Phase 2 Project Report – GillPay™: A Personal Finance Tracker for Students

Course: ISTM 601
Team: Goofy Goldfishes
Phase 2 Due Date: September 14, 2025

Project Title and Team Information

Project Title: GillPay™: A Personal Finance Tracker for Students

- Team Members & Roles (Phase 2):
  - Enoch Adegbola – Team Leader
  - Ruben Alvarez – Developer (Transaction Input & Validation)
  - Matt Bennett – Developer (Data Storage & Recording)
  - Ally Herleth – Tester/Documenter
  - Jo Ann Kern – Documentation Support

Introduction and Purpose

The GillPay™ project aims to provide college students with a simple, engaging, and reliable tool for managing their personal finances. Many students struggle to track their income and expenses, leading to unnecessary financial stress. By creating an application that allows students to log transactions, categorize their spending, and view summaries, GillPay™ empowers users to develop stronger financial habits.

In Phase 2, the project moved beyond planning and design to its first functional stage. Specifically, this phase focused on implementing the core functionality of the program: recording income and expense transactions, storing them persistently, and providing users with real-time calculations of income, expenses, and net savings. This foundation serves as the backbone of the application, upon which more advanced features such as error handling, reporting, and visualization will be built in future phases.

Overview of Features

The Phase 2 version of GillPay™ introduces several important features that make the tool usable in practice. The application begins by prompting users to enter details for each transaction. These details include the date of the transaction, a short description (for example, “Textbooks” or “Groceries”), a category (such as food, rent, utilities, or entertainment), the monetary amount, and the transaction type (income or expense).

To ensure that transactions are organized and easy to review later, each entry is tagged with a category. All data is saved in a CSV file (gillpay_data.csv), allowing users to maintain a persistent record of their finances across sessions. Once data is stored, the application can calculate and display useful financial summaries, including the total income, total expenses, and net savings.

While these features represent only the initial implementation, they provide immediate value by allowing students to track where their money is going and whether they are living within their means.

Main Program File

The main entry point for GillPay™ is the file **main.py**. This file contains the program’s primary menu system and calls supporting modules for handling transactions and storage.

To run the program, users must ensure they are working in a Python 3.10+ environment. Once installed, the program can be started with the command:

```
python main.py
```

No additional dependencies are required at this stage, making the application lightweight and accessible for students who may not have extensive technical experience.

File Descriptions

The submission for Phase 2 contains several files that together make up the GillPay™ system:

- **/src/main.py** - The main entry file, which presents users with a command-line menu and directs input to the appropriate functions.

- **/src/transaction.py** - Defines the Transaction class and associated functions, including user input collection and input validation; This is the container for the transaction data.
- **/src/gillpay_service.py** - Handles the business logic for the GillPay application and interacts with the transaction_dao.py to help formulate the data.
- **/src/transaction_dao.py** - Handles the reading and writing of the data for the CSV file.
- **/data/gillpay_data.csv** - A persistent storage file that records all income and expense entries in comma-separated format.
- **/docs/goofygoldfishes_project_report_phase2.pdf** - The documentation for this phase of the project.

Each of these files plays a distinct role in the project architecture, and together they support the primary functionality introduced in this phase.

User Instructions

To use GillPay™, the user should first run the main program file as described above. Upon launch, the user is presented with a simple menu:

1 → Add a transaction
2 → View account summary
3 → Exit the program (Farewell!)

When adding a transaction, the program prompts the user for the date, description, category, amount, and type of transaction. Once the information is provided, the program validates the input and records the transaction in gillpay_data.csv.

For example:

```
Press 1: Add Transaction
Press 2: Account Summary
Press 3: Farewell!
Action:
```

The resulting entry is stored in the CSV file as:

```
type,category,amount,date
income,job,12.15,2025/09/14
```

By selecting option 2 from the main menu, the user can quickly view totals for all income and expense entries to date, as well as the resulting net savings.

Assumptions and Design Decisions

Several design decisions were made during Phase 2 to balance usability and technical feasibility. First, the team chose CSV storage over more advanced database solutions such as SQLite. This choice was based on the simplicity and portability of CSV files, which are easy to inspect manually and require no setup for the end user. Second, input validation was added for critical fields such as dates, amounts, and transaction types (categories). This ensures that invalid data does not corrupt the CSV file. Finally, the team opted to keep the interface as a command-line program for now. This allows the team to focus on core logic and correctness before moving on to graphical features in both Phase 3 and Phase 4.

Challenges and Solutions

As with any project, Phase 2 presented challenges. One recurring issue was **date validation**. Users often mistype dates or enter them in different formats. To resolve this, the team implemented Python's built-in datetime.strptime function to enforce the YYYY/MM/DD format. Another challenge was **file persistence**. Without careful handling, each new run of the program risked overwriting the CSV file. This was addressed by implementing append mode when writing new transactions.

The team also faced difficulties with **invalid user input**. To address this, the input functions were designed to loop until valid input is received, providing feedback for incorrect entries. On the organizational side, coordinating contributions across multiple developers required discipline. This was managed using GitHub for version control and Discord for communication, ensuring that all team members worked with the latest codebase.

Use of Generative AI

Generative AI tools, particularly ChatGPT, were incorporated into this phase of the project as supportive resources. The team used AI to draft starter code for the Transaction class and CSV handling, and to generate examples of input validation logic. Additionally, AI was helpful when troubleshooting file handling and debugging common Python errors.

Importantly, the team verified and adapted all AI-generated content to fit the project’s requirements. Code suggestions were tested extensively and revised where needed to ensure they were both functional and fully understood by the developers. The value of AI in this phase was in accelerating brainstorming and reducing the time spent resolving technical issues, while the final implementation reflects team comprehension and effort.

Future Improvements

While the Phase 2 implementation provides a solid foundation, there are several ways the application can be improved in future phases. Exception handling for corrupted or missing CSV files could enhance reliability. A category dropdown menu or predefined category list would standardize entries and make summaries more accurate. More advanced reporting, such as monthly summaries or category-based spending breakdowns, would provide users with deeper insights into their habits. Finally, the addition of unit tests would improve maintainability and ensure that new features do not break existing functionality.

Aggie Honor Code Statement

Statement: "An Aggie does not lie, cheat, or steal, or tolerate those who do".

Purpose: To unify the aims of all Texas A&M students toward a high code of ethics and personal dignity, promoting loyalty to truth and confidence in one another.

Application: The code applies to all academic and personal pursuits, emphasizing honesty and integrity.