亞洲大學
ASIA UNIVERSITY

Final Project Report
Advanced Computer Programming

Expense Tracker

Group : 11

Instructor : DINH-TRUNG VU

2025-06-06

Chapter 1 Introduction

1.1 Group Information

1) **Group Project Repository**:
   https://github.com/113021197/ACP-Group11/tree/main
2) **Group Members:**
   1. Tsogbat Bat-Erdene 1– 113021197 (leader)
   2. Khuslen Gantumur 2– 113021189

1.2 Overview

In this project, the following advanced Python features and libraries were used:

- Data classes and modularization: The project modularizes database functions and logic across files for cleaner code.
- tkinter, ttk, and tkcalendar: Used for building a user-friendly GUI with enhanced widgets like date selectors.
- matplotlib: Used for visualizing expenses through pie charts, bar charts, and waterfall charts.
- csv module: Enables export of transaction data to CSV files.
- SqlLite3

Our project, "Expense Tracker", allows users to input and categorize income and expenses, filter data by month/year/category, and view visual summaries. It offers multiple types of data visualization including pie charts, bar charts, and waterfall plots. It also supports exporting data to CSV.

Implementation

1.1 SafeDateEntry

Description: A subclass of DateEntry from tkcalendar to avoid widget errors during focus changes.

1.1.1 Fields

Inherits from DateEntry.

1.1.2 Methods

_on_focus_out_cal: Overridden to safely handle widget focus events without raising exceptions.

1.2 ExpenseTracker

Description: The main class that initializes the UI, handles all business logic, and interacts with the database and plotting functions.

1.2.1 Fields

- root: Main window object.
- tree: Treeview to display transactions.
- categories: Predefined list of income/expense categories.
- style: Used to toggle between GUI themes.
- Various UI components like Combobox, Label, Entry, Button.

1.2.2 Methods

- __init__: Initializes the UI and prompts for password login.
- login_screen: Handles password authentication using simpledialog.
- setup_ui: Constructs the GUI with widgets for input, filtering, and plotting.
- add_transaction: Adds a new transaction after validating inputs.
- refresh_table: Refreshes the treeview with current transactions and updates the summary.
- apply_filter: Applies filtering by month/year/category.
- remove_transaction: Deletes selected transactions from the database.
- show_bar_chart: Displays a bar chart of expenses by category.
- show_progress_bars: Shows category-wise budget usage using progress bars.
- show_waterfall_chart: Displays a waterfall chart for income and expenses.
- show_pie_chart: Displays a pie chart of expense distribution.
- show_monthly_chart: Compares income and expense trends by month.
- export_to_csv: Allows exporting transactions to a CSV file.
- toggle_theme: Switches between dark/light themes.
- on_type_change: Auto-sets category depending on selected transaction type.

1.2.3 Functions

- connect_db(): Initializes and connects to the SQLite database.
- add_transaction(date, category, amount, type_): Inserts a transaction.
- get_all_transactions(month=None, year=None, category=None): Retrieves filtered transactions.
- get_summary(): Returns total income and expense.
- delete_transaction(transaction_id): Deletes a transaction by ID.
- get_monthly_summary(): Aggregates monthly income and expenses for visualization.

Chapter 2 Results

1.1 Functional Expense Tracker App

- Allows users to add, remove, and filter transactions.
- Offers category-based filtering and real-time summary updates.

1.2 Data Visualization

- Pie Chart: Shows percentage-wise category expense.
- Bar Chart: Displays actual amount spent per category.
- Waterfall Chart: Visual representation of income vs expenses.
- Monthly Chart: Side-by-side comparison of monthly incomes and expenses.
- Progress Bars: Shows budget consumption per category.

.

Chapter 3 Conclusions

This project demonstrates how GUI applications can be built using tkinter combined with powerful libraries like matplotlib for visualization. Modular design, good UI/UX practices, and appropriate data handling make it a robust personal finance tool. The project showcases the integration of database interaction, exception handling, and dynamic plotting to provide meaningful insights to users about their financial behavior