**DKTE SOCIETY’S**

**YASHWANTRAO CHAVAN POLYTECHNIC, ICHALKARANJI.**

DEPARTMENT OF

**COMPUTER SCIENCE AND ENGINEERING**

A

Micro Project Report On,

**“ATM Machine using Java”**

Submitted by,

|  |  |  |
| --- | --- | --- |
| Exam seat no | Students Name | Sign |
|  | Aditya Babaso Birangaddi |  |
|  | Atharav Sunil Barkale |  |
|  | Sumit Tanaji Biranje |  |
|  | Vedant Raju Sambare |  |

Under the guidance of

**Prof. M.J.Kanase**

ACADEMIC YEAR

2020-21

**D.K.T.E. SOCIETY’S**

**YASHWANTRAO CHAVAN POLYTECHNIC, ICHALKARANJI.**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING.

**CERTIFICATE**

This is to certify that,

|  |  |  |
| --- | --- | --- |
| Enrollment No | Students Name | Sign |
| 1915770160 | Aditya Babaso Birangaddi |  |
| 1915770159 | Atharav Sunil Barkale |  |
| 1915770161 | Sumit Tanaji Biranje |  |
| 19157701 | Vedant Raju Sambare |  |

Have successfully completed the micro project entitled,

**“ATM Machine using Java”**

In partial fulfillment of diploma in Computer Science and Engineering at MSBTE, Mumbai.

Date:

Place: Ichalkaranji.

Prof. M.J.Kanase Prof. Hatgine R. A.

Guide H.O.D

Prof. Kothali. A. P.

Principal

**ABSTRACT**

We have tried to explain about the ATM machine in Java Programming . This program consists of different function struct and other functions . This program helps user to handle the more records of ATM accounts . User can add new accounts and modify the accounts and their details. And we can change the passwords of the accounts and we withdraw deposit the money from accounts.

Since the introduction of Automated Teller Machines (ATMs) in Nigeria, cash handling has

really reduced especially for those who live in the city and have easy access to ATM

machines by means of their ATM cards. People no longer see the needs to carry excess cash

at home when they already have an ATM card since ATM machines can be easily

accessible. ATM operation did not turn out to be all rosy as it started out. This research

surveys ATM malfunctions in four states in Nigeria to ascertain the nature of malfunction

that is prevalent and formulates a modification algorithm/routine that should possibly correct

this ATM misfit.

**ACKNOWLEDGEMENT**

An endeavor over long period can be successful only with advice and guidance of many well-wishers. Our sincere thanks to **Prof. Hatgine R. A.** H.O.D. of Computer Science and Engineering, for gives us the opportunity to conduct this micro project.

We would like to thank the Computer Science and Engineering department for assistance and constant source of encouragement. We wish to express our profound and deep sense of gratitude of **Prof. M. J. Kanase** for spending their valuable time to extend help us in every step of our project.

I would like to thank the staff of Computer Science and Engineering department for their generous guidance.

Last but not the least we would like to thank our classmates and family for their help in every way and constant inspiration in project and for the success of this project and project report.

Thanking you.

Mr. Aditya Babaso Birangaddi

Mr. Atharva Sunil Barkale

Mr. Sumit Tanaji Biranje

Mr. Vendant Raju Sambare

INDEX

|  |  |  |
| --- | --- | --- |
| Sr .No. | Title | Page no. |
| 1 | Introduction | 1 |
| 2 | Program. | 6 |
| 3 | Output. | 10 |
| 4 | Conclusion | 13 |
| 5 | Reference | 14 |

**INTRODUCTION**

The aim of the ATM Simulation System project is to build a Java based ATM (Automated Teller Machine) Simulation System. The introduction of ATM’s by various banks have brought about freedom from the interminable queues in front of withdrawal counters at banks. This ATM Simulation System requires the constant updating of records between the bank servers and a spread out network of ATM’s

Security is the foundation of a good ATM system. This system will provide for secure authenticated connections between users and the bank servers. The whole process will be automated right from PIN (Personal Identification Number) validation to transaction completion. ATM Simulation System will enable two important features of an ATM, reduction of human error in the banking system and the possibility of 24 hour personal banking. The card details and PIN database will be a secure module that will not be open to routine maintenance, the only possibility of access to this database will be through queries raised from an ATM in the presence of a valid bank ATM card.

**Program:-**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*OPTIONMANU.JAVA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.text.DecimalFormat;

import java.text.DecimalFormatSymbols;

import java.util.\*;

import java.io.IOException;

public class OptionMenu extends Account {

Scanner menuInput = new Scanner(System.in);

DecimalFormat moneyFormat = new DecimalFormat("'$'###,##0.00");

HashMap<Integer, Integer> data = new HashMap<Integer, Integer>();

public void getLogin() throws IOException {

int x = 1;

do {

try {

data.put(22222, 11111);

data.put(44444, 33333);

System.out.println("Welcome to the ATM Project!");

System.out.println("Enter your ATM Number");

setCustomerNumber(menuInput.nextInt());

System.out.print("Enter your PIN Number: ");

setPinNumber(menuInput.nextInt());

} catch (Exception e) {

System.out.println("\n" + "Invalid Character(s). Only Numbers." + "\n");

x = 2;

}

int cn = getCustomerNumber();

int pn = getPinNumber();

if (data.containsKey(cn) && data.get(cn) == pn) {

getAccountType();

} else

System.out.println("\n" + "Wrong Customer Number or Pin Number" + "\n");

} while (x == 1);

}

public void getAccountType() {

System.out.println("Select the Account you Want to Access: ");

System.out.println(" Type 1 - Checking Account");

System.out.println(" Type 2 - Saving Account");

System.out.println(" Type 3 - Exit");

int selection = menuInput.nextInt();

switch (selection) {

case 1:

getChecking();

break;

case 2:

getSaving();

break;

case 3:

System.out.println("Thank You for using this ATM, bye. \n");

break;

default:

System.out.println("\n" + "Invalid Choice." + "\n");

getAccountType();

}

}

public void getChecking() {

System.out.println("Checking Account: ");

System.out.println(" Type 1 - View Balance");

System.out.println(" Type 2 - Withdraw Funds");

System.out.println(" Type 3 - Deposit Funds");

System.out.println(" Type 4 - Exit");

System.out.print("Choice: ");

int selection = menuInput.nextInt();

switch (selection) {

case 1:

System.out.println("Checking Account Balance: " + moneyFormat.format(getCheckingBalance()));

getAccountType();

break;

case 2:

getCheckingWithdrawInput();

getAccountType();

break;

case 3:

getCheckingDepositInput();

getAccountType();

break;

case 4:

System.out.println("Thank You for using this ATM, bye.");

break;

default:

System.out.println("\n" + "Invalid Choice." + "\n");

getChecking();

}

}

public void getSaving() {

System.out.println("Saving Account: ");

System.out.println(" Type 1 - View Balance");

System.out.println(" Type 2 - Withdraw Funds");

System.out.println(" Type 3 - Deposit Funds");

System.out.println(" Type 4 - Exit");

System.out.print("Choice: ");

int selection = menuInput.nextInt();

switch (selection) {

case 1:

System.out.println("Saving Account Balance: " + moneyFormat.format(getSavingBalance()));

getAccountType();

break;

case 2:

getsavingWithdrawInput();

getAccountType();

break;

case 3:

getSavingDepositInput();

getAccountType();

break;

case 4:

System.out.println("Thank You for using this ATM, bye.");

break;

default:

System.out.println("\n" + "Invalid Choice." + "\n");

getChecking();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ATM.JAVA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.io.IOException;

public class ATM extends OptionMenu{

public static void main(String[] args) throws IOException {

OptionMenu optionMenu= new OptionMenu();

optionMenu.getLogin();

}

};

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ACCOUNT.JAVA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import java.text.DecimalFormat;

import java.util.\*;

public class Account {

Scanner input = new Scanner(System.in);

DecimalFormat moneyFormat = new DecimalFormat("'$'###,##0.00");

public int setCustomerNumber(int customerNumber){

this.customerNumber = customerNumber;

return customerNumber;

}

public int getCustomerNumber(){

return customerNumber;

}

public int setPinNumber(int pinNumber){

this.pinNumber = pinNumber;

return pinNumber;

}

public int getPinNumber(){

return pinNumber;

}

public double getCheckingBalance(){

return checkingBalance;

}

public double getSavingBalance(){

return savingBalance;

}

public double calcCheckingWithdraw(double amount){

checkingBalance = (checkingBalance - amount);

return checkingBalance;

}

public double calcSavingWithdraw(double amount){

savingBalance = (savingBalance - amount);

return savingBalance;

}

public double calcCheckingDeposit(double amount){

checkingBalance = (checkingBalance + amount);

return checkingBalance;

}

public double calcSavingDeposit(double amount){

savingBalance = (savingBalance + amount);

return savingBalance;

}

public void getCheckingWithdrawInput(){

System.out.println("Checking Account Balance: " + moneyFormat.format(checkingBalance));

System.out.print("Amount you want to withdraw from Checking Account: ");

double amount =input.nextDouble();

if((checkingBalance-amount)>=0){

calcCheckingWithdraw(amount);

System.out.println("New Checking Account Balance: " + moneyFormat.format(checkingBalance));

}else{

System.out.println("Balance Cannot be Negative." + "\n");

public void getsavingWithdrawInput(){

System.out.println("Saving Account Balance: " + moneyFormat.format(savingBalance));

System.out.print("Amount you want to withdraw from saving Account: ");

double amount =input.nextDouble();

if((savingBalance-amount)>=0){

calcSavingWithdraw(amount);

System.out.println("New saving Account Balance: " + moneyFormat.format(savingBalance));

}else{

System.out.println("Balance Cannot be Negative." + "\n";

public void getCheckingDepositInput(){

System.out.println("Checking Account Balance: " + moneyFormat.format(checkingBalance));

System.out.print("Amount you want to Deposit from Checking Account: ");

double amount =input.nextDouble();

if((checkingBalance+amount)>=0){

calcCheckingDeposit(amount);

System.out.println("New Checking Account Balance: " + moneyFormat.format(checkingBalance));

}else{

System.out.println("Balance Cannot be Negative." + "\n");

}

}

public void getSavingDepositInput(){

System.out.println("Saving Account Balance: " + moneyFormat.format(savingBalance));

System.out.print("Amount you want to Deposit from saving Account: ");

double amount =input.nextDouble();

if((savingBalance+amount)>=0){

calcSavingDeposit(amount);

System.out.println("New saving Account Balance: " + moneyFormat.format(savingBalance));

}else{

System.out.println("Balance Cannot be Negative." + "\n");

}

}

private int customerNumber;

private int pinNumber;

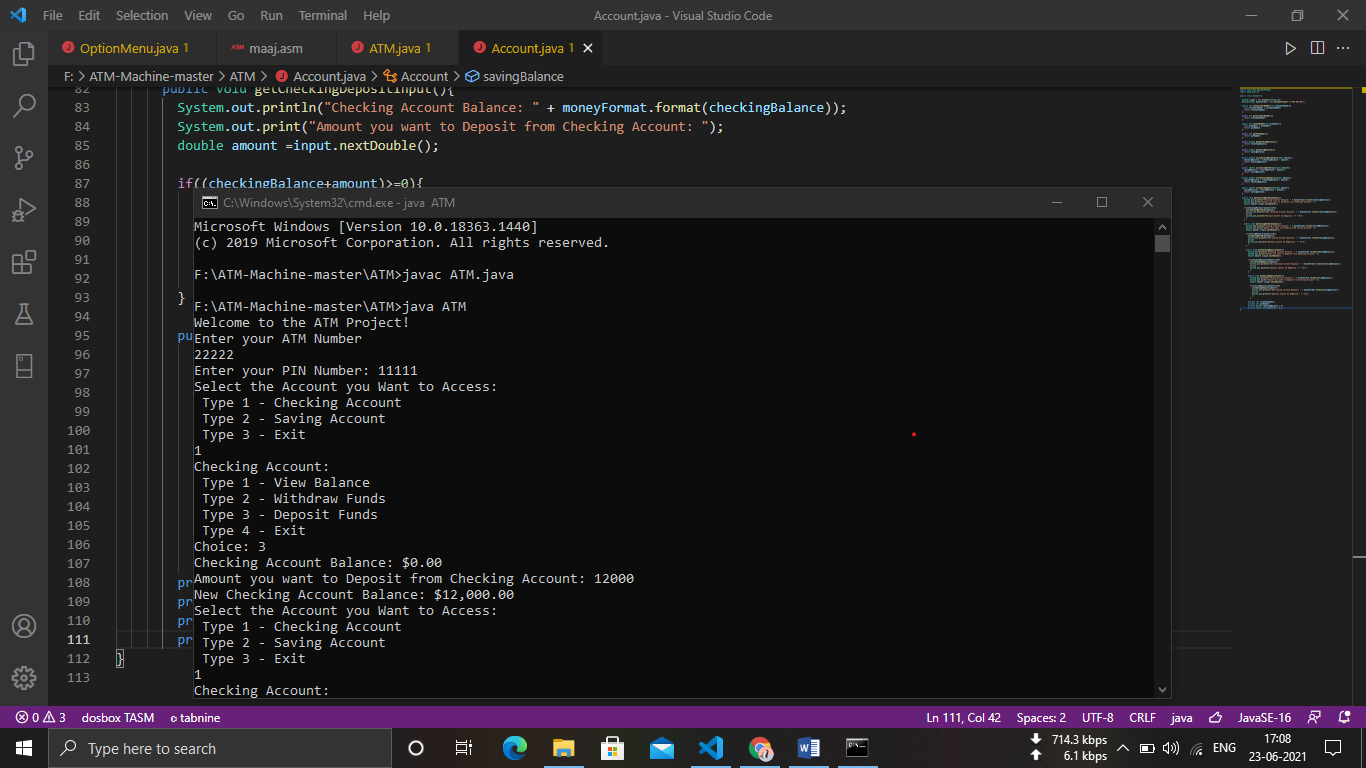
private double checkingBalance = 0;

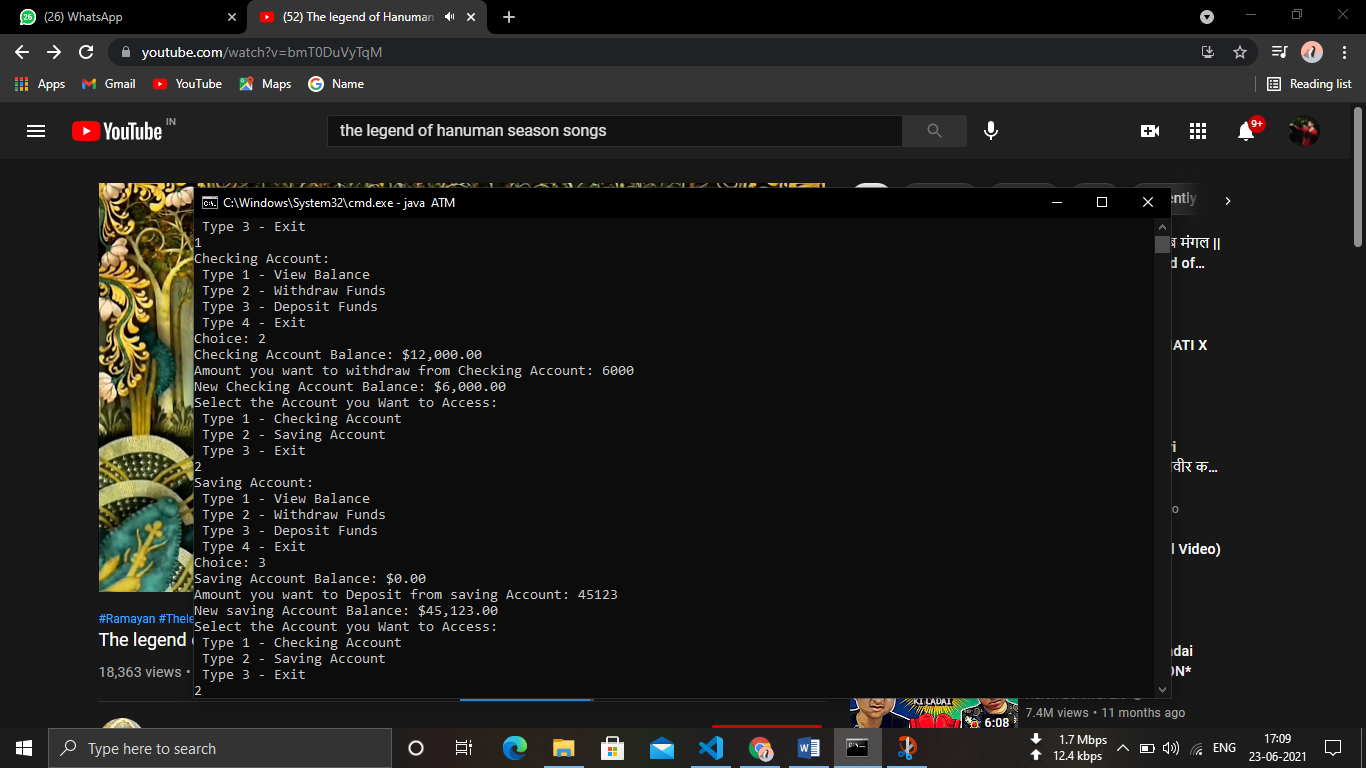
private double savingBalance = 0;

}

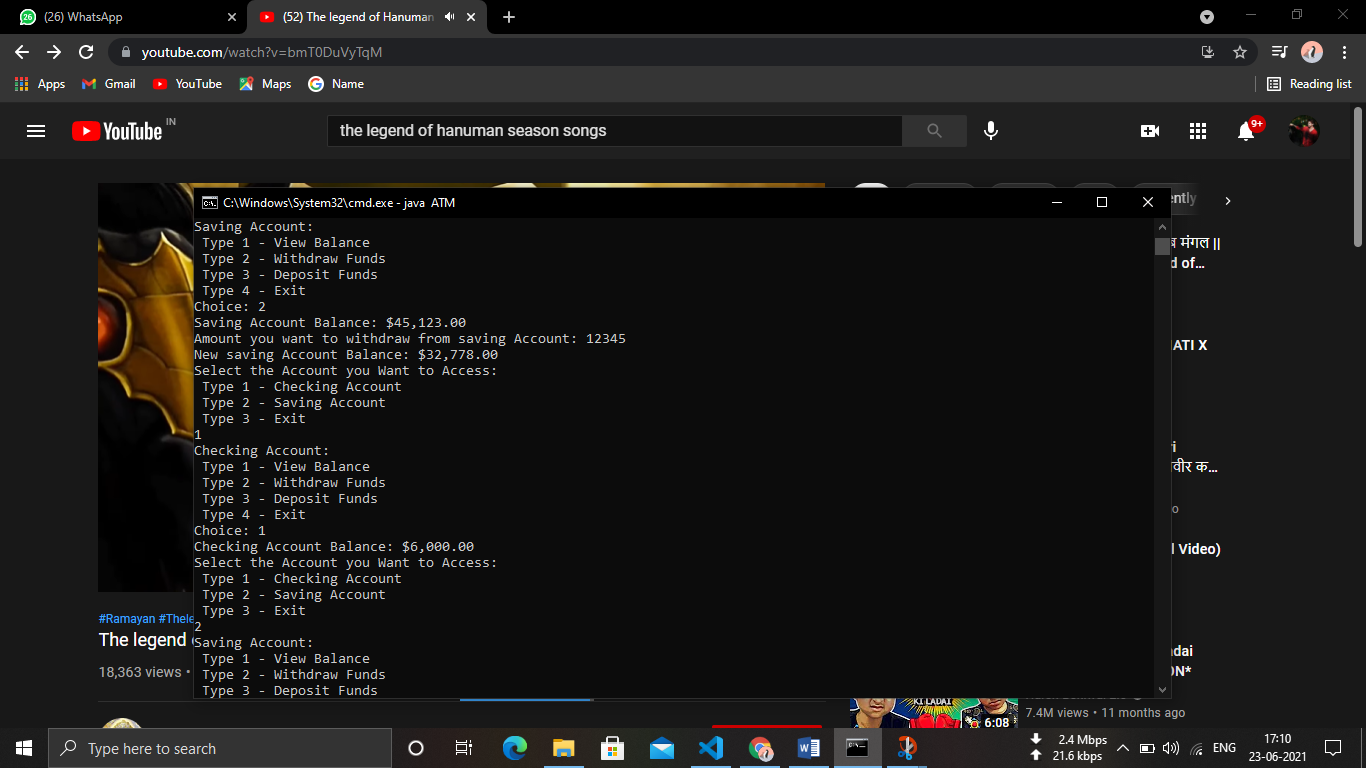
**OUTPUT**

**1]**

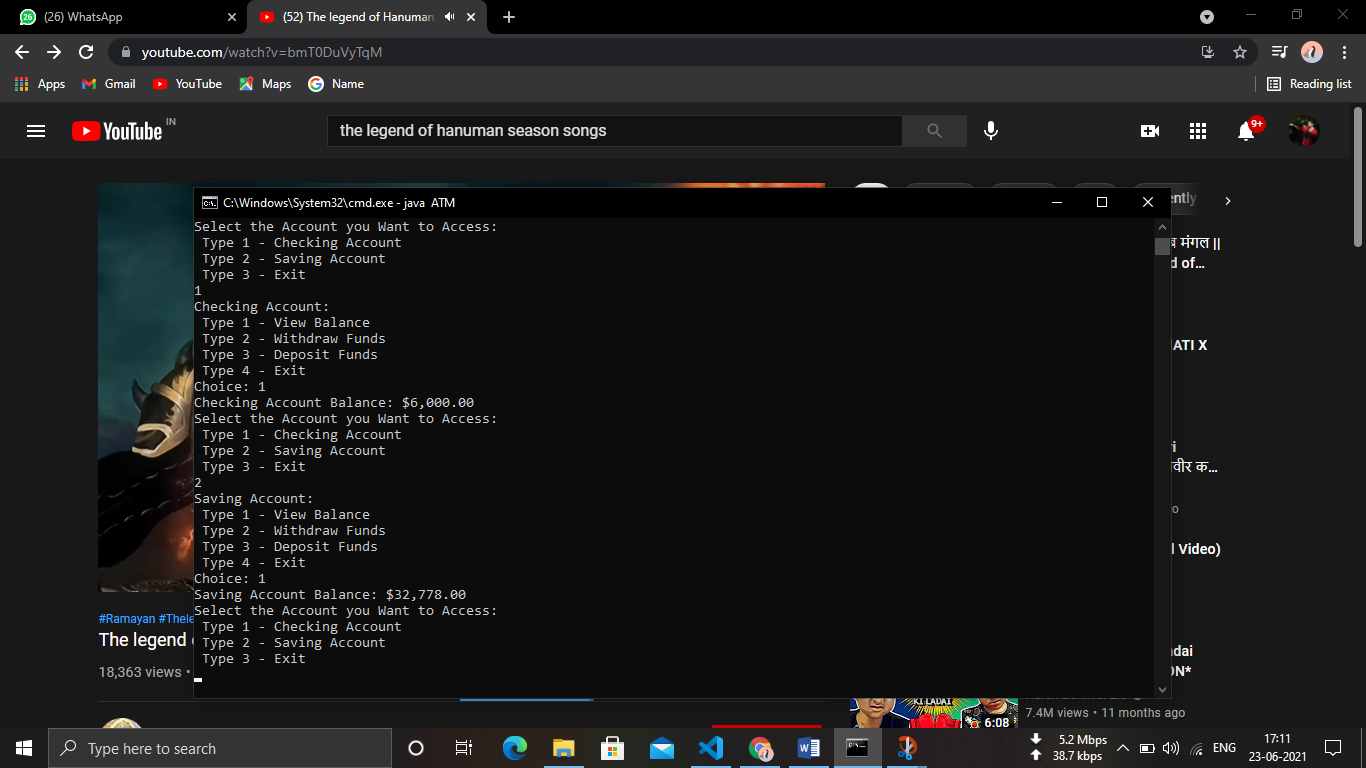
****

**2]**

**3]**

****

**4]**

** Conclusion**

The ATM network has to provide software interfaces to the

software used by different banks and different network software.

Also there should be no restriction of the ATM network to a

specific network protocol as long as the performance

requirements are satisfied. Other performance requirements may

include:

1. Error messages should be displayed at least 30 seconds each

time.

2.If there is no response from the bank computer after a request

within 2 minutes, the card is rejected with an error message.

3. The ATM dispenses money if and only if the withdrawal from

the account is processed and accepted by the bank.

4. Each bank may be processing transactions from several ATMs

at the same time.

**Reference**

Websites

1. ATM Forum, www.atmforum.com. ATM Forum, Worldwide

Headquarters, Mountain View, CA 94040-1313

2. ATM/HFC Network Simulator, Operation and Programming

Guide, Ver 4.0, N. Golmie, F.

3. ATM Angles for IP Edge, D. Pappalardo, Network World, 16,

7, 1999, pp 1 and 61.

4. High Performance Communication Networks, J. Walrand, P.

Varaiya, Morgan Kaufman, 1996