Boxjelly-Testing document

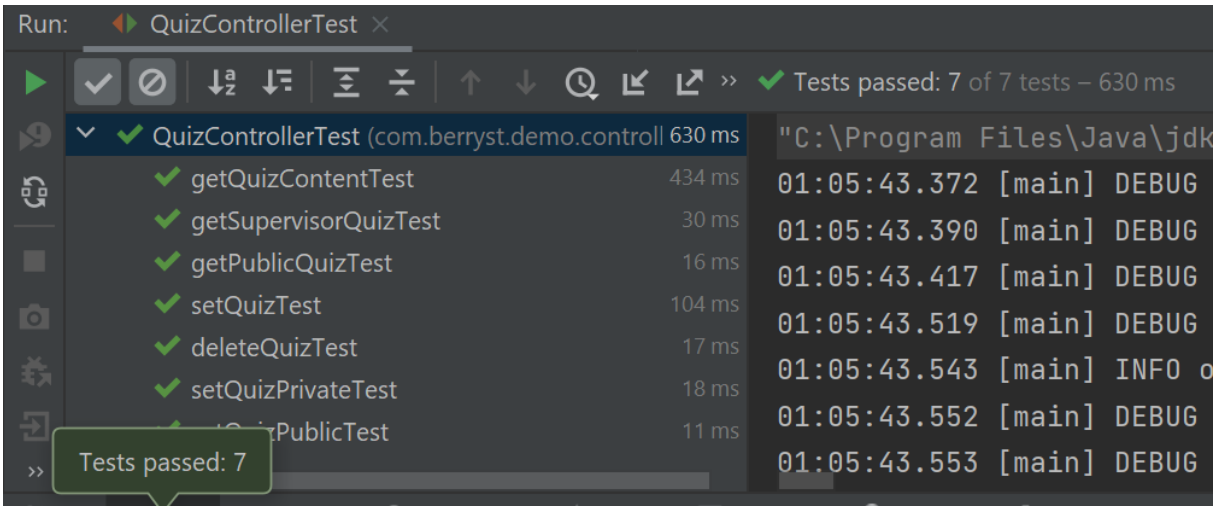
Versions:

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

V2.0

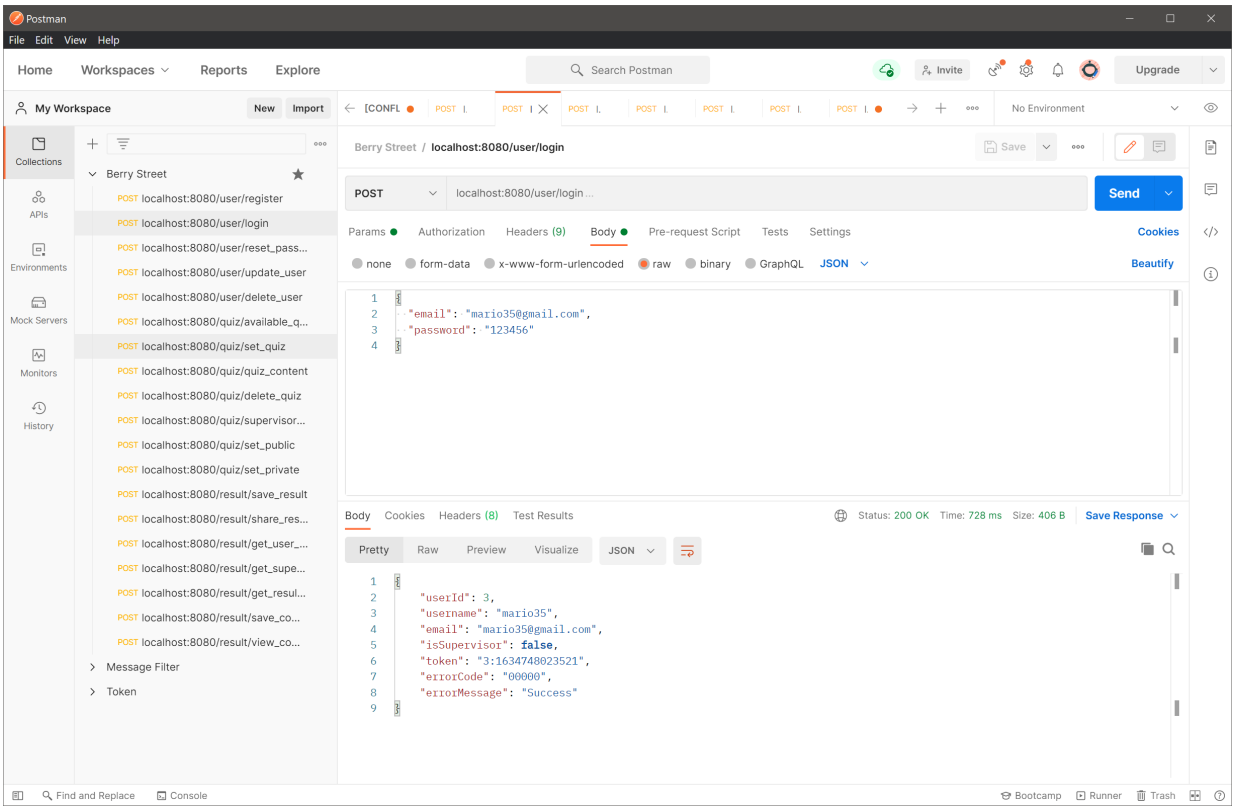
Automatic Testing

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.



Error Handling

Refer to Boxjelly-Error Handling for details.

V1.0

Test case

Case 1: Register

UC1.1: US09 Register Account as LEC

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

Setup:

Username :

1. cannot be empty
2. length: 2-16
3. cannot be repeated
4. not allowed white space

Password:

1. cannot be empty
2. length: 8-16
3. have number combined with character
4. not allowed white space

Repeat Password:

1. cannot be empty
2. length: 8-16
3. have number combined with character
4. same with password
5. not allowed white space

Email

1. cannot be empty
2. cannot be the same email account with other registered account
3. meat with the format "xx@xx.com"
4. not allowed white space

Pre-Conditions:

1. The user can open this register page easily with the guidance of web introduction
2. User can receive the message for the successful account creation

Notes:

The following tasks to implement

1. User can receive the message shows the hint for invalid account input

Time constraint:

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

UC1.2: US19 Register Account as Supervisor

Test Type: Functional	Execution Type: Manual

Objective:

Implement the register function properly, limit the valid account

Setup:

Username :

1. cannot be empty
2. length: 2-16
3. cannot be repeated
4. not allowed white space

Password:

1. cannot be empty
2. length: 8-16
3. have number combined with character
4. not allowed white space

Repeat Password:

1. cannot be empty
2. length: 8-16
3. have number combined with character
4. same with password
5. not allowed white space

Email

1. cannot be empty
2. cannot be the same email account with other registered account
3. meat with the format "xx@xx.com"
4. not allowed white space

As supervisor

1. "As supervisor" button could be chosen

Pre-Conditions:

1. The user can open this register page easily with the guidance of web introduction
2. User can receive the message for the successful account creation

Notes:

The following tasks to implement

1. User can view the recent published quiz

Time constraint:

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

UC1.3: US19 Register Account Controller

Test Type: Functional	Execution Type: Manual

Objective:

Backend UserController should return the correct result when the frontend sends an HTTP message to URL under /test

Setup:

1. Run backend program in IntelliJ IDEA
2. Check URL in Backend/src/main/java/com/berryst/demo/controller/TestController.java
3. Access the URL in the browser and append required parameters (username, password, isSupervisor)
4. Check the returned result

Pre-Conditions:

1. The user can open this register page easily with the guidance of web introduction
2. User can receive the message for the successful account creation

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: 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	

UC2.3: US19 Login Account Controller

Test Type: Functional	Execution Type: Manual
Objective: Backend UserController should return the correct result when the frontend sends an HTTP message to URL under /test	
Setup: 1. Run backend program in IntelliJ IDEA 2. Check URL in Backend/src/main/java/com/berryst/demo/controller/TestController.java 3. Access the URL in the browser and append required parameters (username, password) 4. Check the returned result	
Pre-Conditions: 1. could login with the right username and password pairs	
Notes:	
Time constraint: N/A	

UC2.4: US19 Login Service

Test Type: Functional	Execution Type: 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: Functional	Execution Type: Manual
Objective: Validate if the LEC could view the public and private quiz	
Setup: My quiz: 1. LEC could view the public quizzes 2. Add and delete the published quiz set, the user could view the most recent public quizzes	
Pre-Conditions: 1. user could view the public quiz without login	
Notes: 1. make the direction button for LEC more clear	
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: Functional	Execution Type: Manual
Objective: verify if the LEC could select the quiz to take	
Setup: My quiz: 1. LEC could click on the public quizzes 2. The page correctly redirect to the intro page of the selected quiz.	
Pre-Conditions: 1. user could select the public quiz without login	
Notes: 1. Function works good	
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: Functional	Execution Type: Manual
Objective: verify if the LEC could take the quiz they selected	
Setup: My quiz: 1. LEC could start taking quiz 2. LEC could see the questions and choices.	
Pre-Conditions: 1. user could start taking the quiz without login	
Notes: 1. User interface for showing quiz question and select choice should be more clear to recognize	
Time constraint: N/A, give user enough time to view the question and choice	

UC3.4: US02 Select choice

Test Type: Functional	Execution Type: Manual
Objective: verify if the LEC could correctly choose the choice they want	
Setup: My quiz: 1. LEC could click on any choice 2. LEC could see a reflection on which choice is been selected	
Pre-Conditions: 1. user start taking the quiz	
Notes: 1. User interface for selecting choice should be more clear to recognize	
Time constraint: N/A, give user enough time to view the question and choice	

UC3.5: US02 Go to next/previous question

Test Type: Functional	Execution Type: Manual
Objective: verify if the LEC could correctly go to the next/previous question	

Setup: My quiz: 1. LEC could click on ">" button to the next question 2. LEC could click on "<" button to the previous question
Pre-Conditions: 1. user start taking the quiz
Notes: 1. When user choose to go back to previous question, the system need to show what they have already selected
Time constraint: N/A, give user enough time to view the question and choice

UC3.6: US03 See feedback from quiz

Test Type: Functional	Execution Type: 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	