Document Generator Template: Test Plan Report

- Purpose
- Output Example(s)
- How to use
    - Source data
    - Output format
    - Report assumptions
    - Usage examples
        - Export all details obtained in the context of a given Test Plan
- Understanding the report
    - Layout
        - "Introduction" section
            - Document Overview
            - Test Plan Details
            - Requirements covered by the Tests in this Test Plan
            - Overall Execution Status
            - Defects
            - Tests Summary
        - Test Executions
            - Test Executions Summary
                - Execution Defects
                - Execution Evidences
                - Comment
                - Test Description
                - Test Issue Attachments
                - Preconditions
                - Parameters
                - Iterations
                - Iteration Overall Execution Status
                - Test Run details
                - Iteration precondition definition
                - Iteration parameters details
                - Iteration Test Step Details
                - Test Details
                - Requirements linked with this test
        - Appendix A: Approval
- Customizing the report
    - Sections that can be hidden or shown
    - Adding or removing information to/from the report
        - Exercise: add a field from the related Test issue
- Performance
- Known limitations

Purpose

This report enables you to extract details of a Test Plan, such as the Tests that are part of it, Defects, Requirements and Test Executions, so that you can generate a document report focusing in what matter the most for your team, or even share it with someone else that hasn't access to Jira.

Possible usage scenarios:

- see all the requirements covered by the Test Plan
- see all the defects linked to this Test Plan
- see an overall status summary of the Test Plan
- see a summary of the Tests that are part of the Test Plan
- check a specific detail of a Test Execution (like evidences, attachments, assignee, etc)

Output Example(s)

The following screenshots shows an example of the sections you should expect in this report.

Project Name: **Bookstore**

Issue: **BOOK-30**

Prepared By: **Xpand IT Admin**

Document Date: 24/11/2022

CONFIDENTIAL

TEST PLAN

Test Plan for v1.0

1. Introduction

1.1. Document Overview

This Test Report provides a summary of a test plan. This document has been generated automatically from the test management platform.

1.2. Test Plan BOOK-30 Details

Begin Date	18-02-2019 14:03:00	**End Date**	22-02-2019 14:03:00
Summary	Test Plan for v1.0	**Description**	

1.3. Requirements covered by the Tests in this Test Plan

Key	Summary	Workflow Status	Test Status
BOOK-5	As a visitor, I can change my locale	Resolved	[OK]
BOOK-4	As a visitor, I can edit my account details	Resolved	[OK]
BOOK-1	As a visitor, I can manage my account	Open	[OK]
BOOK-3	As a visitor, I can logout from my account	Resolved	[OK]
BOOK-2	As a visitor, I can login to Bookstore Website	Resolved	[OK]
BOOK-7	As a visitor, I can search for	Resolved	[NOK]

1.6. Tests Summary

Key	Summary	Issue Assignee	Requirements	#TestExecutions	Latest Status
BOOK-18	Test a visitor can change his locale	Bruno Conde	BOOK-5	1	PASS
BOOK-17	Test a logged in visitor can edit the default address	Bruno Conde	BOOK-4 BOOK-1	1	PASS
BOOK-15	Test logged in visitors can logout from their account	Bruno Conde	BOOK-3 BOOK-1	1	PASS
BOOK-16	Test a logged in visitor can edit the account details	Bruno Conde	BOOK-4 BOOK-1	1	PASS
BOOK-14	Test visitors can login to Bookstore Website	Bruno Conde	BOOK-2 BOOK-1	1	PASS
BOOK-28	Test a visitor can filter the search result	Bruno Conde	BOOK-7	2	FAIL
BOOK-27	Test a visitor can do a valid search with multiple keywords	Bruno Conde	BOOK-7	1	PASS
BOOK-26	Test a visitor can do a valid search with a single keyword	Bruno Conde	BOOK-7	1	PASS
BOOK-25	Test a visitor can view all the books in his shopping basket	Bruno Conde	BOOK-11 BOOK-8	1	PASS
BOOK-24	Test visitors can remove books from their shopping basket	Bruno Conde	BOOK-10 BOOK-8	1	PASS

1.5. Defects found

Key	Summary	Priority
BOOK-32	Error when adding Book to shopping basket	Trivial
BOOK-502	global defect	Trivial
BOOK-503	another step level defect	Trivial
BOOK-501	step level defect	Trivial
BOOK-506	dummy def3	Trivial

3. Test Executions

3.1. BOOK-500: Test Execution for Test Plan BOOK-30

3.1.1. BOOK-498: Manual Test

Execution Status	Assignee	Executed By	Started On	Finished On	Versions	Revision
EXECUTING	Xpand IT Admin	xadmin		-	1.0	-

3.1.1.1. Execution Defects

Key	Summary	Priority
BOOK-502	global defect	Trivial

3.1.1.2. Execution Evidence

File Name	Author	File Size	Evidence
pexels-anna-kester-5352939.jpg	xadmin	2573650	

How to use

This report can be generated from the Issue details screen.

> **Learn more**
>
> General information about all the existing places available to export from, and how to perform it, is available in the Exporting page.

Source data

This report is applicable to:

- 1 Test Plan issue

Output format

The standard output format is .DOCX, so you can open it in Microsoft Word, Google Docs, and other tools compatible with this format.

Report assumptions

The template has a set of assumptions that you make sure that your Jira/Xray environment complies with:

1. Issue types having these names
    a. "Test", "Test Plan", "Test Execution"

If any of these assumptions is not met, you need to update the template accordingly.

Usage examples

Export all details obtained in the context of a given Test Plan

1. open the Test Plan issue and export it using this template

Understanding the report

The report shows detailed information about the Test Plan provided.

Layout

The report is composed by several sections. Two major sections are available: Introduction and Test Executions details.

By default, and to avoid overload/redundancy of information, only the "Introduction" section will be rendered; you can change this behavior on the template (more info ahead).

"Introduction" section

This section is divided into 6 sub-sections to have an overview about the Test Plan we have just exported:

1. Document Overview
2. Test Plan Details
3. Requirements covered by the Tests in this Test Plan
4. Overall Execution Status
5. Defects
6. Tests Summary

Each of these sections is explained below.

Document Overview

Brief description of what you will find in this report and how it was generated.

Test Plan Details

In this section we are extracting the Test Plan key in the header and show the Begin and End Date (formatted as demonstrated below), the *Summary* and the *Description* present in the Test Plan.

Field	Description	Sample Code
Begin Date	Timestamp of the Begin Date field present in the Test Plan (with proper format)	*${dateformat("dd-MM-yyyy HH:mm:ss"): Begin Date}*
End Date	Timestamp of the End Date field present in the Test Plan (with proper format)	*${dateformat("dd-MM-yyyy HH:mm:ss"): End Date}*
Summary	Summary of the Test Plan	*${Summary}*
Description	Description of the Test Plan (as this field accepts wiki markup we will use "wiki:" in the code to be interpreted by the document)	*${wiki:Description}*

The output will have the following information, notice that as the *Description* field support wiki markup we are using "*wiki:*" keyword so that it is correctly interpreted.

1.2. Test Plan BOOK-30 Details

Begin Date	18-02-2019 14:03:00	**End Date**	22-02-2019 14:03:00
Summary	Test Plan for v1.0	**Description**	

Requirements covered by the Tests in this Test Plan

In this section we have an overview of all the requirements that are covered by Tests in this Test Plan, we extract the *Key*, *Summary, Workflow* and the *Test Status* Status removing all the repeated entries.

In the Server version we have a query that fetches the Requirements linked with a Test: *testRequirements('${Tests[n].Key}')*.

Field	Description	Sample Code
Key	Key of the requirement (in this case we are adding it as link)	*@{title=${JQLIssues[a].Key}\|href=${BaseURL}/browse/${JQLIssues[a].Key}}*
Summary	Summary of the requirement	*${JQLIssues[a].Summary}*
Workflow Status	Workflow Status of the requirement	*${JQLIssues[a].Status}*
Coverage Status	Requirement coverage status	*${JQLIssues[a].Requirement Status}*

The requirements are listed in a table with the informations explained above.

1.3. Requirements covered by the Tests in this Test Plan

Key	Summary	Workflow Status	Test Status
BOOK-5	As a visitor, I can change my locale	Resolved	[OK]
BOOK-4	As a visitor, I can edit my account details	Resolved	[OK]
BOOK-1	As a visitor, I can manage my account	Open	[OK]
BOOK-3	As a visitor, I can logout from my account	Resolved	[OK]
BOOK-2	As a visitor, I can login to Bookstore Website	Resolved	[OK]
BOOK-7	As a visitor, I can search for	Resolved	[NOK]

Overall Execution Status

As the name suggests we have an overview about the executions of the Tests in this Test Plan, here you will have information about how many Tests you have in this Test Plan and what are the statuses of their executions.

To obtain this information we are using:

Field	Description	Sample Code
TestsCount	The total number of Tests in this Test Plan	*${TestsCount}*
#Tests	To extract the count of the overall execution status per each *<status>* (*TO DO, EXECUTING, PASS, FAIL, ABORTED*)	*${Overall Execution Status.<status>.Count}*
Percentage	To extract the percentage of the overall execution status per each *<status>* (*TO DO, EXECUTING, PASS, FAIL, ABORTED*)	*${Overall Execution Status.<status>.Percentage}*

This will produce the following output:

1.4. Overall Execution Status

Of the 22 Tests contained on BOOK-30:

Status	#Tests	Percentage
TODO	4	18.2 %
EXECUTING	1	4.5 %
PASS	13	59.1 %
FAIL	4	18.2 %
ABORTED	0	0.0 %

Defects

In this section we are listing all the defects found that are associated with this Test Plan, we consider defects associated with TestRuns, defects in Test Steps or defects found during the iterations. We do not print duplicates.

Field	Description	Sample Code

Key	Key of the Defect	• TestRun
		  ○ *@{title=${TestExecutions[n].TestRuns[a].ExecutionDefects[d].Key}\|href=${BaseURL}/browse/${TestExecutions[n].TestRuns[a].ExecutionDefects[d].Key}}*
		• TestSteps
		  ○ *@{title=${TestExecutions[n].TestRuns[a].TestSteps[j].Defects[m].Key}\|href=${BaseURL}/browse/${TestExecutions[n].TestRuns[a].TestSteps[j].Defects[m].Key}}*
		• Iteration TestSteps
		  ○ *@{title=${TestExecutions[n].TestRuns[a].Iterations[it].TestSteps[r].Defects[dc].Key}\|href=${BaseURL}/browse/${TestExecutions[n].TestRuns[a].Iterations[it].TestSteps[r].Defects[dc].Key}}*

Summary	Summary of the Defect	• TestRun
		  ○ *${TestExecutions[n].TestRuns[a].ExecutionDefects[d].Summary}*
		• TestSteps
		  ○ *${TestExecutions[n].TestRuns[a].TestSteps[j].Defects[m].Summary}*
		• Iteration TestSteps
		  ○ *${TestExecutions[n].TestRuns[a].Iterations[it].TestSteps[r].Defects[dc].Summary}*

Priority	Priority of the Defect	• TestRun
		  ○ *${TestExecutions[n].TestRuns[a].ExecutionDefects[d].Priority}*
		• TestSteps
		  ○ *${TestExecutions[n].TestRuns[a].TestSteps[j].Defects[m].Priority}*
		• Iteration TestSteps
		  ○ *${TestExecutions[n].TestRuns[a].Iterations[it].TestSteps[r].Defects[dc].Priority}*

The Defects appear in the document as a table with information regarding the defects found during the executions of the Test Plan.

1.5. Defects

Key	Summary	Priority
BOOK-32	Error when adding Book to shopping basket	Trivial
BOOK-502	global defect	Trivial
BOOK-503	another step level defect	Trivial
BOOK-501	step level defect	Trivial
BOOK-506	dummy def3	Trivial

Tests Summary

In this section we have a table with information regarding the Tests included in this Test Plan. You can find the following information about each Test:

Field	Description	Sample Code
Key	Key of the Test in a link form	*@{title=${Tests[n].Key}\|href=${BaseURL}/browse/${Tests[n].Key}}*
Summary	The Summary of the Test	*${Tests[n].Summary}*
Issue Assignee	Full name of the assignee	*${fullname:Tests[n].Assignee}*
Requirements	List of requirements covered by this Test (Check the template to see the extra cycle we need to list this information)	*${JQLIssues[a].Key}*
#Test Executions	Number of Test Executions for each Test (Check the template to see the extra cycle we need to list this information)	*${jqlcount:issue in testTestExecutions('${Tests[n].Key}') and 'Test Plan' = ${Key}}*
Latest Status	Latest Status of the execution	*${Tests[n].LatestStatus}*

This information is presented in a table as we can see below:

1.6. Tests Summary

Key	Summary	Issue Assignee	Requirements	#TestExecutions	Latest Status
BOOK-18	Test a visitor can change his locale	Bruno Conde	BOOK-5	1	PASS
BOOK-17	Test a logged in visitor can edit the default address	Bruno Conde	BOOK-4 BOOK-1	1	PASS
BOOK-15	Test logged in visitors can logout from their account	Bruno Conde	BOOK-3 BOOK-1	1	PASS
BOOK-16	Test a logged in visitor can edit the account details	Bruno Conde	BOOK-4 BOOK-1	1	PASS
BOOK-14	Test visitors can login to Bookstore Website	Bruno Conde	BOOK-2 BOOK-1	1	PASS
BOOK-28	Test a visitor can filter the search result	Bruno Conde	BOOK-7	2	FAIL
BOOK-27	Test a visitor can do a valid search with multiple keywords	Bruno Conde	BOOK-7	1	PASS
BOOK-26	Test a visitor can do a valid search with a single keyword	Bruno Conde	BOOK-7	1	PASS
BOOK-25	Test a visitor can view all the books in his shopping basket	Bruno Conde	BOOK-11 BOOK-8	1	PASS
BOOK-24	Test visitors can remove books from their shopping basket	Bruno Conde	BOOK-10 BOOK-8	1	PASS

Some particularities to highlight a different behavior about the code needed to show the Tests Runs section:

- Ability to put the Tests with a particular status on top of the table (more info ahead).
- Usage of *${fullname:Tests[n].Assignee}*, this allows us to fetch the full name of the assignee instead of the key associated to it.

Test Executions

This section will gather all the information related to each Test Execution of each Test in the Test Plan with all the possible details.

It is composed with several sub-sections that will be filled with information if it is available or be filled with a message showing that no information is available.

Test Executions Summary

This section have a table with information regarding each Test Execution in this Test Plan (and will repeat these sections for each Test Execution). The information is presented as a table with the following fields:

Field	Description	Sample Code
Execution status	Execution Status of the Test Run	*${TestExecutions[n].TestRuns[a].Execution Status}*
Assignee	Full Name of the Assignee of the Test Run	*${fullname:TestExecutions[n].TestRuns[a].Assignee}*
Executed By	Full Name of the entity that has executed this Test Run	*${fullname:TestExecutions[n].TestRuns[a].Executed By}*
Started On	Timestamp of the Started Date from the TestRun	*${dateformat('dd-MM-yyyy HH:mm:ss'):TestExecutions[n].TestRuns[a].Started On}*
Finished On	Timestamp of the Finished Date from the TestRun	*${dateformat('dd-MM-yyyy HH:mm:ss'):TestExecutions[n].TestRuns[a]. Finished On}*
Versions	Fix Version field associated with the TestRun	*${TestExecutions[n].TestRuns[a].FixVersions}*
Revision	Revision assigned to the TestRun	*${TestExecutions[n].TestRuns[a].Revision}*

All of these fields have code to handle empty fields. The resulting table look like the one below.

2.1.1. BOOK-28: Test a visitor can filter the search result

Execution Status	Assignee	Executed By	Started On	Finished On	Versions	Revision
PASS	Xpand IT Admin	xadmin			1.0	-

Execution Defects

If any Defects was found and associated globally with a TesRun it will appear here in the form of a table with the following fields:

Key	Description	Sample Code
Key	Jira Key of the Defect in the form of a link	*@{title=${TestExecutions[n].TestRuns[a].ExecutionDefects[d].Key}\|href=${BaseURL}/browse/${TestExecutions[n].TestRuns[a].ExecutionDefects[d].Key}}*
Summary	Summary of the Defect	*${TestExecutions[n].TestRuns[a].ExecutionDefects[d].Summary}*
Priority	Priority associated with the defect	*${TestExecutions[n].TestRuns[a].ExecutionDefects[d].Priority}*

The table will be similar to the one below.

2.1.6.1. Execution Defects

Key	Summary	Priority
BOOK-32	Error when adding Book to shopping basket	Trivial

Execution Evidences

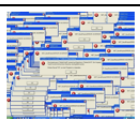
If any Evidence was attached to the TestRun we are showing it in table with the FileName and a screenshot if the Evidence is an image otherwise just a link.

To obtain that information we have used the following code:

Key	Description	Sample Code
File Name	The File Name of the Evidence attached to the Execution	*@{title=${TestExecutions[n].TestRuns[a].ExecutionEvidences[d].Name}\|href=${TestExecutions[n].TestRuns[a].ExecutionEvidences[d].FileURL}}*
Author	Author of the Evidence	*${TestExecutions[n].TestRuns[a].ExecutionEvidences[d].Author}*
File Size	File Size of the Evidence in bytes	*${TestExecutions[n].TestRuns[a].ExecutionEvidences[d].Size}*
Evidence	The Evidence attached to the Execution	*!{${TestExecutions[n].TestRuns[a].ExecutionEvidences[d].Evidence|maxwidth=100}}*

The table in case of an Evidence is of the type image will have the following aspect:

2.1.6.2. Execution Evidence

File Name	Author	File Size	Evidence
funny-error-messages.jpg	xadmin	411864	

Comment

The comment associated to the TestRun (*${wiki:TestExecutions[n].TestRuns[a].Comment}*).

Test Description

The description of the TestRun (*${wiki:TestExecutions[n].TestRuns[a].Description}*).

Test Issue Attachments

This section only appears if you have any attachments associated to the Test.

Key	Description	Sample Code

File Name	File Name of the Attachment	*@{title=${TestExecutions[n].TestRuns[a].Attachments[b].Name}\|href=${TestExecutions[n].TestRuns[a].Attachments[b].FileURL}}*
Author	The Author of the attachment	*${TestExecutions[n].TestRuns[a].Attachments[b].Author}*
File Size	File Size of the attachments in bytes.	*${TestExecutions[n].TestRuns[a].Attachments[b].Size}*

This appears in the document in a table form:

2.1.10.5. Test Issue Attachments

File Name	Author	File Size
template.txt	xadmin	749

Preconditions

This section only appear if you have a Precondition associated with the TestRun.

Key	Description	Sample Code
Key	Key of the Precondition	*@{title=${TestExecutions[n].TestRuns[a].PreConditions[c].Key}\|href=${BaseURL}/browse/${TestExecutions[n].TestRuns[a].PreConditions[c].Key}}*
Summary	Summary of the Precondition	*${TestExecutions[n].TestRuns[a].PreConditions[c].Summary}*
Condition	Condition field present in the Precondition	*${wiki:TestExecutions[n].TestRuns[a].PreConditions[c].Conditions}*

A sub section will appear with the preconditions definitions.

2.1.16.6. Preconditions

Key	BOOK-334
Summary	dummy precondition
Condition	do this

Parameters

This section lists the existing parameters of the TestRun (we are iterating through the Parameters of the TestRun with: *#{for m=TestExecutions[n].TestRuns[a].ParametersCount})*.

Key	Description	Sample Code
Name	Key of the parameter	*${TestExecutions[n].TestRuns[a].Parameters[m].Key}*
Value	Value of the parameter	*${TestExecutions[n].TestRuns[a].Parameters[m].Value}*

It will list the Key and the Value of each parameter in a table.

2.1.16.7. Parameters

There are 2 parameters on this Test Run.

Key	Value
password	somepass
username	somelogin

Iterations

This section uses a sentence to show how many interactions we will go into more details in the next sections.

Key	Description	Sample Code
Iterations	The iterations count of the Test Run	*${TestExecutions[n].TestRuns[a].IterationsCount}*

A sentence is added to the document with this information.

2.3.1.7. Iterations

This Test Run has 4 iterations.

Iteration Overall Execution Status

To obtain the overall execution status of the iteration we use two variables:

Key	Description	Sample Code
List of Statuses	Show the List of Statuses	*${TestExecutions[n].TestRuns[a].Iterations Overall Execution Status}*
TO DO	Overall Execution Status per Status	*${TestExecutions[n].TestRuns[a].Iterations Overall Execution Status.TO DO}*
EXECUTING		*${TestExecutions[n].TestRuns[a].Iterations Overall Execution Status.EXECUTING}*
PASS		*${TestExecutions[n].TestRuns[a].Iterations Overall Execution Status.PASS}*
FAIL		*${TestExecutions[n].TestRuns[a].Iterations Overall Execution Status.FAIL}*
ABORTED		*${TestExecutions[n].TestRuns[a].Iterations Overall Execution Status.ABORTED}*

The above code will produce the below table.

2.3.1.7.1. Iteration Overall Execution Status

List Of Statuses: PASS: 25.0%, FAIL: 25.0%, ABORTED: 0.0%, PENDING: 0.0%, EXECUTING: 25.0%, BLOCKED: 0.0%, TODO: 25.0%

TODO	25.0%
EXECUTING	25.0%
PASS	25.0%
FAIL	25.0%
ABORTED	0.0%

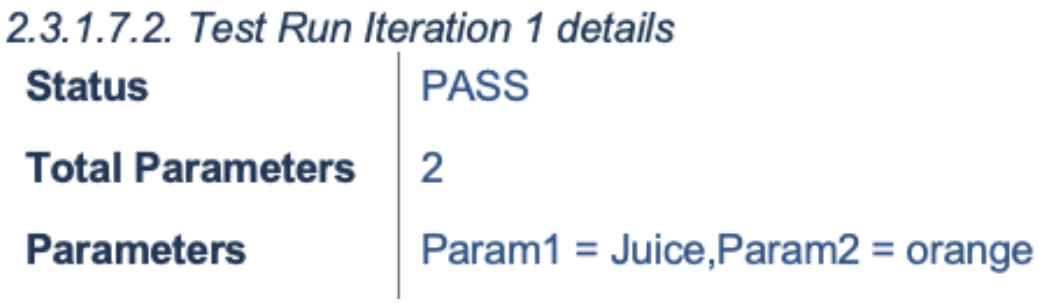
Test Run details

In this section we are showing the Test Run details with the Name, Status and Parameters.

We extract that information using the following fields:

Key	Description	Sample Code
Iteration Name	Name of the iteration	*${TestExecutions[n].TestRuns[a].Iterations[m].Name}*
Status	Status of the iteration	*${TestExecutions[n].TestRuns[a].Iterations[m].Status}*
Total Parameters	Total number of parameters	*${TestExecutions[n].TestRuns[a].Iterations[m].ParametersCount}*
Parameters	Lists all parameters in the form of Key=Value	*${TestExecutions[n].TestRuns[a].Iterations[m].Parameters}*

This section will have the below appearance:



Iteration precondition definition

If a precondition is present we will use the following fields to extract that information:

Key	Description	Sample Code
Key	Iteration precondition key	*${TestExecutions[n].TestRuns[a].Iterations[m].PreConditions[l].Key}*
Definition	Iteration precondition definition	*${wiki:TestExecutions[n].TestRuns[a].Iterations[m].PreConditions[l].PreCondition.Definition}*

This will produce an entry like the one below:

Iteration parameters details

For that given Iteration we are listing the parameters used, that information is extracted with the following fields:

Key	Description	Sample Code
Name	Parameter Key	*${TestExecutions[n].TestRuns[a].Iterations[m].Parameters[l].Key}*
Value	Parameter Value	*${TestExecutions[n].TestRuns[a].Iterations[m].Parameters[l].Value}*

It generates a table of the following form:

2.3.1.7.3. Iteration 1 Parameters details

Key	Value
Param1	Juice
Param2	orange

Iteration Test Step Details

In this section we are listing the details of an iteration, we are listing each step present with details, the code we use for that purpose i present in the below table.

Key	Description	Sample Code
Step	The Step Number	*${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].StepNumber}*
Action	Action defined in the Test Step	*${wiki:TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Action}*
Data	Data defined in the Test Step	*${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Data}*
Expected Result	Expected Result defined in the Test Step	*${wiki:TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].ExpectedResult}*
Attachments	Attachments present in each Test Step (showing the FileURL and a screenshot in case of the Attachment being an image)	*@{title=${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Attachments[sa].Name}\|href=${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Attachments[sa].FileURL}}* *!{${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Attachments[sa].Attachment|maxwidth=100}}*
Comment	Comment	*${wiki:TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Comment}*
Defects	Defects associated to this Iteration	*@{title=${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Defects[dc].Key}\|href=${BaseURL}/browse/${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Defects[dc].Key}}*
Evidence	FileURL and screenshot (if it is an image) of the Evidence	*@{title=${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Evidences[e].Name}\|href=${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Evidences[e].FileURL}} by ${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Evidences[e].Author} - ${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Evidences[e].Size}* *!${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Evidences[e].Evidence|maxwidth=100}*
Status	Test Step Status	*${TestExecutions[n].TestRuns[a].Iterations[m].TestSteps[r].Status}*

The above information is gathered in a table like the one below:

2.3.1.7.4. Iteration 1 Test Steps details
This Manual Test has 3 Steps

	Action	Data	Expected Result	Attachments	Comment	Defects	Evidences	Status
1	Step 1 Action		Expected Result 1	test_environment.docx 20221013_102722.jpg				PASS
2	Insert Juice		Juice is inserted					PASS
3	Inorange sert		orange is inserted					PASS

Test Details

This section shows the Test details, for that we are considering the different possible Test we can have in Xray: Generic, Manual and Cucumber. For each type we will fetch different information.

It may seem similar with the Iteration Test Step Details section but in this section we will show the Test details (not instantiated in each Iteration like the previous section).

Type	Key	Description	Sample Code	Output
Generic	Test Type	Test Type field	*${TestExecutions[n].TestRuns[a].TestType}*	*2.2.1.6. Test Details* **Test Type** | Generic **Specification** | com.xpand.java.CalcTest.CanAddNumbers
	Specification	Definition of the Generic test	*${TestExecutions[n].TestRuns[a].Generic Test Definition}*	
Cucumber	Test Type	Test Type field	*${TestExecutions[n].TestRuns[a].TestType}*	*2.4.2.6. Test Details* **Test Type** | Cucumber **Gherkin Specification** | Scenario Given I have done a search using a valid search term And I see a certain amount of items When I filter using a sub-keyword Then The results are filtered to display a smaller amount of ite
	Gherkin Specification	Gherkin specification of the Test	*${TestExecutions[n].TestRuns[a].Cucumber Scenario}* *#{for h=JQLIssuesCount\|clause=key in testExampleResult ('${TestExecutions[n].TestRuns[a].Key')}* *${TestExecutions[n].TestRuns[a].TestStepsResults[h]}* *#{end}*	
Manual	Step	Step Number	*${TestExecutions[n].TestRuns[a].TestSteps[r].StepNumber}*	*2.1.15.6. Test Details* This Manual Test has 3 Steps
	Action	Action of the Test Step	*${TestExecutions[n].TestRuns[a].TestSteps[r].Action}*	
	Data	Data of the Test Step	*${TestExecutions[n].TestRuns[a].TestSteps[r].Data}*	
	Expected Result	Expected Result of the Test Step	*${TestExecutions[n].TestRuns[a].TestSteps[r].ExpectedResult}*	
	Attachment	Attachment of the Test Step	*${TestExecutions[n].TestRuns[a].TestSteps[r].Attachments[sa].FileURL}*	
	Comment	Comment of the Test Step	*${wiki:TestExecutions[n].TestRuns[a].TestSteps[r].Comment}*	
	Defects	Defects associated with the Test Step	*${TestExecutions[n].TestRuns[a].TestSteps[r].Defects[dc].Key}*	
	Evidence	Evidence with the Test Step	*${TestExecutions[n].TestRuns[a].TestSteps[r].Evidences[e].FileURL}}* *!${TestExecutions[n].TestRuns[a].TestSteps[r].Evidences[e].Evidence|maxwidth=100}*	

	Action	Data	Expected Result	Attachments	Comment	Defects	Evidences	Status
1	Open the App, choose option **My Profile > Logout**.							PASS
2	On the login screen, select option **I lost my password**.							PASS
3	On the email field, provide the address data and press **Reset my Password.**	Email: user@test.com	Information message: "Check your mailbox for further instructions"					PASS

	Status	Status of the Test Step	*${TestExecutions[n].TestRuns[a].TestSteps[r].Status}*

Requirements linked with this test

For each Test we are listing the Requirements linked

Key	Description	Sample Code
Requirement Key	Key of the Requirement	*${TestExecutions[n].TestRuns[a].Requirements[x].Key}*
Requirement Summary	Summary of the Requirement	*${wiki:TestExecutions[n].TestRuns[a].Requirements[x].Summary}*
Workflow Status	Workflow status of the Requirement	*${TestExecutions[n].TestRuns[a].Requirements[x].Status}*

This section present a table with that information like the one below:

2.3.1.8. Requirements linked with this test

Requirement Key	Requirement Summary	Workflow Status
ATLAS-2	As a user, I want to be enforced to have a strong password so my account is properly secured	In Progress
BOOK-507	dummy user story	Open

Appendix A: Approval

This section is added for the cases where you need to have a signature validating the document.

3. Appendix A: Approval

The undersigned acknowledge they have reviewed the **Test Plan** and agree with the approach it presents. Changes to this **Test Plan** will be coordinated with and approved by the undersigned or their designated representatives.

Signature: *Date:*

Print Name:

Title:

Role:

Customizing the report

Sections that can be hidden or shown

The report has some variables/flags that can be used to show or hide some sections whose logic is already implemented in the template.

These variables are defined at the top of each sheet, at the report template; the variables are scoped just to the current sheet.

On the template, use one of these values for flag type of variables:

- **0**: to hide a section
- **1**: to show a section

The format for other types of variables is detailed ahead.

> **Example of setting a variable to, in this case, render information on the section "Test Executions"**
>
> ${set(showTestRunDetails, 1)}

Variable/flag	Purpose	default	example(s)
showTestRunDetails	render the details section • format: 0 or 1	0	*${set(showTestRunDetails, 0)}*
showTestRunEvidences	render this section • format: 0 or 1 (Make sure to define **showTestRunDetails** at 1)	0	*${set (showTestRunEvidences, 0)}*
showTestRunAttachments	render this section • format: 0 or 1 (Make sure to define **showTestRunDetails** at 1)	0	*${set (showTestRunAttachments, 0)}*
showTestRunIterations	render this section • format: 0 or 1 (Make sure to define **showTestRunDetails** at 1)	0	*${set (showTestRunIterations, 0)}*
statusesToShowFirst	render Test Summary section whose reported status is one in this list first (delimited by comma); use an empty string '' to include all statuses • format: '<status1>,<status2>"	'' (i.e., all statuses)	*${set(statusesToInclude, 'FAILED')}* *${set(statusesToInclude, 'FAIL,EXECUTING')}* *${set(statusesToInclude, '')}*

Adding or removing information to/from the report

As this report is a document with different sections, if some sections are not relevant to you, you should be able to simply delete them. Make sure that no temporary variables are created in that section that are used in other subsequent sections or if any all conditional blocks are properly closed.

To add additional information, usually we're thinking of adding:

- fields of the Test Plan itself
- fields of the Tests associated with the Test Plan
- fields of the Test Executions associated with the Test Plan

Eventually, also:

- fields of the Test Runs(s)
- fields of the covered issue(s) associated with the Test that is associated with the Test Plan

The later may be harder to implement, so we won't consider them here.

Exercise: add a field from the related Test issue

Let's say we have a "Severity" field on the Test issue that is connected to the Test Plan, and that we want to show it on the report.

We can copy the column "Summary" from the "Tests Summary" section and adapt it.

1. insert new column in the table
2. on the "Tests Summary" section,
    a. copy "Summary" (i.e., insert a column next to it and copy the values from the existing "Summary" column)
    b. change
        i. ${Tests[n].Summary} to ${Tests[n].Severity}

Performance

Performance can be impacted by the information that is rendered and by how that information is collected/processed.

The number of Test Runs and Tests depending on scenarios, can be considerably high, especially with CI/CD. As this report sum-up quite information, please use it wisely.

Data-driven tests may also add an overhead, as iterations need to be individually processed, for collecting all the reported/linked defects for example.

Some tips

- Use the variables/flags to adjust sections or the Test Plan that will be processed/shown in the report; more info in "Customizing the report"
- limit the number of input issues; in Xporter there's a global setting for this purpose

Known limitations

- Test Plan comments are not formatted
- Gherkin Scenario Outlines are not considered as data-driven (i.e., only one Test Run will appear)