SOFE 4790U: Distributed Systems (Fall 2023)

Instructor: Dr. Ahmed Badr

Assignment #2:

Honour code: By submitting this assignment, I (name and banner id# below) affirm this is my own work, and I have not asked any of my fellow students or others for their source code or solutions to complete this assignment, and I have not offered my source code or solutions for this assignment to any of my fellow students.

Name: Alden O'Cain

Banner ID#: 100558599

[Distributed Assignment 2 DEMO - YouTube]
[PyFinancials GitHub Repository]

1. Application idea

Effective financial management is essential for individuals and families to achieve their financial goals. PyFinancials aims to address this need with a user-friendly web application for budgeting and expense tracking. Designed to simplify financial management and provide users with a comprehensive platform to manage their finances efficiently PyFinancials allows users to make informed financial decisions. With a focus on user-centric design and an array of functionalities accessible via a RESTful API written in Python Flask, PyFinancials aims to improve personal financial management.

2. Client Services

1. **Expense Management:**
    a. GET /expenses: Retrieve a list of all expenses, allowing users to review their spending history.
    b. POST /expenses: Submit new expenses with details like amount, category, and description, making it easy to record and track expenditures.
    c. GET /expenses/{expense_id}: Retrieve details of a specific expense by its unique ID, enabling users to view individual expense details.
    d. PUT /expenses/{expense_id}: Update existing expenses, providing the flexibility to modify expense information such as the amount, category, or description.
    e. DELETE /expenses/{expense_id}: Remove specific expenses from the database, allowing users to maintain a clean and organized financial record.
2. **Budget Management:**
    a. GET /budget: Retrieve the user's budget details, including income and allocated amounts for various expense categories.
    b. POST /budget: Set up a new budget by specifying income and category-wise budget allocations, aiding users in planning and managing their finances.
    c. PUT /budget: Update the budget, allowing users to adapt to changing financial circumstances by modifying income or category budgets.
    d. GET /budget/categories: Get a list of expense categories available in the system, simplifying the budgeting process by providing predefined categories.

e. DELETE /budget: Remove the budget entirely, providing users with the option to start fresh with their financial planning.

3. **User Profile Management:**
    a. POST /users: Create a new user profile with basic information, allowing users to set up their PyFinancials account.
    b. GET /users/{user_id}: Retrieve user details by their unique ID, giving users access to their profile information.
    c. PUT /users/{user_id}: Update user information, enabling users to keep their profile data current and accurate.
    d. DELETE /users/{user_id}: Delete a user's profile, providing an option for account removal when needed.
4. **Reports Generation:**
    a. GET /reports/income: Generate reports on income, including monthly or yearly summaries, helping users visualize their earnings.
    b. GET /reports/expenses: Generate expense reports, such as monthly spending breakdowns, for a detailed view of financial habits.
    c. GET /reports/savings: Calculate and retrieve savings reports, comparing income to expenses, assisting users in understanding their financial health.
5. **User Profile Management:**
    a. GET /categories: Retrieve a list of predefined expense categories for easy budgeting, offering users a set of standard categories to choose from.
    b. POST /categories: Allow users to create custom expense categories for personalization, giving them flexibility in categorizing their expenses.
    c. PUT /categories/{category_id}: Update the name or details of a custom category, allowing users to refine their expense categories.
    d. DELETE /categories/{category_id}: Remove a custom category, offering the option to declutter the list of custom categories.

3. Describe the novel features

PyFinancials has several novel features designed to provide users with an intuitive and creative financial management experience. One of its key functionalities is the ability to generate savings reports, a feature that goes beyond basic expense tracking. These reports allow users to assess their financial health by comparing their income against expenses, providing valuable insights into their savings trends over time. The creative approach lies in its simplicity – an easy-to-understand visual representation of income versus expenditures empowers users to make informed financial decisions. Additionally, PyFinancials offers the creative feature of custom expense categories, allowing users to personalize their expense tracking to suit their unique financial habits. These custom

categories enhance the overall user experience, making it an indispensable tool for individuals seeking intuitive and creative solutions for financial management.

4. Challenges and solutions

Several challenges were encountered, primarily rooted in time constraints and technical intricacies. The limited timeframe posed a challenge in comprehensively completing the project, prompting the adoption of a prioritization strategy that focused on minimal functionalities (5 services, 1 each GET, POST, PUT, DELETE). Organizing the application logic into distinct routes within Flask required planning to ensure proper functionality and data flow across endpoints. To streamline the project structure, a modular approach was employed, utilizing Flask's Blueprint functionalities for improved maintainability and readability. In implementing the website's favicon I never saw it work however the 404 request for the resource stopped occuring. Additionally, configuring Cross-Origin Resource Sharing (CORS) for AJAX requests presented challenges, which were addressed through detailed investigation and implementation of CORS settings to facilitate seamless communication between frontend and backend components.

5. Testing

During the development process unittest test modules were written in Python to validate the functionality of the API routes, although they were not utilized due to time constraints. Testing was primarily minimal and involved rudimentary methods, such as displaying the internal state of the server using an index page. Additionally, the testing process included making Cross-Origin Resource Sharing (CORS) requests to the Flask API through HTML pages using AJAX. The focus on minimal testing was mainly due to time constraints and academic priorities, resulting in a more limited and rudimentary testing phase.

6. Conclusion

In conclusion, PyFinancials is a highly effective financial management application designed to simplify budgeting and expense tracking while offering a range of unique and user-friendly features. With a robust set of client services accessible through a RESTful API, PyFinancials empowers users to manage their finances efficiently. The application introduces innovative functionalities, including the ability to generate savings reports, which provides users with a valuable perspective on their financial health. The

user-centric design and the option to create custom expense categories add a creative dimension to the application, making it a versatile and indispensable tool for individuals seeking intuitive and creative solutions for financial management. PyFinancials is well-equipped to meet the diverse needs of users in their pursuit of financial well-being.