Ishmael Matey Azu 4200920

PROJECT CONTENT

Introduction

Problem Statement

Aim of the project

Specific Objectives of the project

Justification of project

Motivation for undertaking Project

Scope of project

Project limitations

Beneficiaries of the project

Academic and practical relevance of the project

Project activity planning and schedules

Structure of report

Project Deliverables

Chapter 1: Introduction

1.1 Problem Statement

Managing personal finances can be challenging, leading to overspending and financial stress. Existing solutions often lack advanced features like predictive analytics and anomaly detection, which can provide deeper insights into spending patterns.

1.2 Aim of the Project

To develop a budgeting and expense management web application using React that leverages AI for predictive analytics and anomaly detection, helping users make informed financial decisions.

1.3Specific Objectives of the Project

- To build a user-friendly web interface for tracking income and expenses.

- To implement predictive analytics to forecast future spending and savings.

- To incorporate anomaly detection to identify unusual spending patterns.

- To provide comprehensive reporting tools for financial analysis.

1.4 Justification of Project

This project addresses the need for a more intelligent and insightful approach to personal finance management, providing users with advanced tools to maintain financial health.

1.5 Motivation for Undertaking Project

The motivation stems from the increasing complexity of personal finances and the potential of AI to simplify and enhance financial management.as well as the rising demand for digital personal finance tools designed for budgeting to help users make informed decisions

1.6 Scope of Project

The project will cover the development of the web application, integration of AI components for analytics, and user interface design. It will not include mobile app development.

1.7 Project Limitations

- The accuracy of predictive analytics may vary based on the quality of input data.

- The system will require continuous updates to improve AI models.

1.8 Beneficiaries of the Project

- Individuals seeking better control over their finances.

- Financial advisors looking for advanced tools to assist clients.

1.9 Academic and Practical Relevance of the Project

The project combines elements of web development, AI, and finance, providing a practical application of these disciplines and contributing to academic knowledge in these areas.

1.10 Project Activity Planning and Schedules

A detailed Gantt chart outlining the project phases, milestones, and deadlines will be provided in the appendix.

1.11 Structure of Report

- **Chapter 1: Introduction**

- Chapter 2: Review of Related Works

- Chapter 3: Methodology

- Chapter 4: Implementation and Results

- Chapter 5: Findings and Conclusion

1.12 Project Deliverables

- Functional budgeting and expense management web app.
- Documentation of the development process and methodologies.
- User manual for the application.

Chapter 2: Review of Related Works

2.1 Processes of the Existing System

Current budgeting systems often lack real-time updates and advanced analytics. While they offer basic budgeting features, they do not provide personalized insights or predictive analytics, which are essential for proactive financial management.

2.2 The Proposed System

Our proposed system addresses these gaps by offering a user-friendly interface, real-time updates, and advanced analytics through artificial intelligence.

2.3 Conceptual Design

The application will have a modular design, including components for user authentication, budget management, expense tracking, anomaly detection, and predictive analytics.

2.4 Architecture of the Proposed System

The system will use a client-server architecture with a frontend built in React.js and React Router, and a backend using Node.js. AI models will be integrated with TensorFlow or scikit-learn.

2.5 Components Designs and Descriptions

Detailed descriptions of each component will be provided, including their interactions and roles in the system. Diagrams will illustrate these components where necessary.

2.6 Proposed System/Software Features

- User authentication and profile management
- Expense tracking and categorization
- Budget creation and management
- Real-time notifications
- Anomaly detection and predictive analytics

2.7 Development Tools and Environment

- IDE: Visual Studio Code
- Frameworks: React.js, Node.js, TensorFlow/scikit-learn
- Version Control: Git
- Deployment: AWS or other cloud services

2.8 Benefits of Implementing the Proposed System

This system will give users a powerful tool to manage their finances, offering real-time insights and predictive capabilities not found in existing solutions.

The system architecture will be a client-server model with React for the frontend, Node.js for the backend, and TensorFlow.js for AI functionalities.

Components Designs and Components Descriptions

- User Interface: React components for input forms, dashboards, and reports.
- Backend API: Node.js for handling data requests and transactions.
- AI Module: TensorFlow.js for predictive analytics and anomaly detection.

Proposed System/Software Features

- Income and expense tracking.
- Predictive analytics for future financial planning.
- Anomaly detection for unusual spending patterns.
- Comprehensive reporting tools.

Development Tools and Environment

- React: For building the user interface.

- Node.js: For backend development.

- TensorFlow.js: For AI functionalities.

- MongoDB: For database management.

Benefits of Implementation of the Proposed System

- Enhanced financial management through predictive insights.

- Early detection of unusual spending patterns.

- User-friendly interface for easy financial tracking.

Chapter 3: Methodology

Chapter Overview

This chapter outlines the methodologies used for requirement gathering, system design, and development.

3.1 Requirement Specification

Stakeholders of System

- End-users.

- Financial advisors.

- Developers.

3.2 Requirement Gathering Process

Interviews, surveys, and literature reviews were conducted to gather system requirements.

3.3 Functional Requirements

- User authentication.

- Income and expense input forms.

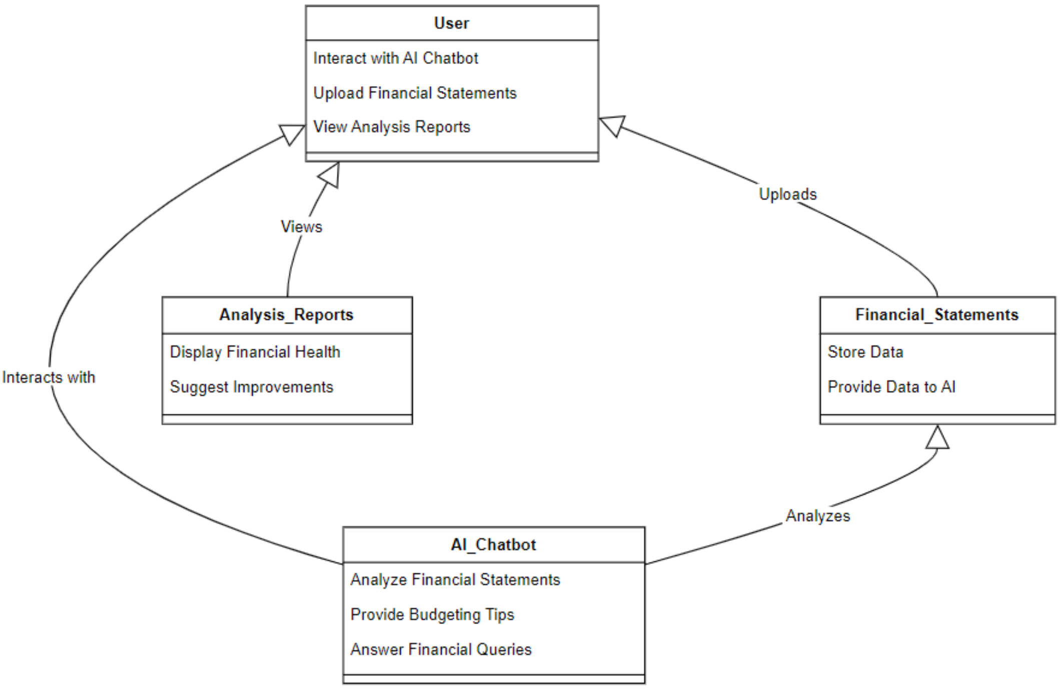
- Predictive analytics dashboard.

- Anomaly detection alerts.

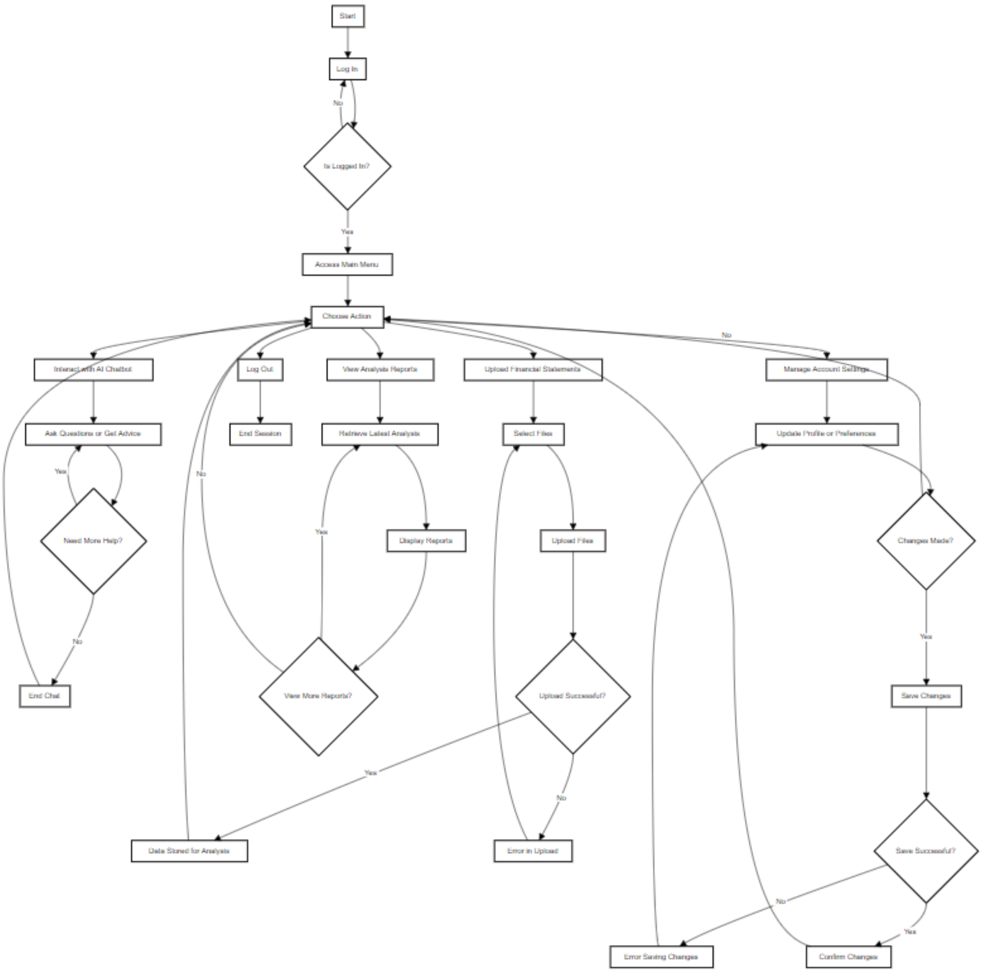
- Reporting tools.

3.4 UML Diagrams

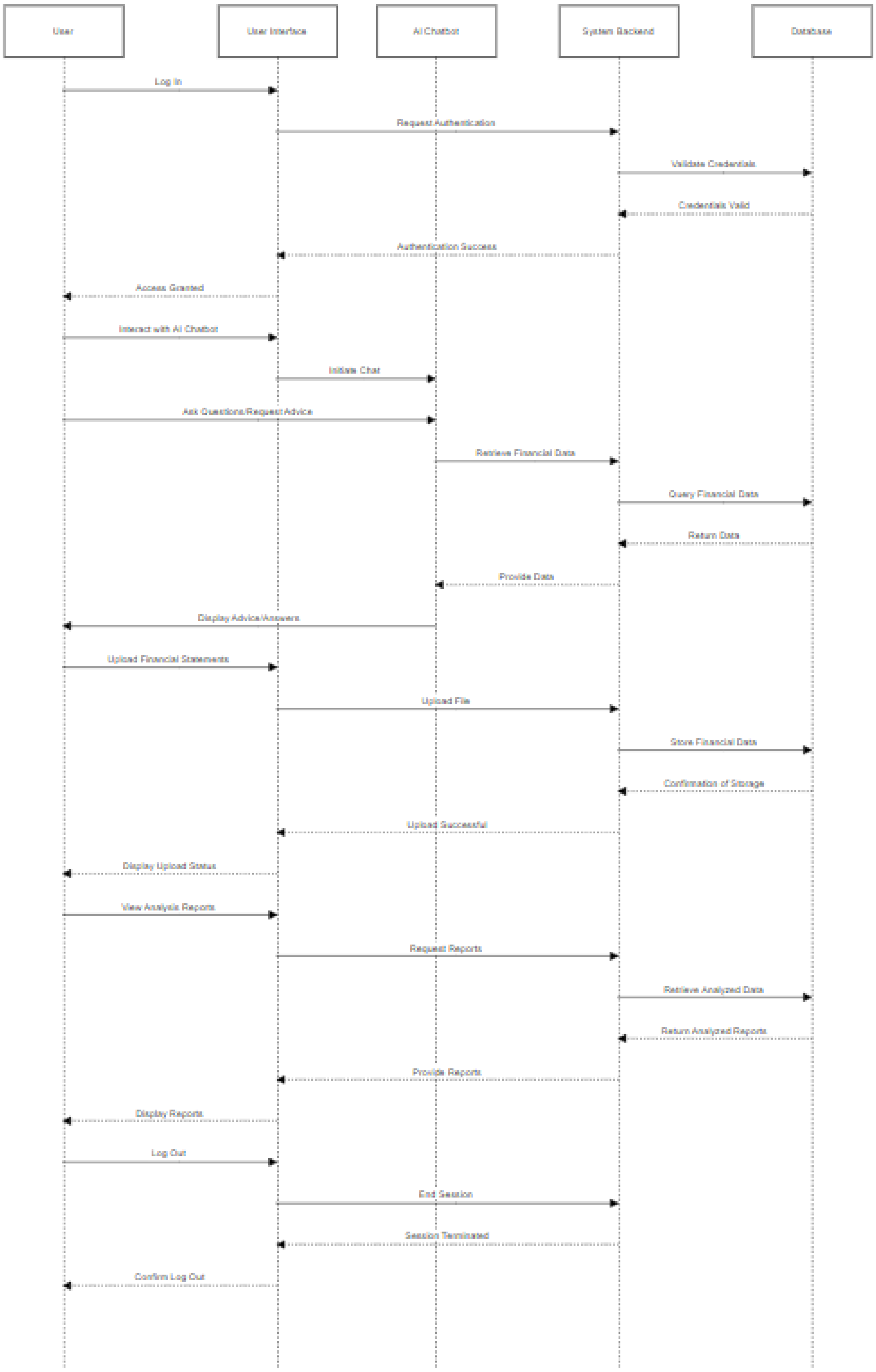
- Use Case Diagrams: For frontend and backend models.



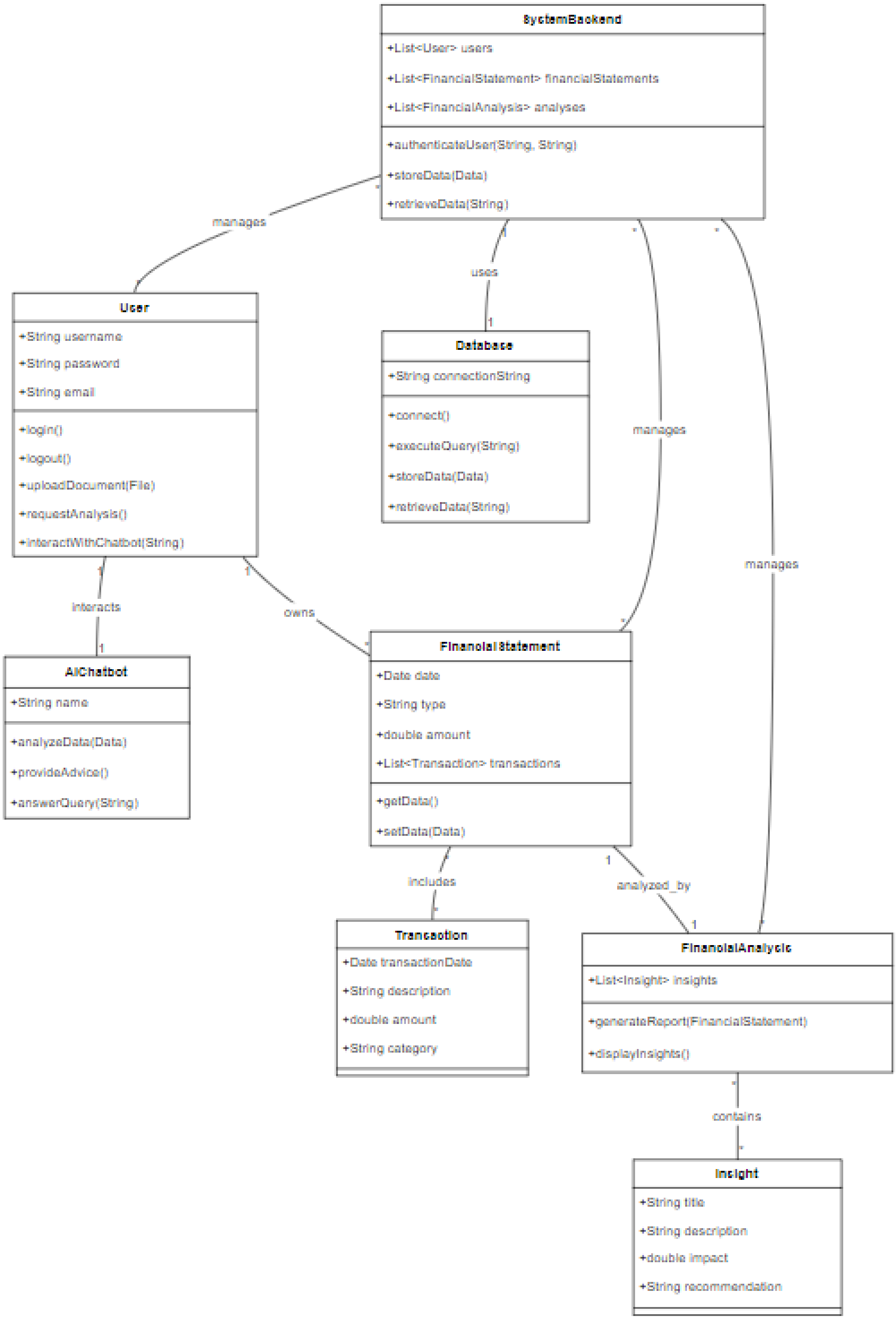
- **Activity Diagrams: For user interactions.**



- Sequence Diagrams: For data flow.



- Class Diagrams: For system components.



Use Case Description

A use case description typically includes details about how each use case functions, the main participants, and the interaction flow. Here's a detailed description for the primary use cases in your budgeting app with an AI chatbot that analyzes financial statements:

1. Interact with AI Chatbot

Primary Actor: User

Goal: To obtain personalized financial advice and answers to queries.

Preconditions: The user must be logged into their account.

Postconditions: The user receives relevant financial insights or information.

Basic Flow:

1. The user initiates a conversation with the AI chatbot.

2. The user inputs a financial query or requests advice.

3. The AI chatbot processes the input and accesses relevant financial data.

4. The AI chatbot responds with advice, insights, or answers based on the analysis.

5. The user can continue the conversation for more detailed information or end it.

Alternative Flows:

- If the AI chatbot cannot process the query due to lack of data, it prompts the user to upload more comprehensive financial statements.

2. Upload Financial Statements

Primary Actor: User

Goal: To upload financial data for analysis.

Preconditions: The user must be logged into their account.

Postconditions: The user's financial statements are stored and ready for analysis.

Basic Flow:

1. The user navigates to the upload section of the app.

2. The user selects the financial statements files from their device.

3. The user uploads the files.

4. The system confirms the successful upload and stores the data securely.

5. The AI chatbot now has access to the new data for analysis.

Alternative Flows:

- If the file format is incorrect, the system notifies the user and rejects the upload.

3. View Analysis Reports

Primary Actor: User

Goal: To view the analysis of uploaded financial statements.

Preconditions: Financial statements must have been uploaded and analyzed.

Postconditions: The user views the insights and reports.

Basic Flow:

1. The user selects the option to view analysis reports.

2. The system retrieves the latest analysis results.

3. The user views various insights such as financial health, trends, and potential areas for improvement.

4. The user can download or save the report for offline use.

Alternative Flows:

- If no new analysis is available, the system informs the user and suggests uploading recent financial statements.

4. Manage Account Settings

Primary Actor: User

Goal: To update personal information and configure application settings.

Preconditions: The user must be logged into their account.

Postconditions: User settings are updated as per the changes made.

Basic Flow:

1. The user navigates to the account settings section.

2. The user updates profile details or modifies application preferences.

3. The user saves the changes.

4. The system updates the account settings and confirms the changes to the user.

Alternative Flows:

- If the system fails to save changes due to a technical issue, it notifies the user and prompts a retry.

Each of these use case descriptions details the steps involved in executing the function, along with handling various scenarios that might occur. This setup will aid in understanding user interactions and the functional requirements of the system.

Non-Functional Requirements

- Performance: The system should handle multiple concurrent users.

- Security: Secure user data and prevent unauthorized access.

Security Concepts

The system will employ encryption for data storage and transmission, and implement secure authentication mechanisms.

Project Methods

Agile methodology will be used for iterative development and continuous feedback.

The Various Software Process Models

- Waterfall: Sequential approach.

- Agile: Iterative and flexible.

- Chosen Model: Agile, for its adaptability and continuous improvement.

Chosen Model and Justification

Agile methodology is chosen due to its flexibility and suitability for incorporating user feedback and making iterative improvements.

Project Design Consideration (Logical Designs)

UI Design

Wireframes will be created to design the user interface.

DB Design

E-R diagrams and DB schemas will be developed for database design.

3.5 Developmental Tools

Detailed descriptions of React, Node.js, TensorFlow.js, and MongoDB will be provided.

Chapter 4: Implementation and Results

4.1Chapter Overview

This chapter covers the implementation process, including code snippets and testing.

4.2 Mapping Logical Design onto Physical Platform

- UI Implementation: Using React components.

- Database Implementation: Using MongoDB.

- AI Implementation: Using TensorFlow.js.

4.3 Construction

4.4Testing

-**Objective:** Ensure individual components function correctly.

- **Approach:** Automated tests for each component.

- **Results**: Highlight coverage and key issues fixed.

4.5Testing Plan

- Component Testing: Algorithms for UI and database testing.

- System Testing: Verification and validation testing algorithms.

4.6 Results

The results of the testing phases will be documented.

Chapter 5: Findings and Conclusion

5.1 Chapter Overview

5.2 Findings

- **User Engagement:** Users interact frequently with the AI chatbot, mainly asking for help with budgeting and financial health checks. Engagement levels are high, showing that users find the chatbot useful.

- **Accuracy of Analysis:** The AI's financial analysis matches up well with standard financial models. Users feel more confident in understanding their financial situation after using the app.

5.3 Conclusions

- **Effectiveness of AI Chatbot:** The AI chatbot is successful in helping users manage their finances by providing quick, reliable advice.

- **Improving Performance:** To ensure a consistently smooth experience, the app needs to handle large amounts of data more efficiently, especially during busy periods.

- **User Interface Improvements:** Although the app is easy to use, making some parts of the interface more straightforward could help users who aren't as tech-savvy.

5.4 Challenges/Limitations of the System

-Integrating ai into manual budgeting and expense management

-Speed of responses

-Matching Industry Security Standard

5.5 Lessons Learned

1. Importance of Clear Requirements

- Insight: Ambiguities in initial requirements can lead to scope creep and delays.
- Action: Emphasize the need for detailed, clear, and agreed-upon requirements before starting development.

2. User-Centric Design

- Insight: Initial feedback indicated that some app features were not intuitive.
- Action: Incorporate user feedback early and often through prototypes and user testing sessions.

3. AI Integration Challenges

- Insight: Integrating AI to analyze financial statements presented technical challenges, particularly in data consistency and accuracy.

- Action: Invest more in preliminary research on AI technologies and involve AI specialists early in the project**.**

5.6 Recommendations for Future Works

1. Enhance User Interface Design

- Rationale: To improve user satisfaction and engagement, as feedback suggested some features were not intuitive.
- Recommendation: Redesign certain UI elements for simplicity and enhanced usability. Consider involving UI/UX experts to incorporate best practices in design.

2. Expand AI Capabilities

- Rationale: The AI chatbot was well-received, but there's potential to offer more detailed analyses and predictions.
- Recommendation: Develop additional AI-driven features, such as predictive budgeting and financial health scoring, to provide more value to users.

3. Strengthen Security Measures

- Rationale: Initial project phases underestimated the importance of security.
- Recommendation: Integrate security as a core aspect of the development process, including regular security audits and updates to ensure data integrity and protection.

4. Optimize for Scalability

- Rationale: Performance issues during peak usage indicate that scalability was not fully considered.
- Recommendation: Reevaluate the architecture to support a growing number of users, possibly incorporating cloud services and scalable databases.

5. Enhance Data Handling and Storage

- Rationale: Efficient data management is crucial for performance and user trust.
- Recommendation: Invest in better data management solutions, focusing on improving data retrieval and storage efficiency.

References

- React Router Documentation: [React Router Documentation](https://reactrouter.com/)

- Node.js Documentation: [Node.js Documentation](https://nodejs.org/)

- React.js Documentation: [React.js Documentation](https://reactjs.org/)

- **TensorFlow Documentation: [TensorFlow Documentation](https://www.tensorflow.org/)**

- **scikit-learn Documentation: [scikit-learn Documentation](https://scikit-learn.org/stable/)**