test case: {text}

Test Case	Input (explanations)
1	“” (empty message)
2	“Hello, I am Winnie” (normal message)

2.7.3 Expected Outputs & Pass/Fail Criteria

Test Case	Expected Output	Pass/Fail Criteria
1	The “send” button do not react	Pass Criteria: The “send” button do not react
2	Send a message “Hello, I am Winnie” and it is displayed in the chat box	Pass Criteria: Send a message “Hello, I am Winnie” and it is displayed in the chat box Fail Criteria: Do not send a message “Hello, I am Winnie” and it is displayed in the chat box

2.8 Set-8 Search

2.8.1 Purpose

To test the user search and post search functionality and ensure that users can search for users by their userID or username and search for posts by various criteria. Thus, we will test the search for users and the search for posts separately. We would like to test the following controls by using black-box testing:

(1) Whether it will show all the user information when the user searches by empty string.
(2) Whether it will show the user information when the user searches by a random string like ‘1234’.
(3) Whether it will show the user information when the user searches by valid string (UserID).
(4) Whether it will show the user information when the user searches by valid string (Username).
(5) Whether the search for users is case-sensitive.
(6) Whether it will show all the posts when the user searches by empty string.
(7) Whether it will show the post when the user searches by a random string like ‘1234’.
(8) Whether it will show the post when the user searches by valid string.
(9) Whether the search for posts is case-sensitive.

2.8.2 Inputs

Input format in each test case: {text}

Test Case	Input (explanation)
1	{“”} (test with empty data)
2	{“1234”} (test with random string)
3	{“3100”} (test with valid UserID)
4	{“Winnie”} (test with valid Username)
5	{“wINnie”} (test with the case sensitivity)
6	{“”} (test with empty data)

7	{“1234”} (test with random string)
8	{“Hello world!”} (test with valid input)
9	{“hELLo World”} (test with the case sensitivity)

2.8.3 Expected Outputs & Pass/Fail Criteria

Test Case	Expected Output	Pass/Fail Criteria
1	It will show the not found message “Sorry, CUChat couldn't find the relevant user.” and a not found image.	Pass Criteria: The not found message and the not found image are displayed correctly. Fail Criteria: The not found message or the not found image is not displayed.
2	It will show the not found message “Sorry, CUChat couldn't find the relevant user.” and a not found image.	Pass Criteria: The not found message and the not found image are displayed correctly. Fail Criteria: The not found message or the not found image is not displayed.
3	It will show the user with the corresponding UserID.	Pass Criteria: The user with the corresponding UserID is displayed correctly. Fail Criteria: The user with the wrong UserID is displayed or there is no user displayed.
4	It will show the user with the corresponding Username.	Pass Criteria: The user with the corresponding Username is displayed correctly. Fail Criteria: The user with the wrong Username is displayed or there is no user displayed.
5	It will show the user with the corresponding Username.	Pass Criteria: The user with the corresponding Username is displayed correctly. Fail Criteria: The user with the wrong Username is displayed or there is no user displayed.
6	It will show the not found message “Sorry, CUChat couldn't find the relevant posts.” and a not found image.	Pass Criteria: The not found message and the not found image are displayed correctly. Fail Criteria: The not found message or the not found image is not displayed.
7	It will show the not found message “Sorry, CUChat couldn't find the relevant posts.” and a not found image.	Pass Criteria: The not found message and the not found image are displayed correctly. Fail Criteria: The not found message or the not found image is not displayed.

8	It will show the posts with the corresponding post content.	Pass Criteria: The posts with the corresponding post content are displayed correctly. Fail Criteria: The posts with the wrong post content are displayed or there is no post displayed.
9	It will show the posts with the corresponding post content.	Pass Criteria: The posts with the corresponding post content are displayed correctly. Fail Criteria: The posts with the wrong post content are displayed or there is no post displayed.

2.9 Set-9 Search

2.9.1 Purpose

To test the follow/unfollow functionality on the website and ensure its reliability and efficiency. This testing will focus on verifying the behaviour of the follow/unfollow feature under various conditions.

Follow Functionality: In this part, we will test the following controls using black-box testing:
(1) Whether the user can successfully follow another user.
(2) Whether the system correctly adds the user to the followers list of the target profile upon successful follow.
(3) Whether the "Follow" button changes to "Unfollow" after successful follow.

Unfollow Functionality: In this part, we will test the following controls using black-box testing:
(1) Whether the user can successfully unfollow a user they are already following.
(2) Whether the system correctly removes the user from the followers list of the target profile upon successful unfollow.
(3) Whether the "Unfollow" button changes back to "Follow" after successful unfollow.

2.9.2 Inputs

The input is a button click.

Test Case	Input (explanation)
1	If the “Follow/Unfollow button” displays “Follow”, then click the button to follow that user.
2	If the “Follow/Unfollow button” displays “Unfollow”, then click the button to unfollow that user.

2.9.3 Expected Outputs & Pass/Fail Criteria

Test Case	Expected Output	Pass/Fail Criteria
1	The “Follow/Unfollow button” turn to “Unfollow”	Pass Criteria: The “Follow/Unfollow button” turn to “Unfollow”

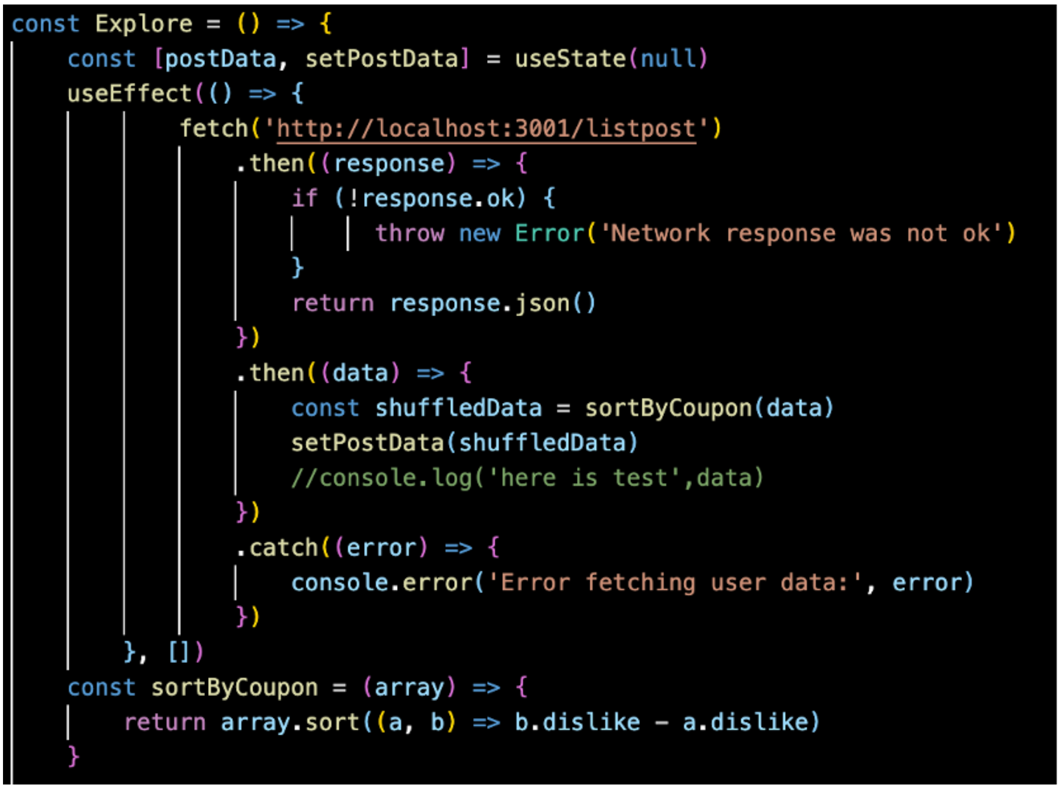
		Fail Criteria: The “Like/Dislike button” do not turn to “Unfollow”
2	The “Follow/Unfollow button” turn to “Follow”	Pass Criteria: The “Follow/Unfollow button” turn to “Follow” Fail Criteria: The “Like/Dislike button” do not turn to “Follow”

2.10 Set-10 Explore

2.10.1 Purpose

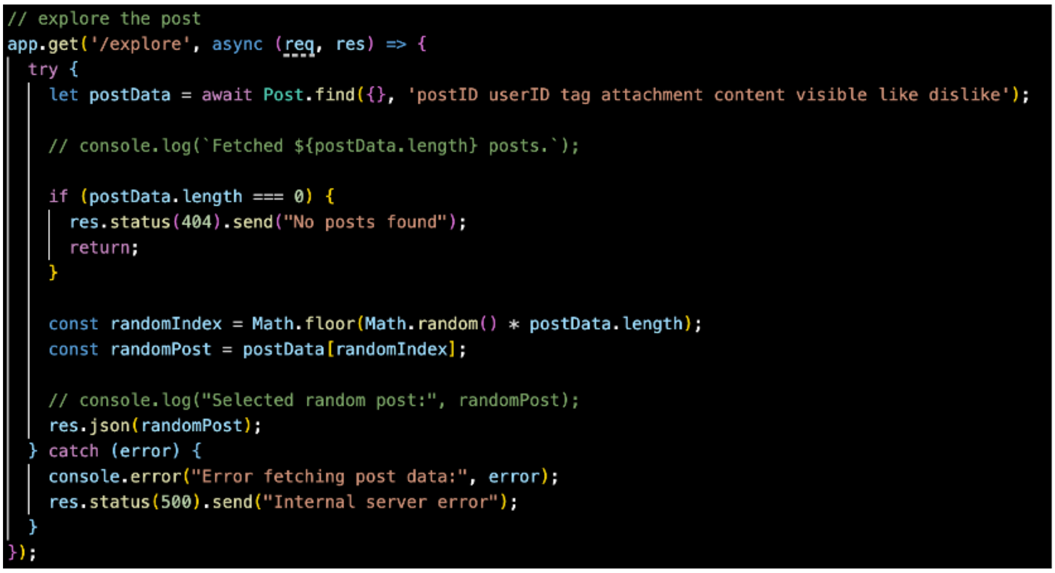
To test whether the posts are displayed with the explore function correctly.

2.10.2 Inputs



```
const Explore = () => {
    const [postData, setPostData] = useState(null)
    useEffect(() => {
            fetch('http://localhost:3001/listpost')
                .then((response) => {
                    if (!response.ok) {
                          throw new Error('Network response was not ok')
                    }
                    return response.json()
                })
                .then((data) => {
                    const shuffledData = sortByCoupon(data)
                    setPostData(shuffledData)
                    //console.log('here is test',data)
                })
                .catch((error) => {
                    console.error('Error fetching user data:', error)
                })
        }, [])
    const sortByCoupon = (array) => {
        return array.sort((a, b) => b.dislike - a.dislike)
    }
```

Figure 1: Code of Function of getting the aimed posts from backend (Frontend).



```
// explore the post
app.get('/explore', async (req, res) => {
  try {
    let postData = await Post.find({}, 'postID userID tag attachment content visible like dislike');

    // console.log(`Fetched ${postData.length} posts.`);

    if (postData.length === 0) {
      res.status(404).send("No posts found");
      return;
    }

    const randomIndex = Math.floor(Math.random() * postData.length);
    const randomPost = postData[randomIndex];

    // console.log("Selected random post:", randomPost);
    res.json(randomPost);
  } catch (error) {
    console.error("Error fetching post data:", error);
    res.status(500).send("Internal server error");
  }
});
```

Test Case	Input (explanation)
1	Click Explore button to browse the recommended posts

2.10.3 Inputs

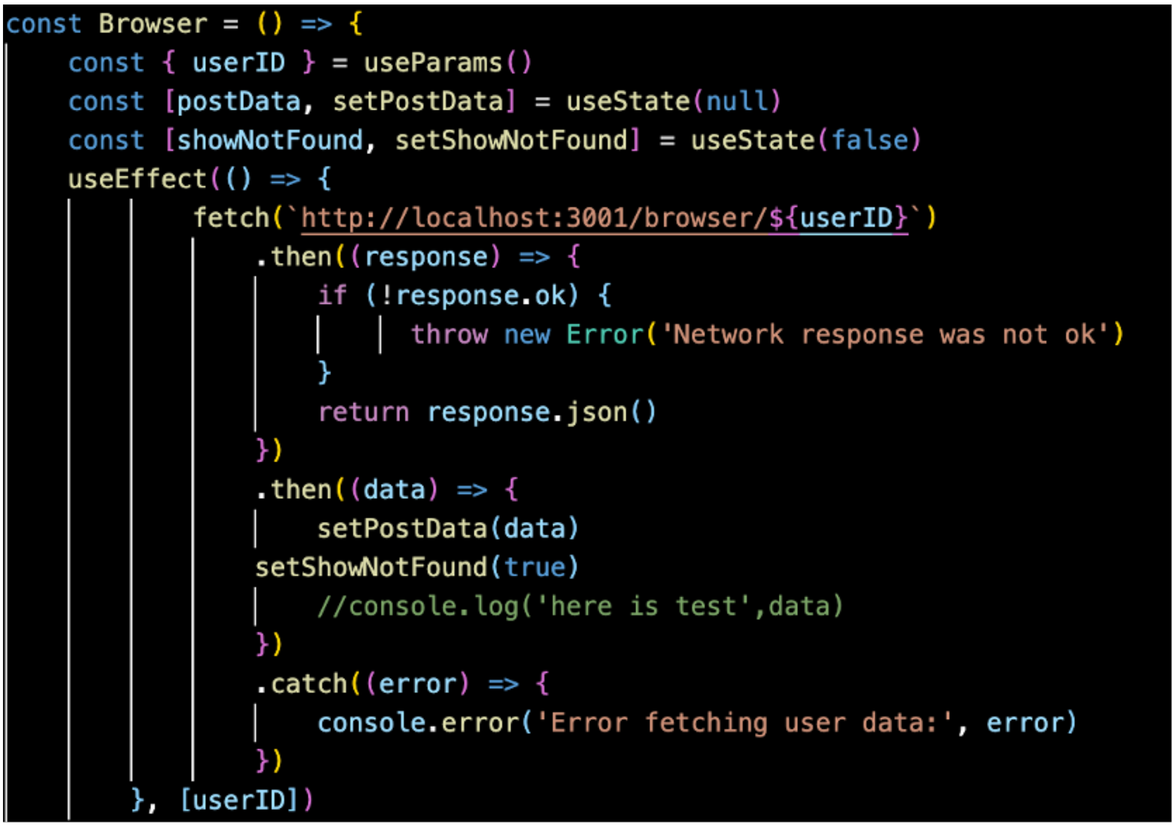
Test Case	Expected output:	Pass/Fail Criteria
1	It will show the recommended posts.	Pass Criteria: The recommended posts will be displayed right. Fail Criteria: The recommended posts are not displayed right.

2.11 Set-11 Explore

2.11.1 Purpose

To test whether the posts of the user's following users have posted are displayed right.

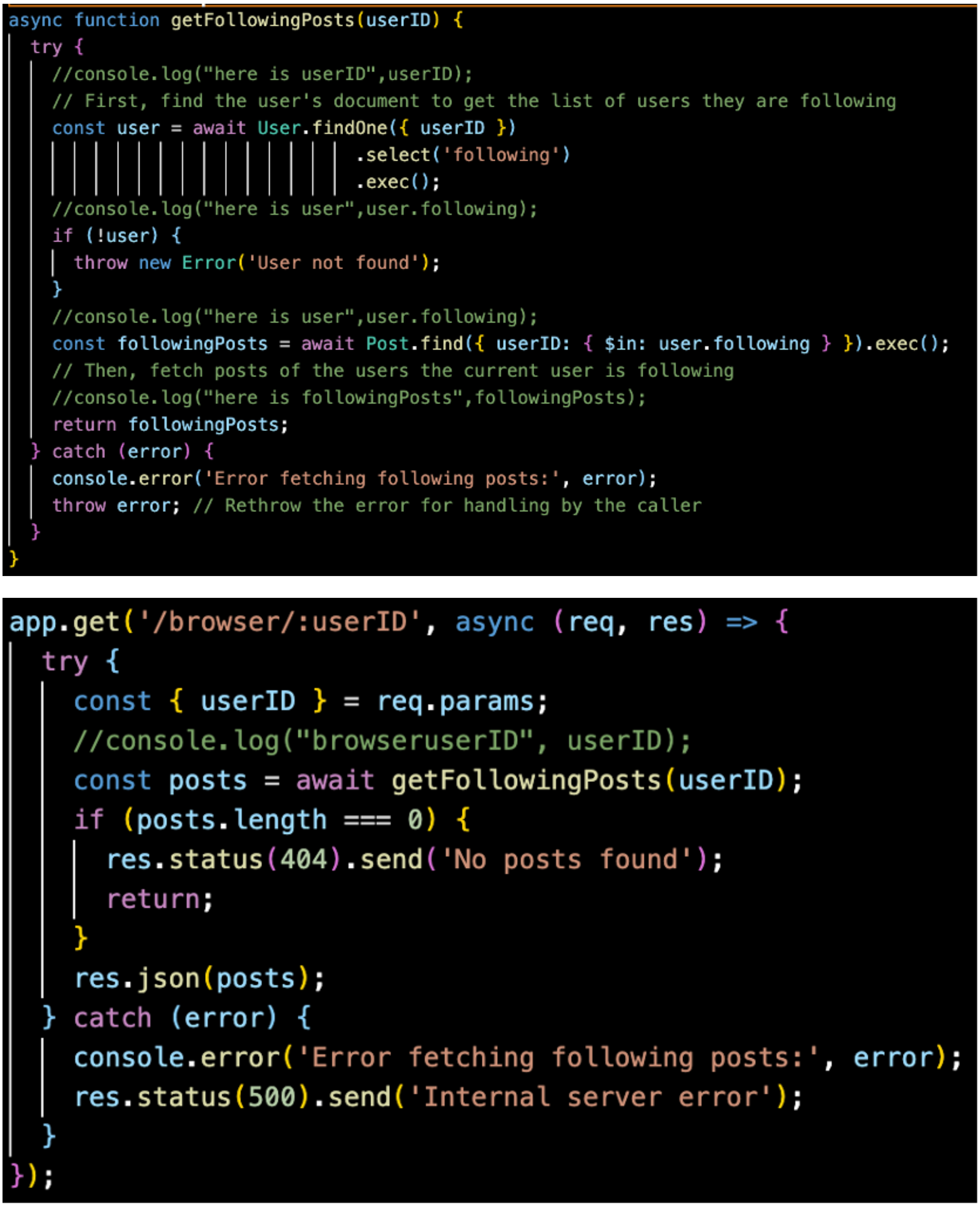
2.11.2 Input



```
const Browser = () => {
    const { userID } = useParams()
    const [postData, setPostData] = useState(null)
    const [showNotFound, setShowNotFound] = useState(false)
    useEffect(() => {
            fetch(`http://localhost:3001/browser/${userID}`)
                .then((response) => {
                    if (!response.ok) {
                        throw new Error('Network response was not ok')
                    }
                    return response.json()
                })
                .then((data) => {
                    setPostData(data)
                setShowNotFound(true)
                    //console.log('here is test',data)
                })
                .catch((error) => {
                    console.error('Error fetching user data:', error)
                })
        }, [userID])
```

Figure 2: Code of Function of getting the aimed posts from backend (Frontend).

Test Case	Input (explanation)
1	Click Browser button to browse the following users' posts



```
async function getFollowingPosts(userID) {
  try {
    //console.log("here is userID",userID);
    // First, find the user's document to get the list of users they are following
    const user = await User.findOne({ userID })
                           .select('following')
                           .exec();
    //console.log("here is user",user.following);
    if (!user) {
      throw new Error('User not found');
    }
    //console.log("here is user",user.following);
    const followingPosts = await Post.find({ userID: { $in: user.following } }).exec();
    // Then, fetch posts of the users the current user is following
    //console.log("here is followingPosts",followingPosts);
    return followingPosts;
  } catch (error) {
    console.error('Error fetching following posts:', error);
    throw error; // Rethrow the error for handling by the caller
  }
}
```

```
app.get('/browser/:userID', async (req, res) => {
  try {
    const { userID } = req.params;
    //console.log("browseruserID", userID);
    const posts = await getFollowingPosts(userID);
    if (posts.length === 0) {
      res.status(404).send('No posts found');
      return;
    }
    res.json(posts);
  } catch (error) {
    console.error('Error fetching following posts:', error);
    res.status(500).send('Internal server error');
  }
});
```

Figure 3: Code of Function of selecting the aimed posts (Backend).

2.11.3 Expected Outputs & Pass/Fail Criteria

Test Case	Expected Output	Pass/Fail Criteria
1	It will show all the posts that the following users have posted.	Pass Criteria: The posts that the following users have posted will be displayed right or show the not found message “Sorry, CUChat couldn't find the relevant posts.” Fail Criteria: The posts that the following users have posted are not displayed right or there is no the not found message.

2.12 Set-12 Edit Profile

2.12.1 Purpose

The purpose of this testing is to verify the functionality and reliability of the "Edit Profile" feature on the website. This testing will focus on ensuring that users can accurately update their profile information and that the system behaves as expected in response to various inputs.

2.12.2 Inputs

Test Case	Input (explanation)
1	{Password: “3100fun”} (test changing password)
2	{User Name: “jessie”} (test changing username)
3	{Introduction: “I love this course”} (test changing introduction)
4	{avatar: “avatar.jpg”} (test changing avatar image)
5	{background: “background.jpg”} (test changing background image)

2.12.3 Expected Outputs & Pass/Fail Criteria

Test case	Expected Output	Pass/Fail Criteria
1	the password of the current user has been successfully changed to new password “3100fun”	Pass Criteria: the password of the current user has been successfully changed to new password Fail Criteria: the password of the current user hasn’t been changed to new password
2	the username of the current user has been successfully changed to new username “jessie”	Pass Criteria: the username of the current user has been successfully changed to new username Fail Criteria: the username of the current user hasn’t been changed to new username
3	the introduction of the current user has been successfully changed to new introduction “I love this course”	Pass Criteria: the introduction of the current user has been successfully changed to new introduction Fail Criteria: the introduction of the current user hasn’t been changed to new username
4	the avatar of the current user has been successfully changed to the new image “avatar.jpg”	Pass Criteria: the avatar of the current user has been successfully changed to new image Fail Criteria: the avatar of the current user hasn’t been changed to new image
5	the background of the current user has been successfully changed to the new image “background.jpg”	Pass Criteria: the background of the current user has been successfully changed to new image Fail Criteria: the background of the current user hasn’t been changed to new image