1. **INTRODUCTION**
   1. **PROJECT BACKGROUND**

Over the last decade e-transactions using credit and debit cards has grown enormously in size, importance and finally, but not last, in complexity. So a proper understanding of the user’s behavior becomes important in order to organize the content in the most efficient way. Users are becoming more and more dependent on these e-transactions for most of their tasks. In this project we will focus on how to perform e-transaction without using credit and debit cards but using a new schema called virtual money transfer which is efficient and easy to use.

In computer science, virtual is an acronym for “simulated, or carried on by means of a computer or computer network .”

E-transactions using credit and debit cards have become very popular amongst its users for their ease of operations and the simplicity they afford in transactions .The credit and debit cards are now increasingly being used at point of sales POS terminals at Merchant Establishment MEs and for making online purchases through the Internet , apart from using at ATMs for drawing cash .With the popularity and usage levels increasing rapidly , it is necessary for the users to understand the risks involved in card based transactions to minimize any attempts to misuse thereof by taking sufficient care in operations.

Though these methods are successful and have been prevalent throughout the world since many years. But now everything in the world is turning towards eco friendly options. So in the place of these cards we implement an alternative way of virtually transferring the money when required. moreover this method also saves a lot of economic investment done during the manufacturing of cards, and also saves lot of time.

**1.2 OBJECTIVES**

* To design an simple user friendly application that perfectly substitutes the existing system and make it more efficient.
* To display the balance of the customer.
* To provide an interface for a new user to register with the application.
* To provide an interface for shopping malls that wish to implement the application, to register into the application
* To generate a session password before every transaction

**2.LITERATURE SURVEY**

**2.1 JAVA**

**2.1.1 OVERVIEW OF JAVA**

Initially the language was called “oak” but it was renamed as “Java” in 1995. The primary motivation of this language was need for a platform-independent (i.e., architecture neutral) language that could be used software to the embedded in various consumer electronic devices. The Java programming language and environment is designed to solve a number of modern programming practices[2].

* Java is a programmer’s language.
* Java is cohesive and consistent.
* Except for those constraints imposed by the Internet environment, Java gives the programmer, full control.

**Java can be used to create two types of programs**

Applications and Applets: An application is a program that runs on our computer under the operating system of that computer. It is more or less like one creating using C or C++. Java’s ability to create Applets makes it important. An apple is an application designed to be transmitted over the Internet and executed by a Java-Compatible web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can react to the user input and dynamically change.

**2.1.2 FEATURES OF JAVA**

Java: A simple, object-oriented, network-savvy, interpreted, robust, secure, architecture neutral, portable, high-performance, multithreaded, dynamic language.

### java.io

The [java.io](http://download.oracle.com/javase/7/docs/api/java/io/package-summary.html) package contains classes that support [input and output](http://en.wikipedia.org/wiki/Input/output). The classes in the package are primarily [stream-oriented](http://en.wikipedia.org/wiki/Stream_(computing)); however, a class for [random access](http://en.wikipedia.org/wiki/Random_access) [files](http://en.wikipedia.org/wiki/Computer_file) is also provided. The central classes in the package are [InputStream](http://download.oracle.com/javase/7/docs/api/java/io/InputStream.html) and [OutputStream](http://download.oracle.com/javase/7/docs/api/java/io/OutputStream.html), which are [abstract](http://en.wikipedia.org/wiki/Class_(computer_science)#Concrete_classes) base classes for reading from and writing to [byte streams](http://en.wikipedia.org/wiki/Byte_stream), respectively. The related classes[Reader](http://download.oracle.com/javase/7/docs/api/java/io/Reader.html) and [Writer](http://download.oracle.com/javase/7/docs/api/java/io/Writer.html) are abstract base classes for reading from and writing to [character](http://en.wikipedia.org/wiki/Character_(computing)) streams, respectively. The package also has a few miscellaneous classes to support interactions with the host [file system](http://en.wikipedia.org/wiki/File_system).

### java.util

The java.util package contains the collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes (a string tokenizer, a random-number generator, and a bit array).

**2.2 CLASSES**

* **BufferedReader**

public class **BufferedReader** extends Reader

Read text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines. The buffer size may be specified, or the default size may be used. The default is large enough for most purposes. In general, each read request made of a Reader causes a corresponding read request to be made of the underlying character or byte stream. It is therefore advisable to wrap a BufferedReader around any Reader whose read() operations may be costly, such as FileReaders and InputStreamReaders. Programs that use DataInputStreams for textual input can be localized by replacing each DataInputStream with an appropriate BufferedReader.

* **DataOutputStream**

public class **DataOutputStream**

extends [FilterOutputStream](http://docs.oracle.com/javase/6/docs/api/java/io/FilterOutputStream.html)

implements [DataOutput](http://docs.oracle.com/javase/6/docs/api/java/io/DataOutput.html)

The class DataOutputStream allow a programmer to write primitive Java data types to an output stream in a portable way. This class provide different method to write different type of data like writeInt() for integer, writeChars() for string etc.

* **IOException**

public class **IOException**

extends Exception

Signals that an I/O exception of some sort has occurred. This class is the general class of exceptions produced by failed or interrupted I/O operations.

* **InputStreamReader**

public class **InputStreamReader**

extends Reader

An InputStreamReader is a bridge from byte streams to character streams: It reads bytes and decodes them into characters using a specified [charset](http://docs.oracle.com/javase/6/docs/api/java/nio/charset/Charset.html" \o "class in java.nio.charset). The charset that it uses may be specified by name or may be given explicitly, or the platform's default charset may be accepted.

Each invocation of one of an InputStreamReader'sread() methods may cause one or more bytes to be read from the underlying byte-input stream. To enable the efficient conversion of bytes to characters, more bytes may be read ahead from the underlying stream than are necessary to satisfy the current read operation.

* **PrintStream**

public class **PrintStream**

extends [FilterOutputStream](http://docs.oracle.com/javase/1.5.0/docs/api/java/io/FilterOutputStream.html)

implements Appendable, Closeable

A PrintStream adds functionality to another output stream, namely the ability to print representations of various data values conveniently. Two other features are provided as well. Unlike other output streams, a PrintStream never throws an IOException; instead, exceptional situations merely set an internal flag that can be tested via the checkError method. Optionally, a PrintStream can be created so as to flush automatically; this means that the flush method is automatically invoked after a byte array is written, one of the println methods is invoked, or a newline character or byte ('\n') is written. All characters printed by a PrintStream are converted into bytes using the platform's default character encoding. The [PrintWriter](http://docs.oracle.com/javase/1.5.0/docs/api/java/io/PrintWriter.html) class should be used in situations that require writing characters rather than bytes.

**2.3 METHODS**

* **readLine()**This method is used to read the data line by line. Then returns the data of a line as a single string.
* **writeBytes()**

This method writes out the string to the underlying output stream as a sequence of bytes.

* **start()**

This method causes this thread to begin execution; the Java Virtual Machine calls the run method of this thread.

* **trim():**This method removes the blank spaces from both ends of the given string (Front and End).
* **valueof()**

Thismethod converts data from its internal format into a human-readable form. It is a static method that is overloaded within Stringfor all of Java's built-in types, so that each type can be converted properly into a string. valueOf( )is also overloaded for type Object, so an object of any class type you create can also be used as an argument.

* **equals()**

This particular method is used to make equal comparison between two objects. There are two types of comparisons in Java. One is using “= =” operator and another is “equals()”. I hope that you know the difference between this two. More specifically the “.equals()” refers to equivalence relations. So in broad sense you say that two objects are equivalent they satisfy the “equals()” condition.

**3.1 PROBLEM DEFINITION**

**3.1 APPLICATION OVERVIEW**

Payments after shopping is generally done through cash r else credit/debit cards. After billing the bill is issued to the customer. Depending on his/her interest the payment is done either by cash or credit cards. Our focus here is on payments done through cards, the customer gives his/her card, then the concerned person swipes the card in specially designed machine called swiping machine. After swiping the card, its verification is done electronically and automatically. If verification fails i.e.; if card is not authorized an error message is shown and transaction is not allowed. On the other hand if card is verified, the transaction/payment is completed and a receipt is generated on which the customer has to endorse his/her sign to confirm the payment.



Figure 3.1  Credit card swiping machine

**Swiping machine overview:**

History of IBM developed the concept of using magnetic strips to store customer information in 1970. A credit or debit card swiping machine to read the data was developed at the same time. The strip's data is stored in the form of positive and negative charges, which a credit or debit card swiping machine converts into 0's and 1's, the binary basis on which computers operate. The computer then converts this binary code into numeric values.

Credit or debit card swiping machines read the information on the card's magnetic strip. The data is transmitted via telephone line to the cardholder's issuing bank to determine whether there is sufficient money to back up the transaction. These swiping machines ensure safe and reliable sales transactions for the merchant. They also offer customers the convenience of not carrying cash or writing checks.

But, If the issuing bank's or merchant's computer system is hacked, cardholders' personal information is stolen and can be used for fraudulent purposes. "Skinformation at the point of sale is becoming more common, too. These swiping machines consume lot of time.

**3.2 PROBLEM DESCRIPTION**

Virtual money transfer includes making payments after the shopping without use of any credit/debit cards and just by using an account number and password involving a simple 5 step procedure and without use of any kind of swiping machines.

Initially the user who uses this application has to select a bank and then should register himself in that bank. This provides a interface for a new user to register with application, and generates an account number and password at the same time assuming that he/she already has an account in the bank with some balance. The mall which uses this application also has to register itself and a VMT code is generated for each mall which is used at the time of transaction. After registration of each user and shopping mall, that particular details of customer and mall are added to database (text files) and each bank has its own database.

The application starts with accepting login info from user and verification from database of application. User enters his bill number and amount used for shopping

and also enters current date on which transaction is done. After verification of date, user has to enter a session password which is unique for each transaction. . For security reasons, each time the customer performs a transaction a fresh and new session password is generated using initial password generated during customer registration. After verification of session password, payment of money and application is completed.

This method surely is far better than existing system that is e-transactions using credit and debit cards as it does not involve any manufacturing of plastic cards and also save lot of money and time.

**3.3 MODULE SPECIFICATION**

* **PAYMENT :**

* + - Designing a GUI using applet framework.
    - Accepting login info from user and verification from the database of the application.
    - Enter bill number ,bill amount for money transaction.
    - Enter the current date and verification of date.
    - Payment of money and completion of application.
* **CUSTOMER MODULE:**
  + - **Customer- registration :** Provides a interface for a new user to register with application, and generates an account number and password at the same time.
    - **Creates- databases :** After registration of every user ,add his/her details to the database(text file) and creates separate database for each bank.
    - **Mall-registration:** Creates a interface for the malls to register in the application and generates VMT code while registration.
    - **Session password generation:** A special type of algorithm that creates a session password every time before the transaction, using the initial password given during customer registration.

**4. ANALYSIS AND DESIGN**

**4.1 ANALYSIS**

The development of a computer-based application includes a systems analysis phase which produces or enhances the data model which itself is a precursor. There are a number of different approaches to system analysis. When a computer-based application is developed, systems analysis (according to the Waterfall model) would constitute the following steps:

* The development of a feasibility study, involving determining whether a project is economically, socially, technologically and organizationally feasible.
* Conducting fact-finding measures, designed to ascertain the requirements of the system's end-users. These typically span interviews, questionnaires, or visual observations of work on the existing system.
* Gauging how the end-users would operate the system (in terms of general experience in using computer hardware or software), what the system would be used for and so on.

Another view outlines a phased approach to the process. This approach breaks systems analysis into 5 phases:

* Scope Definition
* Problem analysis
* Requirements analysis
* Logical design
* Decision analysis

**Use Case Diagram:**

Use cases are a widely-used systems analysis modeling tool for identifying and expressing the functional requirements of a system. Each use case is a business scenario or event for which the system must provide a defined response. Use cases evolved out of object-oriented analysis; however, their use as a modeling tool has become common in many other methodologies for system analysis and design.

**Fig 4.1 USE CASE DIAGRAM**

****

**Fig 4.2 ACTIVITY DIAGRAM**

****

The activity diagram shows the flow of interaction or sequence of interaction of the customer and application .Customer is requested to select the bank first and then enter his login details which were issued by bank during his registration .After verification of his/her details customer is requested to select shopping mall and enter VMT-code of that mall .after this customer has to enter bill number ,bill amount and current date .Date is verified and then he/she has to enter session password which is generated freshly before every transaction using initial password generated during registration .After verification of session password ,transaction is completed.

**DATA FLOW DIAGRAMS:**

A  **data flow diagram** is a graphical representation of the "flow" of data through an information system, modeling its process aspects. The DFDs describe the input-process-output view of the software. The Level 0 DFD (Context DFD) shows the major external entities in application. The Level 1 DFD refines the software system abstraction and describes the detail of processing the input information to produce the output.

****

Fig 4.3 DFD level 0

**** Fig 4.4 DFD level 1

**STATE DIAGRAMS:**

State diagrams are used to give an abstract description of the behavior of a system . This behavior is analyzed and represented in series of events, that could occur in one or more possible states. Hereby "each diagram usually represents objects of a single class and track the different states of its objects through the system".

**** Fig 4.5 State diagram

**4.2 DESIGN**

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. If the broader topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development,"[9] then design is the act of taking the marketing information and creating the design of the product to be manufactured.

* Requirements analysis - analyzes the needs of the end users or customers.
* Benchmarking - is an effort to evaluate how current systems perform.
* Systems architecture - creates a blueprint for the design with the necessary specifications for the hardware, software, people and data resources.
* Design - designers will produce one or more 'models' of what they see a system eventually looking like, with ideas from the analysis section either used or discarded. A document will be produced with a description of the system.

Fig 4.2 Flowchart for virtual-money transfer

**5. SYSTEM REQUIREMENTS**

* 1. **FUNCTIONAL REQUIREMENTS**

The system should provide a better and an efficient way of making payments rather than the existing method of using credit/debit cards. The system must provide the basic functionality of providing a secured way of making payments.

* allows the new user to register in the application
* should store the registered user's details in database(text files)
* allowing user to select the bank on behalf of which he/she wishes to select the bank
* allowing user to enter his application's account details like account number, dob, password
* verification of entered details
* selection of shopping mall and allowing user to enter the corresponding VMT code for the all and performing its verification
* asking user to enter the bill amt details and later asking to enter session password followed by its verification and finally after verification of session password transaction/payment comes to an end.

**5.1.1 HARWARE REQUIREMENTS**

Processor : Pentium 2 266 MHz processor

Memory : 512 MB

**5.1.2 SOFTWARE REQUIREMENTS**

Operating System : Windows 7/ XP/ vista

Programming Language or Tools : Java

Java version :jdk 1.6.0\_45

**5.2 NON FUNCTIONAL REQUIREMENTS**

* The application must prevent unregistered or unauthorized users for using the application.
* If anywhere wrong or irrelevant details are entered the application must prevent the control to move ahead.
* If any impossible transactions/payments are tried to made the application must prevent it from happening.
* Each time a fresh and new session password has to be generated which must not be easily traceable.
* The application must be able to work efficiently in spite any number and any type of inputs. The application must be able to show the status of accounts of both customer's and shopping malls after the payments are done.
* The application must also be portable .

**6. IMPLEMENTATION**

* 1. **BASIC APPROACH**

In this we have three programs namely the customer registration, shopping mall registration and application program. First in the customer registration program a new user is registered. First the user enters the name, dob, sex, mobile number, account balance, bank account number on behalf of which bank he/she wishes to register. Then the details are added to the corresponding text file . The user has to enter his/her details in the specified format only, and here the customer who is going to be registered is assumed to be having a bank account with some minimum assumed balance. These details are written into the corresponding text files in a fixed pattern. Similarly registration of malls is also done through the second program and the corresponding details are written into text files.

This project implements a to implement a secured software application for the shopping malls which allows the e-transactions without the use of any credit/debit cards . Generally such e-transactions are carried out using credit/debit cards. Though these methods are successful and have been prevalent throughout the world since many years. But now everything in the world is turning towards eco friendly options. So in the place of these cards my mini-project implements an alternative way of virtually transferring the money when required. Moreover this method also saves a lot of economic investment done during the manufacturing of cards, and also saves lot of time.

The application has complete control over all the transactions i.e.; payments . It verifies the details entered by the user and i.e. ; checks whether the details entered by the user are authorized and registered in the application's database or not. It takes the name of the shopping mall and the VMT code entered by the user and verifies whether the code matches with the mall or not. Later it takes the bill amount and bill number and session password and checks the conditions like whether sufficient balance is present in the user's account or not and whether the session password entered by the user is correct or not. After performing all the checks the application moves to the further stage of completing the transaction.

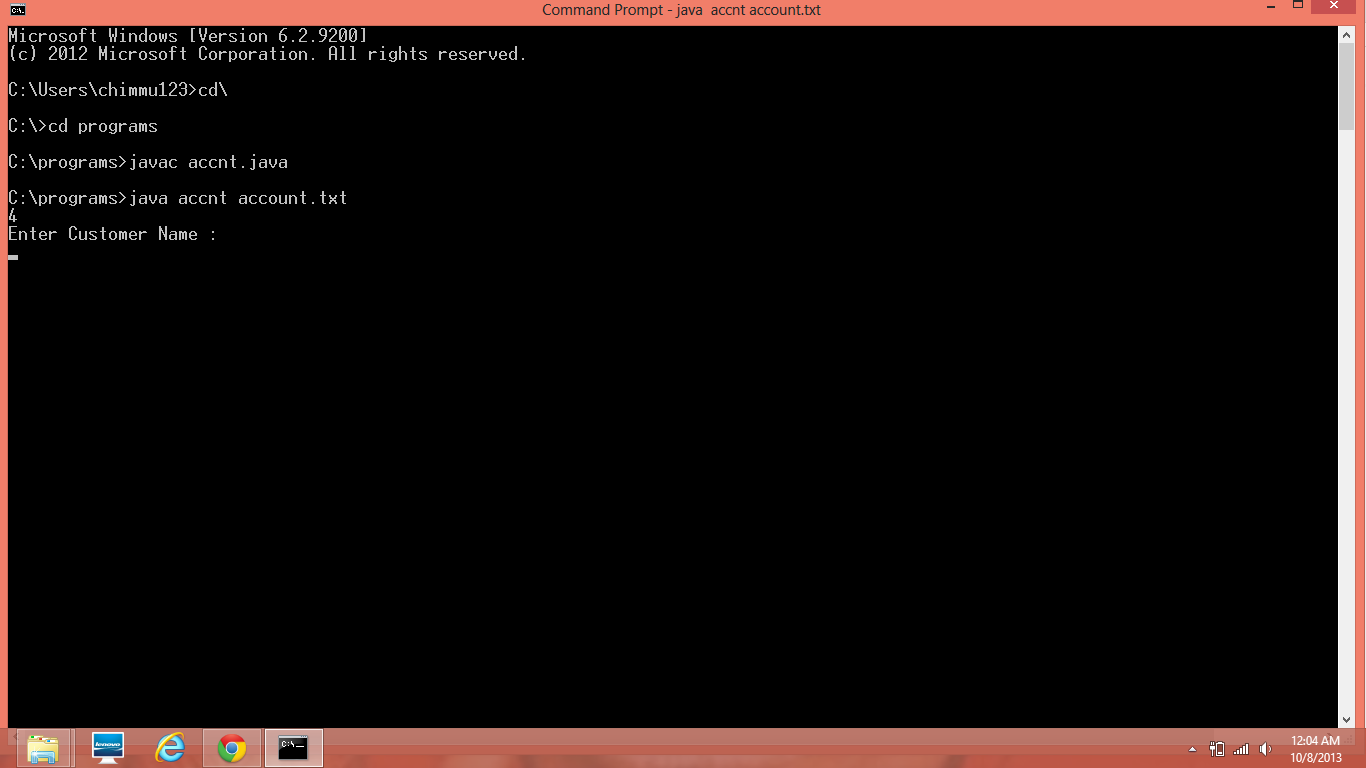
**6.2 CONTROL FLOW**

Control flow (or alternatively, flow of control) refers to the order in which the individual statements, instructions or function calls of an imperative or a declarative program are executed or evaluated. Within an imperative programming language, a **control flow statement** is a statement whose execution results in a choice being made as to which of two or more paths should be followed. For non-strict functional languages, functions and language constructs exist to achieve the same result, but they are not necessarily called control flow statements.

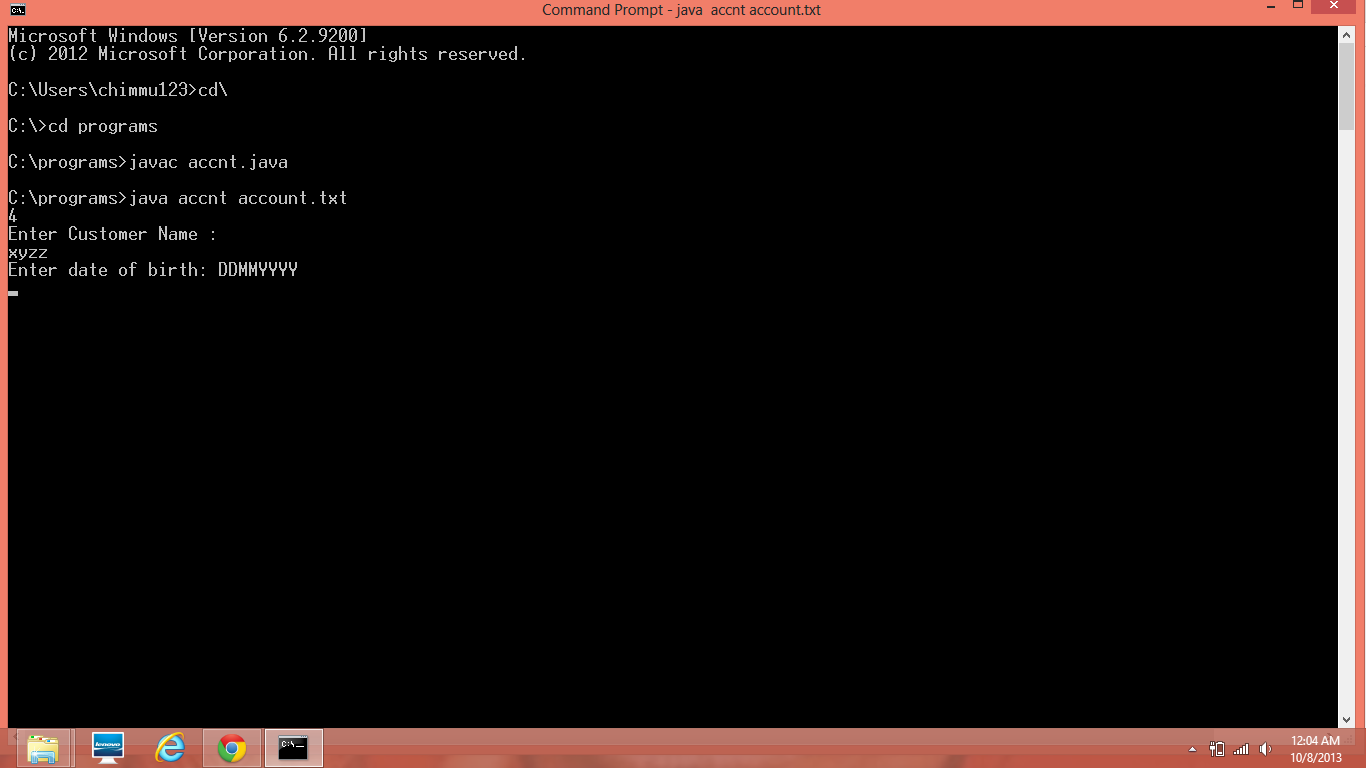
The kinds of control flow statements supported by different languages vary, but can be categorized by their effect:

* continuation at a different statement (unconditional branch or jump),
* executing a set of statements only if some condition is met (choice - i.e., conditional branch),
* executing a set of statements zero or more times, until some condition is met (i.e., loop - the same as conditional branch),
* executing a set of distant statements, after which the flow of control usually returns (subroutines, co routines, and continuations),
* stopping the program, preventing any further execution (unconditional halt).

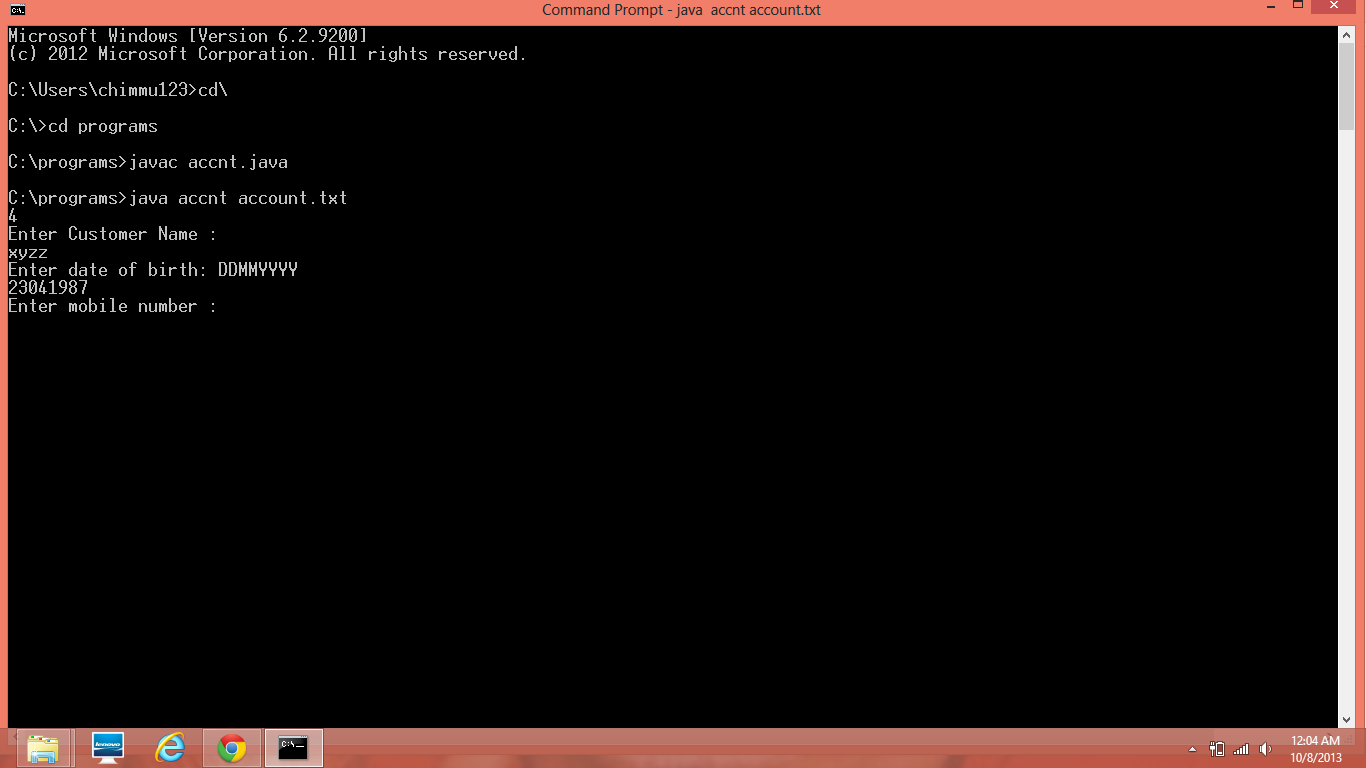
**7. RESULTS**



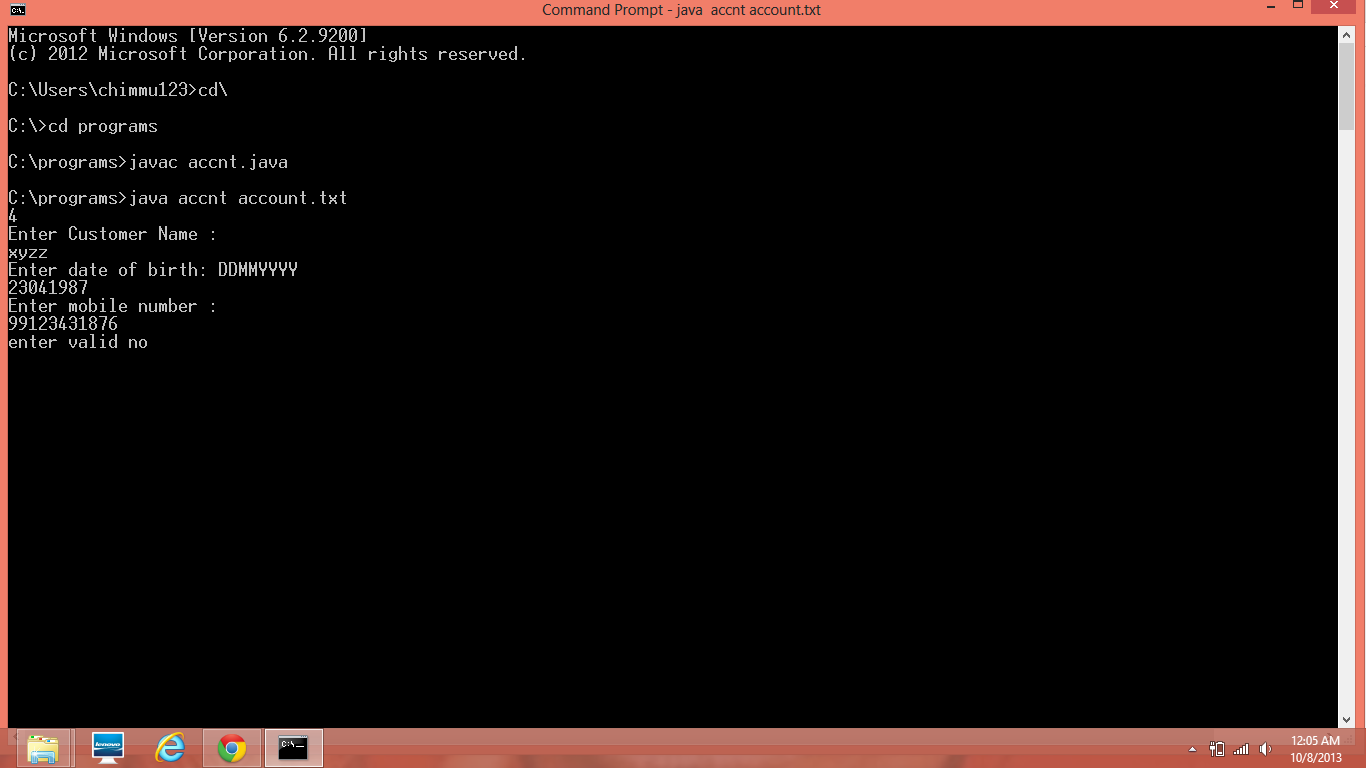
ScreenShot 7.1(a) Customer registration



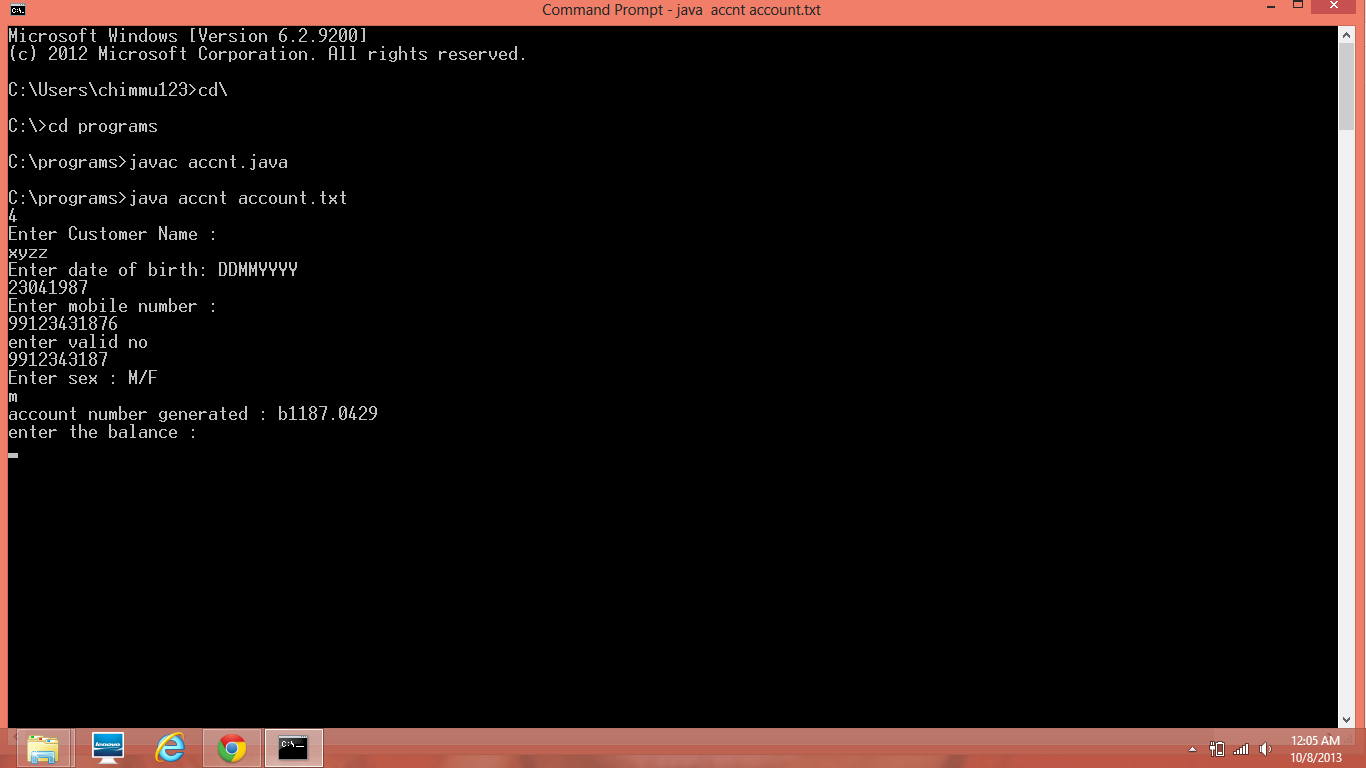
ScreenShot 7.1(b) Adding DOB of customer



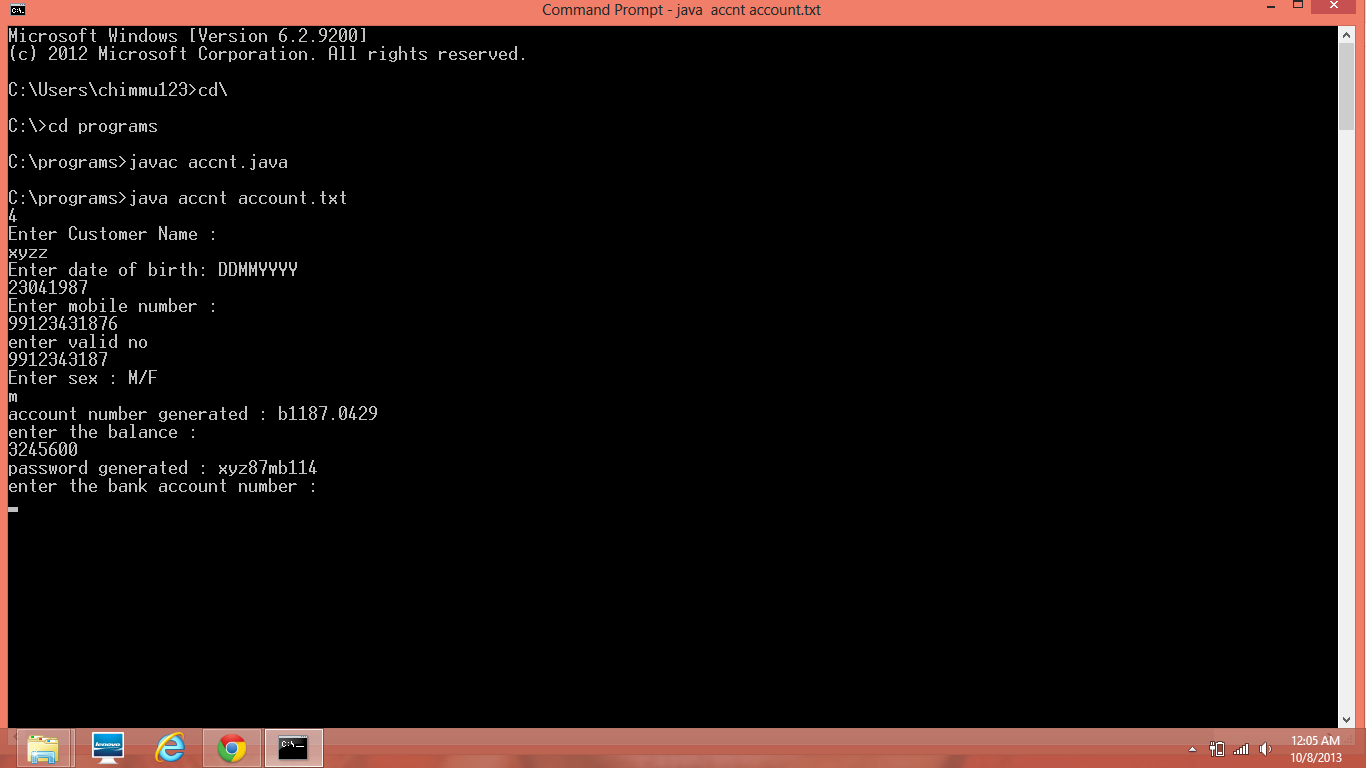
ScreenShot 7.1(c) Adding customer's mobile number



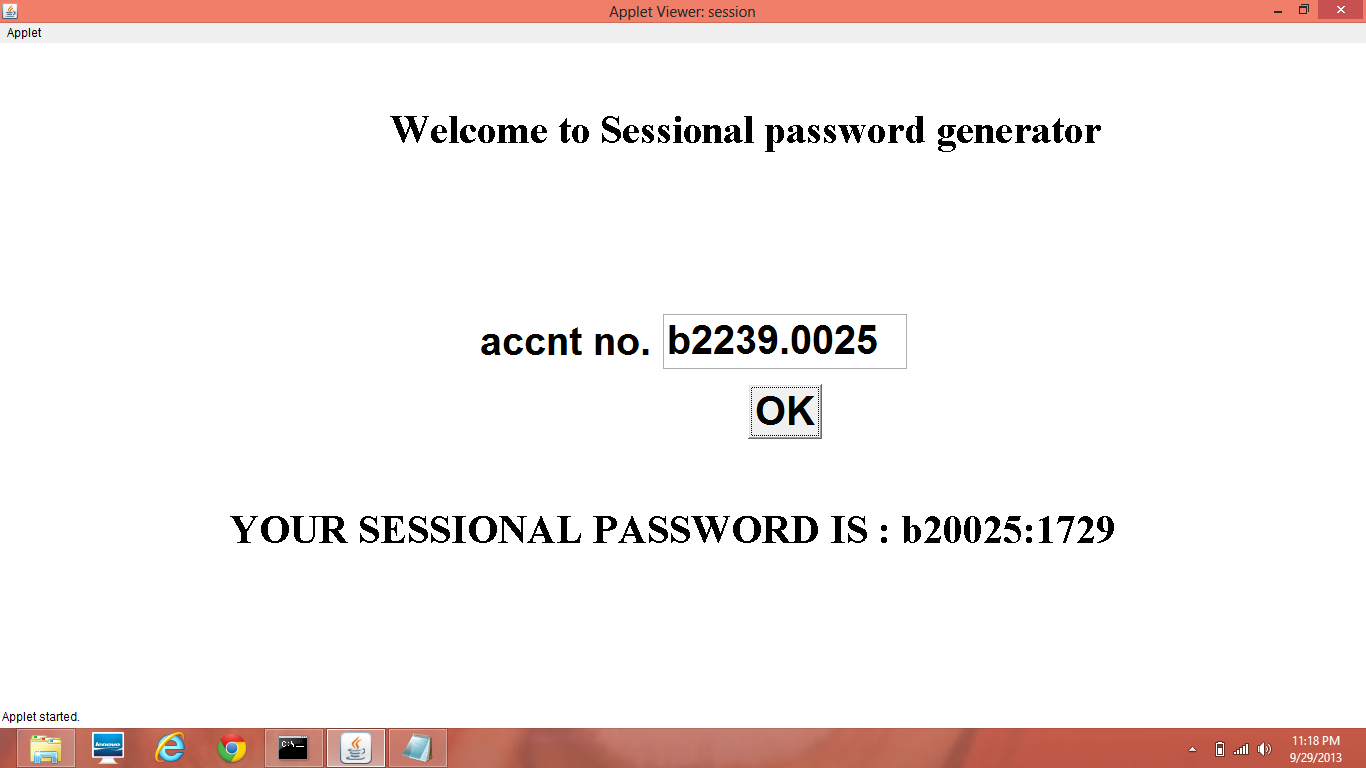
ScreenShot 7.1(d) Entering a valid mobile number



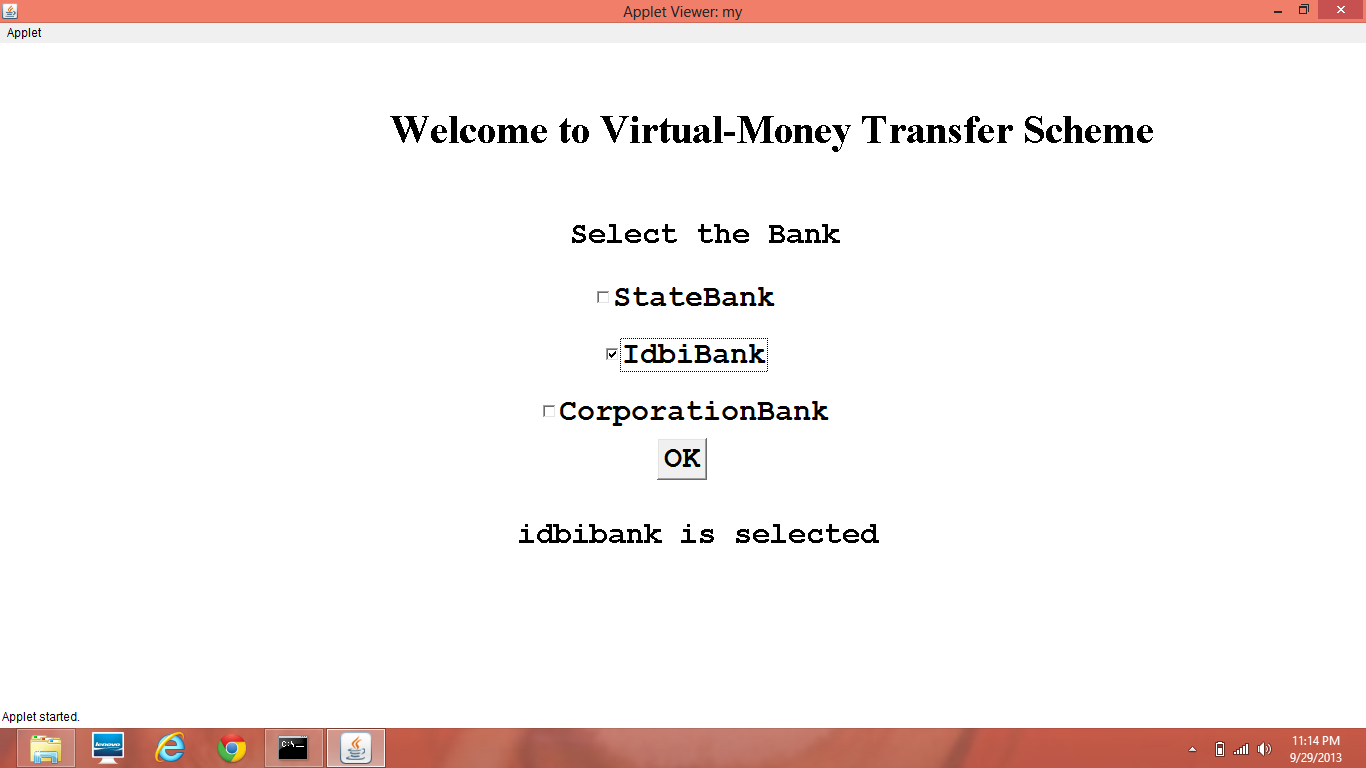
Screenshot 7.1(e) Entering the balance of bank account and application account number generated



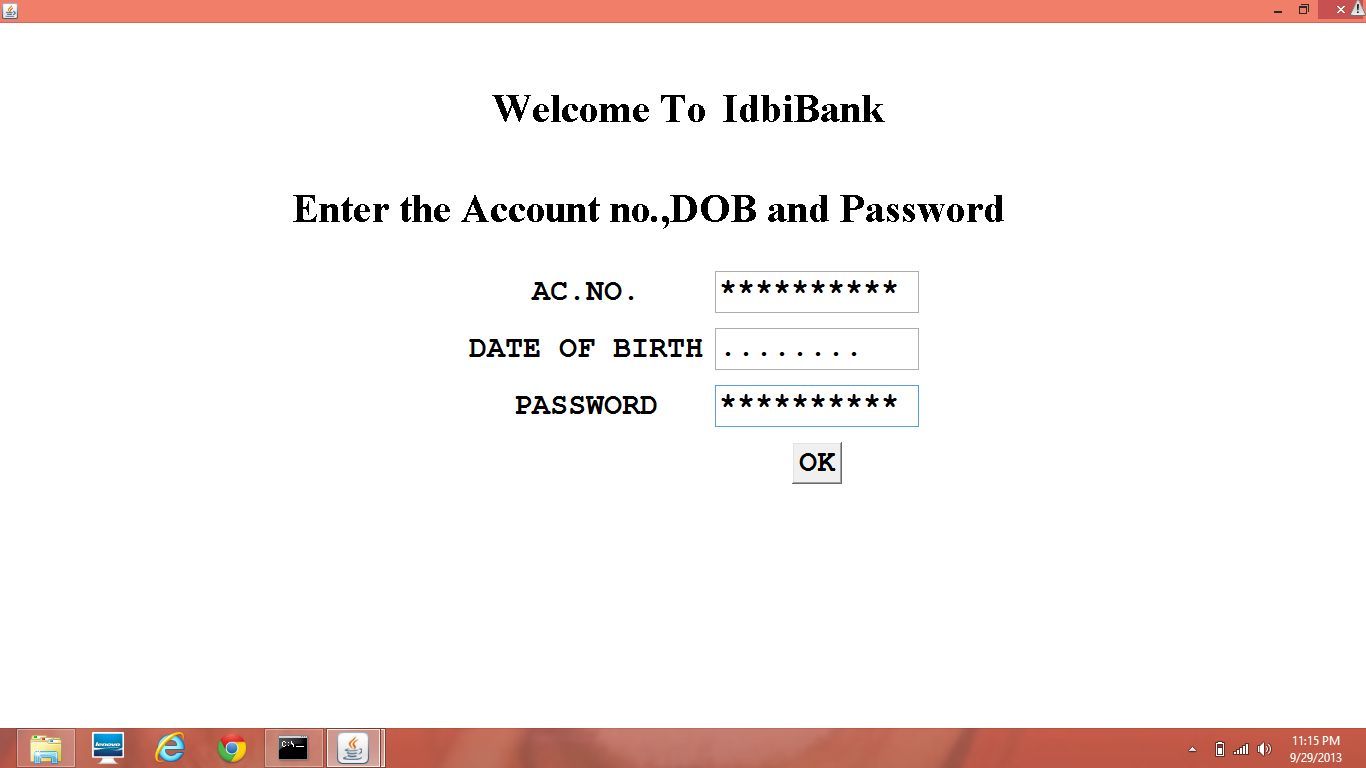
ScreenShot 7.1(f) Entering bank account number and application's password generated



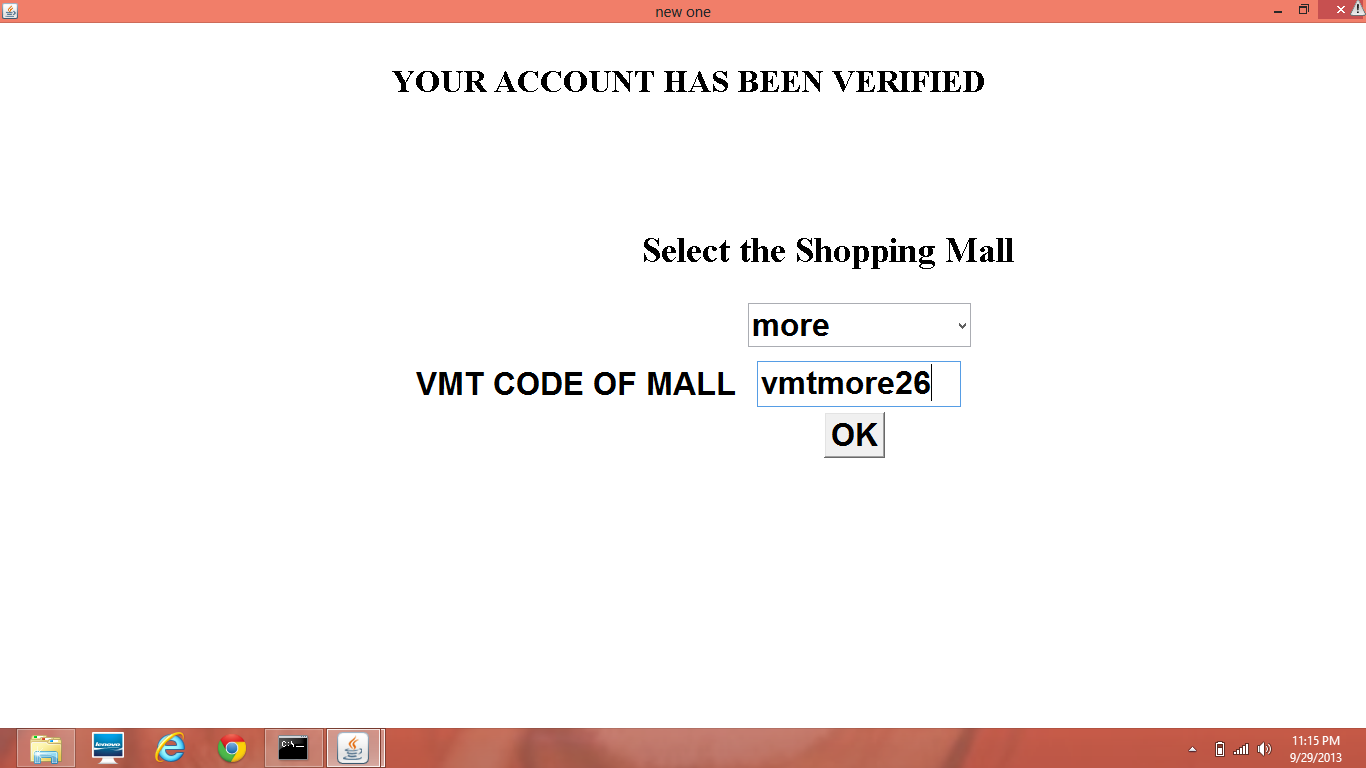
ScreenShot 7.2 Session password generated



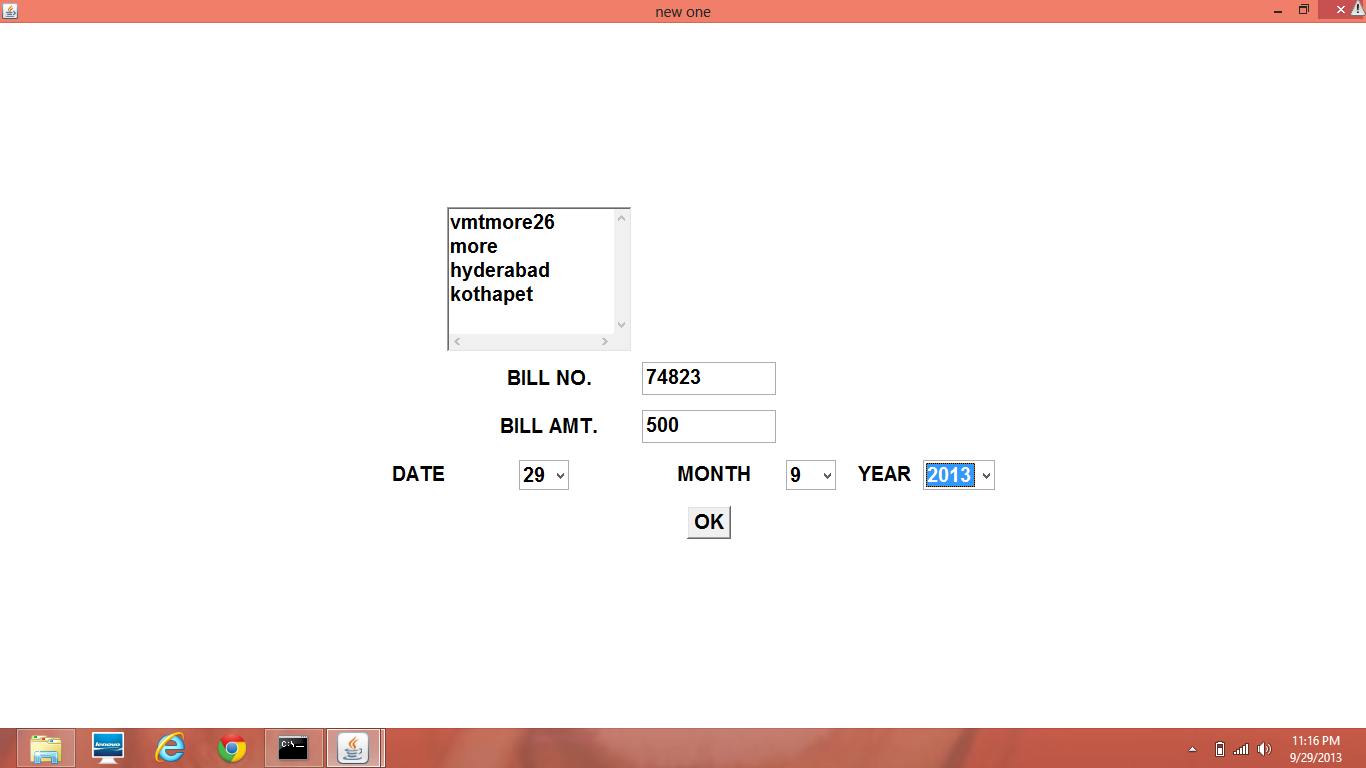
ScreenShot 7.3 Entering into application by selecting the bank



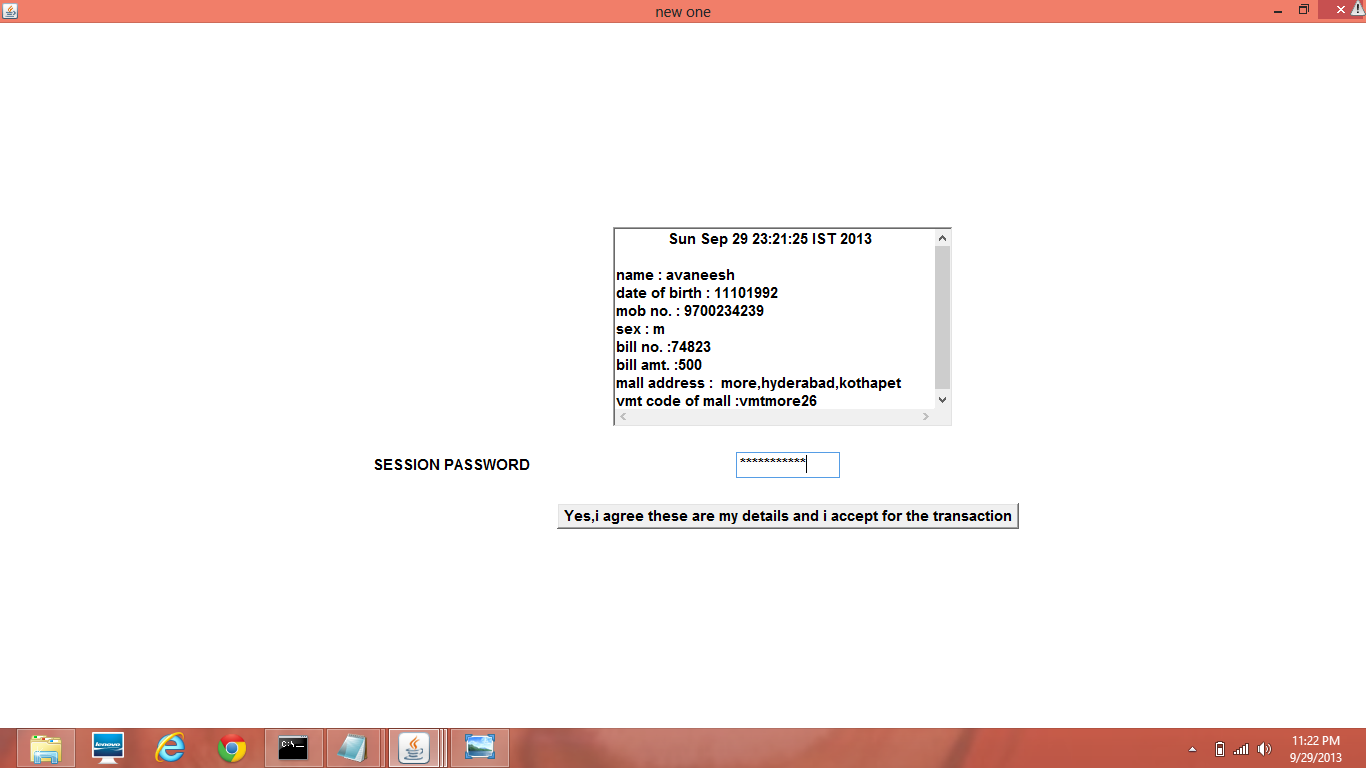
ScreenShot 7.4 Entering the details to begin the application



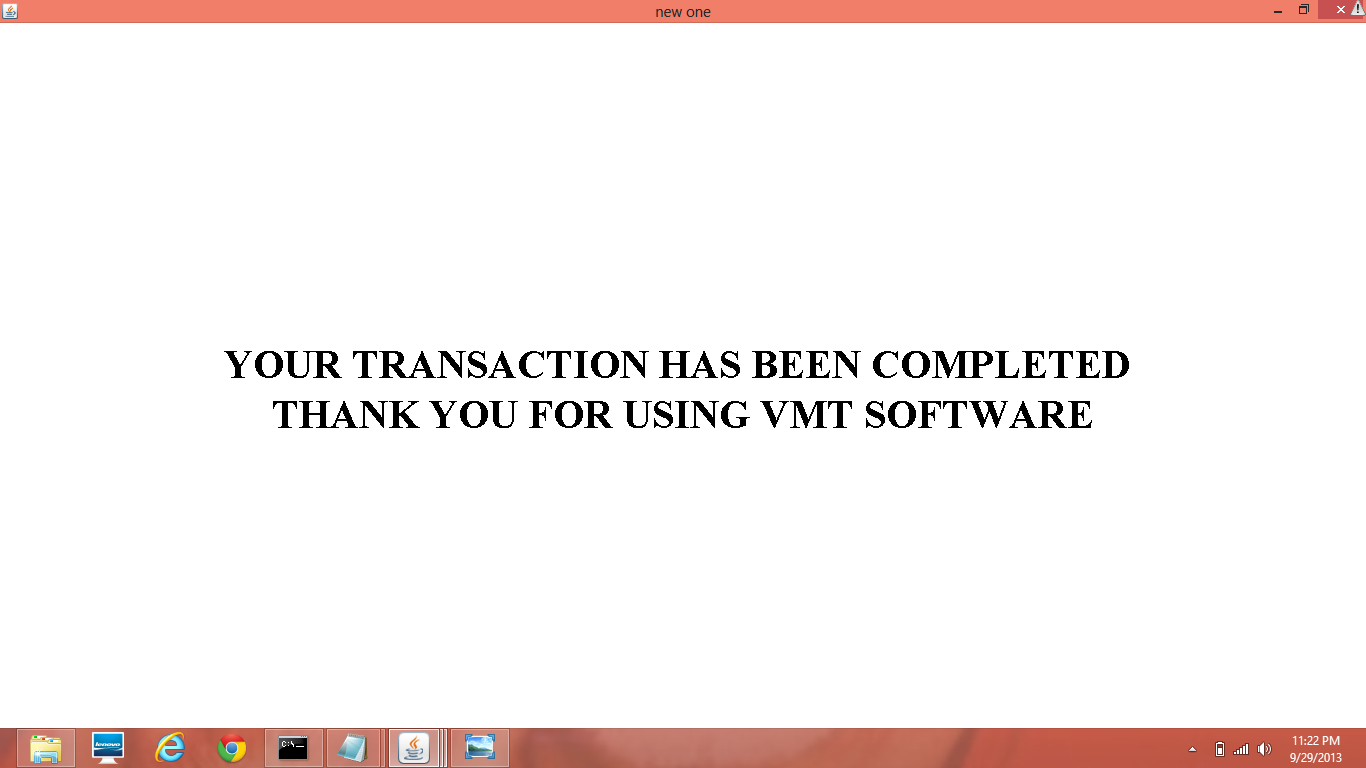
ScreenShot 7.5 Selecting shopping mall and entering its VMT code



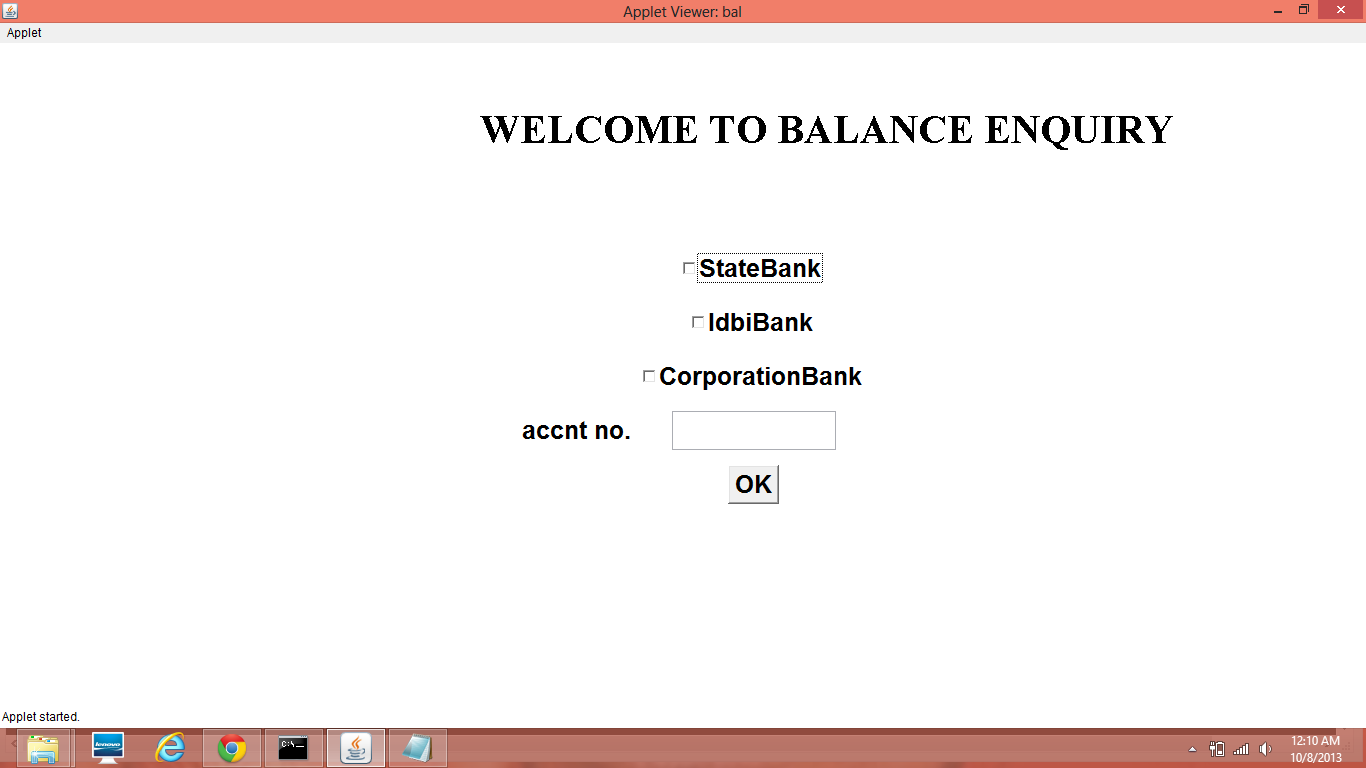
ScreenShot 7.6 Entering bill no. and bill amt and selecting the date of shopping



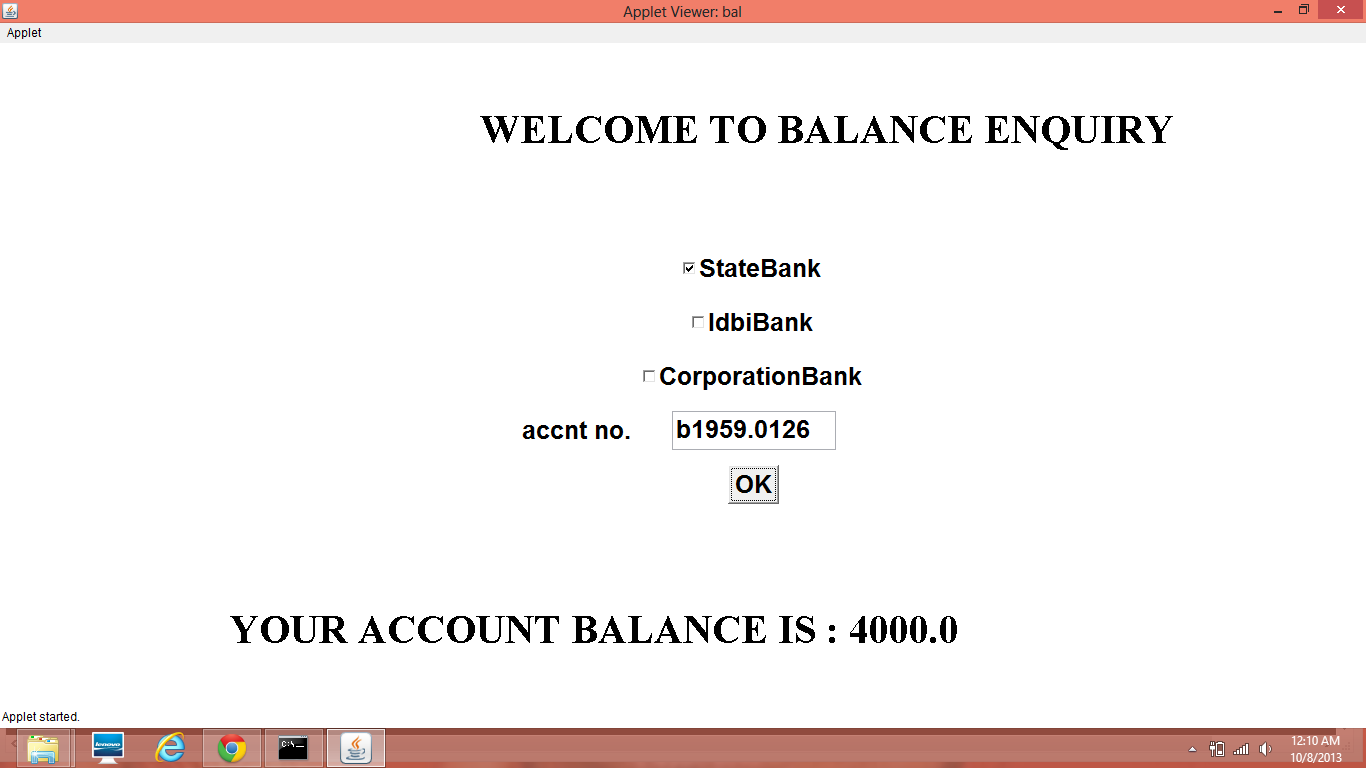
ScreenShot 7.7 Entering the session password and finalizing the transaction after checking the details displayed



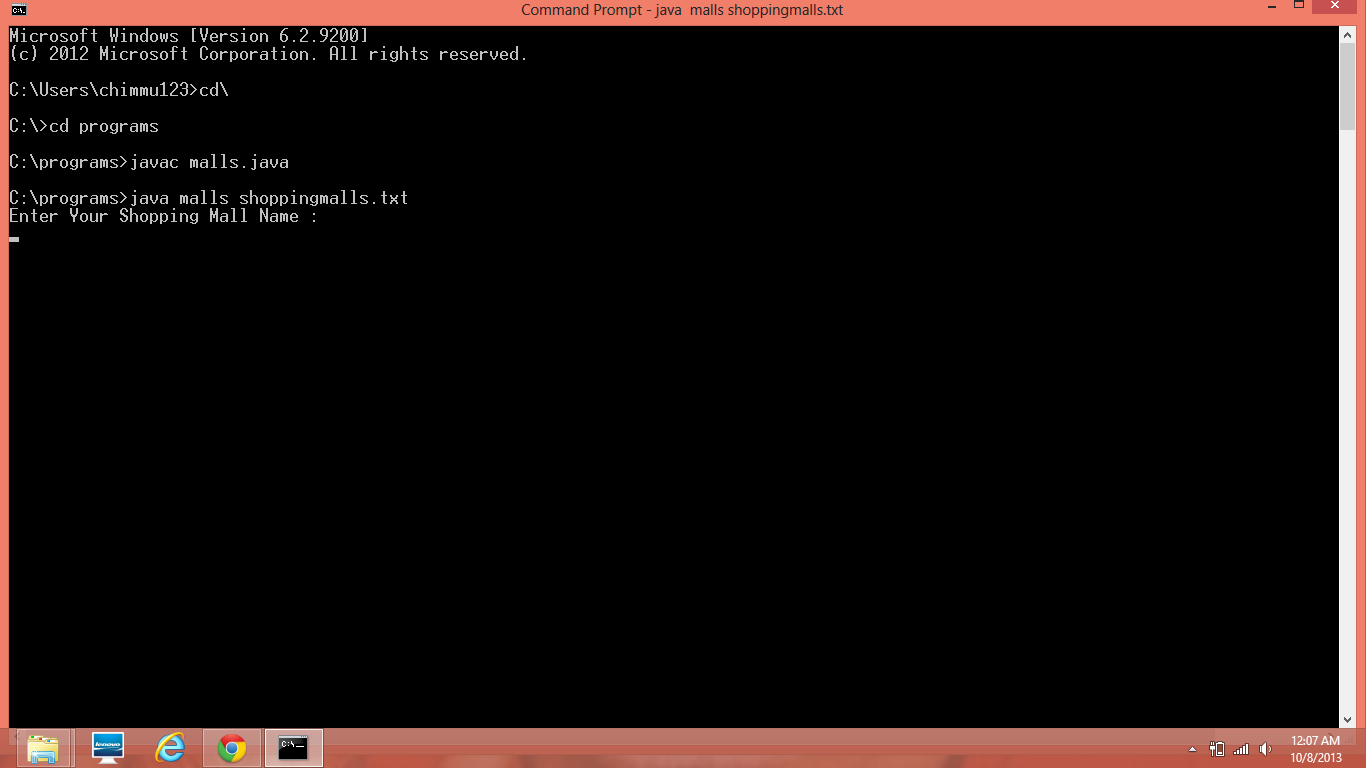
ScreenShot 7.8 Transaction completed



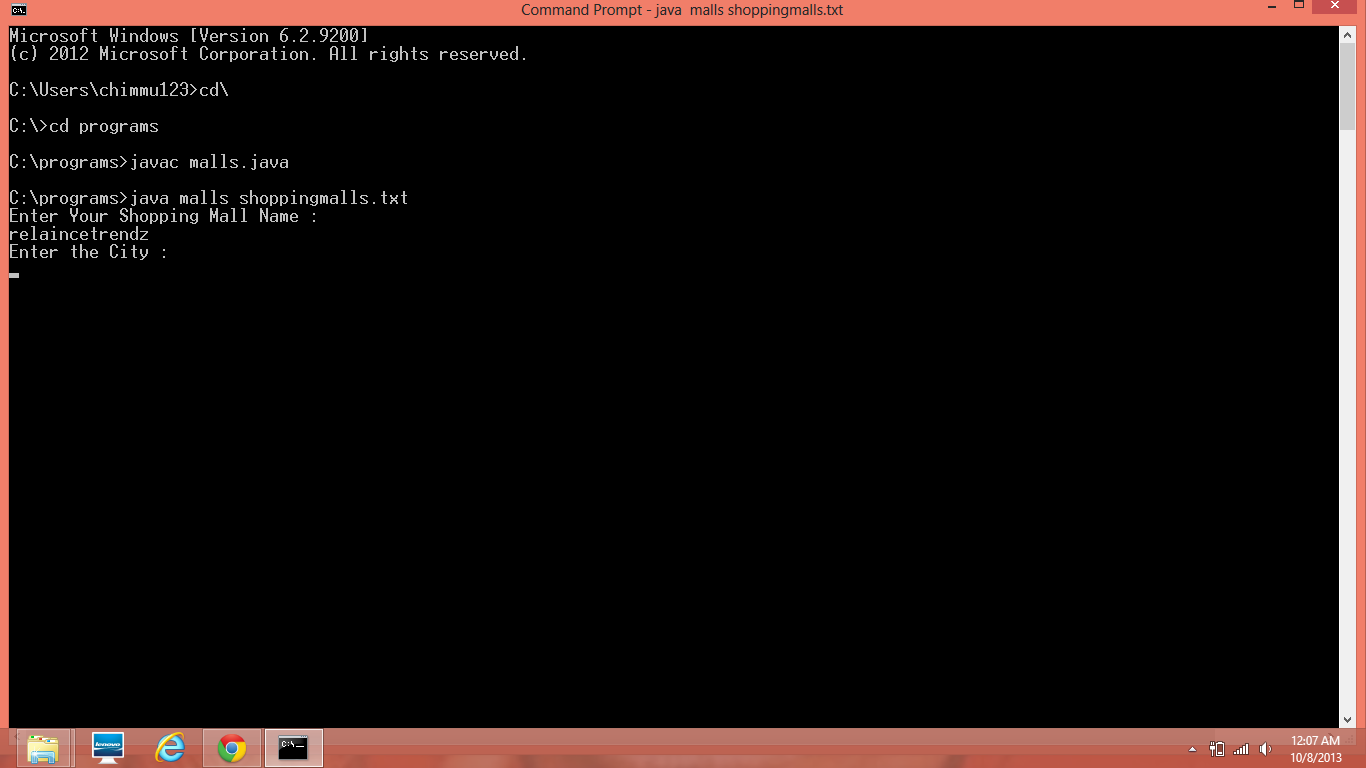
ScreenShot 7.9 Balance enquiry



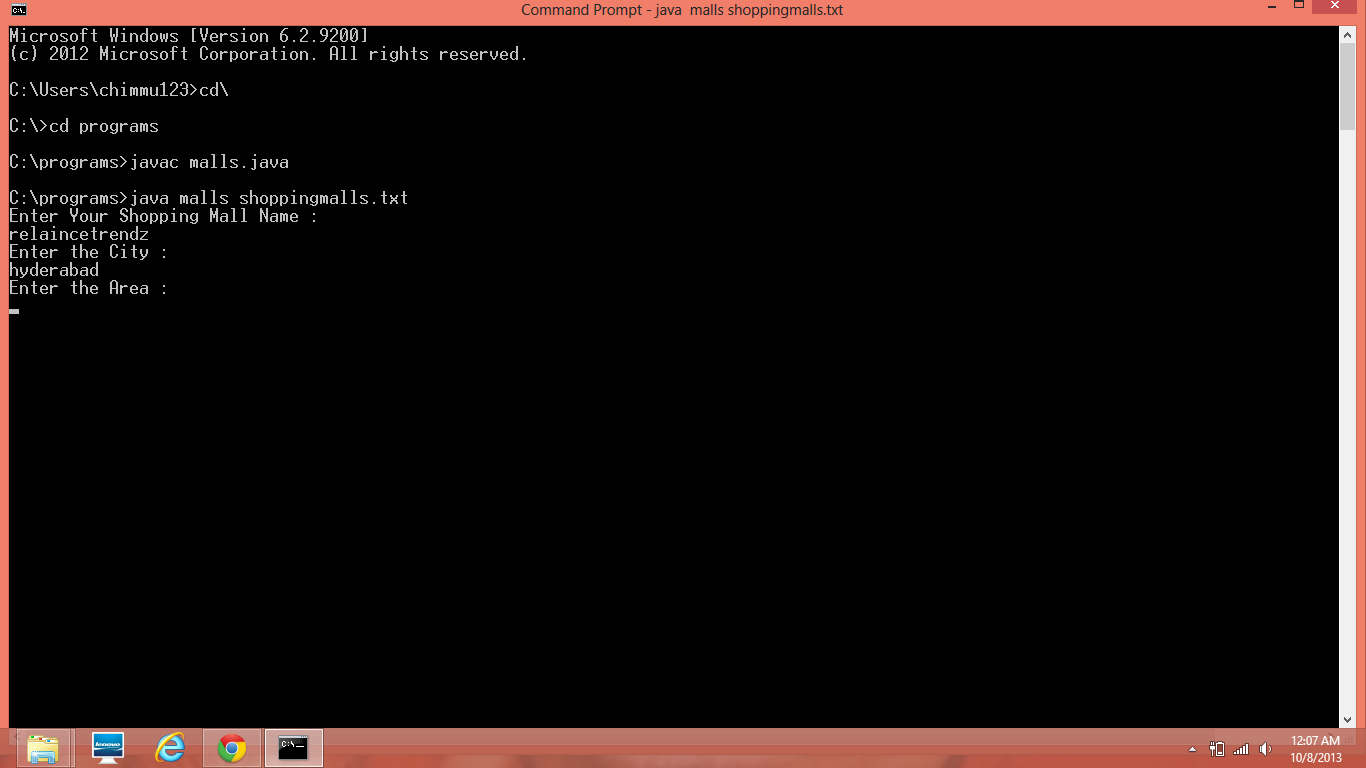
ScreenShot 7.10 Current balance



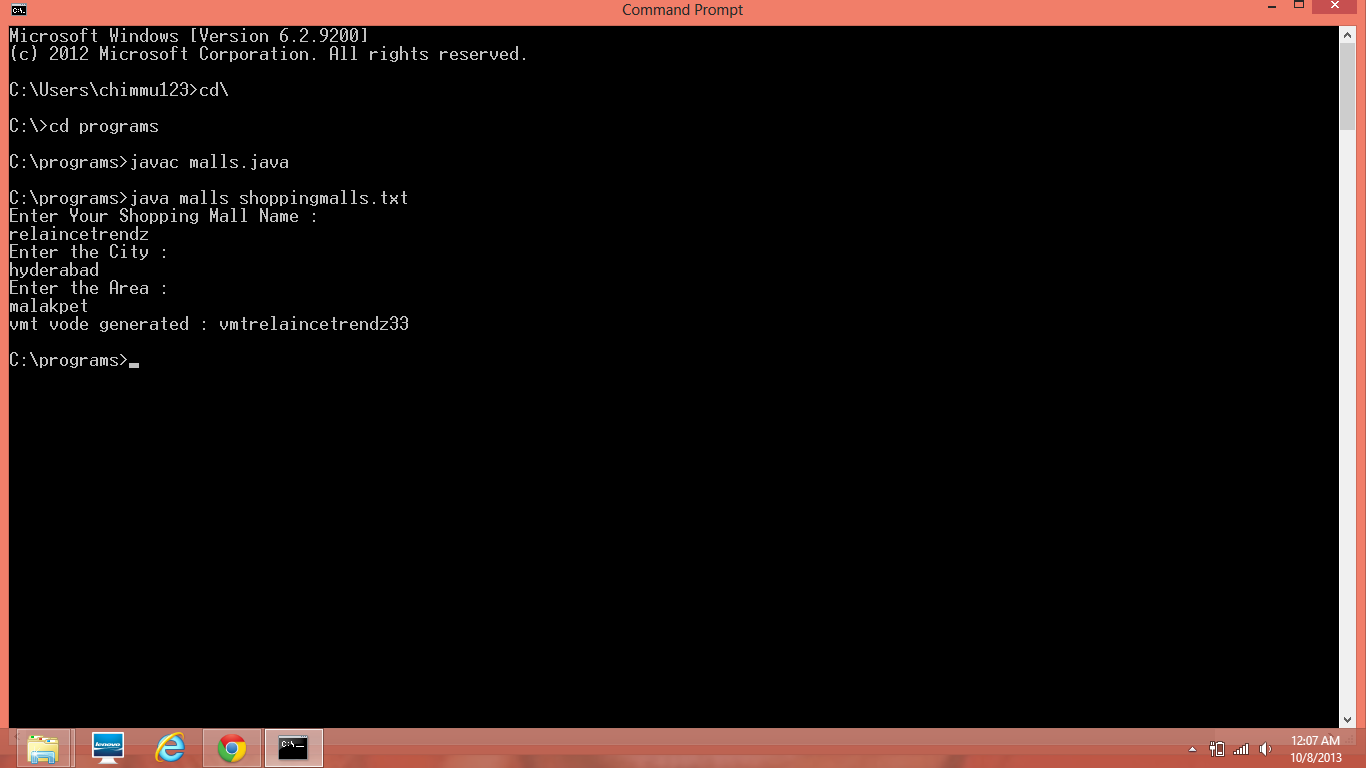
ScreenShot 7.11 Registration of new shopping mall with the application



ScreenShot 7.12 Entering the city in which mall is located



ScreenShot 7.13 Entering the location of the mall



ScreenShot 7.14 VMT code generated for the mall

**CONCLUSION**

This application is simple to use and can be even used by naïve users. As stated earlier VIRTUAL-MONEY TRANSFER 'S main idea is to implement a secured software application for the shopping malls which allows the e-transactions without the use of any credit/debit cards , my applications allows to perform the payments virtually without help of any cards. Initially each registered customer is given an account number and password using which a registered customer can start vmt application. At the further stages the user is required to get a session password using his/her account number to make the payment. For security reasons, each time the customer performs a transaction a fresh and new session password is generated using which he/she can make the payments. So as mentioned in the primary objectives of my project is designed and aimed at providing a safe and secured way of making payments virtually without any use of credit/debit cards which fulfills another important idea of reducing the use of plastic and also money.

**Limitations**

Each and every user who is going to register in this application should have a bank account with minimum assumed balance to perform at least one payment. No algorithms have been used in generating the session and normal passwords. Therefore they might be traceable which raises a question of security. since text files are used as database they have their own disadvantages like redundancy, isolation, inconsistency etc.

**Future Work**

This project can be further extended by adding few more sophisticated features to the VIRTUAL-MONEY TRANSFER like, this can be implemented using GUI and can be made as a web based application. Certain password or unique key generating algorithms can be used to generate the passwords that are more consistent and free from tracing. This application can be made more efficient by avoiding conventional file system and using database for storing and retrieving customer's data.

**REFERENCES**

[1] Herbert Schildt, The Complete Reference JAVA, Tata McGraw-Hill Edition,2007

[2] Big Java by Cay Horstmann, John Wiley and Sons, 2nd Edition

[3] Core Java 2, Vol 1, Fundamentals by Cay.S.Horstmann and Gary Cornell, Pearson Education, Seventh Edition

[4] Beginning in Java 2 by Iver Horton, Wrox Publications.

[5] http://www.llamagraphics.com/resources/documentation/macintosh/.html

[6] http://docs.oracle.com/javase/1.4.2/docs/.html

[7] http://www.sun.com

[8] http://www.w3schools.com

[9] http://www.google.com

[10] http://www.webopedia.com