Mobile Banking System in India: Practices,Challenges and Security Issues

Ravi Gupta[1], Mr. R. Praveen Kumar[2], Anuj Bharadwaj[3]

[#1]*Research Scholar,S.D.B.C.T, Indore India*
[*2]*Mr. R. Praveen Kumar, Reasion IT, Hyderabad India*
[@3] *Research Scholar,S.D.B.C.T, Indore India*

ABSTRACT
The objective of this application is to provide a net banking facility to all potential customers. Using this facility customer can quickly complete their transactions online. The purpose is to design a system which simplifies the banking related operations. In present system all traditional works are done either manually or using Internet. This requires customers to visit the facility to complete the work or one has to access an Internet facility. This often wastes time and results in more effort. The proposed system is a web based application and provides a centralized database with all related information. The application is a banking facility through which all payments can be done at a single place. The system provides a number of interfaces for different functionalities and operations. This enables customers to quickly complete all works and also to verify the results of various transactions.

Key Words: *Check balance, transfer funds, stop payment, request Cheque-book, mini statement and change pin in india.*

1. INTRODUCTION

Mobile Banking is a kind of financial services where ABC provides customers with information inquiry, transfer and remittance, payment, credit card, roaming remittance, loans to farmer households, time-demand deposits transfer, third-party depository, customized message, account management, personal setting, etc. by making using of wireless networks and mobile phones. ABC launches two versions of Mobile Banking based on different user operation procedures and system interface displays: WAP Smooth Version and 4G Fashion Version. Although financial services provided by the two versions are the same, WAP Smooth Version is designed to enhance access speed, with concise text explanations and links, while 4G Fashion Version aims to boost application experience of users and features vivid pictures and fast dropdown lists and buttons. The strong financial service capacity of ABC Mobile Banking is well illustrated by comprehensive functions and personalized design.

After Internet Banking, Mobile Banking or M-Banking has become the buzz word in the industry. It's a fact that Internet Banking has given a boost and has shown a successful way to consider it as a good alternative procedure against physical branch banking. Now where ever you are, you can access your bank account and you can do lot more things like checking your account balance, transfer money to some other account, pay your utility bills online and so on, just by comfortably sitting at your home or office. But, the technical disadvantage of Internet Banking is, you have to have internet connectivity and a computer. Definitely it's not a big hindrance in US or Europe or in the other developed countries, but if one considers the developing economies, then it's a genuine problem and more specifically in the tier II cities. And here Mobile Banking comes into the picture to address the basic limitation of Internet Banking. If we only consider Asian developing countries, the availability of mobile connectivity is really huge. Where one may not find out a landline telephone or an internet connection, but still in those remote places getting mobile connectivity is not a major issue today.

So, Mobile Banking has given the traditional banking a newer look "Anywhere Banking". Now you don't need a PC or a laptop with internet connectivity, just you need your cell phone with you. Considering the Asian economy countries like China, India and Korea have seen the mobile boom in last one decade. A projected value of mobile connectivity in India shows that it will touch 180 Million subscribers by the end of 2008, where it was pegged at around 2 Million in the year 2000. In Korea, more than 70% of the entire population is carrying mobile devices.

The biggest advantage Mobile Banking provides to the banks is that it helps to cut down the costs as it's even more economic than providing tele-banking facilities where banks have to keep hundreds of tele-callers. Additionally, Mobile Banking helps banks to upgrade the quality of services and nature of customer relationship management. Using Mobile

Banking, banks can communicate to the defined cluster of clients. The offers can be customized and this personalization can give the banking industry a huge mileage, even at a lower cost. Again, using the same mobile channels, banks can up-sell and cross-sell their highly complex financial products to the specific set of customers which can be coupled with the selling strategies of Credit Cards, Home Loans and Personal Loans etc. On the contrary, the service providers can also accrue more business by providing the Mobile Banking services to their clients. Countries like Japan, Korea or Singapore where the mobile connectivity has already reached its saturation, the service providers can make handsome business by providing additional banking services to the same static client base.

In the services front, different banking services can be provided, depending upon the banking regulations in respective countries which may include Account Balance Enquiry, Account Statement Enquiry, Credit/Debit Alerts, Bill Payment Alerts, Cheque Book Requisition, Transaction History, Minimum Balance Alerts, Fund Transfer Facilities, etc.

Mobile Banking activities can be categorized in two different manners.

1. By the Nature of Service: It can be any of the two, either Enquiry Based or Transaction Based. For example, Account Balance Enquiry or a Cheque Book Requisition can be the good examples of Enquiry Based Services where a Fund Transfer or a Bill Payment is a Transaction Based activity.

2. Depending on the Originator: Again there can be two different types of services; Push and Pull, depending on the nature of the originator. A Push based service is from the Bank to the Client and vice versa. For example, Bill Payment Alert can be a Push based service, when getting Recent Account History is a Pull based one.

In different countries, Mobile Banking has already gained its popularity. For example, in the South Korean market LG Telecom teamed up with Kookmin Bank to provide their Mobile Banking services in 2004 and since then they have seen a nice and steady growth. In India, Reliance Infocomm has started providing Mobile banking services to ICICI Bank and HDFC Bank through their R-World environment.

The Mobile Banking services will become more popular once the availability of the smart phones or PDA phones shall increase as Smart Phones come with larger screens and bigger memory size. In the application development front, both J2ME and BREW have done excellent work and industry expects by the year 2012, more than 80% of the mobile handsets will be able to run stand alone Mobile Banking applications and that time it will be "Anywhere Banking" in real sense.

2.1 STUDY OF THE SYSTEM

To provide flexibility to the users, the interfaces have been developed that are accessible through a browser. The GUI'S at the top level have been categorized as

1. Administrative user interface
2. The operational or generic user interface

The 'administrative user interface' concentrates on the consistent information that is practically, part of the organizational activities and which needs proper authentication for the data collection. These interfaces help the administrators with all the transactional states like Data insertion, Data deletion and Date updation along with the extensive data search capabilities.

The 'operational or generic user interface' helps the end users of the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information in a customized manner as per the included flexibilities

2.2 INPUT & OUTPOUT REPRESENTETION

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

INPUT STAGES:

The main input stages can be listed as below:

- Data recording
- Data transcription
- Data conversion
- Data verification
- Data control
- Data transmission
- Data validation
- Data correction

INPUT TYPES:

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are computer department's communications to the system?
- Interactive, which are inputs entered during a dialogue.

INPUT MEDIA:

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

- Type of input
- Flexibility of format
- Speed
- Accuracy
- Verification methods
- Rejection rates
- Ease of correction
- Storage and handling requirements
- Security
- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

OUTPUT DESIGN:

In general are:

- External Outputs whose destination is outside the organization.
- Internal Outputs whose destination is with in organization and they are the User's main interface with the computer. Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation. The various types of outputs
- Operational outputs whose use is purely with in the computer department.
- Interface outputs, which involve the user in communicating directly with the system.

OUTPUT DEFINITION

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

For Example

- Will decimal points need to be inserted
- Should leading zeros be suppressed.

OUTPUT MEDIA:

In the next stage it is to be decided that which medium is the most appropriate for the output. The main considerations when deciding about the output media are:

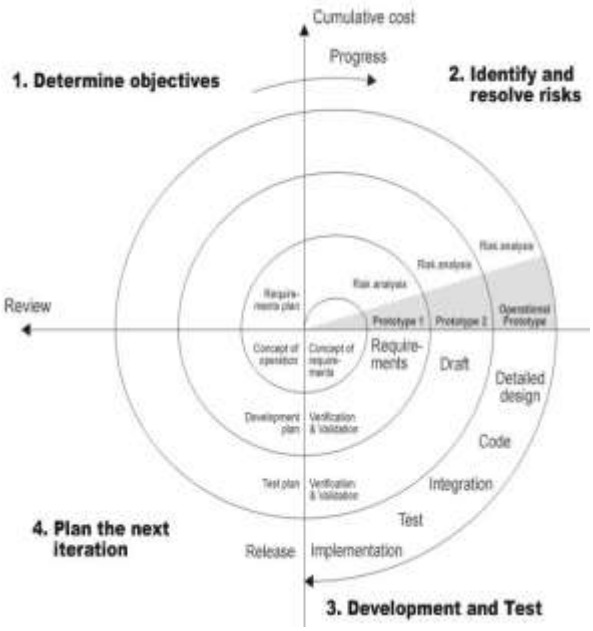
- The suitability for the device to the particular application.
- The need for a hard copy.
- The response time required.
- The location of the users
- The software and hardware available.

Keeping in view the above description the project is to have outputs mainly coming under the category of internal outputs. The main outputs desired according to the requirement specification are:

The outputs were needed to be generated as a hard copy and as well as queries to be viewed on the screen. Keeping in view these outputs, the format for the output is taken from the outputs, which are currently being obtained after manual processing. The standard printer is to be used as output media for hard copies.

2.3 PROCESS MODEL USED WITH JUSTIFICATION

SDLC (Spiral Model):



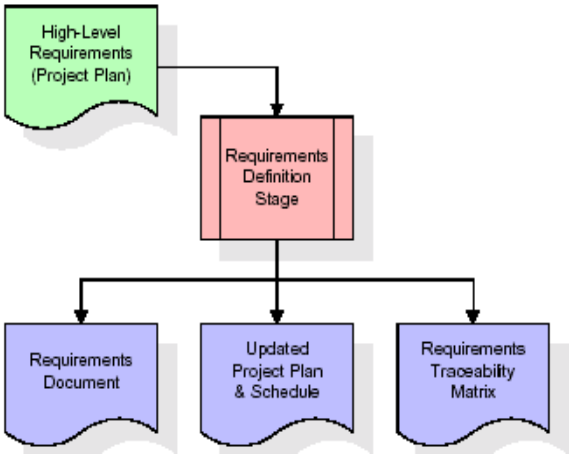
SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

Stages in SDLC:

- Requirement Gathering
- Analysis
- Designing
- Coding
- Testing
- Maintenance

Requirements Gathering stage:

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

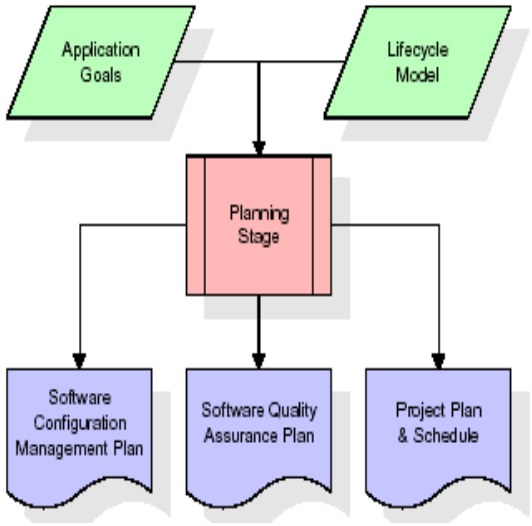
In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term *requirements traceability*.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

- Feasibility study is all about identification of problems in a project.
- No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
- Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator

Analysis Stage:

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.

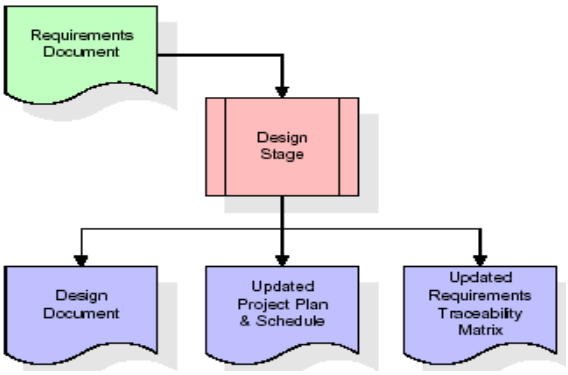


The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

Designing Stage:

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more

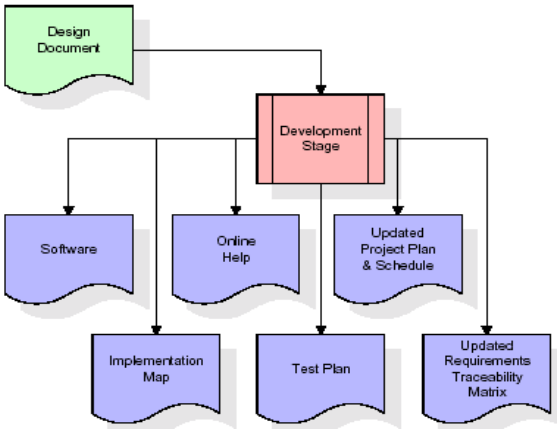
design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.



When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

Development (Coding) Stage:

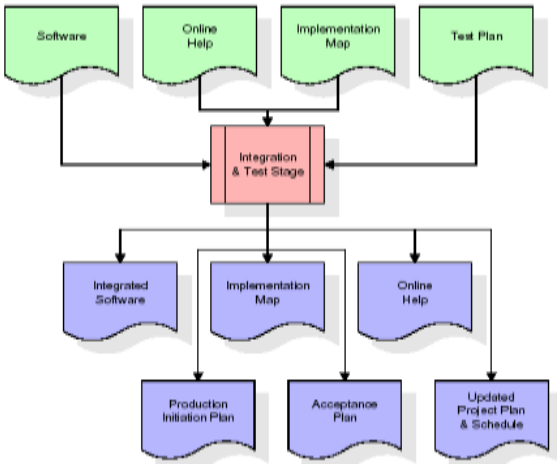
The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

Integration & Test Stage:

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



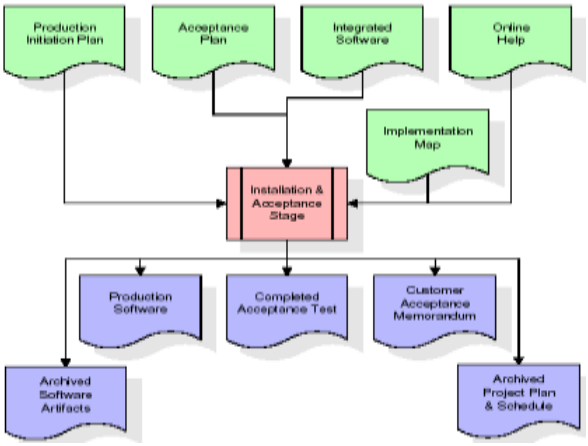
The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

- **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify

the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

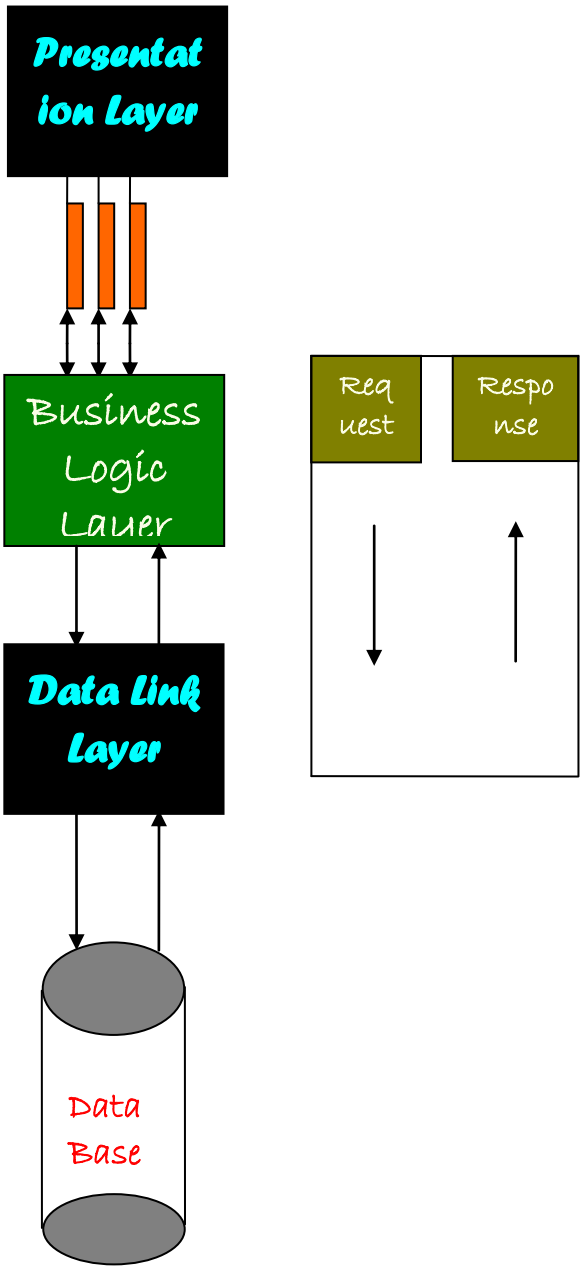
Maintenance:

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will under go training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

2.4 SYSTEM ARCHITECTURE

Architecture flow:

Below architecture diagram represents mainly flow of requests from users to database through servers. In this scenario overall system is designed in three tires separately using three layers called presentation layer, business logic layer and data link layer. This project was developed using 3-tire architecture.



Feasibility Study:

Preliminary investigation examines 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 systems are feasible if they are given unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operation Feasibility

Economical Feasibility

3.1 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?

3.2 OPERATIONAL FEASIBILITY

OPERATIONAL FEASIBILITY

User-friendly : Customer will use the forms for their various transactions i.e. for adding new routes, viewing the routes details. Also the Customer wants the reports to view the various transactions based on the constraints. Theses forms and reports are generated as user-friendly to the Client.

Reliability : The package wills pick-up current transactions on line. Regarding the old transactions, User will enter them in to the system.

Security : The web server and database server should be protected from hacking, virus etc

Portability : The application will be developed using standard open source software (Except Oracle) like Java, tomcat web server, Internet Explorer Browser etc these software will work both on Windows and Linux o/s. Hence portability problems will not arise.

Availability : This software will be available always.

Maintainability : The system called the ewheelz uses the 2-tier architecture. The 1st tier is the GUI, which is said to be front-end and the 2nd tier is the database, which uses My-Sql, which is the back-end.

The front-end can be run on different systems (clients). The database will be running at the server. Users access these forms by using the user-ids and the passwords.

3.3 ECONOMIC FEASILITY

The computerized system takes care of the present existing system's data flow and procedures completely and should generate all the reports of the manual system besides a host of other management reports. It should be built as a web based application with separate web server and database server. This is required as the activities are spread through out the organization customer wants a centralized database. Further some of the linked transactions take place in different locations.Open source software like TOMCAT, JAVA, Mysql and Linux is used to minimize the cost for the Customer.

4.1 FUNCTIONAL REQUIREMENTS SPECIFICATION

This application consists following modules.

1. **Check balance**
2. **Transfer funds**
3. **Stop payment**
4. **Request cheque-book**
5. **Mini statement**
6. **Change pin**

1. Check Balance module:

A customer is allowed to check the present balance in ones account. The customer must already have an account in the bank. Upon entering the account number one can obtain the relevant information.

2. Transfer Funds module:

A customer can transfer funds from one account to another by providing both *from* and *to* account numbers. The *from* account must belong to the customer. The system provides a feedback about the status of the operation.

3. Stop Payment module:

A customer can request to stop the payments for cheques issued to others. The customer must provide relevant details such as cheque number, and to whom the cheque is issued, date etc.

4. Request Cheque-Book module:

A customer may request for a new *cheque* book with the bank and the address to which that cheque book must be delivered.

5. Mini statement module:

This module allows a customer to generate a mini statement with the details of transactions done so far. The customer can use this mini statement to check if the transactions are done properly.

6. Change Pin module:

Each customer must be registered with the system and must have an operational account. The customer uses a *user-id* and *pin* value to perform online banking transactions. For security purposes, a customer is allowed to change the *pin* value as often as desired.

4.2 PERFORMANCE REQUIREMENTS

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely with the users of the existing system to give the requirement specifications because they are the people who finally use the

system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

- The system should be able to interface with the existing system
- The system should be accurate
- The system should be better than the existing system

The existing system is completely dependent on the user to perform all the duties.

4.3 SOFTWARE REQUIREMENTS:

Operating System : Windows
Technology : Java and J2EE
Web Technologies : Html, JavaScript, CSS
IDE : My Eclipse
Web Server : Tomcat
Database : MySql5.0
Java Version : J2SDK1.5

4.4 HARDWARE REQUIREMENTS:

Hardware : Pentium
RAM : 1GB

Additional Tools:

HTML Designing : Dream weaver Tool

Development Tool kit : My Eclipse

4.4.1. INTRODUCTION TO JAVA

About Java:

Initially the language was called as "oak" but it was renamed as "java" in 1995.The primary motivation of this language was the need for a platform-independent(i.e. architecture neutral)language that could be used to create software to be embedded in various consumer electronic devices.

- Java is a programmer's language
- Java is cohesive and consistent
- Except for those constraint imposed by the Internet environment. Java gives the programmer, full control
- Finally Java is to Internet Programming where c was to System Programming.

Importance of Java to the Internet

Java has had a profound effect on the Internet. This is because; java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the server and the personal computer. They are passive information and Dynamic active programs. in the areas of Security and probability. But Java addresses these concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

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 Applet I san application, designed to be transmitted over the Internet and executed by a Java-compatible web browser. An applet I 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 be react to the user input and dynamically change.

Java Architecture

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

Compilation of code

When you compile the code, the Java compiler creates machine code (called byte code)for a hypothetical machine called Java Virtual Machine(JVM). The JVM is supposed t executed the byte code. The JVM is created for the overcoming the issue of probability. The code is written and compiled for one machine and interpreted on all machines .This machine is called Java Virtual Machine.

Compiling and interpreting java source code.

During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be an Intel Pentium windows 95 or sun SPARCstation running Solaris or Apple Macintosh running system and all could receive code from any computer through internet and run the Applets.

Simple:

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ Programmer. Learning Java will oriented features of C++ . Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

Object oriented

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank state. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is

simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

Robust

The multi-platform environment of the web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs. Was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and runtime.

Java virtually eliminates the problems of memory management and deal location, which is completely automatic. In a well-written Java program, all run-time errors can and should be managed by your program.

4.4.2 **Servlets/JSP**

INTRODUCTION

A Servlet Is a generic server extension. a Java class that can be loaded

Dynamically to expand the functionality of a server.Servlets are commonly used with web servers. Where they can take the place CGI scripts.A servlet is similar to proprietary server extension, except that it runs inside a Java Virtual Machine (JVM) on the server, so it is safe and portable Servlets operate solely within the domain of the server. Unlike CGI and Fast CGI, which use multiple processes to handle separate program or separate requests, separate threads within web server process handle all servlets. This means that servlets are all efficient and scalable. Servlets are portable; both across operating systems and also across web servers. Java Servlets offer the best possible platform for web application development.Servlets are used as replacement for CGI scripts on a web server, they can extend any sort of server such as a mail server that allows servelts t extend its functionality perhaps by performing a virus scan on all attached documents or handling mail filtering tasks.Servlets provide a Java-based solution used to address the problems currently associated with doing server-side programming including inextensible scripting solutions platform-specific API's and incomplete interface. Servlets are objects that conform to a specific interface that can be plugged into a Java-based server.Servlets are to the server-side what applets are to the server-side what applets are to the client-side-object byte codes that can be dynamically loaded off the net. They differ form applets in than they are faceless objects(with out graphics or a GUI component).They serve as platform independent, dynamically loadable,plugable helper byte code objects on the server side that can be used to dynamically extend server-side functionality.

For example an HTTP servlet can be used to generate dynamic HTML content when you use servlets to do dynamic content you get the following advantages:

- They're faster and cleaner then CGI scripts
- They use a standard API(the servlet API)
- They provide all the advantages of Java (run on a variety of servers without needing to be rewritten)

A t t r a c t i v e n e s s o f S e r v l e t s:

They are many features of servlets that make them easy and attractive to tuse these include:

- Easily configure using the GUI-based Admin tool]
- Can be Loaded and Invoked from a local disk or remotely across the network.
- Can be linked together or chained, so that on servlet can call another servlet, or several servlets in sequence.
- Can be called dynamically from with in HTML, pages using server-side include-tags.
- Are secure-even when downloading across the network, the servlet security model and servlet and box protect your system from unfriendly behavior.,

Advantages of the servlet API

One of the great advantages of the servlet API is protocol independent. It assumes nothing about:

- The protocol being used to transmit on the net
- How it is loaded
- The server environment it will be running in
- These quantities are important, because it allows the Servlet API to be embedded in many different kinds of servers.There are other advantages to the servelt API as well These include:
- It's extensible-you can inherit all your functionality from the base classes made available to you
- It's simple small, and easy to use.

Features of Servlets:

- Servlets are persistent.Servlet are loaded only by the web server and can maintain services between requests.
- Servlets are fast. Since servlets only need to be l\loaded once, they offer much better performance over their CGI counterparts.
- Servlets are platform independent.
- Servlets are extensible Java is a robust, object-oriented programming language, which easily can be extended to suit your needs.
- Servlets are secure

- Servlets are used with a variety of client.

Servlets are classes and interfaces from tow packages,javax .servlet and javax.servlet.http.The java.servlet package contains classes t support generic, protocol-independent servlets.The classes in the javax.servelt.http package To and HTTP specific functionality extend these classes. Every servlet must implement the javax.servelt interface.Most servlets implement it by extending one of two classes.javax.servlet.GenericServlet or javax.servlet.http.HttpServlet.A protocol-independent servlet should subclass Generic-Servlet.while an Http servlet should subclass HttpServlet, which is itself a subclass of Generic-servlet with added HTTP-specific functionality.

Unlike a java program, a servlet does not have a main() method,Instead the server in the process of handling requests invoke certain methods of a servlet.Each time the server dispatches a request to a servlet, it invokes the servelts Service() method, A generic servlet should override its service() method to handle requests as appropriate for the servlet.The service() accepts two parameters a request object and a response object .The request object tells the servlet about the request, while the response object is used to return a response

InContrast.anHttp servlet usually does not override the service() method.Instead it overrides doGet() to handle GET requests and doPost() to handle Post requests. An Http servlet can override either or both of these modules the service() method of HttpServlet handles the setup and dispatching to all the doXXX() methods.which iswhy it usually should not be overridden

The remainders in the javax.servlet and javax.servlet.http.package are largely support classes .The ServletRequest and ServletResponse classes in javax.servlet provide access to generic server requests and responses while HttpServletRequest and HttpServletResponse classes in javax.servlet provide access to generic server requests and responses while HttpServletRequest and HttpServletResponse in javax.servlet.http provide access a HTTP requests and responses . The javax.servlet.http provide contains an HttpSession class that provides built-in session tracking functionality and Cookie class that allows quickly setup and processing HttpCookies.

Loading Servlets:

Servlets can be loaded from their places. From a directory that is on the CLASSPATH. The CLASSPATH of the JavaWebServer includes service root/classes/, which is where the system classes reside

From the <SERVICE_ROOT/servlets/directory.This is not in the server's classpath. A class loader is used to create servlets form this directory.New servlets can be added-existing servlets can be recompiled and the server will notice these changes. From a remote location.For this a code base like http://nine.eng/classes/foo/ is required in addtion to the servlet's class name.Refer to the admin Gui docs on servlet section to see how to set this up.

Loading Remote Servlets

Remote servlets can be loaded by:

- Configuring the admin Tool to setup automatic loading of remote servlets.
- Selectiong up server side include tags in .html files
- Defining a filter chain Configuration

Invoking Servlets

A servlet invoker is a servlet that invokes the "server" method on a named servlet.If the servlet is not loaded in the server,then the invoker first loades the servlet(either form local disk or from the network) and the then invokes the "service" method.Also like applets,local servlets in the server can be identified by just the class name.In other words, if a servlet name is not absolute.it is treated as local.

A Client can Invoke Servlets in the Following Ways:

- The client can ask for a document that is served by the servlet.
- The client(browser) can invoke the servlet directly using a URL, once it has been mapped using the SERVLET ALIASES Section of the admin GUI
- The servlet can be invoked through server side include tags.
- The servlet can be invoked by placing it in the servlets/directory
- The servlet can be invoked by using it in a filter chain

The Servlet Life Cycle:-

The Servlet life cycle is one of the most exciting features of Servlets.This life cycle is a powerful hybrid of the life cycles used in CGI programming and lower-level NSAPI and ISAPI programming.

The servlet life cycle allows servlet engines to address both the performance and resource problems of CGI and the security concents of low level server API programming.

Servlet life cycle is highly flexible Servers hava significant leeway in how they choose to support servlets.The only hard and fast rule is that a servlet engine must confor to the following life cycle contact:

- Create and initialize the servlets
- Handle zero or more service from clients
- Destroy the servlet and then garbage Collects it.

It's perfectly legal for a servlet t be loaded, created an initialzed in its own JVM,only to be

destroyed an dgarbage collected without hancdling any clientrequest or after handling just one request

The most common and most sensible life cycle implemntations for HTTP servelts are:

Single java virtual machine and astatine persistence.

Init and Destroy:-

Just like Applets servlets can define init() and destroy() methods, A servlets init(ServiceConfig) method is called by the server immediately after the server constructs the servlet's instance.Depanding on the server and its configuration, this can be at any of these times

- When the server states
- When the servlet is first requested, just before the service() method is invoked
- At the request of the server administrator

In any case, nit() is guaranteed to be called before the servlet handles its first request

The init() method is typically used to perform servlet initialization creating or loading objects that are used by the servlet in handling of its request. In order to providing a new servlet any information about itself and its environment, a server has to call a servelts init() method and pass an object that implement the ServletConfig interface.

This ServletConfig object supplies a servlet with information about its initialization parameters.These parameters are given to the servlets and are not associated with any single request.They can specify initial values, such as where a counter should begin counting, or default values, perhaps a template to use when not specified by the request,

The server calls a servlet's destroy() method when the servlet is about to be unloaded. In the destroy() method, a servlet should free any resources it has acquired that will not be garbage collected. The destroy() method also gives a servlet a chance to write out its unsaved. cached information or any persistent information that should be read during the next call to init().

Session Tracking:

HTTP is a stateless protocol, it provides no way for a server to recognize that a sequence of requests is all from the same client. This causes a problem for application such as shopping cart applications. Even in chat application server can't know exactly who's making a request of several clients.

The solution for this is for client to introduce itself as it makes each request, Each clients needs to provide a unique identifier that lets the server identify it, or it needs to give some information that the server can use to properly handle the request, There are several ways to send this introductory information with each request Such as:

USER AUTHORIZATION:

One way to perform session tracking is to leverage the information that comes with

User authorization. When a web server restricts access to some of its resources to only those clients that log in using a recognized username and password. After the client logs in, the username is available to a servlet through getRemoteUser() Wean use the username to track the session. Once a user has logged in, the browser remembers her username and resends the name and password as the user views new pages on the site. A servlet can identify the user through her username and they're by Track her session.

The biggest advantage of using user authorization to perform session tracking is that it's easy to implement. Simply tell the protect a set of pages, and use getRemoteUser() to identify each client.Another advantage is that the technique works even when the user accesses your site form or exists her browser before coming back.

The biggest disadvantage of user authrization is that it requires each user to register for an account and then log in in each time the starts visiting your site. Most users will tolerate registering and lagging in as a necessary evil when they are accessing sensitive information, but its all overkill for simple session tracking.Other problem with user authorization is that a user cannot simultaneously maintain more than one session at the same site.

Hidden Form Fields:

One way to support anonymous session tracking is to use hidden from fields. As the name implies, these are fields added to an HTML, form that are not displayed in the client's browser, They are sent back to the server when the form that contains them is submitted.

In a sense, hidden form fields define constant variables for a form. To a servlet receiving a submitted form, there is no difference between a hidden fields and a visible filed. As more and more information is associated with a clients session . It can become burdensome to pass it all using hidden form fields. In these situations it's possible to pass on just a unique session ID that identifies as particular clients session. That session ID can be associated with complete information about its session that is stored on the server. The advantage of hidden form fields is their ubiquity and support for anonymity. Hidden fields are supported in all the popular browsers, they demand on special server requirements, and they can be used with clients that haven't registered or logged in. The major disadvantage with this technique, however is that works only for a sequence of dynamically generated forms, The technique breaks down immediately with

static documents, emailed documents book marked documents and browser shutdowns.

URL Rewriting:

URL rewriting is another way to support anonymous session tracking, With URL rewriting every local URL the user might click on is dynamically modified. or rewritten, to include extra, information. The extra information can be in the deform of extra path information, added parameters, or some custom, server-specific.URL change. Due to the limited space available in rewriting a URL, the extra information is usually limited to a unique session. Each rewriting technique has its own advantage and disadvantage Using extra path information works on all servers, and it works as a target for forms that use both the Get and Post methods. It does not work well if the servlet has to use the extra path information as true path information

The advantages and disadvantages of URL.rewriting closely match those of hidden form fileds,The major difference is that URL rewriting works for all dynamically created documents, such as the Help servlet, not just forms. With the right server support, custom URL rewriting can even work for static documents.

Persistent Cookies:

A fourth technique to perform session tracking involves persistent cookies. A cookie is a bit of information. sent by a web server to a browser that can later be read back form that browser. When a browser receives a cookie, it saves the cookie and there after sends the cookie back to the server each time it accesses a page on that server, subject to certain rules. Because a cookie's value can uniquely identify a client, cookies are often used for session tracking.

Persistent cookies offer an elegant, efficient easy way to implement session tracking. Cookies provide as automatic an introduction for each request as we could hope for. For each request, a cookie can automatically provide a client's session ID or perhaps a list of clients performance. The ability to customize cookies gives them extra power and versatility.

The biggest problem with cookies is that browsers don't always accept cookies sometimes this is because the browser doesn't support cookies. More often its because

The browser doesn't support cookies. More often its because the user has specifically configured the browser to refuse cookies.

The power of serves:

The power of servlets is nothing but the advantages of servlets over other approaches, which include portability, power, efficiency, endurance, safety elegance,integration,extensibility and flexibility.

Portability:

As servlets are written in java and conform to a well defined and widely accepted API.they are highly portable across operating systems and across server implementation

We can develop a servlet on a Windows NT machine running the java web server and later deploy it effortlessly on a high-end UNIX server running apache. With servlets we can really "write once, serve every where"

Servlet portability is not the stumbling block it so often is with applets, for two reasons

First,Servlet portability is not mandatory i.e. servlets has to work only on server machines that we are using for development and deployment

Second, servlets avoid the most error-prone and inconstancy implemented portion of the java languages.

Power:

Servlets can harness the full power of the core java. API's: such as Networking and Url access, multithreading, image manipulation, data compression, data base connectivity, internationalization, remote method invocation(RMI) CORBA connectivity, and object serialization, among others,

Efficiency and Endurance:

Servlet invocation is highly efficient, Once a servlet is loaded it generally remains in the server's memory as a single object instance, There after the server invokes the servelt to handle a request using a simple, light weighted method invocation .Unlike the CGI, there's no process to spawn or interpreter to invoke, so the servlet can begin handling the request almost immediately, Multiple, concurrent requests are handled the request almost immediately. Multiple, concurrent requests are handled by separate threads, so servlets are highly scalable.

Servlets in general are enduring objects. Because a servlets stays in the server's memory as a single object instance. it automatically maintains its state and can hold onto external resources, such as database connections.

Safety:

Servlets support safe programming practices on a number of levels.

As they are written in java,servlets inherit the strong type safety of the java language. In addition the servlet API is implemented to be type safe. Java's automatic garbage collection and lack of pointers mean that servlets are generally safe from memory management problems like dangling pointers invalid pointer references and memory leaks.

Servlets can handle errors safely, due to java's exception – handling mechanism. If a servlet divides by zero or performs some illegal operations, it throws an exception that can be safely caught and handled by the server.

A server can further protect itself from servlets through the use of java security manager.A server can execute its servlets under the watch of a strict security manager.

Elegance:

The elegance of the servlet code is striking .Servlet code is clean, object oriented modular and amazingly simple one reason for this simplicity is the served API itself. Which includes methods and classes to handle many of the routine chores of servlet development. Even advanced to operations like cookie handling and session tracking tracking are abstracted int convenient classes.

Integration:

Servlets are tightly integrated with the server. This integration allows a servlet to cooperate with the server in two ways . for e.g.: a servlet can use the server to translate file paths, perform logging, check authorization, perform MIME type mapping and in some cases even add users to the server's user database.

Extensibility and Flexibility:

The servlet API is designed to be easily extensible. As it stands today the API includes classes that are optimized for HTTP servlets.But later it can be extended and optimized for another type of servlets.It is also possible that its support for HTTP servlets could be further enhanced. Servlets are also quite flexible, Sun also introduced java server pages. which offer a way to write snippets of servlet code directly with in a static HTML page using syntax similar to Microsoft's Active server pages(ASP)

4.4.3 **JDBC**

What is JDBC?

any relational database. One can write a single program using the JDBC API,and the JDBC is a Java Api for executing SQL,Statements(As a point of interest JDBC is trademarked name and is not an acronym; nevertheless,Jdbc is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java Programming language.JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API.

Using JDBC, it is easy to send SQL statements to virtually program will be able to send SQL .statements to the appropriate database. The Combination of Java and JDBC lets a programmer writes it once and run it anywhere.

What Does JDBC Do?

Simply put,JDBC makes it possible to do three things

Establish a connection with a database
Send SQL statements
Process the results

JDBC Driver Types

The JDBC drivers that we are aware of this time fit into one of four categories

JDBC-ODBC Bridge plus ODBC driver
Native-API party-java driver
JDBC-Net pure java driver
Native-protocol pure Java driver

An individual database system is accessed via a specific JDBC driver that implements the java.sql.Driver interface. Drivers exist for nearly all-popular RDBMS systems, through few are available for free. Sun bundles a free JDBC-ODBC bridge driver with the JDK to allow access to a standard ODBC,data sources, such as a Microsoft Access database, Sun advises against using the bridge driver for anything other than development and very limited development.

JDBC drivers are available for most database platforms, from a number of vendors and in a number of different flavours. There are four driver categories

Type 01-JDBC-ODBC Bridge Driver

Type 01 drivers use a bridge technology to connect a java client to an ODBC database service. Sun's JDBC-ODBC bridge is the most common type 01 driver. These drivers implemented using native code.

Type 02-Native-API party-java Driver

Type 02 drivers wrap a thin layer of java around database-specific native code libraries for Oracle databases, the native code libraries might be based on the OCI(Oracle call Interface) libraries, which were originally designed for **c/c++** programmers, Because type-02 drivers are implemented using native code. in some cases they have better performance than their all-java counter parts. They add an element of risk, however, because a defect in a driver's native code section can crash the entire server

Type 03-Net-Protocol All-Java Driver

Type 03 drivers communicate via a generic network protocol to a piece of custom middleware.

The middleware component might use any type of driver to provide the actual database access. These drivers are all java, which makes them useful for applet deployment and safe for servlet deployment

Type-04-native-protocol All-java Driver

Type o4 drivers are the most direct of the lot. Written entirely in java, Type 04 drivers understand database-specific networking. protocols and can access the database directly without any additional software

JDBC-ODBC Bridge

If possible use a Pure Java JDBC driver instead of the Bridge and an ODBC driver. This completely eliminates the client configuration required by ODBC.It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge(that is, the Bridge native library, the ODBC driver manager library, library, the ODBC driver library, and the database client library)

WHAT IS The JDBC-ODBE Bridge ?

The JDBC-ODBC Bridge is a Jdbc driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge is implemented as the sun.jdbc.odbc Java package and contains a native library used to access ODBC.The Bridge is joint development of Intersolv and Java Soft.

4.4.4 **Oracle**

Oracle is a relational database management system, which organizes data in the form of tables. Oracle is one of many database servers based on RDBMS model, which manages a seer of data that attends three specific things-data structures, data integrity and data manipulation.

With oracle cooperative server technology we can realize the benefits of open, relational systems for all the applications. Oracle makes efficient use of all systems resources, on all hardware architecture; to deliver unmatched performance, price performance and scalability. Any DBMS to be called as RDBMS has to satisfy Dr.E.F.Codd's rules.

Features of Oracle:

Portable

The Oracle RDBMS is available on wide range of platforms ranging from PCs to super computers and as a multi user loadable module for Novel NetWare, if you develop application on system you can run the same application on other systems without any modifications.

Compatible

Oracle commands can be used for communicating with IBM DB2 mainframe RDBMS that is different from Oracle, which is Oracle compatible with DB2. Oracle RDBMS is a high performance fault tolerant DBMS, which is specially designed for online transaction processing and for handling large database applications.

Multithreaded Server Architecture

Oracle adaptable multithreaded server architecture delivers scalable high performance for very large number of users on all hardware architecture including symmetric multiprocessors (sumps) and loosely coupled multiprocessors. Performance is achieved by eliminating CPU, I/O, memory and operating system bottlenecks and by optimizing the Oracle DBMS server code to eliminate all internal bottlenecks.

Oracle has become the most popular RDBMS in the market because of its ease of use

- Client/server architecture.
- Data independence.
- Ensuring data integrity and data security.
- Managing data concurrency.
- Parallel processing support for speed up data entry and online transaction processing used for applications.
- DB procedures, functions and packages.

Dr.E.F.Codd's Rules

These rules are used for valuating a product to be called as relational database management systems. Out of 12 rules, a RDBMS product should satisfy at least 8 rules + rule called rule 0 that must be satisfied.

RULE 0: Foundation Rule

For any system to be advertised as, or claimed to be relational DBMS should manage database with in it self, with out using an external language.

RULE 1: Information Rule

All information in relational database is represented at logical level in only one way as values in tables.

RULE 2: Guaranteed Access

Each and every data in a relational database is guaranteed to be logically accessibility by using to a combination of table name, primary key value and column name.

RULE 3: Systematic Treatment of Null Values

Null values are supported for representing missing information and inapplicable information. They must be handled in systematic way, independent of data types.

RULE 4: Dynamic Online Catalog based Relation Model

The database description is represented at the logical level in the same way as ordinary data so that authorized users can apply the same relational language to its interrogation as they do to the regular data.

RULE 5: Comprehensive Data Sub Language

A relational system may support several languages and various models of terminal use. However there must be one language whose statement can express all of the following:

Data Definitions, View Definitions, Data Manipulations, Integrity, Constraints, Authorization and transaction boundaries.

RULE 6: View Updating

Any view that is theoretical can be updatable if changes can be made to the tables that effect the desired changes in the view.

RULE 7: High level Update, Insert and Delete

The capability of handling a base relational or derived relational as a single operand applies not only retrieval of data also to its insertion, updating, and deletion.

RULE 8: Physical Data Independence

Application program and terminal activities remain logically unimpaired whenever any changes are made in either storage representation or access method.

RULE 9: Logical Data Independence

Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representation or access methods.

RULE 10: Integrity Independence

Integrity constraints specific to particular database must be definable in the relational data stored in the catalog, not in application program.

RULE 11: Distributed Independence

Whether or not a system supports database distribution, it must have a data sub-language that can support distributed databases without changing the application program.

RULE 12: Non Sub-Version

If a relational system has low level language, that low language cannot use to subversion or by pass the integrity rules and constraints expressed in the higher level relational language.

Oracle supports the following Codd's Rules

Rule 1: Information Rule (Representation of information)-YES.

Rule 2: Guaranteed Access-YES.

Rule 3: Systematic treatment of Null values-YES.

Rule 4: Dynamic on-line catalog-based Relational Model-YES.

Rule 5: Comprehensive data sub language-YES.

Rule 6: View Updating-PARTIAL.

Rule 7: High-level Update, Insert and Delete-YES.

Rule 8: Physical data Independence-PARTIAL.

Rule 9: Logical data Independence-PARTIAL.

Rule 10: Integrity Independence-PARTIAL.

Rule 11: Distributed Independence-YES.

Rule 12: Non-subversion-YES.

4.4.5 HTML

Hypertext Markup Language(HTML), the languages of the world wide web(WWW), allows users to produces web pages that included text, graphics and pointer to other web pages (Hyperlinks). HTML is not a programming language but it is an application of ISO Standard 8879,SGML(Standard Generalized Markup Language),but Specialized to hypertext and adapted to the Web. The idea behind Hypertext one point to another point. We can navigate through the information based on out interest and preference. A markup language is simply a series of items enclosed within the elements should be displayed. Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same document. Html can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop

HTML provides tags(special codes) to make the document look attractive.

HTML provides are not case-sensitive. Using graphics,fonts,different sizes, color, etc.. can enhance the presentation of the document. Anything That is not a tag is part of the document it self.

Basic Html Tags:

<!-- -->	Specific Comments.
<A>……</A>	Creates Hypertext links.
<B>……</B>	Creates hypertext links.
<Big>……..</Big>	Formats text in large-font
<Body>…….</Body>	contains all tags and text in the Html-document
<Center>……</Center>	Creates Text
<DD>……..</DD>	Definition of a term.
<TABLE>……</TABLE>	creates table
<Td>……..</Td>	indicates table data in a table.
<Tr>……..</Tr>	designates a table row
<Th>…….</Th>	creates a heading in a table.

A D V A N T A G E S:-

A HTML document is small and hence easy to send over the net.It is small because it does not include formatted information.

HTML is platform independent

HTML tags are not case-sensitive.

4.4.6 JAVA SCRIPT

The Java Script Language

JavaScript is a compact , object-based scripting language for developing client and server internet

applications. Netscape Navigator 2.0 interprets JavaScript statements embedded directly in an HTML page. and Livewire enables you to create server-based applications similar to common gateway interface(cgi) programs. In a client application for Navigator, JavaScript statements embedded in an HTML Page can recognize and respond to user events such as mouse clicks for Input, and page navigation. For example, you can write a JavaScript function to verify that users enter valid information into a form requesting a telephone number or zip code . Without any network transmission, an Html page with embedded Java Script can interpret the entered text and alert the user with a message dialog if the input is invalid or you can use JavaScript to perform an action (such as play an audio file, execute an applet, or communicate with a plug-in) in response to the user opening or exiting a page.

5.1 Systems design

Introduction: Systems design is the process or art 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 and synergy with the disciplines of systems analysis, systems architecture and systems engineering.

5.2 DATA FLOW DIAGRAMS

Data Flow Diagrams

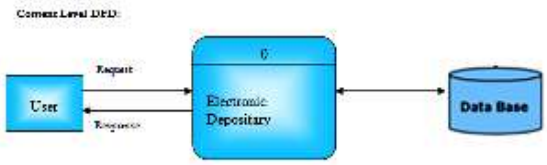
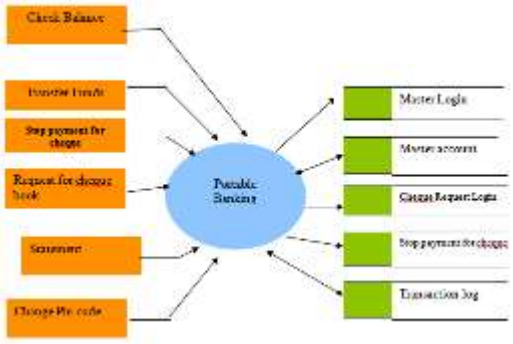


Figure: Level-0 DFD



Unified Modeling Language:

The Unified Modeling Language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

- User Model View
  i. This view represents the system from the users perspective.
  ii. The analysis representation describes a usage scenario from the end-users perspective.
- Structural model view
  i. In this model the data and functionality are arrived from inside the system.
  ii. This model view models the static structures.
- Behavioral Model View

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

- Implementation Model View

In this the structural and behavioral as parts of the system are represented as they are to be built.

- Environmental Model View

In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

UML is specifically constructed through two different domains they are:

✓ UML Analysis modeling, this focuses on the user model and structural model views of the system.

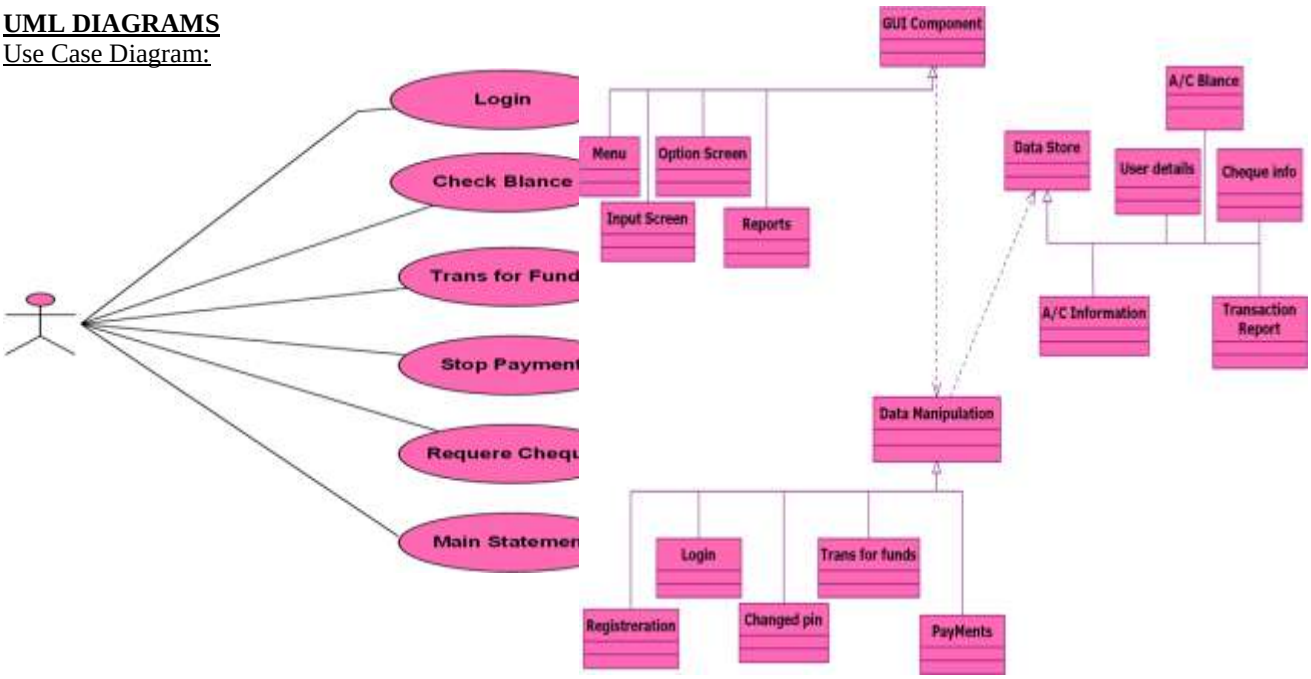
✓ UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

Use case Diagrams represent the functionality of the system from a user's point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view.

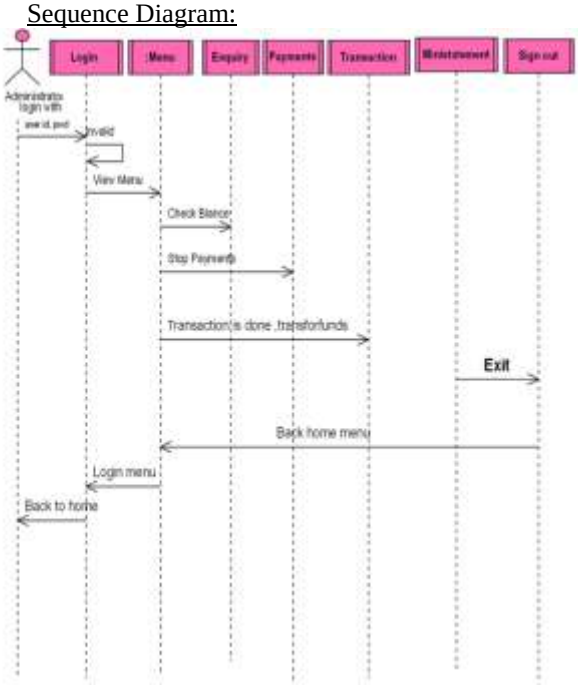
Actors are external entities that interact with the system. Examples of actors include users like administrator, bank customer …etc., or another system like central database.

UML DIAGRAMS

Use Case Diagram:

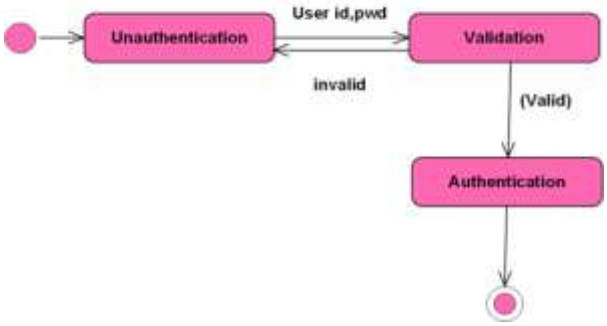


Sequence Diagram:

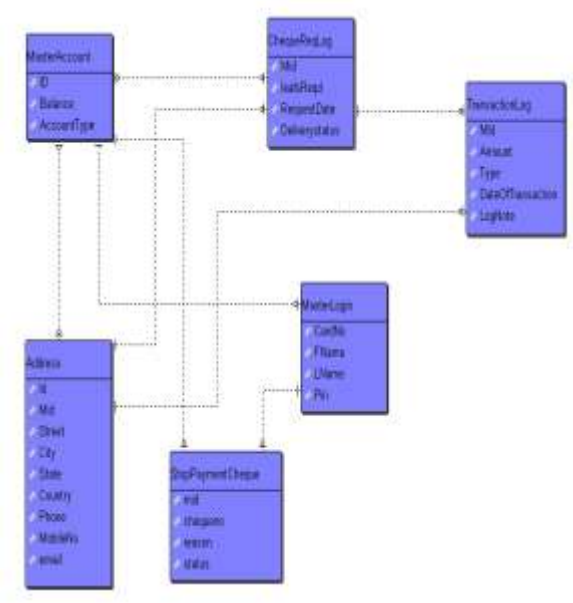


Class Diagram:

State Diagram for User:



5.4 ER-Modeling:



5.5. NORMALIZATION

A Database is a collection of interrelated data stored with a minimum of redundancy to serve many applications. The database design is used to group data into a number of tables and minimizes the artificiality embedded in using separate files. The tables are organized to:

- Reduced duplication of data.
- Simplify functions like adding, deleting, modifying data etc..,
- Retrieving data
- Clarity and ease of use
- More information at low cost

Normalization

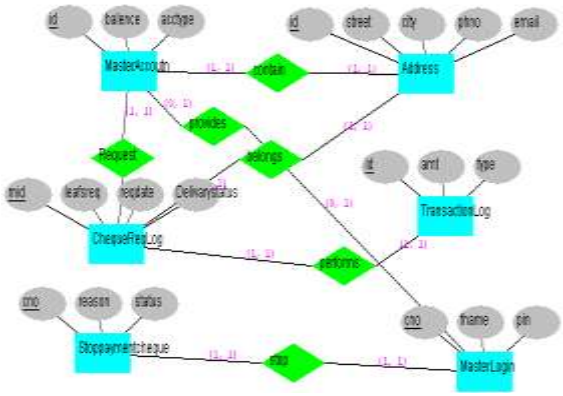
Normalization is built around the concept of normal forms. A relation is said to be in a particular normal form if it satisfies a certain specified set of constraints on the kind of functional dependencies that could be associated with the relation. The normal forms are used to ensure that various types of anomalies and inconsistencies are not introduced into the database.

First Normal Form: A relation R is in first normal form if and only if all underlying domains contained atomic values only.

Second Normal Form: A relation R is said to be in second normal form if and only if it is in first normal form and every non-key attribute is fully dependent on the primary key.

Third Normal Form: A relation R is said to be in third normal form if and only if it is in second normal form and every non key attribute is non transitively depend on the primary key.

5.6. DATA DICTI0NARY



Test cases Log in

Check Balance

Request Cheque Book

Statement

Stop Payment

Transfer Funds

Cheque Status

Change Pin

Log Out

7.1 TESTING
Introduction to Testing:

Testing is a process, which reveals errors in the program. It is the major quality measure employed during software development. During software development. During testing, the program is executed with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as it is expected to perform.

7.2 TESTING IN STRATEGIES

In order to make sure that the system does not have errors, the different levels of testing strategies that are applied at differing phases of software development are:

Unit Testing:
Unit Testing is done on individual modules as they are completed and become executable. It is confined only to the designer's requirements.

Each module can be tested using the following two Strategies: Black Box Testing:

In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program. This testing has been uses to find errors in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structure or external database access
- Performance errors
- Initialization and termination errors.

In this testing only the output is checked for correctness.
The logical flow of the data is not checked.

- **White Box testing:** In this the test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases. It has been uses to generate the test cases in the following cases:
- Guarantee that all independent paths have been Executed.
- Execute all logical decisions on their true and false Sides.
- Execute all loops at their boundaries and within their operational bounds
- Execute internal data structures to ensure their validity.

Integrating Testing :
Integration testing ensures that software and subsystems work together a whole. It tests the interface of all the modules to make sure that the modules behave properly when integrated together.

System Testing :
Involves in-house testing of the entire system before delivery to the user. It's aim is to satisfy the user the system meets all requirements of the client's specifications.

Acceptance Testing :
It is a pre-delivery testing in which entire system is tested at client's site on real world data to find errors.

Test Approach :
Testing can be done in two ways:

- Bottom up approach
- Top down approach

Bottom up Approach:
Testing can be performed starting from smallest and lowest level modules and proceeding one at a time. For each module in bottom up testing a short program executes the module and provides the needed data so that the module is asked to perform the way it will when embedded with in the larger system. When bottom level modules are tested attention turns to those on the next level that use the lower level ones they are tested individually and

then linked with the previously examined lower level modules.

Top down approach:

This type of testing starts from upper level modules. Since the detailed activities usually performed in the lower level routines are not provided stubs are written. A stub is a module shell called by upper level module and that when reached properly will return a message to the calling module indicating that proper interaction occurred. No attempt is made to verify the correctness of the lower level module.

Validation:

The system has been tested and implemented successfully and thus ensured that all the requirements as listed in the software requirements specification are completely fulfilled. In case of erroneous input corresponding error messages are displayed

8.1 INTRODUCTION

System Security:

Setting Up Authentication for Web Applications

Introduction:

To configure authentication for a Web Application, use the <login-config> element of the web.xml deployment descriptor. In this element you define the security realm containing the user credentials, the method of authentication, and the location of resources for authentication.

8.2 SECURITY IN SOFTWARE

To set up authentication for Web Applications:

1. Open the web.xml deployment descriptor in a text editor or use the Administration Console. Specify the authentication method using the <auth-method> element. The available options are:

BASIC

Basic authentication uses the Web Browser to display a username/password dialog box. This username and password is authenticated against the realm.

FORM

Form-based authentication requires that you return an HTML form containing the username and password. The fields returned from the form elements must be: j_username and j_password, and the action attribute must be j_security_check. Here is an example of the HTML coding for using FORM authentication:

```
<form method="POST" action="j_security_check">
        <input type="text" name="j_username">
        <input type="password" name="j_password">
        </form>
```

The resource used to generate the HTML form may be an HTML page, a JSP, or a servlet. You define this resource with the <form-login-page> element.

The HTTP session object is created when the login page is served. Therefore, the session.isNew() method returns FALSE when called from pages served after successful authentication.

9. METHODOLOGY

The study is aimed to evaluate perceptions and opinions of urban mobile banking users. For this a cross sectional descriptive design was adopted with ad-hoc quota sampling.

Sample was comprised of 50 mobile banking users and 50 non-users of Ghaziabad city, India. Non-users were defined as individuals having bank account but not using mobile banking. Of the total respondents 68.16 % were male and 31.84% were female. The sample was comprised of relatively young respondents. Of the total respondents students were 68.18%; remaining were working. 24.4% respondents were graduates and 75.6% were postgraduates [24].

Data was obtained by using structured questionnaire. Data was screened for missing values (available case method was adopted to handle missing values) and outliers. Data was further subject to normality- data was found to be normally distributed as skew index ranged from -.29 to .46 (reference absolute value 3) and kurtosis index from -.1.91 to 2.05 (reference absolute value 10). This questionnaire was analyzed for scale reliability analysis which suggests that items makeup the scale measured the same underlying constructs, as cronbach's alpha coefficient was found to be 0.764 (Annexure 1). At last convergent validity was confirmed as significant correlation (moderate to large, sig .05) was present between items measuring single construct.

Annexure 1

R E L I A B I L I T Y A N A L Y S I S – S C A L E (A L P H A)

Reliability Coefficients N of Class = 100.0 Alpha= .764

10. ANALYSIS AND DISCUSSION

Data was subject to Correlation analysis, Independent Samples T-test, ANNOVA, Percentile analysis.

10. 1 Mobile banking users: Demographic profile

Two-tailed Pearson Correlation was conducted to evaluate the relationship between mobile banking users and demographic variables viz. age, sex, education, occupation and income.

Only demographic variable had significant correlation with user was sex ($r=0.293$, $N=100$, $p<0.05$, correlation strength moderate). Thus analysis suggests that males are more inclined to use mobile banking in comparison to females (Table 2).

Table 2: Results of Correlation Analysis

Demographic variable	Correlation coefficient (N=100, p<0.05)
Age	0.067
Sex	0.293*
Education	-0.071
Occupation	-0.076
Income	0.031

10.2 Mobile banking users: Service usages pattern

Data was further subject to Percentile Analysis which suggested that among mobile banking users majority, 87.87 were availing mobile banking services from public/govt. banks. State Bank of India was largest mobile banking service provider overall followed by Punjab National Bank.

Among private banks ICICI Bank was leading in mobile banking services (Table 3). Percentile Analysis also suggested that majority of users were using services from less than 2 years and frequency of uses among majority of users was weekly (Table 4).

Table 3: Mobile Banking Service usages pattern (a)

Bank from which availing services	% Users
Public Banks 1. SBI 2. PNB	87.87
Private Banks 1. ICICI 2. HDFC	6.06
Both Public & Private Banks	6.06

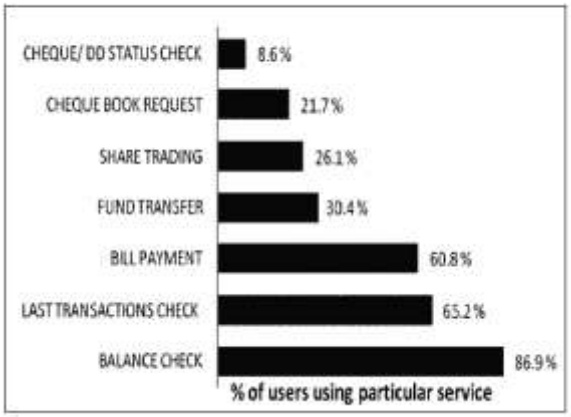
Table 3: Mobile Banking Service usages pattern (b)

Mobile Banking Users			
Time Period	% of users	Frequency of use	% of users
Less than 6 months	39.39	Daily	24.24
6 months to 2 years	42.42	Weekly	45.45
More than 2 years	18.18	Monthly	30.3

10.3 Mobile banking users: Services preference

Data related to various mobile banking services in users were interested was also subject to Percentile Analysis to explore currently most preferred services in mobile banking. Checking balance through mobile banking was most used service where as checking status of D.D. or cheque was least used. (Chart 1)

Chart 1: Mobile banking services preference



10.4 Mobile banking issues: Overall analysis

Overall Percentile Analysis was done for the responses of mobile banking users and non-users to explore selected issues in mobile banking which may impose challenges to service providers (banks), RBI, telecom operators etc. and thus may influence success of mobile banking in India

(Table 4).

Table 4: Analysis of Mobile banking Issues

Table 4: Analysis of Mobile banking Issues

Level of agreement of the respondents (in % terms) with issues					
Mobile banking issues	Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
Mobile handset operability	34.09%	47.27%	10.06%	6.81%	0.75%
Security / Privacy	27.77%	37.06%	20.10%	9.75%	5.30%
Standardization	15.90%	43.18%	24.24%	12.12%	4.54%
Downloading & Installing application software	10.63%	10.30%	33.33%	42.70%	3.02%
Customization	7.57%	26.06%	52.72%	12.12%	1.51%
Telecom Services Quality	10.09%	9.06%	43.93%	27.27%	9.60%

Analysis suggested that majority of respondents i.e. 81.36% strongly agree or agree that mobile handset operability is an issue in mobile banking as different types of handsets support different types of technology leading to complexity. 64.83% respondents think that mobile banking is not secure, also data privacy is absent.

On standardization again majority 59.08 % respondents strongly agree or agree that mobile banking service standards are lacking among Indian banks which makes it difficult to do mobile banking from multiple service providers.

On the issues of download & installation of application s/w, customization (user's preferred language) and telecom service quality; majority of the respondents were either indifferent or disagree. Reason may be that study was conducted in urban area so technological aspect of application s/w, absence of local/preferred language and telecom service quality like network unavailability were not perceived as major issues.

10.5 Mobile banking issues: Users vs. non-users

One way ANOVA was applied to explore any significant difference in opinions of mobile banking users and nonusers on selected issues in mobile banking. Results showed that there was no statistically significant difference in opinions of users and non-users for the selected issues (Annexure 2).

Annexure 2

Annexure 2

Independent Samples Test

		Levene's Test		t-test for Equality of Means	
		F	Sig.	t	Sig. (2-tailed)
PER. UTL.	Equal variances assumed	3.043	.086	.000	1.000
	Equal variances not assumed			.000	1.000

Table 5: Mobile banking users vs. non-users

Table 5: Mobile banking users vs. non-users

Issues	F	Sig. (p<0.05)
Mobile handset operability	0.789	0.378
Security/Privacy	3.313	0.073
Standardization	0.224	0.637
Downloading & Installing application software	0.015	0.902
Customization	0.020	0.888
Telecom Service Quality	0.753	0.389

8.6 Mobile banking perceived utility in comparison

To other channels:

Overall users vs. non-users

Percentile analysis (Annexure 3) suggest that majority of the respondents (53.6%) were indifferent towards perceived utility of mobile banking in comparison to retail banking and internet banking.

Further an Independent Samples T-test was conducted to explore perceived utility of mobile banking in comparison to retail banking and internet banking students of users and nonusers.

There was absolute no significant difference in perceived utility between users (M=2.55, SD=0.83) and nonusers (M=2.55, SD=1.03; t= 0.000, p= 1.000).

Annexure 3

Annexure 3

PER. UTL.

		Output	Percent
Valid	100	1	10.6
Missing	0	2	12.4
		3	53.6
		4	15.2
		5	8.2
		Total	100.0

11. RECOMMENDATIONS

Study shows 'mobile handset operability' is an important
issue in mobile banking, due to availability of various handset models (supporting different type of technology) in
the market. To resolve it service providers i.e. banks must
coordinate with mobile handset manufacturers so that all
handsets irrespective of manufacturer and technology (GSM or CDMA) become compatible with single mobile
banking technology.
Majority customers perceived 'privacy and security' a
critical issue. Here banks are advised to educate customers
on this issue to raise their awareness. Especially for the
customers' worries like losing money if once mobile handset
is lost (substantial number of respondents worried about
it). Secondly banks and telecom operators are suggested to
draft comprehensive joint policy regarding security & privacy so that customers can be assured at both banks and
telecom operator's levels while doing mobile banking.
'Standardization' is another major issue as lack of
standardization of mobile banking services in the country
resulted in increased complexity while using mobile banking
services (especially when using mobile banking services of
multiple banks). For resolving this issue banks are advised to
developed mobile banking standards in guidance of RBI.
Issues of 'download & installation of application s/w',
'customization' (user's preferred language) and 'telecom
service quality' were not perceived critical or important.
Reason may be that study was conducted in urban area so
technological aspect of application s/w, absence of
local/preferred language and telecom service quality like
network unavailability were not perceived as major issues.
But banks are well advised not to overlook above issues as
these may be critical in pan India adoption of mobile banking.

12. CONCLUSION

The Mobile Payment Forum of India (MPFI) has been
formed with Institute for Development and Research in
Banking Technology (IDRBT) and Rural Technology Business Incubator (RTBI), IIT Madras taking the lead
role. It has member s and representatives from the
telecommunications industry, financial institutions (banks and
microfinance institutions) as well members from the Reserve Bank of India. Three sub- committees have been
formed – on technology, on business models and on
regulatory issues. The first meeting of MPFI was held in
Hyderabad on the 15th of September 2007. The subcommittees
are expected to go over their particular concerns
in depth and submit a report shortly. Lots of challenges are
to be overcome for a successful implementation of
mobile payments to be widely accepted as a mode of payment. Businesses, merchants and consumers have to
come forward and make value-producing investments. A
regulatory framework and widely accepted standards will
be the pillars on which mobile payment applications will be
built. Research so far has outlined a diversity of thinking and
innovation that exists in the m- payments ar ena.
Numerous solutions have been tried and failed but the
future is promising with potential new technology innovations.

BIBLIOGRAPHY

References for the Project Development Were Taken From the following Books and Web Sites.

1. Al-Ashban, A.A. an d Burney, M.A. Customer adoption of tele-banking technology: the case of Saudi Arabia, *International Journal of Bank Marketing*,19 (5), pp. 191-200,2001.
2. Amir Herzberg. Payments And Banking With Mobile Personal Devices, Communications of the ACM, Vol. 46, No. 5, May 2003.
3. Banzal S. Mobile Banking & M–Commerce and Related issues, *www.* public.webfoundation.org/…..... 25, Mobile_banking_M-commerce_15.03.pdf, 2010.

4. Barnes,S.J., and Corbitt, B. Mobile Banking: Concept and Potential, International Journal of Mobile Communications, 1 (3), pp. 273-288, 2003.
5. Black, N. J., Lockett, A., Ennew, C., Winklhofer, H. and McKechnie, S. Modeling consumer choice of distribution channels: An illustration from financial services, International Journal of Bank Marketing, 20(4), pp.161- 173, 2002.
6. C. Narendiran, S. Albert Rabara, and N. Rajendran. Public key infrastructure for mobile banking security, Global Mobile Congress 2009, pp. 1-6,2009.
7. C. Narendiran, S. Rabara, and N. Rajendran, ―Performance evaluation on end-to-end security architecture for mobile banking system, ‖ *Wireless Days, 2008. WD '08. 1st IFIP*, pp. 1-5,2008.
8. Comninos, A., Esselaar, S., Ndiwalana, A. and Stork, C. Towards evidence-based ICT policy and regulation mobile banking the unbanked, http:// externo.casafrica.es/aeo/pdf/english / overview_part_2_09_aeo_09.pdf , 2008.
9. Dai Wei and Tang Yanling. Research on Security Payment Technology Based on Mobile Mobile E- Commerce, e-Business and Information System Security (EBISS), pp.1-4,2010.
10. Dian Yan Liou. Four - scenario analysis for mobile banking development contextualized to Taiwan, Management of Engineering & Technology, PICMET, pp. 2634-2642, 2008.
11. F. de la Puente, S. Gonzalez, and J. Sandoval. Virus attack to the PC bank, Security Technology Proceedings. IEEE 33rd Annual 1999 International Carnahan Conference , pp. 304-310,1999.
12. H. Harb, H. Farahat, and M. Ezz. SecureSMSPay: Secure SMS Mobile Payment model, Anti - counterfeiting, Security and Identification ASID, pp. 11- 17,2008.
13. H. Wu, A. Burt, and R. Thurimella. Making secure TCP connections resistant to server failures, Computer Security Applications Conference 2003, Proceedings. 19th Annual, pp. 197-206,2003.
14. Hayat Muhammad Aslam. Mobile p a yme n t s : Will Colombo keep its leadership in South Asia?, http://sundaytimes.lk/090712/FinancialTimes/ft323.html, 2009.
15. I. Brown, Z. Cajee, D. Davies, and S. Stroebel. Cell phone banking: predictors of adoption in South Africa--an exploratory study, International Journal of Information Management,Vol.23, pp. 381-394, Oct.2003.
16. Jin Nie and Xianling Hu. Mobile Banking Information Security and Protection Methods, Computer Science and Software Engineering, 2008 International Conference , pp. 587-590, 2008.
17. Karjaluoto, H., Mattila, M. and Pento, T. Factors underlying attitude formation towards online banking in Finland, International Journal of Bank Marketing, 20 (6), 261-272, 2002.
18. Lyman, T.R., Pickens M. & Porteous D., "Regulating Transformational Branchless Banking: Mobile Phones and Other Technology to Increase Access to Finance", *In: Focus Note 43*. Consultative Group to Assist the Poor (CGAP), Washington, DC, 2008.
19. Mas, I., "Realizing the Potential of Branchless Banking: Challenges Ahead", *In: Focus Note 50*. Consultancy Group to Assist the Poor (CGAP), Washington, D.C., 2008.
20. Mobile Payment Forum of India (MPFI) http://www.mpf.org.in/
21. Muhammad Bilal, Ganesh Sankar, "Trust & Security issues in Mobile banking and its effect on Customers, School of Computing, Blekinge Institute of Technology, SE-371 79 Karlskrona Sweden, 2011.
22. P. Soni,― M-Payment Between Banks Using SMS [Point of View], ‖ *Proceedings of the IEEE*, vol. 98, pp. 903-905. 2010.
23. Polatoglu, V.N. & Ekin, S., "An empirical investigation of the Turkish consumers' acceptance of Internet banking services", *International Journal of Bank Marketing*, 19 (4), 156-165, 2001.
24. Prerna Sharma Bamoriya & Preeti Singh, "Issues & Challenges in Mobile Banking in India: A Customers' Perspective", Research Journal of Finance & Accounting,Vol.2,No.2, 2011.
25. R. Fisher,―Business aspects of trusted third party services in Europe,‖ *Information Technology Applications in Biomedicine, IT IS - ITAB ' 99. IEEE EMBS International Conference on*, pp. 38-39, 1999.
26. Rao, G. R. & Prathima, K., "Online banking in India", Mondaq Business Briefing, 11 April 2003. Sharma, Prerna & Singh, Preeti, "Users' perception about mobile banking- with special reference to Indore & around", *Review of Business & Technology Research*, Vol. 2, 1, pp. 1-5, 2009.
27. S. Alam, H. Kabir, M. Sakib, A. Sazzad, C. Shahnaz, and S. Fattah, ― A secured electronic transaction scheme for mobile banking Bangladesh incorporating digital watermarking, ‖ *Information Theory and Information Security (ICITIS), 2010 IEEE International Conference on*, pp. 98-102, 2010.
28. S. Hung,C. Ku, and C. Chang, ― Critical factors of WAP services adoption: an empirical study, ‖ *Electronic Commerce Research and Applications*, vol. 2, Spring. pp. 42-60, 2003.
29. S. Wijetunge, P. Wijesinghe, and U. Gunawardana, ― Social Impact of SMS in Sri Lanka, ‖ *Information and Automation for Sustainability, 2008. ICIAFS 2008. 4th nternational Conference on*, pp. 346-350, 2008.
30. Suoranta, M., "Adoption of mobile banking in Finland", *Jyväskylä Studies in Business and Economics*, 28, 2003.
31. Suraj - sankaran, "Mobile Banking Architecture: Palisade", http://palisade.plynt.com / issues /2007May / mobile-banking / ,MAY, 2007.
32. T. Holz, M.Engelberth, F. Freiling," Learning More about the Underground Economy", ESORICS 2009, LNCS 5789, pp. 1–18, 2009
33. T. Wilson, ― Malicious mobile ode, ‖ *Internet Business*, pp. 52-3, Feb.1999.
34. Unithan, C. R. & Swatman, P., "Online banking Adaptation and Dot.Com Viability: A Comparison of Australian and Indian Experiences in the Banking Sector", School of Management Information Systems, Deakin University, No. 14, 2001.
35. Xiaoping Zhang and Cheng Zhong, ― A Loss Reportable E - Cash Scheme without TTP Based on ECC, ‖ *Management of e-Commerce and e-Governmen, ICMECG '08. International Conference on*, 2008, pp.