

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

The use of biometric person recognition for secure access to data /services that are restricted, using a mobile phone with Internet connection have been dealt. Biometrics can be divided into two categories based upon the underlying characteristic they are using: physiological and behavioral. There are three general categories of user authentication: 1) something you know, e.g., passwords and personal-identification numbers , 2) something you have , and 3) something you are.

Today, with the advancement of mobile handsets and wireless networking, mobile devices have both the network access and computing capacity to provide users with a diverse range of services. We have, then, a very popular device, with increasing functionality and access to personally and financially sensitive information; therefore, the requirement for additional and/or advanced authentication mechanisms is essential. Hence, biometric face recognition is implemented to provide more secured access for the web services.

The problem of capturing and sending the biometrics to the web server via PC is very easy to solve using embedded applications in the web pages. The proposal of this project is to present a mobile-phone application architecture to capture and send the biometric to the web server based on the use of an embedded web browser. The current mobile technology is not ready for embedded applications in mobile web browsers; however, it is prepared for our solution, which is very easy and effective.

1.2 SCOPE OF THE PROJECT

- It is designed to implement face detection and recognition for secured access to web services.
- Since password is not the only authentication for access to the sites, this method provides additional security.
- It can be applied for any web server that requires additional authentication.
- A user can access their restricted sites anywhere, anytime through their mobile, as implemented for our project.
- The user's mobile IMEI number is also registered which is used for access to the sites. Hence, the sites cannot be accessed from other mobiles.
- It's a user friendly design which is accurate and fast in identification of the user.

CHAPTER 2

SYSTEM ANALYSIS

2.1 LITERATURE REVIEW

Currently, biometrics is becoming popular as it recognizes the user and provides authentication accordingly. Face recognition is a biometric which is applied in this project, which uses computer software to determine the identity of the individual. Face recognition falls into the category of biometrics which is “the automatic recognition of a person using distinguishing traits”. Other types of biometrics include fingerprinting, retina scans, and iris scan.

Generally, passwords are used for authenticating the user, which is not as secure as thought. Hence, in this project face detection and recognition is used for recognizing the user during the login process.

Initially, while registering the user face is recognized and stored at the back end in server. When all registration process is complete the user is provided with automatically generated user id and account number for login process. During this process, the user’s face is verified with registered face and authentication is provided accordingly. Hence, the user’s physical features are used to provide additional authentication.

2.2 EXISTING SYSTEM

There have been a lot of focuses on this topic. However, there have been a lot of reasons that act as barriers to the popularization of biometric recognition, such as cost of modifying biometric platforms, requirement of high resolution capturing devices and also social acceptance drawbacks.

The current approach for accessing the web services is through password but the major problem with it is that if it is easy to remember, it is usually easy

to guess and hack into, but if it is difficult to attack, it is usually difficult to remember; hence, a lot of people write them down and also never change them.

The problem with the passwords is that they authenticate their presence, but not the person. Also, they can be easily forgotten, lost, or stolen, and can be fraudulently duplicated. As a result, biometry appears as a good solution, which is generally used, in addition to the previous authentication methods, to increase security levels.

2.2.1 DRAWBACKS

- Passwords are easy to hack, if they are easy, and difficult to remember if they are complicated.
- It recognizes the presence of the user but not the user itself.
- Passwords, if difficult to remember, are written on a paper or elsewhere that can be stolen or lost. This is not secure.
- Passwords are the only option used to protect the sensitive and financial information of an individual or group, which is risky.

2.3 PROPOSED SYSTEM

Today, with the advancement of mobile handsets and wireless networking, mobile devices have both the network access and computing capacity to provide users with a diverse range of services. Hence, this is used in providing further secured access for web services using biometrics.

The main characteristics are simplicity, low cost and secured access. A lot of works/applications can be found that focus on the use of biometry with mobile devices; however, here a general proposal to capture biometrics by means of a mobile phone during a standard web session is done. This capture

can only be stored in the server or used with remote (i.e., web service) or local (i.e., mobile data or application) restricted access.

2.3.1 ADVANTAGES

- Accurate and fast identification
- High usability and Security
- User friendly design

2.4 REQUIREMENT SPECIFICATION

The requirements specification is a technical specification of requirements for the software products. It is the first step in the requirements analysis process it lists the requirements of a particular software system including functional, performance and security requirements. The requirements also provide usage scenarios from a user, an operational and an administrative perspective. The purpose of software requirements specification is to provide a detailed overview of the software project, its parameters and goals. This describes the project target audience and its user interface, hardware and software requirements. It defines how the client, team and audience see the project and its functionality.

2.4.1 HARDWARE REQUIREMENTS

The hardware components required are

- Hard Disk: 80 GB and above.
- RAM: 1 GB and above.
- Processor: Pentium III and above.
- Mobile Phone with Wi-Fi Access.
- Wi-Fi Router.

2.4.2 SOFTWARE REQUIREMENTS

The software components required are

- Java 1.6.0_24
- Tomcat 7.0.12
- Struts 1.2
- Eclipse Helios

2.5 SOFTWARE DESCRIPTION

2.5.1 JAVA

It is Platform Independent. Java is an object-oriented programming language developed initially by James Gosling and colleagues at Sun Microsystems. The language, initially called Oak (named after the oak trees outside Gosling's office), was intended to replace C++, although the feature set better resembles that of Objective C.

2.5.2 WORKING OF JAVA

Simplistically, a class is the definition for a segment of code that can contain both data (called attributes) and functions (called methods).

When the interpreter executes a class, it looks for a particular method by the name of **main**, which will sound familiar to C programmers. The main method is passed as a parameter an array of strings (similar to the argv[] of C), and is declared as a static method.

To output text from the program, we execute the **println** method of **System.out**, which is java's output stream.

Java consists of two things :

- Programming language
- Platform

2.5.3 THE JAVA PROGRAMMING LANGUAGE

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

- Simple
- Object-oriented
- Distributed & Interpreted
- Robust
- Secure
- Portable
- High-performance

With most programming languages, we either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first we translate a program into an intermediate language called Java byte codes —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

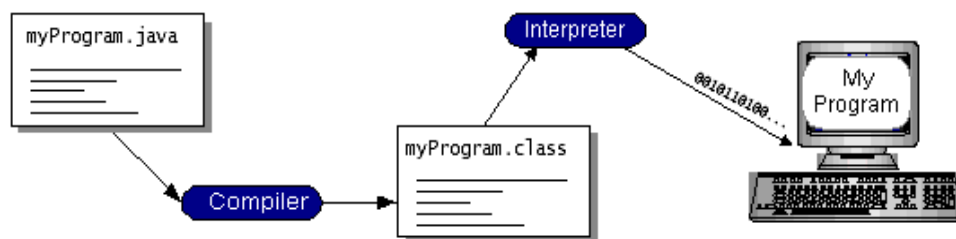


Fig 2.1: Compiler and Interpreter

We can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. We can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.

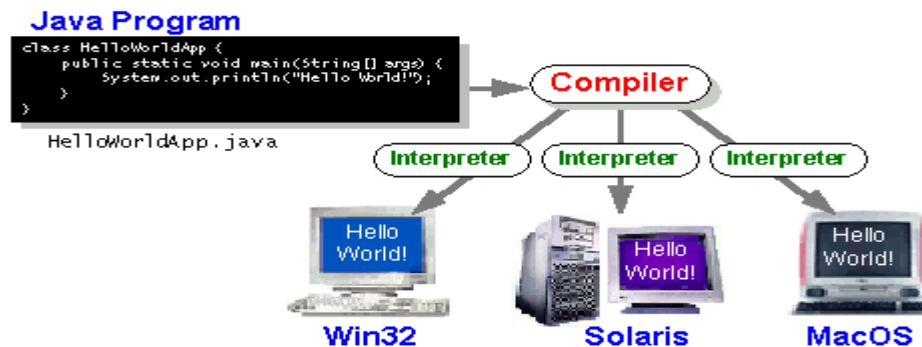


Fig 2.2: Java Execution

2.5.4 THE JAVA PLATFORM

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

The Java platform has two components :

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

You've already been introduced to the JVM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries (**packages**) of related components. The following figure depicts a Java program, such as an application or applet, that's running on the Java platform. As the figure shows, the Java API and Virtual Machine insulates the Java program from hardware dependencies.

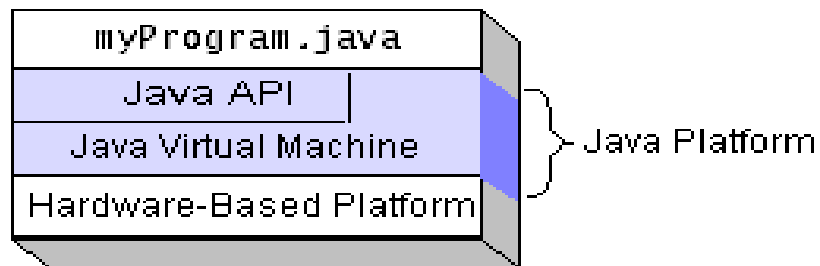


Fig 2.3: Java Platform

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

2.5.5 APACHE TOMCAT SERVER

Apache Tomcat (formerly under the Apache Jakarta Project; Tomcat is now a top level project) is a web container developed at the Apache Software Foundation. Tomcat implements the servlet and the JavaServer Pages ([JSP](#)) specifications from Sun Microsystems, providing an environment for Java code to run in cooperation with a web server. It adds tools for configuration and management but can also be configured by editing configuration files that are

normally [XML](#)-formatted. Because Tomcat includes its own HTTP server internally, it is also considered a standalone web server.

ENVIRONMENT

Tomcat is a web server that supports servlets and JSPs. Tomcat comes with the Jasper compiler that compiles JSPs into servlets. The Tomcat servlet engine is often used in combination with an Apache web server or other web servers. Tomcat can also function as an independent web server. Earlier in its development, the perception existed that standalone Tomcat was only suitable for development environments and other environments with minimal requirements for speed and transaction handling. However, that perception no longer exists; Tomcat is increasingly used as a standalone web server in high-traffic, high-availability environments. Since its developers wrote Tomcat in Java, it runs on any operating system that has a JVM.

2.5.6 INTRODUCTION TO ANDROID

Android is the software platform from Google and the Open Handset Alliance that some say has the potential to revolutionize the global cell phone market. Android is a software environment built for mobile devices. It is not a hardware platform. While components of the underlying OS are written in C or C++, user applications are built for Android in Java. In the Android platform, there is no difference between the built-in applications and applications created with the SDK. This means that powerful applications can be written to tap into the resources available on the device.

LIBRARIES

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

- **System C library** - a BSD-derived implementation of the standard C system library (lib), tuned for embedded Linux-based devices
- **Media Libraries** - based on Packet Video's Open CORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files.
- **Surface Manager** - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications
- **LibWebCore**- a modern web browser engine which powers both the Android browser and an embeddable web view
- **SGL** - the underlying 2D graphics engine 15
- **3D libraries** - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
- **Free Type** - bitmap and vector font rendering
- **SQLite** - a powerful and lightweight relational database engine available to all applications.

ANDROID RUNTIME

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run

multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included "dx" tool. The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management.

2.5.7 ORACLE DATABASE 10G

Oracle Database 10g, released in 2003 and the current release, enables grid (the g in 10g) computing. A grid is simply a pool of computers that provides needed resources for applications on an as-needed basis. The goal is to provide computing resources that transparently scale to the user community, much as an electrical utility company can deliver power to meet peak demand by accessing 16 energy from other power providers' plants via a power grid. Oracle Database 10g further reduces the time, cost, and complexity of database management through the introduction of self-managing features such as the Automated Database Diagnostic Monitor, Automated Shared Memory Tuning, Automated Storage Management, and Automated Disk Based Backup and Recovery.

CHAPTER 3

SYSTEM DESIGN

3.1 INTRODUCTION

This chapter includes the architecture, use case, activity, collaboration, sequence, dataflow diagrams related to the system design.

3.2 SYSTEM ARCHITECTURE

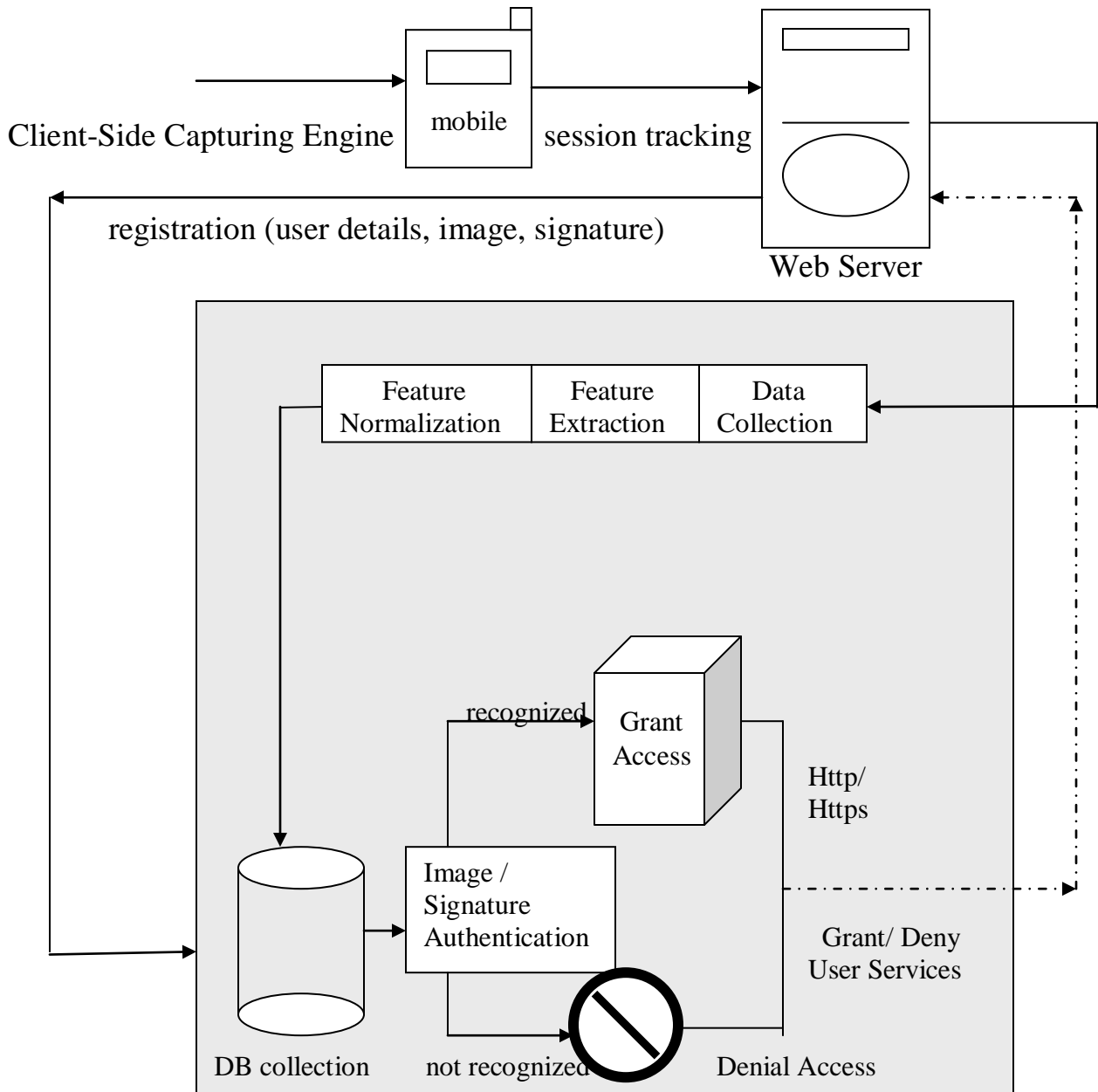


Fig 3.1: System Architecture

3.3 DATA FLOW DIAGRAM

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams.

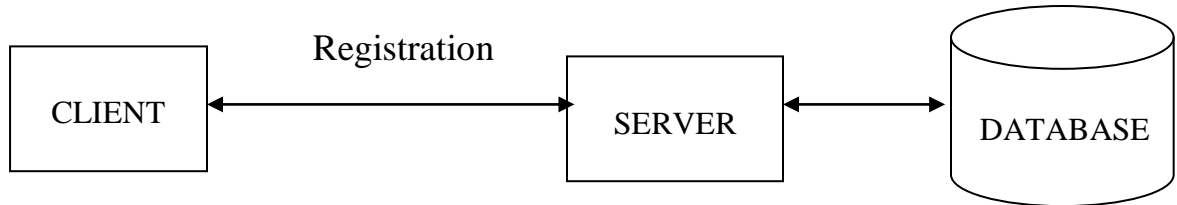
The development of DFD's is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The top-level diagram is often called context diagram. It consists of a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

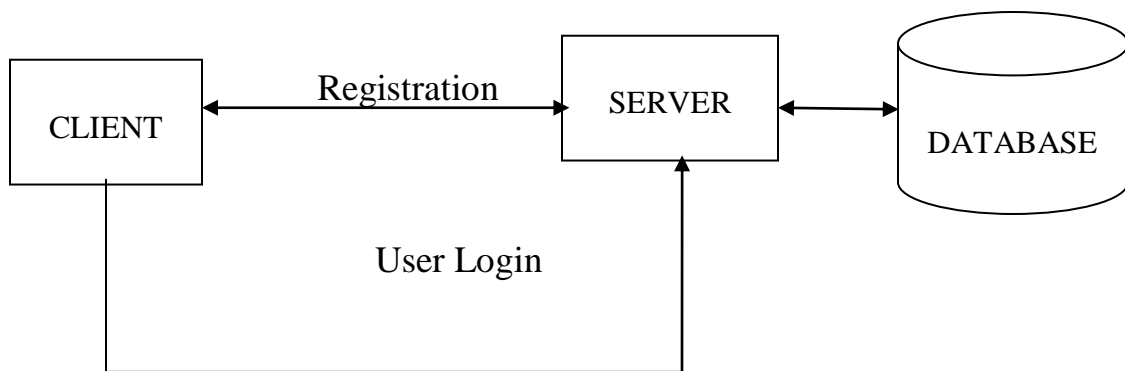
Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical form, this lead to the modular design.

A DFD also known as a "bubble Chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

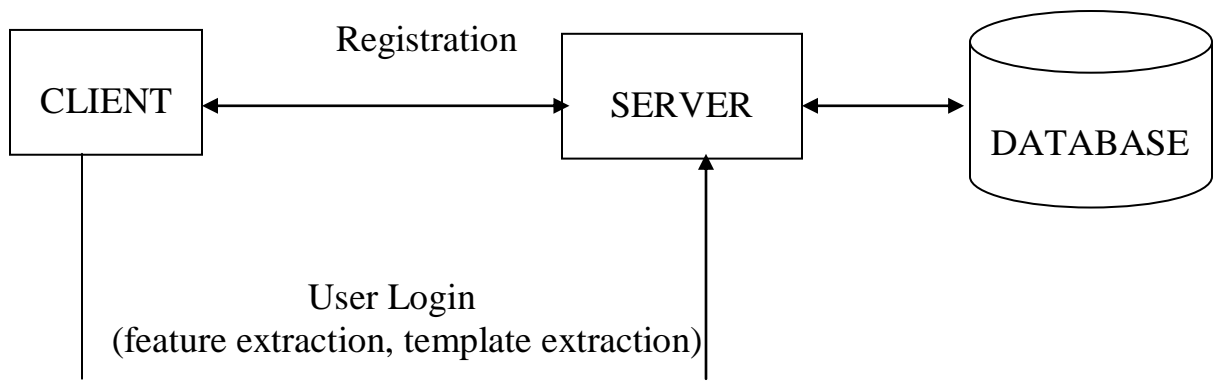
Level 0:



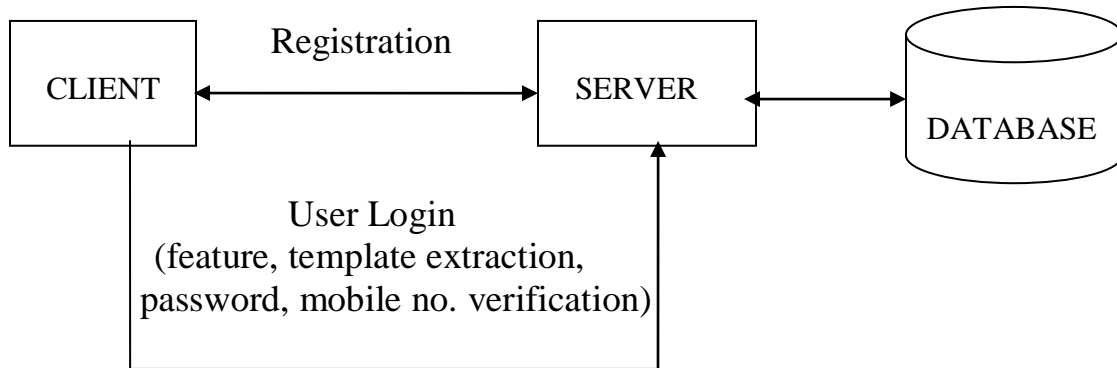
Level 1:



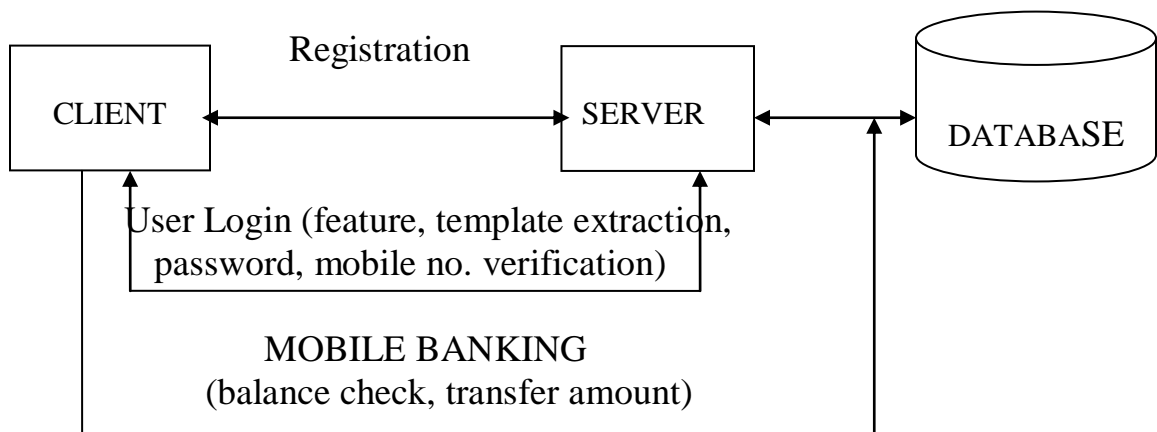
Level 2:



Level 3:



Level 4:



3.4 UML DIAGRAMS

The Unified Modeling Language (UML) is the industry-standard modeling language used for specifying, visualizing, constructing and documenting the artifacts of a software system. The UML is also effective for modeling business and other Non-software systems.

One of the major goals of UML is to present a common modeling language that all developers can use. It is a language whose vocabulary and rules focus on the conceptual and physical representation of the system.

The UML uses Diagram to represent different views of the system being modeled. The purpose of diagram is to present a set of modeled elements, which are rendered as shapes and connectors.

Every complex system is best approached through a small set of nearly independent views of a model. No single viewer is sufficient. Every model may be expressed at different levels of fidelity. The best models are connected to reality.

The UML defines nine graphical diagrams:

- Use-Case Diagram
- Activity Diagram
- Class Diagram
- Collaboration Diagram
- Sequence Diagram

3.4.1 USE CASE DIAGRAM

A use case is a description of a set of sequences of actions, including variants that a system performs to yield an observable result of value to an actor. A use case involves the interaction of actors and the system. Actors (admin) can be human or they can be automated systems.

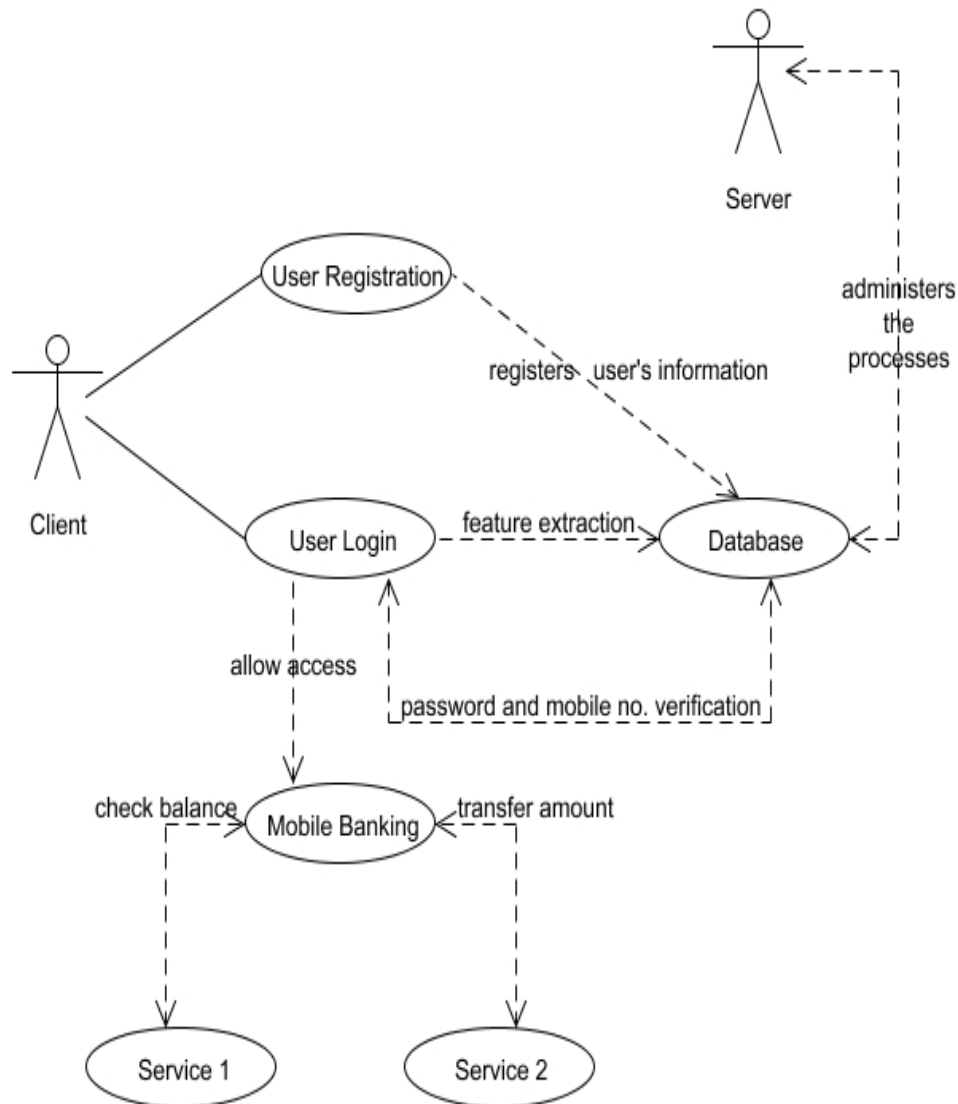


Fig 3.2: Use Case Diagram

3.4.2 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of work flows of step-wise activities and actions with support for choice, iteration and concurrency.

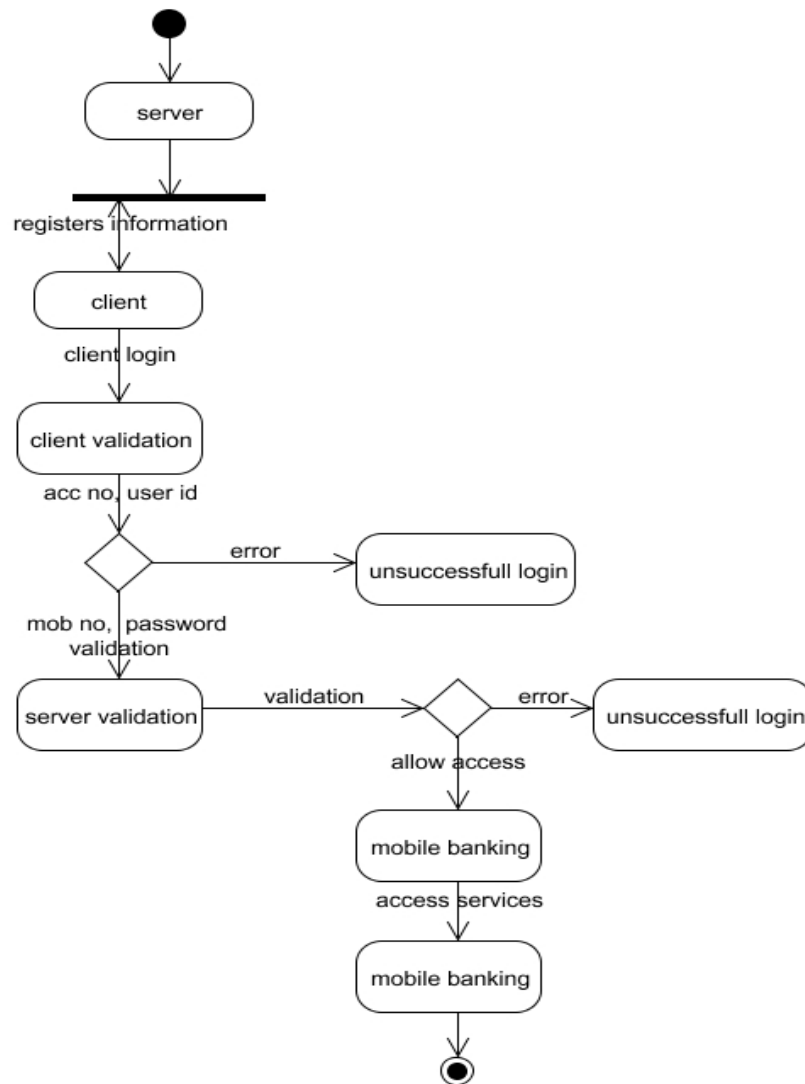


Fig 3.3: Activity Diagram

3.4.3 CLASS DIAGRAM

Class diagrams are the most common diagrams found in modeling object oriented systems. A class diagram shows a set of classes, interfaces, and collaborations and their relationships.

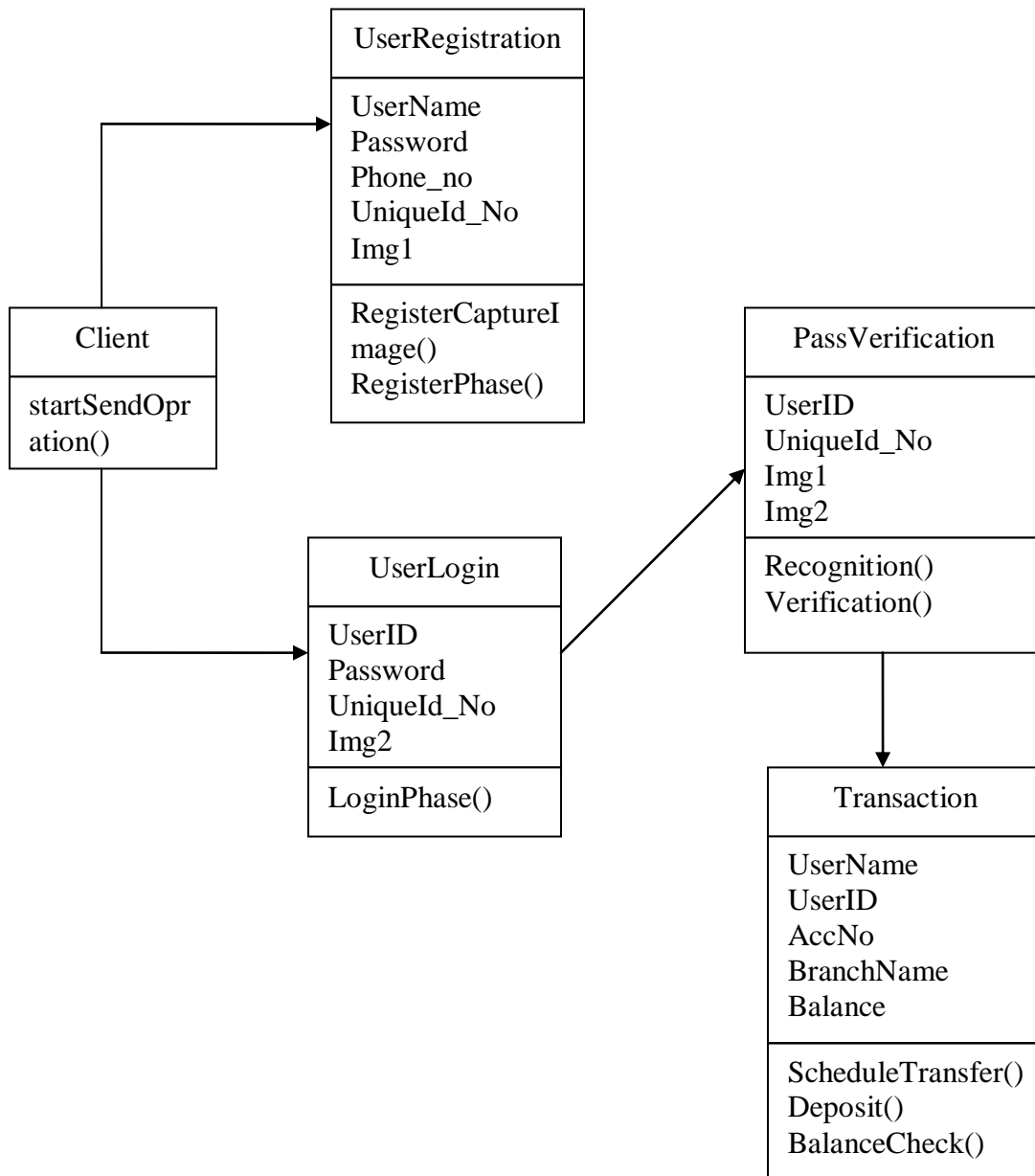


Fig 3.4: Class Diagram

3.4.4 COLLABORATION DIAGRAM

A collaboration diagram is an illustration of the relationships and interactions among software objects in the Unified Modelling Language (UML). Collaboration diagram shows the sequence by numbering the messages on the diagram. This makes it easier to show how the objects are linked together.

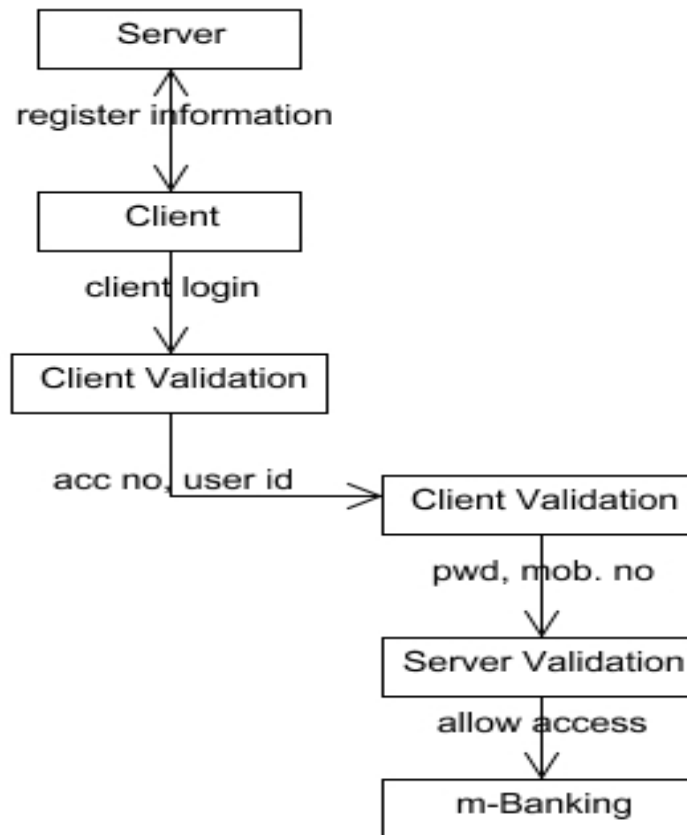


Fig 3.5: Collaboration Diagram

3.4.5 SEQUENCE DIAGRAM

An interaction diagram shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them. A sequence diagram is an interaction diagram that emphasizes the time ordering of messages. Graphically, a sequence diagram is a table that shows objects arranged along the X axis and messages, ordered in increasing time, along the Y axis.

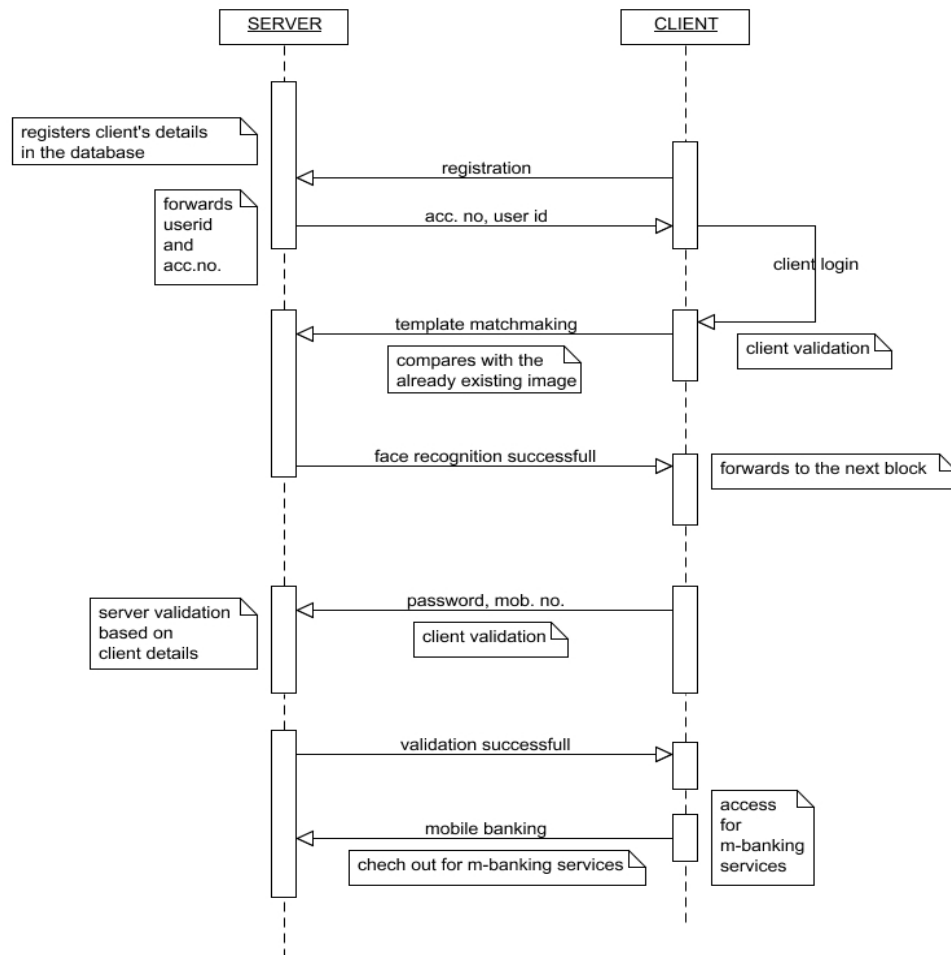


Fig 3.6: Sequence Diagram

3.5 LOGICAL DATABASE DESIGN

TABLE DEFINITIONS:

1. MPB_CLIENTREGISTER

Description: This table stores the user details, bank account details, mobile IMEI number and registered image in encrypted form and its path.

Field Name	Data Type	Size
Username	Varchar	30
Password	Varchar	10
Gender	Varchar	2
Email Id	Varchar	30
Contact No	Varchar	10
Imgpath	Varchar	100
Account No	Varchar	5
User Id	Varchar	6
Balance	Varchar	10
Bank Branch	Varchar	20
IMEI No	Varchar	30
Database Image	Varchar	100

Table 3.1: MPB_CLIENTREGISTER

2. MPB_DRIVINGID

Description: This table contains the user name and driving id details.

Fieldname	Data Type	Size
Username	Varchar	30
Id	Varchar	10

Table 3.2: MPB_DRIVINGID

3. MPB_VOTERSID

Description: This table contains the user name and voter id details.

Fieldname	Data Type	Size
Username	Varchar	30
Id	Varchar	10

Table 3.3: MPB_VOTERSID

4. MPB_PANID

Description: This table contains the user name and pancard id details.

Fieldname	Data Type	Size
Username	Varchar	30
Id	Varchar	10

Table 3.4: MPB_PANID

5. INDIAN BANK

Description: This table contains user name,user id,password,account no,bank balance and branch details of Indian Bank.

Field Name	Data Type	Size
User Name	Varchar	30
Password	Varchar	10
Account No	Varchar	5
User Id	Varchar	6
Balance	Varchar	10
Bank Branch	Varchar	20

Table 3.5: INDIAN BANK

6. STATE BANK

Description: This table contains user name,user id,password,account no,bank balance and branch details of State Bank.

Field Name	Data Type	Size
User Name	Varchar	30
Password	Varchar	10
Account No	Varchar	5
User Id	Varchar	6
Balance	Varchar	10
Bank Branch	Varchar	20

Table 3.6: STATE BANK

CHAPTER 4

IMPLEMENTATION

4.1 MODULES

- **Server Design**
- **Client Registration and ID Generation**
- **User Login and Image Capturing Engine**
- **Feature Extraction and e-Banking**

4.2 MODULE DESCRIPTION

4.2.1 SERVER DESIGN

The server side contains the main parts of the functionality of the proposed architecture. In our first module we deal upon accessing the Server Tier designing part of our project. We deal on working with Tomcat Server and designing our web pages using Struts.

4.2.2 CLIENT REGISTRATION AND ID GENERATION

On the client side, the biometric acquisition software is deployed. Our architecture proposes to leave only the data-capturing module on the client side, with the rest of the modules at the server side. This means that the applications developed need no special memory or processing requirements, since the main computer load falls on the execution of a web navigator and standard mobile devices (e.g., touch screen, microphone, camera, etc.) are used to capture the biometrics. We perform Client Registration from our mobile end to connect it to our server which in turn creates account id and user id for the respective client accordingly.

4.2.3 USER LOGIN AND IMAGE CAPTRING

Biometric Capturer takes in charge of calling and managing the mobile capture devices and a biometric up loader is in charge of sending the biometric data to the server and managing this uploading. A Client makes a login using his account id and user id, which was given after the registration was done successfully, and the server performs a validation using these details. The Biometric Capturer captures the client's image and passes on to the server.

4.2.4 FEATURE EXTRACTION AND E-BANKING

This module collects the raw biometric data and prepares them for processing by the verification engine. This is based on biometric data supplied by the server-side capturing engine, and it generates the feature vectors. In addition to obtaining the features vector or sequence of feature vectors, it is usual to perform further geometric transformations. The database contains information from the users of the system (i.e., user's database subsystem) and their biometric templates (i.e., user's templates subsystem).

The matcher compares the information received against the template of the client stored in the database, thereby generating a numeric comparison score. The Score-normalization is to improve the system performance or to use a universal decision threshold. From the comparison of the score with a decision threshold, this determines whether the user is accepted or rejected and is, thus, granted or denied access to the system or protected services and transmits the output back to the client.

Once the access is granted the user is taken to the e-banking website where he/she can schedule their transfer or check their balance.

4.3. ALGORITHM: STRUCTUTRAL SIMILARITY INDEX:

The **structural similarity** (SSIM) index is a method for measuring the similarity between two images. The SSIM index is an objective image quality metric and a full reference metric, in other words, the measuring of image quality based on an initial uncompressed or distortion-free image as reference. SSIM is designed to improve on traditional methods like peak signal-to-noise ratio (PSNR) and mean squared error (MSE), which have proven to be inconsistent with human eye perception.

The difference with respect to other techniques such as MSE(Mean Squared Error) or PSNR, is that these approaches estimate perceived errors on the other hand SSIM considers image degradation as perceived change in structural information. Structural information is the idea that the pixels have strong inter-dependencies especially when they are spatially close. These dependencies carry important information about the structure of the objects in the visual scene. SSIM Measurement system is illustrated in following diagram.

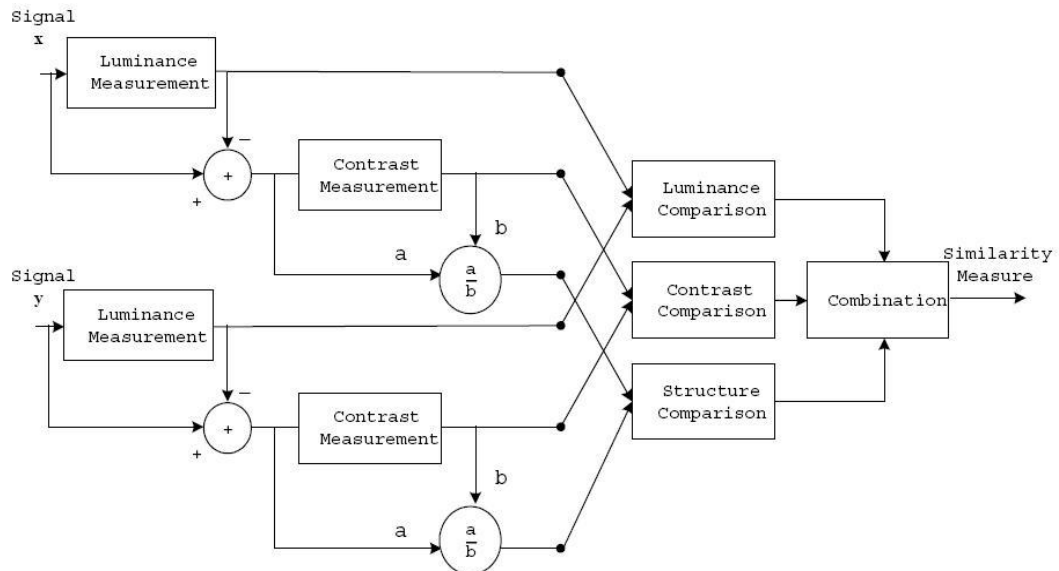


Fig 4.1: SSIM Measurement System

A general form of SSIM is as follows:

$$\text{SSIM}(\mathbf{x}, \mathbf{y}) = [l(\mathbf{x}, \mathbf{y})]^\alpha \cdot [c(\mathbf{x}, \mathbf{y})]^\beta \cdot [s(\mathbf{x}, \mathbf{y})]^\gamma ,$$

where \mathbf{x} and \mathbf{y} are image patches.

[Note that $\alpha > 0$, $\beta > 0$ and $\gamma > 0$ are parameters used to adjust the relative importance of the three components].

- Contrast comparison $c(\mathbf{x}, \mathbf{y})$ - difference of σ_x and σ_y :

$$c(\mathbf{x}, \mathbf{y}) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}$$

- Luminance comparison:

$$l(\mathbf{x}, \mathbf{y}) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}$$

- Structural comparison:

$$s(\mathbf{x}, \mathbf{y}) = \frac{\frac{\sigma_{xy}}{\sigma_x\sigma_y} + C_3}{\frac{\sigma_x^2}{\sigma_x^2} + \frac{\sigma_y^2}{\sigma_y^2} + C_3}$$

Where C_1 , C_2 and C_3 are constants and

- Mean intensity =

$$\mu_x = \frac{1}{N} \sum_{i=1}^N x_i.$$

- Standard deviation =

$$\sigma_x = \left(\frac{1}{N-1} \sum_{i=1}^N (x_i - \mu_x)^2 \right)^{\frac{1}{2}}.$$

The SSIM metric is calculated on various windows of an image. The resultant SSIM index is a decimal value between 0 and 1, and value 1 is only reachable in the case of two identical sets of data. Typically it is calculated on window sizes of 8×8 . The window can be displaced pixel-by-pixel on the image but it is proposed to use only a subgroup of the possible windows to reduce the complexity of the calculation. SSIM gives a much better indication of image quality.

CHAPTER 5

TESTING AND FEASIBILITY REPORT

5.1 SYSTEM TESTING

Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The goal of the testing during phase is to verify that the specification has been accurately and completely incorporated into the design, as well as to ensure the correctness of the design itself. For example the design must not have any logic faults in the design is detected before coding commences, otherwise the cost of fixing the faults will be considerably higher as reflected. Detection of design faults can be achieved by means of inspection as well as walkthrough.

Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases:

- Static analysis is used to investigate the structural properties of the Source code.
- Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

5.1.1 SOFTWARE TESTING STRATEGIES:

The first level of test is unit testing. The purpose of unit testing is to ensure that each program is fully tested. Each module is tested separately.

The second step is integration testing. In this individual program units or programs are integrated and tested as a complete system to ensure that the

software requirements are met. Each module is tested separately and then integrated such as login module integrated with update module and tested.

Acceptance Testing involves planning and the execution of various types of tests in order to demonstrate that the implemented software system satisfies the requirements. Finally our project meets the requirements after going through all the levels of testing.

TEST CASES (ADMIN PAGE)

Test Case Name	Input	Actual Value	Expected Value	Result
Login	Username and password	Username and password is provided	Username and password to be provided	No Error
Customer id selection	Click the drop down list of customer id	Choose the customer id required	Choosing the customer id required	Valid selection of customer id. No Error
Deposit Process	Deposit amount and Bank Branch	The Branch name and the deposit amount of customer is given	The Branch name and the deposit amount of customer to be given	Amount is successfully deposited. No Error

Table 5.1: Test Case (Admin Page)

TEST CASES (ANDROID APPLICATION)

Test Case Name	Input	Actual Value	Expected Value	Result
Validation Process	User name, password, gender, e-mail id, contact number	User's details are given as per the requirements	User's details to be given as per the requirements	Validation is done for details given. No Error
Unique Id	Driving ID, PAN ID or Voter ID of the user	User's driving ID, PAN ID or Voter ID is entered based on option choosen	User's driving ID, PAN ID or Voter ID to be entered based on option choosen	Given value is verified. No Error
Image Capture	Live picture of the user	The image picture of the user is taken	The image picture of the user is taken	The image is stored as a text file. No error
Login Process Details	User id and Account number	The user id and account number provided by server after registration is given	The user id and account number provided by server after registration to be given	The details are verified. No Error

Table 5.2: Test Case (Android Application)

Test Case Name	Input	Actual Value	Expected Value	Result
Login Image Verification	The image of the user for identity verification	The image of the user is taken for identity verification	The image of the user is taken for identity verification	The image is compared and verified with the registered image. No Error
Password Verification	User Password	The password of the user provided during registration is given	The password of the user provided during registration is given	Password is verified. No Error
E-Banking Transaction	Bank Option, Account Number, Branch Name, Transfer Amount	The necessary details is provided by the user for amount transfer to the choosen bank	The necessary details is provided by the user for amount transfer to the choosen bank	Transaction is done successfully. No Error

Table 5.2: Test Case (Android Application)

TEST CASES (BACK END)

Test Case Name	Input	Actual Value	Expected Value	Result
Login	User Name and password	User Name and password is entered	User Name and password to be entered	Access granted. No Error
Unique ID data entry	voter ID, PAN ID and driving ID table entry	User's voter ID, PAN ID and driving ID are entered in the respective tables of the database.	User's voter ID, PAN ID and driving ID are entered in the respective tables of the database.	Entry successful. No Error
Client Registration information storage and updation	User's registration details	User's registration details are to be stored in table that were provided during whole process	User's registration details are to be stored in table that were provided during whole process	All data are entered and updated correctly. No Error

Table 5.3: Test Case (Back End)

5.2 FEASIBILITY REPORT

5.2.1 ECONOMIC FEASIBILITY

Economic analysis is most frequently used for evaluation of the effectiveness of the system. More commonly known as cost/benefit analysis the procedure is to determine the benefit and saving that are expected from a system and compare them with costs, decisions is made to design and implement the system.

This part of feasibility study gives the top management the economic justification for the new system. This is an important input to the management the management, because very often the top management does not like to get confounded by the various technicalities that bound to be associated with a project of this kind. In the system, the organization is most satisfied by economic feasibility. Because, if the organization implements this system, it need not require any additional hardware resources as well as it will be saving lot of time.

5.2.2 TECHNICAL FEASIBILITY

Technical feasibility centres on the existing manual system of the test management process and to what extent it can support the system. According to feasibility analysis procedure the technical feasibility of the system is analyzed and the technical requirements such as software facilities, procedure, inputs are identified. It is also one of the important phases of the system development activities.

The system offers greater levels of user friendliness combined with greater Processing speed. Therefore, the cost of maintenance can be reduced. Since, processing speed is very high and the work is reduced in the maintenance point of view management convince that the project is operationally feasible.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

Face recognition is a technology just reaching sufficient maturity for it to experience a rapid growth in its practical applications. Much research effort around the world is being applied to expanding the accuracy and capabilities of this biometric domain, with a consequent broadening of its application in the near future. Verification systems for physical and electronic access security are available today, but the future holds the promise and the threat passive customization and automated surveillance systems enabled by face recognition.

6.2 FUTURE ENHANCEMENT

Face recognition used today work well under constrained conditions, although all systems work much better with frontal mug-shot images and constant lighting. All current face recognition algorithms fail under the vastly varying conditions under which humans need to and are able to identify other people. Next generation face recognition systems will need to recognize people in real time and in much less constrained situations.

Technology used in smart environments hat to be unobtrusive and allow users to act freely. Wearable systems in particular require their sensing technology to be small, low powered and easily integral with the user's clothing. Considering all the requirements, identification systems that use face recognition seem to us to have the most potential for wide-spread application.

APPENDIX-A

SAMPLE CODE

MOBILEPHONESANDBIOMETRICS.JAVA

```
package mobile.bio;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.telephony.TelephonyManager;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Toast;
public class MobilePhonesAndBiometricsActivity extends Activity {
    /** Called when the activity is first created. */
    ImageView i1;
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        findViewById(R.id.widget29).setSelected(true);
        i1=(ImageView)findViewById(R.id.imageView1);
        Animation a = AnimationUtils.loadAnimation(this, R.anim.rotation);
        i1.startAnimation(a);
    }
}
```

```

        Button login=(Button)findViewById(R.id.login);
        Button register=(Button)findViewById(R.id.register);
        login.setOnClickListener(new LoginListener());
        register.setOnClickListener(new RegisterListener());
    }
    private class LoginListener implements View.OnClickListener{
        public void onClick( View view ){
            //startActivity(new Intent(getApplicationContext(),
            LoginPhase1.class));
        }
    }
    private class RegisterListener implements View.OnClickListener{
        public void onClick( View view ){
            startActivity(new Intent(getApplicationContext(),
            RegisterPhase.class));
        }
    }
}

```

ADMINDEPOSITPROCESS.JAVA

```

package com.yourcompany.struts;

public class AdminDepositProcess extends Action {

    Connection con;
    Statement stmt;
    ResultSet rs;
    String uname,pass,gender,emailid,contactno,accno,userid,succ;
    String balance,dbuserid,newbal;
}

```

```

int bvalue;

public ActionForward execute(ActionMapping mapping,
    ActionForm form, HttpServletRequest request,
    HttpServletResponse response) throws Exception{
    DynaValidatorForm loginForm = (DynaValidatorForm) form;
    String accno = loginForm.get("adminaccnumber").toString();
    String abbranch = loginForm.get("adminbranch").toString();
    String amount = loginForm.get("adminmoney").toString();
    System.out.println("accno....."+accno);
    System.out.println("abbranch....."+abbranch);
    System.out.println("amount....."+amount);
    HttpSession session = request.getSession(true);
    dbconnection dbconn=new dbconnection();
    con=dbconn.getConnect();
    Vector vc=new Vector();
    Statement stat=con.createStatement();
    if(true){
        System.out.println("LOGIN CORRECT");
        ResultSet rs1=stat.executeQuery("select * from
        MPB_CLIENTREGISTER where ACCNO='"+accno+"'");
        while(rs1.next()){
            balance=rs1.getString("BALANCE");
        }
        bvalue=Integer.parseInt(amount)+Integer.parseInt(balance);
        ResultSet rs3=stat.executeQuery("update
        MPB_CLIENTREGISTER set BALANCE='"+bvalue+"' where
        ACCNO='"+accno+"'");
    }
}

```



```

        ResultSet rs4=stat.executeQuery("update
        MPB_CLIENTREGISTER set BANKBRANCH='"+abbranch+"
        where ACCNO='"+accno+"'");
        succ="SUCCESSFULLY DEPOSITED AMOUNT";
    }
    session.setAttribute("depositsuccess",succ);
    return mapping.findForward("depositsuccess");
}
}

```

DBCONNECTION.JAVA

```

package com.yourcompany.struts;
import java.io.FileInputStream;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
import java.util.Properties;
public class dbconnection{
    Connection conn;
    Statement st;
    String DBase="",uname="",pass="";
    Properties prop=null;
    public Connection getConnect() {
    try {
        prop=new Properties();
        prop.load(new FileInputStream("DBConnect.properties"));
        DBase = prop.getProperty("systemname");
    }
    }
}

```

```

        uname = prop.getProperty("username");
        pass =prop.getProperty("password");
        String driver="jdbc:oracle:thin:@" +DBase;
        DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
        conn=DriverManager.getConnection(driver,uname,pass);
        System.out.println("=====Connected=====");
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }
    return conn;
}
}

```

ENCRYPTION

```

package com.yourcompany.struts;

public class Encryption {
    public String g1="";
    public int len;
    public String encrypt(String password)
    {
        int i;
        int b[]=new int[5000];
        int a[]=new int[5000];
        String g=" ";
        char x[]=new char[50];
    }
}

```

```

        char d,s;
String h=password;
len=h.length();
for(i=0;i<len;i++)
{
int aa= h.charAt(i);
a[i]=aa;
}
System.out.print("\n");
System.out.print("HEX-CODE ALG:\n");
for(i=0;i<len;i++)
{
g=" ";
b[i]=a[i]%16;
a[i]=a[i]/16;
if(a[i]==10)
{
        s='A';
        g=g+s;
}
else if(a[i]==11)
{
        s='B';
        g=g+s;
}
else if(a[i]==12)
{

```

```

        s='C';
        g=g+s;
    }
else if(a[i]==13)
{
    s='D';
    g=g+s;
}
else if(a[i]==14)
{
    s='E';
    g=g+s;
}
else if(a[i]==15)
{
    s='F';
    g=g+s;
}
else
{
    s=(char) a[i];
    g=g+a[i];
}
if(b[i]==10)
{
    s='A';
    g=g+s;
}

```

```

        System.out.print(g.trim());
    }
else if(b[i]==11)
{
    s='B';
    g=g+s;
    System.out.print(g.trim());
}
else if(b[i]==12)
{
    s='C';
    g=g+s;
    System.out.print(g.trim());
}
else if(b[i]==13)
{
    s='D';
    g=g+s;
    System.out.print(g.trim());
}
else if(b[i]==14)
{
    s='E';
    g=g+s;
    System.out.print(g.trim());
}
else if(b[i]==15)

```

```

        {
            s='F';
            g=g+s;
            System.out.print(g.trim());
        }
    else
    {
        s=(char) b[i];
        g=g+b[i];
        System.out.print(g.trim());
    }
    g1=g1+g.trim();
}
return g1;
}
}

```

DECRYPTION

```

package com.yourcompany.struts;

public class Decryption {
    public int i,j=0,d=1,h,n,l=8,l2=0,l1=0;
    public char a[]=new char[50000];
    public String t,st="",st1="",vj="";
    public String w =new String();
    public String r =new String();
    public String y="";
    public int c=0;

```

```

public String decrypt(String str2)
{
    int len=str2.length();
    for(i=0;i<len;i++)
    {
        char aa= ((str2).charAt(i));
        a[i]=aa;
        if(a[i]=='A')
        {
            st="1010";
            st1=st;
        }
        else if(a[i]=='B')
        {
            st="1011";
            st1=st;
        }
        else if(a[i]=='C')
        {
            st="1100";
            st1=st;
        }
        else if(a[i]=='D')
        {
            st="1101";
            st1=st;
        }
    }
}

```

```
}  
else if(a[i]=='E')  
{  
    st="1110";  
    st1=st;  
}  
else if(a[i]=='F')  
{  
    st="1111";  
    st1=st;  
}  
else if(a[i]=='0')  
{  
    st="0000";  
    st1=st;  
}  
else if(a[i]=='1')  
{  
    st="0001";  
    st1=st;  
}  
else if(a[i]=='2')  
{  
    st="0010";  
    st1=st;  
}  
else if(a[i]=='3')
```



```

{
    st="0011";
    st1=st;
}
else if(a[i]=='4')
{
    st="0100";
    st1=st;
}
else if(a[i]=='5')
{
    st="0101";
    st1=st;
}
else if(a[i]=='6')
{
    st="0110";
    st1=st;
}
else if(a[i]=='7')
{
    st="0111";
    st1=st;
}
else if(a[i]=='8')
{
    st="1000";

```

```

    }
    else if(a[i]=='9')
    {
        st="1001";
    }
    w=w+st;
}
char w_ch[]=w.toCharArray();
String x=w.toString();
//System.out.println("Length:"+len);
try
{
    if(len==8)
    {
        int k;
        k=len;
        for(j=k-1;j>=0;j--)
        {
            c=c+(w_ch[j]-48)*d;
            y=y+c;
            d=d*2;
        }
        System.out.print("\n");
        vj=vj+(char)c;
        System.out.print((char)c);
    }
    else

```

```

        {
        int k;
        for(i=0;i<10000;i++)
        {
            c=0; d=1;
            k=i*8;
            for(j=k+7;j>=k;j--)
            {
                c=c+(w_ch[j]-48)*d;
                y=y+c;
                d=d*2;
            }
            vj=vj+(char)c;
        }
        //System.out.println("The dec data is"+vj );
    }
}
catch(Exception e)
{

}
return vj;
}
}

```


APPENDIX-B

SCREENSHOTS

DATABASE DESIGNS

Initially, all the required tables are created and the data are entered.

- This table stores the user details that are provided during client registration along with the mobile IMEI number and image path.

MPB_CLIENTREGISTER

Table

Data

Indexes

Model

Constraints

Grants

Statistics

UI Defaults

Triggers

Dependencies

SQL

Add Column

Modify Column

Rename Column

Drop Column

Rename

Copy

Drop

Truncate

Create Lookup Table

Column Name	Data Type	Nullable	Default	Primary Key
USERNAME	VARCHAR2(4000)	Yes	-	-
PASSWORD	VARCHAR2(4000)	Yes	-	-
GENDER	VARCHAR2(4000)	Yes	-	-
EMAILID	VARCHAR2(4000)	Yes	-	-
CONTACTNO	VARCHAR2(4000)	Yes	-	-
IMGPATH	VARCHAR2(4000)	Yes	-	-
ACCNO	VARCHAR2(4000)	Yes	-	-
USERID	VARCHAR2(4000)	Yes	-	-
BALANCE	VARCHAR2(4000)	Yes	-	-
BANKBRANCH	VARCHAR2(4000)	Yes	-	-

1 - 10

- This table stores the user's driving license details that are used during registration process.

MPB_DRIVINGID

TableDataIndexesModelConstraintsGrantsStatisticsUI DefaultsTriggersDependenciesSQL

Add ColumnModify ColumnRename ColumnDrop ColumnRenameCopyDropTruncateCreate Lookup Table

Column Name	Data Type	Nullable	Default	Primary Key
USERNAME	VARCHAR2(4000)	Yes	-	-
ID	VARCHAR2(4000)	Yes	-	-

1 - 2

- This table stores the user's PAN card details that are used during registration process.

MPB_PANID

Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL

Add Column

Modify Column

Rename Column

Drop Column

Rename

Copy

Drop

Truncate

Create Lookup Table

Column Name	Data Type	Nullable	Default	Primary Key
USERNAME	VARCHAR2(4000)	Yes	-	-
ID	VARCHAR2(4000)	Yes	-	-
1 - 2				

- This table stores the user's voter ID details that are used during registration process.

MPB_VOTERSID

Table	Data	Indexes	Model	Constraints	Grants	Statistics	UI Defaults	Triggers	Dependencies	SQL
<div><div>Add Column</div><div>Modify Column</div><div>Rename Column</div><div>Drop Column</div><div>Rename</div><div>Copy</div><div>Drop</div><div>Truncate</div><div>Create Lookup Table</div></div>										
Column Name	Data Type	Nullable	Default	Primary Key						
USERNAME	VARCHAR2(4000)	Yes	-	-						
ID	VARCHAR2(4000)	Yes	-	-						
					1 - 2					

- The data is stored in the mpb_votersid as shown.

MPB_VOTERSID

Create ▾

Table

Data

Indexes

Model

Constraints

Grants

Statistics

UI Defaults

Triggers

Dependencies

SQL

Query

Count Rows

Insert Row

EDIT

USERNAME

ID



d

345

row(s) 1 - 1 of 1

Download

- This table contains the Indian bank users' information and their account details.

INDIANBANK

Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL

Add Column Modify Column Rename Column Drop Column Rename Copy Drop Truncate Create Lookup Table

Column Name	Data Type	Nullable	Default	Primary Key
USERNAME	VARCHAR2(4000)	Yes	-	-
PASSWORD	VARCHAR2(4000)	Yes	-	-
ACCNO	VARCHAR2(4000)	Yes	-	-
USERID	VARCHAR2(4000)	Yes	-	-
BALANCE	VARCHAR2(4000)	Yes	-	-
BANKBRANCH	VARCHAR2(4000)	Yes	-	-

1 - 6

- This table contains the State bank users' information and their account details.

STATEBANK

TableDataIndexesModelConstraintsGrantsStatisticsUI DefaultsTriggersDependenciesSQL

Add ColumnModify ColumnRename ColumnDrop ColumnRenameCopyDropTruncateCreate Lookup Table

Column Name	Data Type	Nullable	Default	Primary Key
USERNAME	VARCHAR2(4000)	Yes	-	-
PASSWORD	VARCHAR2(4000)	Yes	-	-
ACCNO	VARCHAR2(4000)	Yes	-	-
USERID	VARCHAR2(4000)	Yes	-	-
BALANCE	VARCHAR2(4000)	Yes	-	-
BANKBRANCH	VARCHAR2(4000)	Yes	-	-

1 - 6

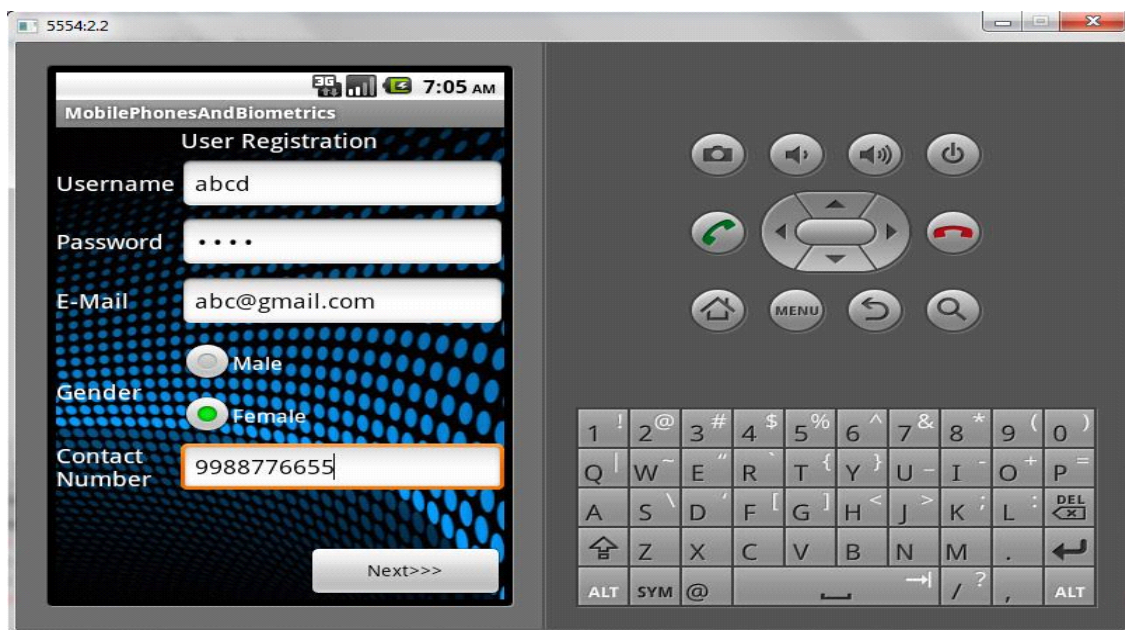
MOBILE PHONE AND BIOMETRICS

REGISTRATION PHASE:

- This is the Home Screen of the android application. The initial step is, the user needs to register first. Click the **Register** button.



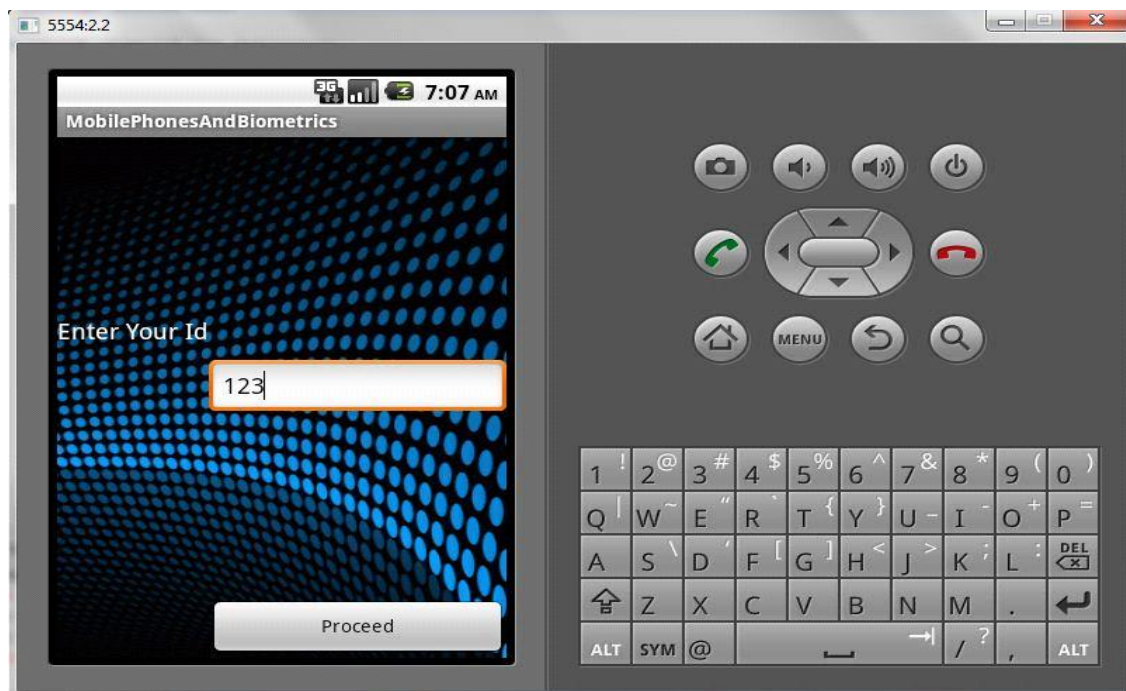
- After clicking the **Register** button, the following details should be given by the user. After giving all the details, click **Next** button to proceed.



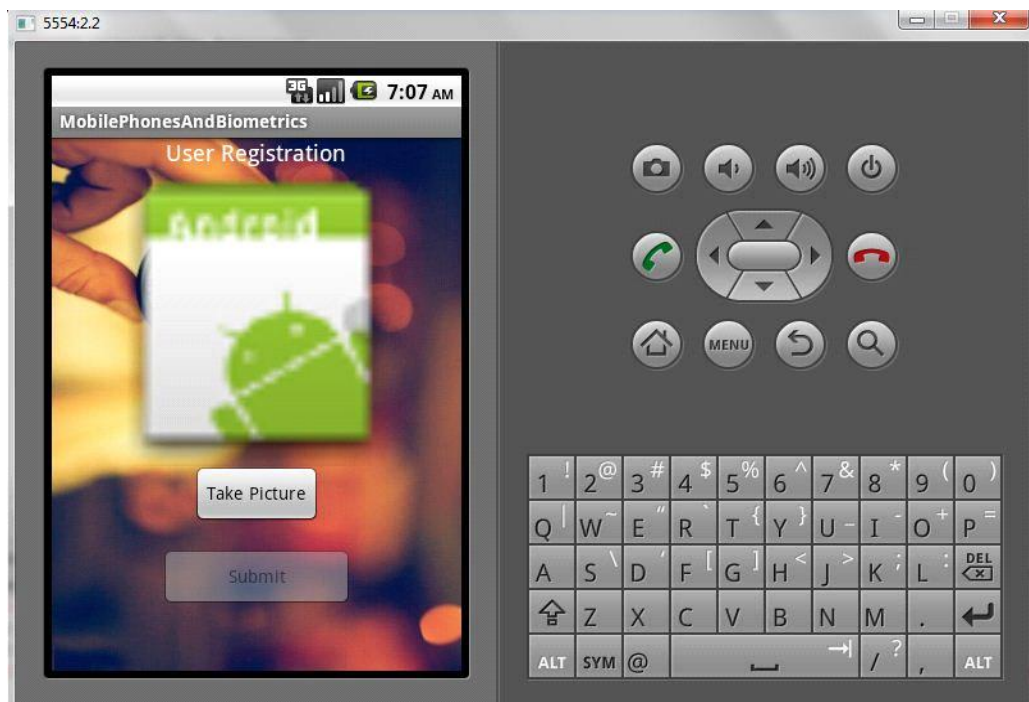
- Next, select the proof which you have and click **Next** button to proceed.



- The next window will prompt you to enter the ID of the particular proof which is already stored in the database. Then click **Proceed** button.




- The Next window will prompt the user to take a photo. Click **Take Picture** button to proceed.



- After clicking the **Take Picture** button, the camera will be displayed. Take the photo of face alone and then click **Submit** button.



- The picture is stored as a text file and its path is stored in the database. After the registration is complete, the details are stored in MPB_CLIENTREGISTER table along with the generated user id and account number.

MPB_CLIENTREGISTER												Create ▼
Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL												
Query Count Rows Insert Row												
EDIT	USERNAME	PASSWORD	GENDER	EMAILID	CONTACTNO	IMGPATH	ACCNO	USERID	BALANCE	BANKBRANCH	IMEINO	DATABASEIMAGE
	abc	1234	Female	abc@gmail.com	9876543210	ENCRYPTEDIMAGES/337331.txt	A68	337331	16000	adayar	911239259164145	-
row(s) 1 - 1 of 1												
Download												

Registration is complete.

LOGIN AND TRANSACTION PHASE

- Next is the Login process where the user id and account number are entered. Then, click **Next** button.
- Click **Take Picture** button and take a picture of the face. Click **Submit** button. If the user is recognized, it takes to the next procedure of password verification and then to e-banking where the transaction and balance check is done.
- The transaction amount is updated in the respective tables in database.

The screenshots of the tables after the transaction are as shown.

STATEBANK

Create

Table

Data

Indexes

Model

Constraints

Grants

Statistics

UI Defaults

Triggers


Dependencies

SQL

Query

Count Rows

Insert Row

EDIT	USERNAME	PASSWORD	ACCNO	USERID	BALANCE	BANKBRANCH
	S	S	123	345	1000	T.Nagar
row(s) 1 - 1 of 1						

Download

INDIANBANK

Create ▼

Table

Data

Indexes

Model

Constraints

Grants

Statistics

UI Defaults

Triggers

Dependencies

SQL

Query

Count Rows

Insert Row

EDIT

USERNAME


PASSWORD

ACCNO

USERID

BALANCE

BANKBRANCH



S

S

678

789

22500

adayar

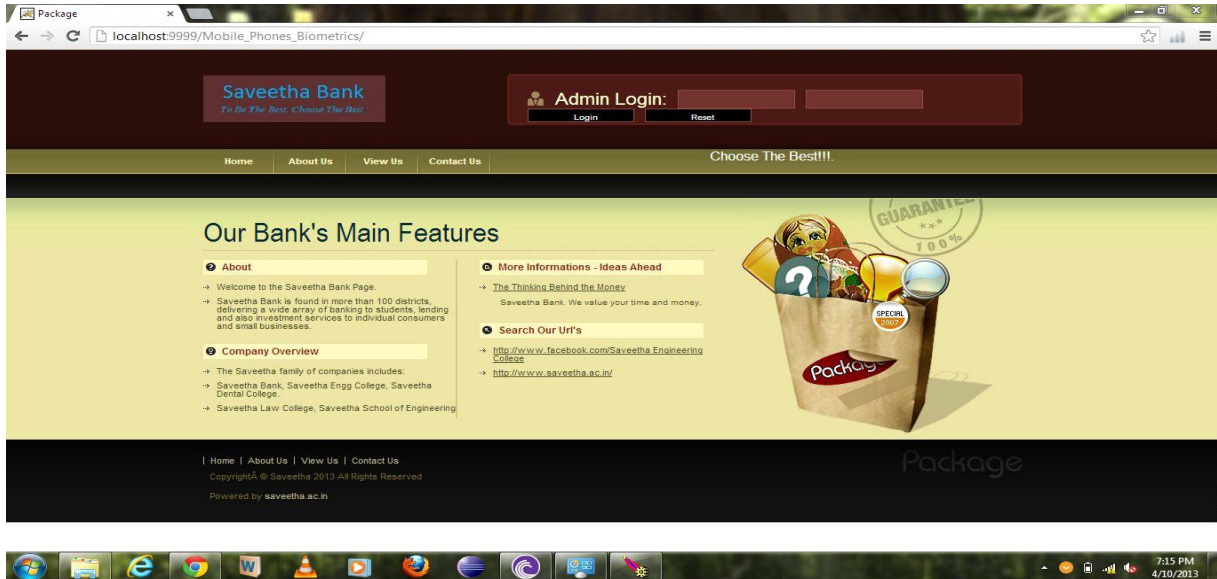
row(s) 1 - 1 of 1

Download

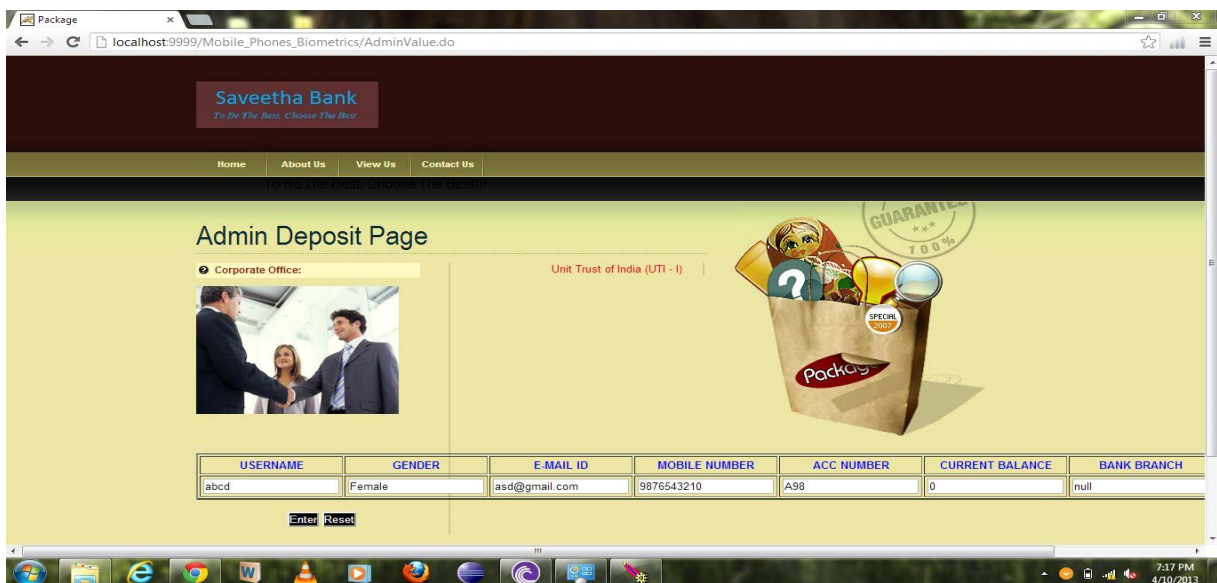
- The amount is updated as per transaction amount entered.
- This completes the Login and transaction phase.

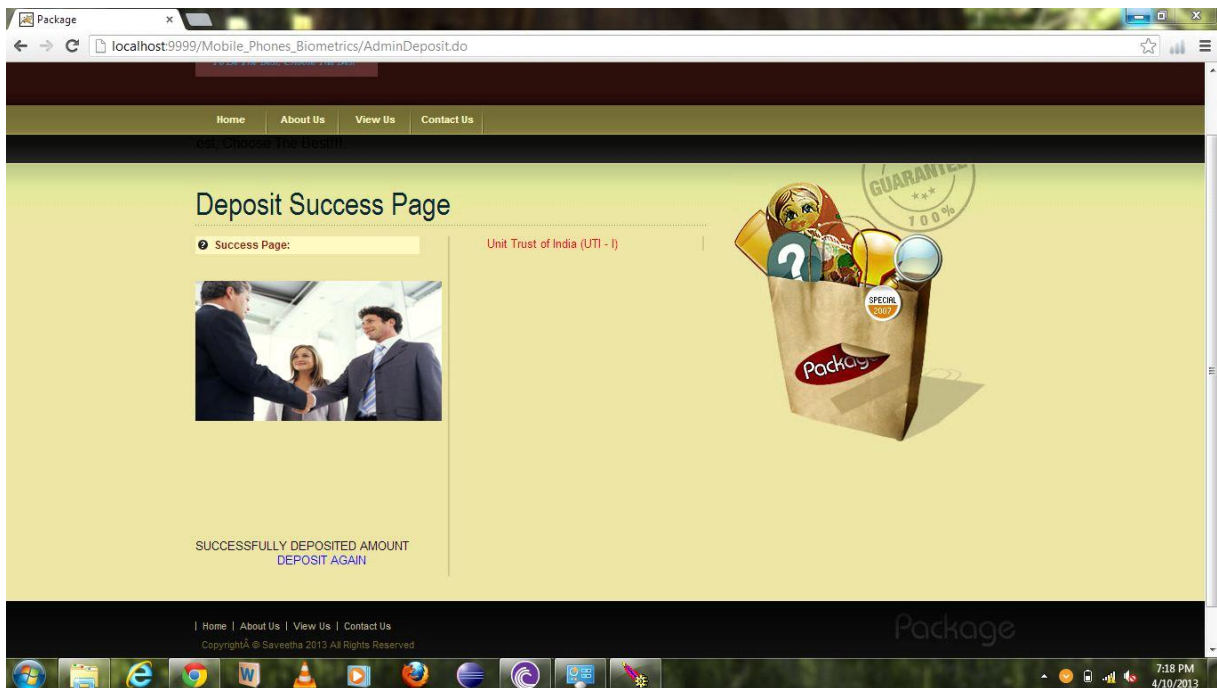
SERVER:

- This is the Home Page of our Web Server. Login name and password should be entered to proceed further for depositing the amount.



- Once logged in, the user can view his details which he gave during registration process along with his bank and account details. Amount can be deposited instantly.





- If an amount is to be deposited again, then click **DEPOSIT AGAIN** button and choose the client's user id from drop down list and proceed further as above.
- Finally, the transaction will be done during e-banking from mobile phone and the transaction amount will be updated both in database and the web server.

REFERENCES

- [1] N. L. Clarke and A. Mekala, “The application of signature recognition to transparent handwriting verification for mobile devices,” *Inf. Manage. Comput. Secur.*, vol. 15, no. 3, pp. 214–225, 2007.
- [2] R. L. Kay. (2003). Protecting mobility, IDC White Paper [Online].
- [3] R. M. Godbole and A. R. Pais, “Secure and efficient protocol for mobile payments,” in *Proc. 10th Int. Conf. Electron. Commerce*, 2008, pp. 1–10.
- [4] D. S. Jeong, H.-A. Park, K. R. Park, and J. Kim, “Iris recognition in mobile phone based on adaptive gabor filter,” *Lect. Notes Comput. Sci.*, vol. 3832, pp. 457–463, 2005.
- [5] Q. Zhang, J. N. Moita, K. Mayes, and K. Markantonakis, “The secure and multiple payment system based on the mobile phone platform,” presented at Workshop Inf. Secur. Appl., Jeju Island, Korea, 2004.
- [6] N. L. Clarke and S. M. Furnell, “Authentication of users on mobile telephones—A survey of attitudes and practices,” *Comput. Secur.*, vol. 24, no. 7, pp. 519–527, 2005.
- [7] K. R. Park, H.-A. Park, B. J. Kang, E. C. Lee, and D. S. Jeong, “A study on iris localization and recognition on mobile phones,” *EURASIP J. Adv. Signal Process.*, vol. 2008, pp. 1–11, 2008.

- [8] M. Martinez-Diaz, J. Fierrez, J. Galbally, and J. Ortega-Garcia, “Towards mobile authentication using dynamic signature verification: Useful features and performance evaluation,” in *Proc. 19th Int. Conf. Pattern Recogn.*, Dec. 2008, pp. 1–5.
- [9] D. J. Hurley, B. Arbab-Zabar, and M. S. Nixon, “The ear as a biometric,” in *Handbook on Biometrics*, A. K. Jain, A. A. Ross, and P. Flynn, Eds. New York: Springer-Verlag, 2008, pp. 131–150.
- [10] S. yi Han, H.-A. Park, D.H. Cho, K. R. Park, and S. Lee, “Face recognition based on near-infrared light using mobile phone,” in *Proc. ICANNGA, Part II (Lect. Notes Comput. Sci. vol. 4432)*, 2007, pp. 440–448.