Working in a Team

Introduction

This exercise covers both hard and soft skills within QA. You'll be practicing effective communication, writing a test plan, and evaluating software methodologies. Each of these sections highlights part of the team environment found in many tech companies. Working in a team requires great communication and documentation regardless of what style your future team works in.

Part 1 of this exercise covers how to send in status data or information to your QA Manager, QA Lead, or QA Director. We'll cover a negative status report example.

Your QA Manager will typically provide you with a template, but for learning purposes use the examples.

We'll also cover managing Imposter Syndrome when working within a Software Development team. Finally, we'll cover accountability and how to professionally learn from your mistakes.

In Part 2, you'll read over two currently used styles of creating Requirements specifications and User Stories with Acceptance Criteria. You will analyze the acceptance criteria and pull what tests you need to create. I've added a few tips to ensure your test plan (and test cases) cover all of the acceptance criteria/requirements.

In Part 3, you'll review Software Development Lifecycles. Different companies and teams use different processes and it's important to know how these processes work and what the benefits and drawbacks of each are.

Part One

Introduction

The Status Report. Often in Agile teams, your Status Report occurs during your daily standup. You will report what user stories you are currently testing, and if you are having any issues or need confirmation. If you're in the standup, take the opportunity to request help from Product Management or Development if you have questions on the meaning of a story and its acceptance criteria. It's the Product team - be it a Product Manager, Product Analyst or Business Analyst who typically create the user stories and add acceptance criteria to them.

If your QA Manager or leader requests you send them a Status Report then create a document to attach to an email or put your report in the body.

A possible example:

QA Name: Bubbles Tester

Assigned Feature: Medical Provider Prescribing

Date: March 1, 2022

Date Due in QA Project Plan: March 4, 2022

Testing Progress: 80% complete, 175 of 200 test cases passed, 18 failures, 18 defects entered, 6 enhancement requests added.

NOTE: The last 25 tests require setup with a production level API not accessible in the test server. Working with Joan Doe in Development to add a mock API for testing. Once the API is configured and working, I will need 2 additional days to execute tests. Expected completion date: March 8, 2022.

Steps

1. Choose an app from a previous lab exercise and create 2 Status Reports. Put yourself in the shoes of a QA on that project, what might they update their leaders on? Write one report delivering bad news and another delivering good news.
2. Using an app from a previous lab exercise (can be the same or different from step 1), create a Test Summary Report. Here is an example:

Test Report Template

Project Information

- Project or Release Name/Number
- Team assigned (if relevant add team members and roles)

Test Objectives

- Testing Executed:
  - Regression
    - Feature/Function
    - Integration
  - Performance & Load
  - Unit tests - or any developer side test execution

List out the objective of each type of test. For example, Regression testing for a new feature. Your objective for Unit Tests may be to ensure each code checkin passes the Unit test suite indicating the main functions are working as expected.

Test Execution Details

- Name of Tests or Test Suites

Location of tests, test suites

Test Execution Results

- Number of tests by type
- Passed
- Failed
- Skipped
- User or team who executed tests, on what server and what date(s)
- Defect number and description for all defects entered from the test effort.

Test Coverage

- Graphical illustration of lines of code tested, compared to lines of code in application. Information that gives the reader a better idea of what is covered by testing.

Test Summary

- Paragraph summary of tests executed, results, and defects entered. Reads like a technical extract - gives you the basic information without the explicit details.

Tip

Keep your status updates professional. Never imply blame to others and be honest. If you're stuck, you're stuck. If you have more tests than you can complete, say so. Be upfront. Most QA Managers want to know that you're reaching out for help and how it's going. If needed, they may need to step in and assist.

3. Read the information below and answer the questions in bold.

You'll miss something. Some defect will get missed in a release and a customer will find it, report it, and you will hear about it. Guaranteed. All you can do is review your Test Plan and Test Cases. Did you include a test for that particular test scenario? Ask yourself - How did I miss seeing the defect? It may be that the defect is only present in the customer's instance. Verify by testing on your test server. Do you see it? If you do, then add it to your test case. Include it in any regression test executions in the future to avoid the defect appearing again. If you cannot reproduce the defect in testing, request help from a developer to determine the root cause. The defect may be a result of a specific configuration setting or platform the customer is using, but the test server is not. Add any specific testing you need to add to your test case(s) and be sure to add it to any regression testing suites for future test execution. Accountability is recognizing something is missing from your test cases, finding it, and adding in any missing test setup or steps needed. When you get another new feature, keep the scenario in mind when reviewing requirements, acceptance criteria or user stories. Make sure customer scenarios are considered, and then include them in your test cases when relevant.

Imposter Syndrome is feeling you're not qualified to report a defect or question a design decision. As a professional QA tester, you are the only defense between defects and the customer. Your goal is to test to find all the defects you can before it gets to the customer, as much as possible. You're the QA - it's literally your job to enter defects and ask as many questions as it takes so you understand. Enter defects when you find them. By entering them, you're adding to the quality of the release.

Do you feel qualified to test an application? Do you feel confident reporting a defect? Describe why or why not.

Part Two

Introduction

In this section, you'll write a Test Plan using the tool of your choice. The app you'll be testing is called Helo, you can see it here.

Setup

- Create a Trello board (or Jira, Word, Google Docs or Sheets, or Excel). You should have cards for each test group, add test cases to each card.
- Review the Requirement Specifications below.
  -The purpose is to practice your ability to dissect requirements and user stories and create a thorough test plan with valid test cases, and write or document defect reports.
- Remember to add Traceability to your test cases indicating what requirement or acceptance criteria the test covers.
    - *Where do you add Traceability typically?*
    - *What is the purpose of Traceability?*

Requirement Specifications

Product

Helo is a blog posting application that allows a user to register, login, and then post new blog content or search and view existing blog content.

Helo blog content is an intranet site for employees only to facilitate cross department communication.

Purpose

Improve internal employee communication with an online tool for posting blogs.U74E

Overview

Helo is used to communicate between employees and departments. Users can post blogs for How-Tos, usage guides, procedures or cross department training. Users add new posts as needed. Users can view existing posts upon login.

Functional Requirements

1. User must authenticate via login page.
2. User can enter Name and Password, and Register.
3. Once registered, user can login by entering a valid Name and Password and login.
4. Create a new Post with Title, Image, and Content.
5. View Posts on Home page.
6. Search Posts on Home page.

Steps

1. Review the user stories and acceptance criteria in the Helo list on Trello, here.
2. Critically review each story, and create test cases for each acceptance criteria.
3. Once you have your Test Plan, then write out a test case for each Acceptance Criteria, or combine acceptance criteria if it makes sense in a customer workflow.
4. Now execute your tests on the application above.
5. Report any defects by adding to the Test Case you create. Mark the defect in red, and write up a defect report with the following:
    1. *Description or General Summary that includes a basic description of the defect*
    2. *Steps to reproduce*
    3. *Expected Results - TIP: This is the acceptance criteria*
    4. *Actual Results - Description of the defect and how it breaks the acceptance criteria*

Part Three

Introduction

In this section, you'll be evaluating the waterfall and agile methodologies, respectively.

Steps

1. Create a table or chart to list out the pros and cons
2. List at least 3 pros and 3 cons for each method: waterfall and agile (at least 12 total)
3. Answer this question: How do you think planning a project differs between these 2 methodologies?

Save your work to GitHub.