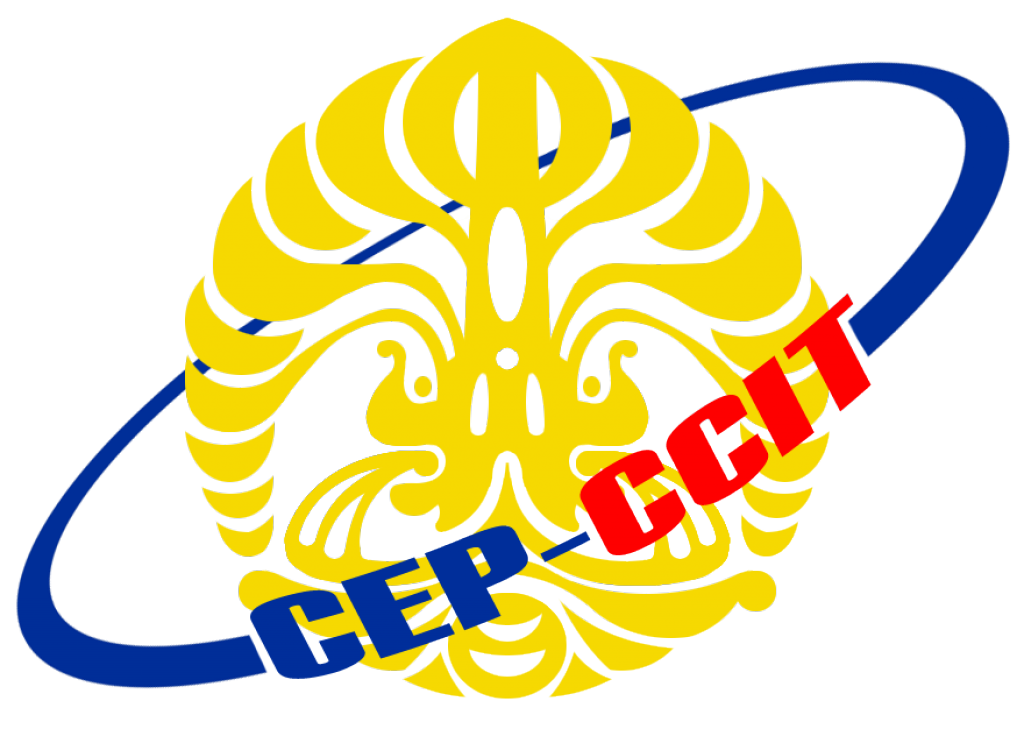
**PROJECT REPORT**

**ATM CONSOLE SIMULATION SYSTEM**

**Name : Aji Trinioferi** { 4817050025 }

**Cahya Mulyadi** { 4817050069 }

**Julius Danes Nugroho** { 4817050199 }

**Faculty : Mr. Musawarman S.Kom**

**ENGINEERING CENTRE 3rd FLOOR, FACULTY OF ENGINEERING**

**UNIVERSITY OF INDONESIA**

**PROJECT ON**

**ATM CONSOLE SIMULATION SYSTEM**

**Developed By**

**Name : Aji Trinioferi** { 4817050025 }

**Cahya Mulyadi** { 4817050069 }

**Julius Danes Nugroho** { 4817050199 }

**NIIT**

**ATM CONSOLE SIMULATION SYSTEM**

Batch Code : 2NAP1

Start Date : March 22, 2018

End Date : April 2 , 2018

Name Of The Coordinator : Mr. Musawarman, S.Kom

Name Of Developer : Aji Trinioferi { 4817050025 }

Cahya Mulyadi { 4817050069 }

Julius Danes N. { 4817050199 }

Date Of Submission : April 3, 2018

**NIIT**

**NIIT**

**CERTIFICATE**

This is to certify that this report, titled “ATM Console Simulation System”, embodies the original work done by Aji Trinioferi, Cahya Mulyadi, and Julius Danes Nugroho. In partial fulfillment of his/her course requirement at NIIT.

Coordinator :

***Mr. Musawarman, S.Kom***

**ACKNOWLEDGEMENT**

Praise and huge thanks to Allah SWT. for giving us blessings, health, ability to be able to finish this Project with the title “ATM Console Simulation System”, without any of it we won’t be able to complete it right on time. Despite many hurdles and obstacles that we experienced in the course of the work, but we managed to finish it properly.

We have benefited a lot from the feedback and suggestions given to us by Mr. Musawarman, S.Kom.

**System Analysis**

**System Summary :**

Automated Teller Machine (ATM) is an electronic banking outlet, which allows customers to complete basic transactions without the aid of a branch representative or teller. Anyone with a credit card or debit card can access most ATMs. Console application that can be used for doing transaction in Bank such as Saving Balance, Money Transfer, Balance Check, and many more.

**Menu in The Application :**

* Master Main Display (For Login as Admin or User)
  + Master Menu Admin Display

- Create User Display

- All User Display

- Search User Display

- Update User Display

- Delete User Display

* + Master Menu User Display

- Check Balance Display

- Transfer Cash Display

- Cash Withdraw Display

- Cash Deposit Display

- Mutation Display

- Change PIN Display

**CLASS**

1. **Program Main**

* Methods :
* Main
* Swipe
* Emphasis
* Warning

1. **MasterMenuAdmin**

* Methods :
* Admin

1. **MasterMenuUser**

* Methods :
* User

1. **ProgramAdmin**

* Methods :
* CreateUserData
* DisplayUserData
* SearchUserData
* UpdateUserData
* DeleteLines
* SearchDataUserForDelete

1. **ProgramUser**

* Methods :
* GetID
* CashDeposit

**CLASS**

* Cash Withdraw
* ScanMoney
* Mutation
* CheckBalance
* ChangePIN
* Transfer
* TransferRek
* TransferBank

**CONSOLE INTERFACE**

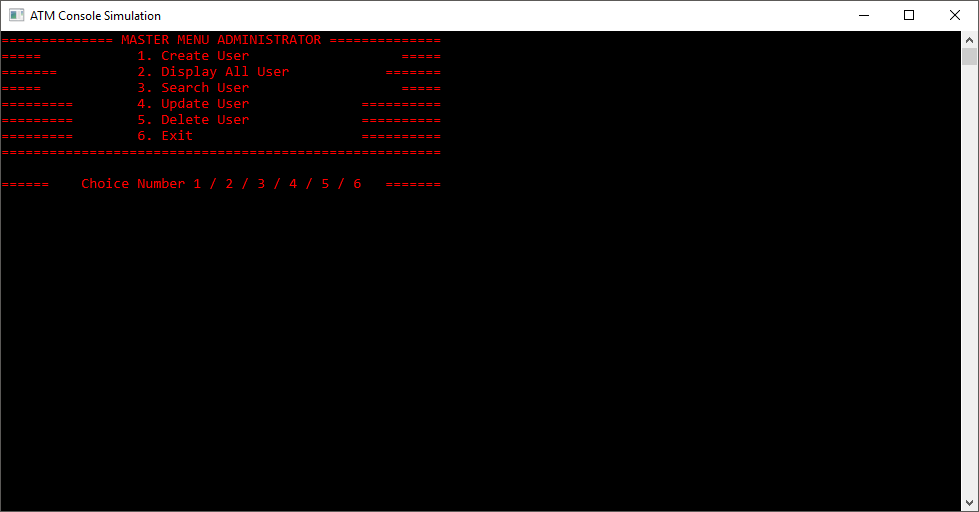
1. Enter RFID Code



1. Enter PIN

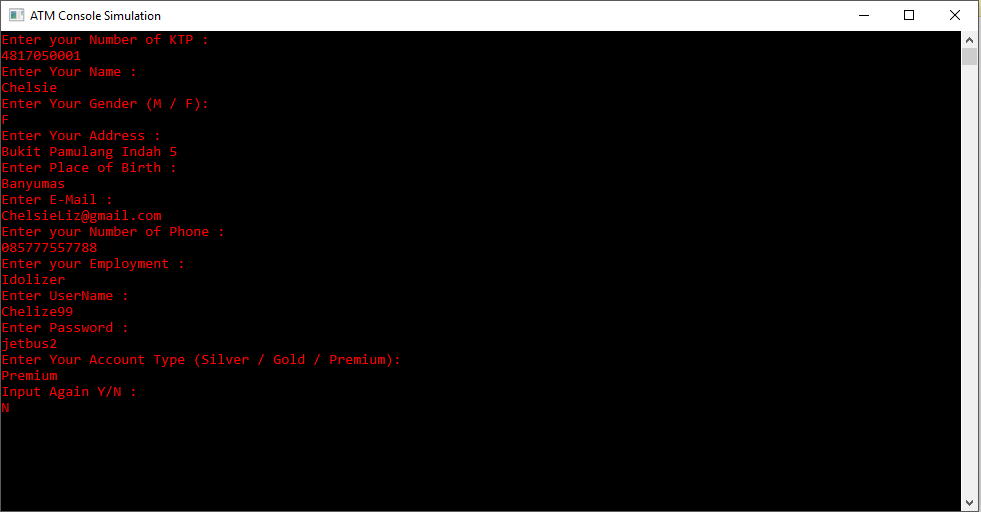


1. Master Menu Admin



**CONSOLE INTERFACE**

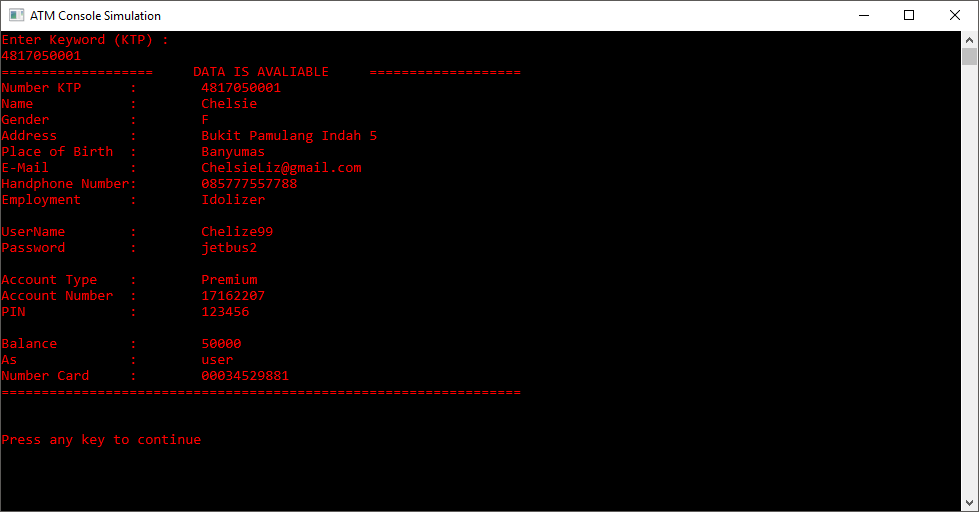
1. Create User Data



1. Display All User

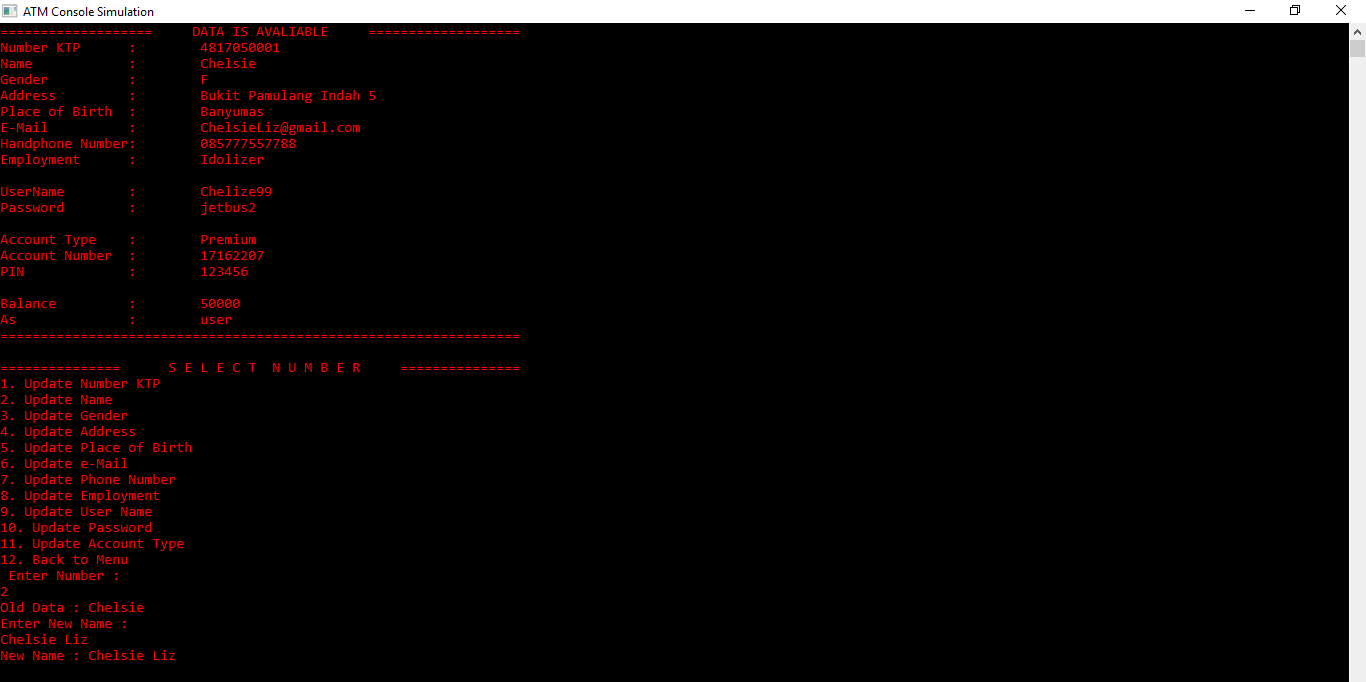


1. Search User

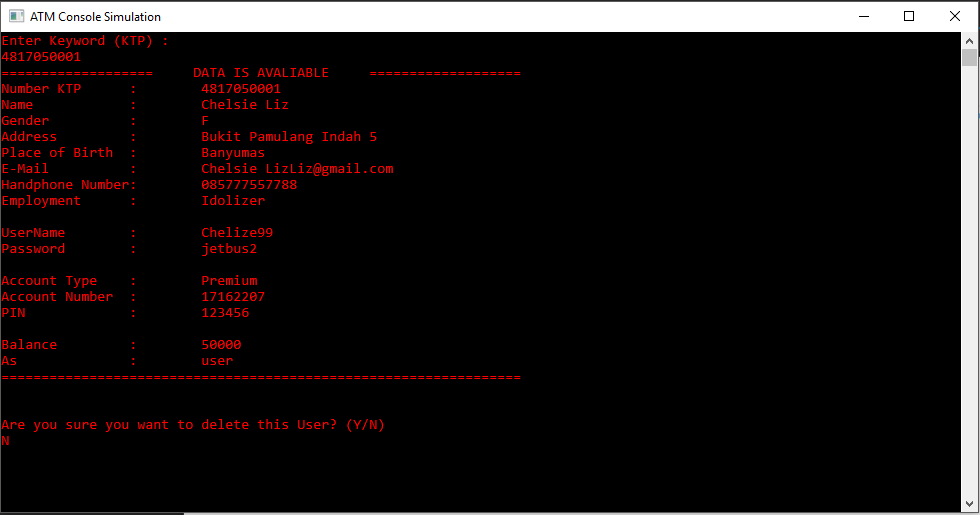


**CONSOLE INTERFACE**

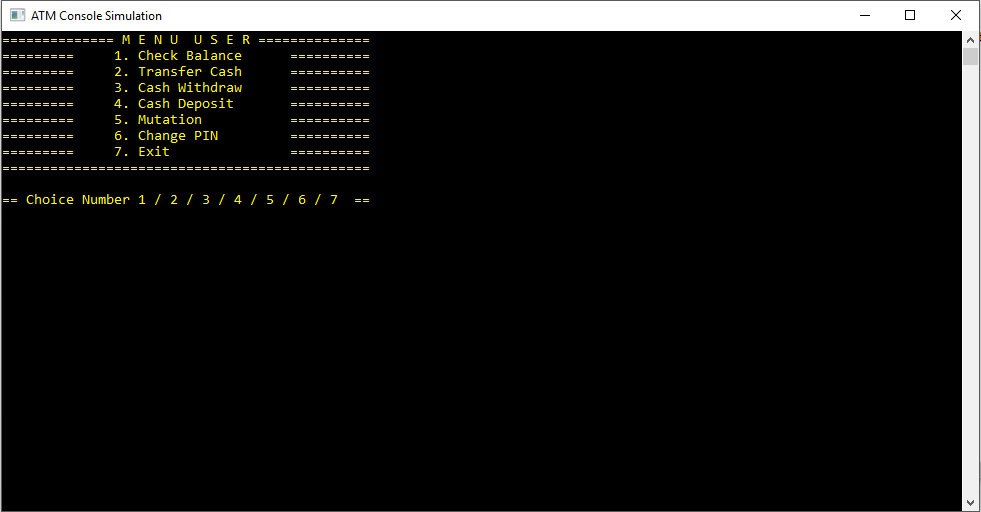
1. Update User



1. Delete User

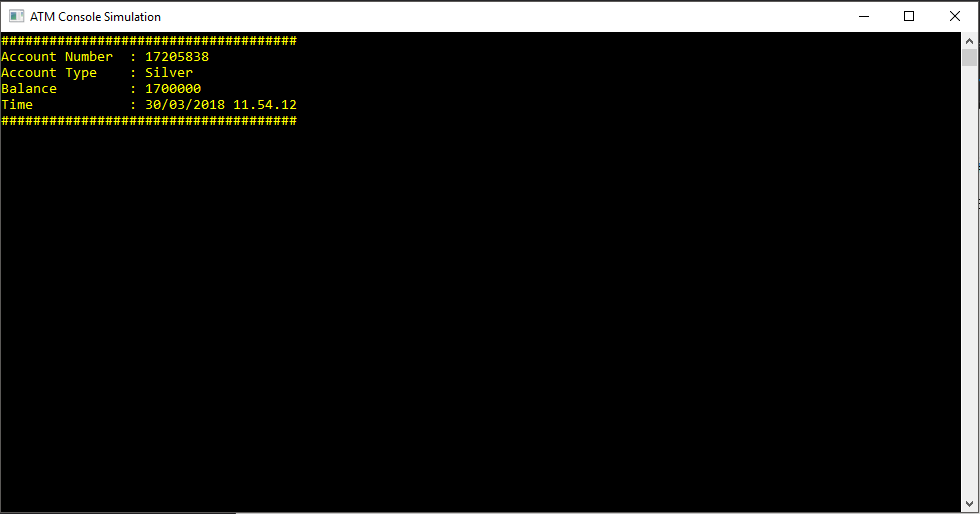


1. Master Menu User



**CONSOLE INTERFACE**

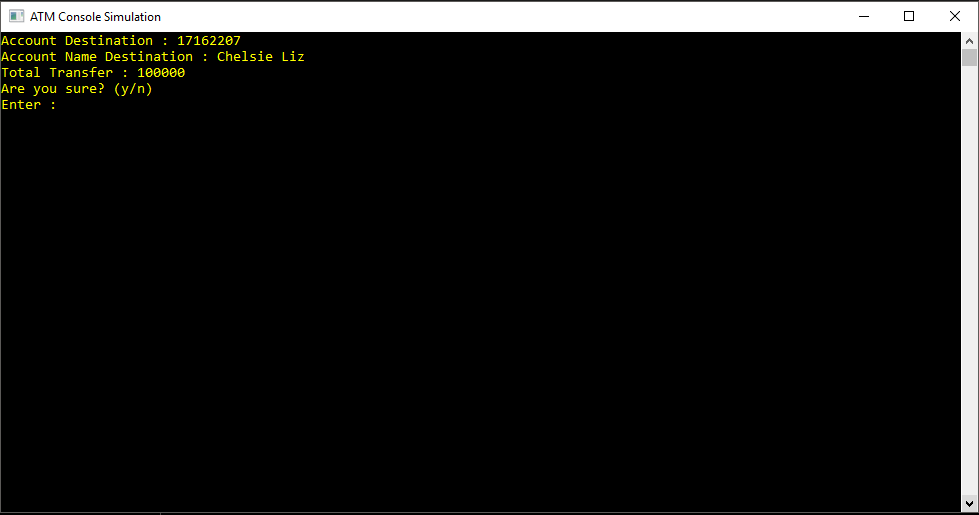
1. Check Balance



1. Transfer Cash

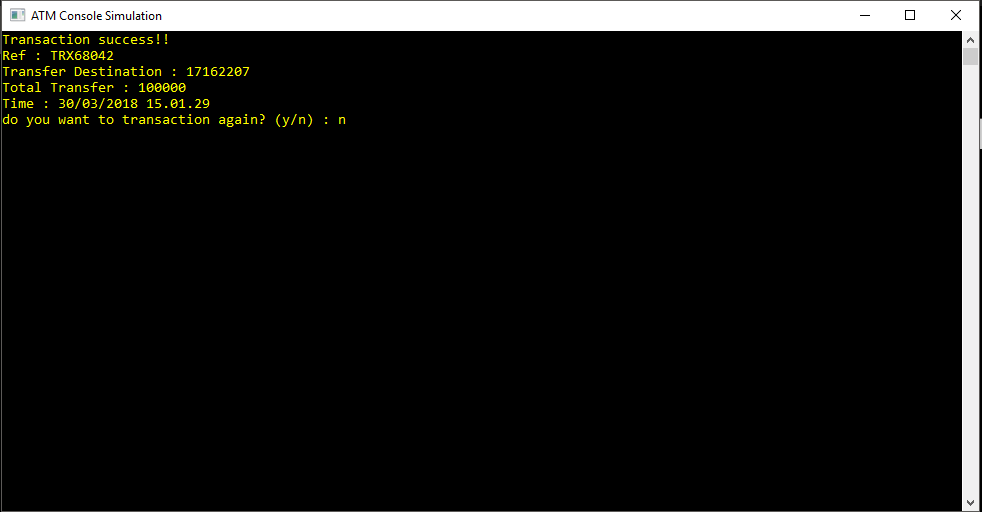


1. Confirmation Transfer

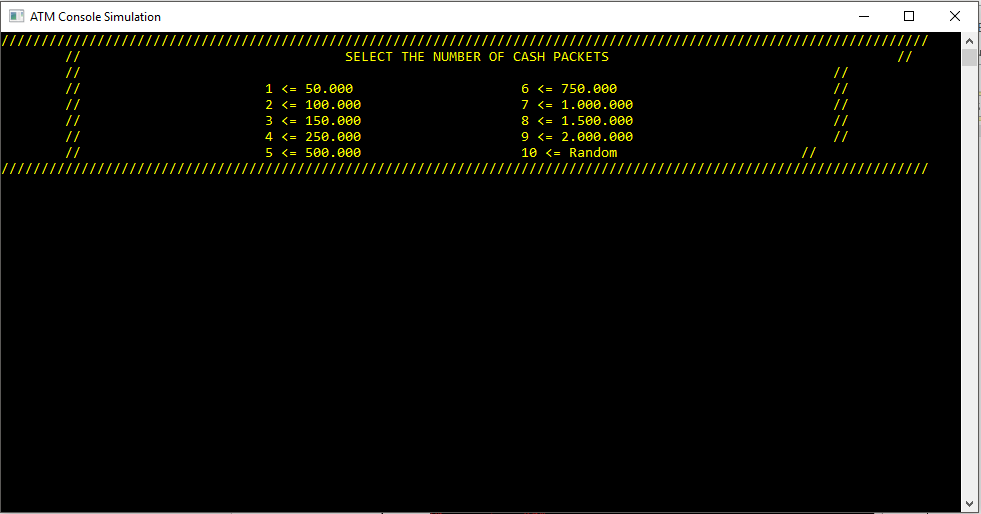


**CONSOLE INTERFACE**

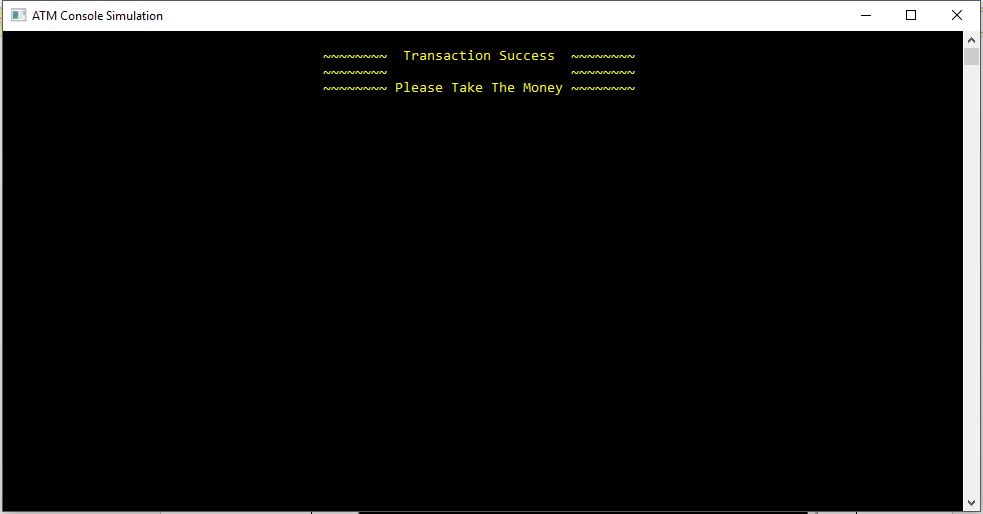
1. Notice After Transfer



1. Cash Withdraw / Cash Deposit

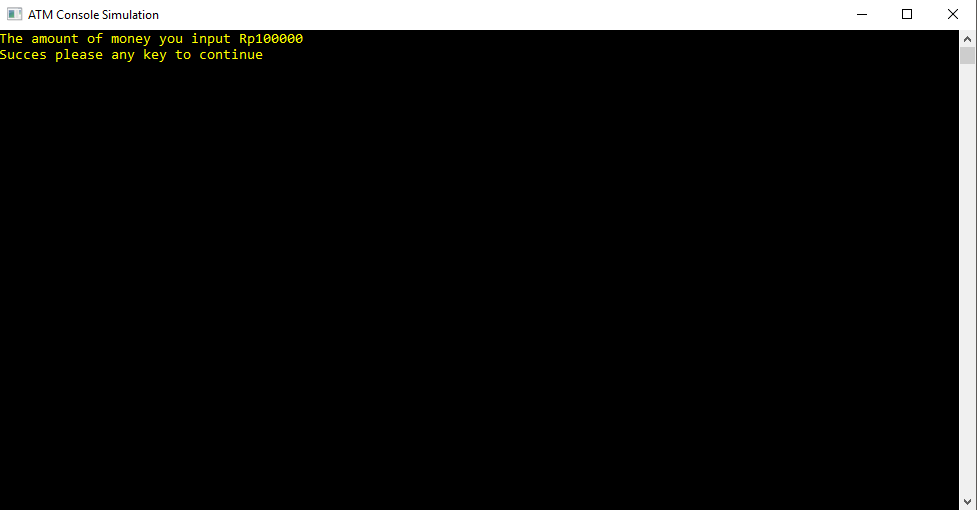


1. Cash Withdraw Notice



**CONSOLE INTERFACE**

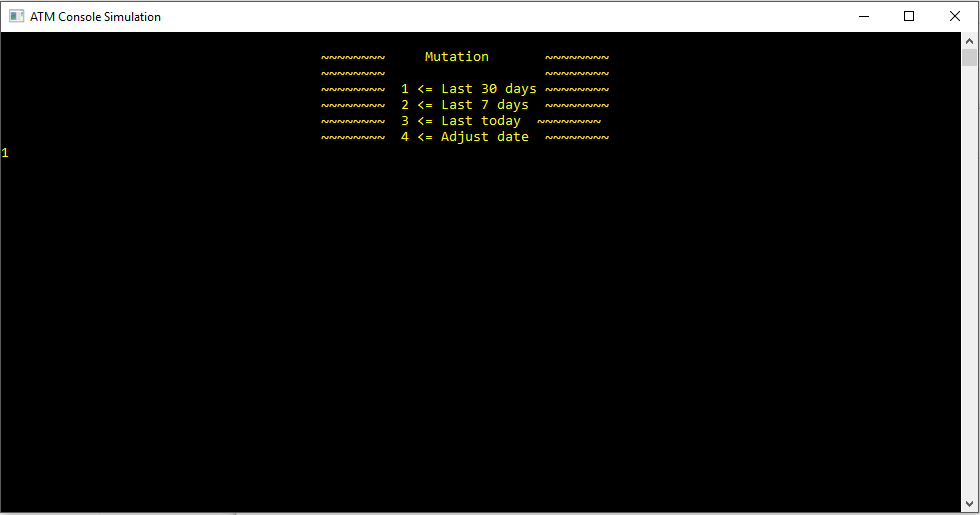
1. Notice After Enter Money (Cash Deposit)



1. Cash Deposit Notice

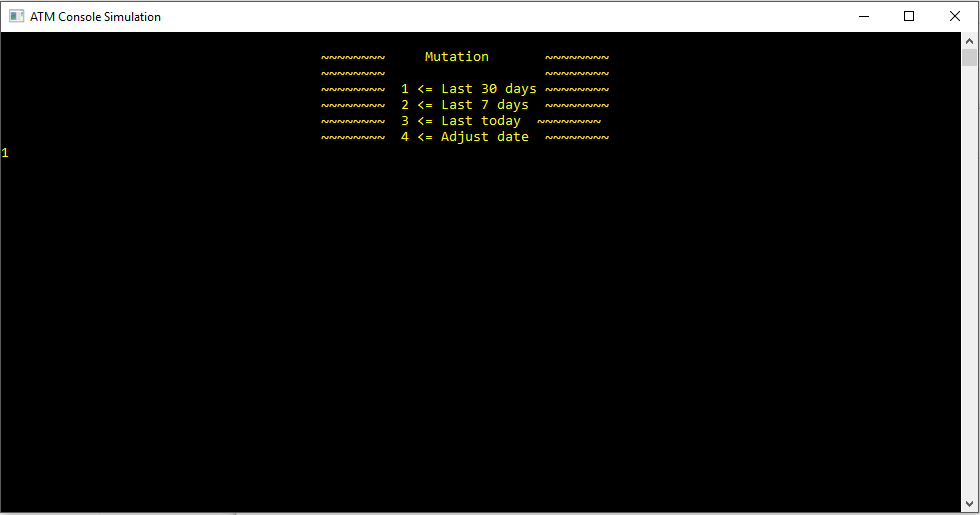


1. Mutation

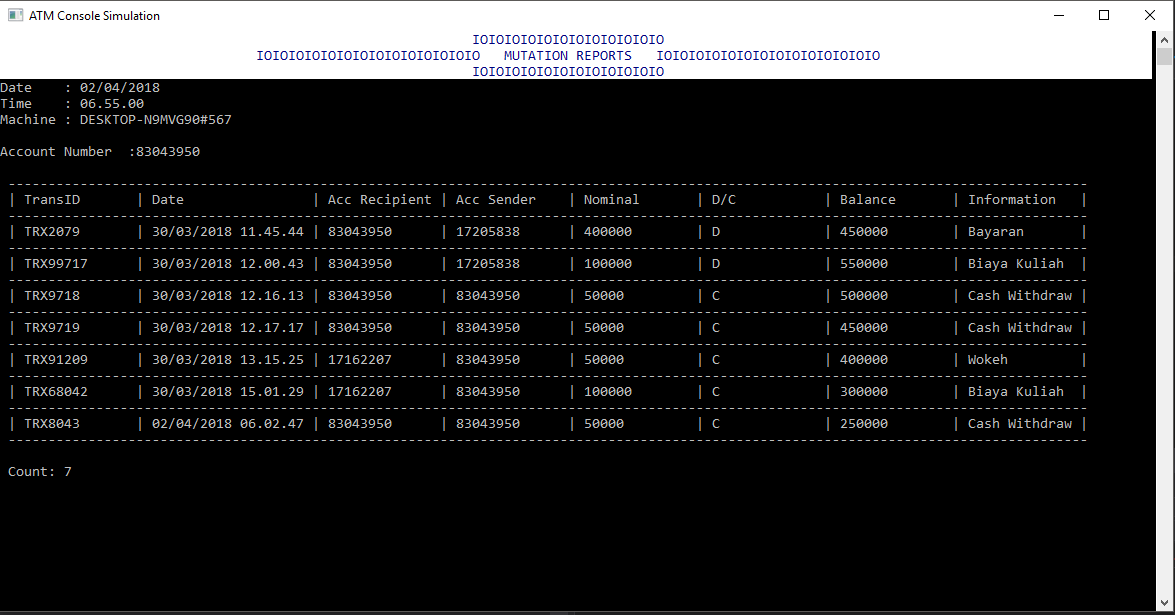


**CONSOLE INTERFACE**

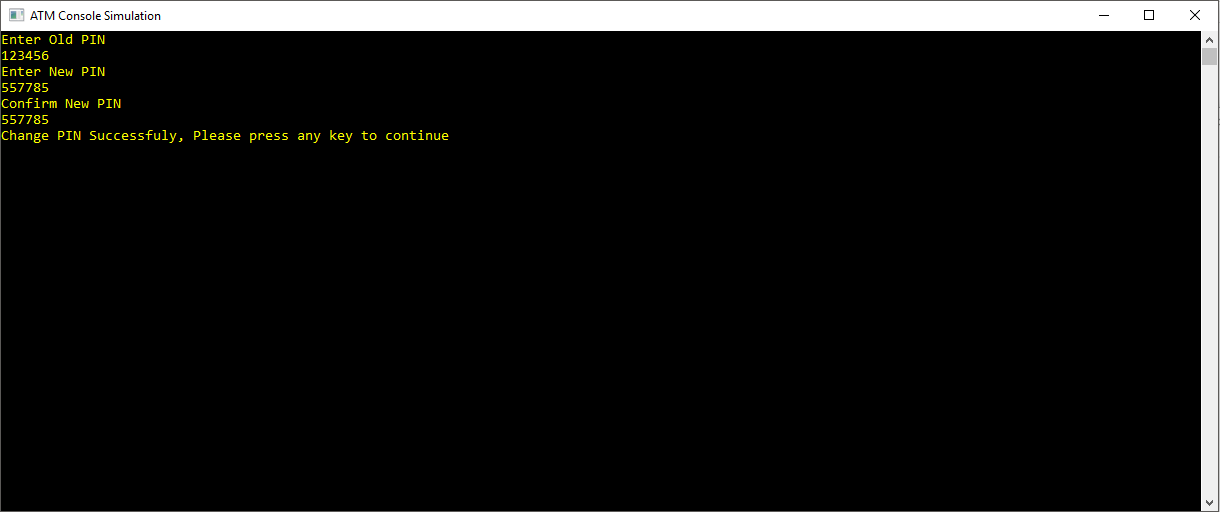
1. Select Mutation



1. Mutation Display



1. Change PIN



**CODES (MASTER MAIN)**

using System.Text;

using System.IO;

using System.Data;

using ConsoleTables;

using System.Threading.Tasks;

using System.Threading;

namespace Wokeh

{

class ProgramMain

{

static string nokartu = "";

static string path = (@"" + Environment.CurrentDirectory + "\\");

static void Main()

{

Console.Title = "ATM Console Simulation";

Console.OutputEncoding = System.Text.Encoding.UTF8;

bool a = true;

while (a)

{

Console.WriteLine("Waiting for card swipe...");

if (string.IsNullOrWhiteSpace(nokartu))

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

nokartu = Console.ReadLine();

Console.ResetColor();

}

Console.Clear();

Console.WriteLine("Input PIN");

string pin = "";

ConsoleKeyInfo key;

do

{

key = Console.ReadKey(true);

// Backspace Should Not Work

if (key.Key != ConsoleKey.Backspace && key.Key != ConsoleKey.Enter)

{

pin += key.KeyChar;

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.Write("\*");

Console.ResetColor();

}

else

{

if (key.Key == ConsoleKey.Backspace && pin.Length > 0)

{

pin = pin.Substring(0, (pin.Length - 1));

Console.Write("\b \b");

}

}

}// Stops Receving Keys Once Enter is Pressed

**CODES (MASTER MAIN)**

while (key.Key != ConsoleKey.Enter);

var data = new object[20];

foreach (string line in File.ReadLines(path + "UserData.txt"))

{

data = line.Split('#');

if (nokartu.Equals(data[16]))

{

break;

}

else

{

continue;

}

}

if (nokartu.Equals(data[16]) && pin.Equals(data[12]))

{

int NoRek = Convert.ToInt32(data[11]);

a = false;

if (data[15].Equals("user"))

{

Console.Clear();

MasterMenuUser.User(NoRek);

}

else

{

Console.Clear();

MasterMenuAdmin.Admin();

}

}

else

{

Warning("\n\u0416\u0416\u0416 Wrong card or PIN \u0416\u0416\u0416");

Console.ReadKey();

Console.Clear();

nokartu = "";

}

}

}

static void Swipe()

{

}

public static void Emphasis(string message)

{

Console.BackgroundColor = ConsoleColor.White;

Console.ForegroundColor = ConsoleColor.DarkBlue;

Console.WriteLine(message);

Console.ResetColor();

}

public static void Warning(string message)

{

Console.BackgroundColor = ConsoleColor.DarkRed;

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine(message);

Console.ResetColor();

}

}

}

**CODES (MASTER MENU ADMIN)**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace Wokeh

{

class MasterMenuAdmin

{

public static void Admin()

{

int choice;

Console.Clear();

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("============== MASTER MENU ADMINISTRATOR ==============");

Console.WriteLine("===== 1. Create User =====");

Console.WriteLine("======= 2. Display All User =======");

Console.WriteLine("===== 3. Search User =====");

Console.WriteLine("========= 4. Update User ==========");

Console.WriteLine("========= 5. Delete User ==========");

Console.WriteLine("========= 6. Exit ==========");

Console.WriteLine("=====================================================\n");

Console.WriteLine("====== Choice Number 1 / 2 / 3 / 4 / 5 / 6 =======");

choice = Convert.ToInt32(Console.ReadLine());

switch (choice)

{

case 1:

ProgramAdmin.CreateUserData();

Console.Clear();

MasterMenuAdmin.Admin();

break;

case 2:

ProgramAdmin.DisplayUserData();

Console.Clear();

MasterMenuAdmin.Admin();

break;

case 3:

ProgramAdmin.SearchUserData();

Console.Clear();

MasterMenuAdmin.Admin();

break;

case 4:

ProgramAdmin.UpdateUserData();

Console.Clear();

MasterMenuAdmin.Admin();

break;

case 5:

ProgramAdmin.SearchDataUserForDelete();

Console.Clear();

MasterMenuAdmin.Admin();

break;

case 6:

Environment.Exit(0);

break;

default:

Console.WriteLine("\nInvalid Choice!!");

Console.ReadKey();

Console.Clear();

MasterMenuAdmin.Admin();

break;

}

Console.ReadKey();

Console.Clear();

}

}

}

**CODES (MASTER MENU USER)**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace Wokeh

{

class MasterMenuUser

{

public static void User(int asd)

{

int NoRek = asd;

int choice;

Console.ForegroundColor = ConsoleColor.Yellow;

Console.WriteLine("============== M E N U U S E R ==============");

Console.WriteLine("========= 1. Check Balance ==========");

Console.WriteLine("========= 2. Transfer Cash ==========");

Console.WriteLine("========= 3. Cash Withdraw ==========");

Console.WriteLine("========= 4. Cash Deposit ==========");

Console.WriteLine("========= 5. Mutation ==========");

Console.WriteLine("========= 6. Change PIN ==========");

Console.WriteLine("========= 7. Exit ==========");

Console.WriteLine("==============================================\n");

Console.WriteLine("== Choice Number 1 / 2 / 3 / 4 / 5 / 6 / 7 ==");

choice = Convert.ToInt32(Console.ReadLine());

switch (choice)

{

case 1:

Console.Clear();

ProgramUser.CheckBalance(NoRek);

Console.Clear();

MasterMenuUser.User(NoRek);

break;

case 2:

Console.Clear();

ProgramUser.Transfer(NoRek);

Console.Clear();

MasterMenuUser.User(NoRek);

break;

case 3:

Console.Clear();

ProgramUser.CashWithdraw(NoRek);

Console.Clear();

MasterMenuUser.User(NoRek);

break;

case 4:

Console.Clear();

ProgramUser.CashDeposit(NoRek);

Console.Clear();

MasterMenuUser.User(NoRek);

break;

case 5:

Console.Clear();

ProgramUser.Mutation(NoRek);

Console.Clear();

MasterMenuUser.User(NoRek);

break;

case 6:

Console.Clear();

ProgramUser.ChangePIN(NoRek);

MasterMenuUser.User(NoRek);

break;

**CODES (MASTER MENU USER)**

case 7:

Environment.Exit(0);

break;

default:

Console.WriteLine("\nInvalid Choice");

Console.ReadKey();

Console.Clear();

MasterMenuUser.User(NoRek);

break;

}

Console.ReadKey();

Console.Clear();

}

}

}

**CODES (PROGRAM ADMIN)**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

using System.Text.RegularExpressions;

namespace Wokeh

{

class ProgramAdmin

{

public static void CreateUserData()

{

string KTP = "", Name = "", Address = "", PlaceOfBirth = "", UserName = "",

Password = "", Email = "", HandphoneN = "", Employment = "",

Gender = "", AccountType = "";

string NoKartu = "";

Random rnd = new Random();

Regex rgxstr = new Regex("^[a-zA-Z ]+$");

Regex rgxint = new Regex("^[0-9]+$");

Regex rgxmail = new Regex(@"^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-

Z]{2,4}$");

Regex rgxpass = new Regex("^[A-Za-z0-9]+$");

int AccountN = 0;

long Balance = 50000, PIN = 123456, FBalance = 50000;

string choice = "n";

while (choice == "n" || choice == "N")

{

Console.Clear();

var dbUser = File.ReadAllLines("UserData.txt");

bool i = true;

while (i)

{

AccountN = rnd.Next(10000000, 99999999);

foreach (string data in dbUser)

{

data.Split('#');

if (AccountN.Equals(data[12]))

{

i = true;

break;

}

else

{

i = false;

continue;

}

}

}

**CODES (PROGRAM ADMIN)**

bool v = true;

while (v)

{

Console.Clear();

Console.WriteLine("Waiting for card swipe...");

NoKartu = Console.ReadLine();

foreach (string data in dbUser)

{

string[] ary = data.Split('#');

if (NoKartu.Equals(ary[16]))

{

v = true;

Console.WriteLine("Duplicate card detected");

Console.ReadKey();

break;

}

else if (!NoKartu.Equals(ary[16]))

{

v = false;

}

}

}

bool x = true;

while (x)

{

Console.Clear();

Console.WriteLine("Enter your Number of KTP :");

KTP = Console.ReadLine();

foreach (string data in dbUser)

{

string[] ary = data.Split('#');

if (KTP.Length == 0)

{

x = true;

Console.WriteLine("Your Number Of KTP is null, Please Enter

Again");

Console.ReadKey();

Console.Clear();

ProgramAdmin.CreateUserData();

}

else if (KTP.Length < 10)

{

x = true;

Console.WriteLine("Number of KTP Must 10 Digit, please Enter

Again");

Console.ReadKey();

Console.Clear();

ProgramAdmin.CreateUserData();

}

else if (ary[0].Equals(KTP))

{

x = true;

Console.WriteLine("Nomor KTP already used, Please Enter

Again");

Console.ReadKey();

ProgramAdmin.CreateUserData();

}

else if (rgxint.IsMatch(KTP))

{

x = false;

}

else

{

x = true;

}

}

}

**CODES (PROGRAM ADMIN)**

bool k = true;

while (k)

{

Console.Write("Enter Your Name :\n");

Name = Console.ReadLine();

if (Name.Length == 0)

{

k = true;

Console.WriteLine("Your Name is Null, Please Enter Your Name

Again ");

}

else if (rgxstr.IsMatch(Name))

{

k = false;

}

else

{

k = true;

}

}

bool b = true;

while (b)

{

Console.WriteLine("Enter Your Gender (M / F):");

Gender = Console.ReadLine();

switch (Gender)

{

case "M":

b = false;

break;

case "F":

b = false;

break;

default:

Console.WriteLine("Your Input not Valid, Please Enter Again

:");

break;

}

}

bool m = true;

while (m)

{

Console.WriteLine("Enter Your Address :");

Address = Console.ReadLine();

if (Address.Length == 0)

{

m = true;

Console.WriteLine("Your Address is Null, Please Enter Your

Address Again");

}

else

{

m = false;

}

}

**CODES (PROGRAM ADMIN)**

bool n = true;

while (n)

{

Console.WriteLine("Enter Place of Birth :");

PlaceOfBirth = Console.ReadLine();

if (PlaceOfBirth.Length == 0)

{

n = true;

Console.WriteLine("Your Place of Birth is Null, Please Enter

Again");

}

else if (rgxstr.IsMatch(PlaceOfBirth))

{

n = false;

}

else

{

n = true;

}

}

bool y = true;

while (y)

{

Console.WriteLine("Enter E-Mail :");

Email = Console.ReadLine();

foreach (string data in dbUser)

{

string[] ary = data.Split('#');

if (Email.Length == 0)

{

y = true;

Console.WriteLine("Your e-Mail is Null, Please Enter Again");

break;

}

else if (ary[5].Equals(Email))

{

y = true;

Console.WriteLine("E-Mail already used, Please Enter Again");

}

else if (rgxmail.IsMatch(Email))

{

y = false;

}

else

{

y = true;

}

}

}

**CODES (PROGRAM ADMIN)**

bool p = true;

while (p)

{

Console.WriteLine("Enter your Number of Phone :");

HandphoneN = Console.ReadLine();

foreach (string data in dbUser)

{

string[] ary = data.Split('#');

if (HandphoneN.Length == 0)

{

p = true;

Console.WriteLine("Your Number of Phone is Null, Please Enter

Again");

break;

}

else if (ary[6].Equals(HandphoneN))

{

p = true;

Console.WriteLine("Number Phone already used, Please Enter

Again");

}

else if (rgxint.IsMatch(HandphoneN))

{

p = false;

}

else

{

p = true;

}

}

}

bool t = true;

while (t)

{

Console.WriteLine("Enter your Employment :");

Employment = Console.ReadLine();

if (Employment.Length == 0)

{

t = true;

Console.WriteLine("Your Employment is null, Please Enter Again

:");

}

else if (rgxstr.IsMatch(Employment))

{

t = false;

}

else

{

t = true;

}

}

**CODES (PROGRAM ADMIN)**

bool h = true;

while (h)

{

Console.WriteLine("Enter UserName :");

UserName = Console.ReadLine();

foreach (string data in dbUser)

{

string[] ary = data.Split('#');

if (UserName.Length == 0)

{

h = true;

Console.WriteLine("UserName is null, Please Enter Again :");

break;

}

else if (ary[8].Equals(UserName))

{

h = true;

Console.WriteLine("UserName already used");

}

else if (rgxpass.IsMatch(UserName))

{

h = false;

}

}

}

bool w = true;

while (w)

{

Console.WriteLine("Enter Password :");

Password = Console.ReadLine();

if (Password.Length == 0)

{

w = true;

Console.WriteLine("Password is null, Please Enter Again :");

}

else if (rgxpass.IsMatch(Password))

{

w = false;

}

else

{

w = true;

}

}

**CODES (PROGRAM ADMIN)**

bool q = true;

while (q)

{

Console.WriteLine("Enter Your Account Type (Silver / Gold /

Premium):");

AccountType = Console.ReadLine();

switch (AccountType)

{

case "Silver":

q = false;

break;

case "Gold":

q = false;

break;

case "Premium":

q = false;

break;

default:

Console.WriteLine("Your Input not Valid, Please Enter Again

:");

break;

}

}

Console.Clear();

Console.WriteLine("=================================================================");

Console.WriteLine("Number KTP\t:\t {0}", KTP);

Console.WriteLine("Name\t\t:\t {0}", Name);

Console.WriteLine("Gender\t\t:\t {0}", Gender);

Console.WriteLine("Address\t\t:\t {0}", Address);

Console.WriteLine("Place of Birth\t:\t {0}", PlaceOfBirth);

Console.WriteLine("E-Mail\t\t:\t {0}", Email);

Console.WriteLine("Handphone Number:\t {0}", HandphoneN);

Console.WriteLine("Employment\t:\t {0}", Employment);

Console.WriteLine("\nUserName\t:\t {0}", UserName);

Console.WriteLine("Password\t:\t {0}", Password);

Console.WriteLine("\nAccount Type\t:\t {0}", AccountType);

Console.WriteLine("Account Number\t:\t {0}", AccountN);

Console.WriteLine("=================================================================\n\n");

Console.WriteLine("Are data above correct (Y/N)");

choice = Console.ReadLine();

Console.Clear();

}

FileStream fs = new FileStream("UserData.txt", FileMode.Append,

FileAccess.Write);

StreamWriter sw = new StreamWriter(fs);

sw.WriteLine(KTP + "#" + Name + "#" + Gender + "#" + Address + "#" +

PlaceOfBirth + "#" + Email + "#" + HandphoneN +

"#" + Employment + "#" + UserName + "#" + Password + "#" + AccountType +

"#" + AccountN + "#" + PIN + "#" + Balance + "#" + FBalance + "#" +

"user"+"#"+NoKartu);

sw.Flush();

sw.Close();

fs.Close();

Console.WriteLine("Input Again Y/N :");

choice = Console.ReadLine();

}

Console.Clear();

}

**CODES (PROGRAM ADMIN)**

public static void DisplayUserData()

{

Console.Clear();

FileStream fs = new FileStream("UserData.txt", FileMode.Open,

FileAccess.Read);

StreamReader sr = new StreamReader(fs);

string data = sr.ReadLine();

while (data != null)

{

string[] ary = data.Split('#');

Console.WriteLine("======================================================

===========");

Console.WriteLine("Number KTP\t:\t {0}", ary[0]);

Console.WriteLine("Name\t\t:\t {0}", ary[1]);

Console.WriteLine("Gender\t\t:\t {0}", ary[2]);

Console.WriteLine("Address\t\t:\t {0}", ary[3]);

Console.WriteLine("Place of Birth\t:\t {0}", ary[4]);

Console.WriteLine("E-Mail\t\t:\t {0}", ary[5]);

Console.WriteLine("Handphone Number:\t {0}", ary[6]);

Console.WriteLine("Employment\t:\t {0}", ary[7]);

Console.WriteLine("\nUserName\t:\t {0}", ary[8]);

Console.WriteLine("Password\t:\t {0}", ary[9]);

Console.WriteLine("\nAccount Type\t:\t {0}", ary[10]);

Console.WriteLine("Account Number\t:\t {0}", ary[11]);

Console.WriteLine("PIN\t\t:\t {0}", ary[12]);

Console.WriteLine("\nBalance\t\t:\t {0}", ary[13]);

Console.WriteLine("As\t\t:\t {0}", ary[15]);

Console.WriteLine("======================================================

=========\n\n");

data = sr.ReadLine();

}

sr.Close();

fs.Close();

Console.ReadKey();

Console.Clear();

}

**CODES (PROGRAM ADMIN)**

static string path = @"D:\Wokeh!! - fix banget\Wokeh!!\bin\Debug\";

public static void SearchUserData()

{

Console.Clear();

string keyword;

var ary = new object[20];

Console.WriteLine("Enter Keyword (KTP) : ");

keyword = Console.ReadLine();

foreach (string data in File.ReadLines(path+"UserData.txt"))

{

ary = data.Split('#');

if (keyword.Equals(Convert.ToString(ary[0])))

{

Console.WriteLine("=================== DATA IS AVALIABLE

===================");

Console.WriteLine("Number KTP\t:\t {0}", ary[0]);

Console.WriteLine("Name\t\t:\t {0}", ary[1]);

Console.WriteLine("Gender\t\t:\t {0}", ary[2]);

Console.WriteLine("Address\t\t:\t {0}", ary[3]);

Console.WriteLine("Place of Birth\t:\t {0}", ary[4]);

Console.WriteLine("E-Mail\t\t:\t {0}", ary[5]);

Console.WriteLine("Handphone Number:\t {0}", ary[6]);

Console.WriteLine("Employment\t:\t {0}", ary[7]);

Console.WriteLine("\nUserName\t:\t {0}", ary[8]);

Console.WriteLine("Password\t:\t {0}", ary[9]);

Console.WriteLine("\nAccount Type\t:\t {0}", ary[10]);

Console.WriteLine("Account Number\t:\t {0}", ary[11]);

Console.WriteLine("PIN\t\t:\t {0}", ary[12]);

Console.WriteLine("\nBalance\t\t:\t {0}", ary[13]);

Console.WriteLine("As\t\t:\t {0}", ary[15]);

Console.WriteLine("Number Card\t:\t {0}", ary[16]);

Console.WriteLine("=================================================================");

Console.WriteLine("Press any key to continue");

Console.ReadKey();

break;

}

}

}

**CODES (PROGRAM ADMIN)**

public static void UpdateUserData()

{

Console.Clear();

StringBuilder newFile = new StringBuilder();

var dUser = File.ReadAllLines("UserData.txt");

string keyword;

int choice;

Regex rgxstr = new Regex("^[a-zA-Z ]+$");

Regex rgxint = new Regex("^[0-9]+$");

Regex rgxmail = new Regex(@"^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$");

Regex rgxpass = new Regex("^[A-Za-z0-9]+$");

bool g = true;

Console.WriteLine("Enter Keyword ( KTP ):");

keyword = Console.ReadLine();

if (keyword.Length == 0)

{

Console.WriteLine("Keyword is null, please enter again");

Console.ReadLine();

ProgramAdmin.UpdateUserData();

}

foreach (string data in dUser)

{

if (data.Contains(keyword) && rgxint.IsMatch(keyword))

{

var result = data.Split('#');

string temp = "";

string KTP = "", Name = "", Address = "", PlaceOfBirth = "",

UserName = "", Password = "", Email = "",

HandphoneN = "", AccountType = "", Employment = "", Gender = "";

Console.Clear();

Console.WriteLine("=================== DATA IS AVALIABLE

===================");

Console.WriteLine("Number KTP\t:\t {0}", result[0]);

Console.WriteLine("Name\t\t:\t {0}", result[1]);

Console.WriteLine("Gender\t\t:\t {0}", result[2]);

Console.WriteLine("Address\t\t:\t {0}", result[3]);

Console.WriteLine("Place of Birth\t:\t {0}", result[4]);

Console.WriteLine("E-Mail\t\t:\t {0}", result[5]);

Console.WriteLine("Handphone Number:\t {0}", result[6]);

Console.WriteLine("Employment\t:\t {0}", result[7]);

Console.WriteLine("\nUserName\t:\t {0}", result[8]);

Console.WriteLine("Password\t:\t {0}", result[9]);

Console.WriteLine("\nAccount Type\t:\t {0}", result[10]);

Console.WriteLine("Account Number\t:\t {0}", result[11]);

Console.WriteLine("PIN\t\t:\t {0}", result[12]);

Console.WriteLine("\nBalance\t\t:\t {0}", result[13]);

Console.WriteLine("As\t\t:\t {0}", result[15]);

Console.WriteLine("======================================================

===========\n");

Console.WriteLine("=============== S E L E C T N U M B E R ===============");

Console.WriteLine("1. Update Number KTP");

Console.WriteLine("2. Update Name");

Console.WriteLine("3. Update Gender");

Console.WriteLine("4. Update Address");

Console.WriteLine("5. Update Place of Birth");

Console.WriteLine("6. Update e-Mail");

Console.WriteLine("7. Update Phone Number");

Console.WriteLine("8. Update Employment");

Console.WriteLine("9. Update User Name");

Console.WriteLine("10. Update Password");

Console.WriteLine("11. Update Account Type");

Console.WriteLine("12. Back to Menu");

Console.WriteLine(" Enter Number : ");

**CODES (PROGRAM ADMIN)**

choice = int.Parse(Console.ReadLine());

switch (choice)

{

case 1:

while (g)

{

Console.WriteLine("Old Data : {0}", result[0]);

Console.WriteLine("Enter New Number : ");

KTP = Console.ReadLine();

foreach (string data1 in dUser)

{

string[] ary = data.Split('#');

if (KTP.Length == 0)

{

g = true;

Console.WriteLine("Your Number Of KTP is null,

Please Enter Again");

}

else if (KTP.Length < 10)

{

g = true;

Console.WriteLine("Number of KTP Must 10 Digit,

please Enter Again");

}

else if (ary[0].Equals(KTP))

{

g = true;

Console.WriteLine("Nomor KTP already used, Please

Enter Again");

}

else if (rgxint.IsMatch(KTP))

{

g = false;

}

else

{

g = true;

}

}

}

temp = data.Replace(result[0], KTP);

newFile.Append(temp + "\r\n");

Console.WriteLine("New Number KTP : {0}", KTP);

Console.ReadKey();

Console.Clear();

continue;

**CODES (PROGRAM ADMIN)**

case 2:

while (g)

{

Console.WriteLine("Old Data : {0}", result[1]);

Console.WriteLine("Enter New Name : ");

Name = Console.ReadLine();

if (Name.Length == 0)

{

g = true;

Console.WriteLine("Your Name is Null, Please Enter

Your Name Again ");

}

else if (rgxstr.IsMatch(Name))

{

g = false;

}

else

{

g = true;

Console.WriteLine("Your name is not valid");

}

}

temp = data.Replace(result[1], Name);

newFile.Append(temp + "\r\n");

Console.WriteLine("New Name : {0}", Name);

Console.ReadKey();

Console.Clear();

continue;

case 3:

while (g)

{

Console.WriteLine("Old Data : {0}", result[2]);

Console.WriteLine("Enter New Gender ( M / F ) : ");

Gender = Console.ReadLine();

switch (Gender)

{

case "M":

g = false;

break;

case "F":

g = false;

break;

default:

Console.WriteLine("Your Input not Valid, Please

Enter Again :");

break;

}

}

temp = data.Replace(result[2], Gender);

newFile.Append(temp + "\r\n");

Console.WriteLine("New Gender : {0}", Gender);

Console.ReadKey();

Console.Clear();

continue;

**CODES (PROGRAM ADMIN)**

case 4:

while (g)

{

Console.WriteLine("Old Data : {0}", result[3]);

Console.WriteLine("Enter New Address : ");

Address = Console.ReadLine();

if (Address.Length == 0)

{

g = true;

Console.WriteLine("Your Address is Null, Please Enter

Your Address Again");

}

else

{

g = false;

}

}

temp = data.Replace(result[3], Address);

newFile.Append(temp + "\r\n");

Console.WriteLine("New Address : {0}", Address);

Console.ReadKey();

Console.Clear();

continue;

case 5:

while (g)

{

Console.WriteLine("Old Data : {0}", result[4]);

Console.WriteLine("Enter New Place Of Birth : ");

PlaceOfBirth = Console.ReadLine();

if (PlaceOfBirth.Length == 0)

{

g = true;

Console.WriteLine("Your Place of Birth is Null,

Please Enter Again");

}

else if (rgxstr.IsMatch(PlaceOfBirth))

{

g = false;

}

else

{

g = true;

Console.WriteLine("Your Place of Birth is not

valid");

}

}

temp = data.Replace(result[4], PlaceOfBirth);

newFile.Append(temp + "\r\n");

Console.WriteLine("New Place Of Birth : {0}", PlaceOfBirth);

Console.ReadKey();

Console.Clear();

continue;

**CODES (PROGRAM ADMIN)**

case 6:

while (g)

{

Console.WriteLine("Old Data : {0}", result[5]);

Console.WriteLine("Enter New e-Mail : ");

Email = Console.ReadLine();

foreach (string data1 in dUser)

{

string[] ary = data.Split('#');

if (Email.Length == 0)

{

g = true;

Console.WriteLine("Your e-Mail is Null, Please

Enter Again");

break;

}

else if (ary[5].Equals(Email))

{

g = true;

Console.WriteLine("E-Mail already used, Please

Enter Again");

}

else if (rgxmail.IsMatch(Email))

{

g = false;

}

else

{

g = true;

Console.WriteLine("Your e-Mail is not valid");

}

}

}

temp = data.Replace(result[5], Email);

newFile.Append(temp + "\r\n");

Console.WriteLine("New e-Mail : {0}", Email);

Console.ReadKey();

Console.Clear();

continue;

**CODES (PROGRAM ADMIN)**

case 7:

while (g)

{

Console.WriteLine("Old Data : {0}", result[6]);

Console.WriteLine("Enter New Number Phone : ");

HandphoneN = Console.ReadLine();

foreach (string data1 in dUser)

{

string[] ary = data.Split('#');

if (HandphoneN.Length == 0)

{

g = true;

Console.WriteLine("Your Number of Phone is Null,

Please Enter Again");

break;

}

else if (ary[6].Equals(HandphoneN))

{

g = true;

Console.WriteLine("Number Phone already used,

Please Enter Again");

}

else if (rgxint.IsMatch(HandphoneN))

{

g = false;

}

else

{

g = true;

Console.WriteLine("Your Phone Number is not

valid");

}

}

}

temp = data.Replace(result[6], HandphoneN);

newFile.Append(temp + "\r\n");

Console.WriteLine("New Number Phone : {0}", HandphoneN);

Console.ReadKey();

Console.Clear();

continue;

**CODES (PROGRAM ADMIN)**

case 8:

while (g)

{

Console.WriteLine("Old Data : {0}", result[7]);

Console.WriteLine("Enter New Employment : ");

Employment = Console.ReadLine();

if (Employment.Length == 0)

{

g = true;

Console.WriteLine("Your Employment is null, Please

Enter Again :");

}

else if (rgxstr.IsMatch(Employment))

{

g = false;

}

else

{

g = true;

Console.WriteLine("Your Employment is not valid");

}

}

temp = data.Replace(result[7], Employment);

newFile.Append(temp + "\r\n");

Console.WriteLine("New Employment : {0}", Employment);

Console.ReadKey();

Console.Clear();

continue;

case 9:

while (g)

{

Console.WriteLine("Old Data : {0}", result[8]);

Console.WriteLine("Enter New User Name : ");

UserName = Console.ReadLine();

foreach (string data1 in dUser)

{

string[] ary = data1.Split('#');

if (UserName.Length == 0)

{

g = true;

Console.WriteLine("UserName is null, Please Enter

Again :");

break;

}

else if (ary[8].Equals(UserName))

{

g = true;

Console.WriteLine("UserName already used");

}

else if (rgxpass.IsMatch(UserName))

{

g = false;

}

else

{

g = true;

Console.WriteLine("Your UserName is not valid");

}

}

}

temp = data.Replace(result[8], UserName);

newFile.Append(temp + "\r\n");

Console.WriteLine("New User Name : {0}", UserName);

Console.ReadKey();

Console.Clear();

continue;

**CODES (PROGRAM ADMIN)**

case 10:

while (g)

{

Console.WriteLine("Old Data : {0}", result[9]);

Console.WriteLine("Enter New Password : ");

Password = Console.ReadLine();

if (Password.Length == 0)

{

g = true;

Console.WriteLine("Password is null, Please Enter

Again :");

}

else if (rgxpass.IsMatch(Password))

{

g = false;

}

else

{

g = true;

Console.WriteLine("Your Password is not valid");

}

}

temp = data.Replace(result[9], Password);

newFile.Append(temp + "\r\n");

Console.WriteLine("New Password : {0}", Password);

Console.ReadKey();

Console.Clear();

continue;

case 11:

while (g)

{

Console.WriteLine("Old Data : {0}", result[10]);

Console.WriteLine("Enter New Account Type ( Silver / Gold

/ Premium ) : ");

AccountType = Console.ReadLine();

switch (AccountType)

{

case "Silver":

g = false;

break;

case "Gold":

g = false;

break;

case "Premium":

g = false;

break;

default:

Console.WriteLine("Your Input not Valid, Please

Enter Again :");

break;

}

}

temp = data.Replace(result[10], AccountType);

newFile.Append(temp + "\r\n");

Console.WriteLine("New Type Account : {0}", AccountType);

Console.ReadKey();

Console.Clear();

continue;

**CODES (PROGRAM ADMIN)**

case 12:

MasterMenuAdmin.Admin();

break;

default:

Console.WriteLine("Invalid Number");

Console.ReadKey();

break;

}

}

newFile.Append(data + "\r\n");

}

File.WriteAllText(@"UserData.txt", newFile.ToString());

}

public static void DeleteLines(string strLineToDelete)

{

string strSearchText = strLineToDelete;

string strOldText;

string a = "";

StreamReader sr = File.OpenText("UserData.txt");

while ((strOldText = sr.ReadLine()) != null)

{

if (!strOldText.Contains(strSearchText))

{

a += strOldText + Environment.NewLine;

}

}

sr.Close();

File.WriteAllText("UserData.txt", a);

}

**CODES (PROGRAM ADMIN)**

public static void SearchDataUserForDelete()

{

Console.Clear();

var dbUser = File.ReadAllLines("UserData.txt");

string keyword;

Regex rgxint = new Regex("^[0-9]+$");

Console.WriteLine("Enter Keyword (KTP) : ");

keyword = Console.ReadLine();

foreach (string data in dbUser)

{

if (data.Contains(keyword) && rgxint.IsMatch(keyword))

{

string[] ary = data.Split('#');

Console.WriteLine("=================== DATA IS AVALIABLE

===================");

Console.WriteLine("Number KTP\t:\t {0}", ary[0]);

Console.WriteLine("Name\t\t:\t {0}", ary[1]);

Console.WriteLine("Gender\t\t:\t {0}", ary[2]);

Console.WriteLine("Address\t\t:\t {0}", ary[3]);

Console.WriteLine("Place of Birth\t:\t {0}", ary[4]);

Console.WriteLine("E-Mail\t\t:\t {0}", ary[5]);

Console.WriteLine("Handphone Number:\t {0}", ary[6]);

Console.WriteLine("Employment\t:\t {0}", ary[7]);

Console.WriteLine("\nUserName\t:\t {0}", ary[8]);

Console.WriteLine("Password\t:\t {0}", ary[9]);

Console.WriteLine("\nAccount Type\t:\t {0}", ary[10]);

Console.WriteLine("Account Number\t:\t {0}", ary[11]);

Console.WriteLine("PIN\t\t:\t {0}", ary[12]);

Console.WriteLine("\nBalance\t\t:\t {0}", ary[13]);

Console.WriteLine("As\t\t:\t {0}", ary[15]);

Console.WriteLine("=============================================

====================\n\n");

string choice;

Console.WriteLine("Are you sure you want to delete this User?

(Y/N)");

choice = Console.ReadLine();

if ((choice == "Y") || (choice == "y"))

{

ProgramAdmin.DeleteLines(keyword);

}

else

{

Console.WriteLine("Press any key to continue");

Console.Clear();

MasterMenuAdmin.Admin();

}

}

}

}

}

}

**CODES (PROGRAM USER)**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

using ConsoleTables;

using System.Data;

using System.Threading;

using System.Text.RegularExpressions;

namespace Wokeh

{

class ProgramUser

{

static string path = @"D:\Wokeh!! - fix banget\Wokeh!!\bin\Debug\";

private static string TID;

public static void GetTID(int b)

{

int NoRek = b;

int count = 1;

if (File.Exists(path + "mutasi-"+NoRek+".txt"))

{

var dbTransaction = File.ReadAllLines(path + "mutasi-" + NoRek + ".txt");

Array.Reverse(dbTransaction);

foreach (string line in dbTransaction)

{

var data = line.Split(',');

count = int.Parse(data[1].Substring(data[1].Length - 4)) + 1;

break;

}

}

if (count < 10)

{

TID = "TRX000" + Convert.ToString(count);

}

else if (count < 100)

{

TID = "TRX00" + Convert.ToString(count);

}

else if (count < 1000)

{

TID = "TRX0" + Convert.ToString(count);

}

else if (count < 10000)

{

TID = "TRX" + Convert.ToString(count);

}

else

{

TID = "TID Limited";

TID = null;

}

}

**CODES (PROGRAM USER)**

public static void CashDeposit(int a)

{

Console.Clear();

int NoRek = a;

var dbUserData = File.ReadAllLines(path + "UserData.txt");

int packet;

double nominal = 0;

bool chk;

do

{

chk = false;

Console.WriteLine("////////////////////////////////////////////////////////////////////////////////////////////////////////////////////");

Console.WriteLine("\t//\t\t\t\t SELECT THE NOMIMAL OF PACKAGE DEPOSITS

\t\t\t\t//");

Console.WriteLine("\t//\t\t\t\t\t\t\t\t\t\t\t\t\t//");

Console.WriteLine("\t//\t\t\t 1 <= 50.000 \t\t\t 6 <= 750.000

\t\t\t\t\t//");

Console.WriteLine("\t//\t\t\t 2 <= 100.000 \t\t\t 7 <= 1.000.000

\t\t\t\t//");

Console.WriteLine("\t//\t\t\t 3 <= 150.000 \t\t\t 8 <= 1.500.000

\t\t\t\t//");

Console.WriteLine("\t//\t\t\t 4 <= 250.000 \t\t\t 9 <= 2.000.000

\t\t\t\t//");

Console.WriteLine("\t//\t\t\t 5 <= 500.000 \t\t\t 10 <= Adjust

Nominal\t\t\t\t//");

Console.WriteLine("////////////////////////////////////////////////////////////////////////////////////////////////////////////////////");

packet = Convert.ToInt32(Console.ReadLine());

switch (packet)

{

default:

Console.WriteLine("Please input 1-10");

Console.ReadKey();

Console.Clear();

CashDeposit(NoRek);

break;

case 1:

nominal = 50000;

break;

case 2:

nominal = 100000;

break;

case 3:

nominal = 150000;

break;

case 4:

nominal = 250000;

break;

case 5:

nominal = 500000;

break;

case 6:

nominal = 750000;

break;

case 7:

nominal = 1000000;

break;

**CODES (PROGRAM USER)**

case 8:

nominal = 1500000;

break;

case 9:

nominal = 2000000;

break;

case 10:

do

{

Console.WriteLine("Please Enter The Nominal");

nominal = Convert.ToDouble(Console.ReadLine());

chk = false;

if (((nominal % 1000) != 0) || (nominal >

5000000) || nominal < 10000)

{

Console.WriteLine("Please enter nominal

multiples of Rp1.000 AND minimum nominal

is Rp10.000 AND maximum nominal is

Rp5.000.000");

Console.ReadKey();

chk = true;

}

} while (chk);

break;

case 11:

MasterMenuUser.User(NoRek);

Console.Clear();

break;

}

ScanMoney(nominal,NoRek);

} while (chk);

}

**CODES (PROGRAM USER)**

public static void CashWithdraw(int b)

{

int NoRek = b;

Console.Clear();

GetTID(NoRek);

StringBuilder newfile = new StringBuilder();

FileStream fs = new FileStream(path + "mutasi-"+NoRek+".txt.",

FileMode.Append, FileAccess.Write);

StreamWriter sw = new StreamWriter(fs);

var dbUserData = File.ReadAllLines(path + "UserData.txt");

int packet;

double nominal = 0;

double remain = 0;

bool chk;

do

{

chk = false;

Console.WriteLine("////////////////////////////////////////////////////////////////////////////////////////////////////////////////////");

Console.WriteLine("\t//\t\t\t\t SELECT THE NUMBER OF CASH PACKETS

\t\t\t\t\t\t//");

Console.WriteLine("\t//\t\t\t\t\t\t\t\t\t\t\t\t\t//");

Console.WriteLine("\t//\t\t\t 1 <= 50.000 \t\t\t 6 <= 750.000

\t\t\t\t\t//");

Console.WriteLine("\t//\t\t\t 2 <= 100.000 \t\t\t 7 <= 1.000.000

\t\t\t\t//");

Console.WriteLine("\t//\t\t\t 3 <= 150.000 \t\t\t 8 <= 1.500.000

\t\t\t\t//");

Console.WriteLine("\t//\t\t\t 4 <= 250.000 \t\t\t 9 <= 2.000.000

\t\t\t\t//");

Console.WriteLine("\t//\t\t\t 5 <= 500.000 \t\t\t 10 <=

Random\t\t\t\t//");

Console.WriteLine("////////////////////////////////////////////////////////////////////////////////////////////////////////////////////");

packet = Convert.ToInt32(Console.ReadLine());

switch (packet)

{

default:

Console.WriteLine("Please input 1-10");

Console.ReadKey();

Console.Clear();

CashWithdraw(NoRek);

break;

case 1:

nominal = 50000;

break;

case 2:

nominal = 100000;

break;

case 3:

nominal = 150000;

break;

case 4:

nominal = 250000;

break;

case 5:

nominal = 500000;

break;

case 6:

nominal = 750000;

break;

**CODES (PROGRAM USER)**

case 7:

nominal = 1000000;

break;

case 8:

nominal = 1500000;

break;

case 9:

nominal = 2000000;

break;

case 10:

do

{

Console.WriteLine("Please Enter The Nominal");

nominal = Convert.ToDouble(Console.ReadLine());

chk = false;

if (((nominal % 50000) != 0) || (nominal > 5000000) ||

(nominal < 50000))

{

Console.WriteLine("Please enter nominal multiples of

Rp50.000 AND minimum nominal is Rp50.000 AND maximum

nominal is Rp5.000.000");

Console.ReadKey();

chk = true;

}

foreach (string line in dbUserData)

{

var data = line.Split('#');

if (long.Parse(data[11]).Equals(NoRek))

{

remain = Convert.ToDouble(data[13]) - nominal;

if (remain < Convert.ToDouble(data[14]))

{

Console.WriteLine("Your saldo is Rp{0} \nPlease

Leave Rp{1} balance", data[13]);

Console.ReadKey();

chk = true;

break;

}

}

}

} while (chk);

break;

case 11:

MasterMenuUser.User(NoRek);

Console.Clear();

break;

}

foreach (string line in dbUserData)

{

var data = line.Split('#');

string temp = "";

if (long.Parse(data[11]).Equals(NoRek))

{

remain = Convert.ToDouble(data[13]) - nominal;

if (remain < Convert.ToDouble(data[14]))

{

Console.WriteLine("Your saldo is Rp{0} \nPlease Leave Rp{1}

balance", data[13]);

Console.ReadKey();

chk = true;

CashWithdraw(NoRek);

break;

}

**CODES (PROGRAM USER)**

else

{

temp = line.Replace(Convert.ToString(data[13]),

Convert.ToString(remain));

newfile.Append(temp + "\r\n");

continue;

}

}

newfile.Append(line + "\r\n");

}

File.WriteAllText(@"UserData.txt", newfile.ToString());

sw.WriteLine(DateTime.Now + "," + TID + "," + NoRek + "," + NoRek +"," +

"K" + "," + nominal + "," + remain + "," + "Cash Withdraw");

sw.Flush();

sw.Close();

fs.Close();

} while (chk);

Console.Clear();

Console.WriteLine("\n\t\t\t\t\t~~~~~~~~ Transaction Success ~~~~~~~~");

Console.WriteLine("\t\t\t\t\t~~~~~~~~ ~~~~~~~~");

Console.WriteLine("\t\t\t\t\t~~~~~~~~ Please Take The Money ~~~~~~~~");

Console.ReadKey();

}

**CODES (PROGRAM USER)**

private static void ScanMoney(double nominals,int c)

{

Console.Clear();

int NoRek = c;

GetTID(NoRek);

StringBuilder newfile = new StringBuilder();

FileStream fs = new FileStream(path + "mutasi-"+NoRek+".txt",

FileMode.Append, FileAccess.Write);

StreamWriter sw = new StreamWriter(fs);

var dbUserData = File.ReadAllLines(path + "UserData.txt");

var dbMoneyData = File.ReadAllLines(path + "MoneyData.txt");

dynamic[] MID = new dynamic[1000];

MID[0] = "0003456596";

MID[1] = "0003452961";

//string InpMID = "MD5E"; /\*When using RFID\*/

double SnMoney = 0, remain = 0, NewSaldo = 0;

bool chk, chkCD = true;

int j = 0;

do

{

chk = false;

while (chkCD)

{

for (int i = j; i < MID.GetLength(0); i++)

{

Console.Clear();

Console.WriteLine("The amount of money you input Rp{0}",

SnMoney);

bool GotKey = false;

DateTime start = DateTime.Now;

string countdown = "";

int k = 5;

while (((DateTime.Now - start).TotalSeconds < 5) && (SnMoney <

nominals))

{

countdown = String.Format("Waiting for input for {0}

seconds...", k);

Console.Write(countdown);

Thread.Sleep(1000);

for (Int32 l = 0; l < countdown.Length; l++)

{

Console.Write("\b \b"); // backspace - space - backspace

}

if ((Console.KeyAvailable) == true)

{

GotKey = true;

break;

}

k--;

//Console.WriteLine((DateTime.Now - start).TotalSeconds);

}

**CODES (PROGRAM USER)**

if (GotKey)

{

chkCD = true;

string InpMID="";

Console.WriteLine("Waiting for card swipe...");

if (true)

{

InpMID = Console.ReadLine();

}

MID[i] = InpMID;

foreach (string line in dbMoneyData)

{

var data = line.Split('\*');

if (data[0].Equals(Convert.ToString(MID[i])))

{

SnMoney += Convert.ToDouble(data[1]);

}

}

j = i + 1;

}

else

{

if(SnMoney >= nominals)

{

Console.WriteLine("Succes please any key to continue");

Console.ReadKey();

}

Console.WriteLine("Timed out");

chkCD = false;

break;

}

}

}

if (SnMoney < nominals)

{

remain = nominals - SnMoney;

Console.WriteLine("the money you input is still less Rp{0}, please

input the lack of money.", remain);

Console.ReadKey();

chk = true;

chkCD = true;

}

else

{

if (SnMoney > nominals)

{

remain = SnMoney - nominals;

Console.WriteLine("Money you input excess Rp{0}, please take the

money.", remain);

Console.ReadKey();

}

foreach (string line in dbUserData)

{

var data = line.Split('#');

string temp = "";

if (long.Parse(data[11]).Equals(NoRek))

{

NewSaldo = Convert.ToDouble(data[13]) + nominals;

temp = line.Replace(data[13], Convert.ToString(NewSaldo));

newfile.Append(temp + "\r\n");

continue;

}

**CODES (PROGRAM USER)**

newfile.Append(line + "\r\n");

}

File.WriteAllText(@"UserData.txt", newfile.ToString());

sw.WriteLine(DateTime.Now + "," + TID + "," + NoRek + "," + NoRek +

"," + "D" + "," + nominals + "," + NewSaldo + "," + "Cash Deposit");

sw.Flush();

sw.Close();

fs.Close();

Console.Clear();

Console.WriteLine("\n\t\t\t\t\t~~~~~~~~ Transaction Success

~~~~~~~~");

Console.WriteLine("\t\t\t\t\t~~~~~~~~

~~~~~~~~");

Console.WriteLine("\t\t\t\t\t~~~~~~~~ Cash Deposit

~~~~~~~~");

Console.ReadKey();

}

} while (chk);

}

**CODES (PROGRAM USER)**

public static void Mutation(int asd)

{

int NoRek = asd;

Console.Clear();

bool chk = false;

Random rnd = new Random();

int choice = 0, diff = 0;

DateTime now = System.DateTime.Now;

DateTime dateF = System.DateTime.Now;

DateTime dateT = System.DateTime.Now;

Console.WriteLine("\n\t\t\t\t\t~~~~~~~~ Mutation ~~~~~~~~");

Console.WriteLine("\t\t\t\t\t~~~~~~~~ ~~~~~~~~");

Console.WriteLine("\t\t\t\t\t~~~~~~~~ 1 <= Last 30 days ~~~~~~~~");

Console.WriteLine("\t\t\t\t\t~~~~~~~~ 2 <= Last 7 days ~~~~~~~~");

Console.WriteLine("\t\t\t\t\t~~~~~~~~ 3 <= Last today ~~~~~~~~");

Console.WriteLine("\t\t\t\t\t~~~~~~~~ 4 <= Adjust date ~~~~~~~~");

choice = Convert.ToInt32(Console.ReadLine());

switch (choice)

{

default:

Console.WriteLine("Please input 1-4");

Console.ReadKey();

Console.Clear();

Mutation(NoRek);

break;

case 1:

dateF = Convert.ToDateTime((now.AddDays(diff = -30)));

break;

case 2:

dateF = Convert.ToDateTime((now.AddDays(diff = -7)));

break;

case 3:

dateF = Convert.ToDateTime((now.AddDays(diff = -1)));

break;

case 4:

do

{

try

{

Console.Clear();

Console.WriteLine("Enter date form in the format dd-mm-

yyyy");

Console.WriteLine("Example 02 March 2018 = 02-04-2018");

dateF = DateTime.Parse(Console.ReadLine());

Console.WriteLine("Enter date to in the format dd-mm-yyyy");

Console.WriteLine("Example 02 March 2018 = 02-04-2018");

dateT = DateTime.Parse(Console.ReadLine()).AddHours(23.9999);

diff = (dateT - dateF).Days;

}

catch

{

Console.WriteLine("Your date input does not match the format,

Please Enter The Date Again ");

chk = true;

Console.ReadKey();

}

if (((now - dateF).Days > 30) || ((now - dateT).Days > 30))

{

Console.WriteLine("Please enter a maximum of the last 30

days");

chk = true;

Console.ReadKey();

}

**CODES (PROGRAM USER)**

else if (diff < 0)

{

Console.WriteLine("Input your date reversed, Please Check And

Enter Your The Date Again ");

Math.Abs(diff);

chk = true;

Console.ReadKey();

}

else if ((dateF > DateTime.Now) || (dateT > DateTime.Now))

{

Console.WriteLine("Input your date beyond now, Please Check

And Enter Your The Date Again ");

chk = true;

Console.ReadKey();

}

} while (chk);

break;

case 5:

MasterMenuUser.User(NoRek);

Console.Clear();

break;

}

Console.Clear();

ProgramMain.Emphasis(@"

IOIOIOIOIOIOIOIOIOIOIOIO

IOIOIOIOIOIOIOIOIOIOIOIOIOIO MUTATION REPORTS

IOIOIOIOIOIOIOIOIOIOIOIOIOIO

IOIOIOIOIOIOIOIOIOIOIOIO ");

Console.WriteLine("Date\t: {0} \nTime\t: {1} \nMachine\t: {2} \n",

now.ToShortDateString(), now.ToLongTimeString(), Environment.MachineName +

"#" + rnd.Next(100, 999));

Console.WriteLine("Account Number\t:" + NoRek + "\n");

Console.SetWindowSize(150, 30);

var table = new ConsoleTable("TransID", "Date", "Acc Recipient", "Acc

Sender", "Nominal", "D/C", "Balance", "Information");

foreach (string line in File.ReadLines(path + "mutasi-" + NoRek + ".txt"))

{

var data = line.Split(',');

if (Convert.ToDateTime(data[0]) > dateF && Convert.ToDateTime(data[0]) <

dateT)

{

table.AddRow(data[1], data[0], data[3], data[2], data[5], data[4],

data[6], data[7]);

}

}

table.Write();

Console.WriteLine();

Console.ReadKey();

}

**CODES (PROGRAM USER)**

public static void CheckBalance(int asd)

{

int NoRek = asd;

var data = new dynamic[20];

foreach (string lines in File.ReadLines(path + "UserData.txt"))

{

data = lines.Split('#');

if (NoRek.Equals(Convert.ToInt32(data[11])))

{

Console.WriteLine("#####################################");

Console.WriteLine("Account Number\t: " + data[11]);

Console.WriteLine("Account Type\t: " + data[10]);

Console.WriteLine("Balance\t\t: " + data[13]);

Console.WriteLine("Time \t\t: " + DateTime.Now);

Console.WriteLine("#####################################");

Console.ReadKey();

}

}

}

public static void ChangePIN(int asd)

{

var data = new object[20];

var data\_input = new object[4];

int rek\_pengirim = asd;

int linenumber = 0;

string[] line\_detect;

string[] line\_specific;

foreach (string line in File.ReadLines(path + "UserData.txt"))

{

data = line.Split('#');

linenumber++;

if (rek\_pengirim.Equals(Convert.ToInt32(data[11])))

{

break;

}

}

Console.WriteLine("Enter Old PIN");

data\_input[0] = Console.ReadLine();

Console.WriteLine("Enter New PIN");

data\_input[1] = Console.ReadLine();

Console.WriteLine("Confirm New PIN");

data\_input[2] = Console.ReadLine();

if (true)

{

if (!data\_input[0].Equals(data[12]))

{

Console.WriteLine("Wrong Old PIN");

Console.ReadKey();

Console.Clear();

ChangePIN(rek\_pengirim);

}

else

{

if (!data\_input[1].Equals(data\_input[2]))

{

Console.WriteLine("Confirmation PIN Not Valid");

Console.ReadKey();

Console.Clear();

ChangePIN(rek\_pengirim);

**CODES (PROGRAM USER)**

else

{

line\_detect = File.ReadAllLines(path + "UserData.txt");

line\_specific = line\_detect[linenumber - 1].Split('#');

File.WriteAllText(path + "UserData.txt", File.ReadAllText(path +

"UserData.txt").Replace(line\_detect[linenumber - 1],

line\_detect[linenumber - 1].Replace(line\_specific[12],

Convert.ToString(data\_input[1]))));

Console.WriteLine("Change PIN Successfuly, Please press any key

to continue");

Console.ReadKey();

Console.Clear();

MasterMenuUser.User(rek\_pengirim);

}

}

}

}

**CODES (PROGRAM USER)**

public static void Transfer(int asd)

{

int NoRek = asd;

Console.WriteLine("1. Transfer Between Account");

Console.WriteLine("2. Transfer Between Bank ");

Console.WriteLine("3. Back to Menu");

Console.WriteLine();

Console.Write("Input : ");

int choice = Convert.ToInt32(Console.ReadLine());

switch (choice)

{

default:

Console.WriteLine("Please input 1-3");

break;

case 1:

TransferRek(NoRek);

break;

case 2:

TransferBank(NoRek);

break;

case 3:

MasterMenuUser.User(NoRek);

break;

}

}

public static void TransferRek(int asd)

{

int num = new Random().Next(1000, 99999);

string[] line\_detect;

string[] line\_specific;

int linenumber;

int rek\_pengirim = asd;

var data = new object[20];

var data\_trf = new object[5];

var rmv\_empty\_line = File.ReadAllLines(path + "UserData.txt").Where(arg =>

!string.IsNullOrWhiteSpace(arg));

File.WriteAllLines(path + "UserData.txt", rmv\_empty\_line);

bool a = true;

while (a)

{

linenumber = 0;

Console.WriteLine("Enter Account Destination :");

data\_trf[0] = Console.ReadLine();

Console.WriteLine("Enter Total Transfer");

data\_trf[1] = Console.ReadLine();

Console.WriteLine("Enter Description");

data\_trf[2] = Console.ReadLine();

foreach (string line in File.ReadLines(path + "UserData.txt"))

{

data = line.Split('#');

linenumber++;

if (data\_trf[0].Equals(data[11]))

{

break;

}

}

**CODES (PROGRAM USER)**

Console.Clear();

Console.WriteLine("Account Destination : " + data\_trf[0]);

Console.WriteLine("Account Name Destination : " + data[1]);

Console.WriteLine("Total Transfer : " + data\_trf[1]);

Console.WriteLine("Are you sure? (y/n)");

Console.Write("Enter : ");

string input = Console.ReadLine().ToLower();

if (input == "y")

{

if (true)

{

foreach (string line in File.ReadLines(path + "UserData.txt"))

{

data = line.Split('#');

if (rek\_pengirim.Equals(Convert.ToInt32(data[11])))

{

break;

}

else

{

continue;

}

}

if (Convert.ToInt32(data\_trf[1]) > Convert.ToInt32(data[13]))

{

Console.Clear();

Console.WriteLine("Transaction Failed!!");

Console.WriteLine("Not Enought Balance");

}

else

{

foreach (string line in File.ReadLines(path +

"UserData.txt"))

{

data = line.Split('#');

if (data\_trf[0].Equals(data[11]))

{

break;

}

else

{

continue;

}

}

if (!data\_trf[0].Equals(data[11]))

{

Console.Clear();

Console.WriteLine("Transaction Failed!!");

Console.WriteLine("Data not Found");

}

**CODES (PROGRAM USER)**

else

{

Console.Clear();

Console.WriteLine("Transaction success!!");

Console.WriteLine("Ref : " + "TRX" + num);

Console.WriteLine("Transfer Destination : " +

data\_trf[0]);

Console.WriteLine("Total Transfer : " + data\_trf[1]);

Console.WriteLine("Time : " + DateTime.Now);

line\_detect = File.ReadAllLines(path + "UserData.txt");

line\_specific = line\_detect[linenumber - 1].Split('#');

File.WriteAllText(path + "UserData.txt",

File.ReadAllText(path +

"UserData.txt").Replace(line\_detect[linenumber - 1],

line\_detect[linenumber - 1].Replace(line\_specific[13],

Convert.ToString((Convert.ToInt32(data[13]) +

Convert.ToInt32(data\_trf[1]))))));

using (var sw = new StreamWriter(path + ("mutasi-" +

data[11] + ".txt"), true))

{

sw.WriteLine(DateTime.Now + "," + "TRX" + num + "," +

rek\_pengirim + "," + data[11] + "," + "D" + "," +

data\_trf[1]+","+

Convert.ToString((Convert.ToInt32(data[13]) +

Convert.ToInt32(data\_trf[1]))) + "," + data\_trf[2]);

}

linenumber = 0;

foreach (string line in File.ReadLines(path +

"UserData.txt"))

{

data = line.Split('#');

linenumber++;

if (rek\_pengirim.Equals(Convert.ToInt32(data[11])))

{

break;

}

}

line\_detect = File.ReadAllLines(path + "UserData.txt");

line\_specific = line\_detect[linenumber - 1].Split('#');

File.WriteAllText(path + "UserData.txt",

File.ReadAllText(path +

"UserData.txt").Replace(line\_detect[linenumber - 1],

line\_detect[linenumber - 1].Replace(line\_specific[13],

Convert.ToString((Convert.ToInt32(data[13]) –

Convert.ToInt32(data\_trf[1]))))));

using (var sw = new StreamWriter(path + ("mutasi-" +

data[11] + ".txt"), true))

{

sw.WriteLine(DateTime.Now + "," + "TRX" + num + "," +

rek\_pengirim + "," + data\_trf[0] + "," + "C" + "," +

data\_trf[1] + ","+

Convert.ToString((Convert.ToInt32(data[13]) -

Convert.ToInt32(data\_trf[1]))) + "," + data\_trf[2]);

}

}

}

}

**CODES (PROGRAM USER)**

Console.Write("do you want to transaction again? (y/n) : ");

string ask\_2 = Console.ReadLine().ToLower();

if (ask\_2 == "y")

{

Console.Clear();

}

if (ask\_2 == "n")

{

Console.Clear();

MasterMenuUser.User(asd);

}

}

else

{

Console.Clear();

}

}

}

**CODES (PROGRAM USER)**

public static void TransferBank(int asd)

{

int num = new Random().Next(1000, 99999);

string[] line\_detect;

string[] line\_specific;

int linenumber;

int rek\_pengirim = asd;

var data = new object[20];

var data\_trf = new object[5];

var rmv\_empty\_line = File.ReadAllLines(path + "UserData.txt").Where(arg =>

!string.IsNullOrWhiteSpace(arg));

File.WriteAllLines(path + "UserData.txt", rmv\_empty\_line);

bool a = true;

while (a)

{

linenumber = 0;

Console.WriteLine("Enter Bank Code");

data\_trf[3] = Console.ReadLine();

Console.WriteLine("Enter Account Destination :");

data\_trf[0] = Console.ReadLine();

Console.WriteLine("Enter Total Transfer");

data\_trf[1] = Console.ReadLine();

Console.WriteLine("Enter Description :");

data\_trf[2] = Console.ReadLine();

string nama\_other\_bank = "";

foreach (string line in File.ReadLines(path + "kodebank.txt"))

{

data = line.Split(',');

linenumber++;

if (data\_trf[3].Equals(data[0]))

{

nama\_other\_bank = Convert.ToString(data[0]);

break;

}

else

{

continue;

}

}

Console.Clear();

Console.WriteLine("Bank Destination : " + data[1]);

foreach (string line in File.ReadLines(path + "UserData.txt"))

{

data = line.Split('#');

linenumber++;

if (data\_trf[0].Equals(data[11]))

{

break;

}

}

Console.WriteLine("Account Destination : " + data\_trf[0]);

Console.WriteLine("Account Name Destination : " + data[1]);

Console.WriteLine("Total Transfer : " + data\_trf[1]);

Console.WriteLine("Are you sure? (y/n)");

Console.Write("Enter : ");

string input = Console.ReadLine().ToLower();

if (input == "y")

**CODES (PROGRAM USER)**

{

if (true)

{

foreach (string line in File.ReadLines(path + "UserData.txt"))

{

data = line.Split('#');

if (rek\_pengirim.Equals(Convert.ToInt32(data[11])))

{

break;

}

else

{

continue;

}

}

if (Convert.ToInt32(data\_trf[1]) + 6500 >

Convert.ToInt32(data[13]))

{

Console.Clear();

Console.WriteLine("Transaction Failed!!");

Console.WriteLine("Not Enought Balance");

}

else

{

foreach (string line in File.ReadLines(path + "bank-" +

data\_trf[3] + ".txt"))

{

data = line.Split('#');

if (data\_trf[0].Equals(data[11]))

{

break;

}

else

{

continue;

}

}

if (!data\_trf[0].Equals(data[11]))

{

Console.Clear();

Console.WriteLine("Transaction Failed!!");

Console.WriteLine("Data Not Found");

}

else

{

Console.Clear();

Console.WriteLine("Transaction Success!");

Console.WriteLine("Ref : " + "TRX" + num);

Console.WriteLine("Bank Destination : " +

nama\_other\_bank);

Console.WriteLine("Account Destination : " +

data\_trf[0]);

Console.WriteLine("Total Transfer : " + data\_trf[1]);

Console.WriteLine("Time : " + DateTime.Now);

linenumber = 0;

**CODES (PROGRAM USER)**

foreach (string line in File.ReadLines(path +

"UserData.txt"))

{

data = line.Split('#');

linenumber++;

if (rek\_pengirim.Equals(Convert.ToInt32(data[11])))

{

break;

}

}

line\_detect = File.ReadAllLines(path + "UserData.txt");

line\_specific = line\_detect[linenumber - 1].Split('#');

File.WriteAllText(path + "UserData.txt",

File.ReadAllText(path +

"UserData.txt").Replace(line\_detect[linenumber - 1],

line\_detect[linenumber - 1].Replace(line\_specific[13],

Convert.ToString((Convert.ToInt32(data[13]) –

Convert.ToInt32(data\_trf[1]))))));

using (var sw = new StreamWriter(path + ("mutasi-" +

data[11] + ".txt"), true))

{

sw.WriteLine(DateTime.Now + "," + "TRX" + num + "," +

rek\_pengirim + "," + data\_trf[3] + "+" + data\_trf[0] + "," + "C"+ "," + data\_trf[1] + "," + Convert.ToString((Convert.ToInt32(data[13]) - Convert.ToInt32(data\_trf[1]))) + "," + data\_trf[2]);

}

}

}

}

Console.Write("Do you want to Transaction Again? (y/n) : ");

string ask\_2 = Console.ReadLine().ToLower();

if (ask\_2 == "y")

{

Console.Clear();

}

if (ask\_2 == "n")

{

Console.Clear();

MasterMenuUser.User(asd);

}

}

else

{

Console.Clear();

}

}

}

}

}