System Test Plan

For IOT BASED ALERT SYSTEM TO HIGHLIGHT PROCESS

Votary Softech Solutions Pvt. Ltd.

Plot No: 76, Lumbini layout,   
Near Euro school,   
Gachibowli-I (V), Hyderabad,  
Telangana - 500032,   
India.

**Revision History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Version (x.y) | Date of Revision | Description of Change | Reason for Change | Affected Sections | Approved By |
| 1.0 |  | New Definition | New | ALL |  |
| 2.0 | 03-08-2017 | Designed for multiple repositories | To improve capability | 1.1,1.4, 1.5,2.6 |  |
|  |  |  |  |  |  |

**Approval History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version (x.y) | Prepared By | Reviewed By/Date | Approved By/Date |
| 1.0 | Dayanand Patil |  |  |
| 2.0 | Dayanand Patil |  |  |
|  |  |  |  |

**Contents**

[1 Define 4](#_Toc474767832)

[1.1 Overview 4](#_Toc474767833)

[1.2 Test Deliverables 4](#_Toc474767834)

[1.3 Prerequisites 4](#_Toc474767835)

[1.4 Assumptions 4](#_Toc474767836)

[1.5 Limitations 4](#_Toc474767837)

[1.6 Glossary 5](#_Toc474767838)

[2 Specify 5](#_Toc474767839)

[2.1 Test Items 5](#_Toc474767840)

[2.2 Testing Tasks 5](#_Toc474767841)

[2.3 Features to Be Tested 5](#_Toc474767842)

[2.4 Features Not To Tested 5](#_Toc474767843)

[2.5 Team Structure 6](#_Toc474767844)

[2.6 Training Needs 6](#_Toc474767845)

[2.7 Schedule 6](#_Toc474767846)

[2.8 Defect Reporting and Management 6](#_Toc474767847)

[3 Architecture 6](#_Toc474767848)

[3.1 Test Environment Block Diagram 6](#_Toc474767849)

[3.2 Testing Tools 6](#_Toc474767850)

[3.3 Test Configuration Information 6](#_Toc474767851)

[4 Design 7](#_Toc474767852)

[4.1 Approach 7](#_Toc474767853)

[4.2 Entry/Exit Criteria 8](#_Toc474767854)

[4.3 Suspension Criteria and Resumption Requirements 8](#_Toc474767855)

[4.4 Risks and Contingencies 9](#_Toc474767856)

[5 Implement 9](#_Toc474767857)

[6 Validate 9](#_Toc474767858)

[7 Deploy 10](#_Toc474767859)

[8 Maintain 10](#_Toc474767860)

# Define

## Overview

This document describes the required information and approach for Build and continuous Integration testing.

1.Jenkins Monitors git repository for any check-in to GIT and Triggers the Build.

2.if Build fails send the Notification to respective e-mail saying Build failed. If Build get success/fail trigger the Test Scripts and trigger GUI pop-up and if build fail then trigger alarm.

3.To read e-mail notification coming from Jenkins server if Build fails/success send the Notification to respective e-mail saying Build failed/success and trigger the Test Scripts, pop-up GUI window for each Jenkins user and if build fails trigger the alarm.

## Test Deliverables

Define the outputs of the test effort, for example:

* **Test Plan**: It describes the scope, approach for Build and continuous Integration testing. testing to satisfy customer requirements.
* **Test Design Documents**:Required to check whether our design meet all requirements with customer requirements.
* **Test Specifications**: Defines steps to be follow for testing and pass/fail criteria.
* **Test Cases/Scripts**: Required to check system functionality using pass/fail status.
* **Test Progress Reports:** Test report gives project and requirement matching status
* **Defect (Fault) Reports:** final result of test execution
* **Test Results/Summary Reports:** final test report which gives short details of no. of test case pass/fail.

## Prerequisites

* Linux environment
* Jenkins setup
* GIT

## Assumptions

* Assume that source code has push on to GIT server.
* Jenkins has tracked changes from GIT server.
* Jenkins server is sending notification on same e-mail for multiple GIT servers build fail/success.
* Raspberry pi set up has done.

## Limitations

* Remote client support single notification at a time.
* Email notification not possible to read on more secured access email.
* If we get more than one build failure at a time we are able to give one alarm that can be closed by the user. Number of pop- up displayed is depending on the number of build fails/success which carries the message of corresponding build.

## Glossary

| **Term** | **Definition** |
| --- | --- |
| DM | Device Management |
| GIT | Version control Tool |
| Jenkins | Build and Integration Tool |
| Raspberry Pi | Development board |

# Specify

## Test Items

* GIT server URL is input to Jenkins.
* Email notifications is input to raspberry pi.
* For testing:

Refer Test case document and test report document.

## Testing Tasks

|  |  |  |
| --- | --- | --- |
| s | Task | Description |
| 1 | Test environment setup | Test Lead/ Tester |
| 2 | Create the test specification/document | Test Lead |
| 3 | Perform Test Execution | Tester |
| 4 | Automation of test cases | Test Automation Engineer |
| 5 | Test Report | Test Lead |

## Features to Be Tested

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SL # | Requirements | Description | Identification Method | Test criteria |
| 1 | REQ – 001 | Jenkins should be able to access GIT Repository | Jenkins should able to detect the path of GIT | Tester should check for permission deneid for the GIT repository for no access/wrong repository |
| 2 | REQ – 002 | Build Cloning from GIT | Check new build available in Jenkins workspace. | Tester has to check manually in Jenkins workspace |
| 3 | REQ – 003 | Jenkins tracking build changes from GIT | 1. Check for new build available in Jenkins workspace.  2. Check for automatic initiation of build execution. | Tester has to check manually in Jenkins workspace  and Jenkins web server for build execution |
| 4 | REQ – 004 | Build success/fail email notification | 1. build mail should appear in provided mail id. | Tester should check email manually. |
| 5 | REQ – 005 | Generating test report(For automation test script) | Check in Jenkins workspace | Tester has to check manually in Jenkins workspace. |
| 7 | REQ – 006 | GUI pop-up | Pop-up window should show on display | Tester has to check manually on display. |
| 6 | REQ – 007 | Alarm Triggering | Alarm trigger on hardware connected to raspberry pi after build fail | Tester has to check for build fail, and generation of alarm tone. |

## Team Structure

|  |  |  |
| --- | --- | --- |
| SL # | Task | Description |
| 1 | Test environment setup | Test Lead/ Tester |
| 2 | Create the test specification/document | Test Lead |
| 3 | Perform Test Execution | Tester |
| 4 | Automation of test cases | Test Automation Engineer |
| 5 | Test Report | Test Lead |

## Training Needs

* Jenkins setup
* basic Linux
* GIT setup

## Schedule

**Version 1.0:**

Generation of test cases ,test report documents, bug fix has to be done in between 18 July – 24 July.

**Version 2.0:**

## Defect Reporting and Management

Defects found during testing will be logged in Defect Tracking Tool (Bugzilla/mantis/excel) by Votary Softech QA or Votary Softech developer.

# Architecture

## Test Environment Block Diagram

## Testing Tools

* Mantis
* Jenkins
* python 2.7
* GCC

## Test Configuration Information

* **Hardware Configuration** : Follow raspberry pi setup document
* **Software Configuration**: Follow Jenkins setup document and DM setup document.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Unit Test Server | Unit Test Clients | System Test Server | System Test Client |
| Hardware | Raspberry Pi, speaker |  |  |  |
| Processor |  |  |  |  |
| Memory | 2 - 4 GB RAM |  |  |  |
| Storage | 512GB |  |  |  |
| OS | Linux |  |  |  |
| Database |  |  |  |  |
| Other Software | Jenkins |  |  |  |

# Design

## Approach

**Requirements phase**

Requirements phase – identify all requirements [in storage management] documents that will be produced in this phase, and all inspection results that will be generated as a result of static testing.

* Requirements Analysis will be performed for clear understanding of the Solution and all the documents to verified
* For each requirement, test cases shall be identified and documented in test cases document.

**Systems design phase**

System design phase - identify all requirements documents that will be produced in this phase and plan for test team participation in the associated inspections, along with all inspection results that may be generated as a result of static testing

* Design Analysis will be performed for clear understanding of the Solution and all the documents to verified
* For each requirement, test cases shall be identified and documented in test cases document.

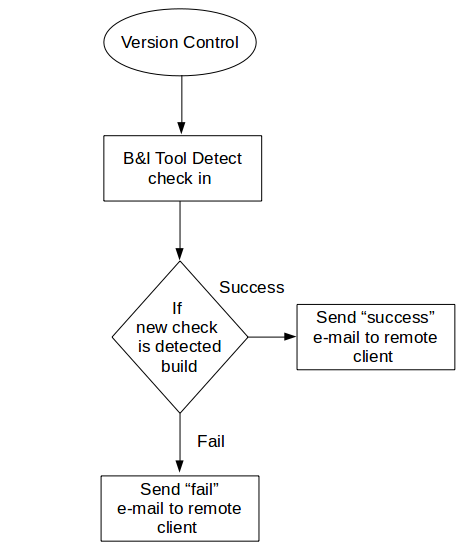
**Testing Phase**

1. Black box techniques such as BVP (Boundary Value Analysis), ECP (Equivalence class partitioning) and decision table shall be used to generate test data.
2. Planning for test execution will be made to ensure test coverage with better cost and testing time optimization
3. Dry run/Sanity Testing shall be performed to determine further testing which are system, regression, stability, load and compatibility testing
4. Regression Testing shall be performed in case of bug fixes or any other changes to ensure that they did not impact the build.
5. The QA Team will perform Functional testing only on latest build.
6. Every increment intended towards release shall be tested for its compatibility on different browsers and formats
7. During Functional testing, QA Team will use preloaded data which is available on the system at the time of execution
8. Every increment intended towards release shall be tested for its performance and stability on various platforms and versions

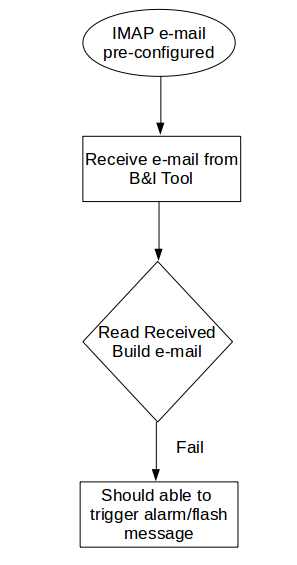
9. Release notes will be validated for Feature List and Unit/Integration Test reports to determine further testing.

**4.1.2 Flowcharts**

→ Flow Chart for B&I



→ Flow Chart for Remote Client



## Entry/Exit Criteria

Entry Criteria

1.System with linux OS.

2.Jenkin Setup

3.GIT setup

4.Internet connection

Exit Criteria:

All planned tests have been run;

All fixed bugs have been verified;

All new bugs have been reported;

All exceptions to the test plan have been documented.

Test Pass/Fail Criteria: Follow 2.3

## Suspension Criteria and Resumption Requirements

## Risks and Contingencies

Consider what is most likely to go wrong during the test effort, and then define the plan for fixing the problems that arise. The following are common areas of risk:

* Hardware availability
* Software availability
* Test cases not completed
* Availability of people for testing
* Training or skill levels have not been achieved prior to start of testing
* Requirements changing during development or during testing

# .Implement

Manual Test cases

Hardware and software setup documents.

# Validate

<References to Review defect logs, Checklist of Test Plan Document, Approval Emails, Traceability >

# Deploy

<Commits to SVN for team reference

Create project repository for Scripts / Documents and Test environment with approved Plan>

# Maintain

<Changes to Plan with appropriate references>