

Software Testing Technical Challenge

What Is Software Testing?

There are many definitions but broadly:

Software testing is a process of executing an app / website / program / application / cloud service / product etc. with the intent of finding the software defects / bugs. (We use the term ‘app’ from now on to refer to any software under test)

Software testing is done by planning and recording the execution of the app to check that what we expect to happen, actually happens!

Any variation between expected and actual results is a bug or defect.

Software testing cannot prove that the app is defect / bug free, only that for the tests executed the results recorded were observed.

This provides the business an objective view of the app (never trust a developer!) to allow the business to appreciate and understand the risks of implementing the app.

A good software tester is one that is able to provide the business with confidence that all likely scenarios have been tested and bugs identified for the least cost..

We suggest you read this document fully before starting.

In this activity you will:

Learn what software testing is.

Test a simple app.

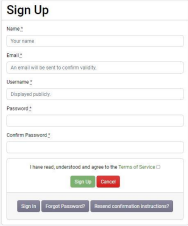
Optionally learn how to automate software testing.



Task 1: Learn what software testing is:

1) Sign up for the (free) Club membership of the Ministry of Testing at:

a. Ninja training for software testers | MoT (ministryoftesting.com)



2) Complete the essentials courses at: Ministry of Testing - Introduction to Software Development and Testing



Task 2: Test a simple app: Scenario:



As part of their online offer a training business wants to be able to validate a subscriber’s name, email address and credit card number when they book a course.

The commercial director, marketing director and CTO were identified as stakeholders for this app and identified the User Stories in Appendix A:

3) Review the User Stories in Appendix A

4) In the table below record the risks you can identify for this app?

Based on what you learned about Risks from the Ministry of Testing complete the table below with any Risks you can identify.

| ID | Risk | Impact | Likelihood | Mitigation |
| --- | --- | --- | --- | --- |
| 1 | <Describe the Risk> | <What is the  impact if the risk happens?> | <How likely is the risk?> | <What can be  done to mitigate it> |
| 2 | Example: SQL Injection | High: lost  database | High: Someone is going to do it. | Test for SQL  Injection |
| 3 | Database may not accept characters from non-roman alphabets/syllabaries | Low: some users may struggle to input names or emails | moderate | Test inputting non-roman characters |
| 4 | Webpage may not work on some browsers | Moderate: Lost potential users | High: There are a lot of device types which function differently | Test application on as many device types as possible |
| 5 | Webpage may not work on some devices | Ibid | ibid | ibid |
| 6 | Webpage may display inconsistent branding | Low to moderate, depending on severity of inconsistency | Moderate, depending on volume of branding and the manner of implementation | Check all instances of branding on application for consistent hex codes, colours, placement etc. |
| 7 | Webpage may fail to meet accessibility guidelines | Moderate: application shall be inaccessible to many potential users | High: most applications and webpages do not comply with this. | Test each page of application is compliant with wc3 accessibility initiative |
| 8 | Webpage may not render properly on some screens | Moderate, lost potential users | High: device displays use a functionally limitless range of aspect ratios and renderers. | Test webpage rendering on as many device displays as possible. |
| 9 | Form may accept impossible credit card numbers | Low: users may input incorrect numbers and have these accepted | Low: credit card numbers follow a precise formula and it is probable that the input criteria reflect this. | Test that only LUHN-algorithm compliant card numbers can be entered |
| 10 | Form may accept impossible email addresses | Low: ibid, mutatis mutandis | Moderate - email addresses are fairly varied in syntactic layout | Test that only correctly formatted email addresses can be input. |
| 11 | Form may accept invalid names | Low, provided SQL injection is not possible. Users may not be notified of errors. | High - there is no formula for names. | Test that only properly formatted names can be input successfully. |
| 12 | Form may not notify user of mistakes in input | Moderate - could cause user frustration | Low - this is a fairly standard feature. | Test that incorrect inputs are met with an error notification |
| 13 | Application may slow down server | High - Slowdown could cause user frustration, or unforeseen emergent errors. | High - slowdown in input forms is a common problem. | Test that all validation is done on-page. |

Often User Stories are missing all the information a tester needs to define tests. This can be considered a requirement defect or bug. These situations can be tracked through issues or by arranging a session to discuss the user stories to get clarification.

5) Record the issues you can find in the User Stories?

Create a list of issues that you would like to discuss with the stakeholders.

| ID | User  Story ID | Issue / Defect |
| --- | --- | --- |
| 1 | 10 | ‘As CTO I need the application to follow our UX (User Experience)’ The UX guide is not available. |
| 2 | 4 | Further clarification on what constitutes a “valid name” would be useful, as unlike email addresses or credit cards, names follow no defined formula. |
| 3 | 6 (the second one) | Unclear if page should only use brand colours, or if brand colour requirements are limited to instances of similar colours on page, or only to instances of copyrighted branding on page. |
| 4 | 13 | “As CTO I need to ensure that all applications can run on all common browsers, devices and screen sizes.” - clarification needed on which devices, screen ratios, and browsers are “common”. |
| 5 | 13 | Clarification on what it means here for applications to “run” on all common browsers devices and screens - does this mean all aspects functional or identical/near identical display of webpage. |
| 6 | 11 | Confirmation that ‘@’ is also an accepted character for input, or the email entry box will not work for any email addresses. |



If we cannot get answers from the stakeholders, then we can use assumptions to carry on testing. 6) Record the assumptions you need to make to allow you to carry on testing?

| ID | Issue ID | Assumption |
| --- | --- | --- |
| 1 | 1 | Only the UX requirements in the user stories will be tested i.e.  Issues being highlighted to the user ASAP, preferably when they are still entering a field.  Correct fields should be shown as green  Errors should show in DN Pink |
| 2 | 2 | A “name” shall be assumed to be any string of only alphabetic or analogous characters, and hyphens. Thus all punctuation symbols with the exception of “-” shall be treated as making a string an invalid name. |
| 3 | 3 | All instances of green, grey, and pink shall be checked for compliance. Other colours e.g. white shall not be counted as anomalous. |
| 4 | 4 | For purposes of practicality I can only test for the devices and operating systems I own - these shall thus be taken to constitute “all standard devices”. This is obviously somewhat inadequate.  - Samsung Galaxy A14 phone - android operating system, 6.6” display. Firefox and Chrome browsers  - Lenovo ideapad 5 laptop, Windows 10 operating system, chrome, firefox, and edge browsers. Connected to a 22” monitor. |
| 5 | 5 | Provided webpage allows all required inputs to be made successfully on a device, display, or it shall be assumed to be running |
| 6 | 6 | ‘@’ shall be treated as an acceptable character. |



Once we have identified the risks, made assumptions about the areas where we don’t have enough information, we are ready to start designing our testing activities. Typically, testers use a mix of structured (sometimes call scripted) and exploratory techniques. You will now design a series of structured tests.

Note: It can be useful to think about writing these tests for somebody else to execute.

Test cases are used to define the tests that we will execute. These can be grouped into a test suite. We’ve simplified things for this example and have provided a Template Test for you to add your tests to in Appendix B. The Test Suite Template contains the following columns

**Test Case ID** A unique ID for the Test case used in the defect report if the test fails,

**User Story ID** Test cases should relate back to a User Story, if it isn’t in a user story, it shouldn’t be tested…

**Acceptance Condition** User Stories will often contain acceptance criteria that the software tester will propose, and the stakeholders will agree to. i.e. Our User Story could say As a user I need to enter my name so that I can… acceptance criteria for that story could be: Western Style First Name and Surname would be a valid name. Title, First Name and Surname would be valid.

**Preconditions** What needs to have happened before we can execute this test?

**Test Data** What test data do we need to use?

**Steps / Instructions** What does the tester have to do?

**Expected Results** What do we expect the outcome to be?

**Actual Results** What actually happened (this can include screenshots as evidence.)

**Pass / Fail** Did the Actual match the Expected?

**Notes / Defect ID Raised.** Any notes from the test or If the test failed record the ID of the defect. 7) Create a suite of the tests you plan to execute in the Template in Appendix B.



Having written our ‘structured’ or ‘scripted’ tests we will try some ad-hoc testing to get to grips with the app.

8) Open up the app to test at: Software Testing Work Trial Activity – Digital Native (dn-uni.com) review it and try it out, 9) Record below any bugs that you find while trying the application out (ad-hoc testing). Ad-hoc test Defects:

| ID | Issue | Evidence |
| --- | --- | --- |
| 1 | Example: Screen has other Digital Native Branding |  |
| 2 | Input boxes are not aligned | ibid |
| 3 | ‘44’ is a valid name |  |
| 4 | Successful entry opens mail application |  |
| 5 | Text is larger than other DN pages | See evidence for issue 1 and |
| 6 | 0000 0000 0000 0000 is technically a LUNH compliant card number but is treated as invalid |  |
| 7 | 8888888888888888 is not a LUNH algorithm compatible number but is treated as a valid input |  |
| 8 | Blank spaces are treated as valid name inputs |  |
| 9 | If data is input from a chrome autosuggestion the data is not displayed as valid or invalid |  |
| 10 | Invalid inputs are displayed in red rather than in DN pink |  |

10) Now execute your test suite – record the actual results of each test in your test plan.

11) Create defect log entries for any tests that fail.

Structured test defects:

| Defect ID | Test  ID | Issue | Evidence |
| --- | --- | --- | --- |
| 1 | 6 (second one) | Example: Screen has other Digital Native Branding |  |
| 2 | 4 | ‘Jon4than Do£’ is treated as a valid name |  |
| 3 | 5 | Invalid email input displays red box instead of pink |  |
| 4 | 6 | Invalid card displays red box instead of pink |  |
| 5 | 8 | Software Testing Work Trial Activity' is in Montserrat font |  |
| 6 | 9 | Font size is not consistent with other DN pages | and ibid |
| 7 | 10 | Input boxes are misaligned |  |
| 8 | 11 | Webpage opens another application during loading |  |
| 9 | 12 | no text is supplied to clearly correspond to digital native branding images |  |
| 10 | 13 | Input box does not turn the correct shade of green when showing a valid input |  |
| 11 | 14 | Input turns green for invalid name input |  |
| 12 | 15 | Invalid character input is accepted ‘name’ submission box |  |
| 13 | 16 | Box displays a card number that does not match LUNH algorithm as green |  |

12) Reflect on the activity.

a. Did your structured testing find all your ad-hoc testing defects?

Structured testing was a lot less effective at catching precisely what was wrong with the name input field, and missed the chrome autofill problem.

b. What have you learned?

* Structured testing requires a broader sequence of counterexample entries in order to identify the source of a problem.
* A different defect can cause an otherwise successful test to fail.
* Accessibility checklist for webpages
* How to calculate LUNH algorithm, and how to automate this with Python
* Basics of how testing is integrated into a software development team
* Difference between waterfall and agile development environments#
* Testing is fun!

c. What would you do differently?

* I would create a more thorough suite of false input tests for each input box in order to more clearly identify what is going wrong in the event that there is a problem.
* If I had access, I would use a larger range of devices, operating systems, screens, and browsers,

d. What would be a better way of capturing the steps to recreate a defect?

* If the steps were automated using selenium or similar, I could provide the code to execute the automated test exactly the same way every time. .

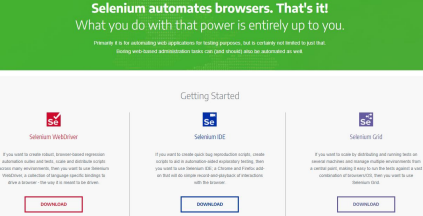
13) Send your completed Technical Challenge and reflections to hello@dn-uk.com



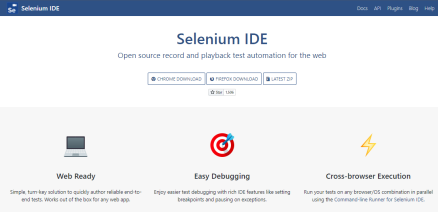
Task 3: Optional – Learn how to Automate Testing:

Running the same tests every time software changes is expensive, slow and a poor use of brilliant tester resource. There are several tools that can be used to automate test execution.

Selenium is a simple tool which allow browser/platform actions to be recorded and played back. 1) Research Selenium About Selenium



2) Watch the Selenium IDE demo https://www.youtube.com/watch?v=ZG3VFDMaAlk 3) Download the Selenium IDE Extension through Google Chrome/Firefox Web Store or click the image below:



4) Record one of your test cases with the Selenium IDE extension

5) Locate review and then execute the recorded code in the Selenium IDE



6) modify the code to run one of your other tests

7) Insert the modified code below.

{

"id": "4f8c511e-7b09-46fa-a4b0-e19d59fddeee",

"version": "2.0",

"name": "Test 1",

"url": "https://dn-uni.com",

"tests": [{

"id": "47b1c235-05f6-44cf-ba51-e390cee6075d",

"name": "Test 1",

"commands": [{

"id": "6fbf22f7-bd59-4251-9133-0fedb072baec",

"comment": "",

"command": "open",

"target": "/sftl4-trial-activity/",

"targets": [],

"value": ""

}, {

"id": "1bd9c001-f090-4903-8eb6-cb6a92a6264e",

"comment": "",

"command": "setWindowSize",

"target": "1296x696",

"targets": [],

"value": ""

}, {

"id": "ead03bbf-3517-4668-84ef-5a1c30740072",

"comment": "",

"command": "click",

"target": "id=name",

"targets": [

["id=name", "id"],

["name=Name", "name"],

["css=#name", "css:finder"],

["xpath=//input[@id='name']", "xpath:attributes"],

["xpath=//div[@id='divtest']/form/p/input", "xpath:idRelative"],

["xpath=//input", "xpath:position"]

],

"value": ""

}, {

"id": "ac5dc65d-4bc9-460e-ae1e-69af41f0aa8e",

"comment": "",

"command": "type",

"target": "id=name",

"targets": [

["id=name", "id"],

["name=Name", "name"],

["css=#name", "css:finder"],

["xpath=//input[@id='name']", "xpath:attributes"],

["xpath=//div[@id='divtest']/form/p/input", "xpath:idRelative"],

["xpath=//input", "xpath:position"]

],

"value": "Jon4than Do£"

}, {

"id": "9cc63d5e-a057-4fab-af96-13f6d88a4cdc",

"comment": "",

"command": "click",

"target": "css=p:nth-child(2)",

"targets": [

["css=p:nth-child(2)", "css:finder"],

["xpath=//div[@id='divtest']/form/p[2]", "xpath:idRelative"],

["xpath=//p[2]", "xpath:position"]

],

"value": ""

}]

}],

"suites": [{

"id": "5f5c489e-15dc-4fb5-a63e-3db29fbf693b",

"name": "Default Suite",

"persistSession": false,

"parallel": false,

"timeout": 300,

"tests": ["47b1c235-05f6-44cf-ba51-e390cee6075d"]

}],

"urls": ["https://dn-uni.com/"],

"plugins": []

}

Appendix A – User Stories



| ID | Stakeholder | User Story |
| --- | --- | --- |
| 1 | Commercial Director | As a user I need to be able to insert my name into the form |
| 2 | Commercial Director | As a user I need to be able to insert my email into the form |
| 3 | Commercial Director | As a user I need to be able to insert my Credit/Debit Card into the form |
| 4 | Commercial Director | As the Commercial Director I need users to only be able to enter valid names |
| 5 | Commercial Director | As the Commercial Director I need users to only be able to enter valid email address |
| 6 | Commercial Director | As the Commercial Director I need users to only be able to enter valid credit card |
| 6 | Marketing  Director | As the marketing director I need the page to conform to our branding guidelines and use our brand colours: Green RGB 137,200,46 Hex 89c82e Grey RGB 60,60,59 Hex 3c3c3b Pink RGB 231,0,100 Hex 700064 |
| 7 | Marketing  Director | As the marketing director I need the page to be consistent with our other pages i.e. using the same Calibri font and text size, centred horizontally and vertically and to match the wireframe below: |
| 8 | CTO | As CTO I need the application to reduce the load on the back-end server, validation should be done on page. |
| 9 | CTO | As CTO I need to ensure all applications can be accessed by all users, the application needs to conform to w3c accessibility initiative. Introduction to Web Accessibility | Web Accessibility Initiative (WAI) | W3C  Checklist of Checkpoints for Web Content Accessibility Guidelines 1.0 (w3.org) |
| 10 | CTO | As CTO I need the application for follow our UX (User Experience) guide so that each user has the best possible experience. This should include any issues being highlighted to the user when they are still entering a field of as soon as possible after.  Correct fields should be shown as DN green, errors shown in DN Pink. |
| 11 | CTO | As CTO I need to ensure that any data capture is secure and there is no risk of database corruption through SQL injection. Only standard upper / lower case. letters and printable characters: !#$%&'\*+-/=?^\_`{|}~ should be allowed in all fields. |
| 12 | CTO | As CTO I need to ensure that all accepted credit cards conform to the LUHN algorithm. |
| 13 | CTO | As CTO I need to ensure that all applications can run on all common browsers, devices and screen sizes. |



Appendix B Simplified Test Suite Template

Below is a simplified Test Suite Template this contains the test cases that need to be completed.

Copy this into Excel. Tools such as Jira and Azure DevOps are often used to track tests for User Stories.

| Test  Case  ID | User  Story  ID | Acceptance Condition | Preconditions | Test Data | Steps /  Instructions | Expected Results | Actual Results | Pass  / Fail | Notes |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | ‘John Doe’ is a valid name | Web page showing  validation screen is  loaded.  Web page contains input capable ‘Name’ field. | John Doe | Enter ‘John  Doe’ into Name field. | On leaving the field (tab or mouse  away) the name  field entry box  turns green. |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

# 