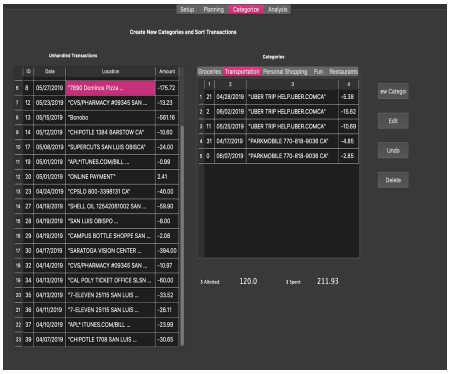
Robert Scott // Collaborator // rdavidscott1@gmail.com //
(408)-833-3831

https://github.com/kscott27/TransactionSorter

Transaction Sorter

Application Overview

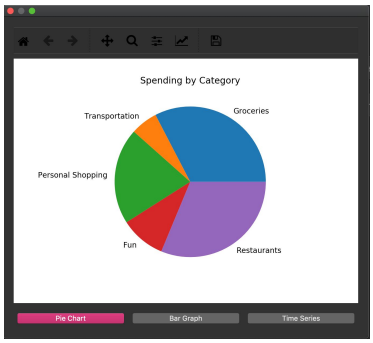


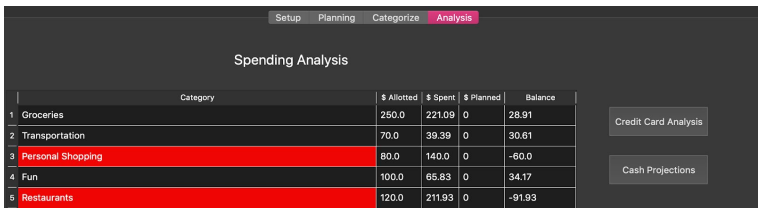
Transaction Sorter

The Transaction Sorter is a personal financial management application. It offers several analytical tools for a comprehensive overview of the user's financial standing- past, present, and future. The application takes several user-defined benchmarks that will then itemize transactions and project balances based on the user's financial goals.

Analytical Capabilities

The Analysis tab has a window that tracks the essential figures for each category: allotment, planning, spent, and balance. Below that table, there are figures that tell the user how much of their budget they have spent. The figures will be checked to see if they are close to overspending. If there is a chance of overspending, the application will flag the category and represent it in a different color. Within the Analysis tab, there are two buttons that will open pop-up windows with plotting capabilities.





Financial Planning

Users can specify "Planned Transactions" in a particular category as supplemental data. These are used for future cash projections and planning.

Code Structure

The GUI was created using PyQt5 in Qt Designer. It is all held in BootGUI.py. Any button functions and user entries will go through Application.py.The API was implemented to completely isolate the front-end from the back-end. The functionality of the backend was its own engine that simply hooked into the GUI I created using PyQt5. The back-end's compartmentalized code structure allows for efficient, adjustable functionality. The three major branches consist of the AnalysisManager, the TransactionManager, and the PersistentDataManager.

The first branch, which has the most modular structure within itself, is the "Analysis Manager". Its various sub-divisions work to give a comprehensive overview of the user's financial standing. The subdivisions' jobs include parsing, sorting, preparing useful information for visualization purposes. Another important sub-module is the "Plotting Data Factory". This module utilizes pandas to store all data in DataFrames. The DataFrame allows for quick and specific access to retrieve data that the user needs. Right now, this class procures data in array form to be plotted in the GUI. The arrays are ready to be used in matplotlib once they leave this object.

While there is an object for analysis, the "Transaction Manager" itemizes transactions in correct categories, setting up the foundation for useful analysis. Whenever the user manually sorts transactions, or creates a new category or plans and major transactions, this class will register the transaction using the Transaction and Category factory. This enables all functionality to make this data ready for analysis.

The "Persistent Data Manager's" main responsibility is to retain vital user information between reboots of the application. Whenever the application is opened, critical information (user-defined categories and planned transactions) is written to and saved in an .xml file.