**Elias Tuma**

**316355148**

**Atid College**

**QA in Python**

**Banking System**

**System Design Document (SDD) for Banking System:**

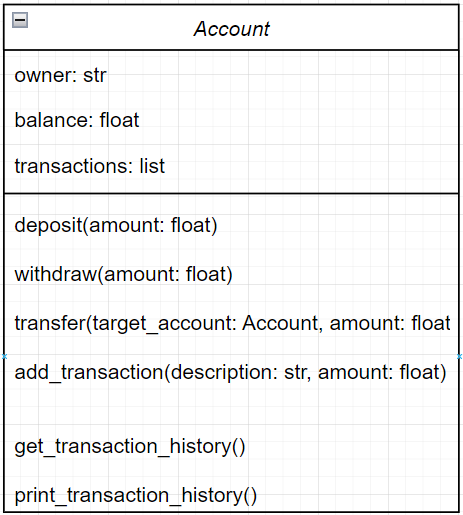
### 1. Overview

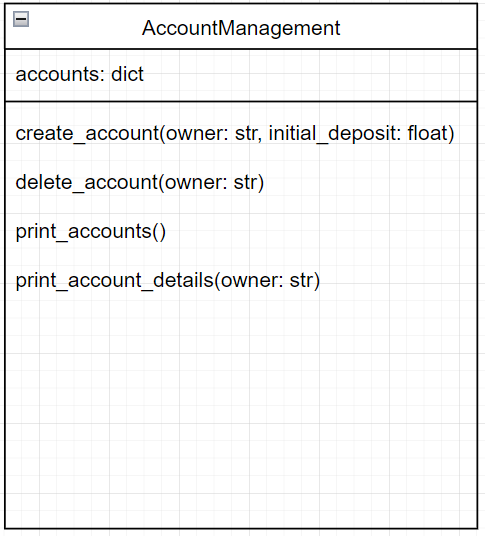
The Banking System is a console-based application designed to manage bank accounts. It supports operations such as creating and deleting accounts, depositing and withdrawing funds, transferring money between accounts, and viewing account details. The system is implemented in Python and is intended to be simple and easy to use.

### Design Diagrams

#### Class Diagram

The Class Diagram illustrates the main classes in the banking system, their attributes, methods, and relationships.





#### 2.2 Sequence Diagram

The Sequence Diagram shows how different components interact during a fund transfer operation:

User -> Banking System: Request Transfer

Banking System -> Account: withdraw(amount)

Account -> Account: Update balance

Account -> Account: Add transaction

Banking System -> Account: deposit(amount)

Account -> Account: Update balance

Account -> Account: Add transaction

Banking System -> User: Confirm Transfer

#### 2.3 Use Case Diagram

The Use Case Diagram outlines the interactions between the user and the system.

+-------------------+

| Banking |

| System |

+-------------------+

^

|

|

+--------------------+ +--------------------+

| User | | System |

+--------------------+ +--------------------+

| + Create Account | | + Create Account |

| + Deposit Funds | | + Deposit Funds |

| + Withdraw Funds | | + Withdraw Funds |

| + Transfer Funds | | + Transfer Funds |

| + View Details | | + View Details |

| + View All Accounts | | + View All Accounts|

+---------------------+ +--------------------+

## 3. System Components

### 3.1 Account Class

**Purpose:** Manages individual bank accounts, including balance, transactions, and account operations.

**Attributes:**

* owner: The name of the account holder.
* balance: The current balance of the account.
* transactions: A list of transactions associated with the account.

**Methods:**

* deposit(amount): Deposits a specified amount into the account.
* withdraw(amount): Withdraws a specified amount from the account.
* transfer(target\_account, amount): Transfers funds to another account.
* add\_transaction(description, amount): Records a transaction.
* get\_transaction\_history(): Retrieves the transaction history.
* print\_transaction\_history(): Prints the transaction history.

### 3.2 AccountManagement Class

**Purpose:** Manages multiple bank accounts and provides methods for account operations.

**Attributes:**

* accounts: A dictionary where keys are account owners and values are Account objects.

**Methods:**

* create\_account(owner, initial\_deposit): Creates a new account with an initial deposit.
* delete\_account(owner): Deletes an account.
* get\_account(owner): Retrieves an account.
* print\_accounts(): Prints all accounts.
* print\_account\_details(owner): Prints the details of a specific account.

## 4. Interaction Scenarios

### 4.1 Creating an Account

1. **User Input:** Provide account owner name and initial deposit.
2. **System Actions:** Call create\_account method in AccountManagement.
3. **System Response:** New account is created and initial transaction is added.

### 4.2 Depositing Funds

1. **User Input:** Provide amount to deposit.
2. **System Actions:** Call deposit method on the account.
3. **System Response:** Account balance is updated, and transaction is recorded.

### 4.3 Transferring Funds

1. **User Input:** Provide target account and amount.
2. **System Actions:** Call withdraw on the source account, deposit on the target account, and update transactions.
3. **System Response:** Both accounts are updated, and transactions are recorded.

## 5. Conclusion

This design document outlines the structure and functionality of the banking system. It includes class and sequence diagrams, a description of components, and interaction scenarios to provide a clear understanding of how the system operates and how different components interact.