Software Testing Report

<Sydney Database Analysis>

Thien Thao My Bui – s5273753

Chelzie Castanares – s5259144

Table of Contents

[1.0 Unit Tests 3](#_Toc49779837)

[2.0 Coverage Report 4](#_Toc49779838)

[3.0 Requirements Acceptance Testing 5](#_Toc49779839)

# Unit Tests

Delete the RED text and replace with your own

(In this table you fill out details about what unit tests you have done using the unittest module)

| **No** | **Test Case** | **Expected Results** | **Actual Results** |
| --- | --- | --- | --- |
| **1.0** | **WordCount Functions** |  |  |
| 1.1 | Test a wrong filename | Exception Handled | Exception Handled |
| 1.2 | Test empty input file | Display error message and exit | Display error message and exit |
| **2.0** | **Histogram Functions** |  |  |
| 2.1 | Empty input dictionary | Display error message and exit | Display error message and exit |

Data Storage / Loading Time /

Keyword finding accuracy (some simple word might be mistaken with other words containing the exact strings because using series.str.contains )

Empty input options (if not input any time period -> will the analysis still run or not / what will show when there is no input but click search…etc)

(I think it is best if you can check out the workshop recordings to see if they mention or give examples about it)

# Coverage Report

A description of the coverage of your unit tests, including how you evaluated coverage (function, statement, branch, condition)

# Requirements Acceptance Testing

(You will need to fill out the column on the left with the requirements listed in software design documents and the columns on the right with the results of your own testing)

| **Software  Requirement No** | **Test** | **Implemented (Full /Partial/ None)** | **Test Results (Pass/ Fail)** | **Comments (for partial implementation or failed test results)** |
| --- | --- | --- | --- | --- |
| 1 | Accept multiple file names as arguments from the command line |  |  |  |
| 2 | Display the details of all valid files |  |  |  |
| 3 | Display an appropriate message if a file does not exist or if a file name is invalid |  |  |  |
| 4 | Display a message if an argument is a directory instead of a file |  |  |  |
| 5 | File name can be a simple file name or include the full path of the file with one or more levels |  |  |  |
| 6 | file names must start with an alphabetical character |  |  |  |
| 7 | Valid file name extensions must be 3 or 4 alphabetical characters preceded by a dot) |  |  |  |
| 8 | Directory/level names must start with an alphabetical character to be considered valid |  |  |  |
| 9 | The program should be able to accept as many levels for each file name as the user wants to input. This is limited only by the number of levels allowed in Windows (approximately 120) |  |  |  |