Roadmap

⚠️ Xray’s roadmap is continuously reviewed and redefined. We update often, depending on the feedback we receive from our clients and internal stakeholders.

Our release plan is available in our Jira issue tracker. Feel free to view and to vote on the issues that you would like to see implemented (account registration required).

Here you can find a list of features that define our main goals for future releases. This doesn’t mean that other, potentially smaller features, won’t be implemented as well.

Shipped

The Test Steps component was the target of a major facelift with the goal of providing a wider, more usable UI that will have the ability to edit all step-related fields at once, in a grid or column-based layout.

Editing, reading and navigating through Test steps are now much easier with this new UI.
Along with the Test steps UI revamp, Xray now also provides the ability to configure and specify new custom step fields for manual test cases that can complement the standard ones (Action/Step, Data, Expected Result). The standard custom fields can also be hidden if desired. All of this can be configured in the project settings.

**Edit Steps**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example</td>
<td>Execute the following database script.</td>
<td><code>SELECT * FROM orders WHERE ID = &lt;ORDER_ID&gt;</code></td>
</tr>
<tr>
<td>Example 2</td>
<td>Execute the following request.</td>
<td><code>https://testing-server/local/BSE/store/order/&lt;ORDER_ID&gt;</code></td>
</tr>
</tbody>
</table>

**In the works**

We plan to add the ability to configure custom fields, that can be used to store additional information in test runs. These custom fields are managed entirely by Xray (these are not Jira custom fields) and can be configured also in the project settings. Test Run reports will also be updated to provide these custom fields as well.

**Wish list for future versions**

The goal behind the testing board is to provide a centralized hub for accessing Xray entities and activities in order to improve navigation and discoverability. For now, this board features the Test repository, the Test Plan board, and project reports. The testing board will be improved to include other Xray entities such as Preconditions, Test Sets, and Test Executions. The Test Run and execution screen will have the testing board in context so that users can navigate to any other test activity.

The **Xray Connector app for Bamboo** will be updated in order to support Xray cloud APIs and connectivity.

Ability to call an existing Test from the step of another Test and thus use tests as reusable building blocks for the composition of more complex testing scenarios. The called test can be executed independently or be used as a step of a broader test (e.g. "login as the admin user").
Ability to define variables in the Test specification (e.g. the steps) that can be replaced by values inherited from different sources (a calling test for example). In that scenario, the called test is used as a parameterized template (e.g. "login as the specified user").

Ability to perform data-driven testing (i.e. execute the same test against multiple parameters, for different combinations of data).

Data-driven testing will be implemented using the foundations of test parameterization.

Data may be defined at multiple levels including:

- the test level (default values)
- the caller test
- the Test Plan
- the Test Execution

Data might also be imported from external sources (like CSV files).

Currently, the Test Plan is composed of a static list of Test cases. This means you must explicitly add the Tests to the Test Plan. If the Tests are all known and well defined when you start your Test Plan, this is ok. However, if you are working in an agile context where a Test Plan is created for a specific sprint, Tests will only be specified during the sprint and later added to the Test Plan. This process is not ideal because users might forget to add the Tests to the Test Plan.

Dynamic Test Plans can be defined with a JQL query which will be the source for Test cases. Considering the use case described above, we can define a JQL query that will get all Tests covering requirements of a specific sprint.

This report will feature a list of Test Execution issues along with metrics. Possible usage scenarios:

- analyze both the progress of the Test Execution and the success rate (i.e., the % of Tests contributing to the requirement's OK status)
- see the number of manual Tests vs. others in the Test Execution
- see the overall execution status (i.e., the current status of the Test Runs)
- see the number of opened/closed linked defects, in the context of the Test Execution

Refer to the Test Executions Report on Xray server.

This feature will cover a common scenario where manual Test cases evolve to automated Tests. In this case, the only option is to create separate Test issues to cover all the natures for the Test.

With Test natures, multiple Test definitions can coexist within the same Test case. Users will also be able to set the current definition so that Test Runs are always created using the "current" definition.

Right now, a Test Run contains the start and finish date which are set automatically, although users can also change the start date manually. However, users can not stop and start the execution as needed and the total time spent on the Test Run (finish - start) might not reflect the time spent by the user executing the Test.

We plan to provide a feature to allow users to pause and continue with the execution directly on the Test Run screen.

This report will show a daily historical view of Requirements coverage. Users will be able to analyze the evolution of the Requirement's coverage status over time for a particular analysis scope: Latest, Version or Test Plan (for each Environment). This way they can estimate if they are going to keep the planned released date according to the coverage status trend.