TEST PLAN

CALIFORNIA- MARKETING PROJECT

1. INTRODUCTION

The Test Plan is designed to prescribe the scope, approach,resources, and schedule of all testing activities of the project California-Marketing website https://qasvus.wixsite.com/ca-marketing

The plan identifies the items to be tested, the features to be tested, the types of testing to be performed, the personnel responsible for testing, the resources and schedule required to complete testing, and the risks associated with the plan.

Customers want a perfect website, which has passed the full cycle of manual testing. Given the specificity of the site it is very important to have the same quality and the site.

2. SCOPE

2.1 In Scope:

All the features of website California-Marketing which were defined in software requirement specs need to be tested. The document mainly targets to check the Registration on the site, GUI of the website, website API, website Performance, and validates data in report output as per Requirements Specifications provided by Client.

Functions to be tested:

- GUI
- Registration
- Music Media Module
- API Collections
- Website Performance

2.2 Out of scope:

These features are not to be tested, they are not included in the software requirements specs.

Functions not to be tested:

- Hardware Interfaces
- Software Interfaces
- Database logical
- Communications Interfaces

3.QUALITY OBJECTIVES

The test objectives are to verify the functionality of the website, the project should focus on testing the website's GUI, Registration on website, API, Performance and guarantee all these operations can work normally in a real business environment.

3.1 Primary Objectives:

A *primary objective* of testing application systems is to: ***assure that the system meets the full requirements, including quality requirements (Functional and Non-functional requirements) and fit metrics for each quality requirement and satisfies the use case scenarios and maintain the quality of the product.*** At the end of the project development cycle, the user should find that the project has met or exceeded all of their expectations as detailed in the requirements.

Any changes, additions, or deletions to the requirements document, Functional Specification, or Design Specification will be documented and tested at the highest level of quality allowed within the remaining time of the project and within the ability of the test team.

3.2 Secondary Objective:

The *secondary objective* of testing application systems will be to: ***identify and expose all issues and associated risks, communicate all known issues to the project team, and ensure that all issues are addressed in an appropriate manner before release.*** As an objective, this requires careful and methodical testing of the application to first ensure all areas of the system are scrutinized and, consequently, all issues (bugs) found are dealt with appropriately.

4.ROLES AND RESPONSIBILITIES

No.	Member	Task
1.	Test Manager	Manage the whole project Define project directions Acquire relevant resources
2.	Testers	Identifying and describing appropriate test techniques/tools/automation. Verify and assess the Test Approach. Execute the tests, Log results, Report the defects.
3.	Developer	Implement the test cases, test program, test suite etc.
4.	Test Administrator	Builds up and ensures test environments and assets are managed and maintained Support Tester to use the test environment for test execution
5.	QA members	Take in charge of quality assurance Check to confirm whether the testing process is meeting specified requirements.

5. TEST APPROACHES

In the project California-Marketing there are four types of testing that should be conducted:

- Manual Tests
- Automation Tests
- API Tests
- Performance Tests

The project is using an ***Agile approach***, with weekly iterations every 14 days. At the end of the second week the requirements identified for that iteration will be delivered to the team and will be tested. Team also must use experience-based testing and error guessing to utilize testers's skills and intuition along with their experience with similar applications.

5.1 Overview

<u>Manual Tests:</u>

The California-Marketing **Module "Sign In"**, **Module "Shop"**, will be tested.
Check the ability of functionality in the modules for California-Marketing website with:

Valid datas - Positive testing
Invalid datas - Negative testing

Automation tests with Selenium IDE will be also created in the **"Sign In"** and **"Shop"** parts.
Manual API tests will be performed in the **"My address"**part.

<u>The UI/UX Smoke testing</u>
will be performed for California-Marketing website for verification UI response to Design requirements.

<u>Sub Plan for UI/UX testing:</u>

1. Verify that all images links on Homepage are working correctly according to Business requirements.
2. Verify that Logo leads at the website's Homepage after clicking on it.
3. Verify that the "Search" field is working correctly according to Business requirements.
   Positive /Negative/Ad hoc testing should be used also if necessary.

Environments for this part:

1. **OS**:
   - Windows 10/64
   - Android 11
2. **Devises**:
   - Desktop-FNM TECH
   - Samsung Galaxy Tab A7 light

3. **Browsers**:
   - Google Chrome
   - FireFox

<u>Website Automation Tests:</u>

Tests indicated in this part will be automated:

1. Test for module “Login”and “Shop” (to verify the ability of users to login the site and be able to choose the product).
2. Tests for UI/UX Smoke testing (to verify the response of UI/UX to design requirements).

Tools for this part:

- Selenium IDE
- XPath (ChroPath/TruPth)

<u>Website API Tests:</u>
California-Marketing API Collections will be tested through the Server response.
Tools for this part:

- Postman
- Json Formatter &Validator

<u>Website Performance Tests:</u>
Tests for evaluation of the website’s *speed, responsiveness and stability* under a workload will be done in this part.
Check *Performance, Accessibility, Best practice, SEO.*
Tools for this part:

- Lighthouse
- GT Metrix
- SpeedLab

6. TEST TYPES

There are 9 types of testing that should be conducted:

1. Exploratory testing
2. Smoke testing
3. GUI testing
4. Functional testing
5. Positive testing
6. Negative testing
7. ADHOC testing
8. API testing
9. Performance testing

<u>Exploratory testing:</u>

Exploratory testing will include a type of software testing where Test Cases are not created in advance but QA check system on the fly. QA may note down ideas about what to test before test execution.

Smoke testing:
Smoke testing is a software testing process that determines whether the deployed software build is stable. Smoke testing is a confirmation for the QA team to proceed with further software testing. It consists of a minimal set of tests run on each build to software functionalities. Smoke testing is also known as “Build Verification Testing” or “Confidence Testing”.

GUI testing:
GUI testing includes the type of testing by providing the *valid data* as input. It checks the application behavior as expected with positive inputs.

Negative testing:
Negative testing includes a method that ensures the application can handle the unwanted input and unexpected user behavior. Invalid data is inserted to compare the output against the given input. Negative testing is also known as *failure testing* or *error path testing* where *error messages* are expected.

ADHOC testing:
ADHOC testing includes an informal testing type with an aim to “break” the system.

7. TEST STRATEGY

7.1 QA ROLE IN THE TEST PROCESS:

- Understanding *Requirements*.
- Requirement *Specifications* will be sent by client.
- Understanding of Requirements will be done by QA.
- Preparing Test Cases:

QA will be preparing test cases based on the exploratory testing. This will cover all scenarios for requirements.

- Preparing Test Matrix:

QA will prepare RTM map tests, this will ensure the coverage for requirements.

- Reviewing test cases and matrix
- Peer review will be conducted for test cases and test matrix by QA Lead.
- Suggestions or improvements will be re done by the author and sent for approval.
- *Creating Test Data*. Test data will be created by QA Engineer on client’s site/developments based on scenarios and Test cases.
- *Executing Test Cases.* Test cases will be executed by QA Engineer on client’s site/developments based on scenarios and Test cases.
- Test result (Pass/Fail) will be updated in the test case document: QA will be logging the defect/bug in Jira.
- QA informs the respective developer about the defects.
- QA performs Retesting and Regression testing:

Retesting for fixed bugs will be done by respective QA once it is resolved by the respective Developer and bug status will be updated accordingly. Regression testing will be conducted if required.

- Deployment/Delivery:

Once all defects /bugs reported after complete testing are fixed and no other bugs are found, the report will be deployed to the client's site.
Once a round of testing will be done by respective QA on the client's test site if required, the report will be delivered along with the sample output by email to the respective Lead and Report group.

7.2 Bug Triage

ALL defects should be logged using Jira', to address and debug defects. Adopters are also requested to send a daily defect report to the developer. Developers will update the defect list on Jira and notify the requestor after the defect has been resolved.
Developers and Adopters are required to request an account on Jira for the relative workspace. Debugging should be based on Priority – High &gt; Medium &gt; Low, these priorities are set by the Adopters and are based on how critical the test script is in terms of dependency and mainly based on use case scenarios.
Below screenshot displays 'Add new Defect' screen, fields marked with (*) are mandatory fields and Adopters should also upload the evidence file for all the defects listed.

All High priority defects should be addressed ***within 1 day*** of the request and resolved/closed
within 2 days of the initial request
All Medium priority defects should be addressed ***within 2 days*** of the request and resolved/closed within 4 days of the initial request
All Low priority defects should be resolved/closed ***no later than 5 days*** of the initial request.

7.2.2 Bug Severity List

Severity ID	Severity Level	Severity Description
1	**Highest**	The module/product *crashes* or the bug causes non-recoverable conditions. System crashes, GP Faults, or database or file corruption, or potential data loss, program hangs requiring reboot are all examples of a **Sev. 1** bug**.**
2	**High**	Major system *components unusable due to failure* or incorrect functionality. **Sev. 2** bugs cause serious problems such as a lack of functionality, or insufficient or unclear error messages that can have a major impact on the user, prevent other areas of the app from being tested, etc. Sev. 2 bugs can have a work around, but the work around is inconvenient or difficult.
3	**Medium**	*Incorrect functionality* of component or process. There is a simple work around for the bug if it is **Sev. 3.**
4	**Low**	Documentation errors or signed off **Sev 4** bugs. Low severity bug occurs when there is almost no impact on the functionality but it is still a valid defect that should be corrected.

7.2.3 Bug Priority List

Priority ID	Priority Level	Priority Description
1	**Highest**	This bug must be fixed *immediately or within 1 day*; the product cannot ship with this bug.
2	**High**	These are important problems that should be fixed *as soon as possible.* It would be an embarrassment to the company if this bug shipped.
3	Medium	The problem should be fixed within the time available. If the bug does not delay the shipping date, then fix it.
4	Low	It is not important (at this time) that these bugs be addressed. Fix these bugs after all other bugs have been fixed.
5	Trivial	Enhancements/ Good to have features incorporated- just are out of the current scope.

7.2.4 Document Change History

Version Number	Date	Contributor	Description
V1.0	Date of change	Name of person made changes	Description of the changes were made

8. RESOURCE AND ENVIRONMENTAL NEEDS

8.1 Testing Tools

Process	Tool
Test case creation	Microsoft Word, Microsoft Excel, Jira
Test case tracking	Jira, Confluence
Test case execution	Manual, Selenium IDE
Test case management	Microsoft Excel, Jira
Defect case management	Microsoft Word, Jira, Confluence
Test reporting	Jira
API testing	Postman
Performance testing	Lighthouse, GT Metrix, Speed Lab
Automation testing	Selenium IDE, XPath

8.2 Test Environment/ Support browsers:

Windows 10/64: Chrome, Firefox
Android

9. TEST SCHEDULE

Task name	Start	Finish	Effort	Comments

10. APPROVALS

	Project Manager	QA Leader
Name		
Signature		

11. TERMS/ACRONYMS

The terms are used as examples, please add/remove any terms relevant to the document

TERM/ACRONYM	DEFINITION
API	Application Program Interface
GUI	Graphical user interface
PM	Project Manager
UAT	User acceptance testing

CM	Configuration Management
QA	Quality Assurance
RTM	Requirements Traceability Matrix