Industry's Best Practice QA Process

2023

QUALITY ASSURANCE

"Quality is not expensive, it's priceless"
"Because quality matters"

Prepared By:
Nazrul Islam Anik

PRODUCT QUALITY ASSURANCE

SCOPE

This document aims to introduce the quality assurance processes applied in the design and development of native applications and mobile apps to ensure that the product delivered conforms to specifications. As a Quality Assurance Analyst and test engineer, the role will include the following:

• Application of Client business requirements and functional specs to define and develop proper testing procedure(s).
• Response to incidents (defects), recreating the event and analyzing the results to arrive at a proper resolution.
• Documenting all test results, recording them into an established repository (database), and submitting a daily test report to the team lead.

Quality Assurance (QA) for a web system (or apps) takes into account all desired functions explicitly stated in the scope of work and sitemap, and translates them into a process document (testing script)defining “action” and “response” where the intended function is tested and verified against the expected result.

Testing for all web systems and apps will adhere to the following FURPS+ model, the fundamental guide for assuring that each Software system/app is tested for:

• Functionality: the Software system/app adheres to the Client’s specifications.
• Usability: the Software system/app meets the Client’s expectations for layout and design; Consistency exists between Software systems and their live counterpart.
• Reliability: the Software system/app operates the same way with maximum accuracy regardless of external factors and conditions.
• Performance: the Software system/app functions in a fluid manner, with minimal navigation time or lag between pages such as logging in, sending a mail, making an entry, or when performing a search.
• Support: the Software system/app system can be tested and maintained correctly.
• Plus: any additional specifications required by Clients will be included in the testing process.

QUALITY ASSURANCE LIFECYCLE

INITIATION

A QA will include a full review of the SRS/BRS to ensure that the proposed functions and/or features are part of the final product.

PLANNING

A test plan will be drafted to establish what functions are going to be tested, how they will be tested, and the expected result. Obtain Functional Specs, Design Specs and makes sure that all requirements as specified in the SRS/BRS are part of the testing documentation. Likewise, Test Accounts (dummy account and cc#) are required as part of testing relevant to e-commerce or booking.

DESIGN& DEVELOPMENT

During the design phase, QA's tasks are:

- Drafting Test Cases
- Drafting Test Scripts(For regression test and keynote)
- Test on Browsers
- Test on devices

IMPLEMENTATION

During this phase, your tasks include checking:

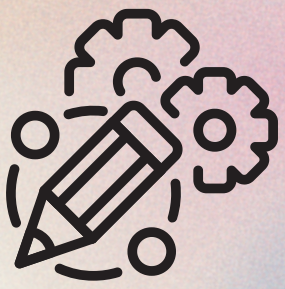
- Test the “production/live” version of the Software system/app
- Verifying Client issues and reporting them

TESTING

QA activities during the testing phase include:

- Ensure that all tests are performed according to test plans and procedures
- Reporting any defects.
- Ensure that test cases/test scripts are complete, accurate, and up-to-date.
- Verify that the test is completed and the software and documentation are ready for delivery; All reported issues during “production” have been addressed and resolved prior to deployment.

QUALITY ASSURANCE PROCESS



ACQUIRE THE PROJECT

a. If the project to test is a web system, acquire the URL by test environment readiness.
b. If the project is an app, and is in production, do the following:
i. Acquire the device build/project file from Developer/Project Mgr.
ii. Install onto a device

TESTING THE PROJECT

a. Regression Test: check that everything works as stated in Functional Spec.
b. Validation Test – Blank Check: test that forms validate for missing data on forms that require information and yield warnings
c. Validation Test – Invalid Data: test that forms validate for proper data formats and yield expected warnings
d. Verify input and output results against the "Parent" web system
e. User Interface (UI): verify app UI against Design Spec./Site Maps

WRITE TEST CASE(S) / TEST SCRIPT(S)

a. Compose "Test Case(s)/ Scripts" for each function found on the SRS/BRS and sign off the document to include the following:

- Test Case
- Criticality
- Description of Test Case (reason for the test)
- Required Parameter(s), Preconditions
- Input Trigger (i.e., button, link)
- Sequence of steps
- Expected Result
- Actual Result
- Test data
- Testing information
- Ticket information
- Verification (compare to web system)

REPORT ERRORS

a. Report errors in ERP
i. Mention the environment first where you have found the issue
ii. Include a description of the error
iii. Write the steps taken to produce the error (Path)
iv. Mention Expected and Actual Results
v. Verification (consistent w. SRS/BRS)
vi. Include notes as well ex. tested on IE 11.0, Issue has been found only for the 'Loan Entry' category, etc.
b. Send your results and updated test script to the PM and QA team lead
c. Submit Daily Test Report to Team Lead at the end of the day.

QUALITY CONTROL (QC)

ERROR REPORTING

Error reporting for the testing of all apps consists of tickets that are noted in ERP. The tickets fall into two (2) classes:

Tasks: Issue(s) to be brought to the attention of the PM and to be addressed with the Client.

Bugs: Issue(s) to be brought to the attention of the PM and to be resolved by the Developer.

INCIDENT RESPONSE

While the testing process is accurate, it cannot address all permutations of a specific scenario. Responding to an issue (defect) is a simple process:

o A Client submits a ticket with an issue that requires immediate attention from the PM

o If the situation is valid, the issue is forwarded to the QA Analyst working with the PM. The scenario is recreated. Non-technical issues are resolved immediately. Technical issues require escalation, and a ticket is created for the developer and assigned to the PM generating a said ticket.

o When the situation has been resolved, the PM replies to the initial ticket and it is returned to the QA Analyst for resolution.

o The Client receives the notification of the action taken and its resolution. The scenario is re-tested by the Client to ensure that the issue has been resolved. Any issues with an application system are checked in the test environment and in the actual live web system itself to further validate the problem.

FOLLOW UP REPORT

All errors reported in JIRA are to be met with a proper follow-up to ensure issues are being addressed in a timely manner. Create Follow-Up Report – A report is generated and consists of the following parameters:

o Ticket #: A reference to the ticket created.

o Short Summary: Short description of the found issue.

o Assigned To The PM to whom the project corresponds.

o Status: the state of the ticket at the time of the report. For the purpose of the report, the status should be, not started.

o Version: the display of app version tested (If any).

o Date Created: the date the incident was reported (ticket created)

o Project Developer: the developer assigned to the project.

QUALITY CONTROL (QC)
More

Which Ticket Status refers to what?

Not Started/Open – When QA creates a ticket.

Need feedback – Needs information from QA.

In-progress –
i) When QA provides more info for Developer/PM and re-sends to PM
ii) When QA does not see the issue fixed and send it to PM for further clarification.

Fixed – When the developer fixes the issue

Ready for QA – A ticket is ready for QA to verify whether the issue has been fixed or not.

Resolved/Done – QA confirms the issue has been resolved

Invalid – When the PM sees an issue that is out of the scope

Closed – QA confirms the issue has been closed.

QUALITY ASSURANCE TESTING PROCEDURE

TESTING PROCEDURE

QA procedure describes the detailed tasks when doing Daily QA, Production QA, and App QA.

Daily Regression Test

i. Task-1: Perform a regression test. Follow the test script.

• Use Internet explorer basic testing. Use other browsers if QA has been requested to do so.

• Clear cache and cookies after testing each test case

ii. Task-2: Add a ticket for the issue that you find on the Software system.

• First QA has to verify the issue with the browser system.

• If QA cannot reproduce the issue on the browser system, s/he will add a ticket to JIRA and assign it to PM

• If you see the same issue on their system, ignore that error.

iii. Task-3: Update the test script as per your results.

• Mention the result details, additional information, and ticket number, if QA has added tickets in JIRA.

i**v. Task-4:** Add/Update the test cases if QA feels it's necessary

• Add/update a test case for new functionality.

v. Task-4: Send the updated script to the PM for that project and the QA lead.

• Send your regression test results along with the updated script

• Mention the ticket number, summary, and priority in the email

PRODUCTION QA TEST SYSTEM

Production QA

i. Task-1: Test the production system

• Test for the new systems and/or expansion of the current systems.

• Obtain system map, timeline, and QA signoff document

ii. Task-2: Follow up with a PM and/or QA Lead and get the updates for the production system.

• Gather information for the functionalities that are added and/or for the functionalities which have been implemented.

iii. Task-3: Test for the new functional implementation and/or design updates

• Test the new functionalities and make changes to the test script accordingly

WRITE TEST CASE

Write/Update Test Case

i. Task-1: Follow the script (if available) received from the PM or QA lead

ii. Taks-2: If you find any test functionality that is not mentioned in the test script, add a test case(s) for the following:

- Test Case
- Criticality
- Description
- Preconditions
- Steps
- Expected result(s)
- Result (Pass/Fail/Def)
- Result Detail(s)
- Date Tested
- Tested by
- Ticket Info
- Comments

APPLICATION (APP) TESTING PROCEDURE

Testing an application in production

a. Install App

i. Task-1: Acquire the proper device and send it to PM.

ii. Task-1: Acquire mobile provision file and ".ipa" file from Developer/Project Mgr.

iii. Task-2: Copy the mobile provision file and ".ipa" to the Library/Apps section of SHAREit

iv. Task-3: Connect the device to the host computer/laptop via the device USB cable and "sync" the app onto said device

- Locate the device under the Devices section of iTunes
- Press the Apps tab on the navigation pane
- Locate the application and select it
- Press the "sync" button on the lower right side of the screen

b. Test App

i. Task-1: Regression Test: to check that everything works as stated in Functional Spec.

ii. Task-2: Validation Test – Blank Check: test that forms validate for missing data on forms that require information and yield warnings

iii. Task-3: Validation Test – Invalid Data: test that forms validate for proper data formats and yield expected warnings

iv. Task-4: Verify input and output results against "Parent" web system

v. Task-5: User Interface (UI): verify app UI against Design Spec./Site Maps

c. Write Test Case

i. Task-1: Compose “Test Case(s)” for each function found on the app to include the following:

• Test Case
• Criticality
• Description
• Preconditions
• Steps
• Expected result(s)
• Result (Pass/Fail/Def)
• Result Detail(s)
• Date Tested
• Tested by
• Ticket Info
• Comments

Testing a Live App

a. Install App

i. Task-1: Share the application using “SHAREit” and install it onto device

b. Test App

i. Task-1: Regression Test: to check that everything works as stated in Functional Spec. and all issues have been corrected

ii. Task-2: Validation Test – Blank Check: test that forms validate for missing data on forms that require information and yield warnings

iii. Task-3: Validation Test – Invalid Data: test that forms validate for proper data formats and yield expected warnings

iv. Task-4: Verify input and output results against “mobile”& (HTML) web system

v. Task-5: User Interface (UI): verify app UI against Design Spec./Site Maps

c. Update Test Case

i. Task-1: Compose “Test Case(s)” for each function found, if none exist

ii. Task-2: Update Test Case(s) to reflect changes on app, if such exist

Please, Click on GitHub to see Sample Test Cases/Scripts, Automation Scripts, Jmeter (Performance testing), Bug reports, and more.

Thank You

Sector-5, Uttara, Dhaka.
https://github.com/anik788
+880 16 800 70 218
nianik.cse@gmail.com