

The Investigation and Development of a Personal Finance Tool to Improve Financial Capability

Project specification

Kiran Sanganee

Department of Computer Science

University of Warwick

1 Problem Discussion

Managing your personal finances has always been hard, whether you use a limited mobile app, a basic Excel spreadsheet, or you don't at all. The current cost of living crisis affecting the UK only makes matters worse, with household essentials increasing faster than incomes.

The concept of financial capability..

The aim of this project is to analyse the best strategies for individuals to better their understanding of money management and improve their financial capability; then implement these ideas into a web-based, free personal finance tool.

2 Background

I will categorise the existing software into three sections, all of which have their own problems.

Firstly, there is the mobile banking apps such as Revolut and Monzo, they track your spending within those specific accounts, but often struggle or don't at all integrate across banks to have a centralised view. Furthermore, to use these services you have to sign up with that bank; people are less likely to trust mobile apps with their personal data than websites; and to get started the app has to be installed over just accessing a site.

Then there are the websites which allow you to sync data with your bank. After signing up, you enable the website to access your bank accounts, which then very slowly imports all your data. Navigating on these are unintuitive and eventually after you find a service which doesn't cost a fortune, the obstructive adverts will pop up and block you from even using it. In addition, some sites are known to not have enough features to entice you to make it worth using and others have too many useless features which don't actually help and instead overwhelm the user.

The third category are the websites or apps which do not let you integrate your accounts. These expect you to manually add your banking information

and then they perform analysis to provide suggestions on how to save. They are little better than an Excel spreadsheet and are by far the worst of the three categories.

Ideally, my software will solve the problems with the existing solutions, incorporate the best features they have into one, and have additional functionalities which I deem to be helpful in improving money management from my research.

In trying to avoid the category of software which doesn't let individuals sync across their banking information, the project will utilise the data from endpoints as a result of the open banking movement. In particular, it will use the Plaid API, enabling the solution to obtain information from all bank accounts a user gives permission to access, in one centralised location for easy analysis and better results.

3 Objectives

3.1 Functional Requirements

- The software must be a web-based application
- Users must be able to create and log into their account
- Authenticated users must be able to link their bank accounts with Plaid
- A user must be able to access their dashboard
- A user's dashboard must display their bank accounts and recent transactions
- For each strategy, the software must enable users to opt-in and benefit from it
- An investments strategy must be implemented
- A budgeting strategy must be implemented

- A savings strategy must be implemented
- A categorising bills strategy must be implemented

3.2 Non-functional Requirements

- The software must be intuitive to navigate and use
- The website must be fast to load and responsive to user input
- The solution must be secure and protect user data
- The software must be reliable and robust so should not be able to produce errors
- The application must be easy to maintain and extend
- The website must be visually appealing
- The software must be scalable in order to handle a large number of users
- It must be accessible to all users including those with impairments

3.3 Possible Extensions

- Users may be able to update their details and delete their account
- Further strategies found during research may be implemented
- Users will be able to provide feedback and recommendations based on their experience

4 Methodology

The project can be divided into three sections: research, development and testing, with development being further subdivided into the initial backbones of the software, and then repeated strategy implementation.

4.1 Research

The initial research will be into creating a thorough understanding of the technologies and development tools. The frontend will utilise Next.js, a react framework with faster page loading times making the user experience better; and the backend will be written in JavaScript with express and use a Mongo database to store user data. In order to use the Plaid APIs with the correct flow for authentication, the frontend requests must go through the backend as a proxy before reaching plaid and returning the data.

Beyond the initial research, during the repeated strategy implementation, academic papers, books, reports and other sources will be read in order to discover the money management strategies. Furthering this, within each strategy there may be conflicts and sub-strategies which must be weighed up and analysed to decide which is the best to add to the software. The intention of the research is as to not just copy what the existing tools already have, but find methods and evidence to support that it actually benefits an individual, when taking control of their finances. A trivial example is that people often have many bank accounts for different purposes but in order to properly view all the transactions and determine where money could be saved, they ought to be collated into one central view. Research will be done into finding evidence and reasoning to support that this is true, but if not and a better solution is found then that idea will be implemented into the tool.

4.2 Development

The initial backbones of the website will be developed using the waterfall strategy, as the plan based methodology lends itself nicely to getting the basic software ready for further development. Full design, implementation, and testing will be done on this skeleton to confirm it works as expected. This foundation involves having full user authentication with signing up and logging in, a basic dashboard where users can link their banks, and a page to view all their most recent transactions from any of their accounts. The benefit of having a clear end goal makes the waterfall strategy perfect as detailed planning can be completed beforehand and then methodical progression through

the stages is clear and with reduced risk. Moreover, it eases time management as set lengths of time can be accurately estimated for each part, helping ensure the projects objectives are reached.

During the repeated strategy development phase, an agile approach will be adopted. This synergizes with the aim to do large amounts of research on effective strategies for individuals to increase their financial capability, followed by the implementation of that strategy into the website. The features that will be added are currently unknown so are unable to be planned. Additionally, the fact that at this stage the process will be quite repetitive works well with the cyclic nature of agile of planning (and researching), designing, developing and testing. At each cycle, the software should be ready to be distributed, so at this point during development, a working version is always ready.

Throughout development, git will be used to as version control to manage changes and features effectively. If a mistake is ever made, reverting to a working version can be done quickly. The Git flow practise will also be followed, using main, release, development and feature branches along with frequent merges to keep the repository clean and professional. Finally, GitHub will be used to store a remote version of the repository and also aid documentation.

4.3 Testing

Similar to the development phase, the testing will be split up into two sections. Initially when working on the website's foundation, frequent unit testing will be done on the frontend and backend. In some instances, the tests will be written before the code is written, as this follows the test-driven development style helping break down the problem into manageable chunks as well as improves general code quality. Some of the tests will cover the integration of the frontend and backend, for example when a user signs up, the backend will add the user to the database and the frontend update the state to reflect this.

In the second phase of development, the testing will be more user-based and occur more frequently due to the testing sections of each sprint in agile development. Following the implementation of a strategy, a user will be asked to use that strategy and test it works as expected. A list of steps will not be

given as to also test the how intuitive and easy to navigate the UI is. The user's feedback will be acted upon and the feature will be improved if necessary.

After the major development has finished, the software will be tested by a larger group of users with various levels of experience. This will be more use-case testing where the users are allowed to use the software as they would normally and report any issues they find. The results of this testing will be analysed and any issues will be fixed before the software is released.

5 Project Timetable

Dates	Event
T1 W1-2	Project specification
T1 W1-2	Research into development tools
T1 W3-9	Development of website foundation
T1 W4	Design basic UI
T1 W5-7	User authentication
T1 W8-9	Integration with PlaidAPI
T2 W1	Testing of website foundation
T2 W2-10	Repeated strategy development*
Easter	Finishing touches
Throughout	Document research
T3 W1	Dissertation due

*Repeated strategy development is: 1 week of research, 1 week of development, and 1 week of testing

6 Legal, Social, Ethical and Professional Issues

The software will have an authentication system enabling users to sign up and log in, making the user experience smooth. Consequently, the current guidance and best practises for security will have to be followed. This includes enforcing a long and complex password for users to avoid unauthorised access from others, but also storing their password and data in a fashion which complies with the GDPR regulations.

As a result of using the Plaid API, no extremely sensitive banking information will be stored locally by me; all this financial data is handled by the Plaid service itself. When the data is displayed to the user, they can be sure that it is verified and secured, so long as the correct flow described in the Plaid documentation is followed during development. Furthermore, when accessing certain products provided by Plaid, steps will be taken to avoid getting unnecessary data, for example when accessing an account's transactions the account number can be displayed as only the last few digits as to avoid vulnerabilities.

The biggest issue the software must avoid is ensuring that the strategies and suggestions are not considered financial advice. To mitigate this, research will be done into knowing the boundary between advice and guidance. The functionality on the website which helps individuals become more financially capable must be opt-in as to avoid pushing suggestions onto users because it may overwhelm them and is against the laws set out by the Financial Conduct Authority.

References