```
class Account:
    def __init__(self, user_id, pin, balance=0):
        self.user_id = user_id
        self.pin = pin
        self.balance = balance
        self.transactions = []

    def check_pin(self, pin):
        return self.pin == pin

    def deposit(self, amount):
        self.balance += amount
        self.transactions.append(('Deposit', amount))
        return self.balance

    def withdraw(self, amount):
        if amount > self.balance:
            return 'Insufficient balance'
        else:
            self.balance -= amount
            self.transactions.append(('Withdraw', amount))
            return self.balance

    def get_transaction_history(self):
        return self.transactions if self.transactions else 'No transactions available'

    def transfer(self, target_account, amount):
        if amount > self.balance:
            return 'Insufficient balance'
        else:
            self.withdraw(amount)
            target_account.deposit(amount)
            self.transactions.append(('Transfer', amount))
            return self.balance


class ATM:
    def __init__(self):
        self.accounts = {}

    def create_account(self, user_id, pin, balance=0):
        self.accounts[user_id] = Account(user_id, pin, balance)

    def access_account(self):
        user_id = input("Enter your user ID: ")
        pin = input("Enter your pin: ")

        if user_id in self.accounts and self.accounts[user_id].check_pin(pin):
```

```
            return self.accounts[user_id]
        else:
            return None

    def run(self):
        while True:
            print("\nATM Interface")
            print("1. Access Account")
            print("2. Quit")
            choice = input("Choose an option: ")
            if choice == '1':
                account = self.access_account()
                if account:
                    while True:
                        print("\n1. Transaction History")
                        print("2. Withdraw")
                        print("3. Deposit")
                        print("4. Transfer")
                        print("5. Quit")
                        operation = input("Choose an operation: ")
                        if operation == '1':
                            print(account.get_transaction_history())
                        elif operation == '2':
                            amount = float(input("Enter amount to withdraw: "))
                            print(account.withdraw(amount))
                        elif operation == '3':
                            amount = float(input("Enter amount to deposit: "))
                            print(account.deposit(amount))
                        elif operation == '4':
                            target_id = input("Enter target user ID: ")
                            amount = float(input("Enter amount to transfer: "))
                            if target_id in self.accounts:
                                print(account.transfer(self.accounts[target_id], amount))
                            else:
                                print("Target account not found.")
                        elif operation == '5':
                            break
                        else:
                            print("Invalid operation.")
                else:
                    print("Invalid user ID or pin.")
            elif choice == '2':
                break
            else:
                print("Invalid choice. Please try again.")

# Create an ATM instance
atm = ATM()
```

```
# Create some accounts for testing (normally this would be handled externally)
atm.create_account('user1', '1234', 1000)
atm.create_account('user2', '5678', 500)

# Run the ATM interface
atm.run()

# The atm.run() line is commented out to prevent the code from actually running in this environment, as it requires user input.
# To use the code, uncomment the line and run it in a local Python environment where you can interact with it via the console.
```