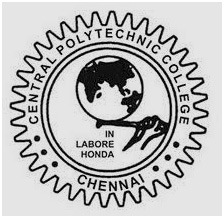
**CENTRAL POLYTECNIC COLLEGE**



**DEPARTMENT OF COMPUTER ENGINEERING**

**PROJECT WORK**

**INTELLIGENT RULE-BASED PHISHING WEBSITES CLASSIFICATION ON URL FEATURES**

Guided by

**N. Chandrika B.E.,**

Submitted by

**M.SANTHIYA 501117044**

**M.ISHWARYA 501117022**

**V.SUDHAVALLI 501118506**

**Of VI SEMESTER** Diploma In Computer Engineering during the Academic Year 2019-2020

# INTELLIGENT RULE-BASED PHISHING WEBSITES

**CLASSIFICATION ON URL FEATURES**

**DEPARTMENT OF COMPUTER ENGINEERING BONAFIDE CERTIFICATE**

### This is to Certify that this is Bonafide Record of the project entitled

**“Intelligent rule-based phishing websites classification on url features”** done by

|  |  |
| --- | --- |
| M.SANTHIYA | 501117044 |
| M.ISHWARYA | 501117022 |
| V.SUDHAVALLI | 501118506 |

**Of VI SEMESTER** Diploma In Computer Engineering during the Academic Year 2019-2020

**PROJECT GUIDE HEAD OF THE DEPARTMENT**

Submitted for the Board Examination held on

## INTERNAL EXAMINER EXTERNAL

**ACKNOWLEDGEMENT**

**“Success is doing ordinary things extraordinary well”**

### I take this opportunity to thank God, the Almighty, with whose blessing I have been able to successfully, carryout this project.

We wish to express our gratitude to our beloved parents for what they are to us. We would like to thank our Principal , **Dr M.S Padmanaban ph.D.,** for his kind patronage, guidance, encouragement and facilities offered.

We feel pleasure in thanking the Head of The Department of Computer Engineering, **Mrs.B.Gnana priya M.E.,** for granting me permission to pursue this project.

We would like to express our sincere thanks to our guide **Ms.N.Chandrika B.E.,** for her valuable suggestions throughout the project.

We thank **Mrs.S.Durgadevi B.E.,** our laboratory assistance who provided us with lab facilities during free hours and finally heartfelt thanks to all my friends who mad to complete this project excellently.

**Table content**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **CONTENT** | **PAGE NO** |
| **1.** | **ABSTRACT** | **5** |
| **2.** | **INRODUCTION** | **6** |
| **3.** | **LITERATURE SURVEY** | **9** |
| **4.** | **SYSTEM ANALYSIS**   * 1. EXISTING SYSTEM   2. PROPOSED SYSTEM 4.3REQUIREMENTS SPECIFICATION | **15** |
| **5.** | **SYSTEM DESIGN**   * 1. SYSTEM ARCHITECTURE   2. USE CASE DIAGRAM   3. COLLABRATION DIAGRAM   4. SEQUENCE DIAGRAM   5. CLASS DIAGRAM   6. DATAFLOW DIAGRAM   7. ER DIAGRAM | **18** |

|  |  |  |
| --- | --- | --- |
| **6.** | **SOFTWARE DESCRITION**   * 1. JAVA PROGRAMMING LANGUAGUE   2. WORKING OF JAVA | **26** |
| **7.** | **SYSTEM IMPLEMENTATION** | **31** |
| **8.** | **FEASIBILITY STUDY** | **35** |
| **9.** | **CODING** | **37** |
| **10.** | **SCREENSHOTS** | **71** |
| **11.** | **CONCLUSION** | **74** |
| **12.** | **REFRENCES** | **76** |

**1. ABSTRACT**

There are number of users who purchase products online and make payment through e-banking. There are e-banking websites who ask the users to provide sensitive data such as username, password or credit card details etc.., often for malicious reasons. This type of Websites are called as phishing websites. In order to detect and predict the e-banking phishing websites we propose an intelligent, flexible and effective system based on using classification Data mining algorithm. The e-banking phishing websites can be detected based on some important characteristics like URL, domain identity, security and encryption criteria in the final phishing detection rate. Once user makes transaction through online and when user makes payments through the e-banking websites our system will use data mining algorithm to detect whether the e-banking websites is phishing websites or not.

**CHAPTER 2**

**2.1 INTRODUCTION**

Online transactions are nowadays become very common and there are various attacks present behind this. In these kinds of attacks, phishing is identified as a major security threat and new innovative ideas are arising with this in each second so preventive mechanism should also be so effective. Thus the security in these cases be very high and should not be easily tractable with implementation easiness. Today, most applications are only as secure as their underlying system. Since the design and technology of middleware has improved steadily, their detection is a difficult problem. As a result, it is nearly impossible to be sure whether a computer that is connected to the internet can be considered trustworthy and secure or not. Phishing scams are also becoming a problem for online banking and e-commerce users. The question is how to handle applications that require a high level of security. Phishing is a form of online identity theft that aims to steal sensitive information such as online banking passwords and credit card information from users. Phishing scams have been receiving extensive press coverage because such attacks have been escalating in number and sophistication.

One definition of phishing is given as “it is a criminal activity using social engineering techniques. Phishers attempt to fraudulently acquire sensitive information, such as passwords and credit card details, by masquerading as a trustworthy person or business in an electronic communication”. The conduct of identity theft with this acquired sensitive information has also become easier with the use of technology and identity theft can be described as “a crime in which the impostor obtains key pieces of information such as Social Security and driver's license numbers and uses them for his or her own gain”. Phishing attacks rely upon a mix of technical deceit and social engineering practices. In the majority of cases the phisher must persuade the victim to intentionally perform a series of actions that will provide access to confidential information. Communication channels such as email, webpages, IRC and instant messaging services are popular. In all cases the phisher must impersonate a trusted source for the victim to believe. To date, the most successful phishing attacks have been initiated by email – where the phisher impersonates the sending authority so here introduces a new method which can be used as a safe way against phishing which is named as "A novel approach against Anti-phishing using visual cryptography". As the name describes, in this approach website cross verifies its own identity and proves that it is a genuine website (to use bank transaction, E-commerce and online booking system etc.) before the end users and make the both the sides of the system secure as well as an authenticated one.

**2.2 NEED OF PROJECT**

Online transactions are nowadays become very common and there are various attacks present behind this. In these types of various attacks, phishing is identified as a major security threat and new innovative ideas are arising with this in each second so preventive mechanism should also be so effective. Thus the security in these cases be very high and should not be easily tractable with implementation easiness. Today, most applications are only as secure as their underlying system. Since the design and technology of middleware has improved steadily, their detection is a difficult problem. As a result, it is nearly impossible to be sure whether a computer that is connected to the internet can be considered trustworthy and secure or not. Phishing scams are also becoming a problem for online banking and e-commerce users. The question is how to handle applications that require a high level of security.

**2.3 OBJECTIVE OF PROJECT**

Phishing is a form of online identity theft that aims to steal sensitive information such as online banking passwords and credit card information from users. Phishing scams have been receiving extensive press coverage because such attacks have been escalating in number and sophistication. One definition of phishing is given as “it is a criminal activity using social engineering techniques. Phishers attempt to fraudulently acquire sensitive information, such as passwords and credit card details, by masquerading as a trustworthy person or business in an electronic communication”. So here introduces a new method which can be used as a safe way against phishing which is named as “On the Relation of Random Grid and Deterministic Visual Cryptography". As the name describes, in this approach website cross verifies its own identity and proves that it is a genuine website (to use bank transaction, E-commerce and online booking system etc.) before the end users and make the both the sides of the system secure as well as an authenticated one. The concept of image processing and an improved visual cryptography is used. Image processing is a technique of processing an input image and to get the output as either improved form of the same image and/or characteristics of the input image. Visual Cryptography (VC) is a method of encrypting a secret image to shares, such that stacking a sufficient number of shares reveals the secret image.

**2.4 SCOPE OF THE PROJECT**

The concept of image processing and an improved visual cryptography is used. Image processing is a technique of processing an input image and to get the output as either improved form of the same image and/or characteristics of the input image. Visual Cryptography (VC) is a method of encrypting a secret image to shares, such that stacking a sufficient number of shares reveals the secret image.

**CHAPTER 3**

**LITERATURE SURVEY**

**TITLE 1: PREVENTING CROSS SITE REQUEST FORGERY ATTACKS**

**AUTHOR:** Nenad Jovanovic, Engin Kirda, and Christopher Kruegel.

**YEAR:** 2006

**DESCRIPTION:**

The Web has become an indispensable part of our lives. Unfortunately, as our dependency on the Web increases, so does the interest of attackers in exploiting Web applications and Web-based information systems. Previous work in the field of Web application security has mainly focused on the mitigation of cross site scripting (XSS) and SQL injection attacks. In contrast, cross site request forgery (XSRF) attacks have not received much attention. In an XSRF attack, the trust of a Web application in its authenticated users is exploited by letting the attacker make arbitrary HTTP requests on behalf of a victim user. The problem is that Web applications typically act upon such requests without verifying that the performed actions are indeed intentional. Because XSRF is a relatively new security problem, it is largely unknown by Web application developers. As a result, there exist many Web applications that are vulnerable to XSRF. Unfortunately, existing mitigation approaches are time-consuming and error-prone, as they require manual effort to integrate defense techniques into existing systems. In this paper, we present a solution that provides a completely automatic protection from XSRF attacks. More precisely, our approach is based on a server-side proxy that detects and prevents XSRF attacks in a way that is transparent to users as well as to the Web application itself. We provide experimental results that demonstrate that we can use our prototype to secure a number of popular open-source Web applications, without negatively affecting their behaviour.

**MERITS:**

* XSRF is a server side proxy.
* It detects and prevents the attack which is transparent to the user.

**DEMERITS:**

* Due to difficulties in client side proxies, the implementation does not support SSL connection.
* The reliability of this technique is difficult to assess.

**TITLE 2:** **A LARGESCALE EXPLORATORY ANALYSIS OF SOFTWARE VULNERABILITY LIFE CYCLES**

**AUTHOR:** Muhammad Shahzad, Muhammad Zubair Shafiq, Alex X. Liu

**YEAR:** 2012

**DESCRIPTION:**

Software systems inherently contain vulnerabilities that have been exploited in the past resulting in significant revenue losses. The study of vulnerability life cycles can help in the development, deployment, and maintenance of software systems. It can also help in designing future security policies and conducting audits of past incidents. Furthermore, such an analysis can help customers to assess the security risks associated with software products of different vendors. In this paper, we conduct an exploratory measurement study of a large software vulnerability data set containing 46310 vulnerabilities disclosed since 1988 till 2011. We investigate vulnerabilities along following seven dimensions: (1) phases in the life cycle of vulnerabilities, (2) evolution of vulnerabilities over the years, (3) functionality of vulnerabilities, (4) access requirement for exploitation of vulnerabilities, (5) risk level of vulnerabilities, (6) software vendors, and (7) software products. Our exploratory analysis uncovers several statistically significant findings that have important implications for software development and deployment.

**MERITS:**

* Complexity ofvulnerabilities are more agile and it is increasing the level of security risk.
* This focus on understanding the technical details of disclosed vulnerabilities.

**DEMERITS:**

* This does not focus on the large scale trend invulnerabilities.
* There is no difference between the open and closed source vendors.

**TITLE 3: DEFINING CODE-INJECTION ATTACKS**

**AUTHOR:** Donald Ray, Jay Ligatti

**YEAR:** 2012

**DESCRIPTION:**

This paper shows that existing definitions of code-injection attacks (e.g., SQL-injection attacks) are flawed. The flaws make it possible for attackers to circumvent existing mechanisms, by supplying code-injecting inputs that are not recognized as such. The flaws also make it possible for benign inputs to be treated as attacks. After describing these flaws in conventional definitions of code-injection attacks, this paper proposes a new definition, which is based on whether the symbols input to an application get used as (normal form) values in the application’s output. Because values are already fully evaluated, they cannot be considered “code” when injected. This simple new definition of code-injection attacks avoids the problems of existing definitions, improves our understanding of how and when such attacks occur, and enables us to evaluate the effectiveness of mechanisms for mitigating such attacks.

**MERITS:**

* These attacks are commonly referred to as Code Injection Attack or Command injection Attack.
* It improves the ability to recognises the attack.

**DEMERITS:**

* It does not alter the syntactic structure if the program.
* We can’t effectively develop or analyse the mechanism.

**TITLE 4:** **REINING IN THE WEB WITH CONTENT SECURITY POLICY**

**AUTHOR:** Sid Stamm, Brandon Sterne, Gervase Markham

**YEAR:** 2010

**DESCRIPTION:**

The last three years have seen a dramatic increase in both awareness and exploitation of Web Application Vulnerabilities. 2008 and 2009 saw dozens of high-profile attacks against websites using Cross Site Scripting (XSS) and Cross Site Request Forgery (CSRF) for the purposes of information stealing, website defacement, malware planting, clickjacking, etc. While an ideal solution may be to develop web applications free from any exploitable vulnerabilities, real world security is usually provided in layers. We present content restrictions, and a content restrictions enforcement scheme called Content Security Policy (CSP), which intends to be one such layer. Content restrictions allow site designers or server administrators to specify how content interacts on their web sites—a security mechanism desperately needed by the untamed Web. These content restrictions rules are activated and enforced by supporting web browsers when a policy is provided for a site via HTTP, and we show how a system such as CSP can be effective to lock down sites and provide an early alert system for vulnerabilities on a web site. Our scheme is also easily deployed, which is made evident by our prototype implementation in Firefox and on the Mozilla Add-Ons web site.

**MERITS:**

* It can leads to significant changes to its appearance and its operations.
* There is enough potential for recovering malicious code.

**DEMERITS:**

* Less sufficient attack.
* Web application does not validate the input after the data is injected.

**TITLE 5: RUN-TIME MONITORING AND FORMAL ANALYSIS OF INFORMATION FLOWS IN CHROMIUM**

**AUTHOR:** Lujo Bauer Shaoying Cai, Limin Jia Timothy Passaro Michael Stroucken Yuan Tian

**YEAR:** 2015

**DESCRIPTION:**

Web browsers are a key enabler of a wide range of online services, from shopping and email to banking and health services. Because these services frequently involve handling sensitive data, a wide range of web browser security policies and mechanisms has been implemented or proposed to mitigate the dangers posed by malicious code and sites. This paper describes an approach for specifying and enforcing flexible information-flow policies on the Chromium web browser. Complementing efforts that focus on information-flow enforcement on JavaScript, our approach focuses on an existing browser and encompasses a broad range of browser features, from pages and scripts to DOM elements, events, persistent state, and extensions. In our approach, which is a coarse-grained, light-weight implementation of taint tracking, entities in the browser are annotated with information-flow labels that specify policy and track information flows. We develop a detailed formal model of our approach, for which we prove noninterference. We also develop a corresponding prototype system built on top of Chromium. We demonstrate, and experimentally confirm, that the system can enforce many existing browser policies, as well as practically useful policies beyond those enforceable in standard web browsers.

**MERITS:**

* Increased convenience and productivity.
* It is more flexible.

**DEMERITS:**

* It does not constraint mainly on the information flows.
* It minimizes the functionality and overhead.

**CHAPTER 4**

**SYSTEM ANALYSIS**

**4.1 EXISTING SYSTEM**.

Some of the web pages look exactly like the real ones. Victims of phishing web pages may expose their bank account, password, credit card number, or other important information to the phishing web page owners. It includes techniques such as tricking customers through email and spam messages, man in the middle attacks, installation of key loggers and screen captures.

**4.1.1 DISADVANTAGES OF EXISTING SYSTEM**

These popular technologies have several drawbacks:

* **BLACKLIST-BASED TECHNIQUE** with low false alarm probability, but it cannot detect the websites that are not in the blacklist database. Because the life cycle of phishing websites is too short and the establishment of blacklist has a long lag time, the accuracy of blacklist is not too high.
* **HEURISTIC-BASED ANTI-PHISHING TECHNIQUE** is finding an approximate solution when classic methods fail to find any exact solution. But it has high probability of false and failed alarm, and it is easy for the attacker to use technical means to avoid the heuristic characteristics detection.
* **SIMILARITY ASSESSMENT BASED TECHNIQUE** is time-consuming. It needs too long time to calculate a pair of pages, so using the method to detect phishing websites on the client terminal is not suitable. And there is low accuracy rate for this method depends on many factors, such as the text, images, and similarity measurement.

**4.2 PROPOSED SYSTEM**

* Phishing websites comprise a variety of cues within its content-parts as well as the browser-based security indicators provided along with the website.
* The use of images is explored to preserve the privacy of image captcha by decomposing the original image captcha into two shares that are stored in separate database servers such that the original image captcha can be revealed only when both are simultaneously available; the individual sheet images do not reveal the identity of the original image captcha. Once the original image captcha is revealed to the user it can be used as the password.
* To develop the captcha we have been using the Visual Cryptography technique. VCS is a cryptographic technique that allows for the encryption of visual information such that decryption can be performed using the human visual system.
* (2, 2)- Threshold VCS scheme- This is a simplest threshold scheme that takes a secret message and encrypts it in two different shares that reveal the secret image when they are overlaid.

**4.2.1 ADVANTAGES OF PROPOSED SYSTEM:**

* It prevents password and other confidential information from the phishing websites.
* URL address on the address bar of your internet browser begins with "https"; the letter’s’ at the end of "https" means 'secured'.
* Look for the padlock symbol either in the address bar or the status bar (mostly in the address bar) but not within the web page display area. Verify the security certificate by clicking on the padlock.

**4.3 REQUIREMENTS SPECIFICATION**

**4.3.1 SOFTWARE REQUIREMENTS**

Operating System : Windows XP or Higher

Languages used : Java (JSP, Servlet), HTML

Tools : JDK 1.7, Net Beans 7.0.1, SQLyog

Backend : My SQL

**4.3.2 HARDWARE REQUIREMENTS**

Processor : Pentium Dual Core 2.3 GHz

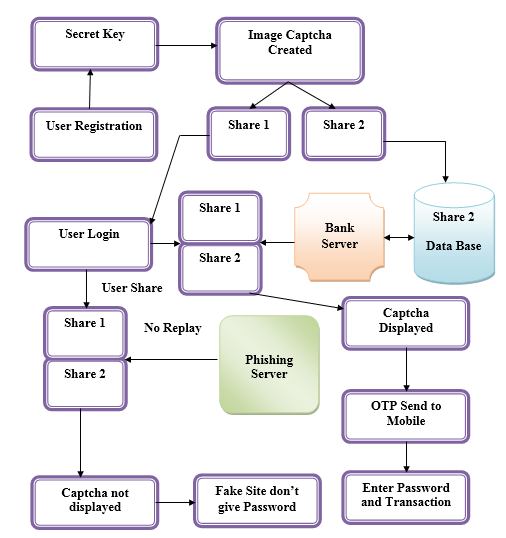
Hard Disk : 250 GB or Higher

Ram : 1 GB (Min)

**CHAPTER 5**

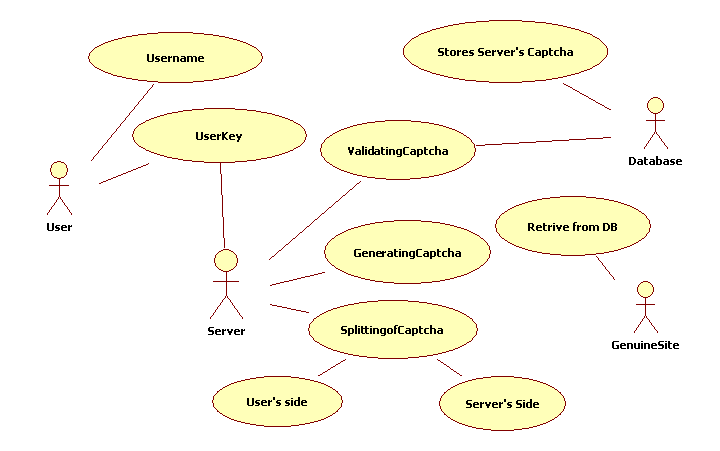
**SYSTEM DESIGN**

**5.1 SYSTEM ARCHITECTURE**



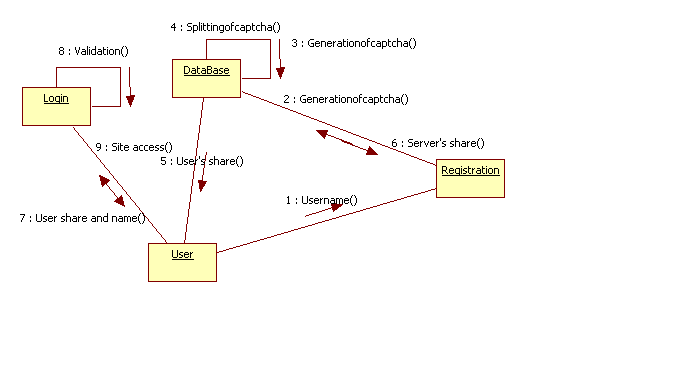
**5.2 USE CASE DIAGRAM**

A use case illustrates a unit of functionality provided by the system. The main purpose of the use-case diagram is to help development teams visualize the functional requirements of a system, including the relationship of "actors" (human beings who will interact with the system) to essential processes, as well as the relationships among different use cases. The use case has two actors: user and server. User gives the image as input and server performs the operation



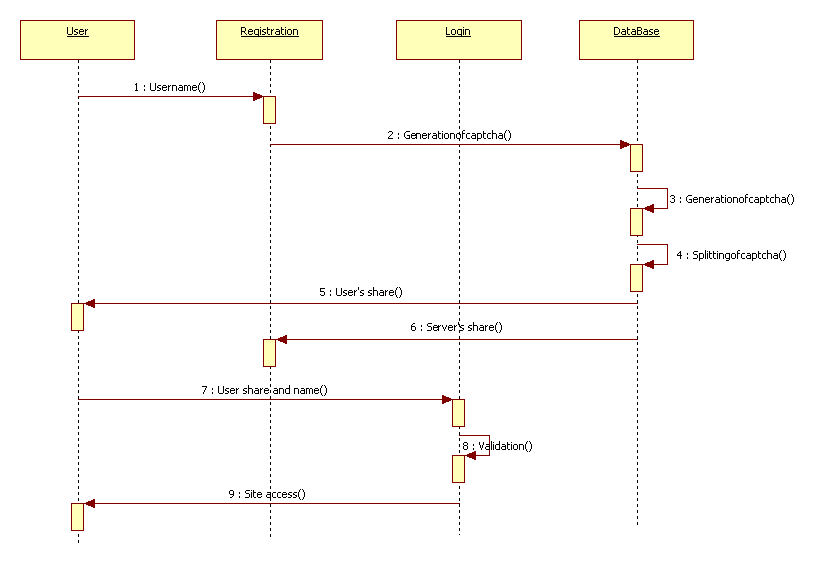
**5.3 COLLABORATION DIAGRAM**

Collaboration diagrams are a technique for defining external object behavior. They include the same information as Sequence Diagrams (or message trace diagrams) but are better able to show asynchronous message passing.  Collaboration diagrams show how objects collaborate by representing objects by icons and their message passing as labeled arrows



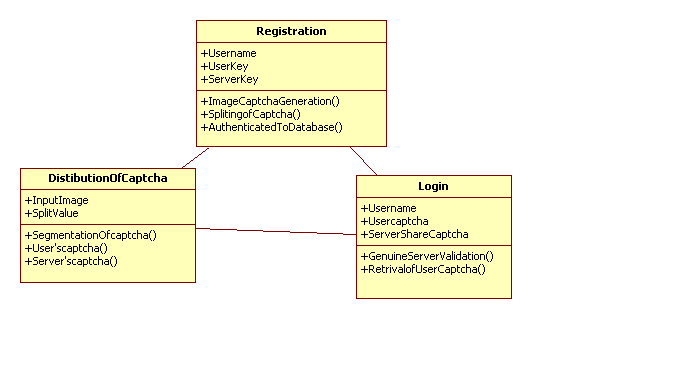
**5.4 SEQUENCE DIAGRAM**

A sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios

****

**5.5 CLASS DIAGRAM**

The class diagram shows how the different entities (people, things, and data) relate to each other; in other words, it shows the static structures of the system. A class diagram can be used to display logical classes. Class diagrams can also be used to show implementation classes, which are the things that programmers typically deal with. A class is depicted on the class diagram as a rectangle with three horizontal sections, as shown in above figure . The upper section shows the class's name; the middle section contains the class's attributes; and the lower section contains the class's operations (or "methods"). The diagram has five main classes which give the attributes and operations used in each class.

****

**5.6 DATA FLOW DIAGRAM**

**5.6.1 LEVEL 0:**

Register With legitimate site

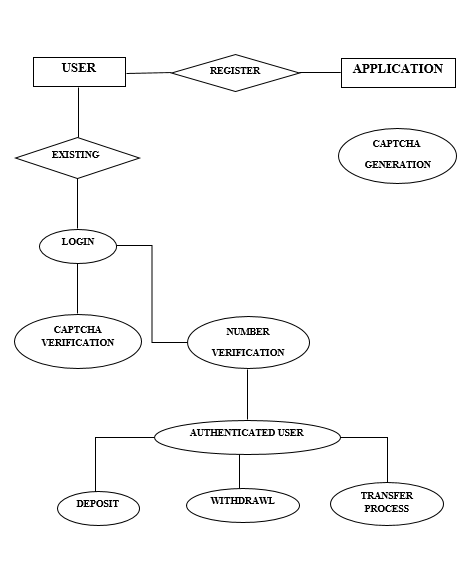
**5.6.2 LEVEL 1:**

Captcha for User string

**5.6.2 LEVEL 2:**

Request with Share1

**5.7 ER DIAGRAM**

****

**CHAPTER 6**

**SOFTWARE DESCRIPTION**

**6.1 JAVA PROGRAMMING LANGUAGE**

**Overview**

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

**Working of Java**

For those who are new to object-oriented programming, the concept of a class will be new to you. Simplistically, a class is the definition for a segment of code that can contain both data (called attributes) and functions (called methods). When the interpreter executes a class, it looks for a particular method by the name of **main,** which will sound familiar to C programmers. The main method is passed as a parameter an array of strings (similar to the args [] of C), and is declared as a static method.

To output text from the program, the **println**method of **System.out is** executed**,** which is java’s output stream. UNIX users will appreciate the theory behind such a stream, as it is actually standard output. Java consists of two things:

* + Programming language
  + Platform

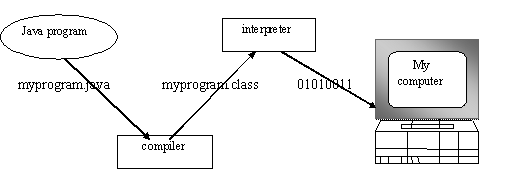
**The Java Programming Language**

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

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

The code and can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows:

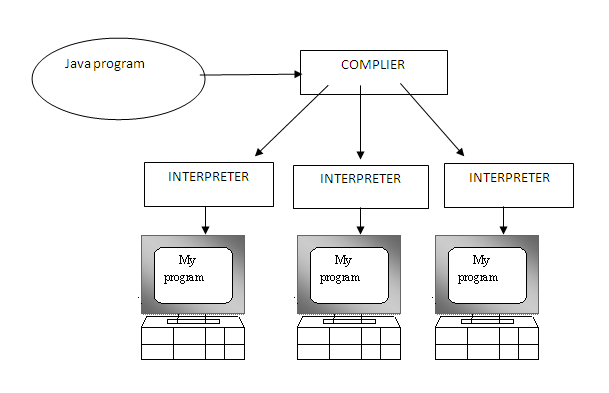
Java is unusual in that each Java program is both co implied and interpreted. With a compiler, you translate a Java program into an intermediate language called **Java byte codes** – the platform independent codes interpreted by the Java interpreter. With an interpreter, each Java byte code instruction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

****

**Java Virtual Machine**

You can think of Java byte codes as the machine code instructions for the **Java Virtual Machine (JVM).** Every Java interpreter, whether it’s a Java development tool or a Web browser that can run Java applets, is an implementation of JVM. That JVM can also be implemented in hardware. Java byte codes help make “write once, run anywhere” possible.

You can compile your Java program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the JVM.

****

**6.2 WORKING OF JVM**

**THE JAVA PLATFORM**

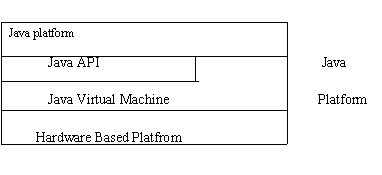
A platform is the hardware or software environment in which a program runs. The Java platform differs from most other platforms in that it’s a software-only platform that runs on top of other, hardware-based platforms. Most other platforms are described as a combination of hardware and operating system.

The Java platform has two components :

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

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

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

****

**JAVA Platform**

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

**CHAPTER 7**

**SYSTEM IMPLEMENTATION**

**7.1 MODULES**

* Registration With Secrete Code
* Image captcha Generation
* Shares Creation(VCS)
* Login Phase

## Module Description

**7.1 Registration With Secrete Code:**

In the registration phase, the user details user name, password, email-id, address and a key string(password) is asked from the user at the time of registration for the secure website. The key string can be a combination of alphabets and numbers to provide more secure environment. This string is concatenated with randomly generated string in the server.

**7.2 Image captcha Generation:**

A key string is converted into image using java classes BufferedImage and Graphics2D. The image dimension is 260\*60 Text color is red and the background color is white. Text font is set by Font class in java. After image generation it will be write into the user key folder in the server using ImageIO class.

**7.3 Shares Creation (VCS):**

The image captcha is divided into two shares such that one of the share is kept with the user and the other share is kept in the server. The user's share and the original image captcha is sent to the user for later verification during login phase. The image captcha is also stored in the actual database of any confidential website as confidential data.

**7.4 Login Phase:**

When the user logs in by entering his confidential information for using his account, then first the user is asked to enter his username (user id).Then the user is asked to enter his share which is kept with him. This share is sent to the server where the user's share and share which is stored in the database of the website for each user, is stacked together to produce the image captcha. The image captcha is displayed to the user.

Here the end user can check whether the displayed image captcha matches with the captcha created at the time of registration. The end user is required to enter the text displayed in the image captcha and this can serve the purpose of password and using this, the user can log in into the website. Using the username and image captcha generated by stacking two shares one can verify whether the website is genuine/secure website or a phishing website.

**CHAPTER 8**

**FEASIBILITY STUDY**

The objective of feasibility study is not only to solve the problem but also to acquire a sense of its scope. During the study, the problem definition was crystallized and aspects of the problem to be included in the system are determined. Consequently benefits are estimated with greater accuracy at this stage. The key considerations are:

* Economic Feasibility
* Technical feasibility
* Operational feasibility

**8.1 ECONOMIC FEASIBILITY**

Economic feasibility studies not only the cost of hardware, software is included but also the benefits in the form of reduced costs are considered here. This project, if installed will certainly be beneficial since there will be reduction in manual work and increase in the speed of work.

**8.2 TECHNICAL FEASIBILITY**

Technical feasibility evaluates the hardware requirements, software technology, available personnel etc., as per the requirements it provides sufficient memory to hold and process.

**8.3 OPERATIONAL FEASIBILITY**

This is the most important step of the feasibility study this study helps us predict the operational ability of the system that is being developed. This study also helps us analyze the approach towards which the system must be developed by which development effort is reduced. Proposed system is beneficial only if they can be turned into information systems that will meet the organization requirements. This system supports in producing good results and reduces manual work. Only by spending time to evaluate the feasibility, do we reduce the chances from extreme embarrassments at larger stager of the project. Effort spend on a feasibility analysis that results in the cancellation of a proposed project is not a wasted effort.

**CHAPTER 9**

**CODING**

<!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>

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

<title>Anti-Phishing</title>

<meta name="keywords" content="corporate blue, theme, free templates, website templates, CSS, HTML" />

<meta name="description" content="Corporate Blue Theme is a free website template provided by tooplate.com" />

<link href="tooplate\_style.css" rel="stylesheet" type="text/css" />

<style type="text/css">

#apDiv1 {

position: absolute;

width: 200px;

height: 193px;

z-index: 2;

left: 1149px;

top: 176px;

}

.side\_con

{

width:200px;

height:200px;

background-color:#6CF;

text-align:center;

font-size:18px;

margin-top:35px;

padding-top:20px;

}

.acc

{

width: 850px;

margin-left: 200px;

position: absolute;

left: 75px;

top: 510px;

margin-top: 30px;

}

.ll

{height:200px;

width:200px;

background-color:#6CF;

}

table

{

margin-left: 250PX;

margin-top: 30PX;

}

h3{

COLOR:darkgreen;

padding-left:40PX;

}

</style>

<script type="text/javascript">

function valid()

{

var p=document.getElementById("email");

var cp=document.getElementById("remail");

if(p.value!=cp.value)

{

alert("Email Id doesnt match!!!");

return false;

}

var pa=document.getElementById("pass");

var cpa=document.getElementById("cpass");

if(pa.value!=cpa.value)

{

alert("Password doesnt match!!!");

return false;

}

if( document.getElementById("uname").value== "")

{

alert("Enter Username-name");

return false;

}

if( document.getElementById("cod").value== "")

{

alert("Enter Secret Code");

return false;

}

if(document.getElementById("mob").value=="")

{

alert("Enter Mobile");

return false;

}

return true;

}

</script>

<!-- Free Website Template by t o o p l a t e . c o m -->

</head>

<body>

<div id="tooplate\_header\_wrapper">

<div id="tooplate\_header">

<div id="site\_title">

<h1><a href="http://www.tooplate.com" rel="nofollow"><img src="images/logo.gif" alt="logo" width="238" height="100" /></a></h1>

</div>

<div id="apDiv2"><H3 align="center" STYLE="color:cadetblue">Never Response Any Mail Or Sms</H3>

</div>

<!-- end of site\_title -->

<div id="header\_right">

<div id="social\_box">

<ul>

<li><a href="#"><img src="images/facebook.png" alt="facebook" /></a></li>

<li><a href="#"><img src="images/twitter.png" alt="twitter" /></a></li>

<li><a href="#"><img src="images/linkedin.png" alt="linkin" /></a></li>

<li><a href="#"><img src="images/technorati.png" alt="technorati" /></a></li>

<li><a href="#"><img src="images/myspace.png" alt="myspace" /></a></li>

</ul>

<div class="cleaner"></div>

</div>

<div id="search\_box">

<form action="#" method="get">

<input type="text" value="Search" name="q" size="10" id="searchfield" title="searchfield" onfocus="clearText(this)" onblur="clearText(this)" />

<input type="submit" name="Search" value="" id="searchbutton" title="Search" />

</form>

</div>

</div>

</div>

</div>

<div id="tooplate\_menu">

<ul>

<li><a href="index.html" class="current">Home</a></li>

<li><a href="accountrst.jsp">Accounts</a></li>

<li><a href="loan.jsp">Loan</a></li>

<li><a href="registeration.jsp">Registration</a></li>

<li><a href="#">Login</a></li>

</ul>

<div class="cleaner"></div>

</div>

<div id="tooplate\_middle\_wrapper">

<div class="post">

<div id="tooplate\_middle">

<img src="images/pwd.jpg" alt="Image 01" height="179" width="329" />

<div id="middle\_content">

<h2>Beware of Phishing Attacks</h2>

<p align="justify"> Phishing is a fraudulent attempt, usually made through email, phone calls, SMS etc seeking your personal and confidential information. <br />

Its representative never sends you email/SMS or calls you over phone to get your personal information, password or one time SMS (high security) password. Any such e-mail/SMS or phone call is an attempt to fraudulently withdraw money from your account through Internet Banking. Never respond to such email/SMS or phone call. </p>

</div>

</div>

</div>

</div>

<!-- end of templatetooplate\_menu -->

<div id="tooplate\_content">

<div id="id\_left">

</div>

</div>

<div class="acc" style="background-color:cadetblue; " >

<span class="wel"><h2>&nbsp;</h2>

<h2 align="center"> REGISTRATION</h2>

</span> <span class="inr\_bllk"><span class="txt">

<form name="i" action="file.jsp" method="post" class="register" onsubmit="return valid();">

<h3>Account Details</h3>

<table>

<table ><tr><td>Email \*</td><td><input type="email" name="email" id="email" required="required"/></td></tr>

<tr><td>Repeat email \*</td><td> <input type="email" name="remail" id="remail" required="required"/></td></tr>

<tr><td>password \*</td><td><input type="password" name="pass" id="pass" required="required"/></td></tr>

<tr><td>Retype Password \*</td><td><input type="password" name="pass" id="cpass" required="required"/></td></tr></table>

<tr><td><h3>Personal Details</h3></td><td></td></tr>

<table ><tr><td>User Name \*</td><td><input type="text" name="userName" id="uname" class="long" required="required" onkeyup="showHint(this.value);"/>

<% String msg=request.getParameter("msg");if(msg!=null){%><label><font color="red"><%=msg%></font></label><%

}

%>

<span id="txtHint"></span> </td></tr>

<tr><td>Enter Secret Code \*</td><td><input type="text" name="string" id="cod" maxlength="8" required="required"/>(Eight Digit)</td></tr>

<tr><td>Mobile \*</td><td><input type="text" name="mobile" maxlength="10" id="mob" required="required"/></td></tr>

<tr><td>Street \*</td><td> <input type="text" name="street" class="long" required="required"/></td></tr>

<tr><td>City \*</td><td><input type="text" name="city" class="long" required="required"/></td></tr>

<tr><td>Country \*</td><td><select name="country" required="required">

<option>

</option>

<option value="United States">United States

</option>

<option value="India">India

</option>

</select></td></tr></table>

<tr><td><h3>Further Information</h3></td><td></td></tr>

<table ><tr><td>Gender \*</td><td> <input type="radio" value="Male" name="gender" required="required"/>

<label class="gender">Male</label>

<input type="radio" value="Female" name="gender" required="required"/>

<label class="gender">Female</label></td></tr>

<tr><td>Birthdate \*</td><td><select class="date" name="date" required="required">

<option value="1">01

</option>

<option value="2">02

</option>

<option value="3">03

</option>

<option value="4">04

</option>

<option value="5">05

</option>

<option value="6">06

</option>

<option value="7">07

</option>

<option value="8">08

</option>

<option value="9">09

</option>

<option value="10">10

</option>

<option value="11">11

</option>

<option value="12">12

</option>

<option value="13">13

</option>

<option value="14">14

</option>

<option value="15">15

</option>

<option value="16">16

</option>

<option value="17">17

</option>

<option value="18">18

</option>

<option value="19">19

</option>

<option value="20">20

</option>

<option value="21">21

</option>

<option value="22">22

</option>

<option value="23">23

</option>

<option value="24">24

</option>

<option value="25">25

</option>

<option value="26">26

</option>

<option value="27">27

</option>

<option value="28">28

</option>

<option value="29">29

</option>

<option value="30">30

</option>

<option value="31">31

</option>

</select>

<select name="month" required="required">

<option value="1">January

</option>

<option value="2">February

</option>

<option value="3">March

</option>

<option value="4">April

</option>

<option value="5">May

</option>

<option value="6">June

</option>

<option value="7">July

</option>

<option value="8">August

</option>

<option value="9">September

</option>

<option value="10">October

</option>

<option value="11">November

</option>

<option value="12">December

</option>

</select>

<input class="year" type="text" size="4" maxlength="4" name="year" required="required"/>e.g 1976</td></tr>

<tr><td>Nationality \*</td><td><select name="nationality" required="required">

<option value="india">India

</option>

<option value="1">United States

</option>

</select></td></tr>

<tr><td></td> </tr><tr><td></td> </tr><tr><td></td> </tr><tr><td></td><td><div class="reg"><button class="button">Register &raquo;</button></div></td></tr>

</table>

</table>

</form>

</span>

</span>

</div>

<!-- Free Website Template by t o o p l a t e . c o m -->

<br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/>

<div id="tooplate\_footer\_wrapper">

<div id="tooplate\_footer">

<p><a href="http://all-free-download.com/free-website-templates/">Home</a> <span>|</span> <a href="http://all-free-download.com/free-website-templates/">Clients </a> <span>|</span> <a href="http://all-free-download.com/free-website-templates/">Services</a> <span>|</span> <a href="http://all-free-download.com/free-website-templates/">About Us </a> <span>|</span> <a href="http://all-free-download.com/free-website-templates/">Contact us</a> </span></p>

<p>&nbsp;</p>

<p>Copyright © 2048 <a href="#">Company Name</a></p>

</div>

<!-- end of templatetooplate\_footer -->

</div>

</body>

</html>

File.jsp

<%@page import="javax.mail.internet.InternetAddress"%>

<%@page import="javax.mail.internet.MimeMessage"%>

<%@page import="javax.mail.\*"%>

<%@page import="java.util.Properties"%>

<%@page import="java.awt.geom.Rectangle2D"%>

<%@page import="java.awt.font.FontRenderContext"%>

<%@page import="java.awt.Font"%>

<%@page import="sun.net.www.content.image.png"%>

<%@page import="java.awt.RenderingHints"%>

<%@page import="java.security.SecureRandom"%>

<%@page import="java.awt.Graphics2D"%>

<%@page import="java.awt.Color"%>

<%@page import="java.awt.image.ColorConvertOp"%>

<%@page import="javax.imageio.ImageIO"%>

<%@page import="java.awt.image.BufferedImage"%>

<%@page import="java.sql.\*"%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@page import="java.io.\*"%>

<%@page import="classes.Pass"%>

<%@page import="classes.Mailsender"%>

<%--<jsp:include page="/Dbconn.jsp">

<jsp:param name="name" value="sos" />

</jsp:include>--%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

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

<title>JSP Page</title>

</head>

<body>

<%

int f=0;

String otp=null;

String sampleText =request.getParameter("string");

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

Pass pwd=new Pass();

sampleText=sampleText+pwd.generatePassword();

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

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

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

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

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

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

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

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

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

String dob=date+"-"+month+"-"+year;

String mob=request.getParameter("mobile");

ServletContext context = getServletContext();

String path=context.getRealPath("/Userkey/");

// out.println(path);

//create a File Object

File newFile = new File(path+"\\captcha"+userName+".jpeg");

String imgn=path+"\\captcha"+userName+".jpeg";

//create the font you wish to use

Font font = new Font("Forte", Font.PLAIN, 40);

//create the FontRenderContext object which helps us to measure the text

FontRenderContext frc = new FontRenderContext(null, true, true);

//get the height and width of the text

Rectangle2D bounds = font.getStringBounds(sampleText, frc);

int w = 260;

int h = 60;

//create a BufferedImage object

BufferedImage image = new BufferedImage(w, h, BufferedImage.TYPE\_INT\_RGB);

//calling createGraphics() to get the Graphics2D

Graphics2D gr = image.createGraphics();

//set color and other parameters

gr.setColor(Color.WHITE);

gr.fillRect(0, 0, w, h);

gr.setColor(Color.RED);

gr.setFont(font);

gr.drawString(sampleText, (float) bounds.getX(), (float) -bounds.getY());

//releasing resources

gr.dispose();

//creating the file

ImageIO.write(image, "jpeg", newFile);

try

{

//String userName=request.getParameter("userName");

//String userImage=request.getParameter("image");

// out.println(userImage);

// ServletContext context = getServletContext();

//String path=context.getRealPath("\\IMAGES\\");

//out.println(path);

try

{

Connection conn = null;

String url = "jdbc:mysql://localhost:3306/";

String dbName = "visual";

String driver = "com.mysql.jdbc.Driver";

String dbuserName = "root";

String password = "admin";

Class.forName(driver).newInstance();

conn = DriverManager.getConnection(url+dbName,dbuserName,password);

System.out.println("Connected to the database");

//out.println("database");

Statement st = conn.createStatement();

// out.println("modify database");

ResultSet rs = st.executeQuery("select \* from userdata");

while(rs.next())

{

String un=rs.getString("username");

//String im=rs.getString("userimage");

if(userName.equals(un))

{ f=1;

System.out.println("USER NAME ALREADY EXIST");

out.println("USER NAME ALREADY EXIST");

}

/\* else{

out.println("invalid user");

//response.sendRedirect("verify.jsp?msg=invalid user");

}\*/

}

if(f==1){ response.sendRedirect("registeration.jsp?msg=SORRY USER NAME NOT AVAILABLE!!!");}

if(f==0)

{

int val = st.executeUpdate("INSERT userdata VALUES('"+userName+"','"+imgn+"','"+pass+"','"+email+"','"+street+"','"+city+"','"+country+"','"+gender+"','"+dob+"','"+mob+"','"+otp+"')");

}

// out.println("1 row affected");

}

catch(SQLException ex)

{

out.println(ex);

}

int width=260; int height=60; boolean resize=false;

File fl=new File(path+"\\captcha"+userName+".jpeg");

BufferedImage imgSrc = null;

try {

imgSrc = ImageIO.read(fl);

} catch (Exception e) {

out.println("\*\*1\*\*\*\*\*\*\*\*\*\*\*");

out.println(e);

}

//out.println("\*\*\*\*\*\*2\*\*\*\*\*\*\*");

//out.println("\*\*\*\*\*\*21\*\*\*\*\*\*\*");

if (resize && (imgSrc.getWidth() > width || imgSrc.getHeight() > height))

//out.println("\*\*\*\*\*3\*\*\*\*\*\*\*\*");

// convert image to ARGB colorspace (if it isn't allready)

if (imgSrc.getType() != BufferedImage.TYPE\_INT\_ARGB) {

//out.println("\*\*\*\*\*\*4\*\*\*\*\*\*\*");

BufferedImage raw\_image = imgSrc;

imgSrc = new BufferedImage(raw\_image.getWidth(), raw\_image.getHeight(), BufferedImage.TYPE\_INT\_ARGB);

new ColorConvertOp(null).filter(raw\_image, imgSrc);

}

// out.println("\*\*\*\*\*5\*\*\*\*\*\*\*\*");

// check if image contains only black + transparent or white pixels

// colored pixels get converted to either black or transparent

for(int i = 0; i < imgSrc.getHeight(); i++) {

for(int j = 0; j < imgSrc.getWidth(); j++) {

int iRgb = imgSrc.getRGB(j, i);

// white to transparent

if(iRgb == Color.WHITE.getRGB()) {

imgSrc.setRGB(j, i, 0x00FFFFFF);

iRgb = imgSrc.getRGB(j, i);

}

// check if pixel is either fully transparent or black

if(!(iRgb>>>24 == 0 || iRgb == Color.BLACK.getRGB())) {

int r = (iRgb & 0x00FF0000)>>16;

int g = (iRgb & 0x0000FF00)>>8;

int b = iRgb & 0x000000FF;

// brightness by euclidian distance)

double brightness = (0.2126 \* r) + (0.7152 \* g) + (0.0722 \* b);

if (brightness > (255/2)) {

// transparent

imgSrc.setRGB(j, i, 0x00FFFFFF);

} else {

// black

imgSrc.setRGB(j, i, Color.BLACK.getRGB());

}

}

}

}

// resize image

if (!resize || (imgSrc.getWidth() == width && imgSrc.getHeight() == height)) {

BufferedImage imgSrcRes = new BufferedImage(width, height, BufferedImage.TYPE\_INT\_ARGB);

Graphics2D g = imgSrcRes.createGraphics();

int x = (width - imgSrc.getWidth()) / 2;

int y = (height - imgSrc.getHeight()) / 2;

g.drawImage(imgSrc, x, y, imgSrc.getWidth() + x, imgSrc.getHeight() + y, 0, 0, imgSrc.getWidth(), imgSrc.getHeight(), null);

//g.drawImage(img, x, y, width, height, observer)

g.dispose();

// File file=new File("C:\\Documents and Settings\\Uniq Tech\\My Documents\\My Pictures\\"+imgSrcRes+".png");

// FileWriter ffw=new FileWriter(fw);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//File file = new File(fileName + "." + ext);

File myPngImage = new File(path+"\\captchabw"+userName+".png");

System.out.println(myPngImage);

ImageIO.write(imgSrcRes, "png", myPngImage);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//out.println("asdfasdfasdf "+imgSrcRes);

//return imgSrcRes;

width \*= 2;

height \*= 2;

// generate empty key image

BufferedImage key = new BufferedImage(width, height, BufferedImage.TYPE\_INT\_ARGB);

Graphics2D keyGraphics = key.createGraphics();

// fill it with a fully transparent "white" (should allready be this way with TYPE\_INT\_ARGB)

keyGraphics.setColor(new Color(255, 255, 255, 0));

keyGraphics.fillRect(0, 0, width, height);

// fill it with the random key structure

keyGraphics.setColor(new Color(0, 0, 0, 255));

// get securerandom. on linux, this uses NativePRNG (e.g. /dev/urandom), on

// windows, it uses SHA1PRNG

SecureRandom secureRandom = new SecureRandom();

// each 2x2-pixel-pack has 2 randomly set pixels

for (int y1 = 0; y1 < height; y1 += 2) {

for (int x1 = 0; x1 < width; x1 += 2) {

// determine the two pixels

int px1 = secureRandom.nextInt(4);

int px2 = secureRandom.nextInt(4);

while (px1 == px2) px2 = secureRandom.nextInt(4);

// determine the coordinates of them

int px1x = (px1 < 2) ? px1 : px1 - 2;

int px1y = (px1 < 2) ? 0 : 1;

int px2x = (px2 < 2) ? px2 : px2 - 2;

int px2y = (px2 < 2) ? 0 : 1;

// write them

keyGraphics.fillRect(x1 + px1x, y1 + px1y, 1, 1);

keyGraphics.fillRect(x1 + px2x, y1 + px2y, 1, 1);

}

}

keyGraphics.dispose();

File myPngkey = new File(path+"\\keyimage"+userName+".png");

ImageIO.write(key, "png", myPngkey);

//return key;

if (key == null || imgSrc == null);

// check for key/source file match

if (imgSrc.getWidth() != key.getWidth() / 2 || imgSrc.getHeight() != key.getHeight() / 2) ;

// resize the source to the size of the key

BufferedImage imgSrcResen = new BufferedImage(key.getWidth(), key.getHeight(), BufferedImage.TYPE\_INT\_ARGB);

Graphics2D gen = imgSrcResen.createGraphics();

gen.setRenderingHint(RenderingHints.KEY\_INTERPOLATION, RenderingHints.VALUE\_INTERPOLATION\_NEAREST\_NEIGHBOR);

gen.drawImage(imgSrc, 0, 0, key.getWidth(), key.getHeight(), 0, 0, imgSrc.getWidth(), imgSrc.getHeight(), null);

gen.dispose();

BufferedImage imgEncr = new BufferedImage(key.getWidth(), key.getHeight(), BufferedImage.TYPE\_INT\_ARGB);

Graphics2D encrGraphics = imgEncr.createGraphics();

// fill it with a fully transparent "white" (should allready be this way with TYPE\_INT\_ARGB)

encrGraphics.setColor(new Color(255, 255, 255, 0));

encrGraphics.fillRect(0, 0, imgEncr.getWidth(), imgEncr.getHeight());

// encrypt

encrGraphics.setColor(new Color(0, 0, 0, 255));

// each 2x2-pixel-pack has 2 pixels to set

for(int y2 = 0; y2 < imgEncr.getHeight(); y2 += 2)

{

for (int x2 = 0; x2 < imgEncr.getWidth(); x2 += 2) {

// because 1 black pixel of the original image is now a square of 4 black pixels,

// only the first pixel has to be checked

if (imgSrcResen.getRGB(x2, y2) == Color.BLACK.getRGB()) {

// write the two pixels to complete the block together with the key

if (key.getRGB(x2, y2)>>>24 == 0) encrGraphics.fillRect(x2, y2, 1, 1);

if (key.getRGB(x2 + 1, y2)>>>24 == 0) encrGraphics.fillRect(x2 + 1, y2, 1, 1);

if (key.getRGB(x2, y2 + 1)>>>24 == 0) encrGraphics.fillRect(x2, y2 + 1, 1, 1);

if (key.getRGB(x2 + 1, y2 + 1)>>>24 == 0) encrGraphics.fillRect(x2 + 1, y2 + 1, 1, 1);

}

else

{

// write the two pixels at the same position in the key

if (key.getRGB(x2, y2) == Color.BLACK.getRGB()) encrGraphics.fillRect(x2, y2, 1, 1);

if (key.getRGB(x2 + 1, y2) == Color.BLACK.getRGB()) encrGraphics.fillRect(x2 + 1, y2, 1, 1);

if (key.getRGB(x2, y2 + 1) == Color.BLACK.getRGB()) encrGraphics.fillRect(x2, y2 + 1, 1, 1);

if (key.getRGB(x2 + 1, y2 + 1) == Color.BLACK.getRGB()) encrGraphics.fillRect(x2 + 1, y2 + 1, 1, 1);

}

}

}

encrGraphics.dispose();

File myPngen = new File(path+"\\encryptimage"+userName+".png");

ImageIO.write(imgEncr, "png", myPngen);

//return imgEncr;

Properties props=new Properties();

props.put("mail.smtp.host", "smtp.gmail.com");

props.put("mail.smtp.socketFactory.port","465");

props.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");

props.put("mail.smtp.auth", "true");

props.put("mail.smtp.port", "465");

Session session1=Session.getDefaultInstance(props, new Authenticator()

{

@Override

protected javax.mail.PasswordAuthentication getPasswordAuthentication(){

return new javax.mail.PasswordAuthentication("unknownid28@gmail.com", "massazeesa");

}

});

try{

// multiple senders

//String to[]={"dharani081990@gmail.com","dharani@uniqtechnologies.co.in"};

String sender=email;

/\*InternetAddress[] addressTo=new InternetAddress[to.length];

for(int i=0;i<to.length;i++)

{

addressTo[i]=new InternetAddress(to[i]);

}\*/

MimeMessage msg=new MimeMessage(session1);

msg.setFrom(new InternetAddress("unknownid28@gmail.com"));

msg.addRecipients(Message.RecipientType.TO, sender);

msg.setSubject("PASSWORD FROM BANK");

// String pass=password;

msg.setText("This is your password : "+pass);

Transport.send(msg);

System.out.println("Mail sent successfully");

}

catch(Exception e)

{

e.printStackTrace();

}

}

}

catch(Exception e)

{

out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*");

out.println("final"+e);

}

%>

<h1>ORIGINAL image</h1>

<%

if(f==0)

{

response.sendRedirect("cregisteration.jsp?captcha="+"Userkey/captchabw"+userName+".png&encrypt="+"Userkey/keyimage"+userName+".png&encrypt1="+path+"Userkey\\keyimage"+userName+".png&name="+userName+"");

}

%>

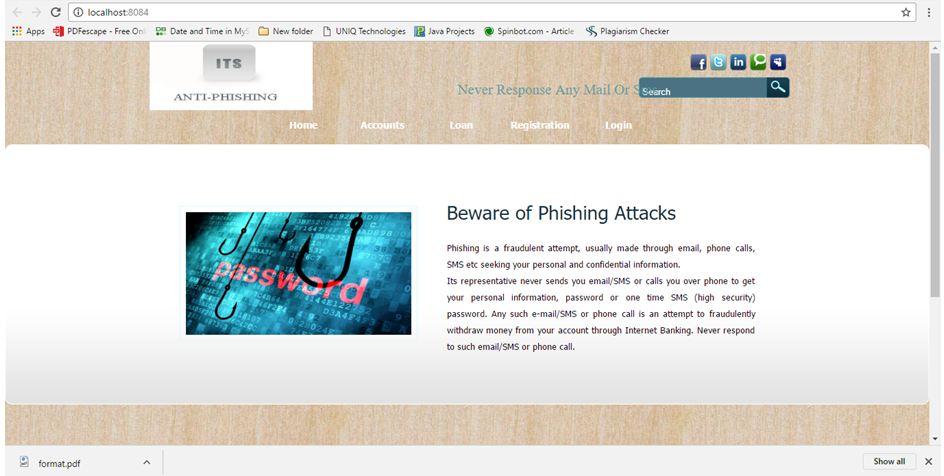
</body>

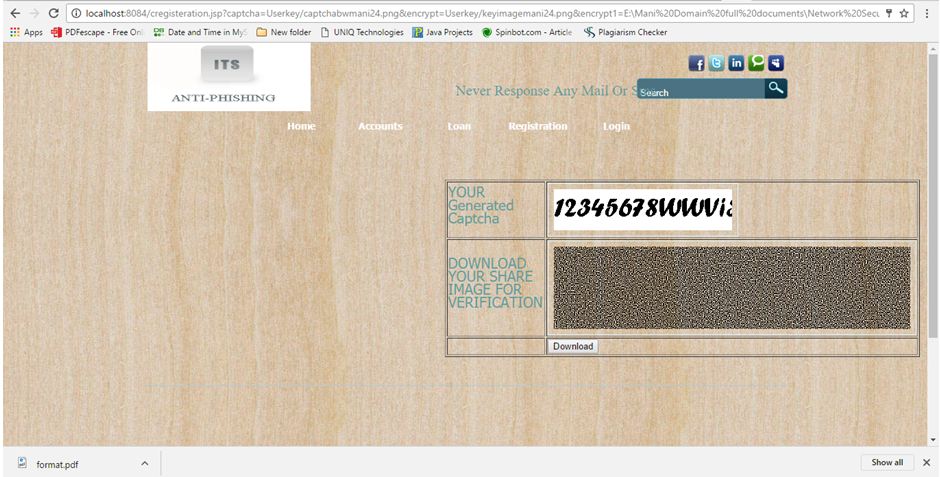
</html>

**CHAPTER 10**

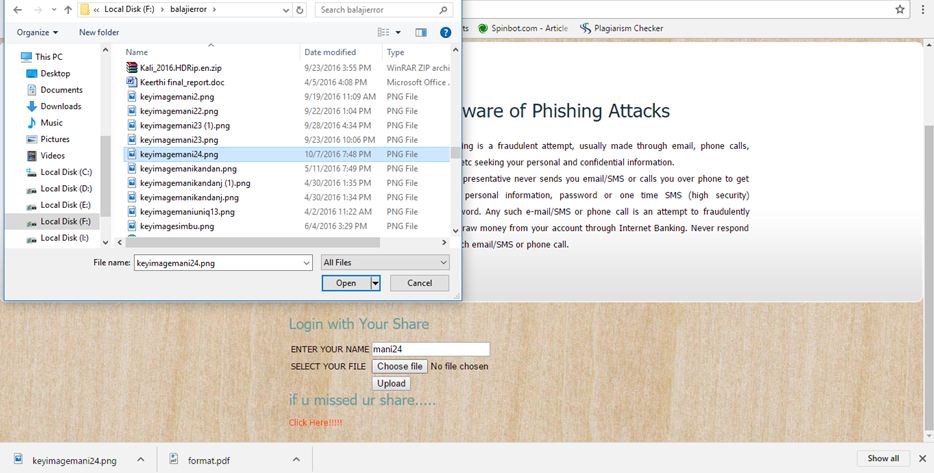
**SCREENSHOTS**

**Home Page:**

**Captcha Generation:**

****

**Login Page:**

****

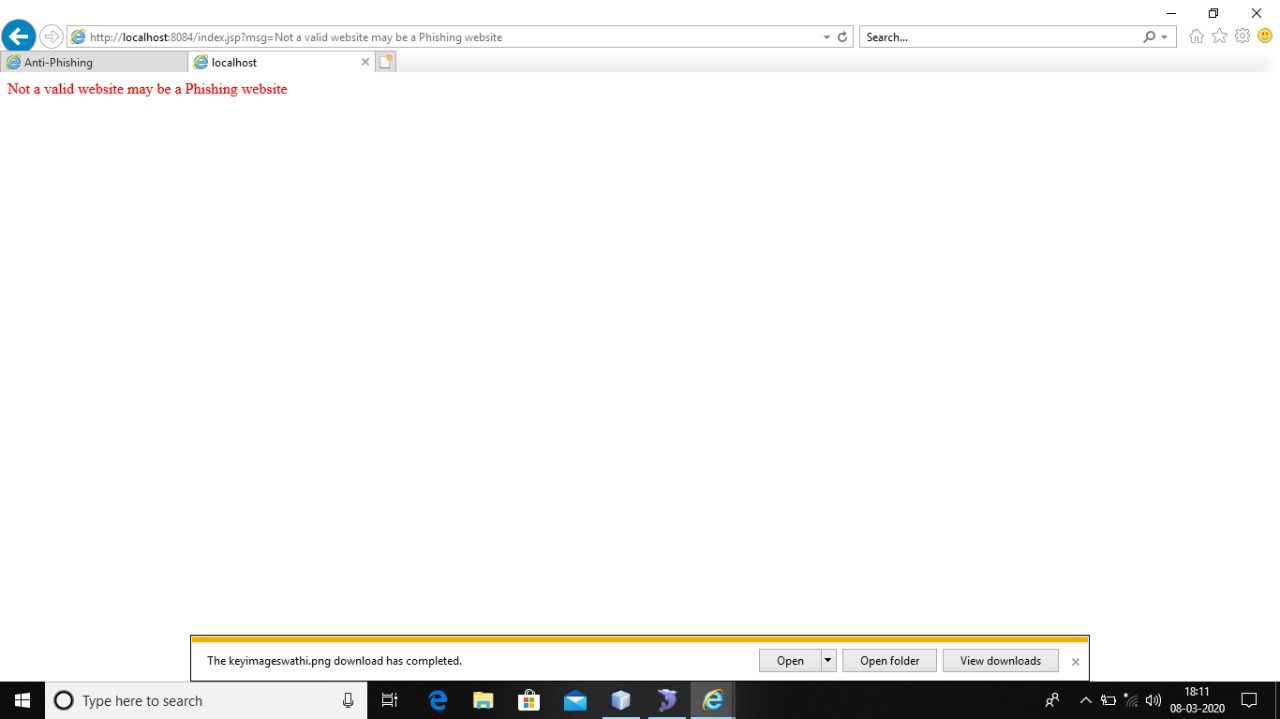
**OTP Generation:**

****

**Online Bank:**

****

**Alert Message:**



**CHAPTER 11**

**CONCLUSION**

Currently phishing attacks are so common because it can attack globally and capture and store the users’ confidential information. This information is used by the attackers which are indirectly involved in the phishing process. Phishing websites as well as human users can be easily identified using our proposed "Anti-phishing framework based on Visual Cryptography”. The proposed methodology preserves confidential information of users. Verifies whether the website is a genuine/secure website or a phishing website. If the website is a phishing website (website that is a fake one just similar to secure website but not the secure website), then in that situation, the phishing website can’t display the image captcha for that specific user (who wants to log in into the website) due to the fact that the image captcha is generated by the stacking of two shares, one with the user and the other with the actual database of the website. The proposed methodology is also useful to prevent the attacks of phishing websites on financial web portal, banking portal, online shopping market.

**FUTURE ENHANCEMENT**

In future we can increase the security by adding many algorithms to encrypt the image. Encryption Phase contains many algorithms like Blowfish, Splitting and Rotating algorithm and (2,2) Visual Cryptography Scheme.

First the “Blowfish Algorithm” is applied to the original image captcha then the image captcha is divided into many blocks and rearranged. After the image captcha blocks are rearranged, the “Splitting and Rotating Algorithm” is applied to the image captcha, and then the rearranged blocks are rotated.

Then the rearranged and rotated blocks are combined. Then (2, 2) VCS scheme is applied to the combined blocks.

This scheme is used to divide the encrypted image captcha into two shares based on white and black pixels. When the two sub pixels are identical blocks it considers as a white pixel. Likewise when the two sub pixels are different the original pixel is considered as black pixel. This VCS scheme adds more complication to the image captcha.

**REFERENCE:**

1. N. Jovanovic, E. Kirda, and C. Kruegel, “Preventing cross site request forgery attacks,” in Proceedings of the Second International Conference on Security and Privacy in CommunicationNetworks. IEEE Computer Society, 2006.
2. M. Shahzad, M. Z. Shafiq, and A. X. Liu, “A large scale exploratory analysis of software vulnerability life cycles,” in ICSE ’12. IEEE Press, 2012, pp. 771–781.
3. Z. Su and G. Wassermann, “The essence of command injection attacks in web applications,” in Proceedings of the 33rd ACM Symposium on Principles of Programming Languages, 2006, pp. 372–382.
4. D. Ray and J. Ligatti, “Defining code-injection attacks,” in POPL ’12. ACM, 2012, pp. 179–190.
5. W. G. Halfond, J. Viegas, and A. Orso, “A classification of SQL-injection attacks and countermeasures,” in Proceedings of the International Symposium on Secure Software Engineering, Mar. 2006.
6. S. Stamm, B. Sterne, and G. Markham, “Reining in the web with content security policy,” in Proceedings of the 19th International Conference on World Wide Web, 2010, pp. 921–930.
7. L. Bauer, S. Cai, L. Jia, P. Timothy, S. Michael, and T. Yuan, “Run-time monitoring and formal analysis of information flows in Chromium,” in NDSS ’15, 2015.
8. D. Wagner and P. Soto, “Mimicry attacks on host-based intrusion detection systems,” in CCS ’02, 2002, pp. 255–264.