**KNOWLEDGE RECALL-BASED AUTHENTICATION**

**SYSTEM**

**INNOVATIVE PRODUCT DEVELOPMENT REPORT**

***Submitted by***

1. **SHIRISHA N. ASHWINI**

**20RH1A6602 20RH1A6640**

***Under the Esteemed Guidance of***

**Mr. MAHESH**

**Assistant Professor**

***in partial fulfillment of the Academic Requirements for the Degree of***

**BACHELOR OF TECHNOLOGY**

 **CSE-AI & ML**

**MALLA REDDY ENGINEERING COLLEGE FOR WOMEN**

## (Autonomous Institution-UGC, Govt. of India)

#### Accredited by NBA & NAAC with ‘A’ Grade, UGC, Govt. of India

**NIRF Indian Ranking–2018, Accepted by MHRD, Govt. of India**

**Permanently Affiliated to JNTUH, Approved by AICTE, ISO 9001:2015 Certified Institution AAAA+ Rated by Digital Learning Magazine, AAA+ Rated by Careers 360 Magazine,**

**6th Rank CSR, Platinum Rated by AICTE-CII Survey, Top 100 Rank band by ARIIA, MHRD, Govt. of India National Ranking-Top 100 Rank band by Outlook, National Ranking-Top 100 Rank band by Times Ne ws Magazine Maisammaguda, Dhullapally, Secunderabad, Kompally-500100**

**Nov- 2022**

**MALLA REDDY ENGINEERING COLLEGE FOR WOMEN**

## (Autonomous Institution-UGC, Govt. of India)

#### Accredited by NBA & NAAC with ‘A’ Grade, UGC, Govt. of India

**NIRF Indian Ranking–2018, Accepted by MHRD, Govt. of India Permanently Affiliated to JNTUH, Approved by AICTE, ISO 9001:2015 Certified Institution**

**AAAA+ Rated by Digital Learning Magazine, AAA+ Rated by Careers 360 Magazine,**

**6th Rank CSR, Platinum Rated by AICTE-CII Survey, Top 100 Rank band by ARIIA, MHRD, Govt. of India National Ranking-Top 100 Rank band by Outlook, National Ranking-Top 100 Rank band by Times Ne ws Magazine Maisammaguda, Dhullapally, Secunderabad, Kompally-500100**

**DEPARTMENT OF CSE - AI &ML**

## CERTIFICATE

This is to certify that the Innovative Product Development work entitled **Knowledge recall-based** **authentication system** is carried out by **A.SHIRISHA (20RH1A6602),N.ASHWINI (20RH1A6640)** in partial fulfillment for the award of degree of **BACHELOR OF TECHNOLOGY** in CSE - AI & ML, Jawaharlal Nehru Technological University, Hyderabad during the academic year 2022-2023.

## Supervisor’s Signature Head of the Department

**Mr. Mahesh Dr. G. Kalpana**

## Assistant Professor Professor

**External Examiner**

**ii**

# 

# MALLA REDDY ENGINEERING COLLEGE FOR WOMEN

#### (Autonomous Institution –UGC,Govt. of India)

(Permanently Affiliated to JNTU, Hyderabad, Approved by AICTE - ISO 9001:2015 Certified)

#### Accredited by NBA & NAAC with ‘A’ Grade

**NIRF India Ranking 2018, Accepted by MHRD, Govt. of India**

Maisammaguda, Dhulapally (Post Via Hakimpet), Secunderabad – 500100

# Department of CSE –AI & ML

**DECLARATION**

We hereby declare that the Innovative Product Development entitled **Knowledge recall-based authentication system** submitted to Malla Reddy Engineering College for Women affiliated to Jawaharlal Nehru Technological University, Hyderabad (JNTUH) for the award of the Degree of Bachelor of Technology in CSE-AI & ML is a result of original research work done by us. It is further declared that the Innovative Product Development report or any part thereof has not been previously submitted to any University or Institute for the award of Degree.

1. **SHIRISHA(20RH1A6602)**
2. **ASHWINI(20RH1A6640)**

**iii**

# ACKNOWLEDGEMENT

We feel ourselves honored and privileged to place our warm salutation to our college **Malla Reddy Engineering College for Women** and **Department of Computer Science and Engineering** which gave us the opportunity to have expertise in engineering and profound technical knowledge.

We would like to deeply thank our Honorable Minister of Telangana State **Sri**.**Ch. Malla Reddy Garu,** founder chairman MRGI, the largest cluster of institutions in the state of Telangana for providing us with all the resources in the college to make our project success.

We wish to convey gratitude to our **Principal Dr. Y. Madhavee Latha,** for providing us with the environment and mean to enrich our skills and motivating us in our endeavor and helping us to realize our full potential.

We would like to thank **Prof. A. Radha Rani**, Director of Computer Science and Engineering & Information Technology for encouraging us to take up a project on this subject and motivating us towards the Project Work.

We express our sincere gratitude to **Dr.G.Kalpana , Head of the Department** of CSE –AI & ML for inspiring us to take up a project on this subject and successfully guiding us towards its completion.

We would like to thank our guide **Mr. Mahesh** , Assistant Professor and all the Faculty members for their valuable guidance and encouragement towards the completion of our project work.

#### **With Regards and Gratitude**

#### **A SHIRISHA (20RH1A6602)**

**N ASHWINI (20RH1A6640)**

# iv

**ABSTRACT**

TEXTUAL passwords have been the most widely used authentication method for decades. Comprised of numbers and upper- and lower-case letters, textual passwords are considered strong enough to resist against brute force attacks. However, a strong textual password is hard to memorize and recollect. Therefore, users tend to choose passwords that are either short or from the dictionary, rather than random alphanumeric strings. Various graphical password authentication schemes were developed to address the problems and weaknesses associated with textual passwords. Based on some studies such as those in , humans have a better ability to memorize images with long-term memory(LTM) than verbal representations. Image-based passwords were proved to be easier to recollect in several user studies As a result, users can set up a complex authentication password and are capable of recollecting it after a long time even if the memory is not activated periodically.

The human actions such as choosing bad passwords for new accounts and inputting passwords in an insecure way for later logins are regarded as the weakest link in the authentication chain. Therefore, an authentication scheme should be designed to overcome these vulnerabilities. In this paper, we present a secure graphical authentication system named Pass Matrix that protects users from becoming victims of shoulder surfing attacks when inputting passwords in public through the usage of one-time login indicators. A login indicator is randomly generated for each pass-image and will be useless after the session terminates. The login indicator provides better security against shoulder surfing attacks, since users use a dynamic pointer to point out the position of their passwords rather than clicking on the password object directly.

Index

**Title……………………………………………………………………………………………………………………………………………….**

**Certificate…………………………………………………………………………………………………………………………ii**

**Declaration……………………………………………………………………………………………………………………….iii**

[Acknowledgement i](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374260)v

[Abstract:](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374261) v

Chapter 1

[INTRODUCTION](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374260) 1

[1.1 Introduction:](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374261) 1

[1.2 Implementation](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374261) 3

Chapter 2

[SYSTEM FEASIBILITY](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374260) 11

2.1 Introduction11

2.2 Existing System15

2.3 Proposed System 15

Chapter 3

[SYSTEM ANALYSIS](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374260) 17

[3.1 The Study of The System](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374261) 17

3.2 Input & Output Representation 18

3.3 Introduction to Java 20

3.4 System Requirements 37

3.4.1 Hardware Requirements 37

3.4.2 Software Requirements 37

Chapter 4

[SYSTEM DESIGN](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374260) 38

4.1 System Architecture 38

4.2 UML Diagrams 43

Chapter 5

[TESTING & VALIDATION](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374260) 50

Chapter 6

OUTPUT [SCREEN SHOTS](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374260) 55

Chapter 7

[CONCLUSIONS](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374260) 60

**FUTURESCOPE……………………………………………………………………………………………………………………………..61**

[REFERENCES](file:///C:\Users\ashwini\Desktop\Temp\e-Learning.docx#_Toc442374260) 79

**List of Figures**

**Fig. No Name of the figure Page no**

1 System Architecture of Project........................................................38

2 Use Case diagram……………….......................................................44

3 Sequence diagram………………………...........................................45

4 Class Diagram of User and E-Commerce.........................................47

**CHAPTER 1**

**INTRODUCTION**

**1.1 Introduction:**

Textual passwords have been the most widely used authentication method for decades. Comprised of numbers and upper- and lower-case letters, textual passwords are considered strong enough to resist against brute force attacks. However, a strong textual password is hard to memorize and recollect. Therefore, users tend to choose passwords that are either short or from the dictionary, rather than random alphanumeric strings. Even worse, it is not a rare case that users may use only one username and password for multiple accounts. According to an article in Computer world, a security team at a large company ran a network password cracker and surprisingly cracked approximately 80% of the employees’ passwords within 30 seconds. Textual passwords are often insecure due to the difficulty of maintaining strong ones.

Various graphical password authentication schemes were developed to address the problems and weaknesses associated with textual passwords. Based on some studies such as those in humans have a better ability to memorize images with long-term memory (LTM) than verbal representations. Image-based passwords were proved to be easier to recollect in several user studies. As a result, users can set up a complex authentication password and are capable of recollecting it after a long time even if the memory is not activated periodically. However, most of these image-based passwords are vulnerable to shoulder surfing attacks (SSAs). This type of attack either uses direct observation, such as watching over someone’s shoulder or applies video capturing techniques to get passwords, PINs, or other sensitive personal information.

The human actions such as choosing bad passwords for new accounts and inputting passwords in an insecure way for later logins are regarded as the weakest link in the authentication chain. Therefore, an authentication schemeshould be designed to overcome these vulnerabilities.

In this paper, we present a secure graphical authentication system named PassMatrix that protects users from becoming victims of shoulder surfing attacks when inputting passwords in public through the usage of one-time login indicators. A login indicator is randomly generated for each pass-image and will be useless after the session terminates. The login indicator provides better security against shoulder surfing attacks, since users use a dynamic pointer to point out the position of their passwords rather than clicking on the password object directly.online business review/rating.

.

**Related work**

In the past several decades, a lot of research on password authentication has been done in the literature. Among all of these proposed schemes, this paper focuses mainly on the graphical-based authentication systems. To keep this paper concise, we will give a brief review of the most related schemes that were mentioned in the previous section. Many other schemes such as those in may have good usability, they are not graphical-based and need additional support from extra hardware such as audio, multi-touch monitor, vibration sensor, or gyroscope, etc.

In the early days, the graphical capability of handheld devices was weak; the color and pixel it could show was limited. Under this limitation, the Draw-a-Secret (DAS) technique was proposed by Jermyn et al. in 1999, where the user is required to re-draw a pre-defined picture on a 2D grid. We directly extract the figure from and show it in Figure 1(b). If the drawing touches the same grids in the same sequence, then the user is authenticated. Since then, the graphical capability of handheld devices has steadily and ceaselessly improved with the advances in science and technology. In 2005, Susan Wiedenbeck et al. introduced a graphical authentication scheme PassPoints, and at that time, handheld devices could already show high resolution color pictures. Using the PassPoint scheme, the user has to click on a set of pre-defined pixels on the predestined photo, as shown in Figure 1(a), with a correct sequence and within their tolerant squares during the login stage. Moreover, Marcos et al.

**1.2 IMPLEMENTATION**

**MODULES:**

**Admin**

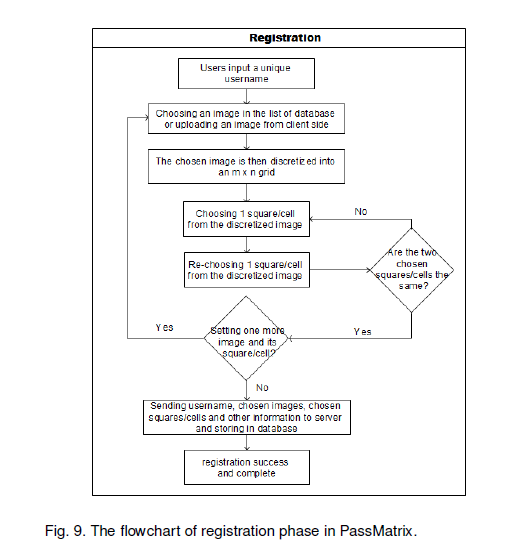
1. In this module, admin has to login with valid username and password. After login successful he can do some operations such as view all user, their details and authorize them, View all users graphical authentication points, view all blocked users (who tried wrong graphical authentication points 3 time), View unblock requests and unblock them, upload documents with image and view all uploaded documents with rank and comments of it, View users results chart based on number of users active and blocked, View the documents results based on rank.

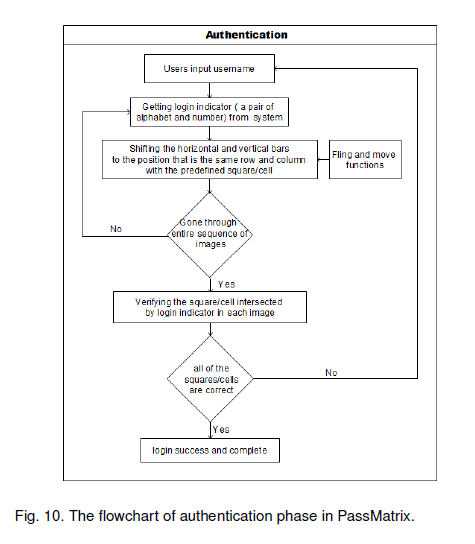
* **User**

In this module, there are n numbers of users are present. User should register before doing some operations and also set graphical authentication points while registration. After registration successful he can login by using valid user name and password and also graphical authentication points . Login successful he will do some operations like view profile details, Chang graphical authentication points, search documents and download/View and comment on it/recommend to others, View all recommended documents.

Module Description

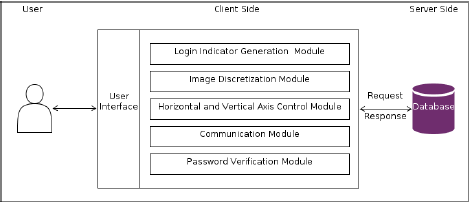
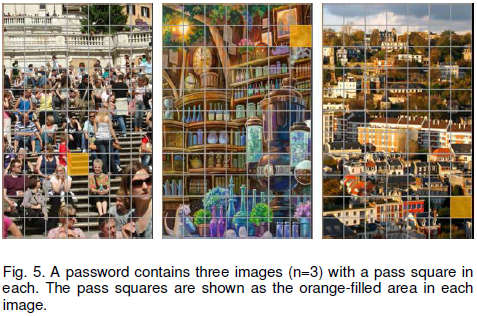
1. **Multi Layer Image Authentication**
2. **Grid Image Authentication**
3. **Color Image Authentication**
4. **Random Guess Attack**
5. **Login / Register**
6. **Upload Image**
7. **View Status**
8. **View Requests**
9. **Approve / Cancel**

****

****

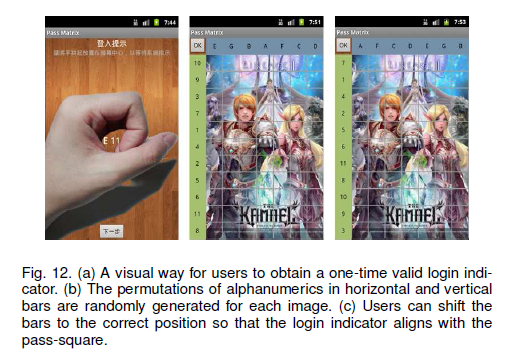
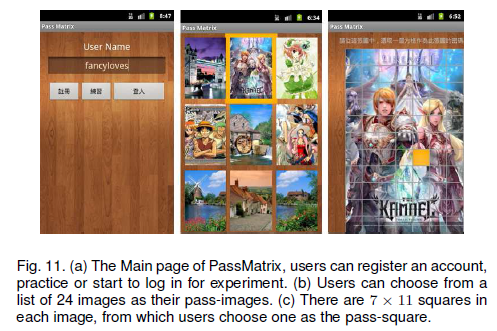
1. **Multi-Layer Image Authentication**

To overcome the security weakness of the traditional PIN method, the easiness of obtaining passwords by observers in public, and the compatibility issues to devices, we introduced a graphical authentication system called Pass Matrix. In Pass Matrix, a password consists of only one pass-square per pass-image for a sequence of n images. The number of images (i.e., n) is user-defined. Bellow figure demonstrates the proposed scheme, in which the first pass-square is located at in the first image, the second pass-square is on the top of the smoke in the second image at , and the last pass-square is at in the third image. In Pass Matrix, users choose one square per image for a sequence of n images rather than n squares in one image as that in the Pass Points scheme. Based on the user study of Cued Click Points . CCP method does a good job in helping users recollect and remember their passwords. If the user clicks on an incorrect region within the image the login will be failed



1. **Grid Image Authentication**

In this type of authentication multiple images can be provided to the user, the user has the select the image that he can to log in, this will the provide more security.



1. **Color Image Authentication**

In this type the authentication is user by the color coordinates of that position. In normal Authentication the password is setting according to the regions. But in this type of authentication we choose the color coordinates for password setting

1. **Random Guess Attack**

To perform a random guess attack, the attacker randomly

tries each square as a possible pass-square for each pass image until a successful login occurs. The key security determinants of the system are the number of pass-images and the degree of discretization of each image. To quantify the security of Pass Matrix against random guess attacks, we define the entropy of a password space as in equation 3. Table 7 defines the notations used in the equation. If the entropy of a password space is k bits, there will be 2kpossible passwords in that space.

Entropy = log2((Dx \_ Dy)i)n

****

1. **Login / Register**

The application will provide a secure user-id/password based secured login mechanism to access its services.

1. **Upload Image**

This is the main module in this application . The Main Process in the Mex application will be worked here. The bill picture is already stored in the mobile gallery . the user will select the picture from the gallery and upload in to the server. And also upload the details like employee name , employee id and Bill details. All the details uploaded here is stored in to the wamp server

1. **View Status**

After uploading the details the user can check the status of the request using the same application. The status will be shown as pending until the higher authority accept or cancel the Request

1. **View Request**

The User Requested data can be view by the Higher authority. Admin is the authority to accept or reject the request. This module is done by using PHP. The Admin will use System to view the request

1. **Approve / Cancel**

After viewing the Request the admin can have the permission to accept or reject the request. The user can check the status

**Algorithms Implemented**

1. **Random Guess Attack**

To perform a random guess attack, the attacker randomly

tries each square as a possible pass-square for each pass image until a successful login occurs. The key security determinants of the system are the number of pass-images and the degree of discretization of each image. To quantify the security of Pass Matrix against random guess attacks, we define the entropy of a password space as in equation 3. Table 7 defines the notations used in the equation. If the entropy of a password space is k bits, there will be 2kpossible passwords in that space.

Entropy = log2((Dx \_ Dy)i)n

****

**CHAPTER: 2**

**SYSTEM FEASIBILITY**

**2.1 Introduction**:

**PRELIMINARY INVESTIGATION**

The first and foremost strategy for development of a project starts from the thought of designing a mail enabled platform for a small firm in which it is easy and convenient of sending and receiving messages, there is a search engine ,address book and also including some entertaining games. When it is approved by the organization and our project guide the first activity, ie. preliminary investigation begins. The activity has three parts:

* **Request Clarification**
* **Feasibility Study**
* **Request Approval**

**REQUEST CLARIFICATION**

After the approval of the request to the organization and project guide, with an investigation being considered, the project request must be examined to determine precisely what the system requires.

Here our project is basically meant for users within the company whose systems can be interconnected by the Local Area Network(LAN). In today’s busy schedule man need everything should be provided in a readymade manner. So taking into consideration of the vastly use of the net in day to day life, the corresponding development of the portal came into existence.

**FEASIBILITY ANALYSIS**

An important outcome of preliminary investigation is the determination that the system request is feasible. This is possible only if it is feasible within limited resource and time. The different feasibilities that have to be analyzed are

* **Operational Feasibility**
* **Economic Feasibility**
* **Technical Feasibility**

###### Operational Feasibility

Operational Feasibility deals with the study of prospects of the system to be developed. This system operationally eliminates all the tensions of the Admin and helps him in effectively tracking the project progress. This kind of automation will surely reduce the time and energy, which previously consumed in manual work. Based on the study, the system is proved to be operationally feasible.

**Economic Feasibility**

Economic Feasibility or Cost-benefit is an assessment of the economic justification for a computer based project. As hardware was installed from the beginning & for lots of purposes thus the cost on project of hardware is low. Since the system is a network based, any number of employees connected to the LAN within that organization can use this tool from at anytime. The Virtual Private Network is to be developed using the existing resources of the organization. So the project is economically feasible.

###### Technical Feasibility

According to Roger S. Pressman, Technical Feasibility is the assessment of the technical resources of the organization. The organization needs IBM compatible machines with a graphical web browser connected to the Internet and Intranet. The system is developed for platform Independent environment. Java Server Pages, JavaScript, HTML, SQL server and WebLogic Server are used to develop the system. The technical feasibility has been carried out. The system is technically feasible for development and can be developed with the existing facility.

**REQUEST APPROVAL**

Not all request projects are desirable or feasible. Some organization receives so many project requests from client users that only few of them are pursued. However, those projects that are both feasible and desirable should be put into schedule. After a project request is approved, it cost, priority, completion time and personnel requirement is estimated and used to determine where to add it to any project list. Truly speaking, the approval of those above factors, development works can be launched.

**SYSTEM DESIGN AND DEVELOPMENT**

**INPUT DESIGN**

Input Design plays a vital role in the life cycle of software development, it requires very careful attention of developers. The input design is to feed data to the application as accurate as possible. So inputs are supposed to be designed effectively so that the errors occurring while feeding are minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range and other related validations.

This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design.

Input design is the process of converting the user created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error is in the input are controlled by the input design. The application has been developed in user-friendly manner. The forms have been designed in such a way during the processing the cursor is placed in the position where must be entered. The user is also provided with in an option to select an appropriate input from various alternatives related to the field in certain cases.

Validations are required for each data entered. Whenever a user enters an erroneous data, error message is displayed and the user can move on to the subsequent pages after completing all the entries in the current page.

OUTPUT DESIGN

The Output from the computer is required to mainly create an efficient method of communication within the company primarily among the project leader and his team members, in other words, the administrator and the clients. The output of VPN is the system which allows the project leader to manage his clients in terms of creating new clients and assigning new projects to them, maintaining a record of the project validity and providing folder level access to each client on the user side depending on the projects allotted to him. After completion of a project, a new project may be assigned to the client. User authentication procedures are maintained at the initial stages itself. A new user may be created by the administrator himself or a user can himself register as a new user but the task of assigning projects and validating a new user rests with the administrator only.

The application starts running when it is executed for the first time. The server has to be started and then the internet explorer in used as the browser. The project will run on the local area network so the server machine will serve as the administrator while the other connected systems can act as the clients. The developed system is highly user friendly and can be easily understood by anyone using it even for the first time.

**2.2 EXISTING SYSTEM**:

TEXTUAL passwords have been the most widely used authentication method for decades. Comprised of numbers and upper- and lower-case letters, textual passwords are considered strong enough to resist against brute force attacks. However, a strong textual password is hard to memorize and recollect. Therefore, users tend to choose passwords that are either short or from the dictionary, rather than random alphanumeric strings.

**2.3 PROPOSED SYSTEM:**

Various graphical password authentication schemes were developed to address the problems and weaknesses associated with textual passwords. Based on some studies such as those in , humans have a better ability to memorize images with long-term memory(LTM) than verbal representations. Image-based passwords were proved to be easier to recollect in several user studies As a result, users can set up a complex authentication password and are capable of recollecting it after a long time even if the memory is not activated periodically.

The human actions such as choosing bad passwords for

new accounts and inputting passwords in an insecure way for later logins are regarded as the weakest link in the authentication chain [16]. Therefore, an authentication scheme should be designed to overcome these vulnerabilities.

In this paper, we present a secure graphical authentication system named Pass Matrix that protects users from becoming victims of shoulder surfing attacks when inputting passwords in public through the usage of one-time login indicators. A login indicator is randomly generated for each pass-image and will be useless after the session terminates. The login indicator provides better security against shoulder surfing attacks, since users use a dynamic pointer to point out the position of their passwords rather than clicking on the password object directly

**CHAPTER: 3**

**SYSTEM ANALYSIS**

**3.1 THE STUDY OF THE SYSTEM**

* To conduct studies and analyses of an operational and technological nature, and
* To promote the exchange and development of methods and tools for operational analysis as applied to defense problems.

Logical design

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modeling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems design are included. Logical design includes ER Diagrams i.e. Entity Relationship Diagrams

Physical design

The physical design relates to the actual input and output processes of the system. This is laid down in terms of how data is input into a system, how it is verified / authenticated, how it is processed, and how it is displayed as output. In Physical design, following requirements about the system are decided.

1. Input requirement,
2. Output requirements,
3. Storage requirements,
4. Processing Requirements,
5. System control and backup or recovery.

Put another way, the physical portion of systems design can generally be broken down into three sub-tasks:

1. User Interface Design
2. Data Design
3. Process Design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them. Data Design is concerned with how the data is represented and stored within the system. Finally, Process Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the systems design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase.

Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc. It involves a detailed design of a user and a product database structure processor and a control processor. The H/S personal specification is developed for the proposed system.

**3.2** **INPUT & OUTPUT REPRESENTATION**

Input Design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

Objectives

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

* 1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
  2. Select methods for presenting information.
  3. Create document, report, or other formats that contain information produced by the system.

## 3.3 Introduction to Java

## Java Technology

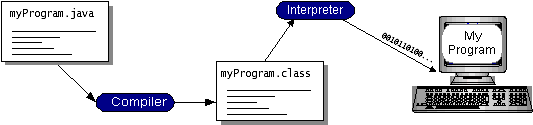
Java technology is both a programming language and a platform.

### The Java Programming Language

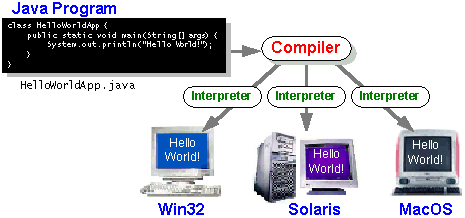
### The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

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

With most programming languages, you 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 you 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.



You 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. You 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.



### The Java Platform

A platform is the hardware or software environment in which a program runs. We’ve already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. 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.

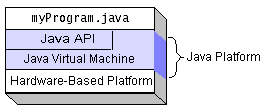
The Java platform has two components:

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

You’ve already been introduced to the Java VM. 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 of related classes and interfaces; these libraries are known as packages. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that’s running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.



Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

## What Can Java Technology Do?

The most common types of programs written in the Java programming language are applets and applications. If you’ve surfed the Web, you’re probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

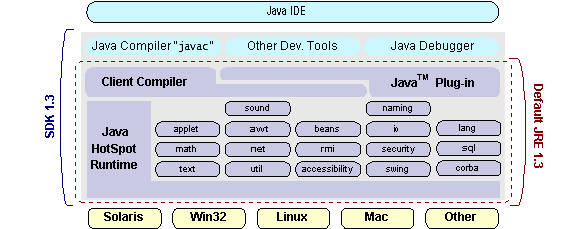
However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a server serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a servlet. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

* **The essentials**: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
* **Applets**: The set of conventions used by applets.
* **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
* **Internationalization**: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
* **Security**: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
* **Software components**: Known as JavaBeansTM, can plug into existing component architectures.
* **Object serialization**: Allows lightweight persistence and communication via Remote Method Invocation (RMI).
* **Java Database Connectivity (JDBCTM)**: Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



## How Will Java Technology Change My Life?

We can’t promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

* **Get started quickly**: Although the Java programming language is a powerful object-oriented language, it’s easy to learn, especially for programmers already familiar with C or C++.
* **Write less code**: Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
* **Write better code**: The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people’s tested code and introduce fewer bugs.
* **Develop programs more quickly**: Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.
* **Avoid platform dependencies with 100% Pure Java**: You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure JavaTM Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
* **Write once, run anywhere**: Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
* **Distribute software more easily**: You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded “on the fly,” without recompiling the entire program.

### ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a de facto standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn’t change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn’t as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

**JDBC**

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

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

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

### JDBC Goals

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

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

1. **SQL Level API**

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC’s complexities from the end user.

1. **SQL Conformance**

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

1. **JDBC must be implemental on top of common database interfaces**   
    The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.
2. **Provide a Java interface that is consistent with the rest of the Java system**

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

1. **Keep it simple**

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

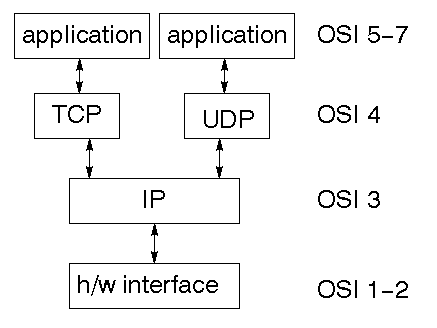
1. **Use strong, static typing wherever possible**

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

## Networking

### TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

### IP datagram’s

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

### UDP

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

### TCP

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

### Internet addresses

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

### Network address

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

### Subnet address

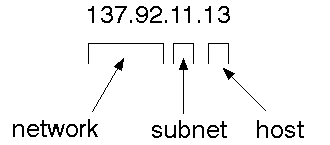
Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

### Host address

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

### 

### Total address



The 32 bit address is usually written as 4 integers separated by dots.

### Port addresses

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

### Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call socket. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

#include <sys/types.h>

#include <sys/socket.h>

int socket(int family, int type, int protocol);

Here "family" will be AF\_INET for IP communications, protocol will be zero, and type will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

**JFree Chart**

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;

A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

## 1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XYPlot class in JFreeChart;

Testing, documenting, testing some more, documenting some more.

## 2. Time Series Chart Interactivity

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

## 3. Dashboards

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFreeChart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

## 4. Property Editors

The property editor mechanism in JFreeChart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

**J2ME (Java 2 Micro edition):-**

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the JavaOne Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

**1. General J2ME architecture**



J2ME uses configurations and profiles to customize the Java Runtime Environment (JRE). As a complete JRE, J2ME is comprised of a configuration, which determines the JVM used, and a profile, which defines the application by adding domain-specific classes. The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. We'll discuss configurations in detail in the The profile defines the application; specifically, it adds domain-specific classes to the J2ME configuration to define certain uses for devices. We'll cover profiles in depth in the The following graphic depicts the relationship between the different virtual machines, configurations, and profiles. It also draws a parallel with the J2SE API and its Java virtual machine. While the J2SE virtual machine is generally referred to as a JVM, the J2ME virtual machines, KVM and CVM, are subsets of JVM. Both KVM and CVM can be thought of as a kind of Java virtual machine -- it's just that they are shrunken versions of the J2SE JVM and are specific to J2ME.

**2. Developing J2ME applications**

Introduction In this section, we will go over some considerations you need to keep in mind when developing applications for smaller devices. We'll take a look at the way the compiler is invoked when using J2SE to compile J2ME applications. Finally, we'll explore packaging and deployment and the role preverification plays in this process.

**3. Design considerations for small devices**

Developing applications for small devices requires you to keep certain strategies in mind during the design phase. It is best to strategically design an application for a small device before you begin coding. Correcting the code because you failed to consider all of the "gotchas" before developing the application can be a painful process. Here are some design strategies to consider:

\* Keep it simple. Remove unnecessary features, possibly making those features a separate, secondary application.

\* Smaller is better. This consideration should be a "no brainer" for all developers. Smaller applications use less memory on the device and require shorter installation times. Consider packaging your Java applications as compressed Java Archive (jar) files.

\* Minimize run-time memory use. To minimize the amount of memory used at run time, use scalar types in place of object types. Also, do not depend on the garbage collector. You should manage the memory efficiently yourself by setting object references to null when you are finished with them. Another way to reduce run-time memory is to use lazy instantiation, only allocating objects on an as-needed basis. Other ways of reducing overall and peak memory use on small devices are to release resources quickly, reuse objects, and avoid exceptions.

**4. Configurations overview**

The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. Currently, two configurations exist for J2ME, though others may be defined in the future:

\* **Connected Limited Device Configuration (CLDC)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the configuration (and the virtual machine) used for developing small J2ME applications. Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC. CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

\* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

**5. J2ME profiles**

**What is a J2ME profile?**

As we mentioned earlier in this tutorial, a profile defines the type of device supported. The Mobile Information Device Profile (MIDP), for example, defines classes for cellular phones. It adds domain-specific classes to the J2ME configuration to define uses for similar devices. Two profiles have been defined for J2ME and are built upon CLDC: KJava and MIDP. Both KJava and MIDP are associated with CLDC and smaller devices. Profiles are built on top of configurations. Because profiles are specific to the size of the device (amount of memory) on which an application runs, certain profiles are associated with certain configurations.

A skeleton profile upon which you can create your own profile, the Foundation Profile, is available for CDC.

**Profile 1: KJava**

KJava is Sun's proprietary profile and contains the KJava API. The KJava profile is built on top of the CLDC configuration. The KJava virtual machine, KVM, accepts the same byte codes and class file format as the classic J2SE virtual machine. KJava contains a Sun-specific API that runs on the Palm OS. The KJava API has a great deal in common with the J2SE Abstract Windowing Toolkit (AWT). However, because it is not a standard J2ME package, its main package is com.sun.kjava. We'll learn more about the KJava API later in this tutorial when we develop some sample applications.

**Profile 2: MIDP**

MIDP is geared toward mobile devices such as cellular phones and pagers. The MIDP, like KJava, is built upon CLDC and provides a standard run-time environment that allows new applications and services to be deployed dynamically on end user devices. MIDP is a common, industry-standard profile for mobile devices that is not dependent on a specific vendor. It is a complete and supported foundation for mobile application

development. MIDP contains the following packages, the first three of which are core CLDC packages, plus three MIDP-specific packages.

\* java.lang

\* java.io

\* java.util

\* javax.microedition.io

\* javax.microedition.lcdui

\* javax.microedition.midlet

\* javax.microedition.rms

**3.4** **SYSTEM REQUIREMENTS**

**3.4.1 HARDWARE REQUIREMENTS:**

* System : Pentium IV 2.4 GHz.
* Hard Disk : 200 GB.
* Floppy Drive : 1.44 Mb.
* Monitor : 15 VGA Colour.
* Mouse : Logitech.
* Ram : 1 GB.

**3.4.2 SOFTWARE REQUIREMENTS:**

* Operating system : Windows XP/7/10.
* Coding Language : Java
* Tool : Netbeans
* Database : MYSQL

.

**CHAPTER: 4**

**SYSTEM DESIGN**

**4.1 SYSTEM ARCHITECTURE**

Architecture Flow:

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

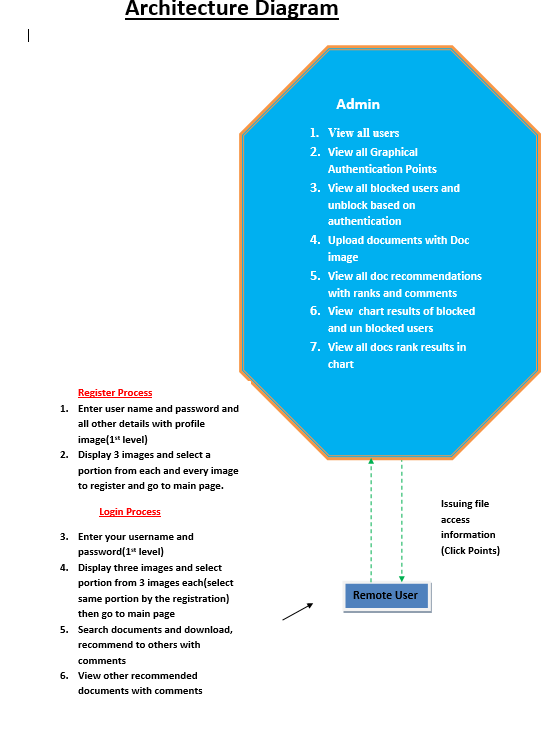


Figure 3.1: Architecture diagram

**FLOW CHART**

Start

**Admin:**

Admin registration

Admin name and password wrong

No

Yes

Login

View all users, View all Graphical Authentication Points

iew all users,View all Graphical Authentication Points

Vie View all blocked users and unblock based on authentication

w all blocked users and unblock based on authentication

Logout

Logout

Uploa Upload documents with Doc image,View all doc recommendations with ranks and comments

d documents with Doc image,View all doc recommendations with ranks and comments

View chart results of blocked and un bl View chart results of blocked and un blocked users,View all docs rank results in chart

ocked users,View all docs rank results in chart

**User:**

user registration

User name and password wrong

No

Yes

Login

Enter user name and password and all other details with profile image

Display 3 images and select a portion from each and every image to register and go to main page

Logout

Enter your username and password,Display three images and select portion from 3 images each then go to main page

Search documents and download, recommend to others with comments

View other recommended documents with comments

3-Tier Architecture:

The three-tier software architecture (a three layer architecture) emerged in the 1990s to overcome the limitations of the two-tier architecture. The third tier (middle tier server) is between the user interface (client) and the data management (server) components. This middle tier provides process management where business logic and rules are executed and can accommodate hundreds of users (as compared to only 100 users with the two tier architecture) by providing functions such as queuing, application execution, and database staging.

The three tier architecture is used when an effective distributed client/server design is needed that provides (when compared to the two tier) increased performance, flexibility, maintainability, reusability, and scalability, while hiding the complexity of distributed processing from the user. These characteristics have made three layer architectures a popular choice for Internet applications and net-centric information systems.

**Advantages of Three-Tier:**

* Separates functionality from presentation.
* Clear separation – better understanding.
* Changes limited to well define components.
* Can be running on WWW.
* Effective network performance.

**SYSTEM DESIGN**

System Design Introduction: The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

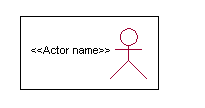
**4.2 UML DIAGRAMS**

Global Use Case Diagrams:

Identification of actors:

Actor: Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.

Graphical representation:



<<Actor name>>

An actor is someone or something that:

Interacts with or uses the system.

* Provides input to and receives information from the system.
* Is external to the system and has no control over the use cases.

Actors are discovered by examining:

* Who directly uses the system?
* Who is responsible for maintaining the system?
* External hardware used by the system.
* Other systems that need to interact with the system.

Questions to identify actors:

* + Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
  + Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.
  + Which external hardware or systems (if any) use the system to perform tasks?
  + What problems does this application solve (that is, for whom)?
  + And, finally, how do users use the system (use case)? What are they doing with the system?

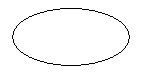
The actors identified in this system are:

1. System Administrator
2. Customer
3. Customer Care

Identification of usecases:

Usecase: A use case can be described as a specific way of using the system from a user’s (actor’s) perspective.

Graphical representation:



A more detailed description might characterize a use case as:

* Pattern of behavior the system exhibits
* A sequence of related transactions performed by an actor and the system
* Delivering something of value to the actor

Use cases provide a means to:

* capture system requirements
* communicate with the end users and domain experts
* test the system

Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

Guide lines for identifying use cases:

* For each actor, find the tasks and functions that the actor should be able to perform or that the system needs the actor to perform. The use case should represent a course of events that leads to clear goal
* Name the use cases.
* Describe the use cases briefly by applying terms with which the user is familiar.

This makes the description less ambiguous

Questions to identify use cases: What are the tasks of each actor?

* Will any actor create, store, change, remove or read information in the system?
* What use case will store, change, remove or read this information?
* Will any actor need to inform the system about sudden external changes?
* Does any actor need to inform about certain occurrences in the system?
* What usecases will support and maintains the system?

1.2 **Flow of Events**

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

* When and how the use case starts and ends
* Use case/actor interactions
* Data needed by the use case
* Normal sequence of events for the use case
* Alternate or exceptional flows

**4.2.1 Construction of Usecase diagrams:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

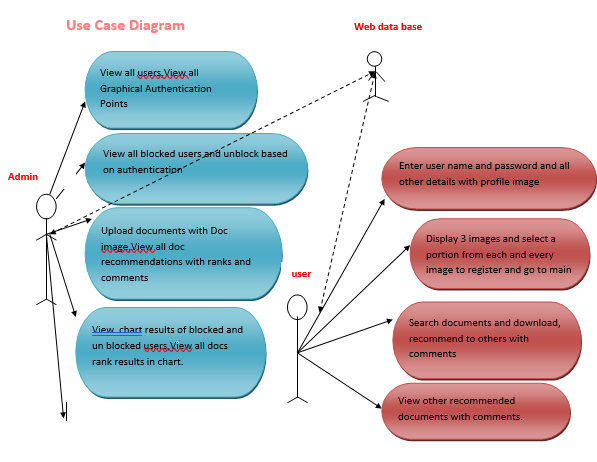


Figure 4.2.1 Use Case Diagram

**4.2.2 SEQUENCE DIAGRAMS:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

User

Admin

Web data base

Register and Login

Display 3 images and select a portion from each and every image to register and go to main page

View all users,View all Graphical Authentication Points

Search documents and download

View all blocked users and unblock based on authentication

Upload documents with Doc image,View all doc recommendations with ranks and comments

View other recommended documents with comments

View chart results of blocked and un blocked users

View all docs rank results in chart.

Figure 4.2.2: Sequence diagram

**Data Flow Diagram:**

**Data Flow Diagram**

**Level -0**

**Web Data base**

**Admin**

**user**

**4.2.3.** **CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

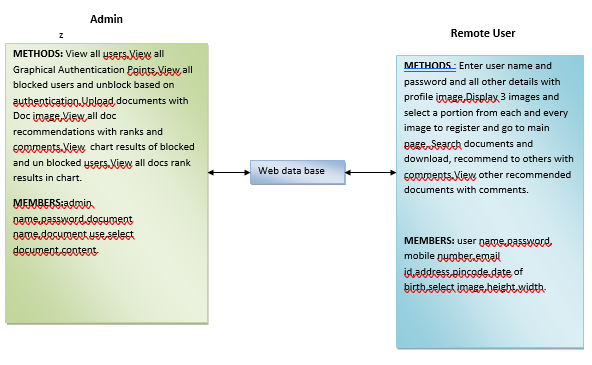
****

Figure 4.2.3: Class Diagram

**CHAPTER: 5**

**TESTING &VALIDATION**

**INTRODUCTION:**

Testing is the debugging program is one of the most critical aspects of the computer programming triggers, without programming that works, the system would never produce an output of which it was designed. Testing is best performed when user development is asked to assist in identifying all errors and bugs. The sample data are used for testing. It is not quantity but quality of the data used the matters of testing. Testing is aimed at ensuring that the system was accurately an efficiently before live operation commands.

Testing objectives:

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error.

1. A successful test is one that uncovers an as yet undiscovered error.
2. A good test case is one that has probability of finding an error, if it exists.
3. The test is inadequate to detect possibly present errors.
4. The software more or less confirms to the quality and reliable standards.

**5.2. Levels of Testing:**

Code testing:

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

Specification Testing:

Executing this specification starting what the program should do and how it should performed under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

Unit testing:

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system.

Each Module can be tested using the following two Strategies:

1. Black Box Testing
2. White Box Testing

**BLACK BOX TESTING**

What is Black Box Testing?

Black box testing is a software testing techniques in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.



The above Black Box can be any software system you want to test. For example : an operating system like Windows, a website like Google ,a database like Oracle or even your own custom application. Under Black Box Testing , you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

Black box testing - Steps

Here are the generic steps followed to carry out any type of Black Box Testing.

* Initially requirements and specifications of the system are examined.
* Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
* Tester determines expected outputs for all those inputs.
* Software tester constructs test cases with the selected inputs.
* The test cases are executed.
* Software tester compares the actual outputs with the expected outputs.
* Defects if any are fixed and re-tested.

Types of Black Box Testing

There are many types of Black Box Testing but following are the prominent ones -

* Functional testing – This black box testing type is related to functional requirements of a system; it is done by software testers.
* Non-functional testing – This type of black box testing is not related to testing of a specific functionality, but non-functional requirements  such as performance, scalability, usability.
* Regression testing – Regression testing is done  after code fixes , upgrades or any other system maintenance to check the new code has not affected the existing code.

**WHITE BOX TESTING**

White Box Testing is the testing of a software solution's internal coding and infrastructure.It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability.White box testing is also known as clear, open, structural, and glass box testing.

It is one of two parts of the "box testing" approach of software testing. Its counter-part, blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing. The term "whitebox" was used because of the see-through box concept. The clear box or whitebox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested

## What do you verify in White Box Testing ?

White box testing involves the testing of the software code for the following:

* Internal security holes
* Broken or poorly structured paths in the coding processes
* The flow of specific inputs through the code
* Expected output
* The functionality of conditional loops
* Testing of each statement, object and function on an individual basis

 The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

## How do you perform White Box Testing?

  To give you a simplified explanation of white box testing, we have divided it into **two basic steps**. This is what testers do when testing an application using the white box testing technique:

**STEP 1) UNDERSTAND THE SOURCE CODE**

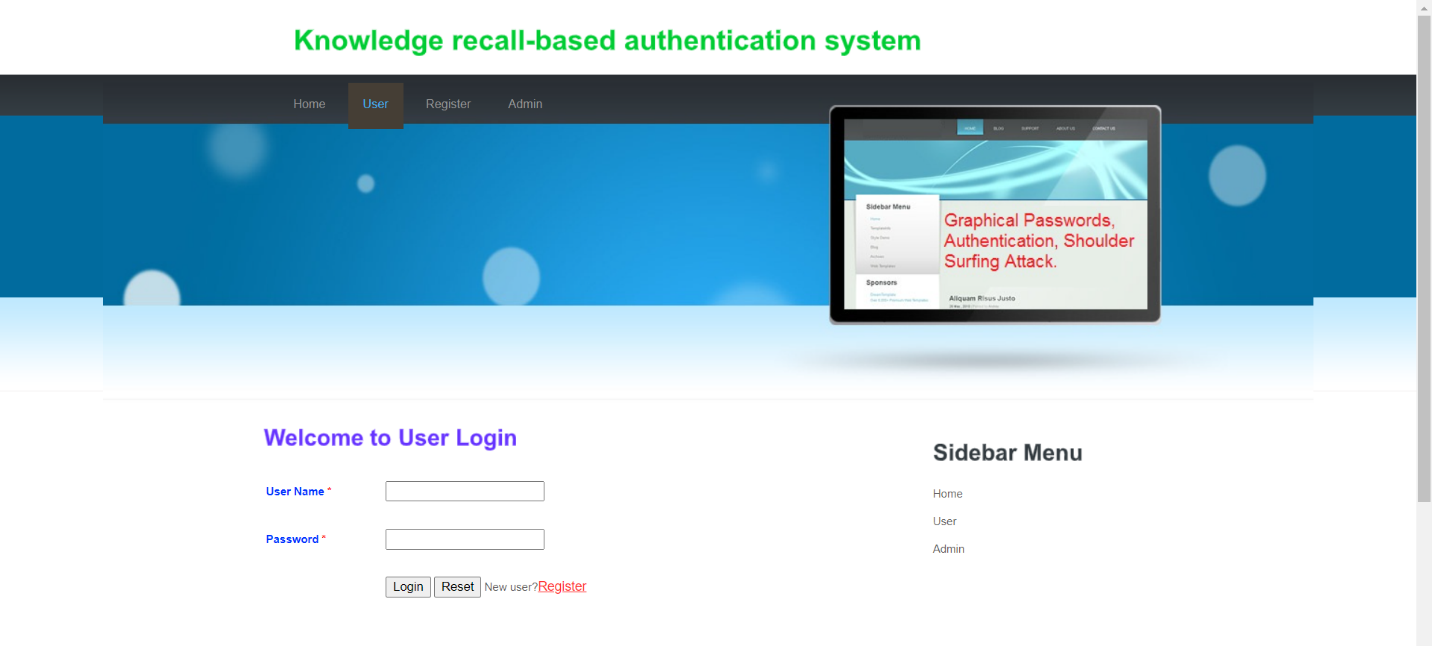
The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

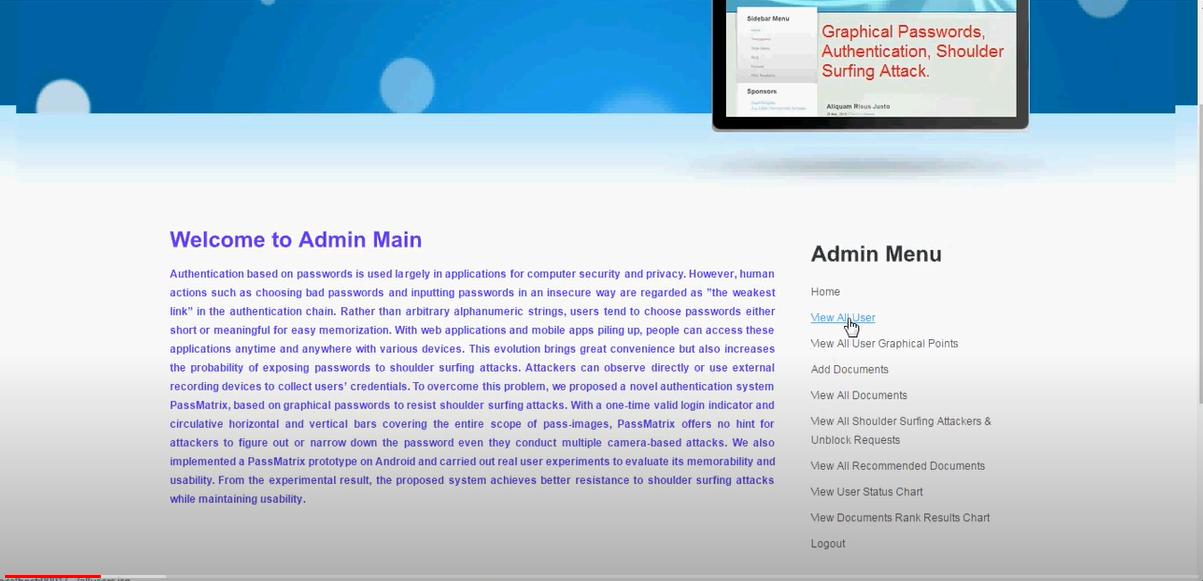
**Step 2) CREATE TEST CASES AND EXECUTE**

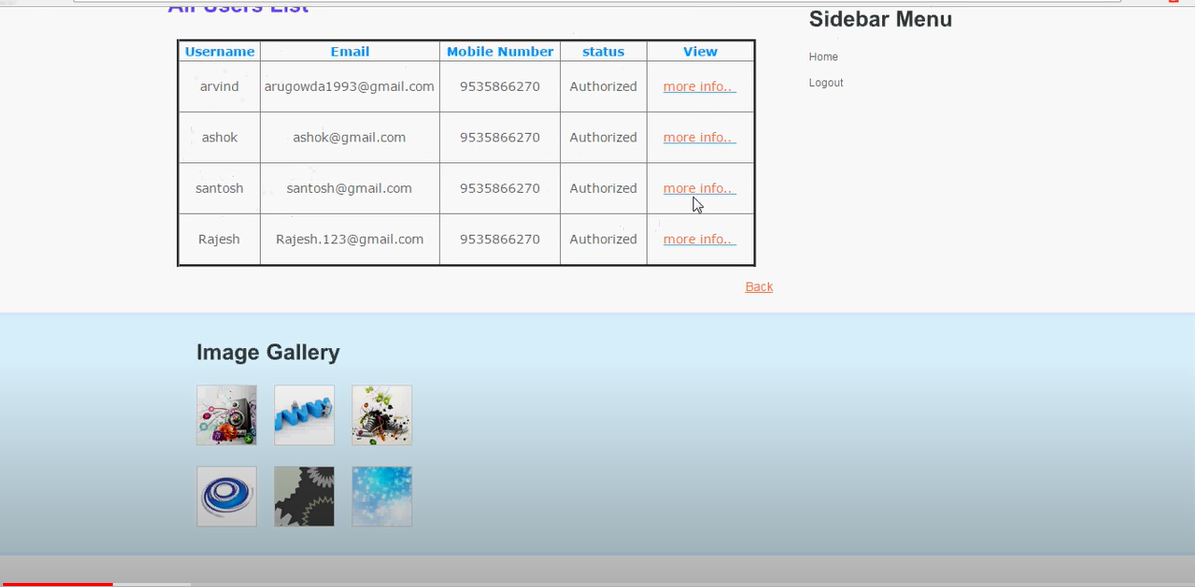
The second basic step to white box testing involves testing the application’s source code for proper flow and structure. One way is by writing more code to test the application’s source code. The tester will develop little tests for each process or series of processes in the application. This  method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include manual testing, trial and error testing and the use of testing tools as we will explain further on in this article.

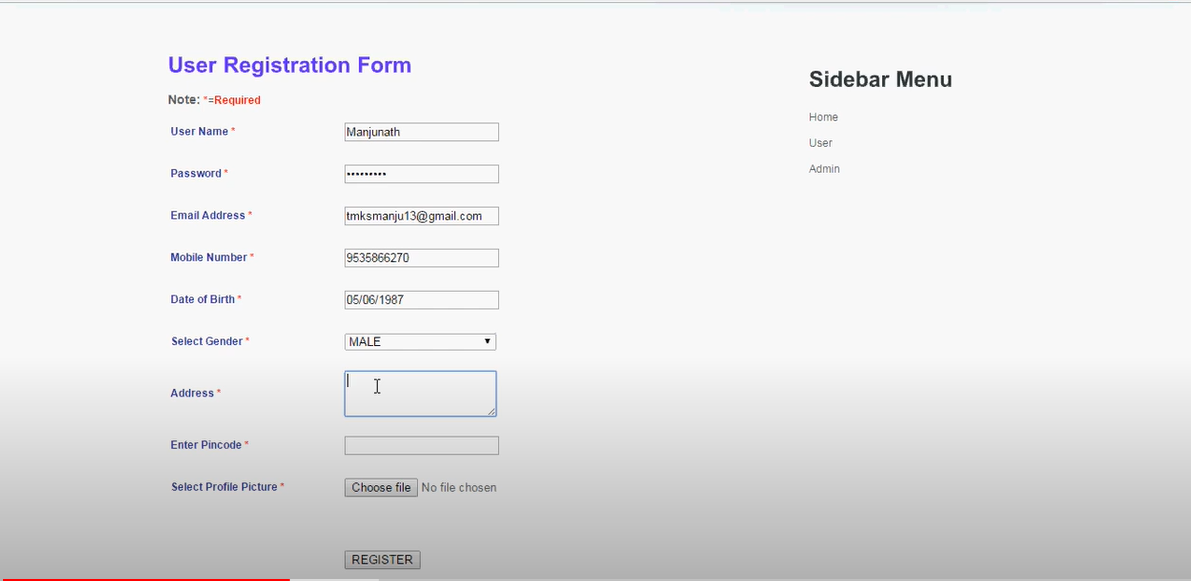
**CHAPTER: 6**

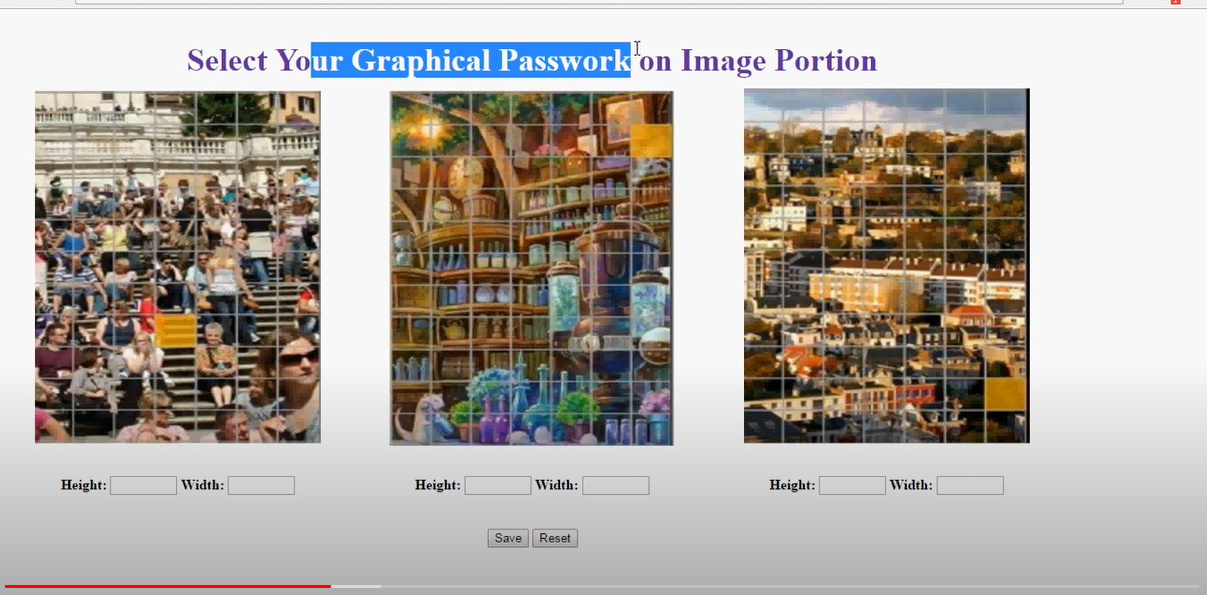
**OUTPUT SCREEN SHOTS**

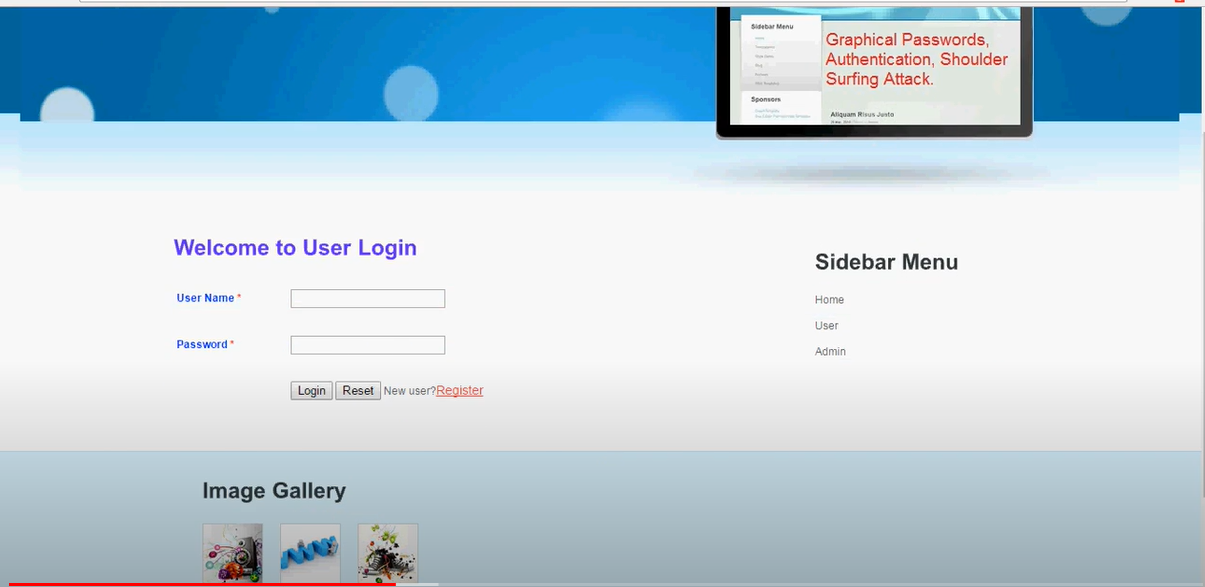


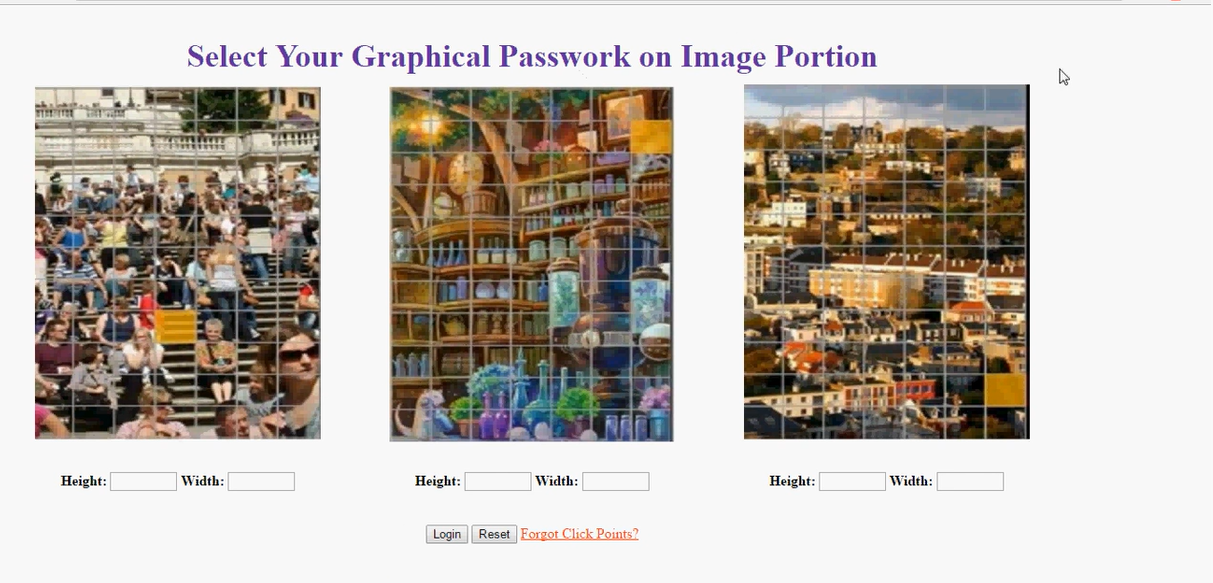


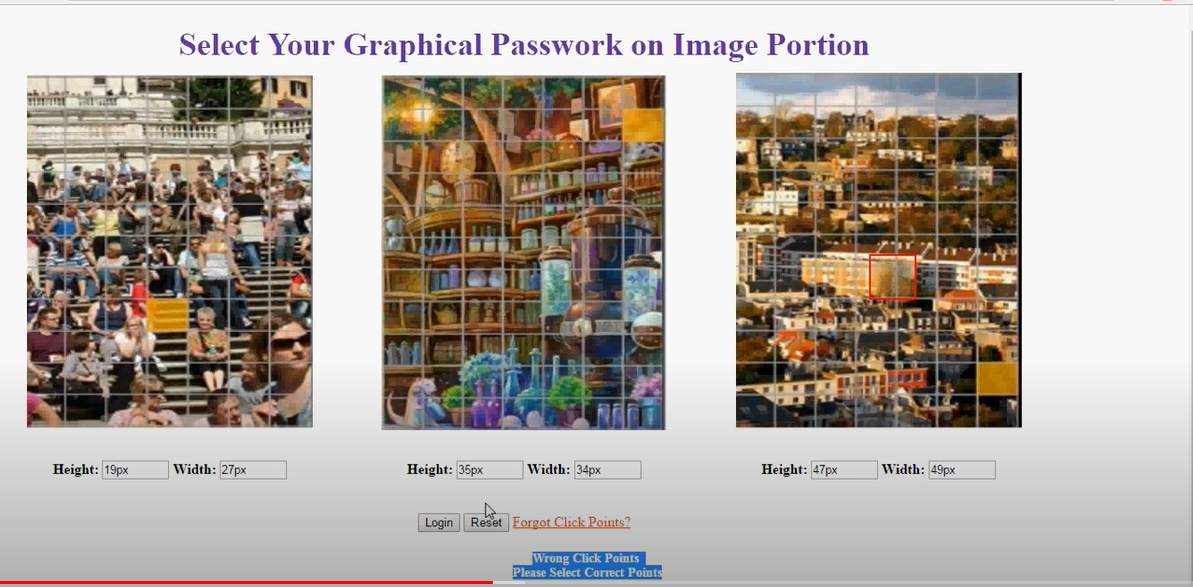














**CHAPTER: 7**

**CONCLUSION**

With the increasing trend of web services and apps, users are able to access these applications anytime and anywhere with various devices. In order to protect users’ digital property, authentication is required every time they try to access their personal account and data. However, conducting the

authentication process in public might result in potential shoulder surfing attacks. Even a complicated password can be cracked easily through shoulder surfing. Using traditional textual passwords or PIN method, users need to

type their passwords to authenticate themselves and thus these passwords can be revealed easily if someone peeks over shoulder or uses video recording devices such as cell phones.

To overcome this problem, we proposed a shoulder surfing resistant authentication system based on graphical passwords, named Pass Matrix. Using a one-time login indicator per image, users can point out the location of their pass-square without directly clicking or touching it,

which is an action vulnerable to shoulder surfing attacks. Because of the design of the horizontal and vertical bars that cover the entire pass-image, it offers no clue for attackers to narrow down the password space even if they have more than one login records of that account. Furthermore,

we implemented a Pass Matrix prototype on Android and carried out user experiments to evaluate the memorability and usability. The experimental result showed that users can log into the system with an average of 1:64 tries (Median=1), and the Total Accuracy of all login trials is 93:33% even two

weeks after registration. The total time consumed to log into Pass Matrix with an average of 3:2 pass-images is between 31:31 and 37:11 seconds and is considered acceptable by 83:33% of participants in our user study.

Based on the experimental results and survey data, Pass Matrix is a novel and easy-to-use graphical password authentication system, which can effectively alleviate shoulder-surfing attacks. In addition, Pass Matrix can be applied To any authentication scenario and device with simple input and output capabilities. The survey data in the user study also showed that Pass Matrix is practical in the real world.

The Application is one of the useful application in the current situation. This is the easy way to communicate with the admin. Employee expense claim workflow became an early candidate for enablement as it could eliminate handling of supporting expense bills and instead use the camera of Smartphone to capture the bill.

**FUTURESCOPE:**

Picture passwords are an alternative to textual alphanumeric password. Most of the existing authentication system has certain drawbacks for that reason graphical passwords are most preferable authentication system where users click on images to authenticate themselves. As authentication techniques generate passwords but they have to face attacks like dictionary attacks, brute force attacks, shoulder surfing. An important usability goal of an authentication system is to support users for selecting the better password.

User creates memorable password which is easy to guess by an attacker and strong system assigned passwords are difficult to memorize. So researchers of modern days have gone through different alternative methods and concluded that graphical passwords are most preferable authentication system. By implementing encryption algorithms and hashing for storing and retrieving pictures and points, one can achieve more security. The proposed system combines the existing cued click point technique with the persuasive feature to influence user choice, encouraging user to select more random click point which is difficult to guess. Picture password is still immature more research is required in this field.

**Code**

**User click points**

<title>add click points</title>

<%@ include file="connect.jsp" %>

<%

String uname = request.getParameter("uname");

String x = request.getParameter("x");

String y = request.getParameter("y");

String x2 = request.getParameter("x2");

String y2 = request.getParameter("y2");

String x3 = request.getParameter("x3");

String y3 = request.getParameter("y3");

try {

if((x.equals(""))&&(y.equals("")))

{

out.print("Please select X and Y Points on Image 1 ");

%>

<br /><br><a href="ur\_clickpoints.jsp?uname=<%=uname%>">Click Here</a>

<%

}

else if((x2.equals(""))&&(y2.equals("")))

{

out.print("Please select X and Y Points on Image 2 ");

%>

<br /><br><a href="ur\_clickpoints.jsp?uname=<%=uname%>">Click Here</a>

<%

}

else if((x3.equals(""))&&(y3.equals("")))

{

out.print("Please select X and Y Points on Image 3 ");

%>

<br /><br><a href="ur\_clickpoints.jsp?uname=<%=uname%>">Click Here</a>

<%

}

else

{

String sql = "insert into clickpoints(username,point1\_x,point1\_y,point2\_x,point2\_y,point3\_x,point3\_y)values('"+uname+"','"+x+"','"+y+"','"+x2+"','"+y2+"','"+x3+"','"+y3+"') ";

Statement stmt = connection.createStatement();

int x1=stmt.executeUpdate(sql);

if(x1>0)

{

String msg="User Registered Successfully";

application.setAttribute("msg",msg);

response.sendRedirect("u\_register.jsp");

}

}

}

catch (Exception ex) {

System.out.println(ex);

}

%>

}

**User change points**

<html>

<head>

<STYLE>

#rubberBand {

position: absolute;

visibility: hidden;

width: 0px; height: 0px;

border: 2px solid red;

}

.style1 {

font-size: 36px;

color: #663399;

}

.style2 {

font-size: 14px;

color: #009933;

}

.style3 {color: #FF3300}

</STYLE>

<title>chage authentication points</title></HEAD>

<BODY>

<%String user=(String)application.getAttribute("user");

%>

<form name="portion" method="post" action="u\_savechanges.jsp?uname=<%=user%>">

<div align="center">

<%

String msg=(String)application.getAttribute("msg");

if(msg!=null)

{

%>

<h3><span class="style2"> <%=msg%> </span></h3>

<%

}application.removeAttribute("msg");

%></div>

<table width="1192">

<tr>

<th height="58" colspan="3"><span class="style1">Select Your Graphical Passwork on Image Portion</span></th>

</tr>

<tr>

<td align="center" valign="middle"><img name="myImage" id="myImage" src="Image1.jpg" width="320" height="400"/>

<DIV ID="rubberBand"></DIV>

<div align="center">

<SCRIPT>

var IMG;

function startRubber (evt) {

if (document.all) {

var r = document.all.rubberBand;

r.style.width = 0;

r.style.height = 0;

r.style.pixelLeft = event.x;

r.style.pixelTop = event.y;

r.style.visibility = 'visible';

IMG.ondragstart = cancelDragDrop; // otherwise IE will try to drag the image

}

else if (document.getElementById) {

// firefox

evt.preventDefault();

var r = document.getElementById('rubberBand');

r.style.width = 0;

r.style.height = 0;

r.style.left = evt.clientX + 'px';

r.style.top = evt.clientY + 'px';

r.style.visibility = 'visible';

r.onmouseup = stopRubber;

}

IMG.onmousemove = moveRubber;

}

function moveRubber (evt) {

if (document.all) { // IE

var r = document.all.rubberBand;

r.style.width = event.x - r.style.pixelLeft;

r.style.height = event.y - r.style.pixelTop;

}

else if (document.getElementById) { // firefox

var r = document.getElementById('rubberBand');

r.style.width = evt.clientX - parseInt(r.style.left);

r.style.height = evt.clientY - parseInt(r.style.top);

document.portion.x.value=r.style.height;

document.portion.y.value=r.style.width;

}

return false; // otherwise IE won't fire mouseup :/

}

function stopRubber (evt) {

IMG.onmousemove = null;

}

function cancelDragDrop()

{

window.event.returnValue = false;

}

IMG = document.getElementById('myImage');

IMG.onmousedown = startRubber;

IMG.onmouseup = stopRubber;

</SCRIPT>

</div>

<br>

<p align="center"><strong>Height: </strong>

<input type="text" name="x" size="6">

<strong>Width:</strong>

<input type="text" name="y" size="6"></p>

</td>

<td align="center" valign="bottom"><img name="myImage2" id="myImage2" src="Image2.jpg" width="320" height="400"/>

<DIV ID="rubberBand"></DIV>

<div align="center">

<SCRIPT>

var IMG2;

function startRubber2 (evt2) {

if (document.all) {

var r2 = document.all.rubberBand;

r2.style.width = 0;

r2.style.height = 0;

r2.style.pixelLeft = event.x;

r2.style.pixelTop = event.y;

r2.style.visibility = 'visible';

IMG2.ondragstart = cancelDragDrop2; // otherwise IE will try to drag the image

}

else if (document.getElementById) {

// firefox

evt2.preventDefault();

var r2 = document.getElementById('rubberBand');

r2.style.width = 0;

r2.style.height = 0;

r2.style.left = evt2.clientX + 'px';

r2.style.top = evt2.clientY + 'px';

r2.style.visibility = 'visible';

r2.onmouseup = stopRubber2;

}

IMG2.onmousemove = moveRubber2;

}

function moveRubber2 (evt2) {

if (document.all) { // IE

var r2 = document.all.rubberBand;

r2.style.width = event.x - r2.style.pixelLeft;

r2.style.height = event.y - r2.style.pixelTop;

}

else if (document.getElementById) { // firefox

var r2 = document.getElementById('rubberBand');

r2.style.width = evt2.clientX - parseInt(r2.style.left);

r2.style.height = evt2.clientY - parseInt(r2.style.top);

document.portion.x2.value=r2.style.height;

document.portion.y2.value=r2.style.width;

}

return false; // otherwise IE won't fire mouseup :/

}

function stopRubber2 (evt2) {

IMG2.onmousemove = null;

}

function cancelDragDrop2()

{

window.event.returnValue = false;

}

IMG2 = document.getElementById('myImage2');

IMG2.onmousedown = startRubber2;

IMG2.onmouseup = stopRubber2;

</SCRIPT>

</div>

<br>

<p align="center"><strong>Height: </strong>

<input type="text" name="x2" size="6">

<strong> Width:</strong>

<input type="text" name="y2" size="6"></p>

</td>

<td align="center" valign="middle"><img name="myImage3" id="myImage3" src="Image3.jpg" width="320" height="400"/>

<DIV ID="rubberBand"></DIV>

<div align="center">

<SCRIPT>

var IMG3;

function startRubber3 (evt3) {

if (document.all) {

var r3 = document.all.rubberBand;

r3.style.width = 0;

r3.style.height = 0;

r3.style.pixelLeft = event.x;

r3.style.pixelTop = event.y;

r3.style.visibility = 'visible';

IMG3.ondragstart = cancelDragDrop3; // otherwise IE will try to drag the image

}

else if (document.getElementById) {

// firefox

evt3.preventDefault();

var r3 = document.getElementById('rubberBand');

r3.style.width = 0;

r3.style.height = 0;

r3.style.left = evt3.clientX + 'px';

r3.style.top = evt3.clientY + 'px';

r3.style.visibility = 'visible';

r3.onmouseup = stopRubber3;

}

IMG3.onmousemove = moveRubber3;

}

function moveRubber3(evt3) {

if (document.all) { // IE

var r3 = document.all.rubberBand;

r3.style.width = event.x - r3.style.pixelLeft;

r3.style.height = event.y - r3.style.pixelTop;

}

else if (document.getElementById) { // firefox

var r3 = document.getElementById('rubberBand');

r3.style.width = evt3.clientX - parseInt(r3.style.left);

r3.style.height = evt3.clientY - parseInt(r3.style.top);

document.portion.x3.value=r3.style.height;

document.portion.y3.value=r3.style.width;

}

return false; // otherwise IE won't fire mouseup :/

}

function stopRubber3 (evt3) {

IMG3.onmousemove = null;

}

function cancelDragDrop3()

{

window.event.returnValue = false;

}

IMG3 = document.getElementById('myImage3');

IMG3.onmousedown = startRubber3;

IMG3.onmouseup = stopRubber3;

</SCRIPT>

</div>

<br>

<p align="center"><strong>Height: </strong>

<input type="text" name="x3" size="6">

<strong> Width:</strong>

<input type="text" name="y3" size="6"></p>

</td>

</tr><br>

<tr><td colspan="3"><br>

<p align="center">

<label>

<input type="submit" name="Submit" value="Save">

</label>

<label>

<input type="reset" name="Submit2" value="Reset">

</label>

</td></tr>

<tr>

<td colspan="3"> <div align="center">

<p>&nbsp;</p>

<p><a href="usermain.jsp" class="style3">Back</a>

</p>

</p>

</div></td>

</tr>

</table>

</form>

</Body>

<html>

**All click points**

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>all unblock requests list</title>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<link href="style.css" rel="stylesheet" type="text/css" />

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

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

<script type="text/javascript" src="js/cuf\_run.js"></script>

<style type="text/css">

<!--

.style2 {font-size: 36}

.style3 {padding:0; margin:0; width:100%; line-height:0; clear: both;}

.style4 {

font-size: 30px;

color: #00CC33;

}

.style7 {color: #6633FF; }

.style11 {

font-size: 14px;

color: #FF6633;

}

.style12 {color: #FF6633}

.style16 {color: #006666}

-->

</style>

</head>

<body>

<div class="main style2">

<div class="header">

<div class="header\_resize">

<div class="logo">

<h2 class="style4">A Shoulder Surfing Resistant Graphical Authentication System</h2>

</div>

<div class="style3"></div>

</div>

</div>

<div class="hbg">

<div class="hbg\_resize">

<div class="menu\_nav">

<ul>

<%String admin=(String)application.getAttribute("admin"); %>

<li><a href="adminmain.jsp">Home</a></li>

<li><a href="#"><%=admin%></a></li>

<li><a href="adminlogin.jsp">Logout</a></li>

</ul>

</div>

<div class="style3"></div>

</div>

</div>

<div class="content">

<div class="content\_resize">

<div class="mainbar">

<!-- <div class="article\_vert"><img src="images/img1.jpg" width="287" height="97" alt="" />

<p>&nbsp;</p>

</div>

<div class="article\_vert"><img src="images/img2.jpg" width="287" height="97" alt="" />

<p>&nbsp;</p>

</div>-->

<h2 class="style7">All Graphical Authentication Points </h2>

<div class="area">

<table width="675" border="3" align="left" cellpadding="0" cellspacing="0" style="border-collapse: collapse; margin:10px 10px 10px 10px; font-family:Verdana, Arial, Helvetica, sans-serif; font-size:14px;">

<tr>

<th width="90" height="40"><span class="style16">User Image</span></th>

<th width="75" height="40"><span class="style16">Username</span></th>

<th width="75"><span class="style16">Point1 X</span></th>

<th width="75"><span class="style16">Point1 Y </span></th>

<th width="75"><span class="style16">Point2 X</span></th>

<th width="75"><span class="style16">Point2 Y</span></th>

<th width="75"><span class="style16">Point3 X</span></th>

<th width="75"><span class="style16">Point3 Y</span></th>

</tr>

<%@ include file="connect.jsp" %>

<%

String s1,s2,s3,s4,s5,s6,s7,s8,s9;

int i=0;

try

{

String query="select \* from clickpoints";

Statement st=connection.createStatement();

ResultSet rs=st.executeQuery(query);

while ( rs.next() )

{

s1=rs.getString(2);

s2=rs.getString(3);

s3=rs.getString(4);

s4=rs.getString(5);

s5=rs.getString(6);

s6=rs.getString(7);

s7=rs.getString(8);

String query1="select id from user where username='"+s1+"'";

Statement st1=connection.createStatement();

ResultSet rs1=st1.executeQuery(query1);

if(rs1.next())

{

i=rs1.getInt(1);

%>

<tr>

<td><input name="image" type="image" src="profile\_pic.jsp?id=<%=i%>&type=<%="user"%>" style="width:70px; height:70px;"/></td>

<td width="77" height="72" align="center" valign="middle" ><%out.println(s1);%></td>

<td align="center" valign="middle"><%out.println(s2);%></td>

<td width="71" height="72" align="center" valign="middle"><%out.println(s3);%></td>

<td width="71" height="72" align="center" valign="middle"><%out.println(s4);%></td>

<td align="center"><%out.println(s5);%></td>

<td width="90" height="72" align="center" valign="middle"><%out.println(s6);%></td>

<td width="90" height="72" align="center" valign="middle"><%out.println(s7);%></td>

</tr>

<%

}

}

connection.close();

}

catch(Exception e)

{

out.println(e);

}

%>

</table>

</div>

<div class="style3"> <p align="right" class="style11"><a href="adminmain.jsp" class="style12">Back</a></p></div>

<!--<div class="article\_vert"><img src="images/img3.jpg" width="287" height="97" alt="" />

<p>&nbsp;</p>

</div>

<div class="article\_vert"><img src="images/img4.jpg" width="287" height="97" alt="" />

<p>&nbsp;</p>

</div>-->

<div class="style3"></div>

</div>

<div class="sidebar">

<div class="gadget">

<h2 class="star">Sidebar Menu</h2>

<ul class="sb\_menu">

<li><a href="adminmain.jsp">Home</a></li>

<li><a href="adminlogin.jsp">Logout</a></li>

</ul>

</div>

<div class="gadget">

<h2 class="star">&nbsp;</h2>

</div>

</div>

<div class="style3"></div>

</div>

</div>

<div class="fbg">

<div class="fbg\_resize">

<div class="col c1">

<h2>Image Gallery</h2>

<a href="#"><img src="images/pix1.jpg" width="66" height="66" alt="" class="ad" /></a> <a href="#"><img src="images/pix2.jpg" width="66" height="66" alt="" class="ad" /></a> <a href="#"><img src="images/pix3.jpg" width="66" height="66" alt="" class="ad" /></a> <a href="#"><img src="images/pix4.jpg" width="66" height="66" alt="" class="ad" /></a> <a href="#"><img src="images/pix5.jpg" width="66" height="66" alt="" class="ad" /></a> <a href="#"><img src="images/pix6.jpg" width="66" height="66" alt="" class="ad" /></a>

<div class="style3"></div>

</div>

<div class="col c2">

<h2>&nbsp;</h2>

</div>

<div class="col c3">

<h2>&nbsp;</h2>

</div>

<div class="style3"></div>

</div>

</div>

<div class="footer">

<div class="footer\_resize">

<div class="style3"></div>

</div>

</div>

</div>

<div align=center></div>

</body>

</html>

**User login click points**

<html>

<head>

<STYLE>

#rubberBand {

position: absolute;

visibility: hidden;

width: 0px; height: 0px;

border: 2px solid red;

}

.style1 {

font-size: 36px;

color: #009966;

}

.style2 { font-size: 36px;

color: #663399;

}

</STYLE>

<title>select click points</title></HEAD>

<BODY>

<%String uname=request.getParameter("uname");

%>

<form name="portion" method="post" action="u\_addclickpoints.jsp?uname=<%=uname%>">

<table width="1192">

<tr>

<th height="58" colspan="3"><span class="style2">Select Your Graphical Passwork on Image Portion</span></th>

</tr>

<tr>

<td align="center" valign="middle"><img name="myImage" id="myImage" src="Image1.jpg" width="320" height="400"/>

<DIV ID="rubberBand"></DIV>

<div align="center">

<SCRIPT>

var IMG;

function startRubber (evt) {

if (document.all) {

var r = document.all.rubberBand;

r.style.width = 0;

r.style.height = 0;

r.style.pixelLeft = event.x;

r.style.pixelTop = event.y;

r.style.visibility = 'visible';

IMG.ondragstart = cancelDragDrop; // otherwise IE will try to drag the image

}

else if (document.getElementById) {

// firefox

evt.preventDefault();

var r = document.getElementById('rubberBand');

r.style.width = 0;

r.style.height = 0;

r.style.left = evt.clientX + 'px';

r.style.top = evt.clientY + 'px';

r.style.visibility = 'visible';

r.onmouseup = stopRubber;

}

IMG.onmousemove = moveRubber;

}

function moveRubber (evt) {

if (document.all) { // IE

var r = document.all.rubberBand;

r.style.width = event.x - r.style.pixelLeft;

r.style.height = event.y - r.style.pixelTop;

}

else if (document.getElementById) { // firefox

var r = document.getElementById('rubberBand');

r.style.width = evt.clientX - parseInt(r.style.left);

r.style.height = evt.clientY - parseInt(r.style.top);

document.portion.x.value=r.style.height;

document.portion.y.value=r.style.width;

}

return false; // otherwise IE won't fire mouseup :/

}

function stopRubber (evt) {

IMG.onmousemove = null;

}

function cancelDragDrop()

{

window.event.returnValue = false;

}

IMG = document.getElementById('myImage');

IMG.onmousedown = startRubber;

IMG.onmouseup = stopRubber;

</SCRIPT>

</div>

<br>

<p align="center"><strong>Height: </strong>

<input type="text" name="x" size="6">

<strong>Width:</strong>

<input type="text" name="y" size="6"></p>

</td>

<td align="center" valign="bottom"><img name="myImage2" id="myImage2" src="Image2.jpg" width="320" height="400"/>

<DIV ID="rubberBand"></DIV>

<div align="center">

<SCRIPT>

var IMG2;

function startRubber2 (evt2) {

if (document.all) {

var r2 = document.all.rubberBand;

r2.style.width = 0;

r2.style.height = 0;

r2.style.pixelLeft = event.x;

r2.style.pixelTop = event.y;

r2.style.visibility = 'visible';

IMG2.ondragstart = cancelDragDrop2; // otherwise IE will try to drag the image

}

else if (document.getElementById) {

// firefox

evt2.preventDefault();

var r2 = document.getElementById('rubberBand');

r2.style.width = 0;

r2.style.height = 0;

r2.style.left = evt2.clientX + 'px';

r2.style.top = evt2.clientY + 'px';

r2.style.visibility = 'visible';

r2.onmouseup = stopRubber2;

}

IMG2.onmousemove = moveRubber2;

}

function moveRubber2 (evt2) {

if (document.all) { // IE

var r2 = document.all.rubberBand;

r2.style.width = event.x - r2.style.pixelLeft;

r2.style.height = event.y - r2.style.pixelTop;

}

else if (document.getElementById) { // firefox

var r2 = document.getElementById('rubberBand');

r2.style.width = evt2.clientX - parseInt(r2.style.left);

r2.style.height = evt2.clientY - parseInt(r2.style.top);

document.portion.x2.value=r2.style.height;

document.portion.y2.value=r2.style.width;

}

return false; // otherwise IE won't fire mouseup :/

}

function stopRubber2 (evt2) {

IMG2.onmousemove = null;

}

function cancelDragDrop2()

{

window.event.returnValue = false;

}

IMG2 = document.getElementById('myImage2');

IMG2.onmousedown = startRubber2;

IMG2.onmouseup = stopRubber2;

</SCRIPT>

</div>

<br>

<p align="center"><strong>Height: </strong>

<input type="text" name="x2" size="6">

<strong> Width:</strong>

<input type="text" name="y2" size="6"></p>

</td>

<td align="center" valign="middle"><img name="myImage3" id="myImage3" src="Image3.jpg" width="320" height="400"/>

<DIV ID="rubberBand"></DIV>

<div align="center">

<SCRIPT>

var IMG3;

function startRubber3 (evt3) {

if (document.all) {

var r3 = document.all.rubberBand;

r3.style.width = 0;

r3.style.height = 0;

r3.style.pixelLeft = event.x;

r3.style.pixelTop = event.y;

r3.style.visibility = 'visible';

IMG3.ondragstart = cancelDragDrop3; // otherwise IE will try to drag the image

}

else if (document.getElementById) {

// firefox

evt3.preventDefault();

var r3 = document.getElementById('rubberBand');

r3.style.width = 0;

r3.style.height = 0;

r3.style.left = evt3.clientX + 'px';

r3.style.top = evt3.clientY + 'px';

r3.style.visibility = 'visible';

r3.onmouseup = stopRubber3;

}

IMG3.onmousemove = moveRubber3;

}

function moveRubber3(evt3) {

if (document.all) { // IE

var r3 = document.all.rubberBand;

r3.style.width = event.x - r3.style.pixelLeft;

r3.style.height = event.y - r3.style.pixelTop;

}

else if (document.getElementById) { // firefox

var r3 = document.getElementById('rubberBand');

r3.style.width = evt3.clientX - parseInt(r3.style.left);

r3.style.height = evt3.clientY - parseInt(r3.style.top);

document.portion.x3.value=r3.style.height;

document.portion.y3.value=r3.style.width;

}

return false; // otherwise IE won't fire mouseup :/

}

function stopRubber3 (evt3) {

IMG3.onmousemove = null;

}

function cancelDragDrop3()

{

window.event.returnValue = false;

}

IMG3 = document.getElementById('myImage3');

IMG3.onmousedown = startRubber3;

IMG3.onmouseup = stopRubber3;

</SCRIPT>

</div>

<br>

<p align="center"><strong>Height: </strong>

<input type="text" name="x3" size="6">

<strong> Width:</strong>

<input type="text" name="y3" size="6"></p>

</td>

</tr><br>

<tr><td colspan="3"><br>

<p align="center">

<label>

<input type="submit" name="Submit" value="Save">

</label>

<label>

<input type="reset" name="Submit2" value="Reset">

</label>

</p></td></tr>

</table>

</form>

</Body>

<html>

**REFERENCES**

[1] S. Sood, A. Sarje, and K. Singh, “Cryptanalysis of password authentication schemes: Current status and key issues,” in Methods and Models in Computer Science, 2009. ICM2CS 2009. Proceeding of International Conference on, Dec 2009, pp. 1–7.

[2] S. Gurav, L. Gawade, P. Rane, and N. Khochare, “Graphical password authentication: Cloud securing scheme,” in Electronic Systems, Signal Processing and Computing Technologies (ICESC), 2014 International Conference on, Jan 2014, pp. 479–483.

[3] K. Gilhooly, “Biometrics: Getting back to business,” Computerworld, May, vol. 9, 2005.

[4] R. Dhamija and A. Perrig, “Deja vu: A user study using images for authentication,” in Proceedings of the 9th conference on USENIX Security Symposium-Volume 9. USENIX Association, 2000, pp. 4–4.

[5] “Realuser,” <http://www.realuser.com/>.

[6] I. Jermyn, A. Mayer, F. Monrose, M. Reiter, and A. Rubin, “The design and analysis of graphical passwords,” in Proceedings of the 8th conference on USENIX Security Symposium-Volume 8. USENIX Association, 1999, pp. 1–1.

[7] S. Wiedenbeck, J. Waters, J. Birget, A. Brodskiy, and N. Memon, “Passpoints: Design and longitudinal evaluation of a graphical password system,” International Journal of Human-Computer Studies, vol. 63, no. 1-2, pp. 102–127, 2005.

[8] A. Paivio, T. Rogers, and P. Smythe, “Why are pictures easier to recall than words?” Psychonomic Science, 1968.

[9] D. Nelson, U. Reed, and J. Walling, “Picture superiority effect,” Journal of Experimental Psychology: Human Learning and Memory, vol. 3, pp. 485–497, 1977.

[10] S. Brostoff and M. Sasse, “Are passfaces more usable than passwords? a field trial investigation,” PEOPLE AND COMPUTERS, pp. 405–424, 2000.

[11] A. De Angeli, M. Coutts, L. Coventry, G. Johnson, D. Cameron, and M. Fischer, “Vip: a visual approach to user authentication,” in Proceedings of the Working Conference on Advanced Visual Interfaces. ACM, 2002, pp. 316–323.

[12] B. Ives, K. Walsh, and H. Schneider, “The domino effect of password reuse,” Communications of the ACM, vol. 47, no. 4, pp. 75–78, 2004.

[13] J. Long and K. Mitnick, No Tech Hacking: A Guide to Social Engineering, Dumpster Diving, and Shoulder Surfing. Elsevier Science, 2011.

[14] T. Kwon, S. Shin, and S. Na, “Covert attentional shoulder surfing: Human adversaries are more powerful than expected,” IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 44, no. 6, pp. 716–727, June 2014.

[15] “Google glass snoopers can steal your passcode with a glance,” <http://www.wired.com/2014/06/google-glass-snoopers-cansteal-> your-passcode-with-a-glance/.

[16] M. Sasse, S. Brostoff, and D. Weirich, “Transforming the weakest linka human/computer interaction approach to usable and effective security,” BT technology journal, vol. 19, no. 3, pp. 122–131, 2001.

[17] “Mobile marketing statistics compilation,” <http://www.smartinsights.com/mobile-marketing/mobilemarketing-> analytics/mobile-marketing-statistics/.

[18] D. Hong, S. Man, B. Hawes, and M. Mathews, “A password scheme strongly resistant to spyware,” in Proceedings of International conference on security and management, 2004.

[19] D. Tan, P. Keyani, and M. Czerwinski, “Spy-resistant keyboard: Towards more secure password entry on publicly observable touch screens,” in Proceedings of OZCHI-Computer-Human Interaction Special Interest Group (CHISIG) of Australia. Canberra, Australia: ACM Press. Citeseer, 2005.

[20] M. Kumar, T. Garfinkel, D. Boneh, and T. Winograd, “Reducing shoulder-surfing by using gaze-based password entry,” in Proceedings of the 3rd symposium on Usable privacy and security. ACM, 2007, pp. 13–19.