[Client Name] Test Plan

[date]

Table of Contents

1. Deliverable Requirements 3

Deliverable Description 3

Deliverable Due Date 3

2. Definition of Testing Terms 3

3. Cost of Testing Suite 4

4. Details for each Type of Testing 4

Coding Standards 4

Developer Testing 4

Peer Review 5

Integration Testing 5

User Acceptance Testing 5

Guidelines for Reporting UAT Issues 5

Performance and Load Testing 5

Appendix A: Testing Tools and Technology 7

Appendix B: Supported Operating Systems and Browsers 9

# 1. Deliverable Requirements

## Deliverable Description

The Test Plan describes the plan for developer, integration, and user acceptance testing and how testing will be managed as part of the development and implementation processes.

## Deliverable Due Date

The Test Plan is due …

# 2. Definition of Testing Terms

| **Term** | **Definition** |
| --- | --- |
| Acceptance testing | The testing conducted by **[the client]** to confirm that the tested functionality is acceptable for release to the public-facing, production environment. |
| Automated test | A test that has been codified so that it can be conducted by a system rather than a person. |
| Branch | A snapshot-version of the Drupal code base, which includes php, javascript, and css files. The codebase does not include html database content, image files, or PDF documents. Branches are managed with robust version control using Git. |
| Developer testing | Manual or automated tests conducted by developers on their branch of the codebase, in local or development environment. |
| Drupal coding standard | The set of standards identified by the Drupal community and documented here: https://drupal.org/coding-standards. |
| Environment | The logical space where a Drupal instance exists.  An environment has its own database, files, codebase, and URLs.  In addition to team members' local laptop environments, the project will use the following Acquia-provided environments: **[environments]** |
| Feature Epic | A collection of requirements (i.e. user stories) that describe website functionality. |
| Integration testing | Set of automated tests run at regular intervals on the current integration branch of the codebase, on the integration environment. This testing also serves the purpose of regression testing. |
| Regression testing | A type of testing that focuses on elements of a system that have already passed testing once, but are being tested again because they may have been impacted by changes elsewhere in the system. |
| Release | An immutable snapshot of the codebase that will be deployed into production. |

# 3. Cost of Testing Suite

Many of the tools used in the proposed testing suite are free and open source, but some require paid subscriptions.

The tools requiring additional payments are listed below:

* Cloudbees - This is a subscription service with a monthly fee plus charges for each minute of usage.  **[Estimated costs are $200 per month],** however this figure will fluctuate with usage.
* GitHub - This is a subscription service with a monthly fee.  **[Estimated costs are $27.50 per month].**
* JIRA - This is a subscription service with a monthly fee.  **[Estimated costs are $165 per month].**
* Sauce Labs - This is a subscription service with a monthly fee.  **[Estimated costs are $165 per month].**
* Blazemeter – This is a service that can be as a subscription or per number of tests run. **[Estimated costs to are $1,650 per round]** of performance and load testing.

A full description of testing tools and technologies is provided in Appendix A.

# 4. Details for each Type of Testing

## Coding Standards

All code will be automatically evaluated against Drupal Coding Standards, documented at https://drupal.org/coding-standards, using the PHP Code Sniffer tool.  The PHP Lint tool is also run to check custom modules' PHP syntax for parse errors.  By using Git and an installer created for this project, the team will ensure that PHP Code Sniffer and PHP Lint execute automatically upon each developer's local code commit.  It will also execute as part of integration testing.

## Developer Testing

Before integrating their custom code with the shared 'integration' codebase branch, developers will carry out appropriate testing to ensure their code is working as expected.  This may involve manual or automated testing as.  Automated tests will utilize the Behat testing framework, referenced in Appendix A.

At a minimum, one automated test will be written for each JIRA user story that creates new site functionality.  At a minimum, one manual test will be conducted for each JIRA epic.

## Peer Review

Before any code is merged into the integration branch, a peer review for best practices will occur (using the GitHub pull request workflow).  The first step of peer review is to check that the requirements and testing instructions stated in the JIRA issue have indeed been completed.  The second step is to review the code itself.  The GitHub pull request workflow allows the developers to quickly identify code changes that have been made and discuss these changes.

## Integration Testing

There will be an integration environment and integration codebase branch that is dedicated to integration testing.   Once code is merged into this branch, the full suite of automated tests will run in this environment.  This continuous integration is a best practice that allows defects, regressions, and malformed code to be discovered as early as possible in the development process.  The outcome of integration testing will be visible in a Jenkins dashboard that provides numerous code quality metrics, including results from automated testing.

The integration environment will also be used by the Acquia team for manual functional testing and content review.  **[the client]** will not need to interact with the integration environment.

## User Acceptance Testing

Once the functionality and content migration for a given release has passed integration testing, the release's code and content will be placed in the staging environment user acceptance testing (UAT).  UAT will be conducted by **[the client].**In most cases, the scope of UAT will include site structure, functionality, and content migration.

### Guidelines for Reporting UAT Issues

All UAT Issues should include the following information:

* Summary
* Description
* Steps to reproduce
* Expected results
* Actual results
* Browser used for testing (supported browsers are listed in Appendix B)

JIRA will be configured so that submitting this information is easy and intuitive.

## Performance and Load Testing

Acquia will conduct performance and load testing against the website in the Production environment before the first Drupal content is made available to the public.  Additional rounds of performance and load testing can be scheduled to run on the production environment at the project team’s discretion. Performance and load testing are meant to simulate how the site will perform while handling web traffic.  For this project, performance testing will measure page load times while the site is under an expected amount of simulated traffic.  Load testing will measure what amount of simulated traffic the site can handle without generating significant errors.  The scripts used to simulate traffic will be written using open source software called Apache Jmeter.  The platform used to run the scripts against the website is a third-party subscription service called Blazemeter.  The results of performance and load testing displayed in the Blazemeter dashboard will be used to determine if changes are necessary before the Drupal site is launched.

Another tool used to monitor and analyze site performance is called New Relic.  This is a third-party subscription service, but there will be no additional cost for this tool because it is included with the Acquia subscription.

# Appendix A: Testing Tools and Technology

This Appendix provides a description of testing tools and technologies used in the project.

| **Tool or Technology** | **Definition** | **Open Source** | **SAAS - required** |
| --- | --- | --- | --- |
| Acquia Insight | An Acquia tool used to automatically measure a Drupal website against Acquia best-practices, and recommend resolution to any identified issues (see <https://www.acquia.com/products-services/acquia-network/cloud-services/insight>). |  |  |
| Apache Ant | An open source tool used by software develoment teams to automate code builds (see <http://ant.apache.org/>). | X |  |
| Apache Jmeter | An open source technology that can be used as a load testing tool for analyzing and measuring the performance of a variety of web applications (see <https://jmeter.apache.org/>). | X |  |
| Behat | A testing tool that allows automated tests to be run from plain-language descriptions of website functionality (see http://behat.org/). | X |  |
| Blazemeter | A third-party, self-service load testing platform as a services (PaaS) which is fully compatible with Apache Jmeter (see <http://blazemeter.com/>). |  | X |
| Cloud Bees | A third-party tool used to host and manage instances of jenkins (see http://www.cloudbees.com/jenkins). | X | X |
| Gherkin | A syntax used to write test cases.  This is the required syntax for automated tests created with Behat (see <http://docs.behat.org/guides/1.gherkin.html>). | X |  |
| Git | An open source tool for code source management and revision control (see http://git-scm.com/). | X |  |
| GitHub | A third-party tool used by software development teams to share code and collaboratively build code using git (see https://github.com/). |  | X |
| GitHub Pull Request Workflow | A process using GitHub for screening code contributions from a distributed development team (see https://help.github.com/articles/using-pull-requests) |  | X |
| Jenkins | An open source tool used by software development teams to schedule and track automated code build, test, or deployment operations (see http://jenkins-ci.org/). | X |  |
| JIRA | A third-party tool used for issue tracking and project management (see https://www.atlassian.com/software/jira). |  | X |
| New Relic | A third-party service used to monitor and evaluate performance of web applications (see <http://newrelic.com/>).  A limited version of New Relic is included with the Acquia hosting subscription. |  |  |
| PHP Code Sniffer | An open source tool used to check PHP, JavaScript, and CSS files to detect violations of a defined coding standard, such as the Drupal coding standard (see http://pear.php.net/manual/en/package.php.php-codesniffer.php). | X |  |
| PHP Lint | An open source tool used to check PHP for syntax errors (see http://www.php.net/manual/en/function.runkit-lint.php). | X |  |
| Sauce Labs | A third-party tool used to access virtual instances of web browsers in order to test websites across many combinations of browsers and operating systems (see https://saucelabs.com/). |  | X |

# Appendix B: Supported Operating Systems and Browsers

While the majority of functionality will operate in all combinations of browsers and operating systems, it is not feasible to ensure all functionality operates in all combinations.  In resolving browser-specific issues, priority will be given to the combinations of operating systems and browsers indicated in the table below.

Percentages of traffic provided below were taken from Google Analytics for the period **[enter date].**

|  | **Windows 2000** | **Windows XP** | **Windows Vista** | **Windows 7** | **Windows 8 or 8.1** | **Mac OS X** |
| --- | --- | --- | --- | --- | --- | --- |
| Internet Explorer 7 | NO | NO | NO | NO | NO | NO |
| Internet Explorer 8 | NO | YES  (**7.02%)** | NO | YES  **(7.48%)** | NO | NO |
| Internet Explorer 9 | NO | NO | NO | YES  **(8.20%)** | NO | NO |
| Internet Explorer 10 | NO | NO | NO | YES  **(7.33%)** | YES  **(3.49%)** | NO |
| Internet Explorer 11 | NO | NO | NO | YES  **(2.47%)** | YES  **(0.21%)** | NO |
| Chrome | NO | NO | NO | YES  **(10.19%)** | YES  **(1.55%)** | YES  (**1.91%)** |
| Firefox | NO | NO | NO | YES  **(5.52%)** | YES  **(0.61%)** | YES  **(1.20%)** |
| Safari | NO | NO | NO | NO | NO | YES  **(5.24%)** |

This sub-set of desktop combinations represents **[62.42%]** of traffic.

In resolving mobile-specific issues, priority will be given to the combinations of devices, operating systems, and browsers indicated in the table below.

|  | **iOS – iPhone** | **iOS – iPad** | **Android – Samsung Galaxy S IV** |
| --- | --- | --- | --- |
| Safari | YES  **(8.51%)** | YES  **(5.18%)** | NO |
| Chrome | NO | No | YES  **(0.50%)** |

This sub-set of mobile combinations represents **[14.19%]** of traffic.

In total, this sub-set of desktop and mobile combinations represents **[76.61%]** all traffic.