Personal Financial Tracker App

Project Plan

By Benjamin Etheridge

Final Year project – cs3810

Supervised By: Dr Donggyun Han

Department of Computer Science

Royal Holloway, University of London

Abstract

Managing personal finances can be a daunting task for many, leading to financial stress and poor decision-making. In the U.K., 47% of adults do not feel confident in financial products and services [5]. This lack of confidence is partially due to financial illiteracy and the overcomplication of financial data presentation, which contributes to poor understanding of personal finances; for instance, 22% of individuals have less than £100 in savings, and 47% have no financial goals for the next five years [5].

Multiple studies have shown that visual information is retained more effectively than written or verbal data [2]. This project will focus on translating financial information into visual representations, such as graphical interfaces, while studying the necessary tools to do so. The emphasis will be on mapping data onto graphs that are both visually appealing and easy to understand. React is the chosen frontend framework due to its reusable components, flexibility, and fast rendering capabilities due its Virtual DOM[3]. Additionally, Recharts will be utilized for data visualization, allowing for seamless integration.

Leveraging N-tier architecture will enable the application to handle large and complex requests while providing scalability [1]. The application will feature a user-friendly interface, ensure a responsive user experience, and efficiently manage data. Essential features will include user authentication, transaction categorization, budget setting, and various visual analytics. Given the sensitivity of the data, data encryption will be crucial, ensuring HTTPS for encryption in transit and potentially encrypting at rest[4]. This project will adhere to best practices in software engineering, with an emphasis on modularity, maintainability, and especially security.

By integrating state-of-the-art digital solutions with targeted financial education, individuals will be better equipped to navigate their financial landscapes and improve their overall financial well-being.

Timeline

Term 1:

- **Week 1/2**: User login and authentication
- **Week 3**: Welcome header, dashboard page, link to other pages
- **Week 4**: Create database for storing data; API for uploading to the database
- **Week 5/6**: Uploading transactions; import bank statements (CSV in correct format)
- **Week 7**: Retrieve transactions from the database and format for the webpage
- **Week 8/9**: Filter functionality to sort transactions by categories
- **Week 10**: Interim reports
- **Week 11**: Finalize reports and presentation

Term 2:

- **Week 1**: Insight page/tab
- **Week 2/3**: Graphical representations of expenditures over time (by category)
- **Week 4/5**: Set budgets page
- **Week 6**: Graphical representation of budget vs. expenditures
- **Week 7**: Polish user interface
- **Week 8**: Testing phase
- **Week 9**: Buffer week
- **Week 10/11**: Final report and presentation

Risks and Mitigations

Security Risk: When developing an application that allows users to upload bank statements, it is essential to encrypt sensitive data during transmission and at rest. Implementing security measures against SQL injection is also necessary, including methods like strict access controls.

Time Allocation Risk: Allocating sufficient time for each project phase is crucial to ensure timely progression. If any stage falls behind, it may hinder the ability to deliver the required outcomes. Proper time management and allocation should be enforced.

Data Storage Considerations: Storing large datasets can be resource intensive. The application will require a scalable solution with potentially 24-hour access, such as cloud storage services (e.g., AWS S3). Ensuring these services implement robust security measures is essential.

Design Risks: Users, particularly older individuals, may struggle to submit statements in the required format, leading to errors and frustration. Providing a generalized guide outlining acceptable formats, along with step-by-step instructions and validation checks, can mitigate this risk.

Coding Risks: Over-reliance on pre-made packages and APIs for functionalities, such as data visualization, may lead to issues if those libraries lack necessary features. It is important to invest time in evaluating libraries to ensure their capabilities and limitations are well understood.

References

[1] Alseelawi, S., Adnan, K., Hazim, T., Alrikabi, H., & Nasser, K. (2020). Design and Implementation of an E-learning Platform Using N-Tier Architecture. *International Association of Online Engineering*. Retrieved from https://www.learntechlib.org/p/216473/.

[2] Mayer, R. E., & Moreno, R. (2003). Nine ways to reduce cognitive load in multimedia learning. *Educational Psychologist*, 38(1), 43–52. https://doi.org/10.1207/S15326985EP3801_6

[3] Psaila, G. (n.d.). Virtual DOM: An Efficient Virtual Memory Representation for Large XML Documents. *Virtual DOM*.

[4] Catalyst2. (n.d.). Encryption at Rest: Its Purpose and Benefits. Retrieved from https://www.catalyst2.com/blog/encryption-at-rest-its-purpose-and-benefits-setbacks/.

[5] Money Advice Service. (2020). Financial Capability Survey, 2018. [Data collection]. 2nd Edition. UK Data Service. SN: 8454, DOI: http://doi.org/10.5255/UKDA-SN-8454-2