

Detecting Mobile Malicious Webpages in Real Time

A "Project Stage-II" Report submitted to JNTU Hyderabad in partial fulfilment of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING (CYBER SECURITY)

Submitted by

M. ABHINAI	21S11A6201
R. LAHARI	21S11A6217
J. SANTOSH GOUD	21S11A6250
T. SRIKANTH	21S11A6256

Under the Guidance of

M.BHARATH KUMAR

B. Tech, M.Tech

Asst.Professor of CSE



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
(CYBER SECURITY)**

MALLA REDDY INSTITUTE OF TECHNOLOGY & SCIENCE

(Approved by AICTE New Delhi and Affiliated to JNTUH)

(Accredited by NBA & NAAC with "A" Grade)

An ISO 9001: 2015 Certified Institution

Maisammaguda, Medchal (M), Hyderabad TS- 500100

MAY 2025

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
(CYBER SECURITY)**

MALLA REDDY INSTITUTE OF TECHNOLOGY AND SCIENCE

(Approved by AICTE New Delhi and Affiliated to JNTUH)

Accredited by NBA and NAAC with “A” Grade

Maisammguda, Medchal (M), Hyderabad-500100, Telangana

MAY 2025



CERTIFICATE

This is to certify that the “Project Stage II” entitled “**DETECTING MOBILE MALICIOUS WEBPAGES IN REAL TIME**” has been submitted by **M. ABHINAI (21S11A6201)**, **R. LAHARI (21S11A6217)**, **J. SANTOSH GOUD (21S11A6250)** and **T. SRIKANTH (21S11A6256)** in partial fulfillment of the requirements for the award of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE IN ENGINEERING & (CYBER SECURITY)**. This record of bonafide work carried out by them under my guidance and supervision. **The result embodied in this major project report has not been submitted to any other University or Institute for the award of any degree.**

Mrs. T. Sai Kumari

HOD-CSE(CS)

M.BHARATH KUMAR

Project Guide

External Examiner

ACKNOWLEDGEMENT

The Mini Project work carried out by our team in the Department of Computer Science and Engineering (CS), Malla Reddy Institute of Technology and Science, Hyderabad. ***This work is original and has not been submitted in part or full for any degree or diploma of any other university.***

We wish to acknowledge our sincere thanks to our project guide **Mr. M. BHARATH KUMAR** Asst.Professor of Computer Science & Engineering (CS) for formulation of the problem, analysis, guidance and her continuous supervision during the course of work.

We acknowledge our sincere thanks to **Dr. Vaka Murali Mohan**, Principal and **T.SAI KUMARI**, Head of the Department and Coordinator, faculty members of CSE(CS) Department for their kind cooperation in making this Mini Project work a success.

We extend our gratitude to **Sri. Ch. Malla Reddy**, Founder Chairman MRGI and **Sri. Ch. Mahender Reddy**, Secretary MRGI, **Dr.Ch. Bhadra Reddy**, President MRGI, **Sri. Ch. Shalini Reddy**, Director MRGI, **Sri. P. Praveen Reddy**, Director MRGI, for their kind cooperation in providing the infrastructure for completion of our Mini Project.

We acknowledge our special thanks to the entire teaching faculty and non-teaching staff members of the Computer Science & Engineering (CS) Department for their support in making this project work a success.

M.ABHINAI	21S11A6201	_____
R. LAHARI	21S11A6256	_____
J. SANTOSH GOUD	21S11A6250	_____
T.SRIKANTH	21S11A6256	_____

INDEX

Chapter	Page no.
ABSTRACT	V
LIST OF FIGURES	VI
1. INTRODUCTION	VII
1.Introduction	1
2. LITERATURE SURVEY	3
3. EXISTING SYSTEM	5
3.1 Existing System	6
3.2 Proposed System	6
4. SYSTEM DESIGN	7
4.1 System Architecture	8
4.1.1 Data Flow Diagram	9
4.2 UML Diagrams	10
4.2.1 Use Case Diagram	11
4.2.2 Class Diagram	12
4.2.3 Sequence Diagram	13
4.2.4 Activity Diagram	14
4.3 System Requirements	15
4.3.1 Hardware Requirements	15
4.3.2 Software Requirements	15
5. INPUT AND OUTPUT DESIGN	16
5.1 Input Design	17
5.2 Output Design	17
6. SYSTEM ENVIRONMENT	18
6.1 Java Technology	19
6.2 JDBC	22
6.2.1 JDBC Goals	23
6.3 J2ME Profiles	25
7. SYSTEM STUDY	27
7.1 Feasibility Study	28
7.2 Economical Feasibility	28
7.3 Technical Feasibility	28
7.4 Social Feasibility	29
8. SYSTEM TESTING	30
8.1 Types of Testing	31
8.1.1 Unit Testing	31
8.1.2 Integration Testing	31
8.1.3 Functional Testing	32
8.1.4 System Test	32
8.1.5 White Box Testing	32
8.1.6 Black Box Testing	32
8.2 Test Strategy and approach	33
9. RESULTS	34
10. CONCLUSION	40

11.REFERENCES	42
12.YUKTHI INNOVATIO CERTIFICATE	44

Detecting Mobile Malicious Webpages in Real Time

ABSTRACT

Mobile specific web pages differ significantly from their desktop counterparts in content, layout and functionality. Accordingly, existing techniques to detect malicious websites are unlikely to work for such web pages. In this paper, we design and implement kayo, a mechanism that distinguishes between malicious and benign mobile webpages. KAYO makes this determination based on static features of a webpage ranging from the number of iframes to the presence of known fraudulent phone numbers. First, we experimentally demonstrate the need for mobile specific techniques and then identify a range of new static features that highly correlate with mobile malicious webpages. We then apply kAYO to a dataset of over 350,000 known benign and malicious mobile webpages and demonstrate 90% accuracy in classification. Moreover, we discover, characterize and report a number of webpages missed by Google Safe Browsing and Virus Total, but detected by kAYO. Finally, we build a browser extension using kAYO to protect users from malicious mobile websites in real-time. In doing so, we provide the first static analysis technique to detect malicious mobile webpages.

LIST OF FIGURES

Fig. No.	Fig. Name	Page. No.
4.1	System Architecture	8
4.1.1	Data Flow Diagram	9
4.2.1	Use Case Diagram	11
4.2.2	Class Diagram	12
4.2.3	Sequence Diagram	13
4.2.4	Activity Diagram	14

LIST OF TABLE

Table Name	Pg.no
Software Requirements	15
Hardware Requirements	15

CHAPTER 1

INTRODUCTION

1. INTRODUCTION

Mobile devices are increasingly being used to access the web. However, in spite of significant advances in processor power and bandwidth, the browsing experience on mobile devices is considerably different. These differences can largely be attributed to the dramatic reduction of screen size, which impacts the content, functionality and layout of mobile webpages. Content, functionality and layout have regularly been used to perform static analysis to determine maliciousness in the desktop space. Features such as the frequency of iframes and the number of redirections have traditionally served as strong indicators of malicious intent. Due to the significant changes made to accommodate mobile devices, such assertions may no longer be true. For example, whereas such behavior would be flagged as suspicious in the desktop setting, many popular benign mobile webpages require multiple redirections before users gain access to content. Previous techniques also fail to consider mobile specific webpage elements such as calls to mobile APIs. For instance, links that spawn the phone's dialer (and the reputation of the number itself) can provide strong evidence of the intent of the page. New tools are therefore necessary to identify malicious pages in the mobile web. In this paper, we present kAYO¹, a fast and reliable static analysis technique to detect malicious mobile webpages. kAYO uses static features of mobile webpages derived from their HTML and JavaScript content, URL and advanced mobile specific capabilities. We first experimentally demonstrate that the distributions of identical static features when extracted from desktop and mobile webpages vary dramatically. We then collect over 350,000 mobile benign and malicious webpages over a period of three months. We then use a binomial classification technique to develop a model for kAYO to provide 90% accuracy and 89% true positive rate. kAYO's performance matches or exceeds that of existing static techniques used in the desktop space. kAYO also detects a number of malicious mobile webpages not precisely detected by existing techniques such as VirusTotal and Google Safe Browsing. Finally, we discuss the limitations of existing tools to detect mobile malicious webpages and build a browser extension based on kAYO that provides realtime feedback to mobile browser users. We make

the following contributions:

- Experimentally demonstrate the differences in the “security features” of desktop and mobile webpages: We experimentally demonstrate that the distributions of static features used in existing techniques (e.g., the number of redirections) are different when measured on mobile and desktop webpages. Moreover, we illustrate that certain features are inversely correlated or unrelated to or non-indicative to a webpage being malicious when extracted from each space. The results of our experiments demonstrate the need for mobile specific techniques for detecting malicious webpages.
- Design and implement a classifier for malicious and benign mobile webpages: We collect over 350,000 benign and malicious mobile webpages. We then identify new static features from these webpages that distinguish between mobile benign and malicious webpages. kAYO provides 90% accuracy in classification and shows improvement of two orders of magnitude in the speed of feature extraction over similar existing techniques. We further empirically demonstrate the significance of kAYO’s features. Finally, we also identify 173 mobile webpages implementing cross-channel attacks, which attempt to induce mobile users to call numbers associated with known fraud campaigns.
- Implement a browser extension based on kAYO: To the best of our knowledge kAYO is the first technique that detects mobile specific malicious webpages by static analysis. Existing tools such as Google Safe Browsing are not enabled on the mobile versions of browsers, thereby precluding mobile users. Moreover, the mobile specific design of kAYO enables detection of malicious mobile webpages missed by existing techniques. Finally, our survey of existing extensions on Firefox desktop browser suggests that there is a paucity of tools that help users identify mobile malicious webpages. To fill this void, we build a Firefox mobile browser extension using kAYO, which informs users about the maliciousness of the webpages they intend to visit in real-time. We plan to make the extension publicly available post publication. We note that we define maliciousness broadly, as is done in the prior literature on the static detection in the desktop space. However, because driveby-downloads are not at all common in the mobile space.

CHAPTER 2
LITERATURE SURVEY

2. LITERATURE SURVEY

1) Detecting malicious websites with low-interaction honeyclients

AUTHORS: A. Ikinici, T. Holz, and F. Freiling. Monkey-spider: Client-side attacks are on the rise: malicious websites that exploit vulnerabilities in the visitor's browser are posing a serious threat to client security, compromising innocent users who visit these sites without having a patched web browser. Currently, there is neither a freely available comprehensive database of threats on the Web nor sufficient freely available tools to build such a database. In this work, we introduce the Monkey-Spider project [Mon]. Utilizing it as a client honeypot, we portray the challenge in such an approach and evaluate our system as a high-speed, Internetscale analysis tool to build a database of threats found in the wild. Furthermore, we evaluate the system by analyzing different crawls performed during a period of three months and present the lessons learned.

2) A guided approach to finding malicious web pages

AUTHORS: L. Invernizzi, S. Benvenuti, M. Cova, P. M. Comparetti, C. Kruegel, and G. Vigna. Evilseed Malicious web pages that use drive-by download attacks or social engineering techniques to install unwanted software on a user's computer have become the main avenue for the propagation of malicious code. To search for malicious web pages, the first step is typically to use a crawler to collect URLs that are live on the Internet. Then, fast prefiltering techniques are employed to reduce the amount of pages that need to be examined by more precise, but slower, analysis tools (such as honey clients). While effective, these techniques require a substantial amount of resources. A key reason is that the crawler encounters many pages on the web that are benign, that is, the "toxicity" of the stream of URLs being analyzed is low. In this paper, we present EVILSEED, an approach to search the web more efficiently for pages that are likely malicious. EVILSEED starts from an initial seed of known, malicious web pages. Using this seed, our system automatically generates search engines queries to identify other malicious pages that are similar or related to the ones in the initial seed. By doing so, EVILSEED leverages the crawling infrastructure of search engines to retrieve URLs that are much more likely to be malicious than a random page on the web. In other words EVILSEED increases the "toxicity" of the input URL stream.

CHAPTER 3
EXISTING SYSTEM

3.1 EXISTING SYSTEM :

The existing system called Content-based and in-depth inspection techniques to detect malicious websites: Dynamic approaches using virtual machines and honey client systems provide deeper visibility into the behavior of a webpage. Therefore, such systems have a very low false positive rate and are more accurate. However, downloading and executing each webpage impacts performance and hinders scalability of dynamic approaches. This performance penalty can be avoided by using static approaches. Static approaches rely on the structural and lexical properties of a webpage and do not execute the content of the webpage. One such technique of detecting malicious URLs is using statistical methods for URL classification based on a URL's lexical and host-based properties. However, URL-based techniques usually suffer from high false positive rates. Using HTML and JavaScript features extracted from a webpage in addition to URL classification helps address this drawback and provides better results. Static approaches avoid performance penalty of dynamic approaches. Additionally, using fast and reliable static approaches to detect benign web pages can avoid expensive in-depth analysis of all web pages.

3.2 PROPOSED SYSTEM:

In the existing system, the system experimentally demonstrates that the distributions of static features used in existing techniques (e.g., the number of redirections) are different when measured on mobile and desktop web pages. Moreover, we illustrate that certain features are inversely correlated or unrelated to or non-indicative to a webpage being malicious when extracted from each space. The results of our experiments demonstrate the need for mobile specific techniques for detecting malicious web pages.

CHAPTER 4
SYSTEM DESIGN

4 SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE:

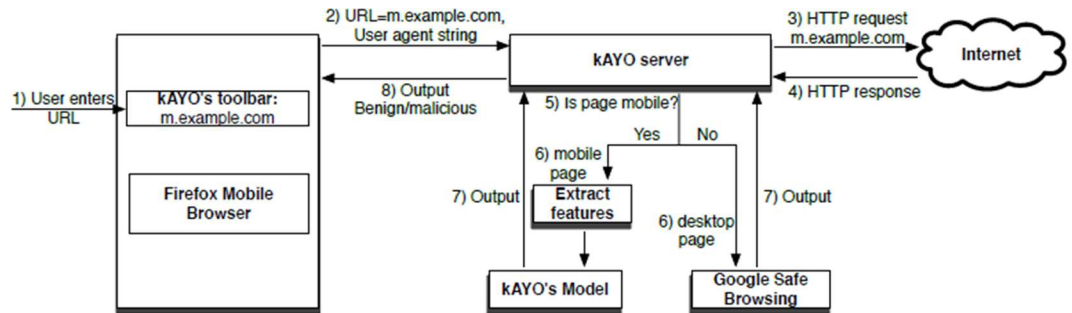


FIG : SYSTEM ARCHITECTURE

Admin

In this module, admin server has to login with valid username and password. After login successful he can do some operations such as -- View all users and authorize and Add Topics with Topic name,URL,Desc(enc),Uses,URL Author, Launched year, attach Topic image, List all topics urls with ranking order by desc and rating order by desc,Set Limit to access malicious WebPages and view, List all Malicious WebPages(if admin name is null,publisher name is Hacker) with attacker names with date and time and IP Address, List all Malicious WebPages accessed user details with date and time and IP Address.

User

In this module, User should register before searching the Website contents. After registration successful the user can login by using valid user name and password. After Login successful the user will do some operations --- View profile, Search WebPages by content keyword - Display only topic name order by description and WebPages and then click on topic name to view all details (increase rank), and recommend to other users, click on web url to display webpage, View all other user recommended Web pages, View Top k web pages ulrs and view the details(increase rank).

4.1.1 DATA FLOW DIAGRAM:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

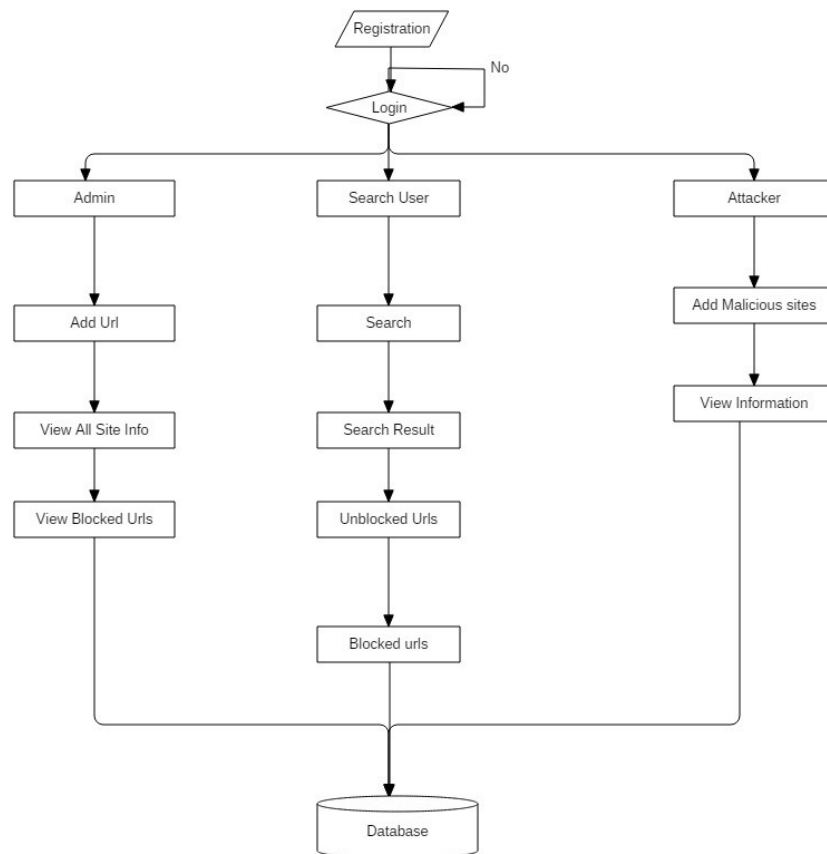


FIG : DATA FLOW DIAGRAM

4.2 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
- 5.** Encourage the growth of OO tools market.

4.2.1 USE CASE DIAGRAM:

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.

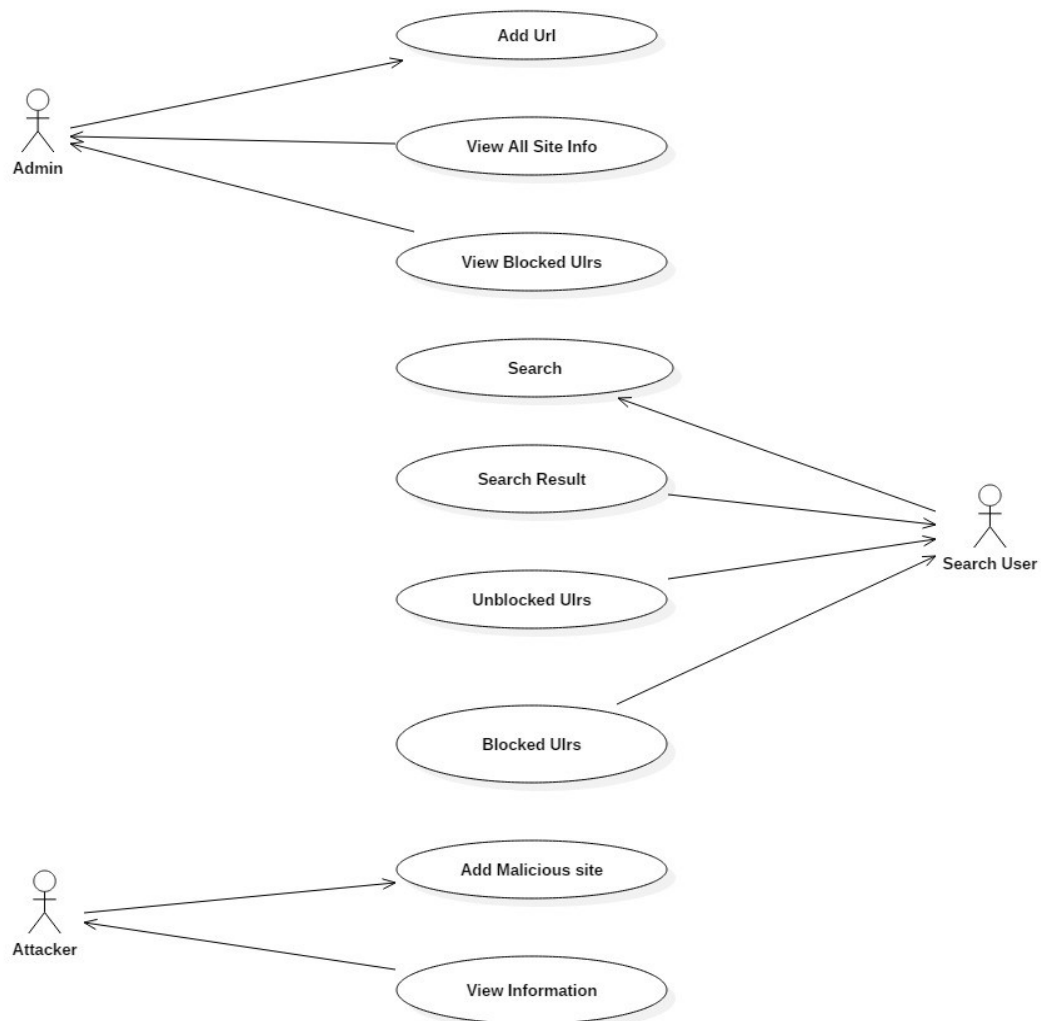


FIG : USE CASE DIAGRAM

4.2.2 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.

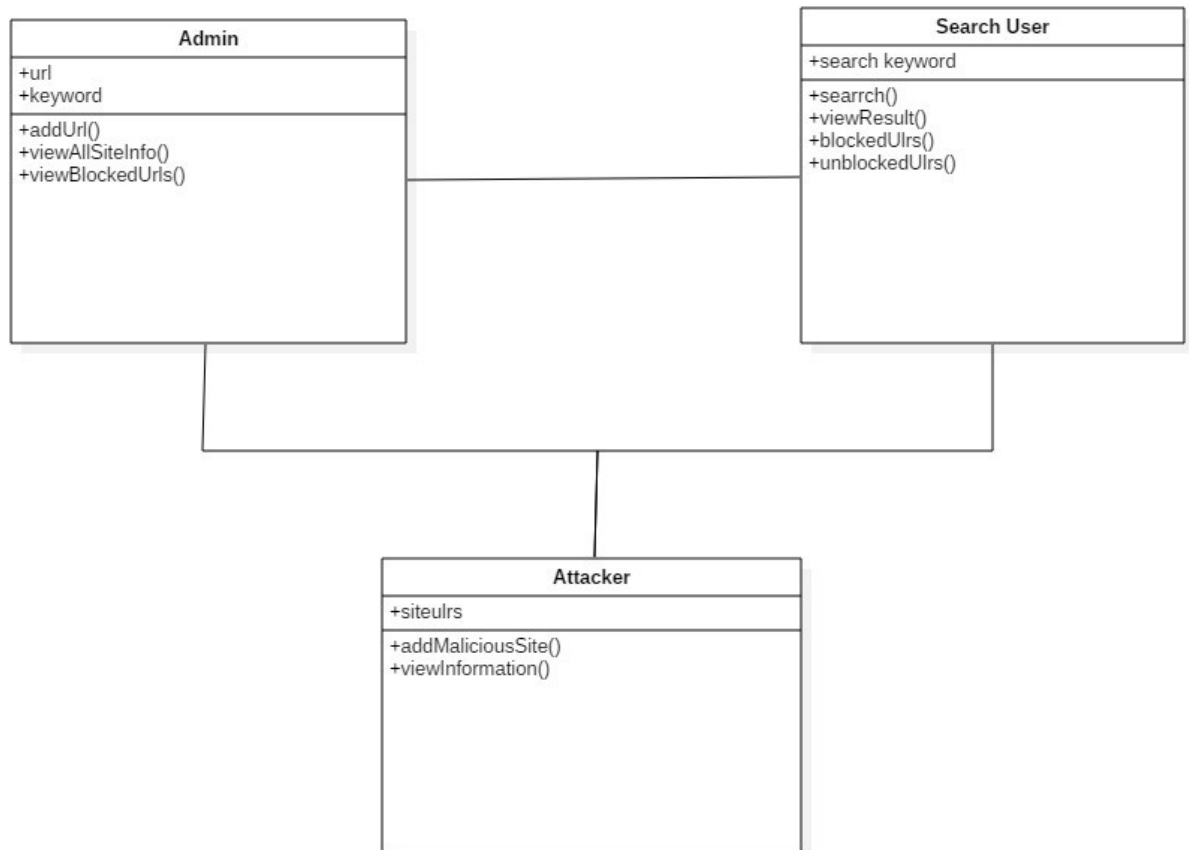


FIG : CLASS DIAGRAM

4.2.3 SEQUENCE DIAGRAM:

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.

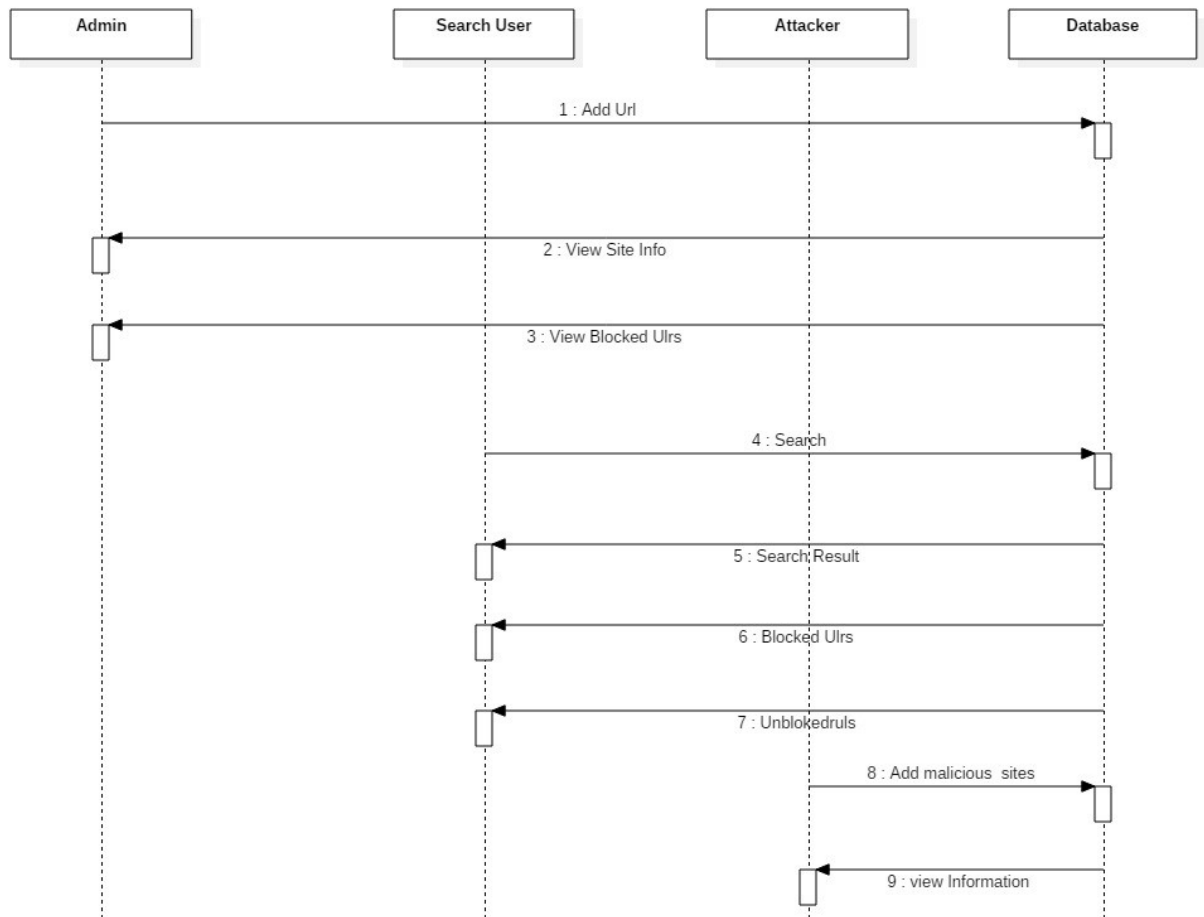


FIG : SEQUENCE DIAGRAM

4.2.4 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

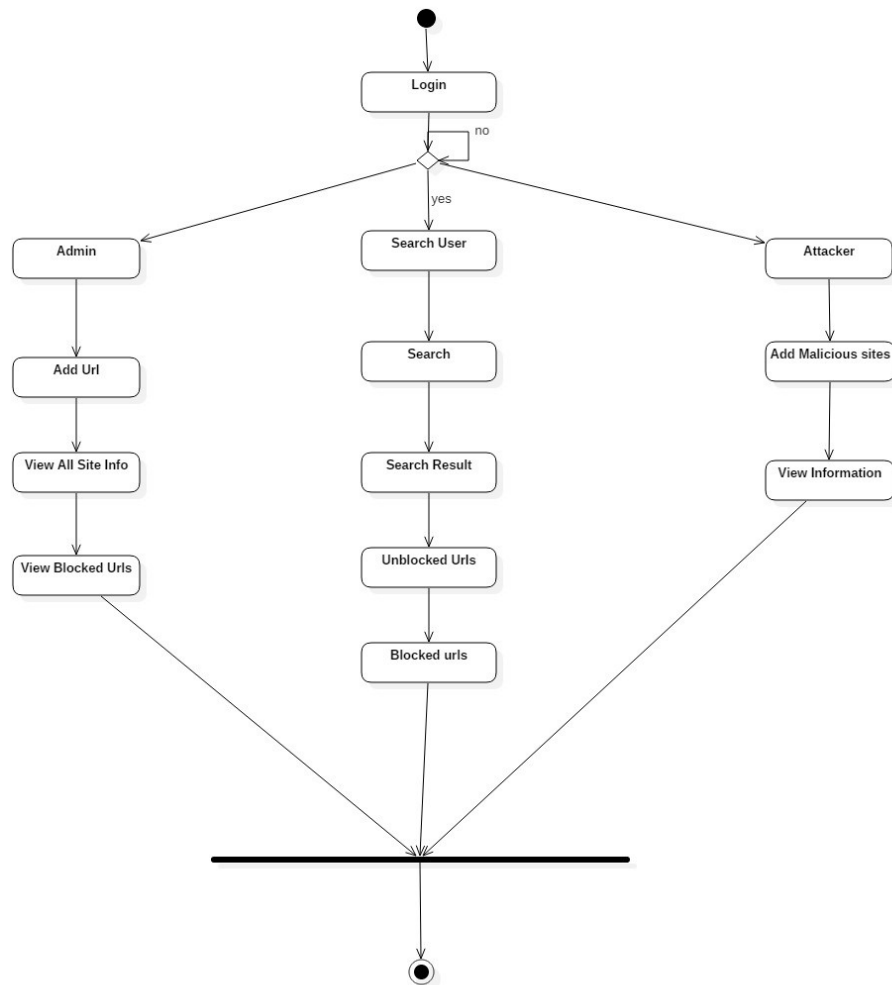


FIG : ACTIVITY DIAGRAM

4.3 SYSTEM REQUIREMENTS:

4.3.1 HARDWARE REQUIREMENTS:

System:	AMD Ryzen 5
Hard Disk:	512 gb
Monitor:	15 VGA Colour.
Mouse:	Logitech.
Ram:	512 Mb.

4.3.2 SOFTWARE REQUIREMENTS:

Operating System:	Windows 11
Programing Language:	Java
DataBase	SQL SERVER 2005

CHAPTER 5
INPUT AND OUTPUT DESIGN

5. INPUT & OUTPUT DESIGN

5.1 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.

5.2 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.
2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system.

CHAPTER 6
SYSTEM ENVIRONMENT

6 SYSTEM ENVIRONMENT

6.1 Java Technology

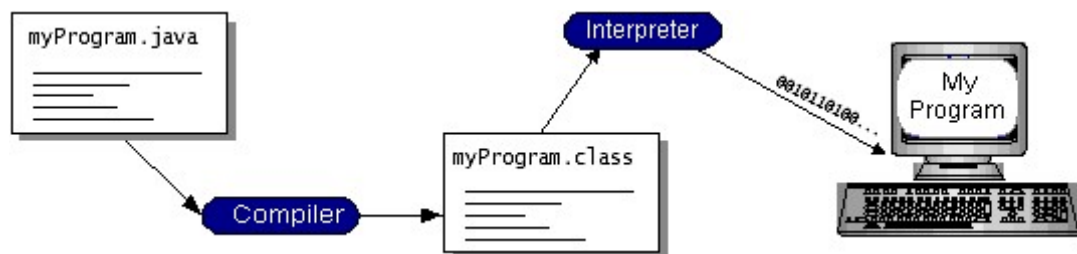
Java technology is both a programming language and a platform.

The Java Programming Language

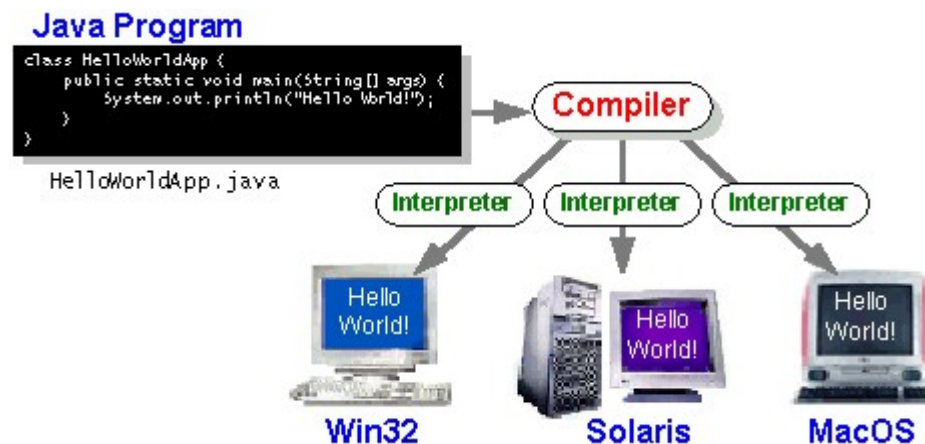
The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

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

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.

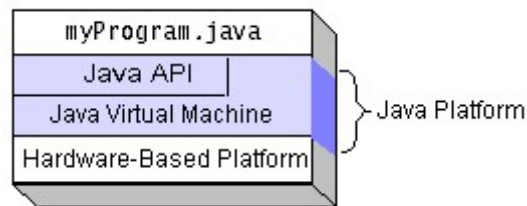
The Java platform has two components:

- 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

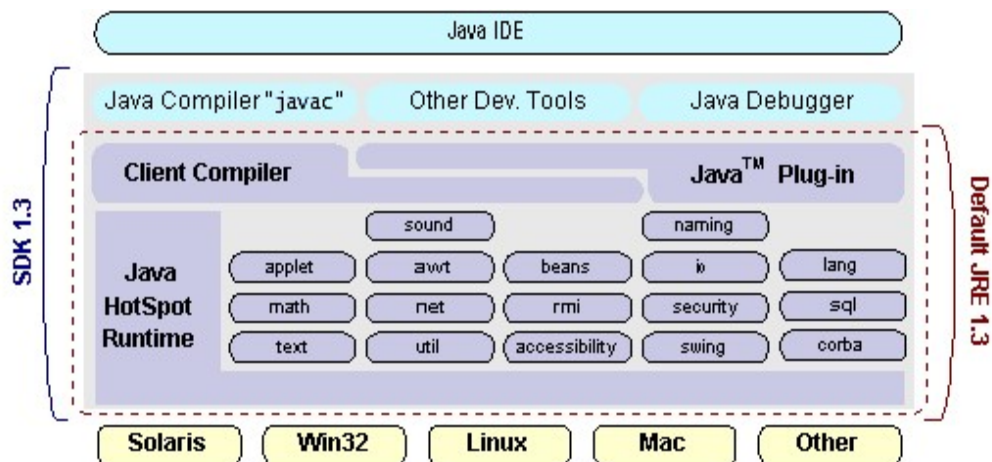
ported onto various hardware-based platforms.

The 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.

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?

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++.
- **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.

6.2 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.

6.2.1 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.

2. 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.

3. 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.

4. 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.

5. 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.

6. 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.

7. Keep the common cases simple

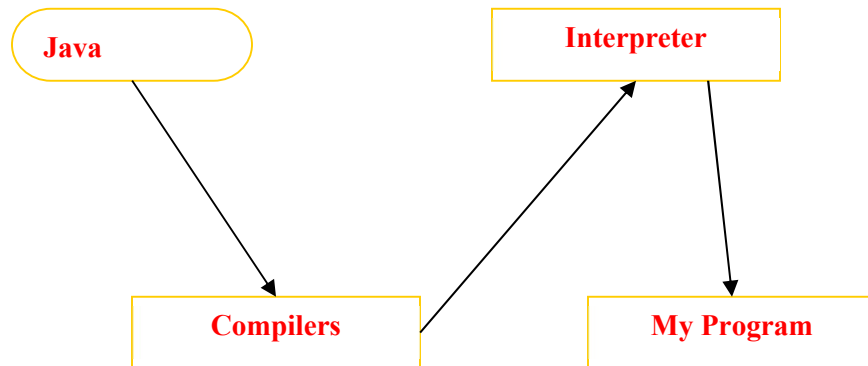
Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible. Finally we decided to proceed the implementation using Java [Networking](#). And for dynamically updating the cache table we go for MS [Access](#) database.

Java has two things: a programming language and a platform.

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

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

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.



6.3 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.

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).

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

CHAPTER 7
SYSTEM STUDY

7 SYSTEM STUDY

7.1 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ◆ ECONOMICAL FEASIBILITY
- ◆ TECHNICAL FEASIBILITY
- ◆ SOCIAL FEASIBILITY

7.2 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

7.3 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available

technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

7.4 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER 8
SYSTEM TESTING

8. SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.1 TYPES OF TESTS

8.1.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

8.1.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

8.1.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

8.1.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

8.1.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings,

structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

8.2 Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

CHAPTER 9

RESULTS

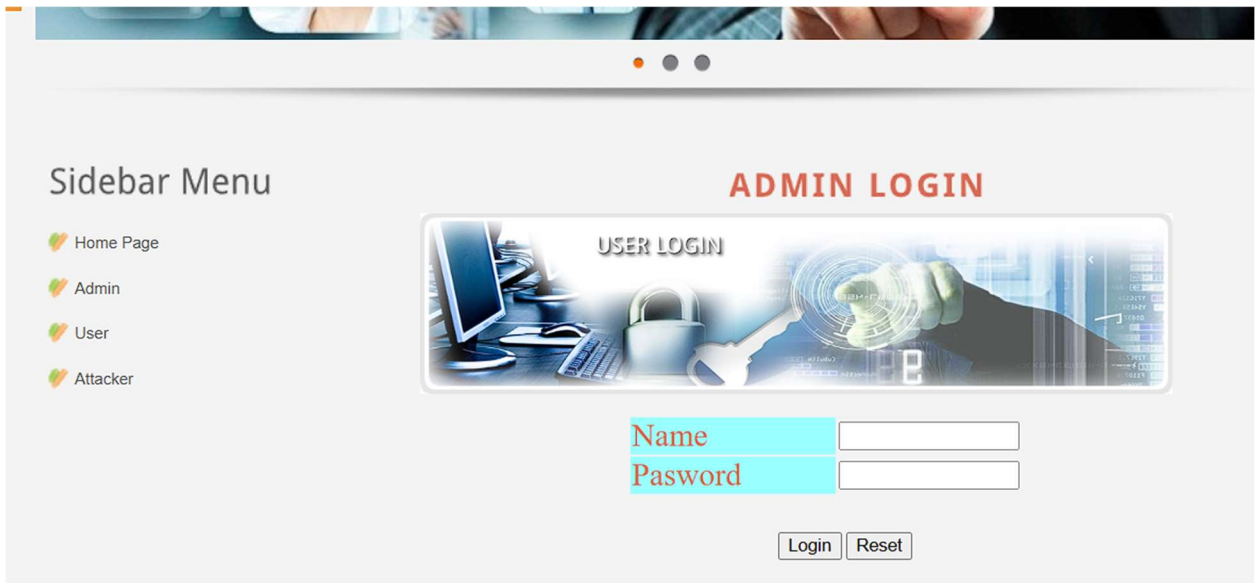
9. RESULTS

9.1 Home Screen



Output Screen: 9.1

9.2 Admin Login screen



Admin Login screen showing the sidebar menu and the login form.

Sidebar Menu

- Home Page
- Admin
- User
- Attacker

ADMIN LOGIN

USER LOGIN

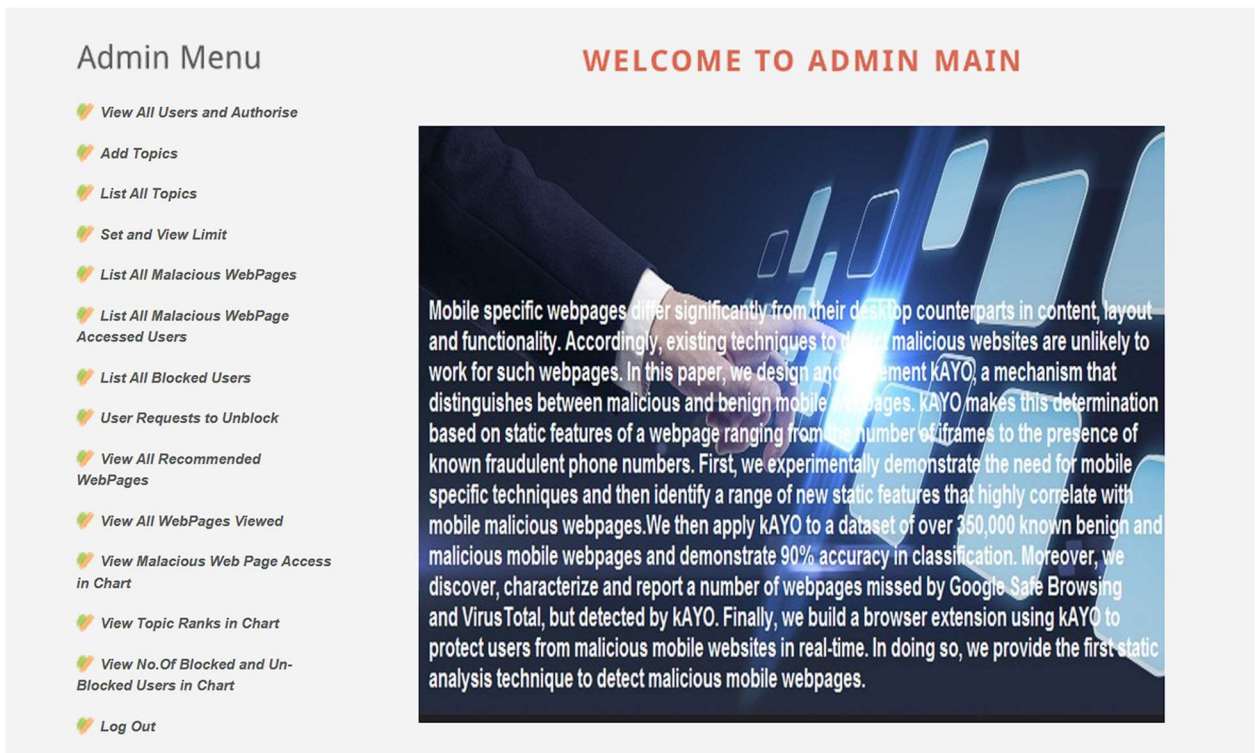
Name

Pasword

Login Reset

Output Screen: 9.2

9.3 Admin login Screen after successful login



Admin Main screen showing the sidebar menu and the welcome message.

Admin Menu

- View All Users and Authorise
- Add Topics
- List All Topics
- Set and View Limit
- List All Malicious WebPages
- List All Malicious WebPage Accessed Users
- List All Blocked Users
- User Requests to Unblock
- View All Recommended WebPages
- View All WebPages Viewed
- View Malicious Web Page Access in Chart
- View Topic Ranks in Chart
- View No.Of Blocked and Un-Blocked Users in Chart
- Log Out

WELCOME TO ADMIN MAIN

Mobile specific webpages differ significantly from their desktop counterparts in content, layout and functionality. Accordingly, existing techniques to detect malicious websites are unlikely to work for such webpages. In this paper, we design and implement kAYO, a mechanism that distinguishes between malicious and benign mobile webpages. kAYO makes this determination based on static features of a webpage ranging from the number of frames to the presence of known fraudulent phone numbers. First, we experimentally demonstrate the need for mobile specific techniques and then identify a range of new static features that highly correlate with mobile malicious webpages. We then apply kAYO to a dataset of over 350,000 known benign and malicious mobile webpages and demonstrate 90% accuracy in classification. Moreover, we discover, characterize and report a number of webpages missed by Google Safe Browsing and VirusTotal, but detected by kAYO. Finally, we build a browser extension using kAYO to protect users from malicious mobile websites in real-time. In doing so, we provide the first static analysis technique to detect malicious mobile webpages.


Output Screen: 9.3

9.4 User login Screen

Sidebar Menu

- Home Page
- Admin
- User
- Attacker

USER LOGIN



The banner features a futuristic theme with a hand using a fingerprint scanner, a padlock, and a computer monitor in the background.

Name

Pasword

Login

Reset

[New User? Register Here](#)


Output Screen: 9.4

9.5 User login Screen after succesfull login

User Menu

- View Profile
- Search Webpages
- View Other User Recommeded WebPages
- View Top K WebPages
- Log Out

WELCOME TO USER MAIN : abhi




The banner shows a person's hands writing on a piece of paper with a pink highlighter. A smartphone and a container of colorful markers are also visible on the desk.

Output Screen: 9.5

9.6 Attacker Screen

Sidebar Menu

- Home Page
- Admin
- User
- Attacker

**Malware**

ADD TOPIC

Topic Name	<input type="text"/>
Url	<input type="text"/>
Description	<input type="text"/>
Uses	<input type="text"/>
Url Author	<input type="text"/>
Launched Year	<input type="text"/>
Attach Image	<input type="button" value="Choose File"/> No file chosen



Output Screen: 9.6

9.7 All Malicious Webpages

Admin Menu

- Admin Main
- Log Out

VIEW ALL MALICIOUS WEBPAGES

Id	Topic Name	Topic Image	URL	Description	Uses	Url Publisher	Launched Year	Up
1	abhi		<input type="text" value="https://www.abhi.com"/>	<input type="text" value="abhi is good boy"/>	<input type="text" value="play boy"/>	<input type="text" value="www.abhi.com"/>	<input type="text" value="2010"/>	<input type="text" value="30/11"/>
2	abhinai		<input type="text" value="https://www.abhi2.com"/>	<input type="text" value="df"/>	<input type="text" value="abhi"/>	<input type="text" value="www.abhi2.com"/>	<input type="text" value="2010"/>	<input type="text" value="11/17"/>

Output Screen: 9.7

9.8 List of blocked users

Admin Menu

Admin Main

Log Out

VIEW ALL BLOCKED USER DETAILS WHO
CROSSED THE ACCESS LIMIT

Id	User Name	Topic Name (Malicious WebPage)	Ip Address	Date
----	-----------	-----------------------------------	------------	------

Back

Output Screen: 9.8

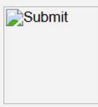
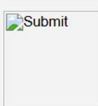
9.9 All Users and permission

Admin Menu

Admin Main

Log Out

VIEW ALL USERS AND AUTHORISE

ID	User Image	User Name	Email	Mobile	DOB	Address	Gender	Pincode
1		abhi	abhi@gmail.com	1234567890	2-11-1980	hyd	Male	34567
2		srikanth	srikanth@gmail.com	1234567890	2-11-1980	hyd	Male	35678

Output Screen: 9.9

CHAPTER 10
CONCLUSION

10. CONCLUSION

Mobile webpages are significantly different than their desktop counterparts in content, functionality and layout. Therefore, existing techniques using static features of desktop webpages to detect malicious behavior do not work well for mobile specific pages. We designed and developed a fast and reliable static analysis technique called kAYO that detects mobile malicious webpages. kAYO makes these detections by measuring 44 mobile relevant features from webpages, out of which 11 are newly identified mobile specific features. kAYO provides 90% accuracy in classification, and detects a number of malicious mobile webpages in the wild that are not detected by existing techniques such as Google Safe Browsing and Virus Total. Finally, we build a browser extension using kAYO that provides real-time feedback to users. We conclude that kAYO detects new mobile specific threats such as websites hosting known fraud numbers and takes the first step towards identifying new security challenges in the modern mobile web.

CHAPTER 11

REFERENCES

11.REFERENCES

- [1] C. Ludl, S. Mcallister, E. Kirda, and C. Kruegel. On the effectiveness of techniques to detect phishing sites. In Proceedings of the 4th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA), 2007.
- [2] Gnu octave: high-level interpreted language. [software/octave/](http://www.gnu.org/software/octave/). <http://www.gnu.org/hphosts>, a community managed hosts file. <http://hphosts.gt500.org/hosts.txt>.
- [3] A. Ikinici, T. Holz, and F. Freiling. Monkey-spider: Detecting malicious websites with low-interaction honeyclients. In Proceedings of Sicherheit, Schutz und Zuverlässigkeit, 2008.
- [4] L. Invernizzi, S. Benvenuti, M. Cova, P. M. Comparetti, C. Kruegel, and G. Vigna. Evilseed: A guided approach to finding malicious web pages. In Proceedings of the 2012 IEEE Symposium on Security and Privacy, 2012.
- [5] P. Kolari, T. Finin, and A. Joshi. Svms for the blogosphere: Blog identification and splog detection. In Proceedings of AAAI Spring Symposium on Computational Approaches to Analysing Weblogs, 2006.
- [6] C. Amrutkar, P. Traynor, and P. C. van Oorschot. Measuring SSL indicators on mobile browsers: Extended life, or end of the road.
- [7] Pindrop phone reputation service. <http://pindropsecurity.com/> phone-fraud-solutions/phone reputation service prs/.
- [8] Scrapy — an open source web scraping framework for python. <http://scrapy.org/>.
- [9] VirusTotal. <https://www.virustotal.com/en/>.
- [10] Google developers: Safe Browsing API. <https://developers.google.com/safe-browsing/>, 2012.

CHAPTER 12
YUKTI CERTIFICATE



INSTITUTION'S INNOVATION COUNCIL
MOE'S INNOVATION CELL



Institute Name:

Malla Reddy Institute of Technology & Science

Title of the Innovation/Prototype:

Detecting Mobile Malicious Webpages in Real Time

Team Lead Name:

Maddela Abhinai

Team Lead Email:

maddela.abhinay2004@gmail.com

Team Lead Phone:

6300364024

Team Lead Gender:

Male

FY of Development:

2024-25

Developed as part of:

Academic Requirement/Study Project

Innovation Type:

Service

TRL LEVEL:

1

Theme:

ICT, cyber-physical systems, Blockchain, Cognitive computing, Cloud computing, AI & ML, Education,

Define the problem and its relevance to today's market / society / industry need:

The problem lies in ensuring the privacy and integrity of scholarly big data while maintaining its accessibility for research purposes. As the volume of academic data grows, there is an increasing need for robust data protection mechanisms that allow data to be searchable and verifiable, while safeguarding sensitive information. This is crucial in today's academic and research environments, where data sharing is essential for innovation, but privacy and accuracy must not be compromised. A secure, transparent system is needed to balance data accessibility with confidentiality in scholarly settings.

Describe the Solution / Proposed / Developed:

In the existing system, the system experimentally demonstrates that the distributions of static features used in existing techniques (e.g., the number of redirections) are different when measured on mobile and desktop web pages. Moreover, we illustrate that certain features are inversely correlated or unrelated to or non-indicative to a webpage being malicious when extracted from each space. The results of our experiments demonstrate the need for mobile specific techniques for detecting malicious web pages

Explain the uniqueness and distinctive features of the (product / process / service) solution:

The real-time detection of mobile malicious webpages is unique due to its ability to identify threats as users browse, preventing harm before it occurs. It combines mobile-specific indicators, such as device fingerprinting and behavioral analysis, with machine learning models to detect emerging threats like phishing or malware. The solution is optimized for mobile devices, ensuring minimal impact on performance and user experience. Leveraging cloud-assisted processing allows for efficient, lightweight detection on resource-constrained devices. Its adaptability across various mobile platforms and browsers, combined with a focus on privacy and user-friendly alerts, makes it a distinctive and powerful security solution.

How your proposed / developed (product / process / service) solution is different from similar kind of product by the competitors if any:

Our solution for detecting mobile malicious webpages in real time stands out by integrating machine learning-driven behavioral analysis and mobile-specific indicators, enabling it to detect both known and emerging threats, such as phishing or malware, with high accuracy. Unlike competitors, we optimize the solution for resource efficiency, ensuring minimal impact on device performance through cloud-assisted processing. Additionally, our system is highly adaptive, offering seamless integration across multiple platforms and browsers. With a focus on privacy, real-time alerts, and a user-friendly experience, our solution provides a more comprehensive and responsive defense against mobile web threats than traditional approaches.

Is there any IP or Patentable Component associated with the Solution?:

No

Has the Solution Received any Innovation Grant/Seedfund Support?:

No

Are there any Recognitions (National/International) Obtained by the Solution?:

No

***Is the Solution Commercialized either through Technology Transfer or Enterprise Development/Startup?:**

No

Had the Solution Received any Pre-Incubation/Incubation Support?:

No

Video URL:

https://drive.google.com/file/d/12vA5xiU4ixH0AjlHkdLyJqGlsGR2_12n/view?usp=sharing

Innovation Photograph:

[View File](#)

Downloaded on: 13-03-2025

This report is electronically generated against Yukti - National Innovation Repository Portal.