**Quote Keeper**

In this project, you will write a feature test for the development of a web application that allows users to save and display a collection of their favorite quotes. We will call this app “Quote Keeper”.

Quote Keeper should allow users to enter the following information about each quote they would like to save to the app:

* The quote itself
* Who the quote is attributed to
* The source of the quote:
  + Book
  + Article
  + Speech
  + IRL

Because this course focuses on feature tests at the top level of the TDD Testing Pyramid, this project will only cover the feature tests process up to the point of getting an error about the exercise phase of your test suite. In order to progress we would need to drop down to the server layer of the Testing Pyramid, which is covered in another course. The completed form of this project will have a feature test suite that is ready to move on to tests at the server level.

The steps below will guide you through the process of using TDD to write a feature test for the Quote Keeper web app.

After you complete each step, run your test suite in the terminal to see the results of your test and evaluate what code to write next in response to the message.

If you get stuck during this project or would like to see an experienced developer work through it, click “**Get Help**“ to see a **project walkthrough video**.

**Tasks**

**13/13Complete**

Mark the tasks as complete by checking them off

**Quote Keeper Feature Test**

**1.**

The overall scope of our testing suite is that we want to test behavior that happens when a user visits our project root, which will be the homepage of our website.

In the file **user-visits-root-test.js**, write a describe block with a docstring that states the context that you will be testing your feature in, followed by an empty callback function.

Hint

A good docstring would be:

User visits root

**2.**

Nest a second describe block with a docstring that states what feature you are testing, followed by an empty callback function.

Hint

Since we are writing a test for the Quote Keeper app, the docstring might be something like:

posting a quote

**3.**

Inside the empty callback function, add an it block with a docstring that describes the desired behavior of the feature you are testing, followed by an empty callback function.

Hint

The desired behavior of our feature could be something like:

saves quote and metadata submitted by user

**4.**

For the next step you will need to have a quote to use as an example in writing your test. The information you will need is

* The quote itself
* Who the quote is attributed to
* The source of the quote

For example:

**The quote itself:** ‘Our deepest fear is not that we are inadequate. Our deepest fear is that we are powerful beyond measure.’

**Who the quote is attributed to:** Marianne Williamson.

**The source of the quote:** A Return to Love: Reflections on the Principles of A Course in Miracles.

[If would like help choosing a quote, you can view Bruce Lee Quotes by clicking here.](https://en.wikiquote.org/wiki/Bruce_Lee)

**5.**

**Setup** Inside the the callback function for the it block, setup three const variables that you will use to mimic the user’s responses to each of the input fields we are testing for. Set each of the variables to a string based on the example quote you chose in the previous step. The variable names should be:

* quote
* attributed
* source

Hint

For example:

const quote = 'Our deepest fear is not that we are inadequate. Our deepest fear is that we are powerful beyond measure.';

const attributed = 'Marianne Williamson';

const source = 'A Return to Love: Reflections on the Principles of A Course in Miracles';

**6.**

**Exercise** Use the global browser variable to set the url of our test browser to the root of our project by calling .url() and passing it an argument of ‘/’.

**7.**

Next, fill in the contents of the quote by telling our browser to set the value of a particular <textarea> element.

To do so, call browser.setValue() with textarea[id=quote], followed by the quote variable that you created earlier.

Hint

You can reference the poetry web example from the lesson for the browser.setValue() syntax:

browser.setValue('textarea[id=poem]', poem);

**8.**

Call .setValue() on the global browser variable to set the value of an input element with id =attributed, to your const variable named attributed.

Hint

Try the same syntax as step 6, except change the arguments to match the elements described above.

**9.**

Call .setValue() on the global browser variable to set the value of an input element with id=source, to your const variable named source.

**10.**

Call .click() on the global browser variable, and pass it as an argument an input with the type=submit.

**11.**

\**Verify \**

Write three assert statements, to verify that the substring of each of your three const variables from the setup phase of your test are included in the string that is made up of all the text from the element with id=quotes.

Hint

To retrieve the text of a particular div you can use browser.getText(), and pass the element you want the text of as an argument.

To write an assert statement you can use the syntax:

assert.include(haystack, needle)

**12.**

Eventually, we will want to verify that once the user submits their data, it will be sent to the server, and then be accessible in the browser, inside a div with id=quotes and name=quotes in our **index.html**. In this course, we are only dealing with tests at the top level of the Testing Pyramid, so the final step in this project will be writing the necessary HTML code in **index.html** to get the test to pass— until the point of receiving an assertion error about the exercise phase.

Go to your **index.html** file.

**13.**

Run your test suite and fix each error you receive one at a time, focusing on the feedback about the HTML that the test expected. input and textarea elements should be given both name and id attributes with the same value.

Address each error by creating the necessary element in your **index.html** page, until you reach the assertion error concerning your exercise phase, which would propel you to move down to the server level test.

Hint

That error will include the following — except it will be your quote instead of the example quote:

