**PROJECT**

**METRICS COLLECTOR**

**TEST APPROACH DOCUMENT**

Team members

Akshara Boppidi

Pooja Jadhav Eshwarlal

Raja Ravi Teja Ponna

Sponsor

Anthony Giorgio

April 3, 2017

Marist College

Poughkeepsie, NY

**Table of Contents**

**Document Revision History 1**

**Purpose1**

**Audience1**

**Test Approach1**

**Test Scenario1**

**Test Types2**

**Development Environment3**

**Error Reporting3**

**Test Risks3**

**Document Revision History:**

|  |  |  |
| --- | --- | --- |
| **Version #** | **Revision**  **Date** | **Major Changes** |
| 1.0 | *03/20/2017* | Test Plan draft |
| 1.1 | *03/28/2017* | Updated Items to be Tested |
| 1.2 | *04/03/2017* | Modified Types of Testing |

**Purpose:**

The test approach document is a track of sequential steps to be taken to test the operation of the Metrics Collector. The first draft of this document was made during the planning phase and design of final draft of the Metrics Collector.

**Audience:**

This document will address 3 main target audience:

* *User/Customer:* They can use metrics collector for extracting details about their system performance.
* *Developer:* They implement test cases and suspect areas of code to focus test efforts on.
* *Tester:* They can review test specifications, defect reports and test results.

**Test Approach**

We use Agile Methodology, where we work in short sprints. Each sprint is focused on specific requirements, so we may not have an extensive detailed test plan for each sprint, but we will design a high level agile test strategy guide.

**Test Scenarios**

We compare statistics of the system performance at various intervals of time by repeatedly running the code. The network, memory and CPU statistics will be tested.

**Test types:**

|  |
| --- |
| Unit Testing |
| Performance Testing |
| Regression Testing |
| Integration Testing |
| Customer Testing |

**Unit Testing:**

* Unit test is performed manually by the developers of the team.
* Used to check whether each unit of the software/code performs as designed minimally.
* JUnit is a unit testing framework for Java programming language.

**Performance Testing:**

* Performance testing is done to provide stakeholders with information about their application regarding speed, stability and scalability.
* It ensures customer satisfaction and protects product failure.
* We will perform this testing by calculating the response time at the end of each iteration.

**Regression Testing:**

* Ensuring the functionality of the modules, even after the code changes.
* User will know that code is at least good, until the prior version.
* Regression testing is performed at least once after adding new modules into code.

**Integration Testing:**

* Integration testing is a logical extension of unit testing.
* In this, we combine two units that have already been tested into a single component and interface between them is tested.

**Customer Testing:**

* Generates feedback on product functionality beyond question at hand as well as insights on customer behavior.
* Customer demo at the end of the project, which is considered to be handover to customer testing.
* Here, customer involvement means we as a team will immerse in the place of customer and perform continuous testing after each sprint of code written.

***Development Environment:***

* Programming language : Java
* Environment : NetBeans IDE
* Database : SQLite
* Build Automation Tool : Maven
* Source Code Repository : GitHub

**Error Reporting:**

Any error encountered during testing process, they are written in bug report document. We will record the bugs by creating issues in the Git repository.

**Test Risks:**

* Manual testing is done for this project, which sometimes might not be accurate all the times due to human errors.
* Creating error situations for better testing of the code we might end up in creating new errors.

**Test Plan Approval:**

Print Name:

Signature: Date: