Test Plan

1. Introduction

Customer wants a perfect website that passes the full cycle of manual and automotive testing. Given the specificity of the site it is important to have the same quality of the site as per requirements.
This document describes approaches and methodologies that will apply to the functional testing of the website https://www.wikipedia.org

2. SCOPE

The document mainly targets the GUI testing and validating data in report output as per Requirements Specifications provided by Client.

2.1 Functions to be tested:

- GUI
- Search Logic
- Announcement placement functionality
- Performance
- Security

3. QUALITY OBJECTIVES

3.1 Primary Objectives A primary objective of testing 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 Objectives The secondary objectives of testing 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 matter 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. TEST APPROACH The approach that is used is Analytical therefore, in accordance with requirements-based strategy, where an analysis of the requirements specification forms the basis for planning,estimating and designing tests. Test cases will be created during exploratory testing. All test types are determined in Test Strategy. Team also must use experience-based testing and error guessing to utilize tester's skills and intuition, along with their experience with similar applications or technologies. The project is using an agile approach, with weekly iterations. At the end of each week the requirements identified for that iteration will be delivered to the team and will be tested.

5. ROLES AND RESPONSIBILITIES Role Staff Member Responsibilities
Project Manager: Sergey Efremov
1. Acts as a primary contact for development and QA team.
2. Responsible for Project schedule and the overall success of the project.
QA Lead: Sergey Efremov
1. Participation in the project plan creation/update process.
2. Planning and organization of test process for the release.
3. Coordinating with QA analysts/engineers on any issues/problems encountered during testing.
4. Report progress on work assignments to the PM.
QA: Anna Sour
1. Understand requirements.
2. Writing and executing Test cases.
3. Preparing RTM.
4. Reviewing Test cases, RTM.
5. Defect reporting and tracking.
6. Retesting and regression testing.
7. Bug Review meeting.
8. Preparation of Test Data.

9. Coordinating with QA Lead for any issues or problems encountered during test preparation/execution/defect handling.

6. ENTRY AND EXIT CRITERIA

6.1 Entry Criteria

- All test hardware platforms must have been successfully installed, configured, and functioning properly.
- All the necessary documentation, design, and requirements information should be available that will allow testers to operate the system and judge the correct behavior.
- All the standard software tools including the testing tools must have been successfully installed and functioning properly.
- Proper test data is available.
- The test environment such as, lab, hardware, software, and system administration support should be ready.
- QA resources have completely understood the requirements.
- QA resources have sound knowledge of functionality.
- Reviewed test scenarios, test cases and RTM.

6.2 Exit Criteria

- A certain level of requirements coverage has been achieved.
- No high priority or severe bugs are left outstanding.
- All high-risk areas have been fully tested, with only minor residual risks left outstanding.
- Cost – when the budget has been spent.
- The schedule has been achieved.

7. SUSPENSION CRITERIAAND RESUMPTION REQUIREMENTS

7.1 Suspension criteria

- The build contains many serious defects which seriously or limit testing progress.
- Significant change in requirements suggested by client.
- Software/Hardware problems.
- Assigned resources are not available when needed by a test team.

7.2 Resumption criteria Resumption will only occur when the problem(s) that caused the suspension have been resolved.

8. TEST STRATEGY

8.1 QA role in test process:

● Understanding Requirements.

● Requirement specifications will be sent by client.

● Understanding of requirements will be done by QA. - Preparing Test Cases. Test strategy includes 5 sub-tasks:

1. Manual testing. QA will be preparing test cases based on the exploratory testing. There will be 5 test cases.
2. Automation testing. QA will transfer manual cases into automation cases with Selenium Webdriver on Python.
3. API testing. QA will be preparing test cases.
There will be 6 test cases by method CRUD. 1 of them will be positive, 5 - negative. Test cases will be automated with Postman API.
4. Website Performance Automation testing will be carried out with Lighthouse and GTMetrix tools
5. Website Automation Security test will be carried out with Mozilla Observatory tool. - Preparing Test Matrix: QA will be preparing a test matrix which maps test cases to respective requirements. 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.

● Any comments or suggestions on test cases and test coverage will be provided by the reviewer respective Author of Test Case and Test Matrix.

● Suggestions or improvements will be re-worked by the author and will be sent for approval.

● Re-worked improvements will be reviewed and approved by the reviewer.

- Creating Test Data: Test data will be created by respective QA on client's developments/test site based on scenarios and Test cases. - Executing Test Cases:

● Test cases will be executed by respective QA on a client's development/test site based on designed scenarios, test cases and Test data.

● Test result (Actual Result, Pass/Fail) will be updated in test case document Defect Logging and Reporting: QA will be logging the defect/bugs in Word document and JIRA, found during execution of test cases. After this, QA will inform a respective developer about the defect/bugs. - Retesting and Regression Testing: Retesting for fixed bugs will be done by respective QA once it is resolved by the respective developer and bug/defect status will be updated accordingly. In certain cases, regression testing will be done if required. - Deployment/Delivery:

● Once all bugs/defects reported after complete testing are fixed and no other bugs are found, the report will be deployed to the client's test site.

● Once a round of testing will be done by QA on the client's test site if required Report will be delivered along with sample output by email to the respective lead and Report group.

● QA will be submitting the filled hard copy of the delivery slip to the respective developer.

● Once the lead gets the hard copy of the delivery slip filled by QA and developer, he will send the report delivery email to the client.

8.2 Bug life cycle: All the issues found while testing will be logged into JIRA

8.3 Testing types GUI Testing: GUI testing will include testing the UI part of the report. It covers users Report format, look and feel, error messages, spelling mistakes, GUI guideline violations. Exploratory testing : 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. Positive testing: Positive testing will include the type of testing that can be performed on the system by providing the valid data as input. It checks whether an application behaves as expected with positive inputs. Negative testing: Negative testing will include a method of testing an application or system that ensures that the plot of the application is according to the requirements and can handle the unwanted input and 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. When performing negative testing exceptions are expected.

Functional Testing: Functional testing is carried out in order to find out unexpected behavior of the report. The characteristics of functional testing are to provide correctness, reliability, testability and accuracy of the report output/data.

Performance Testing:

- Check the optimal time the page is loaded
- Check the operation of the system under load

8.4 Bug Severity and Priority Definition Bug Severity and Priority fields are both very important for categorizing bugs and prioritizing if and when the bugs will be fixed. The bug Severity and Priority levels will be defined as outlined in the following tables below. Testing will assign a severity level to all bugs. The Test Lead will be responsible to see that a correct severity level is assigned to each bug. The QA Lead, Development Lead and Project Manager will participate in bug review meetings to assign the priority of all currently active bugs. This meeting will be known as “Bug Triage Meetings”. The QA Lead is responsible for setting up these meetings on a routine basis to address the current set of new and existing but unresolved bugs.
Severity List
(Severity ID Severity Severity Description)
1 Highest
The module/product crashes or the bug causes nonrecoverable conditions. System crashes, GP Faults, or database or file corruption, or potential data loss, program hangs requiring reboot are all examples of a Severity 1 bug.
2 High
Major system component unusable due to failure or incorrect functionality. Severity 2 bugs cause serious problems such as a lack of functionality, or insufficient or unclear error messages that can have a major impact to the user, prevents other areas of the app from being tested, etc. Severity 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 Severity 3.
4 Low
Documentation errors or signed off Severity 3 bugs.

Priority List

Priority Level 1 Highest
This bug must be fixed immediately; the product cannot ship with this bug.

Priority Level 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.
Priority Level 3 Medium
The problem should be fixed within the time available. If the bug does not delay the shipping date, then fix it.
Priority Level 4 Low
It is not important (at this time) that these bugs be addressed. Fix these bugs after all other bugs have been fixed. Enhancements/ Good to have features incorporated just are out of the current scope.
Priority Level 5
Lowest Documentation errors or signed off Low 4 bugs

9. RESOURCE AND ENVIRONMENT NEEDS

9.1 Testing Tools

Process: Test case creation
Tool: Google Sheet, JIRA
Process:Test case tracking
Tool: JIRA, Confluence
Test Process:case execution
Tool: Frontend Manual, Frontend Selenium Automation with Python, API automation with Postman
Process:Test case management
Tool : Google Sheet, JIRA, Confluence

Process: Defect management
Tool : Google Sheet, JIRA, Confluence
Process:Test reporting
Tool: JIRA
Process: Check list creating
Tool: Google Sheet, JIRA

9.2 Configuration Management

- Documents CM: SVN
- Code CM: Git 9.3 Test Environment x Support level 1 (browsers):
- MacOS Monterey, version 12.6: Edge (Version 95.0.1020.44), Chrome (Version 95.0.4638.69), Firefox (Version 94.0.1)

10.TEST SCHEDULE

Task Name: Test Planning
Time for task:16 hours/2 days
Comments: documentation

Task Name: Review Requirements documents
Time for task: 8 hours/1 day

Task Name: Create test basis
Time for task:8 hours/1 day flex
Effort: with team
Comments: flex

FRONT-END
Task Name: Exploratory testing
Time for task:4 hours/0,5

Task Name: Positive testing
Time for task:4 hours/0,5 day

Task Name:Automating of the manual test-cases

Time for task:16 hours/2 days

API TESTING

Task Name: Exploratory testing
Time for task: 4 hours/0,5 day
Task Name: Positive testing
Time for task: 4 hours/0,5 day
Task Name: Negative testing
Time for task: 4 hours/0,5 day
Task Name: Automating of the manual test-cases
Time for task: 16 hours/2 days
Task Name: Performance testing
4 hours/0,5 day

11.APPROVALS:
Project Manager/QA Lead: Sergey Efremov

Signature___________________________

12.TERMS/ACRONYMS The below 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