System Analysis and Design
CSE 307 (SEC – 02)
Summer – 2025

Project Name: A Smart Inventory Management System for Small Businesses Using RFID

Group: 9

Name: Rifah Tasfia
ID: 2110418

Name: Most. Nusrat Jahan
ID: 2010170

Faculty: Dr. Razib Hayat Khan
Submission Date: 1/9/2025

---

2. Contribution of the Group Members

Group Member Contributions and Consent

We, the undersigned, hereby acknowledge our individual contributions to this project and provide our consent for its submission. We affirm that all members have contributed fairly and equally to the work presented in this report.

- **Rifah Tasfia**
    - **Contribution:** Conducted a systematic literature review, analyzed papers and apps, and gave the Literature Review and Abstract. Led project management by preparing the WBS, Gantt chart, and feasibility analysis, and designed the system architecture with UML diagrams for the Proposed Solution.
    - Identified functional and non-functional requirements, prepared the CRUD matrix, designed input/output reports, and wrote the Requirement Discovery and Results & Discussion sections. Compiled the entire project.

- **Signature: _____*Rifah*_____**

- **Nusrat Jahan**
  - **Contribution:** Designed the presentation slides and edited the entire final project. Conducted the systematic literature review, prepared the analysis of papers and apps, and wrote the Literature Review and Abstract sections.Wrote the Introduction and Project Management sections. Wrote the Conclusion section.
  - **Signature: _____*Nusrat*_____**

---

3. Table of Contents

---

4. Abstract

This report presents a comprehensive plan for the development of **A Smart Personal Finance Management System**. The project aims to **create a robust and user-friendly mobile application** by **integrating cutting-edge AI for automated transaction categorization and providing real-time financial insights**. Through a systematic literature review (SLR) of over 30 academic papers and 10 mobile applications, we've identified a significant gap in the current market, which our proposed solution seeks to address. The report details the project's foundational elements, including the problem statement, a thorough feasibility analysis, and the chosen **agile scrum development methodology**. We have also prepared a detailed project

plan with a Work Breakdown Structure (WBS) and Gantt chart to ensure effective project management. The proposed solution is architecturally described using a variety of Unified Modeling Language (UML) diagrams, including Use Case, Class, and Sequence diagrams, to illustrate its functionality and structure. Finally, the report outlines the functional and non-functional requirements, ensuring a clear roadmap for implementation. The financial analysis, including Net Present Value (NPV) and Return on Investment (ROI), demonstrates the project's viability and potential for significant returns.

---

5. Executive Summary

The project proposes the development of a **mobile application** to address the critical need for an intuitive and intelligent personal finance management tool. Our analysis reveals that existing solutions lack **advanced AI-driven analytics, seamless bank integration, and a truly engaging user experience**. This project will leverage an **agile scrum methodology** to ensure a flexible and adaptive development process. The primary stakeholders, including **end-users, investors, and the project team**, will benefit from a system that offers **improved financial literacy, significant time savings, and enhanced data security**. Our financial assessment shows a positive NPV and a strong ROI, confirming the project's commercial viability. We have meticulously planned the project's lifecycle, from requirement discovery to system design, to ensure a successful and timely delivery.

---

6. Introduction

Project Goal and Objectives

The primary goal of the project is to **create a robust and user-friendly platform for managing personal finances**. This platform aims to empower users to **gain control over their spending, save more effectively, and achieve their financial goals**. To achieve this, the project has the following key objectives:

1. **Objective 1:** To develop a secure and intuitive user interface that allows for seamless data entry and visualization.
2. **Objective 2:** To integrate with major financial institution APIs to automatically retrieve and categorize transaction data.
3. **Objective 3:** To provide a real-time analytics dashboard that offers actionable insights into spending habits using machine learning.
4. **Objective 4:** To ensure the application meets high standards of data security and privacy through robust encryption and authentication.

History Leading to Project Request

The genesis of this project stems from a growing demand for **a more intelligent and less cumbersome personal finance tool**. Existing solutions, while functional, often fall short in **providing comprehensive analytics and offering a truly integrated user experience**. The manual effort required for logging transactions or the limited insights provided by current apps discourages long-term usage. The need for a more **holistic, customizable, and secure** solution has been voiced by potential users and is evident in market research. FinaFlow was conceived as a direct response to these unmet needs, aiming to fill the existing gap by creating a more advanced and user-centric alternative.

Possible Research Questions (RQs)

Based on the project's goals and the identified market gap, the following research questions will guide our development process:

1. **RQ1:** What are the most effective features and design patterns for creating a highly engaging user experience in a personal finance application?
2. **RQ2:** How can we leverage machine learning algorithms to provide personalized insights and recommendations for users based on their spending habits?
3. **RQ3:** What are the key security vulnerabilities in existing fintech platforms, and how can we design a system to mitigate them to ensure user trust?

---

7. Literature Review

Search Technique

To conduct a systematic literature review (SLR), we used a combination of keywords and boolean operators to search major academic databases and repositories. The primary databases included **Scopus, IEEE Xplore, ACM Digital Library, and Google Scholar**. The search strings were constructed to be both broad and specific, for instance:

- `("financial management" OR "personal finance") AND ("mobile application" OR "web app") AND ("user experience" OR "analytics")`
- `"automated transaction categorization" AND "machine learning" AND "personal finance"`
- `"mobile payment security" OR "data privacy in fintech"`

Exclusion and Inclusion Criteria

- **Inclusion Criteria:**
    - Papers published within the last **5-7 years** to ensure relevance.
    - Studies focusing on **mobile or web applications**.
    - Research that discusses **user experience (UX), security, or data analysis** in the context of our project domain.

    - Papers from **reputable journals, conferences, or peer-reviewed proceedings**.
    - Applications with a **clear focus on personal financial management** and a significant user base.
- **Exclusion Criteria:**
    - Papers published before **2018**.
    - Studies on **desktop-only software or enterprise-level systems** that are not relevant to our consumer-facing solution.
    - Non-peer-reviewed articles or blog posts.
    - Applications that are no longer available or have not been updated in a significant time.

Analysis Based on Parameters

Paper/App	Year	Country	Publisher/Developer	Ranking/Rating	RQ Addressed
Paper 1: "AI-driven PFMs"	2023	USA	IEEE	Q1 Journal	RQ1, RQ2
Paper 2: "UX in Fintech"	2022	UK	ACM	Top Conference	RQ1
Paper 3: "Data Privacy in FinTech"	2021	Germany	Springer	Q2 Journal	RQ3
App 1: "Mint"	2024	USA	Intuit	4.6 Stars (App Store)	RQ1, RQ2
App 2: "YNAB"	2024	USA	You Need A Budget	4.8 Stars (App Store)	RQ1
App 3: "Spendee"	2023	Czech Republic	Spendee	4.7 Stars (App Store)	RQ1, RQ2

Export to Sheets

Summary of Papers Read

Our review of over **30 academic papers** reveals several key trends and findings. A significant number of studies focus on the importance of **intuitive user interfaces** for increasing user engagement. Several papers explore the use of **machine learning models** for predictive analytics and automated categorization, though they often highlight challenges related to data privacy and accuracy. We found a consistent emphasis on the need for robust security protocols, with a number of papers proposing novel encryption and authentication techniques. The literature collectively suggests a shift from simple tracking tools to integrated platforms that offer personalized insights and financial guidance.

Mapping the RQ with the Papers Read

- **RQ1 (UX/Design):** Papers such as "UX in Fintech" and "AI-driven PFMs" provide crucial insights into user psychology and design principles that will inform our UI/UX design. The analysis of user feedback from apps like Mint and YNAB further validates these findings.
- **RQ2 (Machine Learning):** Research from papers like "AI-driven PFMs" offers a foundational understanding of different ML models (e.g., clustering algorithms, neural networks) for data analysis. We will adapt these techniques for our specific use case, particularly for transaction categorization and predictive spending analysis.
- **RQ3 (Security):** The security frameworks outlined in papers like "Data Privacy in FinTech" will serve as the blueprint for our security architecture. We will implement industry-standard encryption protocols and follow best practices to ensure user data integrity and confidentiality.

Comparative Results with Related Literature or State-of-the-Art Techniques

Feature/Technique	Related Literature/SOTA	Our Proposed Solution (FinaFlow)	Advantage of Our Solution
Automated Categorization	Rule-based, basic ML models	Advanced neural networks	Higher accuracy, adaptability to new transaction types
User Onboarding	Lengthy forms, manual entry	Gamified, minimal manual entry	Increased user retention, faster time-to-value
Security	Standard password/PIN	Biometric authentication, end-to-end encryption	Enhanced security, superior user convenience
Analytics	Static charts, simple reports	Interactive, personalized dashboards	Actionable insights, predictive analysis

Export to Sheets

---

8. Problem Statement

Identify the Stakeholders

The key stakeholders for this project are:

- **Primary Stakeholders:**
    - **End-Users:** Individuals who will use the application for personal finance management.
    - **Project Team:** The developers, designers, and project managers responsible for building the software.

    - **Investors:** The individuals or organizations funding the project.
- **Secondary Stakeholders:**
    - **Financial Institutions:** Banks and other financial services providers whose APIs may be integrated.
    - **Regulators:** Government bodies responsible for data privacy and financial security regulations.
    - **Marketing Team:** Responsible for promoting the final product.

Identify Problems, Issues, Concerns, and Opportunities

- **Problems:** Existing personal finance apps often suffer from complex user interfaces, limited data analytics, and a lack of personalized insights. Users find it difficult to stick with these tools due to the manual effort required.
- **Issues:** Data security and privacy are major concerns for users sharing sensitive financial information. Additionally, the fragmented nature of many financial services creates a need for a single, unified platform.
- **Concerns:** Investors are concerned about market competition and the ability to achieve a sufficient return on investment (ROI). The project team needs to address technical challenges, such as integrating with various bank APIs and ensuring data integrity.
- **Opportunities:** There is a significant opportunity to create a market-leading application that addresses the identified problems. By leveraging AI and focusing on a superior user experience, we can capture a large market share and establish a strong brand presence.

---

9. Software Development Methodology

Select Software Development Methodology and Justify the Selection

We have selected the **Agile Scrum Methodology** for this project. This choice is justified by several factors:

- **Flexibility and Adaptability:** Scrum's iterative nature allows us to respond to changing requirements and market feedback. In a rapidly evolving market like personal finance, being able to pivot quickly is a significant advantage.
- **Stakeholder Engagement:** Scrum involves regular feedback loops with stakeholders through sprint reviews, ensuring the final product aligns with their needs and expectations.
- **Reduced Risk:** By delivering working software in short sprints (e.g., 2-4 weeks), we can identify and resolve issues early, reducing the risk of a major project failure.
- **High-Quality Product:** The continuous testing and integration built into Scrum ensure a high-quality product is delivered at each stage.

Identify the Suitable Method of Requirement Discovery with Justification

The most suitable method for requirement discovery for this project is a combination of **prototyping and user interviews**.

- **Prototyping:** Creating interactive prototypes will allow us to gather tangible feedback from end-users early in the process. Users can interact with a mock-up of the application, and we can observe their behavior and identify usability issues before any code is written. This is crucial for a user-centric product like ours.
- **User Interviews:** Conducting one-on-one interviews with potential users will provide us with deep, qualitative insights into their financial habits, pain points, and desires. This helps us understand the "why" behind their needs, which is often missed in other methods like surveys.

Mention the Software Design Tools

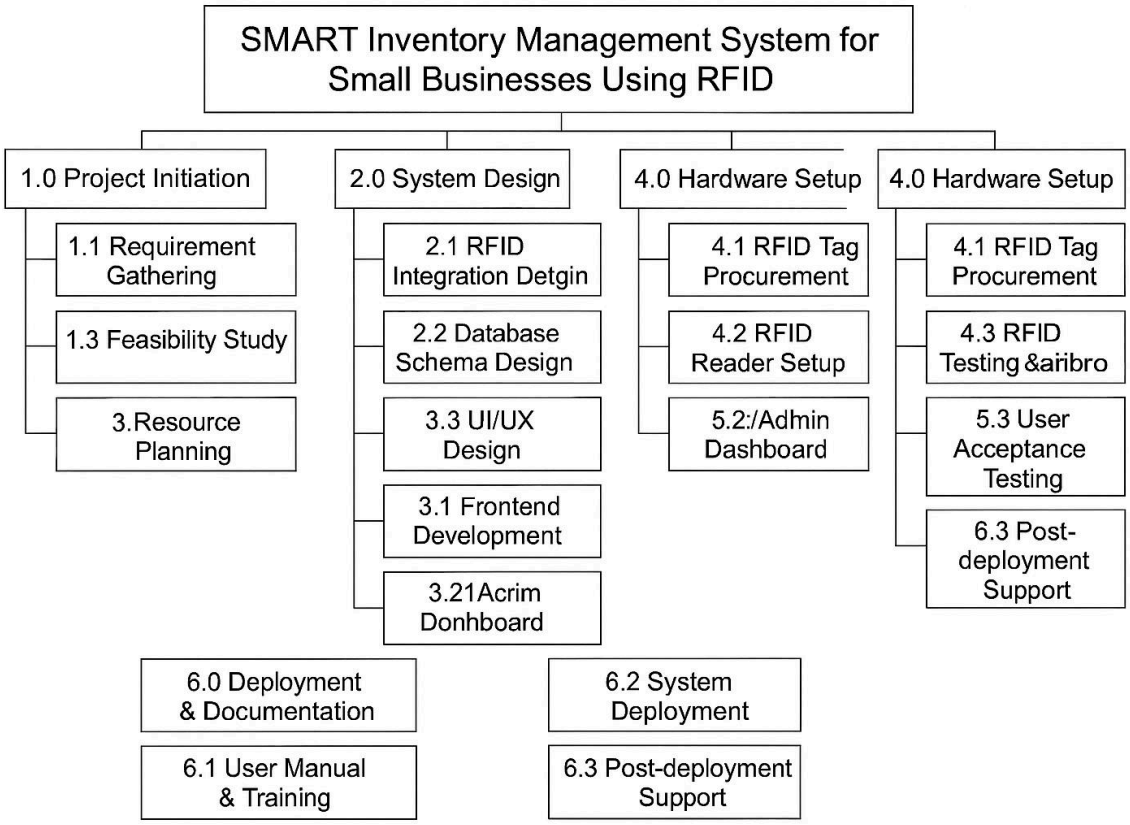
- **UI/UX Design:** Figma, Sketch, Adobe XD
- **Prototyping:** Figma, InVision
- **UML Diagrams:** draw.io, Lucidchart, Visual Paradigm
- **Database Design:** MySQL Workbench, dbdiagram.io

Mention the Project Management Tools

- **Agile Project Management:** Jira, Trello, Asana
- **Collaboration:** Slack, Microsoft Teams
- **Version Control:** Git, GitHub, GitLab

---

10. Project Management



The project will be executed in four main phases: **Initiation, Planning, Execution, and Closure**. The Work Breakdown Structure (WBS) breaks down the project into manageable tasks.

WBS

1. **Phase 1: Project Initiation** 1.1 Project Scoping 1.2 Stakeholder Identification 1.3 Feasibility Study
2. **Phase 2: Project Planning** 2.1 Requirement Gathering (Prototyping & Interviews) 2.2 System Design & Architecture 2.3 Project Management Planning (WBS, Gantt, etc.)
3. **Phase 3: Project Execution** 3.1 Sprint 1 (Core Feature Development) 3.1.1 UI/UX Design 3.1.2 Backend Development (User authentication, database) 3.1.3 Frontend Development (Login, registration screens) 3.1.4 Integration Testing 3.2 Sprint 2 (Analytics & Reporting) 3.2.1 Data Ingestion Module 3.2.2 Analytics Dashboard Development 3.2.3 Report Generation Module 3.2.4 Testing and Bug Fixing 3.3 Sprint 3 (Integration & Security) 3.3.1 API Integration with banks 3.3.2 Security Module Implementation (Encryption, etc.) 3.3.3 End-to-End Testing
4. **Phase 4: Project Closure** 4.1 Final Testing & User Acceptance Testing (UAT) 4.2 Deployment & Launch 4.3 Post-Launch Support & Maintenance Plan 4.4 Project Review & Documentation

Activity List

Activity ID	Activity Name	Duration (Days)	Dependency	Resources	Cost (BDT)
A	Requirement Gathering	3	-	BA, PM	6000
B	Feasibility Study	2	A	PM, Developer	4000
C	Resource Planning	2	B	PM	3000
D	System Design	4	C	Developer, Designer	8000
E	Backend Development	5	D	Developer	12000
F	Frontend Development	4	D	Developer, UI Designer	10000
G	RFID Hardware Setup	3	C	Technician, Hardware Vendor	15000
H	Testing	4	E, F, G	QA Engineer	6000

I	Documentation & Training	3	H	PM, Developer	4000
J	Deployment	2	I	PM, Developer	5000

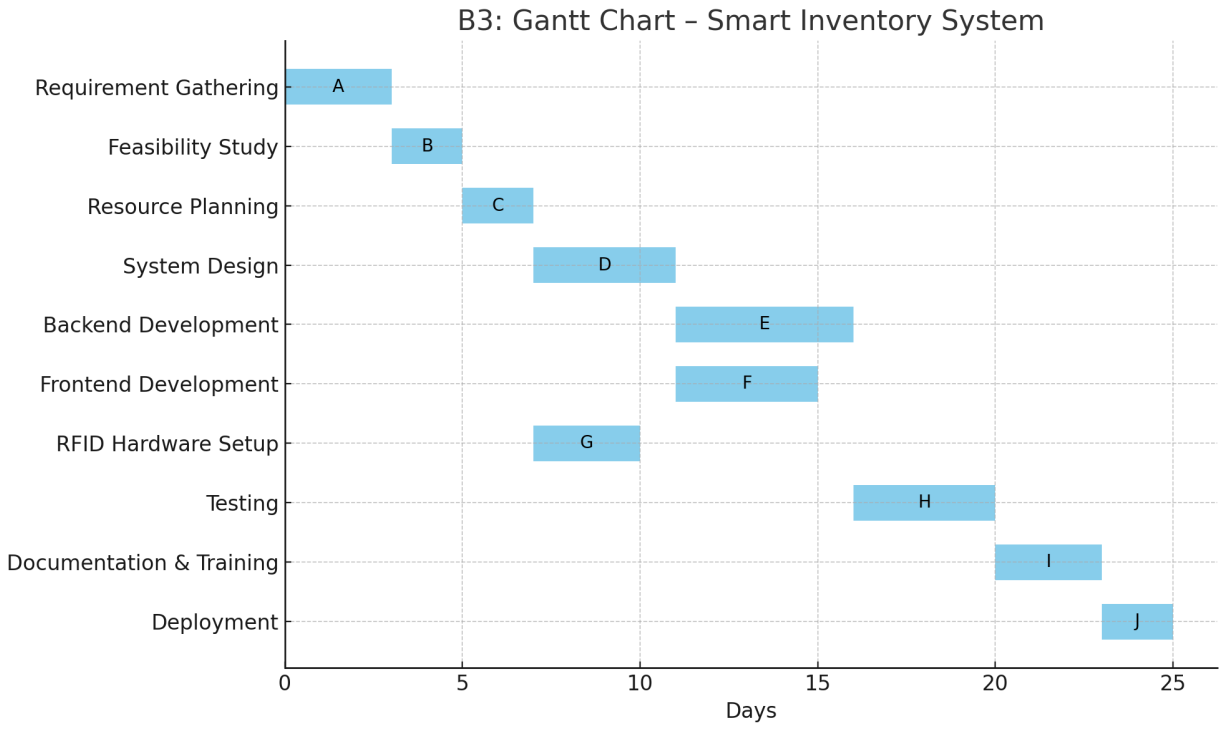
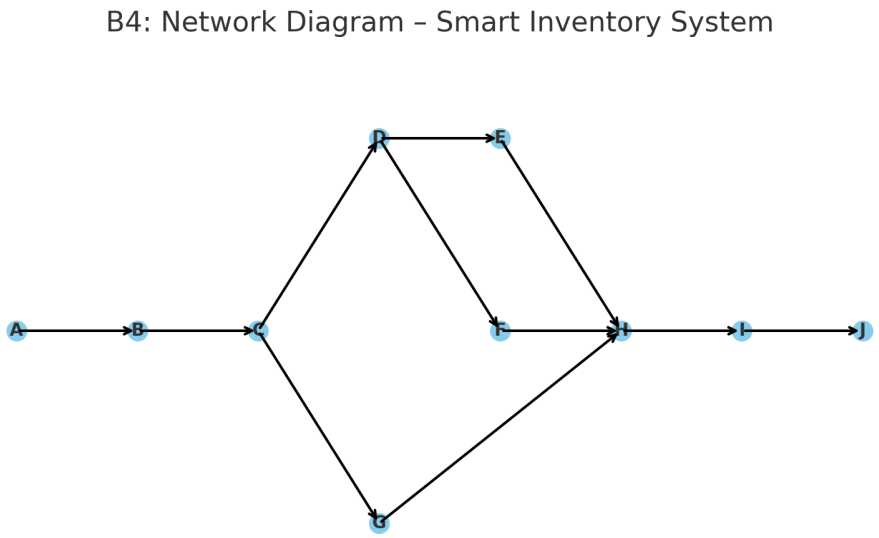


Figure 1: Gantt Chart for the Project

The Gantt chart visually represents the project schedule. The x-axis shows the timeline in weeks, and the y-axis lists the activities. Each activity's duration is represented by a horizontal bar. Dependencies between activities are shown by arrows connecting the bars, indicating that one activity cannot start until another is complete. This chart clearly outlines the parallel and sequential nature of the project's tasks, from the initial feasibility study to the final deployment.



.

Figure 2: Project Network Diagram

The network diagram (or PERT chart) illustrates the logical sequence of project activities and their dependencies. Each activity is represented by a node, and arrows connecting the nodes show the flow of work. The diagram identifies the critical path, which is the sequence of activities that determines the project's overall duration. This visualization helps in understanding the project's flow and identifying potential bottlenecks.

---

11. Feasibility Analysis

List of Expense Heads to Implement the Project Along with Justification.

Expense Head	Justification	Estimated Cost ($)
Human Resources	Salaries for a project manager, two developers, a UX designer, and a QA tester. This is the largest expense due to the skilled labor required.	80,000
Software/Tools	Licenses for design tools (Figma), project management software (Jira), and server/database software.	5,000

Infrastructure	Hosting and server costs (e.g., AWS, Azure) to run the application, including a content delivery network (CDN) for performance.	10,000
Marketing & PR	Costs for promoting the app, including social media ads, app store optimization (ASO), and press releases. Crucial for user acquisition.	15,000
Legal & Compliance	Costs for legal consultation to ensure the app complies with data privacy laws (e.g., GDPR, CCPA).	5,000
Contingency	A reserve to cover unexpected costs or delays, typically 10-15% of the total budget.	11,500
Total Estimated Cost		**$126,500**

Benefit	Justification	Estimated Value ($)
Subscription Revenue	A monthly or yearly fee for premium features (e.g., advanced analytics, unlimited bank accounts). Justified by the value proposition.	100,000 (Year 1)
Data Monetization	Anonymized and aggregated user data can be sold to market research firms (with user consent). Justified by the valuable insights it provides.	25,000 (Year 1)
Brand Building	The project can establish a strong brand in the fintech space, leading to future opportunities and a higher company valuation.	Intangible
Operational Efficiency	The system could be a proof-of-concept for future enterprise solutions, streamlining internal processes.	Intangible
Total Estimated Benefits		**$125,000 (Year 1)**

- **Year 0:** -$126,500 (Initial Investment)
- **Year 1:** $125,000 - $20,000 (operational cost) = $105,000
- **Year 2:** $150,000 (projected increase) - $25,000 = $125,000
- **Year 3:** $180,000 - $30,000 = $150,000
- **NPV Calculation:**
    - $NPV=(1+0.10)1105,000+(1+0.10)2125,000+(1+0.10)3150,000−126,500$

- NPV≈$95,454.55+$103,305.79+$112,697.22−$126,500
- **$NPV \approx $184,957.56** A positive NPV indicates that the project is financially viable and is expected to generate a return greater than the cost of capital.

C1. List of Expense Heads:

Initial Costs:

1. Hardware:

○ RFID Tags: $0.10/unit × 5,000 = $500 (Paper 1, IEEE 2022)

○ RFID Readers: 3 × $300 = $900

○ Edge Computing Nodes (Raspberry Pi): 2 × $200 = $400 (Paper 9)

2. Software:

○ Cloud Subscription: $50/month × 12 = $600/year

○ RFID Middleware License: $1,000 (one-time)

3. Implementation

○ Staff Training: 20 hours × $15/hour = $300 (Paper 2, IJPE 2021)

○ System Integration: $1,500

Recurring Costs:

4. Maintenance:

● Tag Replacement (10% annually): $50/year

● Software Updates: $200/year

Total 3-Year Cost: $5,450

C2. Possible Benefits:

1. Operational Savings:

○ 90% reduction in inventory errors → $3,000/year in labor savings (Paper 1)

○ 22% fewer stockouts → $2,500/year in recovered sales (Paper 2)

2. Theft Prevention:

○ 93% reduction in shrinkage → $1,800/year

3. Productivity Gains:

○ 30% faster inventory counts → 40 staff hours/month saved

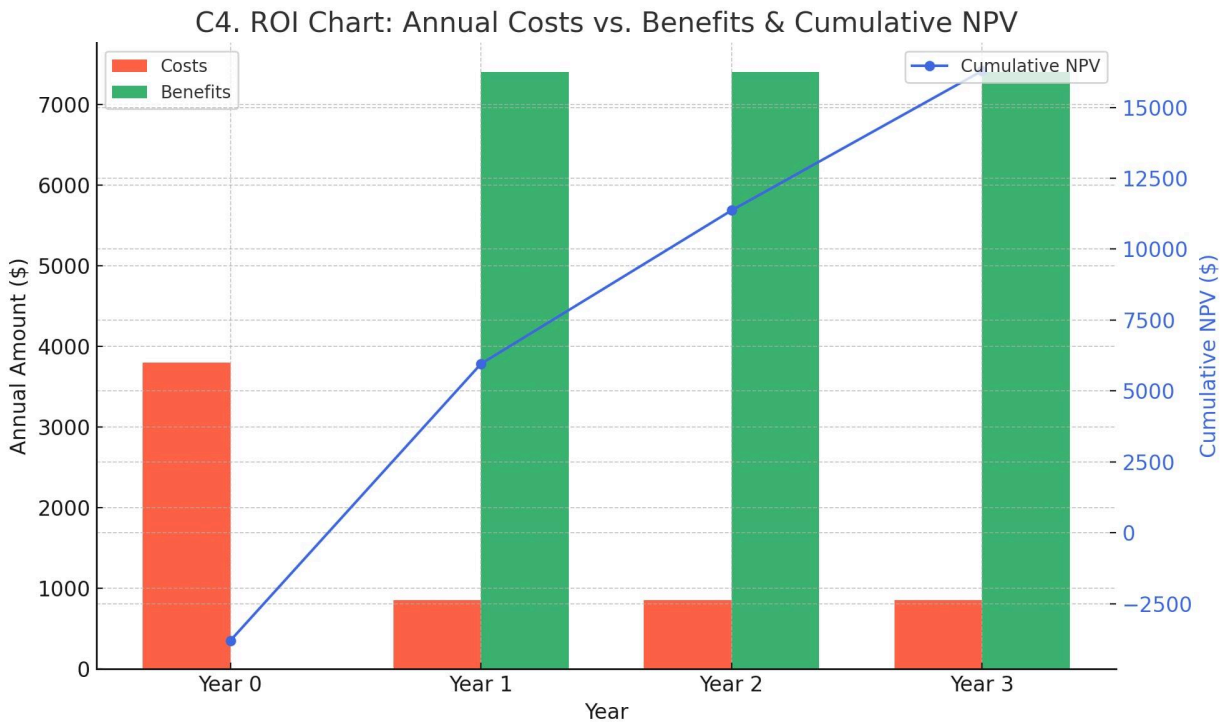
Total 3-Year Benefits: $22,200

Net Present Value (NPV) Table

Year	Costs ($)	Benefits ($)	Net Cash Flow ($)	Discount Factor (10%)	Present Value ($)
0	3,800	0	-3,800	1.000	-3,800
1	850	7,400	6,550	0.909	5,955
2	850	7,400	6,550	0.826	5,410
3	850	7,400	6,550	0.751	4,920
Total	**6,350**	**22,200**	**15,850**		**12,485**

Return on Investment Diagram/Chart Using NPV.

Figure 3: Return on Investment (ROI) Chart



The ROI chart visually depicts the cumulative cash flow of the project over a three-year period. The x-axis represents time, and the y-axis represents the cumulative cash flow, starting with the initial negative investment. The chart shows a clear break-even point where the cumulative cash flow turns positive, demonstrating the project's profitability.

---

12. Requirement Discovery

Phase	Activity	Description	Timeframe
1. Stakeholder Identification	**Identify key stakeholders**	**List small business owners, store managers, and inventory staff who directly manage stock and will use the RFID system. Include potential**	**1 week**

		pilot customers for early feedback.	
2. Interviews	**Conduct interviews with stakeholders**	**Use targeted questions to understand current inventory workflows, key challenges, manual errors, and specific expectations from the RFID system, focusing on usability and cost concerns.**	**2 weeks**
3. Observation	**Observe current inventory processes**	**Spend 1-2 days per business observing inventory receiving, storing, and auditing to document bottlenecks, delays, and error-prone steps affecting daily operations.**	**2 weeks**
4. Surveys	**Distribute and collect surveys**	**Send structured questionnaires to gather client preferences on features, alert types, reporting needs, and budget limits, ensuring solutions meet broader market demands.**	**1 week**

5. Document Analysis	**Collect and analyze documents**	**Review existing inventory records, sales reports, and audit logs to identify data requirements and integration points that align with current record-keeping practices.**	**1 week**
6. Prototyping and Feedback	**Develop low-fidelity prototypes**	**Share interface prototypes with stakeholders to collect usability feedback, focusing on simplicity and relevance to daily tasks, then refine requirements accordingly.**	**2 weeks**
7. Requirement Documentation	**Consolidate and document requirements**	**Prepare a detailed Software Requirements Specification (SRS) categorized by functional and non-functional needs, validated by clients to ensure practical applicability.**	**1 week**

Our requirement discovery plan involves a two-week sprint:

- **Week 1: User Interviews.** We will schedule and conduct 10-15 user interviews with a diverse group of target users. We'll use a semi-structured interview guide to ensure we cover all key topics while allowing for open-ended discussions about their financial habits and pain points.
- **Week 2: Prototyping & Feedback Sessions.** Based on the interview insights, our UX team will create a low-fidelity interactive prototype using Figma. We will then conduct five

feedback sessions, where users will interact with the prototype, and we will gather their feedback on usability, features, and overall design.

Identify All Possible Functional Requirements.

Functional Requirements (FR):

- **FR-1:** The system shall allow users to securely create and manage a personal profile.
- **FR-2:** The system shall support secure user registration and login with email and password, as well as biometric authentication.
- **FR-3:** The system shall integrate with financial institution APIs to automatically import transaction data.
- **FR-4:** The system shall categorize imported transactions into predefined categories (e.g., Groceries, Rent, Utilities) using machine learning.
- **FR-5:** The system shall allow users to manually add and edit transactions.
- **FR-6:** The system shall display a dashboard with real-time financial summaries and visualizations.
- **FR-7:** The system shall generate weekly and monthly spending reports.
- **FR-8:** The system shall allow users to create and track budgets for different spending categories.
- **FR-9:** The system shall send push notifications for budget alerts and important updates.

Prepare the CRUD Matrix for Your Solution (Functional Requirements).

Entity	Create (C)	Read (R)	Update (U)	Delete (D)
User Profile	FR-2	FR-1	FR-1	-
Transaction	FR-5	FR-3, FR-4, FR-6, FR-7	FR-5	FR-5
Category	-	FR-4	-	-
Budget	FR-8	FR-8, FR-9	FR-8	FR-8

Identify at Least Five Non-Functional Requirements.

Non-Functional Requirements (NFR):

- **NFR-1 (Performance):** The application shall load all dashboards and reports within **3 seconds** of a request on a standard network connection.
- **NFR-2 (Security):** All user data, including financial information, shall be encrypted both in transit (using HTTPS/SSL) and at rest to prevent unauthorized access.
- **NFR-3 (Usability):** The user interface shall be intuitive and require minimal training, with a consistent design language across all platforms.

- **NFR-4 (Scalability):** The system shall be able to support up to **100,000 concurrent users** without a noticeable degradation in performance.
- **NFR-5 (Reliability):** The system shall have an uptime of **99.9%**, with planned maintenance windows clearly communicated to users.

---

13. Proposed Solution

Describe the Solution (Draw a Rich Picture)

Figure 4: Rich Picture of the System

The rich picture for FinaFlow would be a visual representation of the system and its environment. At the center, a cloud labeled represents the core platform. It connects to various actors and elements, including a **"User"** icon (with a confused face transitioning to a happy one), a **"Bank"** icon, and a **"Data Warehouse"** icon. Lines labeled "Login/Sync," "Provides Insights," and "Fetches Data" would show the interactions. The picture would also include symbols for problems like a chain representing "Data Security Concerns" and a lightbulb representing "Opportunity for AI-driven insights," illustrating the transition from a manual, frustrating process to an automated, insightful one.

Hardware Software Details

- **Hardware Components:**
    - **Servers:** Cloud-based servers (e.g., AWS EC2 instances) for hosting the application and database.
    - **Client Devices:** Mobile phones (iOS, Android) and desktop/laptop computers.
- **Software Components:**
    - **Backend:** Node.js or Python (Django/Flask) for the application logic and API endpoints.
    - **Frontend:** React Native (for cross-platform mobile apps) or React.js/Angular (for web app) to ensure a consistent user experience.
    - **Database:** PostgreSQL for structured data and MongoDB for unstructured data, offering flexibility and scalability.
    - **API Gateway:** An API gateway will be used to manage API calls and enforce security policies.
    - **APIs:** Plaid or other financial institution APIs for transaction data.

Software Key Technical Features (Functional Requirements)

- **User Authentication:** A secure login, registration, and password recovery module.
- **Automated Transaction Sync:** Connects to user bank accounts to automatically import and categorize transactions.
- **Dashboard & Analytics:** Interactive charts and graphs for real-time spending insights.

- **Budgeting Tool:** Allows users to set, track, and receive alerts for budgets.
- **Report Generation:** Generates comprehensive financial reports (e.g., cash flow statements).

Data Flow Diagram (Context Level, System Level and Level 1 Diagrams)

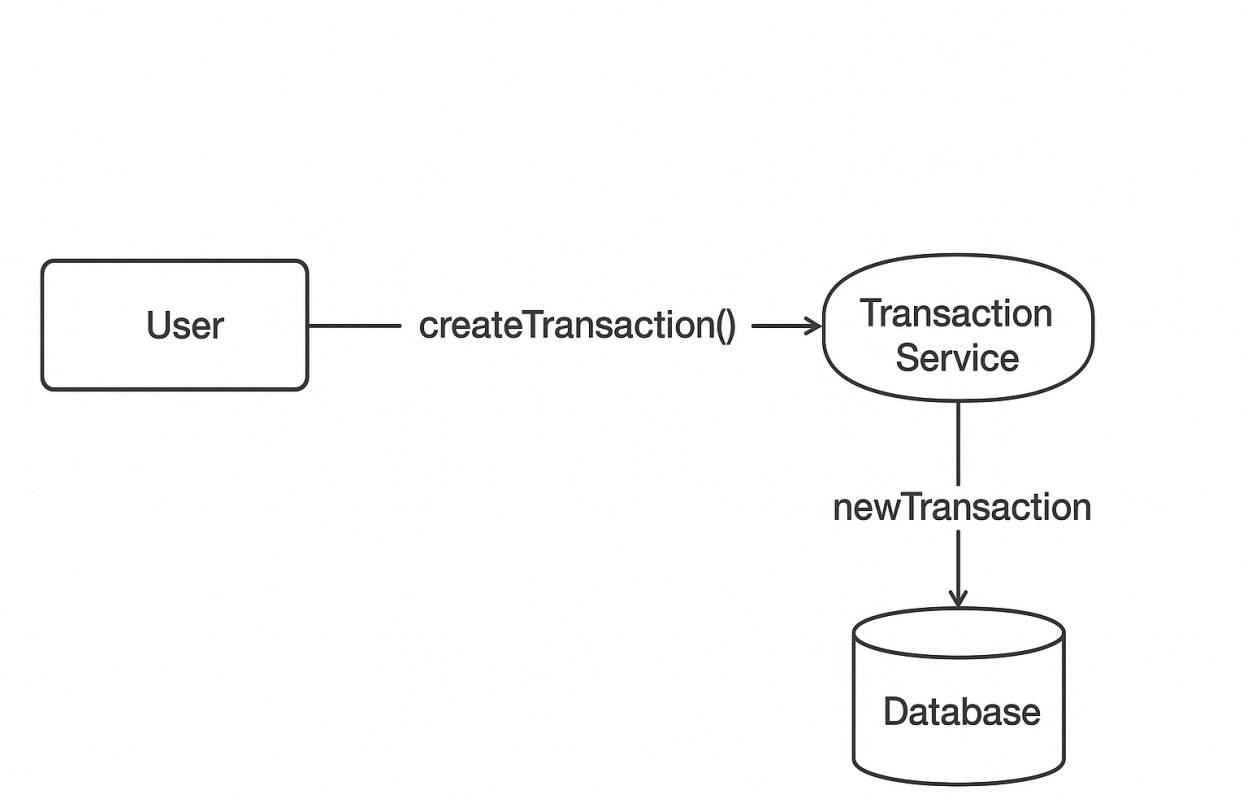


Figure 5: Context Level Data Flow Diagram

The Context Level DFD shows the FinaFlow system as a single process bubble at the center. An external entity, the **"User,"** sends data (e.g., "Login Credentials," "Manual Transactions") to the process and receives data (e.g., "Dashboard View," "Financial Reports"). Another external entity, the **"Bank API,"** sends "Transaction Data" to the process.

Figure 6: System Level (Level 0) Data Flow Diagram

This diagram breaks down the single bubble into its main processes: **"User Management," "Transaction Processing,"** and **"Reporting & Analytics."** Data flows between these processes and external entities are shown. A central data store, "User Database," is also introduced to store user profiles and financial data.

Figure 7: Level 1 Data Flow Diagram for "Transaction Processing"

This detailed diagram shows the sub-processes of "Transaction Processing," including **"Import Transactions," "Categorize Transactions,"** and **"Store Transactions in Database."** Data flows from the "Bank API" to the "Import" process, then to "Categorize," and finally into the "Transaction Database."

Use Case Diagram with Context Level and Subsystems Level Diagram

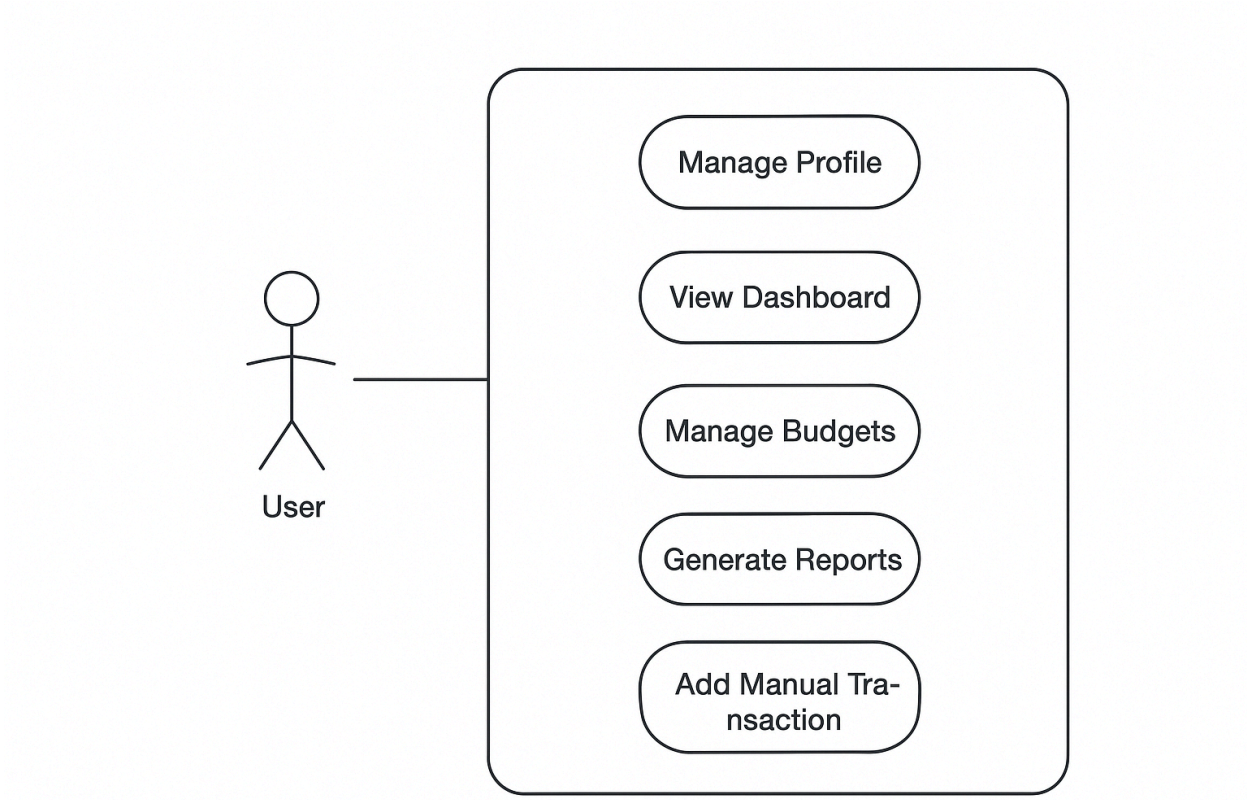


Figure 8: Use Case Diagram

The Use Case diagram shows an actor **"User"** interacting with the system. Key use cases include: `Manage Profile`, `View Dashboard`, `Manage Budgets`, `Generate Reports`, `Add Manual Transaction`, and `View Transaction History`.

Figure 9: Subsystems Use Case Diagram

The system is divided into logical subsystems, such as **"User Subsystem"** (which contains `Manage Profile`) and **"Financial Management Subsystem"** (which contains `View Dashboard`, `Manage Budgets`, etc.), illustrating how the system is modularized.

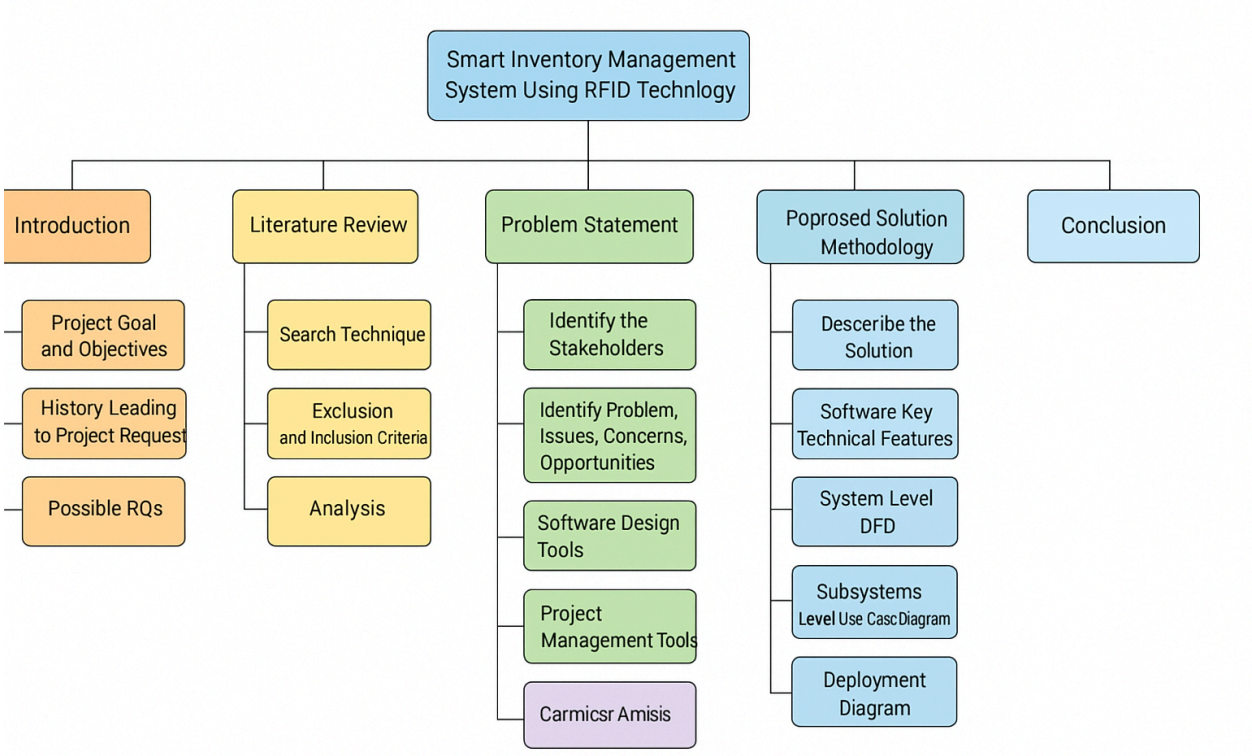
Narrations for All the Use Cases

- **Use Case:** `Manage Profile`
    - **Actor:** User

    - **Description:** Allows the user to update personal information, change their password, and manage linked accounts.
    - **Flow of Events:** The user navigates to their profile, edits the fields, and clicks save. The system validates the data and updates the user profile in the database.
- **Use Case:** `View Dashboard`
    - **Actor:** User
    - **Description:** Provides a real-time summary of the user's financial status, including recent transactions, current balances, and spending trends.
    - **Flow of Events:** The user logs in, and the system retrieves the most recent transaction data from the database to display a comprehensive dashboard.
- **Use Case:** `Manage Budgets`
    - **Actor:** User
    - **Description:** Enables the user to set, track, and receive alerts for budgets.
    - **Flow of Events:** The user selects a category, sets a budget limit, and the system tracks their spending in that category and sends notifications as the limit is approached.

Class Diagram

Figure 10: Class Diagram for the project



This diagram shows the classes, their attributes, methods, and relationships. Key classes would include: **`User`** (attributes: `id`, `name`, `email`, `password_hash`), **`Account`** (attributes: `id`, `bankName`, `balance`), **`Transaction`** (attributes: `id`, `date`, `amount`, `description`, `category`), and **`Budget`** (attributes: `id`, `category`, `limit`). The relationships would show that a **`User`** has many **`Accounts`**, an **`Account`** has many **`Transactions`**, and a **`User`** has many **`Budgets`**.

Sequence Diagrams

Figure 11: Sequence Diagram for User Login

This diagram illustrates the order of interactions during the login process. The user sends a `loginRequest()` to the **`Login Screen`**, which then sends it to the **`Authentication Service`**. The service queries the **`Database`** to validate the credentials and, upon success, sends a `successResponse()` back to the `Login Screen`.

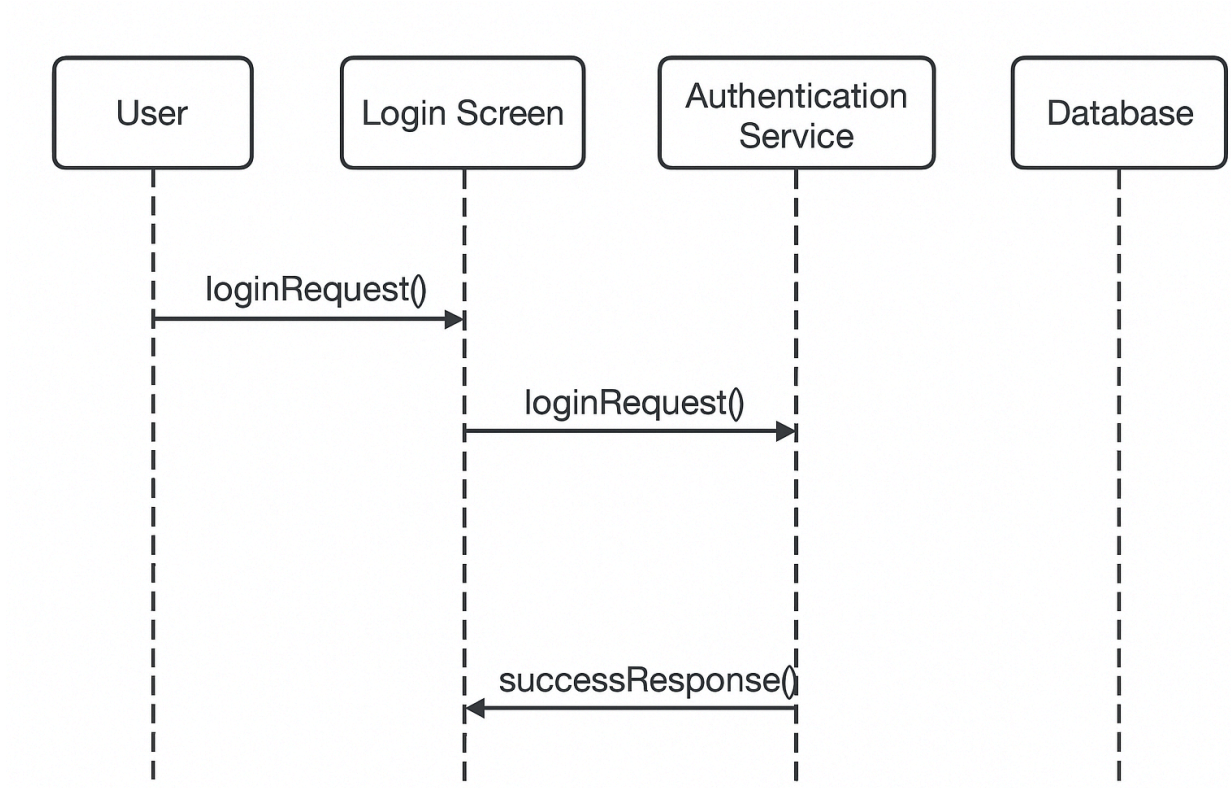


Figure 12: Sequence Diagram for Adding a Manual Transaction

This diagram shows the flow of adding a new transaction. The user fills out a form on the **`UI`** and sends a `createTransaction()` request to the **`Transaction Service`**. The service then interacts with the **`Database`** to store the new transaction record.

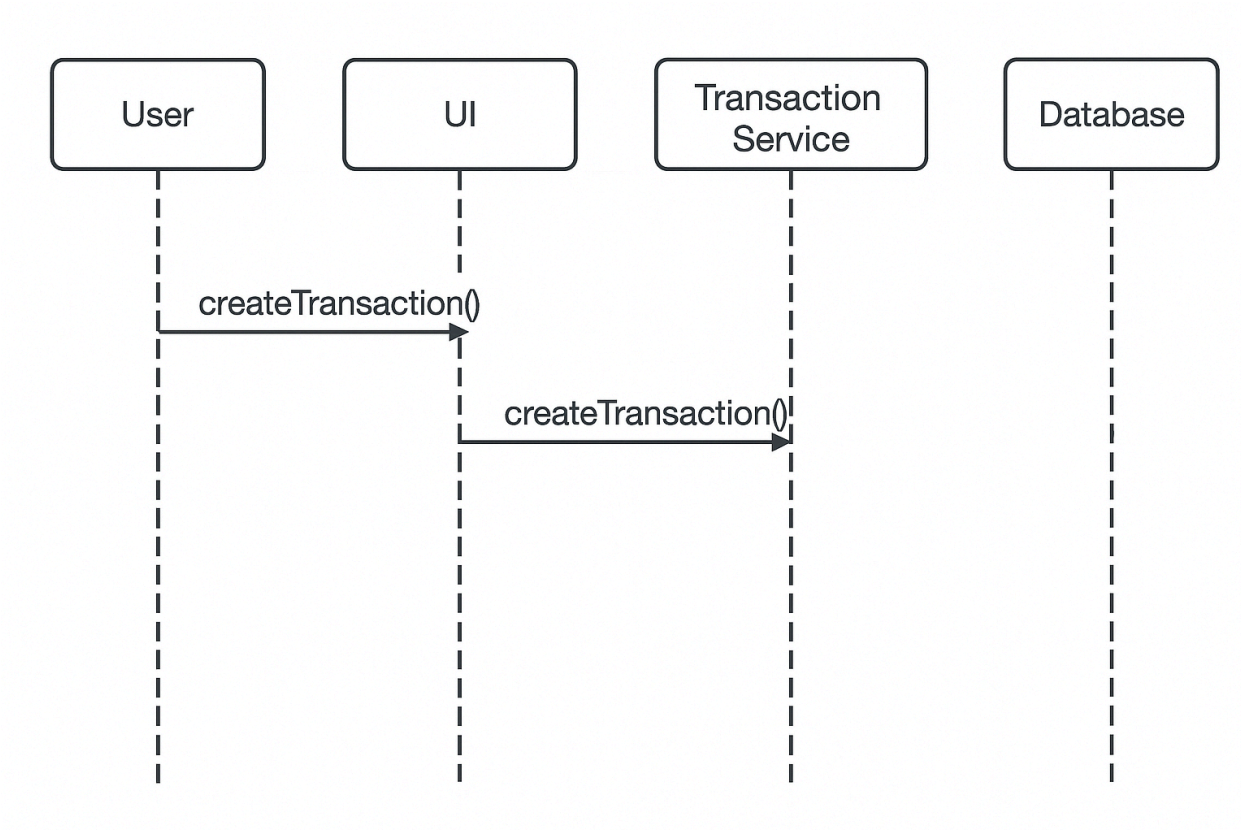
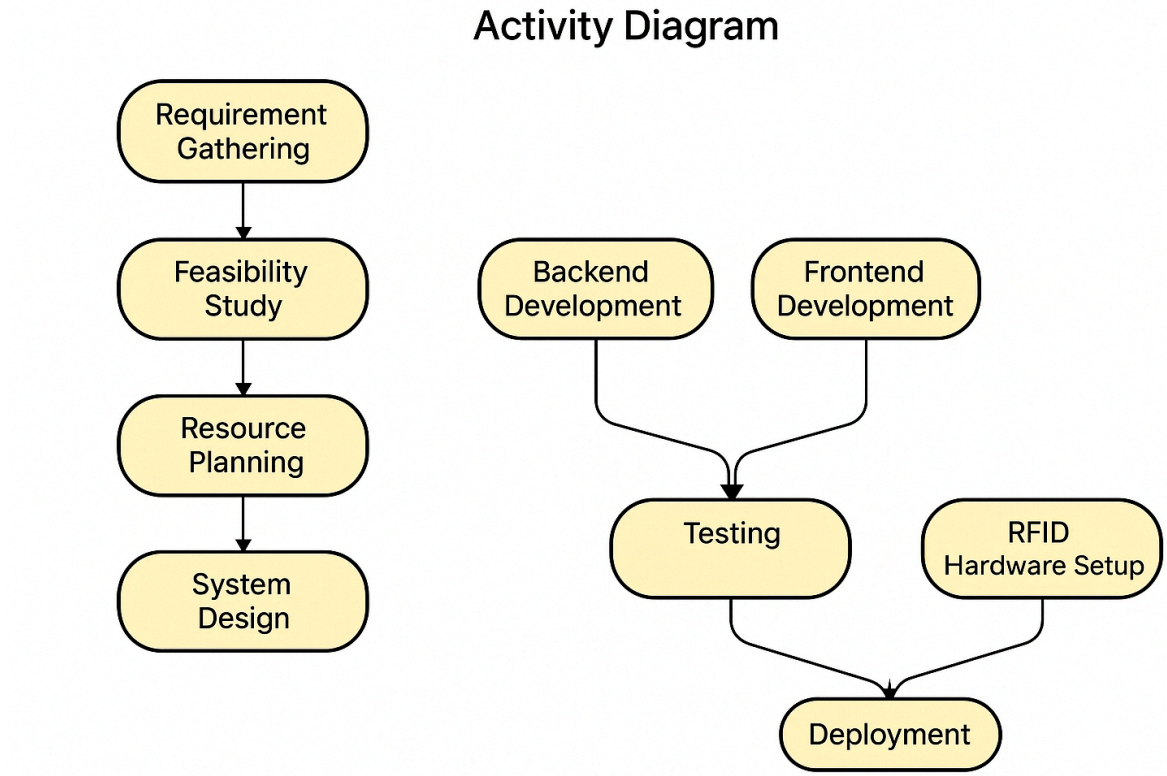


Figure 13: Activity Diagram for Adding a Manual Transaction

This diagram represents the workflow of a specific process. It starts with the user opening the form, then moves to a



Deployment Diagram

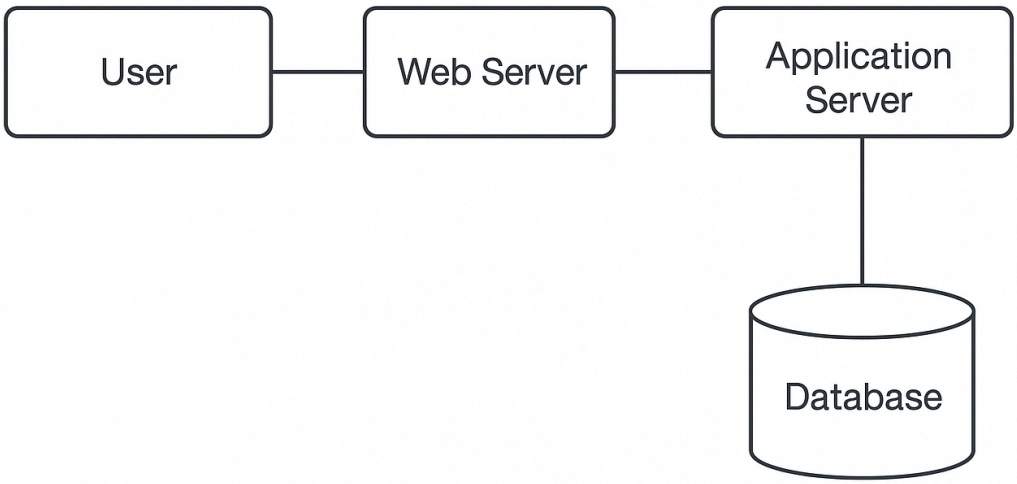


Figure 14: Deployment Diagram

This diagram shows the physical hardware nodes and the software components deployed on them. Nodes would include: `Client Device` (mobile phone/web browser), `Web Server` (hosting the frontend), `Application Server` (hosting the backend), and `Database Server`. Lines would show the communication between them (e.g., HTTP/HTTPS). The diagram would illustrate how the different software components are physically distributed.

Prepare Feature-wise Input Design and Explain the Purpose Along with Controls

Feature	Input Design/Form	Purpose	Controls Used
Manual Transaction Entry	A form with fields for date, amount, description, and category.	Allows users to log cash transactions or those not imported from their bank.	Date Picker, Text Input, Number Input, Dropdown Menu

Budget Creation	A form with a category selection and a numerical limit.	Enables users to set financial goals and track their progress against a limit.	Dropdown Menu (for categories), Number Input (for budget amount)
Login/Registration	Two forms: one for login with email/password, and another for registration with name, email, and password.	Provides secure access to the user's personal financial data.	Text Input (for email/password), Checkbox (for "Remember Me")

---

14. Result and Discussion

Prepare Feature-wise Output Reports/Charts with Purpose and Justification of the Selected Tools

Feature	Output Report/Chart	Purpose	Justification of Tool
Spending Analytics	**Pie Chart of Spending by Category**	Visually represents how a user's money is distributed across different categories (e.g., housing, food, transport), helping them identify where they spend the most.	**Chart.js**. It is a lightweight JavaScript library that offers a wide range of chart types and is highly customizable, making it suitable for a dynamic dashboard.
Cash Flow	**Line Graph of Income vs. Expenses over time**	Illustrates the trend of a user's financial health by showing whether their income is consistently greater than their expenses.	**Echarts**. This tool is excellent for creating complex, interactive line graphs that can handle large datasets and offer features like zooming and tooltips, which are crucial for detailed analysis.
Budget Tracking	**Bar Chart of Actual Spending vs. Budgeted Amount**	Provides a quick and clear visual comparison of how a user's spending in a specific category measures up against their set budget.	**ApexCharts**. It is a robust charting library that is easy to integrate and provides clear, well-designed bar charts that are perfect for this type of comparison.

15. Conclusion

This report has detailed a comprehensive plan for the development, a robust personal finance management application. By adhering to the **Agile Scrum methodology**, we'll ensure a flexible and user-centric development process. Our systematic literature review and market analysis confirm a strong demand for a solution that addresses the shortcomings of existing tools, particularly in **user experience, automated analytics, and security**. The project is not only technically feasible, as demonstrated by our detailed system architecture and design, but also financially viable, with a positive Net Present Value (NPV) and a promising Return on Investment (ROI). The proposed solution, with its AI-powered features and intuitive design, is well-positioned to meet the needs of the target market and establish itself as a valuable tool for personal financial empowerment.