**IP4: Software Metrics**

**Software Test Metrics**

Software Metrics are used to measure the quality of the project. Simply, metric is a unit used for describing an attribute. Metric is a scale for measurement.

The Software Test Metrics are used to:

* Take the decision for next phase of activities such as, estimate the cost & schedule of future projects. Understand the kind of improvement required to success the project. Take decision on process or technology to be modified etc.
* Check that the business requirements and ensure that all of them have been met.
* All the specifications of the application code are complete.
* Before getting started with the User acceptance testing, unit, system testing, and integration will be complete.
* Regression testing needs to be completed.
* Fix all the identified defects before the final test.

These are the metrics that will be considered for the IOT based Weather app project:

**Test Effort**

Metrics measuring test effort will answer the following questions: “how many and how long?” about tests. They help to set baselines, which the final test results will be compared to.

Some of these QA metrics that will be used are:

1. Number of tests in a certain time period = Number of tests run/Total time
2. Test design efficiency = Number of tests designed/Total time
3. Test review efficiency = Number of tests reviewed/Total time
4. Number of bugs per test = Total number of defects/Total number of tests

**Test Effectiveness**

We will use this metric to answer the questions – “How successful are the tests?”, “Are testers running high-value test cases?” In other words, it measures the ability of a test case to detect bugs. This metric is represented as a percentage of the difference between the number of bugs detected by a certain test, and the total number of bugs found for the app.

(Bugs detected in 1 test / Total number of bugs found in tests + after release) X 100

The higher the test effectiveness percentage, the better and robust product it will be. Consequently, the lower the test case maintenance effort required in the long-term.

**Test Coverage**

Test Coverage will measure how much the application has been put through testing.

* Test Coverage achieved would be more than 95%.
* Test case Pass Rate should be 95%.
* All critical Test cases are passed.
* 5% Test cases can be failed but the Failed Test cases are of low priority.
* Complete Functional Coverage is achieved.
* All major functional / business flows are executed successfully with various inputs and are working fine.

**Defect Distribution**

Software quality assurance metrics must also be used to track defects and structure the process of their resolution. Since it is usually not possible to debug every defect in a single sprint, bugs must be allocated by priority and severity

1. Priority of the bugs is associated with scheduling, which means in what order the defects will be fixed. The higher the priority is, the sooner the problem needs to be investigated. Very often, bug priority is determined by its severity. Well, it is reasonable to start fixing with blockers rather than minor defects. For this app, while looking at the bugs, we will be looking into the category in which the bugs fit into and based on it, the higher the priority of the bug, the sooner the bug will be resolved. If the bug is of low priority, it may be fixed in a later sprint.
2. Bug severity is the extent of the impact a particular defect has on the software under test. The higher effect this defect has on the overall functionality or performance, the higher the severity level is. The higher the severity of the bug, for example if it is a showstopper or blocker, the bug will be resolved before the acceptance testing and user acceptance testing. If the bug is of low priority, it may be fixed in a later sprint.

* Defined / Desired bug count is reached.
* All Show Stopper bugs, or blockers are fixed, and no known critical / severity 1 defect is in Open Status.
* All High Priority defects are identified and fixed.
* Defect Rate falls below defined acceptable rate.
* Very few Medium Priority defects are open and have a workaround in place.
* Very few low priority open defects that do not impact software usage.
* All High Priority defects are re-tested and closed and corresponding Regression scenarios are successfully executed.

The following metrics will also be determined before the validation and acceptance testing of the Weather Application system

1. Test Coverage Percentage = (Number of tests runs/Number of tests to be run) X 100
2. Pass Rate = (Total No of Test Cases Passed / Total number of Test Cases) \* 100.
3. Requirements Coverage = (Number of requirements coverage/Total number of requirements) X 100
4. Passed Test Cases Percentage = (Number of Passed Tests/Total number of tests executed) X 100
5. Failed Test Cases Percentage = (Number of Failed Tests/Total number of tests executed) X 100
6. Blocked Test Cases Percentage = (Number of Blocked Tests/Total number of tests executed) X 100
7. Fixed Defects Percentage = (Defects Fixed/Defects Reported) X 100
8. Accepted Defects Percentage = (Defects Accepted as Valid by Dev Team /Total Defects Reported) X 100
9. Defects Deferred Percentage = (Defects deferred for future releases /Total Defects Reported) X 100
10. Critical Defects Percentage = (Critical Defects / Total Defects Reported) X 100

|  |  |  |
| --- | --- | --- |
| **Sl No** | **Metrics** | **Value in Percentage** |
| **1** | **Test Coverage Percentage** | **95** |
| **2** | **Passed Test Cases Percentage** | **95** |
| **3** | **Failed Test Cases Percentage** | **5** |
| **4** | **Critical Defects Percentage** | **0** |
| **5** | **High priority Defects Percentage** | **0** |
| **6** | **Medium Priority Defects Percentage** | **10** |
| **7** | **Low priority Defects Percentage** | **20** |
| **8** | **Blocked Test Cases Percentage** | **5** |
| **9** | **Fixed Defects Percentage** | **90** |
| **10** | **Accepted Defects Percentage** | **100** |
| **11** | **Defects Deferred Percentage** | **10** |
| **12** | **Total Test Cases Percentage** | **100** |

References:

<https://www.softwaretestinghelp.com/how-to-set-defect-priority-and-severity-with-defect-triage-process/>

<https://www.guru99.com/software-testing-metrics-complete-tutorial.html>

[Bug Severity vs Priority, or How to Manage Defect Fixing - QA Madness](https://www.qamadness.com/bug-severity-vs-priority/) [Software testing company](https://www.qamadness.com/)

<https://www.browserstack.com/guide/essential-qa-metrics>