

coachGrasin

CPD

Software Requirements Specification – ContACT Book AppLication

**VERSION 1.0**

APPROVAL SIGNATURES

| CONTRACTOR | | DATE |
| --- | --- | --- |
| <<Deliverable Owner>>  <<John Doe, Manager>> | <<Signature>> |  |
| <<Vendor Project Manager>> | <<Signature>> |  |
| <<Vendor Managing Director>> | <<Signature>> |  |

| **PROJECT MANAGER** | | | | |
| --- | --- | --- | --- | --- |
| Date: | 12.11.2024 | To: | **Nisarg Nirmalkumar,** Project Manager | |
|  | I approve this deliverable and have no further questions or comments. | | | |
|  | I approve this deliverable conditionally, contingent on the review and approval of the following corrections (see comments). | | | |
|  | I reject this deliverable for the following reasons identified (see comments). | | | |
|  | | | | |
| **<<SIGNATURE>>** | | | | **<<DATE>>** |
| Comments | | | | |
|  | | | | |

DOCUMENT HISTORY

| **DOCUMENT APPROVAL HISTORY** | |
| --- | --- |
| Prepared By |  |
| Reviewed By |  |
| Approved By |  |

| **DOCUMENT REVISION HISTORY** | | | |
| --- | --- | --- | --- |
| Date | Document Version | Revision Description | Author |
| 12/11/2024 | 1.0 | Initial Version |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

TABLE OF CONTENTS

1 INTRODUCTION----------------------------------------------------------------------------------------------------1  
1.1. Purpose of This Document------------------------------------------------------------------------1  
1.2. Scope of This Document----------------------------------------------------------------------------1  
1.3. Overview--------------------------------------------------------------------------------------------------1

2 GENERAL DESCRIPTION-------------------------------------------------------------------------------------------1  
2.1. Key Features----------------------------------------------------------------------------------------------1

3 FUNCTIONAL REQUIREMENTS----------------------------------------------------------------------------------2  
3.1. User Management-------------------------------------------------------------------------------------2  
3.1.1. User Registration and Login------------------------------------------------------------------2  
3.1.2. Account Management----------------------------------------------------------------------------3  
3.1.3. Transfer Money-------------------------------------------------------------------------------------3  
3.1.4. Receive Money----------------------------------------------------------------------------------------3  
3.2. Admin Management----------------------------------------------------------------------------------3  
3.2.1. Admin Login-------------------------------------------------------------------------------------------3  
3.2.2. User Account Management (Admin)-------------------------------------------------------3  
3.2.3. Credit/Debit Accounts (Admin)---------------------------------------------------------------4  
3.2.4. Change User Password (Admin)--------------------------------------------------------------4

4 INTERFACE REQUIREMENTS-------------------------------------------------------------------------------------4  
4.1. User Interface (UI) Requirements--------------------------------------------------------------4  
4.1.1. General UI Layout-----------------------------------------------------------------------------------4  
4.2. External Interface Requirements-------------------------------------------------------------4  
4.2.1. Import/Export Interfaces----------------------------------------------------------------------4  
4.3. System Interfaces-------------------------------------------------------------------------------------5  
4.3.1. Database/Storing Interface-------------------------------------------------------------------5

5 PERFORMANCE REQUIREMENTS------------------------------------------------------------------------------5  
5.1. Response Time-------------------------------------------------------------------------------------------5  
5.1.1. User Login and Account Details-------------------------------------------------------------5  
5.1.2. Transfer and Receive Money-------------------------------------------------------------------5  
5.1.3. Admin Actions----------------------------------------------------------------------------------------5  
5.2. Throughput----------------------------------------------------------------------------------------------6  
5.2.1. Account Creation and Management------------------------------------------------------6  
5.3. System Resource Utilization---------------------------------------------------------------------6  
5.3.1. CPU and Memory Usage--------------------------------------------------------------------------6  
5.4. Availability and Reliability-----------------------------------------------------------------------6  
5.4.1. System Uptime----------------------------------------------------------------------------------------6  
5.4.2. Error Handling and Recovery----------------------------------------------------------------6  
5.5. Load Testing---------------------------------------------------------------------------------------------6  
5.5.1. Stress Testing-----------------------------------------------------------------------------------------6  
5.5.2. System Failover and Redundancy-----------------------------------------------------------7

6 DESIGN CONSTRAINTS-------------------------------------------------------------------------------------------7  
6.1. Platform and Device Compatibility----------------------------------------------------------7  
6.1.1. Cross-Platform Support------------------------------------------------------------------------7  
6.1.2. Device Specifications------------------------------------------------------------------------------7  
6.2. Visual Design Constraints------------------------------------------------------------------------7  
6.3. Data Storage and Management----------------------------------------------------------------7  
6.4. Performance Constraints------------------------------------------------------------------------8  
6.4.1. Latency and Load Time----------------------------------------------------------------------------8  
6.4.2. Adding and Deleting Accounts---------------------------------------------------------------8  
6.4.3. Viewing Account Details------------------------------------------------------------------------8  
6.4.4. Data Import/Export-------------------------------------------------------------------------------8  
6.4.5. CPU and Memory Usage--------------------------------------------------------------------------8  
6.4.6. System Uptime----------------------------------------------------------------------------------------8  
6.4.7. Error Handling and Recovery----------------------------------------------------------------8

7 NON-FUNCTIONAL ATTRIBUTES-------------------------------------------------------------------------------9  
7.1. Security----------------------------------------------------------------------------------------------------9  
7.1.1. Data Protection-------------------------------------------------------------------------------------9  
7.1.2. Access Control---------------------------------------------------------------------------------------9  
7.1.3. Compliance---------------------------------------------------------------------------------------------9  
7.1.4. Incident Response and Recovery-------------------------------------------------------------9  
7.2. Reliability------------------------------------------------------------------------------------------------10  
7.2.1. Availability-------------------------------------------------------------------------------------------10  
7.2.2. Fault Tolerance------------------------------------------------------------------------------------10  
7.2.3. Data Integrity--------------------------------------------------------------------------------------10  
7.3. Usability--------------------------------------------------------------------------------------------------10  
7.3.1. User Experience (UX)------------------------------------------------------------------------------10  
7.3.2. Accessibility------------------------------------------------------------------------------------------10  
7.3.3. Localization-----------------------------------------------------------------------------------------10  
7.4. Maintainability---------------------------------------------------------------------------------------11  
7.4.1. Modularity and Extensibility---------------------------------------------------------------11  
7.4.2. Code Quality-----------------------------------------------------------------------------------------11  
7.4.3. Automated Testing and Continuous Integration---------------------------------11  
7.5. Scalability-----------------------------------------------------------------------------------------------11  
7.5.1. Horizontal Scalability--------------------------------------------------------------------------11

8. Requirement Metrics and Status Reporting 20

APPENDIX A. Overview of Requirements Traceability A-1

APPENDIX B. Example Requirements Tagging, Attributes, and Relationships B-1

APPENDIX C. Traceability Metrics and Reporting C-1

APPENDIX D. COTS, MOTS, AND CUSTOM CONSIDERATIONS D-1

# Introduction

## Purpose of this Document

This document provides a structured approach to developing a banking application, ensuring secure, functional, and efficient software. It covers the software development life cycle (SDLC) stages, including requirement gathering, design, development, testing, deployment, and maintenance. By following this structured methodology, the application will meet functional and security expectations while ensuring reliability and scalability.

## Scope of this Document

The banking application allows users to manage their accounts effectively. It provides functionalities such as user authentication, account transactions, and administrative control. Users can log in, view their account details, transfer money, and receive money. Administrators have additional control, including creating and deleting users, assigning account numbers, managing transactions (credit/debit), and resetting user passwords. The system ensures a secure and seamless experience for users and administrators while maintaining data integrity and security.

## Overview

This document outlines each phase of the SDLC, ensuring the banking software's successful implementation. The phases include requirement analysis, where customer needs are documented; planning, which defines the project's scope; system design, where architectural decisions are made; implementation, which involves coding and development; testing, to ensure functionality and security; deployment, where the software is launched; and maintenance, ensuring continuous system improvements. Following this SDLC methodology ensures the development of a robust, user-friendly, and secure banking application.

# GenERAL Description

The banking application is designed to provide a secure and efficient platform for users to manage their accounts. It simplifies financial transactions while ensuring data protection and integrity. The software facilitates secure login, account management, and transaction processing for both users and administrators.

**Key Features:**

* **User Account Management:** Users can create accounts, log in, view account details, and manage their balance securely.
* **Transaction Processing:** Users can transfer and receive money efficiently, with transaction logs maintained for security and auditing.
* **Admin Control Panel:** Administrators have full control over user management, including creating and deleting accounts, assigning account numbers, modifying balances (credit/debit), and resetting passwords.
* **Security Measures:** The system implements encryption and authentication mechanisms to ensure the safety of user credentials and financial data.
* **User-Friendly Interface:** A simple and intuitive UI allows users to perform tasks with ease, enhancing user experience.
* **Performance Optimization:** The software is designed to handle multiple transactions seamlessly, providing a smooth and responsive banking experience.

The primary goal of the banking application is to offer a reliable, secure, and accessible platform for users to perform financial transactions efficiently. Whether for individual users managing their personal accounts or administrators handling multiple accounts, this system aims to streamline banking processes while maintaining the highest security standards.

# FUNCTional Requirements

**3.1. USER MANAGEMENT**

**3.1.1. USER REGISTRATION AND LOGIN**  
• The application must allow users to create an account by entering the following details:

* Username
* Password
* User Mobile Number
* User Email Address  
  • The system should validate the entered data for required fields (username, password, mobile number, email) and ensure data formats are correct (e.g., email format, phone number format).  
  • The application must allow users to log in using their username and password.  
  • The login system should validate the credentials and grant access to the user upon successful login.

**3.1.2. ACCOUNT MANAGEMENT**  
• After login, the system should allow users to view their account details, including:

* Account Number
* Balance  
  • The system should display real-time account details for users, including their current balance and account number.

**3.1.3. TRANSFER MONEY**  
• The application must allow users to transfer money to another account.  
• Users should be able to input the recipient’s account number and the amount they wish to transfer.  
• The system should validate the transaction, including checking the available balance and ensuring the recipient’s account number is valid.  
• Once validated, the system should process the transaction and update the balance of both the sender’s and recipient’s accounts.

**3.1.4. RECEIVE MONEY**  
• The application must allow users to receive money from other users.  
• The system should automatically update the balance of the receiving user after the transaction is complete.

**3.2. ADMIN MANAGEMENT**

**3.2.1. ADMIN LOGIN**  
• The application should have one admin login that allows the administrator to manage user accounts and perform other critical tasks.  
• Admin login should require a separate, secure username and password.

**3.2.2. USER ACCOUNT MANAGEMENT (Admin)**  
• The admin must be able to create new user accounts by assigning:

* Username
* Password
* User Mobile Number
* User Email Address
* Account Number  
  • Admin must be able to delete existing user accounts.  
  • Admin should be able to view, edit, and manage user details, including assigning new account numbers.

**3.2.3. CREDIT/DEBIT ACCOUNTS (Admin)**  
• Admin must be able to credit or debit a user's account balance.  
• Admin must ensure that debiting an account does not result in a negative balance.

**3.2.4. CHANGE USER PASSWORD (Admin)**  
• Admin must have the ability to reset or change a user’s password upon request.  
• The system should prompt the admin to confirm the password change to avoid accidental changes.

# Interface Requirements

**4.1. USER INTERFACE (UI) REQUIREMENTS**

**4.1.1. GENERAL UI LAYOUT**  
• The application should have a clean, simple, and intuitive layout with easy navigation.  
• The main screen should display the user’s account details, including balance, account number, and transaction options (transfer money, view balance).  
• The application should provide separate views for users and admin to access their respective functionalities.  
• The interface should be designed to minimize the number of steps required to perform common tasks, such as transferring money or viewing account details.  
• The login screen should allow users to input their username and password in a user-friendly manner.

**4.2. EXTERNAL INTERFACE REQUIREMENTS**

**4.2.1. IMPORT/EXPORT INTERFACES**  
• The application should provide an interface for importing and exporting user and transaction data.  
• For importing, users should be able to upload files containing relevant information, such as user accounts (e.g., CSV, TXT).  
• For exporting, users should be able to download their transaction or account data in CSV or TXT format, which can be opened and used in other applications for reporting or backup purposes.

**4.3. SYSTEM INTERFACES**

**4.3.1. DATABASE/STORING INTERFACE**  
• The application should use a backend file (e.g., CSV, TXT, or database) to store user account and transaction information.  
• The file or database interface should support CRUD (Create, Read, Update, Delete) operations for user accounts and transaction data.  
• The system must ensure that all sensitive user information is securely stored and, if necessary, encrypted to protect user privacy.  
• The file or database should maintain data integrity, preventing data loss or corruption during usage (e.g., after a transfer or account update).  
• The system should periodically back up the data to avoid data loss in case of unexpected shutdowns or crashes.

# Performance Requirements

**5.1. RESPONSE TIME**

**5.1.1. USER LOGIN AND ACCOUNT DETAILS**  
• The system must authenticate user login and display account details within 1 second under normal conditions.  
• If the user has multiple transactions or data, the system must still load the account information within 3 seconds.

**5.1.2. TRANSFER AND RECEIVE MONEY**  
• The system must process money transfers or receipt of money within 3 seconds of confirming the transaction, ensuring that the balance updates in real-time.  
• The system should provide real-time feedback to users during transfer and receipt actions (e.g., confirmation messages).

**5.1.3. ADMIN ACTIONS**  
• Admin actions such as creating users, updating accounts, or changing passwords must reflect the changes in the system within 1 second after submitting the action.  
• Admin actions that involve updating multiple user accounts should still complete within 5 seconds.

**5.2. THROUGHPUT**

**5.2.1. ACCOUNT CREATION AND MANAGEMENT**  
• The application should be capable of creating and managing up to 1,000 user accounts per minute without any noticeable delay.  
• For large batch account operations (e.g., creating or updating accounts), the system should handle up to 10,000 users within 5 minutes.

**5.3. SYSTEM RESOURCE UTILIZATION**

**5.3.1. CPU AND MEMORY USAGE**  
• The application should operate efficiently, ensuring that CPU usage does not exceed 30% during typical operations, including logging in, transferring money, or retrieving user data.  
• Memory consumption must remain below 200 MB when managing up to 10,000 users and their transactions.

**5.4. AVAILABILITY AND RELIABILITY**

**5.4.1. SYSTEM UPTIME**  
• The system should maintain a 99.9% uptime over a rolling 30-day period, ensuring the banking application is accessible for users at all times.  
• Scheduled maintenance windows should occur during off-peak hours and should not exceed 2 hours per month.

**5.4.2. ERROR HANDLING AND RECOVERY**  
• The application must be resilient to common errors such as database failures and network issues, allowing for quick recovery.  
• In the event of an error, the system should display a user-friendly error message with clear instructions on the next steps.  
• Data loss should be minimized, and in case of a failure, the system should ensure that the user’s transaction or account data is not lost.

**5.5. LOAD TESTING**

**5.5.1. STRESS TESTING**  
• The application must undergo stress testing to verify that it can handle sudden spikes in user activity, especially during peak transaction hours.  
• The system should be able to handle at least 5,000 concurrent users without significant degradation in performance.

**5.5.2. SYSTEM FAILOVER AND REDUNDANCY**  
• The application must include failover mechanisms (e.g., database clustering, load balancing) to ensure performance is not impacted by system failures or high traffic.  
• If a failure occurs in one component (e.g., a server), the system should seamlessly redirect traffic to a backup system, ensuring minimal disruption.

# Design Constraints

**6.1. PLATFORM AND DEVICE COMPATIBILITY**

**6.1.1. CROSS-PLATFORM SUPPORT**  
• The application is designed to run only on Windows PCs, specifically Windows 10 or later. There is no cross-platform support for other operating systems (e.g., macOS, Linux).

**6.1.2. DEVICE SPECIFICATIONS**  
• The application must be compatible with both Intel and AMD processors, ensuring it runs efficiently on a wide range of devices.

**6.2. VISUAL DESIGN CONSTRAINTS**  
• The application’s design should minimize the use of high-resolution images or heavy graphics to optimize loading times and conserve system resources.  
• The color palette used in the interface should be simple and neutral to enhance usability and ensure a professional, non-distracting appearance.

**6.3. DATA STORAGE AND MANAGEMENT**  
• The application will store all user data locally using the Windows file system with text-based files (e.g., .txt format).  
• Data integrity and security should be maintained, and the text file format should be readable and accessible for troubleshooting or manual access if needed.

**6.4. PERFORMANCE CONSTRAINTS**

**6.4.1. LATENCY AND LOAD TIME**  
• The application must return search results for any query (e.g., by account number, user name, or transaction history) within 1 second under normal conditions.  
• For searches involving over 1,000 user accounts or transactions, the system should return results within 2 seconds.

**6.4.2. ADDING AND DELETING ACCOUNTS**  
• When a user or admin adds or deletes an account, the system must immediately reflect the changes and display the updated information within 1 second after the action is completed.  
• The application should ensure that the account list is updated without noticeable delay when users perform these actions.

**6.4.3. VIEWING ACCOUNT DETAILS**  
• Viewing the details of a user’s account (including account number, balance, transaction history) should take no longer than 1 second after the user searches the account.

**6.4.4. DATA IMPORT/EXPORT**  
• The application must be capable of importing user account data from external files (e.g., CSV, TXT) at a rate of 100 accounts per second.  
• The export functionality should allow users to export their entire account data (up to 5,000 accounts) within 30 seconds for backup or transfer purposes.

**6.4.5. CPU AND MEMORY USAGE**  
• The application should operate efficiently, ensuring that CPU usage does not exceed 30% during typical operations, including account searches, transfers, or transactions.  
• Memory consumption must remain within 200 MB when managing up to 10,000 user accounts and their associated data.

**6.4.6. SYSTEM UPTIME**  
• The system should maintain a 99.9% uptime over a rolling 30-day period, ensuring that the application is consistently available for use by all users.  
• Maintenance windows for software updates and patches should be scheduled during off-peak hours and should not exceed 1 hour per month to minimize disruption.

**6.4.7. ERROR HANDLING AND RECOVERY**  
• The system must be resilient to common errors (e.g., database failures or network interruptions) and should recover gracefully, ensuring minimal impact on user experience.  
• If an error occurs, the system should display a clear and user-friendly error message with instructions on how to resolve the issue or contact support.  
• The application should ensure that no critical data is lost during failures, especially during financial transactions or data synchronization.

# Non-functional attributes

**7.1. SECURITY**

**7.1.1. DATA PROTECTION**  
• All user data, including sensitive account information, transaction history, and personal data, must be encrypted both at rest and during transmission to prevent unauthorized access.  
• Strong encryption standards (e.g., AES-256) must be implemented for storing and transmitting user data.

**7.1.2. ACCESS CONTROL**  
• The application must implement role-based access control (RBAC) to ensure that only authorized users (e.g., admins, end-users) can access sensitive information.  
• User authentication must be based on secure methods, such as multi-factor authentication (MFA) for enhanced security.

**7.1.3. COMPLIANCE**  
• The application must comply with privacy and data protection regulations, including GDPR (General Data Protection Regulation), CCPA (California Consumer Privacy Act), and other applicable regulations.  
• Users must have control over their data, including the ability to delete their personal information upon request.  
• The application should notify users about data collection and processing practices during the initial use, including obtaining their consent.

**7.1.4. INCIDENT RESPONSE AND RECOVERY**  
• The system must have logging and monitoring features to detect potential security incidents, such as unauthorized access attempts or data breaches.  
• In case of a breach, the system must notify affected users within 72 hours, following incident response protocols.  
• Recovery procedures must be in place to restore any lost or corrupted data.

**7.2. RELIABILITY**

**7.2.1. AVAILABILITY**  
• The system must ensure 99.9% uptime for both the mobile and web applications, ensuring consistent access to banking services for users.  
• Scheduled maintenance windows should be limited to 1 hour per month, with prior user notification.

**7.2.2. FAULT TOLERANCE**  
• The application should be able to recover gracefully from errors or failures, such as transaction errors or synchronization issues, without data loss.  
• Automated retries and backup systems must be implemented to ensure data integrity and availability.

**7.2.3. DATA INTEGRITY**  
• The system must ensure that user data, including account information and transaction history, remains consistent and accurate at all times.  
• Data integrity checks should be implemented during synchronization and database updates to prevent corruption.

**7.3. USABILITY**

**7.3.1. USER EXPERIENCE (UX)**  
• The application should provide an intuitive and easy-to-use interface, ensuring that users with varying levels of technical knowledge can navigate and interact with the system efficiently.  
• Essential actions, such as logging in, transferring funds, and viewing account details, should require no more than 3 steps to complete.

**7.3.2. ACCESSIBILITY**  
• The application should support keyboard navigation, ensuring it is accessible to users who prefer using a keyboard over a mouse.  
• All interactive elements should be properly labeled for assistive technologies (e.g., screen readers).

**7.3.3. LOCALIZATION**  
• The application should support multiple languages (e.g., English, Spanish, French, and German) to cater to users from different regions.  
• The UI should automatically adapt to language preferences, including localizing date formats, currency symbols, and number formats.

**7.4. MAINTAINABILITY**

**7.4.1. MODULARITY AND EXTENSIBILITY**  
• The application should be built with modular components to facilitate easy updates, bug fixes, and scalability.  
• The architecture should support the addition of new features without requiring significant changes to the core system.

**7.4.2. CODE QUALITY**  
• The codebase must follow clean code principles, ensuring readability, documentation, and easy maintainability.  
• Code should adhere to established coding standards, and version control (e.g., Git) should be used to track changes and facilitate collaborative development.

**7.4.3. AUTOMATED TESTING AND CONTINUOUS INTEGRATION**  
• Automated tests (unit, integration, and functional tests) should be implemented to verify that all components work correctly.  
• A continuous integration (CI) pipeline should be used to automate the build, test, and deployment processes, ensuring quick and safe updates.

**7.5. SCALABILITY**

**7.5.1. HORIZONTAL SCALABILITY**  
• The system should support horizontal scalability, enabling it to handle an increasing number of users and their transactions without a decrease in performance.  
• The database should be capable of scaling to support millions of users and contacts, ensuring consistent performance under load.