**Serenity-bdd**

**Introduction**

Serenity BDD is an open source library that aims to make the idea of living documentation a reality.

Serenity BDD helps you write cleaner and more maintainable automated acceptance and regression tests faster. Serenity also uses the test results to produce illustrated, narrative reports that document and describe what your application does and how it works. Serenity tells you not only what tests have been executed, but more importantly, what requirements have been tested.

One key advantage of using Serenity BDD is that you do not have to invest time in building and maintaining your own automation framework.

Serenity BDD provides strong support for different types of automated acceptance testing, including:

* Rich built-in support for web testing with Selenium.
* REST API testing with RestAssured.
* Highly readable, maintainable and scalable automated testing with the Screenplay pattern.
* It uses selenium 2.0 which contains all new features of selenium.

**Some Features of serenity are :-**

1. Serenity have got beautiful report with graphs.
2. Serenity also provide serenity.property file which is used to pass all the configurations.
3. Serenity pageObject class contains different type of conditional waits.

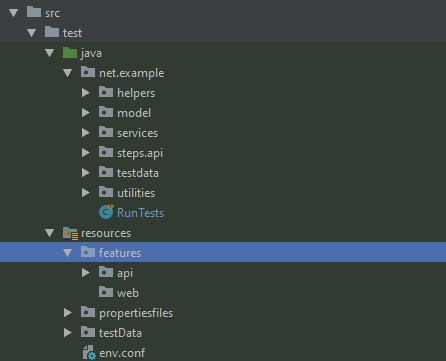
**Technology Which we use :-**

**We use these technology stack for this frameWork :-**

1. **Maven** as project management tool.
2. **Cucumber** for scenario writing.
3. **RestAssured** for api testing.
4. **Serenity** For pageObject class and there predefined function.
5. **Report ,**we use serenity it provide feature to generate report.

**Implementation**

**We implement framework using this structure :-**



**In Java folder we store only single type of files :-**

1. **Java file(.java)** use to write scripts

**We create different packages in java folder :-**

1. **helpers :-** contains **ApiHelper** which is use to return base URI’s with headers so that we further concatenate with end point and **LoadProperties** contains function use to load property file.
2. **model :-** this package contain all the response and request model which we use in api testing.
3. **services :-** This package contains all the endpoints and concatenate with base url
4. **steps/step\_defenition :-** This package contain all the step\_defenation which we use in cucumber files(.feature)
5. **pages :-** This package contains all the pageObjects and their related function, because we follow **Page Object Model** in our project.
6. **utilities :-** This package contains all the classes and function which is common in all over the project.

**In Resources folder we store multiple files like :-**

1. **cucumber file**(.feature) in feature folder.
2. **excel and csv** files in test data folder which uses as input data.
3. **Property files** (.property) in propertyFiles folder.

## **What is Cucumber?**

Cucumber is a tool that supports Behavior Driven Development (BDD). It offers a way to write tests that anybody can understand, regardless of their technical knowledge.

In BDD, users (business analysts, product owners) first write scenarios or acceptance tests that describes the behavior of the system from the customer's perspective, for review and sign-off by the product owners before developers write their codes.

Cucumber use Ruby programming language.

## **Advantages of Cucumber**

1. It is helpful to involve business stakeholders who can't easily read code
2. Cucumber Testing focuses on end-user experience
3. Style of writing tests allow for easier reuse of code in the tests
4. Quick and easy set up and execution
5. Efficient tool for testing

**Properties Files**

**.properties** is a file extension for files mainly used in Java related technologies to store the configurable application. They can also be used for storing strings for Internationalization and localization; these are known as Property Resource Bundles.

Each parameter is stored as a pair of strings, one storing the name of the parameter (called the key), and the other storing the value.

## 

## **Reporting and Living Documentation**

Reporting is one of Serenity’s fortes. Serenity not only reports on whether a test passes or fails, but documents what it did, in a step-by-step narrative format that includes test data and screenshots for web tests. For example, the following page illustrates the test results for our first acceptance criteria:

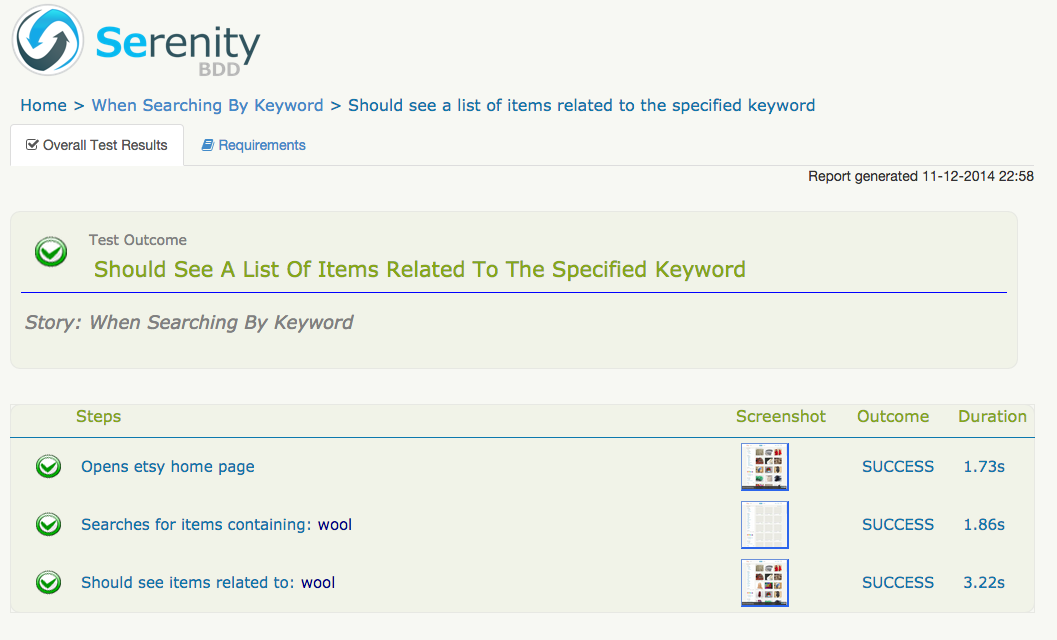


Figure 1. Test results reported in Serenity

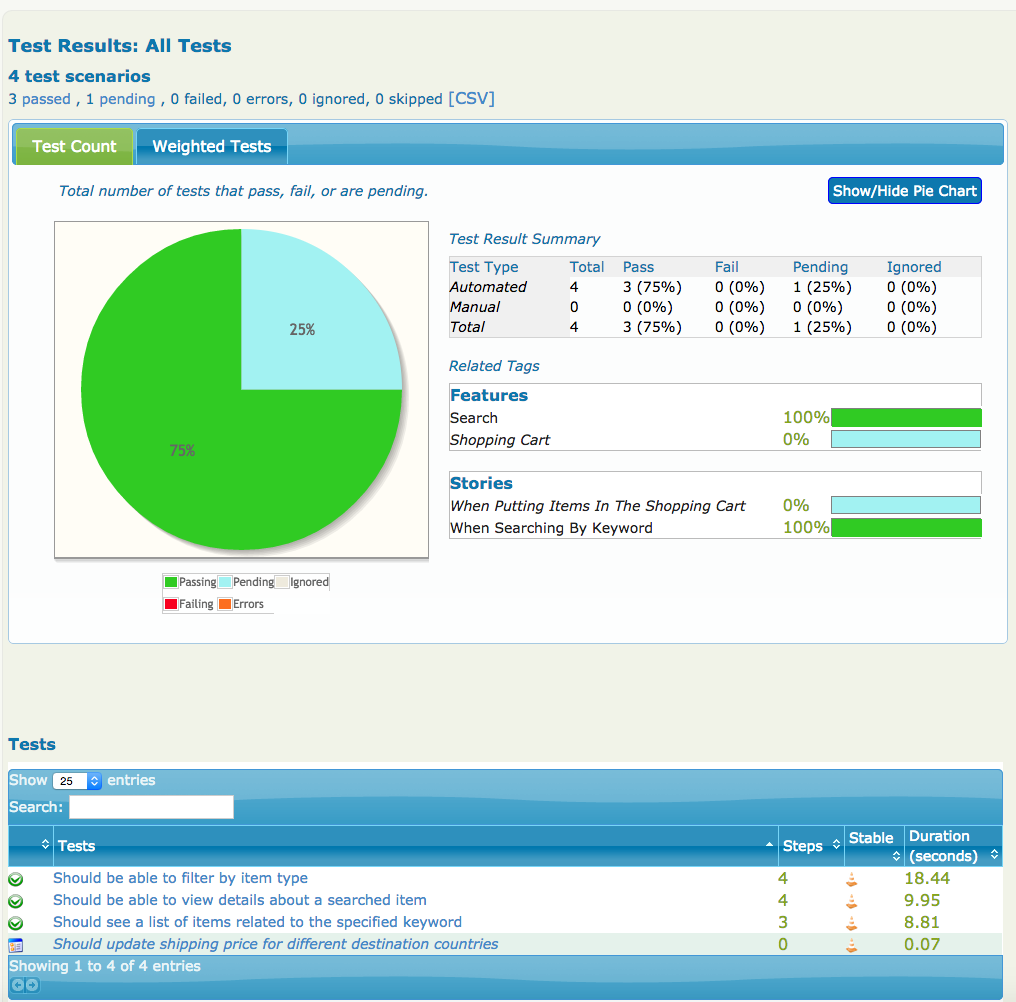
But test outcomes are only part of the picture. It is also important to know what work has been done, and what is work in progress. Serenity provides the @Pending annotation, that lets you indicate that a scenario is not yet completed, but has been scheduled for work, as illustrated here:

@RunWith(CucumberWithSerenity.class)

@CucumberOptions(features = "src/test/resources/features/api/example")

public class RunTests {}

**This test will appear in the reports as *Pending* (blue in the graphs):-**

****

**How to Run and Generate Report?**

**mvn clean** is use to clean previous report

**mvn clean verify** is use to clean previous report and run all scenarios and create new report.

**mvn clean verify "-env=dev"** to pass environment type so that test cases run on particular environment.