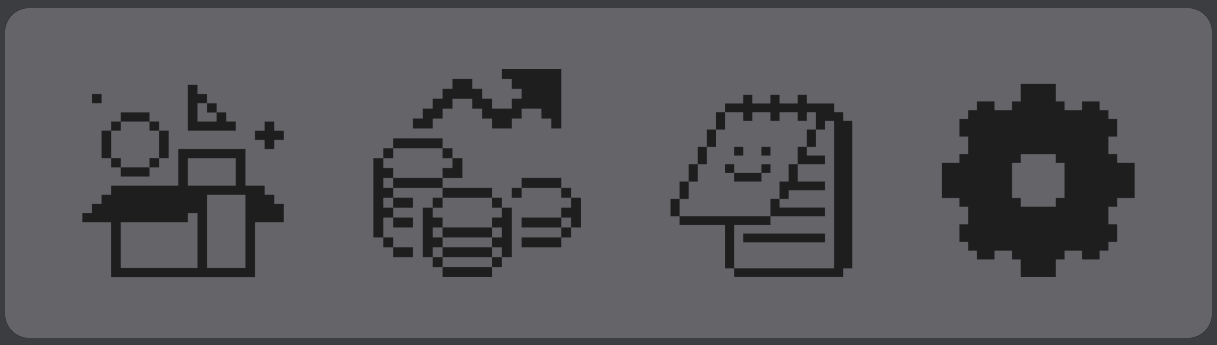
The Manager

ENSE600

Megan Norman | 22164447
Corin Finnin | 21151284



Assignment 2 Report

ENSE600

Contents

1. Introduction

Welcome! Our program "The ____ Manager" is designed for individuals to keep track of items, track spending, as well as natively create and export a shopping list.

2. Project Setup

Once the program has been launched, a home screen should appear with two options:

Option 1: Login

To see a "populated" inventory, budget and shopping list, login in with the following information:

```
Username: admin
Password: admin
```

Option 2: Create New Account

Create a new account with demonstration items preloaded and configure your own user settings.

Note: The data associated with the demonstration account was generated using ChatGPT.
Additional information relating to the use of ChatGPT in this assignment can be found in the Appendices.

3. Individual Contribution

Throughout this project, individual responsibilities for feature implementation were clearly defined, however collaboration was the strategic advantage that allowed for a deeply pleasant development process. Our team members had strengths that precisely offset each other's weaknesses, which was critical for problem solving and bug fixing.

3.1. Group Member 1 (Corin)

This group member was predominantly responsible for the **database components**.

- Core Classes:
  - `InventoryManager.java`
    - Acts as the main link between the database and user interface.
    - Handles loading, saving, adding, and editing inventory records.
  - `Item.java`
    - Defines the structure of individual inventory items with core fields and access methods.
  - `PurchaseLog.java`
    - Records purchase history, tracking price, quantity, and date for each item.
  - `Transaction.java`
    - Records Transaction history, tracking amount, frequency, date, income or expense for each item.
  - `SettingsManager.java`
    - Manages saving and loading of user-specific settings between sessions.
  - `BudgetManager.java`
    - Calculates and loads transaction, income and expense data for the UI using database values.
  - `DatabaseUtils.java`
    - Provides helper functions for database setup, connection handling, and table creation.
- Responsibilities:
  - Designed and maintained the database structure.
  - Collaborated with Group Member 2 to integrate backend data with the front-end interface.

3.2. Group Member 2 (Megan)

This group member was predominantly responsible for the **User Interface (UI)**.

- Template Classes:
  - `BaseScreenPanel.java` and `BaseThemedDialog.java`
    - Define shared layouts and consistent styling for all screens and dialogs.
- Core UI:
  - `Theme.java` - Creates a unified colour scheme and design for the GUI.
  - `HomeScreen.java` - Acts as the main frame and common point for all Panels.
- Panels:
  - `DashboardPanel.java`, `InventoryPanel.java`, `BudgetPanel.java`, `SavingsPanel.java`, `SpendingPanel.java`, `ShoppingListPanel.java`, and `SettingsPanel.java`
    - Each extends `BaseScreenPanel` and implements unique content.
    - The `BudgetPanel` includes budget quadrants for visualising spending categories.
- Custom UI Elements:
  - `AccentHeaderBar.java`, `TileButton.java`, and `ThemedProgressBar.java`
    - Add visual consistency and improved user interaction.
- Responsibilities:
  - Designed a user friendly interface with consistent theming.
  - Collaborated with Group Member 1 to link live data from the backend.

3.3. Contribution Agreement

Student	Contribution Percentage	Signature
Corin Finnin	50%	
Megan Norman	50%	

4. Appendices

4.1. Git

https://github.com/Zalrath/SoftwareConstructionAssignment2

4.2. Unit Tests

The project has two sets of Unit tests, which test the important functions of the Database and the Login Screen.
1. Duplication of accounts, Authentication Success, incorrect password, and User not found.
2. Database creation, connection confirmation and Disconnection.

4.3. Generative AI

During this assignment, our group used AI for two things.
1. Generating large realistic datasets to populate tables which simulate our specific use cases.
2. Locating particularly devilish bugs efficiently, typically involving the database implementation. Our solutions were informed by AI but were adapted, not copy-pasted.

Our datasets have been marked as generated by AI with the following commented convention:
`// ----- GEN AI ----- //`

Our `DatabaseUtils.java` class contains our generated datasets, alongside authentic human functions.

We chose to generate these datasets as it was the most efficient option. As the validity and accuracy of the data was not overly important, we were able to avoid manual data entry as well as sourcing the data which gave us more time to work on the implementation and program design.

This is the full extent of the AI use throughout this assignment.