**Deel-QA Automation/Manual Test Plan**

**Name:** Bright Kwame Nuertey

**Date Started:** 06/07/2021

**TEST COVERAGE/SCOPE**

The scope of this exam is to test 3 different fields on the sign up page of Deel (<https://dev.deel.wtf/>). The fields are as follows:

* First Name
* Date of birth
* Phone number

First Name

Test 1 - Enter first name with characters only

Test 2 - Enter first name containing special characters such as hyphen e.g Ann-Louise

Test 3 - Check the maximum limit of the field by entering characters till the field gets full

Test 4 - Check the minimum limit of the field by entering a single character

Test 5 - Enter first name with a mix of characters and digits

Test 6 - Enter first name with digits only

Test 7 - Leave the name field blank

Test 8 - Create whitespaces in the name field by pressing the spacebar a couple of times

Test 9 - Enter only special characters

Test 10 - Enter first name that has whitespace in them e.g Alphonse Joseph

Date of Birth

Test 1 - Enter DOB in the format DD/MM/YYYY

Test 2 - Enter DOB in the format MM/DD/YYYY

Test 3 - Enter DOB in the format YYYY/MM/DD

Test 4 - Enter DOB in the format MM/YYYY/DD

Test 5 - Enter DOB in the format YYYY/DD/MM

Test 6 - Enter only two digits for year in the format DD/MM/YY

Test 7 - Enter a single digit for day in the format D/MM/YYYY

Test 8 - Enter a single digit for month in the format DD/M/YYYY

Test 9 - Enter single digits for both day and month in the format D/M/YYYY

Test 10 - Enter single digits for day and month, and two digits for year in the format D/M/YY

Test 11 - Leave the DOB field blank

Phone Number

Test 1 - Select correct country code and enter valid number e.g +233 262101670

Test 2 - Select correct country code, enter valid number but starting with 0 e.g +233 0262101670

Test 3 - Do not select country code but add it to the phone number field e.g +233262101670

Test 4 - Select a different country code and enter valid number e.g +234 262101670

Test 5 - Select correct country code and enter invalid invalid number e.g +233 8185278530

Test 5 - Do not select country code,enter valid number e.g 262101670/0262101670

Test 6 - Check the maximum capacity of phone number field by selecting a country code that has phone numbers containing 12 digits and entering a valid 12-digit phone number

Test 7 - Check the minimum capacity of phone number field by selecting a country code that has phone numbers containing 4 digits and entering a valid 4-digit phone number

Test 8 - Enter any digits less than 4 in length

**AUTOMATION**

**Tools:**

Due to dev.deel.wtf being a website/webapp, the primary tool I will use in my automation test is Selenium webdriver. The other tools in the framework are chromedriver for starting a chrome instance, TestNG being the test runner and Java as my programming language of choice.intellijIDEA is the IDE used to write the automation scripts.

**Processes:**

The first step in the automation process is selecting the tools to carry out the test. Test tool selection is largely dependent on the technology the application under test is built on. As an example it is not right to choose a tool like Jmeter for a functional test. These are some of the things I considered before choosing Selenium:

* It has data driven capabilities
* It tool has debugging and logging capabilities
* It is platform independent
* It is easily extensible

The next phase is defining the scope of the test. The scope is the area of an application which will be automated. I chose to automate the first name, date of birth and phone number fields because they are important, users interact with them a lot and they have the potential of not catering to edge cases. Also the automation script generated for these fields can be rescued with little changes.

The next phase of the process is planning, designing and development of test strategy and scripts. During this phase, I go through the flow of the fields I have chosen to automate. This helps me get a general idea going into scripting and also noticing any oddities.The test script can be designed and developed using two models, Single Page or Page Object Model. For this test I will opt for the single page model because the fields are few.

At the test execution stage, I will execute the tests developed using data I have generated for each field. After running detailed test reports will be generated and this can be shared with the development team.Hopefully at this stage we should be able to generate some bugs.

With the addition of new constraints to the fields depending on the bugs generated during successive cycles, it will be necessary to review and maintain automation scripts to make sure all the changes made to the code of the fields under test have been factored in.

**Stages:**

The first stage of testing is doing a unit test. For each field component, the code developed should deliver the desired output. Each unit of the code will be thoroughly tested to ensure it meets the specification before progressing to another unit.

The second stage is component testing. At this stage I will test each component i.e first name field, date of birth field and phone number field individually to verify their expected outputs.The functionality/usability of each component is verified at this stage.

The third stage will be to perform a smoke test. This is to check if the new build is usable (basic functionalities are working or not) enough for in-depth testing.

The next stage is carrying out an integration test. This test will be performed to check how the individual components function together. Ideally, the components should work fine individually and not show bugs when integrated with other components.

Next is the regression tests. It is possible that a modification of a functionality/feature may introduce errors in an application thereby causing it to behave unexpectedly. Regression tests will be undertaken to confirm that code changes have not adversely affected existing features/functionalities.

Sanity tests will be run to determine that modifications made to code have actually fixed bugs and no further bugs have been introduced by the fixes. This will be done instead of regression tests on builds with minor modifications.

The last stage of the automation test is to do system testing. This is where the integrated system including my chosen fields are tested as a whole to verify whether they meet the technical, functional and business requirements of stakeholders.

**Reports:**

Automation reports will be generated and viewed using allure. Test cases, other documentation and bug tracking will be managed via Jira.

**TEST TYPES**

**Sanity (smoke) test:**

Smoke testing is a type of preliminary software testing used to test whether a new software build is stable enough to warrant in-depth testing. If the software build is not stable it will be sent back to the developers to fix the code. I will automate my smoke testing by running pre-recorded tests on the build and recording whether the functions work or not. Below are some of my recommendations on how to go about smoke testing:

* Smoke tests should be carried out early during the early stages of development. As each code is committed a smoke test will be run on it. This will help to find critical bugs that could impede the team’s progress early on and fix them.
* Recording all smoke tests that are carried out and reviewing the test cases periodically. With a test case repository in place it becomes easy to plan a test as all it would require is to get the relevant test scenarios and test cases under them without having to go through the difficult process of creating new test cases from scratch.
* Smoke tests should be quick. The aim of smoke testing is to catch bugs that block development progress at the initial stages of the project. There wouldn’t be a need for a complex test case as more in-depth tests can be executed later on.
* Use smoke tests to determine whether the code is ready for more complex tests. If most of the basic tests based on critical functions pass then the code can move on to the other types of testing. But if they are not passing those tests then the code has to be reworked by the developers.

Sanity tests will be carried out to check whether bug fixes and intended enhancements have been completed as expected and that no further issues are introduced due to these changes. Sanity tests will be executed just after smoke testing is done.

**Regression test:**

These are some of my recommendations on how to go about regression tests:

* Create and maintain a regression test case pack. This is simply a compilation of all regression tests that are performed as the software is updated. With such a pack testers can easily incorporate ad-hoc testing or random tests.
* Give more attention to the most frequent use cases of users. Such use cases contain the core functionalities and features of the software. This requires understanding what customers' needs are and the features and functionalities they most rely on. An effective regression tests pack integrates tests that guarantees that the core functionalities/features are working as they are supposed to.
* Run the full regression test case only when it is needed. For sligh feature modification and bug fixes, it is okay to run a smoke test and sanity test upon commit.The regression test case must be modular too as well, so each component/feature will have a regression test case. In the event of the modification of a particular component, only the regression test case of that component will be executed.
* Regression test cases can be repeated. Test cases that have recognized bugs and glitches should be included in the regression test case pack.

**End2end test:**

End to end testing tests the software as a whole for dependencies, data integrity and communication with other systems, interfaces and databases. This type of testing is carried out after system testing and it simulates production or real-world scenarios. It uses actual production like data and test environment to simulate real-time settings.

These are my recommendations for carrying out an end to end test:

* Build user functions by listing down the features of the system and its interconnected components. List the input data, action and the output data for each feature or function. Identify the relationships between the functions and determine whether a function can be reusable.

In addition to this conditions can be attached to the user functions. Conditions can be sequence, timing or data conditions. For example on the first name field, a compound name can be checked for.

* Build a test scenario for user functions defined. An example is, enter a phone number scenario.
* For each scenario defined one or more test cases can be built. The test case may include each condition as a single test case.