**A Level computer Science**

Component 3

Expense Tracker

By: Cameron McQuade

For: A. Issa

**Stoke on Trent Sixth Form College**

Table of Contents

[Chapter One: Analysis of the problem 3](#_Toc189906778)

[1.1 Introduction 4](#_Toc189906779)

[1.2 Problem Identification 5](#_Toc189906780)

[1.3 Possible Computational Methods 6](#_Toc189906781)

[1.4 Stakeholders analysis 7](#_Toc189906782)

[1.4.1 who are Stakeholders? 7](#_Toc189906783)

[1.4.2 How they make use of the solution? 7](#_Toc189906784)

[1.4.3 Why the solution is appropriate to them? 7](#_Toc189906785)

[1.4.3 Stakeholders’ involvement (interview and conclusion) 7](#_Toc189906786)

[1.5 Research of solutions for similar problems 9](#_Toc189906787)

[1.6 Hardware and software requirements 11](#_Toc189906788)

[1.7 The requirements of the solution 12](#_Toc189906789)

[1.8 Features of the solution 14](#_Toc189906790)

[1.9 Success Criteria 16](#_Toc189906791)

[1.10 Limitations of the solution 19](#_Toc189906792)

# Chapter One: Analysis of the problem

## Introduction

After hearing concerns from my parents about tracking their business’s expenses, I knew I had to create a perfect solution. They informed me how often expense software crashes, or inputting lots of data can be tedious, or an error can occur – which sometimes removes all their tracking progress. My solution tries to be a lightweight, user-friendly app with an interactive GUI, reducing this problem into a simple task. Furthermore, the information displayed on a typical expense tracker can be limited, or hard to understand. These problems often mean that business owners try other methods, such as storing all expenses in one spreadsheet, which can be hard to navigate or the file is so large, modifications cannot be made without crashes or severe lagging is experienced. My app will include a method to add, remove, edit or download expenses as well as graphs and tables which will allow the user to see yearly or all expenses. There will also be a predictor that uses machine learning to predict how the expenses and expense amount will change in the future. There will be two types of users: default users and admins. Admins will have access to more features, such as a simple yet effective account system which allows admin users to add, remove, edit or view all users. Admins will also be able to use money managing/ budgeting methods which will include a loan calculator, a budgeting calculator and an income graph. The ability to manipulate the database like adding test data or wiping all data is also a feature specific to admin users. My solution will be best suited for my stakeholders, allowing me to create these great features with lots of customizability. This will create the perfect experience for the user.

## 1.2 Problem Identification

Nowadays, software has become overly bloated and lots of memory is used for even the simplest task. This becomes apparent in apps such as trackers. Many bugs and severe lag cause users to opt for better solutions, such as inefficient drag and drop programs such as Microsoft Access or huge spreadsheets – which usually take too much time or too much space in the memory. This quickly becomes unorganised, while the user loses motivation; problems like this can cause the whole business to become disorientated and unregulated. Furthermore, generic applications do not contain specialised features – making the process feel boring.

A tailored piece of software will allow my shareholders to easily track, edit and view their expenses much simpler meaning budgets or targets can be seen and met far easily – compared to overly complicated software, which often reduces performance and improvement is not seen. In my software, lots of data such as expenses and the users will be displayed graphically or in tables – which allows for better visualisation of where the company is economically overall or within the year, month, or day.

My main aim is to create a useful desktop application, where the main features consist of adding, removing, editing and downloading expenses and users, viewing graphs and tables that can display expense or user data, a future expense prediction method, calculators which help the user with budgeting compared to monthly expenses. There will be other features too: changing the style/ theme of the application, adding an FAQ page to help users who are unsure or stuck, adding admin accounts who have permissions to add or delete accounts, as well as calculators which will help the user calculate how much is needed to paid with a loan calculator, specialised graphs where the user can view expenses vs income, and database testing/ manipulating methods. These features will create a unique platform for users to experience a well-rounded, and easy to use expense tracker.

Another feature to be added is a target budget setter, where an admin-level user can enter an amount of money which the company will try to stay under or go over – depending on the settings chosen. My application will automatically calculate, using data extrapolation, if the business is on track to reach this target, or if they are not based off the monthly expenses from the company. This will hopefully see an assistance in money-spending visualisation, allowing the user to feel like they are progressing properly.

Problems and the lack of features in modern software creates businesses to become unaware of their spending habits or if they are over-budget. This becomes a further problem if they require a certain amount of money at the end of each year or quarter. Removing these issues and creating a positive environment will be key in my software – as I want the application to be trouble-free and a smooth experience.

## Possible Computational Methods

My application will use a variety of computational methods due to the type of data, and how it is stored and organised. A method my app could use is decomposition:

Decomposition is the process of breaking down a complex problem or system into smaller parts that are more manageable and easier to understand. I can use decomposition to divide up my project into easy, manageable pieces – which will make it more efficient to create. Furthermore, an example of decomposition within the project will be when expenses are shown on a graph; specific data is searched and retrieved, it is placed into the lists which will be used for the different axis, then the data is displayed visually. This process of modular programming (breaking down the data into separate components) will be more efficient and is perfect use of a computational method.

Another method my app could use is abstraction:

Abstraction is the process of removing unnecessary details, and only focusing on the important, essential details. This will help to declutter the code; maintaining efficiency, ease of use, and reducing pointless features which only result in confusion on the user’s end. This ‘simplifying’ of the code allows my application to be accustomed to an object-oriented approach as each part of the project can be treated as an object which can be modified to efficiently handle memory usage whilst maintaining its lightweight feeling, further showing how it is great for a computational method solution.

Furthermore, a method that will be used is iteration:

Iteration means repeating a block of code for several times or until a condition is met. This is in different scenarios in my app. An example is adding multiple expenses in one go. The code will iterate repeatedly, adding these expenses into the database. Another example is when the expenses are being viewed graphically, as the graph type may change or the time frame (yearly, monthly) may change, so the graph must be changed. This task will take longer if each piece of data is fetched one by one. It is much more efficient to use iteration to take lots of data, which will be displayed graphically much faster.

A method specific to my project is the efficient use of calculations and graphical representations:

The user needs to be able to visualize the data they are inputting, so properly handling expenses and user data will be key to this project. Calculations will need to be executed so the correct data is being displayed, and it is important that the graphs are easy to read whilst also being customizable. Furthermore, it should be of benefit to the user to use the graphs and it is faster and can be more accurate to see rather than manually calculating data. Appropriate graphical representations should increase performance due to its speed, which makes this computational method a good approach.

A final method to be used is efficient storage of data:

Storing lots of data and expenses in one text file or one spreadsheet swiftly becomes unorganised and inefficient to search through. It is much more effective to store this information in a database. Multiple tables can be using in a relational database for a fast, useful way to get the data back. In addition, data can be organised much quicker, while also reducing data redundancy or data repetition. This increase in performance and structure allows business to quickly see expenses. Therefore, this is another computational method which will severely benefit the app.

## 1.4 Stakeholders analysis

### 1.4.1 who are Stakeholders?

My stakeholders will be my parents. They operate a conveyancing company and have lots of clients who heavily rely on the business to help them with taking care of the legal aspects of transferring property ownership. This application can help them store the money spend, used for quotes or accessing deeds to properties. Furthermore, this program could be used to store the money coming into the business, helping them keep track of customers.

My father, director of the firm, will be a great stakeholder for this software, since he can try it out and give his feedback back to me easily. He will test all the features and view if it is of benefit to the business. If it is of use, it may be viable to make the application more suitable to a wider audience, allowing more companies to easily track their expenses and budget goals. There are plenty of computers in the office, so the application can be installed on all the machines – making it able to be accessed by anyone at any time within the company.

My mother, practice founder and director, is the other stakeholder. She said she will take great interest in looking at the statistics visually since it will be of great help. In addition, the fee earners within the company will be able to better track the financial setting of the business. Being able to efficiently track all money in and out of the business could skyrocket their performance, and being able to retrieve and view all the data will save lots of time which would have been spent looking hopelessly through a spreadsheet.

### 1.4.2 How they make use of the solution?

Each person in the business will be made a unique account, in which they can add and remove expenses at their own will. Some users will be admins, which gives them access to more functions. This application will make this experience much faster whilst also being very efficient at viewing this inputted data. Whilst remaining light, this program will offer a variety of features, which each employee can freely use, and make good use of.

### 1.4.3 Why the solution is appropriate to them?

As explained by my father, multiple solutions have been tried and tested previously. From spreadsheets to drag and drop databases, nothing has worked. Unorganised data and huge file sizes cause many errors, leading to mistakes which the business must deal with. ‘A neat, and fast program is what the business needs’ in his words sparked a flare in my mind; I want to put a stop to these problems. Every computer will be effortlessly able to run and operate my solution, allowing for lots of time to be saved which can be used helping clients, whilst remaining unambiguous so it is accessible to all users.

### 1.4.3 Stakeholders’ involvement (interview and conclusion)

I had some questions written down to ask my parents, to help me better understand the problem and allow me to create the best program for them:

Q1. How do you currently track your expenses?

Q2. What are the pros and cons of this software?

Q3. Why do you need to track your expenses?

Q4. How do you want to track your expenses?

Q5. What key features should a great expense tracker include?

Q6. What is the most important feature you would want and why?

Q7. How many people currently track the businesses expenses currently?

I believe these questions are vital in helping me better understand what my stakeholders want from this project, and equally importantly how it will benefit them. Here is a summary of the interview of my stakeholders:

Q1: We’ve use multiple of Microsoft products, from excel to access, which each have their benefits, but nothing quite clicked. There were issues with both apps, and we eventually gave up as too much time was being wasted.

Q2: Microsoft excel is easy to start but hard to master, and problems quickly started when the file grew. It took ages to open, and when it did open it kept freezing or just crashed. Access was similar but took some time to first set up. Also, there was some issues where the data was being changed but not saved and the queries were sometimes returning data which we didn’t want.

Q3: We think it is beneficial to track the purchases we make and the money we receive from customers. It can help grow the business and it shouldn’t be a hard task to store.

Q4: We require a simpler application which can store lots of data without any problems. We want the app to be usable by everyone, since this will help us the most. The best apps we’ve seen include graphs and spreadsheets but always come at a price or a subscription. We don’t want this as if it does not suit us, it feels like a waste of money.

Q5: Key features should be graphs to visualise the expenses which can also be edited, a table of the expenses, a way to add and delete accounts, and a simple way to add and remove the expenses. Other features can also be useful such as budgeting calculators or a currency converter.

Q6: We think the most important feature will be the table of expenses, as it can be seen by all the employees and easy to edit. This will hopefully save time and money in the long term.

Q7: Currently, only my father and the fee earner in the business track the expenses. My parents hope this can change, so the job can be done by anyone. This will be much more efficient as people can be easily trained to use to application and track lots of purchases within minutes.

## 1.5 Research of solutions for similar problems

There is an abundance of expense trackers on the market, available for use but lots have a pay wall to access the best features. Here is a photo of a free expense tracker from Microsoft store called MoneyPoint: A screenshot of a computer

Description automatically generated

Lots of key features are shown here, which I will implement in my own software. However, I feel the main GUI may be too cluttered, or too complex for a new user to deal with. Lots of these features can be added in the ‘settings’ tab. MoneyPoint uses a variety of methods to store data, such as using excel to store purchase in a spreadsheet and makes use of databases to store data. The app has a feature to import ‘test data’, which will help me to improve my own app. Here is a photo of the test data in the ‘Account’ tab:

A screenshot of a computer

Description automatically generated

I showed my stakeholders this app, and they indicated that they liked it since there were a lot of features, but they said that they may not use it as it could take too much time to learn and set up. One way this is seen, is in the ‘Help’ tab, as it only offers short sentences of information to the user: A screenshot of a computer

Description automatically generated

My stakeholders also wanted an option to graphically see the expenses, which isn’t apparent in this application. However, an application which does make use of graphs is ‘Spending Tracker’:

A screenshot of a computer

Description automatically generated

My stakeholders agreed that the graphing system on this software was good, yet there were still some problems. The software had an ‘out of date’ feel to it, while also performing slowly, and having features such as ‘recurring expenses’ locked behind a paywall:

A screenshot of a computer

Description automatically generated

This is of no use to my stakeholders. Furthermore, there was no option to automatically add test data to the tracker, and the help button in settings did not work. There is also no ‘extra options’, such as changing the colour of the UI. These issues swiftly made this program a no-go for my stakeholders. They told me that these apps had some key features, yet most lacked some details or specific elements, which make a great expense tracker.

## 1.6 Hardware and software requirements

Hardware:

* The user will need a computer with the ability to use python. My solution will be lightweight, so no high-end parts will be required for the computer. Furthermore, the graphs displayed will be easily shown using integrated graphics. Storage is not also a problem as the program will not be large in size and a large amount of RAM will not be required due to efficient memory management and calculations will be optimized.
* The user will need some basic input/ output devices, such as a mouse and a keyboard – so data can be inputted into the program. The user will also need a monitor to view the application.

Software

* An operating system which can run python 3.12 applications. Most operating systems can handle this such as: Windows 7 or later, Linux, and MacOS.
* Python 3.12 with some modules installed such as: Tkinter, MySQL, Matplotlib, Pandas, etc.

## 1.7 The requirements of the solution

User Interface:

* A simple but effective UI will be implemented, so it is easily navigated. The buttons and menus near to be clearly laid out and labelled well so the experience is as easy as possible.
* There needs to be an option to change the theme of the application, such as changing from light mode to dark mode – giving the user a tailored experience allowing for more accessibility. This can help users if there are in a dark environment and do not want to strain their eyes.

Functionality:

* Creating accounts:
  + There are two types of users: Admin and Default. When the application is loaded, the database is scanned for accounts and if none are present, the user will be prompted to make an admin account – this makes sure no errors appear since without it, the user will not be able to advance. Only admin users can access the accounts system implemented.
  + Once in the application, admin users can advance into the accounts area to create accounts, suppling the details required.
* Deleting accounts:
  + If an account is no longer needed, admin users will be able to access the user list and delete a user/ multiple users that are no longer required.
* Viewing accounts:
  + Admins will be able to view all users in a table-based structure.
* Editing accounts:
  + Admin users will be able to edit aspects of other accounts, such as the first name, last name, email, phone number, account level, and salary.
* Downloading accounts:
  + When viewing accounts on the table, users will also be given the option to download the users into an excel file, which is useful to track all users.
* Adding expenses:
  + Any user will be able to create expenses. The user will enter the name, price, quantity and type of expense. The date and which user submitted it will be automatically detected and entered along with the rest of the data.
* Removing expenses:
  + Both types of users will be able to remove expenses. However, only admin users can remove anyone’s expenses while default users will only be able to remove their own expenses. A table will be given to the user, and they will be able to select which expenses they want to remove.
* View expenses:
  + Admin users will be able to view all expenses, whilst default users will only be able to view their own expenses. Expenses will be shown in a table.
* Edit expenses:
  + All expenses will be shown in a table, and admin users will be able to edit any expense. Default users will only be able to edit their own expenses. The data and UserID of the expenses will not be changeable, but the name, price, quantity and type will be.
* Download expenses:
  + The database will be scanned for which years have expenses. The user will be able to choose whether the download all expenses or the expenses from a specific year. Admin users can download all expenses, but default users can download only their own expenses.
* View information visually:
  + Both expenses and users can be shown in table form.
  + Only expenses will be able to view graphically as it is continuous data. There will be multiple graphs to be able to view, such as a line graph, pie chart and heatmap.
* Options to view / edit the database:
  + Admins will have access to extra functions which affect the database, such as import data, adding test data or wiping the database.
* Extra features to help calculations:
  + Calculators which help the user with loans, or monthly expenses will be added to benefit the user’s experience.
* The app should not contain errors or have invalid inputs:
  + Random errors negatively effect the user’s experience, so a lot of effort is needed so the program is robust.

## 1.8 Features of the solution

My application will be packed with features, all to benefit the user. The features will be:

Clear and functional UI:

* The interface is arguably the most important feature of an application as it needs to maintain functionality whilst being easily accessible and understandable by the user. It needs to contain all other features without being cluttered. There will be an option menu at the top where the user will be able to change the colour theme of the app, get help if they are stuck at any point, and log out.

Database to store all information needed:

* A good database with multiple tables will be required. It will use MySQL so it is online, meaning multiple users can work with the application at the same time – because if it was a local database, it will have to be re-sent to all users before being changed again. This aims to reduce data redundancy and repletion. There will need to be an expense table, user table and an account level/ salary table. The expense table will hold all the details of each expense that has been submitted, such as the expense id, the user id of the user who submitted it, the name of the expense, the quantity, price per unit and type of the expense, and the date it was submitted. The user table will contain the details of each user which are submitted when making each user. It will contain the first name, last name, email, phone number, hashed password and userID which is automatically generated using the first letters of the first and last name. The salary table will contain the salary the user is being paid and the level the user is – admin or default.

An accounts system:

* The ability to add, remove, view or edit accounts will be very useful. There will also be two types of accounts – default and admin. Admin accounts will have these abilities, whilst default accounts will not. When loading the application for the first time, the user will be prompted to make an account which is set as an admin, to prevent any locks in my application. After submitting the account details, to ensure the email address is valid, a verification code will be sent to that email to verify that account. Once verified, the UserID will then be emailed to that email, and the user will be able to sign in using that ID and the password they created.

An expense system:

* This is the main feature of the tracker. Users will be able to add, remove, edit, export and view expenses using graphs and tables. Default users will only be able to remove, edit, export and view their own expenses but admin users will be able to complete these actions for all expenses. This should avoid any expenses being incorrectly tampered with. The graphs will be a line chart of expenses against time, a pie chart of the type of expenses to see the distribution, and a heatmap of expenses over the year. When exporting data, the user will be given the option of downloading all expenses found in the database or from any specific year expenses are found from. It will be downloaded into an excel file. Furthermore, there will be a feature accessible to all users that predicts how the expense amount will change in the future which uses machine learning based on the current expenses. All these features need to be simple to understand, whilst maintaining a lightweight feel as this is where the user will spend the most time. Any inefficiencies will slow down the user, which is the opposite of the aim.

Benefits to admin users:

* Admin users have priority in my application as they have more responsibilities. As previously stated, they have access to add, remove, edit and view users. Admins will be able to view a graph of all expenses against total income, and a bar chart of all incomes. Admins are also given calculators which benefit the business. They have access to a budgeting calculator which will check if the business is on, below or over budget compared to a figure given based off the current monthly expenses. They will also be given a loan calculator which automatically calculates the amount needed to be paid per month and total cost, based off given parameters. A currency converter will also be given to admins, which uses an API to get current conversion rates, which will be able to convert any currency into any other. In addition, admins are the only users to be able to modify the database from within the program. There will be an option to import expenses or users from an excel file, add test data to the database, view the database structure through an image or wipe the database if needed. When the database is wiped, the current state is downloaded to the user’s computer in the form of an excel file, which will include all user and expense information.

Good input verification:

* Making sure the whole system runs flawlessly in key in the success of this application, so all inputs will need to be checked that they are in the correct format or match the right type of data they should be. Since Python is a dynamically typed language, errors can easily arise if the user inputs data which is of the wrong data type. This can lead to confusing errors, but this can be fixed with data validation and good exception handling, such as using ‘try’ and ‘except’ blocks within the code. Since the code will be written modular, it will be easy to return exceptions without the code crashing. Automatic testing using libraries like unittest will also be used to test cases within my code against a set of parameters, which should be asserted as true or equal if no errors arise. This ensure my code runs fluidly – allowing a better experience to the user.

## 1.9 Success Criteria

|  |  |  |  |
| --- | --- | --- | --- |
| Features | What makes it successful? | Proof of its success | How/ why is it successful? |
| User interface | A good user interface is easy to navigate whilst containing all features. It should be customisable and simple to use. | Screenshots will be provided, showing the user interface working as expected whilst having a clear, concise design. | One of the most key aspects to any application is the user interface. It is the visuals of the program, and any user wants to have the best experience with it. |
| Account Login | A simple login screen boosts programs authenticity, which will be provided in a clear frame. Help buttons also guide the user to ensure it is an easy process to log in. | Screenshots of the program will show the input system, and the help system, showing its usefulness. | Another great aspect to a program. There will be a clear input system, guiding the user to enter their username and password. Buttons will also be provided if the user has forgotten their password, allowing for a better experience with the program. |
| Account system | A clear design for the account creation, deleting, and editing whilst having a secure system to store the accounts. Options to view the accounts will also be available. | Screenshots of the account creation screen and information stored in the database will show how the application deals with information | The account will be stored in a database. The passwords will be hashed using a custom module, ensuring security within the program. Encrypting the passwords is key in an online application as businesses can lose lots of data if account information is compromised. |
| Password hashing system | The user’s password cannot be stored as plaintext as anyone can see into the database and find it. A salt will be added to the password, then it will be hashed to add extra security to the user’s account. | Screenshots of the database will show how the password is being stored, after it has been hashed. | Hashing is a very beneficial process as it ensures the security of accounts. It allows the user to know their account is safe and their information is secure. |
| Entry boxes and buttons | Clear and designated widgets to enter information or proceed to another area of the program allows the user to learn this application faster. | Screenshots of the GUI showing a clear layout with many buttons, and entry boxes in designated areas will portray how the program is made to keep simplicity. | Having a cluttered GUI full of buttons looks unprofessional. Widgets are the main way of navigating through the program, so staying unambiguous is a must. |
| Input validation | Before inputs are used or iterated over, they need to be in the correct format or be the correct data type. A class containing multiple functions will be used to ensure the data inputted by the user follows the rules needed for it. | Screenshots showing the results when using correct and incorrect input data to show how my app will handle incorrect details. | Input validation is a key part to any successful program, as it reduces errors or unexpected crashes. It allows for a smoother experience. |
| Adding, removing and editing expenses | Users should be able to efficiently enter lots of expenses, whilst also having a simple way to edit or remove them if mistakes are made. | Screenshots will show the effective way the user can input lots of data, while maintaining minimalism. | The main goal is keeping this task the easiest it can be, as it is the main task the user will be doing in the application. |
| Graphs showing the expenses | Users can view the expenses in a graph format. This will suit any user, who can change the type of graph or time frame of expenses. | Screenshots showing expenses in different graphical forms with different time frames and the data entered to ensure the graphs are accurate. | This is another primary feature, as it allows users to keep the best track of their money. It needs to be clear and straightforward to navigate or change. Results become visualisable and progress becomes apparent. |
| Tables showing expenses | Users will be able to use tables to view a variety of information. Tables can show both expenses and accounts, whilst also being modifiable to specific time periods. | Screenshots of the tables showing multiple pieces of data. | Tables are a key feature as they are easily readable and accessible, allowing the user to understand the data in another way. |
| Exporting data | Users will be able to export/ download data such as expenses or users. Expenses will be able to be downloaded from specific years, or all years. | Screenshots showing how different data and their time periods are downloaded and stored on the user’s device | Downloading data to a file, such as within an excel format (.xls) is a very important feature as these files are easily sharable and can be viewed on any computer. |
| Settings | Users can access the settings at any point within the application, and it should be easy to change settings. There will be a variety of options, such as changing the theme of the UI, or changing the default graph type or time frame. | Screenshots showing the settings being easily accessible and containing all the settings for the user to have a customisable application. | The settings allow the user to make the application feel like their own. They need to be easily navigable and offer a variety of options. |
| Database | The database needs to be online, and accessible by the application so data can be efficiently retrieved for the user. | Screenshots showing the database, and how data is stored within it. | An online relation database will be essential for this application, so multiple people can access it at one time. It needs to be able to store account information, the account type, and expenses. |

## 1.10 Limitations of the solution

There may be some limitations of my application. For example, the online database may be hard to set up and will require a device to constantly keep it running. Furthermore, it will require an internet connection to connect and make changes to. Most devices are always connected to the internet, yet in some areas this is not always an option. Most online databases also require a subscription due to their fast execution times, but free options may produce slower times to lookup and retrieve data.

Another limitation is the python modules. To keep my application lightweight but keep a professional feel, I have opted to use CustomTkinter, which is a python desktop UI-library based on Tkinter, which provides modern looking and fully customizable widgets. Modules like Qt offer better customisation but relies heavily on APIs. This allows for much greater looking applications at the cost of performance, where a better-quality device is required to run it. In addition, if the application was built as a website application using a module like flask, the customisation would be much larger for the user. however, CustomTkinter is more flexible and easier to implement into a program like mine.