|  |
| --- |
| **SOUTHERN CROSS UNIVERSITY** |

**ASSIGNMENT COVER SHEET**

For use with online submission of assignments

Please complete all of the following details and then make this sheet the **first page of each file of your assignment – do not send it as a separate document.**

Your assignments must be submitted as either **Word documents, text documents with .rtf extension or as .pdf documents**. If you wish tosubmit in any other file format please discuss this with your lecturer well before theassignment submission date.

|  |  |
| --- | --- |
| Student Name: | **Kaine Cochrane** |
| Student ID No.: | **22357937** |
| Unit Name: | **Managing Software Development** |
| Unit Code: | **CMP73010** |
| Tutor’s name: | **Barry Wilks** |
| Assignment No.: | **1** |
| Assignment Title: | **Assignment 1** |
| Due date: | **28/08/2017** |
| Date submitted: | **28/08/2017** |

Declaration:

*I have read and understand the Rules Relating to Awards (*[*Rule 3 Section 18 – Academic Misconduct Including Plagiarism*](http://policies.scu.edu.au/view.current.php?id=00140#s18)*) as contained in the SCU Policy Library. I understand the penalties that apply for plagiarism and agree to be bound by these rules. The work I am submitting electronically is entirely my own work.*

|  |  |
| --- | --- |
| Signed:  (please type your name) | Kaine Cochrane |
| Date: | 28/08/2017 |

Table of Contents

[Part 1 – Testing 3](#_Toc491701811)

[Acceptance Test 3](#_Toc491701812)

[*Acceptance Test Criteria 3*](#_Toc491701813)

[*Acceptance Test Plan 3*](#_Toc491701814)

[*Acceptance Test Derivation 5*](#_Toc491701815)

[Detailed Black-Box Test Plan 6](#_Toc491701816)

[Part 2 – Configuration Management 8](#_Toc491701817)

[Code/File Version Management 8](#_Toc491701818)

[Build Management 8](#_Toc491701819)

[Part 3 – Request for Proposal (RFP) 9](#_Toc491701820)

[Quick Facts about the RFP 9](#_Toc491701821)

[Organisation Background 9](#_Toc491701822)

[Contact Information 9](#_Toc491701823)

[Criteria for Making Our Decision (How We Evaluate Proposals) 9](#_Toc491701824)

[System Description 9](#_Toc491701825)

[Appendices 11](#_Toc491701826)

[Appendix A – References for Build Management 11](#_Toc491701827)

[Appendix B – References for Request for Proposal (RFP) 11](#_Toc491701828)

# Part 1 – Testing

## Acceptance Test

### Acceptance Test Criteria

This acceptance test is being developed for an online plagiarism tester for student submissions. The acceptance criteria for this system is as follows: the time and dates of student submissions must be tracked accurately, an analysis must be given of similarities (matches) to other student work submitted to the system as well as other online documents, paper-based books, magazines etc. The analysis the system completes must be available to both the student who submitted the work as well as the designated marker. Analysis the system completes must be configurable as per the following four points – optional bibliography checking, proper quotations included/excluded, set minimum number of matching words, and allowing resubmissions up to a set date/time.

### Acceptance Test Plan

A number of rules need to be established in order to properly plan out the acceptance testing. These mainly concern time (i.e. when scheduled testing should occur), budget, and resources allocated to the testing. In this case of this system – tests should be conducted whenever a change is made to any feature, no matter how minor. If more than a week passes without any changes to the system, a test will be conducted anyway. These weekly tests may expose new bugs in the software which were missed during a previous run-through. The tests will always test for **all** of the above criteria; this is to ensure no bugs or major flaws have been introduced as a result of changes being made. The three major features of this system are the student submissions, the analysis, and the configuration. The order in which the product features will be tested will be:

1. Student submission – time tracking
2. Student submission – date tracking
3. Analysis – matches found in other student work
4. Analysis – matches found in Internet documents
5. Analysis – matches found in paper-based publications
6. Analysis – can be viewed by student
7. Analysis – can be viewed by marker
8. Configuration – bibliography checking
9. Configuration – proper quotations
10. Configuration – minimum number of words
11. Configuration – resubmission of assignments by date/time

Risk mitigation must be considered. There is always the potential for parts of, or the entire system, to fail. Any potential for failure is included in the risk mitigation chart below, along with its potential likelihood and impact, and a strategy for preventing the risk from occurring.



### Acceptance Test Derivation

The following is a list of tests which will be conducted during the acceptance test for this system. They outline typical scenarios which the user will encounter, along with their expected result.

|  |  |
| --- | --- |
| **Scenario** | **Expected Outcome** |
| S1.1: student selects to upload a file | File is uploaded - student is navigated to a page with a confirmation and submission receipt |
| S1.2: student selects to view the plagiarism analysis | Student is shown the full document they submitted alongside highlights and full analysis of where plagiarism is detected |
| S1.3: marker checks student's submission | Marker is able to view student submission, including name, date, time, and full analysis (same analysis as student could view) |
| S2.1: optional bibliography checking is selected | Plagiarism analysis includes the bibliography in the document |
| S2.2: 'proper quotation excluded' selected | Words within proper quotations aren't analysed in the plagiarism check |
| S2.3: minimum number of matching words set to 5 | Plagiarised statements less than five words aren't marked as plagiarism |
| S2.4: resubmissions set to three days after initial due date | Student is able to resubmit up to the specified date and time |

## Detailed Black-Box Test Plan

This detailed black-box test plan will be testing the *Microsoft Word 2016* print dialog/screen. All widgets (interactive elements) of this dialog will be tested using a mixture of typical and atypical inputs in an effort to discover any bugs in the software.

|  |  |  |
| --- | --- | --- |
| ***Screen: Microsoft Word 2016 Print Dialog*** | | |
| **Widget** | **Tests** | **Expected Result** |
| "Copies" Input Field | 1. Number of copies set to 0 | 1. Cannot print 0 copies, set number of copies to 1 |
|  | 2. Number of copies set to 99999 | 2. Cannot set number of copies to more than 32767, set copies to 9999 instead |
|  | 3. Number of copies set to 12.5 | 3. Prevent input as the decimal point is a symbol, display error |
|  | 4. Number of copies set to "abc" | 4. Prevent input as the letters are non-numeric, display error |
| "Printer" Selection | 1. Change default printer while dialog is open | 1. The green tick will move from the previous default printer to the new one |
|  | 2. Select the currently selected printer again from the drop-down | 2. Program doesn't attempt to re-connect to the printer, instead just closes the drop-down menu |
|  | 3. Hover over the Printer drop-down menu | 3. Display pop-up beneath the cursor with printer details |
| "Print All Pages" Selection | 1. Begin typing into "Pages" textbox | 1. Change "Print All Pages" to "Custom Print" |
|  | 2. Type "abc" into "Pages" textbox | 2. Display error dialog box indicating invalid print range |
|  | 3. Type "1.5 - 4.5" into "Pages" textbox | 3. Display error dialog box indicating invalid print range |
|  | 4. Select "Only Print Odd Pages" | 4. Display a tick next to this option |
|  | 5. Then select "Only Print Even Pages" | 5. Remove the tick next to "Only Print Odd Pages", display it next to "Only Print Even Pages" |
|  | 6. Print pages 12 - 24 in a document with 5 pages | 6. Display error dialog box indicating invalid print range |
| "Print One Sided" Selection | 1. Select to print a single page (from "Print All Pages" selection), then try to select "Manually Print on Both Sides" | 1. Grey out "Manually Print on Both Sides" selection |
|  | 2. Then select "Print Current Page" from "Print All Pages" selection | 2. Automatically change back to "Print One Sided" |
| "Collated" Selection | 1. Select "Uncollated" | 1. Collated option changed to "Uncollated" |
|  | 2. Then select "Collated" | 2. Collated option changed back to "Collated" |
| "Orientation" Selection | 1. In an empty default document, select "Landscape" | 1. Orientation option is changed to "Landscape" |
|  | 2. Then change the orientation selection back to "Portrait" | 2. Orientation option is changed back to "Portrait" |
| "Page Size" Selection | 1. Change from "A4" to "A3" | 1. Change page size option to "A3" as well as the page preview |
|  | 2. Change back to "A4" from "A3" | 2. Change page size option to "A4" as well as the page preview |
| "Margins" Selection | 1. Select "Narrow Margins" from the drop-down menu (with "Normal" by default) | 1. Change margin from "Normal Margins" to "Narrow Margins", as well as changing the page preview |
|  | 2. Select "Custom Margins" | 2. Open the "Custom Margins" window |
|  | 3. Select "Normal Margins" again | 3. Change margin back to "Normal Margins" from "Narrow Margins", as well as changing the page preview |
| "Page Per Sheet" Selection | 1. Select "4 Pages Per Sheet" | 1. Page Per Sheet option changed to "4 Pages Per Sheet" |
|  | 2. Then select "Scale to Paper Size" > "A4" | 2. Same output as 1, except displaying scaling information ("Scaled to 21 cm x 29.7cm") |

# Part 2 – Configuration Management

## Code/File Version Management

The Github account used for this project is [kpc11](https://github.com/kpc11). Instructions on how to pull/merge a repository have been included in the pull request for the CMP73010-Ass1-2017 repository.

## Build Management

The Microsoft nightly build system is, as the name implies, is an automated method for testing software which is run every night. Microsoft began adopting this type of testing back in October 2004 during the development of Windows Vista, which had a “continuous integration” system being developed for it by Amitabh Srivastava and his team. Using the slowly refining CI system, Microsoft was able to internally create new builds once every few days (Guth 2005). This type of testing is now applied to other software which runs on Microsoft Windows, such as Microsoft’s *Visual F# Tools*. In order for a build to be pushed out as a *nightly* build, it has to run through Microsoft’s CI system successfully (Visual FSharp Team 2017) at a pre-determined time each day. The nightly build process typically follows these steps:

1. At the same time each day, the latest build of source code is downloaded from a defined source code repository
2. The source code is compiled
3. Required binary files (i.e. Windows installer files and scripts) are packed up, ready for deployment
4. The project is deployed to a test server
5. A full set of BVTs, or “build verification tests”, are run on the test server
6. Finally, an automated, detailed build report is published to members of the project team (Microsoft n.d)

For Microsoft and other systems which employ nightly builds as a main method of testing, there are a fair number of advantages and disadvantages. The main advantage is how it lowers integration risk. McConnell (1996, p. 144) describes integration risk as being ‘when different team members combine … code they have been working on separately, [and] the resulting code does not work well.’ Since automated, complete tests of the software and its components are run every day/night, conflicts or duplication in code can be quickly spotted, automatically reported and resolved appropriately. Another advantage of nightly builds is that, provided an archive is kept of every nightly build, searches can be made through this archive to determine where bugs first appeared or rollbacks can be made with minimal differences in the overall code (Spolsky 2001).

The primary disadvantage to nightly builds is its limited use when applied to smaller projects. Holding back testing for a smaller project to once per night will result in a lot of wasted time and is ultimately pointless. With smaller projects, builds can be compiled and tested in intervals of once per hour, or even shorter depending on the project’s current progress. Typical “continuous integration”, or automatic testing, can be manually run every time something new is added, meaning it would be a much better fit for smaller projects. Another disadvantage applies to projects with many, many smaller components which make up its whole. For example, testing every single component of the Windows 10 OS may not be possible before the following day, so some components have to be omitted.

As for when a release is distributed to the public, Microsoft very recently updated its terminology for Windows 10 feature updates. As of version 1703 of Windows 10 (released in July 2017), feature updates will be rolled out every March and September. This is known as the “Semi-Annual Channel” service option, replacing the previous “Current Branch” name (Savill 2017). According to Microsoft (2017), there is an ‘18-month servicing timeline… for each release.’

# Part 3 – Request for Proposal (RFP)

## Quick Facts about the RFP

* Name of business is *Kool Dudes Mobile*
* Provides new mobile devices, device repairs and mobile device accessories
* Requesting an integrated system to support 5 branch shops
* System is planned to evolve over time, wanting to expand to more locations in future
* Fairly confident in project, business has been established for a few years
* Website for business already exists and receives a good amount of traffic
* Response deadline is lenient – in no rush to begin, deadline is 30 days after sending RFP

## Organisation Background

Our organisation, *Kool Dudes Mobile*, provides customers with brand new mobile devices, complete repairs of any broken device, as well as a wide array of mobile device accessories. We cater to both mobile phone and tablets of many different brands including Apple, Samsung, Huawei, Oppo, Nokia, LG and more. *Kool Dudes Mobile* has been around for almost three years now, and has expanded from a single, small storefront location to five located all across the country. We’ve recently obtained certification from Apple to both sell devices and accessories, as well as repair Apple mobile devices. This is predicted to increase our profits by a significant amount due to the new market share we can accommodate for.

## Contact Information

You can contact us via our business email, phone or Facebook page. If you decide to call, our opening hours are 9am to 5pm, Monday to Friday. Our email address is: [kooldudesmobile@gmail.com](mailto:kooldudesmobile@gmail.com). Our phone number is: 55672248, and our Facebook page is named *Kool Dudes Mobile* *Aus*. These methods of contact can be used as a way of asking any questions that this RFP doesn’t clearly answer, or discussing any other matters regarding the system we wish to have developed.

## Criteria for Making Our Decision (How We Evaluate Proposals)

In order for a proposal to be chosen, it must contain the following important aspects:

* Ability for the system to support at least five branch shops simultaneously
* Flexibility for the system to be modified and expanded upon over time (i.e. not a static, locked down system)
* System includes marketing system/s which incorporate email and social media (i.e. electronic newsletters)
* The proposal itself must contain clear information on how to contact you or your company. If it contains phone numbers, please let us know your opening hours or best time to call
* The proposal must include either an exact or estimate budget and timeframe for how long the system will take to complete, and when it’s due to be completed
* The proposal must take into account the necessary features that have been detailed below. If any of the features are missing, the proposal will not be taken into consideration

## System Description

The system we wish to have developed must follow some pre-determined guidelines, but apart from that, full creative freedom is granted. The system will have a **database** which will contain comprehensive information on each customer device at each branch. This will include information such as customer details, a report of the problem with the device etc. Alongside this, we wish to start a fortnightly **newsletter** for customers who have signed up to our email service (or have had a device repaired with us) informing them on any sales, new branches opening up, or other general news about the business. This newsletter will also be published to our website and distributed via social media (we already have a Facebook page and Twitter account set up, with a fair number of followers).

An important requirement for this project is the **stock management system** which our employees and managers will regularly refer to. This management system should incorporate products that we have for sale, the parts we use for repairs and automated ordering systems for when stocks are low. It’s necessary that this system has data specific to each branch, rather than the business as a whole, so employees and managers of each branch know when new stock has to be ordered in, for example. Access should be given for this stock management system via the use of a straightforward UI/frontend, which also gives options to generate reports for management use.

# Appendices

## Appendix A – References for Build Management

Guth, R.A. 2005, *Battling Google, Microsoft Changes How It Builds Software*, The Wall Street Journal, viewed 23rd August 2017, <http://faculty.salisbury.edu/~xswang/Research/Papers/SERelated/Model/MicrosoftNew.pdf>.

McConnell, S 1996, ‘Daily Build and Smoke Test’, *IEEE Software*, vol. 13, no. 4, pg. 144, viewed 23rd August 2017, <http://stevemcconnell.com/articles/daily-build-and-smoke-test/>.

Microsoft 2017, *Build and Deploy Continuously*, viewed 23rd August 2017, <https://msdn.microsoft.com/en-us/library/ee308011(v=vs.100).aspx>.

Microsoft 2017, *Windows 10 release information*, viewed 23rd August 2017, <https://technet.microsoft.com/en-us/windows/release-info.aspx>.

Savill, J 2017, *What is semi-annual channel for Windows 10*, IT Pro Windows, viewed 23rd August 2017, <http://windowsitpro.com/windows-10/what-semi-annual-channel-windows-10>.

Spolsky, J 2001, ‘Daily Builds Are Your Friend’, *Joel on Software*, viewed 23rd August 2017, <https://www.joelonsoftware.com/2001/01/27/daily-builds-are-your-friend/>.

Visual FSharp Team [MSFT], *Announcing Nightly Releases for the Visual F# Tools*, Microsoft, viewed 23rd August 2017, <https://blogs.msdn.microsoft.com/dotnet/2017/03/14/announcing-nightly-releases-for-the-visual-f-tools/>.

## Appendix B – References for Request for Proposal (RFP)

The guideline used for writing the Request for Proposal is:

Kinard, T 2011, *How to Write an RFP & Manage the Vendor Selection Process*, viewed 23rd August 2017, <http://tonykinard.net/rfp/2011-12_Digital_RFP_How_To_Guide.pdf>.