

MINA Adly

SUEZ CANAL BANK TESTING plane

V 1.0

Table of Contents

[Document Control 2](#_Toc115770904)

[Document History 2](#_Toc115770905)

[Review Panel 2](#_Toc115770906)

[Approvals 2](#_Toc115770907)

[Supporting Documents 2](#_Toc115770908)

[Scope of work 3](#_Toc115770909)

[Testing Scope 3](#_Toc115770912)

[Functional Testing (per sprint) 3](#_Toc115770913)

[System Testing 3](#_Toc115770914)

[Non-Functional Testing (per sprint) 4](#_Toc115770915)

[Test Environments 5](#_Toc115770916)

[Testing tools 5](#_Toc115770917)

[Entry / Exit Criteria 5](#_Toc115770918)

[System testing /UAT Prerequisites 6](#_Toc115770919)

[Defect Management Process 7](#_Toc115770920)

[Defects Definitions 7](#_Toc115770921)

[Defect Tracking 8](#_Toc115770922)

[Defect creation 8](#_Toc115770923)

[Severity classification 9](#_Toc115770924)

[Response to raised defects 11](#_Toc115770925)

[Project Controlling Meetings 12](#_Toc115770926)

[Project Reporting 12](#_Toc115770927)

[Project Stack holders 13](#_Toc115770928)

# Document Control

## Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Author | Comments |
| 05-03-2025 | 1.0 | Mina Adly |  |
|  |  |  |  |
|  |  |  |  |

## Review Panel

|  |  |
| --- | --- |
| Review Panel | |
| Name | Role |
|  |  |
|  |  |

## Approvals

|  |  |  |
| --- | --- | --- |
| Version | Approval Date | Approver Details |
|  |  |  |
|  |  |  |
|  |  |  |

## Supporting Documents

|  |  |  |
| --- | --- | --- |
| Document | Location | Owner |
|  |  |  |
|  |  |  |
|  |  |  |

# Scope of work



## Testing Scope

**SUEZ CANAL BANK** project testing strategy will follow agile principles to ensure delivering the business requirements at frequent intervals and working at a sustainable pace.

Testing activities will be performed against delivered modules based on the context of the provided functional requirement(s) and sprint scope (user stories).

The following test processes are in scope:

## Functional Testing (per sprint)

1. **System Testing**

First stage of testing where problems are identified based on each user story separately. The whole system is tested in depth. Testing focus will be on **COMPLEATE\_PROFILE** and **single transaction level**.

Second stage of testing where all integrations between the different modules and Integration points are tested. The whole system is tested in breadth. Testing focus will be on multiple transaction workflows and complete profile Document within one enterprise area and integration between the different modules.

1. **User Acceptance Test:**

The Final phase of the software testing process to ensure usability of the implemented system by testing the entire system strongly involving end user community. During UAT, actual software users test the software to make sure it can handle the required tasks in real-world scenarios according to specifications. UAT is one of the critical software project procedures that must occur before newly developed software is rolled out to the market.

## System Testing

Testing team will be engaged in the following project phases:

* Planning & Control
* Analysis & test case design
* Execution
* Evaluation & Reporting
* Project Closure

During all project phases testing team will ensure a solid, clear and detailed reporting process. The following project phases will be delivered with the following tasks:

A screenshot of a cell phone

Description generated with very high confidence

## Non-Functional Testing (per sprint)

1. **Performance Test: (if recommended)**

* Is conducted to understand the behavior of the system under a specific expected load. This load can be the expected concurrent number of users on the application performing a specific number of transactions within the set duration.
* The performance testing will be executed on 2 stages
  + Initial performance testing: - will be executed per user story based on the status of the story by creating performance scripts related to the user stories within each sprint , those scripts are subjected to change according to ongoing user/technical changes
    - The user story should be in done status.
    - The user story had been technically tested & passed.
    - The testing environment to be determined as per environment readiness ideally it should be pre-prod (production like environment).
  + Final performance testing :- will be executed per release after full scope has been signed-off
    - The user story should be in done status.
    - The user story had been tested & passed both Functional & UAT
    - The testing environment is pre-prod (production like environment).
* For mobile application, unsecured APK versions may be needed
* Performance testing will be executed on a production like environment
* Performance testing client (32 GB of memory + 4 nodes CPU)

# Test Environments

Two main test environments are required for testing activities:

* Functional Test Environment.
* Performance Test Environment production like (pre-production).

Development environment is separated from testing environment

* Testing team has access to the testing environment
* Development team has access for investigations and deployments
* Development team has to share release note for each build

For all Interfaces, Test Environment has to be connected to all integration points thus each interface could be correctly tested.

# Testing tools

* JIRA 🡪 For [Test\Defect] Management.
* JMETER (open source Tool)🡪 For Performance Testing (Free of charge).

# Entry / Exit Criteria

The concept of establishing pre-requisites (entry criteria) and post conditions (exit criteria) is extremely important in managing any process. The need for using documented entry / exit criteria is to ensure clear understanding where hand offs occur so the sending organization or unit can verify completeness before handing over and receiving unit can validate ‘readiness’ to initiate the process.

Entry and Exist criteria will be defined based on core scenarios readiness per user-story from **SUEZ CANAL BANK** perspective (Project stakeholder’s product owner, Testing team)

By this the overall testing criteria will be defined based on the total sprints exist criteria

The Final User Acceptance sign-off will be considered as the Exit Criteria for the project.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Round** | **Criteria Level** | **Entry Criteria** | **Exit Criteria** |
| **System Testing** | * User story | * Story is ready and confirmed by business team. * Unit tests delivered with zero defects of high criticality * Test cases are created, reviewed and signed-off * Test environment is ready * Test data is ready * All Parameters are set for all user story. | * **Pass Percentage**:   + 100% of core scenarios passed   + More than 60% of overall scenarios passed |
| The user story exist criteria is subjected to change according to project stack holders decision (product owner – business team – testing team ) | | | |
| **UAT** | * User story | * User story met the system testing exist criteria * User story moved to ready for UAT pool (JIRA dashboard) | * Business team signed off /conditional sign off reached |
| The user story exist criteria is subjected to change according to project stack holders decision (product owner – business team – testing team ) | | | |
| **Initial performance testing** | * User story | * User story met UAT exist criteria * Excepted traffic provided by business team | * Performance test scenarios executed. * Performance testing report issued. |
| **Release Sanity testing** | * Production release | * All stories within the release scope met the UAT exist criteria | * All sanity scenarios executed successfully. * All Sanity scenarios passed * No critical or showstopper issues still opened |
| **Release UAT** | * Production release | * Release sanity testing met exist criteria | * Execution percentage: - 100% of test cases had been executed. * All Sanity scenarios passed * No opened issues (critical – or show stoppers) * Business team signed off the UAT |
| **Release performance testing** | * Production release | * Release met Release UAT exist criteria | * Performance report is meeting expected business traffic.(volume) |

# System testing /UAT Prerequisites

Below are the needed prerequisites - in general - for testing:

* BRD’s / User stories are completed and signed-off
* Implementation plan is provided
* System under test is available
* System Test Cases are signed-off
* Required Test environments are ready
* System release is deployed and ready
* Security Matrix is completed and implemented in the system
* Interfaces are implemented and ready
* Integration nodes are connected
* Business team personals are available for UAT
* Access to L&P tool (for Load and Performance)
* L&P servers are ready (for Load and Performance)

# Defect Management Process

The purpose of the defect management process is to implement the govern procedures to manage the tracking and fixing of defects found during testing phases.

## Defects Definitions

A defect is identified when there is a variance in the test results from the expected test results that were predefined for a particular test case.

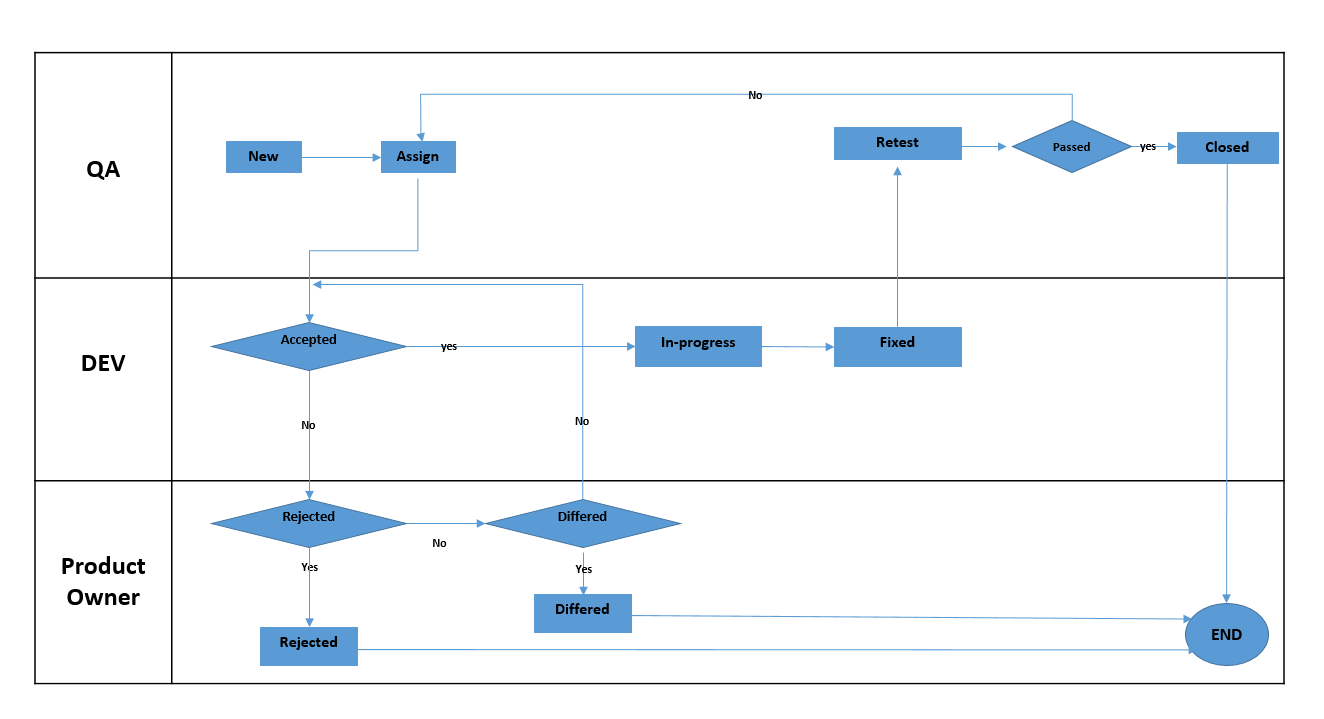
All these defects and any others should be logged, but the cause/type should be established as the action to resolve the defect could differ in each case.

Examples of defects are:

* Bug in the code
* Incorrectly stated requirement
* Overlooked requirement
* Incorrect function

All defects detected during each phase of a project either through static or dynamic testing will be recorded, logged and have cause attributed. These measurable defects will then be used to determine the quality level achievements of prior testing activities and to assist in determining defect forecasts for the remaining phases. The defect logging process and subsequent reports will provide key management information during the project, also enable the prioritization of defect resolution. In addition, the historical information collected will aid in defect rate forecasts for future releases and projects.

## Defect Tracking

The following graph summarizes the defect life cycle that will be used throughout all project phases.

**Re-open**

## Defect creation

When a defect is created, the following information should be filled out in the test/defect management tool (these fields may vary depending on the Test Tool, which will be selected):

* **Defect ID:** unique identification number (should be automatically generated)
* **Title:** short headline description of the defect contains the following
  + Sprint ID.
  + Squad.
  + Module name
  + Descriptive title
* **Description:** a detailed description of the defect and steps to reproduce it, so that the development team or other test team members can reproduce the defect.
* **Status:** the current status of the defect as illustrated in the above diagram
  + **New**.
  + **Assigned.**
  + **In progress: -** once dev team accept as bug and start fixing.
  + **Fixed:** dev team completed the required fix and the bug is ready for re-testing.
  + **Re-open:** the retesting of the fixed bug had been failed.
  + **Rejected:** in case the dev team reject a bug with reason the product owner can put it in rejected state.
  + **Differed**: the dev team reject a bug with reason and the product owner decide to accept it but the fix will be postponed.
  + **Closed** : the bug had been fixed and the fix had been verified by testing team
* **Severity:** the severity of the defect for the business
* **Creation Date:** the date on which the defect was raised (should be automatically generated)
* **Raised By:** test analyst who raised the defect
* **Defect Type Classification:** Functional issue, Configuration issue, Enhancement, Data Issue, Environment Issue, Deployment issue
* **Rejection Reason:** Duplicate, New Requirement, Not a Bug, Not Reproducible, Missing Information, out of scope
* **Impact**: Show stopper defect or a blocking defect
* **Assigned To:** the name of the testing or development or business team member, by whom the defect is assigned to take an action
* **Module:** the module impacted by the defect
* **Phase/Iteration:** test phase or test iteration, in which the defect was raised
* **Test Cycle:** within each phase/iteration, when the defect was discovered (i.e. Cycle 1, 2, 3 etc...)
* **Comments:** in case any other important information related to the defect needs to be clarified/illustrated
* **Attachments:** wherever possible, add relevant screenshots of any error messages or screens showing any deviation to expected test outcome

## Severity classification

The severity of a defect will help determine the level of risk to the business, if the defect went into production. The defect manager should assign a business severity for the defect in the chosen test/defect management tool. The following values can be assigned to the defect severity:

|  |  |
| --- | --- |
| **Severity** | **Description** |
| **1 – Critical** | In this case, a key function in the system is not available and a workaround is not possible or practical. |
| **2 – High** | Severe issue or missing functionality. Potential data integrity issues. A workaround is possible and can be achieved procedurally with a report that will enable a manual temporary resolution. |
| **3 – Medium** | A regular issue or missing functionality that has a contained impact. An alternate functionality may be used to affect the same requirement or a workaround to mitigate the risk is possible. |
| **4 – Low** | An issue that will have minimal impact. An alternate functionality may be used to affect the same requirement or a workaround to mitigate the risk is possible. Affects the look of a particular screen or report or a missing functionality that does not exist today and or infrequently used and has very low impact. |

There will be a flag for blockers defects, hotfix and deployments can be done in cooperation between Testing team and Development on spot and no need to wait until the next planned deployment day.

## Response to raised defects

To enable the provision of an effective and efficient test operation, it is essential that the resources supporting the test process respond in a timely manner according to an agreed service level agreement between the support organizations and the testers.

The table below defines the default SLA for responding to defects found during testing according to the severity of the defect:

|  |  |  |
| --- | --- | --- |
| Test Severity | Acknowledgement and  assign defect within | Target fix date  Or ETA (depends on the type of issue) |
|
| 1 – Critical | Same working day | * 1 working day for development issues. (development issues includes issues related to backend , APIs ), * in case of bug related to the system the SLA will be according to SUEZ CANAL BANK SLA |
| 2 – High | Same working day | * 1 working day for development issues. (development issues includes issues related to backend , APIs ), * in case of bug related to the system the SLA will be according to SLA |
| 3 – Medium | 1 working day | * 2 working days for development issues. (development issues includes issues related to backend , APIs ), |

# Project Controlling Meetings

|  |  |  |
| --- | --- | --- |
|  | **Points of discussion** | **Participants** |
| **Daily stand-up**  **meeting** | Test status, progress , pending issues from last day, planned activities for the current day | Testing team, Dev team |
| **Sprint planning**  **meeting** | Sprint scope, expected modules, relation with previous modules | Testing team, Dev team and project manager |
| **Sprint retrospective meeting** | Lessons learned, overall status with detailed cycle results and metrics, defects [System testing , UAT], risks, next steps including performance testing activity if there will be | Testing team, Dev team and project manager |
| **AD-HOC Meeting** | Unplanned meeting at any phase of the project | Project stakeholders  Testing, Business, PM, product owner and Vendor(s) |
| **Sprint review** | To inspect the outcome of the [Sprint](https://www.scrum.org/resources/what-is-a-sprint-in-scrum) and determine future adaptations. The Team presents the results of their work to key stakeholders and progress toward the Product Goal is discussed.  revise Product Backlog that defines the probable Product Backlog items for the next Sprint. | Testing team  Dev team  Product owner |

# Project Reporting

|  |  |  |
| --- | --- | --- |
|  | **Points of discussion** |  |
| **Daily status report** | Test status, progress , pending issues from last day, planned activities for the current day | Testing team, Dev team |
| **End of Test Phase** | Overall status with detailed results and metrics for all cycles within the completed phase, progress evaluation, performance evaluation, blockers, risks, pending and outstanding issues | Project stakeholders  Testing, Business, PM and Vendor(s) |
| **UAT daily progress** | UAT status, progress, issues and risks | Business, testing team, Dev team and PM |
| **UAT closure report** | Business sign-off | Project stakeholders |
| **Performance report status** | Performance status , progress and performance statistics per each module | Project stakeholders  Testing, Business, PM and Vendor(s) |
| **End of Project** | Overall status with detailed results and metrics for all project phases, progress evaluation, performance evaluation, defects, issues, risks, lessons learned | Project stakeholders  Testing, Business, PM and Vendor(s) |

# Project Stack holders

|  |  |
| --- | --- |
| **Role** | **Name** |
| SUEZ CANAL BANK PM |  |
| Product Owner |  |
| SUEZ CANAL BANK PM |  |
| SUEZ CANAL BANK technical leader |  |
| SUEZ CANAL BANK testing Manger |  |
| SUEZ CANAL BANK testing manger |  |
| SUEZ CANAL BANK Testing team (one for each squad) |  |
| SUEZ CANAL BANK Testing team |  |
| SUEZ CANAL BANK dev team |  |
| SUEZ CANAL BANK MW & Back end |  |
| SUEZ CANAL BANK DB admin |  |
| SUEZ CANAL BANK Network admin |  |
| SUEZ CANAL BANK IT system admin |  |
| SUEZ CANAL BANK Security Ops. |  |
| SUEZ CANAL BANK info Sec. |  |