Bike Testers exercise

Starting point

What do I have

Bike Shop App  
Set of unit tests

What do I need to do

Design/build/document/run an integration test suite.

How to run the app

Open cmd  
Cd C:\Users\User\Documents\GitHub\test-testers\app  
Python -m SimpleHTTPServer  
Go to browser  
Localhost:8000 to open app  
Localhost:8000/test to open tests

Understand how the unit test works

Jasmine was used to conduct the unit tests.  
The Jasmine index file uses the app source js file and the jasmine unit test js file.  
Understand how this works, and possibly use as a basis for understanding what integration tests to perform.

Initial thoughts on the Integration tests

In the problem spec, it is stated that the integration tests need to prove that the app functions according to the User story:

**The User story**

As a user, assuming I have access to a modern web browser (Chrome) with an internet connection and I am visiting the page

* I would like to see a list/grid of bikes based on /app/bikes.json
* I would like to see a the name, an image, a description and the class for each bike
* I would like to be able to sort the bikes into a custom order based on class
* I would like my custom order to be saved and not change when I refresh the page
* A JSON object with all the relevant bike data is available at <https://jujhar.com/bikes.json>; feel free to call that URL directly from your page or host it on your server if you're using one.

This suggests to me that I need to test that the UI is outputting what the controller is giving to it, as the unit tests have already exercised the controller functions.

A internet search on UI integration testing yielded some good information, the following link being of note:

https://stackoverflow.com/questions/11741738/frontend-testing-what-and-how-to-test-and-what-tool-to-use

There seem to be two candidate tools:

Jasmine-jquery

or

Cypress

Cypress is causing issues when extracting from zip, therefore I have gone with jasmine-jquery.

I have created a file at app/test/jasmine-jquery.js  
I have modified file at app/test/index.html

Test Plan

Scope of testing: Testing should verify the User story given in the Problem spec.

Test Scenario

Typical user workflow as per user story to be tested

Test cases

See Test cases.xls

Current progress

I have used Plunker (plnkr.co) to edit and run the suite.

I have opted to use Jasmine-Jquery as the test framework.

All files needed to run the suite are in the ‘Integration Test Suite’ folder.

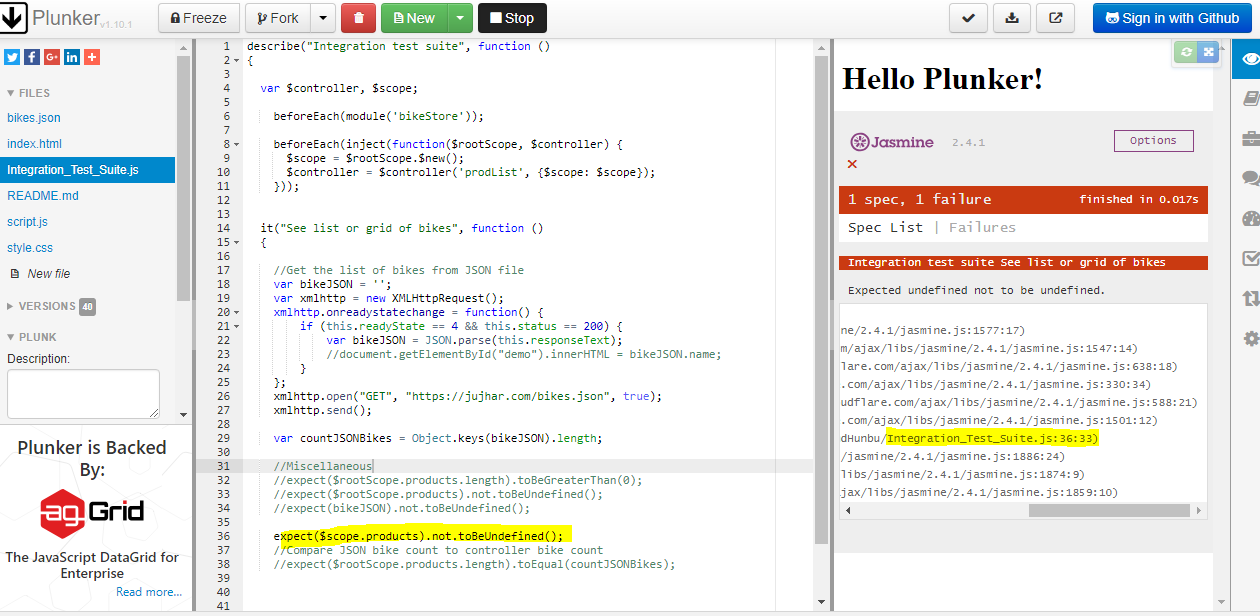
As of yet, I have not achieved a successful run of any tests. I have constructed the first test, the code for which can be found in ‘Integration\_Test\_Suite.js’.

The issue seems to be that when I run the test, the $rootScope properties are undefined.

So, even though I am setting up a controller, and the controller/$rootScope can be shown to exist, the properties (such as ‘Products’) remain undefined, and this is pretty fatal to my tests.

I cant overcome this issue. I am new to angular/jasmine, and I have searched for solutions but I cannot find anything wrong with what I am attempting to do.

Screenshot of error and offending line (although this line is a symptom rather than the cause).



Protractor

I have decided to abandon my attempt to conduct integration tests with Jasmine, as it appears that there is no way for me to run the app and get back the real JSON data. HTTPBackend mock cannot be replaced by a real call to the JSON data.

I have therefore turned to Protractor.

To use protractor:

Set up python HTTP Server as above.

In a new cmd:

Now start up a server with:

webdriver-manager start

This will start up a Selenium Server and will output a bunch of info logs. Your Protractor test will send requests to this server to control a local browser. Leave this server running throughout the tutorial. You can see information about the status of the server at http://localhost:4444/wd/hub

In another cmd:

Cd C:\Users\User\Documents\GitHub\test-testers\app\Protractor\_Integration\_Test

Protractor config.js

I have focused on getting a set of tests working, rather than concerning myself with optimising, factoring out the code, as I notice that refactoring code creates errors, which require a more in-depth understanding of how angular/node/protractor works, which is difficult within the allotted time.