**Software Testing Reports.**

**Why Do We Need Them?**

Any report is an important and laconic form of information transfer from the executor to the customer. Let me remember you about software testing process. Here we have the following stages:

* Project creating
* Test Plan preparing
* CI/CD integration
* Test Case execution
* Finding bugs
* Making reports

As you can see reports, which can be prepared, have to contain the information about the activities from the preceding stages.

So, we can define test report as a document containing information about the performed actions (run test cases, detected bugs, spent time etc.) and the results of this performance (failed/passed test cases, the number of bugs and crashes etc.).

If you think that your report is simply the heap of your work results, this is exactly what it would be like when you finish it. And if you want to express your intelligence via heavy-crammed report full of statistical data, it won’t be appraised. Besides, people who will read it might be slightly puzzled when they get it.



**What are benefits of test report?**

1. Well done test report allows us (and not only us) to evaluate the current status of project and quality of the product.
2. There is an ability to take corrective actions if it is necessary.
3. Highlight existing quality process issues (found/fixed/closed bugs)
4. The test report can be the final document which determines if the product is ready for release or not.

But what is most important, the report is a communication conductor. Thus, it shouldn’t be boring and too scientific.

If you want to show what a great worker you are, don’t write a report that only you understand how to decipher. You will scarcely benefit from it and most likely will have to rewrite it.

Treat your report as if it is a story you want to tell and make sure all the report sections correspond to this story. In such a way you get a logically structured and easy to read report that puts you and your company in a beneficial position.

**Whom test report is prepared to?**

When creating a report, you must fully understand who it is for and who will read it. Based on the priorities of the target audience, we must determine what information the report should contain. Three groups of target audiences can be distinguished:

**1.** **Technical users (Test managers)**

The understanding the progress of testing, and also how problems arise, how they are solved, the construction of the testing process, the description of the applied methods and technologies have a priority significance for them.

**2. Product Managers**

They are focused on the implementation deadlines, the pure test results without unnecessary technical details and the overall statistics (digital and comparative metrics).

**3. Business users (Product owners)**

As a rule, this is the people who make decisions on the end of testing. They also determine the quality of the work done. The final result, generated in the shortest and clearest format (“Yes” or “No”) is most important for them. the visual presentation of information (graphs, diagrams), expert opinion on the possibility of producing a product in the industrial environment, etc., without going into detail. Information should be presented in a visual form (graphs, diagrams). It is also important to have an expert opinion on the possibility of a release without going into details.

Of course, it is almost impossible to write a report, which will suit all the groups. That is why, be sure to identify the target audience, before preparing a test report. Depending on it, the content will be very different in structure and contain different details that are necessary for a specific group.

**Test report time sampling**

Test reports can be divided into types relative to time: intermediary or final.

**Intermediate test report** should reflect the current state of the project. State is not constant but dynamic and it is determined by comparing actual APP behavior and expected, based on a set of checks (unit/integration/performance tests).

Checks are created for each project individually, based on the goals that are defined for successful testing. They allow to make an overall comparative representation of the project available and quickly enough.

**Iteration test report** is another important and often used type of the intermediary report. It describes the work progress of the testing team for a particular version (iteration) of the product. It can include test automation coverage, pass rate, bug report status.

The **final report** shows a general view of the work done (in the context of established metrics) and the evolution of the product. Also, you need to give exhaustive information about the status of the product at the moment (the number of unfixed bugs remaining, whether the product is fully tested or an additional testing cycle is required, product readiness for the release etc.).

**Content of the test report**

So, let’s consider what exactly test report should contain. At the picture below, you can see the info required to be described in the informative report.

**

**Project Info**

Naturally, that you should specify the title of your project, product name, and version in the test report.

**Test Summary**

The next points are obligatory to be specified in this section:

* The number of the executed test cases
* The number of the passed test cases
* The number of the failed test cases
* Passed test cases percentage
* Failed test cases percentage
* Comments

**Test Objective**

The objectives of each stage of software testing process (functional testing, performance testing, UI testing etc.) are required to be described in the test report.

**Defects**

This section should contain:

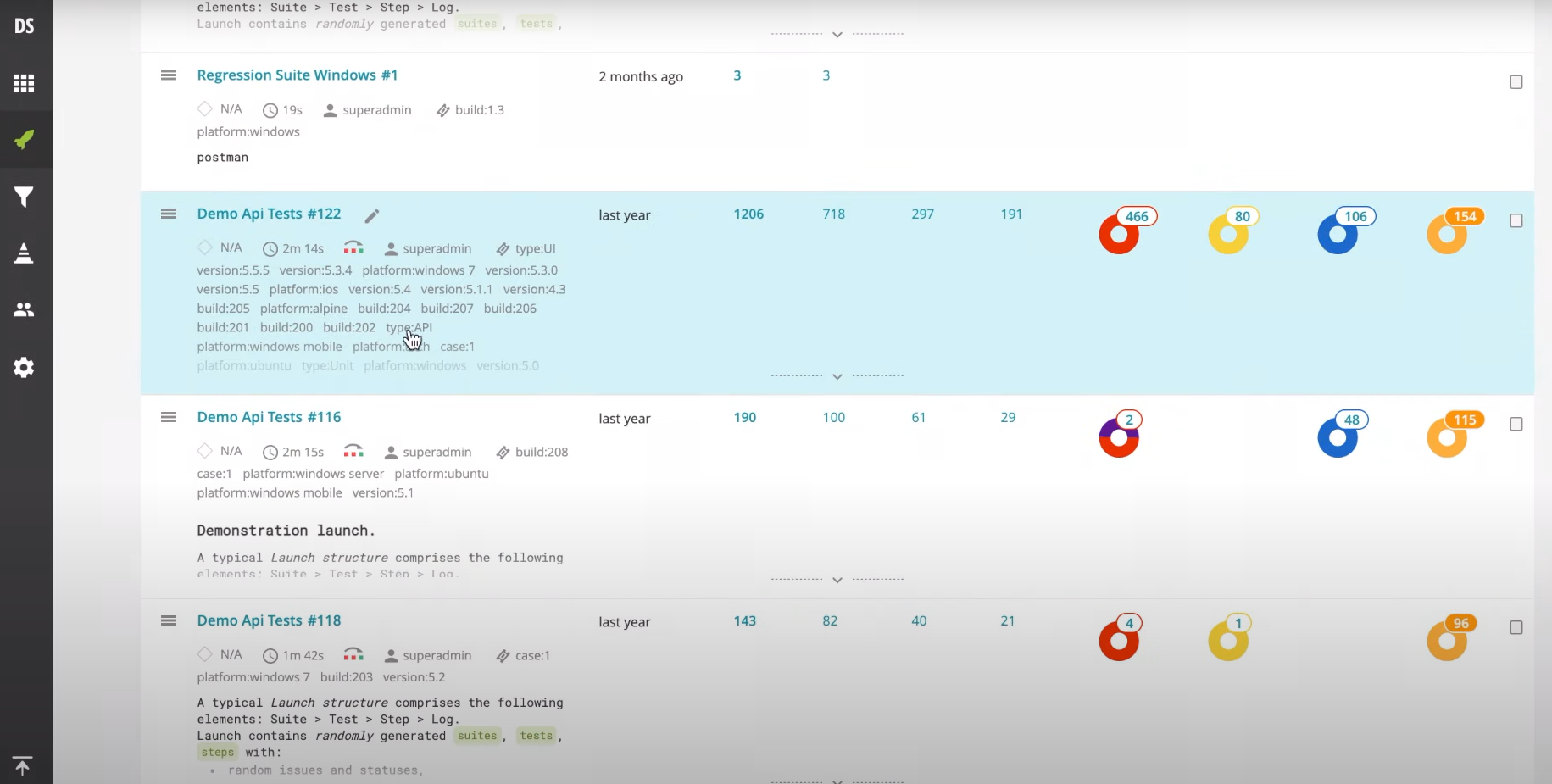
* Total number of detected bugs
* Bugs statuses (open, closed, fixed etc.)
* Number of bugs by each status (open, closed, fixed etc.)
* Severity and priority breakdowns

**Test report on a new build (Intermediate test report**)

It is a good practice to include our automated tests into CI process and setting up a proper automated test report generation. So, in case of APP code changes we will get a clear picture of current APP quality status.

It is not required, but possible to send such type of the report each time we get a new build, but we should be able to find the report data in case we asked to provide such information (ex. some tests of new build are failed)

As [example](https://katalon.com/resources-center/blog/test-reporting-tools), we can use a tool <https://reportportal.io>



**Iteration(sprint) test report**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Created By** |
| Version 2.1 | 22/03/2023 | Tarek Rushdie |
|

**Introduction**

This Test summary report is used to clarify the testing activities that happened for the previous **Android** build of the XYZ Mobile application in the period from (18-3-2023) to (22-3-2023).

The goal of this report is to show the following:

* Test automation coverage (or test coverage)
* Status of test cases executed
* Defects found & their status
* Suggestions for next period of testing

**Test Automation coverage**

This is the status of test automation coverage (Features implemented/ Features covered with tests \* 100%):

|  |  |  |
| --- | --- | --- |
| **Test Automation status** | **Current Build Results** | **Previous Build results** |
| Total Number of Stories in Sprint (APP features) | 26 (⇑) | 19 |
| Number of stories not required to cover | 0 | 2 |
| Test automation coverage | 100% | 89% |

**Test automation coverage notes**:

* + The number of stories not required for test automation decreased from 2 to 0 which also indicates, that we extend the ability for regression testing.
  + The tests automation coverage is increased to 100% which indicates an improvement in test automation planning and close communication between DEV and QA parts of the team.

**Test Case Execution Status**

This is the status of all the test cases that are run on the **Android** version:

|  |  |  |
| --- | --- | --- |
| **Test Case Status** | **Current Build Results** | **Previous Build results** |
| Total Number of TCs | 70 (⇑) | 63 |
| Passed TCs | 40(⇑) | 31 |
| Failed TCs | 18(⇑) | 16 |
| Blocked/Skipped TCs | 12(⇓) | 16 |

**Test Case Execution notes**:

* + The number of passed test cases increased from 31 to 40, which indicates an increase in the quality of the application
  + The number of failed & blocked/skipped test cases decreased from 32 to 20 which also indicates an increase in the quality of the application
  + Note that all test cases that included integrations between two users were executed on two android devices because there is no iOS build now. Testing integrations using iOS might cause more test cases to fail

**Defects Status for Android**

This table lists the number of defects in the **Android** version based on their priority:

|  |  |  |
| --- | --- | --- |
| **Defect Status** | **Current Build Defects** | **Previous Build Defects** |
| Total Number | 47(⇑) | 33 |
| High | 23 (⇑) | 12 |
| Medium | 13 (⇑) | 10 |
| Low | 8 (⇓) | 11 |

This table lists the number of defects in the **Android** version based on their type:

|  |  |  |
| --- | --- | --- |
| **Defect Status** | **Current Build Defects** | **Previous Build Defects** |
| Total Number | 47(⇑) | 33 |
| Functional | 22 (⇑) | 20 |
| UI/UX | 23 (⇑) | 10 |
| Performance | 2 (⇓) | 3 |

**Android Defects notes:**

* + The number of defects in the new build is more than the number of defects in the previous build, but although, we consider that the quality has increased. The cause of the high number of defects might be that new functionalities are added in the new build.

**Testing suggestions for next period**

* We suggest that a defect tracking tool like (JIRA) is used in writing defects so that the defect is tracked from reporting to resolving. This will reduce the time required to write new defect reports for each build.