

Testing for Kanban based Teams



By adopting a methodology like Kanban, your team will be working in a living backlog (that can change at any moment) and they will limit work in progress according to their capacity. Kanban is about continuous delivery, so there will be a release every time the team is ready to do it.

Testing is always taking place and is essential to consider the story ready for delivery.

1

Work as a team to understand user stories and create Tests

Just like in a SCRUM team, the work can be divided into user stories that usually represent the expected requirements. As a team, you need to make sure that everyone understands what needs to be done, so when someone picks the user story from the backlog it can be delivered without interruptions.

Testers are part of the team and need to make sure that everyone knows exactly how and when the work items are going to be tested.

To know how to create tests with Xray check the [Writing Tests](#) module, from the Tester course.



Tests, Pre-Conditions and Test Sets are typically reusable entities; thus, they probably should not belong to the scope of a release.

2

Select your Test Planning approach

After creating Tests, you should create the Test Plan. Depending on the approach that makes more sense to you, you can have at Test Plan by feature.

Approach	Recommendations
Test Plan by feature	<ul style="list-style-type: none">• Create a Test Plan that aggregates all the stories related with a specific feature (usually an epic)
Test Plans for Regression Tests & for Non-Regression Tests	<ul style="list-style-type: none">• Create two Test Plans: one for regression tests and other for non-regression tests
Multiple Test Plans	<ul style="list-style-type: none">• Create multiple Test Plans to have insights about the testing progress of a certain subset of tests.• Divide the test per type, that can be managed by different users.• More complex approach, be aware of the overhead.



It could make sense for your team to avoid creating Test Plan. There is nothing wrong with this approach, but be aware that you will loose some functionality, especially in terms of tracking and reports.

To know more on how to create Test Plans with Xray check the [Planning Tests](#) module from the Tester course.

3

Schedule Test Executions

You can schedule multiple Test Executions, from the Test Plan (if you have created one) or directly from each Story as Sub-Test Executions.

If you choose to create Sub Test Executions from each story, you will have the:

- ability to track the progress of the Test Execution in the Agile board, by seeing it in context with the parent Story
- ability to count the time estimates of the Sub-Test Executions within the overall time estimate at the parent Story
- ability to optionally restrict workflow transition on the parent requirement based on the workflow status of the related sub-tasks (e.g. allow transition of the Story to "closed" only if the Sub-Test Executions are also "closed")



It is not mandatory to use Sub-Test Executions, you can simply create a Test Execution and add several test to it.

However, as mentioned above, they may provide some interesting benefits for some specific use cases, including the ability to track them as ordinary sub-tasks of Stories.

To know more on how to create Test Executions with Xray check the [Executing Tests](#) from the Tester course.

4

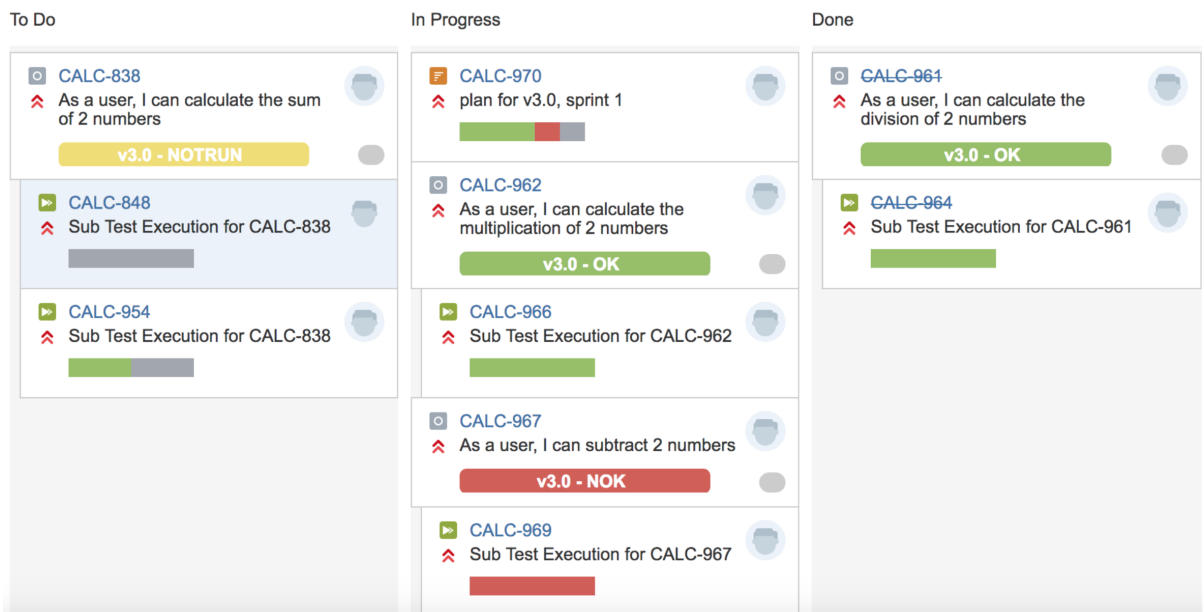
Track progress and get feedback

Using an Agile Kanban Board you can have immediate feedback on:

- **coverage status of the user stories** (or other entities that you cover with tests); this status includes real-time feedback about the latest test results
- **progress of Test Plans** (if you want, you may include Test Plans in the board)
- **progress of Test Executions** (if you want, you may include Test Executions in the board)

Sprint 1

QUICK FILTERS: [Only My Issues](#) [Recently Updated](#)



Besides this, you can also track overall progress at the Test Plan Level or by using Xray Reports like:

- **Overall Coverage Report**, to evaluate the current coverage status of the coverable entities (e.g. user stories)
- **Traceability Report**, to quickly filter out relevant entities based on their coverage, check the runs of related Tests and reported defects

- **Test Executions Report** and/or **Test Plans Report**, to see the results obtained on different environments, different revisions of the SUT, and to check if the reported defects are resolved or not.

Learn more about Tracking Progress & Reporting in the Advanced Reporting module, coming next in this course.
