Document Generator Template: Test Execution Report

Purpose

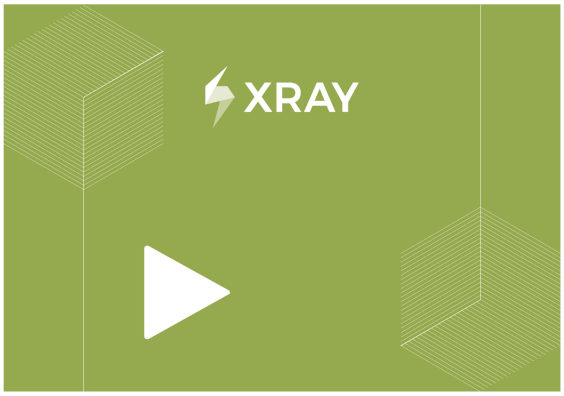
This report enables you to extract details of a Test Execution, such as the Tests that are part of it, Defects, Requirements and iterations details, so that you can generate a document report focusing in what matter the most for your team, or even share it with someone else that hasn't access to Jira.

Possible usage scenarios:

- see all the requirements covered by the Tests in the Test Execution
- see all the defects linked to the Tests in this Test Execution
- see an overall status summary of the Test Execution
- check a specific detail of a Test Runs (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-31
Prepared By: Xpand IT Admin

Document Date: 28/11/2022

CONFIDENTIAL

TEST EXECUTION

Test Execution for Test Plan BOOK-30

1. Introduction

1.1. Document Overview

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

1.2. Test Execution BOOK-31 Details

Description	*None*
Begin Date	18-02-2019 14:03:00
End Date	22-02-2019 14:03:00
Revision	65d64uyg7t6r
Test Environments	[chrome, windows]
Test Plan	BOOK-30

1.3. Requirements covered by this Test Execution

Key	Summary	Workflow Status	Test Status
BOOK-7	As a visitor, I can search for books in the store	Resolved	NOK
BOOK-11	As a visitor, I can view all the books in my shopping basket	Resolved	OK
BOOK-8	As a visitor, I can manage my Shopping Basket	Open	NOK
BOOK-10	As a visitor, I can remove books from my shopping basket	Resolved	OK

1.4. Overall Execution Status

Of the 16 Test Runs contained on BOOK-31:

Status	#TestRuns	Percentage
TODO	1	6.2 %
EXECUTING	0	0.0 %
PASS	13	81.3 %
FAIL	2	12.5 %
ABORTED	0	0.0 %

1.6. Test Runs

Rank	Key	Summary	Test Type	#Req	#Def	Test Sets	Assignee	Status
1	BOOK-28	Test a visitor can filter the search result	Cucumber	1	0	BOOK-508	xadmin	PASS
2	BOOK-27	Test a visitor can do a valid search with multiple keywords	Cucumber	1	0		xadmin	PASS
3	BOOK-26	Test a visitor can do a valid search with a single keyword	Cucumber	1	0		xadmin	PASS
4	BOOK-25	Test a visitor can view all the books in his shopping basket	Cucumber	2	0		xadmin	PASS
5	BOOK-24	Test visitors can remove books from their shopping basket	Cucumber	2	0		xadmin	PASS
6	BOOK-23	Test visitors can add books to their shopping basket	Cucumber	2	1		xadmin	FAIL
7	BOOK-22	Test visitors can navigate to the book details page	Cucumber	1	0		xadmin	PASS
8	BOOK-20	Test Checkout with incorrect delivery details	Cucumber	1	0		xadmin	PASS
9	BOOK-19	Test a visitor can checkout items in his basket	Cucumber	1	0		xadmin	PASS
10	BOOK-18	Test a visitor can change his locale	Cucumber	1	0	BOOK-29	xadmin	PASS
11	BOOK-17	Test a logged in visitor can edit the default address	Cucumber	2	0	BOOK-29	xadmin	PASS
12	BOOK-16	Test a logged in visitor can edit the account details	Cucumber	2	0	BOOK-29	xadmin	PASS
13	BOOK-15	Test logged in visitors can logout from their account	Cucumber	2	0	BOOK-29	xadmin	PASS

2. Test Run Details

2.1. Bookstore / Test Execution: BOOK-31 / Test: BOOK-28 - Test a visitor can filter the search result

Execution Status	Assignee	Executed By	Started On	Finished On	Versions	Revision
PASS	xadmin	xadmin	25-02-2019 14:07:47	25-02-2019 14:07:47	1.0	65d64uyg7t6r

2.1.1. Execution Defects

There are 0 Execution Defects on this Test Run.

2.1.2. Execution Evidences

There are 0 Execution Evidences on this Test Run.

2.6. Bookstore / Test Execution: BOOK-31 / Test: BOOK-23 - Test visitors can add books to their shopping basket

Execution Status	Assignee	Executed By	Started On	Finished On	Versions	Revision
FAIL	xadmin	xadmin	25-02-2019 14:10:46	25-02-2019 14:10:46	1.0	65d64uyg7t6r

2.6.1. Execution Defects

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

2.6.2. Execution Evidences

File Name	Author	File Size	Evidence
funny-error-messages.jpg	xadmin	402 kB	

2.6.3. Comment

Error when trying to add book with title "There is something strange with this book $%&" to the shopping basket

2.6.4. Test Description

None.

3. Appendix A: Approval

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

Signature: *Date:*

Print Name:

Title:

Role:

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 Execution 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 Set", "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 Execution

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

Understanding the report

The report shows detailed information about the Test Execution provided.

Layout

The report is composed by several sections. Two major sections are available: Introduction and Test Runs 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 Execution Details
3. Requirements covered by the Tests in this Test Execution
4. Overall Execution Status
5. Defects
6. Test Runs

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

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 Execution BOOK-31 Details

Description	*None*
Begin Date	18-02-2019 14:03:00
End Date	22-02-2019 14:03:00
Revision	65d64uyg7t6r
Test Environments	[chrome, windows]
Test Plan	BOOK-30

Requirements covered by the Tests in this Test Execution

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=${TestRuns[n].Requirements[x].Key}\|href=${BaseURL}/browse/${TestRuns[n].Requirements[x].Key}}*
Summary	Summary of the Requirement	*${wiki:TestRuns[n].Requirements[x].Summary}*
Workflow Status	Workflow Status of the Requirement	*${TestRuns[n].Requirements[x].Status}*
Coverage Status	Status of the Test coverage of the Requirement	*${TestRuns[n].Requirements[x].Requirement Status}*

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

1.3. Requirements covered by this Test Execution

Key	Summary	Workflow Status	Coverage Status
BOOK-7	As a visitor, I can search for books in the store	Resolved	NOK
BOOK-11	As a visitor, I can view all the books in my shopping basket	Resolved	OK
BOOK-8	As a visitor, I can manage my Shopping Basket	Open	NOK
BOOK-10	As a visitor, I can remove books from my shopping basket	Resolved	OK

Overall Execution Status

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

To obtain this information we are using:

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

This will produce the following output:

1.4. Overall Execution Status

Of the 16 Test Runs contained on BOOK-31:

Status	#TestRuns	Percentage
TODO	1	6.2 %
EXECUTING	0	0.0 %
PASS	13	81.3 %
FAIL	2	12.5 %
ABORTED	0	0.0 %

Defects

In this section we are listing all the defects found that are associated with this Test Execution, 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=${TestRuns[n].ExecutionDefects[d].Key}\|href=${BaseURL}/browse/${TestRuns[n].ExecutionDefects[d].Key}}*
		• TestSteps
		  ◦ *@{title=${TestRuns[n].TestSteps[j].Defects[m].Key}\|href=${BaseURL}/browse/${TestRuns[n].TestSteps[j].Defects[m].Key}}*
		• Iteration TestSteps
		  ◦ *@{title=${TestRuns[n].Iterations[it].TestSteps[r].Defects[dc].Key}\|href=${BaseURL}/browse/${TestRuns[n].Iterations[it].TestSteps[r].Defects[dc].Key}}*

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

Priority	Priority of the Defect	• TestRun
		  ○ *${TestRuns[n].ExecutionDefects[d].Priority}*
		• TestSteps
		  ○ *${TestRuns[n].TestSteps[j].Defects[m].Priority}*
		• Iteration TestSteps
		  ○ *${TestRuns[n].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 Execution.

1.5. Defects

Key	Summary	Priority
BOOK-32	Error when adding Book to shopping basket	Trivial
BOOK-510	Iteration global defect added	Trivial
BOOK-502	global defect	Trivial

Test Runs

In this section we have a table with information regarding the Test Runs in this Test Execution. You can find the following information about each Test Run:

Field	Description	Sample Code
Rank	Rank of the Test Step	*${TestRuns[n].Rank}*
Key	Key of the Test Run	*${TestRuns[n].Key}*
Summary	The Summary of the Test Run	*${wiki:TestRuns[n].Summary}*
Test Type	The type of Test that was executed in this Test Run	*${TestRuns[n].Test Type}*
#Req	Number of Requirements associated to this Test Run	*${TestRuns[n].RequirementsCount}*
#Def	Calculation of the number of Defects associated to this Test Run	*%{var total=${TestRuns[n].ExecutionDefectsCount} + ${TestRuns[n].TestStepsDefectsCount}; var total2=''+total;var totalParts= total2.split('.'); totalParts[0];}*
Test Sets	Test Set Key (if this Test Run was part of a Test Set)	*${TestRuns[n].TestSets[ts].Key}*
Assignee	Full name of the Assignee	*${fullname:TestRuns[n].Assignee}*
Status	Status of the Test Run	*${TestRuns[n].Execution Status}*

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

1.6. Test Runs

Rank	Key	Summary	Test Type	#Req	#Def	Test Sets	Assignee	Status
1	BOOK-28	Test a visitor can filter the search result	Cucumber	1	0	BOOK-508	xadmin	PASS
2	BOOK-27	Test a visitor can do a valid search with multiple keywords	Cucumber	1	0		xadmin	PASS
3	BOOK-26	Test a visitor can do a valid search with a single keyword	Cucumber	1	0		xadmin	PASS
4	BOOK-25	Test a visitor can view all the books in his shopping basket	Cucumber	2	0		xadmin	PASS
5	BOOK-24	Test visitors can remove books from their shopping basket	Cucumber	2	0		xadmin	PASS
6	BOOK-23	Test visitors can add books to their shopping basket	Cucumber	2	1		xadmin	FAIL
7	BOOK-22	Test visitors can navigate to the book details page	Cucumber	1	0		xadmin	PASS
8	BOOK-20	Test Checkout with incorrect delivery details	Cucumber	1	0		xadmin	PASS
9	BOOK-19	Test a visitor can checkout items in his basket	Cucumber	1	0		xadmin	PASS
10	BOOK-18	Test a visitor can change his locale	Cucumber	1	0	BOOK-29	xadmin	PASS

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

- Usage of *${fullname:Tests[n].AssigneeId}*, this allows us to fetch the full name of the assignee instead of the key associated to it.

Test Run Details

This section will gather all the information related to each Test Run of each Test in the Test Execution 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 Run in this Test Execution (and will repeat these sections for each Test Run). The information is presented as a table with the following fields:

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

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

2. Test Run Details

2.1. Bookstore / Test Execution: BOOK-31 / Test: BOOK-28 - Test a visitor can filter the search result

Execution Status	Assignee	Executed By	Started On	Finished On	Versions	Revision
PASS	xadmin	xadmin	25-02-2019 14:07:47	25-02-2019 14:07:47	1.0	65d64uyg7t6r

Execution Defects

If any Defects was found and associated globally with a Test Run 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=${TestRuns[n].ExecutionDefects[d].Key}\|href=${BaseURL}/browse/${TestRuns[n].ExecutionDefects[d].Key}}*
Summary	Summary of the Defect	*${wiki:TestRuns[n].ExecutionDefects[d].Summary}*
Priority	Priority associated with the defect	*${TestRuns[n].ExecutionDefects[d].Priority}*

The table will be similar to the one below.

2.15.1. Execution Defects

Key	Summary
BOOK-510	Iteration global defect added

Execution Evidences

If any Evidence was attached to the TestRun we are showing it in table with the FileName.

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

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

2.16.2. Execution Evidences

File Name	Author	File Size	Evidence
Untitled.svg	xadmin	4 kB	

Comment

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

Test Description

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

Test Issue Attachments

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

Key	Description	Sample Code
File Name	File Name of the Attachment	*@{title=${TestRuns[n].Attachments[a].Name}\|href=${TestRuns[n].Attachments[a].FileURL}}*
Author	The Author of the attachment	*${fullname:TestRuns[n].Attachments[a].Author}*
File Size	File Size of the attachments in bytes.	*${TestRuns[n].Attachments[a].Size}*

This appears in the document in a table form:

2.10.5. 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=${TestRuns[n].PreConditions[c].Key}\|href=${BaseURL}/browse/${TestRuns[n].PreConditions[c].Key}}*
Summary	Summary field present in the Precondition	*${wiki:TestRuns[n].PreConditions[a].Summary}*
Conditions	Condition present in the Precondition	*${wiki:TestRuns[n].PreConditions[a].Conditions}*

A sub section will appear with the preconditions definitions.

2.16.6. 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=TestRuns[n].ParametersCount})*.

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

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

2.16.7. 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	*${TestRuns[n].IterationsCount}*

A sentence is added to the document with this information.

2.16.7. 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	*${TestRuns[n].Iterations Overall Execution Status}*
TO DO	Overall Execution Status per Status	*${TestRuns[n].Iterations Overall Execution Status.TODO}*
EXECUTING		*${TestRuns[n].Iterations Overall Execution Status.EXECUTING}*
PASS		*${TestRuns[n].Iterations Overall Execution Status.PASS}*
FAIL		*${TestRuns[n].Iterations Overall Execution Status.FAIL}*
ABORTED		*${TestRuns[n].Iterations Overall Execution Status.ABORTED}*

The above code will produce the below table.

2.16.7.1. Iteration Overall Execution Status

List Of Statuses: PASS: 25.0%, FAIL: 0.0%, ABORTED: 0.0%, PENDING: 0.0%, EXECUTING: 25.0%, BLOCKED: 0.0%, TODO: 50.0%

Status	Percentage
TODO	50.0%
EXECUTING	25.0%
PASS	25.0%
FAIL	0.0%
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	*${TestRuns[n].Iterations[m].Name}*
Status	Status of the iteration	*${TestRuns[n].Iterations[m].Status}*
Total Parameters	Total number of parameters	*${TestRuns[n].Iterations[m].ParametersCount}*
Parameters	Lists all parameters in the form of Key=Value	*${TestRuns[n].Iterations[m].Parameters}*

This section will have the below appearance:

2.16.7.2. Test Run Iteration 1 details

Status	PASS
Total Parameters	2
Parameters	username = somelogin,password = somepass

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	*${TestRuns[n].Iterations[m].PreConditions[l].Key}*
Definition	Iteration precondition definition	*${wiki:TestRuns[n].Iterations[m].PreConditions[l].Conditions}*

This will produce an entry like the one below:

2.16.7.2.1. Iteration Precondition BOOK-334 Definition

do this

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	*${TestRuns[n].Iterations[m].Parameters[l].Key}*
Value	Parameter Value	*${TestRuns[n].Iterations[m].Parameters[l].Value}*

It generates a table of the following form:

2.16.7.2.2. Iteration 1 Parameters details

Key	Value
password	somepass
username	somelogin

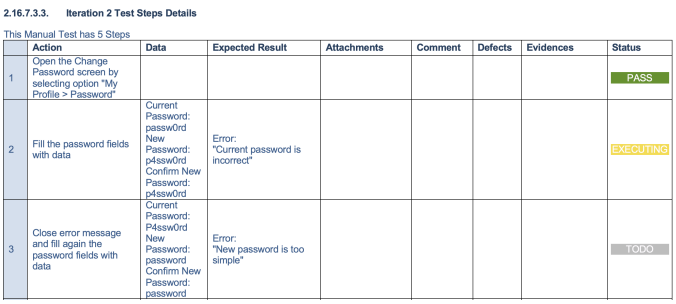
Iteration Test Step Details

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 is present in the below table.

Key	Description	Sample Code

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

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



2.16.7.3.3. Iteration 2 Test Steps Details

This Manual Test has 5 Steps

	Action	Data	Expected Result	Attachments	Comment	Defects	Evidences	Status
1	Open the Change Password screen by selecting option "My Profile > Password"							PASS
2	Fill the password fields with data	Current Password: passw0rd New Password: p4ssw0rd Confirm New Password: p4ssw0rd	Error: "Current password is incorrect"					EXECUTING
3	Close error message and fill again the password fields with data	Current Password: P4ssw0rd New Password: password Confirm New Password: password	Error: "New password is too simple"					TODO

Test Details

This section shows the Test details, for that we are considering the different possible Tests 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	*${TestRuns[n].Test Type}*	**2.2.6. Test Details** **Test Type** Generic **Specification** [chromium] › login.spec.ts:14:5 › Login validations › Login with invalid credentials.Login validations Login with invalid credentials
	Specification	Definition of the Generic test	*${TestRuns[n].Generic Test Definition}*	
Cucumber	Test Type	Test Type field	*${TestRuns[n].Test Type}*	

	Gherkin Specification	Gherkin specification of the Test	*${TestRuns[n].Cucumber Scenario}*

2.1.6. Test Details

Test Type	Cucumber
Gherkin Specification	Given I want to see the gherkin scenarios Then I will see something in the report When I will push the export button

Manual	Step	Step Number	*${TestRuns[a].TestSteps[r].StepNumber}*
	Action	Action of the Test Step	*${TestRuns[a].TestSteps[r].Action}*
	Data	Data of the Test Step	*${TestRuns[a].TestSteps[r].Data}*
	Expected Result	Expected Result of the Test Step	*${TestRuns[a].TestSteps[r].ExpectedResult}*
	Attachments	Attachment of the Test Step	*${TestRuns[a].TestSteps[r].Attachments[sa].FileURL}* *!{${TestRuns[n].TestSteps[r].Attachments[sa].Attachment}|width=100}*
	Comment	Comment of the Test Step	*${wiki:TestRuns[a].TestSteps[r].Comment}*
	Defects	Defects associated with the Test Step	*${TestRuns[a].TestSteps[r].Defects[dc].Key}*
	Evidence	Evidence with the Test Step	*${TestRuns[a].TestSteps[r].Evidences[e].FileURL}}* *!${TestRuns[a].TestSteps[r].Evidences[e].Evidence|maxwidth=100}*

2.15.7. Test Details

This Manual Test has 3 Steps

	Action	Data	Expected Result	Attachments	Comment	Defects	Evidence	Status
1	Open the App, choose option **My Profile > Logout**.							TODO
2	On the login screen, select option **I lost my password**.							TODO
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"					TODO

	Status	Status of the Test Step	*${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	*@{title=${TestRuns[n].Requirements[x].Key}\|href=${BaseURL}/browse/${TestRuns[n].Requirements[x].Key}}*
Requirement Summary	Summary of the Requirement	*${wiki:TestRuns[n].Requirements[x].Summary}*

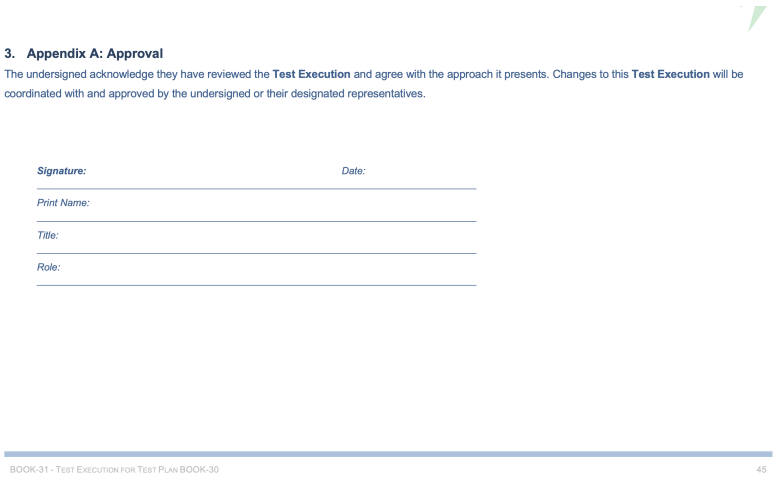
This section present a table with that information like the one below:

2.15.8. Requirements linked with this test

Requirement Key	Requirement Summary
BOOK-39	As a user, I want to reset my password so I can regain access when I forget it
BOOK-40	Account Security

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 Execution** and agree with the approach it presents. Changes to this **Test Execution** will be coordinated with and approved by the undersigned or their designated representatives.

Signature: *Date:*

Print Name:

Title:

Role:

BOOK-31 - Test Execution for Test Plan BOOK-30 45

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)}*

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 Execution itself
- fields of the Tests associated with the Test Execution
- fields of the Test Runs associated with the Test Execution

Eventually, also:

- fields of the Test Plan (if associated to any)
- fields of the covered issue(s) associated with the Test that is associated with the Test Execution

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 Defect that is connected to the Test Execution, and that we want to show it on the report.

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

1. insert new column in the table
2. on the "Tests Details" section,
    a. copy "Comment" (i.e., insert a column next to it and copy the values from the existing "Comment" column)
    b. change
        i. *${wiki:TestRuns[n].TestSteps[r].Comment} to ${TestRuns[n].TestSteps[r].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 Execution comments are not formatted
- Gherkin Scenario Outlines are not considered as data-driven (i.e., only one Test Run will appear)