**Royal Mail Group**

**Digital Labs Project Test Strategy**

**Author: kirubanand ramasamy**

1. Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Issue | Date | Author | Change history |
| 0.1 | 11/10/15 | kirubanand | First working draft |
| 0.2 | 11/13/15 | Kirubanand | Updated based on Comments from Kevin harris |

1. Document Reviewers

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Name | Date | Issue |
| 1 | Uday Deorukhakar – Capgemini Engagement Manager |  |  |
| 2 | Ahuja Girish – Capgemini Service Delivery manager |  |  |
| 3 | Kevin Harris- Capgemini Process Consultant | 11/12/2015 |  |

1. Document Approvals

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Name | Date | Issue |
| 1 | Uday deorukhakar – Capgemini Project Manager |  |  |
| 2 | Ahuja Girish – Capgemini Service Delivery manager |  |  |

**Contents**

[1 Introduction 4](#_Toc432499054)

[1.1 Overview 4](#_Toc432499055)

[1.2 Digital labs Development Process 4](#_Toc432499056)

[1.3 Testing in Scope 5](#_Toc432499057)

[1.4 Not in Scope 5](#_Toc432499058)

[1.5 Test Phase Overview 7](#_Toc432499062)

[1.6 Test Deliverables 8](#_Toc432499063)

[1.7 RMG digital labs Test Acceptance 9](#_Toc432499064)

[2 Daily Test Execution 10](#_Toc432499065)

[2.1 Start of Day Testing 10](#_Toc432499066)

[2.2 End of Day Testing 10](#_Toc432499067)

[2.3 Daily & Final Test Reporting 10](#_Toc432499068)

[2.4 Test Recording 10](#_Toc432499069)

[2.5 Test Incident Escalation 10](#_Toc432499070)

[2.6 Regression Testing 10](#_Toc432499071)

[3 Test Data 11](#_Toc432499072)

[4 Test Assurance / Governance 11](#_Toc432499073)

[5 Assumption & Dependencies 11](#_Toc432499074)

[5.1 Assumptions 11](#_Toc432499075)

[5.2 Dependencies 12](#_Toc432499076)

[6 Test Governance 12](#_Toc432499094)

[6.1 Test Monitoring 12](#_Toc432499095)

[6.2 Defect Management 13](#_Toc432499096)

[6.3 Configuration Management 13](#_Toc432499097)

[7 Test tools and environments 13](#_Toc432499098)

[7.1 Tools 13](#_Toc432499099)

[7.2 Automation Testing 14](#_Toc432499100)

[7.3 Environment Usage 14](#_Toc432499101)

[8 Project Roles and Responsibilities 14](#_Toc432499105)

[8.1 Roles 14](#_Toc432499106)

1 Introduction

* 1. Overview

This document is the test strategy for RMG digital labs project.

* 1. RMG Digital Labs Development Process

The chosen development process for the RMG digital labs solution will follow Iterative Development process this will provide a lightweight, but disciplined, approach to software development, utilising a number of Agile practices. It will be risk-driven, iterative and incremental, with a focus on high collaboration and low ceremony; it will reduce process overheads without sacrificing quality, whilst delivering efficiently produced development.

The goal will be to build upon client-approved requirements as the project progresses, as opposed to delivering one large application at the end of the project. This approach is the RMG digital labs development process as governed by the current delivery agreement between Capgemini and RMG.

During each sprint phase all user requirements will be reviewed to ensure items are being built are as per RMG’s expectations and these will be managed via the product backlog. The product backlog will be analysed and reprioritised if necessary allowing for different set of functionality to be selected for future sprints during the RMG digital labs delivery process.

* 1. Testing in Scope

Capgemini will take the lead on management of the following test phases: -

**Functional Testing:**

* **Unit Testing**: Unit Testing is a level of software testing where individual units/ components of software are tested.
* **Smoke Testing**: Smoke testing is the initial testing process exercised to check whether the software under test is ready/stable for further testing.
* **Sanity Testing**: Sanity testing is carry out to check whether the bugs reported in previous build are fixed and verifies new functionality.
* **Skim Testing**: A testing technique used to determine the fitness of a new build or release of an AUT to undergo further, more thorough testing.
* **System Testing**: Black-box type testing that is based on overall requirements specifications; covers all combined parts of a system
* **Regression Testing**: The purpose of regression testing is to confirm that a recent program or code change has not adversely affected existing features.

**Non-Functional Testing:**

* **BaseLine Testing:** Baseline tests form a very important part of the performance test methodology. Done for each script with 1, 2, 5, 10, 20 and 50 users to determine baselines for mainly response times.
* **Load Testing:** Test system with multiple users to determine performance under load. The number of users is usually specified by the performance test requirements.
* **Stress Testing:** Load the system to its breakpoint. This is to determine the system break point or threshold. How the system breaks and recovers should also be monitored.
* **Performance Testing:** Performance testing, a non-functional testing technique performed to determine the system parameters in terms of responsiveness and stability under various workload. Performance testing measures the quality attributes of the system, such as scalability, reliability and resource usage.
* **Soak Testing:** Testing a system under load for an extended period of time to establish stability and behavior under sustained use.
* **Volume Testing:** Testing a system with a certain amount of data. Usually includes high data and throughput volumes.
  1. Not in Scope

Need HLD document

* 1. Test Phase Overview RMG Digital labs

Each test phase is a discrete form of testing with its own objectives, methods and requirements coverage and therefore a set of its own tests. This document will highlight the key flow of test stages to be employed for the (RMG digital labs).

|  |  |  |
| --- | --- | --- |
| **Responsible** | **Test Stage** | **Overview** |
| Capgemini | Unit Testing | Unit testing provides immediate feedback on the quality of the code being built in theory improve the overall quality of the deployment into proceeding phases. This testing will demonstrate that each code component or package configuration/integration functions according to its specification |
| Capgemini | System Test | To test, as part of each iteration, that the latest integrated build meets the specified requirements for different types of valid/invalid conditions. This will include: regression testing of functionality (from previous Iterations) retesting of any defect fixes. This will be a mixture of both manual and automated testing where permissible |

* 1. Test Deliverables

As part of the rmg digital labs delivery a number of test deliverables will be created by Capgemini and issued during each of the Test Phase.

| **Document Type** | **Purpose** |
| --- | --- |
| Test Strategy (This document) | Overall scope / approach for RMG digital labs testing highlighting any deviations to the current RMG Test Strategy (this document) |
| Test Plan / Approach Document | This will be a simple test approach which can be part of the change plan document itself. It will summarize the types of testing to be done for the particular requirements and also reference to the test cases if any |
| Testing Traceability Matrix based on user stories from JIRA (per Capgemini test phase) | Stories will be defined in JIRA. Each story describes one end result of value to the business. Stories are primarily described in terms of Acceptance test scenarios - concrete set(s) of steps describing the important test scenario(s).  Non-Functional stories such as Volume, Performance & Security will be defined separately from the Functional Tests.  Any such split would be detailed within the relevant Test Plan Stage  .  For every Test Phase describe the rationale for the selection of the chosen technique.  All Tests to be reviewed internally before being sent to RMG for review and subsequent approval.  The traceability matrix will be drawn from JIRA whereby stories have been linked to the tests to provide full testing traceability coverage |
| Digital Labs – Test Phase Exit Report | This report will be reviewed internally before being issued to RMG. |

* 1. RMG Digital labs Test Acceptance

The test acceptance criteria will be agreed for each enhancement request.

Sign off of Capgemini test phases will be given after review of the relevant Test Exit report. Overall acceptance for go-live will be given following completion of qa and qt testing.

The numbers in the grid below is kept for sample

|  |  |
| --- | --- |
| **Severity Type** | **Exit Level** |
| Critical | Agreed exit level = 0 |
| Major | Agreed exit level = 0 |
| Minor | Agreed exit level <=20 |
| Cosmetic | Agreed exit level <=30 |

1. Daily Test Execution

There will be no variation from the scripts during formal testing, except when problems are found with the scripts themselves (e.g. typing errors; the script instructs the user to use an incorrect button; expected result is incorrectly described; etc.). Above all, common sense must prevail.

* 1. Start of Day Testing

Each day on which testing is to be performed will start with a briefing session for all participants, outlining the plans for the day, issuing of Tests

* 1. End of Day Testing

There will be a de-briefing (“wash-up”) session at the end of each day, to review any problems or issues encountered and to review and agree the priorities of any issues raised.

* 1. Daily & Final Test Reporting

Daily test reporting will be issued to an agreed set of stakeholders based on the results of the testing undertaken that day.

A final Test Phase Exit Report will be produced detailing the testing phase undertaken which can be used by RMG as an input to make their Go / No Go decision on whether, or not, to continue to the next phase.

* 1. Test Recording

Tests will be recorded in JIRA during Testing. Each test will be signed by the tester to certify that the test ran as expected and any incidents have been recorded.

A summary of all Issues, Defects, Observations or Queries arising during Capgemini Testing will also be recorded in JIRA

* 1. Test Incident Escalation

Test Incidents, Observations or Queries raised during (Capgemini) System Testing will be taken up at the next level of governance. Wherever possible, additional evidence (e.g. screenshots) will be provided to assist problem resolution process.

* 1. Regression Testing

Regression testing is performed when a component part of the overall product is modified, and ensures that no new faults have been introduced as a result of this modification.

1. Test Data

Pending waiting for high level and requirement document.

1. Test Assurance / Governance

Pending waiting for high level and requirement document.

1. Assumption & Dependencies
   1. Assumptions

In creating this document, the following assumptions have been identified for test related activities:

|  |  |  |  |
| --- | --- | --- | --- |
| **Ref** | **Work stream** | **Assumption Description** | **Comments** |
| A01 | Test | Security impact has been completed but the overall requirements have yet to be agreed. | This will be drawn out in a future SoW |

* 1. Dependencies

No dependencies as now.

1. Test Governance
   1. Test Monitoring

Testing progress will be reviewed at least once a week but at critical points this will be more frequent (e.g. on a daily basis during formal test execution). The primary means of reporting will be a measure of completed deliverables/activities against those planned.

For the acceptance test phases, test preparation progress will be reported as the number of Test Conditions/Scenarios/Scripts completed against the number planned.

During the Development Iterations test execution, reporting will be email on a daily basis to the Work stream and Project Management team.

For the acceptance test, phases Test Execution will be reported both daily (snapshot information) and weekly (more detailed information) as follows:

* Summary status of Test Scenarios/Scripts executed (passed/failed/blocked/outstanding)
* Summary of test incidents/defects currently outstanding - showing quantities by state and severity
* Any serious problems encountered e.g. with quality of component under test, quality of test scripts/procedures, test environment, test resources, defect correction turnaround etc.
* Any changes to plans recommended or required because of testing to date.

The written content of test progress will be kept to a minimum (to fulfil their intended purpose) but will be supported by spreadsheets, tables, etc. in order to enable corrective actions to be taken quickly and for test progress to be expedited.

* 1. Defect Management

This project will use JIRA defect management for all type of Testing.

These tools will manage the systematic defect tracking process during the life of the project. Each defect identified will be reviewed by a senior tester or developer. Defects logged in the system will need associated defect numbers, to support cross-referencing and efficient defect management process.

* 1. Configuration Management

Capgemini is responsible for ensuring all test environments are subjected to release control. Test environments will be formally controlled after the Component Testing/Component Integration Testing phase. The Capgemini Test Manager will be responsible for authorising any changes to any hardware or software configuration in the test environments.

1. Test tools and environments
   1. Tools

The following will be used:

* Team forge: document repositories, under version control. These tool will be used to store and maintain formal test documents which have been agreed & signed off;
* Behat open source tool as identified for automation testing;
* Jmeter is used for performance, volume and stress testing;
* JIRA for Agile backlog, Defect logging and Test Management;
* Confluence working document repository;
* SauceLabs:Used to test our application on different platforms and browsers on cloud services.
  1. Automation Testing

Automated test tools are used to address both functional and non-functional automation testing. The functional automation is based on the Behat automation framework and non-functional testing is delivered by using Jmeter (performance testing).

* 1. Environment Usage

|  |  |
| --- | --- |
| **Environment** | **Use** |
| QA | The Quality Assurance (QA) environment caters for independent functional system tests to verify that the new system functions according to the specified requirements. |
| Development | This environment is used to perform system testing, system integration testing and user acceptance testing of the system and its associated interworking systems. |
| Pre-prod (Pre-production) | This environment is more representative of the target Production environment and is used to support the key non-functional testing including Performance and Volume Resilience testing. |

1. Project Roles and Responsibilities
   1. Roles

The table on the following page provides a summary of key responsibilities for the core testing related roles across the Programme.

| **Resource** | **Specific Responsibilities or Comments** |
| --- | --- |
| Testers  (Capgemini) | * Analyse business scope, requirements, user stories and /or use cases * Create manual test scripts based on the scope, requirements, user stories and /or use cases * Execute manual test scripts based on the scope, requirements, user stories and /or use cases * Asses the impacts and risk of the new functionality on the system & advise relevant stake holders * Test analysis – understand requirements for test (review of documentation, attendance at workshops/meetings, cross-team collaboration, etc.) * Test design - generate test conditions and test cases/scenarios/scripts (i.e. ‘what’ to test) * Test implementation – test scripting/procedures (i.e. ’how’ to test) * Test execution – execute tests, including the logging of results and raising of test defects * Execution of regression tests and defect retesting when required * Peer review QA checks of test scripts/test execution of other analysts * Report progress to Capgemini SDM and Technical Lead |
| Performance Test Specialist  (Capgemini) | * Provide Performance testing input to project test plans * Prepare and execute Performance Test scenarios and Scripts * Escalation of any issues affecting progress to the Capgemini test manager * Report progress to the Capgemini test manager |