**Prompt:**

**Manager:**  
Alright, let’s finalize our plan for the internal banking system. For the first version, we’re keeping it strictly **admin-only** — no customer access yet. The idea is that our internal team should be able to manage accounts, view balances, record deposits or withdrawals, and keep track of transaction history.

**Architect:**  
Got it. So this first phase will focus entirely on the admin interface and basic banking functions — login, account overview, transaction history, and the ability for admins to perform deposits and withdrawals.

**Manager:**  
Exactly. We’ll use our existing SSO system for login so that only authorized admins can access it. If for some reason SSO isn’t available, we can have a backup login system using credentials.

**Architect:**  
That makes sense. We’ll make SSO the default option, and the fallback login will just be for testing or special cases. Once an admin logs in, they should land on a dashboard where they can see all accounts, their current balances, and overall status.

**Manager:**  
Yes, the dashboard should make it easy to search and filter accounts — by name, number, or even account status. Admins should be able to quickly check if an account is active or frozen.

**Architect:**  
Right. And when an admin clicks on an account, they’ll be able to see its detailed information — including transaction history. Each transaction record will include details like date, type (deposit or withdrawal), amount, and any notes or references.

**Manager:**  
Good. For the actual account operations, let’s focus only on **deposits** and **withdrawals** for now. We don’t need to handle direct fund transfers between accounts yet.

**Architect:**  
Understood. So the main actions will be:

* Deposit money into an account
* Withdraw money from an account  
  Both of these will automatically record a transaction and update the account’s balance.

**Manager:**  
Exactly. And we should have some checks in place — like making sure the withdrawal amount doesn’t exceed the account’s balance.

**Architect:**  
Of course. We’ll make sure all balance changes are validated before processing. Deposits will always increase the balance, while withdrawals will reduce it — but not below zero. Everything will happen securely and reliably, so we don’t end up with inconsistencies.

**Manager:**  
Perfect. Also, every action like deposit or withdrawal should have a clear confirmation message — something like “Deposit successful” or “Insufficient funds” if it fails.

And on the transaction list, admins should be able to filter records by date or transaction type.

**Architect:**  
That’s simple enough. We can include a date range picker and a type dropdown. Also, since it’s an internal tool, we’ll make sure it’s straightforward to navigate — login, dashboard, account details, deposit/withdrawal pages.

**Manager:**  
Exactly. No unnecessary complexity — just fast and secure. Also, let’s make sure all communication is secure, so it’s HTTPS only.

And since this is admin-only, restrict access to the allowed domains.

**Architect:**  
Absolutely. All sensitive data like admin credentials or tokens will be kept secure and never exposed. We’ll store them safely in our system environment, and all connections will be encrypted.

We’ll also ensure that all failed login attempts or failed operations return clear but safe error messages, without exposing system details.

**Manager:**  
Good. For the interface, keep it clean — maybe use our standard colors, simple tables for account and transaction lists, and input forms for deposits and withdrawals.

**Architect:**  
Got it. And for project rollout, here’s how I suggest we phase it:

* **Phase 1 (Weeks 1–2):** Authentication (SSO + fallback), Admin dashboard, and Account management.
* **Phase 2 (Weeks 2–3):** Deposit and Withdrawal features, including balance validation and transaction history updates.
* **Phase 3 (Weeks 3–4):** Error handling improvements, performance checks, and preparing for production.

**Manager:**  
That sounds good. Also, make sure it’s easy to deploy on our current setup but flexible enough to move to the cloud later.

And maybe add some demo data for testing — a few sample accounts and transactions so we can walk through the flow easily.

**Architect:**  
Yes, we’ll prepare some seeded data for that.  
So to summarize:

Admins can log in securely, view all accounts, perform deposits and withdrawals, and check full transaction histories — all from one interface. No transfers, no customer login. Just a reliable internal tool for our operations team.

**Manager:**  
Exactly. That’s the version one goal. Let’s move ahead with this plan — once we get the design mockups and the SSO setup finalized, we’ll start development.

**Architect:**  
Perfect. I’ll prepare the requirement document and acceptance criteria based on this plan.

go through the transcript carefully and generate a requirement document for this. The requirement document should be in two part . The first part should contain the requirements and next part should contain the gherkin's user acceptance along with user stories. Skip the technical part from it.  
  
  
  
**Banking System - Internal Admin Portal**

**Requirements Document v1.0**

**Part 1: Functional Requirements**

**1. Authentication and Access Control**

**1.1 Single Sign-On (SSO) Integration**

* The system shall integrate with the existing organizational SSO system as the primary authentication method
* Only authorized administrators shall be granted access to the system
* Access shall be restricted to allowed organizational domains

**1.2 Fallback Authentication**

* The system shall provide a backup credential-based login mechanism
* The fallback login shall be available when SSO is unavailable or for testing purposes
* All login attempts shall be validated and tracked

**1.3 Security Requirements**

* All communications shall use HTTPS encryption
* Sensitive credentials and tokens shall be stored securely in the system environment
* Failed login attempts shall return clear error messages without exposing system details

**2. Dashboard and Account Overview**

**2.1 Admin Dashboard**

* Upon successful login, administrators shall land on a dashboard displaying all accounts
* The dashboard shall show current balances for all accounts
* Account status (active or frozen) shall be visible at a glance

**2.2 Search and Filter Capabilities**

* Administrators shall be able to search accounts by name
* Administrators shall be able to search accounts by account number
* Administrators shall be able to filter accounts by status (active/frozen)

**3. Account Management**

**3.1 Account Details View**

* Administrators shall be able to click on any account to view detailed information
* Account details shall include complete transaction history
* Each transaction record shall display date, transaction type, amount, and reference notes

**3.2 Transaction History Filtering**

* Administrators shall be able to filter transactions by date range
* Administrators shall be able to filter transactions by type (deposit/withdrawal)
* Transaction lists shall be presented in a clear tabular format

**4. Banking Operations**

**4.1 Deposit Functionality**

* Administrators shall be able to deposit money into any account
* Each deposit shall automatically update the account balance
* Each deposit shall create a transaction record with all relevant details
* The system shall display a confirmation message upon successful deposit

**4.2 Withdrawal Functionality**

* Administrators shall be able to withdraw money from any account
* The system shall validate that withdrawal amounts do not exceed available balance
* Withdrawals shall not reduce account balance below zero
* Each withdrawal shall automatically update the account balance
* Each withdrawal shall create a transaction record with all relevant details
* The system shall display appropriate messages for successful or failed withdrawals

**4.3 Balance Validation**

* All balance changes shall be validated before processing
* Deposits shall always increase the account balance
* Withdrawals shall decrease the account balance only if sufficient funds exist
* The system shall prevent inconsistencies in balance calculations

**5. User Interface Requirements**

**5.1 Navigation**

* The interface shall provide straightforward navigation between login, dashboard, account details, and operation pages
* The design shall be clean and minimize complexity
* Standard organizational colors shall be used throughout the interface

**5.2 Data Presentation**

* Account lists shall be displayed in simple, readable tables
* Transaction lists shall be displayed in simple, readable tables
* Input forms shall be provided for deposit and withdrawal operations

**5.3 Feedback and Messaging**

* All operations shall provide clear confirmation messages
* Error messages shall be informative without exposing technical details
* Insufficient funds scenarios shall display appropriate warnings

**6. System Capabilities**

**6.1 Scope Limitations**

* Version 1 shall not support customer login or access
* Version 1 shall not support direct fund transfers between accounts
* The system is strictly for internal administrative use

**6.2 Deployment Requirements**

* The system shall be deployable on current infrastructure
* The architecture shall support future migration to cloud platforms
* Demo data (sample accounts and transactions) shall be available for testing and demonstrations

## User Acceptance Criteria

## Epic 1: Administrator Authentication

## User Story 1.1: SSO Login

As an administrator, I want to log in using my organizational SSO credentials, so that I can securely access the banking system without managing separate passwords.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario | Given | When | Then |
| Successful SSO login | I am an authorized administrator AND I am on the login page | I click "Sign in with SSO" AND I complete the SSO authentication process | I should be redirected to the admin dashboard AND I should see all account information |
| Unauthorized SSO access attempt | I am not an authorized administrator | I attempt to log in via SSO | I should see an "Access Denied" message AND I should remain on the login page |
| SSO from non-allowed domain | I have valid SSO credentials from an unauthorized domain | I attempt to log in | I should see "Access restricted to authorized domains" message AND I should not gain access to the system |

## User Story 1.2: Fallback Login

As an administrator, I want to use credential-based login when SSO is unavailable, so that I can still access the system during SSO outages or for testing.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario | Given | When | Then |
| Successful fallback login | SSO is unavailable or I choose fallback login AND I have valid admin credentials | I enter my username and password AND I click "Login" | I should be redirected to the admin dashboard |
| Failed fallback login with incorrect credentials | I am on the fallback login page | I enter incorrect username or password AND I click "Login" | I should see "Invalid credentials" message AND I should remain on the login page AND system details should not be exposed |

## Epic 2: Dashboard and Account Overview

## User Story 2.1: View All Accounts

As an administrator, I want to see all accounts on my dashboard, so that I can get a quick overview of the banking system status.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario | Given | When | Then |
| View dashboard after login | I am logged in as an administrator | I land on the dashboard | I should see a list of all accounts AND each account should display account number, name, current balance, and status |
| View account status indicators | I am on the dashboard | I view the account list | I should clearly see which accounts are active AND I should clearly see which accounts are frozen |

## User Story 2.2: Search and Filter Accounts

As an administrator, I want to search and filter accounts, so that I can quickly find specific accounts I need to manage.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario | Given | When | Then |
| Search account by name | I am on the dashboard | I enter a customer name in the search field | I should see only accounts matching that name |
| Search account by account number | I am on the dashboard | I enter an account number in the search field | I should see the matching account if it exists |
| Filter accounts by status | I am on the dashboard | I select "Active" from the status filter | I should see only active accounts |
| Filter accounts by status (frozen) | I am on the dashboard | I select "Frozen" from the status filter | I should see only frozen accounts |
| No results found | I am on the dashboard | I search for a non-existent account | I should see "No accounts found" message |

## Epic 3: Account Details and Transaction History

## User Story 3.1: View Account Details

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario | Given | When | Then |
| Access account details | I am on the dashboard | I click on an account | I should see the account details page AND I should see account holder information AND I should see current balance AND I should see account status AND I should see complete transaction history |
| View transaction record details | I am viewing an account's details | I look at the transaction history | Each transaction should display date AND type (deposit or withdrawal) AND amount AND reference notes if available |

As an administrator, I want to view detailed information for a specific account, so that I can

understand the account's complete history and current state.

## User Story 3.2: Filter Transaction History

As an administrator, I want to filter transaction history, so that I can find specific transactions quickly.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario | Given | When | Then |
| Filter transactions by date range | I am viewing an account's transaction history | I select a start date and end date | I should see only transactions within that date range |
| Filter transactions by type (deposits) | I am viewing an account's transaction history | I select "Deposits" from the type filter | I should see only deposit transactions |
| Filter transactions by type (withdrawals) | I am viewing an account's transaction history | I select "Withdrawals" from the type filter | I should see only withdrawal transactions |
| Combined filtering | I am viewing an account's transaction history | I apply both date range and transaction type filters | I should see only transactions matching both criteria |

## Epic 4: Deposit Operations

## User Story 4.1: Deposit Money into Account

As an administrator, I want to deposit money into an account, so that I can increase the account balance when customers make deposits.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario | Given | When | Then |
| Successful deposit | I am viewing an account details page | I click on "Deposit" button AND I enter a valid deposit amount AND I add reference notes AND I confirm the deposit | The account balance should increase by the deposit amount AND a new transaction record should be created AND I should see "Deposit successful" confirmation message AND the transaction should appear in the history with correct details |
| Deposit with invalid amount | I am on the deposit page | I enter zero or negative amount AND I attempt to confirm | I should see "Please enter a valid amount" message AND the deposit should not be processed |
| Deposit to frozen account | I am attempting to deposit into a frozen account | I try to confirm the deposit | I should see an appropriate warning message AND the system should indicate the account status |

## Epic 5: Withdrawal Operations

## User Story 5.1: Withdraw Money from Account

As an administrator, I want to withdraw money from an account, so that I can decrease the account balance when customers make withdrawals.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario | Given | When | Then |
| Successful withdrawal with sufficient funds | I am viewing an account with sufficient balance | I click on "Withdraw" button AND I enter a withdrawal amount less than or equal to the balance AND I add reference notes AND I confirm the withdrawal | The account balance should decrease by the withdrawal amount AND a new transaction record should be created AND I should see "Withdrawal successful" confirmation message AND the transaction should appear in the history |
| Withdrawal attempt with insufficient funds | I am viewing an account | I attempt to withdraw an amount greater than the current balance AND I try to confirm the withdrawal | I should see "Insufficient funds" message AND the withdrawal should not be processed AND the account balance should remain unchanged |
| Withdrawal that would result in negative balance | An account has a balance of 100 | I attempt to withdraw 101 | The system should prevent the withdrawal AND I should see "Insufficient funds" message AND the balance should remain at 100 |
| Withdrawal with invalid amount | I am on the withdrawal page | I enter zero or negative amount AND I attempt to confirm | I should see "Please enter a valid amount" message AND the withdrawal should not be processed |

## Epic 6: System Security and Error Handling

## User Story 6.1: Secure Communication

As an administrator, I want all system communications to be encrypted, so that sensitive banking data remains protected.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario | Given | When | Then |
| HTTPS enforcement | I attempt to access the system via HTTP | I navigate to the system | I should be automatically redirected to HTTPS AND all subsequent communications should use HTTPS |
| Secure credential storage | Admin credentials and tokens exist in the system | The system is running | They should be stored securely in the system environment AND they should never be exposed in responses or logs |

## User Story 6.2: Error Handling

As an administrator, I want clear error messages when operations fail, so that I understand what went wrong without exposing system vulnerabilities.

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario | Given | When | Then |
| Clear error messages on failed operations | An operation fails for any reason | The error occurs | I should see a clear, user-friendly error message AND the message should not expose system internals AND the message should not reveal technical details AND the message should guide me on what to do next |
| Failed login attempts | I enter incorrect credentials | The login fails | I should see "Invalid credentials" message AND the message should not reveal whether username or password was incorrect AND the message should not expose any system paths or configurations |