

Author(s):

Carl Arndt – QA Manager

Carl.arndt@fcb.com

08

**Fall**

[Client Name]

[Project Name]

Test Plan

Date: 12/21/2016

Version: 4.4

# Index

[Client Name] 1

[Project Name] 1

Test Plan 1

0 Index 2

1 Document Control 3

2 Overview 4

2.1 Introduction 4

2.2 Purpose 4

3 Features to be Tested 5

4 Features not to be Tested 5

5 Test Approach 5

5.1 Planning 5

5.2 Execution 6

5.3 Defect Process 6

5.4 Defect Priorities 7

5.5 Test Reporting 7

5.5.1 Metrics 7

5.5.2 Certification 8

6 Exit / Acceptance Criteria 8

7 Suspension / Resumption Criteria 8

7.1 Suspension Criteria 8

7.2 Resumption Criteria 8

8 Test Deliverables 9

9 Configurations to Test 9

10 Responsibilities 11

11 Schedule 11

12 Testing Tasks 12

12.1 Tools 12

12.2 Testing Types 12

12.2.1 Smoke Testing 12

12.2.2 Exploratory Testing 12

12.2.3 Functional testing 12

12.2.4 Compatibility Testing 12

12.2.5 Retesting (AKA Validation Testing) 13

# Document Control

# Document location

|  |
| --- |
| Location |
| TBD |

# Author

|  |  |  |
| --- | --- | --- |
| Name | Position | Contact |
| Carl Arndt | QA Manager | Carl.arndt@fcb.com |

# Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author/editor | Description/Summary of changes |
| 4.4 | 12/21/2016 | Carl Arndt | Added Device section, Schedule section, Reviewer section, reworded Suspension/Resumption Criteria section. Corrected bullet points, bolding, etc. |
| 4.3 | 07/19/2016 | Carl Arndt | Various formatting improvements, updated Exit/Acceptance Criteria, |
| 4.2 | 10/12/2015 | Carl Arndt | Updated Summary and Purpose |
| 4.1 |  | Carl Arndt | Initial document creation and styling. |

# Reviewed by

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Name | Position | Approval date |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Overview

## Introduction

FCB Chicago Digital is developing a new site for [Client Name]. The QA Department will test the [Project Name] within the timelines provided. The following details describe the methodology and strategy involved in testing the project.

## Purpose

“Adventure is just bad planning.”

- Roald Amundsen

Every project that passes through QA has either an implicit or explicit test plan.

The implicit test plan is the plan that covers testing activities across a group of generally smaller projects. An email campaign is an example of a project that falls under an implicit test plan.

In contrast, the explicit test plan is written for one specific project – typically written for large project, such as a large web site or other large project.

The Purpose of this test plan is to describe all the testing activities for the [Project Name] project and to ensure that the reader understands how these test activities play a role in the overall software development process.  
  
This test plan is written so it follows IEEE829 specifications of test plan writing to help ensure that every relevant objective in the area of quality is planned for. IEEE is the Institute of Electrical and Electronics Engineers – the largest organization of technical professionals in the world. The IEEE has written process and procedures specifications for all areas of engineering. Specification 829 describes the software test plan writing process and provides a time proven process for creating test plans.

# Features to be Tested

1. Feature 1
2. Feature 2
3. Feature 3

# Features not to be Tested

* Tagging / Tracking Testing
* Functional Load / Performance Testing
* Spellchecking / Proofreading / Fonts

# Test Approach

## Planning

The QA Department will prepare for testing in a number of ways.

1. Creating this Test Plan – Creating a test plan and obtaining the proper buy-in from the other stakeholders on the project will be an important first-step in the process of delivering high quality products. This plan will be the overarching document that describes all the facets of testing that will be performed with this project moving forward. When all parties are on the same page with how QA is planning on proceeding, there will be less risks and more efficient testing.
2. Review Test Deliverables – Any documents that pertain to the project will be reviewed by QA prior to creating test cases. Examples include Use cases, Wireframes, Design documents.
3. <TBD> Create / Review Test Cases – QA will write the test cases that will be executed with the project under test. These test cases will be reviewed by at minimum the QA Manager, and ideally, the Development manager on the project as well. This will ensure that both development and QA are on the same page with what is being tested to ensure proper test case coverage and reduce the risk of something not being tested. If test cases are created for this project they will be housed in or test case manager named TestRail.

## Execution

QA will test as much of the site for functionality and browser compatibility as possible in the time allotted, as described in the Schedule section below. New feature testing will have a higher priority over retests as the test cycle progresses.

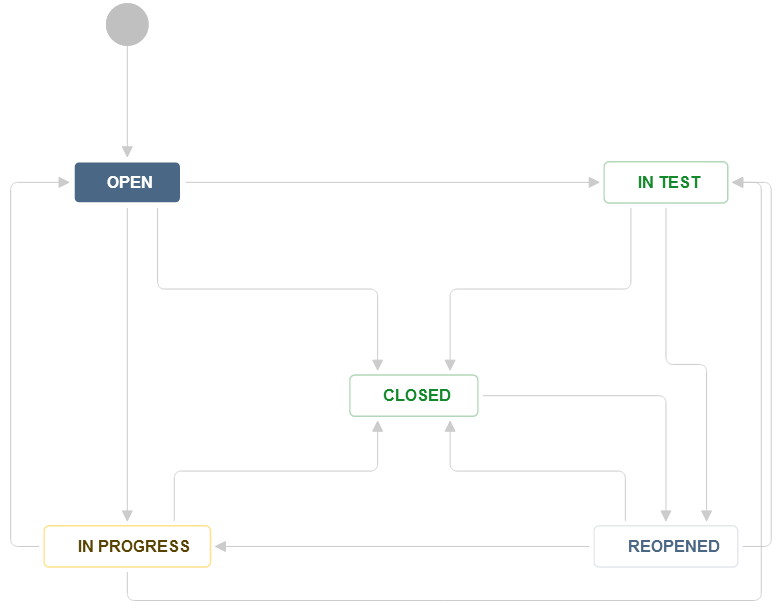
Any defects found will be entered into Jira and assigned to the Lead Role in Jira for the project. These tickets will follow the standard defect ticket workflow, as specified in the workflow section of Jira. These new defects will be vetted by the Project Manager to determine if they are indeed valid defects. Any valid defects will be assigned to the Development Manager for resolution.

The QA Manager will report status on defects entered and facilitate the issue resolution of these tickets.

The QA department will follow established practices and processes on planning for and executing test cases to uncover any issues with the system under test. When a potential defect is discovered, the tester will log the defect into Jira following the defect process below.

## Defect Process

As the testers are following testing processes and running tests of the preceding testing types, issues are likely to be found. When a perceived issue is found, it will be entered into the defect process. The defect workflow below illustrates how the issues move through the statuses.



**Example 1**: QA logs an issue into Jira. The ticket has all the pertinent information that the team needs to triage and deal with the issue. In the simple process flow, the Development Manager receives the defect, assigns the ticket to a developer who fixes it, marks the defect as Fixed and assigns it back to QA for retesting (verification that the defect is fixed). QA verifies the fix, and closes the issue.

**Example 2:** QA logs an issue, Development Manager determines that the issue is not valid and closes it with a resolution of “Won’t fix” and adds a comment as example ‘working as designed’.

**Example 3:** QA logs an issue; the Development Manager assigns the issue to a developer who addresses the issue. The developer marks the issue fixed, assigns it back to QA and the tester retests it and finds that the issue remains. The tester adds their comments and changes the status to ‘Reopen’ and assigns it back to the developer.

In most cases, the person who creates the issue most of the time is the one responsible for closing it. Example 1 is the most common workflow.

Test cases will continue to be executed until complete at such time the tester can perform exploratory / non scripted testing. Retests (verifying that defects are fixed) can be performed after all functional/compatibility tests have been run first.

## Defect Priorities

These are all of the possible priorities that are associated with an issue. Priorities allow the mitigation of issues in order of importance.

* **Blocker** - The project cannot release with this defect unresolved or this issue is preventing QA from testing other features
* **Critical** – Issue is causing crashes and/or data loss
* **Major** – Functionality missing / broken, no work-around
* **Minor** – Functionality missing / broken, work-around available
* **Trivial** – Cosmetic, very small issue, lowest priority issue

## Test Reporting

Two areas of reporting the information that QA finds are through Metrics and the Certification.

### Metrics

* Tests Executed / Percentage of tests executed
* Tests Passed / Percentage of tests passed
* Tests Failed / Percentage of tests failed
* Defects found
* Defects by Status
* Defects by Priority
* Defects by Assignee
* Hours used testing / Hours remaining to use on testing

### Certification

The build is certified when it passes all Exit / Acceptance criteria for said build. When testing has passed all criteria, the build is said to have passed the Acceptance criteria and QA attaches a QA Certification Document to the ticket in Jira. The certification document is how the QA team communicates ‘Yes, this build meets the required specifications and is ready to be delivered to the client.’

# Exit / Acceptance Criteria

The Exit or Acceptance criteria is a list of all things that must happen before the build is released to the customer – in other words, before the build exits our department and is accepted by the customer. I included a general template of typical criteria here, but more can be added.

* The build/files has been tested on required browsers and platforms
* All Blocker Defects Resolved
* All Critical Defects Resolved
* All Major Defects Resolved
* Minor and Trivial Defects are entered into Jira when found and triaged. A plan for resolution is created and approved by the Account team
* Overall Functionality of site working as expected
* Overall look and feel of the build is as expected

# Suspension / Resumption Criteria

## Suspension Criteria

* The Development Manager or Project Manager asks us to put testing on hold
* QA finds a number of blocking defects or other issues that prevent executing tests on the build/website

## Resumption Criteria

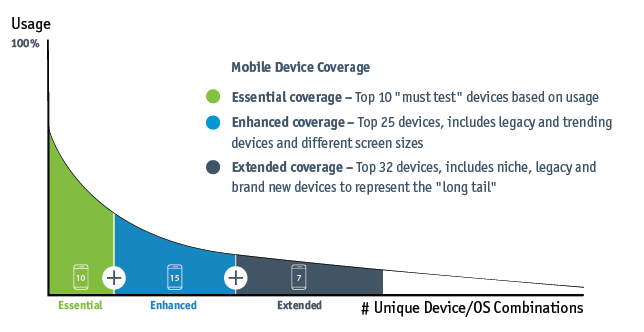
* + When the build/website blockage has been resolved and QA can make reasonable progress to completing testing if they resume testing
  + All issues that caused the testing to be suspended have been resolved.

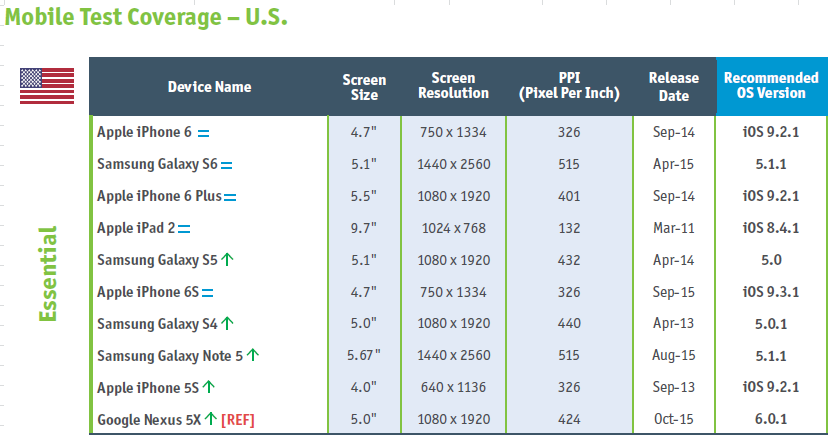
# Test Deliverables

* This Test Plan
* Defect Report
  + List of defects including status and priority
  + Blocking defects should be clearly marked
  + High level descriptions of ‘the big picture’ of the defects
  + Test Case Structure
* Test Status Reports – The progress of testing will be communicated to the team and will include the following metrics:
  + # of Test Cases Written
  + Status of Testing (Testing / On Hold / Testing Resumed (Date/Time) )
  + # of Test Cases Executed
  + # of Test Cases Passed / Failed / Blocked
  + # of Test Cases N/A – Due to a test case covering a now out of scope requirement
* Test Certificate – QA attaches a certificate test request ticket in Jira at the end of testing when it is decided that the project under test meets the Acceptance/Exit Criteria.

# Configurations to Test

According to Perfecto Mobile’s (a Testing Consulting firm that partners with test teams to allow for automated testing integrations and support) Digital Test Coverage table for July 2016, FCB has most of the ‘essential’ devices in our test lab. Note: this chart is current and was published before Apple’s release of the iPhone 7S and Samsung’s Galaxy S7, which we both have in our test lab.





The following Platforms and Browsers will be included in the testing effort.

<Add/Remove configurations in the following table as needed. The included configurations are for examples>

|  |  |  |
| --- | --- | --- |
| Operating System | Browser | Device |
| Win 7 | IE11 | N/A |
| Win 7 | Chrome 51 | N/A |
| Win 8.1 | Firefox 47 | N/A |
| Win 10 | Edge 14 | N/A |
| OS X El Capitan | Safari 9.1 | N/A |
| Android 5.0 | Chrome 51 | Android S5 |
| Android 5.1.1 | Chrome 51 | Samsung Galaxy Note 4 |
| Android 5.1.1 | Chrome 51 | Google Nexus 6 |
| Android 6.0 | Chrome 51 | Android tablet |
| IOS 8.1 | Safari 8.1 | iPhone 6 |
| IOS 9.1 | Safari 9.1 | iPad Air 2 |

The following table contains the devices we need to acquire in order to provide essential test device coverage.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Operating System | Browser | Device | Cost | Link to Acquire |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Responsibilities

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Administers Project | Delivers Files / Build | Functional Test Plan / Administer Testing | Functionality Testing | Creative & Content Test Plan | Functional Issue Resolution |
| Project Manager | **** |  |  |  |  |  |
| Development Manager |  | **** |  |  |  | **** |
| Functional QA Manager |  |  | **** |  |  |  |
| Functional QA Analyst |  |  |  | **** |  |  |
| Strategic Analysis Team |  |  |  |  | **** |  |

# Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| Task Name | Start | Finish | Comments |
| Test Planning |  |  |  |
| Site IR 1 |  |  |  |
| QA Testing – Cycle 1 |  |  |  |
| Site IR 2 |  |  |  |
| Client Review #1 |  |  |  |
| QA Testing – Cycle 2 |  |  |  |
| Cycle 3 Regression |  |  |  |
| Launch |  |  |  |

# Testing Tasks

Testers will use the following tools and testing types during their testing of the project.

## Tools

QA will be utilizing the following tools during the testing process for this project.

* Basecamp – <Project Management> Any QA progress reports or status updates will be communicated to the team here. Team communications will all flow through Basecamp
* Jira - <Issue Tracking> – Defects will be entered and tracked in Jira
* TestRail <Test Case Management> - If it is determined that test cases will be created, they will be manager in TestRail

## Testing Types

QA will be running different types of test cases during testing of the project. Here is a description of those types.

### Smoke Testing

There will be a small sub-set of tests that will be marked as ‘Smoke Tests’. These tests will be high level general tests that are run at the beginning of a test cycle to ensure that the latest build is ‘test worthy’. If these tests pass, then regularly scheduled testing can occur. If the tests fail, we know that the build is not suitable for testing to start. These tests will be identified in the test case set.

### Exploratory Testing

This is testing where the user has no set script and tests functionality freely. While the software is being tested, the tester learns things that together with experience and creativity generate new good tests to run.

### Functional testing

Verifying that the code functions as it should. Any unexpected behavior will be investigated and defects created and assigned to development.

### Compatibility Testing

Compatibility testing is verifying that the site/project is compatible with different browsers and devices. Any specifics pertaining to the project should be described here.

### Retesting (AKA Validation Testing)

Testing defects that have been fixed or resolved by development is called ‘Retesting’ or ‘Validation testing’. Retesting generally occurs after at least one round of functional testing has taken place and a majority of the defects have been found and logged already.