https://www.travelrepublic.co.uk/imagesv3/dnata/dnata.png

|  |  |
| --- | --- |
| **Project Name:** | Honeycomb Test Plan |
| **Version:** | 0.1 |
| **Date:** | 24/01/22 |
| **Author:** | Venkata Satya Prasad Sajja |
| **Reviewed:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Change Summary** |
| 0.1 | 24/10/22 | Venkata Satya Prasad Sajja | First Draft |

Table of Contents

[1. Introduction 4](#_Toc95734318)

[2. Test Objective 4](#_Toc95734319)

[3. Scope 4](#_Toc95734320)

[In Scope 4](#_Toc95734321)

[Out of Scope 4](#_Toc95734322)

[4. Test Environments 5](#_Toc95734323)

[5. Tools 5](#_Toc95734324)

[6. Browsers / Device Coverage 6](#_Toc95734325)

[7. Test Approach 7](#_Toc95734326)

[Audience 7](#_Toc95734327)

[Smoke Testing 7](#_Toc95734328)

[Feature Testing 8](#_Toc95734329)

[Systems integration (SIT) or End to end testing 8](#_Toc95734330)

[Regression Testing 9](#_Toc95734331)

[User Acceptance Testing 9](#_Toc95734332)

[Automation Testing 10](#_Toc95734333)

[Non-Functional Testing 10](#_Toc95734334)

[8. Test Design 11](#_Toc95734335)

[9. Test Execution 11](#_Toc95734336)

[10. Test Data Approach 12](#_Toc95734337)

[11. Defect Management 12](#_Toc95734338)

[12. CI/CD Integration 13](#_Toc95734339)

[13. Assumptions 13](#_Toc95734340)

[14. Risk and Mitigation Factors 14](#_Toc95734341)

[15. Project Timelines 16](#_Toc95734342)

[16. Communication Plan & Resources 16](#_Toc95734343)

[Project Manager 17](#_Toc95734344)

[Business Analyst 17](#_Toc95734345)

[QA Manager 17](#_Toc95734346)

[QA Engineer 18](#_Toc95734347)

[Engineering Manager 18](#_Toc95734348)

[17. Deliverables 18](#_Toc95734349)

[18. Test Plan Review/Approval 19](#_Toc95734350)

# Introduction

This document describes the overall approach of the testing for any Honeycomb projects from initiation till it goes live. It briefs the testing activities, roles, responsibilities and practices to be used during the project. It covers

* Test Strategy: rules the test will be based on, including the givens of the project (e.g.: start / end dates, objectives, assumptions); description of the process to set up a valid test (e.g.: entry / exit criteria, creation of test cases, specific tasks to perform, scheduling, data strategy).
* Execution Strategy: describes how the test will be performed and process to identify and report defects, and to fix and implement fixes.
* Test Management: process to handle the logistics of the test and all the events that come up during execution (e.g.: communications, escalation procedures, risk and mitigation, team resources)
* Scope, testing efforts, entry and exit criteria that must be fulfilled prior to accepting products/features into testing environment.

It ensures the stakeholders are clear on the test phases approval criteria.

QA Managers, Test Leads, Product owner, Engineering Manager, Project Manager and any Business UAT resources shall review the document prior to the end to end testing phase.

# Test Objective

To ensure quality delivery of Honeycomb based mobile and web applications. High level end to end and User Acceptance scenarios approved by the business works as expected across the full architecture.

# Scope

This section lists on what’s in the scope of testing, what’s out of scope, so that all the stakeholders are clear on the expectations.

### **In Scope**

* Travel Republic (UK & IE), Sunmaster on Honeycomb platform
* Regression on Adaptive V3 Website (until migration to Honeycomb platform)
* Non-Functional requirements (TBC ... waiting on to get more details on the same)

### **Out of Scope**

* BAU on V2, V3 and Sunmaster
* Performance testing
* Security testing

# Test Environments

Environments will be provided by the infrastructure team and owned by development team. Detailed requirements of the environments and specifications are outside the scope of this document. Below are the environments available for Honeycomb testing.

* QA Environment
* Staging / Pre-Production Environment

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Testing Type** | **Environment** |
| 1 | Unit Testing | Dev |
| 2 | Feature Testing | QA |
| 3 | System Integration Testing (SIT) | Staging |
| 4 | Non-Functional Testing | Staging (TBC) |
| 5 | User Acceptance Testing (UAT) | Staging |

# Tools

This section list the tools used by the QA team to support the activities of both the manual and automation teams.

* TestRail - Test Management tool (Test case creation, Execution and Reporting)
* Selenium WebDriver - UI Automation Tool
* API Automation – Rest Sharp
* Test Development - C#, NUnit, Specflow
* Test Reporting - Report Portal
* Defect Tracking - Jira
* Source Control – Bitbucket

# Browsers / Device Coverage

This section lists the browsers that are tested on desktop and mobile devices. It also lists the physical devices available in-house.

**Browsers tested for Desktop Websites**

|  |  |  |  |
| --- | --- | --- | --- |
| **OS (Versions)** | **Browser** | **Priority** | **Devices** |
| Windows 10 | Chrome | P1 |  |
| Macintosh Catalina | Safari | P1 |  |
| Windows 10 | Edge | P2 |  |
| Windows 10 | Internet Explorer | P2 |  |
| Tablet – IOS 12 /13 | Safari | P1 | iPad Pro / iPad |
| Tablet – Android 9/4.4 | Chrome | P1 | Galaxy Tab |

* Testing will be done only on the latest version browsers.
* Functional testing is done on all P1 browsers. Any UI changes are tested on both P1 & P2 browsers
* Considering the timelines of testing, we might take a risk based approach, where in we will not repeat entire testing on all browsers on all domains.

Eg: TR UK on Chrome, TR IE on Safari, TR UK on Edge, TR IE on IE, TR UK on IOS and TR IE on Android

* Automation tests are run on the Chrome browser and additional browsers supported by Automation. Manual regression and Visual validation checks are done on P2 browsers.

**Devices and Browsers tested for Mobile Website**

|  |  |  |  |
| --- | --- | --- | --- |
| OS (Version) | Browser | Priority | Devices |
| IOS (15) | Safari | P1 | iPhone 13 |
| Android (11) | Chrome | P1 | Galaxy S21 |
| Android (10) | Chrome | P2 | Galaxy S21 |
| IOS (14) | Chrome | P2 | iPhone 12 |

* Functional testing is done on all P1 devices on respective browsers.
* UI changes will be tested on ­­both P1 & P2 devices on respective browsers.
* Mobile testing is mostly done on the real devices available in-house or on browser stack.
* Automation tests will be run on Android and IOS simulators on chrome browser.
* Considering the timelines of testing, we might take a risk based approach, where in we will not repeat entire testing on all browsers on all domains.

***Physical Devices Available In-house:***

<https://dnatatravel.atlassian.net/wiki/spaces/TD/pages/1207402691/Device+List>

# Test Approach

To achieve sign off for Honeycomb projects, the end to end testing will use SME’s from the business to validate systems. We will cover five types of test phases; Smoke Testing, Feature Testing, System Integration Testing, Regression Testing and User Acceptance testing.

Each functional test phase will have its Entry and Exit criteria defined.

### Audience

* **Project** **Team Members** perform tasks specified in this document and provide input and recommendations on this document.
* **Stakeholders Representatives** will take part in the end to end and UAT testing to ensure the business is aligned with the results of the test.
* **Stakeholders** will provide sign-off for each test phases.
* **Technical Team** ensures that the test plan and deliverables are in line with the design, provides the environment for testing and follows the procedures related to the fixes of defects.
* **Business Analysts** will provide their inputs on functional changes and test case sign off.

### Smoke Testing

**Purpose**: Smoke Testing, also known as “Build Verification Testing”, is a type of software testing that comprises of a non-exhaustive set of tests that aim at ensuring that the most important functions work.

The results of this testing are used to decide if a build is stable enough to proceed with further testing and to make sure critical defects are removed before the next levels of testing can start.

**Entry criteria**

* Environment access for testing the application should be given to the testing team.
* Happy paths test scope that will be used during smoke tests are approved by the Business Analyst.
* Adequate Test data is available.
* Completed and reviewed Test Cases/ Test scripts.
* Any test booking conventions are understood and communicated (e.g. limits on price, airlines that cannot be used for test bookings etc.)

**Exit criteria**

* Severity 1 (Critical) and 2(Major) defects are resolved and implemented.
* Smoke tests are executed and passed.
* **S3** defects are communicated to the business team.
* Defects were documented, reported and can be found in Jira.
* Applicable sign-off on testing was obtained.

### Feature Testing

**Purpose:** Actual implementation of the feature or enhancement is tested thoroughly in this phase. All the tests written are executed to make sure the implementation is as per the requirements. Any UI changes made are tested on all supported domains, browsers and device combinations.

**Entry criteria**

* Smoke tests have been completed successfully.
* Code freeze across all systems in scope.
* Environment access for testing the application should be given to the testing team.
* Test scope that will be used during the feature testing is approved by the Business Analysts.

**Exit criteria:**

* All feature tests have been completed successfully.
* S1 and S2 defects found as part of the feature testing phase have been fixed
* **S3** defects are communicated to the business team

### Systems integration (SIT) or End to end testing

**Purpose**: End-to-end testing is a methodology used to test whether the flow of an application or system is performing as designed from start to finish. The purpose of carrying out end-to-end tests is to identify system dependencies and to ensure that the right information is passed between various system components and systems. This is the primary goal of this plan.

**Entry criteria**

* Feature tests have been completed successfully.
* Code freeze across all systems in scope.
* Environment access for testing the application should be given to the testing team.
* Test scope that will be used during the End to End testing are approved by the Business Analysts.

**Exit criteria:**

* System test and System integration test cases have been successfully executed.
* Priority 1 End to End tests have been completed successfully, covering all key Business components
* S1 and S2 defects found as part of the End to End test phase have been fixed
* **S3** defects are communicated to the business team and agreed.
* Results of executed tests must be shared /discussed with product management/Business analyst

### Regression Testing

Regression Testing is defined as a type of software testing to confirm that a recent program or code change has not adversely affected existing features. This testing is done to make sure that new code changes or the migration of code base should not have side effects on the existing functionalities.

The Regression test phase is a crucial testing phase before a Release, as it gives stakeholders a high level of confidence on the integrity of a Release before it is deployed onto the LIVE/Prod environment.

**Entry Criteria**

* Exit criteria for system functional testing have been met.
* There is a code freeze on the pre-production environment.
* Regression test scope and Test Scripts are ready and reviewed

**Exit Criteria:**

* No S1 and S2 defects identified
* All S3 defects are communicated to the business team.
* Successful completion of the Regression testing phase
* All bugs are tracked and recorded. They should be shared with the SMEs and Business

### User Acceptance Testing

User Acceptance Test (UAT) focuses on validating the business logic and providing product approval. It allows the business SME’s, stakeholders and end users to complete one final review of the system prior to deployment.

The UAT will be performed by the business SME’s, end users and relevant stakeholders. However, the QA team can support by giving the end user or Business representative, guidelines on the execution of the UAT test phase. The test scope of the UAT phase will be devised by the business resources conducting the UAT.

Since the business users have the most knowledge on how the system adapts to them and meets their use case, it may happen that the users do some validation not contained in the test cases.

**Entry criteria**

* All prior test phase has met exit criteria
* The scope of UAT test is defined and test cases are written using the Test case template provided in this test plan.
* Test data is set up for UAT
* User(s), business SME’s and stakeholders have been identified for UAT

**Exit criteria**

* Zero unresolved S1 and S2 defects open
* Business process works
* UAT Sign off meeting held with all stakeholders.

### Automation Testing

Automation testing helps us to improve the pace at which we can release the changes to live. It increases the efficiency of testing, productivity, throughput, higher levels of accuracy, improved coverage and consistency.

In Dnata, we have an in-house developed framework that will support development of API and UI automation. For API automation, we use Rest Sharp library to send and capture request details to validate the APIs. For UI Automation, we use selenium web driver and page object models for driving the UI tests. We currently use UI automation for desktop and mobile websites. ‘Browser / Device Coverage’ section of the document covers the details of the browsers and devices covered.

Below is the priority in which the test cases are picked up automation

* Smoke
* Regression

**Smoke Test Packs:**

Smoke Test Packs are run before accepting the build to the QA or PP environments. They are also run on live environment once the release goes out.

**Regression Test Packs:**

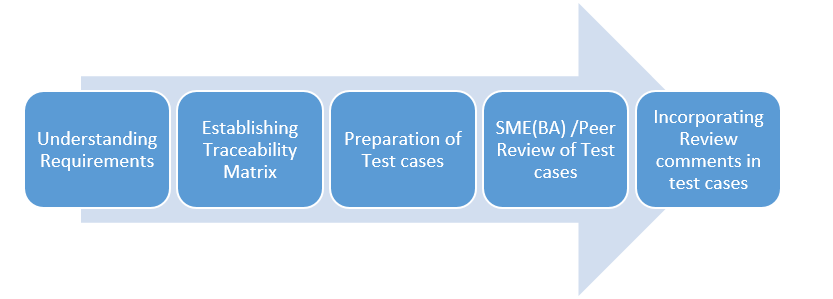
Regression Test Packs are scheduled to run nightly on QA environments where the changes are deployed frequently. They are also run on PP environment before we signoff the release.

### Non-Functional Testing

Where possible, the QA function will attempt to undertake Non-functional testing of the applications to ensure that current levels of performance, concurrency and reliability are preserved or improved where possible.

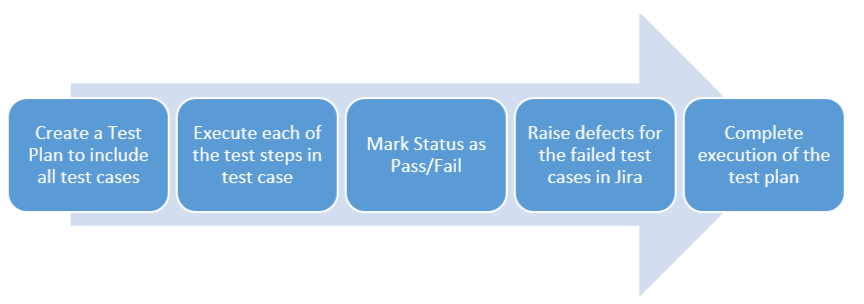
Honeycomb QA function does not currently have the capability to test and baseline information to support extensive non-functional testing, such as Performance and Stress Testing due to the limitation on test environment. However, a level of testing and observation should be attempted.

# Test Design



* Tester will understand each requirement and prepare corresponding test cases to ensure all requirements are covered.
* Each Test case will be mapped to Story as part of Traceability matrix.
* Each of the Test cases will undergo review by the Business Analyst. If there are updates needed to Test cases, the testers will update and finally obtain approval and sign-off.
* Sign-off for the test cases would be communicated through mail by Business Analysts.
* Any subsequent changes to the test cases will be communicated and agreed with the Business Analyst.

# Test Execution



* Once all test cases are approved and the test environment is ready for testing, Testers will start the Smoke test to ensure the application is stable for testing.
* Test Plan is created in TestRail for Test Execution with all the applicable test cases for all supported browsers and devices.
* Each Tester will be assigned Test cases to execute.
* If there are any showstopper (S1) found during Smoke testing, it will be escalated to the lead developer to triage for fixes.
* Each tester performs step by step execution and updates the executions status. The tester enters Pass or Fail Status for each test step.
* If any issues are detected, defect will be raised as per severity guidelines specified in this Test Strategy, which can be found under the Defect Management section, detailing steps to simulate along with screenshots if appropriate.
* If there are any defects that were discovered outside of the test steps, such defects need to be captured in Jira and mapped against the test case level or at the specific step that issue was encountered after confirming with the Test Manager.
* This process is repeated until all test cases are executed fully with Pass/Fail status.
* During the subsequent cycle, any defects fixed applied will be tested.

# Test Data Approach

During all test phases, the test environment will contain pre-loaded test data to be used for testing activities.

In case the data is not pre-loaded, we need to identify the team members who have all the required access to create test data both on PP and Live, to avoid any delays in the test execution or releases.

# Defect Management

Defect is the difference observed between expected behavior and the actual results. Essentially it marks the deviation of the products behavior from customer requirements.

* Defects will be raised and tracked in Jira. The development team will gather additional information, when needed from the Defect Reporter to enable the development team to work on a fix.
* It is the responsibility of the tester to open the defects, link them to the corresponding script, assign an initial severity and status, retest and close the defect; it is the responsibility of the Business Analyst to triage, review the severity of the defects. It is the responsibility of the PO to prioritize.
* It is the responsibility of the developer to assign a fixed defect to the correct status so that the tester can validate the fix provided.
* UAT testing and validation is carried out by the Business. The Testing team will be on hand to help with additional information and guidance if needed for raising defects

Defects are assessed and assigned a Severity level to indicate the impact of the defect on the solution as outlined below:

|  |  |  |
| --- | --- | --- |
| **Severity** | **Title** | **Description** |
| S1 | Critical / Show Stopper | A defect that completely blocks testing of the product or renders the solution completely unusable. |
| S2 | Major or Severe | A major defect that occurs when the functionality vastly deviates away from the expectations or not doing what it should be doing.  It causes a lack of vital program functionality, however there is a workaround. |
| S3 | Moderate/ Normal | A defect that occurs when the product or application doesn’t meet certain criteria or still exhibits some unexpected behavior. However, the functionality is not impacted. |
| S4 | Low or Minor | A minor bug occurs when there is almost no or little impact to the functionality, but it is still a valid defect that should be corrected. |

Standards for raising defects can be found here: [Raising a defect](https://dnatatravel.atlassian.net/wiki/spaces/TD/pages/203685908/Defects)

# CI/CD Integration

CI is a modern software development practice in which incremental code changes are made frequently and reliably. Automated build-and-test steps triggered by CI ensure that code changes being merged into the repository are reliable. The code is then delivered quickly and seamlessly as a part of the CD process. In the software world, the CI/CD pipeline refers to the automation that enables incremental code changes from developers’ desktops to be delivered quickly and reliably to production.

In Honeycomb project, we will have separate pipelines for Automation. Automation pipeline is will be integrated with development pipeline. Auto tests will be triggered when the changes are merged and deployed to environments.

Plan as of now will be, Smoke tests will get triggered once the changes are merged to QA, PP or Live environments. Regression tests will run on QA or PP environment nightly as scheduled.

# Assumptions

**Key assumptions**

* Business will be available to check/sign off the end to end requirements and test cases.
* Business will sign off the feature through show and tell sessions and UAT phase.
* Business will be available to check/sign off the end to end tests and data flow to downstream applications (Internal / External)
* Defect management required for issues raised by business UAT resource. E.g. Issues logged in Jira on appropriate work streams linked to an appropriate epic on the Honeycomb Jira board.

**General Assumptions**

* The QA team and business resource will be provided with access to Test environment
* The QA team assumes all necessary inputs required during Test design and execution will be supported by Development/Business Analysts appropriately
* Smoke test activities will be executed as an acceptance criterion before a build is accepted for a test cycle
* Test environment and preparation activities will be owned by the Dev Team
* Dev team will provide Defect fix plans based on the Defect meetings during each cycle to plan.
* Business Analyst will review and sign-off all Test cases prepared by QA team prior to start of Test execution
* Defects will be raised and tracked through Jira. Any defect fixes planned will be shared with the QA team prior to applying the fixes on the Test environment
* Project Manager/Business Analyst will review and sign-off all test deliverables, including the test plan article
* QA team will manage the testing effort with close coordination with Project PM/Business Analyst
* There is no environment downtime during test due to outages or defect fixes.

# Risk and Mitigation Factors

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Prob.** | **Impact** | **Mitigation Plan** |
| **SCHEDULE**  Any blockers during the functional testing will result in testing being suspended until the issues are resolved. | Low | High | ·     The testing team can control the preparation tasks (in advance) and the early communication with involved parties.  ·     Some buffer has been added to the schedule for contingencies. |
| QA End-to-end testing may be impacted due to late delivery of the features/product | Medium | High |  |
| Additional deployment to the testing environment after the completion of the test case design or after the test has been executed will cause delay, as more test need to be created and run to validate the changes. Therefore, may in turn affect testing effort/resources. | Medium | High |  |
| **RESOURCES**  Not enough resources during the test phases, due to QA team members going on holiday/Sick leave. This may impact the testing quality. | Medium | Medium | Holidays and vacation have been estimated and built into the schedule. Sick leaves however cannot be mitigated against. |
| **DEFECTS**  Defects discovered late, may be due to unclear specifications would take longer or cause rework | Medium | High | Defect management plan is in place to ensure prompt communication to fix defects. |
| **SCOPE**  Scope completely undefined | Low | High | Communication between the Business Analysts and Testers to ensure expectations and expected results are well understood before the testing phases. |
| Changes in requirements after test cases have been created or tests have been executed may lead to repeating the testing and updating test cases. This may cause major delay. | Low | High | The requirements scope is locked down before the testing phase commences. No changes are to be made to the current workflow and requirements. Any changes should come as Change Request. |
| **Environment**  Issues that are not present on the test environment(s) may be discovered on Production. | Low | High | Post production smoke test will be carried out to after code deployment. In case of any issues found on live will be triaged and decision will be taken to rollback or fix forward. |
| Environment issues like No search results, hotel rooms sold out, APIs are down | Medium | High | We need to alert the right people to get this sorted and work towards a long-term solution to avoid such problems in further. |
| Any failure in smoke testing will result in interruption in Testing. | Medium | Medium | A smoke test Health-check functionality is now implemented. There is now an automated Smoke test pack that is run after a new deployment or fix – Derwent\existing brand websites. |

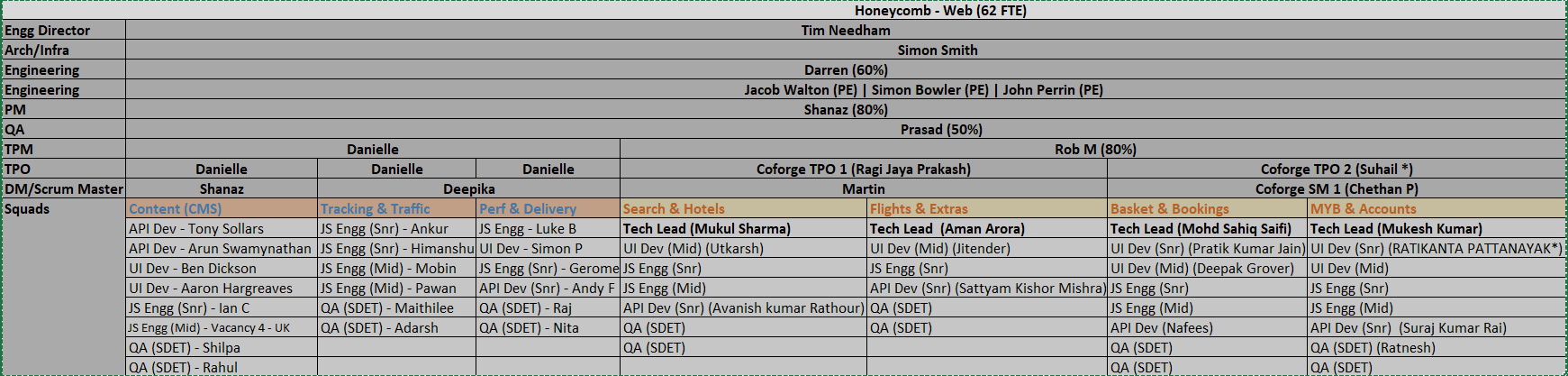
# Project Timelines

This section shows the project plan, timelines and major milestones where we will release the changes to Live environments.

Section will be updated once we have more details.

# Communication Plan & Resources

The following list defines in general terms the expectations related to the roles directly involved in the management, planning or execution of the test for the project.



### Project Manager

* Reviews the content of the Test Plan, signs off on it.
* Managing the scope of the project
* Coordinates with the business resource for End to End SME’s and UAT activities.
* Attend triage meetings
* Keep project team and leadership informed of potential software delivery date slips based on the current schedule
* Keeps the Programme Manager up to date on project progress

### Business Analyst

* Add Acceptance criteria and high level scenarios
* Schedule 3 Amigo meeting with Dev and QA
* Reviews end to end test cases, SME checks and UAT tests
* Attend Triage meetings

### QA Manager

* Ensure entry criteria is used as input before starting the execution of a test phase.
* Develop test plan and the guidelines to create test conditions, test cases, expected results and execution scripts.
* Provide guidelines on how to manage defects.
* Communicate to the QA team any changes that need to be made to the test deliverables or application and when they will be completed.
* Provide support to business SME’s\ UAT activity.
* Acknowledge the completion of a section within a cycle.
* Gives the OK to start next level of testing.
* Facilitate defect communications between testing team and technical / development team.
* Attend Triage meetings

### QA Engineer

* Develop test conditions, test cases, expected results, and execution scripts for Smoke test and Functional test phase.
* Perform execution and validation of the Smoke and Functional test phase
* Identify, document and prioritize defects per the guidance provided by the Development Lead
* Lead the Re-test of fixes after software modifications have been made per the schedule.
* Prepare testing metrics and provide regular status updates
* Provide support to business SME’s\ UAT activity.

### Engineering Manager

* Review testing deliverables (test plan) and provide timely feedback.
* Ensure the global governance and end to end of the delivery is in line with agreed approach.
* Support the development and testing processes being used to in the project.
* Keep project team and leadership informed of potential software delivery date slips based on the current schedule.
* Define processes to facilitate the initial and ongoing migration of components.

# Deliverables

This section lists the outcomes of the QA Activities, Ownership and responsibility of reviewing those outcomes. Deliverables will be shared over an email or will part of tools used by the team. In case the deliverables are inside a tool, team will make sure required access is provided to the reviewers before requesting for the review.

|  |  |  |
| --- | --- | --- |
| **Deliverable** | **Author** | **Reviewer** |
| Test Plan | QA Manager | Project Manager/ Engineering Mangers/QA Managers/Business Analyst |
| End to End Test Cases | Test Team | Business Analyst |
| UAT test cases | UAT Team | QA Team/ Business Analyst |
| Logging Defects | Test Team | QA Manager/ Business Analyst/ Triage group |
| Status report | Test Team | QA Manager / Project Manager |
| Test Closure report | QA Manager | Project Manager |

# 

# Test Plan Review/Approval

This section lists the team members responsible for reviewing and approving the test plan. As part of the review process they can request to add additional sections or remove some of them that are not required or add value to this document.

|  |  |  |
| --- | --- | --- |
| Name | **Role** | **Approval Date** |
| Tim Needham | Engineering Director |  |
| Darren Carr | Engineering Manager |  |
| Liam Hanlon | Business |  |
| Shanaz Begum | Project Manager |  |
| YTD | Product Owner |  |
| Danielle Hewitt / Robert Mead | Business Analyst |  |
| Deepika Chaudhry | Delivery Manager |  |
| Venkata Satya Prasad Sajja | QA Manager |  |