Exercise (Instructions): End-to-End Testing Angular Applications

Objectives and Outcomes

In this exercise, you will learn about end-to-end (e2e) testing in your Angular applications. You will learn about Protractor and learn how to use them to carry out e2e testing. At the end of this exercise, you should be able to:

- Configure a protractor configuration file
- Set up e2e tests using Jasmine and carry out the e2e test automatically

Note: You should have completed the previous exercise on unit testing before proceeding with this exercise. Some of the set up is common for both unit tests and e2e tests. These were already done in the previous exercise.

Setting up the E2E Test Environment

- Set up the protractor tool globally for use in e2e testing:

```
npm install protractor –g
```

Remember to use sudo if you are in OSX or Linux environments.

- The above also installs webdriver-manager. Then, update your web drivers by typing:

```
webdriver–manager update
```

Remember to use sudo if you are in OSX or Linux environments.

- Make sure that you are running the json-server to serve up the REST API for accessing the JSON data by your Angular application.
- Next, you need to start a server to serve up the Angular application. Fortunately, Gulp is already set up to do that. Make sure you are in the *conFusion* folder. At the command prompt, type:

```
gulp watch
```

This will start the server and serve the page at http://localhost:3001/. Make sure about the port number (3001) where your server is serving up the web page.

Configuring Protractor

- Next, move to the *test* folder and create a file named *protractor.conf.js* and add the following configuration code to it:

```
exports.config = {
  allScriptsTimeout: 11000,
  specs: [
    'e2e/*.js'
  ],
  capabilities: {
    'browserName': 'chrome'
  },

  baseUrl: 'http://localhost:3001/',

  framework: 'jasmine',
    directConnect: true,

  jasmineNodeOpts: {
    defaultTimeoutInterval: 30000
  }
};
```

Here we are using the directConnect option to directly connect to the browser to conduct the tests, rather than through the Selenium web server.

- Next, create a folder named *e2e* in the *test* folder, and move to the *e2e* folder.
- Create a file named *scenarios.js*. This fill will contain all the e2e tests that we will execute on the application.
- Add the following code to the file:

```
'use strict';


describe('conFusion App E2E Testing', function() {


});
```

- Now we will introduce the tests into this file. First check to make sure that you are redirected to the index.html file:

```
  it('should automatically redirect to / when location hash/fragment is empty', function(
) {


    browser.get('index.html');
```

```
    expect(browser.getLocationAbsUrl()).toMatch("/");

  });
```

- Next, we will set up a simple test for the index file to check if the page title is set correctly:

```
  describe('index', function() {
    beforeEach(function() {
      browser.get('index.html#/');
    });

    it('should have a title', function() {
      expect(browser.getTitle()).
        toEqual('Ristorante Con Fusion');
    });
  });
```

- Next, we will navigate to the first menu item, and test a few properties there:

```
  describe('menu 0 item', function() {
    beforeEach(function() {
      browser.get('index.html#/menu/0');
    });

    it('should have a name', function() {
          var name = element(by.binding('dish.name'));
          expect(name.getText()).
             toEqual('Uthapizza Hot $4.99');
    });

    it('should show the number of comments as', function() {
         expect(element.all(by.repeater('comment in dish.comments'))
            .count()).toEqual(5);

    });

    it('should show the first comment author as', function() {
          element(by.model('orderText')).sendKeys('author');
```

```
            expect(element.all(by.repeater('comment in dish.comments'))
            .count()).toEqual(5);
          var author = element.all(by.repeater('comment in dish.comments'))
                      .first().element(by.binding('comment.author'));

          expect(author.getText()).toContain('25 Cent');

    });
});
```

- Save the changes and then move back to the *test* folder.
- At the command prompt, type the following to execute the e2e tests:

```
    protractor protractor.conf.js
```

- All the tests should pass successfully. You can modify some of the inputs to the text to see them fail.

Conclusions

In this exercise, you used Protractor together with Jasmine to carry out several e2e tests on your Angular application.