A

PROJECT REPORT ON

ACCURATE DECENTRALIZED APPLICATION IDENTIFICATION VIA ENCRYPTED TRAFFIC ANALYSIS USING GRAPH NEURAL NETWORKS

Submitted in partial fulfillment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

By

GOLAMARI MANIKANTA
(21P11F0042)

Under the esteemed guidance of

Dr.K. SAILAJA, MCA, M. Tech, Ph.D

Professor & HOD

DEPARTMENT OF COMPUTER APPLICATIONS

CHADALAWADA RAMANAMMA ENGINEERING COLLEGE

(AUTONOMUS)

Accredited by NAAC with 'A' Grade, Approved by A.I.C.T.E, New Delhi, Affiliated to JNTUA, Anantapur CHADALAWADA

NAGAR, RENIGUNTA ROAD, TIRUPATI-517506

CHITTOOR(Dist.), A.P, INDIA

2022-2023

***

CHADALAWADA RAMANAMMA ENGINEERING COLLEGE
(AUTONOMUS)

DEPARTMENT OF COMPUTER APPLICATIONS
2021-2023

CERTIFICATE

This is to certify that the project work entitled "**ACCURATE DECENTRALIZED APPLICATION IDENTIFICATION VIA ENCRYPTED TRAFFIC ANALYSIS USING GRAPH NEURAL NETWORKS**", is a bonafide work done by GOLAMARI MANIKANTA (21P11F0042), in the Department of "MASTER OF COMPUTER APPLICATIONS", and submitted to **Chadalawada Ramanamma Engineering College (Autonomous), Tirupati** in partial fulfillment of the requirements for the award of the Master of Computer Applications and the project work carried out by them under my guidance during the academic year **2022-2023.**

PROJECT GUIDE	HEAD OF THE DEPARTMENT	PRINCIPAL & DIRECTOR
Dr. K. SAILAJA	**Dr. K. SAILAJA**	**Dr. BHASKAR PATEL**
Professor & HOD	Professor & HOD	CREC(A) & Krishna Teja
Department of Computer Applications	Department of Computer Applications	Educational Institution

Submitted for the project viva-voce examination held on__________________

INTERNAL EXAMINER: EXTERNAL EXAMINER:

ACKNOWLEDGEMENT

I am thankful to my guide **Dr.K.SAILAJA Professor & HOD** of Chadalawada Ramanamma Engineering College, Tirupati, for her valuable guidance and encouragement. His helping attitude and suggestions have helped me in the successful completion of the project**.**

I hereby wish to express my deep sense of gratitude to **Dr. K. SAILAJA, HOD, Department of Computer Applications** for all provisions made and for her constant encouragement throughout the work.

I express my sincere thanks to our beloved **principal & Director Dr. BHASKER PATEL M.Tech, Ph.D** for providing all the facilities and support in completing my project report successfully.

I sincerely thank our **Chairperson Dr. CHADALAWADA. KRISHNA MURTHY** for providing necessary infrastructure and resources for the accomplishment of the topic at **Chadalawada Ramanamma Engineering College (Autonomous), Tirupati.**

Successful completion of any project cannot be done without proper support and encouragement. My sincere thanks to the Management for providing all the necessary facilities during the Course of my study.

I would like to thank my parents and friends, who have the greatest contributions in all my achievements, for the great care and blessings in making me successful in all my endeavors.

I would like to express my deep gratitude to all those who helped me directly or indirectly to transform an idea into my working project.

GOLAMARI MANIKANTA

(21P11F0042)

ABSTRACT

Decentralized Applications (DApps) are increasingly developed and deployed on blockchain platforms such as Ethereum. DApp fingerprinting can identify users' visits to specific DApps by analyzing the resulting network traffic, revealing much sensitive information about the users, such as their real identities, financial conditions and religious or political preferences. DApps deployed on the same platform usually adopt the same communication interface and similar traffic encryption settings, making the resulting traffic less discriminative. Existing encrypted traffic classification methods either require hand- crafted and fine-tuning features or suffer from low accuracy. It remains a challenging task to conduct DApp fingerprinting in an accurate and efficient way. In this paper, we present GraphDApp, a novel DApp fingerprinting method using Graph Neural Networks (GNNs). We propose a graph structure named Traffic Interaction Graph (TIG) as an information-rich representation of encrypted DApp flows, which implicitly reserves multiple dimensional features in bidirectional client-server interactions. Using TIG, we turn DApp fingerprinting into a graph classification problem and design a powerful GNN-based classifier. We collect real-world traffic datasets from 1,300 DApps with more than 169,000 flows. The experimental results show that GraphDApp is superior to the other state-of-the-art methods in terms of classification accuracy in both closed- and open-world scenarios. In addition, GraphDApp maintains its high accuracy when being applied to the traditional mobile application classification.

Index Terms— Decentralized applications, encrypted traffic classification, deep learning, graph neural networks, blockchain.

TABLE OF CONTENTS

S.NO	DESCRIPTION	PAGE NO
1	**1. INTRODUCTION**	1 -2
2	**2.LITERATURE SURVEY** 2.1 Different authors discussions	3-4
3	**3.EXISTING SYSTEM** 3.1 EXISTING SYSTEM 3.1.1 DISADVANTAGES OF EXISTING SYSTEM	5
4	**4.PROPOSED SYSTEM** 4.2 PROPOSED SYSTEM 4.2.1 ADVANTAGES OF PROPOSEDSYSTEM	6
5	5.**MODULE DISCRIPTION** 5.1 MODULES EXPLANATION	7-8
6	**6.PROJECT DESIGN** 6.1 SYSTEM ARCHITECTURE 6.2 UML DIAGRAM 6.3 PROJECT REQUIREMENTS	9-17
7	7.**PROJECT IMPLEMENTATION** 7.1 TECHNOLOGIES USED 7.2 SAMPLE CODE 7.3 SYSTEM TESTING	18-52
8	**8. SCREENSHOTS**	55-64
9	**9.CONCLUSION** 9.1 CONCLUSION 9.2 FUTURE ENHANCEMENT	65
10	10. **REFERENCES**	66-67

1. INTRODUCTION

With the recent development of blockchain technology, the number of decentralized applications (DApps) are increasing dramatically. Unlike regular mobile or web applications deployed on centralized servers, DApps run their backend codes on a decentralized peer-to-peer (P2P) network without the control of a single entity. Among the blockchain platforms on which DApps are hosted, such as EOS, NEO, Stellar, and Tron, we focus on Ethereum [1] in this paper, as it attracts the largest developer community, where more than 3,200 DApps are deployed on Ethereum and the number of daily active users has reached nearly 110,000 by 2020 [2]. The prosperity of DApps is attributed greatly to their resistance to censorship [3], [6].

Compared with the traditional mobile or web applications, DApps are completely open sourced and autonomously managed without a single authority to manage all codes and data. Once a piece of information in DApp is added to the underlying blockchain, it gets stored permanently and thus cannot be removed or modified. In addition, blockchain technology naturally provides anonymity to each participant and thus can potentially protect identity privacy of DApp users. These features make DApps trustable to censorship and governance. Although these service-level enchantments provide users with safety and security, the network traffic generated when individual users visit DApps can still reveal plenty of sensitive information of users.

Passive adversaries (e.g., campus network administrators, residential network service providers, or malicious eavesdroppers) could conduct DApp fingerprinting to identify the specific DApps that a user visits by analyzing the resulting network traffic. An adversary can infer users' financial conditions from their usage of gambling DApps or learn their religious preferences and political views from their visits to social DApps. DApp fingerprinting can also be conducted by governors to deanonymize DApp users or even block access attempts to certain DApps without affecting visits to the rest DApps on the same platform (e.g., Ethereum).

2.LITERATURE SURVEY

2.1Different authors discussions

[1] A global, longitudinal Internet censorship measurement platform

Researchers have studied Internet censorship for nearly as long as attempts to censor contents have taken place. Most studies have however been limited to a short period of time and / or a few countries; the few exceptions have traded off detail for breadth of coverage. Collecting enough data for a comprehensive, global, longitudinal perspective remains challenging. In this work, we present ICLab, an Internet measurement platform specialized for censorship research. It achieves a new balance between breadth of coverage and detail of measurements, by using commercial VPNs as vantage points distributed around the world. ICLab has been operated continuously since late 2016. It can currently detect DNS manipulation and TCP packet injection, and overt "block pages" however they are delivered. ICLab records and archives raw observations in detail, making retrospective analysis with new techniques possible. At every stage of processing, ICLab seeks to minimize false positives and manual validation. Within 53,906,532 measurements of individual web pages, collected by ICLab in 2017 and 2018, we observe blocking of 3,602 unique URLs in 60 countries. Using this data, we compare how different blocking techniques are deployed in different regions and/or against different types of content. Our longitudinal monitoring pinpoints changes in censorship in India and Turkey concurrent with political shifts, and our clustering techniques discover 48 previously unknown block pages. ICLab's broad and detailed measurements also expose other forms of network interference, such as surveillance and malware injection.

[2] Adaptive encrypted traffic fingerprinting with bidirectional dependence

Recently, network traffic analysis has been increasingly used in various applications including security, targeted advertisements, and network management. However, data encryption performed on network traffic poses a challenge to these analysis techniques. In this paper, we present a novel method to extract characteristics from encrypted traffic by utilizing data dependencies that occur over sequential transmissions of network packets. Furthermore, we explore the temporal nature of encrypted traffic and introduce an adaptive model that considers changes in data content over time. We evaluate our analysis on two packet encrypted applications: website fingerprinting and mobile application (app) fingerprinting. Our evaluation shows how the proposed approach outperforms previous worksespecially in the open-world scenario and when defense mechanisms are considered.

[3] “An optimal lower bound on the number of variables for graph identifications

In this paper we show that Ω(*n*) variables are needed for first-order logic with counting to identify graphs on*n* vertices. The*k*-variable language with counting is equivalent to the (*k*−1)-dimensional Weisfeiler-Lehman method. We thus settle a long-standing open problem. Previously it was an open question whether or not 4 variables suffice. Our lower bound remains true over a set of graphs of color class size 4. This contrasts sharply with the fact that 3 variables suffice to identify all graphs of color class size 3, and 2 variables suffice to identify almost all graphs. Our lower bound is optimal up to multiplication by a constant because*n* variables obviously suffice to identify graphs on*n* vertices.

[4] Analyzing Android encrypted network traffic to identify user actions

Mobile devices can be maliciously exploited to violate the privacy of people. In most attack scenarios, the adversary takes the local or remote control of the mobile device, by leveraging a vulnerability of the system, hence sending back the collected information to some remote web service. In this paper, we consider a different adversary, who does not interact actively with the mobile device, but he is able to eavesdrop the network traffic of the device from the network side (e.g., controlling a Wi-Fi access point). The fact that the network traffic is often encrypted makes the attack even more challenging. In this paper, we investigate to what extent such an external attacker can identify the specific actions that a user is performing on her mobile apps. We design a system that achieves this goal using advanced machine learning techniques. We built a complete implementation of this system, and we also run a thorough set of experiments, which show that our attack can achieve accuracy and precision higher than 95%, for most of the considered actions. We compared our solution with the three state-of-the-art algorithms, and confirming that our system outperforms all these direct competitors.

[5] “A Web traffic analysis attack using only timing information

We introduce an attack against encrypted Web traffic that makes use only of packet timing information on the uplink. This attack is therefore impervious to existing packet padding defenses. In addition, unlike existing approaches, this timing-only attack does not require the knowledge of the start/end of web fetches and so is effective against traffic streams. We demonstrate the effectiveness of the attack against both wired and wireless traffic, achieving mean success rates in excess of 90%. In addition to being of interest in its own right, this timing-only attack serves to highlight deficiencies in existing defenses and so to areas where it would be beneficial for virtual private network (VPN) designers to focus further attention.

3. EXISTING SYSTEM

3.1 EXISTING SYSTEM

- It remains a challenging task to accurately and efficiently identify DApps via traffic analysis. Unlike regular mobile applications or websites, DApps deployed on Ethereal implement the same frontend interface, adopt similar settings of SSL/TLS protocols, and share the same decentralized block chain network for running their backend codes and managing their data.
- As a result, the traffic of different DApps has lots of common features, leading to low accuracy of existing fingerprinting methods employing SSL/TLS packet flags or packet length statistics.
- Efficiency is also of importance to construct a DApp fingerprinting method. Existing studies that employ machine learning classifiers usually resort to hand-crafted features (e.g., fusing multiple dimensional features or even require compute-intensive feature processing (e.g., dynamic time warping of packet time series.

3.1.1 DISADVANTAGES OF EXISTING SYSTEM

Efficiency is also of importance to construct a DApp fingerprinting method. Existing studies that employ machine learning classifiers usually resort to hand-crafted features (e.g., fusing multiple dimensional features, or even require compute-intensive feature processing (e.g., dynamic time warping of packet time series. It is more desirable to simplify the sophisticated feature selection process and reduce the computational overhead for training classifiers.
The classifier can automatically extract features from input TIGs and distinguish different graph structures by mapping them to different representations in the embedding space. We conduct extensive experiments using real-world datasets to evaluate the performance of GraphDApp in the closed- and open-world settings.
Compared with our previous work the novelty of this paper includes the new graph-based representation of encrypted flows and the new classifier using GNNs. We further extend performance evaluation by conducting a comprehensive comparison with the state-of-the-art on real-world traffic datasets.

4. PROPOSED SYSTEM

4.1 PROPOSED SYSTEM

1) We propose Traffic Interaction Graph (TIG) to represent each individual encrypted flow, where vertices in a TIG represent packets and edges represent the packet-level interactions between a pair of client and server. We alsoprovide quantitative measures to demonstrate the advantages of representing flows using TIGs over the traditional packet length sequence.
2) We design GraphDApp, a powerful GNN-based classifier using Multi-Layer Perceptions (MLPs) and a fully connected layer. It maps TIGs of different DApp flows to different representations in the embedding space and does not require hand-crafted features so that the classification can be conducted in an effective and accurate way.
3) We collect real-world traffic datasets from 1,300 DApps on Ethereal with more than 169,000 flows. We demonstrate the accuracy and efficiency of GraphDApp in closed- and open-world settings. Compared with the state-of-the-art methods, GraphDApp has the highest classification accuracy with the shortest training time. In addition, it is also applicable to traditional mobile application classification.

4.2.1 ADVANTAGES OF PROPOSED SYSTEM

- We propose Traffic Interaction Graph (TIG) to represent each individual encrypted flow, where vertices in a TIG represent packets and edges represent the packet-level interactions between a pair of client and server. We alsoprovide quantitative measures to demonstrate the advantages of representing flows using TIGs over the traditional packet length sequence.
- DApp fingerprinting is a traffic analysis attack that aims at identifying users' visits to a certain collection of monitored DApps. To be able to perform this attack, an adversary needs to observe the traffic when a victim is visiting DApps.
- Following the common assumption on threat model in existing literature, we assume that the potential adversary is local and passive, which means that the adversary exists in the local network of victims (e.g., campus network administrators) and performs merely traffic collection without any active attacks such as traffic hijacking.

5.MODULE DISCRIPTION

5.1 MODULAR EXPLANATION

Decentralized applications

Decentralized Applications (DApps) are increasingly developed and deployed on block chain platforms such as Ethereum. DApp fingerprinting can identify users' visits to specific DApps by analyzing the resulting network traffic, revealing much sensitive information about the users, such as their real identities, financial conditions and religious or political preferences. DApps deployed on the same platform usually adopt the same communication interface and similar traffic encryption settings, making the resulting traffic less discriminative.

Once a piece of information in DApp is added to the underlying block chain, it gets stored permanently and thus cannot be removed or modified. In addition, blockchain technology naturally provides anonymity to each participant and thus can potentially protect identity privacy of DApp users.

Encrypted traffic classification

With the rapidly emerging encryption techniques for network traffic, the classification of encrypted traffic has increasingly become significantly important in network management and security. In this paper, we propose a novel deep neural network that combines both the convolutional network and the recurrent network to improve the accuracy of the classification results. The convolutional network is used to extract the packet features for a single packet.

The recurrent network is trained to pick out the flow features based on the inputs of the packet features of any three consecutive packets in a flow. The proposed model surpasses the existing studies which ask for the first packets of a flow, and it provides more flexibility in real practice. We compare our model with the existing work under deep learning for encrypted traffic classification, based on the public dataset.

Deep learning

With TIG, we convert the encrypted flow classification problem into a graph classification problem. As deep learning has shown its superiority over traditional machine learning techniques, we design a GNN-based classifier using multi-layer perceptions.

The classifier can automatically extract features from input TIGs and distinguish different graph structures by mapping them to different representations in the embedding space.

We conduct extensive experiments using real-world datasets to evaluate the performance of GraphDApp in the closed- and open-world settings. closed- and open-world settings.

Graph neural networks

In this proposed, we present GraphDApp, a novel DApp fingerprinting method using Graph Neural Networks (GNNs). We propose a graph structure named Traffic Interaction Graph (TIG) as an information-rich representation of encrypted DApp flows, which implicitly reserves multiple dimensional features in bidirectional client-server interactions. Using TIG, we turn DApp fingerprinting into a graph classification problem and design a powerful GNN-based classifier. We collect real-world traffic datasets from 1,300 DApps with more than 169,000 flows.

In this project, we extend our previous work and propose GraphDApp, a DApp fingerprinting method using Graph Neural Networks (GNNs). We are motivated by an observation that each traffic flow, consisting of a series of packets resulting from client-server interactions, can be represented by an information-rich graph structure named Traffic Interaction Graph (TIG).

.

6.PROJECT DESIGN

6.1 ARCHITECTURE DIAGRAM

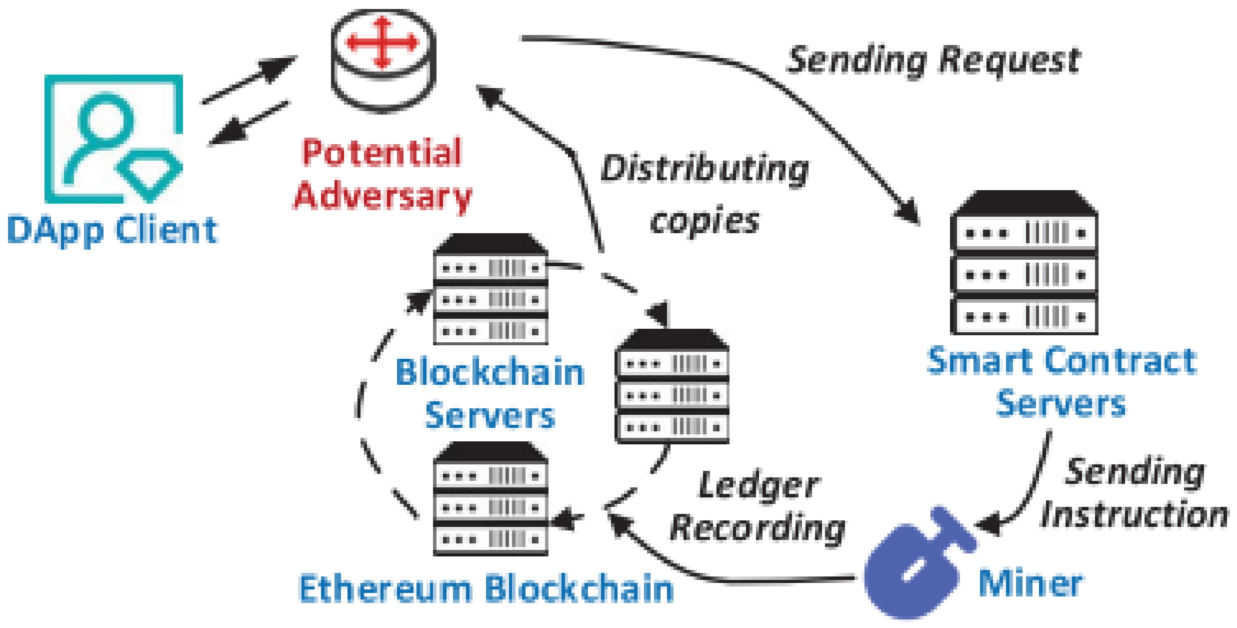


Fig 6.1 System Architecture

6.2 UML DIAGRAMS

The Unified Modeling Language (UML) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complexsystems.1 The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects. Using the UML helps project teams communicate, explore potential designs, and validate the architectural design of the software.

6.2.1 CLASS Diagram:

A class diagram is a type of UML diagram that represents the structure of a system or software application by showing the classes, their attributes, methods, and the relationships between them. It is used to visualize the conceptual design of a system and to describe the objects and their interactions.

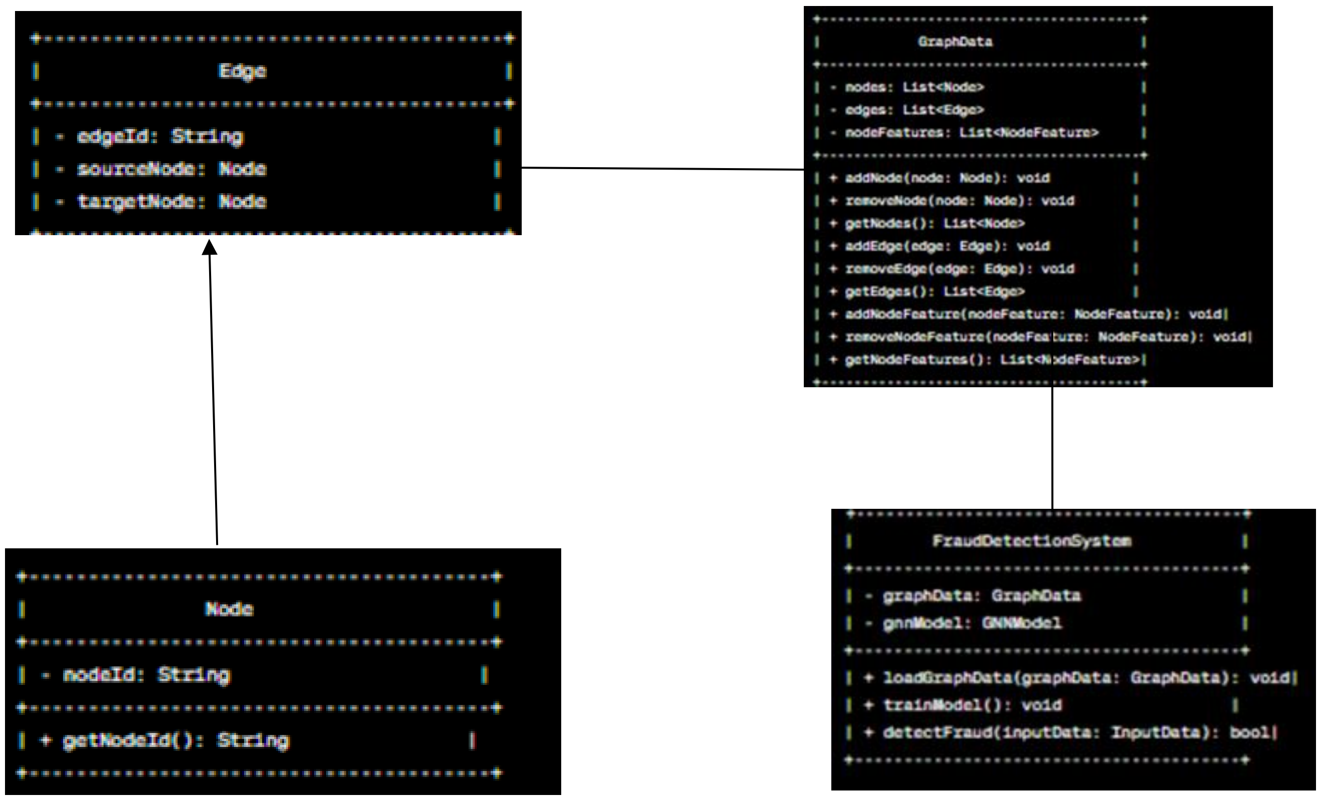

Edge
- edgeId: String
- sourceNode: Node
- targetNode: Node
GraphData
- nodes: List<Node>
- edges: List<Edge>
- nodeFeatures: List<NodeFeature>
+ addNode(node: Node): void
+ removeNode(node: Node): void
+ getNodes(): List<Node>
+ addEdge(edge: Edge): void
+ removeEdge(edge: Edge): void
+ getEdges(): List<Edge>
+ addNodeFeature(nodeFeature: NodeFeature): void
+ removeNodeFeature(nodeFeature: NodeFeature): void
+ getNodeFeatures(): List<NodeFeature>
Node
- nodeId: String
+ getNodeId(): String
FraudDetectionSystem
- graphData: GraphData
- gnnModel: GNNModel
+ loadGraphData(graphData: GraphData): void
+ trainModel(): void
+ detectFraud(inputData: InputData): bool

Fig:6.2.1 **CLASS Diagram**

6.2.2 Sequence Diagram:

A sequence diagram is a type of interaction diagram because it describes howandin what order group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios. A sequence diagram is the most commonly used interaction diagram. Interaction diagram–An interaction diagram is used to show the interactive behavior of a system. Since visualizing the interactions in a system can be a cumbersome task, we use different types of interaction diagrams to capture various features and aspects of interaction in a system.

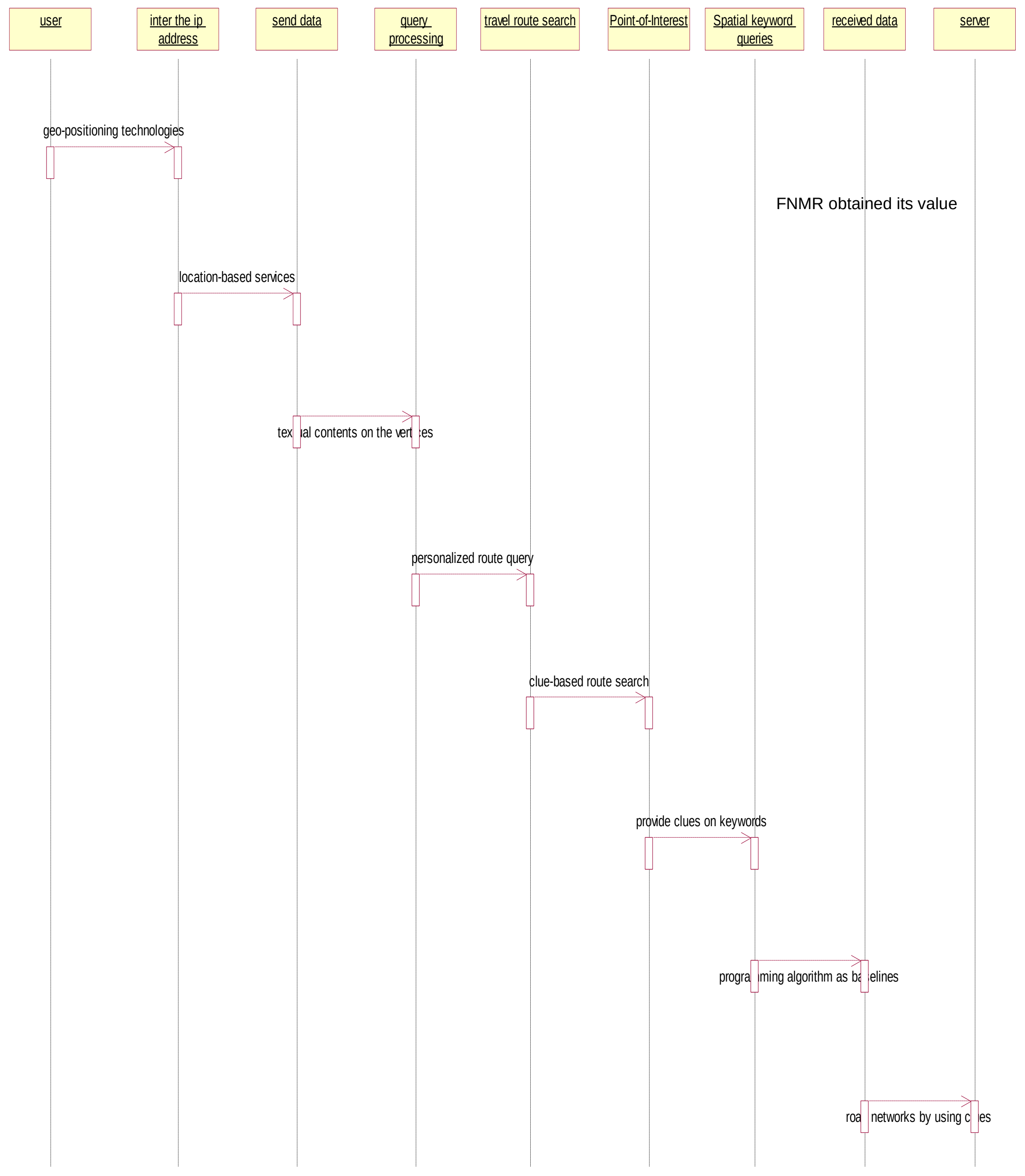


Fig 6.2.2 sequence diagram

6.2.3 USE CASE diagram:

A class diagram is a type of UML diagram that represents the structure of a system or software application by showing the classes, their attributes, methods, and the relationships between them. It is used to visualize the conceptual design of a system and to describe the objects and their interactions within the system.

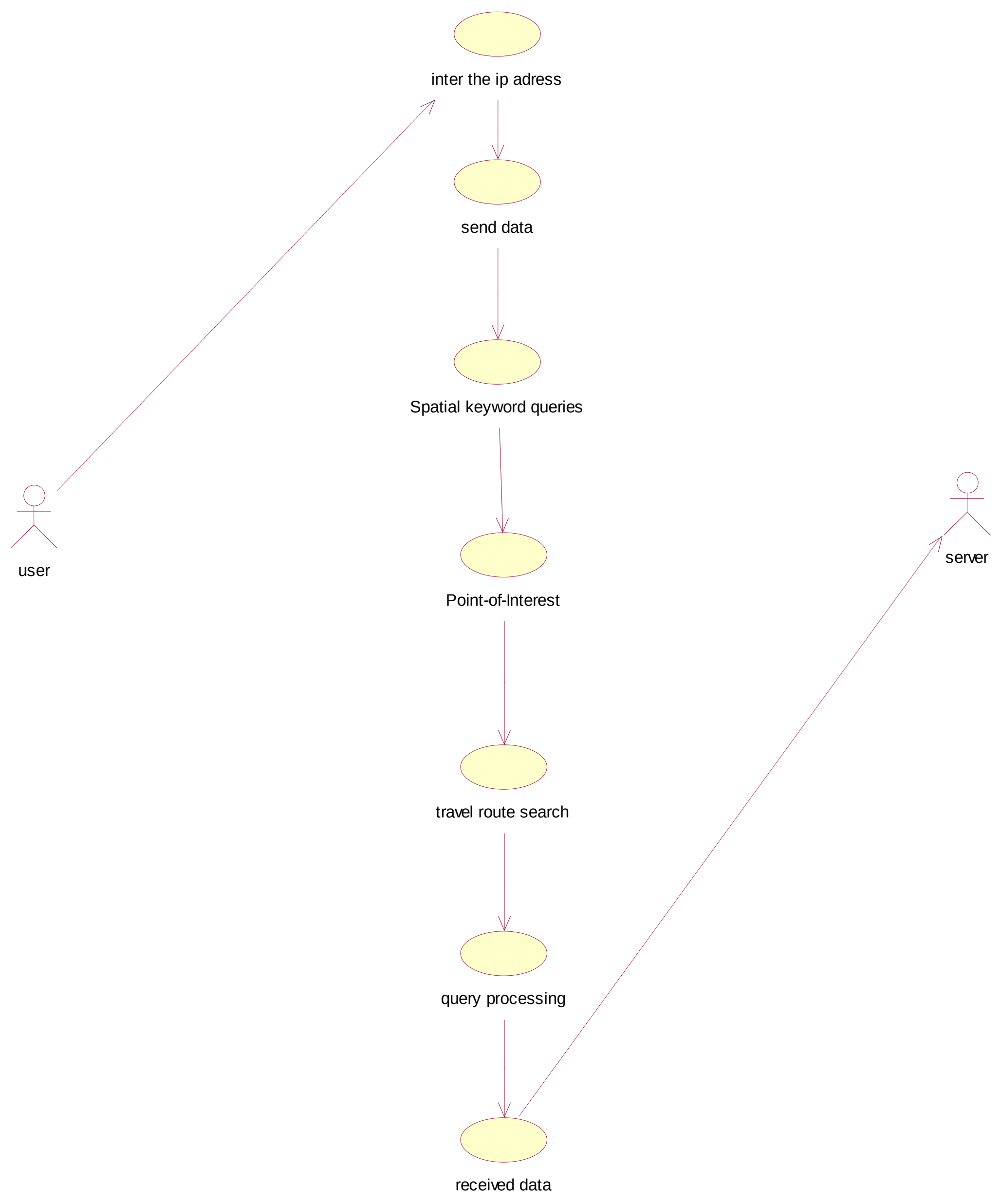


Fig 6.2.3 USE CASE diagram

6.2.4 Data Flow Diagram

A Data Flow diagram is a type of UML diagram that represents the structure of a system or software application by showing the classes, their at server, router, and the between them. client It is used to visualize the conceptual design of a system and to describe the objects and their interactions within the system.

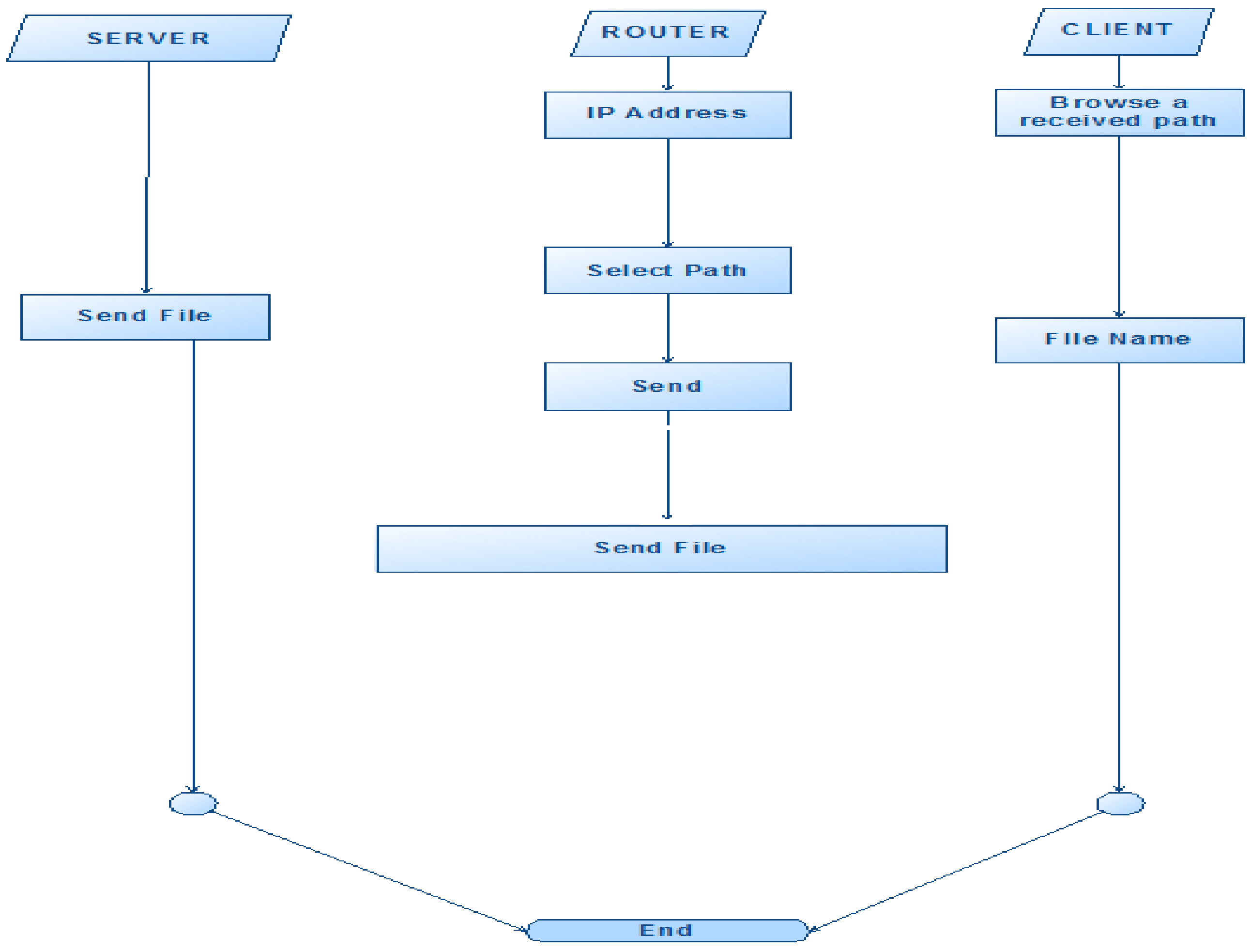


Fig 6.2.4 Data flow diagram

6.2.5 Collaboration diagram:

A collaboration diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. Itis a construct of a Message collaboration Chart. A collaboration diagram depicts the sequence of actions that occur in a system. The invocation of methods in each object, and the order in which the invocation occurs is captured in a collaboration diagram. This makes the Sequence diagrams very useful tool to easily represent the dynamic behavior of a system.

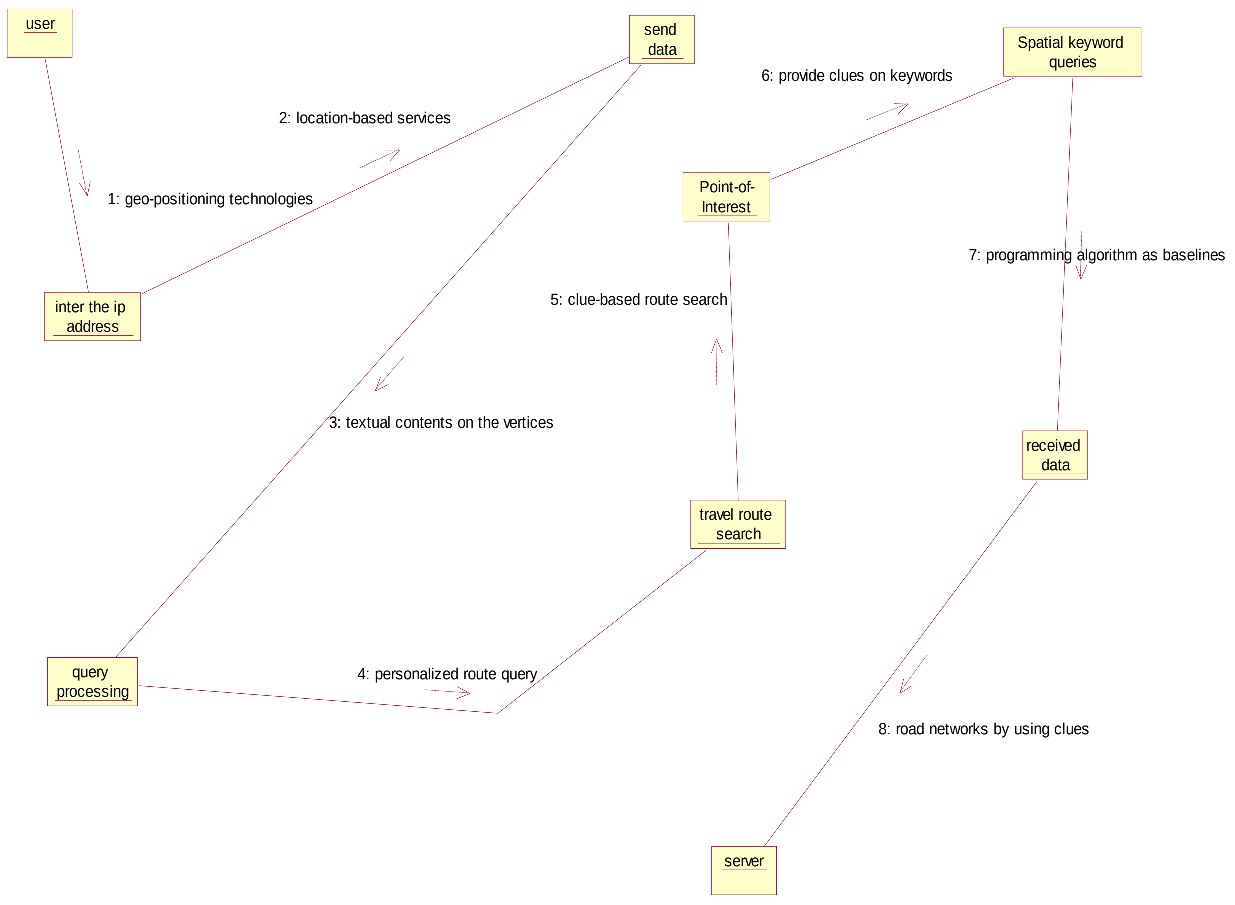

user
send data
Spatial keyword queries
6: provide clues on keywords
2: location-based services
1: geo-positioning technologies
Point-of-Interest
7: programming algorithm as baselines
inter the ip address
5: clue-based route search
3: textual contents on the vertices
received data
travel route search
query processing
4: personalized route query
8: road networks by using clues
server

Fig 6.2.5 collaboration diagram

6.2.6 Activity Diagram

Activity diagram modeling communication between client application and server software application by showing the classes, their at server, router, and the between them. client It is used to visualize the conceptual design of a system and to describe the objects and their interactions within the system.

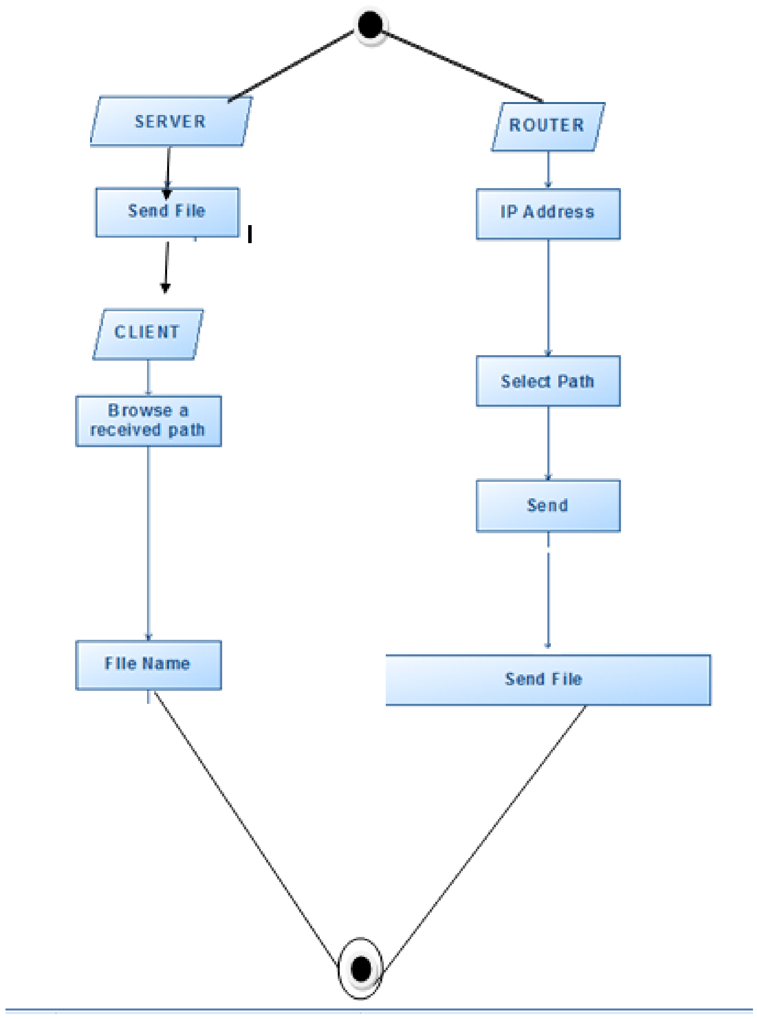


Fig 6.2.6 Activity Diagram

6.3 PROJECT REQUIREMENTS

6.3.1 Hardware Requirements:

• System	:	Pentium IV 2.4 GHz.
• Hard Disk	:	40 GB.
• Floppy Drive	:	1.44 Mb.
• Monitor	:	15 VGA Colour.
• Mouse	:	Logitech.
• RAM	:	256 Mb.

6.3.2 Software Requirements:

• Operating system	: -	Windows 7.
• Front End	: -	JAVA or JSP
• Database	: -	SQL SERVER 2008
• Tools	:-	Eclipse IDE

7. PROJECT IMPLEMENTATION

7.1 TECHNOLOGIES USED

Java Technology

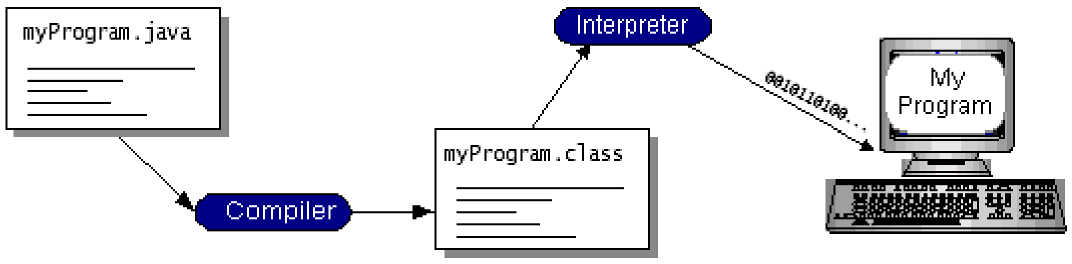
Java technology is both a programming language and a platform.

The Java Programming Language

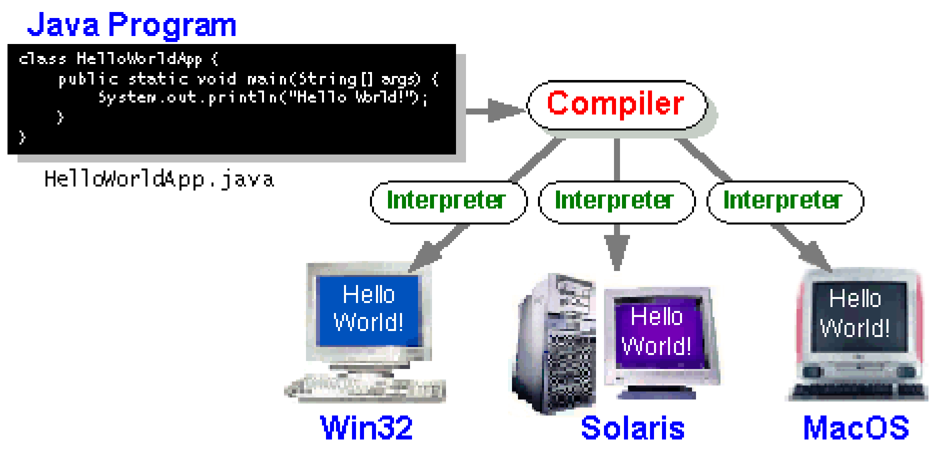
The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

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

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



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


```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

The Java Platform

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

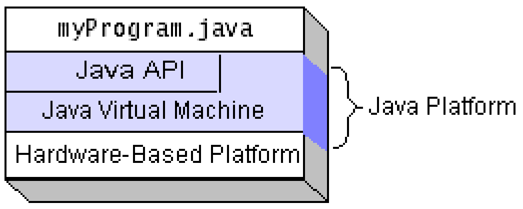
The Java platform has two components:

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

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

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

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



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

What Can Java Technology Do?

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

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

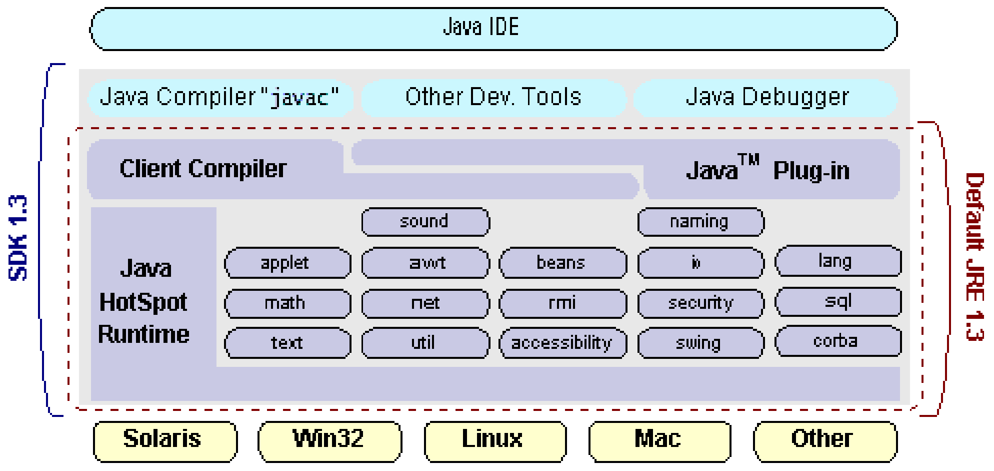
An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of

CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

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

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

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



How Will Java Technology Change My Life?

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

- **Get started quickly**: Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code**: Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
- **Write better code**: The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.
- **Develop programs more quickly**: Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.

- **Avoid platform dependencies with 100% Pure Java**: You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java^{TM} Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
- **Write once, run anywhere**: Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
- **Distribute software more easily**: You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded "on the fly," without recompiling the entire program.

ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a *de facto* standard for Windows programs to interface with database systems, programmers had to use proprietary languages

for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

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

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

ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources.

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

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

JDBC

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

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

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after. The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

JDBC Goals

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

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

1. ***SQL Level API***

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

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.

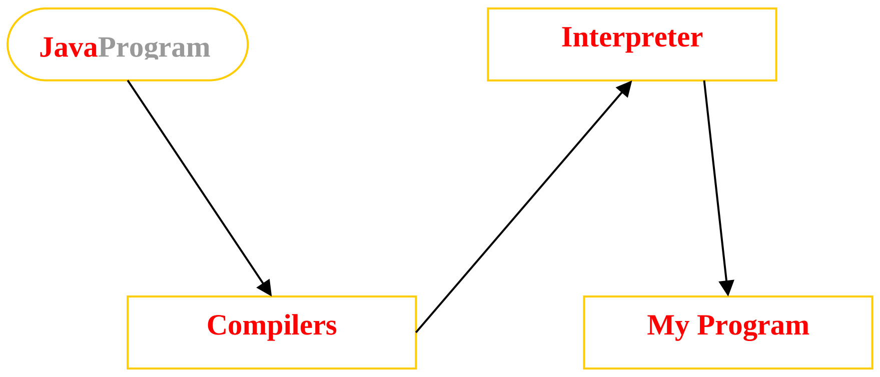
Java ha two things: a programming language and a platform.

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 Secure

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.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.



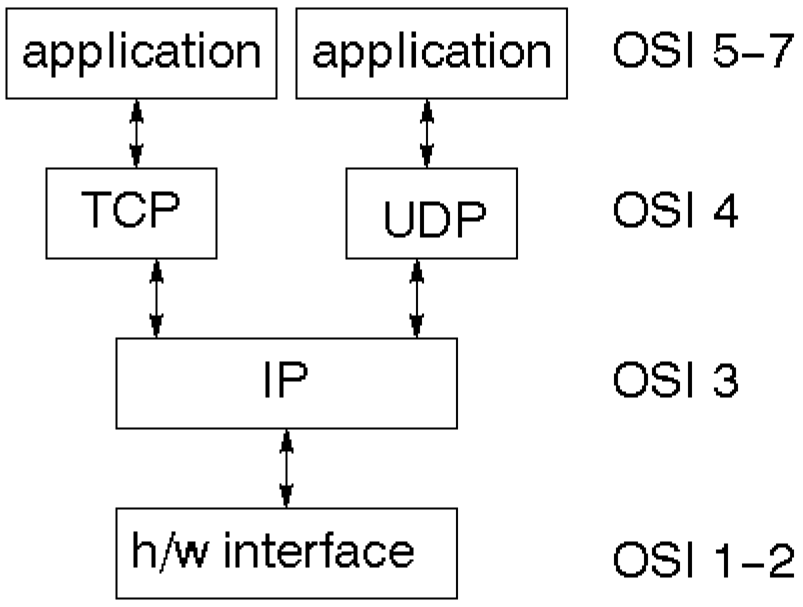
You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make "write once, run anywhere" possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

Networking

TCP/IP stack

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



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

IP datagram’s

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

UDP

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

TCP

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

Internet addresses

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

Network address

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

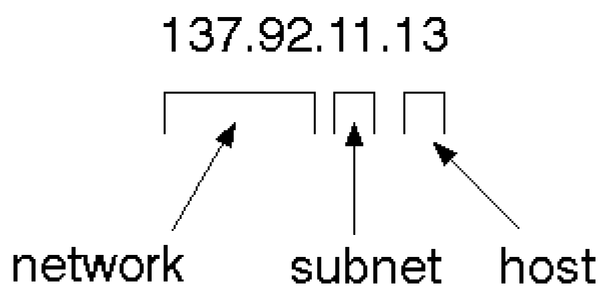
Subnet address

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

Host address

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

Total address



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

Port addresses

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

Sockets

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

```
#include <sys/types.h>
#include <sys/socket.h>
int socket(int family, int type, int protocol);
```

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

JFree Chart

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

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

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

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

JFreeChart is "open source" or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

1. Map Visualizations

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

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

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

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

2. Time Series Chart Interactivity

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

3. Dashboards

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

4. Property Editors

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

7.1.2 Algorithm:

We adopt the Adam optimizer in GraphDApp. Adam is a stepwise optimization algorithm based on a stochastic objective function of adaptive low-order moment estimation. It is an effective stochastic optimization method that requires only first-order gradients and little memory.

To get a better understanding of the representation capacity of TIG, we select 3 kinds of DApps (i.e., Aigang, Aragon and AaveProtocol) and visualize the corresponding TIGs constructed with Algorithm 1, as shown in Fig. 3. Since it is impossible to enumerate the TIGs for all the flows, we randomly select a flow for each DApp and use Matplotlib to draw the corresponding TIG.

It is clear to observe the differences in terms of graph structure: Aigang owns spindle shaped TIGs for its flows, Aragon has simple and compact TIGs, while AaveProtocol exhibits complex and fish-shaped TIGs.

Technique:

The first limitation is that GraphDApp needs a relatively long time to label an unknown flow. This can be solved by appropriately reducing the number of packets in TIGs to shorten feature extraction time, and employing less MLP layers and hidden units to accelerate the prediction speed.

The second limitation is that, as a fingerprinting solution, when the fingerprints of an application changes, the accuracy will decrease accordingly. To address this problem, we can regularly update the TIGs of the application and fine-tune the parameters in the classifier.

With TIG, we convert the encrypted flow classification problem into a graph classification problem. As deep learning has shown its superiority over traditional machine learning techniques, we design a GNN-based classifier using multi-layer perceptions. The classifier can automatically extract features from input TIGs and distinguish different graph structures by mapping them to different representations in the embedding space.

Methodology:

1) Web Application Classification Methods: The goals of this category mainly focus on website fingerprinting and webpage fingerprinting. Website fingerprinting aims to identify the traffic generated from visits of certain websites. In practice, homepages are usually used as representatives of the corresponding websites.

2) Mobile Application Classification Methods: The goals of this category mainly contain mobile

application classification and user action identification. Taylor et al. proposed Appscanner that combined statistical features of packet length with the random forest classifier to identify applications on smartphones. Several studies resorted to SSL/TLS flags in encrypted traffic and applied Markov Models to classify different smartphone applications.

3) DApp Fingerprinting Methods: DApps are emerging as a new service paradigm that relies on the underlying blockchain platform. In our previous study we obtained several key observations. First, statistical features of packet length are similar among DApps, e.g., the statistics of a DApp named Kitty are similar to those of Origin Protocol. Second, the adoption of the same blockchain platform makes the SSL/TLS message types of differ

7.2 SAMPLE CODE

Inder.html

```
<!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-equal="Content-Type" content="text/html; charset=utf-8" />
<title>Multi-Cloud Storage</title>
<meta name="keywords" content="secured theme, free template, template, red layout" />
<meta name="description" content="Secured Theme is provided by templatemo.com" />
<link href="templatemo_style.css" rel="stylesheet" type="text/css"
<link rel="stylesheet" type="text/css" href="style.css" />
<script type="text/javascript" src="js/jquery.min.js" ></script>
<script type="text/javascript" src="js/jquery-ui.min.js" ></script>
<script type="text/javascript">
        $(document).ready(function(){
                $("#featured > ul").tabs({fx:{opacity: "toggle"}}).tabs("rotate", 5000, true);
        });
<link rel="stylesheet" type="text/css" href="css/ddsmoothmenu.css" />
<link rel="stylesheet" href="css/slimbox2.css" type="text/css" media="screen" />

<script type="text/JavaScript" src="js/slimbox2.js"></script>
```

```
</head>
<body>

<div id="templatemo_wrapper">
        <div id="templatemo_header">
        <h2><font color="#FFFFFF" size="6" face="Times New Roman, Times, serif"
style="line-height: 35px;"></font></h2>

      <div id="templatemo_search">

      </div>
   </div> <!-- END of header -->

<br>
       <div id="templatemo_menu" class="ddsmoothmenu">
         <ul>

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

     <li><a href="login.jsp">Login</a> </li>

     <li><a href="reg.jsp">Registeration</a></li>

         </ul>
         <br style="clear: left" />
      </div> <!-- end of templatemo_menu -->
  <div id="featured">
   <!-- First Content -->
   <img src="images/header.jpg" width="940" height="220">
   <!-- Second Content -->
   <!-- Third Content -->
   <!-- Fourth Content -->
   <div id="fragment-4" class="ui-tabs-panel ui-tabs-hide" style=""> <img
src="images/featured_list/image4.jpg" alt="" />
     <div class="info" > </div>
   </div>
  </div>
   <div id="templatemo_main">
        <div class="content_wrapper content_mb_30">
        <div class="col_3 service_home">

         </div>
```

```
           <div class="col_3 service_home">

           </div>
           <div class="col_3 no_margin_right service_home">

           </div>
        </div>
          <div class="content_wrapper">
           <div class="col_2">
                      <fieldset style="border: 2px solid #F07305 ;
border-radius: 10px ;
    </div> <!-- END of templatemo_footer -->
</div> <!-- END of templatemo_wrapper -->

</body>
</html
```

Login.jsp

```
<!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>Multi-Cloud Storage</title>
<meta name="keywords" content="secured theme, free template, templatemo, red layout"
/>
<meta name="description" content="Secured Theme is provided by templatemo.com" />
<link href="templatemo_style.css" rel="stylesheet" type="text/css" />

<link rel="stylesheet" type="text/css" href="style.css" />
<script type="text/javascript" src="js/jquery.min.js" ></script>
<script type="text/javascript" src="js/jquery-ui.min.js" ></script>
<script type="text/javascript">
        $(document).ready(function()
$("#featured > ul").tabs({fx:{opacity: "toggle"}}).tabs("rotate", 5000,true);
        });
</script>
<link rel="stylesheet" type="text/css" href="css/ddsmoothmenu.css" />
<link rel="stylesheet" href="css/slimbox2.css" type="text/css" media="screen" />
<script type="text/JavaScript" src="js/slimbox2.js"></script>
</head>
<body>
<div id="templatemo_wrapper">
        <div id="templatemo_header">
        <h2><font color="#FFFFFF" size="6" face="Times New Roman, Times, serif"
```

```
style="line-height: 35px;"></font></h2>
        <div id="templatemo_search"
        </div>
    </div> <!-- END of header -->

<br>
        <div id="templatemo_menu" class="ddsmoothmenu">

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

      <li><a href="login.jsp">Login</a> </li>

      <li><a href="reg.jsp">Registeration</a></li>


            </ul>
            <br style="clear: left" />
        </div> <!-- end of templatemo_menu
  <div id="featured">
    <!-- First Content -->
    <img src="images/header.jpg" width="940" height="220">
    <!-- Second Content -->
    <!-- Third Content -->
 <!-- Fourth Content -->
    <div id="fragment-4" class="ui-tabs-panel ui-tabs-hide" style=""> <img
src="images/featured_list/image4.jpg" alt="" />
      <div class="info" > </div>
    </div>
  </div>
    <div id="templatemo_main">
          <div class="content_wrapper content_mb_30">
          <div class="col_3 service_home">

            </div>
            <div class="col_3 service_home">

            </div>
            <div class="col_3 no_margin_right service_home">
   </div>
        </div>
          <div class="content_wrapper">
            <div class="col_2">
                <h2>Login</h2>
                         <fieldset style="border: 2px solid #F07305 ;
```

```
border-radius: 10px ;
-moz-border-radius: 10px ;
-webkit-border-radius: 10px ; width:400px;  margin: 0px
     <table width="398" height="160" align="center">
        <form action="userlogin.jsp" method="post" name="form" onsubmit="return 
validation();">
          <tr>
            <td width="148" height="37"><font 
size="2"><strong>UserName</strong></font></td>
            <td width="238"><input type="text" name="name" id="s" size="25" 
style="border: 2px solid #F07305" /></td>
          </tr>
          <tr>
            <td height="35"><font size="2"><strong>Password</strong></font></td>
            <td><input type="password" name="password" id="s" size="25" style="border: 
2px solid #F07305"/></td>
          </tr>           <tr>
            <td height="35"><font size="2"><strong>UserType</strong></font></td>
            <td><select name="occupation" style="border: 2px solid #F07305">

<option value="">----------Select----------        <option 
value="Publisher">Publisher</option>
         <option value="Subscriber">Subscriber</option

         </select></td>
          </tr>
          <tr>
            <td height="39"></td>
      <td><input type="submit" name="submit" class="button2" value="Login" 
style="border: 2px solid #F07305">
       </div>
       <div class="clear"></div>
   </div> <!-- END of templatemo_footer -->
</div> <!-- END of templatemo_wrapper --
</body>
</htMI
```

Reg.jsp

```
<%@ page import="java.sql.*"  import="databaseconnection.*" errorPage="" %>
<%
                String myname=(String)session.getAttribute("myname");
                String myemail=(String)session.getAttribute("myemail");
                String myoccupation=(String)session.getAttribute("myoccupation");
                System.out.println(myoccupation);
```

```
%>
<!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>Multi-Cloud Storage</title>
<meta name="keywords" content="secured theme, free template, templatemo, red layout"
/>
<meta name="description" content="Secured Theme is provided by templatemo.com" />
<link href="templatemo_style.css" rel="stylesheet" type="text/css" />

<link rel="stylesheet" type="text/css" href="style.css" />
<script type="text/javascript" src="js/jquery.min.js" ></script>
<script type="text/javascript" src="js/jquery-ui.min.js" ></script>
<script type="text/javascript">
        $(document).ready(function(){
                $("#featured > ul").tabs({fx:{opacity: "toggle"}}).tabs("rotate", 5000, true);
        });
</script>

<link rel="stylesheet" type="text/css" href="css/ddsmoothmenu.css" />
<link rel="stylesheet" href="css/slimbox2.css" type="text/css" media="screen" />

<script type="text/JavaScript" src="js/slimbox2.js">
</head>
<body>
<div id="templatemo_wrapper">
        <div id="templatemo_header">
        <h2><font color="#FFFFFF" size="6" face="Times New Roman, Times, serif"
style="line-height: 35px;"></font></h2>
      <div id="templatemo_sea
      </div>
   </div> <!-- END of header -->
       <div id="templatemo_menu" class="ddsmoothmenu">
         <ul
     <li><a href="index.html" class="selected">Home</a></li
     <li><a href="index.html">Login</a> </li
     <li><a href="reg.jsp">Registeration</a></li>
     <li><a href="share_message.jsp">Share Message</a></li>
<li><a href="graph.jsp">View Graph</a></li>
     <li><a href="index.html">LogOut</a></li>
         </ul>
         <br style="clear: left" />
      </div> <!-- end of templatemo_menu -->
     <div id="featured">
   <!-- First Content -->
   <img src="images/header.jpg" width="940" height="220">
   <!-- Second Content -->
   <!-- Third Content -->
   <!-- Fourth Content -->
   <div id="fragment-4" class="ui-tabs-panel ui-tabs-hide" style=""> <img
src="images/featured_list/image4.jpg" alt="" />
     <div class="info" > </div>
   </div>
 </div>
   <div id="templatemo_main">
        <h2>WelcomePublisher:<%=myname%>!</h2>
        <div class="content_wrapper">
         <div class="col_2">
      <h5 align="justify"><font size="3">Publish Secret Message Such As About
        news distribution, stock exchange, environmental monitoring, traffic
        control, and public sensing</font
 <fieldset style="border: 2px solid #F07305 ;
border-radius: 10px ;
-moz-border-radius: 10px ;
-webkit-border-radius: 10px ; width:400px;  margin: 0px  50px;">

     <table width="398" height="160" align="center">
        <form action="insert_message.jsp" method="post" name="form"
enctype="multipart/form-data" onsubmit="return validation();">
         <tr>
          <td width="148" height="37"><font size="2"><strong>Share To
</strong></font></td>
             <td width="238"><select name="tname" style="border: 2px solid #F07305">
             <option value="" >Select Name</option>

        st4=con4.createStatement()
                String sql4="select * from profile where name!='"+myname+"'";

        while(rs5.next()){%>
             <option
value="<%=rs5.getString("occupation")%>"><%=rs5.getString("occupation")%></optio
n>                    <%}
        catch (Exception eq5)
        out.println(eq5.getMessage());
        }
                            %>
          </select>
             </td>
          </tr>
             <tr>
              <td height="35"><font size="2"><strong>Event
TitleName</strong></font></td>
              <td><input type="text" name="event" id="s" size="25" style="border: 2px solid
#F07305"/></td>
          </tr>            <tr>
              <td height="35"><font size="2"><strong>Headlines</strong></font></td>
              <td><input type="text" name="head" id="s" size="25" style="border: 2px solid
#F07305"/></td>
          </tr>
                          <tr>
              <td height="35"><font size="2"><strong>RelatedTopic</strong></font></td>
              <td><input type="text" name="top" id="s" size="25" style="border: 2px solid
#F07305"/></td>
          </tr>
                           <tr>
              <td height="35"><font
size="2"><strong>MessageContent</strong></font></td>
              <td><input type="text" name="message" id="s" size="25" style="border: 2px
solid #F07305"/></td>
          </tr>
             <tr>
              <td height="35"><font size="2"><strong>Image</strong></font></td>
              <td><input  name="image" type="file"  id="s" size="25" style="border: 2px
solid #F07305"/></td>
          </tr>
          <tr>

   <td height="39"></td>
              <td><input type="submit" name="submit" class="button2" value="Share"
style="border: 2px solid #F07305">
               <input type="reset" name="reset" class="button2" value="Clear"
style="border: 2px solid #F07305">
              </td>
          </tr>
        </form>
      </table>
     </fieldset>
          </div>
     <div class="news_list">
<br><br><br><br><br><br><br>
```

```
                &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<img src="images/share copy.PNG">
                  <div class="clear"></div>
                </div>
                <div class="clear"></div>

              </div>
            </div>
            <div class="clear"></div>
          </div
              <div id="templatemo_footer">
              <div class="col_4">
            </div>
            <div class="col_4">
           </div>
            <div class="col_4"
            </div>
            <div class="col_4 no_margin_right"
            </div>
            <div class="clear"></div>
          </div> <!-- END of templatemo_footer -->
</div> <!-- END of templatemo_wrapper -->
</body>
</htmi>
```

Contact.html

```
<%@ page import="java.sql.*"  import="databaseconnection.*" errorPage="" %>
<%
          String myname=(String)session.getAttribute("myname");
          String myemail=(String)session.getAttribute("myemail");
          String myoccupation=(String)session.getAttribute("myoccupation");
          System.out.println(myoccupation);
%>
<!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>Multi-Cloud Storage</title>
<meta name="keywords" content="secured theme, free template, templatemo, red layout"
/>
<meta name="description" content="Secured Theme is provided by templatemo.com" />
<link href="templatemo_style.css" rel="stylesheet" type="text/css" />

<link rel="stylesheet" type="text/css" href="style.css" />
<script type="text/javascript" src="js/jquery.min.js" ></script>
```

```
<script type="text/javascript" src="js/jquery-ui.min.js" ></script>
<script type="text/javascript">
        $(document).ready(function(){
                $("#featured > ul").tabs({fx:{opacity: "toggle"}}).tabs("rotate", 5000, true);
        });
</script>

<link rel="stylesheet" type="text/css" href="css/ddsmoothmenu.css" />
<link rel="stylesheet" href="css/slimbox2.css" type="text/css" media="screen" />

<script type="text/JavaScript" src="js/slimbox2.js"></script>
</head>
<body>
<div id="templatemo_wrapper">
        <div id="templatemo_header">
        <h2><font color="#FFFFFF" size="6" face="Times New Roman, Times, serif"
style="line-height: 35px;"></font></h2>
      <div id="templatemo_search"
      </div>
   </div> <!-- END of header -->
       <div id="templatemo_menu" class="ddsmoothmenu">
         <ul>

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

     <li><a href="index.html">Login</a> </li>

     <li><a href="reg.jsp">Registeration</a></li>

     <li><a href="share_message.jsp">Share Message</a></li>
                <li><a href="graph.jsp">View Graph</a></li>
     <li><a href="index.html">LogOut</a></li>


         </ul>
   <br style="clear: left" />
      </div> <!-- end of templatemo_menu -->
  <div id="featured">
   <!-- First Content -->
   <img src="images/header.jpg" width="940" height="220">
   <!-- Second Content -->
   <!-- Third Content -->
   <!-- Fourth Content -->
   <div id="fragment-4" class="ui-tabs-panel ui-tabs-hide" style=""> <img
src="images/featured_list/image4.jpg" alt="" />
```

```
      <div class="info" > </div>
     </div>
   </div>
     <div id="templatemo_main">
           <h2>WelcomePublisher:<%=myname%>!</h2>
           <div class="content_wrapper">
            <div class="col_2">
               {
           out.println(eq5.getMessage());

            %>
            </select>
             </td>
            </tr>
            <tr>
     <td height="35"><font size="2"><strong>Event TitleName</strong></font></td>
              <td><input type="text" name="event" id="s" size="25" style="border: 2px solid
#F07305"/></td>
        <tr>
              <td height="35"><font size="2"><strong>Headlines</strong></font></td>
              <td><input type="text" name="head" id="s" size="25" style="border: 2px solid
#F07305"/></td>
            </tr>
                              <tr>
              <td height="35"><font size="2"><strong>RelatedTopic</strong></font></td>
              <td><input type="text" name="top" id="s" size="25" style="border: 2px solid
#F07305"/></td>
            </tr>
                               <tr>
              <td height="35"><font
size="2"><strong>MessageContent</strong></font></td>
              <td><input type="text" name="message" id="s" size="25" style="border: 2px
solid #F07305"/></td>
            </tr>
             <tr>
              <td height="35"><font size="2"><strong>Image</strong></font></td>
              <td><input  name="image" type="file"  id="s" size="25" style="border: 2px
solid #F07305"/></td>
            </tr>
            <tr>
              <td height="39"></td>
              <td><input type="submit" name="submit" class="button2" value="Share"
style="border: 2px solid #F07305">
               <input type="reset" name="reset" class="button2" value="Clear"
```

```
style="border: 2px solid #F07305">
              </td>
            </tr>
          </form>
        </table>
                </div>
                 <div class="news_list">
 <br><br><br><br><br><br><br>
                 &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<img src="images/share copy.PNG">
                    <div class="clear"></div>
                </div>
                <div class="clear"></div>

            </div>
        </div>
        <div class="clear"></div>
    </div>
<div id="templatemo_footer">
          <div class="col_4"
        </div>
        <div class="col_4">
        </div>
        <div class="col_4">
        </div>
        <div class="col_4 no_margin_right"
        </div>
        <div class="clear"></div
  </div> <!-- END of templatemo_footer -->
</div> <!-- END of templatemo_wrapper -->
</body>
</HTML>
```

Search.jsp

```
<%@ page import="java.sql.*"%>
<%@ page import="databaseconnection.*"%>
<%
String myname=(String)session.getAttribute("myname");
%>
<!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>Multi-Cloud Storage</title>
<meta name="keywords" content="secured theme, free template, templatemo, red layout"
/>
<meta name="description" content="Secured Theme is provided by templatemo.com" />
<link href="templatemo_style.css" rel="stylesheet" type="text/css" />

<link rel="stylesheet" type="text/css" href="style.css" />
<script type="text/javascript" src="js/jquery.min.js" ></script>
<script type="text/javascript" src="js/jquery-ui.min.js" ></script>
<script type="text/javascript">
        $(document).ready(function(){
                $("#featured > ul").tabs({fx:{opacity: "toggle"}}).tabs("rotate", 5000, true);
        });
</script>

<link rel="stylesheet" type="text/css" href="css/ddsmoothmenu.css" />
<link rel="stylesheet" href="css/slimbox2.css" type="text/css" media="screen" />

<script type="text/JavaScript" src="js/slimbox2.js"></script>


</head>
<body>

<div id="templatemo_wrapper">
        <div id="templatemo_header">
        <h2><font color="#FFFFFF" size="6" face="Times New Roman, Times, serif"
style="line-height: 35px;"></font></h2>
      <div id="templatemo_search">


      </div>
   </div> <!-- END of header -->


       <div id="templatemo_menu" class="ddsmoothmenu">
         <ul>

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

     <li><a href="index.html">Login</a> </li>

     <li><a href="reg.jsp">Registeration</a></li>
```

```
    <li><a href="select_share.jsp">View</a></li>

    <li><a href="index.html">LogOut</a></li>


       </ul>
       <br style="clear: left" />
     </div> <!-- end of templatemo_menu -->


  <div id="featured">
   <!-- First Content -->
   <img src="images/header.jpg" width="940" height="220">
   <!-- Second Content -->
   <!-- Third Content -->
   <!-- Fourth Content -->
   <div id="fragment-4" class="ui-tabs-panel ui-tabs-hide" style=""> <img
src="images/featured_list/image4.jpg" alt="" />
    <div class="info" > </div>
   </div>
  </div>
   <div id="templatemo_main">
      <div class="content_wrapper content_mb_30">
      <div class="col_3 service_home">

       </div>
       <div class="col_3 service_home">
            <h2>WelcomeSubscriber:<%=myname%>!</h2>
       </div>
       <div class="col_3 no_margin_right service_home">

       </div>
     </div>
      <div class="content_wrapper">
       <div class="col_2">

         <h2>Search Here</h2>


                <fieldset style="border: 2px solid #F07305 ;
border-radius: 10px ;
-moz-border-radius: 10px ;
-webkit-border-radius: 10px ; width:400px;  margin: 0px  50px;">
  <br><br>
```

```
          <table width="496" height="160" align="center">

                   <form method="post" name="form" action="search_share.jsp" 
onsubmit="return validation();">
           <tr>
             <td width="201" height="37"><font size="2"><strong>Search By Publisher 
Name</strong></font></td>
             <td width="283"><input type="text" name="sname" id="s" size="45" 
style="border: 2px solid #F07305" /></td>
           </tr>

           <tr>
             <td height="39"></td>
             <td><input type="submit" name="submit" class="button2" value="Submit" 
style="border: 2px solid #F07305">
               <input type="reset" name="reset" class="button2" value="Clear" 
style="border: 2px solid #F07305">
             </td>
           </tr>
          </form>
        </table>
      </fieldse>
               </div>
      <div class="news_list">
       <div class="clear"></div>
               </div>
               <div class="clear"></div>

           </div>
        </div>
        <div class="clear"></div>
    <div id="templatemo_footer">
          <div class="col_4">
       </div>
        <div class="col_4">
       </div>
  <div class="col_4">
      </div>
     <div class="col_4 no_margin_right">
        </div>
        <div class="clear"></div>
    </div> <!-- END of templatemo_footer -->
</div> <!-- END of templatemo_wrapper -->
```

```
</body>
</html>
```

Userlogin.jsp

```
<%@ page import="java.sql.*"%>
<%@ page import="databaseconnection.*"%>
<%@page
import="com.oreilly.servlet.*,java.sql.*,java.lang.*,databaseconnection.*,java.text.Simpl
eDateFormat,java.util.*,java.io.*,javax.servlet.*, javax.servlet.http.*"
errorPage="Error.jsp"%
<%
ServletContext context=getServletContext();
String name1=context.getInitParameter("name1");
String pass1=context.getInitParameter("pass1");
%>
<html>
<head>
<title>Secure</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"
</head>
<body>
Statement st = null;
ResultSet rs = null;
String name = request.getParameter("name");
String password = request.getParameter("password");
String occupation = request.getParameter("occupation")
        Connection con=databasecon.getconnection();
        st = con.createStatement();
        String qry ="select * from profile where  name='"+name+"' AND
password='"+password+"' AND occupation='"+occupation+"'";
        rs = st.executeQuery(qry);
        if(rs.next())
        {
           session.setAttribute("myid",Integer.toString(rs.getInt("id")));
                session.setAttribute("myname",rs.getString("name"));
                System.out.println("myname");
                session.setAttribute("myemail",rs.getString("email"));
                session.setAttribute("mygender",rs.getString("gender"));
                session.setAttribute("myoccupation",rs.getString("occupation"));
session.setAttribute("myp",rs.getString("prime"));
                session.setAttribute("myg",rs.getString("generator"));
                session.setAttribute("mys",rs.getString("privatekey"));
        if(occupation.equals("Publisher"))
  {
```

```
  <jsp:forward page="share_message.jsp"/>

  <%
  }
  if(occupation.equals("Subscriber"))
  {
  %
  <jsp:forward page="search.jsp"/>
  <%
  }
  }
        else{
                System.out.println("---------------------");
                System.out.println("Successfully login by "+rs.getString("name"));
                System.out.println("---------------------");

        }
        con.close();
        st.close();
}
catch(Exception ex){
        out.println(ex);
}
%>
</body>
</html>
```

7.3 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.

TYPES OF TESTS

7.4.1 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.

7.4.2 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 satisfaction, 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.

7.4.3 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.

7.5 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.

7.5.1 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.

7.5.2 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.

7.6 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

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.

7.7 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results:

All the test cases mentioned above passed successfully. No defects encountered.

8.SCREEN SHOTS

Screen Layout to Home page :

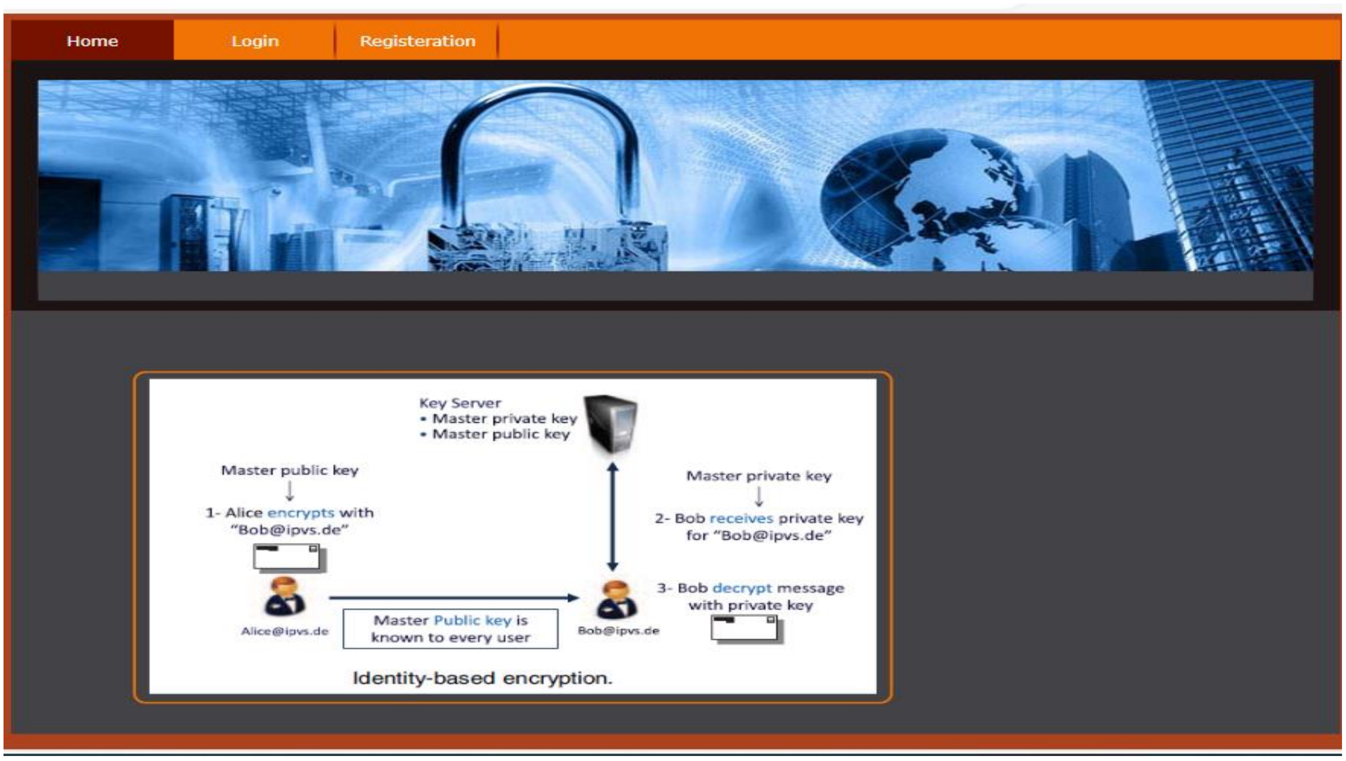

Home
Login
Registeration
Key Server
• Master private key
• Master public key
Master public key
1- Alice encrypts with "Bob@ipvs.de"
Master private key
2- Bob receives private key for "Bob@ipvs.de"
3- Bob decrypt message with private key
Alice@ipvs.de
Master Public key is known to every user
Bob@ipvs.de
Identity-based encryption.

Fig :HOME PAGE

Screen Layout to Login page :

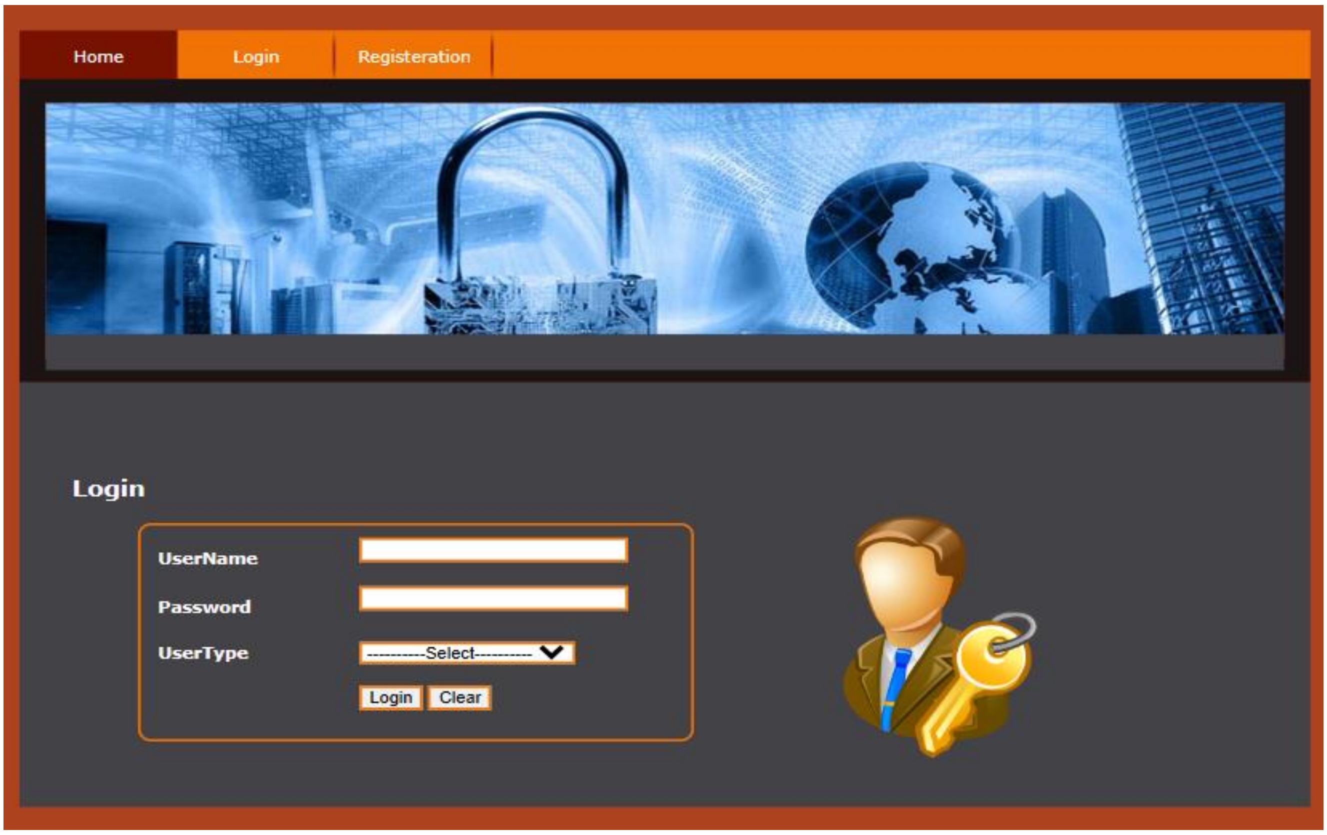
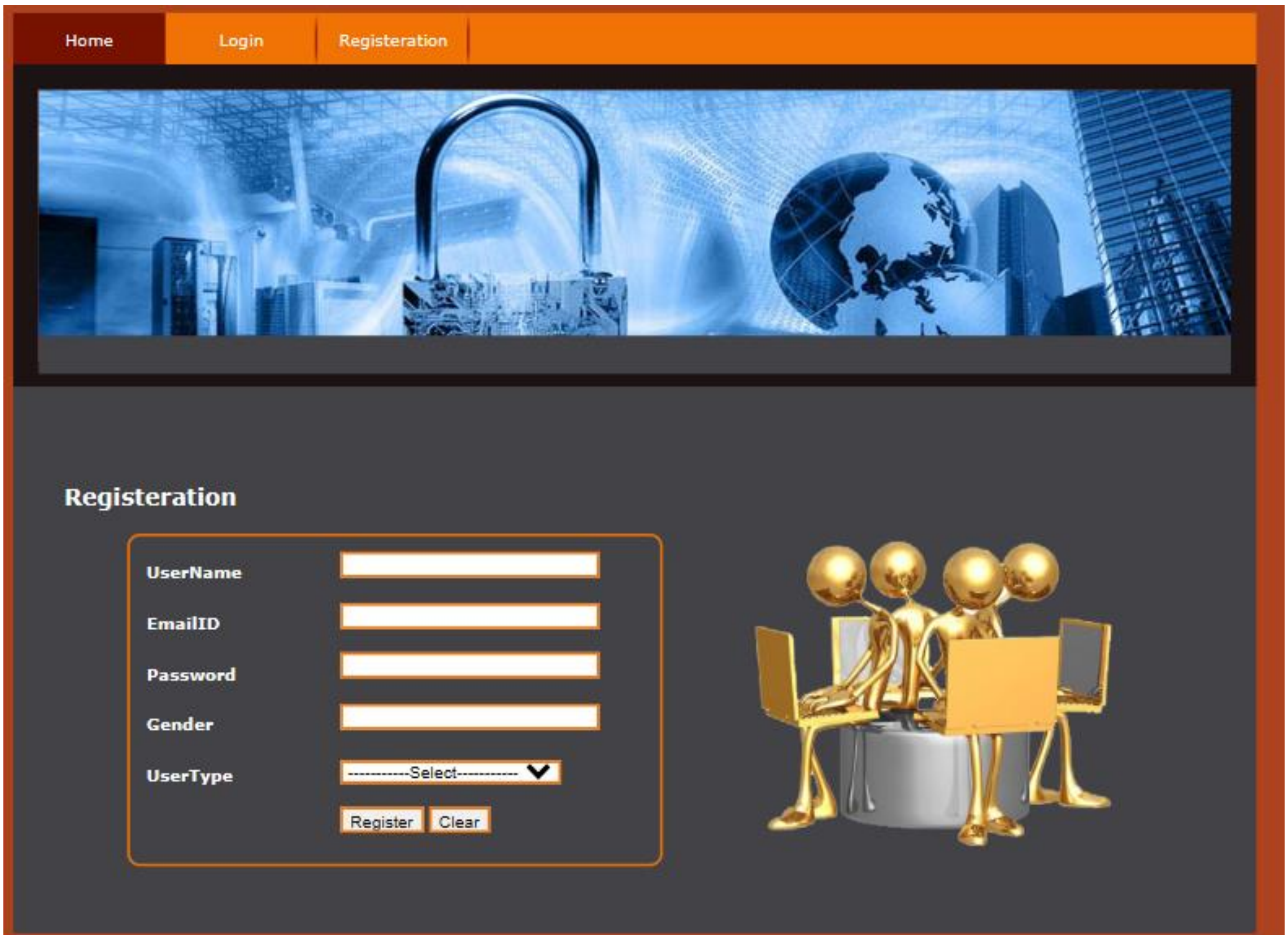


Fig: Login page

Screen Layout to Registration page :

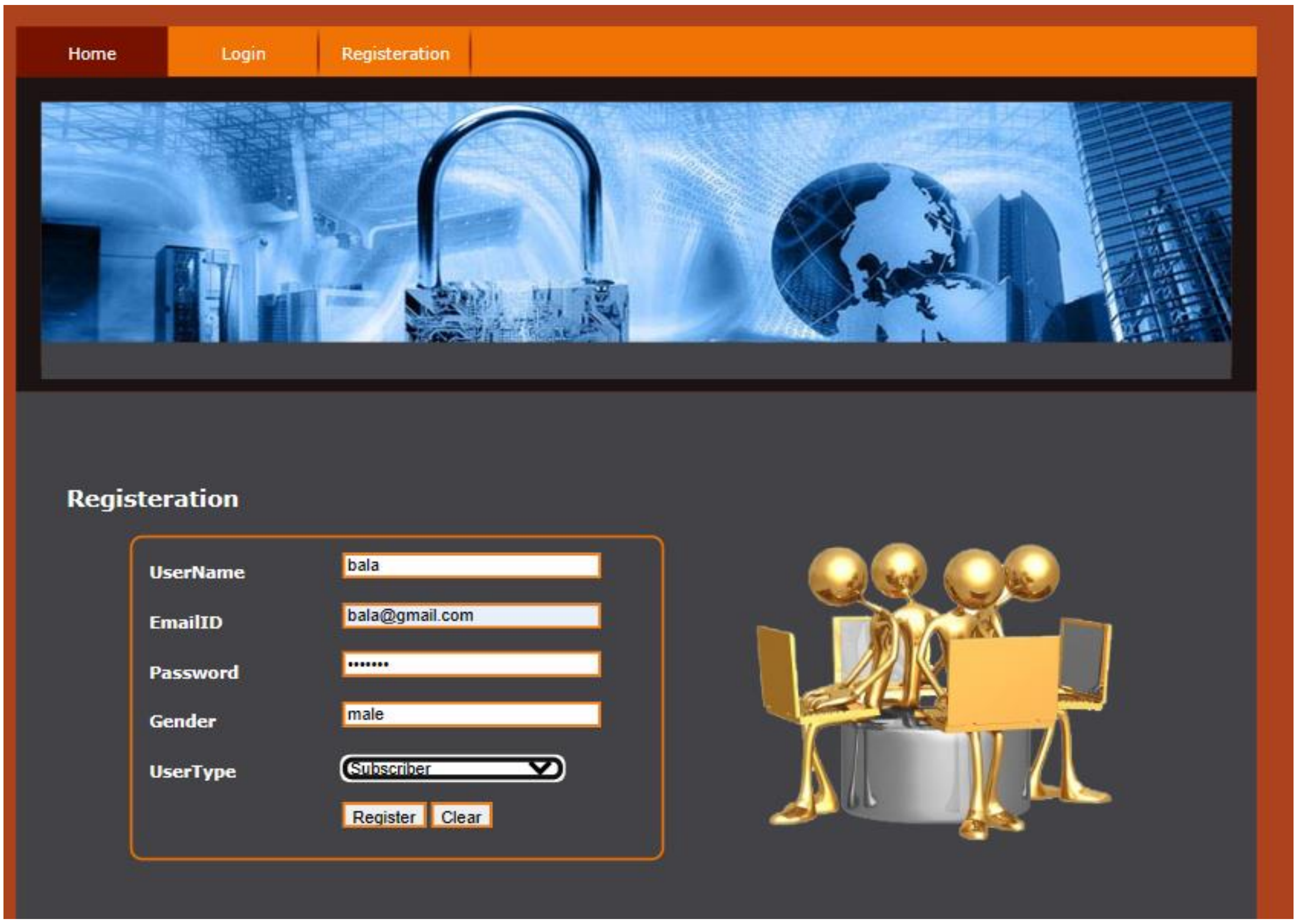


Registeration

UserName	
EmailID	
Password	
Gender	
UserType	-----------Select-----------

Fig: Registeration page

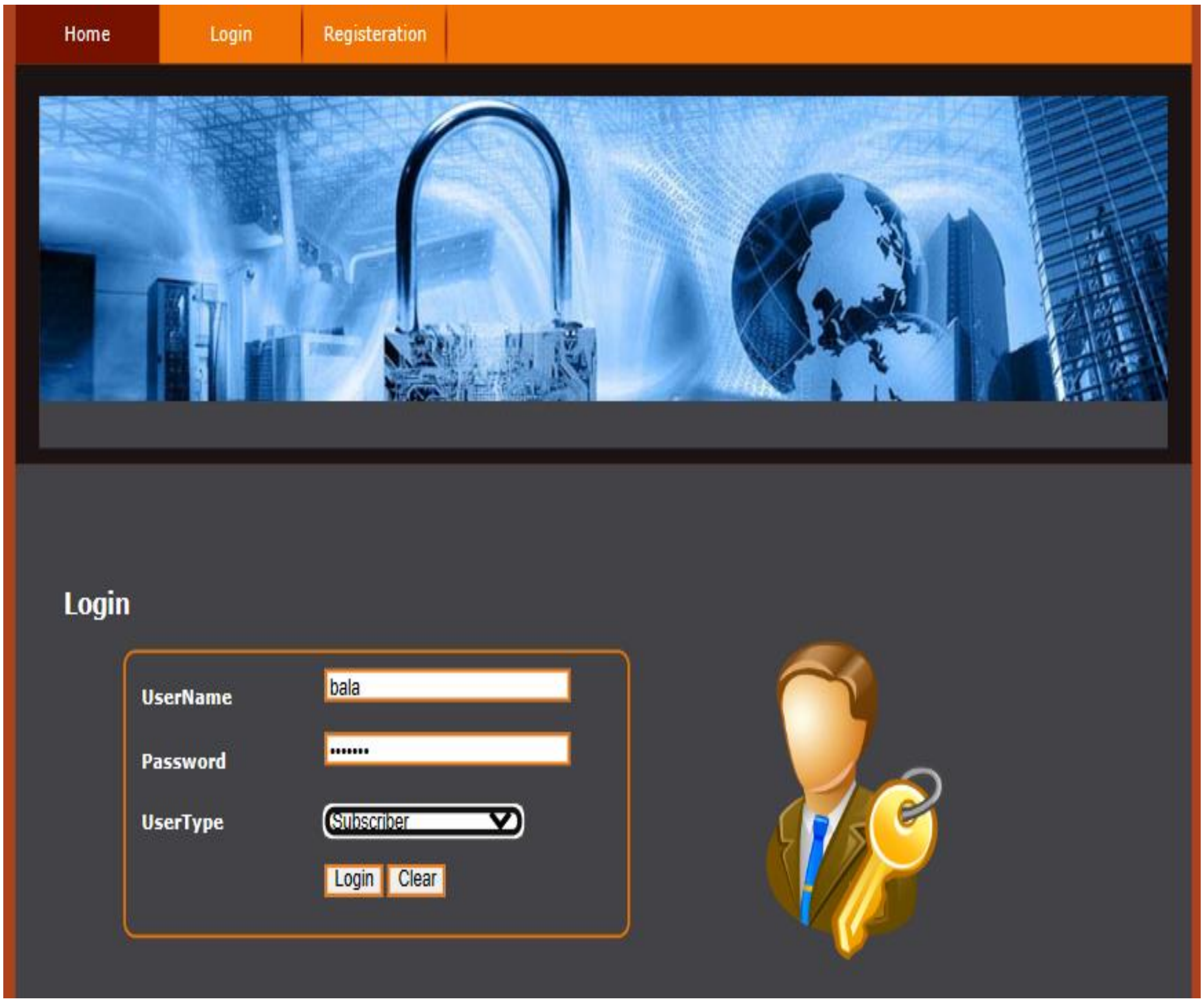
Screen Layout to First of all Registeration :



UserName	bala
EmailID	bala@gmail.com
Password	•••••••
Gender	male
UserType	Subscriber

Fig : registration

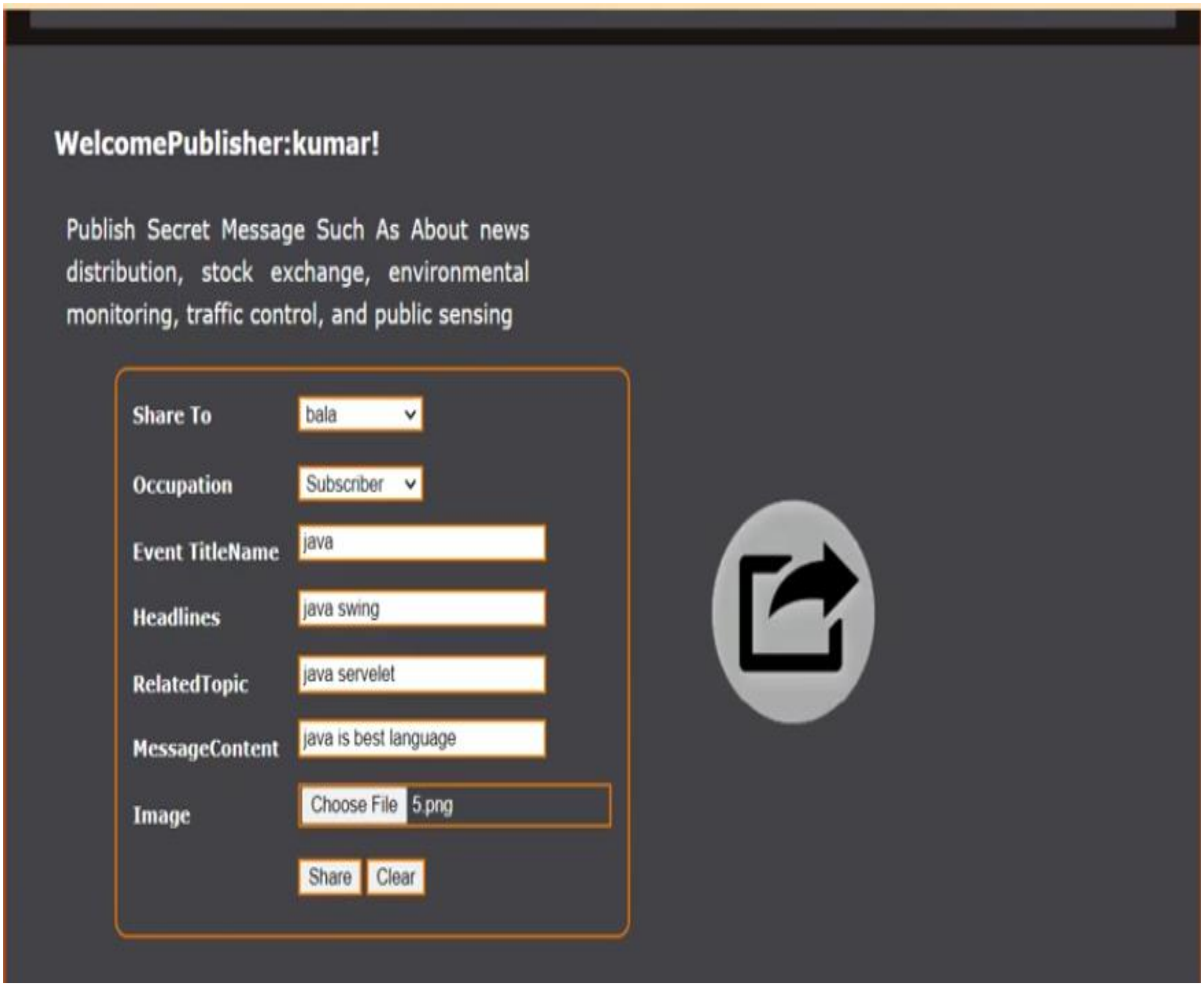
Screen Layout to RELOGIN :



UserName: bala
Password: •••••••
UserType: Subscriber

Fig : Relogin

Screen Layout to Share information :



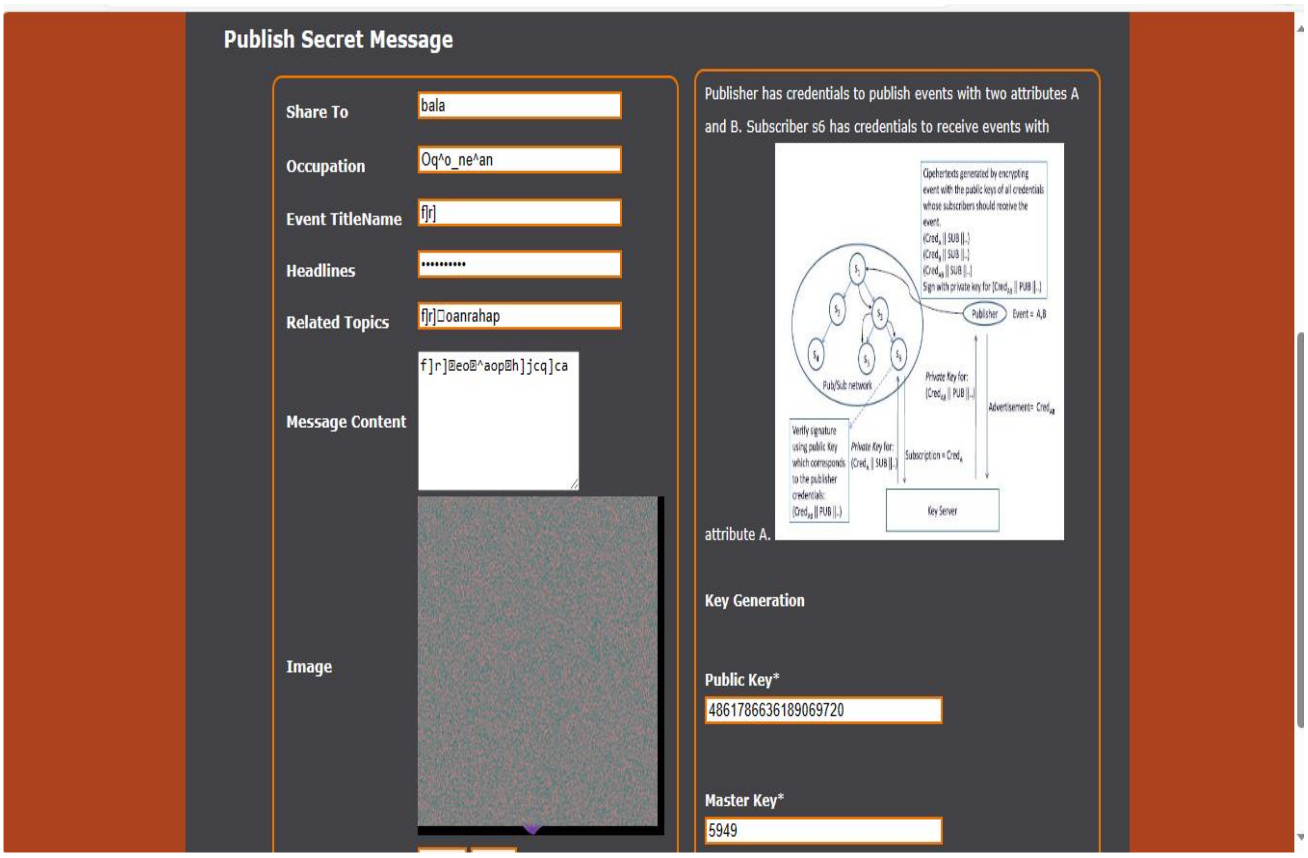
WelcomePublisher:kumar!

Publish Secret Message Such As About news distribution, stock exchange, environmental monitoring, traffic control, and public sensing

Share To	bala
Occupation	Subscriber
Event TitleName	java
Headlines	java swing
RelatedTopic	java servelet
MessageContent	java is best language
Image	Choose File 5.png

Fig : share information

Screen Layout to publish secret Message:



Publish Secret Message

Share To	bala
Occupation	Oq^o_ne^an
Event TitleName	f]r]
Headlines	••••••••••
Related Topics	f]r]oanrahap
Message Content	f]r]eo^aoph]jcq]ca
Image	

Publisher has credentials to publish events with two attributes A and B. Subscriber s6 has credentials to receive events with attribute A.

Key Generation

Public Key*

4861786636189069720

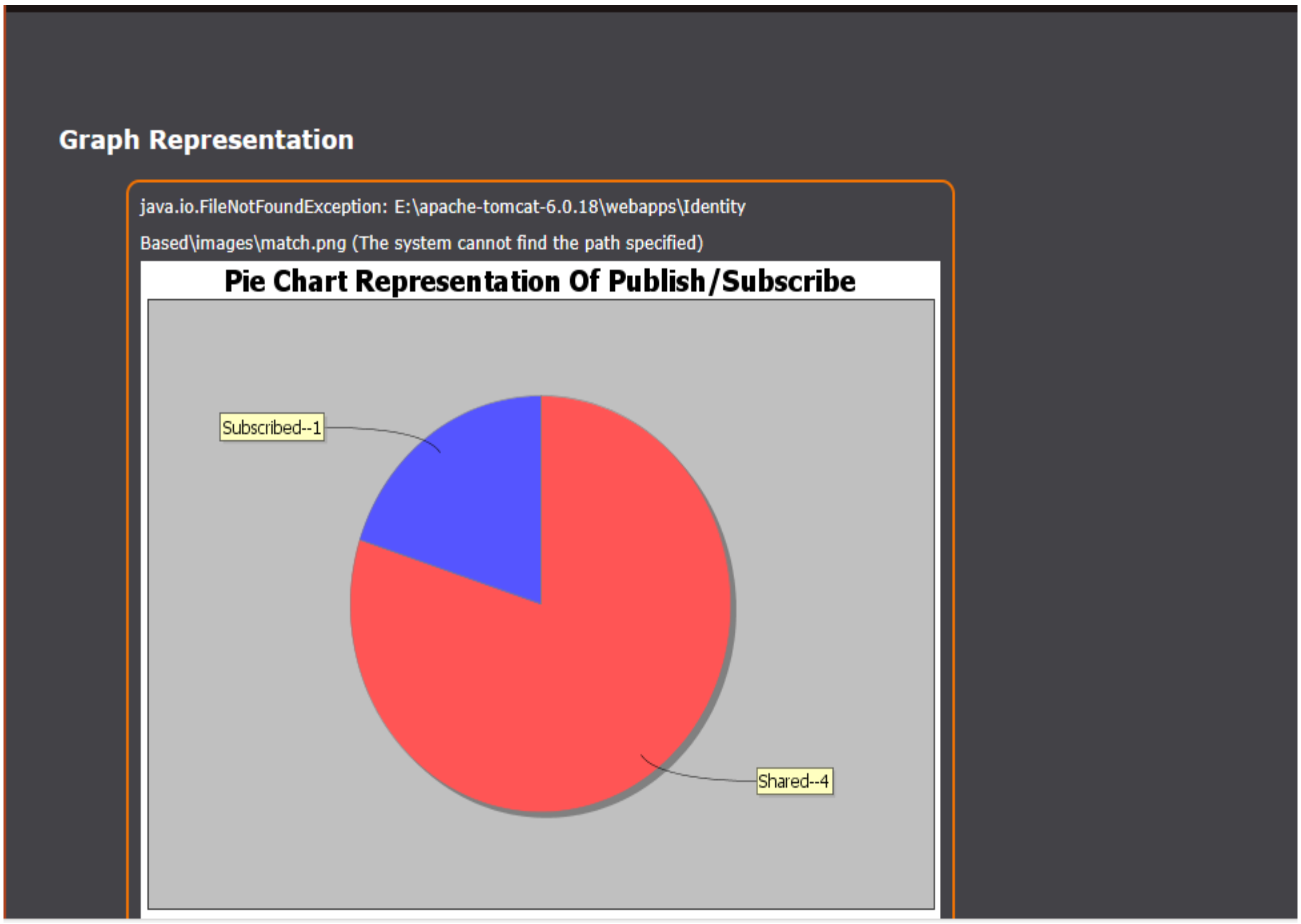
Master Key*

5949

Ciphertexts generated by encrypting event with the public keys of all credentials whose subscribers should receive the event.
$(Cred_A \| SUB \|..)$
$(Cred_B \| SUB \|..)$
$(Cred_{AB} \| SUB \|..)$
Sign with private key for $(Cred_{AB} \| PUB \|..)$
S_1
S_2
S_3
S_4
S_5
S_6
Pub/Sub network
Publisher
Event = A,B
Private Key for: $(Cred_{AB} \| PUB \|..)$
Advertisement= $Cred_{AB}$
Verify signature using public Key which corresponds to the publisher credentials: $(Cred_{AB} \| PUB \|..)$
Private Key for: $(Cred_A \| SUB \|..)$
Subscription = $Cred_A$
Key Server

Fig : publish secret message

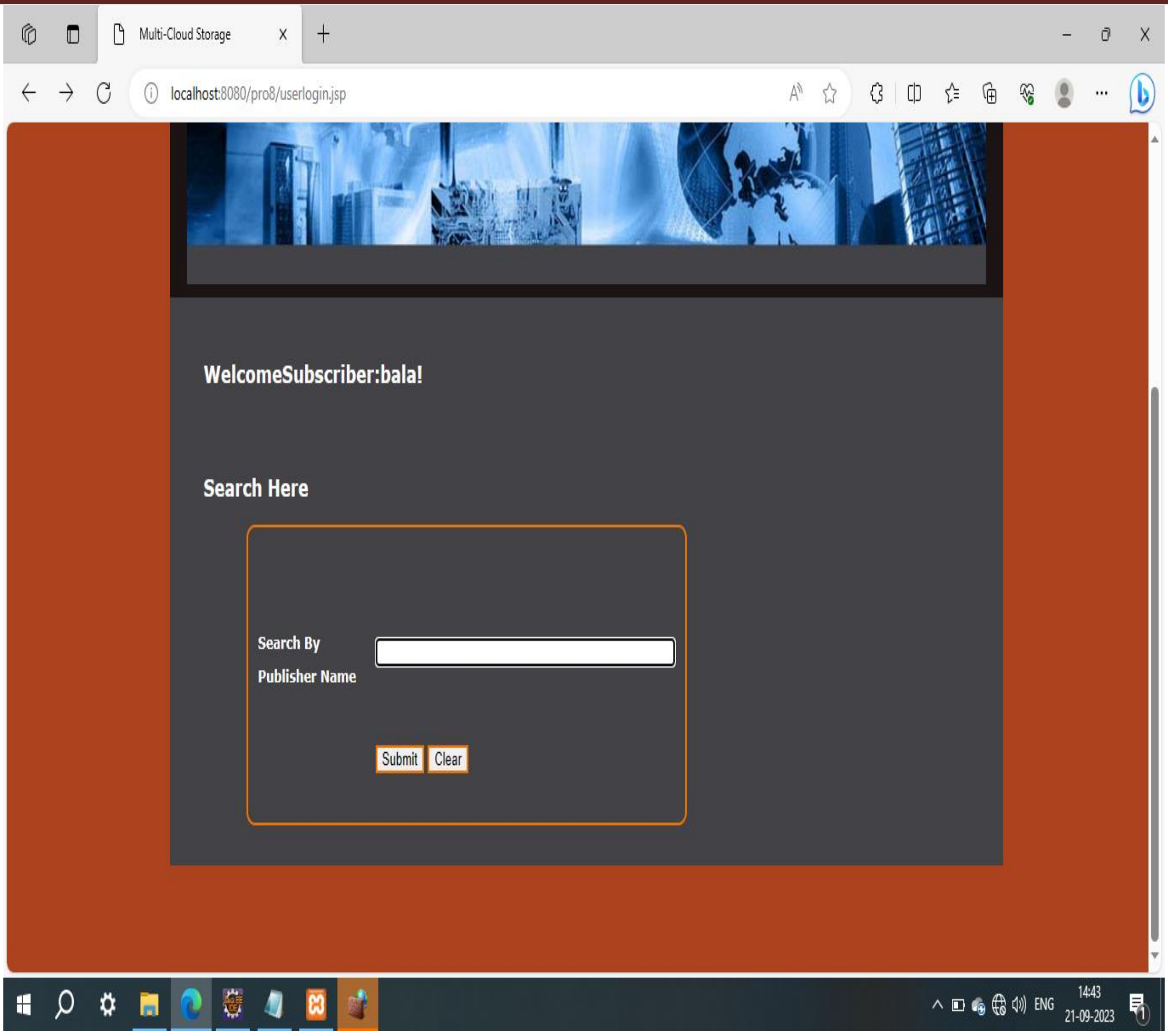
Screen Layout to Graph Representation :



```
java.io.FileNotFoundException: E:\apache-tomcat-6.0.18\webapps\Identity
Based\images\match.png (The system cannot find the path specified)
```

Fig : graph representation

Screen Layout to search Here :



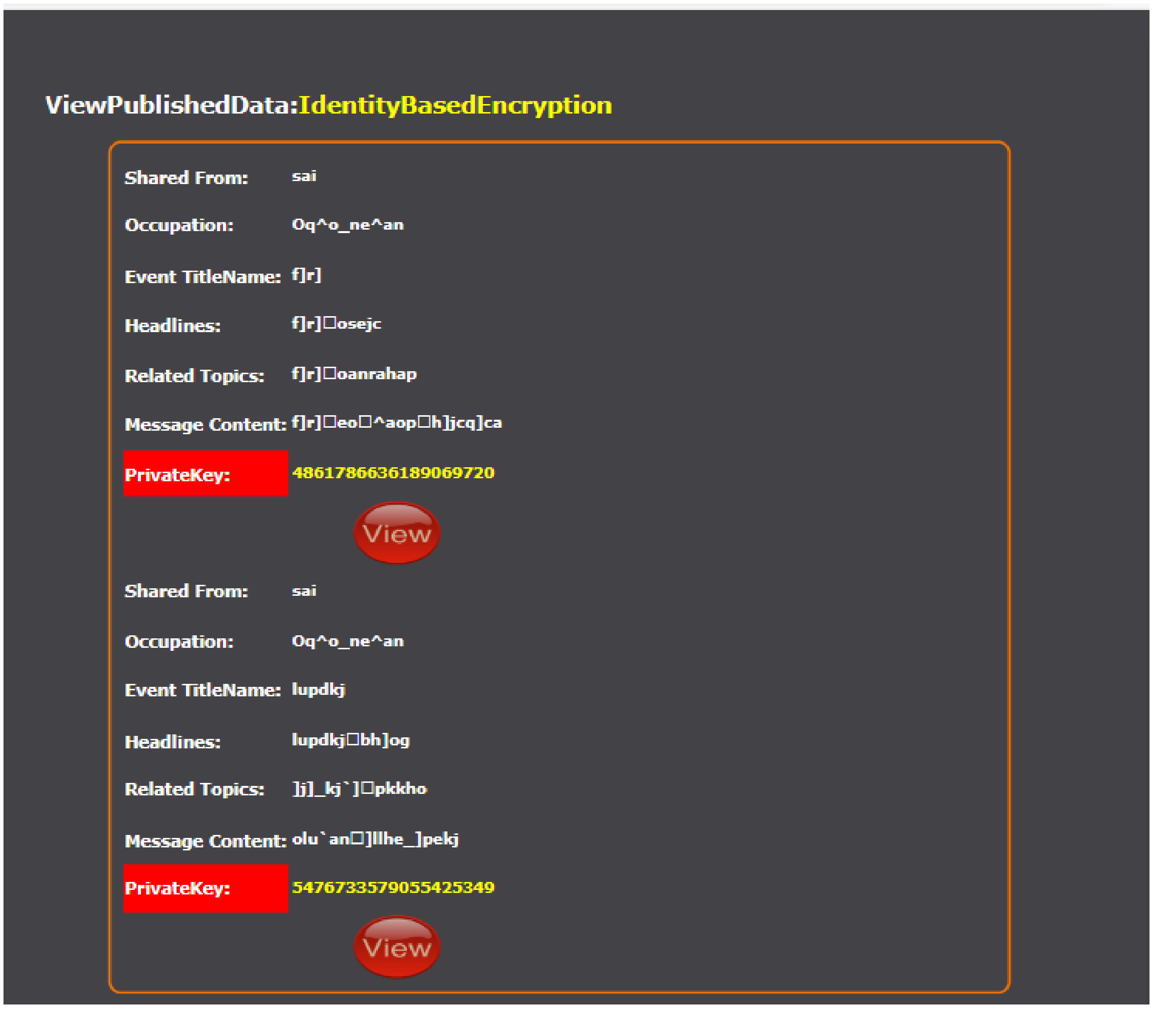
WelcomeSubscriber:bala!

Search Here

Search By Publisher Name

Fig : search here

Screen Layout to view published Data :



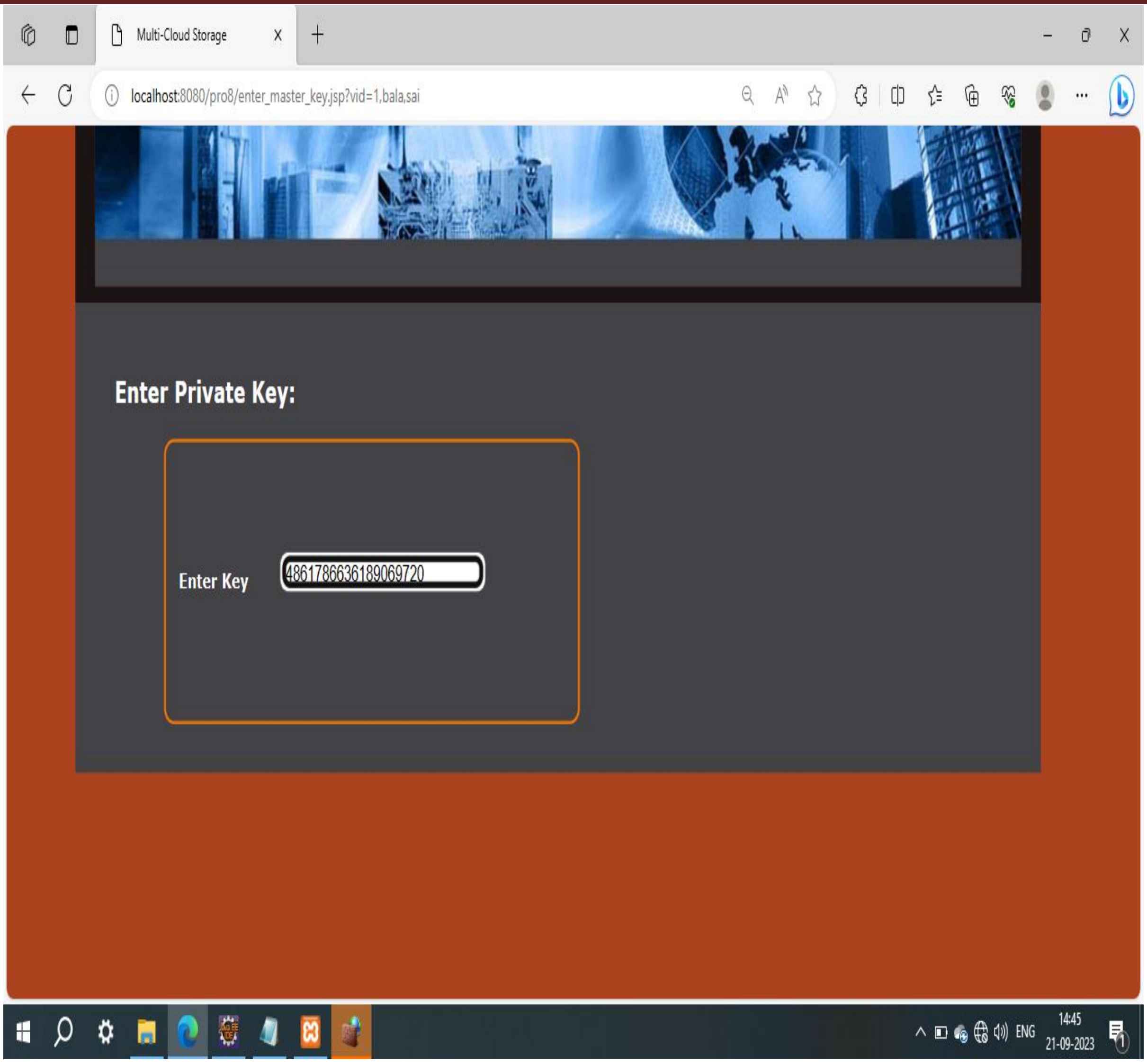
ViewPublishedData:IdentityBasedEncryption

Shared From:	sai
Occupation:	Oq^o_ne^an
Event TitleName:	f]r]
Headlines:	f]r]□osejc
Related Topics:	f]r]□oanrahap
Message Content:	f]r]□eo□^aop□h]jcq]ca
PrivateKey:	48617866361890697 20

Shared From:	sai
Occupation:	Oq^o_ne^an
Event TitleName:	lupdkj
Headlines:	lupdkj□bh]og
Related Topics:	]j]_kj`]□pkkho
Message Content:	olu`an□]llhe_]pekj
PrivateKey:	5476733579055425349

Fig :view published data

Screen Layout to private key:

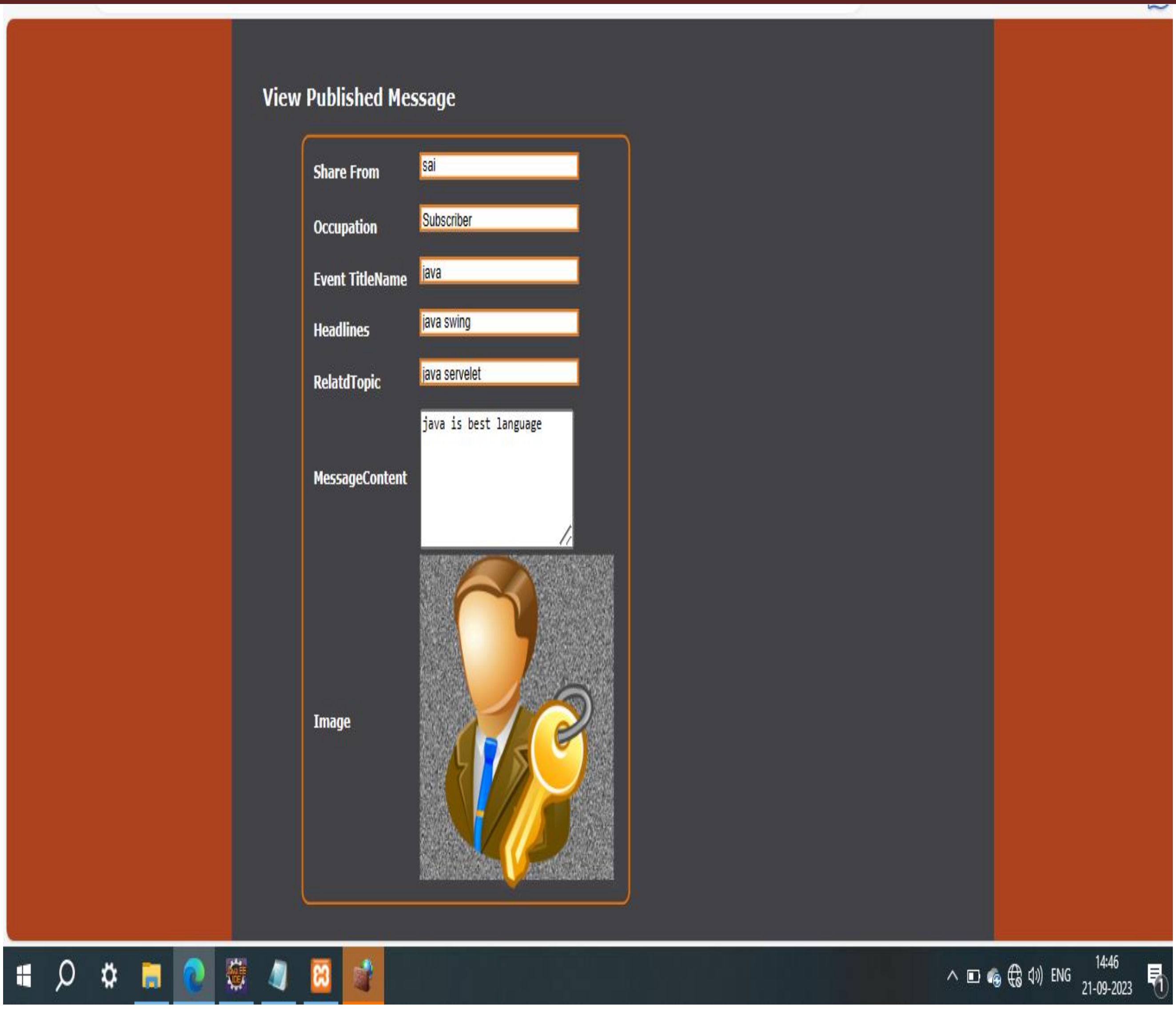


Enter Private Key:

Enter Key 4861786636189069720

Fig : private key

Screen Layout to view published Message :



View Published Message

Share From	sai
Occupation	Subscriber
Event TitleName	java
Headlines	java swing
RelatdTopic	java servelet
MessageContent	java is best language
Image	

Fig : view published message

9. CONCLUSION

9.1 CONCLUSION :

In this paper, we proposed GraphDApp, which can identify encrypted traffic of DApps using GNNs. We constructed the TIGs of encrypted flows based on the packet length and packet direction and turned the DApp identification into a graph classification problem. Then, we employed multi-layer perceptions to build a GNN-based classifier. Experiments are conducted to evaluate our method on real traffic datasets collected from DApps on Ethereum, including 40 monitored DApps and 1,260 unmonitored DApps. The experimental results show that GraphDApp can improve the accuracy by at least 10% over the state-of-the-art. We also demonstrated the effectiveness of GraphDApp on mobile application fingerprinting.

9.2 FUTURE ENHANCEMENT:

In the future work, we will further investigate techniques to make GraphDApp more adaptable to traffic changes.

Future work, we plan to make GraphDApp more adaptable to traffic changes and further improve its time efficiency.

10.REFERENCES

Good Teachers are worth more than thousand books, we have them in Our Department

References Made From:

1. User Interfaces in C#: Windows Forms and Custom Controls by Matthew MacDonald.

2. Applied Microsoft® .NET Framework Programming (Pro-Developer) by Jeffrey Richter.

3. Practical .Net2 and C#2: Harness the Platform, the Language, and the Framework by Patrick Smacchia.

4. Data Communications and Networking, by Behrouz A Forouzan.

5. Computer Networking: A Top-Down Approach, by James F. Kurose.

6. Operating System Concepts,by Abraham Silberschatz.

7. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski,G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia,“Above the clouds: A berkeley view of cloud computing,” University ofCalifornia, Berkeley, Tech. Rep. USB-EECS-2009-28, Feb 2009.

8. “The apache cassandra project,” http://cassandra.apache.org/.

9. L. Lamport, “The part-time parliament,” ACM Transactions on Computer Systems, vol. 16, pp. 133–169, 1998.

10. N. Bonvin, T. G. Papaioannou, and K. Aberer, “Cost-efficient and differentiated data availability guarantees in data clouds,” in Proc. of the ICDE, Long Beach, CA, USA, 2010.

11. O. Regev and N. Nisan, “The popcorn market. online markets for computational resources,” Decision Support Systems, vol. 28, no. 1-2, pp. 177 – 189, 2000.

12. A. Helsinger and T. Wright, "Cougaar: A robust configurablemulti agent platform," in Proc. of the IEEE Aerospace Conference, 2005.

13. J. Brunelle, P. Hurst, J. Huth, L. Kang, C. Ng, D. C. Parkes, M. Seltzer, J. Shank, and S. Youssef, "Egg: an extensible and economics-inspired open grid computing platform," in Proc. of the GECON, Singapore, May 2006.

14. J. Norris, K. Coleman, A. Fox, and G. Candea, "Oncall: Defeating spikes with a free-market application cluster," in Proc. of the International Conference on Autonomic Computing, New York, NY, USA, May 2004.

15. C. Pautasso, T. Heinis, and G. Alonso, "Autonomic resource provisioning for software business processes," Information and Software Technology, vol. 49, pp. 65–80, 2007.

16. A. Dan, D. Davis, R. Kearney, A. Keller, R. King, D. Kuebler, H. Ludwig, M. Polan, M. Spreitzer, and A. Youssef, "Web services on demand: Wsla-driven automated management," IBM Syst. J., vol. 43, no. 1, pp. 136–158, 2004.

17. M. Wang and T. Suda, "The bio-networking architecture: a biologically inspired approach to the design of scalable, adaptive, and survivable/available network applications," in Proc. of the IEEE Symposium on Applications and the Internet, 2001.

18. N. Laranjeiro and M. Vieira, "Towards fault tolerance in web services compositions," in Proc. of the workshop on engineering fault tolerant systems, New York, NY, USA, 2007.

19. C. Engelmann, S. L. Scott, C. Leangsuksun, and X. He, "Transparent symmetric active/active replication for servicelevel high availability," in Proc. of the CCGrid, 2007.

20. J. Salas, F. Perez-Sorrosal, n.-M. M. Pati and R. Jim´enez- Peris, "Ws-replication: a framework for highly available web services," in Proc. of the WWW, New York, NY, USA, 2006.