**CT4021 Introduction to Programming Fundamentals**

**Assignment 001**

Technical Report

for the Expense Management Program

**Student Name**: Nathaniel Clarke

**Student Number**: s1906563

**Course**: Computer and Cyber Security

**Submission Date**: Thursday, 16 January 2020

**University of Gloucestershire**

Contents

[Introduction 3](#_Toc29977684)

[Design 4](#_Toc29977685)

[Testing 5](#_Toc29977686)

[Appendix 2 - User Manual 6](#_Toc29977687)

[References 7](#_Toc29977688)

# Introduction

# Design

# Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ref.** | **Scope** | **Expected Results** | **Actual Results** | **Fixes** |
| 001 | Program start   1. Correct input 2. Incorrect input | Main menu brought up, input takes user to corresponding menu, incorrect input just loops to make input again. User can navigate through menus | As expected | N/A |
| 002 | Quit program from any menu   1. ‘q’ input in menu | Database closed, program ended | As expected | N/A |
| 003 | Set monthly income   1. Input source name 2. Input income | Values saved to dB, total income calculated and updated, user alerted/brought back to menu | As expected | N/A |
| 004 | Add new category   1. Input category name | Category saved to dB, user alerted/brought back to menu | As expected | N/A |
| 005 | Add new expense   1. Input expense name 2. Input category 3. Input cost 4. Input date | If category input matches existing category - add expense to dB and calculate then update over/under for expense and category  If category input invalid - user alerted, back to menu | To add over/under Numpy float could not be concatenated as integer | Dataframe value had to be converted to string first |
| 006 | Update income   1. Select source to change 2. Enter new values | If input matches existing - old values replaced with new ones in dB, user alerted  If input invalid – user alerted, back to menu | Error inserting into db | Was trying to use INSERT with WHERE clause instead of UPDATE |
| 007 | Update category   1. Select category to change 2. Input new name | If input matches existing category – update category name in db  If input invalid – user alerted, back to menu | As expected | N/A |
| 008 | Update expenses   1. Select expense to change 2. Input new name, category, cost | If expense and category input match existing – update values  If expense or category input invalid – user alerted, back to menu | Over/under calculation values undefined | Sql execute query had to be above calculation logic |

# Conclusion

To improve the software next time I would:

1. Have better validation on the date inputs, can easily still be incorrect
2. Create functions with different parameters rather than several functions that do slightly different things, more efficient

# References

Check exists

<https://stackoverflow.com/questions/43246384/pythonsqlite3-if-statement-to-see-if-value-is-in-database>

SQLite db

<https://www.sqlitetutorial.net/sqlite-python/creating-database/>

<https://www.sqlitetutorial.net/sqlite-python/?fbclid=IwAR1D-QLVuo6-sUxtf8IaqLqEtTvOKNKsvfdq739s6dWdxcLm5OFvRWuUlp8>

Dictionary

<https://www.w3schools.com/python/python_dictionaries.asp>

Pandas print table

<https://stackoverflow.com/questions/37051516/printing-a-properly-formatted-sqlite-table-in-python>

XlsxWriter

<https://xlsxwriter.readthedocs.io/example_pandas_multiple.html>

<https://stackoverflow.com/questions/32957441/putting-many-python-pandas-dataframes-to-one-excel-worksheet>

# Appendix 1 - User Manual