appical

Backend Engineer

Technical Assessment

Document Owner: CTO
Information Classification: Public

Assessment

Bank Account Management API

Context

This section describes a simplified problem domain for which an ASP .NET Core API must be implemented.

Important
Please disregard authentication and authorization - they are out of scope. We will assume that the API implements the necessary authorization to restrict operations to the correct actors (e.g. Account Owner and Bank Clerk).

Also, please disregard currency when dealing with money - we assume the accounts can only be opened in a single default currency so we only care about the numerical value of money and not about currency.

Technical Requirements

Technical Stack

Please adhere to the following technology stack.

- C# (feel free to use C# 9 or latest including the latest language constructs e.g. init access modifier)
- .NET 5 (or latest)
- ASP .NET Core
- EF Core
    - The database engine selected should be MS SQL not In Memory or another database provider
    - Please use the EF core code-first approach

Unit Testing

Please implement the Unit Tests that you deem necessary. They must be part of the solution provided.

Publishing

Please publish the solution to your own Git repository of choice (GitHub, BitBucket, etc.). It must be publicly accessible so we can review your solution.

Please ensure that everything necessary is included in the repository.

Deployment

If possible, provide the necessary dockerfile / docker-compose files so we can run your solution as Docker container(s).

In any case, please use EF Migrations to ensure that the Database is set up correctly upon first run of the application so we don't have to manually create the necessary tables ourselves.

Please use Swagger (including Swagger UI) to document the API so we can easily evaluate it.

Functional Requirements

General domain knowledge

Consider the following conceptual model when designing the application.

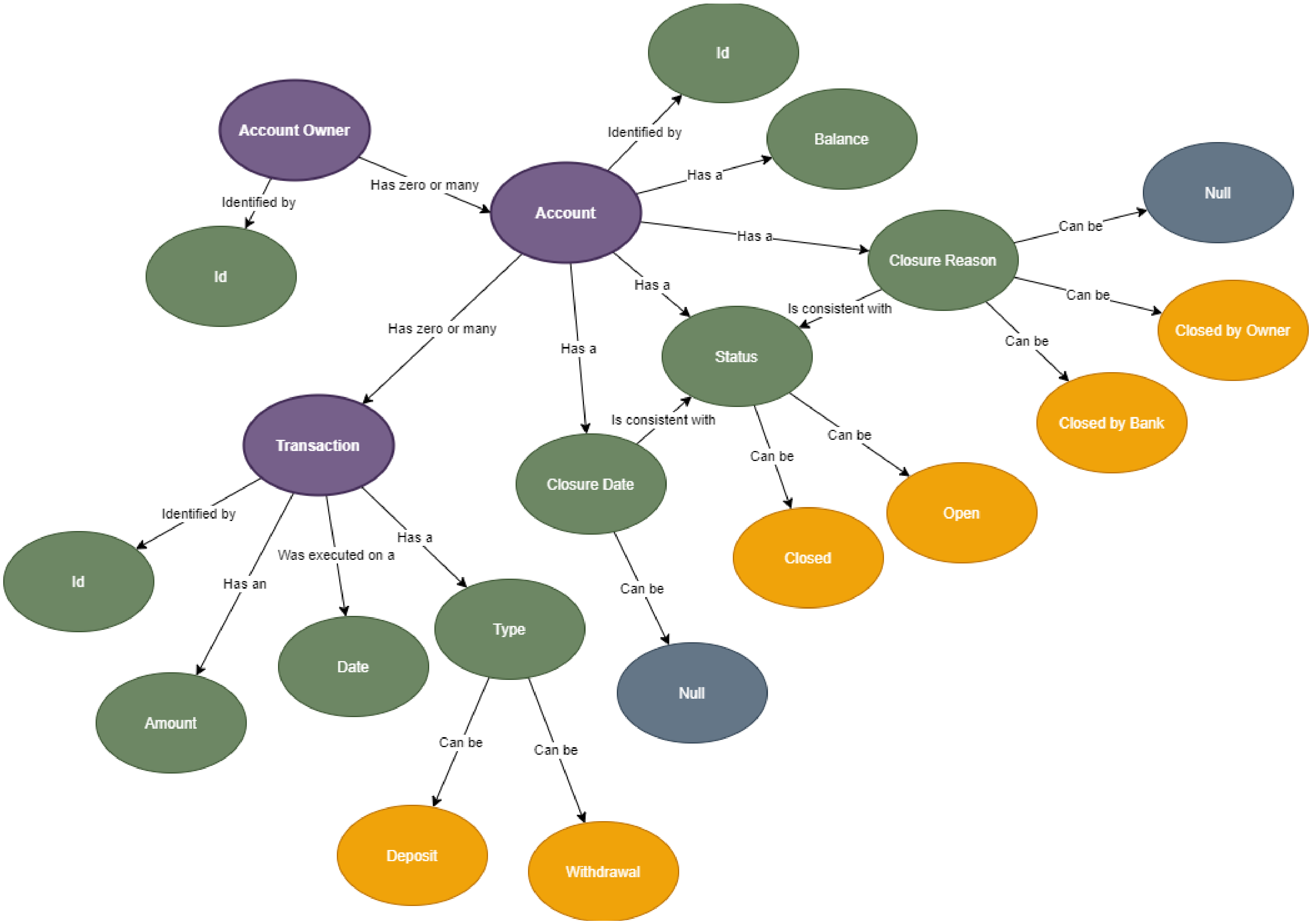

Fig 1. Account Management - Conceptual Model

Use Cases

1. Open a new Account

As a Bank Clerk I can Open an Account for a new Account Owner so they can use it to Deposit and Withdraw cash.

Acceptance Criteria

- A new Account is created for the specified Account Owner with a Balance of 0.
- The new Account has a unique Id
- The new Account has a status of “Open”

Non-functional Criteria

- The new Account is identified by a unique Id described as a Guid
- The Account Owner is described as a Guid which uniquely identifies them

2. Deposit cash into Account

As an Account Owner I can Deposit cash into my Account

Acceptance Criteria

- The Deposited Amount is added to the Account Balance

Non-functional Criteria

- Multiple Deposit and Withdrawal operations can be executed in parallel for the same Account
- A Deposit operation must be idempotent so that it’s safe to retry

3. Withdraw cash from an Account

As an Account Owner I can Withdraw cash from my Account

Business Rules

- An account Balance cannot be negative.

- A Withdrawal Transaction either succeeds for the entire amount or fails due to insufficient funds.

Acceptance Criteria

- The Withdrawal Amount is deducted from the Account Balance.

Non-functional Criteria

- Multiple Deposit and Withdrawal operations can be executed in parallel for the same Account
- A Withdrawal operation must be idempotent so that it’s safe to retry

4. Close an Account

As an Account Owner I can Close my Account

Business Rules

- An account can only be Closed if the Balance is 0.
- Cash cannot be Deposited into or Withdrawn from a Closed Account.
- Once Closed, an Account cannot be re-Opened by the Account Owner (this operation is reserved for Bank Clerks and is out of scope).

Acceptance Criteria

- The Account Status is Closed, the Closure Date is recorded and the Account’s Closure Reason is set to "Closed by Owner" (the value "Closed by Bank" is also possible and is reserved for future use).

5. View Accounts overview

As an Account Owner I can see an overview of all my Accounts including their status (Open / Closed).

Acceptance Criteria

- The list of all accounts is provided including their
  - Id
  - Status (Open / Closed)
  - Closure Date (if applicable)

    - Closure Reason (if applicable)
    - Balance

6. View Account transactions

As an Account Owner I can see the list of all transactions for the specified Accounts.

Acceptance Criteria

- For each transaction the following information is provided:
    - Id
    - Type (Deposit or Withdrawal)
    - Amount
    - Date (the UTC Date when the transaction was executed)

Bonus Non-Functional Requirements (this step is optional)

Please describe how you would approach this problem if the system had to support 20,000 transactions / second. What would have to be changed in the architecture and why?