Personal Finance Tracker
Final Report

South East Technological University Carlow
Software Development

Name: Matthew Ufumeli
Student no: C00273575
Supervisor: Chris Staff
Date: 28/04/2025

Abstract

The myFi web application is designed to simplify personal finance management by providing users with an intuitive platform to track income, expenses, and budgets. Developed as a Final Year Project, myFi integrates real-time bank data through the Moneyhub API[3], automates transaction categorization using rule-based and AI-driven methods (OpenAI)[4], and offers budget tracking and conversational insights via an AI chatbot.

Built with a React front end[5], FastAPI[6] back end, and Firestore database[1][2], the application ensures a responsive and scalable user experience. Comprehensive testing validated its reliability across authentication, data integration, and user interface functionalities.

Despite time constraints limiting features like notifications and custom machine learning, myFi successfully empowers users to monitor their finances, promoting financial literacy and discipline. The project demonstrates proficiency in full-stack development, API integration, and user-centric design, with potential for future enhancements like mobile support and advanced analytics. MyFi addresses the growing need for accessible financial tools, offering significant value to users seeking to manage their finances effectively.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, whose guidance was instrumental in shaping this project. Their support with early documentation and clarifying the core features of myFi provided a strong foundation for development, ensuring the project aligned with its objectives. I am also deeply thankful to South East Technological University (SETU) Carlow for fostering an environment that encouraged innovation and learning. The resources, facilities, and academic support at SETU Carlow were crucial in equipping me with the technical skills and confidence to tackle this ambitious project.

Additionally, I appreciate the encouragement from my peers and family, whose support kept me motivated throughout the development process. This project would not have been possible without these contributions, and I am grateful for the opportunity to grow as a developer through this experience.

Table of Contents

Abstract **2**
Acknowledgements **3**
Table of Tables and Figures **5**
1. Introduction **6**
2. Screenshots of App **7**
3. General Issues **13**
3.1 Problems Encountered and How They Were Resolved 13
4. Achievements **14**
4.1 Secure User Authentication and Data Management 14
4.2 Real-Time Bank Account Integration 14
4.3 Automated Transaction Categorization 14
4.4 Budget Tracking and Monitoring 15
4.5 AI-Powered Financial Chatbot 15
4.6 Responsive and Intuitive User Interface 15
4.7 Scalable Backend Infrastructure 15
4.8 Successful End-to-End Integration 15
5. What Was Not Achieved **16**
5.1 Notification System 16
5.2 User Profile and Settings 16
5.3 Machine Learning for Transaction Categorization 16
6. Learnings **17**
6.1 Mastering Full-Stack Development 17
6.2 Optimizing User Experience 17
6.3 Project Management and Prioritization 17
6.4 Managing Third-Party API Dependencies 17
7. What I Would Do Differently **18**
8. User Impact and Benefits **19**
9. Technical Issues **20**
10. Differences from Earlier Design and Additional Research **21**
11.Screenshots of Data Structures **24**
12. Testing Used to Assess Reliability **26**
Unit Testing 26
Integration Testing 26
Error Handling Testing 26
User Interface Testing 26
13. System Architecture **27**
13.1 Front End 27
13.2 Back End 27

13.3 Database ... 28
13.4 APIs ... 28
14. Security and Privacy Considerations ... 29
15. Future Work ... 31
16. Conclusion ... 31
17. References ... 32
Declaration ... 33

Table of Tables and Figures

Figure 0.1: Dashboard screen ... 8
Figure 0.2: CSV Upload screen ... 8
Figure 0.3: Budgets screen ... 9
Figure 0.4: Reports screen ... 10
Figure 0.5: Connect Bank screen ... 11
Figure 0.6: Sign In screen ... 12
Figure 0.7: Budget screen mock design ... 20
Figure 1.1: Final Budget screen design ... 20
Figure 0.8: CSV upload screen mock design ... 20
Figure 0.9: Final CSV upload screen design ... 21
Figure 1.0: Reports screen mock design ... 22
Figure 1.2: Reports screen mock design ... 23
Figure 1.3: Users Data structure ... 23
Figure 1.4: Budgets Data structure ... 24
Figure 1.5: Chat messages Data structure ... 24
Figure 1.6: Transactions Data structure ... 24
Figure 1.7: Bank Accounts Data structure ... 24
Figure 1.8: Bank Accounts Data structure ... 26

1. Introduction

The "myFi" project is a full-stack web application designed to empower users to manage their personal finances effectively by integrating real-time bank data, transaction tracking, budget management, and AI-driven financial insights. In an era where financial literacy and proactive money management are increasingly vital, myFi addresses the need for an accessible, user-friendly platform that consolidates financial data and provides actionable insights. The application leverages modern technologies, including React for a dynamic frontend, FastAPI for a robust backend, Firebase for authentication and data storage, Moneyhub's API for bank account integration, and OpenAI for intelligent chatbot functionality. As my Final Year Project (FYP), myFi represents a significant undertaking to apply software engineering principles to a real-world problem, demonstrating my growth as a developer through the challenges faced and lessons learned.

Motivation

The primary motivation for myFi stems from the complexity individuals face in tracking their finances across multiple bank accounts and budgeting effectively. Many existing tools are either too simplistic, lacking real-time data integration, or overly complex, deterring non-technical users. myFi bridges this gap by offering a seamless interface to connect bank accounts via Moneyhub's OAuth 2.0 API[7], categorize transactions automatically, set and monitor budgets, and interact with a chatbot for personalized financial queries. The project's scope includes secure user authentication, real-time data syncing with Moneyhub, Firestore-based transaction and budget storage, and a responsive React frontend that updates dynamically based on user actions.

Report Objective

This report documents the development process, reflecting on both successes and setbacks to showcase my learning journey. It is structured to address the FYP requirements, covering general issues (problems, achievements, lessons learned, and user impact), technical details (design changes, modules, data structures, testing, architecture, and security), future work, and a conclusion. Appendices include code listings, a glossary, and supporting visuals, with a bibliography adhering to Harvard referencing standards. Through this report, I aim to demonstrate that I have become a more capable software developer, capable of tackling complex projects and learning from mistakes to deliver a valuable product.

2. Screenshots of App

Dashboard

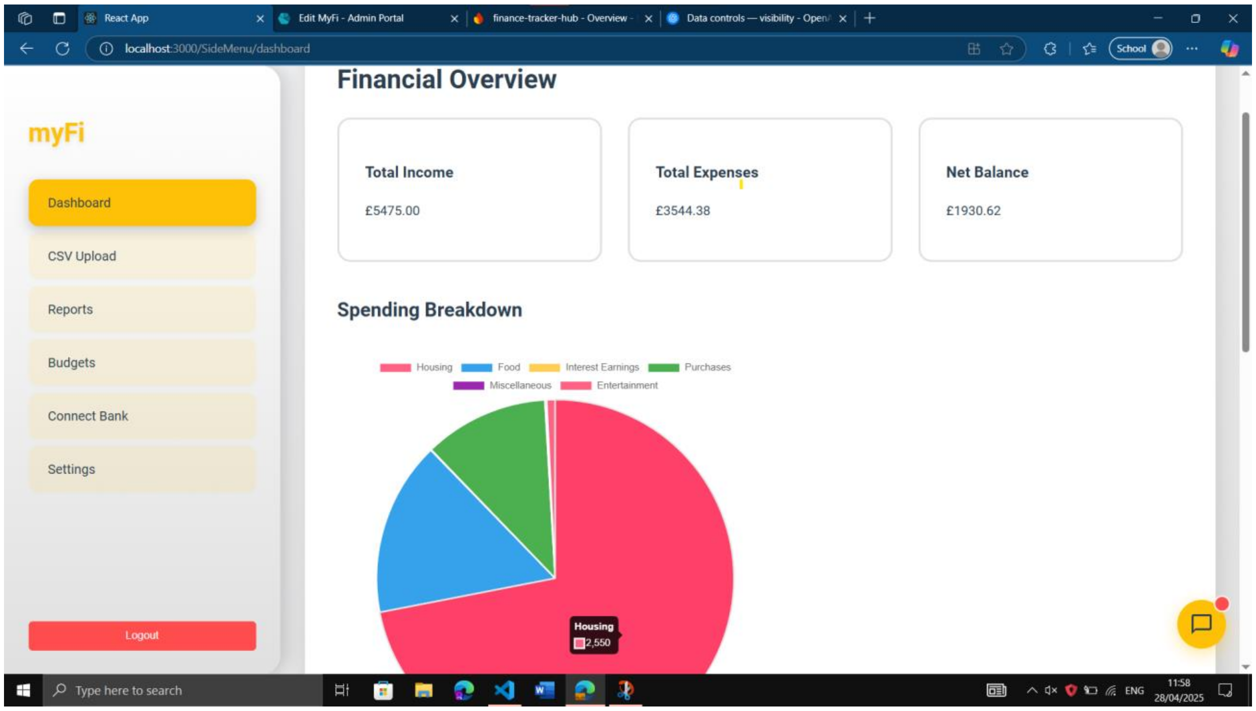


Figure 0.1: Dashboard screen

The Dashboard is your central hub for a quick overview of your finances. It displays key statistics like total income, expenses, and savings for the current month, along with a list of recent transactions. You'll also find personalized financial insights, such as your top spending category, to help you understand your habits at a glance. This screen is perfect for staying informed about your financial health without diving into details.

CSV Upload

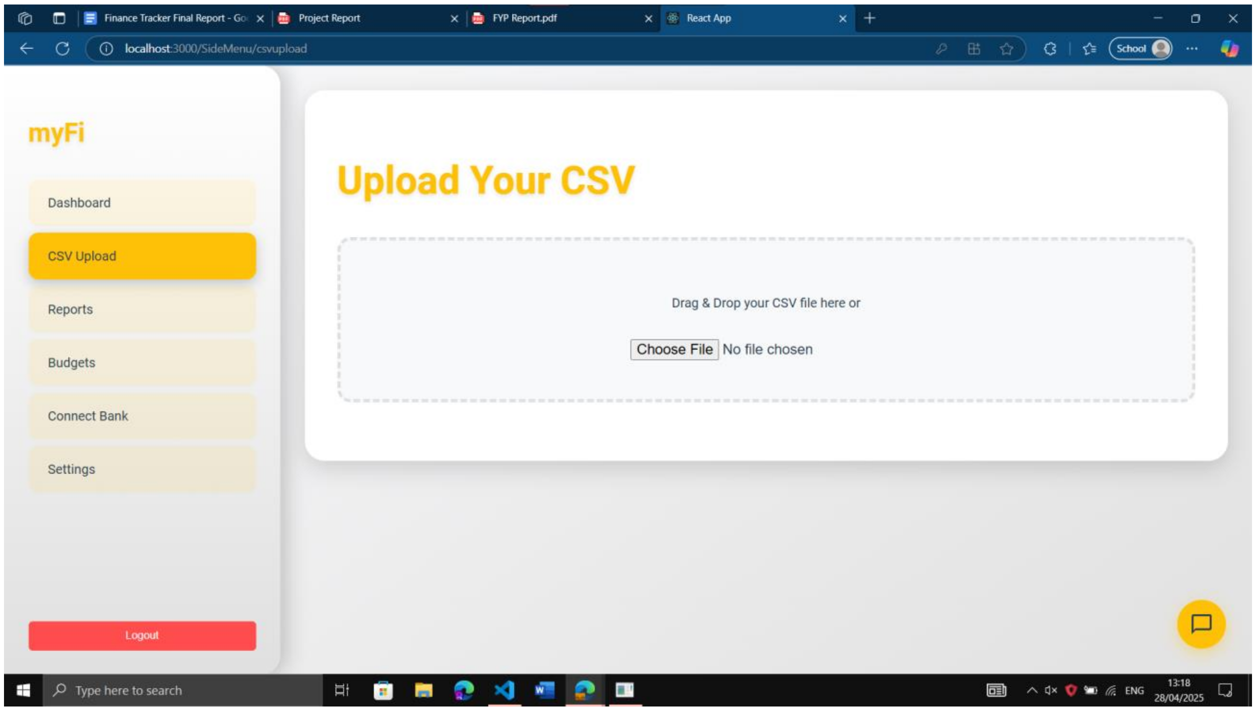
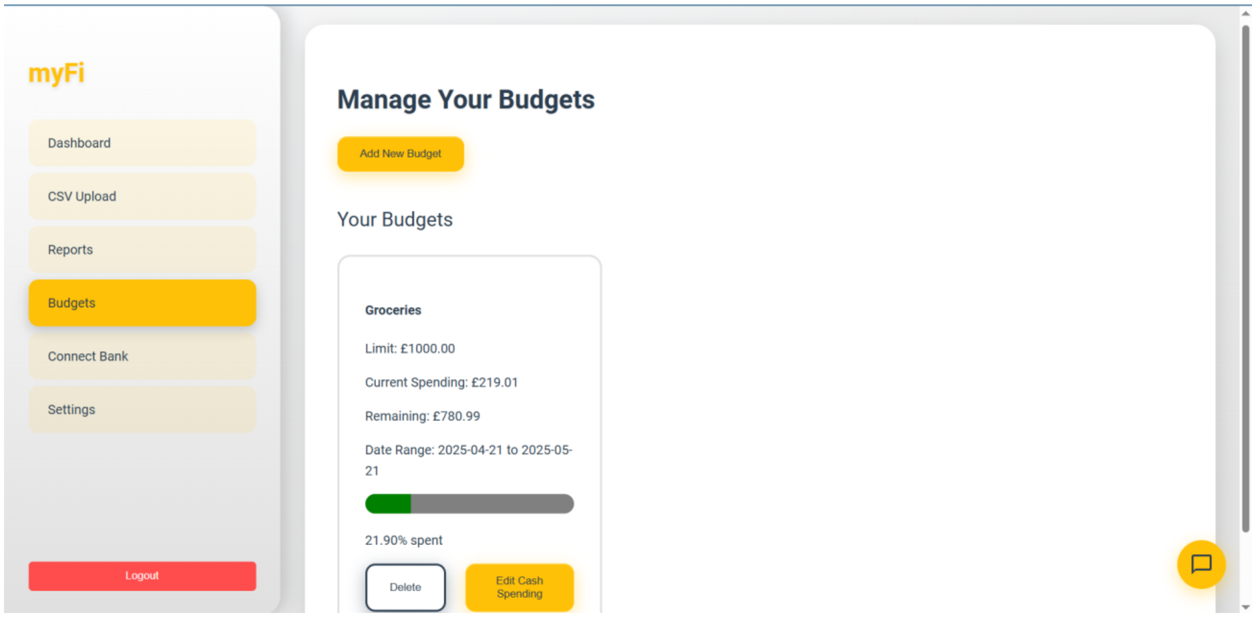


Figure 0.2: CSV Upload screen

The CSV Upload screen lets you import transaction data from bank statements in CSV format. You can drag and drop a file or select it manually, then map the file's columns (e.g., date, amount, description) to the app's fields. Once uploaded, transactions are categorized and added to your records, making it easy to track spending without manual entry. This feature is ideal for users who prefer managing data from multiple sources.

Budgets



Manage Your Budgets

Your Budgets

Groceries

Limit: £1000.00

Current Spending: £219.01

Remaining: £780.99

Date Range: 2025-04-21 to 2025-05-21

21.90% spent

Figure 0.3: Budgets screen

The Budgets screen allows you to create, manage, and track budgets for categories like groceries or entertainment. You can set spending limits, define start and end dates, and monitor progress with visual progress bars that highlight when you're nearing or exceeding your budget. You can also update cash spending manually or delete budgets as needed. This feature helps you plan and control your expenses effectively.

Reports

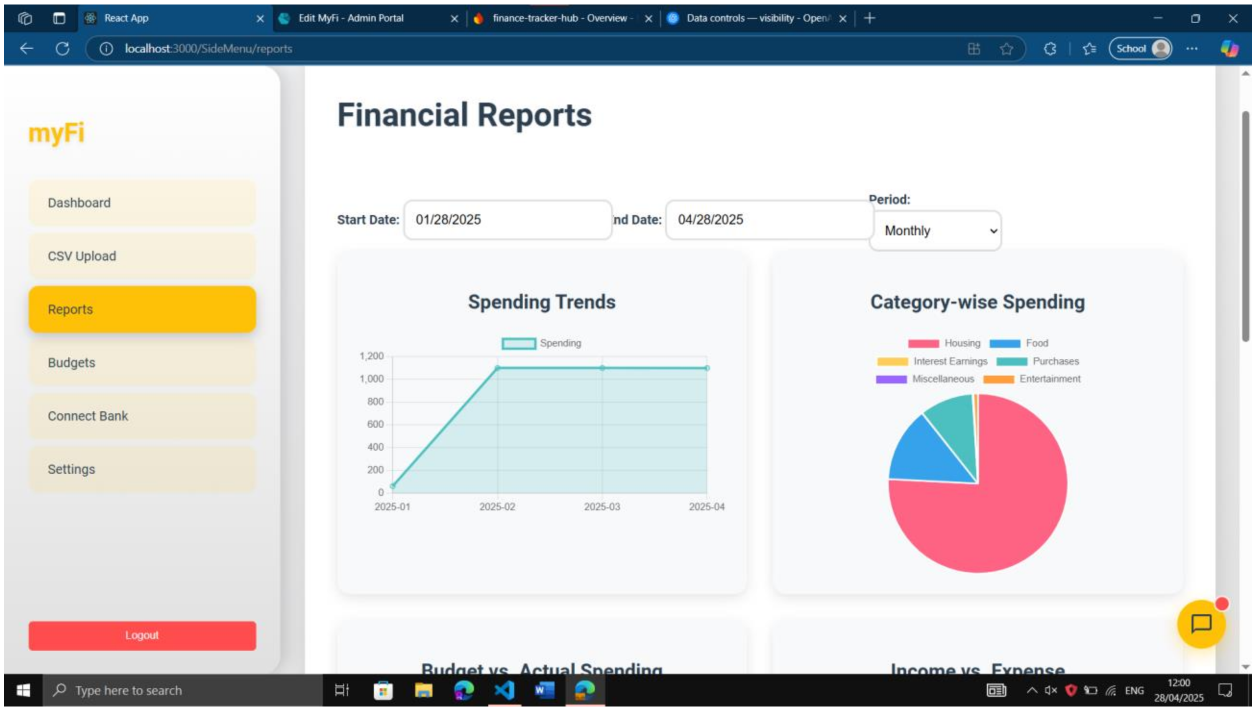
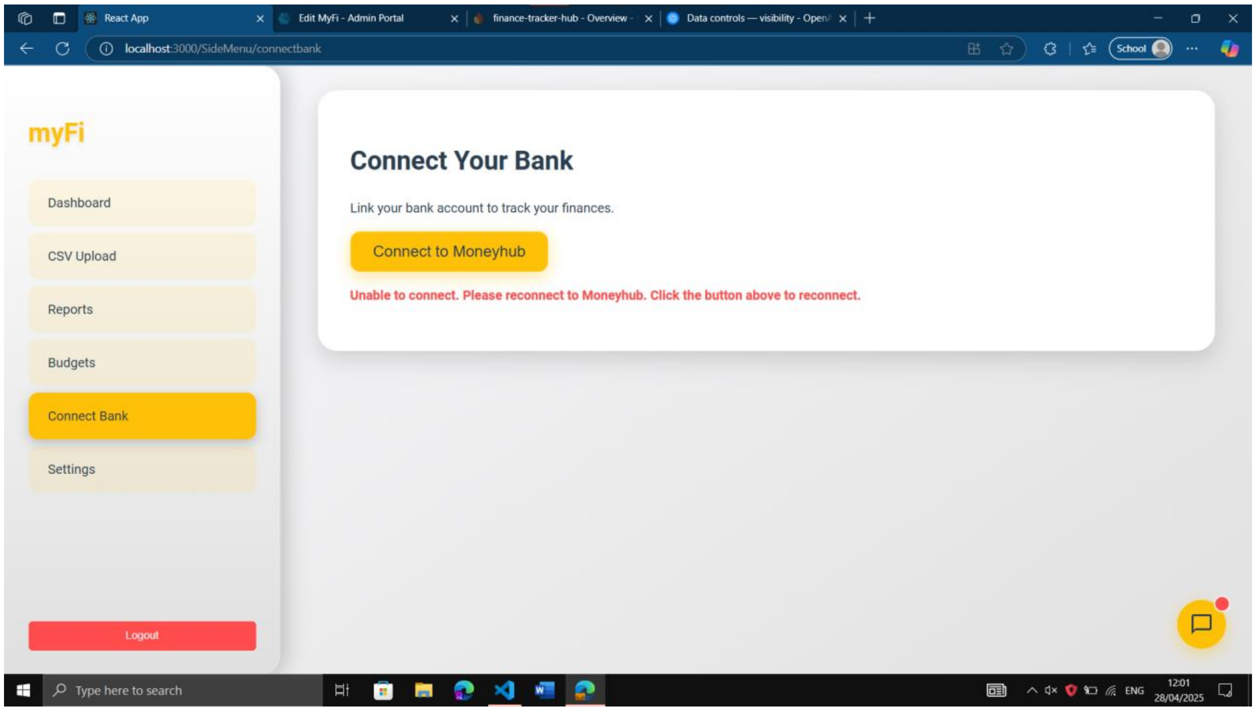


Figure 0.4: Reports screen

The Reports screen provides detailed insights into your financial activity through interactive charts and tables. You can view spending trends over time, category-wise spending, budget vs. actual spending, and income vs. expense breakdowns, with filters for date ranges and weekly or monthly views. A transaction history table lets you sort and search past transactions. This screen helps you analyze patterns and make informed financial decisions.

Connect Bank



Connect Your Bank

Link your bank account to track your finances.

Unable to connect. Please reconnect to Moneyhub. Click the button above to reconnect.

Figure 0.5: Connect Bank screen

The ConnectBank screen enables you to link your bank accounts to myFi for real-time financial tracking. After securely connecting via a trusted financial service, the app automatically fetches and displays your account balances and transactions. You can refresh data or reconnect if needed, with clear feedback on connection status. This screen simplifies keeping your financial data up-to-date without manual input.

Sign in

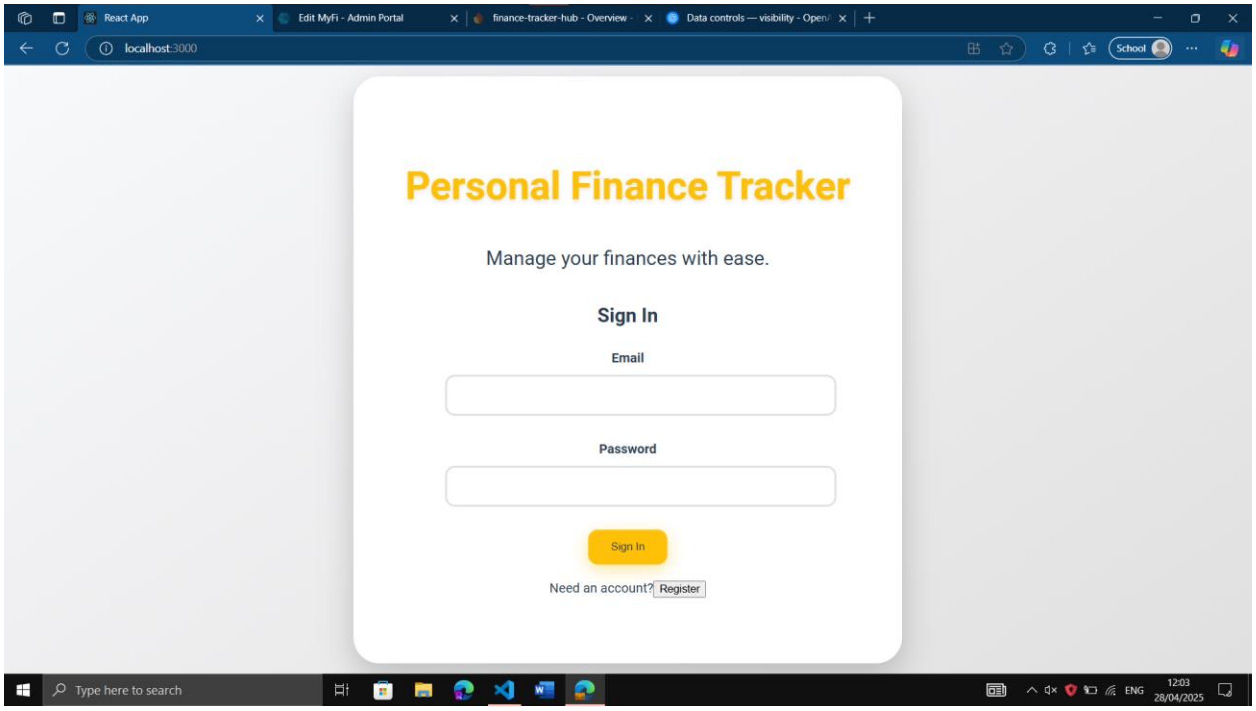


Figure 0.6: Sign In screen

The Sign In screen is your gateway to myFi, allowing secure access to your financial data. Simply enter your email and password to log in and reach the dashboard, where you can track expenses and budgets. If you encounter issues, clear error messages guide you. This screen ensures quick, safe entry, keeping your financial information protected while offering a seamless start to managing your money.

3. General Issues

The development of myFi, a personal finance tracking web application, was a complex endeavor that presented numerous challenges, opportunities for achievement, and valuable lessons. This section reflects on the problems I encountered, the milestones I reached, the goals I fell short of, the knowledge I gained, the changes I would make if starting anew, and the impact myFi has on its users. By addressing these aspects, I aim to demonstrate my growth as a software developer and my ability to learn from mistakes while delivering a functional product.

3.1 Problems Encountered and How They Were Resolved

As a first-time developer working with APIs, integrating the Moneyhub Open Banking API into myFi was both a challenging and rewarding experience that significantly shaped my technical growth.

- Having never worked with APIs before, I spent several weeks grappling with the complexities of Moneyhub's API, which required a steep learning curve to understand and implement correctly. This process highlighted common struggles that beginners, like myself, could face when integrating Moneyhub's Open Banking API into an application.

  The primary challenge was understanding OAuth 2.0, the authentication protocol Moneyhub uses to securely connect to bank accounts. As a beginner, the concept of managing access tokens, refresh tokens, and authorization codes was overwhelming. I struggled to set up the correct flow, where users are redirected to Moneyhub's authentication page, return with a code, and exchange it for tokens. For weeks, I encountered errors like invalid redirects or expired tokens because I didn't fully grasp the sequence or how to store and refresh tokens securely. Moneyhub's documentation, while detailed, assumed familiarity with OAuth terminology, which was a hurdle for a novice like myself.

  These struggles consumed weeks of trial and error, but through persistent research, testing, and community forums, I eventually implemented a robust integration. I learned to break down OAuth into manageable steps, use logging to diagnose errors, and implement retry logic for rate-limited requests. This experience taught me the importance of patience and thorough documentation reading, transforming me from an API novice to a developer confident in integrating third-party services.

Throughout the development process, I faced several significant challenges that tested my problem-solving skills and required creative solutions.

- Another major issue was ensuring that the application correctly displayed a user’s financial data after logging in, as there were instances where data from a previous user persisted. This was caused by improper handling of user sessions, which I resolved by implementing a robust mechanism to clear all user-specific data upon logout, ensuring each session started fresh.

4. Achievements

The myFi project successfully delivered a comprehensive personal finance tracking web application, integrating multiple technologies to provide users with powerful tools for managing their financial lives. The application combines secure user authentication, real-time bank data integration, automated transaction categorization, budget tracking, and an AI-powered chatbot to offer a seamless and intuitive user experience. Below, I outline the key achievements of the project, highlighting the functional components and their benefits to users.

4.1 Secure User Authentication and Data Management

I implemented a robust user authentication system that ensures secure access to the application. Users can sign up, log in, and log out with confidence, knowing their personal and financial data is protected. The system stores user profiles and financial data in a cloud-based NoSQL database, enabling real-time updates and scalability. This achievement provides a foundation for user trust and data integrity, critical for a financial application handling sensitive information.

4.2 Real-Time Bank Account Integration

A core feature of myFi is its ability to connect to users' bank accounts through a third-party financial API, allowing real-time access to account balances and transaction histories. I successfully integrated this API, enabling users to link their accounts securely via an OAuth-based authentication flow. Once connected, the application fetches and displays account details, such as balances and transaction records, providing users with an up-to-date view of their finances without manual input.

4.3 Automated Transaction Categorization

The application automatically categorizes transactions (e.g., "Food," "Transport," "Bills") based on their descriptions, streamlining financial tracking. I developed a rule-based system that analyzes transaction details and assigns appropriate categories, reducing the need for manual sorting. This feature enhances user efficiency, allowing them to quickly understand their spending patterns across various categories.

4.4 Budget Tracking and Monitoring

I implemented a budget management system that lets users create, update, and delete budgets for specific spending categories, such as groceries or entertainment. The system tracks expenses within each budget's date range and updates spending progress in real time. For example, if a user sets a £200 monthly budget for dining, the application calculates and displays how much they've spent, helping them stay within their limits. This achievement empowers users to plan and control their finances effectively.

4.5 AI-Powered Financial Chatbot

A standout feature is the AI-driven chatbot, which answers users' financial queries in a conversational manner. I integrated a large language model to process questions like "How much did I spend on groceries this month?" and provide accurate, friendly responses based on the user's transaction data. The chatbot uses pattern matching to identify query types and fetches relevant data from the database, enhancing user engagement and accessibility. This feature makes financial insights more approachable, especially for non-technical users.

4.6 Responsive and Intuitive User Interface

The frontend of myFi is a responsive web interface built with a modern JavaScript framework, ensuring compatibility across devices, from desktops to mobile phones. I designed an intuitive layout with clear navigation, allowing users to access bank connections, budgets, transactions, and the chatbot effortlessly. Visual feedback, such as loading indicators and error messages, improves usability, making the application user-friendly and professional.

4.7 Scalable Backend Infrastructure

The backend, developed with a high-performance Python framework, handles API requests efficiently and integrates with external services like the financial API and AI model. I implemented endpoints for authentication, data fetching, budget management, and chatbot interactions, ensuring low latency and reliability. The backend's scalability supports future growth, such as adding more users or features, without compromising performance.

4.8 Successful End-to-End Integration

I achieved seamless integration across the frontend, backend, database, and third-party services, creating a cohesive system. For instance, when a user connects their bank account, the backend securely stores authentication tokens, fetches transaction data, categorizes it, and updates budgets, while the frontend displays the results in real time.

5. What Was Not Achieved

Despite the successful delivery of myFi’s core functionalities, time constraints before the project deadline prevented the implementation of several planned features. Below, I outline the key components that were not achieved, explaining their intended purpose and the reasons for their exclusion.

5.1 Notification System

- **Purpose**: A notification system was planned to alert users about budget overspending, upcoming bill payments, or significant transactions via in-app messages or emails.
- **Reason**: Limited time prevented the integration of a reliable notification framework, as priority was given to core features like bank integration and budget tracking.

5.2 User Profile and Settings

- **Purpose**: A user profile page would have allowed users to customize preferences, such as currency, notification settings, or account details, enhancing personalization.
- **Reason**: Developing a comprehensive settings interface was deprioritized due to time constraints, focusing instead on functional financial tools.

5.3 Machine Learning for Transaction Categorization

- **Purpose**: Training a custom machine learning model to categorize transactions more accurately based on user-specific patterns was intended to improve automation.
- **Reason**: The complexity and time required for data collection, model training, and integration exceeded the project timeline, leading to reliance on rule-based and API-based categorization.

6. Learnings

The development of myFi, was a transformative experience that deepened my understanding of software development and project management. Through tackling complex integrations, resolving unexpected issues, and delivering a functional product, I acquired a range of technical skills, problem-solving strategies, and professional insights. Below, I outline the key learnings from the project, demonstrating how challenges and successes shaped me into a more capable developer.

6.1 Mastering Full-Stack Development

Building myFi required integrating a frontend, backend, database, and third-party APIs, teaching me the intricacies of full-stack development. I learned to create a responsive user interface that communicates seamlessly with a server, which in turn fetches and processes data from external financial services. This process enhanced my ability to design systems where each component—user interface, server logic, and data storage—works harmoniously, a critical skill for developing scalable applications.

6.2 Optimizing User Experience

Designing an intuitive and responsive user interface taught me the value of user-centric development. I learned to anticipate user needs, such as providing clear feedback during data loading or error states, and to ensure the application works across devices. Iterating on the interface based on testing feedback helped me understand how small design choices, like button placement or error message clarity, significantly impact user satisfaction.

6.3 Project Management and Prioritization

Managing a large-scale project taught me to prioritize features and allocate time effectively. Early in development, I underestimated the complexity of API integrations, leading to delays. This mistake prompted me to adopt an iterative approach, focusing on core functionality (e.g., bank connections, budget tracking) before adding advanced features like the chatbot. I learned to balance ambition with practicality, a key skill for delivering projects on time.

6.4 Managing Third-Party API Dependencies

Integrating a financial API to fetch bank data introduced challenges like rate limits, token expiration, and error handling. I learned to design robust systems that gracefully handle API failures, retry requests when appropriate, and cache data to reduce unnecessary calls. This experience underscored the importance of thoroughly understanding external service documentation and planning for their limitations in application design.

7. What I Would Do Differently

Reflecting back, several aspects of the development process could be improved to enhance efficiency, reliability, and user experience. These insights, gained through challenges and successes, highlight how I would approach a similar project differently to achieve better outcomes.

Improved Planning and Prioritization

- **Earlier Testing**: Implement automated testing from the start to catch issues like data display errors sooner.
- **Feature Scoping**: Prioritize core features (e.g., bank integration) earlier, delaying complex additions.

Enhanced System Design

- **Centralized State Management**: Use a robust state management tool to simplify user interface updates and avoid inconsistencies.
- **API Optimization**: Cache financial data locally to reduce API calls, improving performance and handling rate limits better.

Focus on User Experience

- **User Testing**: Conduct early user testing to refine the interface, ensuring intuitive navigation and clearer feedback.

8. User Impact and Benefits

The myFi application significantly enhances users’ ability to manage their finances by providing an intuitive, all-in-one platform for tracking income, expenses, and budgets. By integrating real-time bank data, automated transaction categorization, and budget monitoring, myFi saves users time and effort compared to manual tracking methods. The AI-powered chatbot further simplifies financial management, allowing users to ask questions like “How much did I spend on groceries?” and receive instant, friendly responses. This accessibility empowers users, including those with limited financial expertise, to gain clear insights into their spending habits and make informed decisions to achieve their financial goals.

Beyond convenience, myFi promotes financial discipline and awareness. The budget tracking feature helps users set and adhere to spending limits, while visual reports highlight trends, such as high spending in specific categories, encouraging better budgeting practices. The responsive interface ensures accessibility across devices, making financial oversight seamless at home or on the go. By offering these tools, myFi has the potential to improve users’ financial literacy and confidence, fostering long-term financial stability and reducing stress associated with money management.

9. Technical Issues

Developing myFi presented several technical challenges, primarily related to integrating third-party APIs, managing real-time data, and ensuring a responsive user interface. These issues were addressed to deliver a reliable application, though they required significant effort and adaptation.

- **API Integration Challenges**: Integrating the Moneyhub API for real-time bank data and the OpenAI API for transaction categorization and chatbot functionality posed significant hurdles. The Moneyhub API required complex OAuth authentication, and inconsistencies in bank data formats led to occasional transaction parsing errors. Similarly, the OpenAI API's rate limits and response variability caused delays in categorizing transactions accurately. These challenges extended development time, as extensive debugging and configuration adjustments were needed. To address them, I implemented fallback rule-based categorization and added error-handling mechanisms to display user-friendly messages when API calls failed, ensuring the application remained operational despite integration setbacks.

- **Web App Unresponsiveness**: The web application occasionally became unresponsive, with features like the dashboard or budget tracker stopping loading entirely. This issue typically occurred after saving changes to the backend file (`Main.py`), yet no error messages appeared, complicating diagnosis. The problem likely stemmed from disrupted connections between the React front end and FastAPI back end, possibly due to hot-reloading issues or database sync delays with Firestore. To mitigate this, I restarted the server and tested incrementally, though a permanent fix was not achieved due to time constraints.

10. Differences from Earlier Design and Additional Research

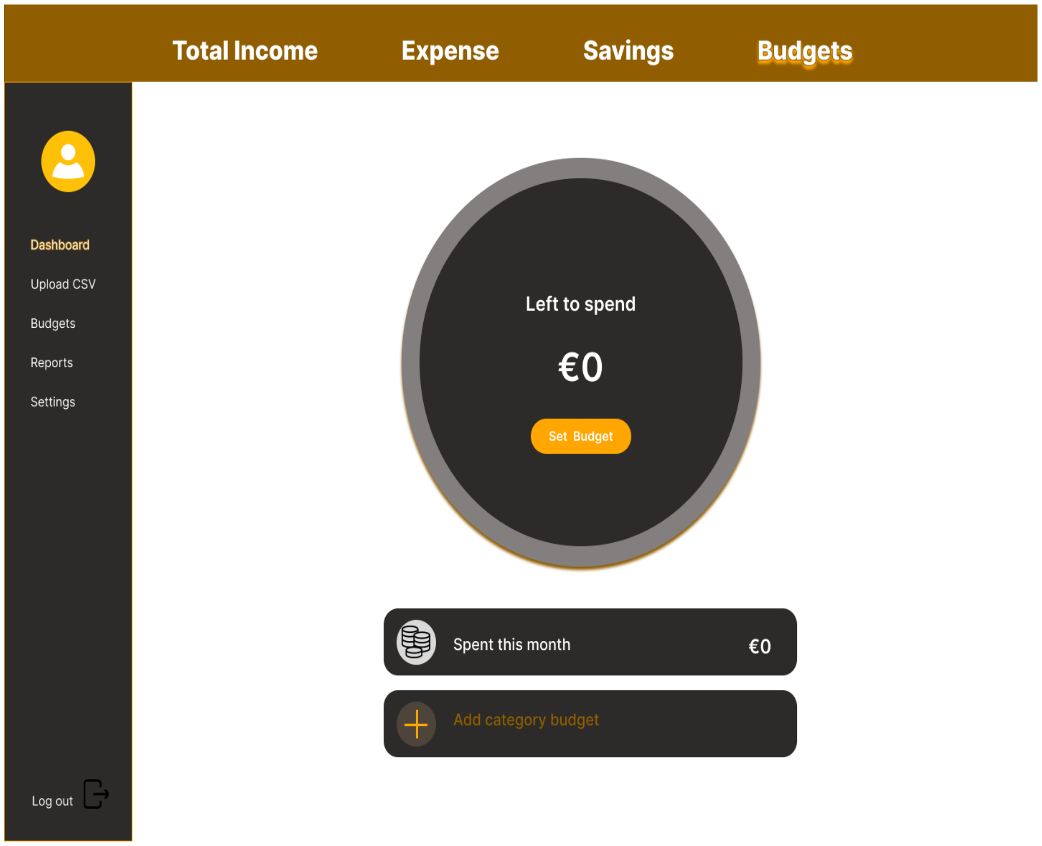
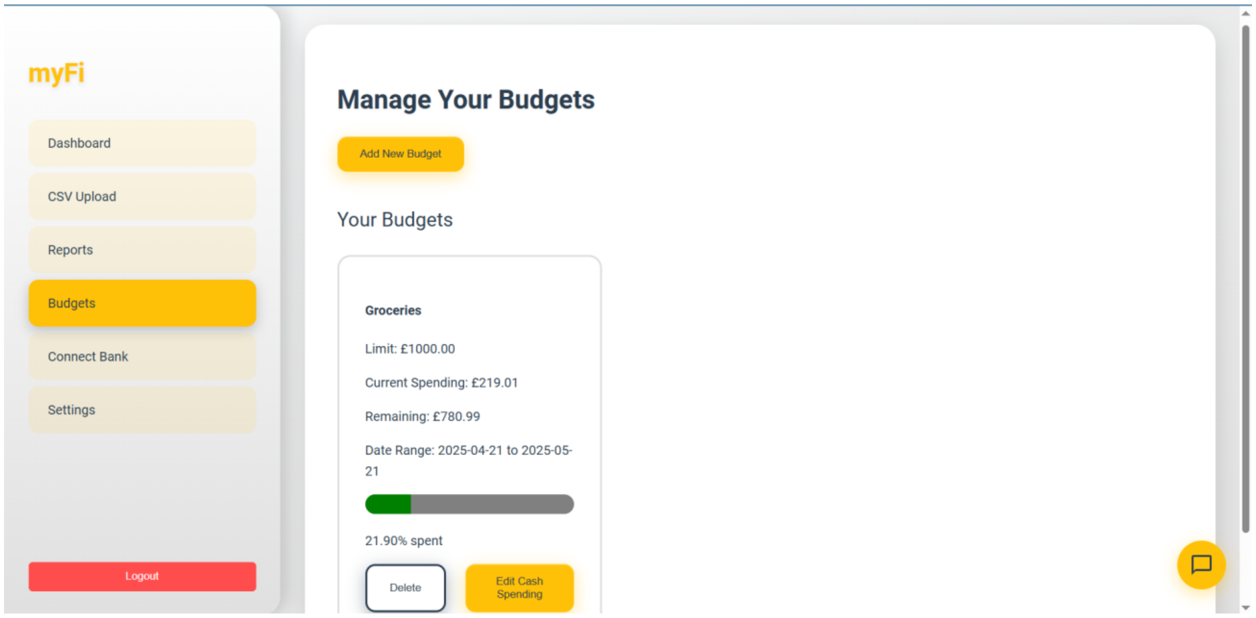


Figure 0.7: Budget screen mock design



Manage Your Budgets

Your Budgets

Groceries

Limit: £1000.00

Current Spending: £219.01

Remaining: £780.99

Date Range: 2025-04-21 to 2025-05-21

21.90% spent

Figure 1.1: Final Budget screen design

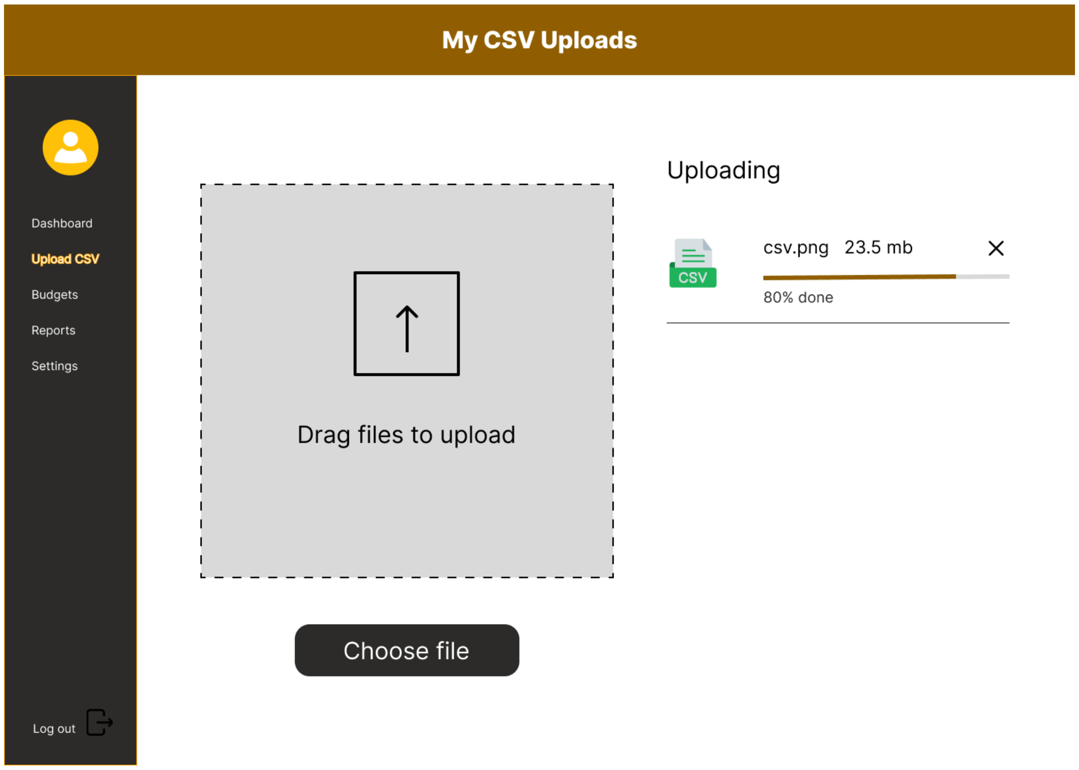


Figure 0.8: CSV upload screen mock design

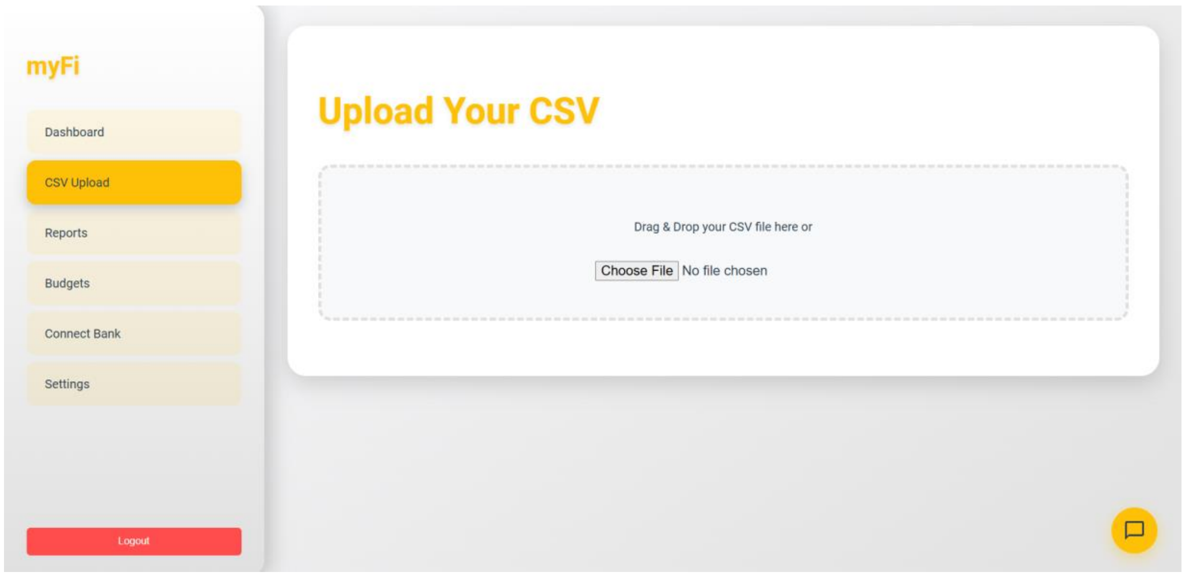


Figure 0.9: Final CSV upload screen design

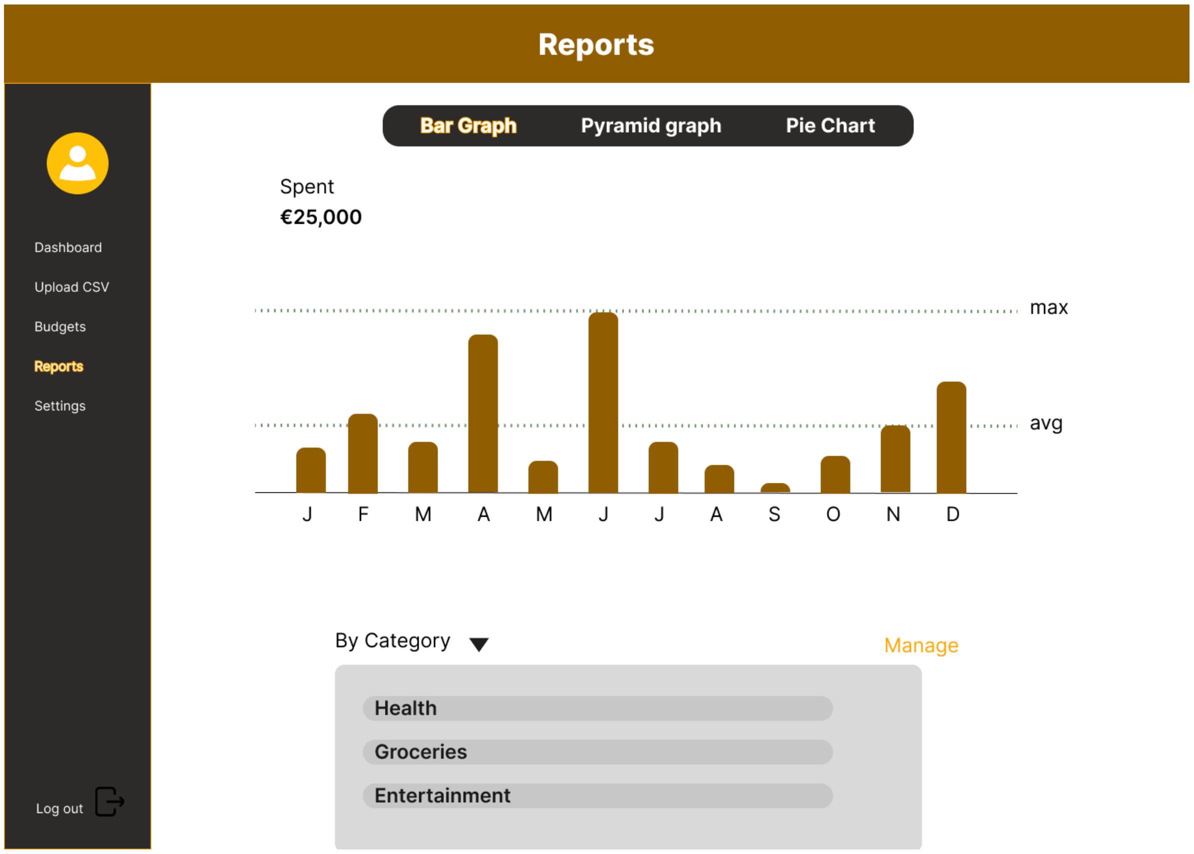


Figure 1.0: Reports screen mock design

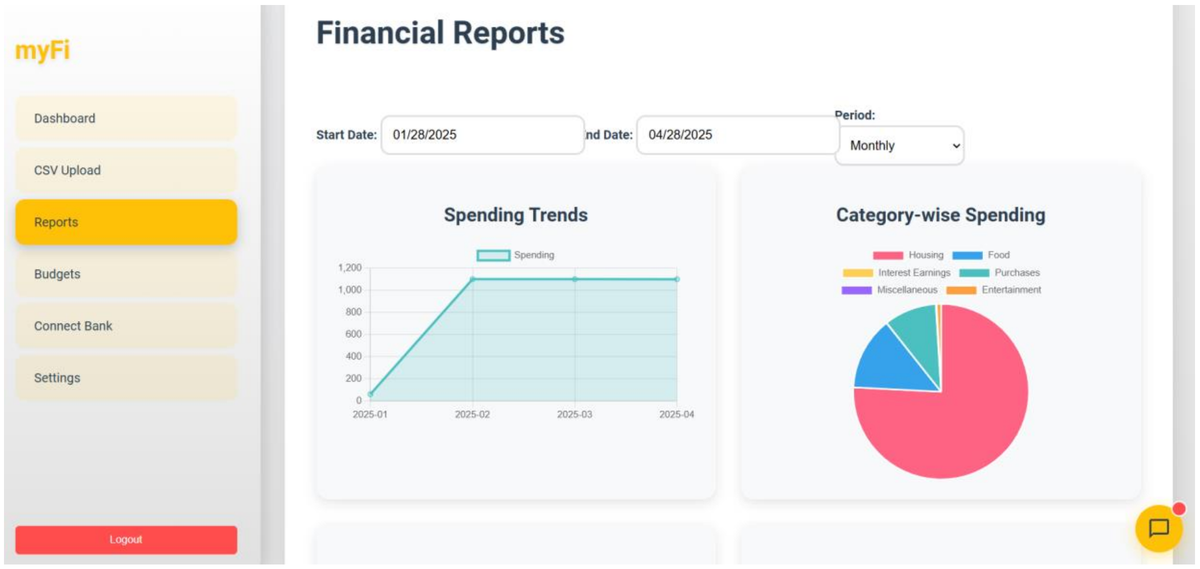
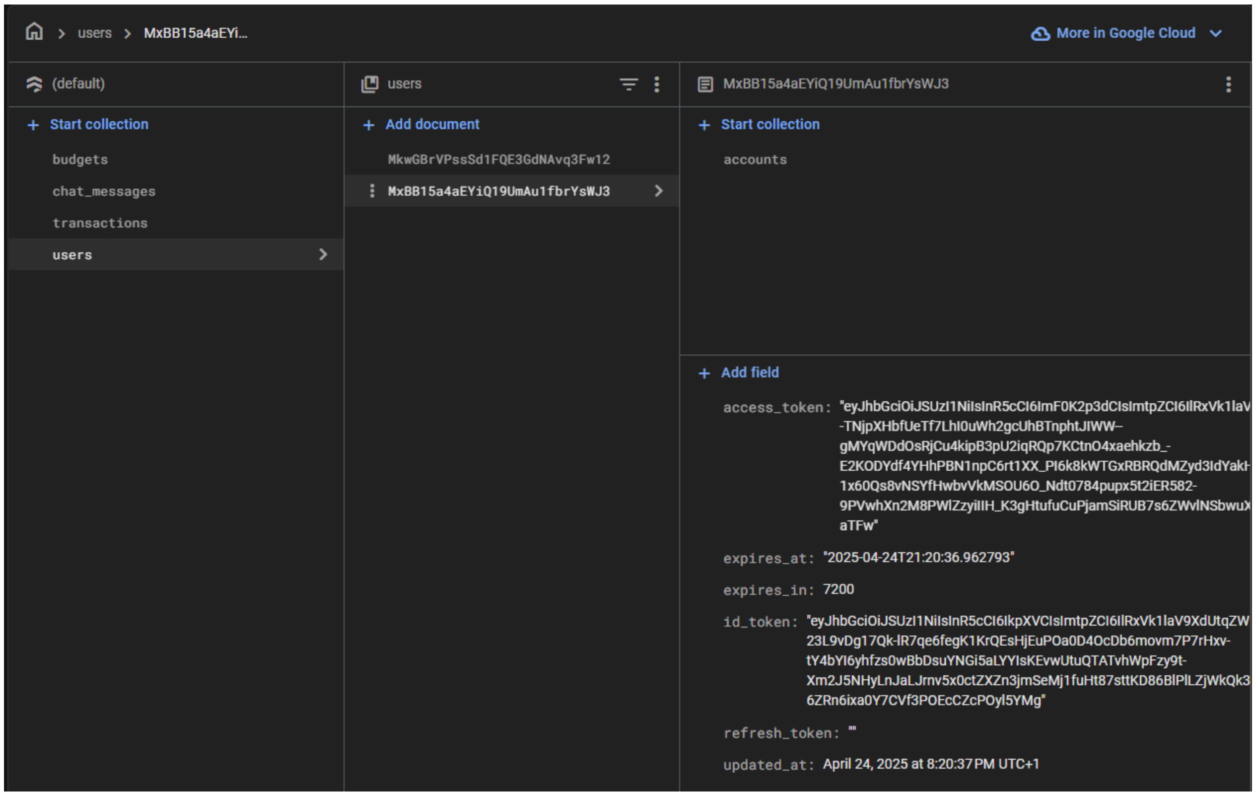


Figure 1.2: Reports screen mock design

11.Screenshots of Data Structures



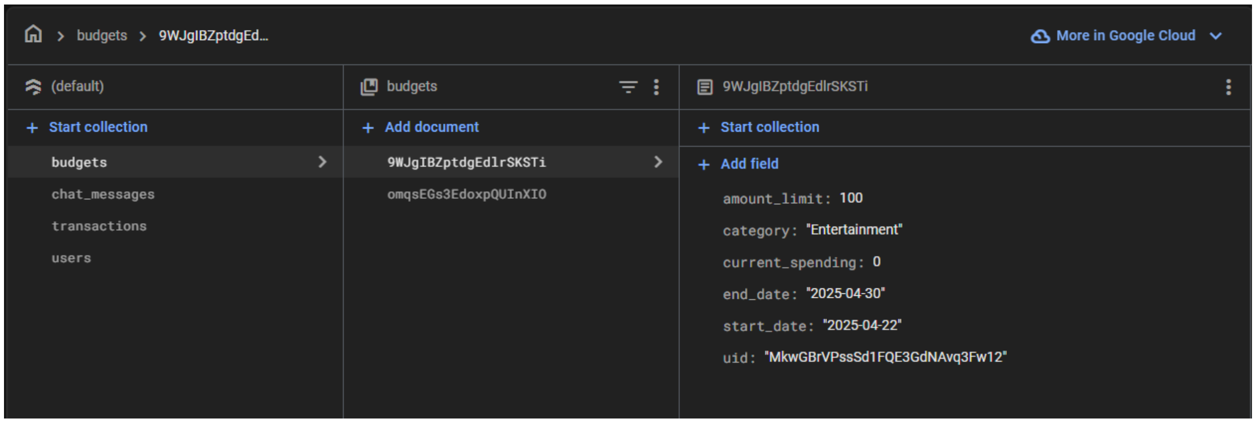
budgets
chat_messages
transactions
users

MkwGBrVPssSd1FQE3GdNAvq3Fw12
MxBB15a4aEYiQ19UmAu1fbrYsWJ3

accounts

```
access_token: "eyJhbGciOiJSUzI1NiIsInR5cCI6ImF0K2p3dCIsImtpZCI6IlRxVk1laV
              -TNjpXHbfUeTf7LhI0uWh2gcUhBTnphtJIWW–
              gMYqWDdOsRjCu4kipB3pU2iqRQp7KCtnO4xaehkzb_-
              E2KODYdf4YHhPBN1npC6rt1XX_PI6k8kWTGxRBRQdMZyd3IdYakH
              1x60Qs8vNSYfHwbvVkMSOU6O_Ndt0784pupx5t2iER582-
              9PVwhXn2M8PWlZzyiIIH_K3gHtufuCuPjamSiRUB7s6ZWvlNSbwuX
              aTFw"
expires_at: "2025-04-24T21:20:36.962793"
expires_in: 7200
id_token: "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6IlRxVk1laV9XdUtqZW
          23L9vDg17Qk-lR7qe6fegK1KrQEsHjEuPOa0D4OcDb6movm7P7rHxv-
          tY4bYI6yhfzs0wBbDsuYNGi5aLYYIsKEvwUtuQTATvhWpFzy9t-
          Xm2J5NHyLnJaLJrnv5x0ctZXZn3jmSeMj1fuHt87sttKD86BlPlLZjWkQk3
          6ZRn6ixa0Y7CVf3POEcCZcPOyl5YMg"
refresh_token: ""
updated_at: April 24, 2025 at 8:20:37 PM UTC+1
```

Figure 1.3: Users Data structure

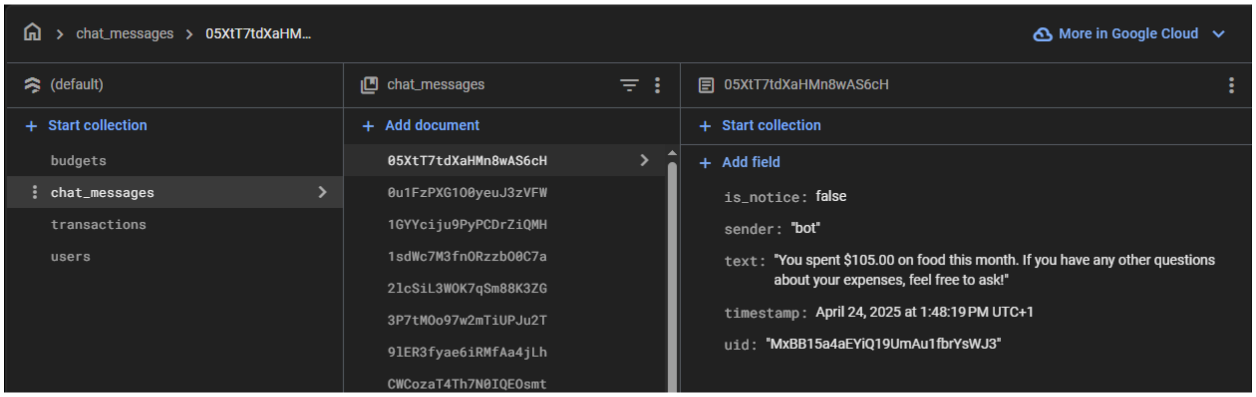


budgets
chat_messages
transactions
users

9WJgIBZptdgEdlrSKSTi
omqsEGs3EdoxpQUInXIO

```
amount_limit: 100
category: "Entertainment"
current_spending: 0
end_date: "2025-04-30"
start_date: "2025-04-22"
uid: "MkwGBrVPssSd1FQE3GdNAvq3Fw12"
```

Figure 1.4: Budgets Data structure



chat_messages > 05XtT7tdXaHM...

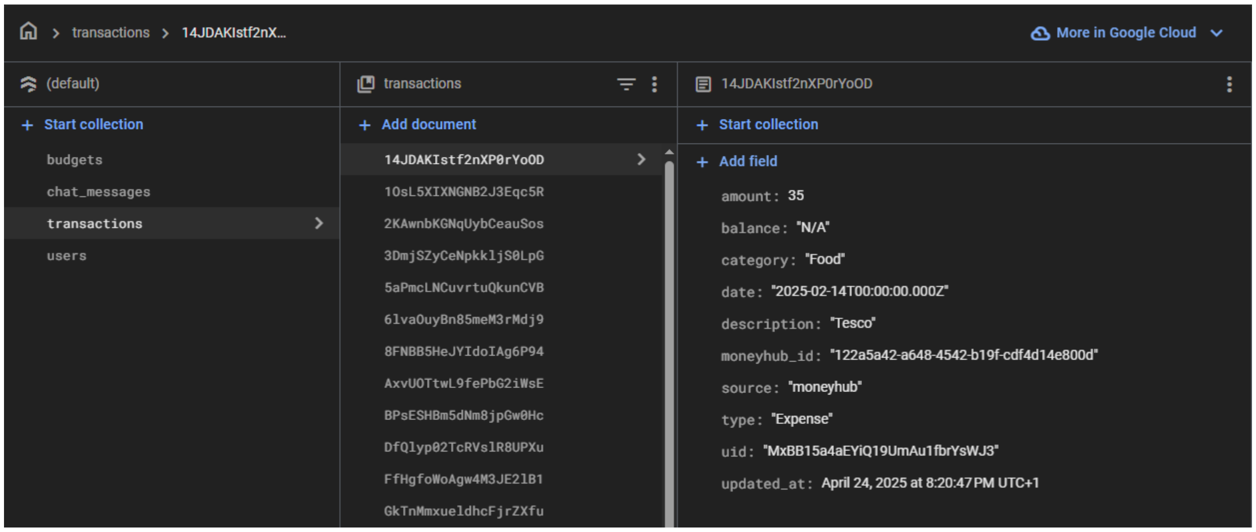
(default)
- budgets
- chat_messages
- transactions
- users

chat_messages
- 05XtT7tdXaHMn8wAS6cH
- 0u1FzPXG1O0yeuJ3zVFW
- 1GYYciju9PyPCDrZiQMH
- 1sdWc7M3fnORzzbO0C7a
- 2lcSiL3WOK7qSm88K3ZG
- 3P7tMOo97w2mTiUPJu2T
- 9lER3fyae6iRMfAa4jLh
- CWCozaT4Th7N0IQEOsmt

05XtT7tdXaHMn8wAS6cH

```
is_notice: false
sender: "bot"
text: "You spent $105.00 on food this month. If you have any other questions
      about your expenses, feel free to ask!"
timestamp: April 24, 2025 at 1:48:19 PM UTC+1
uid: "MxBB15a4aEYiQ19UmAu1fbrYsWJ3"
```

Figure 1.5: Chat messages Data structure



transactions > 14JDAKIstf2nX...

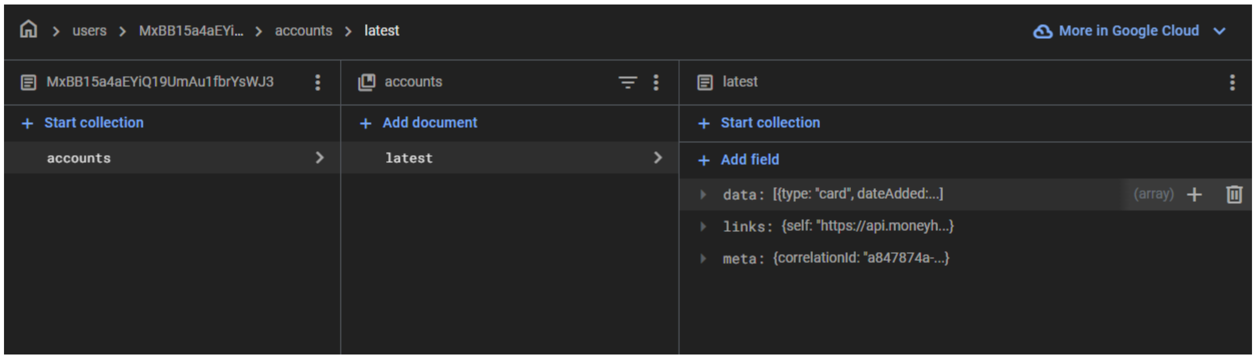
(default)
- budgets
- chat_messages
- transactions
- users

transactions
- 14JDAKIstf2nXP0rYoOD
- 1OsL5XIXNGNB2J3Eqc5R
- 2KAwnbKGNqUybCeauSos
- 3DmjSZyCeNpkkljS0LpG
- 5aPmcLNCuvrtuQkunCVB
- 6lvaOuyBn85meM3rMdj9
- 8FNBB5HeJYIdoIAg6P94
- AxvUOTtwL9fePbG2iWsE
- BPsESHBm5dNm8jpGw0Hc
- DfQlyp02TcRVslR8UPXu
- FfHgfoWoAgw4M3JE2lB1
- GkTnMmxueldhcFjrZXfu

14JDAKIstf2nXP0rYoOD

```
amount: 35
balance: "N/A"
category: "Food"
date: "2025-02-14T00:00:00.000Z"
description: "Tesco"
moneyhub_id: "122a5a42-a648-4542-b19f-cdf4d14e800d"
source: "moneyhub"
type: "Expense"
uid: "MxBB15a4aEYiQ19UmAu1fbrYsWJ3"
updated_at: April 24, 2025 at 8:20:47 PM UTC+1
```

Figure 1.6: Transactions Data structure



users > MxBB15a4aEYi... > accounts > latest

MxBB15a4aEYiQ19UmAu1fbrYsWJ3
- accounts

accounts
- latest

latest

```
data: [{type: "card", dateAdded:...]   (array)
links: {self: "https://api.moneyh...}
meta: {correlationId: "a847874a-...}
```

Figure 1.7: Bank Accounts Data structure

12. Testing Used to Assess Reliability

Unit Testing

- **Scope**: Tested individual components, such as the transaction categorization function, to ensure accurate assignment of categories (e.g., “Food” for “Tesco”). Mock data simulated diverse transaction descriptions.
- **Outcome**: Confirmed reliable categorization and error handling for missing or ambiguous data.

Integration Testing

- **Scope**: Evaluated interactions between modules, such as the bank integration fetching data via the Moneyhub API and updating the database. Tests checked if transactions were correctly stored and displayed on the dashboard.
- **Outcome**: Verified seamless data flow across front end, back end, and database, with proper error messages for failed API calls.

Error Handling Testing

- **Scope**: Simulated failures, such as invalid user credentials, API rate limits, or database connectivity issues, to test system resilience.
- **Outcome**: Confirmed that myFi displays user-friendly error messages and maintains stability under adverse conditions.

User Interface Testing

- **Scope**: Conducted manual tests across devices (desktop, mobile) to ensure the responsive interface functioned correctly. Scenarios included creating budgets, querying the chatbot, and navigating menus.
- **Outcome**: Ensured consistent rendering, intuitive navigation, and clear feedback (e.g., loading indicators, error alerts).

13. System Architecture

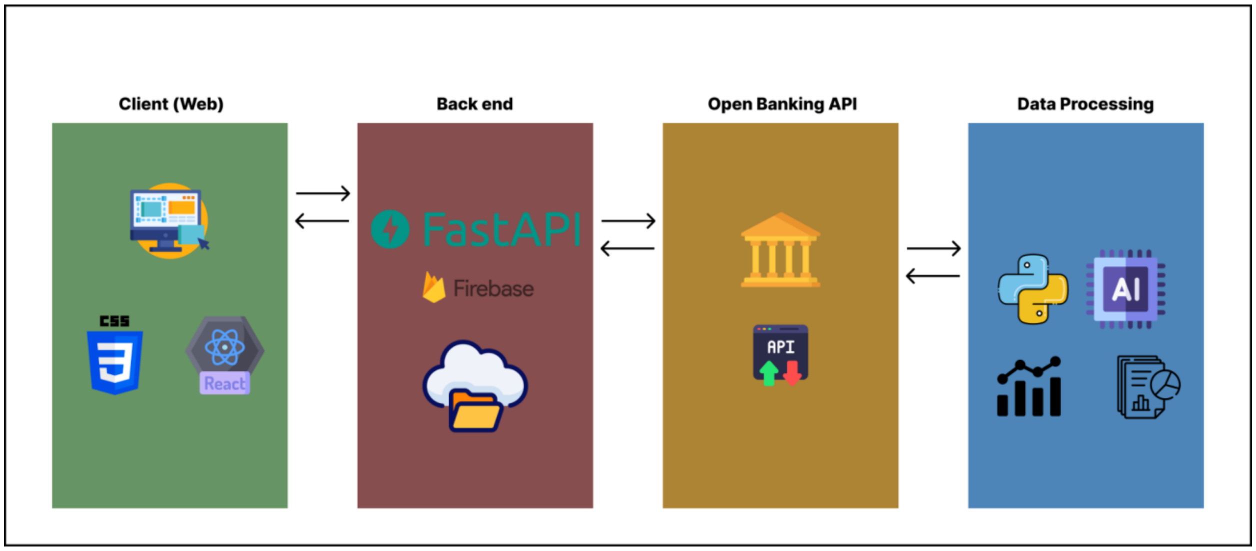


Figure 1.8: Bank Accounts Data structure

The web app is built with a modular, client-server architecture, integrating a responsive front end, a robust back end, a cloud-based database, and third-party APIs to deliver a seamless financial tracking experience. This structure ensures scalability, security, and real-time data processing.

13.1 Front End

The front end is a responsive web interface built using a modern JavaScript framework (React). It provides an intuitive user experience with components like dashboards, budget trackers, and a chatbot, accessible across devices. Dynamic visuals and interactive elements enhance usability for managing finances.

13.2 Back End

The back end, developed with a high-performance Python framework (FastAPI), handles business logic, API requests, and data processing. It manages user authentication, transaction categorization, and chatbot interactions, ensuring low-latency responses and secure data handling.

13.3 Database

A cloud-based NoSQL database (Firestore) stores user profiles, transactions, budgets, and chat messages. Its real-time capabilities support instant data updates, while its scalability accommodates growing user data and ensures reliable access.

13.4 APIs

MyFi integrates two key APIs:

- **Financial API (Moneyhub)**: Fetches real-time bank account balances and transactions via OAuth-based authentication.
- **AI API (OpenAI)**: Powers the chatbot and enhances transaction categorization with natural language processing.

These components interact seamlessly to provide a cohesive, user-friendly financial management tool.

14. Security and Privacy Considerations

The myFi application handles sensitive financial data, such as bank account details and transaction histories, making security and privacy paramount to building user trust. As a first-time developer integrating the Moneyhub Open Banking API, I faced significant challenges in ensuring robust security, particularly with authentication and data protection. This section outlines the security measures implemented, evaluates their effectiveness in safeguarding user data, and discusses privacy considerations to reassure users that their information is handled responsibly.

Authentication

- To secure user authentication, myFi leverages a trusted third-party service for user login, which employs industry-standard encryption to protect credentials. This service verifies user identities securely, ensuring that only authorized individuals can access the application. For connecting to bank accounts via Moneyhub's Open Banking API, I implemented the OAuth 2.0 protocol, which allows users to grant access to their financial data without sharing login credentials. Despite initial struggles with OAuth, such as weeks spent troubleshooting token management and redirect errors—I successfully configured a system that securely exchanges authorization codes for access tokens. These tokens are stored in a cloud-based database with restricted access, ensuring that sensitive financial data remains protected from unauthorized access.

Full Stack Security

- Data transmission between the frontend, backend, and external APIs is secured using HTTPS, which encrypts all communications to prevent interception by malicious actors. The application's database is configured with access rules that limit data retrieval to authenticated users, ensuring that each user can only view their own financial information. For instance, transaction and budget data are tied to unique user identifiers, preventing cross-user data leakage. Additionally, the backend includes mechanisms to refresh Moneyhub access tokens automatically before they expire, reducing the risk of failed connections or exposure due to outdated credentials.

Safety Reassurance

- Privacy is a core consideration in myFi's design. The application collects only the data necessary for its functionality such as bank account details, transactions, and budget settings and does not share this information with third parties beyond the Moneyhub and AI services required for core features. The AI chatbot, which provides financial insights, processes queries securely without storing sensitive user data beyond the duration of the session. To further enhance privacy, user data is cleared from the application's interface upon logout, minimizing the risk of residual data exposure on shared devices.

User Access

- MyFi prioritizes user security by implementing robust measures for app access and cloud-stored financial data. Access to the application on users' devices is secured through a cloud-based authentication system (Firebase Authentication), requiring email and password verification to ensure only authorized users can log in. Financial data, including transactions and budgets, is stored in a secure, cloud-based NoSQL database (Firestore) with encrypted data transmission and access controls to prevent unauthorized access. This setup protects sensitive information, such as bank account details fetched via the Moneyhub API, while allowing seamless access across devices, ensuring both convenience and trust in the application's handling of personal financial data.

15. Future Work

MyFi provides a solid foundation for personal finance management, but additional features could enhance its functionality and user engagement. Future development will focus on implementing unachieved features and introducing new capabilities to make the application more comprehensive and tailored to diverse user needs.

These enhancements I covered in the not achieved section will elevate myFi’s utility, making it a more dynamic and user-centric financial management tool.

16. Conclusion

The myFi project successfully delivered a user-friendly web application that empowers individuals to manage their finances effectively. By integrating real-time bank data, automated transaction categorization, budget tracking, and an AI-powered chatbot, myFi simplifies financial oversight and promotes informed decision-making. Despite time constraints preventing features like notifications and custom machine learning, the application meets its core objectives, offering a reliable and intuitive platform. Testing ensured stability across key functionalities, while the modular architecture supports future scalability. MyFi’s impact lies in its ability to enhance financial literacy and discipline, benefiting users with accessible tools for tracking and planning. This project has been a valuable learning experience, equipping me with skills in full-stack development, API integration, and user-centric design, and laying the groundwork for future enhancements to make myFi an even more powerful financial companion.

17. References

Resources/ Tools used:

[1] Firebase (2024), Firebase Authentication Documentation, Google, https://firebase.google.com/docs/auth.

[2] Google Cloud (2024), Cloud Firestore Documentation, Google, https://cloud.google.com/firestore/docs.

[3] Moneyhub (2024), Moneyhub Open Banking API Documentation, Moneyhub Financial Technology, https://developer.moneyhub.co.uk/docs.

[4] OpenAI (2024), OpenAI API Documentation, OpenAI, https://platform.openai.com/docs/introduction.

[5] React (2024), React Official Documentation, Meta, https://react.dev/learn.

[6] FastAPI (2024), FastAPI Documentation, Sebastián Ramírez, https://fastapi.tiangolo.com.

[7] OAuth (2023), OAuth 2.0 Authorization Framework, Internet Engineering Task Force (IETF), https://datatracker.ietf.org/doc/html/rfc6749.

Other/ Tools used:

Flaticon
Flaticon, 2024. *Flaticon: Free icons for your projects*. Available at: https://www.flaticon.com/ [Accessed 28 April 2025].

Figma
Figma, 2024. *Figma: The collaborative interface design tool*. Available at: https://www.figma.com/ [Accessed 28 April 2025].

ChatGPT
OpenAI, 2024. *ChatGPT: AI language model for various applications*. Available at: https://chatgpt.com/ [Accessed 28 April 2025].

Declaration

I, the undersigned, declare the following in relation to my Final Year Project:

- I understand that plagiarism involves presenting someone else's work, ideas, writings, or materials (including those from books, journals, the Internet, software, or other sources) as my own, whether intentionally or unintentionally, without proper acknowledgement.
- I confirm that this project, except where duly acknowledged and appropriately referenced, is entirely my own work. All quotations, paraphrases, summaries, code, or other materials sourced from external works have been cited in accordance with the referencing guidelines provided by SETU Carlow.
- I have included a complete bibliography of all sources used in the preparation of this project.
- I acknowledge that I have not copied another person's work, downloaded unpermitted material from the Internet, or allowed others to copy my work with the intent of passing it off as their own.
- I understand that plagiarism is a serious academic offence, equivalent to cheating, and that failure to comply with the university's regulations may result in severe consequences, including the project not being graded.

- I confirm that any incorporation of external material was discussed with and approved by my project supervisor, and all such material is appropriately referenced.

I hereby declare that I have read and understood the above statements and that this submission complies with SETU Carlow's policies on plagiarism.

Signature(Printed):	MATTHEW UFUMELI
Student Number:	C00273575
Date:	28/04/2025