A Report

ON

"Oop Project"

"Banking And ATM System"

Representors

Group 3

Muhammad Uneeb Khan 243729

Muhammad Hamza Khan 243774

Muhammad Anas Akram 243766

Under the Guidance of

Sir Saad-Ur-Rehman

Department of Computer Science Air University Multan

TABLE OF CONTENT:

Contents

Banking ATM System

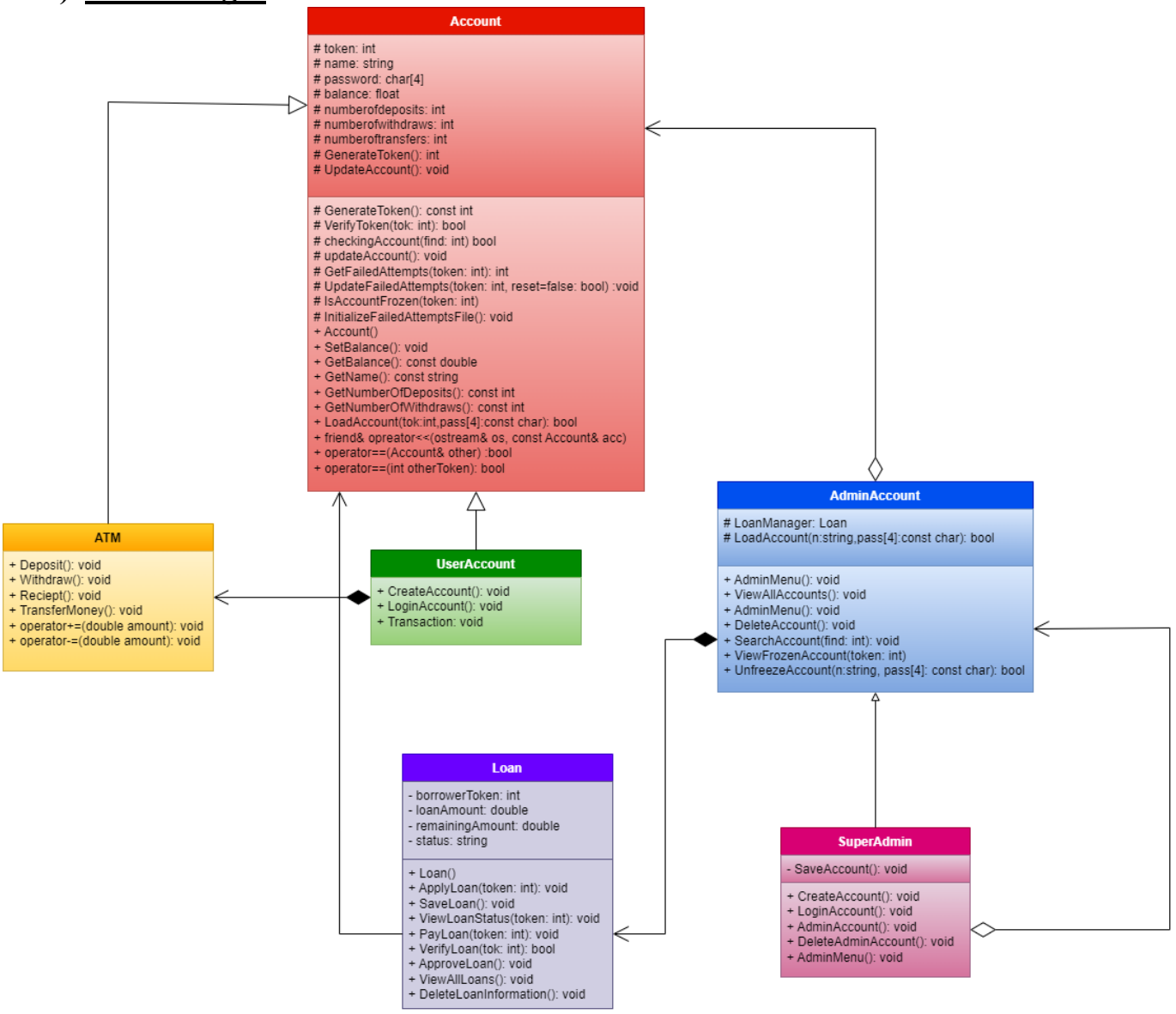
1) Problem Statement:

- Managing bank accounts manually is slow and risky.
- Users need an easy way to create accounts, deposit, and withdraw money.
- Weak passwords and no login attempt limits.
- Loan process is slow. Customers wait too long for approvals.
- Admins struggle to track accounts, freeze suspicious ones, or approve loans.
- No receipts. Users can't check transaction history easily.

Solution:

- A digital banking system to automate everything.
- Secure login with account lock after wrong tries.
- Deposit, withdraw, and transfer money easily.
- Apply for loans and track status. We make a non-interest loan system to make it Islamic bank.
- Admins can manage accounts and approve loans faster.
- Faster, safer, and more convenient banking.
- Make banking simple for users and admins.

2) OOP Design:



Classes and their Core Functions:

Account

- GenerateToken(): Creates a unique account number.
- SaveAccount(): Stores account details in Account.txt.
- LoadAccount(): Retrieves account data for login.
- updateAccount(): Updates balance/transactions in the file.
- checkingAccount(): Verifies if an account exists.
- VerifyToken(): Checks if a token is valid.
- GetFailedAttempts(): Tracks failed logins.

- UpdateFailedAttempts(): Locks accounts after 3 failed tries.
- IsAccountFrozen(): Checks if an account is locked.

ATM

- Deposit(): Adds funds to balance (operator+=).
- WithDraw(): Deducts balance (operator-=).
- TransferMoney(): Moves money between accounts.
- Receipt(): Prints transaction details (uses operator<< overload).

Loan

- ApplyLoan(): Submits a loan request.
- SaveLoan(): Stores loan data in Loan.txt.
- ViewLoanStatus(): Checks approval/pending/paid status.
- PayLoan(): Processes loan repayments.
- ApproveLoan(): (Admin-only) Approves pending loans.
- DeleteLoanInformation(): Removes paid loans.

UserAccount

- CreateAccount(): Registers a new user (calls SaveAccount()).
- LoginAccount(): Authenticates users (uses LoadAccount()).
- Transaction(): Menu for deposits/withdrawals/loans (uses ATM and Loan).

AdminAccount

- AdminLogin(): Authenticates admins.
- ViewAllAccounts(): Lists all user accounts.
- DeleteAccount(): Removes user accounts.
- SearchAccount(): Finds accounts by token.
- ViewFrozenAccounts(): Shows locked accounts.
- UnfreezeAccount(): Unlocks frozen accounts.

SuperAdmin

- CreateAccount(): Adds new admins.
- DeleteAdminAccount(): Removes admins.
- Extended AdminMenu(): Extra options (e.g., manage admins).

Relationships Between Classes:

- **Inheritance:**

- Account is the Base class and is inherited by ATM, UserAccount and AdminAccount.
- SuperAdmin Inherits the Account class.

- **Composition:**

- ATM and UserAccount has a relation (Composition).
- AdminAccount has a LoanManager (Composition).

- **Aggregation:**

- Account and AdminAccount has a relation (Aggregation).
- SuperAdminAccount and AdminAccount has a relation (Aggregation).

- **Association:**

- Loan is associated with Account through the borrowerToken.
- ATM interacts with Account for transactions.

3) Operator Overloading:

- Overload ostream operator<< for the receipt to show the details of the account.
- Overload = = operator for the comparison of two accounts in TransferAccount that the user cannot transfer money to its account.
- Overload + = operator to deposit money.
- Overload - = operator to withdraw money.

4) Overloading Functions

<< Insertion Operator Overload:

To Display Account Display.

```
friend ostream& operator<<(ostream& os, const Account& acc)
{
    os<<"      Token Number: "<<acc.GetToken()<<endl
    <<"      Name: "<<acc.GetName()<<endl
    <<"      Current Balance: "<<acc.GetBalance()<<endl
    <<"      Number of Deposits: "<<acc.GetNumberOfDeposits()<<endl
    <<"      Number of Withdraws: "<<acc.GetNumberOfWithdraws()<<endl
    <<"      Number of Transfers: "<<acc.GetNumberofTransfers()<<endl<<endl;
    return os;
}
```

+= and -= Operator Overload:

```
void operator+=(double amount)
{
    balance+=amount;
}
void operator-=(double amount)
{
    balance-=amount;
}
```

= = Operator Overload:

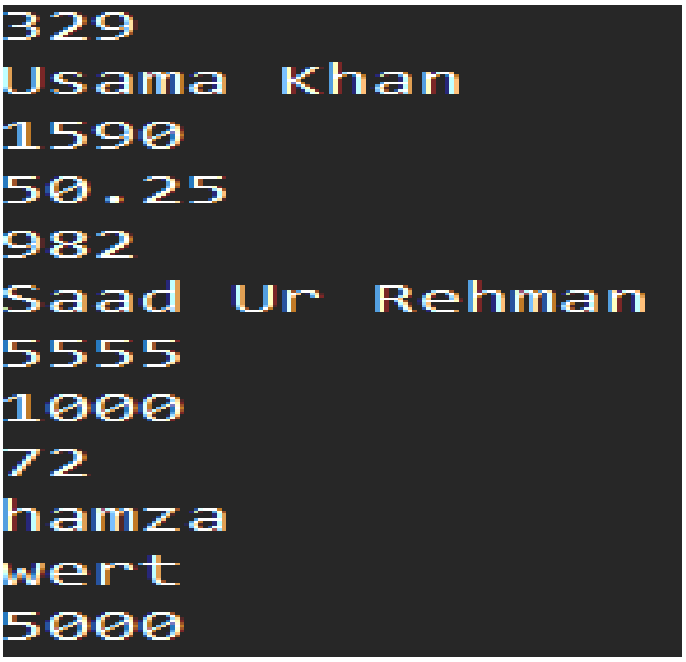
```
bool operator==(const Account& other) const
{
    return (this->token==other.token);  // Co
}
```

5) File Handling Functions:

Account Management:

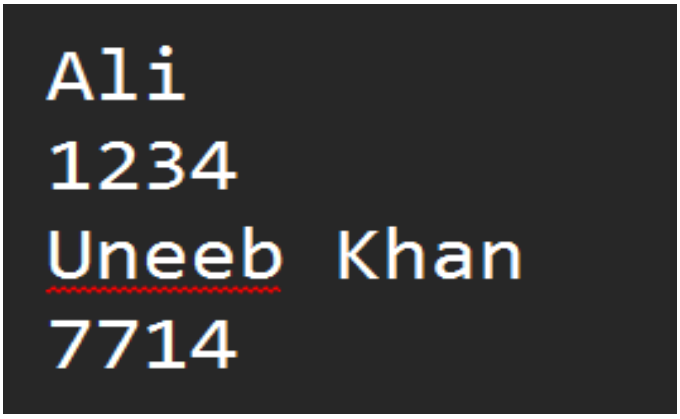
Structure of writing data(Account.txt):

- Token
- Name
- Password
- Balance



Structure of Writing Data (Admin.txt):

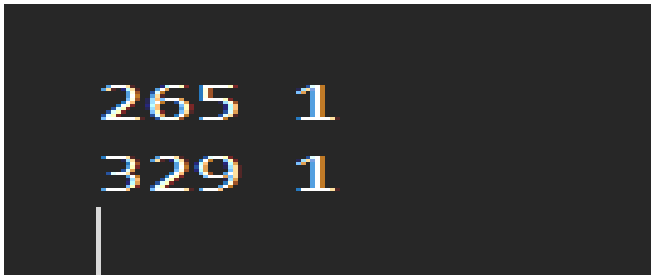
- Name
- Password



Security Feature:

Structure Writing Data (FailedAttempts.txt):

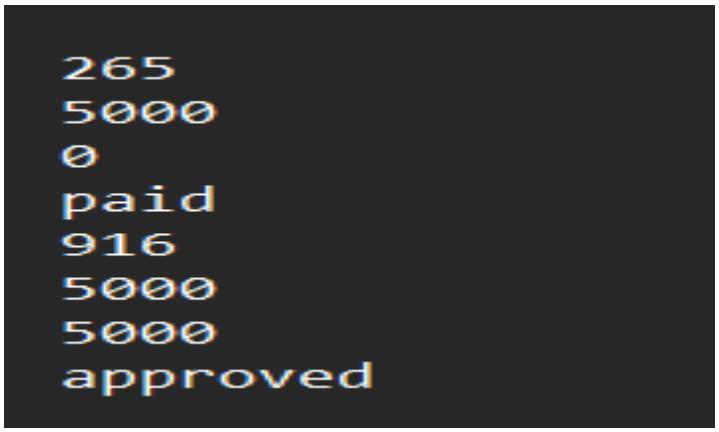
- Token Number Number of Failed Attempts



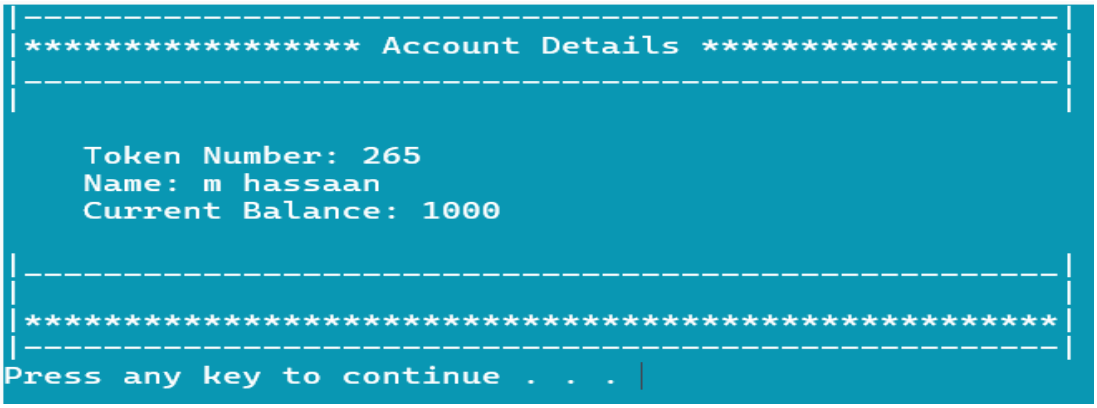
Loan Management:

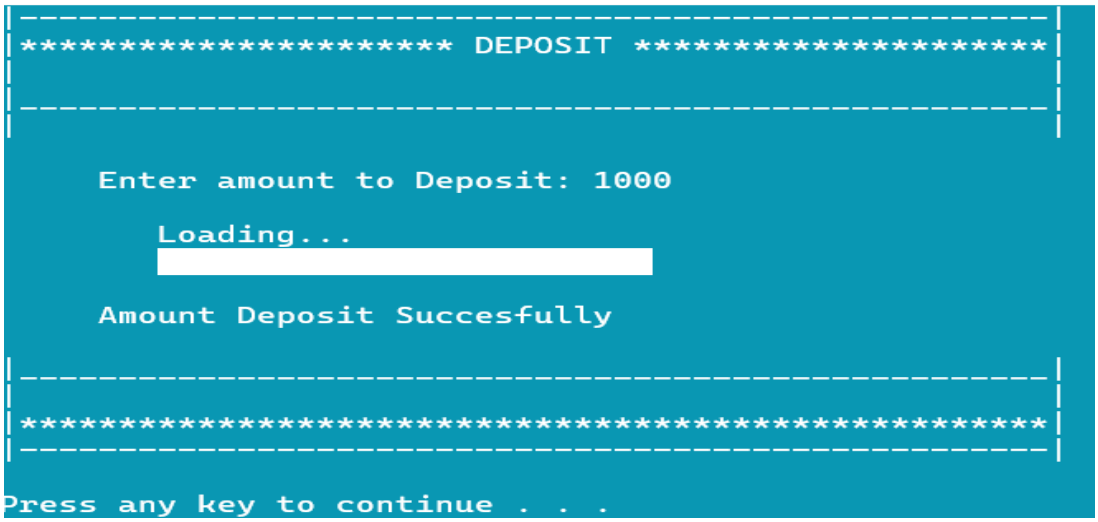
Structure of writing Data (Loan.txt):

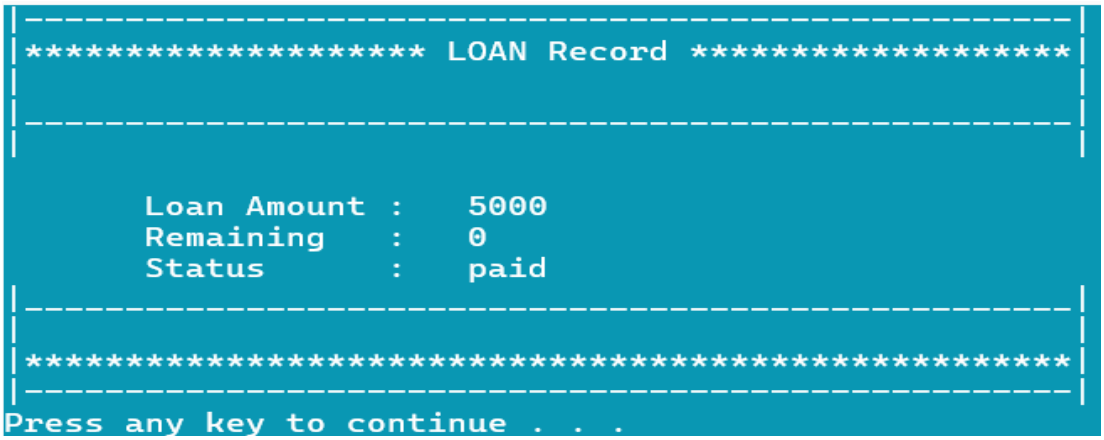
- Borrower Token
- Amount Borrowed
- Remaining Amount
- Status

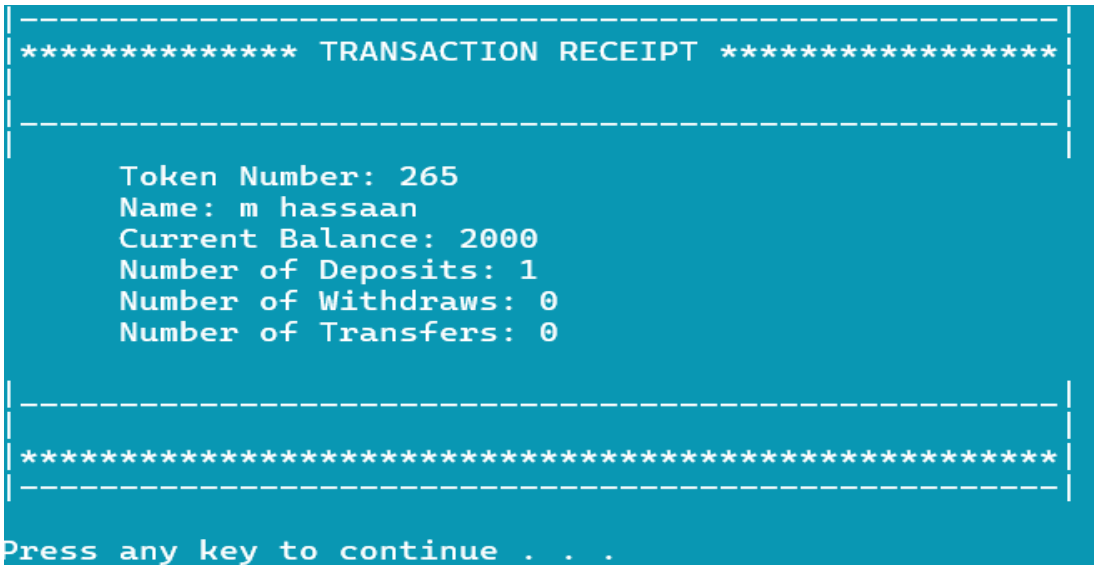


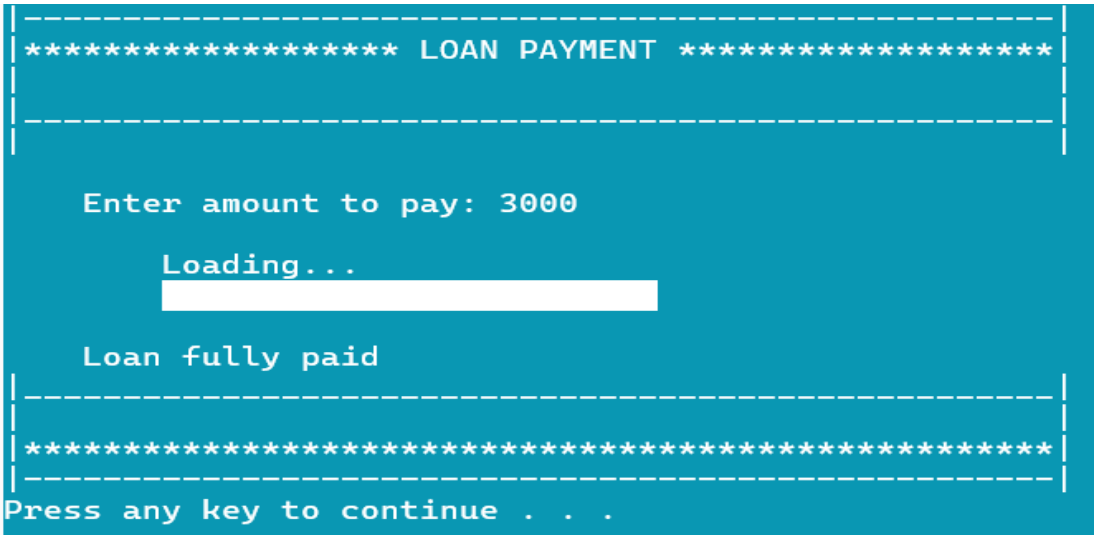
6) Sample Outputs:






********** LOAN Record **********
Loan Amount : 5000
Remaining : 0
Status : paid
****
Press any key to continue . . .


********** TRANSACTION RECEIPT **********
Token Number: 265
Name: m hassaan
Current Balance: 2000
Number of Deposits: 1
Number of Withdraws: 0
Number of Transfers: 0
****
Press any key to continue . . .


********** LOAN PAYMENT **********
Enter amount to pay: 3000
Loading...
Loan fully paid
****
Press any key to continue . . .

```
|-----------------------------------------------------|
|******************** LOAN PAYMENT ********************|
|                                                     |

1- Token: 916
   Total Amount: 5000
   Remaining Amount: 5000
   Status: approved

2- Token: 265
   Total Amount: 3000
   Remaining Amount: 0
   Status: paid

|-----------------------------------------------------
|
|*****************************************************
|-----------------------------------------------------
Press any key to continue . . .
```

```
|-----------------------------------------------------|
|********************** WITHDRAW **********************|
|                                                     |
|-----------------------------------------------------|
|                                                     |


    Enter amount to Withdraw: -784
Invalid Input...Please try again
    Enter amount to Withdraw:
```

```
|-----------------------------------------------------|
|******************* Transfer Balance ******************|
|                                                     |
|-----------------------------------------------------|
|                                                     |


    Enter receiver's token number: 265
    Enter amount to transfer: 1000

       Loading...


    Amount transferred successfully.

|-----------------------------------------------------|
|                                                     |
|*****************************************************|
|-----------------------------------------------------|

Press any key to continue . . .
```