HAND OVER 3

TESTING

ALEN JOHN VARGHESE
POOJA PRAKASH
SHAFANA NIZAM

Test Cases

Black-Box Test Cases

These test cases are defined based on the available documentation and execution of the program. The code was not inspected.

#	Test case (very brief description)	Preconditions (any required setup)	Test steps (steps executed during testing)	Expectation	Observation (“pass” or failure description)
1	Verify that the signup form is displayed correctly.	✓ The application is running.	Open the signup page. Enter a name, username , password and email. Click the "Register" button.	The user should be redirected to the home page.	Pass
2	Ensure that incorrect signup credentials display an appropriate error message.	✓ The application is running.	Open the signup page. Enter a name, username , password and email. Click the "Register" button.	An error message should be displayed.	Fail
3	Verify that an error is returned when trying to register with an existing username.	✓ The application is running. ✓ A user with the username "existingUser" already exists.	Open the signup page. Enter a name, username (taken.), password and email. Click the "Register" button.	An error message should be displayed.	Fail
4	Verify that an error is returned when trying to register with empty username or password fields.	✓ The application is running.	Open the signup page. Click the "Register" button	An error message "Username cannot be empty" should be displayed for an empty username. An error message "Password cannot be	Fail , but cannot signup

				empty" should be displayed for an empty password.	
5	Verify that the login form is displayed correctly.	✓ The application is running.	Open the login page.	The login form should be displayed with all required fields.	Pass
6	Verify that an error message is shown for empty username or password fields.	✓ The application is running.	Submit the form with an empty username. Submit the form with an empty password.	An error message "Username cannot be empty" should be displayed for an empty username. An error message "Password cannot be empty" should be displayed for an empty password.	Fail, can login without username and password
7	Verify that the user is redirected to the correct page after a successful login.	✓ The application is running. ✓ Valid user details are available.	Enter a valid username and password. Submit the form.	The user should be redirected to the "HomePage".	Pass
8	Verify that the chat interface is displayed correctly.	✓ The application is running.	Open the chat page.	The chat interface should be displayed with all required elements.	Pass
9	Verify that a message can be sent successfully.	✓ The app is running. ✓ The chat interface is displayed.	Enter a message . Click the "Send" button	The message should be displayed in the chat window.	Pass

10	Verify that sending an empty message displays an appropriate error.	✓ The app is running. ✓ The chat interface is displayed.	✓ Enter a message . ✓ Click the "Send" button.	An error message "Message cannot be empty" should be displayed.	Fail
11	Verify that the search results window is displayed correctly.	✓ The app is running. ✓ The chat interface is displayed.	Open the search window.	The search results window should be displayed with all required elements.	Pass
12	Verify that resizing the application window adjusts the layout correctly.	✓ The application is running. ✓ The main application window is displayed.	Resize the application window to different dimensions.	The layout should adjust correctly.	Fail
13	Verify that the edit profile window is displayed correctly.	✓ The application is running. ✓ The main application window is displayed.	Click the edit profile option	Display change username and change password fields and should able change them.	Pass
14	Verify that the Group message works correctly.	✓ The application is running.	Click the group message option.	Should able to send message to different users at a time.	Pass
15	Verify that the logout works correctly	✓ The application is running. ✓ The main application window is displayed	Click the logout button.	Should logout from that user.	Pass

White-Box Test Cases

These additional test cases were defined during inspection of the code.

#	Test case (very brief description)	Preconditions (any required setup)	Test steps (steps executed during testing)	Expectation	Observation (“pass” or failure description)
1	Verify UI components initialization.	AppGUI instance created.	1.Create AppGUI instance 2. Retrieve loginbutton and signupbutton via reflection.	loginbutton and signupbutton should not be null.	Pass
2	Test login button action.	AppGUI instance created.	1. Click loginbutton.	LoginFrontend should be created and visible.	Pass
3	Test signup button action.	AppGUI instance created.	1. Click signupbutton.	SignupFrontend should be created and visible.	Pass
4	Verify UI components initialization.	ChatFrontend instance created.	1. Create ChatFrontend instance. 2. Retrieve messagebox and chatscreen via reflection.	messagebox and chatscreen should not be null.	Pass
5	Test sending a message.	ChatFrontend instance created.	1. Set text in messagebox. 2. Click sendButton.	sendMessage method should be called with "Hello, world!". Message box should be cleared.	Pass
6	Test sending an emoji.	ChatFrontend instance created.	1. Click emojiButton. 2. Manually invoke sendMessage method	sendMessage method should be called with emoji. messagebox should be cleared.	Fail
7	Test refreshing messages	ChatFrontend instance created.	1. Set backend messages 2. Manually invoke refreshMessages method	chatscreen should display all messages from backend correctly.	Fail
8	Verify message storing.	Messages table created, empty.	1. Store a message. 2. Query the messages table for the stored message	The message should be stored correctly in the database.	Pass

9	Verify fetching messages.	Messages table created, empty.	1. Store multiple messages between two users. 2. Fetch messages between the two users.	The messages should be fetched correctly in the correct order.	Pass
10	Verify searching messages.	Messages table created, empty.	1. Store multiple messages. 2. Search messages containing a specific word.	The messages containing the specific word should be fetched correctly.	Pass
11	Verify UI components initialization.	LoginFrontend instance is created.	1. Create an instance of LoginFrontend. 2. Check for UI components using reflection.	All UI components (usernameField, passwordField, login button) should be initialized and not null.	Pass
12	Valid login credentials.	LoginFrontend instance is created.	1. Set valid credentials. 2. Simulate login button click	Login should be successful, and the LoginFrontend window should be disposed.	Pass
13	Invalid login credentials.	LoginFrontend instance is created.	1. Set invalid credentials ("user", "wrongpassword"). 2. Simulate login button click	Login should fail, and the LoginFrontend window should remain visible.	Pass
14	Verify successful authentication with correct credentials.	The USERS1 table exists and contains user data.	1. Set up the database and insert test user data. 2. Call LoginBackend.authentication("user1", "password1").	The method should return true.	Fail
15	Verify failed authentication due to incorrect password.	The USERS1 table exists and contains user data.	1. Set up the database and insert test user data. 2. Call LoginBackend.authentication("user1", "wrongPassword")	The method should return false.	Fail
16	Verify clearing of USERS1 table before each test.	The USERS1 table exists and contains user data.	1. Set up the database and insert test user data 2. Verify that the table is cleared before each test by checking the contents of the table at the start of each test.	The USERS1 table should be empty at the start of each test.	Pass
17	Verify SQL exception handling.	The USERS1 table does not exist or the database connection is invalid.	1. Simulate SQL exception by using an incorrect database URL 2. Call LoginBackend.authentication("user1", "password1") inside a try-catch block	The method should return false, and the exception should be an instance of SQLException.	Fail

18	Verify window properties.	The SearchResultsWindow instance is created.	1. Create an instance of SearchResultsWindow with test search results. 2. Verify the window title, dimensions, and default close operation	The window title should be "Search Results".	Pass
19	Verify search results display in the UI	The SearchResultsWindow instance is created.	1. Create an instance of SearchResultsWindow with test search results. 2. Check if the content pane contains a JScrollPane with a JPanel. 3. Verify the layout and labels.	The JPanel inside the JScrollPane should have a vertical BoxLayout and the correct number of JLabel components with expected text.	Pass
20	Verify initial visibility of the window.	The SearchResultsWindow instance is created.	1. Create an instance of SearchResultsWindow with test search results. 2. Check if the window is visible.	The window should be visible	Pass
21	Verify UI components initialization.	The SignupFrontend instance is created.	1. Create an instance of SignupFrontend. 2. Access private fields using reflection. 3. Verify that nameField, usernameField, passwordField, and emailField are not null.	All fields should be initialized and not null.	Pass
22	Verify actionPerformed method with valid input.	The SignupFrontend instance is created and all input fields are set.	1. Set values for nameField, usernameField, passwordField, and emailField using reflection. 2. Simulate a button click on the signup button.	Verify that the actionPerformed method processes the input correctly.	Pass
23	Verify the signup backend method is called with correct arguments.	The SignupFrontend instance is created and all input fields are set.	1. Set values for nameField, usernameField, passwordField, and emailField using reflection. 2. Simulate a button click on the signup button	SignupBackend.Signup should be called with the values set in the fields.	Pass
24	Verify button initialization.	The StadiumButtongreen instance is created.	1. Create an instance of StadiumButtongreen. 2. Check initial text. 3. Check initial background and foreground colors.	Buttons should be active	Fail
25	Verify setForegroundColor method, paintComponent	The StadiumButtongreen instance is created.	Create an instance of StadiumButtongreen. 2. Set the size of the button.	The color should be updated to the new color	Pass

	method, paintBorder method.				
26	Verify setBackgroundColor method.	The StadiumButtongreen instance is created.	1. Create an instance of StadiumButtongreen. 2. Set a new background color using setBackgroundColor method. 3. Verify the background color is updated correctly.	The background color should be updated to the new color.	Fail
27	Verify password field initialization.	The StadiumPasswordField instance is created.	Create an instance of StadiumPasswordField.	Field should be displayed.	Pass
28	Verify password input.	The StadiumPasswordField instance is created.	1. Create an instance of StadiumPasswordField. 2. Set a password using setText method. 3. Verify the password is correctly set using getPassword method.	The password should be correctly set and retrievable.	Pass
29	Verify text field initialization.	The StadiumTextField instance is created.	Create an instance of StadiumTextField with 20 columns.	The column count should be 20.	Pass
30	Verify text input.	The StadiumTextField instance is created.	1. Create an instance of StadiumTextField. 2. Set some text using setText method. 3. Verify the text is correctly set using getText method.	The text should be correctly set and retrievable.	Pass