Abstract

The automated teller machine, or ATM, is such a complicated piece of technology that it does not have a single inventor. Instead, the ATMs we use today are an amalgam of several different inventions. Some of these proto-ATMs dispensed cash but did not accept deposits, for example, while others accepted deposits but did not dispense cash. Today's ATMs are sophisticated computers that can do almost anything a human bank teller can, and have ushered in a new era of self-service in banking. Today the money transactions in ATM are done with Cards, and Net banking is done form account to account.

In the proposed system we are introducing an new feature that is we can allow the person to withdraw the amount from ATM even if he/she has no bank account. In this application the account holder will be sending the request to server to allow the withdrawal of the amount from the ATM by using the mobile. Here

the user will be sending and request to the server with the withdrawer mobile number and amount to be withdrawn, then the server sends an message to the person after the confirmation is done by the account holder.

To withdraw the amount the person has to visit the ATM and enter his mobile number. Then an OTP is generated and sent to the withdrawer, the withdrawer has to put the OTP and then withdraw the amount.

The most important feature in this application is that the withdrawer can only withdraw the amount which has been set the account holder and even the original balance is hidden from the withdrawer.

INTRODUCTION

INTRODUCTION

An Automated Teller Machine (ATM) allows customers to perform banking transactions anywhere and at anytime without the need of human teller. By using a debit or ATM card at an ATM, individuals can withdraw cash from checking or savings accounts, make a deposit or transfer money from one account to another or perform other functions. You can also get cash advances using a credit card at an ATM. Individuals should be aware that many banks charge transaction fees – generally ranging from Rs 50-150 per transaction -for using another bank's

SCOPE
We have introduced a Card less cash withdrawal facility which allows customers to transfer money from their account to anyone in India with a mobile number.

We foresee tremendous growth potential in the use of electronic payments in our country. 'Instant Money Transfer and Withdrawal without Bank Account ' provides an added facility in an array of electronic payment options that Bank offers to its customers."

The sender first needs to register the recipient's name, mobile number and address. The sender will get a four-digit verification code while the recipient a six digit reference code, over SMS. The recipient can withdraw cash from ATM by entering the mobile number, cash amount along with the verification and reference code.

This service can also be used by the Bank's account holders to withdraw cash from their own accounts without using a debit card.

Existing System:

Today, there are almost 2 million ATMs around the globe. Although use of the machines has declined in recent years, likely because more people make purchases using credit and debit cards instead of cash, the ATM continues to have a place in modern culture. Today's machines sell everything from airline tickets to movie tickets to medicine.

The automated teller machine, or ATM, is such a complicated piece of technology that it does not have a single inventor. Instead, the ATMs we use today are an amalgam of several different inventions. Some of these proto-ATMs dispensed cash but did not accept deposits, for example, while others accepted deposits but did not dispense cash. Today's ATMs are sophisticated computers that can do almost anything a human bank teller can, and have ushered in a new era of self-service in banking.

Today the money transactions in ATM are done with Cards, and Net banking is done form account to account.

Features:

- In the proposed system we are introducing an new feature that is we can allow the person to withdraw the amount from ATM even if he/she has no bank account.
- In this application the account holder will be sending the request to server to allow the withdrawal of the amount from the ATM by using the mobile. Here
- the user will be sending and request to the server with the withdrawer mobile number and amount to be withdrawn, then the server sends an message to the person after the confirmation is done by the account holder.
- To withdraw the amount the person has to visit the ATM and enter his mobile number.
- Then an OTP is generated and sent to the withdrawer, the withdrawer has to put the OTP and then withdraw the amount.
- The most important feature in this application is that the withdrawer can only withdraw the amount which has been set the account holder and even the original balance is hidden from the withdrawer.

<u>Explanation of Modules</u>

- ❖ In this project there are 4 modules.
- ❖ First, Second & Fourth modules , JAVA swing programming language is used and in Third module HTML, PHP, CSS, java script and Bootstrap has been used.

<u>MODULE 1:</u>

- ➢ The module one is RBI i.e, Reserve bank of India.
- ➢ RBI is India central banking institution, which controls the monetary policy of Indian rupee.
- ➢ RBI allows to create banks. In this project using bank master option we can create banks.
- ➢ In bank master option there some columns to fill details of bank such as Bank name, branch code, branch address, landline number, password, re-enter password and IFSC code.
- ➢ The IFSC code is Indian financial system code is of 11 character code for identifying bank and branch which an account is held . the IFSC code is used both by NFET, RTGS and IMPS finance transfer system.

- **<u>MODULE 2:</u>**
  - The module 2 is Bankers, in this module bank clerk select bank, then should enter branch code and password for creating user account in particular selected bank.
  - Here in bankers there are 2 option i.e, customer master and transaction master.
  - Using customer master option a bank clerk can create customer account in selected bank.
  - Using transaction master option, bank clerk can deposit user money in his/her account with consuming less amount of time.
  - When the transaction is successfully done, then a message is received in user mobile phone (i.e, your amount is deposited to your account and even display account number of user account).

MODULE 3:

- Basically concept is Cardless Transaction for this we have designed a website for all the users throughout India transferring money at any corner of the world with any device, operating system and Browser. The receiver can withdrawal money even if he/she doesn't have ATM card or a bank account.
- So for transferring money the user needs to go to the browser and search for Dropin.in
- Before transferring the money, the sender must have a DROPIN account and a bank account to register.
- To register with DropIn account go to the register form fill all the following details and click register button.
- After successful registration. The specified bank account will be linked to DropIn account.
- Then go to login form, login with the correct pin password make sure that you enter the correct password, because entering 3 wrong password trials will block your drop in account for next 24 hours or you need to go to your bank branch to activate it instantly.
- Once you login to drop in account go to money transfer form and to transfer the money you need to enter the receivers mobile number and the amount you want to transfer
- Click confirm, a confirmation form will open and check all the details are correct after confirming click on transfer button.
- After successful transfer of money the sender will get a message of amount deducted from your respective Bank.
- Another message will be received by the receiver with one time password OTP to withdrawal the cash from ATM with the amount up to which he/she can withdraw from ATM.

Security features :

A question arise is money secured with DropIn account?

The following are the security features:

- While the sender is attempting to login and if someone gets to know the email ID of the sender and if they try 3 wrong password trials the DropIn account will be blocked for next 24 hours other wise the DropIn account can be opened instantly.
- By visiting the sender's bank branch and the sender will get alert someone was trying to access your DropIn account if it wasn't you please change your email ID.

- While the sender is attempting to login if unauthorized gets to know the email ID and the password both of the sender.
- Later if unauthorized try to open the DropIn account, then all the details of their device will be stored and later the sender can file a FIR to the police. Using the following details of the robber such as robbers IP address, operating system, browser.
- If the sender forgot to logout the DropIn account, then his/her DropIn account will be logged out automatically and if there is 3 minutes of inactivity in the account.
- If an authorized person opens the browser in which the DropIn account was logged in, but the DropIn account appears to be logged out.
- If the sender is in hurry! then the sender just need to click to the logo of DropIn, Hence automatically the sender/user DropIn account will be logged out.

MODULE 4:

- After the sender transfer the amount to the Receiver. The Receiver receives a text message on his/ her mobile number with a onetime password(OTP) on his mobile that is temporary(OTP).
- The Receiver needs to go to the ATM for the withdrawal of money sent by the sender.
- On the interface of ATM the User needs to select the cardless option.
- Enter his/her mobile number and select continue this generates a onetime password (OTP) on his mobile that is Temporary(OTP).
- Enter the Temporary one time password (OTP) and then select continue.
- Enter the original (OTP) and select continue .
- Then select withdraw and enter the amount you want o withdraw.
- Then you will be able to take the cash with a receipt containing the details of Transaction.

REQUIREMENTS

2. REQUIREMENTS

HARDWARE REQUIREMENTS:

- **PROCESSOR** **:** 1.7GHZ or MORE
- **RAM** **:** Min 1GB or More
- **MEMORY SPACE** **:** Min 5GB

SOFTWARE REQUIREMENTS:

- **FRONT END** **:** Java Swings (ATM Side)
- **FRONT END FOR ONLINE APPLICATION** **:** HTML/CSS
- **SERVER SIDE SCRIPTING** **:** PHP
- **BACK END** **:** MYSQL
- **IDE** **:** Netbeans

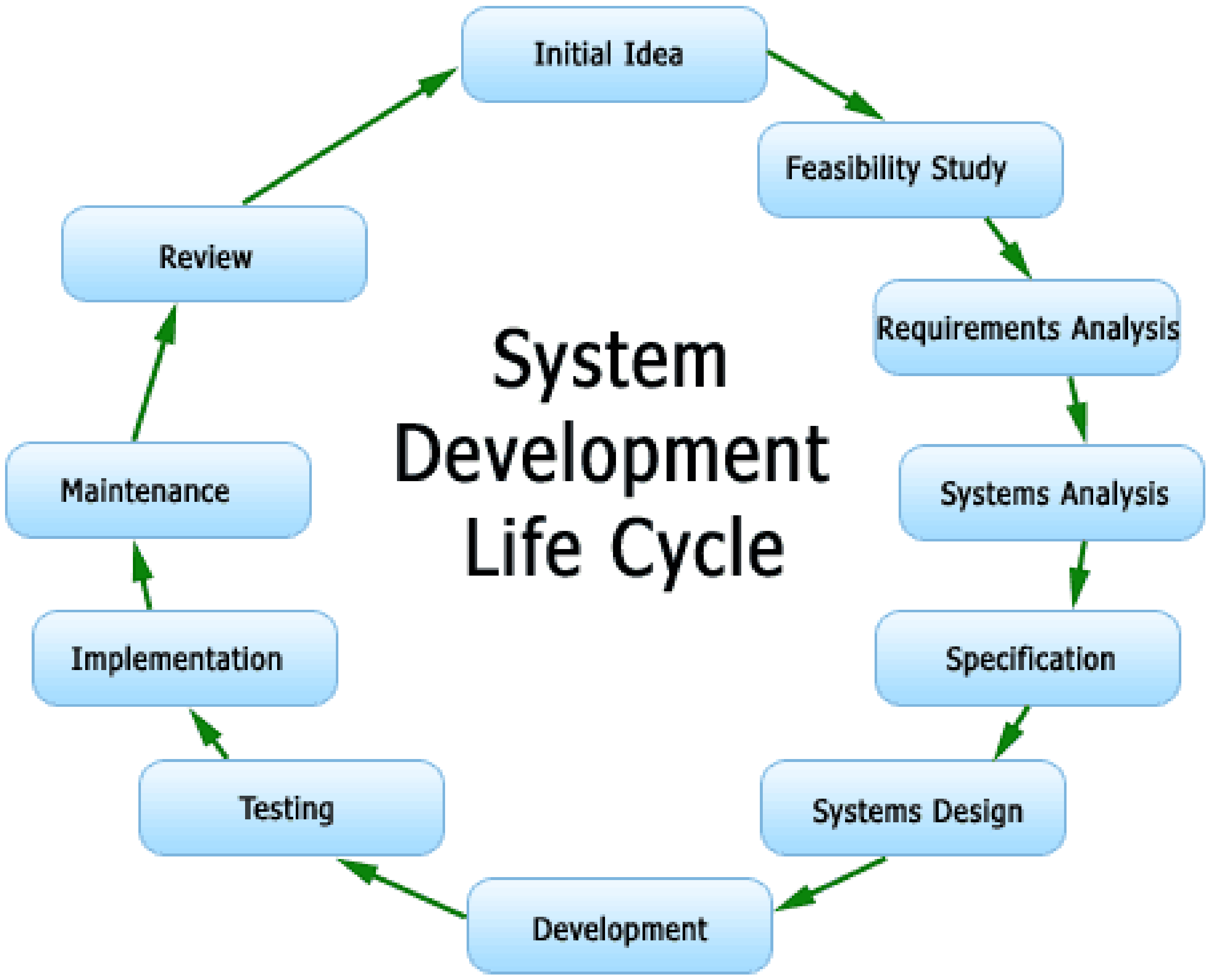
SOFTWARE DEVELOPMENT LIFE CYCLE

3. Software Development Life Cycle

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software's. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

SDLC consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific softare. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.



SDLC is a process followed for a software project, within a software organization.

Software development life cycle (**SDLC**) is a series of phases that provide a common understanding of the software building process. How the software will be realized and developed from the business understanding and requirements elicitation phase to convert these business ideas and requirements into functions and features until its usage and operation to achieve the business needs. The good software engineer should have enough knowledge on how to choose the SDLC model based on the project context and the business requirements.

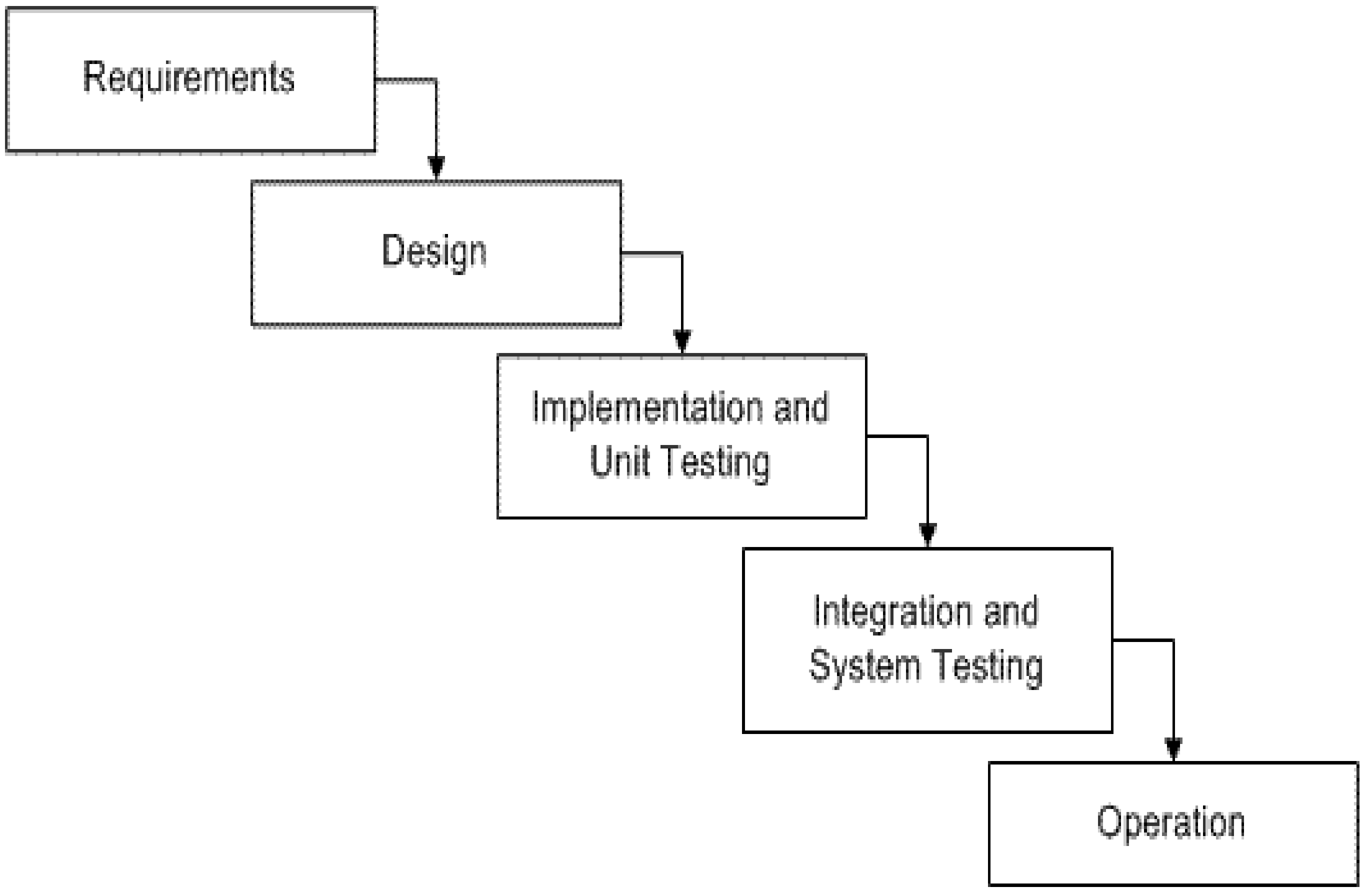
Therefore, it may be required to choose the right SDLC model according to the specific concerns and requirements of the project to ensure its success. I wrote another article on how to choose the right SDLC, you can follow this link for more information.

The System Life Cycle Is :

The period of time that starts when a software product is conceived and ends when the product is no longer available for us. The software life cycle typically includes requirements phase, design phase, implementation phase, installation and check out phase , operation and maintenance phase and sometime retirement phase.

In this project Waterfall model and Spiral model has been used.

Waterfall Model :


Requirements
Design
Implementation and Unit Testing
Integration and System Testing
Operation

The Waterfall Model was first Process Model to be introduced. It is very simple to understand and use. In a Waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. Waterfall model is the earliest SDLC approach that was used for software development.

In “The Waterfall” approach, the whole process of software development is divided into separate phases. The outcome of one phase acts as the input for the next phase sequentially. This means that any phase in the development process begins only if the previous phase is complete. The waterfall model is a sequential design process in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance.

As the Waterfall Model illustrates the software development process in a linear sequential flow; hence it is also referred to as a Linear-Sequential Life Cycle Model**.**

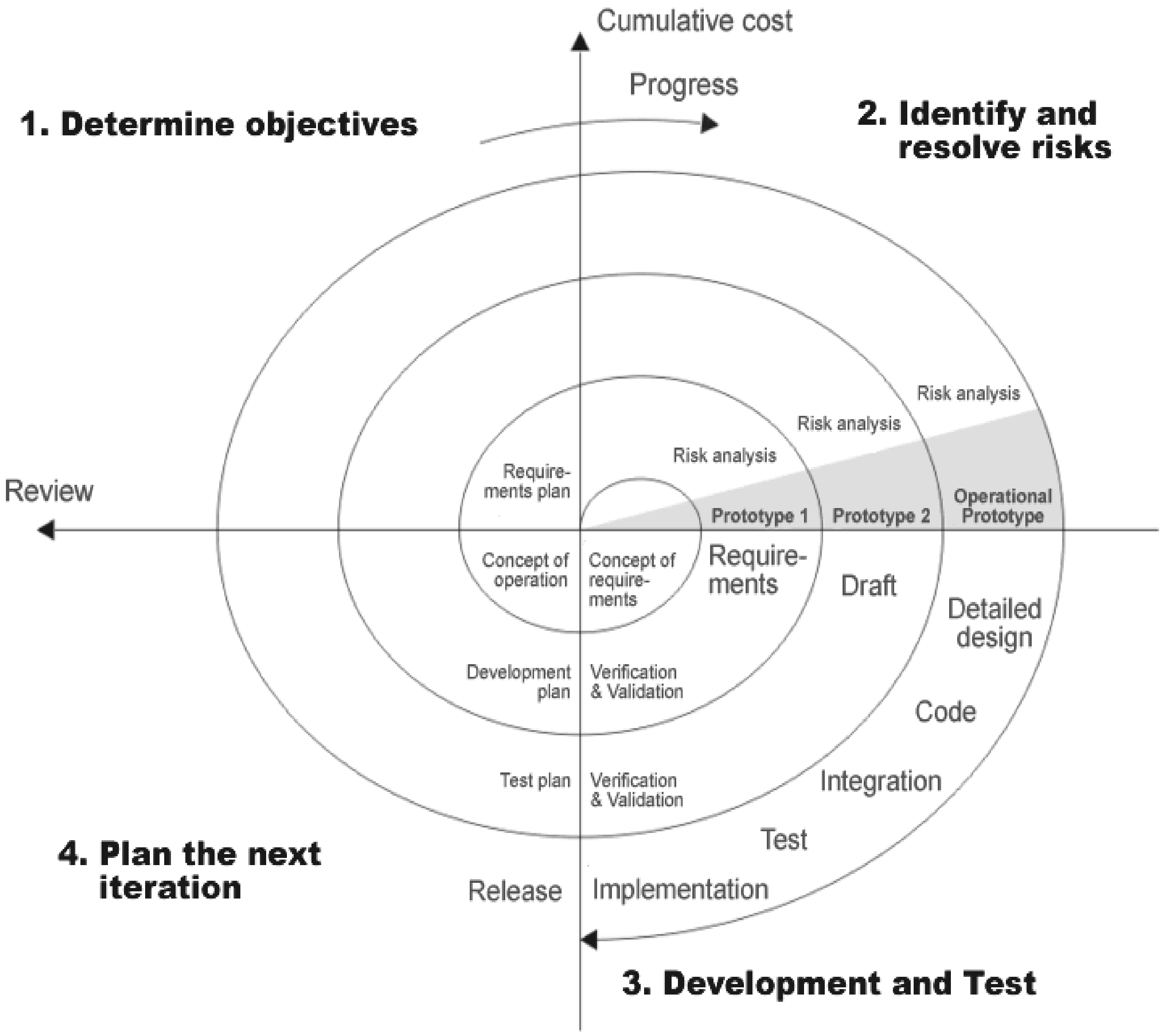
The Waterfall Model is a linear sequential flow. In which progress is seen as flowing steadily downwards (like a waterfall) through the phases of software implementation. This means that any phase in the development process begins only if the previous phase is complete. The waterfall approach does not define the process to go back to the previous phase to handle changes in requirement. The waterfall approach is the earliest approach and most widely known that was used for software development.

Waterfall model is an example of a Sequential model. In this model, the software development activity is divided into different phases and each phase consists of series of tasks and has different objectives.

Waterfall model is the pioneer of the SDLC processes. In fact, it was the first model which was widely used in the software industry. It is divided into phases and output of one phase becomes the input of the next phase. It is mandatory for a phase to be completed before the next phase starts. In short, there is no overlapping in Waterfall model
In waterfall, development of one phase starts only when the previous phase is complete. Because of this nature, each phase of waterfall model is quite precise well defined. Since the phases fall from higher level to lower level, like a waterfall, It's named as waterfall model.

Spiral Model:

The spiral model is similar to the incremental model, with more emphasis placed on risk analysis. The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation. A software project repeatedly passes through these phases in iterations (called Spirals in this model). The baseline spiral, starting in the planning phase, requirements are gathered and risk is assessed. Each subsequent spirals builds on the baseline spiral.



Planning Phase : Requirements are gathered during the planning phase. Requirements like ‘BRS’ that is ‘Business Requirement Specifications’ and ‘SRS’ that is ‘System Requirement specifications’.

Risk Analysis : In the **risk analysis phase**, a process is undertaken to identify risk and alternate solutions. A prototype is produced at the end of the risk analysis phase. If any risk is found during the risk analysis then alternate solutions are suggested and implemented.

Engineering Phase : In this phase software is **developed**, along with testing at the end of the phase. Hence in this phase the development and testing is done.

Evaluation phase : This phase allows the customer to evaluate the output of the project to date before the project continues to the next spiral.

Advantages of Spiral model :

- High amount of risk analysis hence, avoidance of Risk is enhanced.
- Good for large and mission-critical projects.
- Strong approval and documentation control.
- Additional Functionality can be added at a later date.
- Software is produced early in the software life cycle.

Disadvantages of Spiral model:

- Can be a costly model to use.
- Risk analysis requires highly specific expertise.
- Project’s success is highly dependent on the risk analysis phase.
- Doesn’t work well for smaller projects.

When to use Spiral model:

- When costs and risk evaluation is important
- For medium to high-risk projects
- Long-term project commitment unwise because of potential changes to economic priorities
- Users are unsure of their needs

- Requirements are complex

ANALYSIS

Requirements analysis :

Requirements analysis encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product or project, taking account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements.

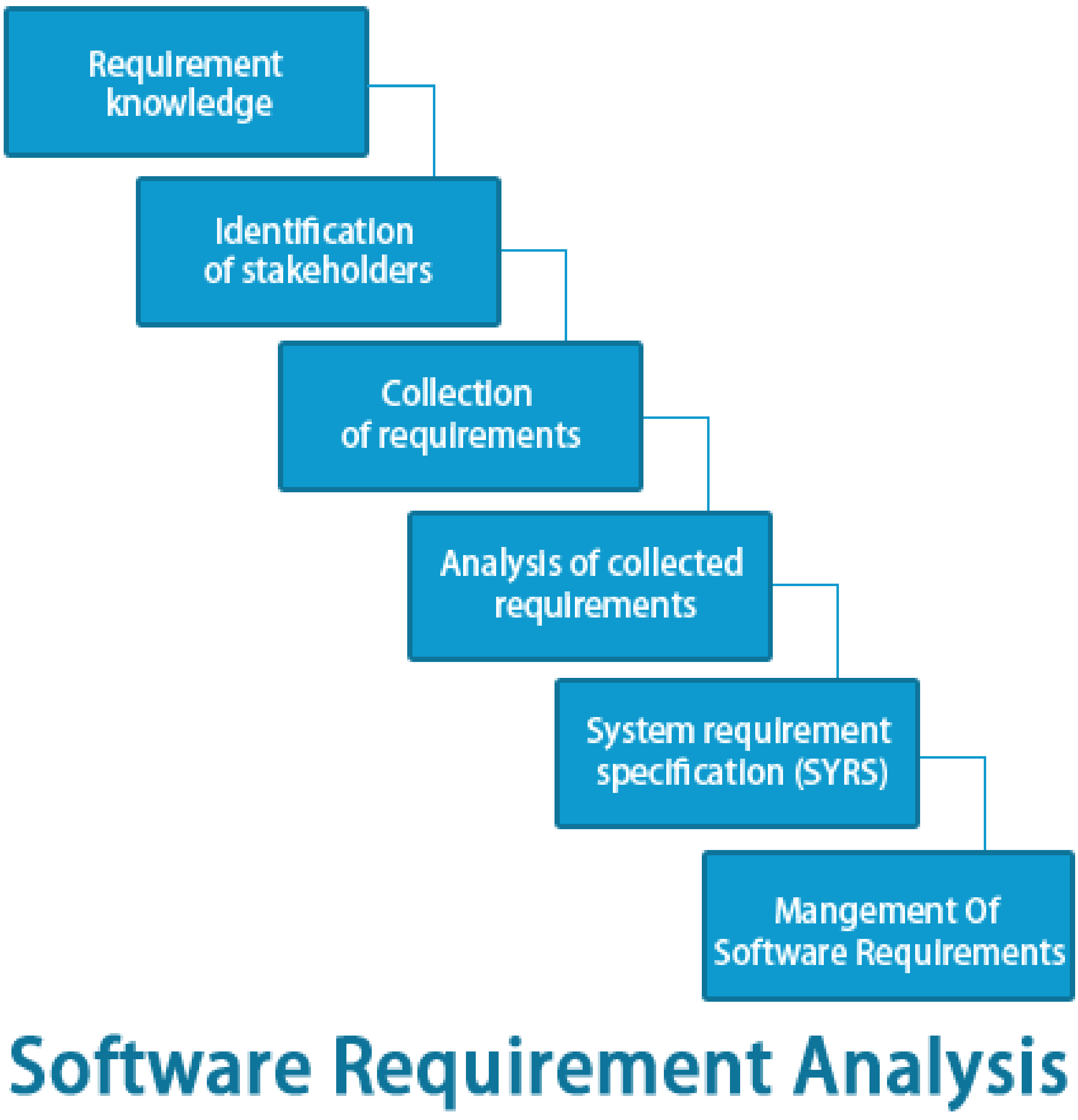
Requirements analysis is critical to the success or failure of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design.

Conceptually, requirements analysis includes three types of activities:

- Eliciting requirements : the business process documentation, and stakeholder interviews. This is sometimes also called requirements gathering or requirements discovery.
- Analyzing requirements : determining whether the stated requirements are clear, complete, consistent and unambiguous, and resolving any apparent conflicts.
- Recording requirements : Requirements may be documented in various forms, usually including a summary list and may include natural-language documents, use cases, user stories, process specifications and a variety of models including data models.

Requirements analysis can be a long and tiring process during which many delicate psychological skills are involved. Large systems may confront analysts with hundreds or thousands of system requirements.[4] New systems change the environment and relationships between people, so it is important to identify all the stakeholders, take into account all their needs and ensure they understand the implications of the new systems. Analysts can employ several techniques to elicit the requirements from the customer. These may include the development of scenarios (represented as user stories in agile methods), the identification of use cases, the use of workplace observation or ethnography, holding interviews, or focus groups (more aptly named in this context as requirements workshops, or requirements review sessions) and creating requirements lists. Prototyping may be used to develop an example system that can be demonstrated to stakeholders. Where necessary, the analyst will employ a combination of these methods to establish the exact requirements of the stakeholders, so that a system that meets the business needs is produced. Requirements quality can be improved through these and other methods

- Visualization : Using tools that promote better understanding of the desired end-product such as visualization and simulation.
- Consistent use of templates : Producing a consistent set of models and templates to document the requirements.
- Documenting dependencies : Documenting dependencies and interrelationships among requirements, as well as any assumptions and congregations.


Requirement knowledge
Identification of stakeholders
Collection of requirements
Analysis of collected requirements
System requirement specification (SYRS)
Mangement Of Software Requirements
Software Requirement Analysis

Feasibility Study :

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operation Feasibility
- Economical Feasibility

Technical Feasibility

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.

The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

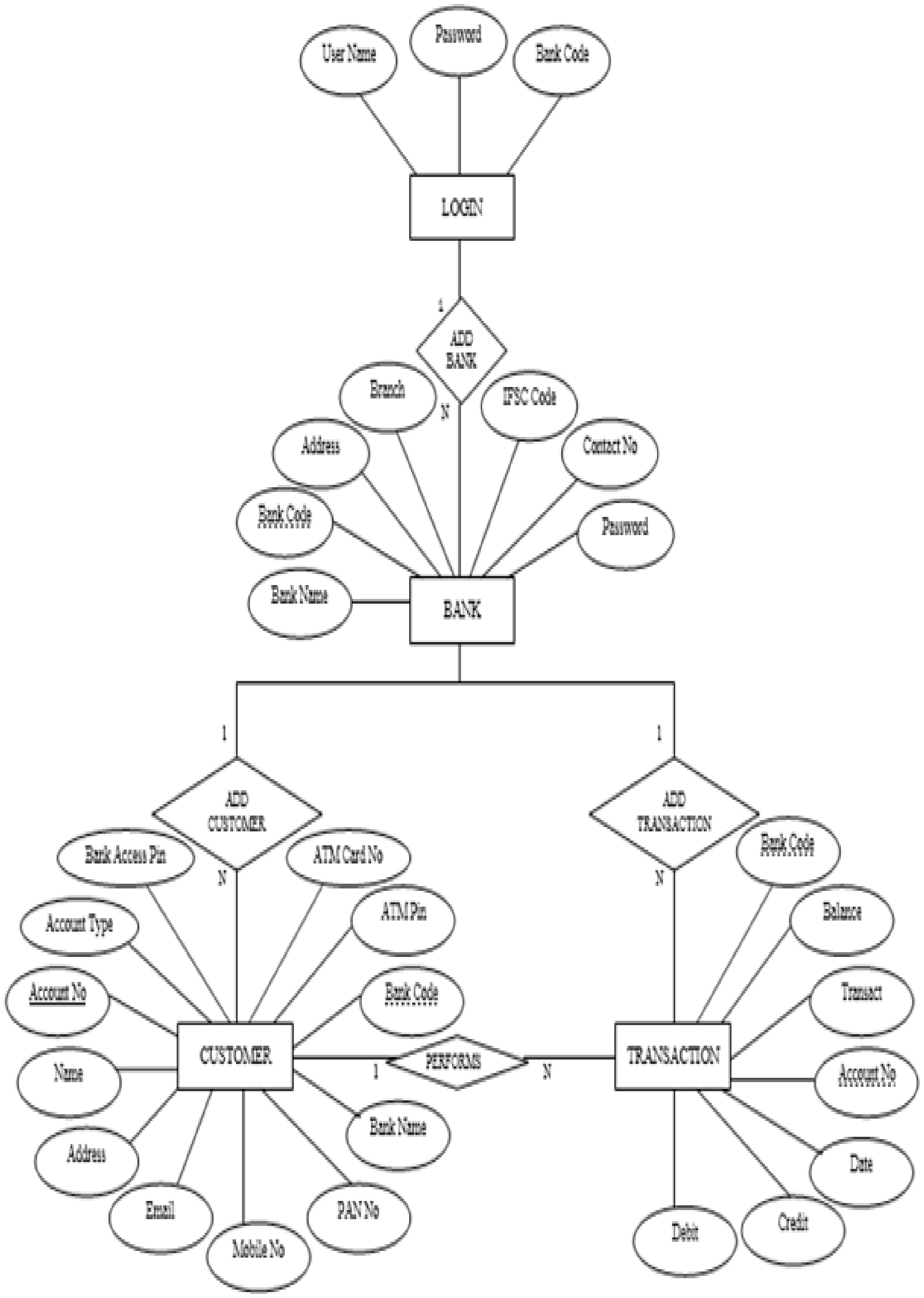
Economic Feasibility

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

Data Flow Diagram’s

E – R Diagram


User Name
Password
Bank Code
LOGIN
1
ADD BANK
N
Branch
IFSC Code
Address
Contact No
Bank Code
Password
Bank Name
BANK
1
1
ADD CUSTOMER
ADD TRANSACTION
N
N
Bank Access Pin
ATM Card No
Account Type
ATM Pin
Account No
Bank Code
CUSTOMER
PERFORMS
1
N
Name
Address
Email
Mobile No
PAN No
Bank Name
Bank Code
Balance
Transact
TRANSACTION
Account No
Date
Credit
Debit

Entity–Relationship Model (E-R Diagram)

An **entity–relationship model** (**ER model** for short) describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types.

An entity–relationship diagram for an MMORPG using Chen's notation. In software engineering, an ER model is commonly formed to represent things that a business needs to remember in order to perform business processes. Consequently, the ER model becomes an abstract data model, that defines a data or information structure which can be implemented in a database, typically a relational database.

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases. At first glance an entity relationship diagram looks very much like a flowchart. It is the specialized symbols, and the meanings of those symbols, that make it unique.

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.

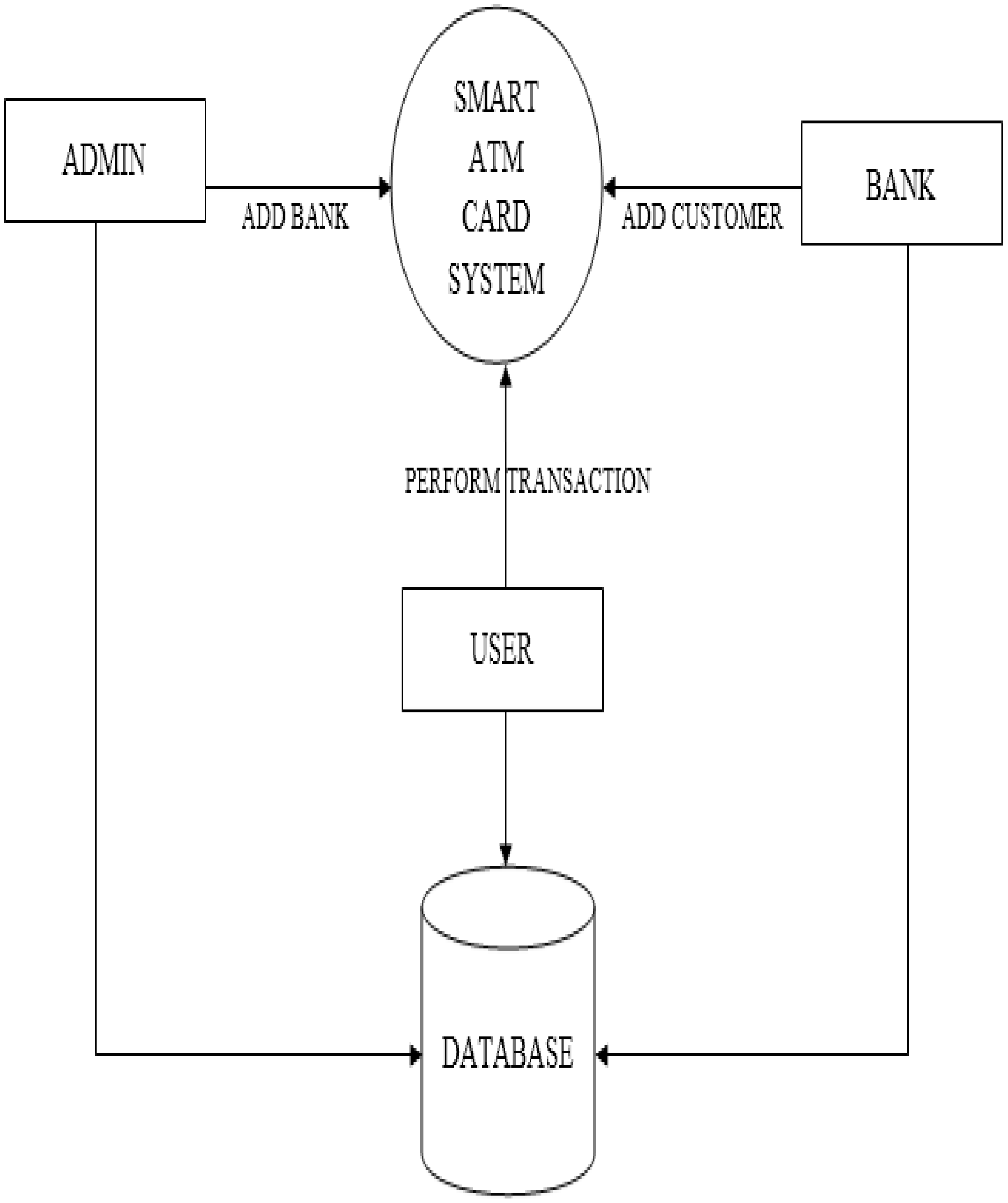
- **Database design:** ER diagrams are used to model and design relational databases, in terms of logic and business rules (in a logical data model) and in terms of the specific technology to be implemented (in a physical data model.) In software engineering, an ER diagram is often an initial step in determining requirements for an information systems project. It’s also later used to model a particular database or databases. A relational database has an equivalent relational table and can potentially be expressed that way as needed.

- **Database troubleshooting:** ER diagrams are used to analyze existing databases to find and resolve problems in logic or deployment. Drawing the diagram should reveal where it’s going wrong.

- **Business information systems:** The diagrams are used to design or analyze relational databases used in business processes. Any business process that uses fielded data involving entities, actions and interplay can potentially benefit from a relational database. It can streamline processes, uncover information more easily and improve results.

- **Business process re-engineering (BPR):** ER diagrams help in analyzing databases used in business process re-engineering and in modeling a new database setup.

- **Education:** Databases are today's method of storing relational information for educational purposes and later retrieval, so ER Diagrams can be valuable in planning those data structures.


ADMIN
SMART ATM CARD SYSTEM
BANK
ADD BANK
ADD CUSTOMER
PERFORM TRANSACTION
USER
DATABASE

Data Flow Diagram (DFD)

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show information about process timing or whether processes will operate in sequence or in parallel, unlike a traditional structured flowchart which focuses on control flow, or a UML activity workflow diagram, which presents both control and data flows as a unified model.

Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.

ER diagrams are related to data structure diagrams (DSDs), which focus on the relationships of elements within entities instead of relationships between entities themselves. ER diagrams also are often used in conjunction with data flow diagrams (DFDs), which map out the flow of information for processes or systems.

IMPLEMENTATION

IMPLEMENTATION

Implementation is the realization of an application or execution of a plan, idea, model, design, specification, standard, algorithm or policy.

IMPLEMENTATION OF CARDLESS TRANSACTION

The Cardless Transaction we have introduced a cardless cash withdrawal facility which allows customers to transfer money from their account to anyone in India with a mobile number.
We foresee tremendous growth potential in the use of electronic payments in our country. 'Instant Money Transfer and Withdrawal without Bank Account ' provides an added facility in an array of electronic payment options that Bank offers to its customers.

The sender first needs to register the recipient's name, mobile number and address. The sender will get a four-digit verification code while the recipient a six digit reference code, over SMS. The recipient can withdraw cash from ATM by entering the mobile number, cash amount along with the verification and reference code. This service can also be used by the Bank's account holders to withdraw cash from their own accounts without using a debit card.

There are almost 2 million ATMs around the globe. Although use of the machines has declined in recent years, likely because more people make purchases using credit and debit cards instead of cash, the ATM continues to have a place in modern culture. Today’s machines sell everything from airline tickets to movie tickets to medicine.

The automated teller machine, or ATM, is such a complicated piece of technology that it does not have a single inventor. Instead, the ATMs we use today are an amalgam of several different inventions. Some of these proto-ATMs dispensed cash but did not accept deposits, for example, while others accepted deposits but did not dispense cash. Today’s ATMs are sophisticated computers that can do almost anything a human bank teller can, and have ushered in a new era of self-service in banking.

Today the money transactions in ATM are done with Cards, and Net banking is done form account to account.

In the proposed system we are introducing an new feature that is we can allow the person to withdraw the amount from ATM even if he/she has no bank account. }In this application the account holder will be sending the request to server to allow the withdrawal of the amount from the ATM by using the mobile. Here the user will be sending and request to the server with the withdrawer mobile number and amount to be withdrawn, then the server sends an message to the person after the confirmation is done by the account holder.

To withdraw the amount the person has to visit the ATM and enter his mobile number. Then an OTP is generated and sent to the withdrawer, the withdrawer has to put the OTP and then withdraw the amount. The most important feature in this application is that the withdrawer can only withdraw the amount which has been set the account holder and even the original balance is hidden from the withdrawer.

INTRODUCTION TO PHP

"PHP is an html-embedded scripting language. Much of its syntax is borrowed from c, java and Perl with a couple of unique PHP-specific features thrown in. the goal of the language is to allow web developers to write dynamically generated pages quickly." This is generally a good definition of PHP. However, it does contain a lot of terms you may not be used to. Another way to think of PHP is a powerful, behind the scenes scripting language that your visitors won't see! When someone visits your PHP webpage, your web server processes the PHP code. It then sees which parts it needs to show to visitors(content and pictures) and hides the other stuff(file operations, math calculations, etc.) then translates your PHP into html. After the translation into html, it sends the webpage to your visitor's web browser.

PHP - What's it do?

It is also helpful to think of PHP in terms of what it can do for you. PHP will allow you to:

• Reduce the time to create large websites.

• Create a customized user experience for visitors based on information that you have gathered from them.

• Open up thousands of possibilities for online tools. check out PHP - hot scripts for examples of the great things that are possible with PHP.

• Allow creation of shopping carts for e-commerce websites.

PHP is a server-side, html-embedded scripting language that may be used to create dynamic WebPages. It is available for most operating systems and web servers, and can access most common databases, including MySQL. PHP may be run as a separate program or compiled as a module for use with a web server.

MySQL Database :

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is developed, marketed, and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons:

- MySQL is released under an open-source license. So you have nothing to pay to use it.
- MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
- MySQL uses a standard form of the well-known SQL data language.
- MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
- MySQL works very quickly and works well even with large data sets.
- MySQL is very friendly to PHP, the most appreciated language for web development.
- MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).

PHP AND DATABASE

- PHP and MYSQL are a perfect companion.
- Largely because they both free and they have numerous capabilities.
- PHP as of version # supports inherently MySQL i.e specialized build-in function handle database interaction.
- Same goes with ORACLE but not with Microsoft database(Access SQL server)

INTRODUCTION TO CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. While most often used to change the style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG and XUL. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications.[1]

Apache HTTP Server

The Apache HTTP Server, colloquially called Apache, is the world's most widely used web server software. Originally based on the NCSA HTTPd server, development of Apache began in early 1995 after work on the NCSA code stalled. Apache played a key role in the initial growth of the World Wide Web, quickly overtaking NCSA HTTPd as the dominant HTTP server, and has remained the most popular HTTP server since April 1996. In 2009, it became the first web server software to serve more than 100 million websites.[5]

Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. Most commonly used on a Unix-like system (usually Linux), the software is available for a wide variety of operating systems, including Windows, OS X, Linux, Unix, FreeBSD, Solaris, NetWare, OS/2, TPF, OpenVMS and eComStation. Released under the Apache License, Apache is free and open-source software.

Database

A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.

Other kinds of data stores can be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those types of systems.

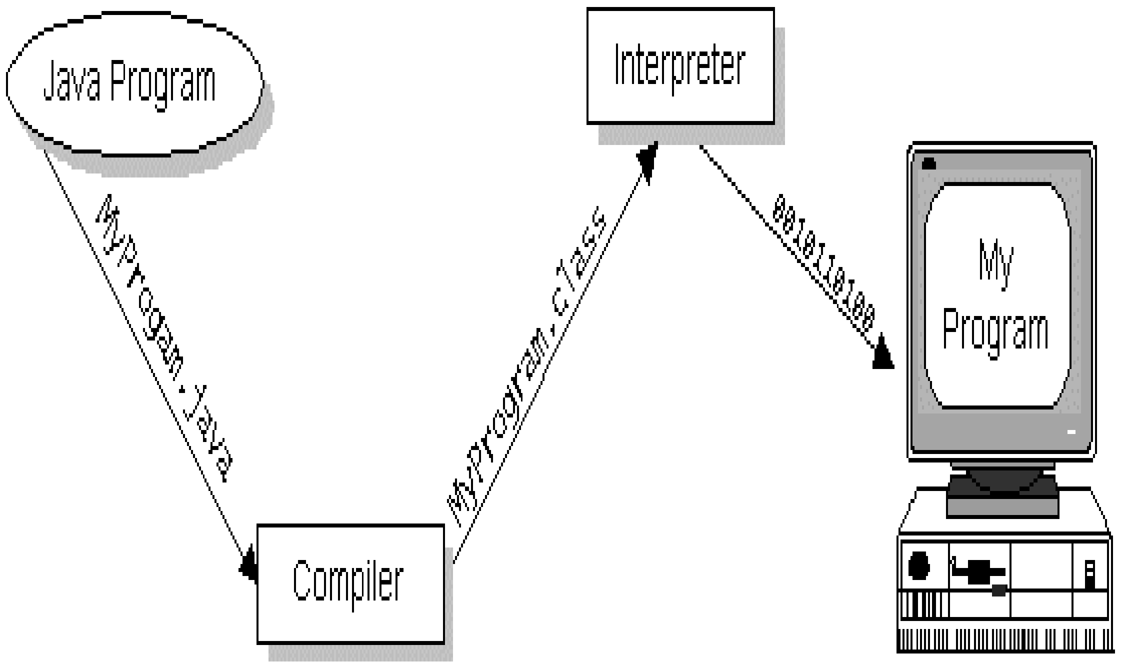
So nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as foreign keys.

Java Language

Java is a high-level programming language which is all of the following:

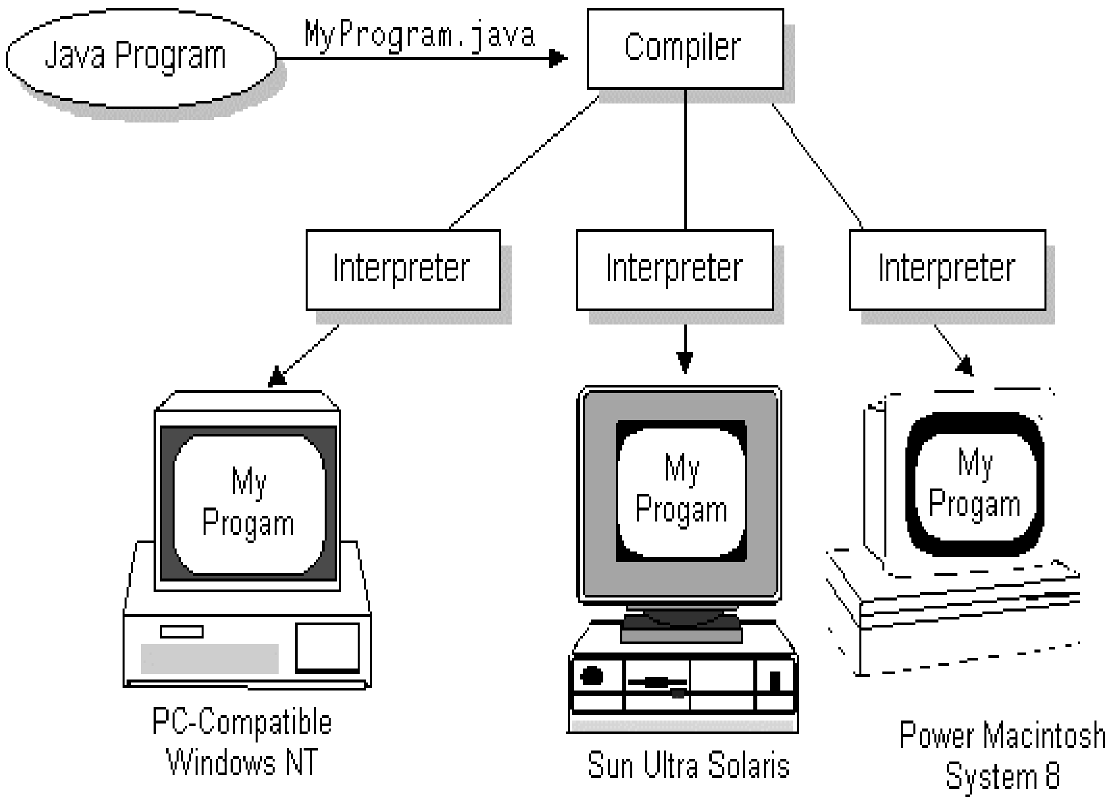
- Simple
- Object-oriented
- Distributed
- Interpreted
- Robust
- Secure
- Architecture-neutral
- Portable
- High-performance
- Multithreaded
- Dynamic

Java is also unusual in that each Java program is both compiled and interpreted. With a compiler, you translate a Java program into an intermediate language called Java byte codes--the platform-independent codes interpreted by the Java interpreter. With an interpreter, each Java byte code instruction is parsed and run on the computer. Compilation happens just once; interpretation occurs each time the program is executed. This figure illustrates how this works.



Java byte codes can be considered as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make "write once, run anywhere" possible. The Java program can be compiled into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. For example, the same Java program can run on Windows NT, Solaris, and Macintosh.



The Java Platform

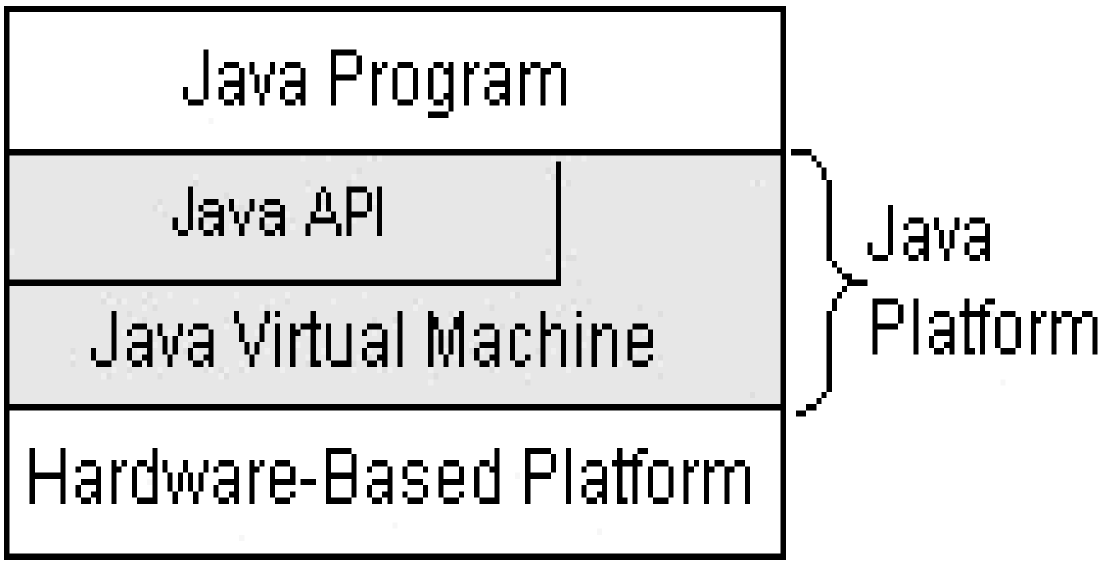
A platform is the hardware or software environment in which a program runs. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other, hardware-based platforms. Most other platforms are described as a combination of hardware and operating system.

The Java platform has two components:

- The Java Virtual Machine (Java VM)
- The Java Application Programming Interface (Java API)

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries (packages) of related components.

The following figure depicts a Java program, such as an application or applet, that's running on the Java platform. As the figure shows, the Java API and Virtual Machine insulates the Java program from hardware dependencies.



As a platform-independent environment, Java can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring Java's performance close to that of native code without threatening portability.

Java Libraries

The Java Class Library is a set of dynamically loadable libraries that Java applications can call at run time. Because the Java Platform is not dependent on any specific operating system, applications cannot rely on any of the existing libraries. Instead, the Java Platform provides a comprehensive set of standard class libraries, containing much of the same reusable functions commonly found in modern operating systems.

The Java class libraries serve three purposes within the Java Platform:

Like other standard code libraries, they provide the programmer a well-known set of useful facilities, such as container classes and regular expressions.

- In addition, the class libraries provide an abstract interface to tasks that would normally depend heavily on the hardware and operating system. Tasks such as network access and file access are often heavily dependent on the native capabilities of the platform.
- Finally, some underlying platforms may not support all of the features a Java application expects. In these cases, the class libraries can either emulate those features using whatever is available, or provide a consistent way to check for the presence of a specific feature.

JDBC

In an effort to set an independent database standard API for Java, Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMS. This consistent interface is achieved through the use of “plug-in” database connectivity modules, or *drivers*. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC’s framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

- **Provide a Java interface that is consistent with the rest of the Java system**
  Because of Java's acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.
- **Keep it simple** : This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.
- **Use strong, static typing wherever possible** : Strong typing allows for more error checking to be done at compile time; also, less errors appear at runtime.
- **Keep the common cases simple** : Because more often than not, the usual SQL calls used by the programmer are simple SELECT's, INSERT's, DELETE's and UPDATE's, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

SOURCE CODE

Samples codes of website :

```
<?php
ini_set('error_reporting', 0);
ini_set('display_errors', 0);
session_start();
if ($_SESSION['id'] == "") {
echo "<script>window.location.href='index.php'</script>";
}
include 'dbconnection.php';
$queryt = "Select * from logintime where email='{$_SESSION['id']}'";
$resultt = mysql_query($queryt);
$rowt = mysql_fetch_assoc($resultt);
date_default_timezone_set('Asia/Kolkata');
$h = date('H');
$i = date('i');
$c = $i - $rowt['minutes'];
if ($c >= 2) {
echo "<script type='text/javascript'>alert('Time Out. ReLogin to continue');</script>";
echo "<script>window.location.href='login.php'</script>";
} else {
$queryt1 = "UPDATE logintime SET hour='$h',minutes='$i' WHERE
email='{$_SESSION['id']}'";
$resultt1 = mysql_query($queryt1);
}
?>
```

```
<html lang="en">

<head><style type="text/css">            #firstname{                font-family: "Bauhaus 93";
font-size: 40px;             }        </style>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>

<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"/>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

<script type="text/javascript" src="js/jquery-1.6.js" ></script>

<script type="text/javascript" src="js/cufon-yui.js"></script>

<script type="text/javascript" src="js/cufon-replace.js"></script>

<script type="text/javascript" src="js/Vegur_300.font.js"></script>

<script type="text/javascript" src="js/PT_Sans_700.font.js"></script>

<script type="text/javascript" src="js/PT_Sans_400.font.js"></script>

<script type="text/javascript" src="js/atooltip.jquery.js"></script>

<style type="text/css">

table {

border-collapse: collapse;

width: 100%;           }

th, td {

text-align: left;

padding: 3px;

}   tr:hover {background-color:white;}

         th {

background-color: #4299DA  ;
```

```
color: white;  }
.aa input[type="text"]{
width: 255px;
height: 35px;
border-radius: 5px;
padding-left: 5px;
background-color: transparent;
border:1;            }
.aa input[type="password"]{
width: 255px;
height: 35px;
border-radius: 5px;
padding-left: 5px;
background-color: transparent;
border:1;}
.aa input[type="email"]{
width: 255px;
height: 35px;
border-radius: 5px;
padding-left: 5px;
background-color: transparent;
border:1;}
.aa input[type="submit"]{
width: 160px;
height: 35px;
```

```
border-radius: 5px;

background-color: transparent;

border:1;

font-weight: bolder;

}

.aa input[type="submit"]:hover{

mouse:pointer;

}

</style>

<!-- Required meta tags -->

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

<title>DROP IN</title>

<!-- Favicon -->

<link rel="shortcut icon" type="image/icon" href="assets/images/favicon.ico"/>

<!-- Font Awesome -->

<link href="https://maxcdn.bootstrapcdn.com/font-awesome/4.6.3/css/font-awesome.min.css"
rel="stylesheet">

<!-- Bootstrap CSS -->

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
beta/css/bootstrap.min.css" integrity="sha384-
/Y6pD6FV/Vv2HJnA6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M"
crossorigin="anonymous">

<!-- Slick slider -->

<link href="assets/css/slick.css" rel="stylesheet">

<!-- Gallery Lightbox -->

<link href="assets/css/magnific-popup.css" rel="stylesheet">
```

```
<!-- Skills Circle CSS  -->

<link rel="stylesheet" type="text/css"
href="https://unpkg.com/circlebars@1.0.3/dist/circle.css">

<!-- Main Style -->

<link href="style.css" rel="stylesheet">

<!-- Fonts -->

<!-- Google Fonts Raleway -->

<link href="https://fonts.googleapis.com/css?family=Raleway:300,400,400i,500,500i,600,700"
rel="stylesheet">

<!-- Google Fonts Open sans -->

<link href="https://fonts.googleapis.com/css?family=Open+Sans:400,400i,600,700,800"
rel="stylesheet">

</head>

<script>

function check(){

if((isNaN(document.f.cardno.value)) || (document.f.cardno.value.length<16)){

alert("Invalid card no");

return false;

}

if((isNaN(document.f.mobile.value)) || (document.f.mobile.value.length<10)){

alert("Invalid mobile no");

return false;

}

if(((document.f.pass.value)) != (document.f.repass.value)){

alert("Password does not match");

return false;

}
return true;
}
</script>
<body>
<!--START SCROLL TOP BUTTON --><a class="scrollToTop" href="#">
<i class="fa fa-angle-up"></i>
</a>
<!-- END SCROLL TOP BUTTON -->

<!-- Start Header -->
<header id="mu-hero">

<nav class="navbar navbar-expand-lg navbar-light mu-navbar">
<!-- Text based logo -->
<a class="navbar-brand mu-logo" href="index.php"><p id="firstname"><img
src="DROP.png"/><span><i>DROP IN</i></span></p></a>
<!-- image based logo -->
<!-- <a class="navbar-brand mu-logo" href="index.html"><img src="assets/images/logo.png"
alt="logo"></a> -->
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
<span class="fa fa-bars"></span><br/></span>
</button>
<right><div class="collapse navbar-collapse" id="navbarSupportedContent">
<ul class="navbar-nav mr-auto mu-navbar-nav">
```

```
<li class="nav-item"><a href="home.php">Home</a></li>

<li class="nav-item"><a href="viewaccount.php">VIEW ACCOUNT</a></li>

<li class="nav-item"><a href="moneytransfer.php">MONEY TRANSFER</a></li>

<li class="dropdown">

<a class="dropdown-toggle" class="nav-item active" data-toggle="dropdown"
href="#">GENERAL SETTINGS</a>

<ul class="dropdown-menu" >

<div class="aa">

<li class="nav-item"><a href="changepassword.php">CHANGE PASSWORD</a></li>

<li class="nav-item"><a href="logindetails.php">LOGIN DETAILS</a></li>

</div>

</ul>

</li>

<li class="nav-item active">

<a href="logindetails.php">LOGIN DETAILS</a>

</li>

<li class="nav-item"><a href="logout.php">LOGOUT</a></li>

</ul>

</div>

</nav>

</div>

</header>

<!-- End Header -->

<!-- Start Page Header area -->

<div id="mu-page-header">

<div class="container">
```

```
<div class="row">
<div class="col-md-12">
<div class="mu-page-header-area">
<div class="aa">
<article id="content"><div class="ic"></div>
<div class="wrapper">
<center>
<h2>Login Details</h2><br/>
<script>
$.get("https://ipinfo.io/json", function (response) {
$("#ip").html("IP: " + response.ip);
$("#address").html("Location: " + response.city + ", " + response.region);
$("#details").html(JSON.stringify(response, null, 4));
}, "jsonp");
</script>
<?php
$query = "select * from logindetails where email='{$_SESSION['id']}' order by slno desc limit 
100";
$result = mysql_query($query);
if (mysql_num_rows($result) > 0) {
?>
<br/><br/>
<table>
<tr align="center">
<th>Sl No</th>
<th>Time</th>
<th>IP Address</th>
<th>Browser</th>
<th>Operating system</th>
</tr>
<?php
$i = "1";
while ($row = mysql_fetch_assoc($result)) {
?>
<tr><td align="center"><?php echo "<font color='#4299DA'>" . $i ?></td>
<td align="center"><?php echo "<font color='#4299DA'>" . $row['time'] ?></td>
<td align="right"><?php echo $row['ip'] ?></td>
<td align="right"><?php echo $row['browser'] ?></td>
<td align="right"><?php echo $row['os'] ?></td>
<?php
$i++;
}
echo "</tr></table>";
} else {
echo "<script type='text/javascript'>alert('Server is busy.');</script>";
// echo "<script>window.location.href='viewaccount.php'</script>";
}
?>
</center>
</div>
</form>
```

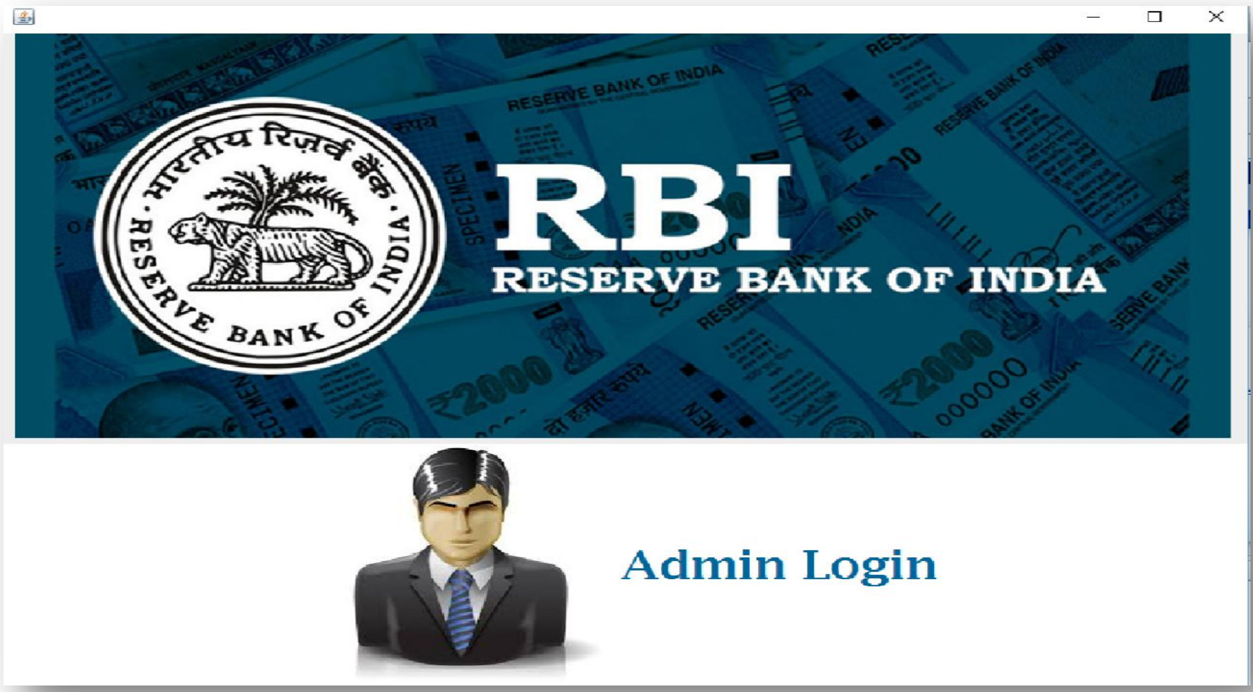
```
</div>
<!-- End Page Header area -->
<!-- Start Breadcrumb -->
<!-- End Breadcrumb -->
<!-- Start main content -->
<!-- Start About -->
<!-- End About -->

<!-- Start Skills -->

<!-- End Team -->

<!-- Start Clients -->
<div id="mu-clients">
<div class="container">
<div class="row">
<div class="col-md-12">
<div class="mu-clients-area">
<!-- Start Clients brand logo -->
<div class="mu-clients-slider">
<div class="mu-clients-single">
<img src="assets/images/client-brand-1.jpg" alt="Brand Logo">
</div>
<div class="mu-clients-single">
<img src="assets/images/client-brand-2.jpg" alt="Brand Logo">
```

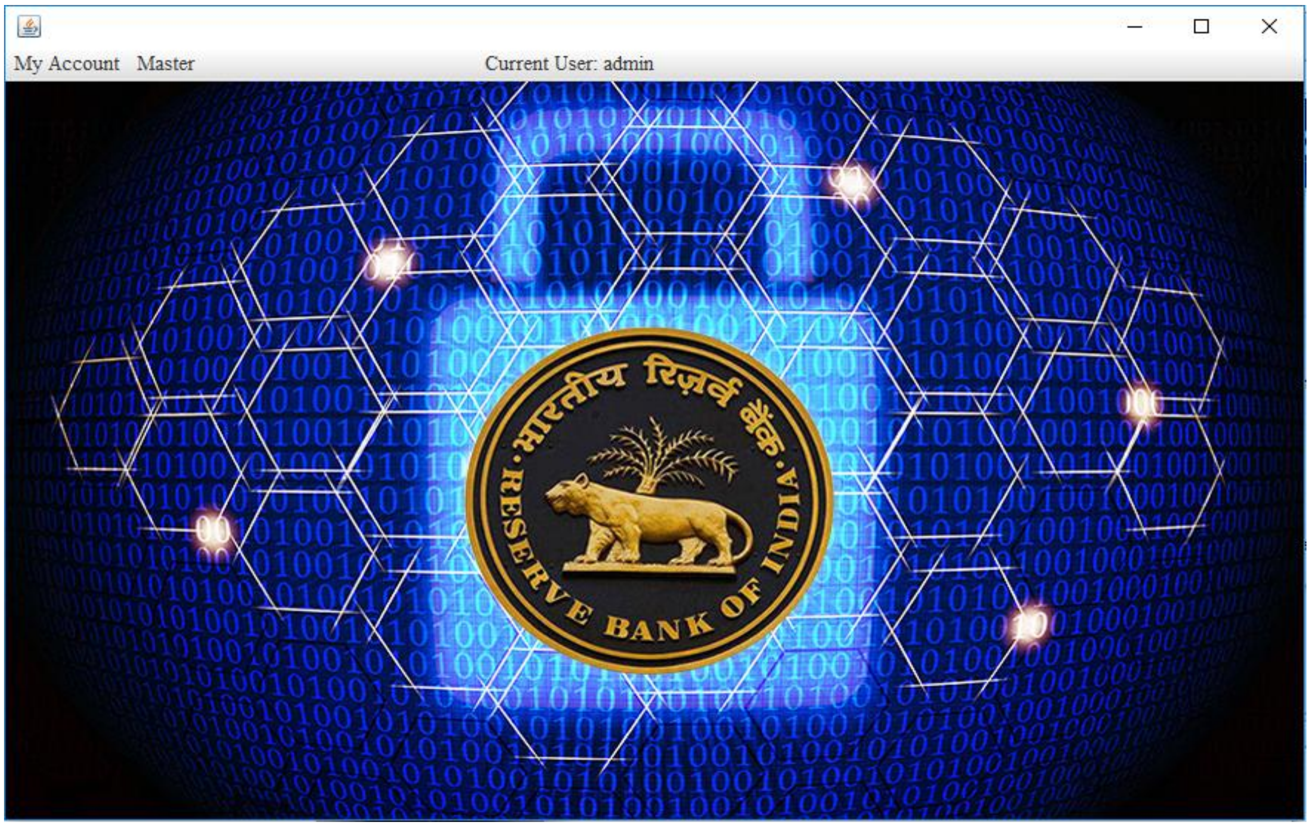
```
</div>
<div class="mu-clients-single">
<img src="assets/images/client-brand-3.jpg" alt="Brand Logo">
</div>
<div class="mu-clients-single">
<img src="assets/images/client-brand-4.jpg" alt="Brand Logo">
</div>
<div class="mu-clients-single">
<img src="assets/images/client-brand-5.jpg" alt="Brand Logo">
</div>
<div class="mu-clients-single">
<img src="assets/images/client-brand-6.jpg" alt="Brand Logo">
</div>
</div>
<!-- End Clients brand logo -->
</div>
<!-- End Clients -->
<footer id="mu-footer">
<div class="mu-footer-bottom">
<div class="container">
<div class="row">
<div class="col-md-12">
<div class="mu-footer-bottom-area">
<p class="mu-copy-right">&copy; Copyright 2018 <a rel="nofollow" href="index.php">Drop
In</a>. All right reserved.</p>
</div>
```

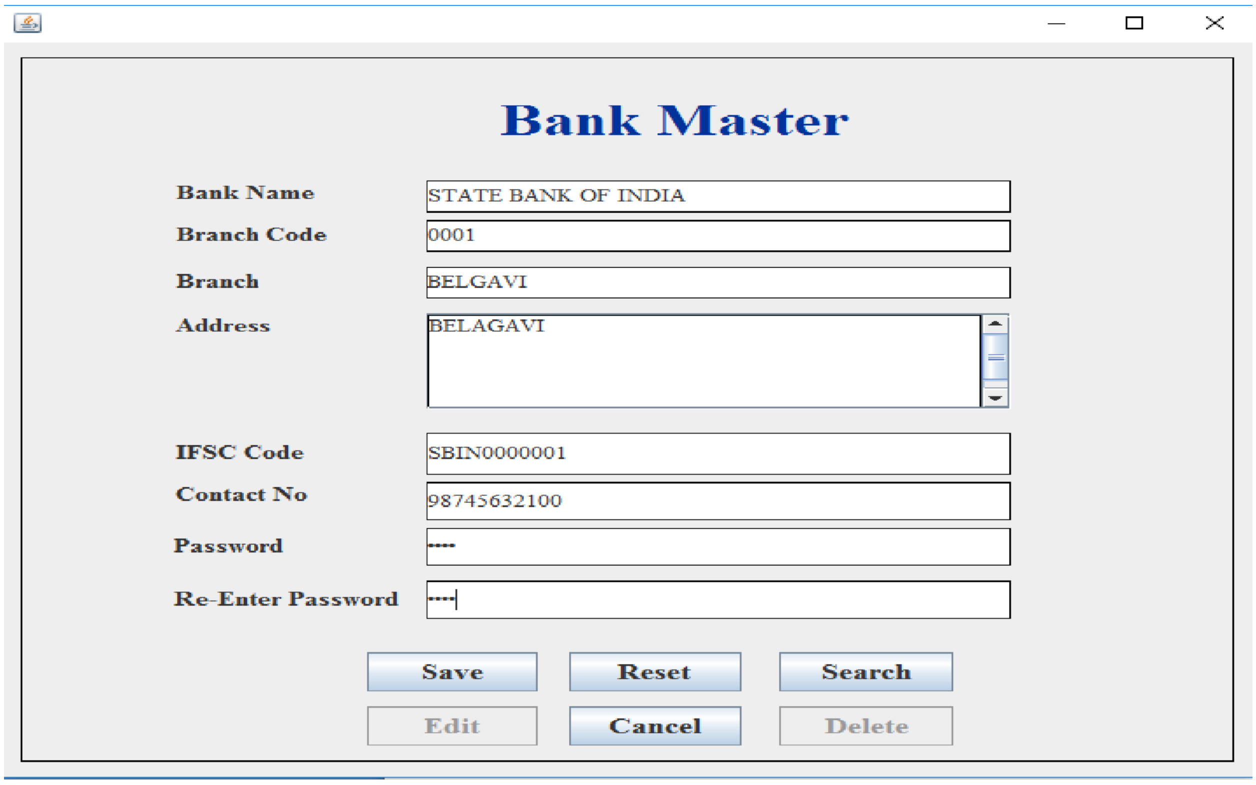
```
</footer>

<!-- End footer -->

<!-- JavaScript -->

<!-- jQuery first, then Popper.js, then Bootstrap JS -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.11.0/umd/popper.min.js"
integrity="sha384-
b/U6ypiBEHpOf/4+1nzFpr53nxSS+GLCkfwBdFNTxtclqqenISfwAzpKaMNFNmj4"
crossorigin="anonymous"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/js/bootstrap.min.js"
integrity="sha384-
h0AbiXch4ZDo7tp9hKZ4TsHbi047NrKGLO3SEJAg45jXxnGIfYzk4Si90RDIqNm1"
crossorigin="anonymous"></script>

<!-- Slick slider -->

<script type="text/javascript" src="assets/js/slick.min.js"></script>

<!-- Progress Bar -->

<script src="https://unpkg.com/circlebars@1.0.3/dist/circle.js"></script>

<!-- Filterable Gallery js -->

<script type="text/javascript" src="assets/js/jquery.filterizr.min.js"></script>

<!-- Gallery Lightbox -->

<script type="text/javascript" src="assets/js/jquery.magnific-popup.min.js"></script>

<!-- Counter js -->

<script type="text/javascript" src="assets/js/counter.js"></script>

<!-- Ajax contact form  -->

<script type="text/javascript" src="assets/js/app.js"></script>

<!-- Custom js -->

<script type="text/javascript" src="assets/js/custom.js"></script>

<!-- About us Skills Circle progress  -->
```

```
<script>
// First circle
new Circlebar({
element : "#circle-1",
type : "progress",
maxValue:  "90"
});
// Second circle
new Circlebar({
element : "#circle-2",
type : "progress",
maxValue:  "84"
});
// Third circle
new Circlebar({
element : "#circle-3",
type : "progress",
maxValue:  "60"
});

// Fourth circle
new Circlebar({
element : "#circle-4",
type : "progress",
maxValue:  "74"
});
$(window).load(function(){
$('#slider')._TMS({
banners:true,
waitBannerAnimation:false,
preset:'diagonalFade',
easing:'easeOutQuad',
pagination:true,
duration:400,
slideshow:8000,
bannerShow:function(banner){
banner.css({marginRight:-500}).stop().animate({marginRight:0}, 600)
},
bannerHide:function(banner){
banner.stop().animate({marginRight:-500}, 600)
}
})
})
</script>

</body>
</html>
```

SCREEN SHOTS


RBI
RESERVE BANK OF INDIA
Admin Login

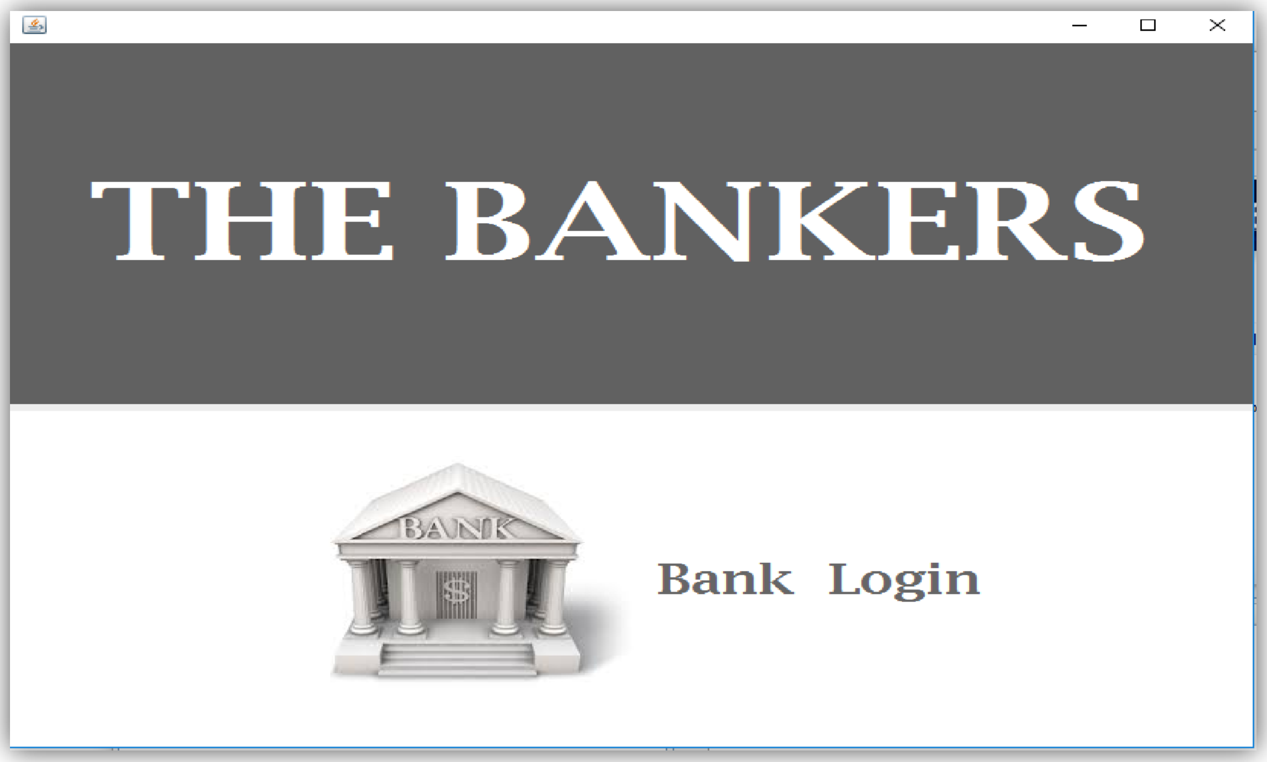

My Account
Master
Current User: admin

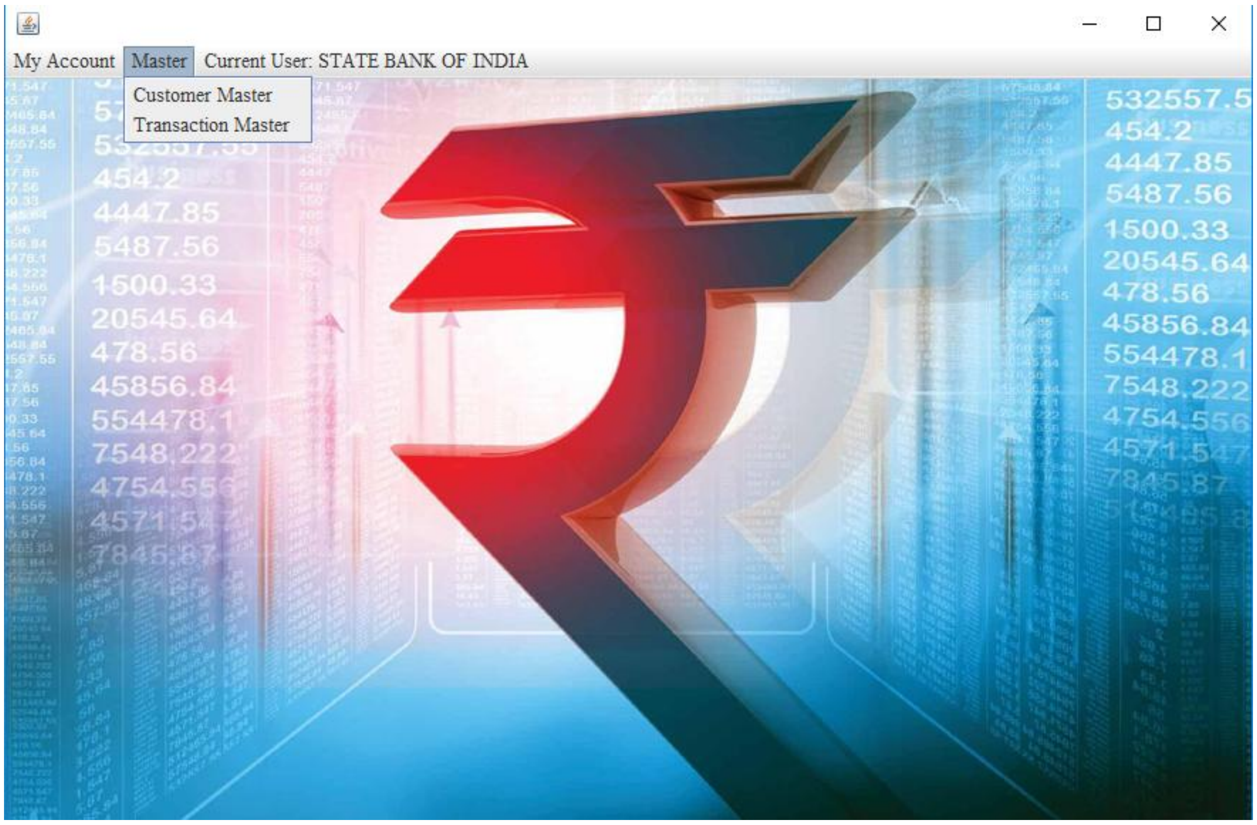


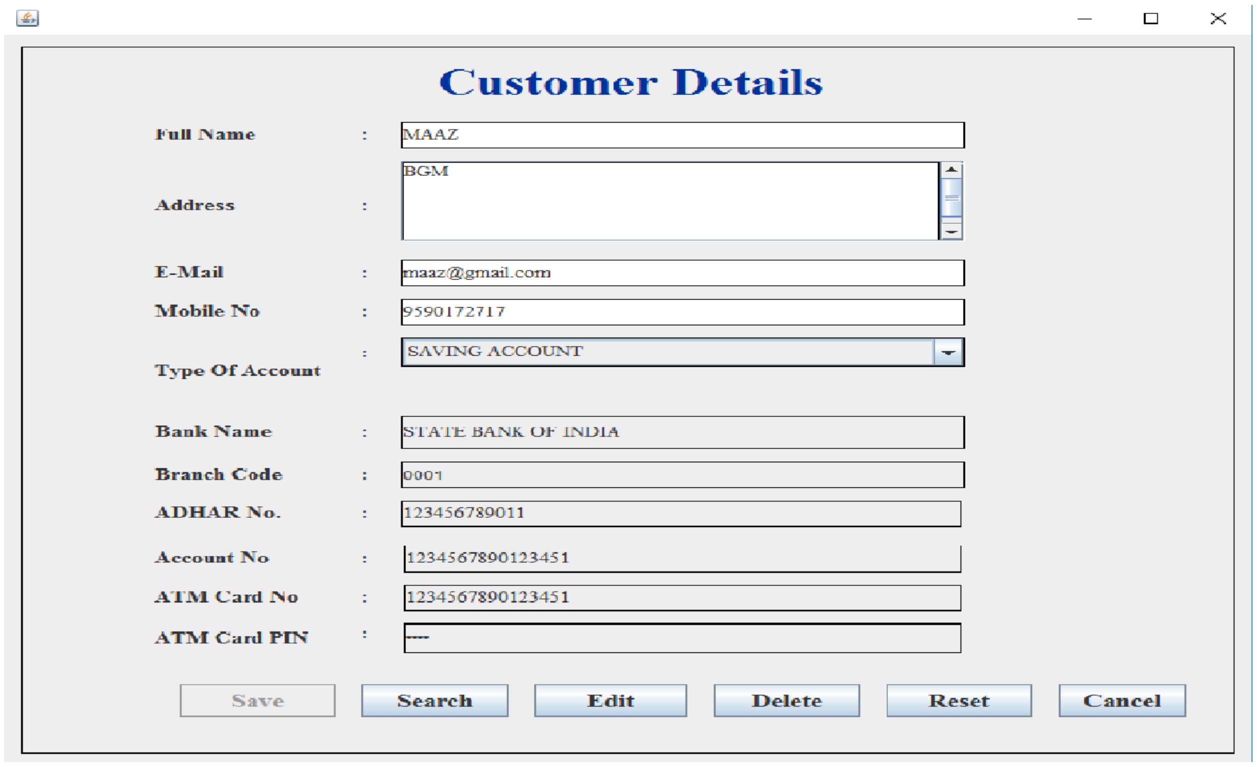
Bank Master

Bank Name	STATE BANK OF INDIA
Branch Code	0001
Branch	BELGAVI
Address	BELAGAVI
IFSC Code	SBIN0000001
Contact No	98745632100
Password	----
Re-Enter Password	----

MODULE 2 :

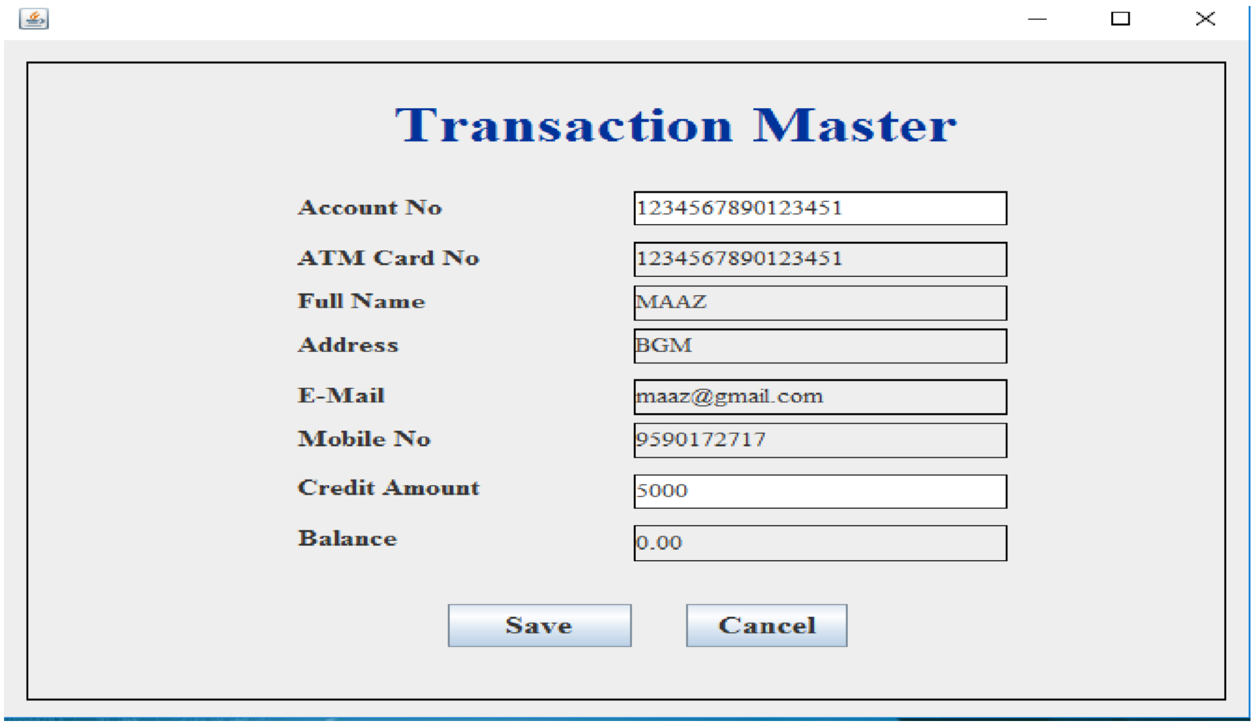






Customer Details

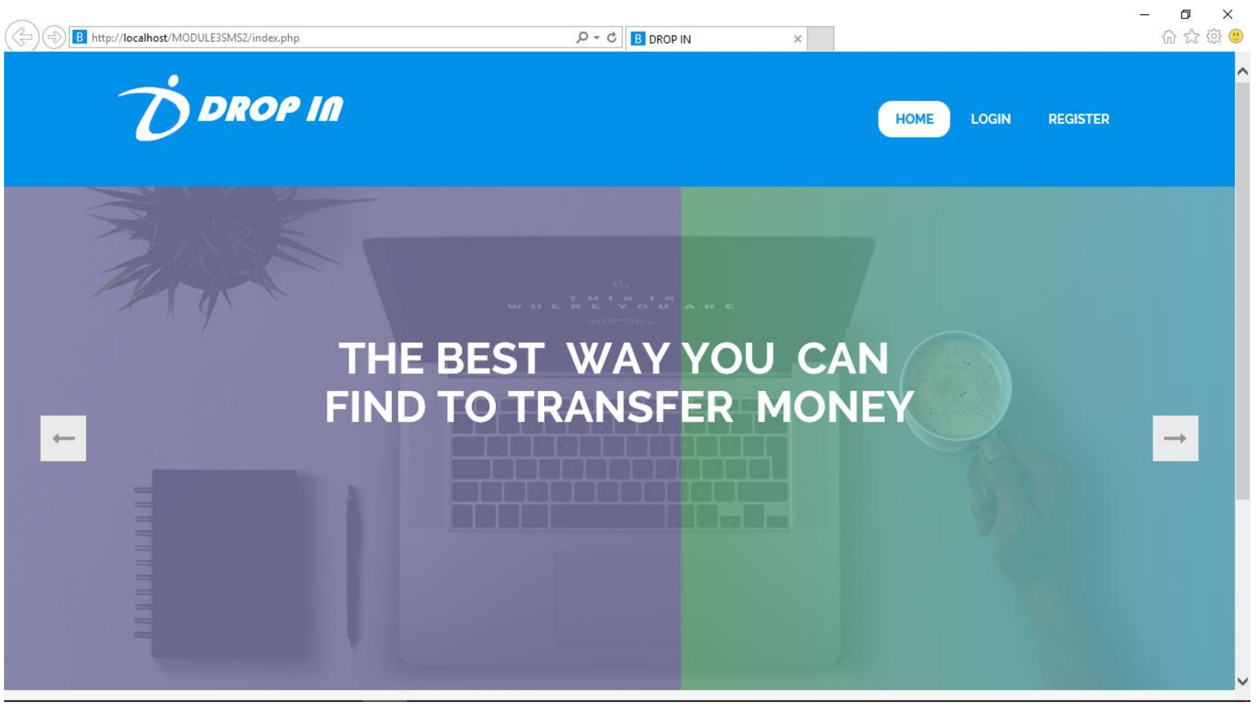
Full Name	:	MAAZ
Address	:	BGM
E-Mail	:	maaz@gmail.com
Mobile No	:	9590172717
Type Of Account	:	SAVING ACCOUNT
Bank Name	:	STATE BANK OF INDIA
Branch Code	:	0001
ADHAR No.	:	123456789011
Account No	:	1234567890123451
ATM Card No	:	1234567890123451
ATM Card PIN	:	----



Transaction Master

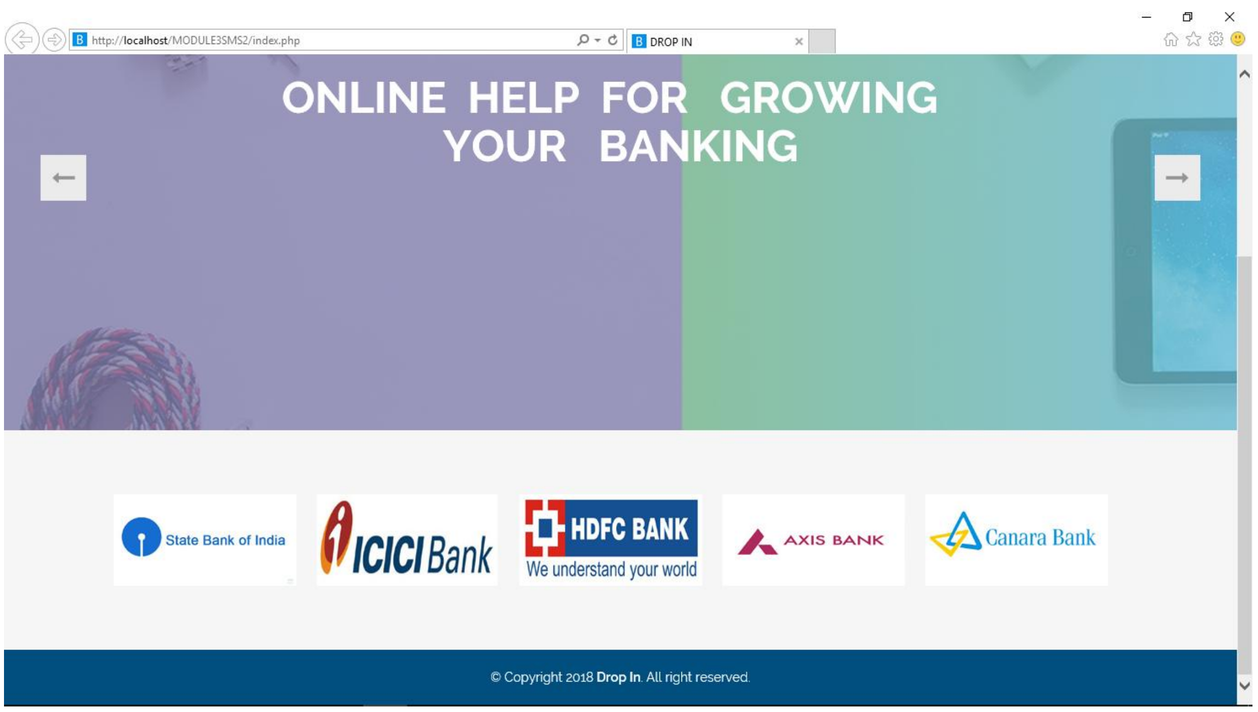
Account No	1234567890123451
ATM Card No	1234567890123451
Full Name	MAAZ
Address	BGM
E-Mail	maaz@gmail.com
Mobile No	9590172717
Credit Amount	5000
Balance	0.00

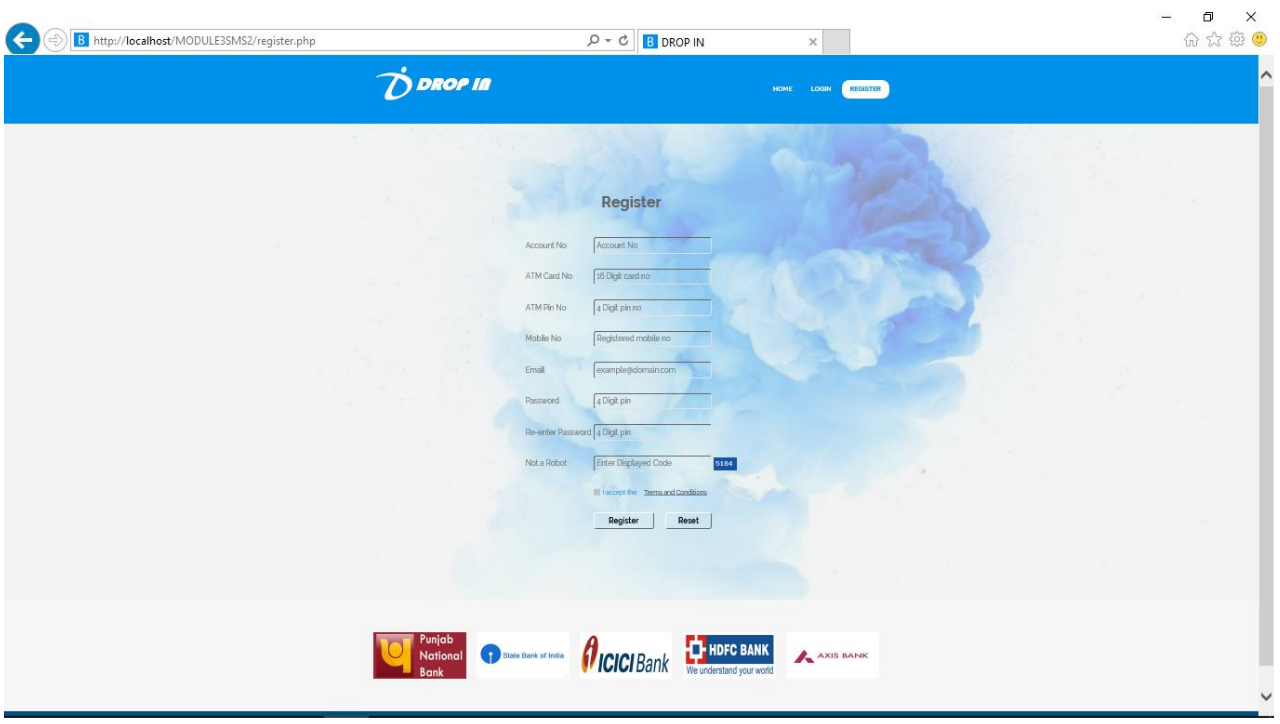
MODULE 3 :



http://localhost/MODULE3SMS2/index.php

THE BEST WAY YOU CAN FIND TO TRANSFER MONEY


http://localhost/MODULE3SMS2/index.php
DROP IN
ONLINE HELP FOR GROWING YOUR BANKING
State Bank of India
ICICI Bank
HDFC BANK
We understand your world
AXIS BANK
Canara Bank
© Copyright 2018 Drop In. All right reserved.

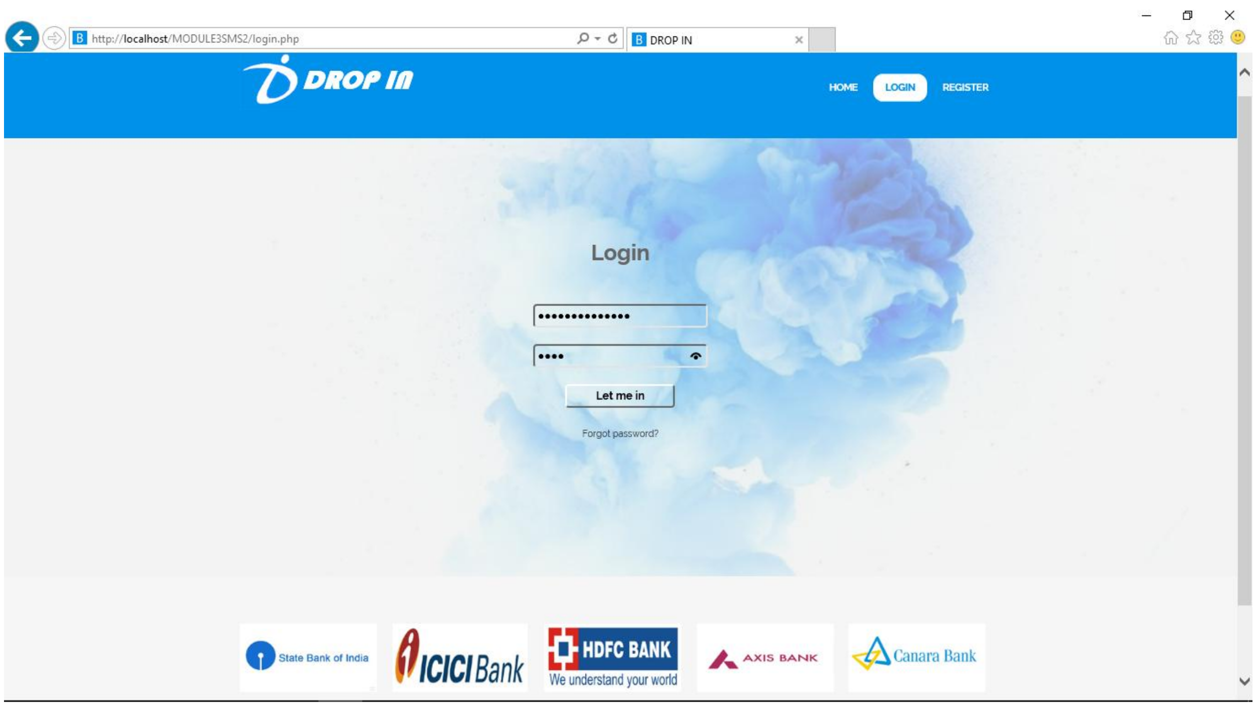


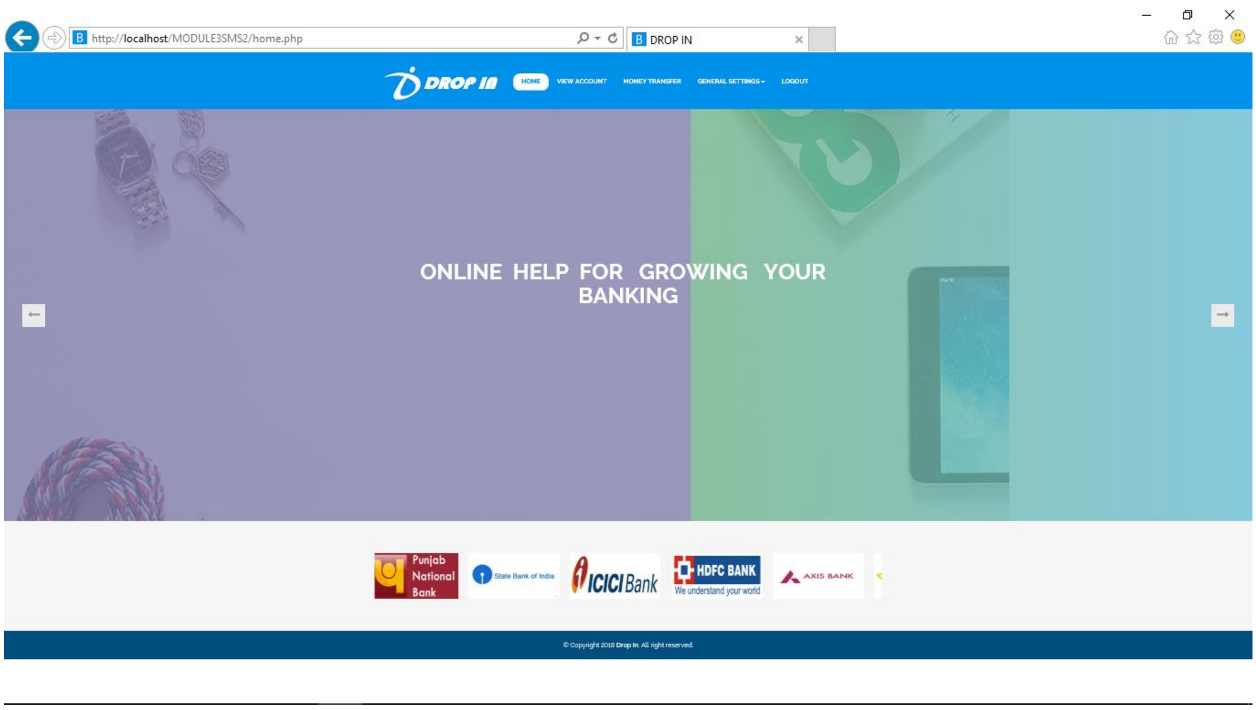
Register

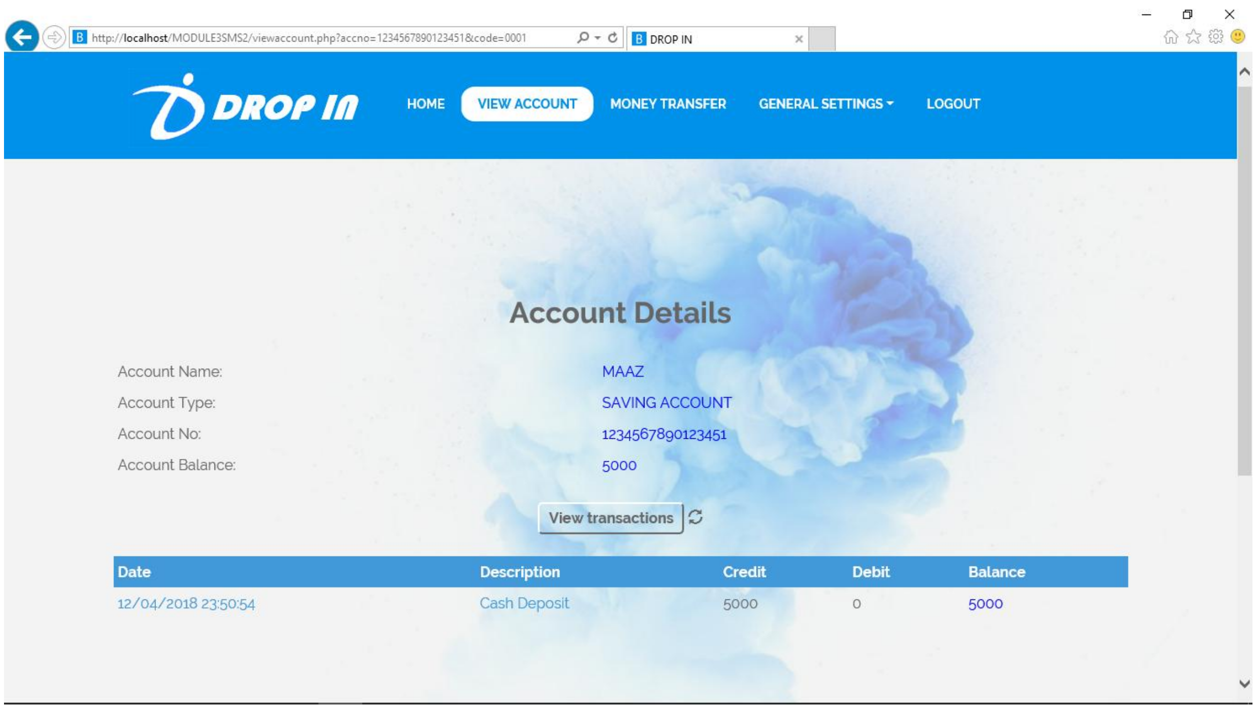
Account No	Account No
ATM Card No	16 Digit card no
ATM Pin No	4 Digit pin no
Mobile No	Registered mobile no
Email	example@domain.com
Password	4 Digit pin
Re-enter Password	4 Digit pin
Not a Robot	Enter Displayed Code 5184

I accept the Terms and Conditions

Register Reset


http://localhost/MODULE3SMS2/login.php
DROP IN
DROP IN
HOME
LOGIN
REGISTER
Login
••••••••••••••
••••
Let me in
Forgot password?
State Bank of India
ICICI Bank
HDFC BANK
We understand your world
AXIS BANK
Canara Bank

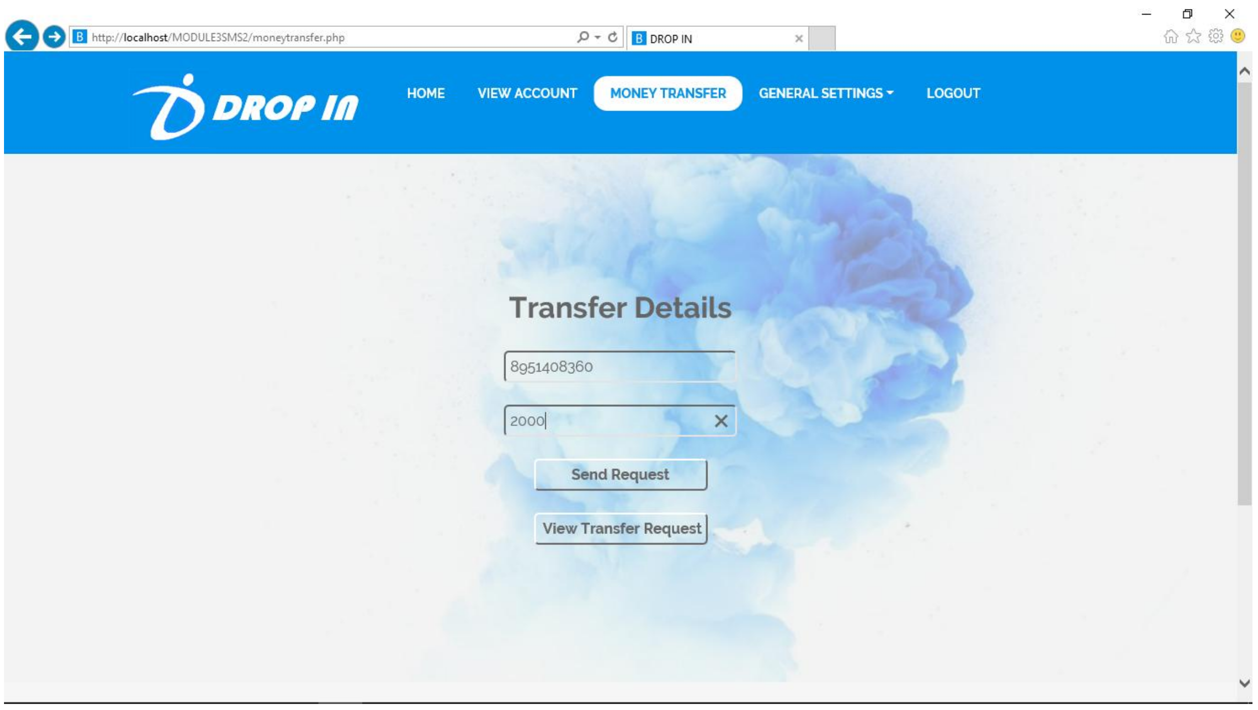

http://localhost/MODULE3SMS2/home.php
DROP IN
DROP IN
HOME
VIEW ACCOUNT
MONEY TRANSFER
GENERAL SETTINGS
LOGOUT
ONLINE HELP FOR GROWING YOUR BANKING
Punjab National Bank
State Bank of India
ICICI Bank
HDFC BANK
We understand your world
AXIS BANK
© Copyright 2018 Drop In. All right reserved.



Account Details

Account Name:	MAAZ
Account Type:	SAVING ACCOUNT
Account No:	1234567890123451
Account Balance:	5000

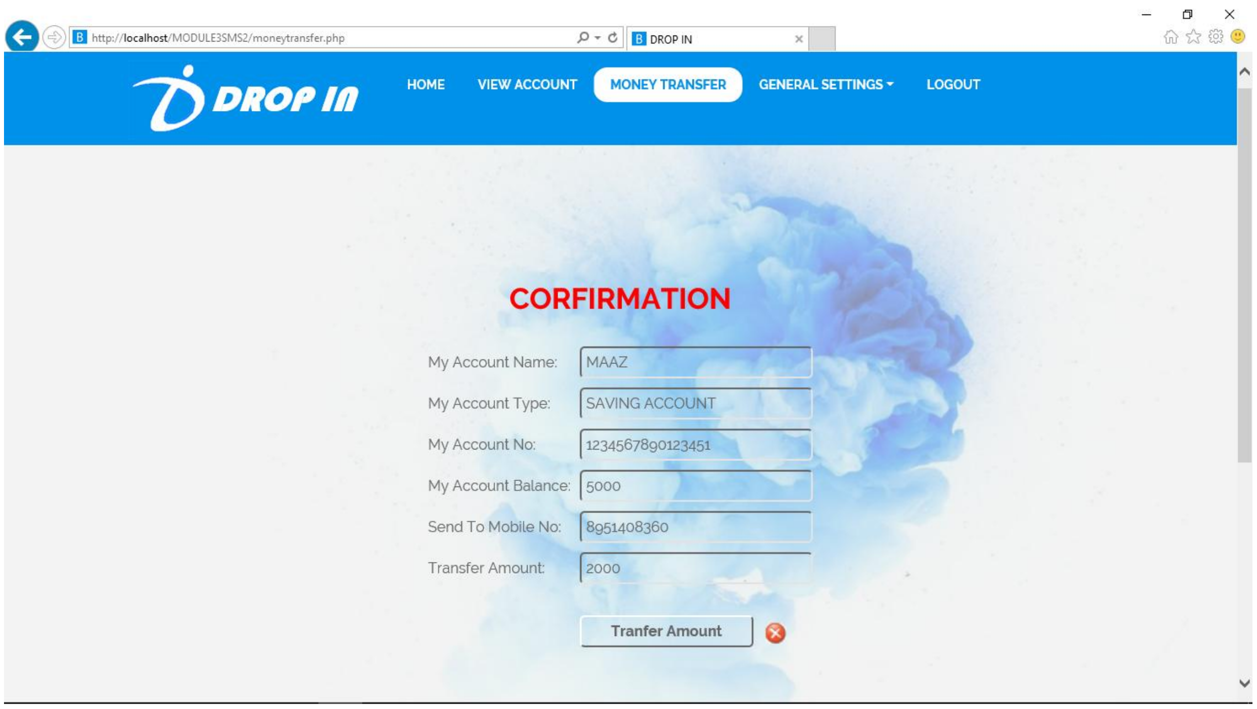
Date	Description	Credit	Debit	Balance
12/04/2018 23:50:54	Cash Deposit	5000	0	5000



Transfer Details

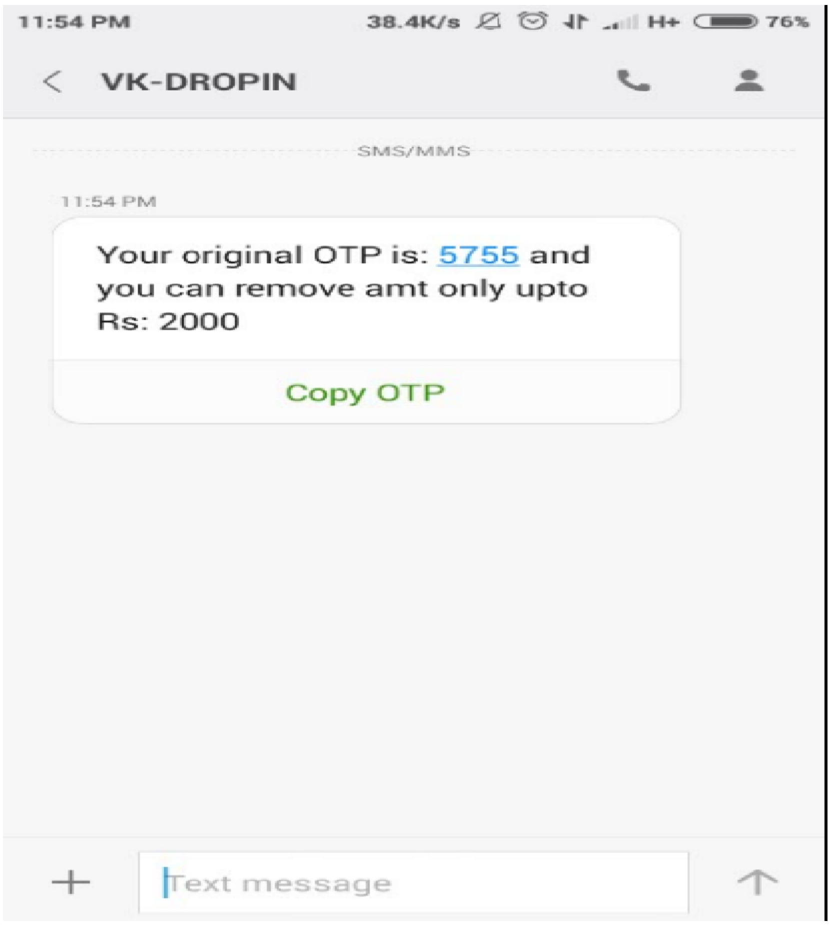
8951408360

2000



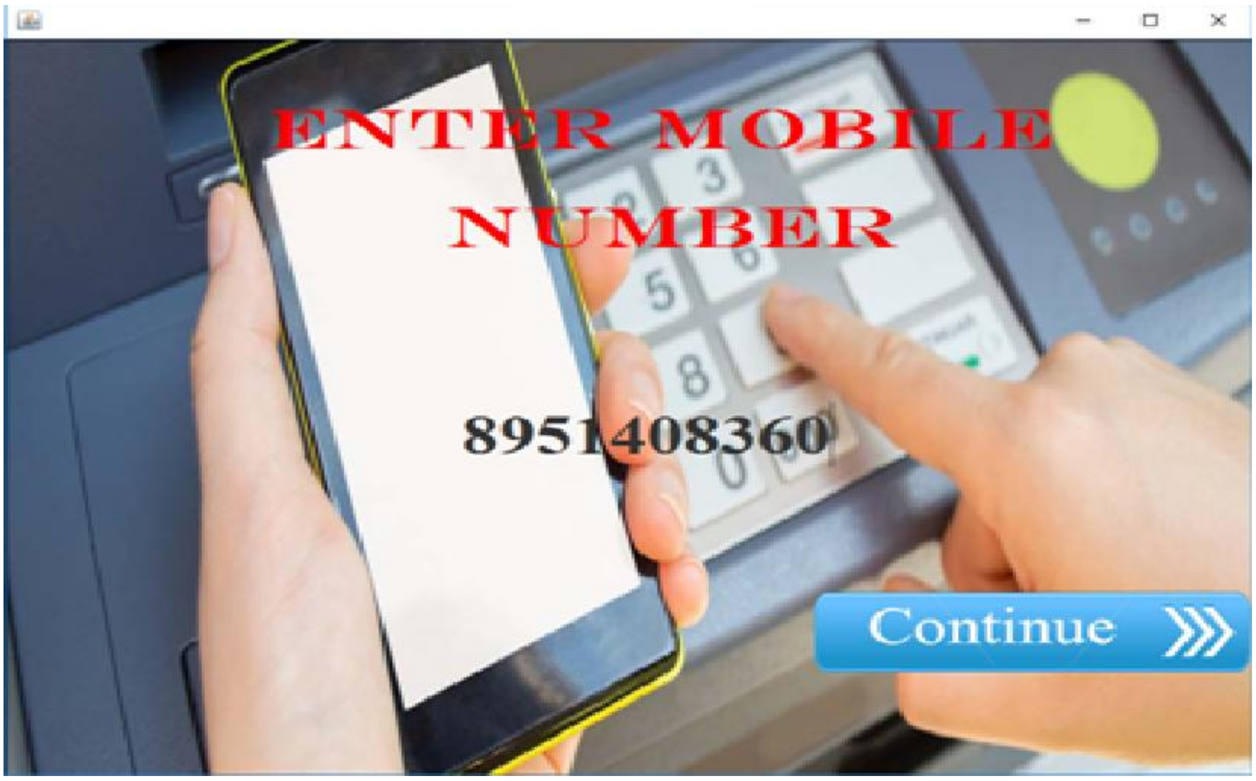
CORFIRMATION

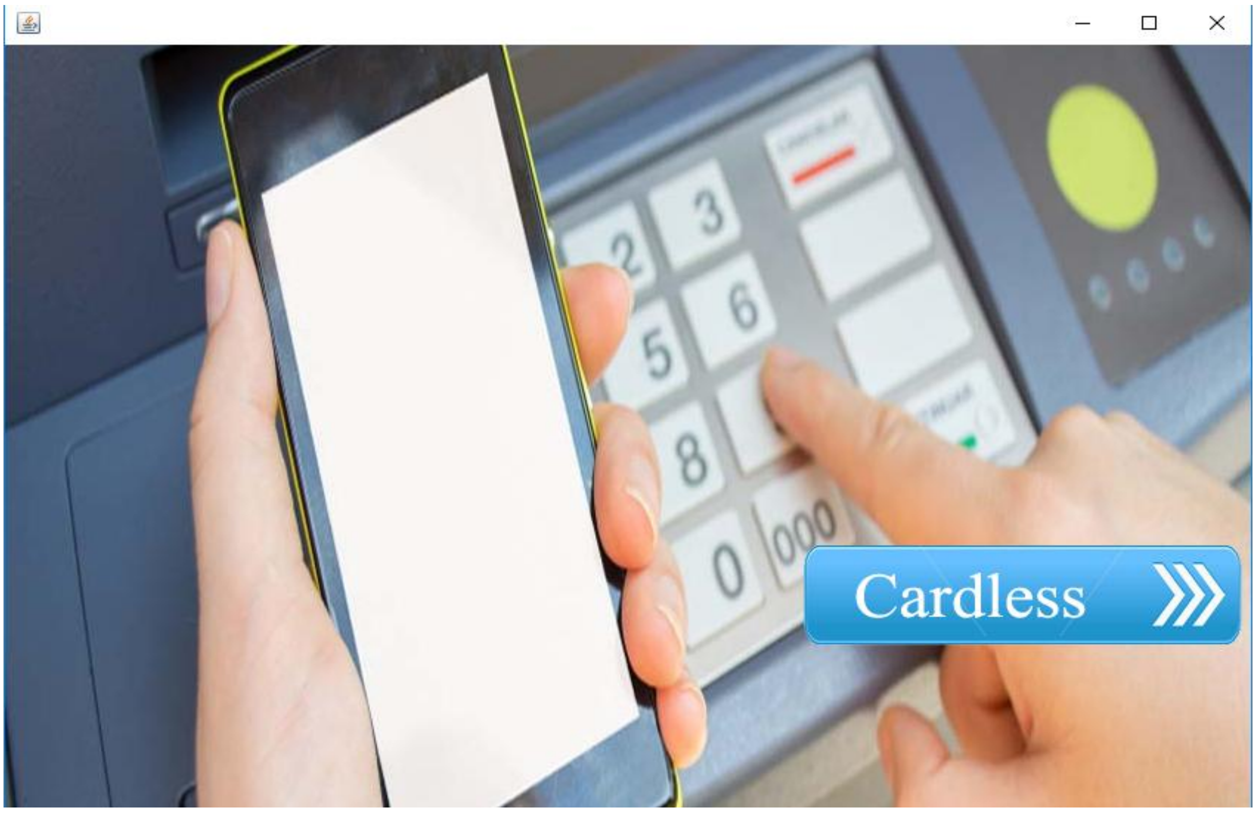
My Account Name:	MAAZ
My Account Type:	SAVING ACCOUNT
My Account No:	1234567890123451
My Account Balance:	5000
Send To Mobile No:	8951408360
Transfer Amount:	2000

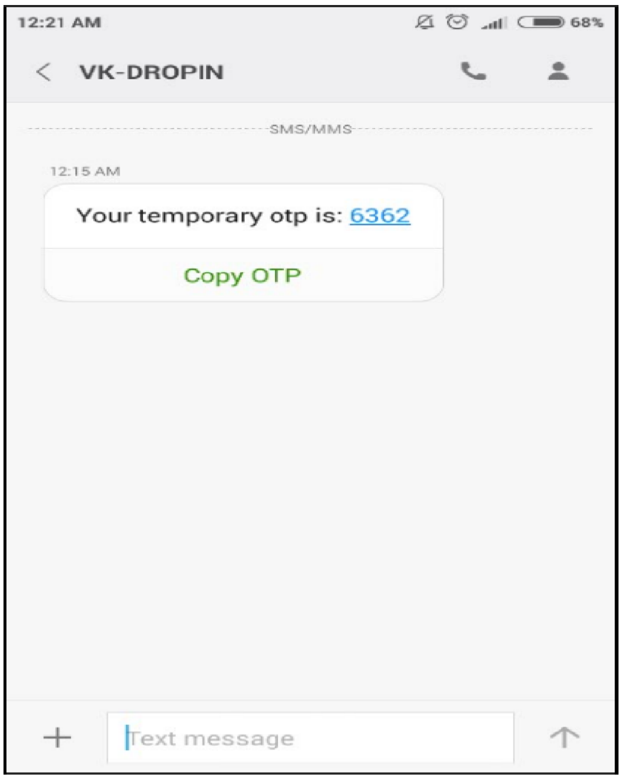


Your original OTP is: 5755 and you can remove amt only upto Rs: 2000

MODULE 4 :







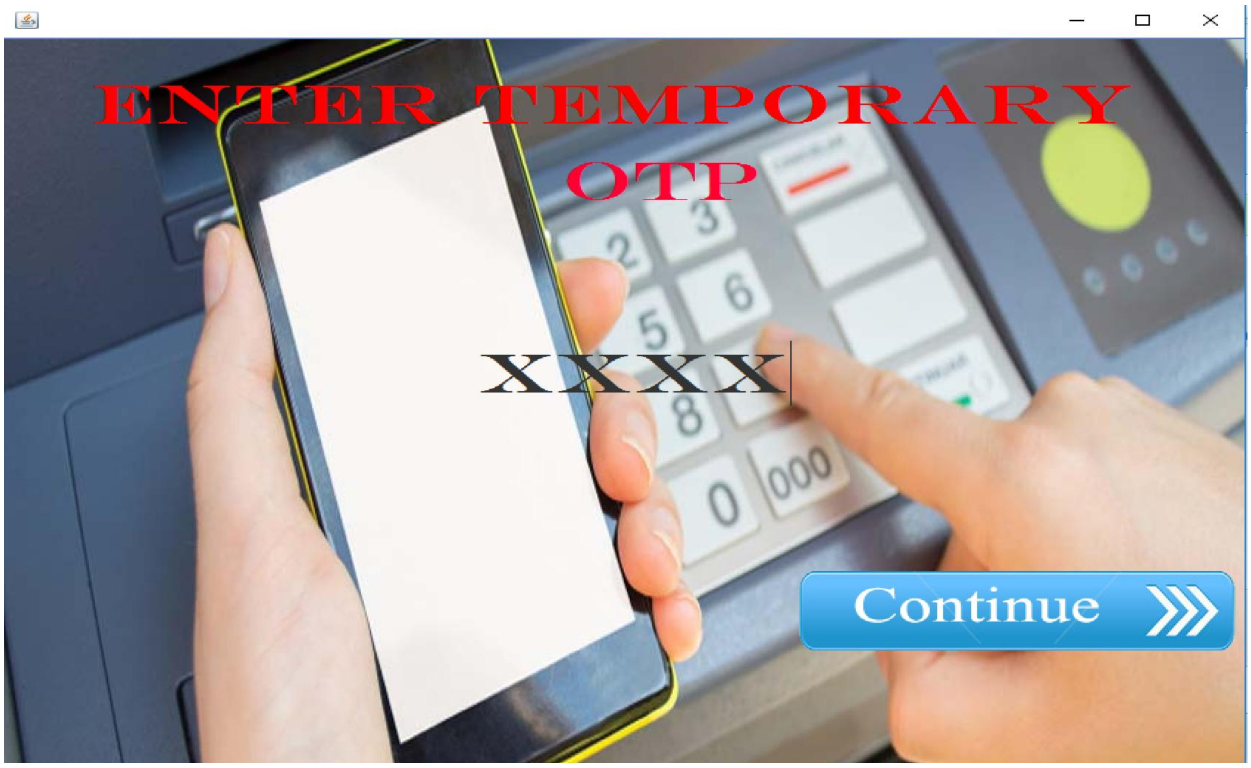
12:21 AM 68%

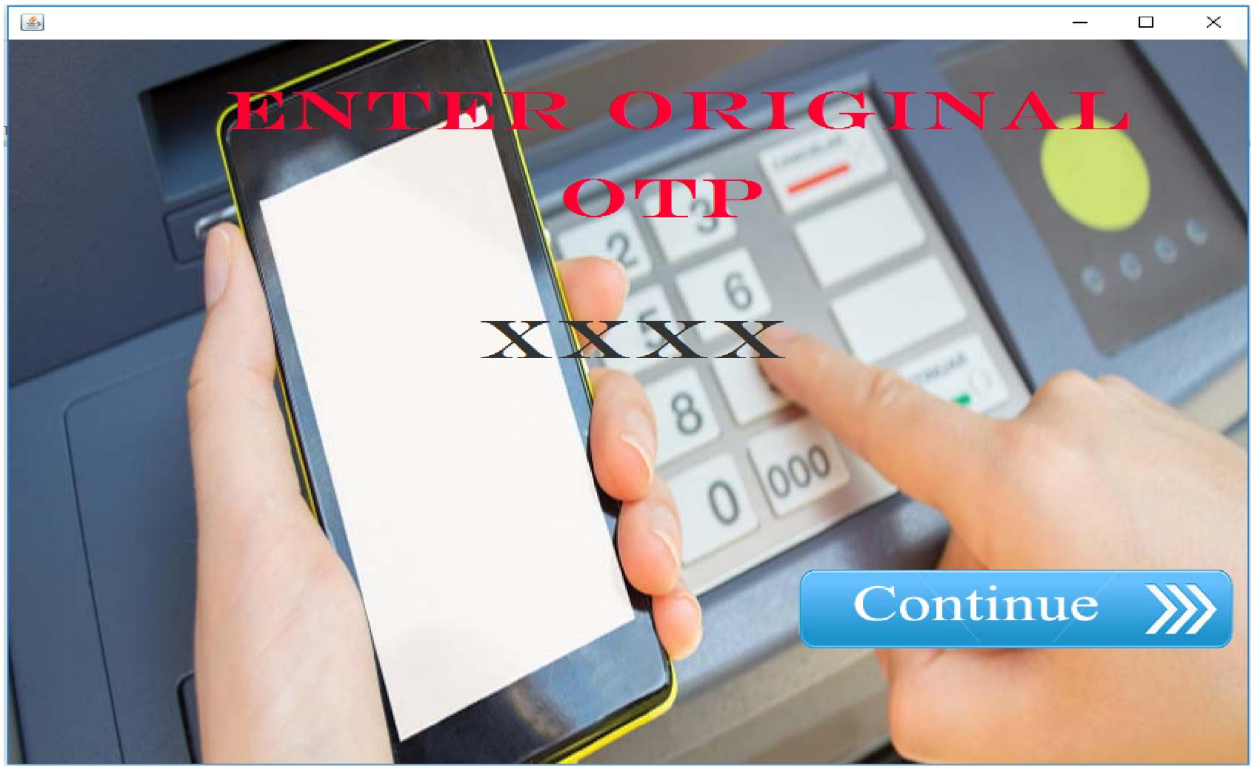
VK-DROPIN

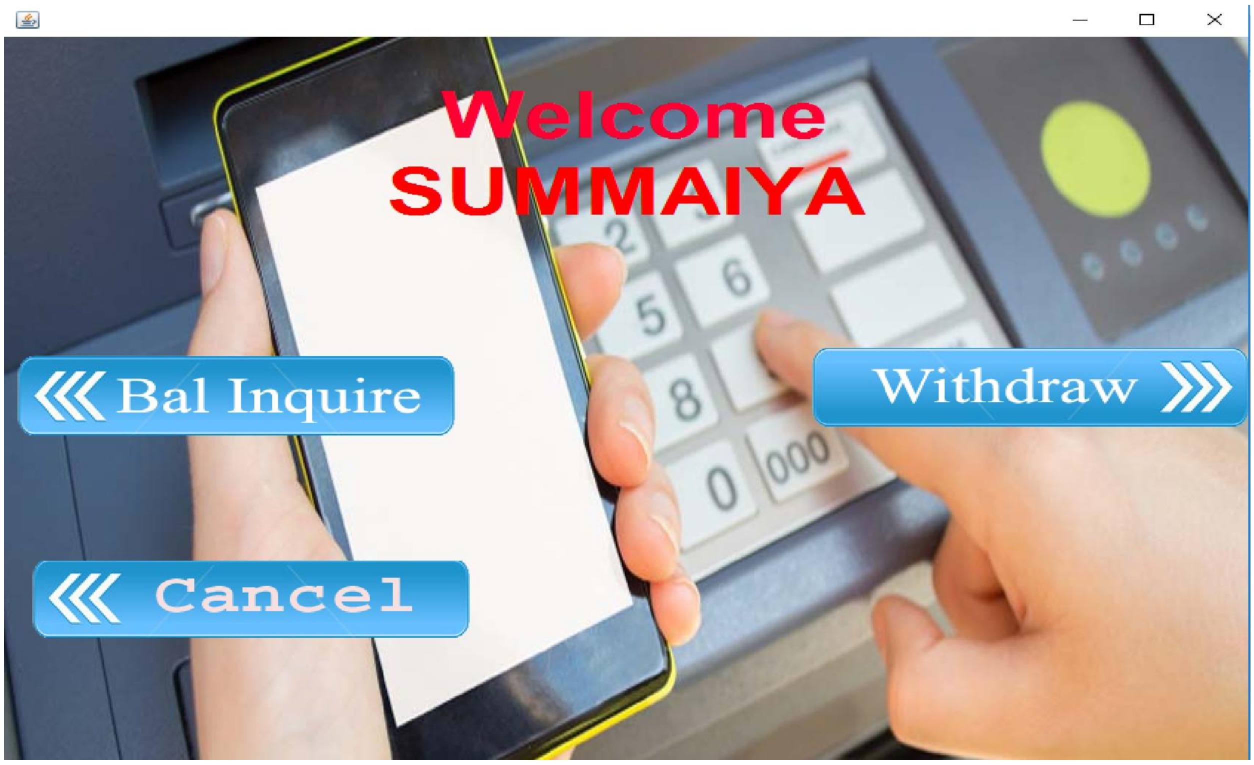
SMS/MMS

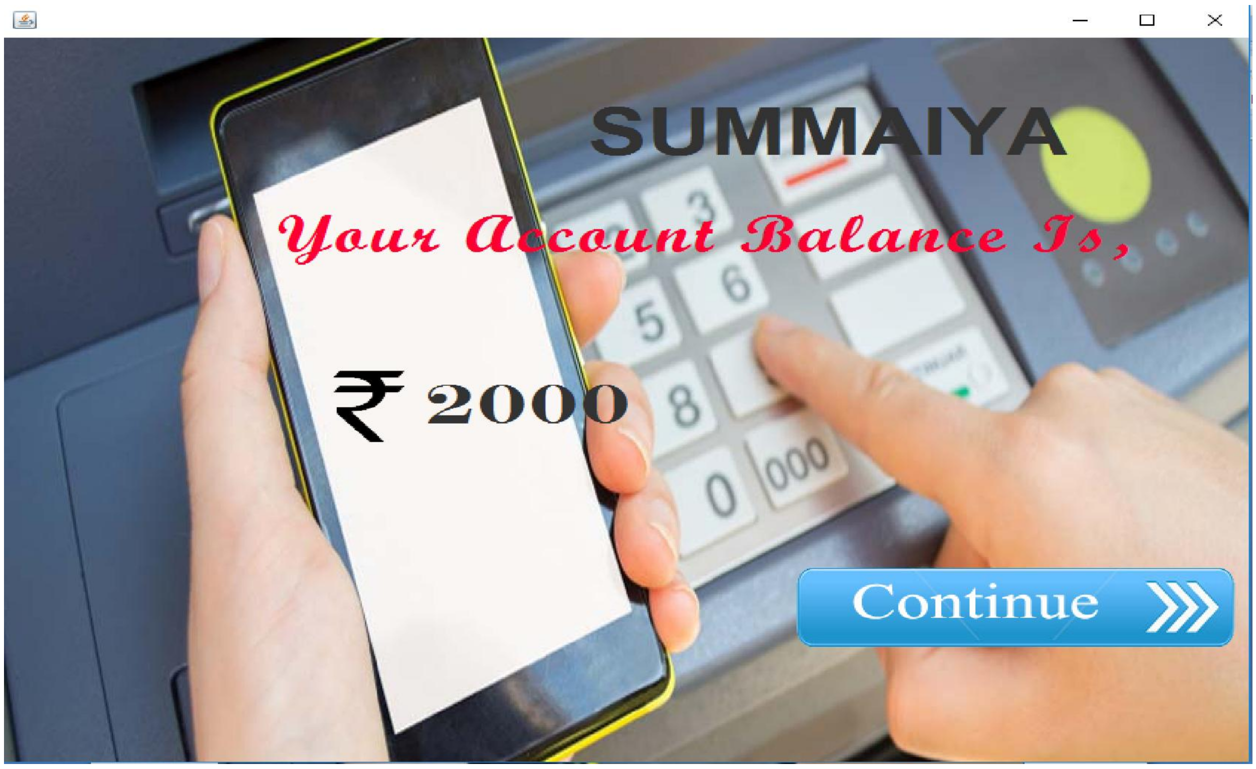
12:15 AM

Your temporary otp is: 6362



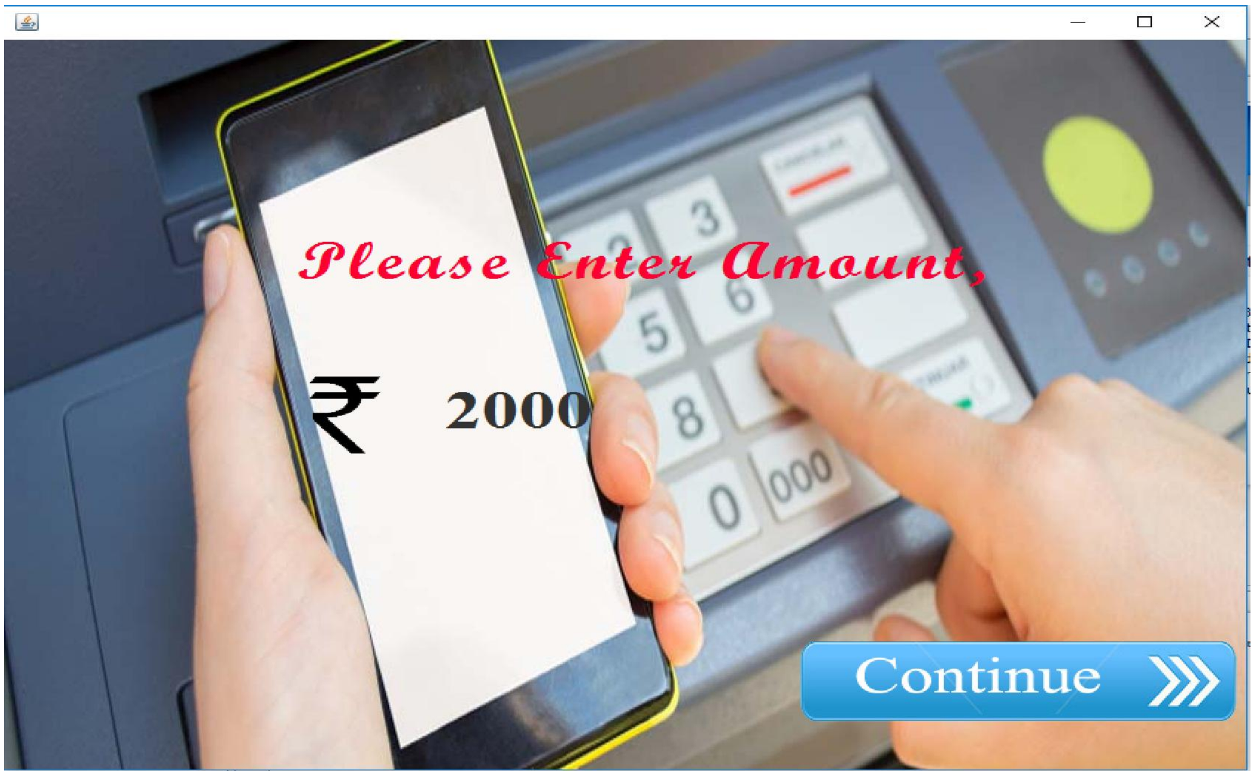

ENTER ORIGINAL
OTP
XXXX
Continue


Welcome
SUMMAIYA
Bal Inquire
Withdraw
Cancel

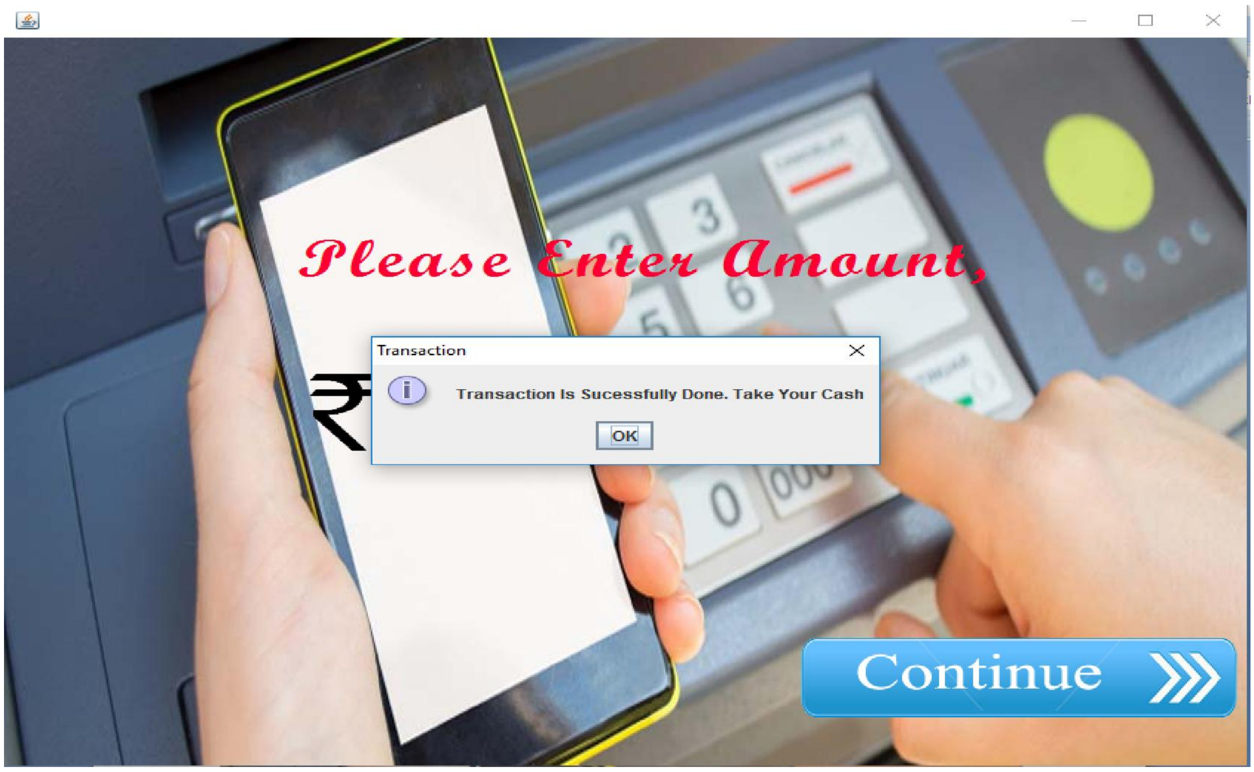

SUMMAIYA

Your Account Balance Is,

₹ 2000


Please Enter Amount,

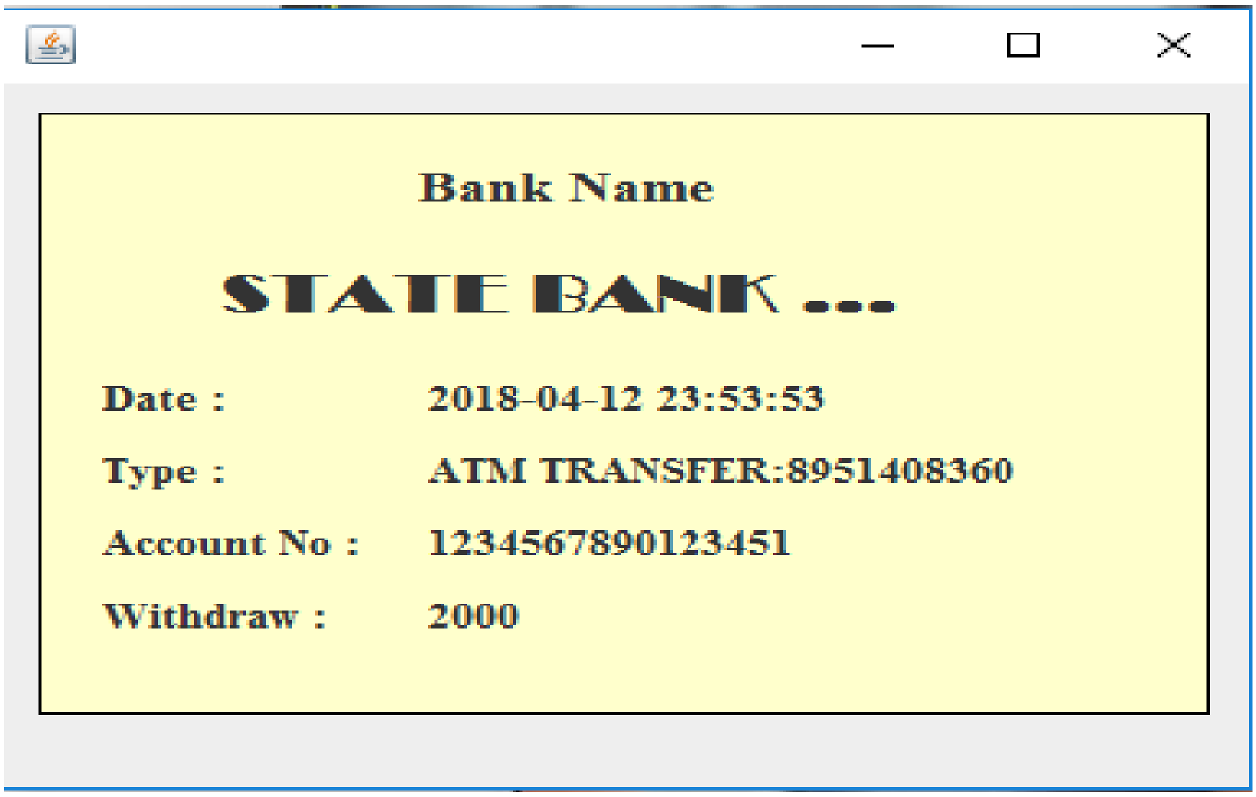
₹ 2000



Transaction

Transaction Is Sucessfully Done. Take Your Cash

OK



Bank Name

STATE BANK ...

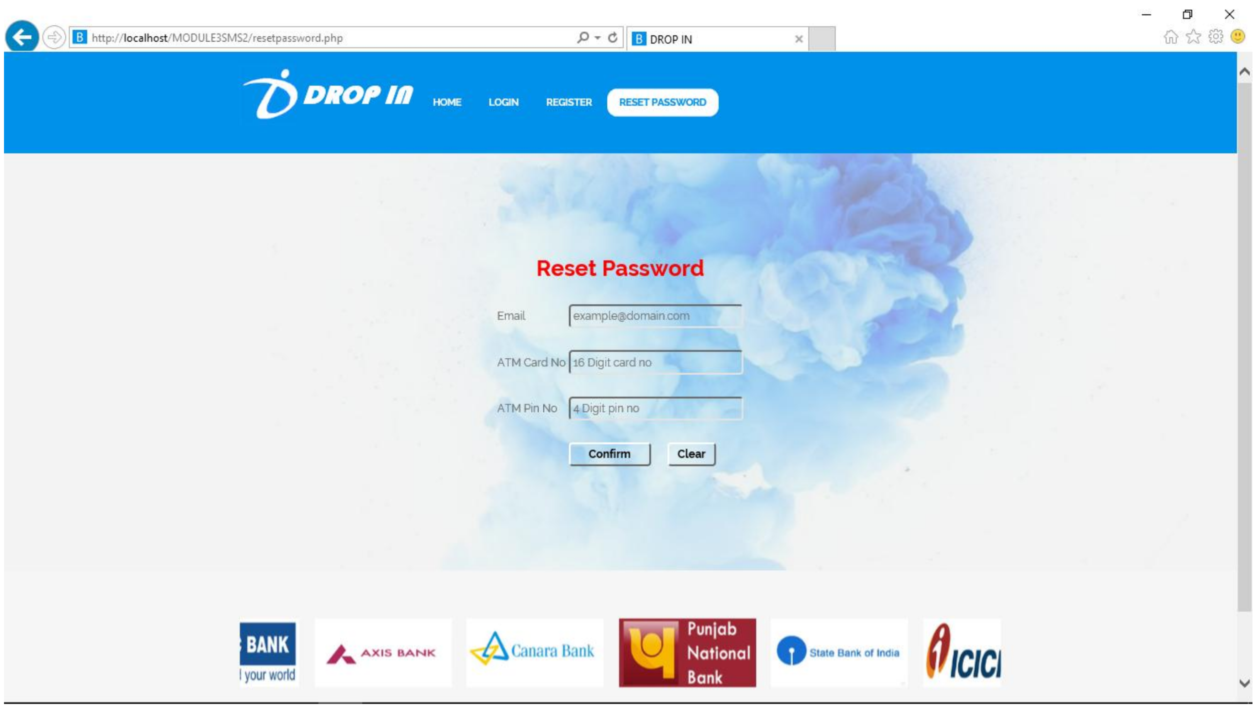
Date : 2018-04-12 23:53:53

Type : ATM TRANSFER:8951408360

Account No : 1234567890123451

Withdraw : 2000

Security features:

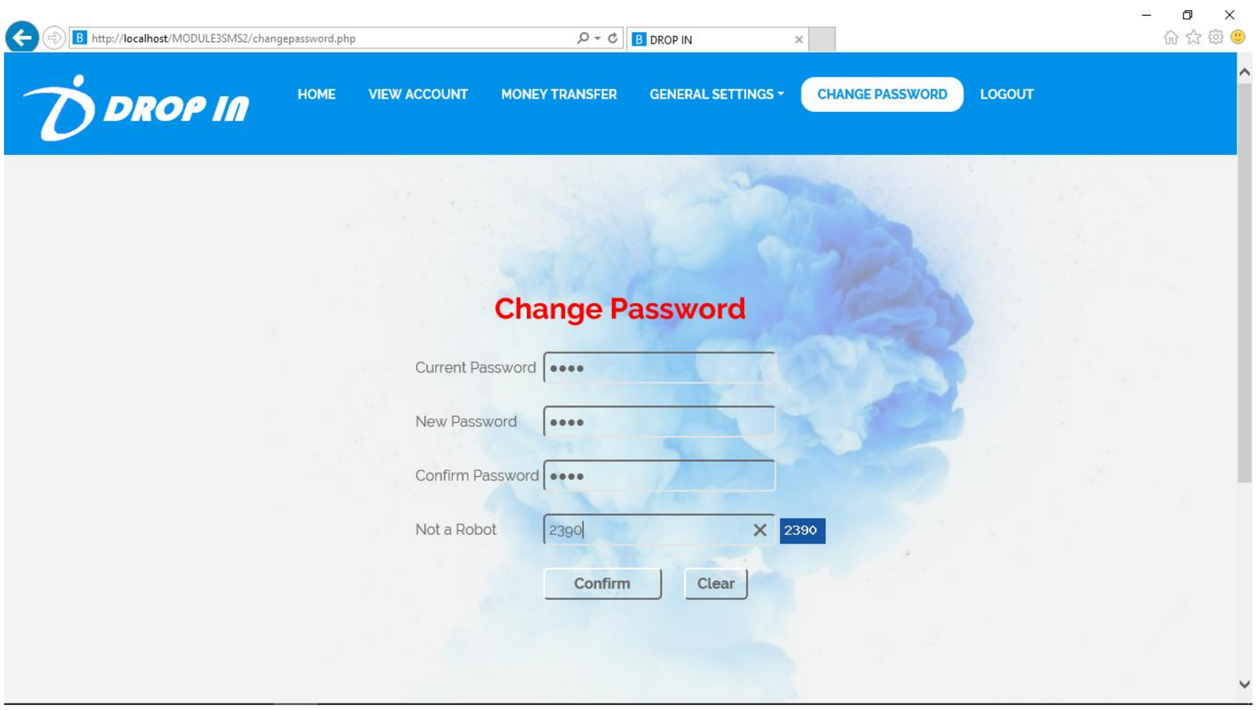


Reset Password

Email: example@domain.com

ATM Card No: 16 Digit card no

ATM Pin No: 4 Digit pin no



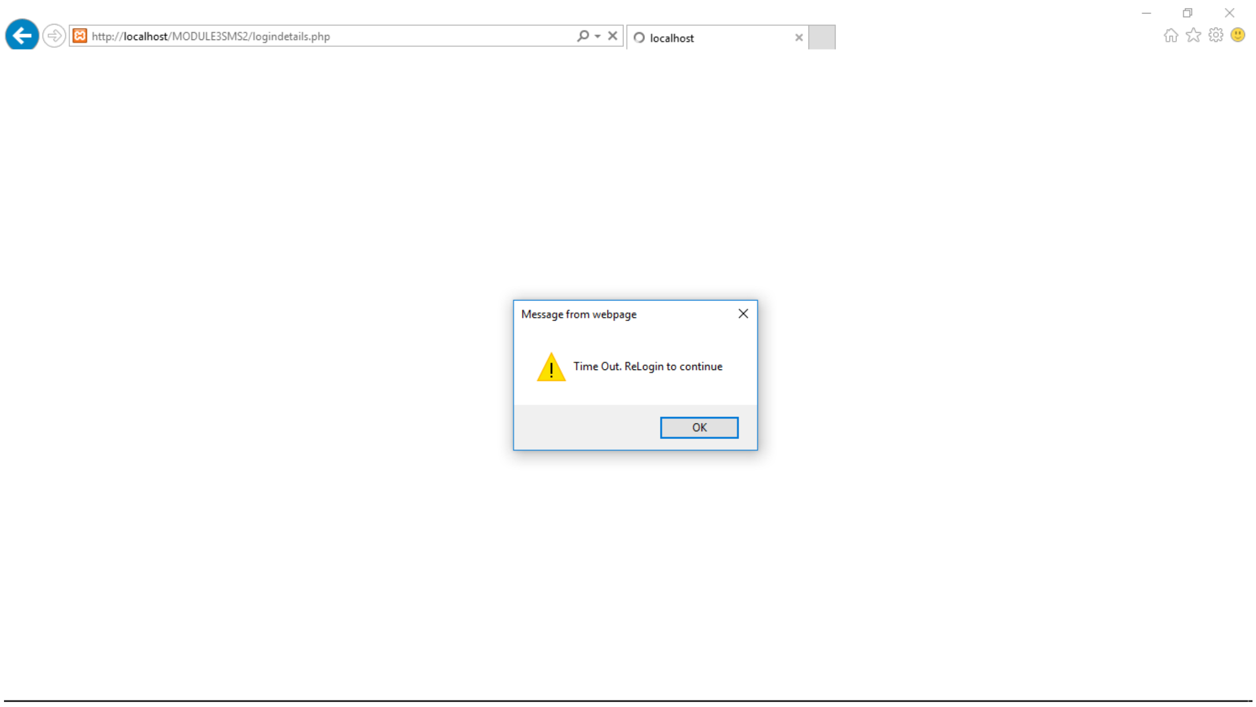
Change Password

Current Password: ••••

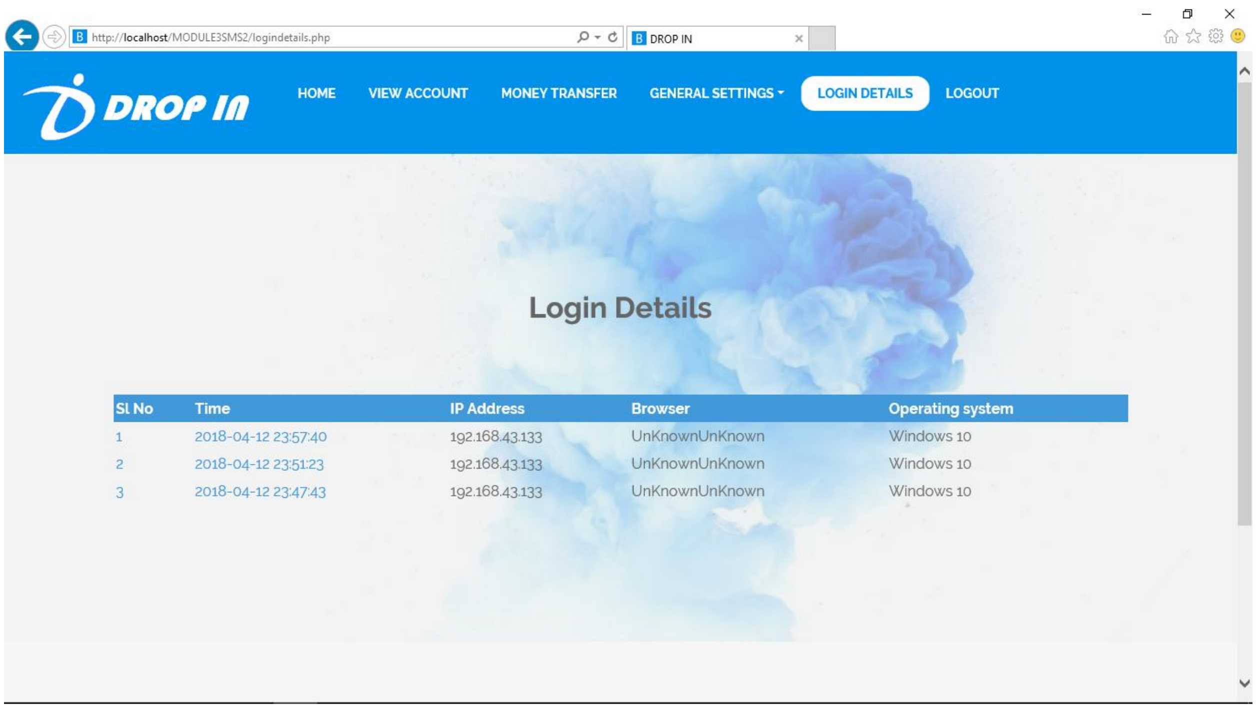
New Password: ••••

Confirm Password: ••••

Not a Robot: 2390 2390



Time Out. ReLogin to continue



Login Details

Sl No	Time	IP Address	Browser	Operating system
1	2018-04-12 23:57:40	192.168.43.133	UnKnownUnKnown	Windows 10
2	2018-04-12 23:51:23	192.168.43.133	UnKnownUnKnown	Windows 10
3	2018-04-12 23:47:43	192.168.43.133	UnKnownUnKnown	Windows 10

TESTING

Testing

Software Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not.

Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

This tutorial will give you a basic understanding on software testing, its types, methods, levels, and other related terminologies.

It depends on the process and the associated stakeholders of the project(s). In the IT industry, large companies have a team with responsibilities to evaluate the developed software in context of the given requirements. Moreover, developers also conduct testing which is called **Unit Testing**. In most cases, the following professionals are involved in testing a system within their respective capacities:

- Software Tester
- Software Developer
- Project Lead/Manager
- End User

Different companies have different designations for people who test the software on the basis of their experience and knowledge such as Software Tester, Software Quality Assurance Engineer, QA Analyst, etc.It is not possible to test the software at any time during its cycle. The next two sections state when testing should be started and when to end it during the SDLC.

HISTORY

The separation of debugging from testing was initially introduced by Glenford J. Myers in 1979.Although his attention was on breakage testing ("A successful test case is one that detects an as-yet undiscovered error. it illustrated the desire of the software engineering community to separate fundamental development activities, such as debugging, from that of verification.

Software testing began in the 1950s and it was basic. However it was found that the kind of testing that was being done was not able to find all the bugs. In the 1960s and 1970s a method that was known as correctness proof was developed that was designed but it was time consuming and it did not de-bug the system sufficiently. The next method of software testing was developed in the late 1970s and this method tried to find errors in the system that had not yet been discovered. This was an improvement of the previous method that tried to remove errors that had already been discovered. The way these types of software testing worked is by testing the program as if there were errors to void assuming that there are no errors which would cause the programs to thoroughly inspect each part of the program as well as inspect each test results.

In the 1980s defect prevention was included in the process of software testing. This was defined as designing tests and it was a huge improvement on the other methods of testing that existed until then. In this method of testing reviews of the whole program was done as it was developed. This included testing the requirements of the program, the design of the program, the code, the program as well as the tests that were carried out to ensure that they were effective.

Computers could do more tests compared to people which was how efficiency was increased. In the 1990s the early test design was developed. This mode of testing was done by doing tests in the planning, design, building, maintaining, and in the testing of the program. It also involved quality assurance of the program. Other methods of testing that were developed at this time were testing tools that were more advanced in scripting languages and reporting facilities.

In the 2000s a method of testing known as BTO (Business Technology Optimization) was developed. In this method of testing, the business goals of an organization's program and the effectiveness of the program were taken into consideration. The way this method was developed to work is to measure the value of the program and maximize it by enhancing the way the program works.

White box testing

Sometimes called glass-box testing is a test case design method that uses the control structure of the procedural design to derive test cases. Using white-box testing methods, the software engineer can derive test cases that

(1) guarantee that all independent paths within a module have been exercised at least once,

(2) exercise all logical decisions on their true and false sides,

(3) execute all loops at their boundaries and within their operational bounds, and

(4) exercise internal data structures to ensure their validity.

White-box testing of software is predicated on close examination of procedural detail. Providing test cases that exercise specific sets of conditions and/or loops tests logical paths through the software. The “status of the program” may be examined at various points to determine if the expected or asserted status corresponds to the actual status. Basis path testing is a white-box testing technique first proposed by Tom McCabe. The basis path method enables the test case designer to derive a logical complexity measure of a procedural design and use this measure as a guide for defining a basis set of execution paths. Test cases derived to exercise the basis set are guaranteed to execute every statement in the program at least one time during testing.

In this system, the system was tested for the calculation matters were the data provided for giving the right output or not. If wrong data was provided then what it is throwing error or accepting.

Black box testing

Also called behavioral testing, focuses on the functional requirements of the software. That is, black box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program. Black box testing is not an alternative to white-box techniques. Rather, it is a complementary approach that is likely to uncover a different class of error than white-box methods. When computer software is considered, black box testing alludes to tests that are conducted at the software interface. Although they are designed to uncover errors, black-box tests are used to demonstrate that software functions are operational, that input is

Properly accepted and output is correctly produced and that the integrity of external information is maintained. A black-box test examines some fundamental aspect of a system with a little regard for the internal logical structure of the software. Black-box testing attempts to find errors in the following categories:

1. Incorrect or missing functions,
2. Interface errors,
3. Errors in data structures or external database access,
4. Behavior or performance errors, and
5. Initialization and termination errors. By applying back-box techniques, we derive a set of test cases that satisfy the following criteria:
    - Test cases that reduce, by a count that is greater than one, the number of additional test cases that must be designed to achieve reasonable testing and
    - Test cases that tell us something about the presence or absence of classes of errors, rather than an error associated only with the specific test at hand.

White-box testing should not, however, be dismissed as impractical. A limited number of important logical paths can be selected and exercised. Important data structures can be probed for validity. The attributes of both black and white box testing can be combined to provide an approach that validates the software interface and selectively ensures that the internal workings of the software are correct.

Black box testing for this system was done to check the internal testing i.e, the system is working properly in each case or no. What kind of errors are there in database design.

Static vs. dynamic testing

There are many approaches available in software testing. Reviews, walkthroughs, or inspections are referred to as static testing, whereas actually executing programmed code with a given set of test cases is referred to as dynamic testing. Static testing is often implicit, as proofreading, plus when programming tools/text editors check source code structure or compilers (pre-compilers) check syntax and data flow as static program analysis. Dynamic testing takes place when the program itself is run. Dynamic testing may begin before the program is 100% complete in order to test particular sections of code and are applied to discrete functions or modules. Typical techniques for this are either using stubs/drivers or execution from a debugger environment.

Static testing involves verification, whereas dynamic testing also involves validation. Together they help improve software quality. Among the techniques for static analysis, mutation testing can be used to ensure the test cases will detect errors that are introduced by mutating the source code.

White-box testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing and structural testing, by seeing the source code) tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user. In white-box testing, an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT).

While white-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it might not detect unimplemented parts of the specification or missing requirements.

Techniques used in white-box testing include:

- API testing – testing of the application using public and private APIs (application programming interfaces)
- Code coverage – creating tests to satisfy some criteria of code coverage (e.g., the test designer can create tests to cause all statements in the program to be executed at least once)
- Fault injection methods – intentionally introducing faults to gauge the efficacy of testing strategies
- Mutation testing methods
- Static testing methods

Code coverage tools can evaluate the completeness of a test suite that was created with any method, including black-box testing. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested. Code coverage as a software metric can be reported as a percentage for:

- Function coverage, which reports on functions executed
- Statement coverage, which reports on the number of lines executed to complete the test
- Decision coverage, which reports on whether both the True and the False branch of a given test has been executed

100% statement coverage ensures that all code paths or branches (in terms of control flow) are executed at least once. This is helpful in ensuring correct functionality, but not sufficient since the same code may process different inputs correctly or incorrectly.

Black-box testing

Black-box testing treats the software as a "black box", examining functionality without any knowledge of internal implementation, without seeing the source code. The testers are only aware of what the software is supposed to do, not how it does it. Black-box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, state transition tables, decision table testing, fuzz testing, model-based testing, use case testing, exploratory testing, and specification-based testing.

Testing levels

There are generally four recognized levels of tests: unit testing, integration testing, component interface testing, and system testing. Tests are frequently grouped by where they are added in the software development process, or by the level of specificity of the test. The main levels during the development process as defined by the SWEBOK guide are unit-, integration-, and system testing that is distinguished by the test target without implying a specific process model. Other test levels are classified by the testing objective. There are two different levels of tests from the perspective of customers: low-level testing (LLT) and high-level testing (HLT). LLT is a group of tests for different level components of software application or product. HLT is a group of tests for the whole software application or product.

Unit testing

Unit testing refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors.

These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected. One function might have multiple tests, to catch corner cases or other branches in the code. Unit testing alone cannot verify the functionality of a piece of software, but rather is used to ensure that the building blocks of the software work independently from each other.

Unit testing is a software development process that involves a synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time, and costs. It is performed by the software developer or engineer during the construction phase of the software development lifecycle. Unit testing aims to eliminate construction errors before code is promoted to additional testing; this strategy is intended to increase the quality of the resulting software as well as the efficiency of the overall development process.

Depending on the organization's expectations for software development, unit testing might include static code analysis, data-flow analysis, metrics analysis, peer code reviews, code coverage analysis and other software testing practices.

Integration testing

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be located more quickly and fixed.

Integration testing works to expose defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system.

Component interface testing

The practice of component interface testing can be us+ed to check the handling of data passed between various units, or subsystem components, beyond full integration testing between those units. The data being passed can be considered as "message packets" and the range or data types can be checked, for data generated from one unit, and tested for validity before being passed into another unit. One option for interface testing is to keep a separate log file of data items being passed, often with a timestamp logged to allow analysis of thousands of cases of data passed between units for days or weeks. Tests can include checking the handling of some extreme data values while other interface variables are passed as normal values. Unusual data values in an interface can help explain unexpected performance in the next unit. Component interface testing is a variation of black-box testing, with the focus on the data values beyond just the related actions of a subsystem component.

System testing

System testing tests a completely integrated system to verify that the system meets its requirements. For example, a system test might involve testing a logon interface, then creating and editing an entry, plus sending or printing results, followed by summary processing or deletion (or archiving) of entries, then logoff.

Operational acceptance testing

Operational acceptance is used to conduct operational readiness (pre-release) of a product, service or system as part of a quality management system. OAT is a common type of non-functional software testing, used mainly in software development and software maintenance projects. This type of testing focuses on the operational readiness of the system to be supported, or to become part of the production environment. Hence, it is also known as operational readiness testing (ORT) or Operations readiness and assurance (OR&A) testing. Functional testing within OAT is limited to those tests that are required to verify the non-functional aspects of the system.

In addition, the software testing should ensure that the portability of the system, as well as working as expected, does not also damage or partially corrupt its operating environment or cause other processes within that environment to become inoperative.

Testing artifacts

A software testing process can produce several artifacts. The actual artifacts produced are a factor of the software development model used, stakeholder and organizational needs.

Test plan

A test plan is a document detailing the objectives, target market, internal beta team, and processes for a specific beta test. The developers are well aware what test plans will be executed and this information is made available to management and the developers. The idea is to make them more cautious when developing their code or making additional changes. Some companies have a higher-level document called a test strategy.

Traceability matrix

A traceability matrix is a table that correlates requirements or design documents to test documents. It is used to change tests when related source documents are changed, to select test cases for execution when planning for regression tests by considering requirement coverage.

Test case

A test case normally consists of a unique identifier, requirement references from a design specification, preconditions, events, a series of steps (also known as actions) to follow, input, output, expected result, and the actual result. Clinically defined, a test case is an input and an expected result. This can be as terse as 'for condition x your derived result is y', although normally test cases describe in more detail the input scenario and what results might be expected. It can occasionally be a series of steps (but often steps are contained in a separate test procedure that can be exercised against multiple test cases, as a matter of economy) but with one expected result or expected outcome. The optional fields are a test case ID, test step, or order of execution number, related requirement(s), depth, test category, author, and check boxes for whether the test is automatable and has been automated. Larger test cases may also contain prerequisite states or steps, and descriptions. A test case should also contain a place for the actual result. These steps can be stored in a word processor document, spreadsheet, database, or other common repositories. In a database system, you may also be able to see past test results, who generated the results, and what system configuration was used to generate those results. These past results would usually be stored in a separate table.

Test script

A test script is a procedure or programming code that replicates user actions. Initially, the term was derived from the product of work created by automated regression test tools. A test case will be a baseline to create test scripts using a tool or a program.

Test suite

The most common term for a collection of test cases is a test suite. The test suite often also contains more detailed instructions or goals for each collection of test cases. It definitely contains a section where the tester identifies the system configuration used during testing. A group of test cases may also contain prerequisite states or steps, and descriptions of the following tests.

Test fixture or test data

In most cases, multiple sets of values or data are used to test the same functionality of a particular feature. All the test values and changeable environmental components are collected in separate files and stored as test data. It is also useful to provide this data to the client and with the product or a project.

Test harness

The software, tools, samples of data input and output, and configurations are all referred to collectively as a test harness.

REFERENCE

Reference

- http://www.w3schools.com
- http://php.net
- http://tutorialspoint.com
- www.codecademy.com/en/tracks/php