

# Boxjelly-Testing document

Versions:

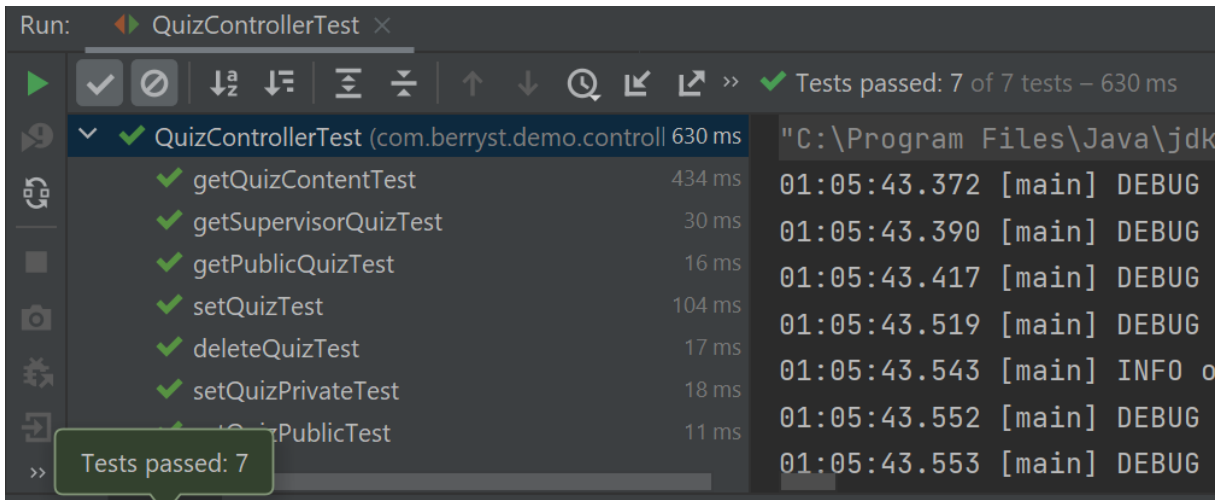
Version ID	Date	Changes
1.0	18/09/2021	Add manual tests and automatic tests for sprint 1
2.0	23/09/2021	Add automatic testing and manual integration testing for backend Add manual testing for frontend

## V2.0

### Backend

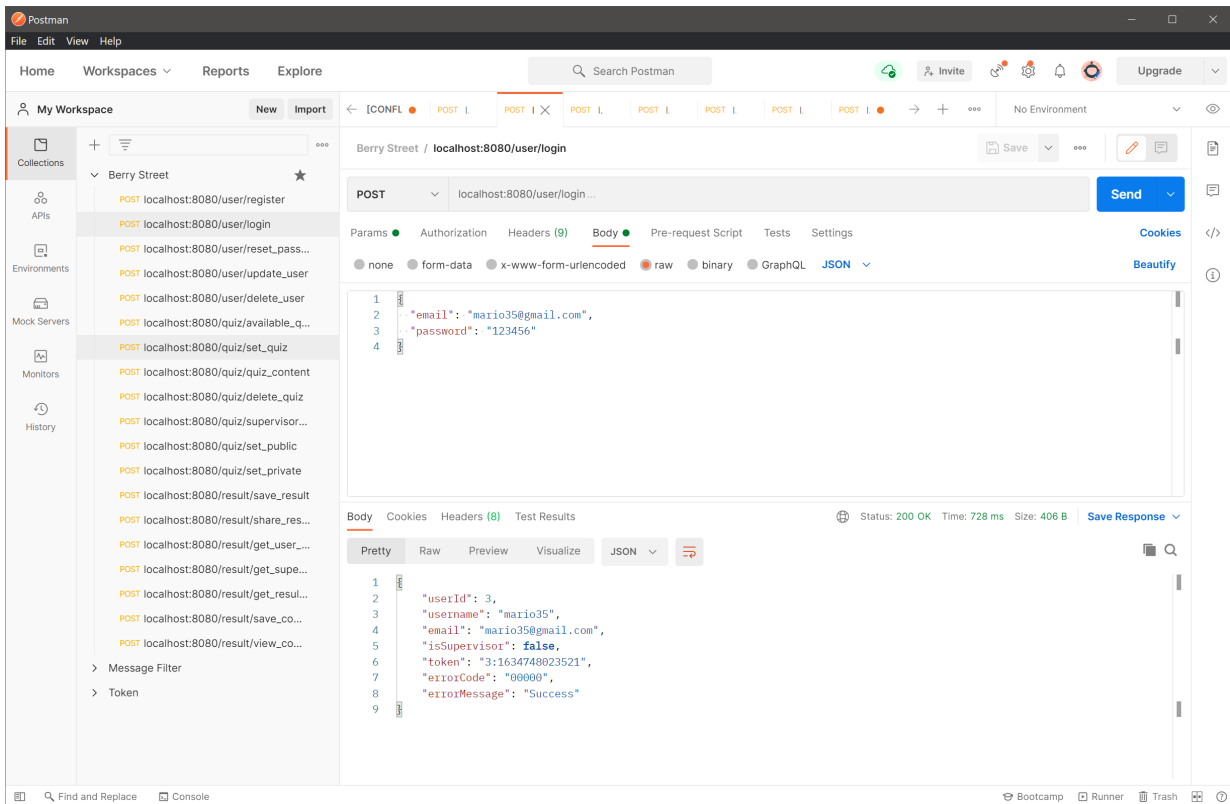
#### Automatic Testing

The automatic tests for our SpringBoot backend can be found under [Backend/src/test/java/com/berryst/demo/](#). By running QuizControllerTest.java, UserControllerTest.java, and ResultControllerTest.java, all the functions of our backend program will be automatically tested. These include 19 tests for our controllers. Tests use JUnit 5.0 to assert if the result of a function matches the correct result and MockMVC to mock the input of a function and corresponding services without actually running the service. Because MySQL server does not support transactions, we use MockMVC to avoid database contamination during testing. We will write and execute automatic testing when we finished a function or user story. If a test fails, we will debug and fix this issue and push a new commit to git with "fix(scope):" beginning.



#### Integration Testing

We use Postman to do our integration testing for backend. Since our SpringBoot controller actually receives HTTP messages from the frontend and Postman let us define an HTTP message and send it to backend. This test requires backend program running and will perform tests as an actual user by sending real HTTP POST messages. To determine if a test is passed, the test staff need to manually check database results after running a test and investigate backend logs to see if there is an Exception. If a test fails, we will debug and fix this issue and push a new commit to git with "fix(scope):" beginning. Our test data can be reused by importing a JSON configuration from postman which can be found on [data/Berry Street.postman\\_collection.json](#). This file contains all the HTTP requests corresponding to different backend controller functions.



## System Testing/Manual Testing

The list of manual testing can be found below. We have first built the database with some predefined data. All the three components of our project, including database, backend, and frontend, need to be run. MySQL database is run on port 3306, backend is run on port 8080, and frontend is run on port 3000. The testing will make actions with some specific data or behavior, and see if the system gives out a correct response. Since the automatic and integration tests are all passed, if a system test fails, it means some of the codes in communicating between frontend and backend need to be fixed.

## Error Handling

Refer to [Boxjelly-Error Handling](#) for details.

## Frontend

### Case 3: Take quiz

#### UC3.7: US02 Go to next/previous question

Test Type:	Execution Type:
Functional	Manual
Objective:	
verify if the LEC could view the chosen selection at previous question	
Setup:	
My quiz:	
1. LEC could click on "<" button to the previous question	

Pre-Conditions:

1. user start taking the quiz
2. user select an answer at previous question

Notes:

1. When user choose to go back to previous question, the system need to show what they have already selected
2. The chosen answer is highlighted

Time constraint:

N/A, give user enough time to view the question and choice

## Case 4: Reset Password

### UC4.1: US20 Login in

Test Type:	Execution Type:
Functional	Manual
Objective:	
Implement the update password function properly when user commit the valid username and email pair	
Setup:	
Username :	
<ol style="list-style-type: none"><li>1. cannot be empty</li><li>2. length: 2-16</li><li>3. cannot be repeated</li><li>4. not allowed white space</li><li>5. existed username in dataset</li></ol>	
Email	
<ol style="list-style-type: none"><li>1. cannot be empty</li><li>2. cannot be the same email account with other registered account</li><li>3. meat with the format "<a href="#">xx@xx.com</a>"</li><li>4. not allowed white space</li><li>5. same email address for this username, stored in dataset</li></ol>	
Password:	
<ol style="list-style-type: none"><li>1. cannot be empty</li><li>2. length: 8-16</li><li>3. have number combined with character</li><li>4. not allowed white space</li></ol>	
Repeat Password:	
<ol style="list-style-type: none"><li>1. cannot be empty</li><li>2. length: 8-16</li><li>3. have number combined with character</li><li>4. same with password</li><li>5. not allowed white space</li></ol>	
Pre-Conditions:	
<ol style="list-style-type: none"><li>1. The user can open this reset password page easily with the guidance of web introduction "forget password?" link</li><li>2. User can receive the message for the successful password update</li><li>3. User can receive error message for the username or email input</li></ol>	
Notes:	
The following tasks to implement	
<ol style="list-style-type: none"><li>1. User can receive error message for the unmatched username and email pair</li></ol>	

Time constraint:

N/A, give user enough time to update the new password

## Case 5: Create quiz

### UC5.1: US13 Create quiz

Test Type:	Execution Type:
Functional	Manual
Objective:	
Implement the create quiz function properly when supervisor want to create a new quiz	
Setup:	
<ol style="list-style-type: none"><li>1. User can choose the create quiz button</li><li>2. User can edit the quiz title</li><li>3. User can write the quiz introduction</li></ol>	
Pre-Conditions:	
<ol style="list-style-type: none"><li>1. Supervisor can access the dashboard to view the create quiz button</li></ol>	
Notes:	
<ol style="list-style-type: none"><li>1. Modify the UI style to keep the quiz format consistent.</li></ol>	
Time constraint:	
N/A	

### UC5.2: US13 Create quiz

Test Type:	Execution Type:
Functional	Manual
Objective:	
Assign the score for each candidate answer.	
Setup:	
<ol style="list-style-type: none"><li>1. User can add a new answer.</li><li>2. New answer will automatically add 1 score</li><li>3. User can edit the answer content</li><li>4. User can change the answer score</li></ol>	
Pre-Conditions:	
<ol style="list-style-type: none"><li>1. Supervisor can access the create quiz page</li></ol>	
Notes:	
<ol style="list-style-type: none"><li>1. Modify the UI style to keep the quiz format consistent.</li><li>2. The data will be sent to the dataset when committed</li></ol>	
Time constraint:	
N/A	

# Test case

## Case 1: Register

### UC20: US09 Register Account as LEC

Test Type:	Execution Type:
Functional	Manual
Objective:	
Implement the register function properly, limit the valid account	
Setup:	
Username :	
<ol style="list-style-type: none"><li>1. cannot be empty</li><li>2. length: 2-16</li><li>3. cannot be repeated</li><li>4. not allowed white space</li></ol>	
Password:	
<ol style="list-style-type: none"><li>1. cannot be empty</li><li>2. length: 8-16</li><li>3. have number combined with character</li><li>4. not allowed white space</li></ol>	
Repeat Password:	
<ol style="list-style-type: none"><li>1. cannot be empty</li><li>2. length: 8-16</li><li>3. have number combined with character</li><li>4. same with password</li><li>5. not allowed white space</li></ol>	
Email	
<ol style="list-style-type: none"><li>1. cannot be empty</li><li>2. cannot be the same email account with other registered account</li><li>3. meet with the format "xx@xx.com"</li><li>4. not allowed white space</li></ol>	
Pre-Conditions:	
<ol style="list-style-type: none"><li>1. The user can open this register page easily with the guidance of web introduction</li><li>2. User can receive the message for the successful account creation</li></ol>	
Notes:	
The following tasks to implement	
<ol style="list-style-type: none"><li>1. User can receive the message shows the hint for invalid account input</li></ol>	
Time constraint:	
N/A, give user enough time to create the valid account	

### UC1.2: US19 Register Account as Supervisor

Test Type:	Execution Type:
Functional	Manual
Objective:	
Implement the register function properly, limit the valid account	

<p>Setup:</p> <p>Username :</p> <ol style="list-style-type: none"> <li>1. cannot be empty</li> <li>2. length: 2-16</li> <li>3. cannot be repeated</li> <li>4. not allowed white space</li> </ol> <p>Password:</p> <ol style="list-style-type: none"> <li>1. cannot be empty</li> <li>2. length: 8-16</li> <li>3. have number combined with character</li> <li>4. not allowed white space</li> </ol> <p>Repeat Password:</p> <ol style="list-style-type: none"> <li>1. cannot be empty</li> <li>2. length: 8-16</li> <li>3. have number combined with character</li> <li>4. same with password</li> <li>5. not allowed white space</li> </ol> <p>Email</p> <ol style="list-style-type: none"> <li>1. cannot be empty</li> <li>2. cannot be the same email account with other registered account</li> <li>3. must with the format "xx@xx.com"</li> <li>4. not allowed white space</li> </ol> <p>As supervisor</p> <ol style="list-style-type: none"> <li>1. "As supervisor" button could be chosen</li> </ol>
<p>Pre-Conditions:</p> <ol style="list-style-type: none"> <li>1. The user can open this register page easily with the guidance of web introduction</li> <li>2. User can receive the message for the successful account creation</li> </ol>
<p>Notes:</p> <p>The following tasks to implement</p> <ol style="list-style-type: none"> <li>1. User can view the recent published quiz</li> </ol>
<p>Time constraint:</p> <p>N/A, give user enough time to create the valid account</p>

### UC1.3: US19 Register Account Controller

Test Type:	Execution Type:
Functional	Manual
<p>Objective:</p> <p>Backend UserController should return the correct result when the frontend sends an HTTP message to URL under /test</p>	
<p>Setup:</p> <ol style="list-style-type: none"> <li>1. Run backend program in IntelliJ IDEA</li> <li>2. Check URL in Backend/src/main/java/com/berryst/demo/controller/TestController.java</li> <li>3. Access the URL in the browser and append required parameters (username, password, isSupervisor)</li> <li>4. Check the returned result</li> </ol>	
<p>Pre-Conditions:</p> <ol style="list-style-type: none"> <li>1. The user can open this register page easily with the guidance of web introduction</li> <li>2. User can receive the message for the successful account creation</li> </ol>	

Notes:

The following tasks to implement

1. User can view the recent published quiz

Time constraint:

N/A, give user enough time to input the valid account

## UC1.4: US19 Register Account Service

Test Type:

Functional

Execution Type:

Automatic

Objective:

Backend UserService should return the correct object when calling functions related to register the account.

Setup:

1. Navigate test file in Backend/src/test/java/com/berryst/demo/service/impl/ResultServiceImplTest.java
2. Run UserServiceImplTest

## Case 2: Log in

### UC2.1: US10 Log in as LEC

Test Type:

Functional

Execution Type:

Manual

Objective:

the login function can work properly

Setup:

test pairs:

1. empty username
2. empty password
3. wrong password
4. wrong username

Expected status:

1. receive error message for wrong username and password pairs
2. receive error message for empty input or cannot press login button with empty input

Pre-Conditions:

1. could login with the right username and password pairs

Notes:

1. required to add "forget password " link to find the password
2. required to add detailed message to hint the error for wrong username and password input.

Time constraint:

N/A, give user enough time in case forget the right username and password pairs

### UC2.2: US20 Login as supervisor

Test Type:	Execution Type:
Functional	Manual
Objective:	
the login function can work properly	
Setup:	
test pairs:	
<ol style="list-style-type: none"> <li>1. empty username</li> <li>2. empty password</li> <li>3. wrong password</li> <li>4. wrong username</li> </ol>	
Expected status:	
<ol style="list-style-type: none"> <li>1. receive error message for wrong username and password pairs</li> <li>2. receive error message for empty input or cannot press login button with empty input</li> </ol>	
Pre-Conditions:	
<ol style="list-style-type: none"> <li>1. could login with the right username and password pairs</li> </ol>	
Notes:	
<ol style="list-style-type: none"> <li>1. required to add "forget password " link to find the password</li> <li>2. required to add detailed message to hint the error for wrong username and password input.</li> </ol>	
Time constraint:	
N/A, give user enough time in case forget the right username and password pairs	

### UC2.3: US19 Login Account Controller

Test Type:	Execution Type:
Functional	Manual
Objective:	
Backend UserController should return the correct result when the frontend sends an HTTP message to URL under /test	
Setup:	
<ol style="list-style-type: none"> <li>1. Run backend program in IntelliJ IDEA</li> <li>2. Check URL in Backend/src/main/java/com/berryst/demo/controller/TestController.java</li> <li>3. Access the URL in the browser and append required parameters (username, password)</li> <li>4. Check the returned result</li> </ol>	
Pre-Conditions:	
<ol style="list-style-type: none"> <li>1. could login with the right username and password pairs</li> </ol>	
Notes:	
Time constraint:	
N/A	

### UC2.4: US19 Login Service

Test Type:	Execution Type:
Functional	Automatic
Objective:	
Backend UserService should return the correct object when calling functions related to the user login.	



Setup:

1. Navigate test file in Backend/src/test/java/com/berryst/demo/service/impl/ResultServiceImplTest.java
2. Run UserServiceImplTest

## Case 3: Undertake Quizzes

### UC3.1: US01 View reflection quizzes

Test Type:	Execution Type:
Functional	Manual
Objective: Validate if the LEC could view the public and private quiz	
Setup: My quiz: <ol style="list-style-type: none"><li>1. LEC could view the public quizzes</li><li>2. Add and delete the published quiz set, the user could view the most recent public quizzes</li></ol>	
Pre-Conditions: <ol style="list-style-type: none"><li>1. user could view the public quiz without login</li></ol>	
Notes: <ol style="list-style-type: none"><li>1. make the direction button for LEC more clear</li></ol>	
Time constraint: N/A, give user enough time to view the quiz page and find out the one they want to take	

### UC3.2: US02 Select quiz

Test Type:	Execution Type:
Functional	Manual
Objective: verify if the LEC could select the quiz to take	
Setup: My quiz: <ol style="list-style-type: none"><li>1. LEC could click on the public quizzes</li><li>2. The page correctly redirect to the intro page of the selected quiz.</li></ol>	
Pre-Conditions: <ol style="list-style-type: none"><li>1. user could select the public quiz without login</li></ol>	
Notes: <ol style="list-style-type: none"><li>1. Function works good</li></ol>	
Time constraint: N/A, give user enough time to view the quiz page and find out the one they want to take	

### UC3.3: US02 Undertake quiz

Test Type:	Execution Type:
Functional	Manual
Objective:	
verify if the LEC could take the quiz they selected	
Setup:	
My quiz:	
<ol style="list-style-type: none"> <li>1. LEC could start taking quiz</li> <li>2. LEC could see the questions and choices.</li> </ol>	
Pre-Conditions:	
<ol style="list-style-type: none"> <li>1. user could start taking the quiz without login</li> </ol>	
Notes:	
<ol style="list-style-type: none"> <li>1. User interface for showing quiz question and select choice should be more clear to recognize</li> </ol>	
Time constraint:	
N/A, give user enough time to view the question and choice	

#### UC3.4: US02 Select choice

Test Type:	Execution Type:
Functional	Manual
Objective:	
verify if the LEC could correctly choose the choice they want	
Setup:	
My quiz:	
<ol style="list-style-type: none"> <li>1. LEC could click on any choice</li> <li>2. LEC could see a reflection on which choice is been selected</li> </ol>	
Pre-Conditions:	
<ol style="list-style-type: none"> <li>1. user start taking the quiz</li> </ol>	
Notes:	
<ol style="list-style-type: none"> <li>1. User interface for selecting choice should be more clear to recognize</li> </ol>	
Time constraint:	
N/A, give user enough time to view the question and choice	

#### UC3.6: US03 See feedback from quiz

Test Type:	Execution Type:
Functional	Manual
Objective:	
verify if the LEC could correctly get the result and feedback	

Setup:

My quiz:

1. LEC could click finish the quiz after select choice of the last question
2. LEC could view the feedback based on the choice they selected

Pre-Conditions:

1. LEC finish taking the quiz

Notes:

1. The function works well

Time constraint:

N/A, give user enough time to view the question and choice