Personal Finance Tracker

An interactive tool for personal finance management

University Of Illinois Chicago

IDS 401 Final Project

Professor Yingda Lu, TA Deepali Baswa

Group 5:

Neelansh Singh Rathore - nrath@uic.edu

Omkar Nehete - onehet2@uic.edu

Supriya Narendra - snaren4@uic.edu

Tanishq Padwal - tpadwa2@uic.edu

Utsav Virat Shukla - ushukl2@uic.edu

1. Introduction:

1.1 Overview:

Effortless Finance Management simplifies personal finance through easy tracking, intuitive design, and automated budgeting. It streamlines daily transactions, offers user-friendly interfaces for quick data entry, sets personalised budgets with alerts, and visualises complex financial data for better decision-making. Taking ownership of our financial planning and management and putting it into action, putting theory into practice is critical for everyone. This is not merely for the purpose of establishing our household budget, but also to save, invest, and plan for retirement. The project aims to empower users, especially students, in managing finances effortlessly, analysing spending patterns, and retrieving data within their preferred time frame.

The aim of this project is to create a finance tracker as a tool that helps people manage their finances by keeping track of what they spend. Users can set budgets for each area in the tracker, which frequently includes predefined categories such as shopping, transportation, and entertainment. When a user makes a purchase, they enter the amount and category into the tracker. By examining the tracker on a regular basis, users may analyze their spending, assess how it compares to their budget, and make necessary adjustments to help them meet their financial goals. In general, a finance tracker is a great tool for anyone attempting to take control of their finances and make prudent financial choices.

1.2 Key Features:

The key features of the project include Transaction Logging, which simplifies recording and categorising financial transactions for oversight; Budget Management, allowing easy setup and tracking of personalised spending limits; Visual Analytics, presenting graphical charts to visualise spending patterns and trends; Export Transactions, facilitating the export of transaction data for analysis; and an Intuitive GUI, providing a user-friendly interface with real-time financial updates.

Languages: This application is primarily developed in Java. The program makes use of SQLite to retrieve data from a SQL database.

Libraries: Several libraries were used during the creation of this application:

- Java Swing
- JFreeChart
- SQLite JDBC Driver
- Java Standard Library

1.3 Premise:

The application has 5 options for the user to choose from namely: Manage transactions, Manage Budgets, View Transactions, Export Transactions, View Charts. The first option of managing transactions allows the user to capture the details of financial transactions, including amount, date, description, and category. The option of manage budgets allows the user to manage financial limits across categories, enabling users to set and track spending limits. The option of view chart lets the user to visualize financial data with interactive charts for enhanced insights.

1.4 Modular Design and Key Components

● Modular Design: Enhances scalability with each class addressing specific application aspects.

● FinanceTrackerGUI: Main user interface for managing transactions and budgets.

● DatabaseHandler: Manages database operations, ensuring data accuracy and reliability.

● Account: Represents user accounts, overseeing transactions and budget management.

● Transaction: Captures key details of financial activities for tracking and analysis.

● Budget: Handles budget setting and monitors spending by category.

● PieChart: Visualizes financial data with interactive charts for enhanced insights.

2. Functional Overview:

2.1 Transaction Management:

The Transaction Management functionality serves as the backbone for capturing the intricate details of financial activities. Each transaction allows the user to meticulously document, encompassing vital information such as the monetary value(amount), transaction date, a descriptive account(description), and the specific category of expenditure (eg. Groceries, utilities).

Key Attributes:

- amount: Monetory value of the transaction.
- date: Transaction date (using java.util.Date).
- description: Textual description of the transaction.
- category: Category of expenditure (eg: groceries, utilities).

Adding Transactions:

Within this framework, the Adding Transactions process engages users in inputting transaction details, initiating the creation of a dedicated Transaction object. This dynamic approach ensures that every financial interaction is systematically recorded and stored. The

integration of the **addTransaction** method within the *Account class* facilitates the seamless logging of each transaction. Consequently, the Impact on Financial Data is substantial; every transaction becomes a building block, influencing the overall financial data and propelling the budget tracking mechanism.

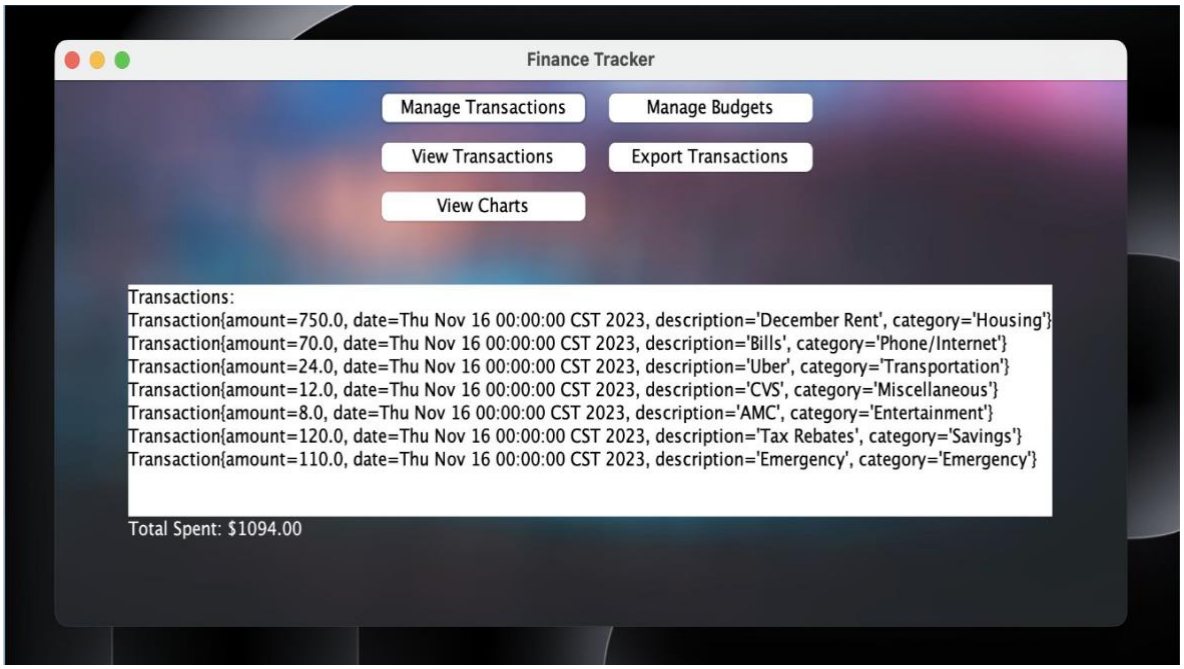

Fig 1. GUI for the View Transaction Tab

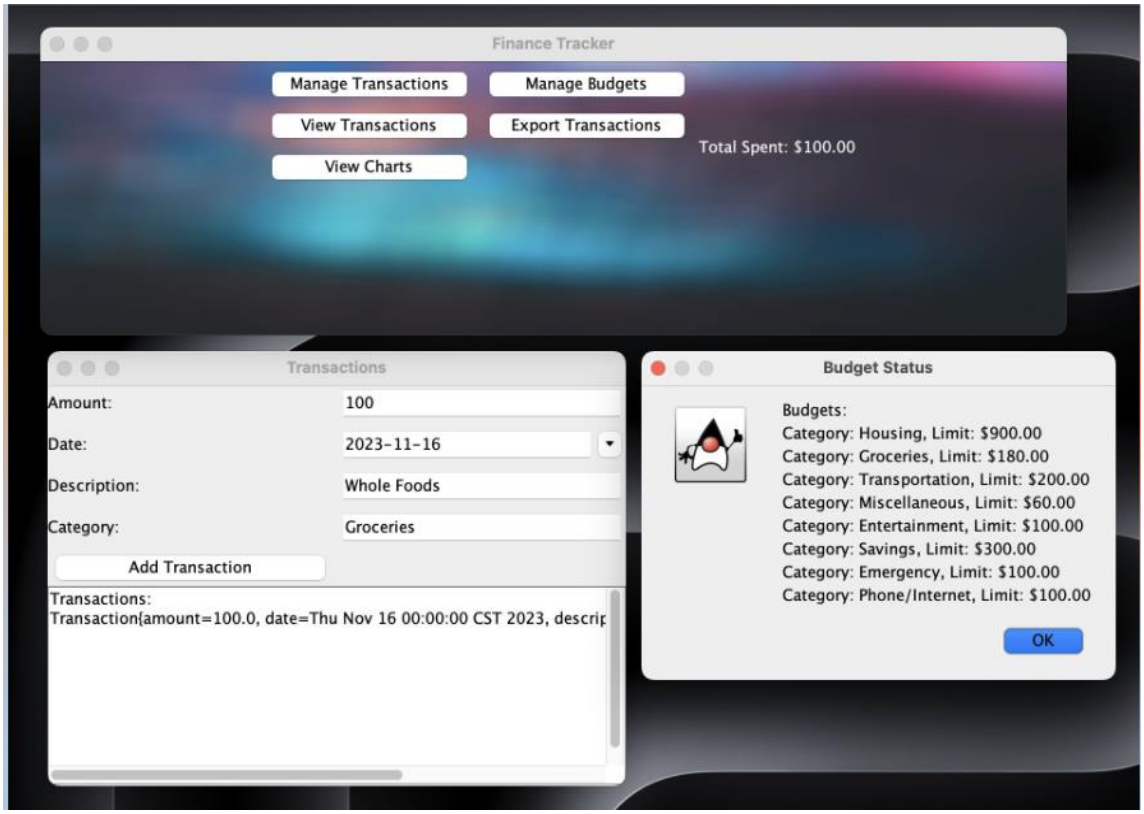

Fig 2. GUI for adding the transaction tab.

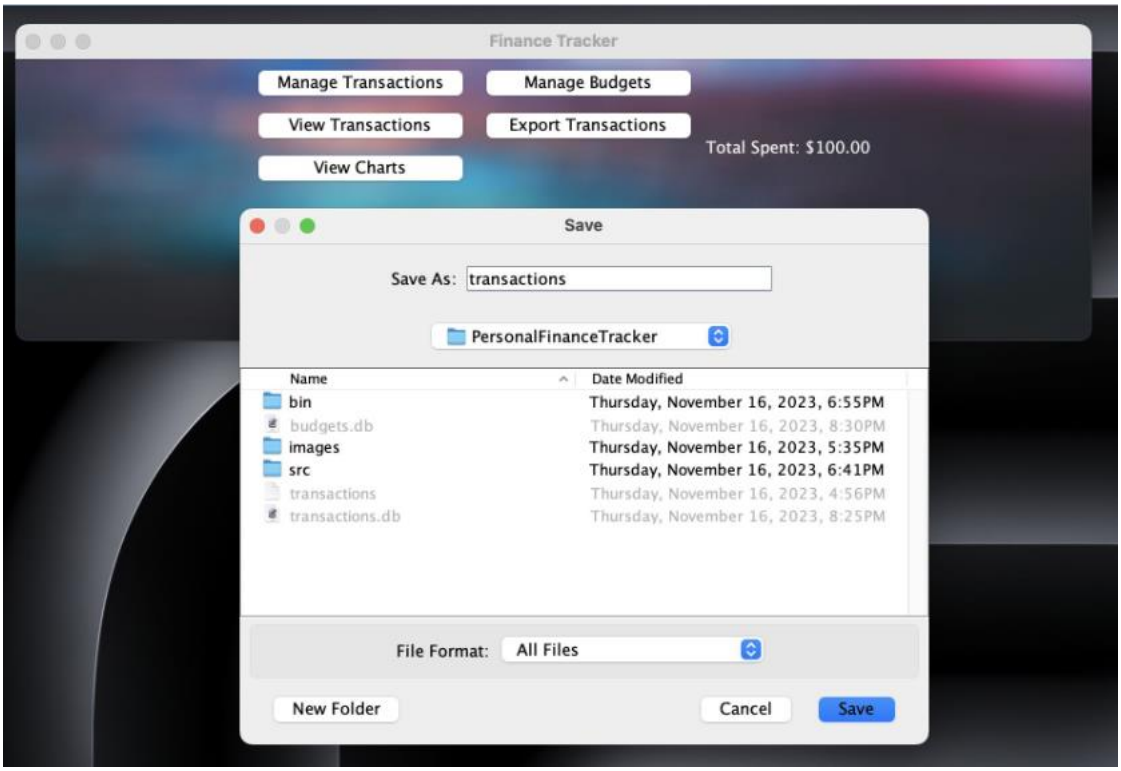

Fig 3. Exporting transactions 1

Fig 4. Exporting transactions 2

2.2 Budget Management:

The Budget Management section plays a pivotal role in the Personal Finance Tracker, ensuring users maintain financial discipline by effectively managing their spending across various categories. The *Budget Class* serves as the backbone of this functionality, orchestrating the process of setting, tracking, and monitoring spending limits. By categorizing expenditures into areas such as groceries or entertainment, users can define maximum spending limits, providing a structured approach to their financial management.

2.2.1 Budget Creation: Users initiate the budgeting process by setting personalized limits for specific categories. This step allows for a tailored financial plan, aligning with individual priorities and financial goals.

2.2.3 Budget Monitoring: Once budgets are established, The Personal Finance Tracker diligently monitors and updates spending against these predefined limits with each new

transaction. This real-time tracking mechanism ensures that users have a clear and immediate understanding of their financial standing within each category.

Key Attributes:

- **category:** Defines the budget area.
- **limit**: Sets the maximum allowable spending.
- **spent**: Tracks current expenditures in the category.

2.2.4 Alerts for Overspending: The **isOverLimit** function is a crucial element of this feature, promptly notifying users if their spending surpasses the predefined budget. This real-time alerting mechanism empowers users to make informed financial decisions, fostering responsible spending habits.

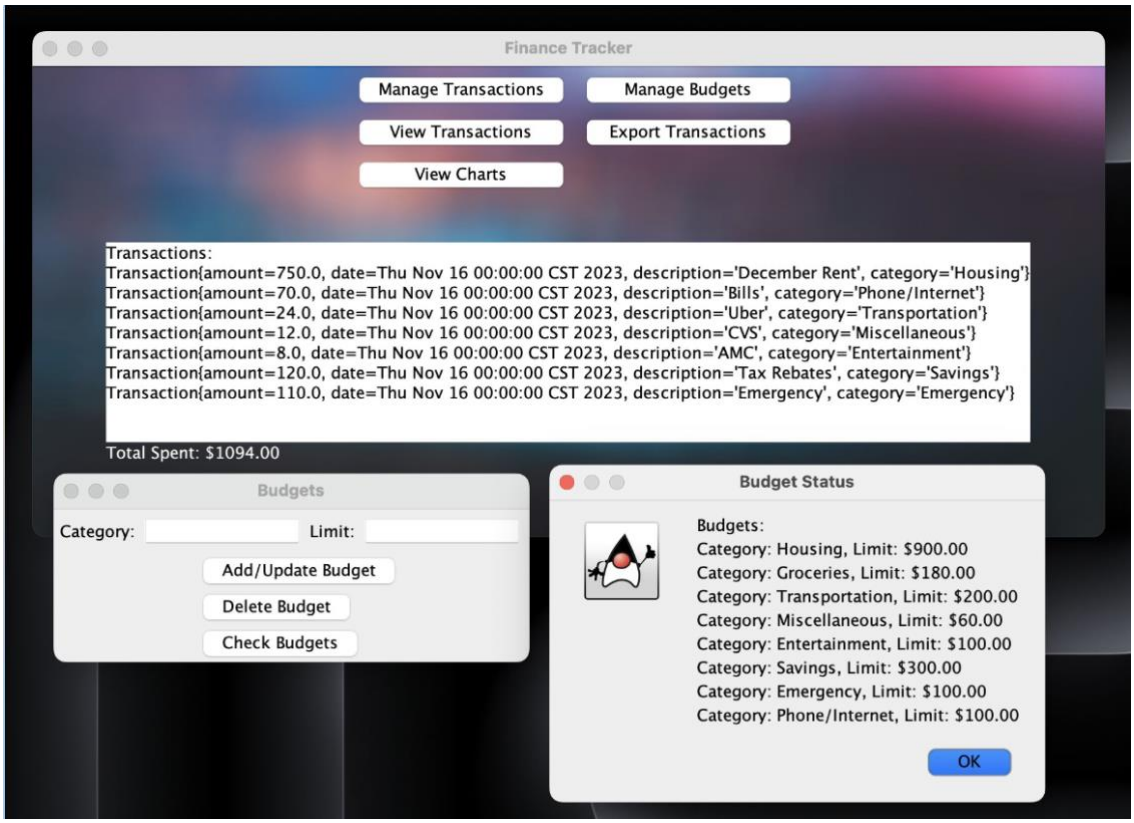

Fig 5. GUI representation of Budget Management Tabs

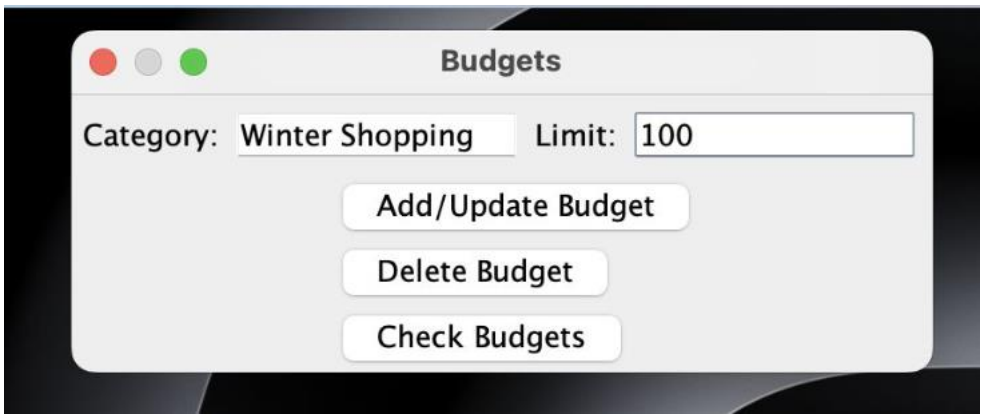

Fig 6. Budget Management

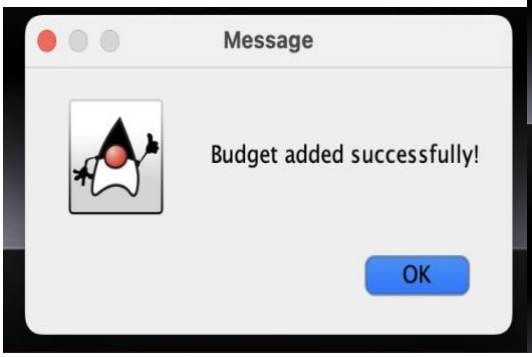

Fig 7. Budget Management

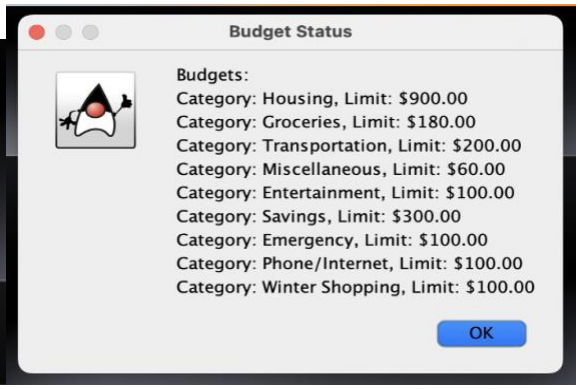

Fig 8. Budget Management

2.3. Visualizing Finances with PieChart:

The *PieChart* Class collaborates seamlessly with the *DatabaseHandler* to retrieve essential financial data. Visualizes financial data with pie and bar charts, offering a clear view of one's financial health. Leveraging the capabilities of *JFreeChart*, it transforms raw data into engaging visualizations. These visualizations serve as dynamic tools, displaying budget allocations and expense categories in an interactive manner.

- **Key Visualizations:**
  - **Budget Distribution:** Visualizes how the budget is spread across different categories.
  - **Expense Analysis:** Highlights spending in various categories to pinpoint major expenses.
- **Benefits of Visual Data:**
  - Simplifies complex financial data into understandable charts.
  - Aids in spotting spending patterns for smarter budgeting.
  - Provides immediate insights for effective financial management.

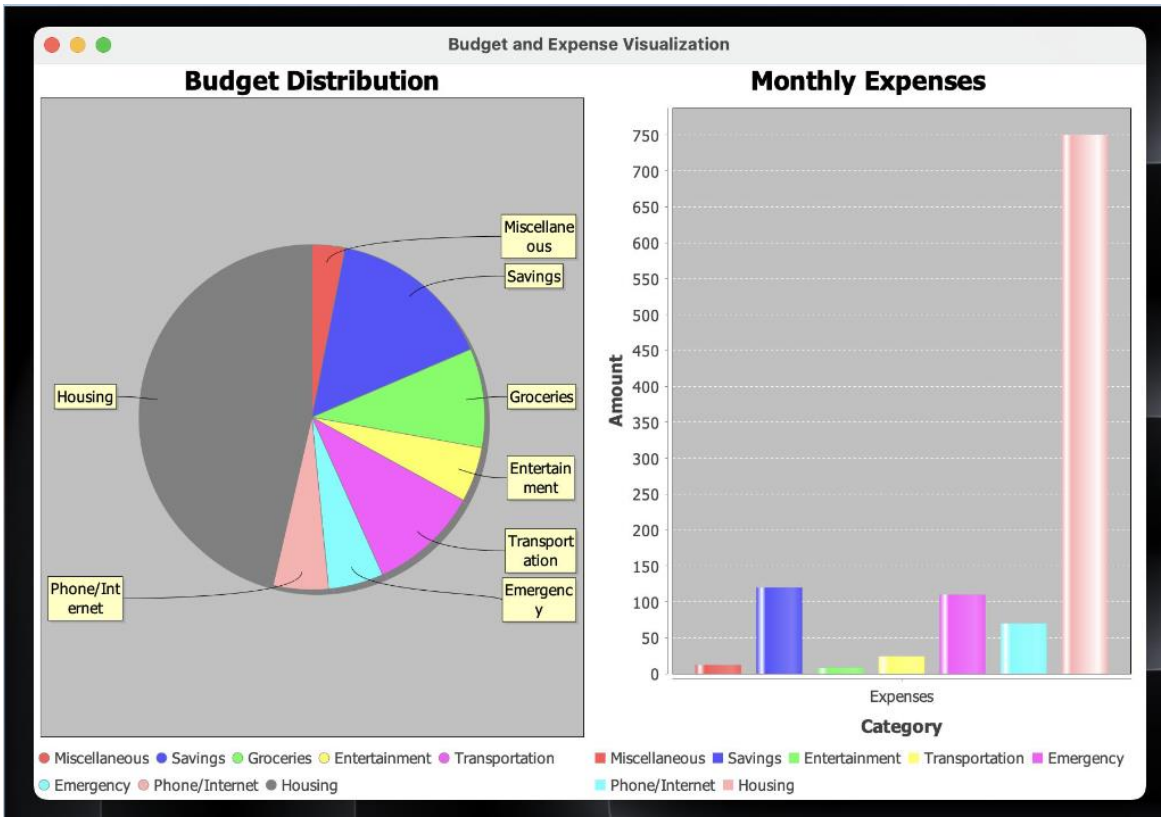

Fig 9. Visualization of Budget and Expense

2.4 Database Management with DatabaseHandler:

Functioning as the central hub for all database operations, this class takes on the responsibility of managing connections to the SQLite database with finesse.

2.4.1 Purpose of *DatabaseHandler*:

The *DatabaseHandler* plays a crucial role in ensuring the integrity and accessibility of financial data. It achieves this through the following key functionalities:

- **Managing Database Connections:** The *connectTransactions* and *connectBudgets* methods establish connections to the respective SQLite databases, laying the foundation for secure and reliable data transactions.
- **Initialization of Database:** The *initializeDatabase* method ensures the existence of necessary tables, creating them if they don't already exist, providing a structured foundation for data storage.
- **Transaction Operations:** Methods like *insertTransaction* facilitate the addition of new transaction records, contributing to the real-time tracking of financial activities.
- **Budget Operations:** Functions such as *insertBudget*, *updateBudgetLimit*, and *deleteBudget* manage budget records, offering users control over their spending limits.
- **Budget Verification:** The *budgetExists* method checks the existence of a budget category, adding an extra layer of validation to user inputs.

- **Data Retrieval:** The *getAllTransactions* and *getAllBudgets* methods retrieve all transaction and budget records, respectively, facilitating comprehensive data analysis and financial management.
- **Financial Analytics:** Analytical operations like *getMonthlyExpenses*, *getBudgetLimits*, and *getBudgetLimit* empower users with insights into their spending patterns, allowing for informed budget adjustments.

2.4.2 Core Functionalities:

The *DatabaseHandler* performs critical core functionalities to ensure the smooth functioning of the Personal Finance Tracker application:

- **Transaction Processing:** Efficiently processes new transactions and updates existing records based on user inputs, maintaining an accurate and up-to-date financial database.
- **Data Retrieval and Display:** Retrieves and displays financial data, such as transactions and budgets, within the GUI using methods like *getAllTransactions*, fostering a transparent and user-friendly interface.
- **Visualization Support:** Supplies financial data to visualization components like *PieChart*, enabling users to gain valuable insights into their financial health through graphical representations.

2.4.3 Integration with *FinanceTrackerGUI* and Accounts:

The *DatabaseHandler* seamlessly integrates with the *FinanceTrackerGUI* and Accounts components, ensuring effective communication between the user interface and the underlying databases:

- **User Interface Communication:** Handles data input/output between the GUI and databases, exemplified by methods like *insertTransaction*, which is invoked when users add transactions via the GUI.
- **Data Display:** Retrieves and displays crucial financial data in the GUI, enhancing the user's understanding and control over their financial activities.
- **User Feedback:** Provides user alerts and confirmations after database operations, enhancing the overall user experience and providing assurance regarding the success of their actions.
- **Data Synchronization:** Maintains synchronization between in-memory financial data and database changes, ensuring that the user always interacts with the most recent and accurate financial information.
- **Budget Management:** Manages the storage, retrieval, and updates of budget data in the database, enabling users to effectively set and monitor their spending limits.
- **Analytical Operations:** Supports reporting and analysis by fetching and processing data, providing users with valuable insights through features such as monthly expense summaries.
-

2.5 SQL Schema for Transactions:

```
CREATE TABLE IF NOT EXISTS

transactions (

id INTEGER PRIMARY KEY,

amount REAL NOT NULL,

date TEXT NOT NULL,

description TEXT,

category TEXT NOT NULL

);
```

2.6 SQL Schema for Budgets:

```
CREATE TABLE IF NOT EXISTS

budgets (

category TEXT PRIMARY KEY,

"limit" REAL NOT NULL

);
```

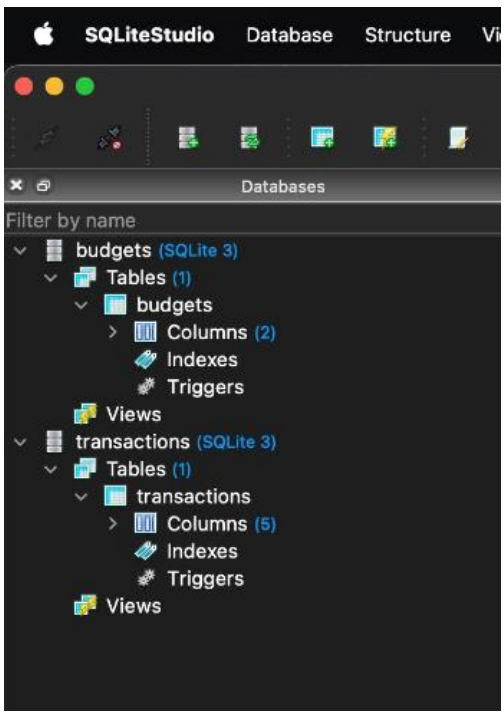

Fig 10. SQL schema for transactions table and budgets table

2.7 Skeleton Overview of FinanceTrackerGUI:

The *FinanceTrackerGUI* class serves as the cornerstone of the user interface in the Personal Finance Tracker application, seamlessly managing the layout and user interactions. This class encapsulates various key elements and functionalities that contribute to a smooth and intuitive user experience.

At its core, the *FinanceTrackerGUI* class includes a distinctive *BackgroundPanel* inner class, a custom *JPanel* adorned with a background image. This element is strategically employed to set the aesthetic tone of the main frame, providing a visually engaging backdrop for the entire application.

The user interface is constructed with essential GUI elements, each playing a pivotal role in user interaction. The *JFrame* frame stands as the main application window, while the *BackgroundPanel* contributes to the visual appeal. The *JPanel buttonPanel* conveniently organizes buttons that trigger various actions, such as managing transactions, budgets, viewing transactions, exporting data, and accessing charts. Two essential *JDialogs*—*transactionDialog* and *budgetDialog*—efficiently handle transaction and budget inputs, enhancing the overall usability of the application.

User interaction components include text fields for entering amounts, descriptions, categories, and budget limits. These are complemented by buttons that facilitate diverse actions. A specialized *Date Picker*, represented by the *JXDatePicker datePicker*, simplifies the selection of transaction dates.

Utility and display elements, such as the *JTextArea outputArea* for presenting transaction and budget information, and the *JLabel totalSpentLabel* to display total expenditure, contribute to a comprehensive and transparent financial overview. The *SimpleDateFormat dateFormat* ensures a well-formatted display of dates.

In terms of event handling, the *FinanceTrackerGUI* class implements methods like *showTransactionDialog*, *addTransaction*, *refreshTransactions*, among others, effectively managing user-triggered actions. These methods orchestrate the flow of information and ensure a responsive and dynamic user interface.

The integration of the user interface with the backend is a critical aspect, with instances of the *Account* and *DatabaseHandler* classes seamlessly interacting with the database and managing financial data. This integration is pivotal in maintaining data accuracy and reliability.

During initialization and execution, the *FinanceTrackerGUI* class sets up the UI, initializes the necessary components, and stands ready as the gateway to the application. Its main method orchestrates the initiation of the application, ensuring a user-friendly and feature-rich financial management experience.

Fig 11. Finance Tracker GUI

3. Conclusion:

This finance tracker application is a useful tool for monitoring personal finances and encouraging saving habits. By allowing users to enter their income and expenses on a daily basis, the software supports users in keeping track of their money and minimizing impulsive and unexpected spending. If a person knows exactly where their money goes each month, they may immediately see where they can save money and make some sacrifices. The project is successful in avoiding the use of manual computations to estimate monthly revenue and expenses. The modules have been intended to be both functional and visually pleasing. Overall, the Personal Finance Tracker app is an excellent way to promote financial responsibility and increase financial sustainability.

In conclusion, the Personal Finance Tracker is a robust application that combines user-friendly design with powerful features for effective personal finance management. The project is designed to demonstrate a meticulous approach to transaction and budget management, incorporating visualizations for enhanced understanding. The modular design, SQL database integration, and intuitive GUI contribute to the application's scalability and user accessibility. The provided code snippets and SQL schemas offer insights into the application's underlying structure and data management processes. The walkthrough of the GUI highlights the simplicity and efficiency of the user interface, emphasizing the application's practicality for users seeking a comprehensive personal finance solution.

3.1 Future Scope :

3.1.1 Alerts & Notifications :

Customizable Alerts:

- Implement a feature allowing users to set personalized alerts for upcoming bills, ensuring timely payments and avoiding late fees.
- Users can define budget limit alerts to receive notifications when approaching or exceeding set spending limits, promoting financial discipline.

Proactive Financial Advice:

- Introduce a proactive advisory system providing real-time financial advice based on spending patterns and budget analysis.
- Reminders for savings milestones to encourage users to meet their financial goals, fostering a habit of responsible financial planning.

3.1.2 Enhanced Report Generation:

Detailed Financial Reports:

- Develop features for generating comprehensive financial reports, including monthly spending summaries, enabling users to gain insights into their expenditure patterns.
- Implement category-wise expenditure analysis to highlight major expense areas and identify opportunities for cost-cutting or budget adjustments.

Year-End Financial Overviews:

- Provide functionality for generating year-end financial overviews, offering a holistic perspective on the user's financial health.
- Enable users to assess annual savings, investment growth, and identify areas for improvement in their financial planning.

Export Options:

- Offer users the flexibility to export generated reports in various formats such as PDF, Excel, or HTML.
- Facilitate easy sharing and printing of reports, ensuring users can communicate their financial data effectively with advisors or for personal records.

Admin and User-based Login:

- Introduce a secure login system with distinct roles for administrators and users.
- Admins can access and manage system settings, perform data analysis, and ensure overall system integrity.
- Users will have personalized accounts, ensuring data privacy and tailored financial insights based on their individual transactions and budgets.