Sri Lanka Institute of Information Technology

Assignment Report

Software Engineering Process and Quality Management

Submitted by:

IT15146366 - H. A. I. S. Hettiarachchi
IT16124936 - P. G. R. S. H. Gamlath
IT16160262 - M. S. B. W. T. M. P. S. B. Thennakoon
IT16119086 - W. A. Geeth Sameera

Static Analysis Tool

Static analysis tools are generally used by developers as part of the development and component testing process. The key aspect is that the code (or other artefact) is not executed or run but the tool itself is executed, and the source code we are interested in is the input data to the tool[1].

JSHint

JSHint is a static code analysis tool for JavaScript. It is a community driven - open source tool that can be run online or using a command line. JSHint was created in 2011 as a fork of JSLint[2] project by Anton Kovalyov. Tool is written JavaScript to find common code errors like syntax errors, a bug due to implicit type conversion, leaking variables, etc. Tool will also helps to find potential bugs that would come in the compile time.

Pros and Cons

Pros

- Has both online and command line versions
- Most of the settings can be configured
- Can configure every rule
- Has a good documentation for each of the rule
- Basic ES6 support
- Support many libraries out of the box like jQuery, QUnit, NodeJS, Mocha, etc.

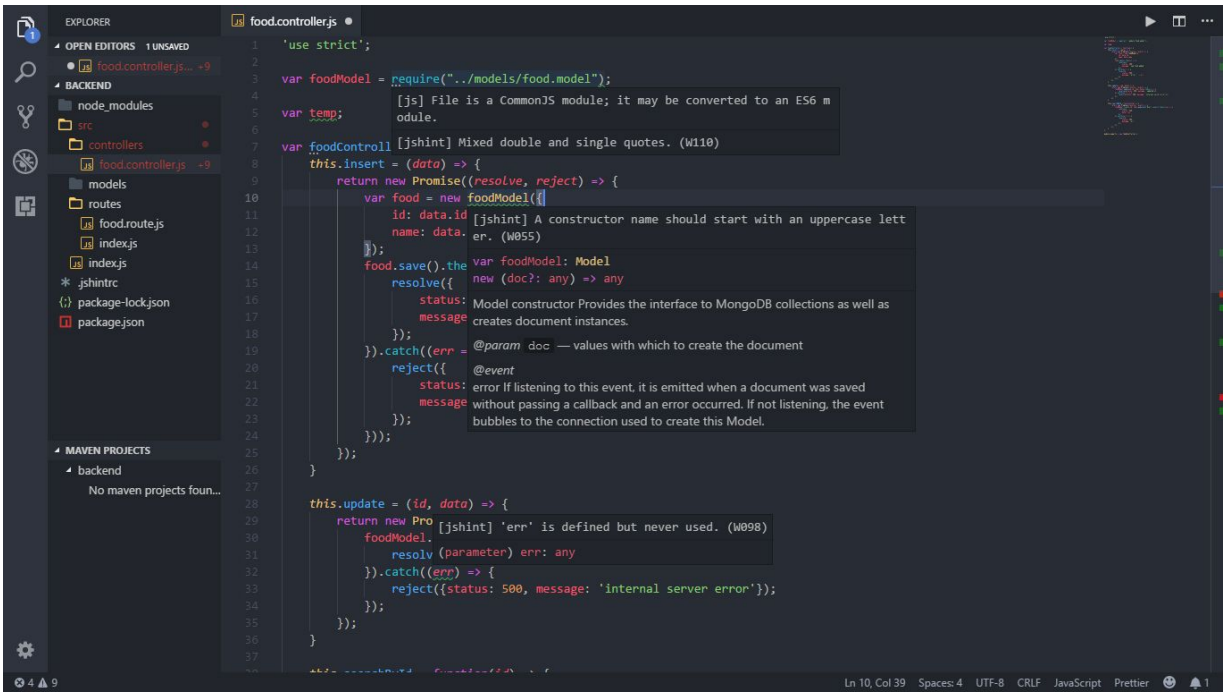
Cons

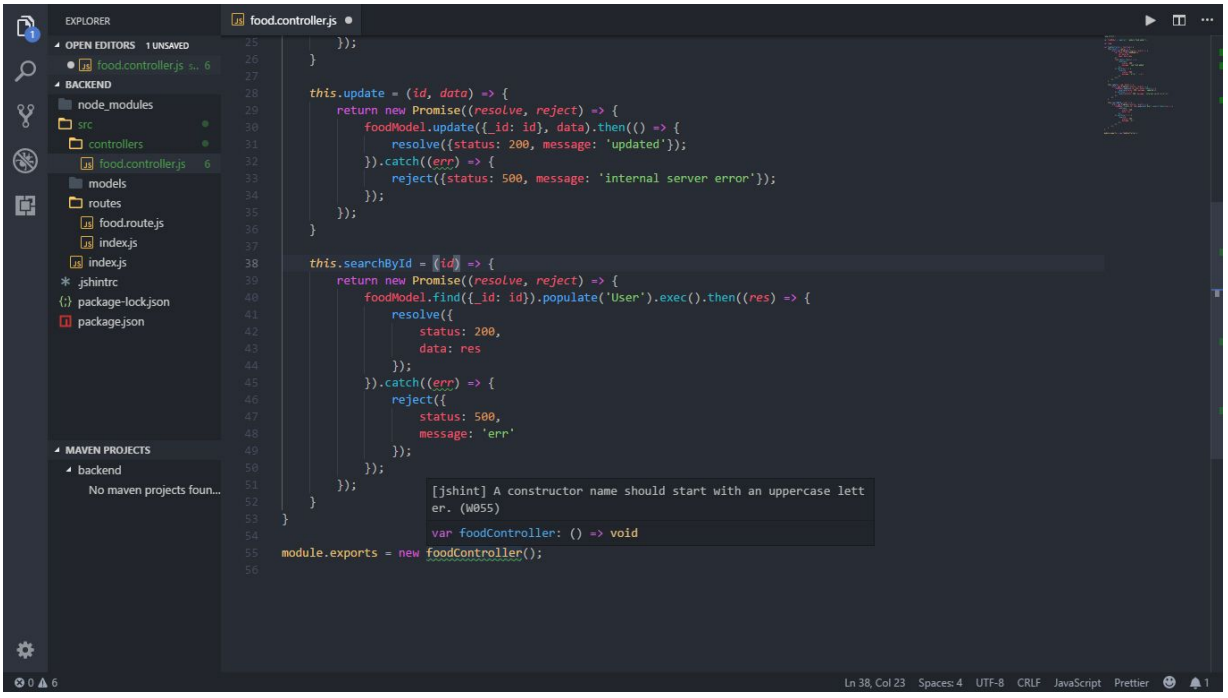
- Difficult to know which rule is causing the error
- No custom rule support
- Slightly confusing to configure the tool

Use case scenario of JSHint

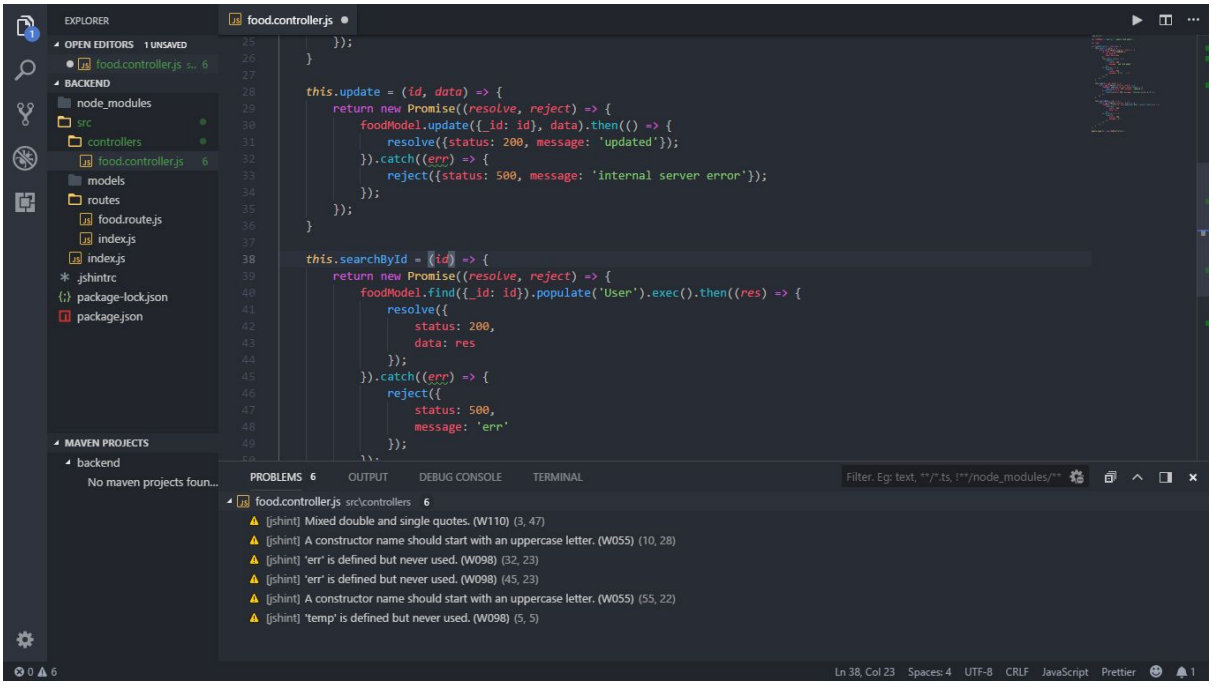
IDE Integration and command line

JSHint is installed as a Visual Studio Code extension and installed as a dev dependency to JavaScript Express JS project. JSHint configuration file has been added and all the rules are configured as true to check in JSHint. There are intentional syntax errors done to the code for this demonstration.



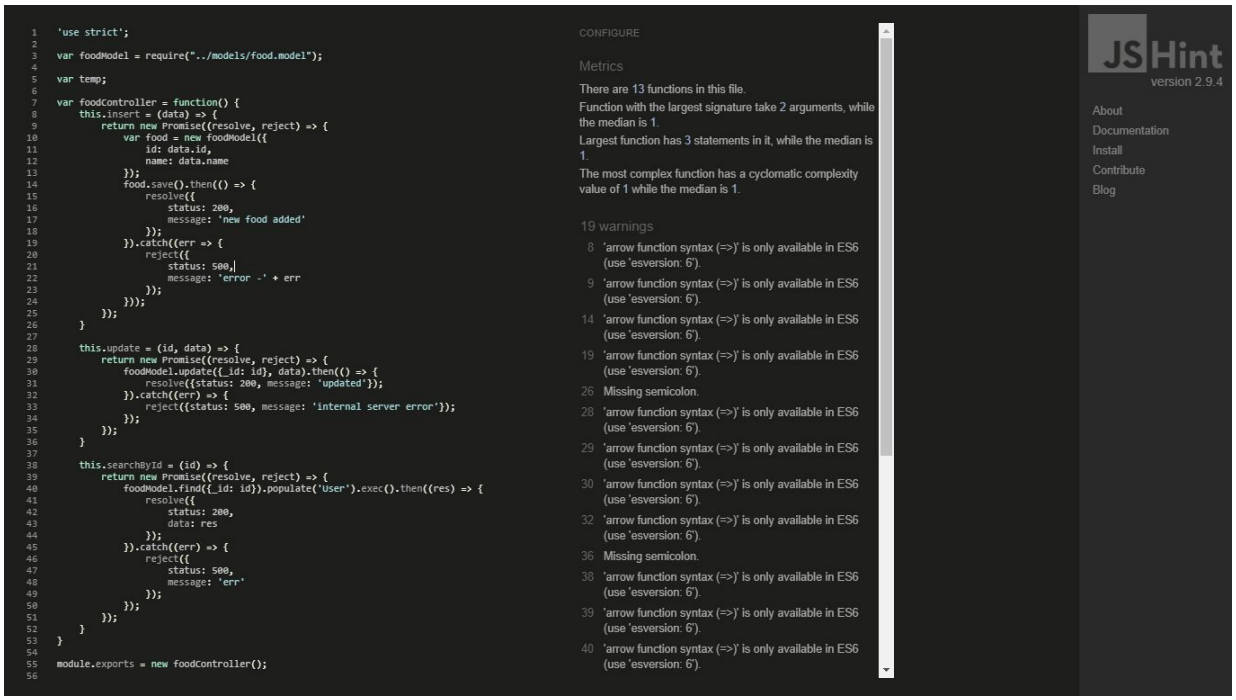


JSHint errors are underlined and the editor. Moving the mouse pointer to those underlined codes will show a description of the error and possible fixes



In here all the JSHint errors are show in the VSCode problems tab. Clicking on each problem will take the focus to the error line in the editor.

JSHint Online



```
'use strict';

var foodModel = require("../models/food.model");

var temp;

var foodController = function() {
    this.insert = (data) => {
        return new Promise((resolve, reject) => {
            var food = new foodModel({
                id: data.id,
                name: data.name
            });
            food.save().then(() => {
                resolve({
                    status: 200,
                    message: 'new food added'
                });
            }).catch((err => {
                reject({
                    status: 500,
                    message: 'error -' + err
                });
            }));
        });
    }

    this.update = (id, data) => {
        return new Promise((resolve, reject) => {
            foodModel.update({_id: id}, data).then(() => {
                resolve({status: 200, message: 'updated'});
            }).catch((err) => {
                reject({status: 500, message: 'internal server error'});
            });
        });
    }

    this.searchById = (id) => {
        return new Promise((resolve, reject) => {
            foodModel.find({_id: id}).populate('User').exec().then((res) => {
                resolve({
                    status: 200,
                    data: res
                });
            }).catch((err) => {
                reject({
                    status: 500,
                    message: 'err'
                });
            });
        });
    }
}

module.exports = new foodController();
```

CONFIGURE

Metrics

There are 13 functions in this file.
Function with the largest signature take 2 arguments, while the median is 1.
Largest function has 3 statements in it, while the median is 1.
The most complex function has a cyclomatic complexity value of 1 while the median is 1.

19 warnings

8 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').
9 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').
14 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').
19 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').
26 Missing semicolon.
28 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').
29 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').
30 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').
32 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').
36 Missing semicolon.
38 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').
39 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').
40 'arrow function syntax (=>)' is only available in ES6 (use 'esversion: 6').

JSHint version 2.9.4

About
Documentation
Install
Contribute
Blog

Same JavaScript is copied to the JSHint online checker. It gives an overview of the code under metrics, warnings and other problems separately in right side panel.

Bug/Issue tracking tool

Mantis BT

MantisBT is an open source bug/issue tracker that provides a delicate balance between simplicity and power. Users are able to get started in minutes and start managing their projects while collaborating with their teammates and clients effectively.

MantisBT initially released in 2000. Mantis is known to be one of the oldest tool. It's written in PHP and available in 49 different languages. MantisBT is a widely used bug tracking tool.

Mantis got its name from the Mantidae family of insects, colloquially referred to as bugs. That's the reason why they use bug as their logo.

With the release of Mantis BT version 1.2.0, an event-driven plugin system was introduced. As being one of the elder kids in town, I personally found Mantis BT quite old school. Also if you compare it to other issue tracking tools.

Pros and Cons

Pros

- It sends out emails of updates, comments, resolutions to related peoples
- Administrator can control the user access at project level
- Users can easily customize Mantis as per their requirements.
- Mantis supports iPhone, Android and Windows Phone Platforms.
- An ever-expanding library of plugins are there to add custom functionalities to Mantis.
- MantisBT provides users with several collaboration, notification, reporting and management features that help streamline and simplify the bug tracking process.[14]
- Reduce deployment delays
- Helps to prevent downtime once the application is launched
- The MantisBT allows users to upload and attach images or other files with bug reports to provide additional information to the debugging initiative[14]
- The issue fields and workflows for individual bugs are also customizable depending on the needs of the particular debugging activities[14]

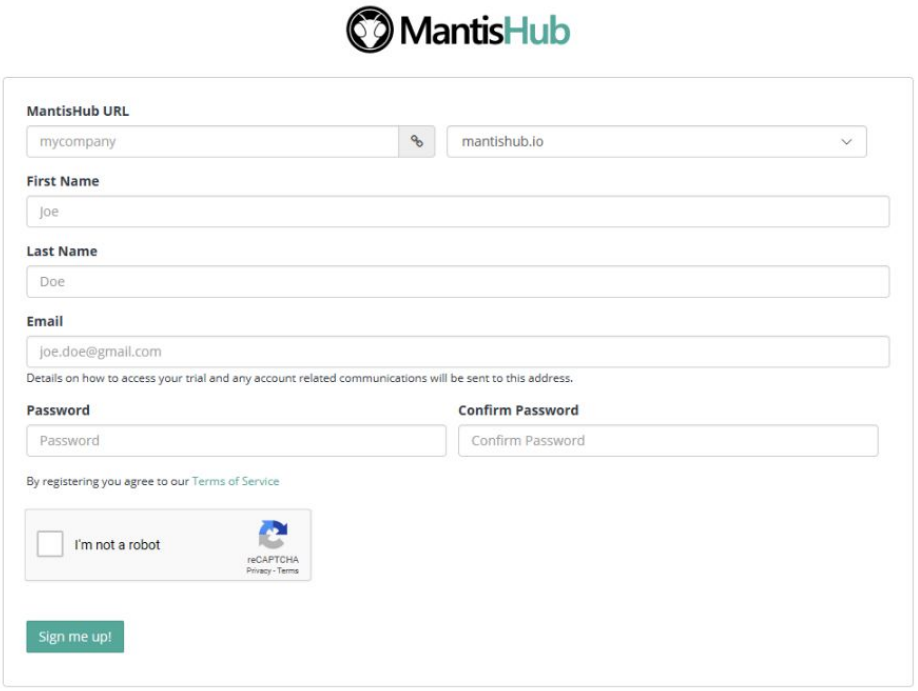
Cons

- MantisBT does not provide any tools for diagnosing or debugging developed applications or software
- Installation requires XAMPP or WAMP server installed.
- Data is stored in a relational database. MantisBT supports MySQL. Support for other databases is known to be problematic.

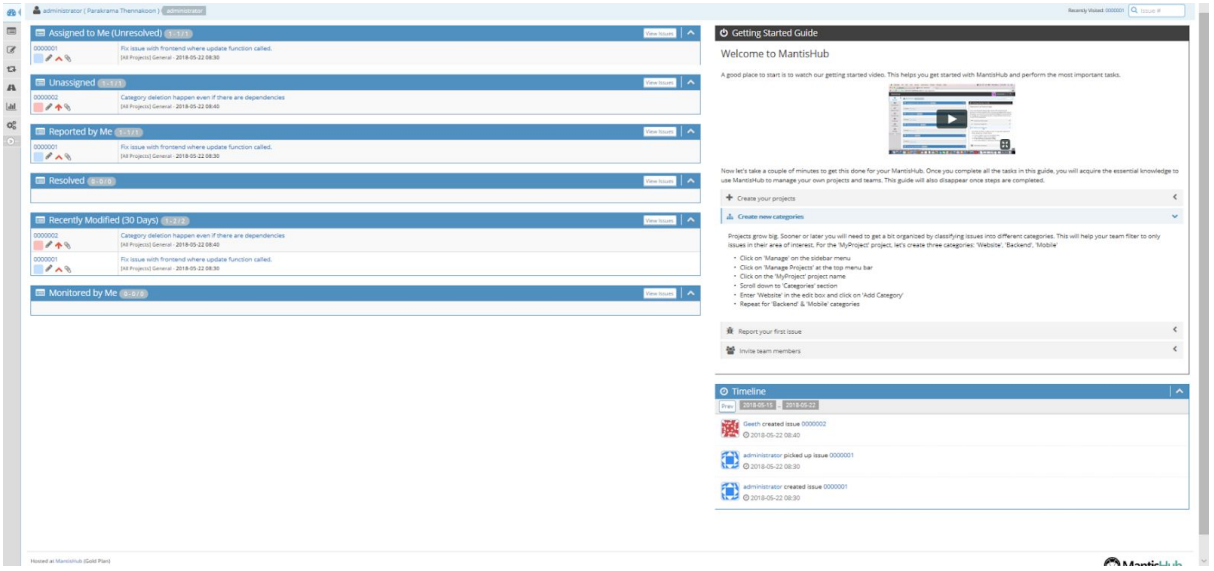
Use case scenario of Mantis

Web based API

Mantis can be run on browser in both LAN server and internet. Registration takes less than 1 minute for trial version. Provided images shows its interface with added issues for certain project.

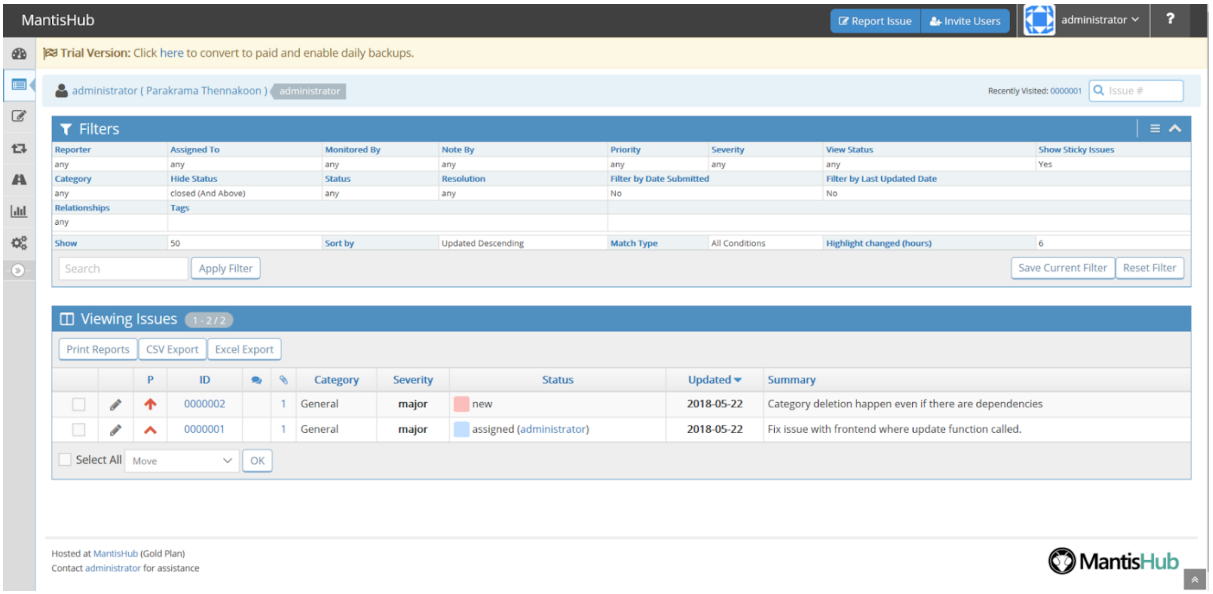


Registration process interface where the user will create his own domain network. Administrator role can add other users to his domain.

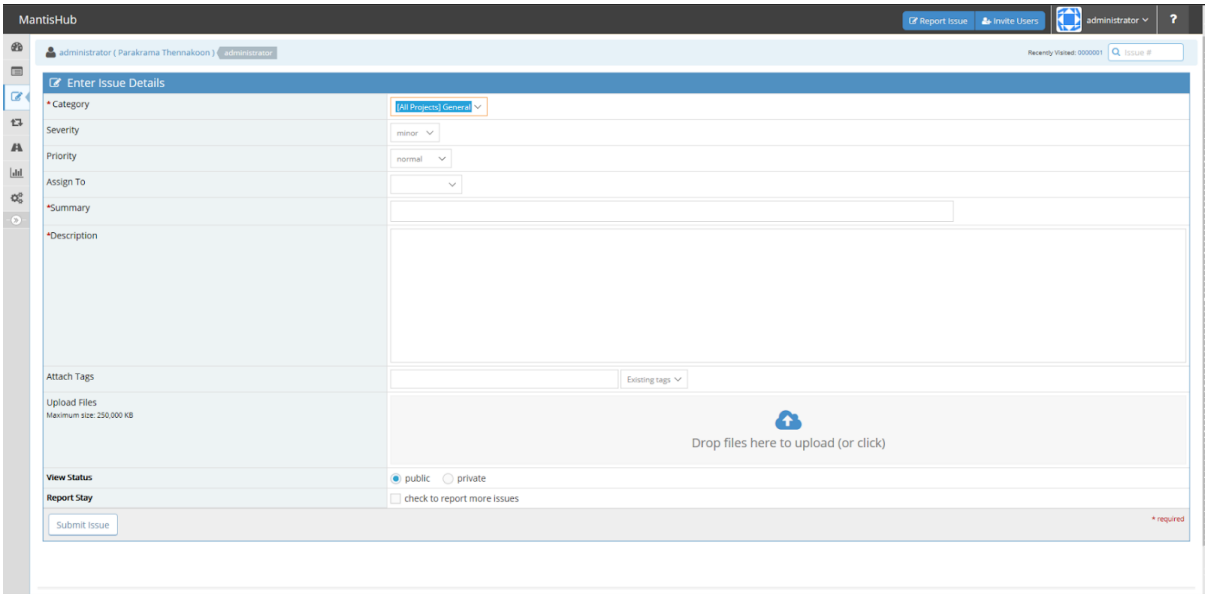


View of that user currently logged in shown above. Here notification of issues, timeline of issues and hints are shown. User will see this interface , the first time logged in(without these added issues.)

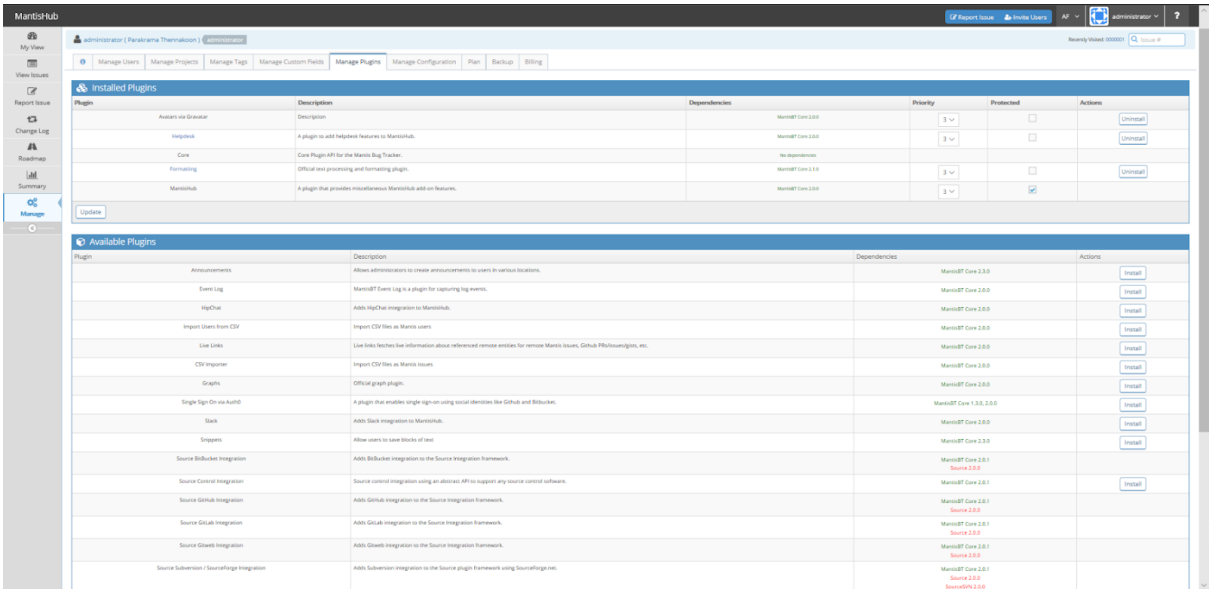
Interfaces for Bug/Issue view, assignment , manage(plugins that can be integrated) are show in brelow images.



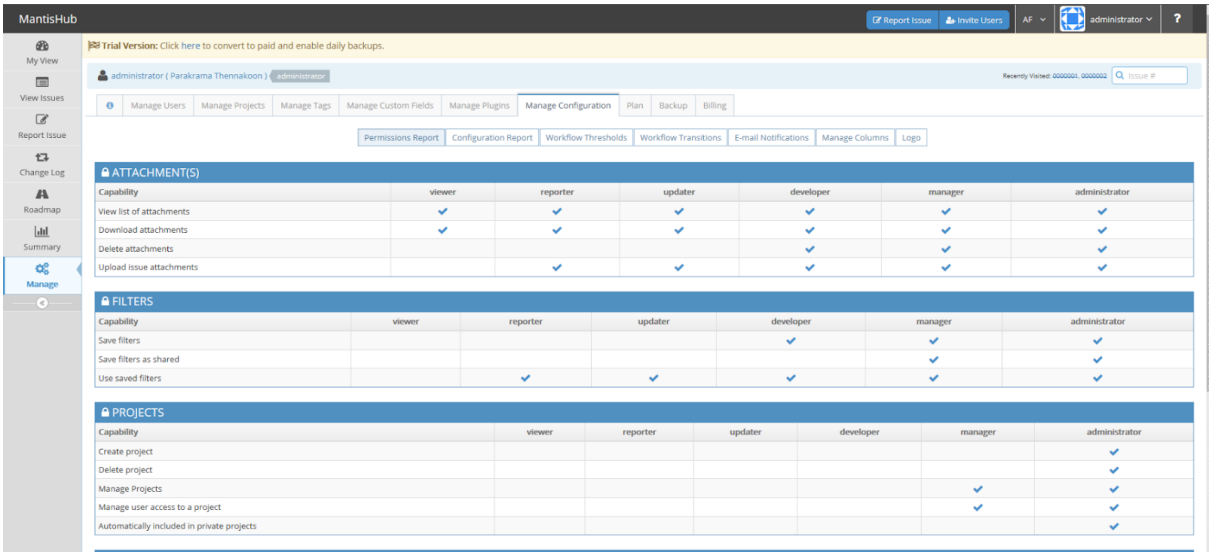
Issues assigning interface shown below. This is where user story issues filtered and assigned



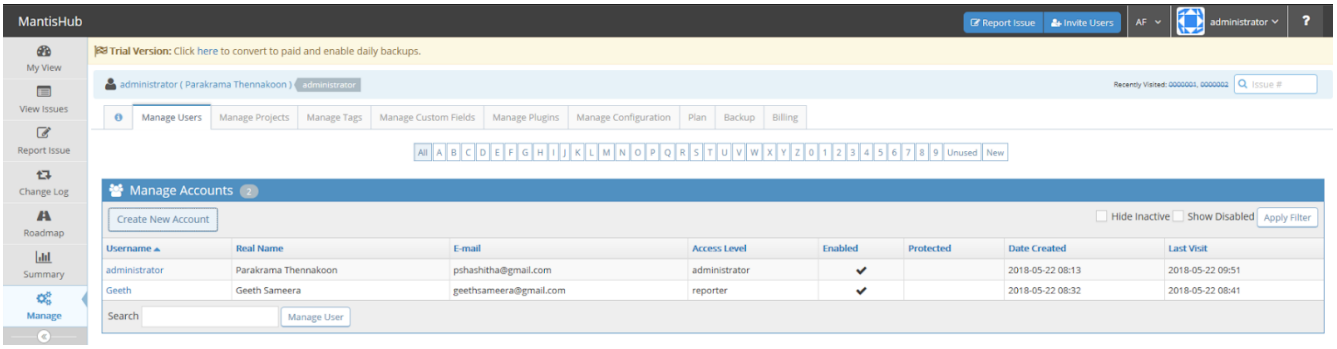
Below is where the user can manage projects,configurations, plugins etc.



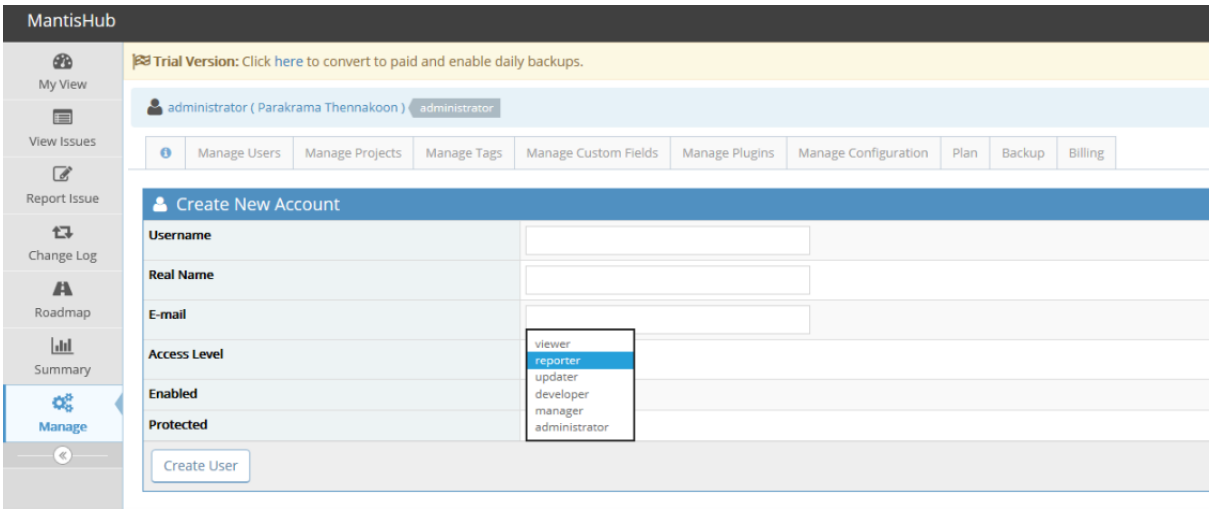
Here shown the capabilities for each registered user in this network of administrator.



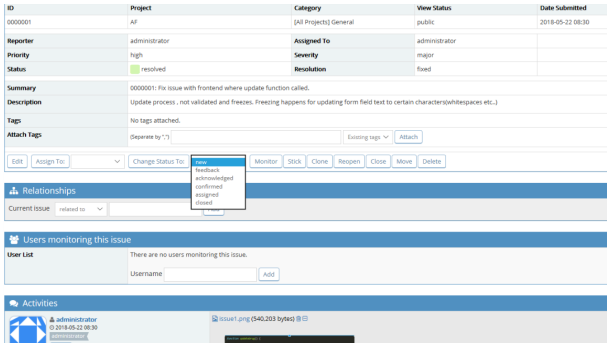
Administrator has capability to register above roles. New user will be shown in Manage Accounts/Users view.

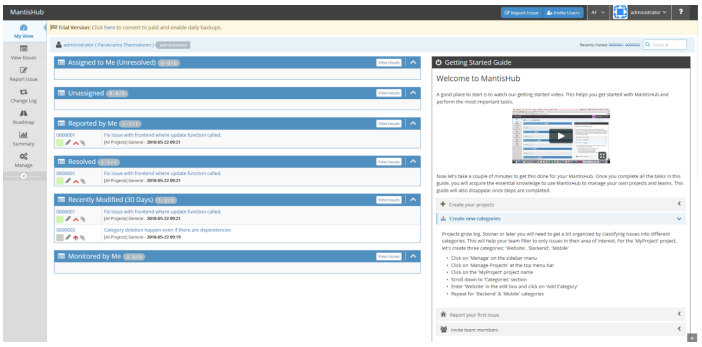


After clicking Create new Account button, administrator can add another role. Creation of new user is shown below.

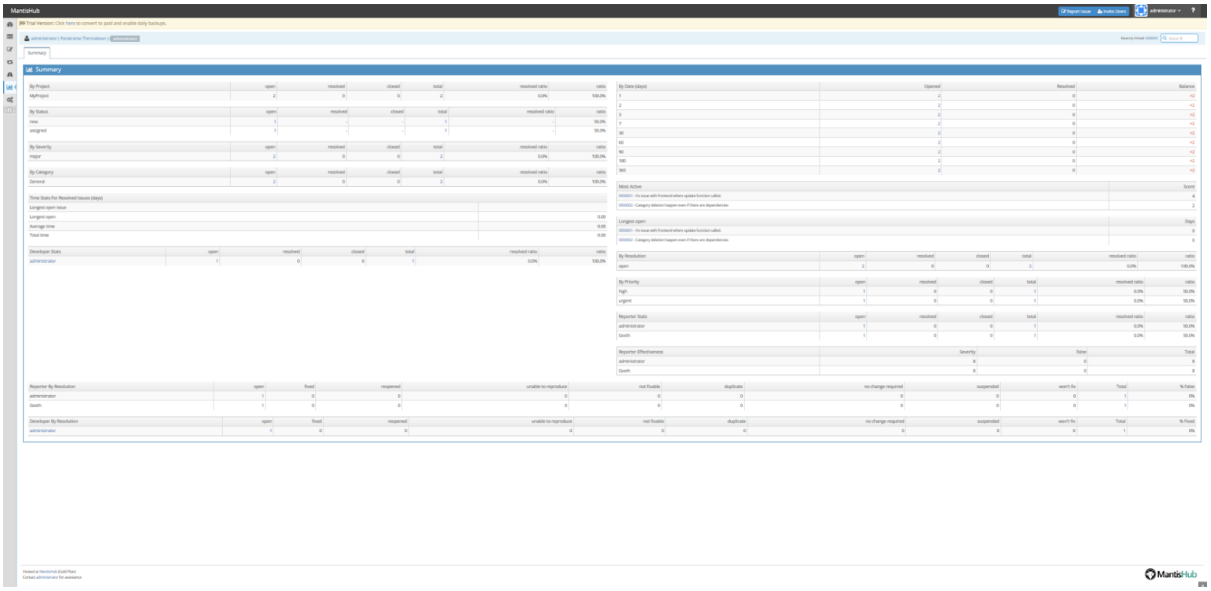


After completion of issue, status will changed to resolve by the assignee. There are multiple statuses we can assign to issue as show below.

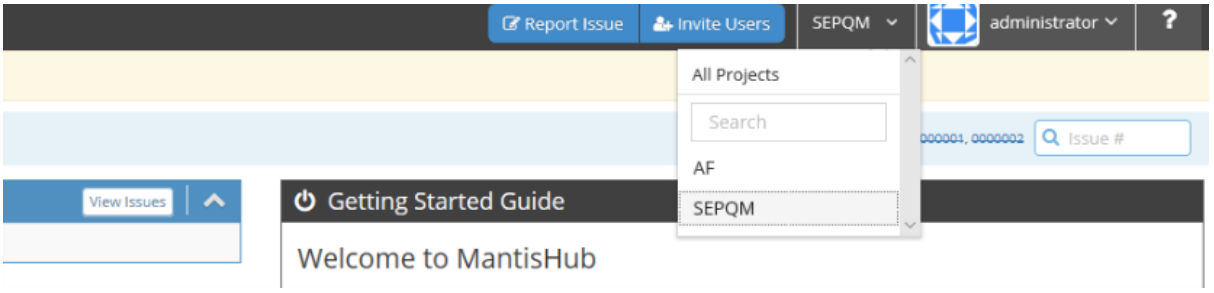




A great view of summary for the logged in use can be viewed in Summary tag.



Select currently working project(issue reporting and tracking) from upper right corner.



Code Coverage Tool

Code Coverage is a metric where we measure how much of our code is tested (or covered) by some test case (or test suite). So it actually measures the quality of the tests that are designed to measure the quality of the code rather than measuring the quality of the code. The amount of code is measured by either number of lines, number of statements or number of blocks. Usually the output is a percentage calculated by the amount of code covered by the test case (or the suite) over the whole code. So, higher the percentage, better the quality of the test case (or test suite).[4]

There are many tools out there that have been developed to measure the above mentioned metric. Most of them works well with the popular Integrated Development Environments. As a result, these tools can be used not just to calculate the code coverage percentage, but to visually display the code segments covered and not covered by a particular test case (or suite).

JaCoCo by EclEmma

JaCoCo stands for Java Code Coverage. As its name implies, it covers Java code. It is a free and open source tool developed by EclEmma team and licensed under the terms of the Eclipse Public License. It can integrate with Ant, Maven, Gradle, Jenkins, Visual Studio, and many others. It replaced EMMA as the code coverage tool that powers the EclEmma plugin for Eclipse IDE. Its latest release is 0.8.1. It is one of the few open source code coverage tools that are being actively developed[5][6].

Pros and Cons

Pros

- Has both pre instrumentation(offline instrumentation) and on-the-fly instrumentation[8] for class files.
- Follows test driven[13] development in its development process(all components pretested upon deployment) and lets the client see its test results.
- Does not instrument the code, but uses a JRE agent to capture execution[7] information.
- Provides JaCoCo Command Line Interface, Java Agent , Java API with the module. All required are downloaded in one module[6].
- Jacoco reports coverage in its HTML report in a detailed and appealing manner
- Has a good documentation.[5]
- JaCoCo provides Method, Line, Branch, Global Coverage and Per-Test coverage, as its coverage metrics[7].
- Provide the capability for Cross-Report linking[7].

Cons

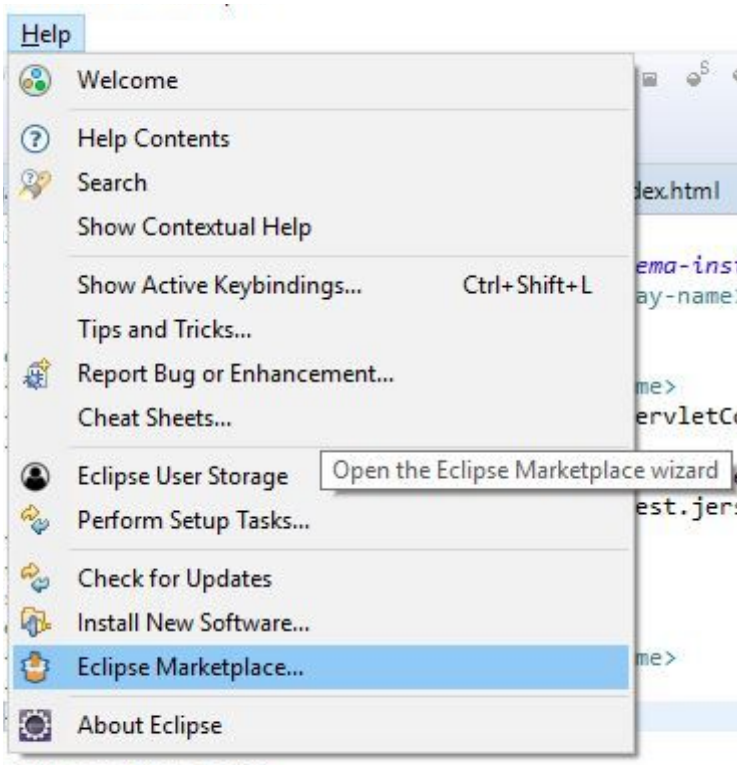
- Statement, MC/DC, Mutation like code coverage metrics not supported.[7]
- Does not provides instrumentation on Source files.[7]
- Does not provide PDF reporting and reports generated as only web based[5][7] documents(HTML, CSV and XML)
- Offline instrumentation has several drawbacks with Java Agents[10] (which used to track code coverage).
- Limited number of test frameworks[7] supported(JUnit, TestNG)

Demonstration of the tool

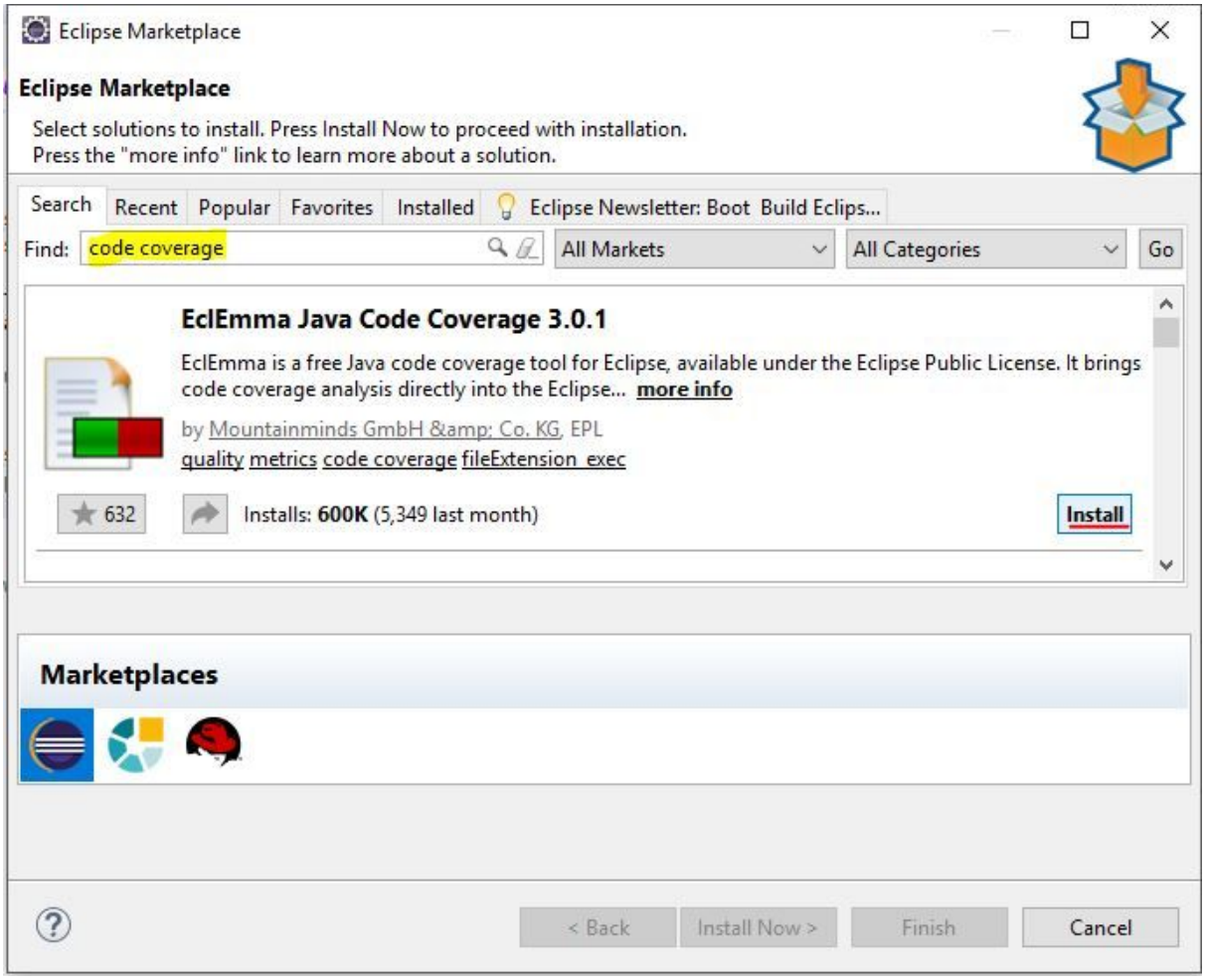
Setting up the environment

We will demonstrate the tool using EclEmma plugin for Eclipse using Eclipse Oxygen.

First we'll have to install the plugin using the Eclipse Marketplace.
Goto the Help menu in eclipse and select Eclipse Market.



Then in the search tab, search for ‘code coverage’. That will bring up the EclEmma Java Code Coverage x.x.x.
Click on the ‘Install’ button to install the plugin.



Accept the license agreement and click finish.
You will have to restart the eclipse IDE for changes to take place.

Scenario for the Demonstration

We will create java class to generate first n numbers in Fibonacci series, when n is given and a JUnit test class to test the functions of the above class. We will deliberately omit testing of some functions for the sake of the demonstration. Then we will be testing the code coverage of the created test using EclEmma plugin.

Demonstration

CrunchifyFibonacci.java contains 4 methods including the main method.

```java
import java.util.Scanner;

public class CrunchifyFibonacci {

    @SuppressWarnings("resource")
    public static void main(String args[]) {

        // input to print Fibonacci series upto how many numbers
        log("Enter number upto which Fibonacci series to print: ");
        int number = new Scanner(System.in).nextInt();

        log("\nUsing Method-1: Using Recursion. Provided Number: " + number);
        // printing Fibonacci series upto number
        for (int i = 1; i <= number; i++) {
            log(fibonacciRecusion(i) + " ");
        }

        log("\nMethod-2: Fibonacci number at location " + number + " is ==> " +
             (fibonacciLoop(number) + ""));

    }

    // Method-1: Java program for Fibonacci number using recursion.
    public static int fibonacciRecusion(int number) {
        if (number == 1 || number == 2) {
            return 1;
        }

        return fibonacciRecusion(number - 1) + fibonacciRecusion(number - 2);
        // tail recursion
    }

    // Method-2: Java program for Fibonacci number using Loop.
    public static int fibonacciLoop(int number) {
        if (number == 1 || number == 2) {
            return 1;
        }
        int fibo1 = 1, fibo2 = 1, fibonacci = 1;
        for (int i = 3; i <= number; i++) {
            fibonacci = fibo1 + fibo2; // Fibonacci number is sum of previous
            two Fibonacci number
            fibo1 = fibo2;
            fibo2 = fibonacci;

        }
        return fibonacci; // Fibonacci number
    }

    private static void log(String number) {
        System.out.println(number);

    }

}
```

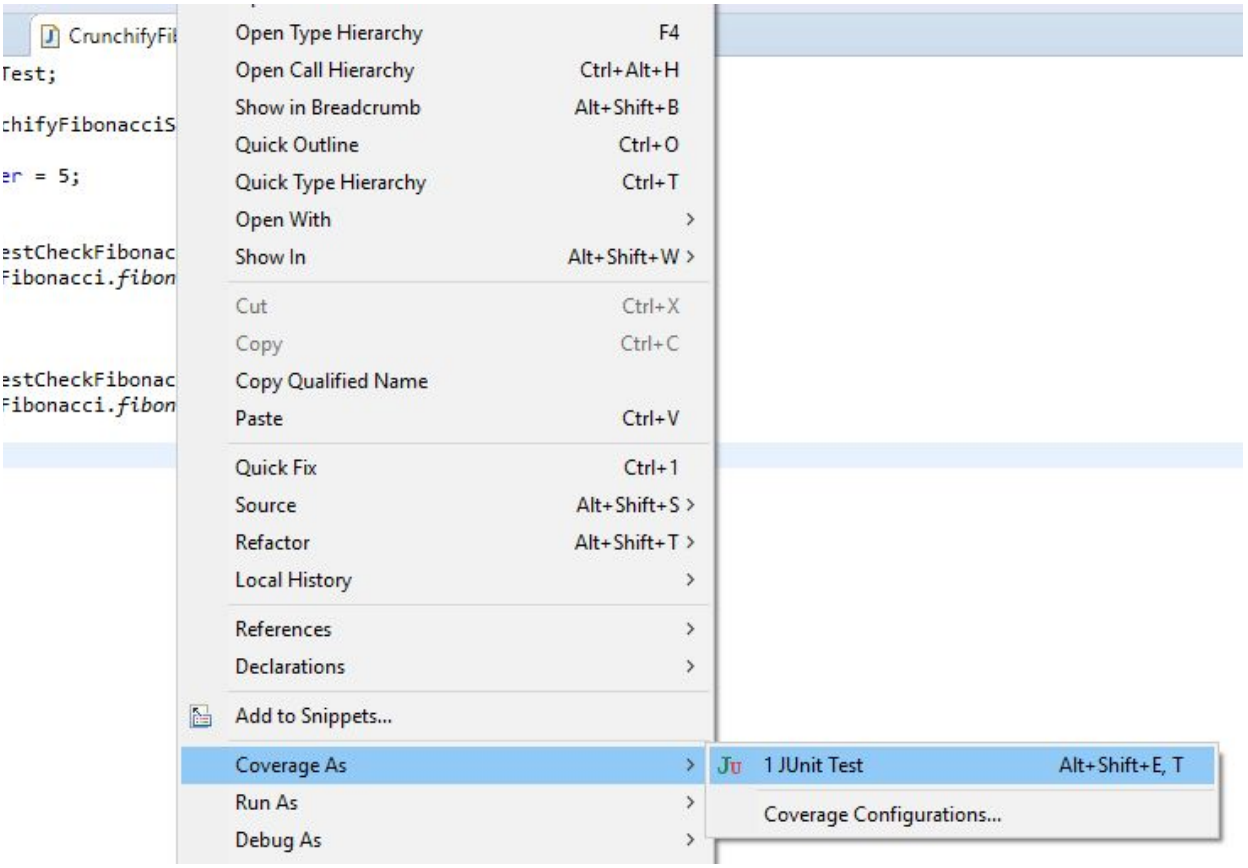
CrunchifyFibonacci.java

CrunchifyFibonacciSeriesTest.java is a JUnit test class that will test the *fibonacciRecusion()* method and *fibonacciLoop()* method. It will not test the *main* method and the *log()* method.

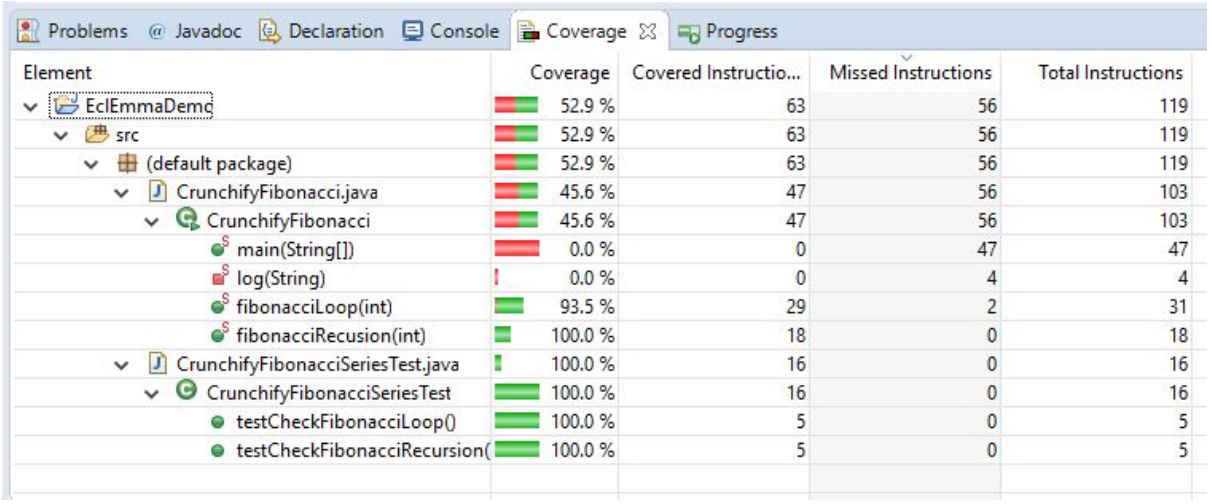
```java
import org.junit.Test;

public class CrunchifyFibonacciSeriesTest {

    int totalNumber = 5;

    @Test
    public void testCheckFibonacciRecursion() {
        CrunchifyFibonacci.fibonacciRecusion(totalNumber);
    }

    @Test
    public void testCheckFibonacciLoop() {
        CrunchifyFibonacci.fibonacciLoop(totalNumber);
    }
}
```

CrunchifyFibonacciSeriesTest.java

Right click on the *CrunchifyFibonacciSeriesTest.java* class' code and select *Coverage As* > *JUnit Test*



It will run the test class and generate some statistics on the code coverage. The report can be accessed from the Coverage view near the console view. It will decompose the coverage to each and every method.



Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
EclEmmaDemo	52.9 %	63	56	119
src	52.9 %	63	56	119
(default package)	52.9 %	63	56	119
CrunchifyFibonacci.java	45.6 %	47	56	103
CrunchifyFibonacci	45.6 %	47	56	103
main(String[])	0.0 %	0	47	47
log(String)	0.0 %	0	4	4
fibonacciLoop(int)	93.5 %	29	2	31
fibonacciRecusion(int)	100.0 %	18	0	18
CrunchifyFibonacciSeriesTest.java	100.0 %	16	0	16
CrunchifyFibonacciSeriesTest	100.0 %	16	0	16
testCheckFibonacciLoop()	100.0 %	5	0	5
testCheckFibonacciRecursion(	100.0 %	5	0	5

From the code editor, you can see which code segments are covered by the test and which are not. Covered code segments are highlighted with green color and red colored code segments have not been covered by the test.

CrunchifyFibonacciSeriesTest.java | CrunchifyFibonacci.java

```
1  import org.junit.Test;
2
3  public class CrunchifyFibonacciSeriesTest {
4
5      int totalNumber = 5;
6
7      @Test
8      public void testCheckFibonacciRecursion() {
9          CrunchifyFibonacci.fibonacciRecusion(totalNumber);
10     }
11
12     @Test
13     public void testCheckFibonacciLoop() {
14         CrunchifyFibonacci.fibonacciLoop(totalNumber);
15     }
16 }
17
```

CrunchifyFibonacciSeriesTest.java

CrunchifyFibonacci.java

```java
import java.util.Scanner;

public class CrunchifyFibonacci {

    @SuppressWarnings("resource")
    public static void main(String args[]) {

        // input to print Fibonacci series upto how many numbers
        log("Enter number upto which Fibonacci series to print: ");
        int number = new Scanner(System.in).nextInt();

        log("\nUsing Method-1: Using Recursion. Provided Number: " + number);
        // printing Fibonacci series upto number
        for (int i = 1; i <= number; i++) {
            log(fibonacciRecusion(i) + " ");
        }

        log("\nMethod-2: Fibonacci number at location " + number + " is ==> " + (fibonacciLoop(number) + ""));

    }

    // Method-1: Java program for Fibonacci number using recursion.
    public static int fibonacciRecusion(int number) {
        if (number == 1 || number == 2) {
            return 1;
        }

        return fibonacciRecusion(number - 1) + fibonacciRecusion(number - 2); // tail recursion
    }

    // Method-2: Java program for Fibonacci number using Loop.
    public static int fibonacciLoop(int number) {
        if (number == 1 || number == 2) {
            return 1;
        }
        int fibo1 = 1, fibo2 = 1, fibonacci = 1;
        for (int i = 3; i <= number; i++) {
            fibonacci = fibo1 + fibo2; // Fibonacci number is sum of previous two Fibonacci number
            fibo1 = fibo2;
            fibo2 = fibonacci;

        }
        return fibonacci; // Fibonacci number
    }

    private static void log(String number) {
        System.out.println(number);

    }

}
```

References

[1] istqbexamcertification.com. (n.d.). *What is Static analysis tools in software testing?*. [Online] Available at: http://istqbexamcertification.com/what-is-static-analysis-tools-in-software-testing/. [Accessed: 21 May 2018].

[2] En.wikipedia.org. (n.d.). *JSLint*. [Online] Available at: https://en.wikipedia.org/wiki/JSLint. [Accessed: 21 May 2018].

[3] *Mantisbt.org*, 'Admin Guide', 2018. [Online]. Available: https://www.mantisbt.org/docs/master/en-US/Admin_Guide/html-desktop/#admin.lifecycle.create. [Accessed: 22- May- 2018].

[4] stackify.com, 'The Ultimate List of Code Coverage Tools: 25 Code Coverage Tools for C, C++, Java, .NET, and More', 2017. [Online]. Available: https://stackify.com/code-coverage-tools/. [Accessed: 20- May- 2018].

[5] www.jacoco.org, 'Java Code Coverage for Eclipse', 2017. [Online]. Available: https://www.jacoco.org/index.html. [Accessed: 20- May- 2018].

[6] www.jacoco.org, 'JaCoCo Java Code Coverage Library', 2017. [Online]. Available: https://www.jacoco.org/jacoco/. [Accessed: 20- May- 2018].

[7] confluence.atlassian.com, "Comparison of Code Coverage Tool", 2017. [Online]. Available https://confluence.atlassian.com/clover/comparison-of-code-coverage-tools-681706101.html. [Accessed: 20- May- 2018].

[8] www.eclemma.org, 'Offline Instrumentation', 2018. [Online]. Available: https://www.eclemma.org/jacoco/trunk/doc/offline.html. [Accessed: 20- May- 2018].

[9]www.jacoco.org, 'Documentation', 2018. [Online]. Available: https://www.jacoco.org/jacoco/trunk/doc/index.html. [Accessed: 20- May- 2018].

[10] www.eclemma.org, 'Java Agent', 2018. [Online]. Available: https://www.eclemma.org/jacoco/trunk/doc/agent.html. [Accessed: 20- May- 2018].

[11] crunchify.com, 'Eclipse and EclEmma: Best Code Coverage Plugin – Complete Working Testcase Tutorial', 2017. [Online]. Available: https://crunchify.com/what-is-the-best-code-coverage-plugin-you-should-use-in-eclipse-ide/. [Accessed: 20- May- 2018].

[12] crunchify.com, 'Write Java Program to Print Fibonacci Series up-to N Number [4 different ways]', 2017. [Online]. Available: https://crunchify.com/write-java-program-to-print-fibonacci-series-upto-n-number/. [Accessed: 20- May- 2018].

[13] www.jacoco.org , 'Test Driven Approach, JUnit Test Results', 2018. [Online]. Available: https://www.jacoco.org/jacoco/trunk/test/index.html. [Accessed: 20- May- 2018].

[14]"Mantis Bug Tracker helps developers manage defects", *SearchSoftwareQuality*, 2018. [Online]. Available: https://searchsoftwarequality.techtarget.com/feature/Manage-defect-info-using-the-MantisBT-bug-tracking-software. [Accessed: 23- May- 2018].