Bank Management System Requirements Specification

Version 3.0

March 8th, 2025

Project Overview //Visi and Fabio

1. Project Title: Bank Management System

2. Introduction:The Bank Management System (BMS) is a software solution designed to automate and streamline banking operations, ensuring efficient customer service, secure transactions, and regulatory compliance. This system will allow customers to manage their accounts, transfer funds, apply for loans, and receive notifications while enabling bank employees to manage transactions, loan approvals, and account security.

3. Objectives:

•To provide secure and efficient banking services for customers.

•To automate fund transfers, account management, and loan applications.

•To enhance security with encryption and fraud detection mechanisms.

•To improve customer experience through an intuitive online banking platform.

•To ensure compliance with financial regulations and reporting requirements.

4. Scope of the Project:

Included Features:

✓ Customer Features:

•Secure Login & Authentication (with OTP)

•Account Overview (Balance, Transaction History)

•Fund Transfers (Internal & External)

•Loan Application & Repayment Tracking

•Notifications & Alerts

✓ Bank Employee Features:

•Customer Management (New Accounts, Profile Updates)

•Transaction Monitoring & Approvals

•Loan Processing & Approval System

•Report Generation

✓ Admin Features:

•User & Role Management

•Fraud Detection & Security Logs

•Compliance & Audit Reporting

Excluded Features:

✕ Cryptocurrency transactions

✕ Stock market trading integration

✕ ATM management

5. Target Users:

•Bank Customers: Individuals & businesses managing their finances.

•Bank Employees: Customer service agents, loan officers, and account managers.

•System Administrators: IT personnel managing security and compliance.

Product/Service Description //Alonso, David and Enklajd

In this section, describe the general factors that affect the product and its requirements. This section should contain background information, not state specific requirements (provide the reasons why certain specific requirements are later specified).

1. Product Context

How does this product relate to other products? Is it independent or self-contained? Does it interface with a variety of related systems? Describe these relationships or use a diagram to show the major components of the larger system, interconnections, and external interfaces.

2. User Characteristics

Create general customer profiles for each type of user who will be using the product. Profiles should include:

- Student/faculty/staff/other
- experience
- technical expertise
- other general characteristics that may influence the product

3. Assumptions

List any assumptions that affect the requirements, for example, equipment availability, user expertise, etc. For example, a specific operating system is assumed to be available; if the operating system is not available, the Requirements Specification would then have to change accordingly.

4. Constraints and Dependencies

List constraints and dependencies that affect the requirements. Examples:

- parallel operation with an old system
- audit functions (audit trail, log files, etc.)
- access, management and security
- criticality of the application

- This new product will require a daily download of data from X,
- Module X needs to be completed before this module can be built.

-First question: How does this product relate to other products?

It's not about how your **Bank Management System** interacts with *other* Bank Management Systems. Instead, it's about how your **Bank Management System interacts with other financial systems** that it needs to function properly. Think of it like this: Your **Bank Management System is the core**, and it **connects** with different **external systems** to provide banking services.

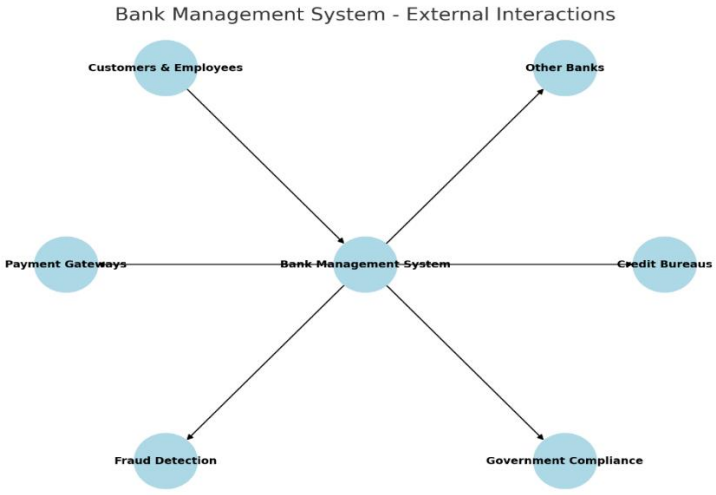
-Second question: Is it independent or self-contained?

I will say that my Bank Management system in order to give me the service that I need it needs to interact with other platforms so it is not independent which means that cannot work on its ow,.

-Third question : Describe these relationships or use a diagram to show the major components of the larger system, interconnections, and external interfaces.

Your **Bank Management System (BMS)** is at the center, and it interacts with multiple external systems to provide banking services. So in few words I mean I would say that first of all as we said eralier, this Bank Management system it shows the interaction with other systems

Here is a simple diagram:



Fourth question: ***User Characteristics***

Create general customer profiles for each type of user who will be using the product. Profiles should include:

- Student/faculty/staff/other

- experience
- technical expertise
- other general characteristics that may influence the product

Different type of users that will interact with our Bank Management System are:

1. Customers (Regular Bank Users)

Profile Type: General public (students, workers, business owners, retirees)
Experience: Varies—some are tech-savvy, others may struggle with online banking.
Technical Expertise: Basic to intermediate (some may be first-time users of digital banking).
General Characteristics:

- Need **simple and user-friendly interfaces**.
- Expect **24/7 access** to banking services.
- May need **customer support** for troubleshooting.
- Security-conscious (prefer **two-factor authentication** and fraud alerts).

2. Bank Employees (Tellers, Account Managers, Loan Officers)

Profile Type: Staff members who manage customer accounts and transactions.
Experience: Experienced in banking but may have different levels of tech skills.
Technical Expertise: Intermediate to advanced (familiar with financial software).
General Characteristics:

- Require **efficient tools** to manage accounts and process transactions.
- Need access to **customer details, transaction history, and account management options**.
- May need **training** for new system updates.
- Must follow **security protocols** (e.g., only accessing relevant customer data).

3. Administrators (System Admins & Bank IT Staff)

Profile Type: IT professionals responsible for system security, user management, and settings.
Experience: Highly experienced in banking IT systems and cybersecurity.
Technical Expertise: Advanced (knowledge of databases, APIs, security protocols).
General Characteristics:

- Have **full control** over system settings, security, and user access levels.
- Manage **software updates, system monitoring, and integrations** with other financial platforms.
- Ensure **data security, encryption, and compliance with banking regulations**.
- Need **detailed reporting and analytics dashboards**.

-Fifth question：*Assumptions*

List any assumptions that affect the requirements, for example, equipment availability, user expertise, etc. For example, a specific operating system is assumed to be available; if the

operating system is not available, the Requirements Specification would then have to change accordingly.

The system is built based on certain assumptions. If any of these assumptions change, we may need to adjust the system requirements.

Possible Assumptions for our Bank Management System

1. **Internet Availability**
    - Assumption: Users and employees will have a stable internet connection to access the system.
    - Impact: If internet access is unreliable, the system may need **offline functionalities** or alternative access methods.
2. **User Expertise**
    - Assumption: Customers and employees have basic digital skills to use the system.
    - Impact: If many users struggle with technology, the system needs to be **extra user-friendly** and offer more **customer support**.
3. **Hardware & Device Compatibility**
    - Assumption: Users will access the system through modern devices (PCs, smartphones, tablets).
    - Impact: If older devices need support, the system might require **lighter versions** or **compatibility adjustments**.
4. **Operating System & Browser Support**
    - Assumption: The system will be accessed on **Windows, macOS, and mobile OS (Android/iOS)** using common browsers (Chrome, Edge, Safari).
    - Impact: If certain OS/browsers aren't available, the system may need **custom apps or additional development**.
5. **Security Measures**
    - Assumption: Users will follow secure practices (e.g., strong passwords, 2FA).
    - Impact: If users neglect security, the system might need **more automated fraud detection or password recovery options**.
6. **Banking Regulations & Compliance**
    - Assumption: The system will follow all legal banking regulations.
    - Impact: If laws change, the system may need **updates** to remain compliant with financial authorities.
7. **High User Load Support**
    - Assumption: The system can handle thousands of users at once without crashing.
    - Impact: If traffic is higher than expected, **server capacity may need to be upgraded**.

Sixth *question: Constraints and Dependencies*

List constraints and dependencies that affect the requirements. Examples:

- parallel operation with an old system
- audit functions (audit trail, log files, etc.)
- access, management and security
- criticality of the application
- This new product will require a daily download of data from X,
- Module X needs to be completed before this module can be built.

This question is asking about **constraints (limitations)** and **dependencies (things the system relies on to function properly)** that affect our **Bank Management System**.

And these limitations are:

1. Constraints (Limitations)

These are restrictions that **affect how the system is designed or built**.

1. **Security & Compliance Requirements**
   - The system **must follow strict banking regulations** (e.g., anti-money laundering laws, customer data protection).
   - **Constraint:** Security protocols (e.g., encryption, two-factor authentication) **must be implemented** before launching.
2. **High Availability & Performance**
   - Banks operate **24/7**, so downtime is **not an option**.
   - **Constraint:** The system **must handle high traffic loads** without slowing down or crashing.
3. **User Access & Authentication**
   - Different users (customers, employees, admins) **have different levels of access**.
   - **Constraint:** The system must enforce **role-based access control (RBAC)** to prevent unauthorized access.
4. **Integration with Existing Systems**
   - The bank may already have an **old system** that must run in **parallel** while transitioning to the new one.
   - **Constraint:** The system must **support data migration and coexistence** with legacy systems.

2. Dependencies (Things the System Needs to Work)

These are **external factors or other systems** that the Bank Management System **depends on**.

1. **Connection to Other Banks (Interbank Transactions)**
   - The system **needs access to other banking networks** for fund transfers.
   - **Dependency:** If external banking APIs are unavailable, **transactions between different banks won't work**.
2. **Payment Gateways (Card & Online Payments)**
   - The system **relies on Visa, MasterCard, and PayPal** for processing payments.
   - **Dependency:** If a payment gateway goes down, **users can't complete transactions**.
3. **Fraud Detection & Security Services**
   - The system depends on **fraud prevention tools** to flag suspicious activities.
   - **Dependency:** If fraud detection services fail, **the bank becomes vulnerable to fraud attacks**.
4. **Regulatory & Compliance Updates**
   - Banking laws change frequently, and the system must adapt.
   - **Dependency:** If government regulations change, **the system may need updates to remain compliant**.

3 Requirements //Flori and Erdi

Functional Requirements

1. Customer shall be able to create a bank account because they need to store and manage their money
2. Customer shall be able to view account balance because they need to check their available funds
3. Customer shall be able to transfer money to another account because they need to make transactions
4. Customer shall be able to request a loan because they need financial assistance
5. Customer shall be able to view loan details because they need to track the status and amount of their loan
6. Customer shall be able to withdraw money from their account because they need access to their funds
7. Customer shall be able to deposit money into their account because they need to increase their balance
8. Customer shall be able to change their account PIN because they need to secure their account
9. Customer shall be able to request a checkbook because they need to make payments via checks
10. Bank Employee shall be able to approve or reject account creation request because accounts must be verified before activation
11. Bank Employee shall be able to view all client accounts because they need to manage customer data
12. Bank Employee shall be able to update client account information because clients may request changes to personal details
13. Bank Employee shall be able to check client loan status because they need to manage loan repayments and approvals
14. Administrator shall be able to manage employee accounts because they need to assign roles and permissions
15. Admin shall be able to monitor all transactions because fraud detection and auditing are necessary
16. Admin shall be able to freeze or close accounts because fraudulent activities or client requests may require account suspension
17. Administrator shall be able to generate reports on transactions because financial oversight
18. System shall be able to validate account details during creation because to prevent errors and fraud
19. System shall be able to send transaction alerts to customers because customers need to be notified of account activity
20. System shall be able to generate monthly account statements for customers because customers need a record of their financial activities
21. System shall be able to apply security measures for online transactions because to prevent unauthorized access
22. System shall be able to set loan eligibility criteria because to ensure that loans are given based on financial capability.
23. System shall be able to apply interest rates to loan accounts because to calculate the amount to be paid back.
24. Customer shall be able to view transaction history because they need to keep track of all their financial transactions.
25. Customer shall be able to set account preferences (e.g., notifications) because they need to customize their experience.
26. Customer shall be able to request account suspension because they may need to temporarily disable their account.
27. Bank Employee shall be able to approve or deny loan applications because they need to assess loan risk.

28. System shall be able to calculate loan repayment schedules because customers need clear repayment plans.
29. System shall be able to perform automatic currency conversion for international transactions because customers need to send and receive funds across borders.
30. System shall be able to block suspicious transactions because fraudulent activities need to be prevented.

Non-Functional Requirements

Product Requirements.

3.2.1.1 Usability Requirements

- The system should be easy to use for customers, requiring minimal training to perform common actions
- The user documentation and help section should be complete and accessible to help users understand all system features
- The system should have an intuitive interface that minimizes the learning curve for new users
- The system should provide helpful error messages when an invalid action is performed

3.2.1.2 Performance Requirements

- The system should be able to handle at least 1000 simultaneous users without degradation in performance
- 95% of transactions shall be processed in less than 1 sec
- The system should be able to process at least 10000 transactions per day during break periods
- The system should handle data storage requirements of up to 1TB of transactionsal data without performance issues

3.2.1.3 Availability

- The system should have an availability of 99.9%, ensuring minimal downtime
- The system should support access from multiple geographic regions, including remote branches and online banking services
- Scheduled maintenance should be communicated to users at least 24 hours in advance, and downtime should not exceed 2 hours per month
- The maximum allowed downtime for unscheduled outages should be 4 hours per year.
- The system should be able to recover from failure within 5 minutes to minimize impact on user activities

3.2.1.4 Security

- All user data and transactions should be encrypted using AES-256 encryption
- The system should log all user activity for auditing and troubleshooting purposes, with logs being stored for a minimum of 1 year.

- Access to the system's backend should be restricted to authorized bank employees only, with role-based access control
- The system should conduct integrity checks on user data and transactional records to ensure no unauthorized modifications occur
- Two-factor authentication should be required for all admin-level users

3.2.2 Organizational Requirements

- The bank's internal policies must be adhered to during the development, ensuring compliance with financial regulations
- The system should be compatible with the bank's existing infrastructure, including hardware and software systems
- The development process must follow the Agile methodology for continuous improvement and rapid feedback cycles
- The system should support integration with existing third-party systems, such as payment gateways and credit scoring providers

3.2.3 External Requirements

- The system should comply with the GDPR for data protection and privacy of EU customers
- The system must comply with industry standards such as PCI DSS for secure handling of payment card data.
- The system should integrate with government systems for tax reporting and financial audits where necessary.
- The system should support interoperability with other banks for cross-bank transfers (e.g., SWIFT, SEPA).
- The system must ensure that all software components comply with local and international banking regulations.

4. User Scenarios/Use Cases

Provide a summary of the major functions that the product will perform. Organize the functions to be understandable to the customer or a first time reader. Include use cases and business scenarios, or provide a link to a separate document (or documents). A business scenario:

- Describes a significant business need
- Identifies, documents, and ranks the problem that is driving the scenario
- Describes the business and technical environment that will resolve the problem
- States the desired objectives
- Shows the "Actors" and where they fit in the business model
- Is specific, and measurable, and uses clear metrics for success

Use cases are associated with a particular Functional Requirement. Assuming you have the first functional requirement named BR_01, you will map it into the Use Case called UC_01 and user scenario US_01. Please keep this naming convention throughout all your use cases and diagrams.

5. Diagrams

In this section you are going to place all of the diagrams that you build throughout to the course, in following with the slides presented throughout the weeks.

5.1 ER Diagram

Standard ERD for your project. Not much but the skills gained in the DBMS course are required.

5.2 Use Case Diagram (general)

Use Case Diagram (only one, with all the use cases).

5.3 Activity Diagram

Each Activity Diagram should be associated with an use case, associated with a particular requirement which is further associated with a particular use-case. E.g BR_01 which becomes UC_01 which becomes AC_01.

5.4. Class diagram.

One class diagram (general) for all the classes. Edit it afterwards with the design pattern implemented in it.

5.5 State diagram

Place all the relevant state diagrams here.

5.6 Sequence diagram.

All sequence diagrams are associated with an Activity Diagram. A Sequence Diagram is built based on an activity diagram. If the activity diagram is named AC_07, the Sequence Diagram will be named SC_07.

5.7. Collaboration diagram

All collaboration diagrams directly relate to a sequence diagram. If a sequence diagram is named SC_07, then the collaboration diagram is named CC_07

6. Design Patterns

Choose the relevant design patterns for your project. For each, give a reasoning and the associated class and sequence diagram. These are NOT part of the above diagrams, and need not carry the following naming scheme.

7. Appendix.

Organizing the Requirements

This section is for information only as an aid in preparing the requirements document.

Detailed requirements tend to be extensive. Give careful consideration to your organization scheme. Some examples of organization schemes are described below:

By System Mode

Some systems behave quite differently depending on the mode of operation. For example, a control system may have different sets of functions depending on its mode: training, normal, or emergency.

By User Class
Some systems provide different sets of functions to different classes of users. For example, an elevator control system presents different capabilities to passengers, maintenance workers, and fire fighters.

By Objects
Objects are real-world entities that have a counterpart within the system. For example, in a patient monitoring system, objects include patients, sensors, nurses, rooms, physicians, medicines, etc. Associated with each object is a set of attributes (of that object) and functions (performed by that object). These functions are also called services, methods, or processes. Note that sets of objects may share attributes and services. These are grouped together as classes.

By Feature
A feature is an externally desired service by the system that may require a sequence of inputs to affect the desired result. For example, in a telephone system, features include local call, call forwarding, and conference call. Each feature is generally described in a sequence of stimulus-response pairs, and may include validity checks on inputs, exact sequencing of operations, responses to abnormal situations, including error handling and recovery, effects of parameters, relationships of inputs to outputs, including input/output sequences and formulas for input to output.

By Stimulus
Some systems can be best organized by describing their functions in terms of stimuli. For example, the functions of an automatic aircraft landing system may be organized into sections for loss of power, wind shear, sudden change in roll, vertical velocity excessive, etc.

By Response
Some systems can be best organized by describing all the functions in support of the generation of a response. For example, the functions of a personnel system may be organized into sections corresponding to all functions associated with generating paychecks, all functions associated with generating a current list of employees, etc.

By Functional Hierarchy
When none of the above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be used to show the relationships between and among the functions and data.

Additional Comments
Whenever a new Requirements Specification is contemplated, more than one of the organizational techniques given above may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.

There are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; and when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.