SIX WEEKS SUMMER TRAINING REPORT

on

ATM MANAGEMENT SYSTEM

Submitted by

Nitesh Kumar Srivastava

Registration No: 12110770

Program Name: B.Tech.(CSE)

Under the Guidance of

Mr . Adhiraj Chauhan

(CipherSchool)

School of Computer Science and Engineering
Lovely Professional University. Phagwara
(June- July, 2023)

DECLARATION

I hereby declare that I have completed my six weeks summer training at **CipherSchool** from **10th June 2023** to **29th July 2023** under the guidance of **Mr. Adhiraj Chauhan** . I declare that I have worked with full dedication during these six weeks of training and my learning outcomes fulfill the requirements of training for the award of degree of **B.Tech.(Computer Science & Engineering)** at Lovely Professional University, Phagwara.

Nitesh Kumar Srivastava
Registration No: 12110770

CODE

```
#include <iostream>
#include <unordered_map>
using namespace std;

struct Transaction {
   string type;
   int amount;
   Transaction* next;
};

unordered_map<int, int> accounts;
unordered_map<int, int> accountsBalance;
unordered_map<int, Transaction*> transactionHistory;
string languages[] = {"English", "Hindi"}; // Array for multi-language support

void createAccount() {
   cout << "Enter a customer ID: ";
   int customerId;
   cin >> customerId;

   cout << "Enter a PIN: ";
   int pin;
   cin >> pin;

  accounts[customerId] = pin;
   cout << "Account created successfully." << endl;
   }

  int login() {
```

```
  cout << "Enter your customer ID: ";
  int customerId;
  cin >> customerId;

  cout << "Enter your PIN: ";
  int pin;
  cin >> pin;
if (accounts.count(customerId) && accounts[customerId] == pin) {
    cout << "Login successful." << endl;
    return customerId;
 } else {
    cout << "Invalid customer ID or PIN." << endl;
    return 0;
  }
  }

void addToTransactionHistory(int customerId, const string& type, int amount) {
  Transaction* transaction = new Transaction();
  transaction->type = type;
  transaction->amount = amount;
  transaction->next = nullptr;

  if (!transactionHistory.count(customerId)) {
    transactionHistory[customerId] = transaction;
  } else {
   Transaction* curr = transactionHistory[customerId];
    while (curr->next) {
    curr = curr->next;
  }
    curr->next = transaction;
    }
   }
 void displayTransactionHistory(int customerId) {
 cout << "Transaction History:" << endl;
 if (transactionHistory.count(customerId)) {
 Transaction* curr = transactionHistory[customerId];
  while (curr) {
        cout << curr->type << ": " << curr->amount << endl;
        curr = curr->next;
   }
   } else {
    cout << "No transaction history found." << endl;
   }
   }

void transaction(int customerId) {
 while (true) {
    cout << "Choose a transaction:" << endl;
    cout << "1. Withdraw" << endl;
    cout << "2. Deposit" << endl;
    cout << "3. Check Balance" << endl;
    cout << "4. Fast Withdraw" << endl;
    cout << "5. Transfer Funds" << endl;
    cout << "6. Transaction History" << endl;
    cout << "7. Change Language" << endl;
    cout << "8. Logout" << endl;

    int choice;
    cin >> choice;
```

```
      if (choice == 1) {
          cout << "Enter amount to withdraw: ";
       int amount;
          cin >> amount;


        if (amount > 0) {
              int balance = 0;
        if (accountsBalance.count(customerId)) {
                  balance = accountsBalance[customerId];
              }


          if (balance >= amount) {
           accountsBalance[customerId] = balance - amount;
          cout << "Transaction successful. Current balance:"<<
accountsBalance[customerId] << endl;


                  addToTransactionHistory(customerId, "Withdraw", amount);
            } else {
                  cout << "Insufficient funds." << endl;
            }
            } else {
              cout << "Invalid amount." << endl;
            }
        } else if (choice == 2) {
            cout << "Enter amount to deposit: ";
            int amount;
            cin >> amount;


            if (amount > 0) {
              int balance = 0;
              if (accountsBalance.count(customerId)) {
                balance = accountsBalance[customerId];
            }

        accountsBalance[customerId] = balance + amount;
            cout << "Transaction successful. Current balance: " <<
accountsBalance[customerId] << endl;

            addToTransactionHistory(customerId, "Deposit", amount);
        } else {
            cout << "Invalid amount." << endl;
        }
    } else if (choice == 3) {
     int balance = 0;
        if (accountsBalance.count(customerId)) {
            balance = accountsBalance[customerId];
        }

        cout << "Your balance is: " << balance << endl;
    } else if (choice == 4) {
        cout << "Enter amount to withdraw: ";
        int amount;
        cin >> amount;

        if (amount > 0) {
            int balance = 0;
            if (accountsBalance.count(customerId)) {
            balance = accountsBalance[customerId];
            }

            if (balance >= amount + 10) {
             accountsBalance[customerId] = balance - amount - 10;
```

```
                    cout << "Transaction successful. Current balance: " <<
accountsBalance[customerId] << endl;

            addToTransactionHistory(customerId, "Fast Withdraw", amount);
         } else {
            cout << "Insufficient funds." << endl;
          }
          } else {
         cout << "Invalid amount." << endl;
          }
          } else if (choice == 5) {
          cout << "Enter recipient's customer ID: ";
          int recipientId;
          cin >> recipientId;
          if (accounts.count(recipientId)) {
          if (!accountsBalance.count(recipientId)) {
            accountsBalance[recipientId] = 0;
         }

         cout << "Enter amount to transfer: ";
         int amount;
         cin >> amount;

         if (amount > 0) {
            int balance = 0;
         if (accountsBalance.count(customerId)) {
               balance = accountsBalance[customerId];
            }

          if (balance >= amount) {
          accountsBalance[customerId] = balance - amount;
```

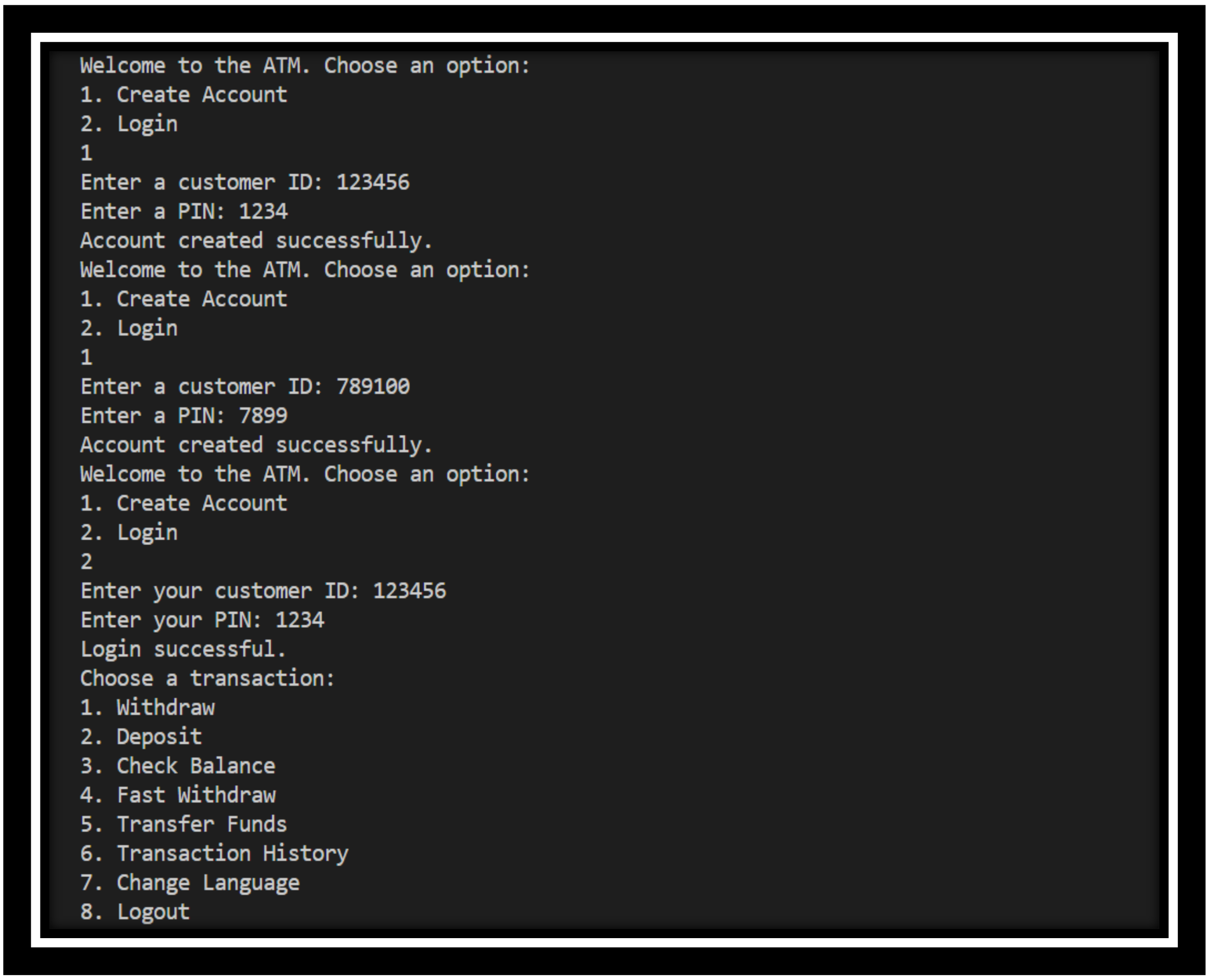
```
                accountsBalance[recipientId] += amount;
                        cout << "Transaction successful. Current balance: " <<
accountsBalance[customerId] << endl;


                        addToTransactionHistory(customerId, "Transfer to " +
to_string(recipientId), amount);
    } else {
            cout << "Insufficient funds." << endl;
    }
    } else {
          cout << "Invalid amount." << endl;
    }
    } else {
      cout << "Recipient's customer ID not found." << endl;
    }
    } else if (choice == 6) {
    displayTransactionHistory(customerId);
    } else if (choice == 7) {
    cout << "Select Language:" << endl;
    for (int i = 0; i < sizeof(languages) / sizeof(languages[0]); i++) {
      cout << i + 1 << ". " << languages[i] << endl;
    }


    int langChoice;
    cin >> langChoice;


            if (langChoice >= 1 && langChoice <= sizeof(languages) /
sizeof(languages[0])) {
   cout << "Language changed to: " << languages[langChoice - 1] << endl;
      // Perform language-specific operations here
   } else {
      cout << "Invalid language choice." << endl;
```

```
 }
 } else if (choice == 8) {
      cout << "Logout successful." << endl;
      break;
} else {
      cout << "Invalid choice. Please try again." << endl;
}
}
}

int main() {
while (true) {
  cout << "Welcome to the ATM. Choose an option:" << endl;
  cout << "1. Create Account" << endl;
  cout << "2. Login" << endl;

  int choice;
  cin >> choice;

  if (choice == 1) {
      createAccount();
  } else if (choice == 2) {
      int customerId = login();
      if (customerId != 0) {
          transaction(customerId);
      }
  } else {
      cout << "Invalid choice. Please try again." << endl;
  }
  }     return 0;  }
```

x

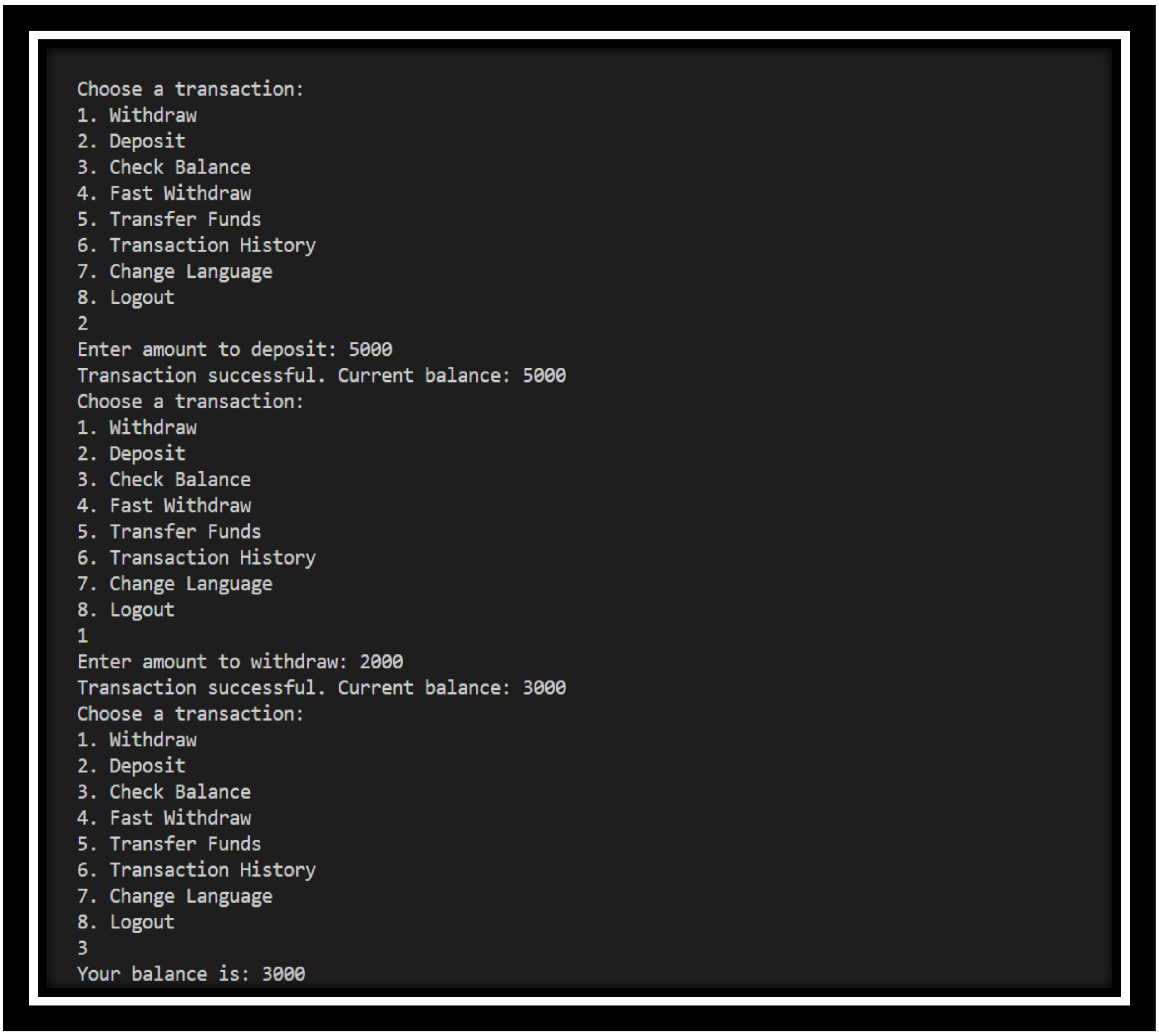
SCREENSHOT

- **Created two account second one for recipient's id which will use later on transfer fund (option 5).**



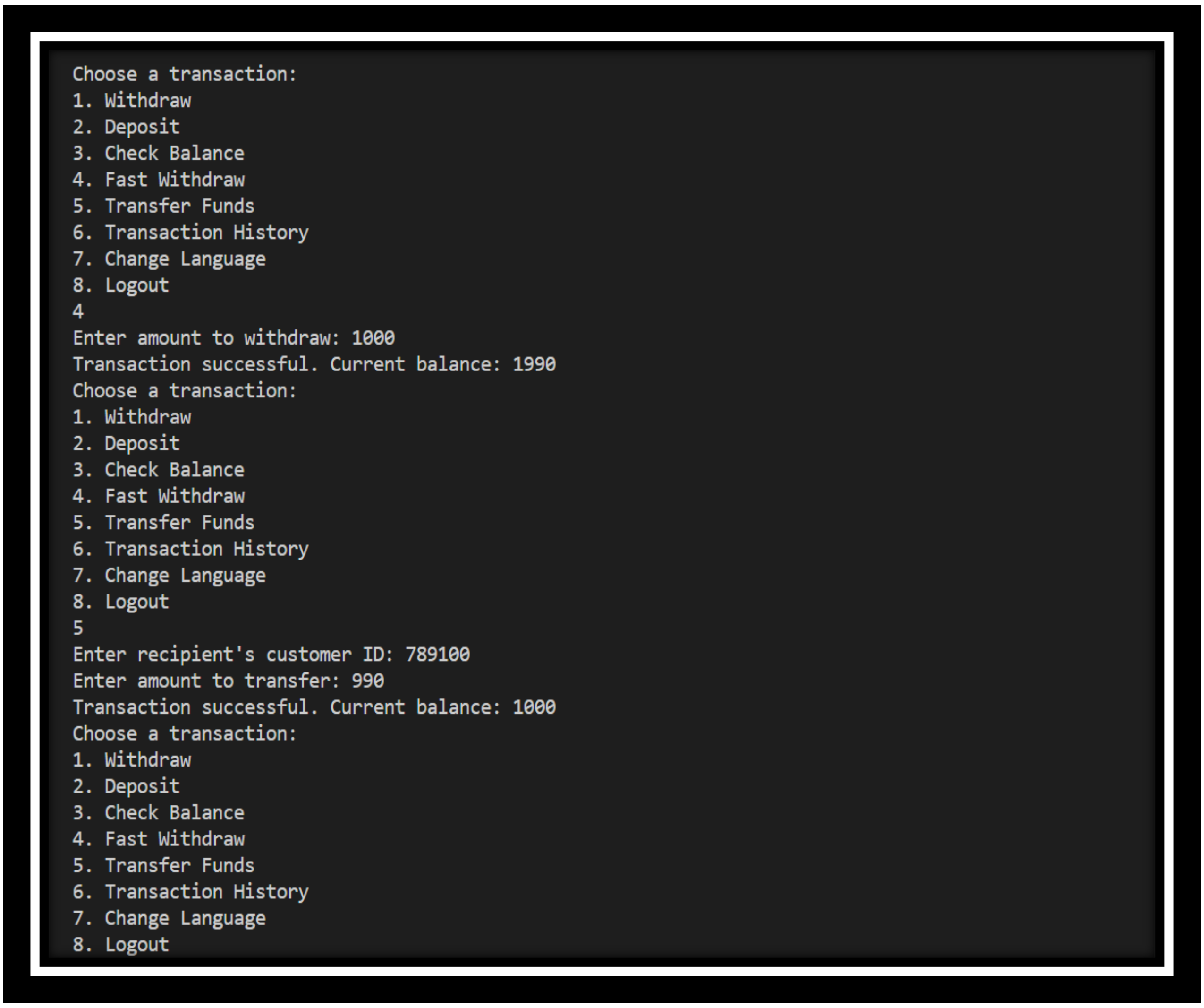
```
Welcome to the ATM. Choose an option:
1. Create Account
2. Login
1
Enter a customer ID: 123456
Enter a PIN: 1234
Account created successfully.
Welcome to the ATM. Choose an option:
1. Create Account
2. Login
1
Enter a customer ID: 789100
Enter a PIN: 7899
Account created successfully.
Welcome to the ATM. Choose an option:
1. Create Account
2. Login
2
Enter your customer ID: 123456
Enter your PIN: 1234
Login successful.
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
```

- **You can deposit the money and withdraw the amount by clicking the needed option.**



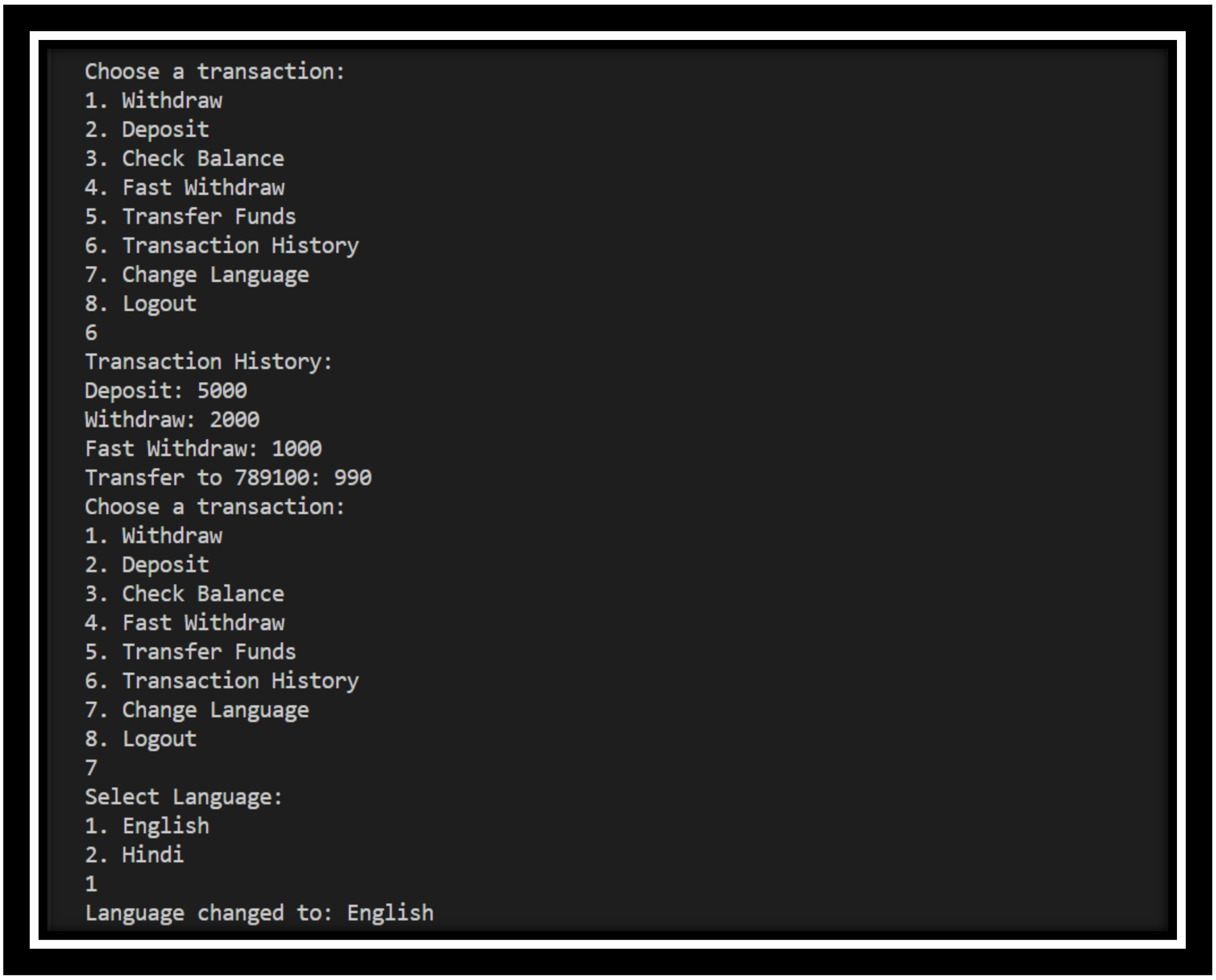
```
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
2
Enter amount to deposit: 5000
Transaction successful. Current balance: 5000
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
1
Enter amount to withdraw: 2000
Transaction successful. Current balance: 3000
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
3
Your balance is: 3000
```

- **Fast withdraw will cut the amount by 10% from original amount used for immediate need of money for the user.**



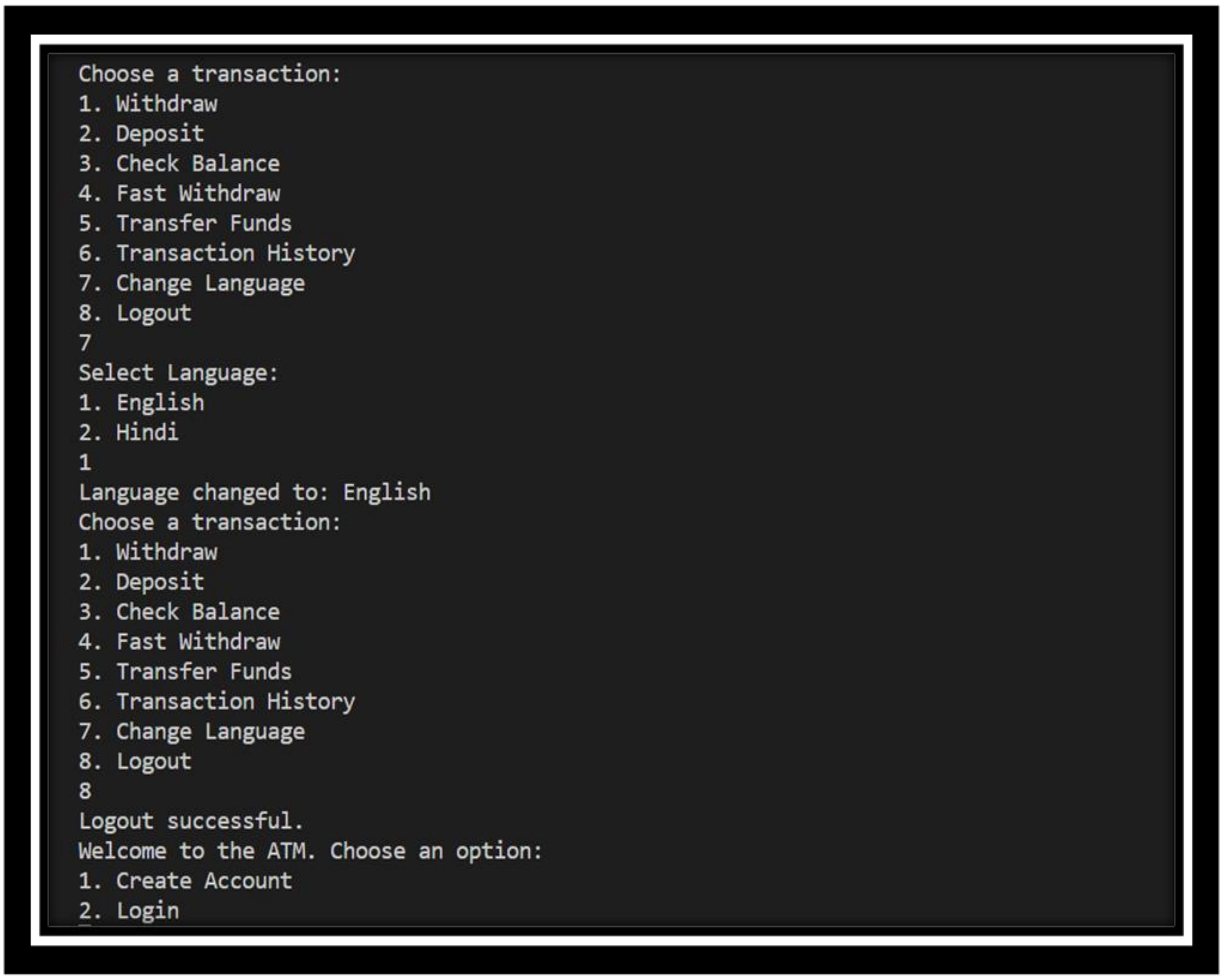
```
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
4
Enter amount to withdraw: 1000
Transaction successful. Current balance: 1990
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
5
Enter recipient's customer ID: 789100
Enter amount to transfer: 990
Transaction successful. Current balance: 1000
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
```

- **we can also see our transaction history whatever we have done. And I used Linked list data structure for it.**



```
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
6
Transaction History:
Deposit: 5000
Withdraw: 2000
Fast Withdraw: 1000
Transfer to 789100: 990
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
7
Select Language:
1. English
2. Hindi
1
Language changed to: English
```

- **we can also change the language to hindi. And I used Array data Structure for it.**



```
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
7
Select Language:
1. English
2. Hindi
1
Language changed to: English
Choose a transaction:
1. Withdraw
2. Deposit
3. Check Balance
4. Fast Withdraw
5. Transfer Funds
6. Transaction History
7. Change Language
8. Logout
8
Logout successful.
Welcome to the ATM. Choose an option:
1. Create Account
2. Login
```