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

Progress report

Kiran Sanganee

Department of Computer Science

University of Warwick

WARWICK

THE UNIVERSITY OF WARWICK

1 Introduction

The aim of this project is to investigate how individuals can improve their financial capability by managing their money more effectively; then produce a web application in which users can easily access these strategies. Progress on this aim has been made in accordance with the project specification timetable (Appendix: A), with a basic proof of concept in development and ongoing research into the required aspects. This report explains the project in more detail, summarises the research and development that has been done so far, and outlines the next steps to be taken.

2 Background

The idea of an individual being financially capable is defined by the Financial Capability Strategy for the UK as them having the "ability to manage money well" and the "attitude ... [and] confidence to put (their) money skills into practise" [1]. After researching what tools will be best to help users achieve this, they will be implemented into the application. This will not only give users confidence, but also the information they need to improve their money management skills and make the right decisions.

2.1 Limitations of Current Applications

Motivation for such software comes from the limitations in the current applications available to users. The project specification (Appendix: A) goes into more detail but in summary, there are three different types of software. Firstly, there are mobile banking apps which track customers' spending on that account, however, these these do not integrate with any accounts from different companies. Then, there are websites which perform the analysis on various accounts but require the user to manually import their financial data. Finally, there are websites which automatically sync the data, but they lack many saving features, are difficult to use, and often require a paid subscription [2].

2.2 Software

This project is an attempt to solve these problems. The resulting software will be a web application which can import users' financial data securely and automatically. This is done with the Plaid API [3], a set of endpoints to access a user's bank account data as a result of the recent open banking movement [4]. A secure authentication system will be implemented to ensure that only the correct user can access their data. After signing in, the user will be able to link any of their bank accounts for analysis to be performed. Their dashboard will include all their recent transactions, and they will be able to switch to a category view to understand their spending habits. Further features and analysis tools will be implemented based on the research into which strategies are most effective. Examples include a budgeting section to organise a budget and encourage saving; as well as an investments tab where users can see their current investments and the performance of their portfolio.

2.3 UK Cost of Living Crisis

The timing of this project is also extremely appropriate as the UK is currently facing a cost of living crisis and is only made worse by the war in Ukraine [5]. The inflation rate of the household utility sector is at 26.6% according to the UK Office for National Statistics [6]. Users could make use of the software as they can group their transactions by category, enabling them to better compare their bills and see where they can save. Furthermore, by using the budgeting feature, they can set strict limits on what is spent and what is saved to be in a more comfortable position in the future.

3 Progress

Progress so far can be divided into two sections. Firstly, research has been made on the distinction between financial advice and guidance, the technologies to use for development, and the money management strategies. The second section is the development of a proof of concept web application, using the waterfall methodology, to demonstrate the feasibility of the project.

3.1 Research

3.1.1 Financial Advice vs. Financial Guidance

As outlined in the risks section of the project specification (Appendix: A), a major factor to consider is distinguishing between financial advice and financial guidance; therefore this was heavily prioritised during the early stages of research.

As defined in the report from the Financial Conduct Authority (FCA) and HM Treasury [7], financial advice is defined as "a service which recommends a specific course of action based on consumers' individual circumstances and goals". This is in contrast to financial guidance, which is defined as "provides information and/or options to narrow down consumers' choices, without making explicit recommendation". These definitions are extremely reliable as the report is part of the Financial Advice Working Group's Review (FAWG), whose sole purpose is to demystify the differences and is what is considered in court.

It is important to consider this distinction when developing the application, as in the UK, it is illegal to provide financial advice on regulated investments without being vetted by the FCA's fact-find process [8]. In addition, the repercussions of providing financial advice include the potential that if a user were to lose money, they can sue the software for damages which is a major risk to consider [9].

In particular, the application will be providing information specific to each user, as a result of using Plaid [3], and so emphasis must be made that the strategies are not based on this, but rather are just general tools to help the user manage their money more effectively. An example an of implementation that follows is in the budgeting strategy. The user will be able to see their recent transactions, but they must opt in to begin budgeting and then set their strategy. The application will recommend the most effective strategy based on research later on in the project, but this will never be pushed onto the user.

3.1.2 Technologies

Also in accordance with the original timetable, research was conducted before development to help determine what technologies will be used. The main technologies that were considered were the web application framework, the database and the API for the bank connections.

Next.js was chosen as the frontend web application framework. It is a react-based framework that is extremely popular and has a large community. It is also very easy to use and has a lot of documentation. The project specification's non-function requirements were also considered when choosing the framework. For example, the requirement that the "website must be fast to load and responsive to user input" (Appendix: A) is satisfied by Next.js as it has in-built functionality for server-side rendering and makes many optimisations by default [10]. Server-side rendering is when the more powerful server will render the page rather than the client; this means the pages feel faster and more responsive [11]. Another requirement was that the application "must be visually pleasing". This will be accomplished by using TailwindCSS [12], a set of CSS classes that allows for complete customisation of the website's appearance without using a heavy component library.

To incorporate the users' bank data, Plaid was chosen as the API. It gives the most freedom to a user when specifying exactly what accounts are shared with the application; furthermore, they get peace of mind that their data is secure by Plaid using strict security measures such as end-to-end encryption and AES-256 with TLS [3]. The developer never sees or has to store the user's bank credentials as Plaid handles all of this, meaning the only data that is stored is the user's access token. This is stored in a secure database discussed in the next section. Moreover, Plaid has very detailed documentation and several tutorials on getting started [13].

Originally, a full backend to perform the required server functionality was planned; however, after further research and understanding of the benefits of Next.js and Plaid, a conclusion of just using a simple database and performing all the querying using Next's API routes was made. Only the data for user authentication and a user's access token (or Plaid) needed to be stored, so a lightweight database was opted for. After comparing the various SQL and NoSQL alternatives, Pocketbase was the final decision thanks to its simplicity and ease of use [14]. It is effectively a wrapper for SQLite, written in Go with a JavaScript SDK and REST API [15], so is the perfect choice for an application like this.

3.1.3 Budgeting

The budgeting strategy will be the first strategy to be developed as it is the most straightforward, and will be the easiest to demonstrate the feasibility of the project. This development is set to begin in a later phase of the project, but research has been conducted to help determine what the best budgeting approach is and how to implement it into the app already.

Budgeting is a concept that most would associate with money, but after reading some of the academic papers available such as [16] and [17], it is clear the budgeting can be applied to any type of resources such as time or energy. Furthermore, budgets can vary on a scale from personal to organisational; for example this book [18] discusses how budgeting is carried out at a government level. This means that the best aspects of all types and scales can be applied to build the best strategy, and then implement it into the application.

So far some research has gone into finding which budgeting approach and the current conclusion is to do the 50/30/20 strategy. This is where an individual separates their monthly income into 50% for needs, 30% for wants and 20% for savings or repaying debt [19]. It originates from the book "All Your Worth: The Ultimate Lifetime Money Plan" [20] which itself references over 20 years of research. It is considered the best approach for the web application as it is simple to understand, yet effective according to many studies [21]; on top of that, it is also considered very appropriate for the current cost of living crisis over other methods like the digital envelope approach [22].

The implementation of this 50/30/20 strategy is also somewhat simple, as users will be able to opt in to let the software calculate how much money is assigned to each section. Furthermore, it would be possible to let users categorise their previous transactions into these and determine how they would need to change their previous spending habits to match. Also, they could have the 20%, that is used to repay debt and save, automatically accounted for in a separate loans section of the application.

The research at this stage is not final and other options are still being considered, however, the 50/30/20 strategy is the current favourite. As a result of performing the second phase of development in sprints, it will be possible to test the effectiveness of the strategy and make changes if necessary without majorly affecting the project timeline.

3.2 Development

The development of the proof of concept web application is using the waterfall methodology - a sequential approach to software development. This is enabled as the requirements are well defined and there is a clear end goal. So far, two separate concepts have been developed. The first is the basic authentication system using Next.js, TailwindCSS and Pocketbase. It is currently working and passing all the unit tests that have been written. The second is the Plaid integration, where the quickstart project has been set up and is currently working with some slight adjustments. The next step is to combine the two concepts and then move on to repeated strategy implementation.

4 Project management

As alluded to in the previous section, the progress made so far is in accordance with the project timeline set out in the specification, therefore the timetable remains unchanged. Ideally, the full proof of concept application will be finished by the end of term 1, however due to the large amount of other coursework deadlines, this can be extended to the beginning of term 2 and will not affect the rest of the timeline.

Following the proof of concept, the repeated strategy implementation phase begins where sprints occur in cycles of three weeks. The first week is dedicated to research and planning to find the best strategy for the particular feature that spring. For example, if the focus is investments, then research will be done into how investments affect a user's financial capability and what could be done to aid them. The second week is for development where this chosen approach for that strategy is implemented into the application. This will be done as a separate git branch in accordance with the git flow feature branch practice [23]. The third week is testing and bug fixing, before being merged into the main application and the cycle begins again. This is appropriate methodology as the requirements at this stage are less clear and quite open, so need to be flexible. Following this, the application will be tested more thoroughly and focus will shift to preparing for the presentation and writing the dissertation.

References

[1] FinCap UK. What is financial capability? *Financial Capability Strategy UK*, 2020. URL `https://www.fincap.org.uk/en/articles/what-is-financial-capability`.

[2] Latoya Irby. Best personal finance apps. *The balance*, June 2022. URL `https://www.thebalancemoney.com/best-personal-finance-apps-4170650`.

[3] Plaid. Plaid api documentation, 2022. URL `https://plaid.com/docs/`.

[4] OBIE. What is open banking. *OpenBanking.org*, Sept 2020. URL `https://www.openbanking.org.uk/what-is-open-banking/`. Accessed: 21/11/2022.

[5] Premila Webster Keith Neal. The 'cost of living crisis'. *Journal of Public Health*, 44(3):475–476, Aug 2022. ISSN 1741-3842. doi: 10.1093/pubmed/fdac080. URL `https://doi.org/10.1093/pubmed/fdac080`.

[6] D. Clark. Inflation rate for the consumer price index in the united kingdom in october 2022, by sector. *Statista*, Oct 2022. URL `https://www.statista.com/statistics/281724/consumer-price-index-cpi-united-kingdom/`. Accessed: 21/11/2022.

[7] FCA + HM Treasury. Consumer explanations of "advice" and "guidance", March 2017. URL `https://www.fca.org.uk/publication/research/fawg-consumer-explanations-advice-guidance.pdf`. Accessed: 20/11/2022.

[8] Andrew Craig. #53 - financial advice in the uk. *Plain English Finance*, March 2019. URL `https://plainenglishfinance.co.uk/opinion/financial-advice-in-the-uk`. Accessed: 20/11/2022.

[9] Thomas Brock Brian J. Block. Do you dare sue your broker? *Investopedia*, May 2022. URL `https://www.investopedia.com/articles/investing/031113/do-you-dare-sue-your-broker.asp`. Accessed: 20/11/2022.

[10] Ben Schwarz. Next.js performance: Making a fast framework even faster. *Calibre*, Dec 2021. URL `https://calibreapp.com/blog/nextjs-performance/`. Accessed: 22/11/2022.

[11] Bartosz Szczecinski. What's server side rendering and do i need it? *Medium*, Sept 2018. URL `https://medium.com/baphemot/whats-server-side-rendering-and-do-i-need-it-cb42dc059b38`. Accessed: 22/11/2022.

[12] Anna Fitzgerald. Tailwind css: What it is, why use it & examples. *Hubspot*, May 2022. URL `https://blog.hubspot.com/website/what-is-tailwind-css`. Accessed: 22/11/2022.

[13] Todd Kerpelman. Plaid academy: Getting started with the plaid api. YouTube, Feb 2022. URL `https://www.youtube.com/watch?v=sGBvKDGgPjc`.

[14] Alim Arslan Kaya. Here's why you don't have to code your own backends anymore. *Medium*, July 2022. URL `https://javascript.plainenglish.io/you-dont-have-to-code-your-own-backends-anymore-try-pocketbase-instead-70924fe45040`. Part of the JavaScript in Plain English series, Accessed: 22/11/2022.

[15] pocketbase. Pocketbase js sdk. GitHub, 2022. URL `https://github.com/pocketbase/js-sdk`. Accessed: 22/11/2022.

[16] Stephen C Hansen, David T Otley, and Wim A Van der Stede. Practice developments in budgeting: an overview and research perspective. *Journal of management accounting research*, 15(1):95–116, 2003.

[17] Jae K Shim, Joel G Siegel, and Allison I Shim. *Budgeting basics and beyond*, volume 574. John Wiley & Sons, 2011.

[18] Aaron B Wildavsky. *Budgeting: a comparative theory of the budgeting process*. Transaction Publishers, 1986.

[19] n26. The 50/30/20 rule: how to budget your money more efficiently. *N26 Blog*, Aug 2022. URL `https://n26.com/en-eu/blog/50-30-20-rule`. Accessed: 22/11/2022.

[20] Elizabeth Warren and Amelia Warren Tyagi. *All your worth: The ultimate lifetime money plan*. Simon and Schuster, 2005.

[21] Eric Whiteside. The 50/30/20 budget rule explained with examples. *Investopedia*, Sept 2022. URL `https://www.investopedia.com/ask/answers/022916/what-502030-budget-rule.asp`. Accessed: 22/11/2022.

[22] Team Starling. The 50/30/20 rule: Is it realistic in a cost of living crisis? *Starling Bank*, Sept 2022. URL `https://www.starlingbank.com/blog/50-30-20-budgeting-rule-is-it-realistic-in-a-cost-of-living-crisis/`. Accessed: 22/11/2022.

[23] Vincent Driessen. A successful git branching model, Jan 2010. URL `https://nvie.com/posts/a-successful-git-branching-model/`.

A Project Specification

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 the cost of household essentials increasing faster than incomes [1].

The concept of financial capability is the combination of financial knowledge, skills, attitudes, and confidence that lead to positive financial behaviours and money management decisions that fit the circumstances of an individual's life [2]. It gives people the power to make the most of their money and improve their lives [3].

This project aims 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

The existing software can be categorised into three sections, all of which have their problems.

Firstly, there are mobile banking apps, such as Revolut and Monzo, which track customers' spending within those specific accounts [4] [5]. They often struggle to, or don't even at all, integrate across other banks to have a singular centralised view and force users to create bank accounts with them to use. In addition, users must download a mobile app, which they are less likely to trust with their personal data, rather than just accessing a website [6].

Then there are the websites which allow users to sync data with their banks. After signing up, users allow the website to access their bank accounts, which then very slowly imports all their data. Navigating on these is unintuitive and eventually, after a user would find a service which doesn't cost a fortune, the obstructive adverts will pop up and block them from even using it [7]. In addition, some sites are known to not have enough features to make them worth using, and others have too many useless features which don't actually help, and instead overwhelm the user.

The third category is the websites or apps which do not let users integrate their accounts. These expect users to manually add their banking information and then they perform analysis to provide suggestions on how to save money. They are barely better than an Excel spreadsheet and are by far the worst of the three categories.

Ideally, the software will solve most of the problems with the existing solutions, incorporate the best features they have into one, and, have additional functionalities which are deemed to help improve money management found from the 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 [8], enabling the solution to obtain information from all bank accounts that a user gives permission to access, and compiles it into 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 [9]
- A budgeting strategy must be implemented [10]
- A savings strategy must be implemented [11]

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 users cannot produce errors
- The application must be easy to maintain and extend
- The website must be visually appealing
- The software must be scalable to handle a large number of users
- The UI 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
- A categorising bills strategy may be implemented
- 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 [12]; the backend will be written in JavaScript with Express, and use a Mongo database to store user data [13]. 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 the plaid servers to process the request and return 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 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 so that the software has methods, with evidence to support these, that benefit a user in taking control of their finances; not just a clone of the existing tools. A trivial example is that people often have several bank accounts, each for different purposes. In order to properly view all the transactions and determine where money could be saved, they ought to be collated into one central view [14]. Further 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 [15]. 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, followed by methodical progression through the stages. Moreover, it eases time management as set lengths of time can be accurately estimated for each part, helping ensure the project's objectives are reached.

During the repeated strategy development phase, an agile approach will be adopted. This synergises with the intent 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's planning, designing, developing and testing sprints [15]. At each cycle, the software should be ready to be distributed so a working version is always ready.

Throughout development, git will be used as version control to manage changes and features effectively [16]. If a mistake is ever made, reverting to a working version can be done quickly. The Git flow practice will also be followed, using main, release, development and feature branches, along with frequent merges to keep the repository clean and professional [17]. Finally, GitHub will be used to store a remote version of the repository and also aid documentation [18].

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, as this follows the test-driven development style to help break down the problem into smaller manageable chunks, as well as improve code quality in general [19]. 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 tester will be asked to use that strategy and see if it works as expected. A list of steps will not be given in an attempt to also test 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 experiment with the software as they would normally, and report any issues they find [20]. 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 practices for data security will have to be followed. This includes enforcing a long and complex password for users to avoid unauthorised access to accounts [21], but also storing user's passwords and data in a fashion which complies with the GDPR [22].

As a result of using the Plaid API, no extremely sensitive banking information will be stored locally; All the financial data is handled by the Plaid service itself. When the data is displayed to the user, they can be sure that it is secured and accurate, so long as the correct flow that is described in the Plaid documentation is followed during development [23]. 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 several asterisks followed by only the last few digits in plaintext, in an attempt 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 [24].

References

[1] Peter Hourston. Cost of living crisis. *The Institute for Government*, Aug 2022. URL `https://www.instituteforgovernment.org.uk/explainers/cost-living-crisis`.

[2] Russell Research. What is financial capability? *Financial Capabilty Research Australia*, 2022. URL `https://www.financialcapability.gov.au/strategy/part2`.

[3] FinCap UK. What is financial capability? *Financial Capability Strategy UK*, 2020. URL `https://www.fincap.org.uk/en/articles/what-is-financial-capability`.

[4] Revolut. Budgetting and analytics. Advertising Page, 2022. URL `https://www.revolut.com/best-budget-planner/`.

[5] Richard Cook. How to budget with monzo. *Monzo*, Aug 2018. URL `https://monzo.com/blog/2018/08/21/budget-with-monzo`.

[6] Anton Diduh. Mobile app vs. mobile website. *Cleveroad*, Sept 2022. URL `https://www.cleveroad.com/blog/mobile-app-vs-mobile-website/`.

[7] Latoya Irby. Best personal finance apps. *The balance*, June 2022. URL `https://www.thebalancemoney.com/best-personal-finance-apps-4170650`.

[8] Plaid. Plaid api documentation, 2022. URL `https://plaid.com/docs/`.

[9] Phil Town. *Rule #1: The Simple Strategy for Successful Investing in Only 15 Minutes a Week*. Random House, 2010. ISBN 9781409060048.

[10] Jae K. et al. *Budgetting Basics and Beyond*. John Wiley & Sons, 2011.

[11] Bernard DĊoleman. *Money How to save it spend it and make it*. Elsevier Science, Jun 2016. ISBN 9781483136394.

[12] tarunsinghwap7. Nextjs vs. reactjs, Aug 2022. URL `https://www.geeksforgeeks.org/nextjs-vs-reactjs-which-one-to-choose/`.

[13] MongoDB. Mongodb documentation, 2022. URL `https://www.mongodb.com/docs/`.

[14] David Weliver & Lauren Graves. How to use multiple bank accounts for budgetting. *moneyunder30*, Oct 2022. URL `https://www.moneyunder30.com/multiple-bank-accounts-to-control-spending`.

[15] Mike McCormick. Waterfall vs. agile methodology. *MPCS*, 3, 2012.

[16] Scott Chacon. About git + advantages, 2022. URL `https://git-scm.com/about`.

[17] Vincent Driessen. A successful git branching model, Jan 2010. URL `https://nvie.com/posts/a-successful-git-branching-model/`.

[18] GitHub. Github documentation, 2022. URL `https://docs.github.com/en`.

[19] Kent Beck. *Test-driven development by example*. Addison-Wesley Professional, 2003.

[20] Kuldeep Rana. Use case testing the complete guide. *Art of Testing*, Apr 2021. URL `https://artoftesting.com/use-case-testing`.

[21] Richard Shay et al. Encountering stronger password requirements. In *Proceedings of the sixth symposium on usable privacy and security*, pages 1–20, 2010.

[22] Paul Voigt et al. The eu general data protection regulation. *A Practical Guide*, 10(3152676):10–5555, 2017.

[23] Todd Kerpelman. Plaid academy: Getting started with the plaid api. YouTube, Feb 2022. URL `https://www.youtube.com/watch?v=sGBvKDGgPjc`.

[24] FCA. Understanding advice and guidance. *Financial Conduct Authority*, Feb 2018. URL `https://www.fca.org.uk/consumers/understanding-advice-guidance-investments`.