Capstone Functional Requirements
Desktop Financing Application

Danny Le, Aman Sahu, Andrew Miller & Rohan Mankame

Problem Statement :

It's known how important documentation and economics are to companies in the real-world. How this connects to computer science is through software for internal financing. Software tends to be more secure, fast, and available than paper.

Our aim for this app will allow a company to efficiently manage budget, expenses, income, payables/receivables, cash flow, taxes, audits and other document handling matters such as Profit/Loss statements. We will be working with an external database to store files (PDF statements) and other confidential information like login information and user information.

We will use JavaFX as our base application integrated with Firebase as the database platform that can not only store key-value information, but in addition to media, which pertains to the application's use of economic documents. We will also potentially use Python in some of the backend modules.

User Authentication and Authorization:

- Users will be given a secure username and password, this will be used to authenticate the user with the application.
- When a user is prompted to login on application startup, their information will be verified by connecting to the 'userLoginInformation' Database.

Database Implementation:

This section will focus on the main implementations required for this application.

- Database 1 : userLoginInformation: This database will include all the information required for a user to login to the application. This table will have 2 columns:
    - username : auto generated and will be used as a **primary key.**
    - password : autogenerated. Password would be less than 20 characters and include alphanumeric characters
- Database 2: userInformation: This database will contain the user information. Its access will be restricted to admins and have encrypted table values and have 4 columns:
    - username: will be automatically added from database 'userLoginInformation'.
    - Email : contact information
    - Phone Number : contact information
    - Address: Billing Address
- Database 3 : userDocuments: This database will include documents the user has uploaded into the database using the desktop app. This table should use the **FireStore Database**. The table will contain 3 columns:
    - fileID : generated by API.
    - documentType : the type of financial document.
    - fileAddress : this will contain the path to the uploaded file.
- Database 4 : financialData: This database will include the parsed financial document data from the file parser.

Product Functions:

- Accounting Home:
    - The Accounting Home Page will contain the overview of the module of Budget Management. This Budget Management Module will contain the following sub-modules: Expense and Income Tracking, and Cash Flow Management.

    - A module on the dashboard, when clicked, should open up its own respective window.
- Graph Generation:

  Alongside the modules, the main homepage will feature summary graphs of the following financial aspects:
- Profit Calculation
- Monthly Revenue

- Base Module Features:

  This section will cover the basic functionalities of what each of the **modules** will have and what functionalities it should entail.
- Object Creation: This functionality will enable the user to create any type of object with the attributes, depending from module to module.
- CRUD Operations: Each module will be able to implement all of the respective Create, Read, Update, and Delete functionalities.

- Budget Management:
    - Module will help enable users to create, modify, and track various budgets for their business or corporation.
    - Users must be able to create, rename, and delete Reports.
    - Reports must be categorized by Fiscal Quarters.
    - Reports must be able to be duplicated into another quarter.
    - Duplicated Reports must be related to each other by name but not by fiscal quarter.
    - Renaming Duplicate Reports must rename all other related reports.
    - Removing Duplicated Reports must prompt the user to delete multiple similar reports or to delete the selected report.
    - Enable users to request a report summary for one or multiple reports by applying certain filters based on fiscal quarters and report names.
    - Reports encompass all of below:
        - Expense and Income Tracking:
            - Users must be able to view expected expenses and income through a table format
            - Users must be allowed to create, edit, and remove expected expenses

        - Users must be allowed to create, edit, and remove expected income
        - Users must be able to create, edit, and remove nameable groups
        - Existing expenses and incomes must be addable and removable from groups
        - The table must sum up the expenses and income of the entire report as well as at each group
        - Income tracking will include the complete summary of the monthly revenue of the company.
        - Profit Calculation will be implemented using Expense and Income Tracking.
    - Cash Flow Management:
        - This module will not be shown as an interactive module on the dashboard, it should be a backend module.
        - This module will be used to provide the real time cash flow of a business, and be shown on the Dashboard. This feature will be available based on if the business supports real time cash flow dependencies.

Document Management:

This section will cover the implementation of document management. This section is important as most financial data will be imported and extracted from various files.

Financial Document Handling:

- Users will have the ability to upload and store financial documents of given file types into the Database 2 : ‘userDocuments’.

File Parsing:

- Program will be able to generate and read Balance Sheets, Income Statements, and Cash Flow Statements of all types.
- File types will primarily be limited to .pdf, .xlsx, .json.
- After extracting all the data, the program should place the extracted data into the database of ‘financialData’. This will include data sorting, meaning all the respective data will go under the respective columns in the database.

Speech & Artificial Intelligence:

This section will cover the implementation of speech detection and AI communication between the user and our program.

Speech Detection:

- Users will have the ability to interact with our program through their voice. Users may want to say “sold another 2 units of product x” this will automatically interact with our AI to input the sale into database without direct input of the user through text.
- There will be the ability to undo the command processed through voice incase any errors were made in speech dedication.
- The Speech detection will be performed by sphinx 4 API.

Artificial Intelligence:

- Program will have AI interaction. After processing the voice of the user, data will be fed to ai to generate appropriate responses to the user. For example, a user could ask for help finding a specific page, which the ai will direct the user’s trough the correct screens.
- Currently we have not decided on a API to use as our research is in progress.