

TEST PLAN TEMPLATE

[Type the document subtitle/type]

[Type the company name]

Version [X.X]

Contents

[**Revision history**](#_vv2i44mjlzgr) **3**

[**Introduction**](#_30j0zll) **4**

[Purpose](#_1fob9te) 4

[Terminology](#_3znysh7) 4

[References](#_2et92p0) 4

[Overview](#_tyjcwt) 4

[**Test items**](#_3dy6vkm) **5**

[Features to be tested](#_1t3h5sf) 5

[Features not to be tested](#_4d34og8) 6

[**Test strategy**](#_2s8eyo1) **6**

[Test approach](#_17dp8vu) 6

[Test types](#_3rdcrjn) 7

[Test coverage](#_26in1rg) 8

[QC Activities workflow](#_lnxbz9) 8

[Test planning](#_35nkun2) 10

[Test planning exit criteria](#_1ksv4uv) 10

[Test planning process](#_44sinio) 10

[Test planning stop/continue criteria](#_2jxsxqh) 11

[Test planning exit criteria](#_z337ya) 11

[Test design](#_3j2qqm3) 11

[Test design entry criteria](#_4i7ojhp) 11

[Test design process](#_2xcytpi) 11

[Test design stop/continue criteria](#_1ci93xb) 12

[Test design exit criteria](#_3whwml4) 12

[Test Execution](#_2bn6wsx) 12

[Test execution entry criteria](#_3as4poj) 12

[Test execution process](#_1pxezwc) 13

[Defect/Issue management process](#_49x2ik5) 14

[Defect registration](#_r9j82cpp1olt) 14

[Defect verification](#_gr9xm1pjkhb) 17

[Test execution stop/continue criteria](#_2p2csry) 17

[Test execution exit criteria](#_3o7alnk) 17

[Test environment](#_1hmsyys) 18

[Configurations](#_2grqrue) 18

[Testing tools](#_vx1227) 18

[Test deliverables](#_1v1yuxt) 18

[**Roles and Responsibilities**](#_4f1mdlm) **19**

[**Risks**](#_19c6y18) **20**

[**Schedule**](#_3tbugp1) **21**

# Revision history

| **Date** | **Version** | **Description** | **Author** |
| --- | --- | --- | --- |
| MM/DD/YYYY | <x.x> | <details> | <Name> |
| MM/DD/YYYY | <x.x> | Approved | <Name> |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

*[State the purpose of the test plan, possibly identifying the level of the plan (master etc.).]*

[Example:

The purpose of this document is to work out in detail the QC activities required to perform for <Project Name> project; to define the test strategy and approach to testing; to define the scope of the QC activities and to identify responsibilities. Provide reference documents and guidelines to perform the QC activities; provide the standards, practices and conventions used in carrying out QC activities; and provide the tools, techniques, and methodologies to support QC activities, and reporting.]

## Terminology

| **Term** | **Definition** |
| --- | --- |
| QC | Quality Control |
| QCE | Quality Control Engineer |
| PM | Project Manager |
| PO | Product Owner |
|  |  |

## References

| **#** | **Version** | **Date** | **Name** | **Published by** |
| --- | --- | --- | --- | --- |
| 1 | <X.X> | MM/DD/YYYY | <Document name> | <Publishing organization> |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Overview

*[Describe briefly the purpose of the system/application to which this test plan applies. Overview business problems and benefits from project’s implementation. This information can be copied from project plan, requirement specification or any other document describing the product.]*

[Example:

The application allows user to display producer fields on a map, associate them with all the officially registered land units considering location of the fields, their boundaries and sizes.

The application provides the interface for custom (not registered yet) fields creation and management.

Additionally, the application provides the possibilities to form and print brochures with maps and producer fields on it in .pdf format but different sizes and quality as handouts for producers.]

# Test items

*[Identify test items ( the parts of the system/application that can include numbers of features) intended to be tested within the scope of this test plan. Test items level of detalization depend on level of test plan]*

[Example:

The following essential system components are to be tested:

1. Application
2. Documentation Products (.pdf brochures generated by application)
3. Integration with official sources of land units information]

## Features to be tested

*[ List the features to be tested. Specify a test item (if there are several test items) the feature belongs to. Describe features using user terminology, not technical ones.]*

[Example:

The main testing areas are listed below:

1. Application:

* Feature 1;
* ……………...
* Feature N;

1. Documentation Products:

* Feature 1;
* ……………...
* Feature N;

1. Integration with official sources of land units information:

* Feature 1;
* ……………...
* Feature N;

Actual testing scope will be defined for each iteration accordingly to the scope of particular iteration]

## Features not to be tested

*[ List the features which are out of testing scope for this project. Identify the reason why the feature is not to be tested.*

*Describe features using user terminology, not technical ones.]*

[Example:

The following features will not be tested:

* The correctness of data provided by official sources of land units information;
* Base map is provided by BING.]

# Test strategy

*[Describe the overall test strategy for this test plan. It should be in agreement with all other levels of test plan if any]*

[Example:

The goal of testing is to provide the information about the level of consistency between test items and customer requirements. This information will be provided by means of QC activities performed to control quality and defects registered in result of testing and discrepancies detected. All defects will have a Priority attribute determined by analyzing the importance and frequency of functionality used by the customer and risk of its appearing in operational use.

Quality control will be performed by QC staff on the System level only. Testing on Unit and Integration levels will not be done by QCEs. Acceptance level will be covered by customers after iteration results and product delivering.

The analytical, requirements-based strategy, where an analysis of the requirements is the basis for planning, estimating and designing tests will be used as a main test strategy. Also, exploratory testing can be used for discovering dependencies in functionality and usage of this knowledge in test design and testing.]

## Test approach

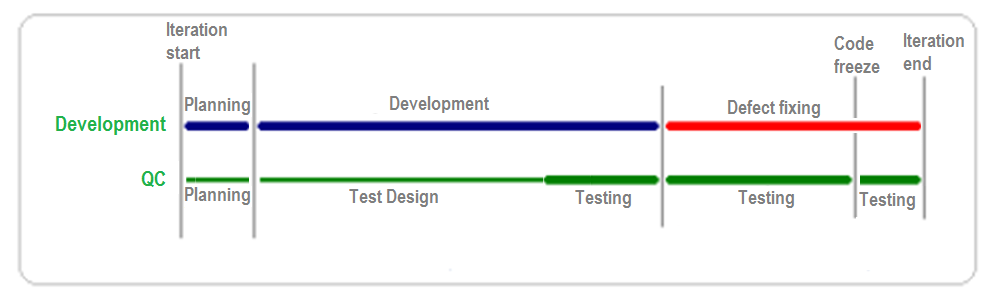
*[Describe the approach how the test strategy will be implemented]*

[Example:

As the whole software development process is divided into iterations (2 weeks each), the testing activities will be divided in the same way to be consistent with the development process and to serve the goal to deliver a potentially shippable product at the end of each iteration to the client. The list of new functionality or improvements to existing functionality to be developed in scope of the iteration is a basement for the QC activities to be conducted for ensuring quality control.

The full set of QC activities will be performed within each of iterations starting from planning and ending with test closure activities.

The basic Iteration workflow is shown on chart below:



The detailed description of each QC activity is described in QC Activities Workflow chapter.]

### Test types

*[Identify and describe all the testing types to be applied in scope of testing.]*

[Example:

The following types of testing will be performed within project:

**Smoke testing** defines the initial tests performed after initial construction of or critical change to any component. It assures that the most critical features work.

Smoke testing is performed by developers before releasing functionality to QC. Also, smoke tests will be used by QCEs to accept software builds from the Development team. See Entry Criteria section to know what criteria will be used to decide whether a system under test can enter the testing phase.

**UI testing** is performed by QCEs to check whether the GUI adheres to common and project specific UI standards.

**Functional testing** refers to the set of comprehensive tests that verify system functioning against functional requirements. It verifies that all functional requirements are implemented properly. Functional testing is QCEs responsibility.

**Confirmation testing,** performed by QCEs, runs test cases that failed the last time they were run, in order to verify the success of corrective actions.

**Regression testing**is performed by QC staff after new components and defect fixes are integrated into the existing system to validate that unchanged components (that were implemented earlier) still work as originally intended.

The depth of regression testing and the set of tests to be run will depend on the new components and/or defects fixes nature and complexity.

**Configuration testing** is focused on testing all browser dependent application features on the different configurations. Configuration testing is QCEs responsibility. Test configurations are described in the Test Environment section.

**Acceptance testing** is performed by <Client> with respect to end user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.

Separate tests will not be designed and developed for **Smoke**, **Regression** and **Configuration** **testing** types. Part of **UI** and **Functional** tests will be used as smoke, regression and configuration tests. ]

### Test coverage

*[Define and describe the coverage of features to be tested by testing]*

[Example:

All new functionality and improvements to existing functionality included into the scope of iteration should be covered by **Smoke**, **UI**, **Functional** and **Regression** tests.

**Acceptance testing** will be performed by the customer after iteration ends without PlanA participation.]

## QC Activities workflow

***[Example***:

QC process will consist of the following activities united into groups:

1. Planning:

* Analyzing and clarifying scope of work for upcoming iteration;
* Test Planning;

1. Test Design:

* Test Design and Implementation;
* Maintenance of designed tests;

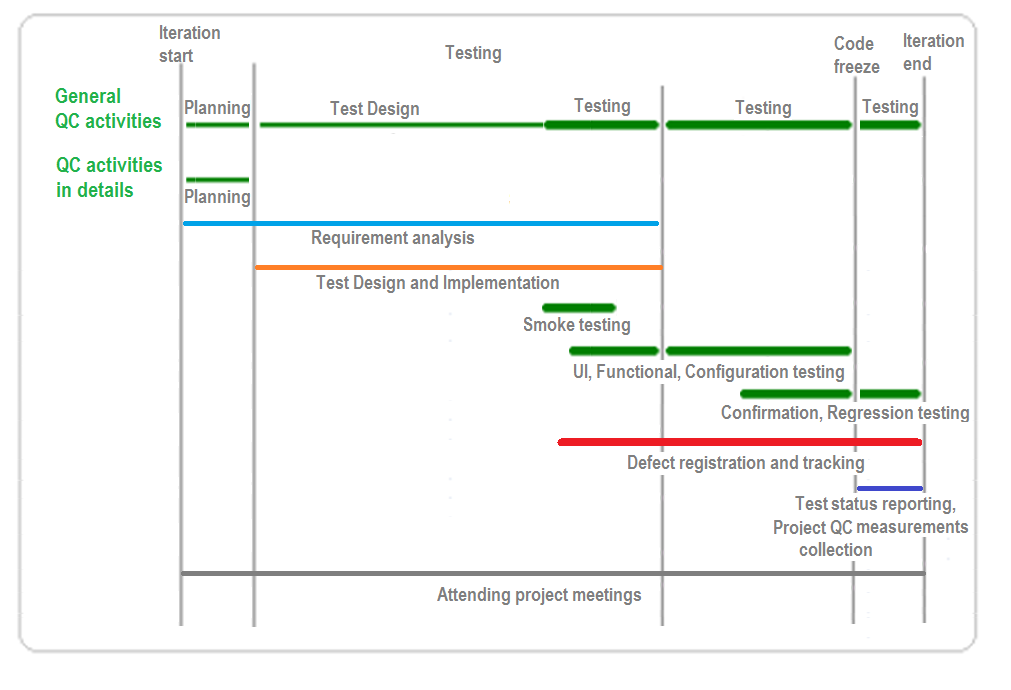
1. Testing:

* Test data preparation;
* Manual Test Execution;
* Defects registration and tracking;
* Defects fix verification;
* Test status reporting;

1. Project QC measurements collection;
2. Attending project meetings

* Iteration Planning;
* Status meetings with Team on daily basis;
* Iteration Demo meetings;
* Iteration Retrospective meetings;
* Status meetings with Customers.

QC activities workflow is shown in details on the chart below:



The responsibilities of every role performing QC activities listed above will be covered in next chapters that correspond to the group of activities to which these activities belong.]

*[The following chapters should describe how the QC activities will be performed.*

*In general for each activity/group of activities the following questions should be answered:*

* *When can we start doing it? – entry criteria*
* *Are there any criteria that prevent us from continuing doing it? – stop (suspension) criteria*
* *What criteria should be satisfied to proceed with it? – continue (resumption) criteria*
* *When can we finish doing it? – exit criteria*
* *What exactly should we do and how is the scope of this activity/group of activities?****]***

## Test planning

*[This chapter contains the detailed description of QC activities to be performed in scope of test planning.*

*This group of QC activities can be omitted from the test plan in case project implementation approach is non-incremental and test planning is performed once for the whole project.]*

### Test planning exit criteria

QC activities planning is performed within Iteration planning. The planning is based on the scope of the Iteration (new functionality or improvements to existing functionality to be developed during the iteration are defined).

Following requirements should be satisfied before QC activities planning:

1. Scope of work is defined and described;
2. List of configurations on which functionality or improvements should be tested is defined;
3. Iteration scope is prioritized.

In case some or all of listed requirements are not satisfied before QC activities planning, the risk of insufficient or inappropriate test planning for iteration exists.

### Test planning process

The following tasks are performed during test planning:

1. User Stories\Improvements analysis;
2. Defining features to be tested accordingly to the scope of Iteration;
3. Defining features not to be tested;
4. Defining the depth of testing needed based on the complexity of functionality to be developed;
5. Defining the depth of regression testing needed in the end of iteration based on the scope of the iteration;
6. Estimation QC activities for the iteration that should include next activities:
7. Test design and test cases implementation;
8. Smoke, UI, Functional testing taking into consideration the number of supported configurations;
9. Confirmation testing based on the complexity of functionality to be developed and the project measurements;
10. Exploratory testing sessions;
11. Regression testing based on the complexity of functionality to be developed and the project measurements.

### Test planning stop/continue criteria

Test Planning stops when a lot of discrepancies and controversial issues in the requirements are revealed during test planning.

Test Planning renews after:

* questions are clarified by client and announced to the Project Team;
* assumptions and agreed solutions are made by the Project Team for not clarified by client issues.

### Test planning exit criteria

The absence of discrepancies, questions and doubts related to the functionality to be delivered at the end of iteration, is the exit criteria to finish this test planning. The Test Plan for iteration might be developed at the end of Test Planning activity.

## Test design

*[This chapter contains the detailed description of QC activities to be performed in scope of test design.]*

Test Cases will be designed for UI and Functional testing only. There will be no separate Test Cases for Smoke, Regression, or Configuration testing unless it is considered necessary for specific iteration. All the Test Cases will be designed to verify that all the requirements have been satisfied. The objective of every Test Case should be formulated accordingly to template “Verify that <expected result>“.

Any specific data needed for test execution will be described in Test prerequisites or created while test execution

### Test design entry criteria

Following requirements should be satisfied before test design started:

1. As the main test strategy is requirement-based, the requirements for iteration should be finalized before test design.
2. Test planning for iteration is finished.

In case some or all of the listed requirements are not satisfied before test design starts, the risk of insufficient or inappropriate test design for story\improvement exists.

### Test design process

The following tasks are performed during test design for particular User Story\Improvement:

1. Analyzing possibility to use existing test cases.
2. Defining a set of tests to be run based on the documented functional requirements, UI Guide and common sense.
3. Defining additional tests to be run based on knowledge gained during exploratory testing sessions.
4. Every test should contain the following attributes to provide requirements and tests traceability:
5. Story\improvement identifier and summary;
6. The particular requirement that test is aimed to verify;
7. Test objective (actually, the test itself).

5. Prioritizing tests accordingly to:

1. The level of possible impact to perform daily tasks in operational use;
2. The logical dependencies between tests;
3. The data dependencies between tests.
4. Test cases implementation which consists of defining and documenting following required information (if it was not changed while test planning):
5. Preconditions that should be met before test case execution;
6. Steps to be followed while test case execution;
7. Instructions to be followed while test execution if any.

### Test design stop/continue criteria

Test Design stops when a lot of discrepancies and controversial issues in the requirements are revealed during any step of the test design phase.

Test Design renews after:

1. questions are clarified by client and announced to the Project Team;
2. assumptions and agreed solutions are made by Project Team for not clarified by client issues.

### Test design exit criteria

In result of test design the Test Case Specification that contains all tests, needed to be executed to verify if implemented\altered functionality works as it was supposed, is written.

Test Case Specification for current iteration is available to the Project Team.

## Test Execution

*[This chapter contains the detailed description of QC activities to be performed in scope of test execution.]*

### Test execution entry criteria

The following requirements should be satisfied before test execution:

1. Test Design is finished for the particular story\improvement;
2. All development tasks for this particular story\improvement are closed or developer has informed QCE that some part of functionality can be tested before all tasks are closed;
3. Testing version of the application is built and deployed to the testing environment;
4. Defect tracking system is configured;
5. Smoke testing is finished and defects that prevent effective test case execution are fixed and verified.

In case some or all of the listed requirements are not satisfied, the test execution cannot start.

### Test execution process

The following tasks are performed during test execution:

1. Preparing test data and bringing the application to the state, specified in Test Prerequisites (preconditions) in Test Case Specification;
2. Performing Steps to ensure that Test Objective is satisfied;
3. In case actual result differs from expected (“Test objective\Expected results”) or result in some intermediate step during test execution differs from described in test case the issue (Sub-Defect or Bug) in Jira should be registered.

The Defect management procedure is described in detail in the Defect[/Issue Management Process](#_49x2ik5) chapter.

1. Test status reporting in Test Case specification:
2. PASS - application performs in the way described in “Test objective\Expected results” and no flaw appears while “Steps” execution;
3. FAIL - application’s behavior differs from described in “Test objective\Expected results” or the defect appeared on some steps during test execution etc.

QCE has to add a defect ID (or link) into the Comments” section of test case specification;

1. SKIPPED - the test was omitted from the executions for some reasons (for example, verification is blocked by another issues, item was excluded from the verification scope, etc);

QCE has to describe the exact reason in the “Comments” section of test case specification;

1. INVALID - Test Cases cannot be executed due to some errors in the steps, test prerequisites, or test objective.

QCE has to describe the exact reason in the “Comments” section of test case specification.

1. Verification of fixed Sub-Defects accordingly to [Defect/Issue Management Process](#_49x2ik5) and following actions:
2. Test Case status updating in Test Case Specification after defect has been fixed and verified;
3. Test Case maintenance to correspond to final agreement in case the requirements have been clarified in process of defect fixing.

Verification of fixed Bugs.

### Defect/Issue management process

Jira is used as a Requirement and Defect Management tool on the project.

At the Iteration Planning stage the iteration is filled by elements from the Backlog. The Backlog consists of the following elements:

* User Story – the unit of work, that can be used as functional unit (requirement) or unit for planning;
* Improvement – an enhancement to existing functionality;
* Bug – defect (flaw, mistake, warning, etc.)
* Iteration Backlog filling consists of next steps:
* Issues with User Story type can be entered to Backlog by Product Owner or Project Team with aim to add value to the product or to divide functionality into planning units;
* Issues with Improvement type can be entered to Backlog by Product Owner or Project Team with aim to add value to the product through enhancing existing functionalities.
* Issues with Bug type can be entered to Backlog by Product Owner or Project Team with aim to register problems in existing functionality. Also, issues with type Bug can be created by converting Sub-Defects, created whiletestingUser StoriesorImprovementsin the Test Execution stage.

Defect tracking procedure in scope of QC activities consists of:

1. Defects registration;
2. Defect verification.

#### Defect registration

Two types of defects can be entered in Jira by QCEs while Test execution: Sub-Defects and Bugs.

In case a flaw in application functioning or UI, has been observed, related to new functionality/improvement that is being implemented in scope of current iteration, the Sub-Defect to Story/Improvement is registered.

In case a flaw in application functioning or UI, has been observed, related to existing functionality that is out of scope of current iteration, the Bug is registered.

There are following reasons when issue with type Bug can appear:

* The defect has been missed while testing the functionality after it had been implemented (Post Delivery defect);
* The defect occurred in result of some changes after functionality had been tested (Regression defect);
* The Sub-Defect was not fixed in scope of iteration where it has occurred and is deferred to the next iterations.

The next attributes if the Defect (Bug or Sub-Defect) should be specified while registration:

| **Field Name** | **Description** |
| --- | --- |
| Summary\* | A short summary of the issue |
| Defect Type\* | Type of defect: Sub-Defect or Bug |
| Priority\* | |  | The importance of the issue in relation to other issues. | | --- | --- |  * **Blocker** — Highest priority. Indicates that this issue takes precedence over all others. * **Critical** — Indicates that this issue is causing a problem and requires urgent attention. * **Major** — Indicates that this issue has a significant impact. * **Minor** — Indicates that this issue has a relatively minor impact. * **Trivial** — Lowest priority. |
| Due date\* | The date by which this issue is scheduled to be completed:   * End of Iteration date for Sub-Defects; * End of Project date for Bugs. |
| Component/s | Project component(s) to which this issue relates. |
| Affects Version/s\* | Versions which are affected by this defect |
| Fix Version/s | Version where a defect is planned to be fixed. When registering  defect is set equal to “Affects Version/s\*”. |
| Assignee | Developer who will fix the defect. In case the exact developer can’t  be determined right now, the “Assignee” should be set to  ”Unassigned” |
| Environment\* | The hardware or software environment to which the issue relates |
| Description\* | A detailed description of the issue that consists of :   * Steps to reproduce; * Actual result; * Expected result. |
| Attachment | Screenshots or other files that is helpful for defect reproducing |
| Labels\* | * The cause of the defect or its source. |
| Grouping\* | Indication of phase of the project in which the defect has been found |
| Regression | Indication if the defect is regression in existing functionalities caused  by new features implementation or not. |
| Scope\* | Filled by default by “Product” value and not edited while defect  registration |
| Fixed in Build | Version of the application (build) in which the defect is fixed.  Remains empty while defect registration. |
| Found in Build\* | Version of the application (build) in which defect is found |

Attributes marked by asterisk (\*) are required while defect registration.

Every defect should be linked to the story\improvement that contains requirements for the corresponding functionality to keep the traceability between defects and requirements.

* In case the flaw is registered as Sub-Defect, the corresponding issue should be created from User Story/ Improvement that describes the functional unit which is being tested;
* In case the flaw is registered as Bug, this issue should be linked to corresponding User Story/ Improvement that describes the functionality to which the defect is related.

All registered Sub-Defects automatically fall into current Iteration and all Bugs fall into Backlog. In case Bug’s priority is higher than Normal (Major, Critical and Blocker) or QCE considers Bug as an issue that can impact Iteration Demo it should be moved to the Iteration manually.

#### Defect verification

Any defect, despite the way of its resolution and current assignee, being in ‘Ready For Verification’ status should be verified by QC engineer.

* Possible ways for defect resolution are the following:
* **Fixed** — A fix for this defect has been implemented;
* **Won't Fix** — defect will not be fixed, e.g. it may no longer be relevant.
* In case Resolution is Won't Fix developer should comment the decision with specified decision maker;
* **Duplicate** — defect is a duplicate of an existing issue.
* it is recommended you create a link to the duplicated issue;
* **Cannot Reproduce** — defect could not be reproduced at this time, or not enough information was available to reproduce the issue.

1. In case the way defect has been fixed differs from described in Expected result part of Defect Description, **the developer has to comment on the final decision and indicate the decision maker.**

In case, QCE confirms that defect no longer exists as a result of verification, the issue in Jira should be closed with specifying “Fix in Build” attribute for defects with resolution Fixed before defect closing.

In case, QCE detects a fix incompletion or for some reason specified by developer resolution is not applicable for this particular defect, the corresponding comment should be added and defect should be reopened and reassigned to the developer.

### Test execution stop/continue criteria

Test Execution stops when a defect that blocks the execution of the rest tests has been detected.

Test Execution renews after all blocker defects are fixed and verified.

### Test execution exit criteria

The following requirement should be met to finish test execution for particular story\improvement:

1. All tests planned to be run are run;
2. All Sub-Defects found are registered in Jira;
3. All Sub-Defects defined to be fixed before delivery, are fixed and verified.
4. All test cases corresponds to the final requirements;
5. Test status for every Test Case corresponds to actual state.

The following requirement should be met to finish test execution for iteration:

1. All exit criteria are satisfied for every story\improvement form iteration scope;
2. All Bugs defined to be fixed before delivery, are fixed and verified.
3. All regression tests defined to be run to ensure that new functionalities\bug fixes does not cause negative impact, are run;
4. All Bugs detected in the result of regression testing and defined to be fixed before delivery, are fixed and verified.

## Test environment

*[Define and describe all the requirements to the environment for testing in scope of this test plan. Special hardware and software of specific versions should be listed.]*

### Configurations

*[List the test configurations required to be covered within this test plan.]*

***[Example:***

Environment dedicated for Quality Control activities- <link>

The primary testing platform is Windows OS 7 X64 and Google Chrome (recent version) browser.

In case any other configurations need to be supported for some functionality\iterations, it will be announced before iteration planning.***]***

### Testing tools

*[List the tools needed for testing within this test plan.]*

***[Example:***

Following tools will be used by QC team for quality control activities:

* Test Cases development and maintenance - GoogleDocs and Excel;
* Defects tracking tool – JIRA;
* Tools for JSON visualization – e.g. JSON viewer plugin for Notepad ++, different online tools, etc.;
* DB administration and management tool – pgAdminIII v.1.18.1***]***

## Test deliverables

*[List what is to be delivered as a result of testing, except the software itself that is listed under test items and is delivered by development.*

*Common list of deliverables is:*

* Test Plan
* Test Design Specifications (list of tests)
* Test Case Specifications (tests)
* Defects entered in the defect tracking tool
* Test Summary Report (upon delivery)
* QA Status Report (weekly)]

***[Example:***

Test deliverables will include:

* List of Completed functionality (story\improvements from Jira);
* List of Not Completed functionality if any (story\improvements from Jira);
* List of Fixed Sub-Defects (from Jira);
* List of Fixed Bugs (from Jira);
* List of deferred Sub-Defects (from Jira)
* Test Case Specification with test status for every test and QC Status Reports.***]***

# Roles and Responsibilities

*[Define the roles which take part in testing within the scope of this test plan and list their responsibilities.]*

| **Role** | **Responsibilities** |
| --- | --- |
| QC Lead | * Establishing and tailoring QC process; * QC activities monitoring and control; * QC team management; * Test planning; * QC activities status reporting; * All the QC Engineer’s responsibilities. |
| QC Engineer | * Requirements Analysis; * Test design and implementation; * Test execution; * Defects tracking; * Test results analysis and reporting; * Maintenance of designed tests. |

# Risks

*[Define and list overall risks to the project with an emphasis on the testing process. Specify what will be done for various events and who is responsible for taking actions]*

***[Example:***

The following risks may occur and delay testing:

| **#** | **Risk** | **Who** | **Action** |
| --- | --- | --- | --- |
|  | Build is not ready on time | QC Lead | To notify the PM about the shortage immediately.  To review Test Case Specification and to reprioritize tests to cover the most important functionality (with the highest priority) to the delivery. |
|  | Installed build is not functioning | QCE | To register an issue in Jira with Blocker priority.  To notify the Dev Team and PM about the issue. |
|  | Environment is not deployed properly corresponding to approved configuration | QC Lead | To notify the PM about the shortage immediately.  Inform about impact on testing. |
|  | Lack of human resources | QC Lead | To notify the PM about the shortage immediately.  To review Test Case Specification and to reprioritize tests to cover the most important functionality (with the highest priority) to the delivery. |
|  | Technical issues (power outage, loss of connection etc) | QC Lead | To notify the PM about the shortage immediately.  To review Test Case Specification and to reprioritize tests to cover the most important functionality (with the highest priority) to the delivery. |

In case some or all of the listed risks occur, QC engineer informs PM about impossibility to finish the current test activity and all the future tasks properly in full extent that can cause the quality issues.

Prioritization of tasks is recommended to avoid\decrease the level of impact on the iteration results and schedule.

QC activities planned for current iteration can be reviewed and altered: some of them might be omitted or performed in limited scope.]

# Schedule