The information on [Protractor’s site](http://www.protractortest.org) is rather complete. The site not only displays how to setup, configure, to write and execute protractor tests it also provides a [style guide](http://www.protractortest.org/#/style-guide). Protractor is a [Node.js](https://nodejs.org/en/) program. To execute protractor, you will need to have Node.js installed. Protractor is built on top [WebDriverJS](https://github.com/SeleniumHQ/selenium/wiki/WebDriverJs). Java Development Kit (JDK) needs to be installed to run a local Selenium Server. Selenium WebDriver supports several browser implementations or drivers. The default behavior driven development (BDD) test framework in the Protractor install is [Jasmine](http://jasmine.github.io/). This repository uses the BDD framework Cucumber and assertion library Chai. Below is the version of Protractor and supporting software installed on my personal computer.

|  |  |
| --- | --- |
| Description | Command |
| Version of Node | $ node --version  v4.4.5 |
| Version of npm | $ npm --version  2.15.5 |
| Version of Protractor (also the version of Jasmine and Selenium WebDriver) | $ protractor --version  Version 3.3.0  $ npm ls -depth=0  protractor@3.3.0 /usr/local/lib/node\_modules/protractor  ├── adm-zip@0.4.7  ├── chalk@1.1.3  ├── glob@6.0.4  ├── jasmine@2.4.1  ├── jasminewd2@0.0.9  ├── optimist@0.6.1  ├── q@1.4.1  ├── request@2.67.0  ├── saucelabs@1.0.1  ├── selenium-webdriver@2.52.0  └── source-map-support@0.4.0 |
| Version of Java | $ java -version  java version "1.8.0\_45"  Java(TM) SE Runtime Environment (build 1.8.0\_45-b14)  Java HotSpot(TM) 64-Bit Server VM (build 25.45-b02, mixed mode) |
| Version of Protractor-Cucumber-Framework | Version 0.6.0 |
| Version of Chai | Version 3.5.0 |
| Version of Chai-As-Promised | Version 6.0.0 |

Protractor needs a test file and a configuration file to run. This [reference.js](https://github.com/angular/protractor/blob/master/docs/referenceConf.js) shows all of the configuration options that may be passed to Protractor. The configuration file tells Protractor where the test file(s) is located and where to talk to the Selenium Server. The test files use the syntax of your test framework, for example Chai, and the [Protractor API](http://www.protractortest.org/#/api). Below demonstrates how to run a Protractor test using PhantomJS headless browser on a specified server or the cloud testing solution Sauce Labs.

|  |  |  |
| --- | --- | --- |
| Test Type | Description | Command |
| Execute Protractor Test Using PhantomJS | 1. Start Selenium Server by either: | |
| a. Start Selenium WebDriver | webdriver-manager start |
| b. Or update the configuration file to reference the selenium standalone server jar | Reference to the selenium standalone server jar.  seleniumServerJar: './selenium\_server\_standalone/selenium-server-standalone-2.53.1.jar', |
|  | 2. Run Protractor Test | protractor conf.js |
| Execute Protractor Test Using Sauce Labs | Run Protractor Test | protractor confSauceLabsCucumber.js |

Below is the Protractor configuration, test file, and page object files used in the [AngularJSWebsiteTestByProtractorCucumberChai](https://github.com/gdombchik/AngularJSWebsiteTestByProtractorCucumberChai) GitHub repository.

|  |  |  |
| --- | --- | --- |
| Functionality | Description | JavaScript Files |
| Protractor Configuration File Using PhantomJS | Protractor configuration options used:   * seleniumAddress: “To connect to a Selenium Server which is already running.” * seleniumServerJar: “Reference to the selenium standalone server jar. * capabilities:{‘browser’:}: “Protractor will launch specified browser. In our example we are launching a headless browser PhantomJS.” * framework: “Framework to use. The framework in the protractor configuration file is specified as custom.” * frameworkPath: “Specifies the path to the Protractor Cucumber Framework.” * specs: “The Cucumber feature files.” * cucumberOpts: “Options to be passed to Cucumber including the step definitions (spec) file.” | conf.js |
| Protractor Configuration File Using Sauce Labs | Protractor configuration file using Sauce Labs options used:   * sauceUser: “The Sauce Labs user name.” * sauceKey: “The Sauce Labs access key.” * framework: “Framework to use. The framework in the protractor configuration file is specified as custom.” * frameworkPath: “Specifies the path to the Protractor Cucumber Framework.” * specs: “The Cucumber feature files.” * cucumberOpts: “Options to be passed to Cucumber including the step definitions (spec) file.” * multiCapabilities: “Specify the browser type, browser version, and operating system” | confSauceLabsCucumber.js |
| Page Object Design Pattern | A Protractor test file to test the AngularJS home page. The angularjsSpec.js test file calls the page objects located in angularjsWebsite/pageObjects folder.  Use the [Page Objects](http://www.protractortest.org/#/style-guide) design pattern.  Reasons to use the Page Objects design pattern as specified by the Protractor style guide:   * Encapsulate information about the elements on the page under test * They can be reused across multiple tests   Decouple the test logic from implementation details | *Cucumber Feature File:*  *angularjsWebsite.feature*  *Cucumber Definition File (spec):*  stepDefinitions.js  *Page Objects:*  homePage.js  downloadAngularJSOnePage.js  theBasics.js  addSomeControl.js  createComponents.js  wireUpABackend.js |

Cucumber uses feature files to specify use cases that describe a specific function of the software being tested. The feature files use the Gherkin language to define the test cases. The Gherkin syntax is designed to be non-technical and human readable in effort to promote business driven development practices across an entire development team. A feature file is separated into three parts: Features, Scenarios, and Steps. The Features describe specific function of the software being tested. Each feature is made of a collection of scenarios and each Scenario is defined by a sequence of Steps. Using java annotations, the Cucumber Steps associate with JUnit tests.

Below is the defined feature file for this application: angularjsWebsite.feature.

Feature: To test the AngularJS website home page.

Background:

Given I am on the AngularJS website home page.

@angularJSWebsite

Scenario: Test basic components on the AngularJS home page.

When I confirm I am on the AngularJS website home page.

| Field | Value |

| DownLoad Angular JS One Button | Download AngularJS 1\n\n (1.6.0-rc.1 / 1.5.8 / 1.2.32) |

Then I click on the Download AngularJS One button.

And I check the properties of the Download AngularJS One page.

| Field | Value |

| Title Label | Download AngularJS |

| Branch | 1.5.x (stable) |

| Build Minified | Minified |

| Build Zip | Zip |

| Build Uncompressed | Uncompressed |

| Cdn | angular.min.js |

| Bower | bower |

| Npm | npm |

| Extras | Browse additional modules |

| Previous Versions | Previous Versions |

| Download Button | angular.min.js |

| Close Button | × |

Then I click on the Close button of the Download AngularJS One page.

@angularJSWebsiteTestTheBasics

Scenario: Test The Basics.

When I fill in the name.

| Field | Value |

| Name | Greg |

Then I confirm the message.

| Field | Value |

| Name Message | Hello Greg! |

@angularJSWebsiteTestAddSomeControl

Scenario: Test Add Some Control.

When I confirm the labels of the current todo items.

| Field | Value |

| First Todo Checkbox | learn angular |

| Second Todo Checkbox | build an angular app |

Then I confirm the todo checkboxes that are selected.

| Field | Value |

| First Todo Checkbox | learn angular |

And I confirm the todo checkboxes that are not selected.

| Field | Value |

| Second Todo Checkbox | build an angular app |

Then I add a new todo item.

| Field | Value |

| New Todo List Item | Go to the dentist |

And I check the values of the todo items.

| Field | Value |

| First Todo Checkbox | learn angular |

| Second Todo Checkbox | build an angular app |

| New Todo List Item | Go to the dentist |

Then I select the check box of the new todo item.

| Field | Value |

| New Todo List Item | Go to the dentist |

And I recheck the value of the todo items.

| Field | Value |

| First Todo Checkbox | learn angular |

| New Todo List Item | Go to the dentist |

@angularJSWebsiteTestWireUpABackend

Scenario: Test Wire Up A Backend.

When I confirm the labels of the current JavaScript Projects.

| Field | Value |

| Angular 2 | Angular 2 |

| AngularJS | AngularJS |

| Backbone | Backbone |

| Cappucino | Cappucino |

| Ember | Ember |

| GWT | GWT |

| jQuery | jQuery |

| Knockout | Knockout |

| Polymer | Polymer |

| React | React |

| Spine | Spine |

| SproutCore | SproutCore |

Then I confirm the labels of the current JavaScript Project Descriptions.

| Field | Value |

| Angular 2 | One framework. Mobile and desktop. |

| AngularJS | HTML enhanced for web apps! |

| Backbone | Models for your apps. |

| Cappucino | Objective-J. |

| Ember | Ambitious web apps. |

| GWT | JS in Java. |

| jQuery | Write less, do more. |

| Knockout | MVVM pattern. |

| Polymer | Reusable components for the modern web. |

| React | A JavaScript library for building user interfaces. |

| Spine | Awesome MVC Apps. |

| SproutCore | A Framework for Innovative web-apps. |

Then Search for, update, and confirm a project values.

| Field | Value |

| GWT\_Current\_Name | GWT |

| GWT\_Current\_Website | http://www.gwtproject.org/ |

| GWT\_Current\_Description | JS in Java. |

| GWT\_Updated\_Name | GWT\_Updated |

| GWT\_Updated\_Website | http://www.gwtproject\_updated.org/ |

| GWT\_Updated\_Description | JS in Java.\_Updated |

@angularJSWebsiteTestCreateComponents

Scenario: Test Wire Create Components.

When I confirm the locales.

| Field | Value |

| United States | US |

| Slovakia | SK |

Then I confirm the localization values for United States.

| Field | Value |

| US\_Date | Date: Sunday, April 1, 2012 |

| US\_Currency | Currency: $123,456.00 |

| US\_Number | Number: 98,765.432 |

And I confirm the pluralization values for United States.

| Field | Value |

| no beers | no beers |

| 1 beer | 1 beer |

| 2 beer | 2 beer |

| 3 beer | 3 beer |

| 4 beer | 4 beer |

| 5 beer | 5 beer |

| 6 beer | 6 beer |

| 7 beer | 7 beer |

Then I confirm the localization values for Slovakia.

| Field | Value |

| SK\_Date | Date: nedeľa, 1. apríla 2012 |

| SK\_Currency | Currency: 123 456,00 € |

| SK\_Number | Number: 98 765,432 |

And I confirm the pluralization values for Slovakia.

| Field | Value |

| žiadne pivo | žiadne pivo |

| 1 pivo | 1 pivo |

| 2 pivá | 2 pivá |

| 3 pivá | 3 pivá |

| 4 pivá | 4 pivá |

| 5 pív | 5 pív |

| 6 pív | 6 pív |

| 7 beer | 7 beer |

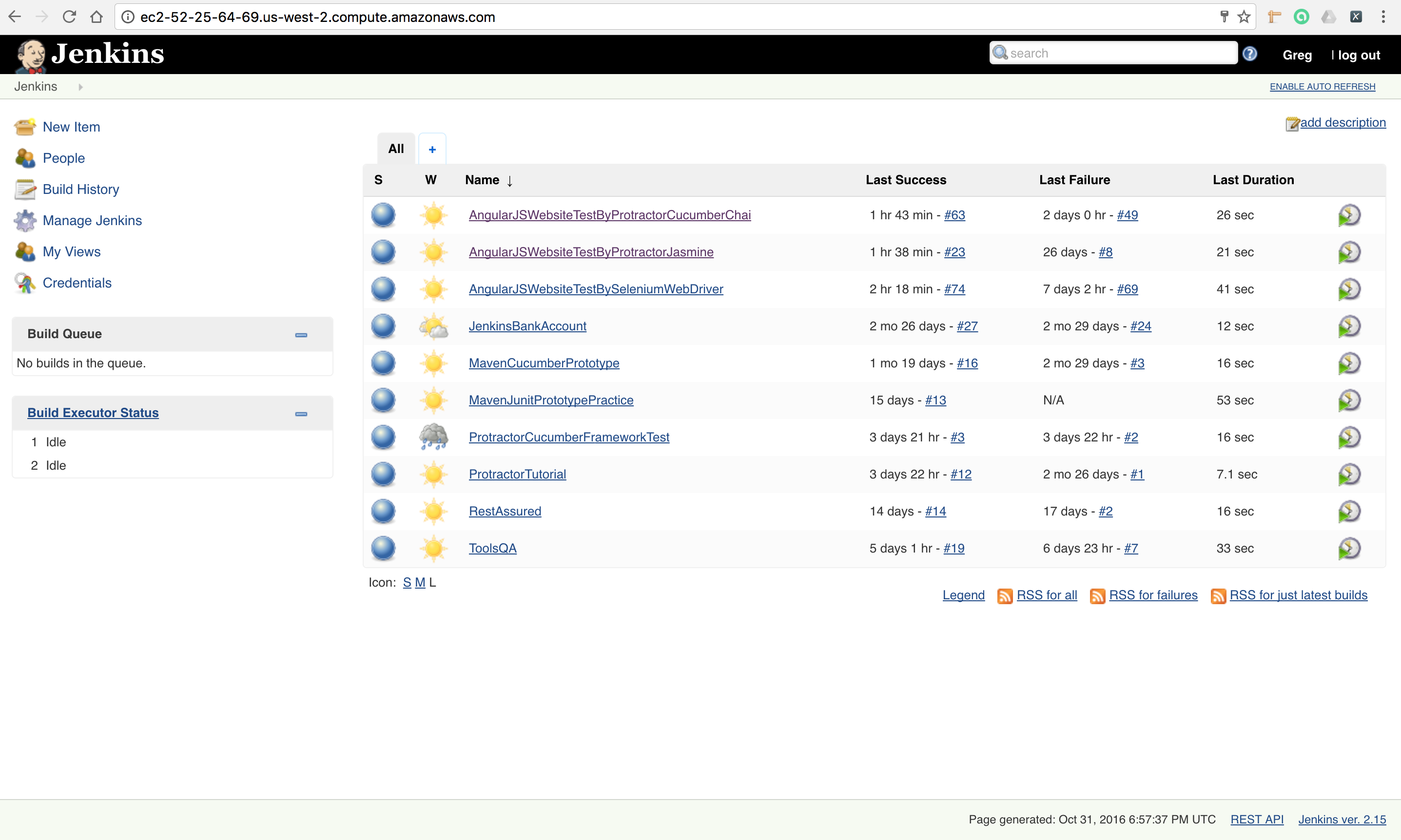
In reviewing how to run the Protractor tests using a continuous integration (ci) tool such as Jenkins, an option is to use the standalone Selenium Server jar start the selenium server, use Jenkins “execute shell” option to execute the Protractor configuration file, run the tests using a headless environment using tools such as PhantomJS, and capture test results.

The steps below are to be executed on the server where Jenkins is hosted. This example Jenkins has been installed on an AWS E2 Ubuntu server.

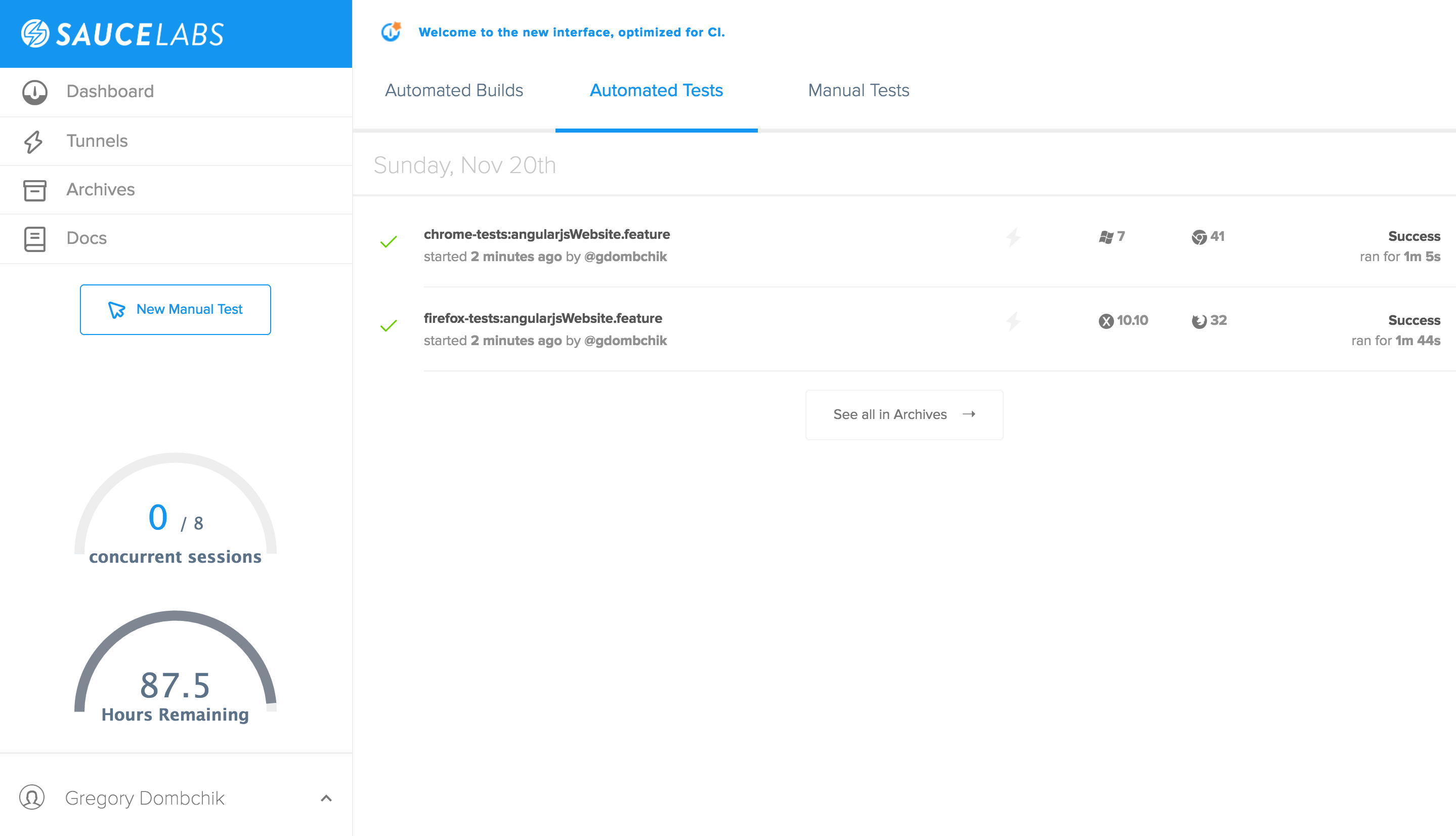
|  |  |
| --- | --- |
| Description | Command |
| Installation of nodejs | $ curl -sL https://deb.nodesource.com/setup\_4.x | sudo -E bash -  $ sudo apt-get install -y nodejs  $ npm –version  2.15.8 |
| Global Installation of Protractor and webdriver-manager | $ sudo npm install -g protractor  $ protractor --version  Version 4.0.2  $ sudo webdriver-manager update |
| Global Installation of PhantomJS | $ sudo npm install -g phantomjs-prebuilt  $ phantomjs --version  2.1.1 |
| Update the configuration file to reference PhantomJS | Update conf.js and change the browserName to phantomjs.  capabilities: {  'browserName': 'phantomjs' }, |
| Update the configuration file to reference the selenium standalone server | Reference selenium server standalone jar.  seleniumServerJar: './selenium\_server\_standalone/selenium-server-standalone-2.53.1.jar', |
| Global installation of Cucumber | $ sudo npm install –g cucumber |
| Installation of Protractor-Cucumber-Framework | $ Local Machine: sudo npm install --save-dev protractor-cucumber-framework  $ Server: sudo npm install -g protractor-cucumber-framework |
| Installation of Chai and Chai-As-Promised | $ Local Machine: sudo npm install chai chai-as-promised  $ Server: sudo npm install –g chai chai-as-promised |

The Jenkins setup of a Protractor project in GitHub. Does not include the installation and setup of Jenkins on AWS E2 Ubuntu server.

|  |  |
| --- | --- |
| Description | Command |
| *AWS E2 Ubuntu server* | |
| Switch to the Jenkins user. | $ sudo su – Jenkins |
| Create public private key pair. | $ cd .ssh  $ ssh-keygen -t dsa |
| Cat the public key and copy the results. | $ cat AngularJSWebsiteTestByProtractorCucumberChai.pub |
| *GitHub* | |
| Add and paste deploy key in GitHub project. | <https://github.com/gdombchik/AngularJSWebsiteTestByProtractorCucumberChai/settings/keys> |
| *Manage Jenkins – Configure System* | |
| Jenkins Location | Scroll to the “Jenkins Location” section.  Jenkins URL:  <http://ec2-52-25-64-69.us-west-2.compute.amazonaws.com/>  System Admin e-mail address:  [greg@gregorydombchik.com](mailto:greg@gregorydombchik.com) |
| E-mail Notification | Scroll to the “E-mail Notification” section.  SMTP server:  mail.gregorydombchik.com |
| *Jenkins Server – New Project* | |
| Add A Jenkins Project for Protractor project. | Select New Item.  Enter an item name.  Select Freestyle project.  Select Ok. |
| Specify the URL of the remote GitHub repository. | Scroll to the “Source Code Management” section.  Select the “Git” option.  Enter Repository URL:  <https://github.com/gdombchik/AngularJSWebsiteTestByProtractorCucumberChai.git> |
| Update the Poll SCM. | Scroll to the “Build Triggers” section.  Select the “Poll SCM” option.  Enter the following in the “Schedule” text box:  H \*/3 \* \* \* |
| Update the Build. | Scroll to the “Build” section.  Select from the “Add build step” and select the “Execute shell” option.  Enter the following in the “Command” text box:  protractor confSauceLabsCucumber.js |
| Update E-mail Notification. | Scroll to the Post-build Actions.  Select from the “Add post-build action” and select the “E-mail Notification” option.  Enter email address in the “Recipients” text box:  [greg@gregorydombchik.com](mailto:greg@gregorydombchik.com) |
| *GitHub* | |
| Add Jenkins GitHub Plugin you can automatically trigger build jobs when  pushes are made to GitHub. | <https://github.com/gdombchik/AngularJSWebsiteTestByProtractorCucumberChai/settings/installations>  Select from the “Add service” and select the “Jenkin’s (Git plugin)” option. (NOT GITHUB PLUGIN. This service does not appear to work).  Enter the following in the “Jenkins url” text box:  <http://ec2-52-25-64-69.us-west-2.compute.amazonaws.com/> |



An alternative to executing the unit tests on a server using the PhantomJS is to use a cloud testing solution such as Sauce Labs. Sauce Labs provides the ability to run the tests on hundreds different browser types, browser version, and operating system combinations. The Sauce Labs account credentials and specified browser/operating system combination(s) are specified in the protractor configuration file (confSauceLabsCucumber.js). When the Jenkins server runs the Sauce Labs specified tests, the tests will be executed on the Sauce Labs cloud testing environment. A Sauce Labs test result is displayed in both the Jenkins and Sauce Labs dashboard.



In summation, the automated testing of a website can be built by creating behavior-driven development Cucumber feature files and associated definition files that reference JavaScript page objects representing the web application using the Protractor and Chai test framework and a headless environment such as PhantomJS Driver or using a cloud testing solution such as Sauce Labs, and then porting the solution into a continuous integration environment (Jenkins).