SOFTWARE QUALITY ASSURANCE (SQA)

PREPARED FOR

Coralie Mercier,

Head of W3C Marketing & Communications

Prime Consultant

Dileep Kumar
Business Development Head
Chance River

Stamford,

http://ww.chanceriver.com
E-mail: dileep.kumar@chanceriver.com

Date Prepared:
December 13 2019

Our Mission

Chanceriver will test early, validate along the way, collaborate with business partners & client to deliver a quality Website Application.

Introduction

Software testing is an essential part of software development cycle. It is considered an important activity where software is validated in compliance to requirements and specifications. This document will address the different standards that will apply to the unit, integration and system testing of the specified application. The design, development and testing of these reports will be based on clients projects. Throughout the testing process we will be applying the test documentation specifications described in the IEEE Standard for Software Test Documentation to ensure systematic and complete test coverage and ensure full visibility of the project's activities. Apply proven testing techniques, tools and methodologies to perform different types of testing and surpass your quality expectations

Objective

Objective of Test plan is to define the various Testing strategies and testing tools used for complete Testing life cycle of this project.

Scope

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

Chanceriver & Testing

Chanceriver is a provider of custom software development, testing and QA services. Driven by 10 years of experience in software testing and QA services, our company can help you to meet high end-user demands and provide fully functional, stable, secure, and user-friendly software. Chanceriver's test engineers and QA professionals are ready to take up a testing project of any scale and complexity, introduce an efficient and mature QA process, or enhance your general strategy of software quality management.

What We Offer

QA outsourcing

We provide end-to-end testing services at each stage of the software development lifecycle to ensure all the pieces integrated into your solution are performing as designed from start to finish. Project-specific testing services

Independent testing services

Depending on the specifics of your project, we can run particular types of testing, be it functional, performance, security or any other software testing type.

QA consulting

An increased workload, inconsistent QA process or methodologies, business expansion may lead to inefficient quality management. We help you detect and address the existing problems through QA strategy development, QA process/project audit, QA aid and pre-certification.

Quality assessment

Chaneriver offers comprehensive product audit from different angles, including code quality, solution functionality, usability, performance and security. As a result, you get a detailed report with the list of the detected issues and recommendations on how to fix them.

Why Quality Assurance

Why is Software Testing Important?

Testing is important because software bugs could be expensive or even dangerous. Software bugs can potentially cause monetary and human loss, and history is full of such examples. An application failure at a crucial point in a transaction can be complex to repair and cost millions in lost sales and customer goodwill. Testing is an absolutely critical element in ensuring that software and applications perform as specified, are rolled-out quickly, and don't threaten business continuity during deployment.

Why test web applications?

- Application/network security

- Poor performance
- Availability
- Scalability
- Multi-browser compatibility
- Disaster recovery
- Data integrity
- Privacy and data security
- Compliance with standards
- Live upgrades

Types of Software Testing

Typically Testing is classified into three categories.

1. Functional Testing
2. Non-Functional Testing or Performance Testing
3. Maintenance (Regression and Maintenance)

Chanceriver Testing Checklist

Usability Testing

What is Usability Testing?

- Usability testing is nothing but the User-friendliness check.
- In Usability testing, the application flow is tested so that a new user can understand the application easily.
- Basically, system navigation is checked in Usability testing.

Functional Testing

What is Functional Testing?

- Testing the features and operational behavior of a product to ensure they correspond to its specifications.
- Testing that ignores the internal mechanism of a system or component and focuses solely on the outputs generated in response to selected inputs and execution conditions.

What is the purpose or Goal of Functional testing?

- The goal of Functional Testing is to verify whether your product meets the intended functional specifications mentioned in your development documentation.

Compatibility Testing

What is Compatibility testing?

- Compatibility testing is used to determine if your software is compatible with other elements of a system with which it should operate, e.g. Browsers, Operating Systems, or hardware.

What is the purpose or Goal of Compatibility testing?

- The purpose of Compatibility testing is to evaluate how well the software performs in a particular browser, Operating Systems, hardware or software.

Database Testing:

What is Database Testing?

- In Database testing backend records are tested which have been inserted through the web or desktop applications. The data which is displayed in the web application should match with the data stored in the Database.

Security Testing

What is Security Testing?

Security Testing involves the test to identify any flaws and gaps from a security point of view.

Performance Testing

What is Performance Testing?

Performance Testing is conducted to evaluate the compliance of a system or component with specified performance requirements.

Performance Testing tool used

- Apache JMeter
- Load Runner
- Borland Silk Performer.
- Rational Performance Tester
- WAPT
- NEO LOAD

Mobile Testing

We offer ample experience in testing iOS, Android, Windows Phone and cross-platform mobile applications in various domains. We also use external tools for Mobile testing and share the detailed report at the end of testing completion.

1. UI testing
2. UX testing
3. Mobile performance testing
4. Mobile security testing
5. Mobile functional (manual) testing
6. Mobile test automation

Mobile Automation Testing Services

Mobile automation testing is a process that uses automated scripts run without human involvement to test native, mobile-web, and hybrid applications. This type of testing executes automated tests on the devices (including real devices & emulators) and verifies that the application works as expected and matches all requirements. Test execution can be performed on different devices, platform versions, screen resolutions, network connections etc.

- Automated Mobile App Testing will
- Speed up your testing process
- Reduce human involvement
- Have a structured approach
- Enable cross-platform testing
- Track the testing results and app status
- Run tests in the cloud

Chanceriver Test Cases

General Test Scenarios

1. All mandatory fields should be validated and indicated by an asterisk (*) symbol.
2. Validation error messages should be displayed properly at a correct position.
3. All error messages should be displayed in the same CSS style (e.g. using red color)
4. General confirmation messages should be displayed using CSS style other than error messages style (e.g. using green color)
5. Tooltips text should be meaningful.
6. Drop-down fields should have the first entry as blank or text like ‘Select'.
7. ‘Delete functionality' for any record on a page should ask for a confirmation.
8. Select/deselect all records option should be provided if page supports record add/delete/update functionality
9. Amount values should be displayed with correct currency symbols.
10. Default page sorting should be provided.
11. Reset button functionality should set default values for all fields.
12. All numeric values should be formatted properly.
13. Input fields should be checked for the max field value. Input values greater than specified max limit should not be accepted or stored in the database.
14. Check all input fields for special characters.
15. Field labels should be standard e.g. field accepting user's first name should be labeled properly as ‘First Name'.
16. Check page sorting functionality after add/edit/delete operations on any record.
17. Check for timeout functionality. Timeout values should be configurable. Check application behavior after operation timeout.
18. Check cookies used in an application.
19. Check if downloadable files are pointing to the correct file paths.
20. All resource keys should be configurable in the config files or database instead of hard coding.
21. Standard conventions should be followed throughout for naming resource keys.
22. Validate markup for all web pages (validate HTML and CSS for syntax errors) to make sure it is compliant with the standards.
23. Application crash or unavailable pages should be redirected to the error page.
24. Check text on all pages for spelling and grammatical errors.
25. Check numeric input fields with character input values. Proper validation message should appear.
26. Check for negative numbers if allowed for numeric fields.
27. Check amount fields with decimal number values.
28. Check functionality of buttons available on all pages.

29. The user should not be able to submit page twice by pressing submit button in quick succession.
30. Divide by zero errors should be handled for any calculations.
31. Input data with the first and last position blank should be handled correctly.

GUI and Usability Test Scenarios

1. All fields on a page (e.g. text box, radio options, drop-down lists) should be aligned properly.
2. Numeric values should be justified correctly unless specified otherwise.
3. Enough space should be provided between field labels, columns, rows, error messages etc.
4. The scrollbar should be enabled only when necessary.
5. Font size, style, and color for headline, description text, labels, infield data, and grid info should be standard as specified in SRS.
6. Description text box should be multi-lined.
7. Disabled fields should be greyed out and users should not be able to set focus on these fields.
8. Upon click of an input text field, mouse arrow pointer should get changed to the cursor.
9. The user should not be able to type in drop down select lists.
10. Information filled by users should remain intact when there is an error message on page submit. The user should be able to submit the form again by correcting the errors.
11. Check if proper field labels are used in error messages.
12. Drop-down field values should be displayed in a defined sort order.
13. Tab and Shift+Tab order should work properly.
14. Default radio options should be pre-selected on the page load.
15. Field-specific and page level help messages should be available.
16. Check if the correct fields are highlighted in case of errors.
17. Check if the drop-down list options are readable and not truncated due to field size limit.
18. All buttons on a page should be accessible by keyboard shortcuts and the user should be able to perform all operations using a keyboard.
19. Check all pages for broken images.
20. Check all pages for broken links.
21. All pages should have a title.
22. Confirmation messages should be displayed before performing any update or delete operation.
23. Hourglass should be displayed when the application is busy.
24. Page text should be left justified.
25. The user should be able to select only one radio option and any combination for checkboxes.

Test Scenarios for Filter Criteria

1. The user should be able to filter results using all parameters on the page.
2. Refine search functionality should load the search page with all user-selected search parameters.
3. When there are at least one filter criteria required to perform the search operation, make sure proper error message is displayed when the user submits the page without selecting any filter criteria.
4. When at least one filter criteria selection is not compulsory, the user should be able to submit the page and the default search criteria should get used to query results.
5. Proper validation messages should be displayed for all invalid values for filter criteria.

Test Scenarios for Result Grid

1. Page loading symbol should be displayed when it's taking more than default time to load the result page.
2. Check if all the search parameters are used to fetch data shown on the result grid.
3. The total number of results should be displayed in the result grid.
4. Search criteria used for searching should be displayed in the result grid.
5. Result grid values should be sorted by default column.
6. Sorted columns should be displayed with a sort icon.
7. Result grids should include all the specified columns with correct values.
8. Ascending and descending sorting functionality should work for columns supported by data sorting.
9. Result grids should be displayed with proper column and row spacing.
10. Pagination should be enabled when there are more results than the default result count per page.
11. Check for Next, Previous, First and Last page pagination functionality.
12. Duplicate records should not be displayed in the result grid.
13. Check if all the columns are visible and horizontal scrollbar is enabled if necessary.
14. Check the data for dynamic columns (columns whose values are calculated dynamically based on the other column values).
15. For result grids showing reports check ‘Totals' row and verify the total for every column.
16. For result grids showing reports check ‘Totals' row data when pagination is enabled and the user gets navigated to the next page.
17. Check if proper symbols are used for displaying column values e.g. % symbol should be displayed for percentage calculation.
18. Check result grid data to know if the date range is enabled.

Test Scenarios for a Window

1. Check if default window size is correct.
2. Check if child window size is correct.
3. Check if there is any field on the page with default focus (in general, the focus should be set on the first input field of the screen).
4. Check if child windows are getting closed on closing parent/opener window.
5. If the child window is opened, the user should not be able to use or update any field in the background or parent window
6. Check window minimize, maximize, and close functionality.
7. Check if the window is resizable.
8. Check scroll bar functionality for parent and child windows.
9. Check cancel button functionality for the child window.

Database Testing Test Scenarios

1. Check if correct data is getting saved in the database upon successful page submit.
2. Check values for columns which are not accepting null values.
3. Check for data integrity. Data should be stored in single or multiple tables based on the design.
4. Index names should be given as per the standards e.g. IND_<Tablename>_<ColumnName>
5. Tables should have a primary key column.
6. Table columns should have description information available (except for audit columns like created date, created by etc.)
7. For every database add/update operation log should be added.
8. Required table indexes should be created.
9. Check if data is committed to the database only when the operation is successfully completed.
10. Data should be rolled back in case of failed transactions.
11. Database name should be given as per the application type i.e. test, UAT, sandbox, live (though this is not a standard it is helpful for database maintenance)
12. Database logical names should be given according to the database name (again this is not standard but helpful for DB maintenance).
13. Stored procedures should not be named with a prefix “sp_”
14. Check if values for table audit columns (like created date, created by, updated, updated by, is deleted, deleted data, deleted by etc.) are populated properly.
15. Check if input data is not truncated while saving. Field length shown to the user on the page and in database schema should be the same.
16. Check numeric fields with minimum, maximum, and float values.
17. Check numeric fields with negative values (for both acceptance and non-acceptance).
18. Check if radio button and drop-down list options are saved correctly in the database.
19. Check if the database fields are designed with the correct data type and data length.

20. Check if all the table constraints like a Primary key, Foreign key etc. are implemented correctly.
21. Test stored procedures and triggers with sample input data.
22. Input field leading and trailing spaces should be truncated before committing data to the database.
23. Null values should not be allowed for the Primary key column.

Test Scenarios for Image Upload Functionality

(Also applicable for other file upload functionality)

1. Check for uploaded image path.
2. Check image upload and change functionality.
3. Check image upload functionality with image files of different extensions (e.g. JPEG, PNG, BMP etc.)
4. Check image upload functionality with images having space or any other allowed special character in the file name.
5. Check duplicate name image upload.
6. Check image upload with image size greater than the max allowed size. The Proper error message should be displayed.
7. Check image upload functionality with file types other than images (e.g. txt, doc, pdf, exe etc.). A proper error message should be displayed.
8. Check if images of specified height and width (if defined) are accepted otherwise rejected.
9. The image upload progress bar should appear for large size images.
10. Check if cancel button functionality is working in between upload process.
11. Check if file selection dialog shows only supported files listed.
12. Check multiple images upload functionality.
13. Check image quality after upload. Image quality should not be changed after upload.
14. Check if the user is able to use/view the uploaded images.

Test Scenarios for Sending Emails

(Test cases for composing or validating emails are not included here)
(Make sure to use dummy email addresses before executing email related tests)

1. Email template should use standard CSS for all emails.
2. Email addresses should be validated before sending emails.
3. Special characters in the email body template should be handled properly.
4. Language specific characters (e.g. Russian, Chinese or German language characters) should be handled properly in the email body template.

5. Email subject should not be blank.
6. Placeholder fields used in the email template should be replaced with actual values e.g. {Firstname} {Last Name} should be replaced with individuals first and last name properly for all the recipients.
7. If reports with dynamic values are included in the email body and report data should be calculated correctly.
8. Email sender name should not be blank.
9. Emails should be checked in different email clients like Outlook, Gmail, Hotmail, Yahoo! mail etc.
10. Check to send email functionality using TO, CC and BCC fields.
11. Check plain text emails.
12. Check HTML format emails.
13. Check email header and footer for company logo, privacy policy and other links.
14. Check emails with attachments.
15. Check to send email functionality to single, multiple or distribution list recipients.
16. Check if a reply to email address is correct.
17. Check to send the high volume of emails.

Test Scenarios for Excel Export Functionality

1. The file should get exported in the proper file extension.
2. The file name for the exported Excel file should be as per the standards e.g. if the file name is using the timestamp, it should get replaced properly with an actual timestamp at the time of exporting the file.
3. Check for date format if exported Excel file contains the date columns.
4. Check number formatting for numeric or currency values. Formatting should be the same as shown on the page.
5. The exported file should have columns with proper column names.
6. Default page sorting should be carried in the exported file as well.
7. Excel file data should be formatted properly with header and footer text, date, page numbers etc. values for all pages.
8. Check if the data displayed on a page and exported Excel file is the same.
9. Check export functionality when pagination is enabled.
10. Check if the export button is showing proper icon according to the exported file type E.g. Excel file icon for xls files
11. Check export functionality for files with a very large size.
12. Check export functionality for pages containing special characters. Check if these special characters are exported properly in the Excel file.

Performance Testing Test Scenarios

1. Check if the page load time is within the acceptable range.

2. Check the page load on slow connections.
3. Check the response time for any action under a light, normal, moderate, and heavy load conditions.
4. Check performance of database stored procedures and triggers.
5. Check the database query execution time.
6. Check for load testing of the application.
7. Check for stress testing of the application.
8. Check CPU and memory usage under peak load condition.

Security Testing Test Scenarios

1. Check for SQL injection attacks.
2. Secure pages should use the HTTPS protocol.
3. Page crash should not reveal application or server info. Error page should be displayed for this.
4. Escape special characters in the input.
5. Error messages should not reveal any sensitive information.
6. All credentials should be transferred over an encrypted channel.
7. Test password security and password policy enforcement.
8. Check application logout functionality.
9. Check for Brute Force Attacks.
10. Cookie information should be stored in encrypted format only.
11. Check session cookie duration and session termination after timeout or logout.
12. Session tokens should be transmitted over a secured channel.
13. The password should not be stored in cookies.
14. Test for Denial of Service attacks.
15. Test for memory leakage.
16. Test unauthorized application access by manipulating variable values in the browser address bar.
17. Test file extension handing so that exe files are not uploaded and executed on the server.
18. Sensitive fields like passwords and credit card information should not have to autocomplete enabled.
19. File upload functionality should use file type restrictions and also anti-virus for scanning uploaded files.
20. Check if directory listing is prohibited.
21. Password and other sensitive fields should be masked while typing.
22. Check if forgot password functionality is secured with features like temporary password expiry after specified hours and security question is asked before changing or requesting a new password.
23. Verify CAPTCHA functionality.
24. Check if important events are logged in log files.

25. Check if access privileges are implemented correctly.

Chanceriver Best Practice for writing Test Cases.

Test Cases need to be simple and transparent:

Create test cases that are as simple as possible. They must be clear and concise as the author of the test case may not execute them. Use assertive language like go to the home page, enter data, click on this and so on. This makes the understanding the test steps easy and tests execution faster.

Create Test Case with End User in Mind

The ultimate goal of any software project is to create test cases that meet customer requirements and is easy to use and operate. A tester must create test cases keeping in mind the end user perspective

Avoid test case repetition.

Do not repeat test cases. If a test case is needed for executing some other test case, call the test case by its test case id in the precondition column

Do not Assume

Do not assume functionality and features of your software application while preparing test case. Stick to the Specification Documents.

Ensure 100% Coverage

Make sure you write test cases to check all software requirements mentioned in the specification document. Use Traceability Matrix to ensure no functions/conditions is left untested.

Test Cases must be identifiable.

Name the test case id such that they are identified easily while tracking defects or identifying a software requirement at a later stage.

Implement Testing Techniques

It's not possible to check every possible condition in your software application. Software Testing techniques help you select a few test cases with the maximum possibility of finding a defect.

Self-cleaning

The test case you create must return the Test Environment to the pre-test state and should not render the test environment unusable. This is especially true for configuration testing.

Repeatable and self-standing

The test case should generate the same results every time no matter who tests it

Peer Review.

After creating test cases, get them reviewed by your colleagues. Your peers can uncover defects in your test case design, which you may easily miss.

Success Story

Customer :- A US based DJ Audio and Video streaming applications.

Solution :- Functional, exploratory, usability and regression testing of each customized version of the B2B hybrid application on 15+ devices, such as TVs, smartphones, tablets, desktop computers and laptops in 7 Customer’s projects. Automation Testing for Web, QA Audit, Manual Testing, Test Documentation design, DevOps, Responsibility for Quality. Our QA team created automated tests for iOS and Android applications from scratch. Smoke and regression tests were written and integrated to the Pipeline process on Jenkins CI.

Tools & Technologies

Atlassian JIRA, Confluence, Google Drive

Customer :- A European based electronic products manufacturing company along with its e-commerce web application

Solution :- Full-scale testing and technical support of a multi-faceted ERP system integrated with a BI module. Since the application was updated constantly, regression testing was carried out each time a new feature was implemented. Our manual QA team reported more than 100 bugs, 30% of which were critical or blocker issues that prevented releasing new versions of the application.

With our full range of ecommerce testing services, QA Mentor can ensure:

1. Minimum downtime
2. Browser compatibility
3. Portability between devices
4. Better security
5. Smooth middleware integration
6. High speeds even at peak times

Tools & Technologies

Atlassian JIRA, Confluence, Google Drive