Testing and Data Entry Document

Test Case

Versions:

Version ID	Date	Changes
1.0	18/09/2021	Add manual tests and automatic tests for sprint 1
2.0	24/10/2021	Add test cases and data entry for sprint 2

1.1 UC001: Register function for supervisor

1.1.1 TC001: Valid email, password, username and code

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the register function works correctly.	
Setup: <List the pre-conditions to carry through this test case > · The email is valid and does not exist in database. · The password is valid with length between 8 and 16. · The username is valid with length between 2 and 16. · The invitation code is correct.	
Pre-Conditions: <List the conditions after the test execution > · The account information is added to the database and the user could log in with this account. · The front end receives status code 200 from the back end. · The front end displays the message “Success!!Please Sign in!” in the screen.	
Notes: [1] Register function. Verify that the register function works correctly. [1.1] Email, username, password and invitation code are mandatory fields. [1.2] Email, username and password have no limit on especial characters. [1.3] Username and password validation works in the front end while email and code validation works in the back end. [1.4] Log in with this account successfully or check the account information in database.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.1.2 TC002: Invalid email, valid password, username and code

Test Type:	**Execution Type:**
Functional	Manual
Objective:	
Verify if the register function works correctly.	
Setup:	
<List the pre-conditions to carry through this test case > · The email is invalid and already exist in database. · The password is valid with length between 8 and 16. · The username is valid with length between 2 and 16. · The invitation code is correct.	
Pre-Conditions:	
<List the conditions after the test execution > · The user information is not added to the database and the user could not log in with this account. · The front end receives status code 500 and message “email exist” from the back end. · The front end displays the message “email exist” in the screen.	
Notes:	
[1] Register function. Verify that the register function works correctly. [1.1] Email, username, password and invitation code are mandatory fields. [1.2] Email, username and password have no limit on especial characters. [1.3] Username and password validation works in the front end while email and code validation works in the back end. [1.4] The email here should exist in database. Other parameters are valid. [1.5] The account information is not added to database.	
Time constraint:	
Minimum: 10 min Maximum: 15 min	

1.1.3 TC003: Valid email, username, code and invalid password

Test Type:	**Execution Type:**
Functional	Manual
Objective:	
Verify if the register function works correctly.	
Setup:	
<List the pre-conditions to carry through this test case > · The email is valid and does not exist in database. · The password is invalid with length<8 or length >16. · The username is valid with length between 2 and 16. · The invitation code is correct.	

Pre-Conditions: <List the conditions after the test execution > · The user information is not added to the database and the user could not log in with this account. · The front end declines this operation because the password validation works in the front end. · The front end displays the message “Password length should between 8 to 16” in the screen.
Notes: [1] Register function. Verify that the register function works correctly. [1.1] Email, username, password and invitation code are mandatory fields. [1.2] Email, username and password have no limit on especial characters. [1.3] Username and password validation works in the front end while email and code validation works in the back end. [1.4] The password is invalid because its length is less than 8 or more than 16. Other parameters are valid. [1.5] The account information is not added to database.
Time constraint: Minimum: 10 min Maximum: 15 min

1.1.4 TC004: Valid email, password, code and invalid username

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the register function works correctly.	
Setup: <List the pre-conditions to carry through this test case > · The email is valid and does not exist in database. · The password is valid with length between 8 and 16. · The username is invalid with length <2 or length>16. · The invitation code is correct.	
Pre-Conditions: <List the conditions after the test execution > · The user information is not added to the database and the user could not log in with this account. · The front end declines this operation because the username validation works in the front end. · The front end displays the message “Your name length should between 2 to 16” in the screen.	

Notes: [1] Register function. Verify that the register function works correctly. [1.1] Email, username, password and invitation code are mandatory fields. [1.2] Email, username and password have no limit on especial characters. [1.3] Username and password validation works in the front end while email and code validation works in the back end. [1.4] The username is invalid because its length is less than 2 or more than 16. Other parameters are valid. [1.5] The account information is not added to database.
Time constraint: Minimum: 10 min Maximum: 15 min

1.1.5 TC005: Valid email, username, password, and invalid code

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the register function works correctly.	
Setup: <List the pre-conditions to carry through this test case > · The email is valid and does not exist in database. · The password is valid with length between 8 and 16. · The username is valid with length between 2 and 16. · The invitation code is incorrect.	
Pre-Conditions: <List the conditions after the test execution > · The user information is not added to the database and the user could not log in with this account. · The front end receives status code 500 and message “wrong invite code” from the back end. · The front end displays the message “wrong invite code” in the screen.	
Notes: [1] Register function. Verify that the register function works correctly. [1.1] Email, username, password and invitation code are mandatory fields. [1.2] Email, username and password have no limit on especial characters. [1.3] Username and password validation works in the front end while email and code validation works in the back end. [1.4] The invitation code is incorrect. Other parameters are valid. [1.5] The account information is not added to database.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.2 UC002: Login function for supervisor

1.2.1 TC001: Login successful

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the login progress of supervisors is performed successfully.	
Setup: <List the pre-conditions to carry through this test case > · The supervisor completes register.	
Pre-Conditions: <List the conditions after the test execution > · The front-end receives a status code 0 and the information of current supervisor.	
Notes: [1]. Input user email. [2]. Input user password. [3]. Click the “Log in” button to submit the form. * Email and Password are mandatory fields. * Email filed only allow the email standard input. * Password filed allows the input length between 8 and 16. [4]. Try to input the registered user email with corresponding password. * The user logs in to the supervisor home page.	

1.2.2 TC002: Login unsuccessful

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify that the login progress of the supervisor is NOT performed successfully.	
Setup: <List the pre-conditions to carry through this test case >	
Pre-Conditions: <List the conditions after the test execution > · The front-end receive a status code 400 and a error message.	

Notes: [1]. Try to input the email and password without register. * The screen displays the message “The user does not exist! Please check your email!” [2]. Try to input the registered email and the unmatched password. * The screen displays the message “Login fail! Please check your password!” [3]. Try to input the registered email and password belongs to a LEC. * The user logs in to the LEC home page. *<describe the expected outcome here>
Time constraint: Minimum: 20 min Maximum: 25 min

1.3 UC003: Password reset function for supervisor

1.3.1 TC001: Input correct old password and valid and consistent new passwords

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the password is reset.	
Setup: <List the pre-conditions to carry through this test case > · The old password should be the same as the password of the supervisor in database. · The two new passwords should be the same. · The two new passwords should meet the requirement: 8<=length<=16	
Pre-Conditions: <List the conditions after the test execution > · The front end receives status code 200 and the message "Reset password successfully" from the back end. · The front end should display the message "Reset password successfully" in the screen. · The password of the supervisor in database should be changed to the new password.	
Notes: [1] Reset password. Verify that the password is changed. [1.1] Use special chars, capital letters, digits, various allowed numbers of chars (8<=new password length<=16) to fill new password fields. [1.2] Log out and log in with the new password or check the database. Verify if the password is reset correctly.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.3.2 TC002: Input wrong old password and valid and consistent new passwords

Test Type:	**Execution Type:**
Functional	Manual
Objective:	
Verify if the password is reset.	
Setup:	
<List the pre-conditions to carry through this test case > · The old password is not the same as the password of the supervisor in database. · The two new passwords should be the same. · The two new passwords should meet the requirement: 8<=length<=16	
Pre-Conditions:	
<List the conditions after the test execution > · The front end receives status code 400 and the message "Password is wrong!" from the back end. · The front end should display the message "Password is wrong!" in the screen. · The password of the supervisor in database does not change.	
Notes:	
[1] Reset password. Verify that the password is changed. [1.1] The old password should be wrong and the new passwords should be valid and consistent. [1.2] The back end declines the password reset function. [1.2] Log out. Then log in with the new password and it should fail because the password does not reset.	
Time constraint:	
Minimum: 10 min Maximum: 15 min	

1.3.3 TC003: Input correct old password and valid and inconsistent new passwords

Test Type:	**Execution Type:**
Functional	Manual
Objective:	
Verify if the password is reset.	
Setup:	
<List the pre-conditions to carry through this test case > · The old password should be the same as the password of the supervisor in database. · The two new passwords are not the same. · The two new passwords should meet the requirement: 8<=length<=16	
Pre-Conditions:	
<List the conditions after the test execution > · The front end declines this operation and displays the message that the two new passwords are not consistent and requires us to check the information. · The password of the supervisor in database does not change.	

Notes: [1] Reset password. Verify that the password is changed. [1.1] The old password should be correct and the new passwords should be valid and inconsistent. [1.2] The front end declines the password reset function rather than the back end. The back end does not receive the request. [1.2] Log out. Then log in with the new password and it should fail because the password does not reset.
Time constraint: Minimum: 10 min Maximum: 15 min

1.3.4 TC004: Input correct old password and invalid and consistent new passwords

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the password is reset.	
Setup: <List the pre-conditions to carry through this test case > · The old password should be the same as the password of the supervisor in database. · The two new passwords should be the same. · The two new passwords should be invalid: length<8 or length>16 or the new password is the same with the old password	
Pre-Conditions: <List the conditions after the test execution > · The front end declines this operation and displays the message that the two new passwords are not valid because of the length or the same as the old password and requires us to check the information. · The password of the supervisor in database does not change.	
Notes: [1] Reset password. Verify that the password is changed. [1.1] The old password should be correct and the new passwords should be consistent but invalid because of the length or the same as the old password. The front end declines the password reset function rather than the back end. The back end does not receive the request. [1.2] Log out. Then log in with the new password and it should fail because the password does not reset.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.4 UC004: Email reset function for supervisor

1.4.1 TC001: Input correct old email and valid new email

Test Type:	**Execution Type:**
Functional	Manual

Objective: Verify if the email is reset.
Setup: <List the pre-conditions to carry through this test case > · The old email should be the same as the email of the supervisor in database. · The new email should be different from the old email.
Pre-Conditions: <List the conditions after the test execution > · The front end receives status code 200 and the message "Reset email successfully". · The front end displays the message "Reset email successfully" in the screen. · The email of the supervisor in database should be changed to the new email.
Notes: [1] Reset email. Verify that the email is changed. [1.1] The new email should be in the format of an email and could receive emails. In sprint 2, we plan to make the system automatically send verification messages to the emails. [1.2] Log out and log in with the new email or check the database. Verify if the email is reset correctly.
Time constraint: Minimum: 10 min Maximum: 15 min

1.4.2 TC002: Input wrong old email and valid new email

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the email is reset.	
Setup: <List the pre-conditions to carry through this test case > · The old email is not the same as the email of the supervisor in database. · The new email should be different from the old email.	
Pre-Conditions: <List the conditions after the test execution > · The front end declines this operation and displays the warn that the email is not correct and requires us to check the information. The back end does not receive this request. · The email of the supervisor in database should not change.	

Notes: [1] Reset email. Verify that the email is changed. [1.1] The new email should be in the format of an email and could receive emails. In sprint 2, we plan to make the system automatically send verification messages to the emails. [1.2] The front end declines this operation and the back end does not receive this request. [1.3] Log out. Then log in with the new email and it should fail because the email is not reset. The front end declines the email reset function rather than the back end. The back end does not receive the request.
Time constraint: Minimum: 10 min Maximum: 15 min

1.4.3 TC003: Input correct old email and invalid new email

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the email is reset.	
Setup: <List the pre-conditions to carry through this test case > · The old email should be the same as the email of the supervisor in database. · The new email is the same with the old email.	
Pre-Conditions: <List the conditions after the test execution > · The front end declines this operation and displays the warn that the new email should be different from the old email and requires us to check the information. The back end does not receive this request. · The email of the supervisor in database should not change.	
Notes: [1] Reset email. Verify that the email is changed. [1.1] The new email is the same with the old email. [1.2] The front end declines the email reset function rather than the back end. The back end does not receive the request.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.5 UC005: Quiz creating function for supervisor

1.5.1 TC001: Create Quiz basic information

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the quiz basic information can be created successfully.	

Setup: <List the pre-conditions to carry through this test case > · The supervisor completes log in and, · The supervisor clicks the “My Quizzes” button and, · The supervisor stays in the “My Quizzes” page and, · Click the “Create new quiz” button and, · The supervisor stays in the “Create Quiz” page.
Pre-Conditions: <List the conditions after the test execution > · The front-end receive a status code 0 and the current quiz ID.
Notes: [1]. Input the Title [2]. Input the Quiz Overview. [3]. Click the “Continue” button to submit the form. * Title and Quiz Overview are mandatory fields * Title filed allow the input length within 1 to 20 words. * Quiz Overview filed allow the input length within 5 to 300 words. [4]. Try to input the valid value of Title and Quiz Overview and click “continue” button. * The website displays massage “Quiz has been Created, Please continue!” and jump to the questions creation page.

1.5.2 TC002: Create Quiz basic information - Unsuccessful

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the quiz basic information can NOT be created successfully.	
Setup: <List the pre-conditions to carry through this test case > · The supervisor completes log in and, · The supervisor clicks the “My Quizzes” button and, · The supervisor stays in the “My Quizzes” page and, · Click the “Create new quiz” button and, · The supervisor stays in the “Create Quiz” page.	
Pre-Conditions: <List the conditions after the test execution > · The front-end receive the status code 400 and error message.	

Notes: [1]. Input the invalid title and overview of the new Quiz * Error tips will be shown on the screen and the submission will fail. [2]. Click the “cancel” button. * A confirm popup asks “You have not finished yet. Are you sure you wanna cancel it”. * The supervisor jump back to the “My Quizzes” page.

1.5.3 TC003: Create a question of a new quiz

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if creating a new question of current new quiz is preformed successfully.	
Setup: <List the pre-conditions to carry through this test case > · The supervisor completes log in and, · The supervisor creates a new quiz and click “continue” or, · The supervisor edits a exist quiz and click “continue” button.	
Pre-Conditions: <List the conditions after the test execution > · The front-end receive a status code 0	
Notes: [1]. Click the “Create new question” button and, [2]. Input the Question name filed. [3]. Input the Option and Point. * The Question, Option and Point are mandatory fields. * Point field only allows bit digital inputs. * There will be at least two options in one question. [4]. Randomly add or delete options. [5]. Click the “Complete” button to submit the form. * The screen displays message “New question added!” * The screen refreshes the current questions list of current quiz.	

1.5.4 TC004: Create a question of a new quiz - Unsuccessful

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if creating a new question of current new quiz is NOT preformed successfully.	

Setup: <List the pre-conditions to carry through this test case > · The supervisor completes log in and, · The supervisor creates a new quiz and click “continue” or, · The supervisor edits a exist quiz and click “continue” button.
Pre-Conditions: <List the conditions after the test execution > · The front-end receive a status code 400 and error message.
Notes: [1]. Input invalid value of Question name. [2]. Input invalid value of Option name. [3]. Input invalid value of Point. * The screen displays the warning below the input box.

1.6 UC006: Quiz updating function for supervisor

1.6.1 TC001: Update quiz title

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the quiz title	
Setup: <List the pre-conditions to carry through this test case > · supervisor is on the edit quiz page and he/she is ready to click the edit button.	
Pre-Conditions: <List the conditions after the test execution > · Supervisors can see the quiz title has been updated.	
Notes: 1] The front end should display the updated quiz title.	
Time constraint: Minimum: 1 min Maximum: 2 min	

1.6.2 TC001: Update quiz overview

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the supervisor can update quiz overview correctly.	
Setup: <List the pre-conditions to carry through this test case > · supervisor is on the edit quiz page and he/she is ready to click the edit button	

Pre-Conditions: <List the conditions after the test execution > · Supervisors can see the quiz overview has been updated.
Notes: [1] The front end should display the updated quiz overview.
Time constraint: Minimum: 10 seconds Maximum: 1 minute

1.6.3 TC001: Update quiz question

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the Supervisor can update quiz question	
Setup: <List the pre-conditions to carry through this test case > · Supervisor is on the edit question page and he/she is ready to click the edit button to edit the question	
Pre-Conditions: <List the conditions after the test execution > · After execution, the question name has been changed.	
Notes: [1] The front end should display the edited question name.	
Time constraint: Minimum: 10 seconds Maximum: 30 seconds	

1.6.4 TC001: Update quiz option

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the supervisor can update question options	
Setup: <List the pre-conditions to carry through this test case > · supervisor is on the edit question page and he/she is ready to click the edit button · supervisor is on the edit question page and he/she is ready to click the delete option button · supervisor is on the edit question page and he/she is ready to click the "addnew" option button	
Pre-Conditions: <List the conditions after the test execution > · supervisor can see the options have been updated, deleted, or added.	

Notes: [1] The front end should display the edited question options.
Time constraint: Minimum: 10 seconds Maximum: 1 minute

1.7 UC007: Quiz deleting function for supervisor

1.7.1 TC001: Supervisor quiz deleting function

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the supervisor could delete the quizzes.	
Setup: <List the pre-conditions to carry through this test case > · Supervisor creates quizzes in UC005 · Supervisor updates quizzes in UC006 (optional) After the above operations, supervisor could choose and delete the quizzes.	
Pre-Conditions: <List the conditions after the test execution > · If successful: · The front-end receives the status code 0 from the back-end if successful. · The quiz table in the database changes the status of this quiz to 4, meaning the quiz is deleted. · The supervisor and LEC could not view the quiz. · If unsuccessful: · The front-end receives the error status code from the back-end. · The quiz is not deleted.	
Notes: [1] Supervisor quiz deleting function. Verify that the function works correctly. [1.1] correct: all done. [1.2] wrong: · Check whether the front-end logic is correct. · Check whether the parameters in the front-end axios request are correct. · Check whether the back-end operations are correct.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.8 UC008: Information updating function for supervisor

1.8.1 TC001: Supervisor information updating function

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the supervisor could updates his information.	
Setup: <List the pre-conditions to carry through this test case > · Supervisor logs in in UC002 After the above operations, supervisor could updates his information like username.	
Pre-Conditions: <List the conditions after the test execution > · If successful: · The front-end receives the status code 0 from the back-end if successful. · The user table in the database changes the username of this supervisor. · The supervisor could see his username changed in the front-end. · If unsuccessful: · The front-end receives the error status code from the back-end. · The supervisor's username is not changed.	
Notes: [1] Supervisor information updating function. Verify that the function works correctly. [1.1] correct: all done. [1.2] wrong: · Check whether the front-end logic is correct. · Check whether the parameters in the front-end axios request are correct. · Check whether the back-end operations are correct.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.9 UC009: Register function for LEC

1.9.2 TC001: Valid email, password and username

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the register function works correctly.	

Setup: <List the pre-conditions to carry through this test case > · The email is valid and does not exist in database. · The password is valid with length between 8 and 16. · The username is valid with length between 2 and 16.
Pre-Conditions: <List the conditions after the test execution > · The account information is added to the database and the user could log in with this account. · The front end receives status code 200 from the back end. · The front end displays the message “Success!!Please Sign in!” in the screen.
Notes: [1] Register function. Verify that the register function works correctly. [1.1] Email, username and password are mandatory fields. [1.2] Email, username and password have no limit on especial characters. [1.3] Username and password validation works in the front end while email validation works in the back end. [1.4] Log in with this account successfully or check the account information in database.
Time constraint: Minimum: 10 min Maximum: 15 min

1.9.2 TC002: Invalid email, valid password and username

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the register function works correctly.	
Setup: <List the pre-conditions to carry through this test case > · The email is invalid and already exist in database. · The password is valid with length between 8 and 16. · The username is valid with length between 2 and 16.	
Pre-Conditions: <List the conditions after the test execution > · The user information is not added to the database and the user could not log in with this account. · The front end receives status code 500 and message “email exist” from the back end. · The front end displays the message “email exist” in the screen.	

Notes: [1] Register function. Verify that the register function works correctly. [1.1] Email, username and password are mandatory fields. [1.2] Email, username and password have no limit on especial characters. [1.3] Username and password validation works in the front end while email validation works in the back end. [1.4] The email here should exist in database. Other parameters are valid. [1.5] The account information is not added to database.
Time constraint: Minimum: 10 min Maximum: 15 min

1.9.3 TC003: Valid email, username and invalid password

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the register function works correctly.	
Setup: <List the pre-conditions to carry through this test case > · The email is valid and does not exist in database. · The password is invalid with length<8 or length >16. · The username is valid with length between 2 and 16.	
Pre-Conditions: <List the conditions after the test execution > · The user information is not added to the database and the user could not log in with this account. · The front end declines this operation because the password validation works in the front end. · The front end displays the message “Password length should between 8 to 16” in the screen.	
Notes: [1] Register function. Verify that the register function works correctly. [1.1] Email, username and password are mandatory fields. [1.2] Email, username and password have no limit on especial characters. [1.3] Username and password validation works in the front end while email validation works in the back end. [1.4] The password is invalid because its length is less than 8 or more than 16. Other parameters are valid. [1.5] The account information is not added to database.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.9.4 TC004: Valid email, password and invalid username

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the register function works correctly.	
Setup: <List the pre-conditions to carry through this test case > · The email is valid and does not exist in database. · The password is valid with length between 8 and 16. · The username is invalid with length <2 or length>16.	
Pre-Conditions: <List the conditions after the test execution > · The user information is not added to the database and the user could not log in with this account. · The front end declines this operation because the username validation works in the front end. · The front end displays the message “Your name length should between 2 to 16” in the screen.	
Notes: [1] Register function. Verify that the register function works correctly. [1.1] Email, username and password are mandatory fields. [1.2] Email, username and password have no limit on especial characters. [1.3] Username and password validation works in the front end while email validation works in the back end. [1.4] The username is invalid because its length is less than 2 or more than 16. Other parameters are valid. [1.5] The account information is not added to database.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.10 UC010: Login function for LEC

1.10.1 TC001: Login successful

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the login progress of supervisors is performed successfully.	
Setup: <List the pre-conditions to carry through this test case > · The LEC completes register.	
Pre-Conditions: <List the conditions after the test execution > · The front-end receives a status code 0 and the information of current LEC.	

Notes: [1]. Input user email. [2]. Input user password. [3]. Click the “Log in” button to submit the form. * Email and Password are mandatory fields. * Email filed only allow the email standard input. * Password filed allows the input length between 8 and 16. [4]. Try to input the registered user email with corresponding password. * The user logs in to the supervisor home page.

1.10.2 TC002: Login unsuccessful

Test Type: Functional	**Execution Type:** Manual
Objective: Verify that the login progress of the supervisor is NOT performed successfully.	
Setup: <List the pre-conditions to carry through this test case >	
Pre-Conditions: <List the conditions after the test execution > · The front-end receive a status code 400 and a error message.	
Notes: [1]. Try to input the email and password without register. * The screen displays the message “The user does not exist! Please check your email!” [2]. Try to input the registered email and the unmatched password. * The screen displays the message “Login fail! Please check your password!” [3]. Try to input the registered email and password belongs to a supervisor. * The user logs in to the supervisor home page. *<describe the expected outcome here>	
Time constraint: Minimum: 20 min Maximum: 25 min	

1.11 UC011: Password reset function for LEC

1.11.1 TC001: Input correct old password and valid and consistent new passwords

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the password is reset.	

Setup: <List the pre-conditions to carry through this test case > · The old password should be the same as the password of the LEC in database. · The two new passwords should be the same. · The two new passwords should meet the requirement: 8<=length<=16
Pre-Conditions: <List the conditions after the test execution > · The front end receives status code 200 and the message "Reset password successfully" from the back end. · The front end should display the message "Reset password successfully" in the screen. · The password of the supervisor in database should be changed to the new password.
Notes: [1] Reset password. Verify that the password is changed. [1.1] Use special chars, capital letters, digits, various allowed numbers of chars (8<=new password length<=16) to fill new password fields. [1.2] Log out and log in with the new password or check the database. Verify if the password is reset correctly.
Time constraint: Minimum: 10 min Maximum: 15 min

1.11.2 TC002: Input wrong old password and valid and consistent new passwords

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the password is reset.	
Setup: <List the pre-conditions to carry through this test case > · The old password is not the same as the password of the LEC in database. · The two new passwords should be the same. · The two new passwords should meet the requirement: 8<=length<=16	
Pre-Conditions: <List the conditions after the test execution > · The front end receives status code 400 and the message "Password is wrong!" from the back end. · The front end should display the message "Password is wrong!" in the screen. · The password of the LEC in database does not change.	

Notes: [1] Reset password. Verify that the password is changed. [1.1] The old password should be wrong and the new passwords should be valid and consistent. [1.2] The back end declines the password reset function. [1.2] Log out. Then log in with the new password and it should fail because the password does not reset.
Time constraint: Minimum: 10 min Maximum: 15 min

1.11.3 TC003: Input correct old password and valid and inconsistent new passwords

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the password is reset.	
Setup: <List the pre-conditions to carry through this test case > · The old password should be the same as the password of the LEC in database. · The two new passwords are not the same. · The two new passwords should meet the requirement: 8<=length<=16	
Pre-Conditions: <List the conditions after the test execution > · The front end declines this operation and displays the message that the two new passwords are not consistent and requires us to check the information. · The password of the LEC in database does not change.	
Notes: [1] Reset password. Verify that the password is changed. [1.1] The old password should be correct and the new passwords should be valid and inconsistent. [1.2] The front end declines the password reset function rather than the back end. The back end does not receive the request. [1.2] Log out. Then log in with the new password and it should fail because the password does not reset.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.11.4 TC004: Input correct old password and invalid and consistent new passwords

Test Type: Functional	**Execution Type:** Manual

Objective: Verify if the password is reset.
Setup: <List the pre-conditions to carry through this test case > · The old password should be the same as the password of the LEC in database. · The two new passwords should be the same. · The two new passwords should be invalid: length<8 or length>16 or the new password is the same with the old password
Pre-Conditions: <List the conditions after the test execution > · The front end declines this operation and displays the message that the two new passwords are not valid because of the length or the same as the old password and requires us to check the information. · The password of the LEC in database does not change.
Notes: [1] Reset password. Verify that the password is changed. [1.1] The old password should be correct and the new passwords should be consistent but invalid because of the length or the same as the old password. The front end declines the password reset function rather than the back end. The back end does not receive the request. [1.2] Log out. Then log in with the new password and it should fail because the password does not reset.
Time constraint: Minimum: 10 min Maximum: 15 min

1.12 UC012: Email reset function for LEC

1.12.1 TC001: Input correct old email and valid new email

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the email is reset.	
Setup: <List the pre-conditions to carry through this test case > · The old email should be the same as the email of the LEC in database. · The new email should be different from the old email.	
Pre-Conditions: <List the conditions after the test execution > · The front end receives status code 0 and the message "Reset email successfully". · The front end displays the message "Reset email successfully" in the screen. · The email of the LEC in database should be changed to the new email.	

Notes: [1] Reset email. Verify that the email is changed. [1.1] The new email should be in the format of an email and could receive emails. In sprint 2, we plan to make the system automatically send verification messages to the emails. [1.2] Log out and log in with the new email or check the database. Verify if the email is reset correctly.
Time constraint: Minimum: 10 min Maximum: 15 min

1.12.2 TC002: Input wrong old email and valid new email

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the email is reset.	
Setup: <List the pre-conditions to carry through this test case > · The old email is not the same as the email of the LEC in database. · The new email should be different from the old email.	
Pre-Conditions: <List the conditions after the test execution > · The front end declines this operation and displays the warn that the email is not correct and requires us to check the information. The back end does not receive this request. · The email of the LEC in database should not change.	
Notes: [1] Reset email. Verify that the email is changed. [1.1] The new email should be in the format of an email and could receive emails. In sprint 2, we plan to make the system automatically send verification messages to the emails. [1.2] The front end declines this operation and the back end does not receive this request. [1.3] Log out. Then log in with the new email and it should fail because the email is not reset. The front end declines the email reset function rather than the back end. The back end does not receive the request.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.12.3 TC003: Input correct old email and invalid new email

Test Type:	**Execution Type:**
Functional	Manual

Objective: Verify if the email is reset.
Setup: <List the pre-conditions to carry through this test case > · The old email should be the same as the email of the LEC in database. · The new email is the same with the old email.
Pre-Conditions: <List the conditions after the test execution > · The front end declines this operation and displays the warn that the new email should be different from the old email and requires us to check the information. The back end does not receive this request. · The email of the LEC in database should not change.
Notes: [1] Reset email. Verify that the email is changed. [1.1] The new email is the same with the old email. [1.2] The front end declines the email reset function rather than the back end. The back end does not receive the request.
Time constraint: Minimum: 10 min Maximum: 15 min

1.13 UC013: Information updating function for LEC

1.13.1 TC001: LEC information updating function

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the LEC could updates his information.	
Setup: <List the pre-conditions to carry through this test case > · LEC logs in in UC010 After the above operations, LEC could updates his information like username.	
Pre-Conditions: <List the conditions after the test execution > · If successful: · The front-end receives the status code 0 from the back-end if successful. · The user table in the database changes the username of this LEC · The LEC could see his username changed in the front-end. · If unsuccessful: · The front-end receives the error status code from the back-end. · The LEC's username is not changed.	

Notes: [1] LEC information updating function. Verify that the function works correctly. [1.1] correct: all done. [1.2] wrong: · Check whether the front-end logic is correct. · Check whether the parameters in the front-end axios request are correct. · Check whether the back-end operations are correct.
Time constraint: Minimum: 10 min Maximum: 15 min

1.14 UC014: Account deleting function for LEC

1.14.1 TC001: LEC account deleting function

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the LEC could delete his account.	
Setup: <List the pre-conditions to carry through this test case > · LEC registers an account in UC009 · LEC logs in in UC010 After the above operations, the LEC could delete his account.	
Pre-Conditions: <List the conditions after the test execution > · If successful: · The front-end receives the status code 0 from the back-end if successful. · The user table in the database changes the role of this user to 'deleted', meaning the user is deleted. · The user could not log in with this account. · If unsuccessful: · The front-end receives the error status code from the back-end. · The user is not deleted.	
Notes: [1] LEC account deleting function. Verify that the function works correctly. [1.1] correct: all done. [1.2] wrong: · Check whether the front-end logic is correct. · Check whether the parameters in the front-end axios request are correct. · Check whether the back-end operations are correct.	

Time constraint: Minimum: 10 min Maximum: 15 min

1.15 UC015: Quiz viewing function for LEC

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the LEC can see the quiz on the page	
Setup: <List the pre-conditions to carry through this test case > · LEC is on the main page and he/she is ready to click the "I'm a person with lived experience" button	
Pre-Conditions: <List the conditions after the test execution > · If successful: · The front-end receives the status code 0 from the back-end. · LEC can see there are some quizzes on the page. · If unsuccessful: · The front-end receives the error status code from the back-end. · LEC could not see any quizzes.	
Notes: [1] The front end should display all the published public quizzes that are ready to be undertaken by LECs. [2] LEC quiz viewing function. Verify that the function works correctly. [2.1] correct: all done. [2.2] wrong: · Check whether the front-end logic is correct. · Check whether the parameters in the front-end axios request are correct. · Check whether the back-end operations are correct.	
Time constraint: Minimum: 10 seconds Maximum: 1 minute	

1.16 UC0016: Quiz taking function for LEC

1.16.1 TC001: Question selection function

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the question selection system works correctly.	

Setup:

<List the pre-conditions to carry through this test case >

· This function only contains LEC selecting the options of the questions.

· LEC selects one option of a question and click the button 'next' and 'previous' to view other questions. And the front end could automatically calculate the score.

· LEC must select the option for all questions.

Pre-Conditions:

<List the conditions after the test execution >

· If successful:

  · LEC could view the questions and take quiz.

  · The front end should print the total score in the console of the website.

  · The front end could transmit the score to the next page 'getFeedback.vue'.

· If unsuccessful:

  · The front-end receives the error status code from the back-end.

  · LEC could not view the questions.

Notes:

[1] Question selection function. Verify that the function works correctly. Test the current functions including option selecting and score calculating.

[1.1] correct: all done.

[1.2] wrong:

· Check whether the front-end logic is correct.

· Check whether the parameters in the front-end axios request are correct.

· Check whether the back-end operations are correct.

Time constraint:

Minimum: 10 min

Maximum: 15 min

1.17 UC017: Feedback viewing function for LEC after finishing quiz

1.17.1 TC001: Feedback receiving function

Test Type:	**Execution Type:**
Functional	Manual

Objective:

Verify if the LEC could receive correct feedback after finishing a quiz.

Setup:

<List the pre-conditions to carry through this test case >

· After LECs take quizzes as is shown in UC016, they could receive feedback from the back-end based on the score calculated during taking quizzes.

Pre-Conditions: <List the conditions after the test execution > · If successful: · The front-end receives the status code 0 from the back-end. · The front end should display the correct feedback transmitted from the back-end on the screen. · If unsuccessful: · The front-end receives the status code 0 or the error status code from the back-end. · LEC could not see feedback or the feedback is wrong.
Notes: [1] Feedback receiving function. Verify that the function works correctly. [1.1] correct feedback: all done. [1.2] wrong feedback: · Check whether score calculation function in UC016 is correct. · Check whether the front-end logic is correct. · Check whether the parameters in the front-end axios request are correct. · Check whether the back-end operations are correct.
Time constraint: Minimum: 10 min Maximum: 15 min

1.18 UC018: Diary writing function for LEC

1.18.1 TC001: Diary writing function

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the LEC could write diary reflection after taking quizzes and receiving feedback.	
Setup: <List the pre-conditions to carry through this test case > · LEC takes quizzes in UC016 · LEC receives feedback in UC017 After the above operations, they could write diary to record their reflection of the finished quiz.	
Pre-Conditions: <List the conditions after the test execution > · The LEC could input the diary reflection in the input area of the page.	
Notes: [1] Diary writing function. Verify that the function works correctly. [1.1] correct input: all done. [1.2] wrong input: Check the front-end user input function.	

Time constraint: Minimum: 10 min Maximum: 15 min

1.19 UC019: Feedback saving function for LEC

1.19.1 TC001: Feedback saving function

Test Type:	**Execution Type:**
Functional	Manual
Objective: Verify if the LEC could save feedback after taking quizzes, receiving feedback and writing diary.	
Setup: <List the pre-conditions to carry through this test case > · LEC takes quizzes in UC016 · LEC receives feedback in UC017 · LEC logs in in UC010 · LEC writes diary in UC018 (optional) · LEC saves feedback in UC019 (optional) · LEC shares feedback in UC021 (optional) After the above operations, they could save feedback of the finished quiz.	
Pre-Conditions: <List the conditions after the test execution > · If successful: · The front-end receives the status code 0 from the back-end and displays the message 'save successfully' on the screen. · The record table in the database adds a piece of data containing the quiz content, the options chosed, the feedback, the reflection diary, the user id and the date. · LEC could view this feedback in his past feedback. · If unsuccessful: · The front-end receives the error status code from the back-end and displays the message 'not success, save again' on the screen.	
Notes: [1] Feedback saving function. Verify that the function works correctly. [1.1] correct: all done. [1.2] wrong: · Check whether the front-end logic is correct. · Check whether the parameters in the front-end axios request are correct. · Check whether the back-end operations are correct.	
Time constraint: Minimum: 10 min Maximum: 15 min	

1.20 UC020: Past feedback viewing function for LEC

1.20.1 TC001: Past feedback viewing function

Test Type:	**Execution Type:**
Functional	Manual

Objective:

Verify if the LEC could view past feedback of the finished quizzes.

Setup:

<List the pre-conditions to carry through this test case >

- LEC takes quizzes in UC016
- LEC receives feedback in UC017
- LEC logs in in UC010
- LEC writes diary in UC018 (optional)
- LEC saves feedback in UC019
- LEC shares feedback in UC021 (optional)

After the above operations, they could view the past feedback of the finished quiz.

Pre-Conditions:

<List the conditions after the test execution >

- If successful:
  - The front-end receives the status code 0 from the back-end.
  - LEC could view all his feedback record.
  - LEC could click 'view' button to view single detailed feedback record.
- If unsuccessful:
  - The front-end receives the error status code from the back-end.
  - LEC could not view his feedback record.

Notes:

[1] Past feedback viewing function. Verify that the function works correctly.

[1.1] correct: all done.

[1.2] wrong:

- Check whether the front-end logic is correct.
- Check whether the parameters in the front-end axios request are correct.
- Check whether the back-end operations are correct.

Time constraint:

Minimum: 10 min

Maximum: 15 min

1.21 UC021: Feedback sharing function for LEC

1.21.1 TC001: Feedback sharing function

Test Type:	**Execution Type:**
Functional	Manual

Objective:

Verify if the LEC could share feedback after taking quizzes, receiving feedback and writing diary.

Setup:

<List the pre-conditions to carry through this test case >

· LEC takes quizzes in UC016

· LEC receives feedback in UC017

· LEC logs in in UC010 (optional)

· LEC writes diary in UC018 (optional)

· LEC saves feedback in UC019 (optional)

· LEC shares feedback in UC021 (optional)

After the above operations, LEC could share feedback of the finished quiz.

Pre-Conditions:

<List the conditions after the test execution >

· If successful:

  · The front-end receives the status code 0 from the back-end and displays the message 'share successfully' on the screen.

  · If the record table does not has a record, it adds a piece of data.

  · The share table in the database adds a piece of data containing the record id, the feedback, the reflection diary, the user id, share email and the date.

  · The person shared could receive an email.

· If unsuccessful:

  · The front-end receives the error status code from the back-end and displays the message 'not success, save again' on the screen.

Notes:

[1] Feedback sharing function. Verify that the function works correctly.

[1.1] correct: all done.

[1.2] wrong:

· Check whether the front-end logic is correct.

· Check whether the parameters in the front-end axios request are correct.

· Check whether the back-end operations are correct.

Time constraint:

Minimum: 10 min

Maximum: 15 min

1.22 UC022: Suggestion function for LEC on the quiz and user experience

1.22.1 TC001: LEC suggestion function

Test Type:	**Execution Type:**
Functional	Manual

Objective: Verify if the LEC could make suggestions on the quiz and user experience.
Setup: <List the pre-conditions to carry through this test case > · LEC logs in in UC010 (optional) · LEC inputs the suggestion After the above operations, LEC could make suggestions.
Pre-Conditions: <List the conditions after the test execution > · If successful: · The front-end receives the status code 0 from the back-end. · The experience table in the database adds a piece of data containing the suggestion and the date. · The supervisor could view the suggestion in UC024. · If unsuccessful: · The front-end receives the error status code from the back-end.
Notes: [1] LEC suggestion function. Verify that the function works correctly. [1.1] correct: all done. [1.2] wrong: · Check whether the front-end logic is correct. · Check whether the parameters in the front-end axios request are correct. · Check whether the back-end operations are correct.
Time constraint: Minimum: 10 min Maximum: 15 min

1.23 UC023: Past shared feedback viewing function for supervisor

1.23.1 TC001: Supervisor past shared feedback viewing function

Test Type: Functional	**Execution Type:** Manual
Objective: Verify if the supervisor could view past shared feedback of LECs.	

Setup:

<List the pre-conditions to carry through this test case >

- LEC takes quizzes in UC016
- LEC receives feedback in UC017
- LEC logs in in UC010 (optional)
- LEC writes diary in UC018 (optional)
- LEC saves feedback in UC019 (at least one of this condition and the next condition)
- LEC shares feedback in UC021

After the above operations, the supervisor could view the past shared feedback of the finished quiz.

Pre-Conditions:

<List the conditions after the test execution >

- If successful:
  - The front-end receives the status code 0 from the back-end.
  - Supervisor could view all past shared feedback record.
  - Supervisor could click 'view' button to view single detailed feedback record.
- If unsuccessful:
  - The front-end receives the error status code from the back-end.

Notes:

[1] Past feedback viewing function. Verify that the function works correctly.

[1.1] correct: all done.

[1.2] wrong:

- Check whether the front-end logic is correct.
- Check whether the parameters in the front-end axios request are correct.
- Check whether the back-end operations are correct.

Time constraint:

Minimum: 10 min

Maximum: 15 min

1.24 UC024: Suggestion viewing function for supervisor

1.24.1 TC001: Supervisor suggestion viewing function

Test Type:	**Execution Type:**
Functional	Manual

Objective:

Verify if the supervisor could view suggestions of LECs.

Setup:

<List the pre-conditions to carry through this test case >

- LEC makes suggestions in UC022

After the above operations, the supervisor could view suggestions of LECs.

Pre-Conditions:

<List the conditions after the test execution >

· If successful:

  · The front-end receives the status code 0 from the back-end.

  · Supervisor could view all LECs' suggestions record.

· If unsuccessful:

  · The front-end receives the error status code from the back-end.

Notes:

[1] Supervisor suggestion viewing function. Verify that the function works correctly.

[1.1] correct: all done.

[1.2] wrong:

  · Check whether the front-end logic is correct.

  · Check whether the parameters in the front-end axios request are correct.

  · Check whether the back-end operations are correct.

Time constraint:

Minimum: 10 min

Maximum: 15 min

Data Entry

This section describes the entry data that will be used by more than one test case, avoiding data replication. These data are referenced by the test cases.

Data Set Name: Sign in Data	
Description: This data set contains input data from users to log into the system	
Field	**Value**
Email	jayden@gmail.com
Password	12345678

Data Set Name: Register Data	
Description: This data set contains input data from users to register an account	
Field	**Value**
Email	jayden@gmail.com
Password	12345678
Username	Jayden
Code	123123

Data Set Name: Reset Password Data	
Description: This data set contains input data from users to reset password	
Field	**Value**
Old Password	12345678
New Password	87654321
Confirm Password	87654321

Data Set Name: Reset Email Data	
Description: This data set contains input data from users to reset Email	
Field	**Value**
Old Email	jayden@gmail.com
New Email	jayden1@gmail.com

Data Set Name: Create Quiz Overview Data	
Description: This data set contains input data from users to create a quiz overview	
Field	**Value**
Title	Communication
Overview	Being able to communicate effectively is perhaps the most important of all life skills. It is what enables us to pass information to other people, and to understand what is said to us. ... Communication skills may take a lifetime to master—if indeed anyone can ever claim to have mastered them.

Data Set Name: Create question Data	
Description: This data set contains input data from users to create questions	
Field	**Value**
Question Name	When the group needs suggestions, I...
Option1	Do not make suggestions
Option2	Tell the group what to do
point for Option1	1
point for Option2	2

Data Set Name: Create Feedback Data	
Description: This data set contains input data from users to create feedbacks	
Field	**Value**
Feedback Name	You did pretty well in ...
Score Range(From)	1
Score Range(To)	5

Data Set Name: Reset Username Data	
Description: This data set contains input data from users to reset user name	
Field	**Value**
New Username	John

Data Set Name: Share Email Data	
Description: This data set contains input data from users to input the supervisor email	
Field	**Value**
Email	Josh@gmail.com