Tampere University Unit of Computing Sciences

COMP.SE.610 Software Engineering Project 1

COMP.SE.620 Software Engineering Project 2

Who is reading Test Report ?

For example

* project manager
* developer company quality manager
* customer.

So this is quite an important document. Groups should think what kind of document structure fits best for their project. You may also ask your coach.

Two last pages (not part of deliverable Test Report) give some guidelines to project groups about the contents.

(and remove this box before document delivery ) 😉

Group name/number

Project name

Test Report

Note: All texts in this document that are coloured blue are instructive and should be replaced with actual text by you. They just tell what should be included in each section.

|  |
| --- |
| student name |
| student namestudent namestudent namestudent namestudent namestudent name |

Version history

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description |
| 0.1 | 17.09.2013 | Tommy Testmanager | First draft |
| 0.2 | 20.09.2013 | Tanya Testmanager | Comments included |
| 0.3 | 09.11.2017 | Tim Testmanager | Section 2.1  |
| 0.4 | 10.11.2020 | Ted Hill  | cosmetic TUNI changes  |
| 0.6 | 12.11.2020 | Ted Hill | added guidelines  |
| 0.7 | 18.08.2021  | Ted Hill  | course numbers updated  |
| 0.9 | 23.01.2024  | tp  | Minor fixes to the cover page  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

1 Introduction 4

1.1 Purpose and scope of document 4

1.2 Product and environment 4

1.3 Project constraints related to testing 4

1.4 Definitions, abbreviations and acronyms 4

2 Testing process 5

2.1 General approach 5

2.2 Testing roles 5

2.3 Test schedule 5

2.4 Test documentation 5

3 Testing Tools 5

4 Test cases and results 5

4.1 Test results 6

4.2 Unit testing 6

4.3 Integration testing 6

4.4 System testing 6

4.5 Special testing 6

4.6 Acceptance testing 6

4.7 Xyz (of your choice..) 6

APPENDIX A […Z] 6

# Introduction

## Purpose and scope of document

Purpose of this document, what testing it covers (unit, integration, system, acceptance, etc.). Usually Test Report do not cover unit and integration testing.

The purpose of Test Documentation is to make clear what have been tested, when, and by who. Group itself also knows what and how to test.

In overall, Test Report gives stakeholders a view (they may have not seen Test Plan) about e.g.

* what was tested (e.g. GUI, usability, functional requirements, DB),
* how was tested (e.g. exploratory testing, system test cases),
* how much was tested (e.g. 42 system test cases),
* who tested (e.g. UX-testing group, three system testers, and four end-users),
* when testing took part (e.g. started three weeks before deployment, lasted 14 working days),
* what kind of test data was used (e.g. customer’s actual data sets)
* what was result of testing (e.g. 33 bugs were found, 30 fixed, three minor class bugs still open).

## Product and environment

Product name and the environment in general.

##  Project constraints related to testing

Such as technology constraints, standards.

Special practices and/or tools set by the customer

 Confidentiality

 Cybersecurity

 Licensing issues

 User testing.

##  Definitions, abbreviations and acronyms

Exploratory testing no test cases but story to follow

Test testing of one requirement, functionality or feature

Test case everything required to one test

Test suite a set of tests and test cases, e.g. for usability testing.

# Testing process

Do you have some kind of coding conventions (code style guide) or quality guidelines.

## General approach

Describe your general strategy for testing: how will testing be done at various levels (unit, integration, system, acceptance).

Are there customer acceptance tests ?

Will there be analytic assessments (such as usability assessment) ?

Do you use error classification (e.g. cosmetic/minor, medium, fatal/major) ?

##  Testing roles

List the roles of personnel related to testing. Remember other parties as well (e.g. customer, end-users).

E.g. test manager, front-end and back-end test group, if exists.

## Test schedule

Describe here your project’s testing schedule; starting, execution and ending time of tests.

Time used for bug fixing and re-testing.

Planned vs. actual.

##  Test documentation

Describe what kind of documents exists from testing. E.g. version control issues lists, test log, test diary, automated unit testing tool reports, listing from external bug database.

# Testing Tools

Describe your tools used in testing, planned and actually used.

What kind of statistics was logged and observed ?

# Test cases and results

Describe here in the following sections 4.x all testing.

In each section, make clear the numbers of test cases; passed, failed, ignored/removed, and if any bugs are still open (not resolved).

## Test results

Write here overall result of system/acceptance testing, so that casual reader easily understands whether the product is tested well, and did it pass.

## Unit testing

## Integration testing

## System testing

Actual end-users involved in system testing ?

Exploratory testing ?

## Special testing

Describe any special testing tasks that require planning. Those include e.g. usability testing, security testing, browser / hardware compatibility testing.

## Acceptance testing

Usually done by customer. If known, explain results here.

## Xyz (of your choice..)

Add more sections (4.x) if needed.

# APPENDIX A […Z]

Include appendices if needed.

E.g. templates / final documents about

- system test logs / diaries

- user test scripts

- acceptance test report.

**GROUPS SHALL THEMSELVES MODIFY THE FORMAT OF TEST REPORT TO GIVE READER VITAL INFORMATION ABOUT PROJECT’S TESTING** (AND QUALITY ASSURANCE) **PROCESS AND ACTIONS.**

- testing goals (at different testing types / levels / methods)

 [ *softwaretestingfundamentals.com/* ]

- test data used (e.g. self-made, or from customer)

- how many testing sessions

- what have been tested (e.g. backend, client(s), DB, GUI/UX, security,…)

- when was tested (start..end), perhaps a timeline chart would be good

- how many testers (in different groups) and who tested

- how many end-users participated, in how many tests, for how long

- results from automated unit testing (e.g. CI/CD pipeline)

- static code analyser results (e.g. SonarQube)

- about (system) test cases (tests, suites), in general results/summary

- how many planned

- how many done

- how many passed

- how many failed

- how many rejected / skipped (omitted), and why

- list of still open issues (bugs), if any.

**Some examples about what stakeholders would NOT like to see in Test Report (as such information does not give much actual data… only indicates a poor testing process):**

* we planned automated unit testing as CI pipeline, but we didn’t had time for that
* we planned to involve end-users in usability testing, but finally we had no time
* the software crashed in the morning when we planned to start system testing, so it is not tested, but we believe it should work
* test manager’s SSD disk broke, so we lost all test data
* We did four hours of exploratory testing, and we did not find any bugs, so the quality is good. We did not document those tests.
* we did 42 systems tests, so testing was comprehensive
* there are still some open issues, but we will fix those before delivering software to customer
* There are some open issues in our GitLab. These will be solved before the final submission.
* Open issue left is one bug that was not fixed in time.
* At this time, there are no issues with the project, except for various kinds of bugs that were either undiscovered or discovered at the very end.