GPM Jammers
Milestone 7
April 30, 2024
Cpt_S 322

The following is our submission for Milestone 7. This document outlines test cases that would ensure our application behaves as defined in previous milestones.Each component outlined in milestone 6 has a related test case table and a short description.

Frontend

Validate Input:
This test case ensures the cleaning of strings entered in the password and username input fields are valid (does not contain spaces and is not the empty string).

Test Type	Unit test - `validate_input_fields()`
Test Target	UI Communicator component
Test Input	`username_input.value = 'hello world'`
Expected Output	`create_account_button = false`

Search User:
This test case ensures that an existing user is included in the database.

Test Type	Unit test - `set_all_users()`
Test Target	User Handler component
Test Input	`username_input.value = 'cesarus1234'`
Expected Output	`true`

Categorize Game Jam:
This test ensures that existing game jams are entered into the correct category of 'past', 'ongoing', or 'future'.

Test Type	Unit test - `draw_all_game_jams()`
Test Target	Central Logic Engine component
Test Input	`jam.date = 'July 20, 2069 at 20:17'`
Expected Output	`insertAfter(past_div, jam_div)`

Categorize Game Jam:
This test ensures that all users tied to a game jam are drawn to the screen.

Test Type	Unit test - `fill_users()`
Test Target	Game Jam Logic component
Test Input	`game_jam.participats = ['user1', 'user2']`
Expected Output	`'user1'` and `'user2'` are displayed to the screen

Login:
This test ensures that an existing user can only be logged in with correct credentials.

Test Type	Unit test - `async function login()`
Test Target	Database Communicator component
Test Input	`username_input.value = 'cesarus1234'` `Password_input.value = 'hello_world'`
Expected Output	`"Not found" (a response with a 404 error)`

Application Programming Interface (Server API)

Sign Up - Successful:
Ensure that a new user with a unique username can be created and returned the public attributes of said user.

Test Type	Unit test - `signUpSuccess()`
Test Target	Application Programming Interface

Test Input	`url = baseURL + "/signUp"`
	`query = {`
	`  {`
	`  username = "cesarus1234"`
	`  name = "Cesarus"`
	`  password_encoded =`
	`  bcrypt("abc", process.BCRYPT_SECRET)`
	`  },`
	`  { method: 'POST' }`
	`}`

Expected Output	`{`
	`  status: 201,`
	`  body:`
	`  {`
	`  "username": "cesarus12345",`
	`  "name": "Cesarus",`
	`  "isAdmin": false,`
	`  "bio": ""`
	`  }`
	`}`

Sign Up - Username Taken:
Ensure that a request to create a user with an already taken username will be rejected.

Test Type	Unit test - `signUpUsernameTaken()`
Test Target	Application Programming Interface

Test Input	`url = baseURL + "/signUp"`
	`query = {`
	`  {`
	`  username = "cesarus12345"`
	`  name = "Cesarus"`
	`  password_encoded =`
	`  bcrypt("abc", process.BCRYPT_SECRET)`
	`  },`
	`  { method: 'POST' }`
	`}`
	
	`url = baseURL + "/signUp"`
	`query = {`
	`  {`
	`  username = "cesarus12345"`
	`  name = "Cesarus"`
	`  password_encoded =`
	`  bcrypt("abc", process.BCRYPT_SECRET)`
	`  },`
	`  { method: 'POST' }`
	`}`

Expected Output	`{`
	`  status: 201,`
	`  body:`
	`  {`
	`  "username": "cesarus12345",`
	`  "name": "Cesarus",`
	`  "isAdmin": false,`
	`  "bio": ""`
	`  }`
	`}`
	`{`
	`  status: **400,**`
	`  body: "**Username taken**"`
	`}`

Sign In - Successful:
Check if users are found and authenticated correctly.

Test Type	Unit test - `signIn_success()`
Test Target	Application Programming Interface

Test Input	`url = baseURL + "/signUp"`
	`query = {`
	`  {`
	`  username = "cesarus1234"`
	`  name = "Cesarus"`
	`  password_encoded =`
	`  bcrypt("abc", process.BCRYPT_SECRET)`
	`  },`
	`  { method: 'POST' }`
	`}`
	
	`url = baseURL + "/signIn";`
	`query = {`
	`  {`
	`  username = "cesarus1234"`
	`  password_encoded =`
	`  bcrypt("abc", process.BCRYPT_SECRET)`
	`  },`
	`  { method: 'POST' }`
	`};`

Expected Output	`{`
	`  status: 201,`
	`  body:`
	`  {`
	`  "username": "cesarus1234",`
	`  "name": "Cesarus",`
	`  "bio": "",`
	`  "isAdmin": false`
	`  }`
	`}`
	
	`{`
	`  status: 200,`
	`  body:`
	`  {`
	`  "username": "cesarus1234",`
	`  "name": "Cesarus",`
	`  "bio": "",`
	`  "isAdmin": false`
	`  }`
	`}`

Sign In - Unsuccessful:
Check if users are found and authenticated correctly.

Test Type	Unit test - `signIn_unsuccessful()`
Test Target	Application Programming Interface

Test Input	`url = baseURL + "/signUp"`
	`query = {`
	`  {`
	`  username = "cesarus1234"`
	`  name = "Cesarus"`
	`  password_encoded =`
	`  bcrypt("abc", process.BCRYPT_SECRET)`
	`  },`
	`  { method: 'POST' }`
	`}`
	
	`url = baseURL + "/signIn";`
	`query = {`
	`  {`
	`  username = "cesarus1234"`
	`  password_encoded =`
	`  bcrypt("abcd", process.BCRYPT_SECRET)`
	`  },`
	`  { method: 'POST' }`
	`};`

Expected Output	`{`
	`  status: 201,`
	`  body:`
	`  {`
	`  "username": "cesarus1234",`
	`  "name": "Cesarus",`
	`  "bio": "",`
	`  "isAdmin": false`
	`  }`
	`}`
	
	`{`
	`  status: 404,`
	`  body: "Not Found"`
	`}`

Get User:
Check if a user can be found and returned appropriately, without exposing passwords.

Test Type	Unit test - `getUser()`
Test Target	Application Programming Interface

Test Input	`url = baseURL + "/getUser"`
	`query = {`
	`  {`
	`  username = "cesarus1234"`
	`  },`
	`  { method: 'GET' }`
	`}`

Expected Output	`{`
	`  status: 200,`
	`  body: {`
	`  "username": "cesarus1234",`
	`  "name": "Cesarus",`
	`  "isAdmin": false,`
	`  "bio": ""`
	`  }`
	`}`

Get Users:
Check if users are found and returned appropriately, without exposing passwords.

Test Type	Unit test - `getUsers()`
Test Target	Application Programming Interface

Test Input	`url = baseURL + "/signUp"`
	`query = {`
	`  {`
	`  username = "cesarus12345"`
	`  name = "Cesarus"`
	`  password_encoded =`
	`  bcrypt("abc", process.BCRYPT_SECRET)`
	`  },`
	`  { method: 'GET' }`
	`}`
	
	`url = baseURL + "/getUsers"`
	`query = {`
	`  { method: 'GET' }`
	`}`

Expected Output	`{`
	`  status: 200,`
	`  body: [{`
	`  "username": "cesarus12345",`
	`  "name": "Cesarus",`
	`  "isAdmin": false,`
	`  "bio": ""`
	`  }]`
	`}`

Update User:
Check if users are found and then updated correctly, if the client is authorized to do so.

Test Type	Unit test - `updateUser()`
Test Target	Application Programming Interface

Test Input	`url = baseURL + "/signIn";`
	`query = {`
	`  {`
	`  username = "cesarus1234"`
	`  password_encoded =`
	`  bcrypt("abc", process.BCRYPT_SECRET)`
	`  },`
	`  { method: 'POST' }`
	`};`
	
	`url = baseURL + "/updateUser";`
	`query = {`
	`  {`
	`  username = "cesarus1234",`
	`  name = "Cesaro",`
	`  bio = "Lorem ipsum"`
	`  },`
	`  { method: 'PUT' }`
	`};`

Expected Output	`{`
	`  status: 200,`
	`  body:`
	`  {`
	`  "username": "cesarus1234",`
	`  "name": "Cesarus",`
	`  "bio": "",`
	`  "isAdmin": false`
	`  }`
	`}`
	
	`{`
	`  status: 200,`
	`  body:`
	`  {`
	`  "username": "cesarus1234",`
	`  "name" = "Cesaro",`
	`  "bio" = "Lorem ipsum",`
	`  "isAdmin": false`
	`  }`
	`}`

Delete User:
Check if users are found deleted correctly.

Test Type	Unit test - `deleteUser()`
Test Target	Application Programming Interface

Test Input	`url = baseURL + "/deleteUser";`
	`query = {`
	`  {`
	`  username = "cesarus12345"`
	`  },`
	`  { method: 'DELETE' }`
	`};`
	
	`url = baseURL + "/getUser";`
	`query = {`
	`  {`
	`  username = "cesarus12345"`
	`  },`
	`  { method: 'GET' }`
	`};`

Expected Output	`{`
	`  status: 200`
	`}`
	
	`{`
	`  status: 404,`
	`  body: "Not Found"`
	`}`

Add Participant:
Check if users are found and returned appropriately, without exposing passwords.

Test Type	Unit test - `addParticipant()`
Test Target	Application Programming Interface

Test Input	`url = baseURL + "/signUp"`
	`query = {`
	`  {`
	`  username = "cesarus1234"`
	`  name = "Cesarus"`
	`  password_encoded =`
	`  bcrypt("abc", process.BCRYPT_SECRET)`
	`  },`
	`  { method: 'POST' }`
	`}`
	
	`url = baseURL + "/postJam";`
	`query = {`
	`  {`
	`  title = "2024 Winter Game Jam"`
	`  description = "Lorem ipsum"`
	`  date = "December 20th, 2024"`
	`  },`
	`  { method: 'POST' }`
	`};`
	
	`url = baseURL + "/addOrRemoveParticipant";`
	`query = {`
	`  {`
	`  title = "2024 Winter Game Jam"`
	`  },`
	`  { method: 'PUT' }`
	`};`
	`credentials = sessionCookie;`

Expected Output	`{`
	`  status: 200,`
	`  body: {`
	`  "title": "2024 Winter Game Jam",`
	`  "description": "Lorem ipsum",`
	`  "date": "December 20th, 2024",`
	`  **"participants": ["cesarus1234"],**`
	`  "posts": []`
	`  }`
	`}`