Project “ADOBE” Test Plan (website)

Author: Katsiaryna Bialiayeva

INTRODUCTION: This document describes the approaches and methodologies that will be applied to site testing. The quality of the site must meet the requirements of the client. Project specification - going through the full cycle of manual testing, API testing, and Performance Automation testing using additional tools.

Requirements Specifications provided by Client

OUT OF SCOPE:

- Performance and Load Testing
- Security and Access Control Testing

IN SCOPE:

Website Manual Test:

- GUI Testing
- Positive Testing*
- Negative Testing*
- Boundary Testing*
- Equivalence Partitioning Testing*
- Validating data in report

Website Automation Test:

Website Performance Automation test with:

- Lighthouse
- GTMetrix
- SpeedLab tools

Website API Test:

Create API queries with Postman. Send requests to an endpoint, retrieve data from a data source, or test API functionality.
Using 4 CRUD methods - create, read, update and delete to interact with REST API resources.

ROLES AND RESPONSIBILITIES:

Project Manager:

- Acts as a primary contact for development and QA team.
- Responsible for Project schedule and the overall success of the project.

QA Lead:

- Participation in the project plan creation/update process.
- Planning and organization of the test process for the release.
- Coordinate with QA analysts/engineers on any issues/problems encountered during testing.
- Report progress on work assignments to the PM

QA:

- Understand requirements
- Writing and executing Test cases
- Preparing RTM
- Reviewing Test cases, RTM
- Defect reporting and tracking
- Retesting and regression testing
- Bug Review meeting
- Preparation of Test Data
- Coordinate with QA Lead for any issues or problems encountered during test preparation/execution/defect handling.

ENVIRONMENTS + TOOLS:

1. Testing Environments:
   - Web Browsers
     - ➔ Google Chrome
     - ➔ Mozilla Firefox
     - ➔ Safari
     - ➔ Opera
   - Operating systems
     - ➔ Windows
     - ➔ MacOS
     - ➔ Linux
   - Mobile Devices (operating systems)
     - ➔ Android
     - ➔ IOS
2. Test Management and Bug Tracking Tool
   - JIRA
   - Confluence
3. Other Tools:
   - Microsoft Word
   - Microsoft Excel
   - Lightshot Screenshot
   - Awesome Screenshot
   - ScreenPal

RESOURCE:

- Test Plan
- Test Scenarios
- Test Cases
- Test Data
- Test Environments
- Documentation

RISKS:

- Incomplete Requirements
- Tight Timelines
- Lack of Test Data
- Communication and Collaboration Challenges
- Regression Issues
- Resource Constraints
- Security and Privacy Concerns

APPROACH:	TIMESCALES:
• Requirements Analysis • Test Planning • Test case creation • Test Data Preparation • Test Execution • Bug Reporting and Tracking • Regression Testing • Collaboration and Communication • Test Coverage Review • Documentation and Reporting • Continuous Improvement	By 15 July, 2023

---

* Positive testing – includes 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 – also known as failure testing or error path testing, is 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.
Bug Severity and Priority Definition.

*Boundary Testing - this is one of the software testing methods and it includes its own data counters based on their boundary dimensions or two dimensions such as minimum and maximum values.

·Equivalence Partitioning Testing - is a software testing technique that allows you to reduce the number of test cases without reducing test coverage. The essence of this technique is to divide all possible input data into classes that are equivalent to each other, and select at least one representative from each class for testing. This allows us to make sure that the program works correctly for all values from each equivalence class, and also to cover all possible inputs. Equivalence classes can be defined based on program requirements or program specifications.