# Introduction

Shoe Salon is a shoeboutique that was built last 2010. All the shoes that were exported here in the Philippines are all come from different countries. For this year, I am proposing a reservation system for our valued customers. This system can reserve the shoe name, shoe size and quantity of the shoes that is not yet here in the Philippines.

# Objectives

* Can recall or view the list of customers who wants to reserve new upcoming shoes
* To provide excellent service to our valued customers
* To be able to create a working reservation system
* To make shoe reservation easier and organized

# Proposed System

Using a File Handling System using Java Programming, I will create a system that can add the customer’s name, shoe name, shoe size and quantity. This system can also update the customer’s details if ever the customer wants to avail new shoe. It can also delete the customer’s information if ever the customer cancelled their reserve product. And lastly the system can view the customer’s information so they can proceed to order their reserve product to different countries.

# Program Outline

Log In Screen Main menu

\*SHOE SALON\*

Username:

Password:

Welcome!

Add [A]

Edit [E]

View [V]

Delete [D]

Choose: Exit [X]

Adding Menu Edit Menu

Editing

Enter Transaction number: 001

Record Found. Edit? [Y/N]

Adding

Transaction Number: 001

Enter Last Name:

Enter First Name:

Shoe Name:

Shoe Size:

Quantity

Viewing

Transaction number: 001

Last Name:

First Name:

Shoe Name:

Shoe Size:

Quantity:

Deleting

Enter Transaction number: 001

Record found. Delete? [Y/N]

Deleting Menu Viewing Menu

# Program Flowchart

LOGIN

ENTER CHOICE

X

D

V

E

A

DELETE

VIEW

EDIT

ADD

DELETE TRANSACTION NUMBER

ENTER TRANSACTION NUMBER

ENTER CUSTOMER INFORMATION

NEW DATA SUCCESFULLY ADDED

RECORD DELETED

EDIT SUCCESS

EXIT

# Program Listing

import java.io.\*;

public class ShoeSalon

{

//Auto number

public static String generate\_id() throws IOException

{

BufferedReader input = new BufferedReader(new InputStreamReader(System.in ));

FileOutputStream out;

PrintStream p;

String ID=null;

int a = 1;

String Record;

try

{

FileInputStream FI = new FileInputStream("ID.txt");

DataInputStream DI = new DataInputStream(FI);

out = new FileOutputStream("ID.txt", true);

p = new PrintStream(out);

while(DI.available()!=0)

{

Record = DI.readLine();

a+=1;

}

ID=String.valueOf(a);

//write in file

p.println(ID);

p.close();

}

catch (Exception e)

{

System.err.println ("Error writing ID");

}

return ID;

}

//Add Function

public static void Add() throws Exception

{

BufferedReader input = new BufferedReader(new InputStreamReader(System.in ));

FileOutputStream out;

PrintStream p;

String transc, ELast, EFirst,Sname, Ssize, qty;

try

{

out = new FileOutputStream("myFile.txt", true);

p = new PrintStream( out );

System.out.print("Transaction Number: ");

transc = "00" + generate\_id();

System.out.println(transc);

System.out.print("Enter Last Name: ");

ELast = input.readLine();

System.out.print("Enter First Name: ");

EFirst = input.readLine();

System.out.print("Shoe Name: ");

Sname = input.readLine();

System.out.print("Shoe Size: ");

Ssize = input.readLine();

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

qty = input.readLine();

//write in file

p.println(transc);

p.println(ELast);

p.println(EFirst);

p.println(Sname);

p.println(Ssize);

p.println(qty);

p.close();

}

catch (Exception e)

{

System.err.println ("Error writing to file");

}

}

//SearchDelete

public static void Delete() throws Exception

{

BufferedReader input = new BufferedReader(new InputStreamReader(System.in ));

try

{

FileInputStream FI = new FileInputStream("myfile.txt");

DataInputStream DI = new DataInputStream(FI);

FileOutputStream out;

PrintStream p;

out = new FileOutputStream("temp.txt");

p = new PrintStream( out );

String transc, Record, Choice;

int Found =0;

System.out.print("Enter Transaction Number: ");

transc = input.readLine();

while(DI.available()!=0)

{

Record = DI.readLine();

if(transc.equals(Record))

{

System.out.println("Record found. Delete?[Y/N]?");

Choice = input.readLine();

Choice = Choice.toUpperCase();

if(Choice.equals("Y"))

{

DI.readLine();

DI.readLine();

DI.readLine();

DI.readLine();

DI.readLine();

Found = 1;

}

else

p.println(Record);

}

else

{

p.println(Record);

}

}

if(Found <=0)

{

System.out.println("No Record");

}

p.close();

DI.close();

//delete old file

File file = new File("myFile.txt");

file.delete();

//saving edited file

File oldFile = new File("temp.txt");

oldFile.renameTo(new File("myFile.txt"));

}

catch(Exception e)

{

System.out.println("File input error");

}

}

//Edit Function

public static void Edit() throws Exception

{

BufferedReader input = new BufferedReader(new InputStreamReader(System.in ));

try

{

FileInputStream FI = new FileInputStream("myfile.txt");

DataInputStream DI = new DataInputStream(FI);

FileOutputStream out;

PrintStream p;

out = new FileOutputStream("temp.txt");

p = new PrintStream( out );

String transc, Record, Choice, transc2, ELast, EFirst, Sname, Ssize, qty;

int Found =0;

System.out.print("Enter Transaction Number: ");

transc = input.readLine();

while(DI.available()!=0)

{

Record = DI.readLine();

if(transc.equals(Record))

{

System.out.println("Record found. Edit?[Y/N]?");

Choice = input.readLine();

Choice = Choice.toUpperCase();

if(Choice.equals("Y"))

{

System.out.println("Transaction Number: " + transc);

System.out.print("Enter Last Name: ");

ELast = input.readLine();

DI.readLine();

System.out.print("Enter First Name: ");

EFirst = input.readLine();

DI.readLine();

System.out.print("Shoe Name: ");

Sname = input.readLine();

DI.readLine();

System.out.print("Shoe Size: ");

Ssize = input.readLine();

DI.readLine();

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

qty = input.readLine();

DI.readLine();

//write in file

p.println(transc);

p.println(ELast);

p.println(EFirst);

p.println(Sname);

p.println(Ssize);

p.println(qty);

Found = 1;

}

else

p.println(Record);

}

else

{

p.println(Record);

}

}

if(Found <=0)

{

System.out.println("No Record");

}

p.close();

DI.close();

//delete old file

File file = new File("myFile.txt");

file.delete();

//saving edited file

File oldFile = new File("temp.txt");

oldFile.renameTo(new File("myFile.txt"));

}

catch(Exception e)

{

System.out.println("File input error");

}

}

//ViewRecord

public static void View() throws Exception

{

BufferedReader input = new BufferedReader(new InputStreamReader(System.in ));

int a=0;

try

{

FileInputStream FI = new FileInputStream("myfile.txt");

DataInputStream DI = new DataInputStream(FI);

while(DI.available()!=0)

{

System.out.println("Transaction Number: " + DI.readLine());

a+=1;

System.out.println("Last Name: " + DI.readLine());

a+=1;

System.out.println("First Name: " + DI.readLine());

System.out.println("Shoe Name: " + DI.readLine());

System.out.println("Shoe Size: " + DI.readLine());

System.out.println("Quantity: " + DI.readLine());

System.out.println("");

}

if(a<=1)

{

System.out.println("No record");

}

DI.close();

}

catch(Exception e)

{

System.out.println("File input error");

}

}

//Main Menu Function

public static void MainMenu() throws Exception

{

BufferedReader input = new BufferedReader(new InputStreamReader(System.in ));

String Choice;

int choose = 0;

do

{

System.out.println("\nAdd [A] \n\nEdit [E] \n\nView [V] \n\nDelete[D] \n\n\n\t\t\t\t\t\t\t\tEXIT[X] ");

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

Choice = input.readLine();

Choice = Choice.toUpperCase();

if(Choice.equals("A"))

{

System.out.println("Adding");

Add();

}

else if(Choice.equals("E"))

{

System.out.println("Editing");

Edit();

}

else if(Choice.equals("V"))

{

System.out.println("Viewing");

View();

}

else if(Choice.equals("D"))

{

System.out.println("Deleting");

Delete();

}

else if(Choice.equals("X"))

{

System.out.println("THANK YOU!");

choose = 1;

}

else

{

System.out.println("Error");

}

}while(choose<=0);

System.in.read();

}

//Main Function

public static void main(String[] args) throws Exception

{

BufferedReader input = new BufferedReader(new InputStreamReader(System.in ));

String Username;

String Password;

int choose=1;

do

{

System.out.println("\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\t\t\t\t\*SHOE SALON\*\n\n\n\n\n\n");

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

Username = input.readLine();

System.out.print("\nPassword: ");

Password = input.readLine();

if((Username.equals("Admin")) && (Password.equals("123456")))

{

System.out.println("\t\t\tWELCOME!");

MainMenu();

}

else if ((Username.equals("")) && (Password.equals("")))

{

System.out.println("\n\nWRONG INPUT!");

choose = 0;

}

else

{

System.out.println("WRONG INPUT PLEASE RE LOGIN");

System.in.read();

}

System.in.read();

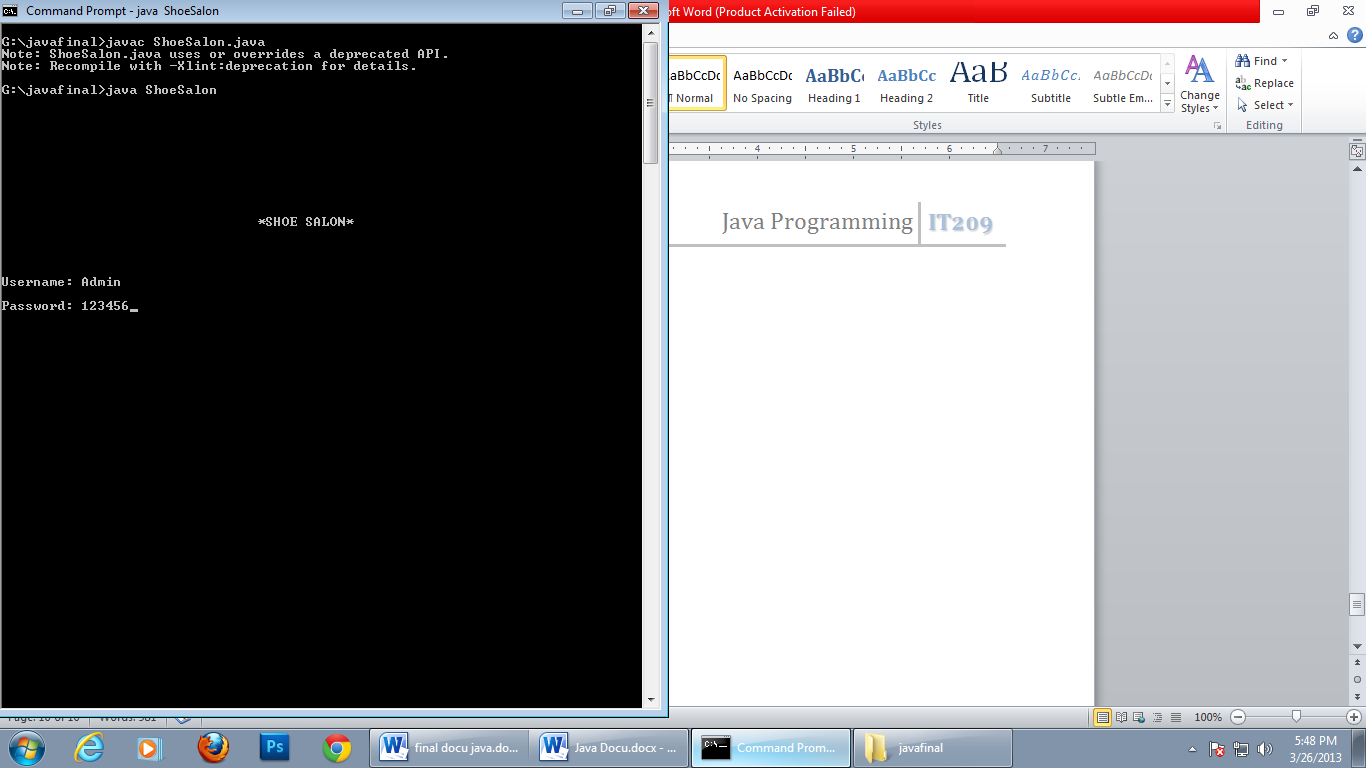
}while(choose>=1);

}

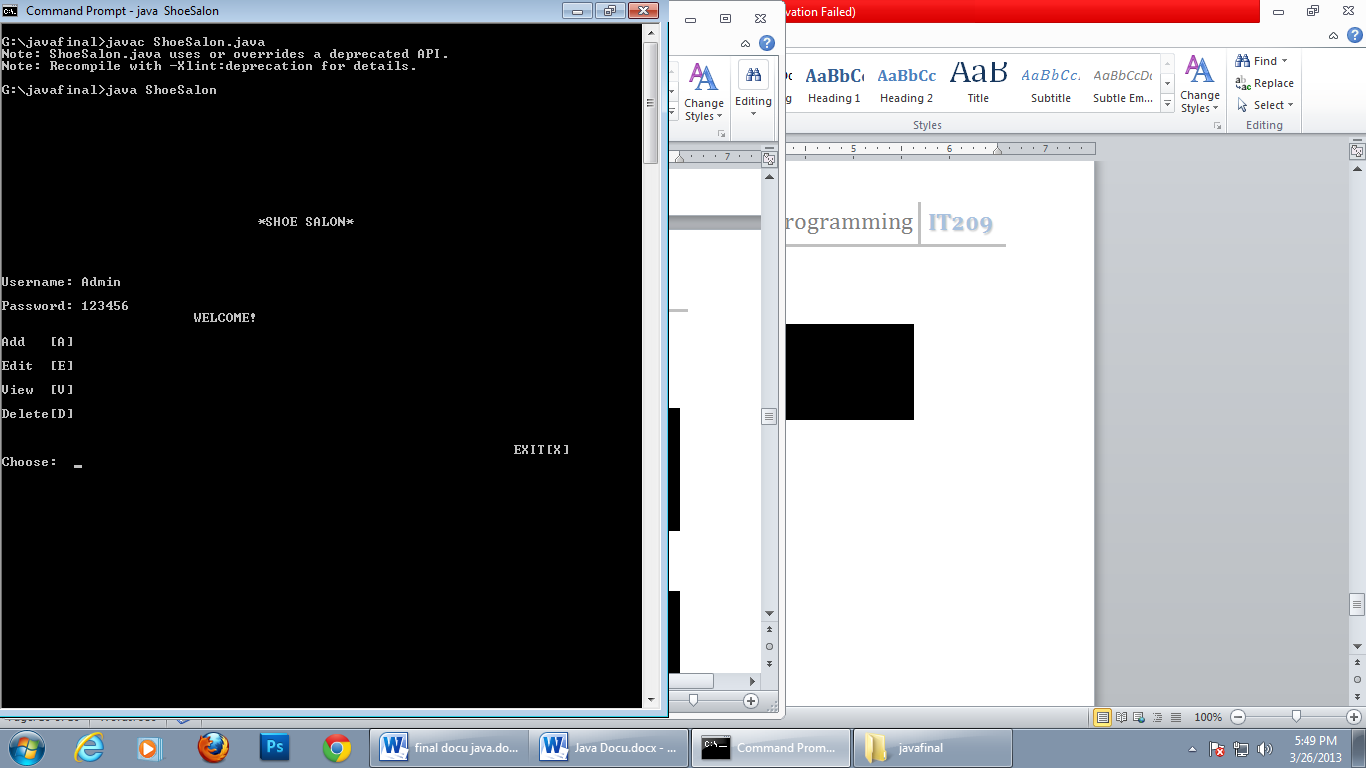
}

# Screen Design

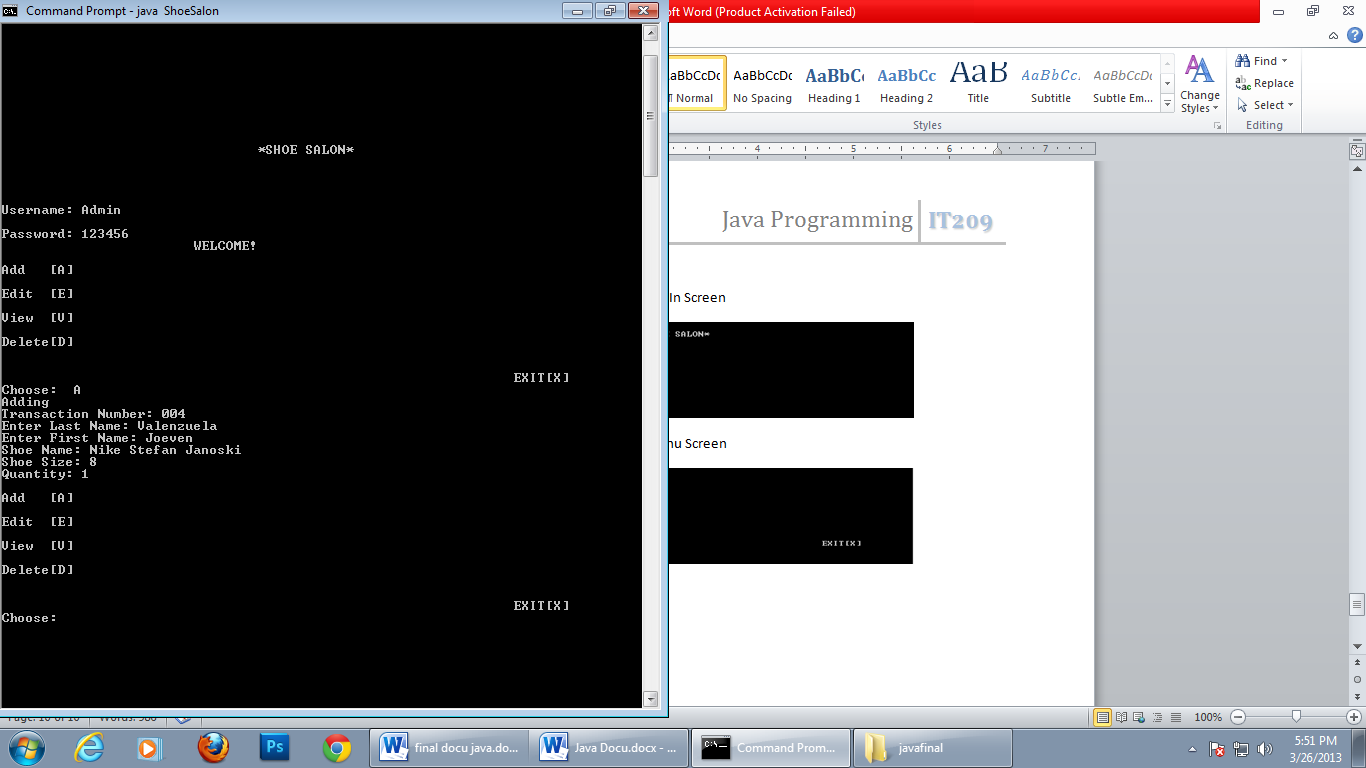
Log In Screen



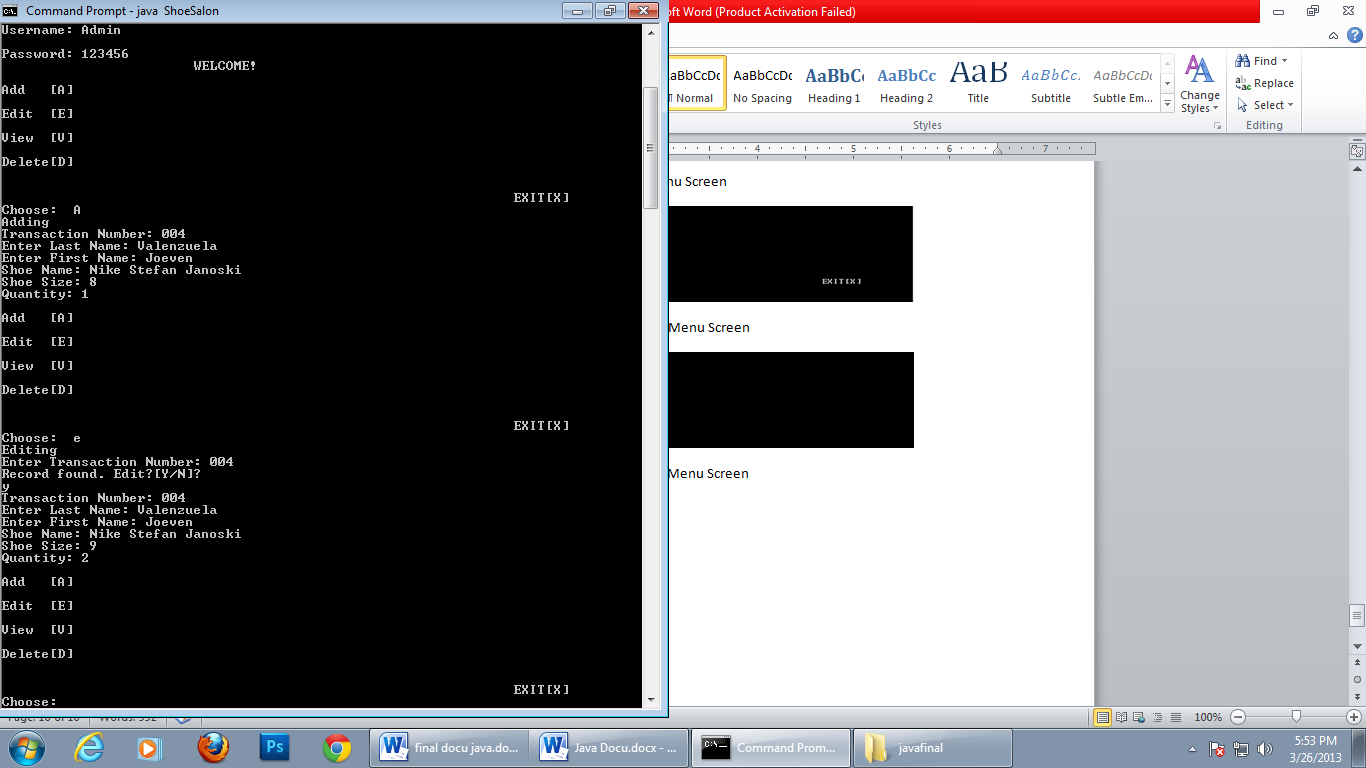
Menu Screen



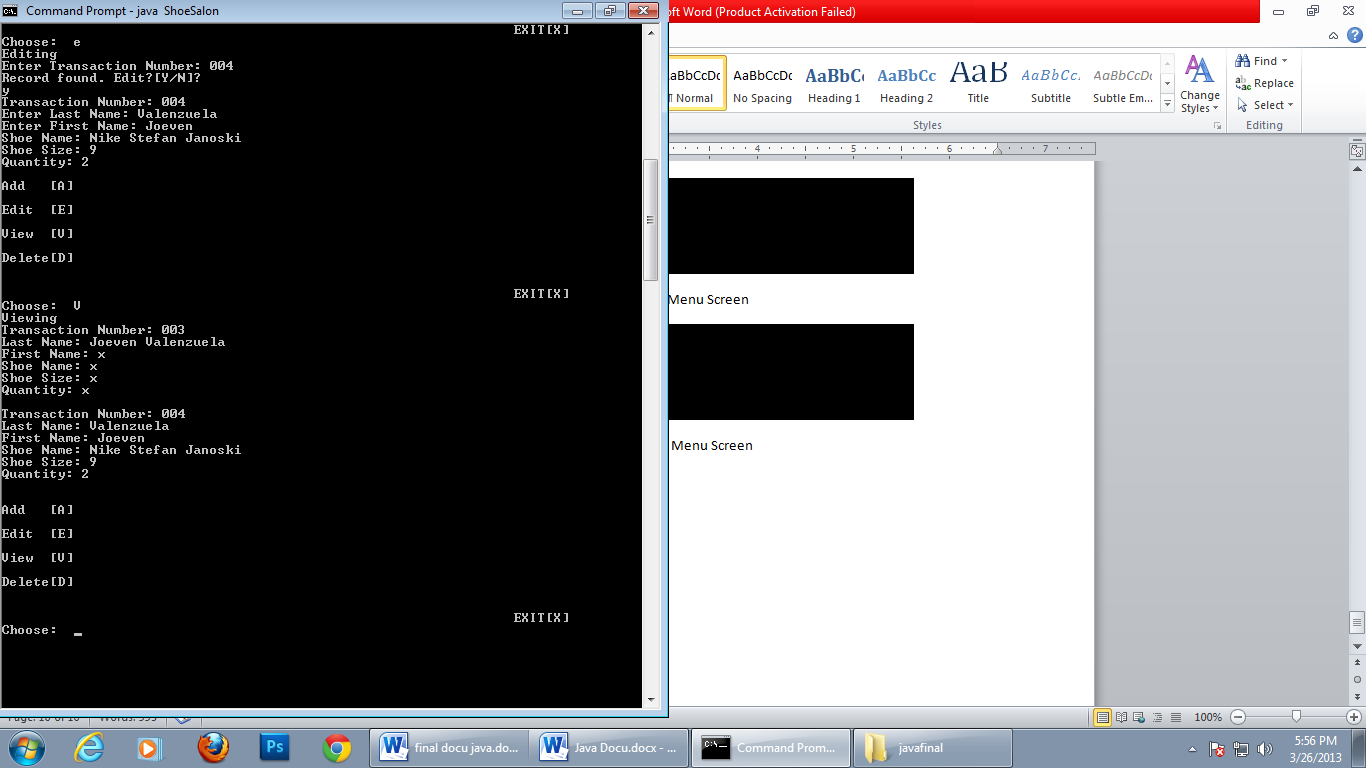
Adding Menu Screen



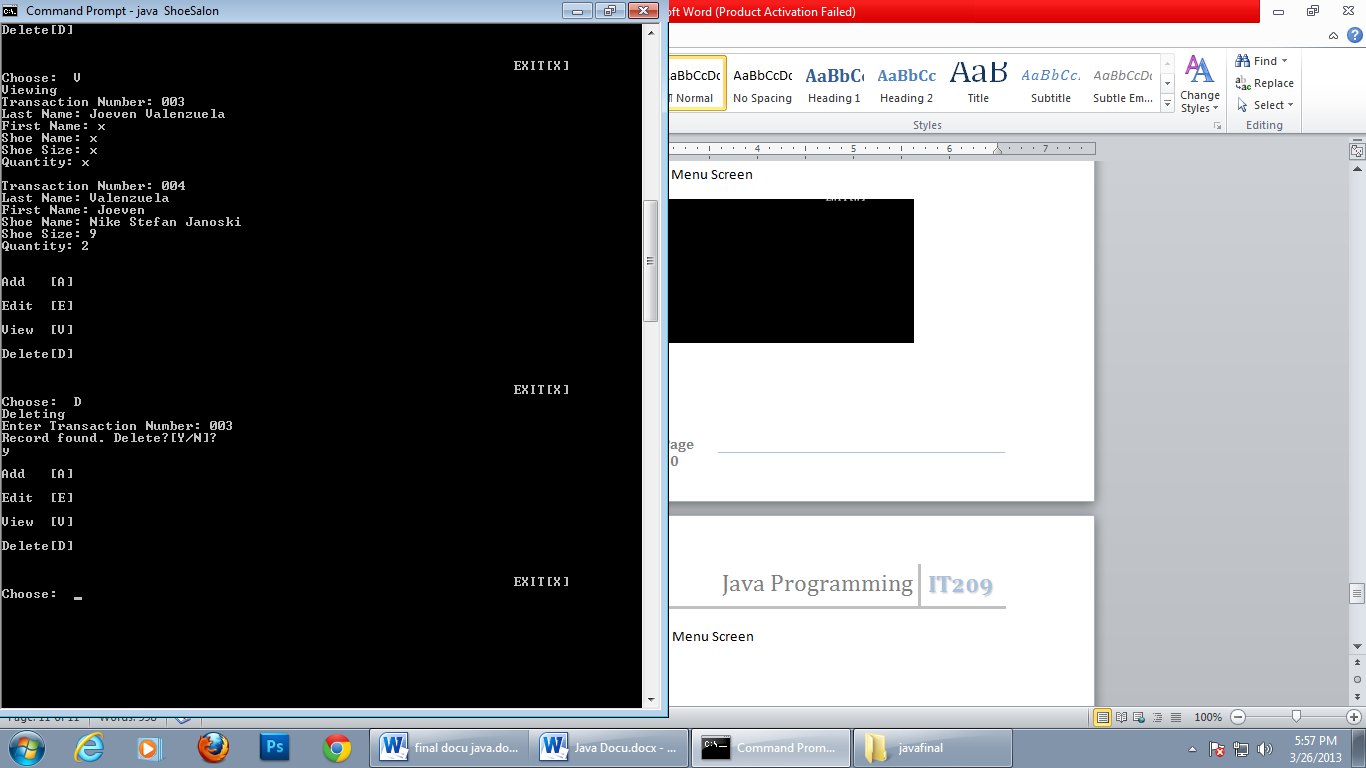
Editing Menu Screen



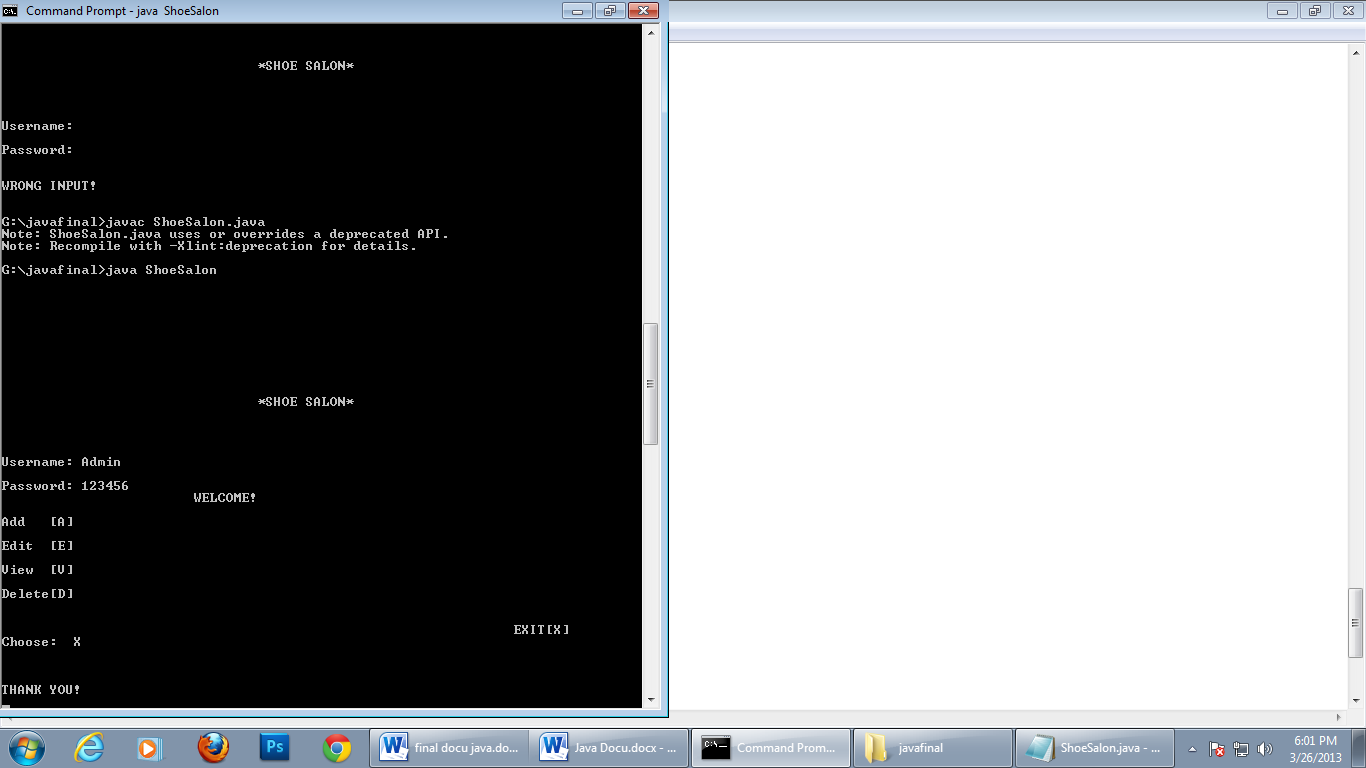
Viewing Menu Screen



Deleting Menu Screen



Log Out Menu Screen



# Program Testing

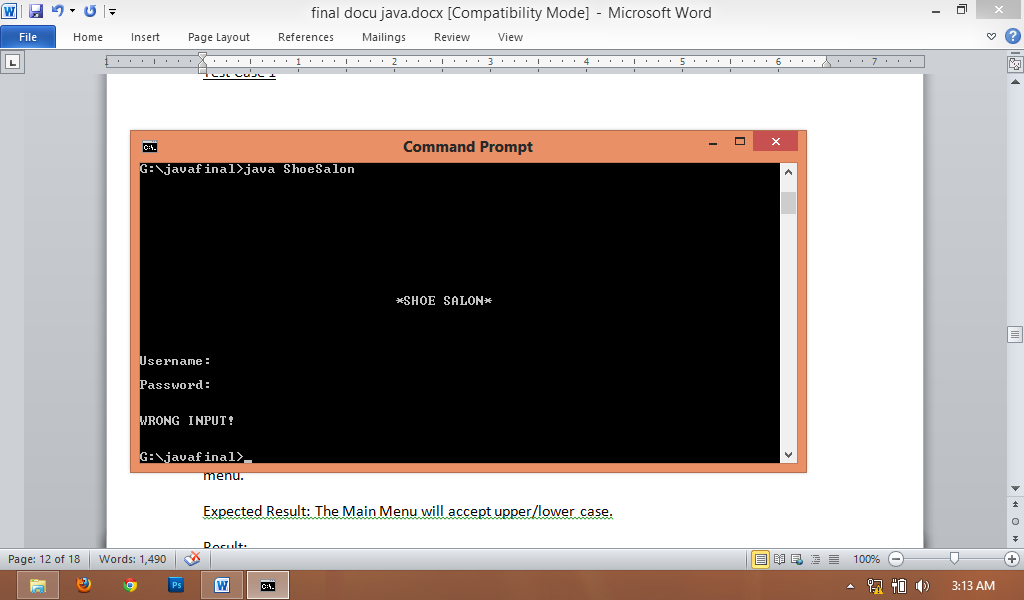
|  |  |
| --- | --- |
| Test Case | Objectives |
| 1 | To check if Username and Password is going to exit when you didn’t put anything. |
| 2 | To check if Main Menu can also accept small cases then proceed to desired menu. |
| 3 | To check if Adding Menu can add the customer’s information. |
| 4 | To check if Editing Menu can update the customer’s information. |
| 5 | To check if Viewing Menu can view the entire customer’s information. |
| 6 | To check if Deleting Menu can delete the stored record of the customer’s information. |
| 7 | To check if Log out can Exit your system. |

Test Case 1

Test Objective: To check if Username and Password is going to exit when you didn’t put anything.

Expected Result: The program will exit if you leave the username and password blank.

Result:

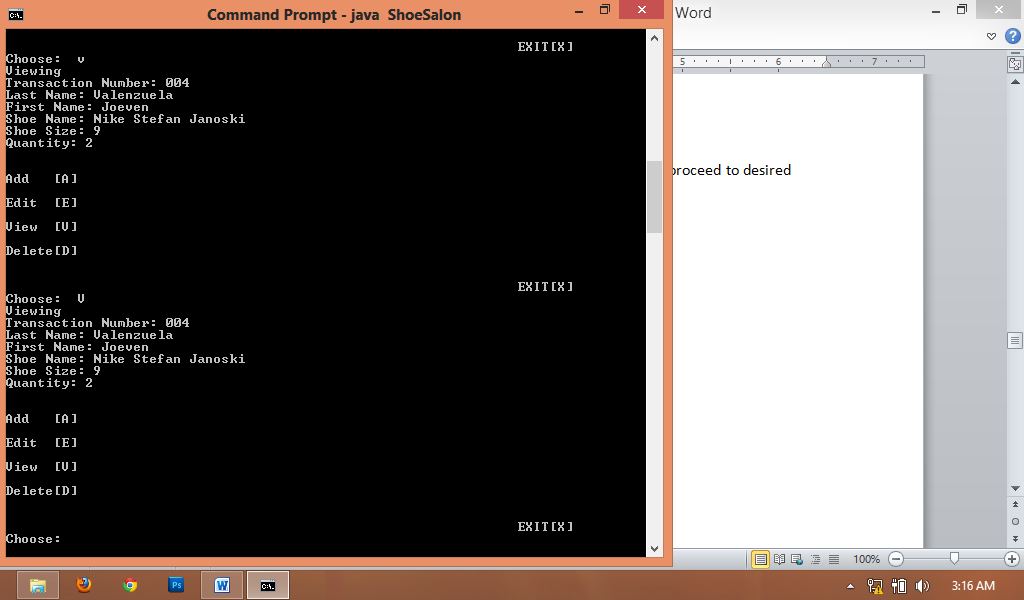


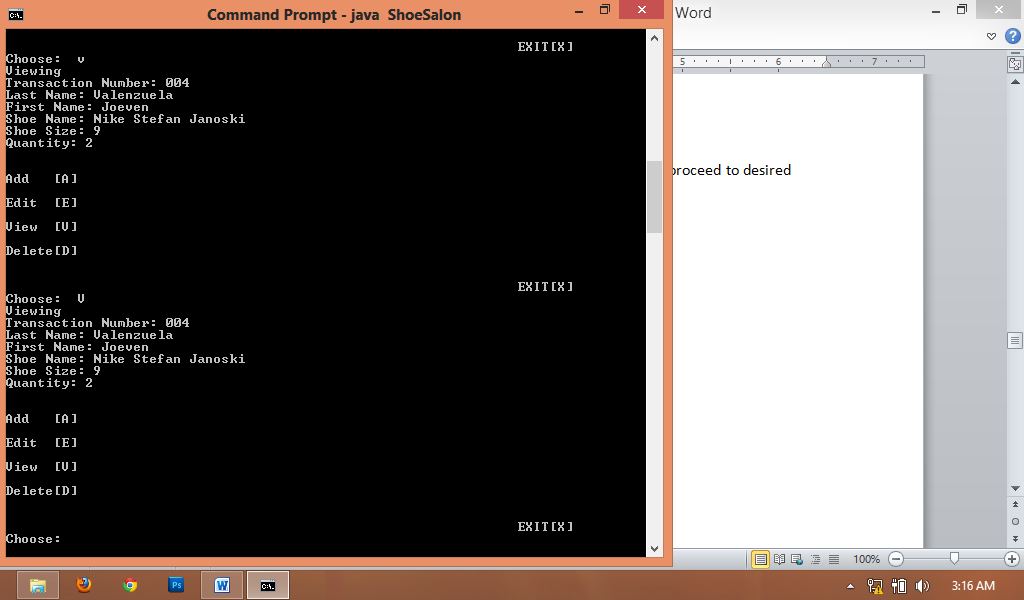
Test Case 2

Test Objective: To check if Main Menu can also accept small cases then proceed to desired menu.

Expected Result: The Main Menu will accept upper/lower case.

Result:



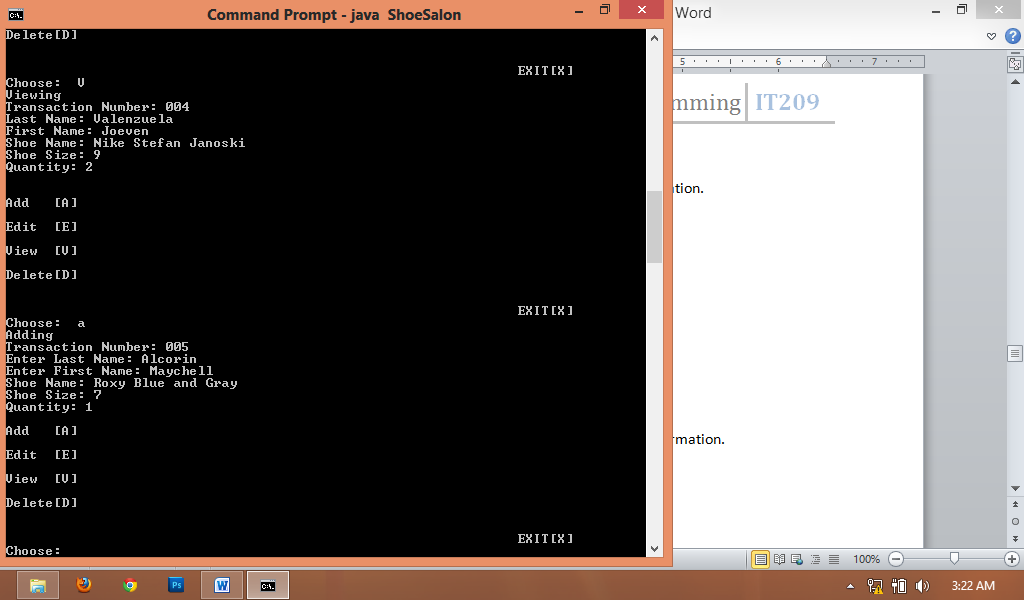


Test Case 3

Test Objective: To check if Adding Menu can add the customer’s information.

Expected Result: You can add the customer’s information.

Result:

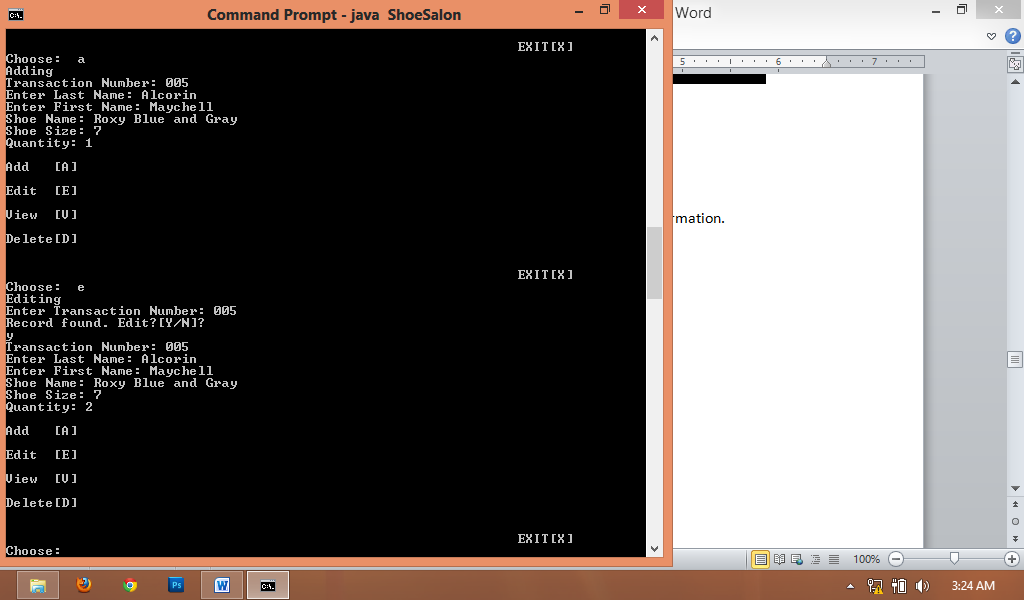


Test Case 4

Test Objective: To check if Editing Menu can update the customer’s information.

Expected Result: You can edit or update the customer’s information.

Result:

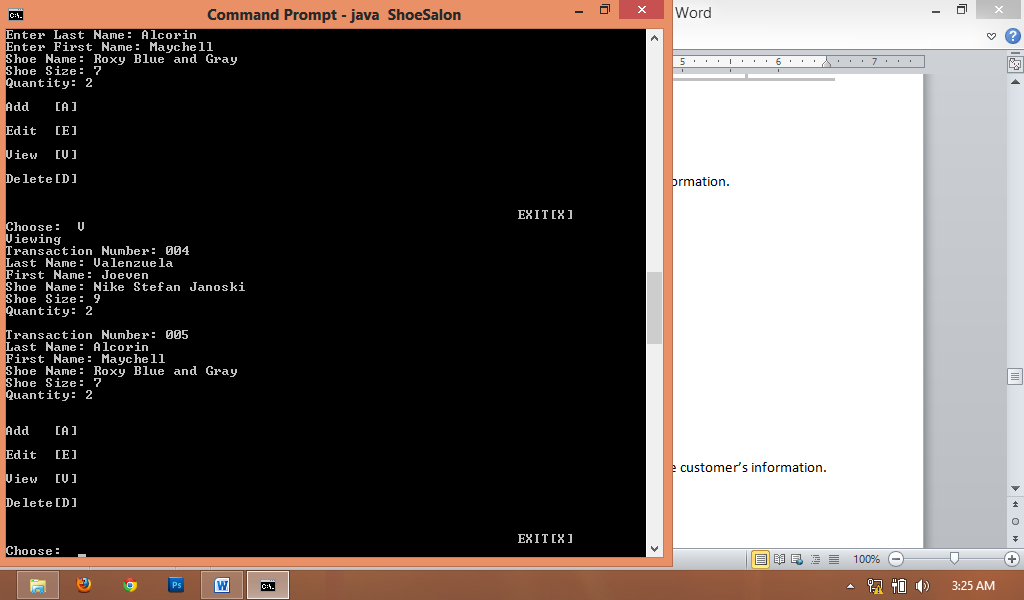


Test Case 5

Test Objective: To check if Viewing Menu can view the entire customer’s information.

Expected Result: You can view all the customers’ information.

Result:

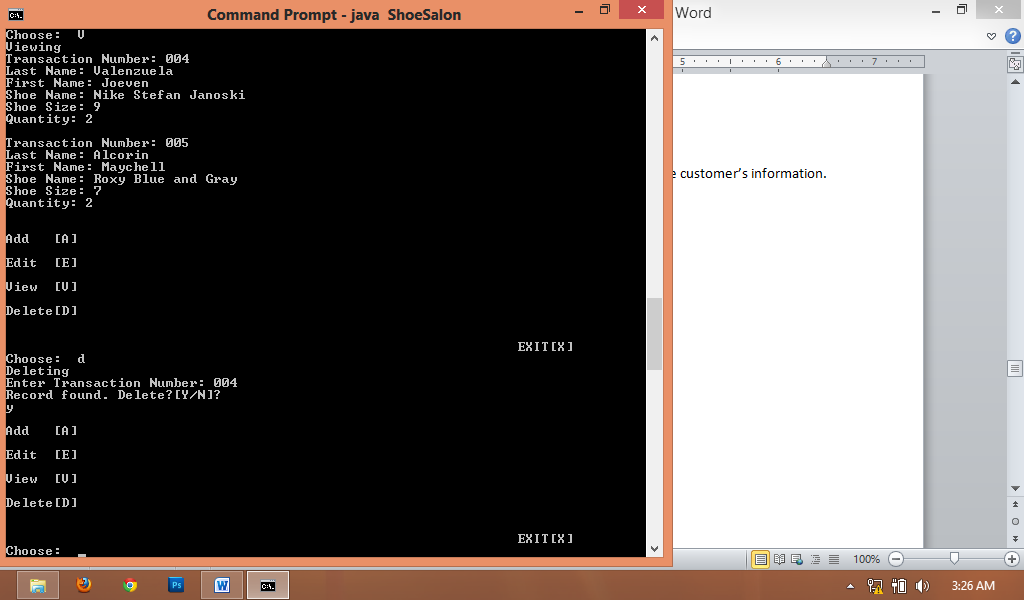


Test Case 6

Test Objective: To check if Deleting Menu can delete the stored record of the customer’s information.

Expected Result: You can delete the customer’s information.

Result:

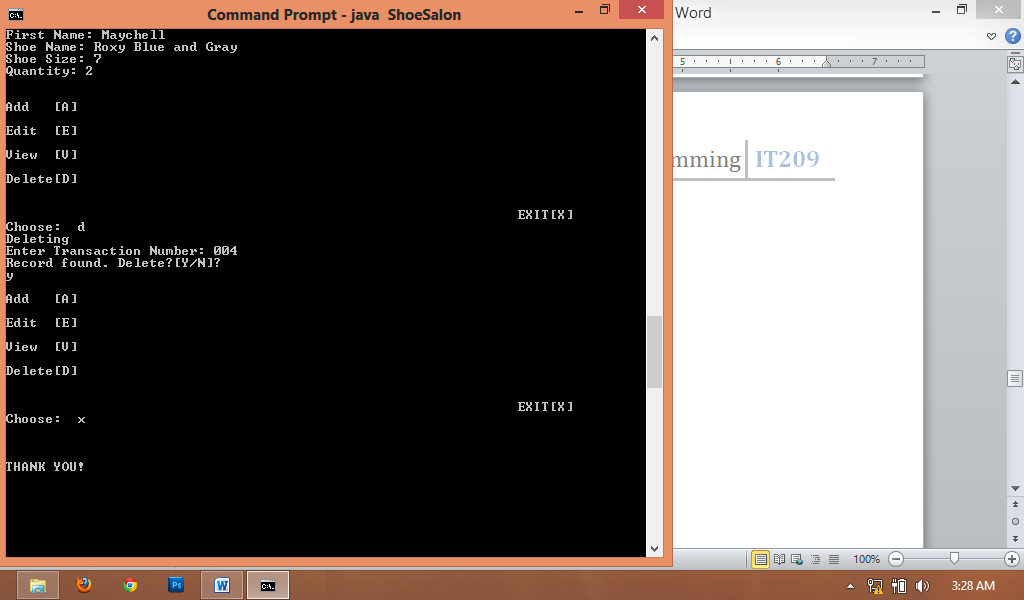


Test Case 7

Test Objective: To check if Log out can Exit your system.

Expected Result: You can log out your program by pressing “x”.

Result:



# Test Results

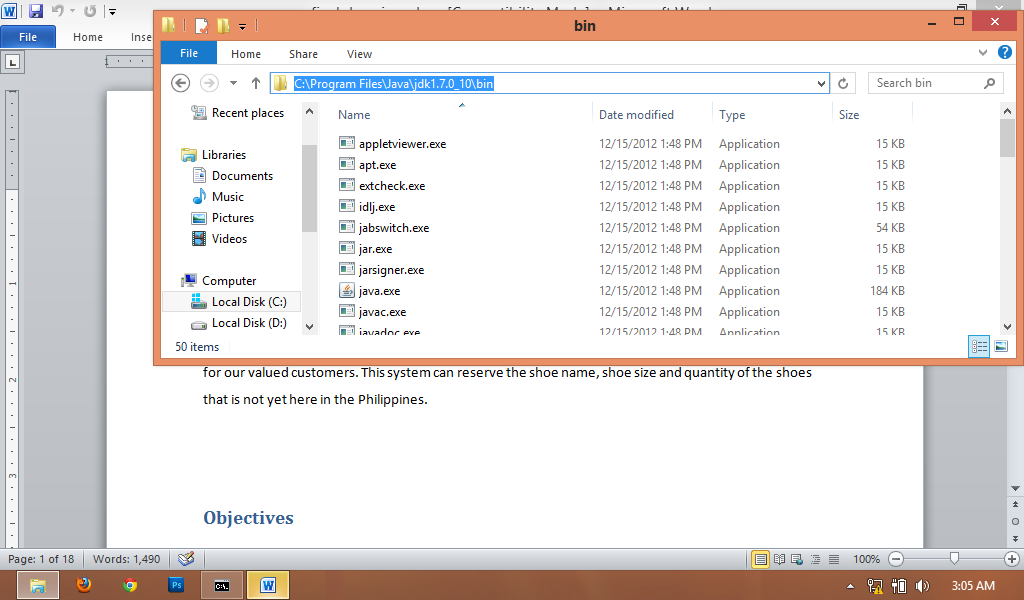
|  |  |
| --- | --- |
| Test Case | Test Results |
| 1 | Successful |
| 2 | Successful |
| 3 | Successful |
| 4 | Successful |
| 5 | Successful |
| 6 | Successful |
| 7 | Successful |

# Input Specification

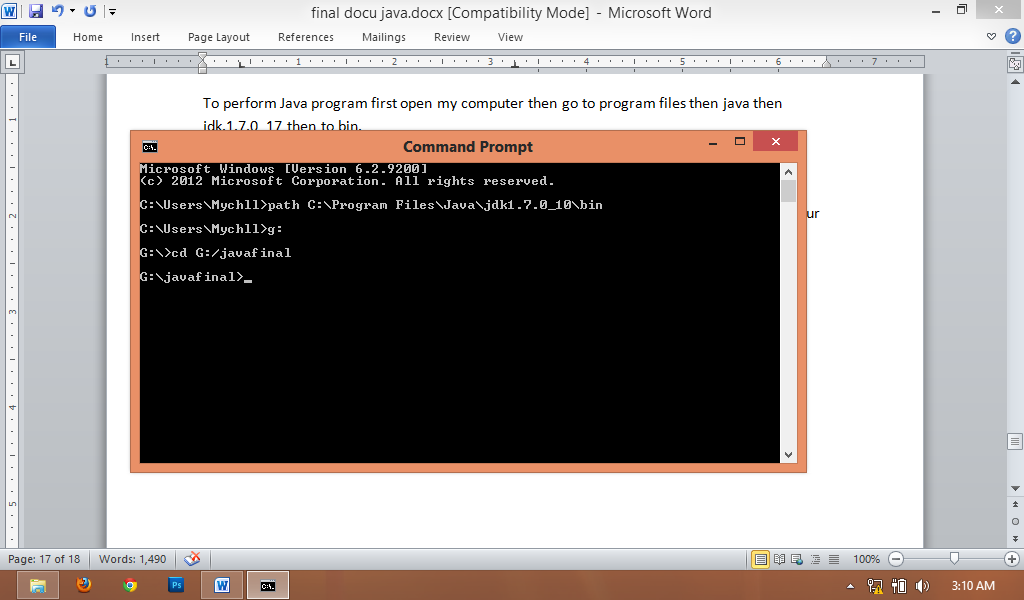
|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Type | Range |
| transc | Transaction Number | String | A-Z,0-9 |
| ELast | Customers Last Name | String | A-Z,0-9 |
| EFirst | Customers First Name | String | A-Z,0-9 |
| Sname | Shoe name | String | A-Z,0-9 |
| Ssize | Shoe size | String | A-Z,0-9 |
| qty | Shoe Quantity | String | A-Z,0-9 |

# User Manual

To perform Java program first open my computer then go to program files then java then jdk.1.7.0\_17 then to bin.

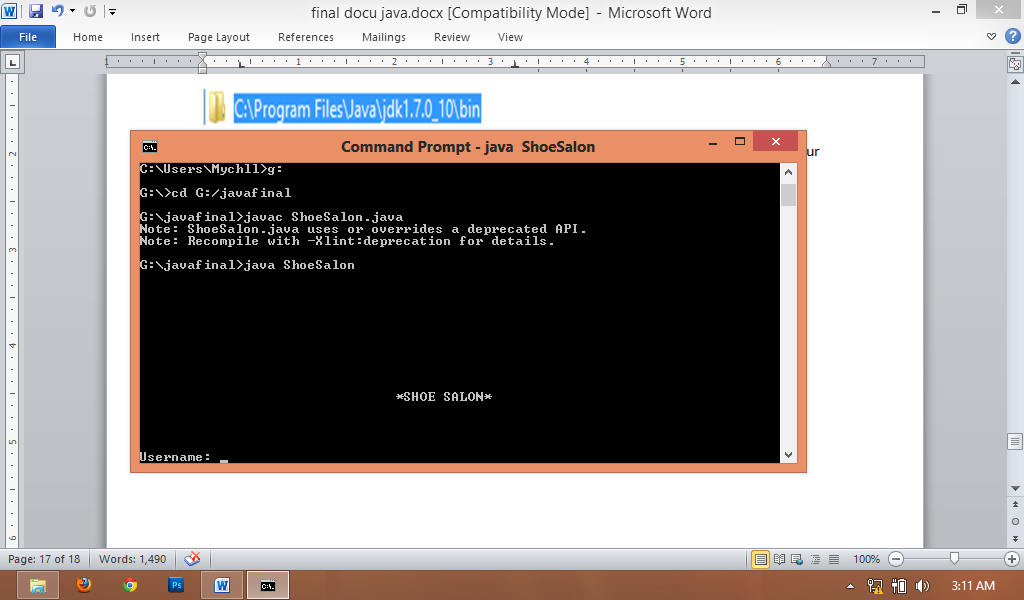


Open your command prompt (cmd). Type your path where the java folder is. Then change your directory where your file is saved.



After changing the directory, the syntax for compiling your file is javac <filename.java>

You will now proceed to Log In Menu where you can type your username and password.



After logging in, The program will proceed you to the Main Menu.

# Implementation

Strength

This program can simply add, update, view and delete information about the customers.

Weaknesses

This program doesn’t have any designs because we are just using a command prompt.

Enhancements

This program needs to encrypt the password, needs more design, need a search bar in case the program will handled 10 or more customers.