**ASSUMPTIONS:**

Framework – Protractor

Language – JavaScript

Assertions – Jasmine

Browser – Chrome, configuration has setup for firefox

**FRAMEWORK -**

**PROTRACTOR :**

Protractor, formally known as E2E(End To End) testing framework, is an open source functional automation framework designed specifically for AngularJS web applications.

**Protractor** is a **Node.js** program built on **top** of WebDriverJS

Protractor is a wrapper (built on the top) around Selenium WebDriver, so it contains every feature that is available in the Selenium WebDriver

Protractor works with both Angular and non-Angular JS applications equally well.

As AngularJs (Extends regular HTML elements) getting popular, in future we no need to switch to new framework.

**Salient features of the Protractor Automation tool:**

1. Built on the top of WebdriverJS and Selenium server
2. Introduced new simple syntax to write tests
3. Allows running tests targeting remote addresses
4. Can take
5. advantage of Selenium grid to run multiple browsers at once
6. Can use Jasmine or Mocha to write test suites

**Protractor vs Selenium WebDriver**

**What makes Protractor different from traditional**[**Selenium WebDriver**](https://www.softwaretestinghelp.com/selenium-webdriver-selenium-tutorial-8/)**?**

Take a minute to answer these questions:

* Is it a struggle to determine when exactly the web page is finally loaded (All of the asynchronous elements are ready and processed)?
* Are you tired of adding waits and sleeps to your code?
* Do you want to get rid of the cumbersome effort to locate the angular elements?
* Frustrated with locating elements with changing Ids?
* Do you want to create your own locator?
* Are you writing lengthy code even for the simplest assertions?
* Are you a JavaScript enthusiast/fan?

*If you answered Yes to these questions,*[*Protractor*](https://www.protractortest.org/#/)*can help.*

JASMINE Framework:

Jasmine is a behavior-driven development framework for testing JavaScript code. It’s a Testing framework.

Jasmine which have assertions i.e.“expect” that can resolve promises internally

Jasmine also provides Setup and teardown Annotations. Some of them are

BeforeAll, BeforeEach, AfterAll,AfterEach etc

## Expectations

Expectations are built with the function expect which takes a value, called the actual. It is chained with a Matcher function, which takes the expected value.

### Matchers

Each matcher implements a boolean comparison between the actual value and the expected value. It is responsible for reporting to Jasmine if the expectation is true or false. Jasmine will then pass or fail the spec.

Read more - https://jasmine.github.io/2.0/introduction.html