Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Fall, Year: 2022), B.Sc. in CSE (Day)

ATM Transaction Management System

Course Title: Operating System Lab
Course Code: CSE 310
Section: 202 D2

Students Details

Name	ID
Md. Rajin Saleh	202002069

Submission Date: 10-01-2023
Course Teacher's Name: Md. Jahidul Islam

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

Chapter 1

Introduction

1.1 Introduction

Atm transaction is a process done in banks in order to make bank transactions easier. Users can check their balance, withdraw money if the given amount is in their account, deposit money into the bank, check transaction history, apply for loans, send money to other accounts, get invoice for each transaction etc. In this project, a similar atm transaction system will be recreated and simulated in a console-based environment using shell script.

1.2 Design Goals/Objectives

The objective for this project is:

- To be able to properly use shell scripting on a real-world application.
- To be able apply Operating system concepts to real-world applications.
- To be able to recreate an actual real-world application.

Chapter 2

Design/Development/Implementation of the Project

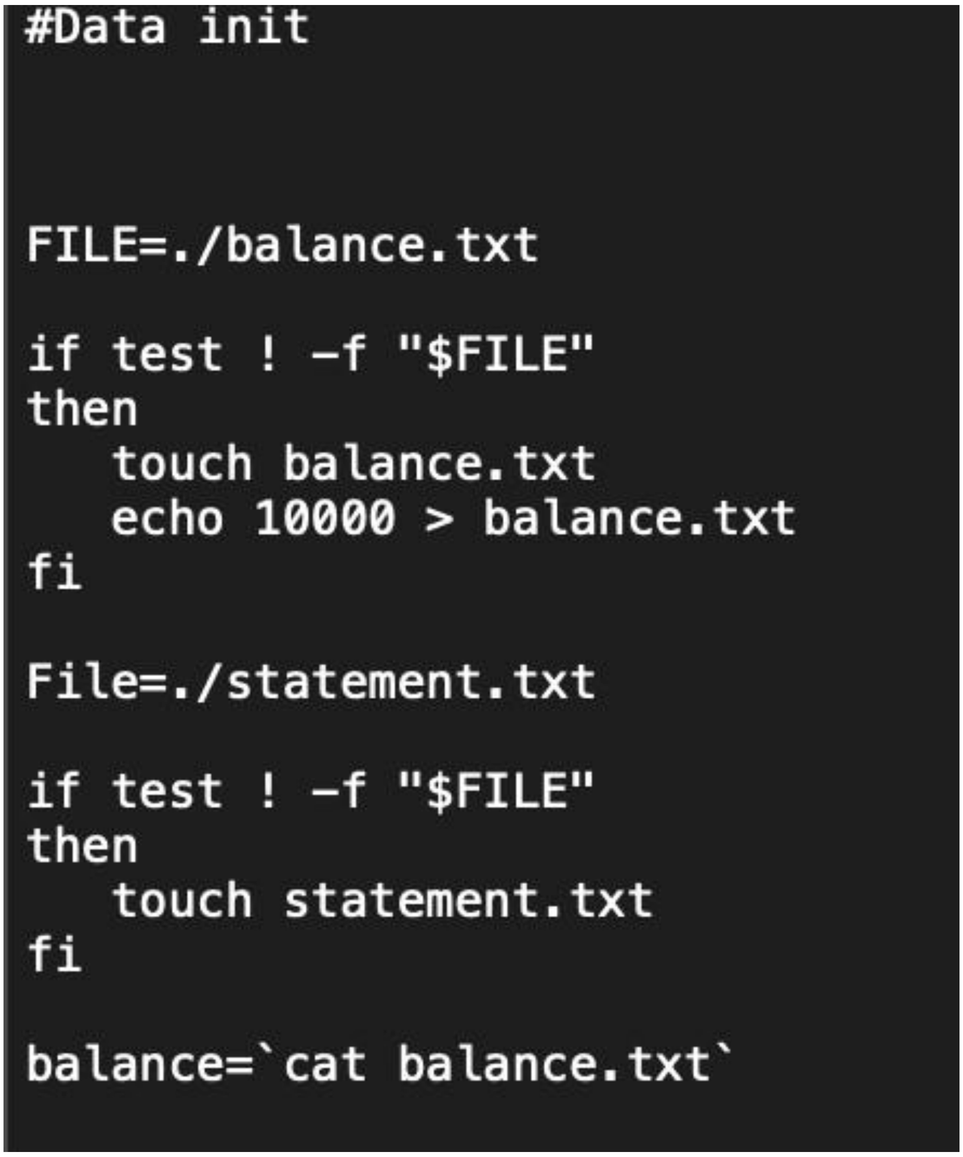
2.1 Algorithm

The Algorithm for the project is given below -

1. Check if balance.txt exists. If not, then create the file using touch command and put initial balance as 10000 unit.
2. Read the balance from balance file.
3. Check if statement.txt exists. If not, then create the file.
4. Take user input.
    - If input is 1, then take input for amount, name and date and add the amount to balance. Then, save the balance and statement to each files and show the statement using gedit command.
    - If input is 2, then take input for amount, name and date and check if the amount exists. If so, then subtract it from balance and save it to balance and statement files and show statement. Otherwise, show insufficient balance and exit using exit 0 command.
    - If input is 3, show the balance.txt file using gedit command.
    - If input is 4, show the balance.txt file using gedit command.

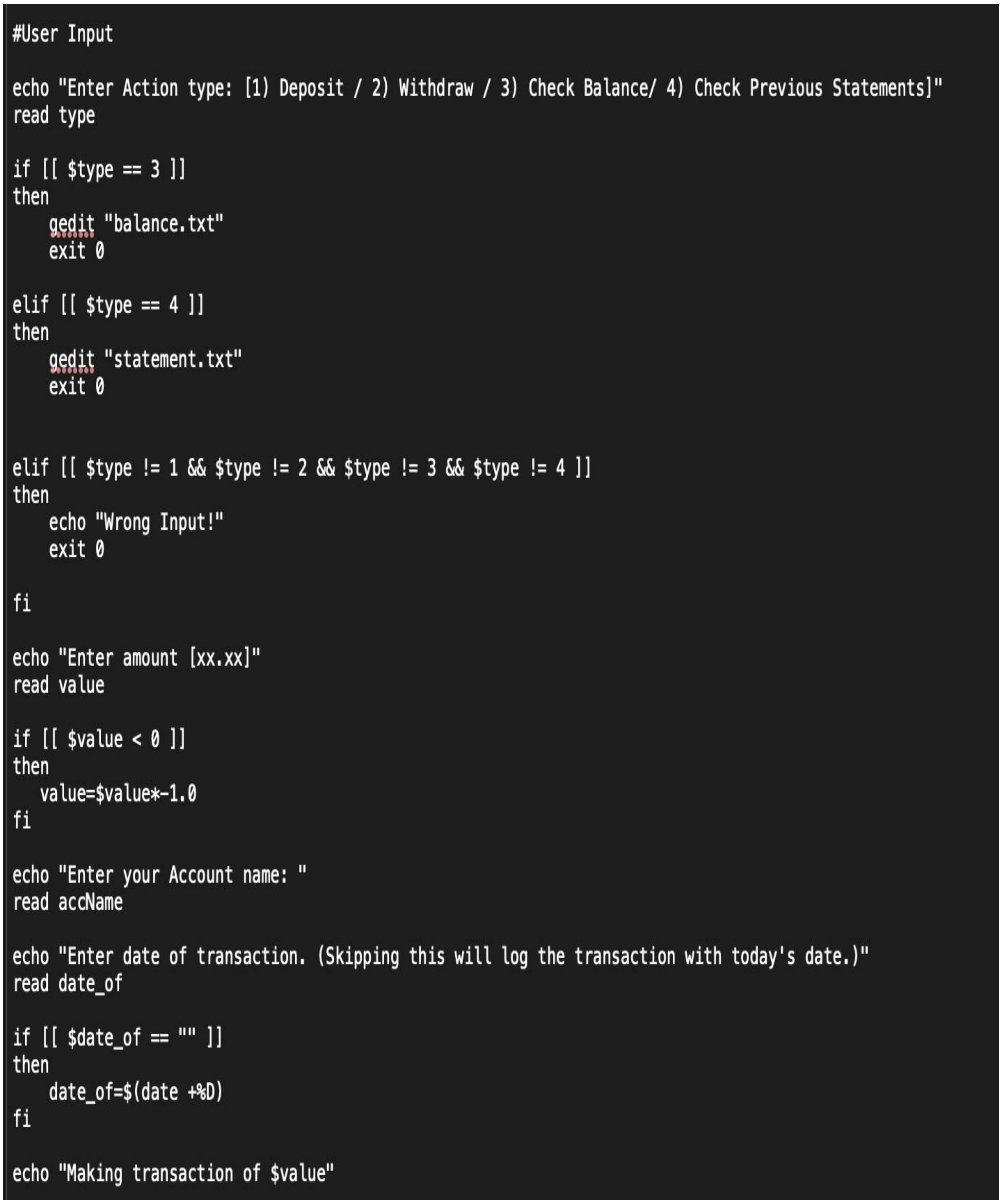
2.2 Implementation

Code snapshot :



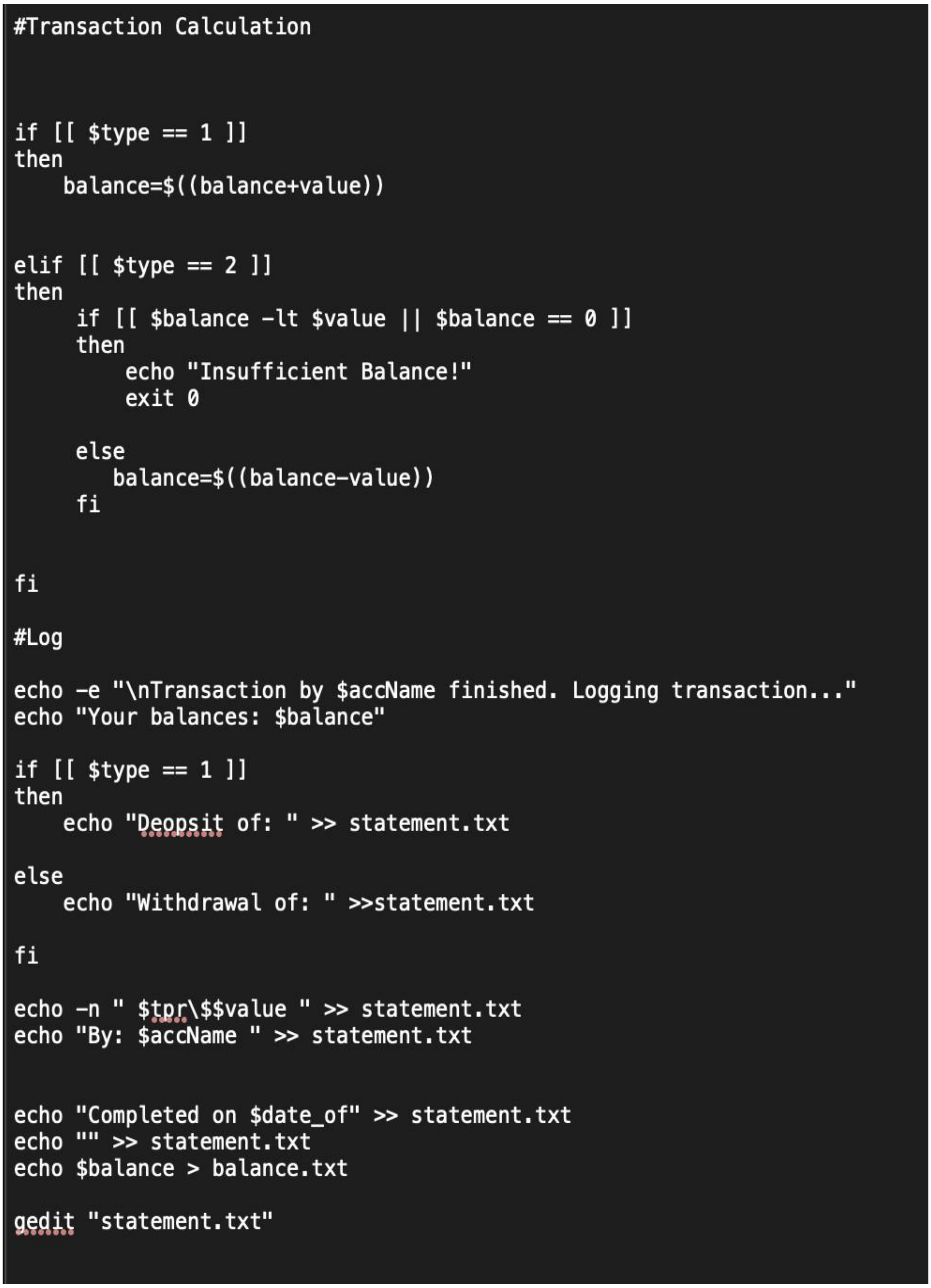
```
#Data init



FILE=./balance.txt

if test ! -f "$FILE"
then
   touch balance.txt
   echo 10000 > balance.txt
fi

File=./statement.txt

if test ! -f "$FILE"
then
   touch statement.txt
fi

balance=`cat balance.txt`
```

Figure 2.1: Data initialization



```
#User Input

echo "Enter Action type: [1) Deposit / 2) Withdraw / 3) Check Balance/ 4) Check Previous Statements]"
read type

if [[ $type == 3 ]]
then
    gedit "balance.txt"
    exit 0

elif [[ $type == 4 ]]
then
    gedit "statement.txt"
    exit 0


elif [[ $type != 1 && $type != 2 && $type != 3 && $type != 4 ]]
then
    echo "Wrong Input!"
    exit 0

fi

echo "Enter amount [xx.xx]"
read value

if [[ $value < 0 ]]
then
   value=$value*-1.0
fi

echo "Enter your Account name: "
read accName

echo "Enter date of transaction. (Skipping this will log the transaction with today's date.)"
read date_of

if [[ $date_of == "" ]]
then
    date_of=$(date +%D)
fi

echo "Making transaction of $value"
```

Figure 2.2: User Input



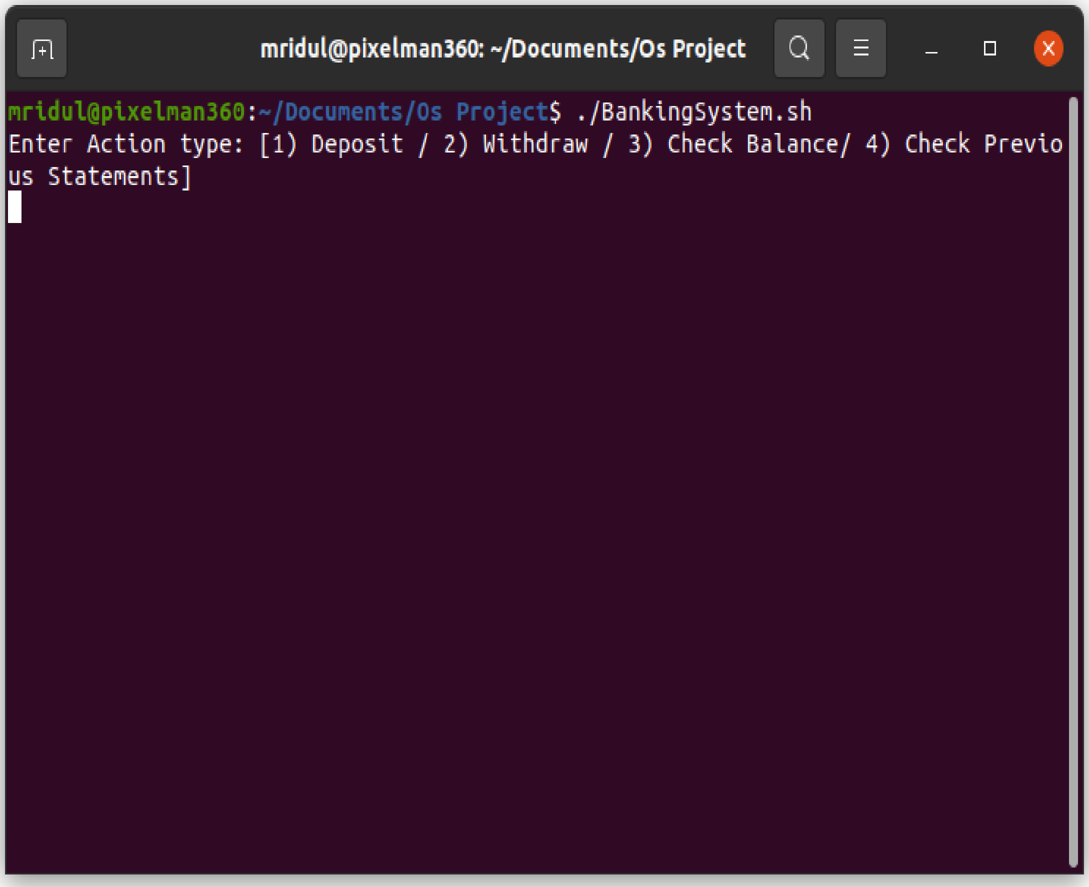
```
#Transaction Calculation



if [[ $type == 1 ]]
then
    balance=$((balance+value))


elif [[ $type == 2 ]]
then
     if [[ $balance -lt $value || $balance == 0 ]]
     then
         echo "Insufficient Balance!"
         exit 0

     else
        balance=$((balance-value))
     fi


fi

#Log

echo -e "\nTransaction by $accName finished. Logging transaction..."
echo "Your balances: $balance"

if [[ $type == 1 ]]
then
    echo "Deopsit of: " >> statement.txt

else
    echo "Withdrawal of: " >>statement.txt

fi

echo -n " $tpr\$$value " >> statement.txt
echo "By: $accName " >> statement.txt


echo "Completed on $date_of" >> statement.txt
echo "" >> statement.txt
echo $balance > balance.txt

gedit "statement.txt"
```

Figure 2.3: Calculations and other commands

Chapter 3

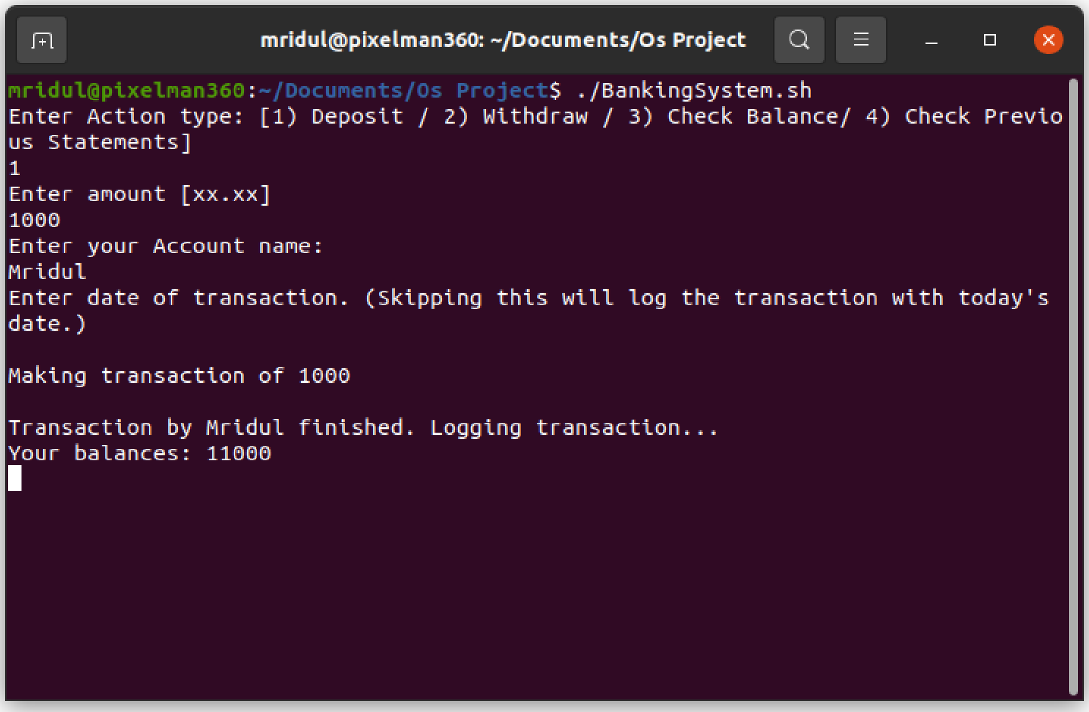
Performance Evaluation

3.1 Results Analysis/Testing



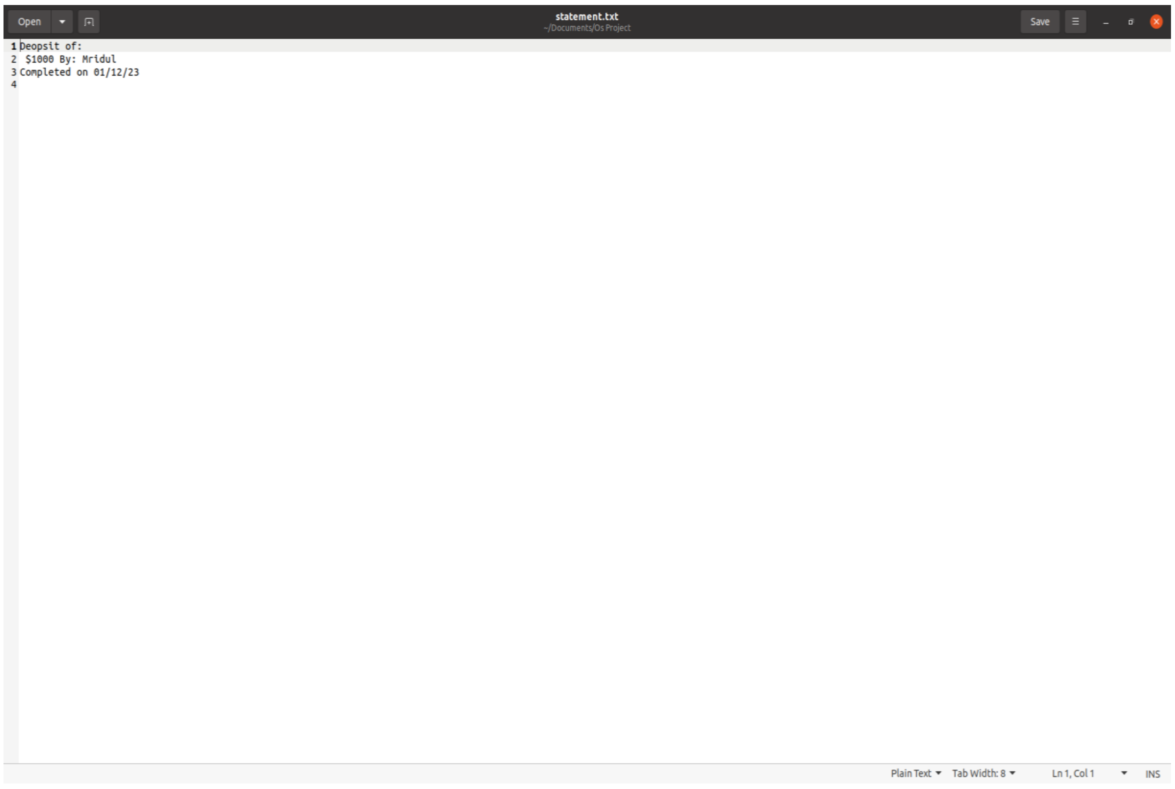
```
mridul@pixelman360:~/Documents/Os Project$ ./BankingSystem.sh
Enter Action type: [1) Deposit / 2) Withdraw / 3) Check Balance/ 4) Check Previo
us Statements]
```

Figure 3.1: Menu



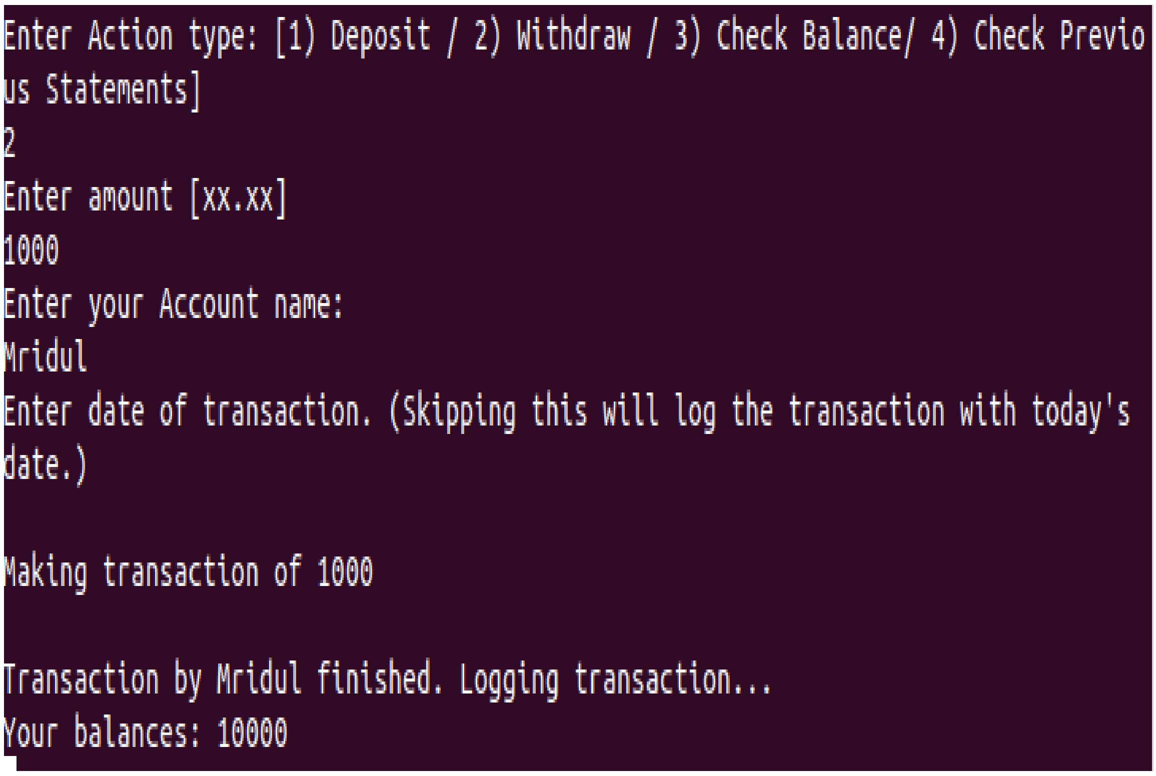
```
mridul@pixelman360:~/Documents/Os Project$ ./BankingSystem.sh
Enter Action type: [1) Deposit / 2) Withdraw / 3) Check Balance/ 4) Check Previo
us Statements]
1
Enter amount [xx.xx]
1000
Enter your Account name:
Mridul
Enter date of transaction. (Skipping this will log the transaction with today's
date.)

Making transaction of 1000

Transaction by Mridul finished. Logging transaction...
Your balances: 11000
```

Figure 3.2: Deposit money



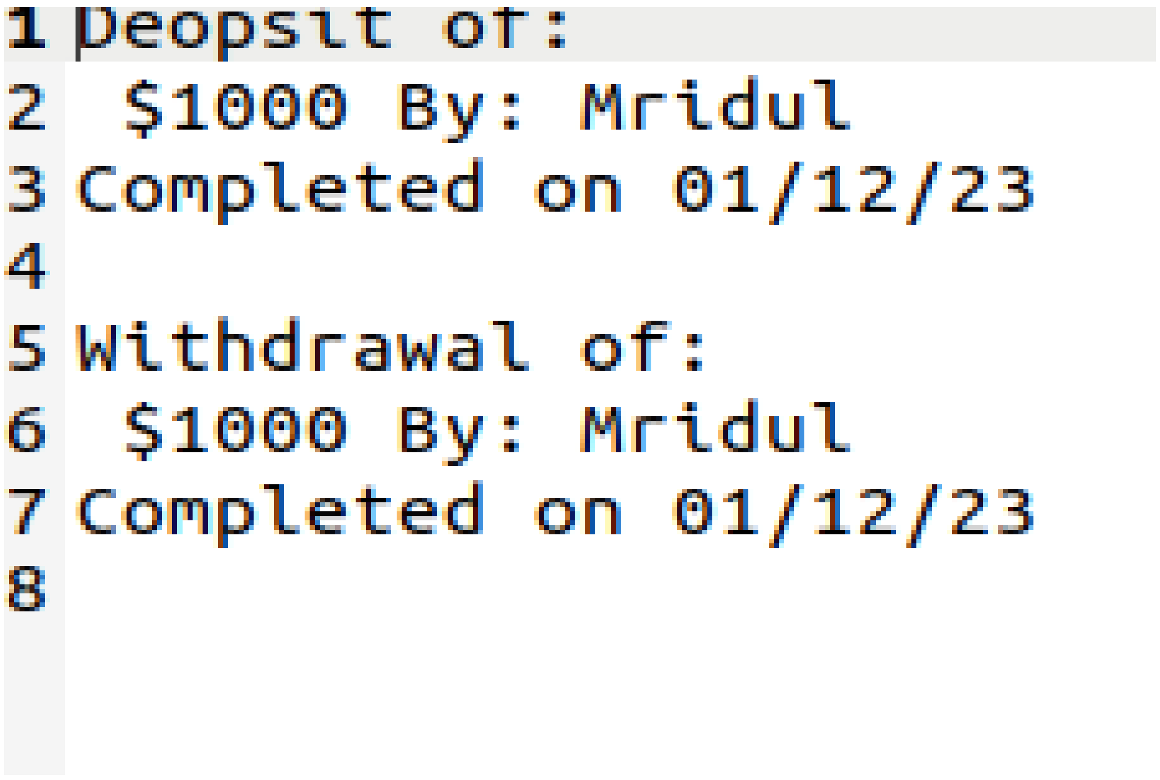
```
Deopsit of:
 $1000 By: Mridul
Completed on 01/12/23
```

Figure 3.3: Deposit Statement



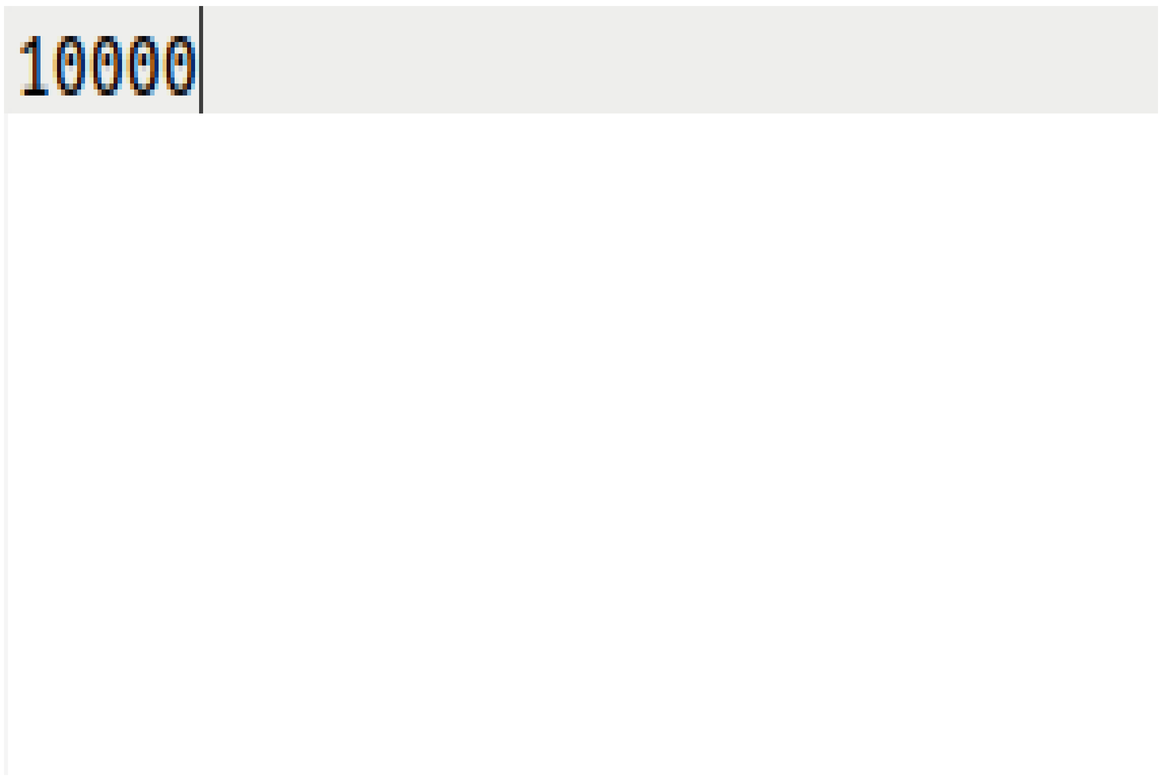
```
Enter Action type: [1) Deposit / 2) Withdraw / 3) Check Balance/ 4) Check Previo
us Statements]
2
Enter amount [xx.xx]
1000
Enter your Account name:
Mridul
Enter date of transaction. (Skipping this will log the transaction with today's
date.)

Making transaction of 1000

Transaction by Mridul finished. Logging transaction...
Your balances: 10000
```

Figure 3.4: Withdraw Money



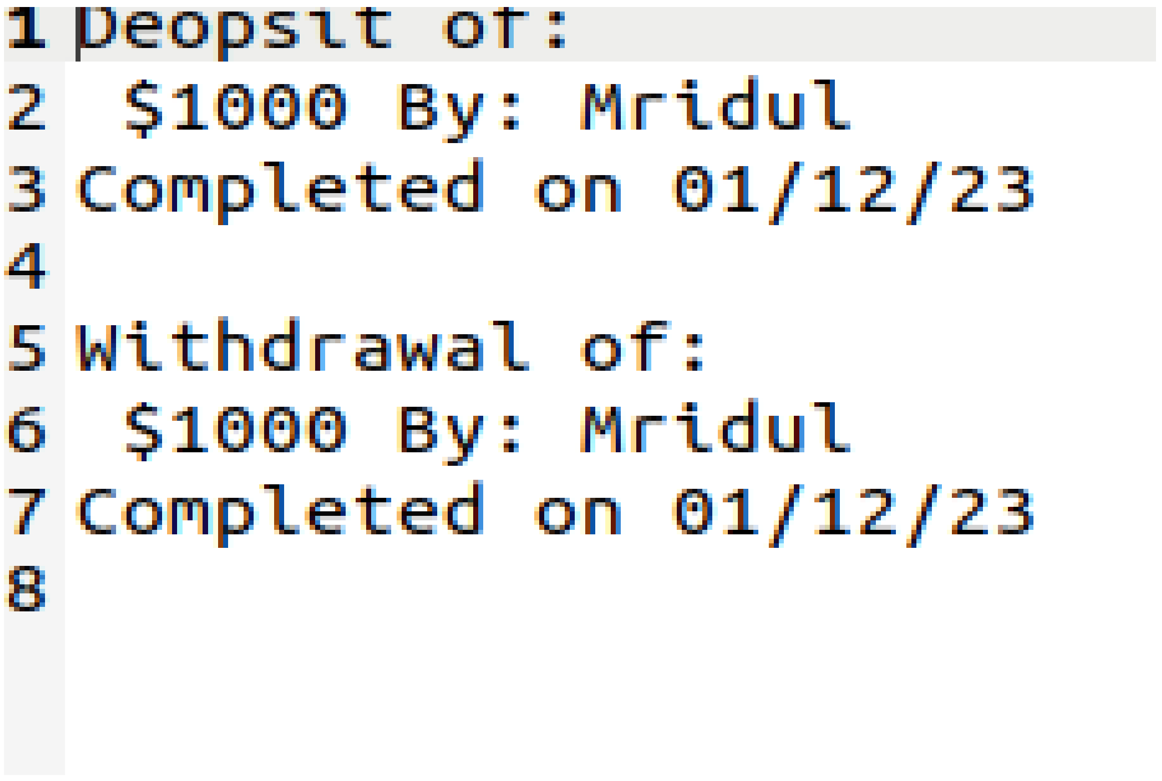
```
Deopsit of:
 $1000 By: Mridul
Completed on 01/12/23

Withdrawal of:
 $1000 By: Mridul
Completed on 01/12/23

```

Figure 3.5: Withdraw Statement



```
10000
```

Figure 3.6: Current Balance



```
Deopsit of:
 $1000 By: Mridul
Completed on 01/12/23

Withdrawal of:
 $1000 By: Mridul
Completed on 01/12/23
```

Figure 3.7: Checking Previous statements

Chapter 4

Conclusion

4.1 Discussion

In this project, a simple atm transaction management system was implemented using shell scripts. It had features like, 1) deposit money, 2) withdraw money, 3) Show balance, 4) Show statement. It doesn't have any errors and due to the data saving system using file management, the project is dynamic, although the data integrity or security wasn't maintained in any way. Also, this project is not a full-fledged bank management system. So, it can be called an incomplete version or early version as of now. But, it can be extended and turned into a better project using shell and some other languages as well.

4.2 Scope of Future Work

Since, the project is incomplete. It has a vast future scope. The scope of future works are:

- Implementation of Simple GUI
- Implementation of Loan giving system.
- Implementation of Receiving back loans with interest.
- Implementation of loan interest and amount count.
- Implementation of a better Database.
- Add Data integrity and security.

4.3 References

1. Author Initial. Author Surname, Title. City: Publisher, Year Published, p. Pages Used.

2. A. Rezi and M. Allam, ”Techniques in array processing by means of transformations, ” in Control and Dynamic Systems, Vol. 69, Multidemsional Systems, C. T. Leondes, Ed. San Diego: Academic Press, 1995, pp. 133-180.

3. O. B. R. Strimpel, ”Computer graphics,” in McGraw-Hill Encyclopedia of Science and Technology, 8th ed., Vol. 4. New York: McGraw-Hill, 1997, pp. 279-283.

4. K. Schwalbe, K. Schwalbe, Information Technology Project Management, 3rd ed. Boston: Course Technology, 2004.

5. M. N. DeMers, Fundamentals of Geographic Information Systems, 3rd ed. New York: John Wiley, 2005.

6. M. Bell, et al., Universities Online: A survey of online education and services in Australia, Occasional Paper Series 02-A. Canberra: Department of Education, Science and Training, 2002.