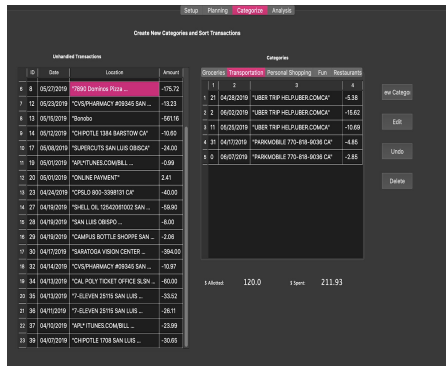


Transaction Sorter

Application Overview



The screenshot shows the 'Categorize' tab of the Transaction Sorter application. It features a table of transactions with columns for ID, Date, Location, and Amount. A category selection menu is open, showing options like Groceries, Transportation, Personal Shopping, Fun, and Restaurants. The 'Personal Shopping' category is currently selected.

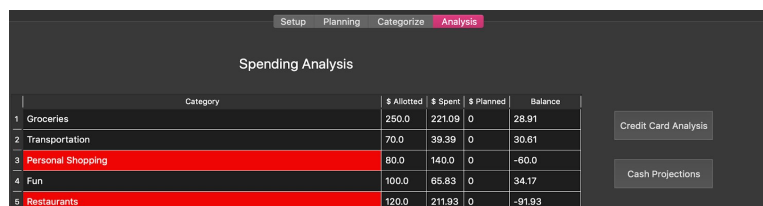
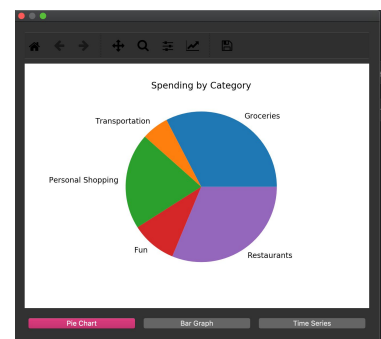
| ID | Date | Location | Amount |
|----|------------|----------------------------------|---------|
| 6 | 06/07/2019 | "H&M Downtown Plaza ... | -175.72 |
| 7 | 06/07/2019 | "CVS PHARMACY #00345 SAN ... | -13.23 |
| 8 | 06/07/2019 | "Santitas | -501.16 |
| 9 | 06/07/2019 | "CHOPOTLE SAN LEANITO CA ... | -10.60 |
| 10 | 06/07/2019 | "SANTITAS SAN LEANITO CA ... | -24.00 |
| 11 | 06/07/2019 | "SAN LEANITO CONNELL ... | -2.00 |
| 12 | 06/07/2019 | "HOLING PARKWAY ... | 2.41 |
| 13 | 06/07/2019 | "SPICE 800-3388731 CA ... | -40.00 |
| 14 | 06/07/2019 | "SHELL OIL 70542081002 SAN ... | -58.90 |
| 15 | 06/07/2019 | "SAN LUIS OBISPO ... | -4.00 |
| 16 | 06/07/2019 | "CAMPUS BOTTLE SHOPPE SAN ... | -2.06 |
| 17 | 06/07/2019 | "SANTITAS VISION CENTER ... | -384.00 |
| 18 | 06/07/2019 | "CVS PHARMACY #00345 SAN ... | -10.87 |
| 19 | 06/07/2019 | "CAL POLY TICKET OFFICE BLVD ... | -60.00 |
| 20 | 06/07/2019 | "7-ELEVEN 20715 SAN LUIS ... | -33.52 |
| 21 | 06/07/2019 | "7-ELEVEN 20715 SAN LUIS ... | -26.51 |
| 22 | 06/07/2019 | "MEXI TONELCONELL ... | -23.00 |
| 23 | 06/07/2019 | "CHOPOTLE SAN LUIS ... | -30.00 |

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.



The screenshot shows the 'Spending Analysis' table in the Analysis tab. It displays a table with columns for Category, \$ Allotted, \$ Spent, \$ Planned, and Balance. The table is color-coded: red for categories where spending is close to or exceeds the allotment, and green for categories where spending is well below the allotment. Buttons for 'Credit Card Analysis' and 'Cash Projections' are visible on the right.

| Category | \$ Allotted | \$ Spent | \$ Planned | Balance |
|---------------------|-------------|----------|------------|---------|
| 1 Groceries | 250.0 | 221.09 | 0 | 28.91 |
| 2 Transportation | 70.0 | 39.39 | 0 | 30.61 |
| 3 Personal Shopping | 80.0 | 140.0 | 0 | -60.0 |
| 4 Fun | 100.0 | 65.83 | 0 | 34.17 |
| 5 Restaurants | 120.0 | 211.93 | 0 | -91.93 |

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.

