A
PROJECT
REPORT ON

"A Hybrid Approach for Detecting Automated Spammers in Twitter"

Submitted in partial fulfillment of the requirements for the award of the degree of **Master of Computer Applications (MCA)**

By

POLU SAI ASHISH
(H.No): 1009-19-862-032

Under the guidance of

Ms Humera Shaziya
Asst. Professor

Department of Informatics
Nizam College (Autonomous), (A Constituent College,O.U)
Basheerbagh, Hyderabad
(2021 – 22)

Nizam College (Autonomous), (A Constituent College,O.U)
Basheerbagh, Hyderabad
(2021 – 22)

CERTIFICATE

This is to certify that the project entitled " **A Hybrid Approach for Detecting Automated Spammers in Twitter** " bearing H.No: "**1009-19-862-032**"in the partial fulfillment of the requirements for the award of the degree of **Master of Computer Applications (MCA)**, Dept. of Informatics, Nizam college, Osmania University, Hyderabad.

PROJECT GUIDE

EXTERNAL EXAMINER

Ms. Humera Shaziya
Assistant Professor

Head of the Department
Dept. of Informatics.

FRATELLO INNOTECH PVT LTD

05/07/2022

CERTIFICATE

This is to certify that **Mr. POLU SAI ASHISH** studying in **MCA** at **Nizam college, Basheerbagh** with enrolment no **1009-19-862-032** successfully completed project work titled "**A Hybrid Approach for Detecting Automated Spammers in Twitter"** in Python technology as part of internship.

He has done this project using Python platform during the period from **03 March 2022** to **05 July 2022** under the guidance and supervision of **Mr. Bhargav Kumar, Sr. Software Developer Fratello InnoTech Private Limited,** Hyderabad.

She has completed the assigned project well within the time frame and is sincere, hardworking and her conduct during her project is commendable.

We wish all the best to her future endeavors

For **Fratello InnoTech Pvt Ltd**

FRATELLO INNOTECH PVT. LTD. ★ HYDERABAD

[illegible]

040-40044139
info@fratelloinnotech.com
fratelloinnotech@gmail.com
www.fratelloinnotech.com
301, 3rd Floor, LIG 630, K.P.H.B
Road No - 5, Hyderabad - 500 072

Training Consulting Development Internships

Acknowledgement

The success and final outcome of this project required a lot of guidance and assistance from many people and I am extremely fortunate to have got this all along the completion of my project work. Whatever I have done is only due to such guidance and assistance and I would not forget to thank them.

I would like to thank the **ALMIGHTY,** who gave me enough strength and health to complete this task without any interruption and **My Parents** who gave me all the support during the project work.

I owe my profound gratitude to my project guide **Ms Humera Shaziya** , Asst. Professor and other teachers who took keen interest on our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good system.

I thank to **Ms SHRAVANTHI,** Head of the Dept. of Computer Science (PG), Nizam College, For her moral Support and Guidance.

I express my whole hearted gratitude to **Prof B. Bhima** and to **Management of Nizam College** for providing us the conductive environment to carry through our academic schedules and project with ease.

I am thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staffs of Department of computer science which helped us in successfully completing our project work. Also, I would like to extend our sincere regards to all the non-teaching staff of department of computer science for their timely support

Last but not the least; I would like to thank all my friends for their compliment

Name	**:**	**POLU SAI ASHISH**
Roll No	**:**	**1009-19-862-032**

ABSTRACT

A Hybrid Approach for Detecting Automated Spammers in Twitter

Twitter is one of the most popular micro blogging services, which is generally used to share news and updates through short messages restricted to 280 characters. However, its open nature and large user base are frequently exploited by automated spammers, content polluters, and other ill-intended users to commit various cybercrimes, such as cyber bullying, trolling, rumor dissemination, and stalking. Accordingly, a number of approaches have been proposed by researchers to address these problems.
However, most of these approaches are based on user characterization and completely disregarding mutual interactions. In this paper, we present a hybrid approach for detecting automated spammers by amalgamating community based features with other feature categories, namely metadata-, content-, and interaction-based features. The novelty of the proposed approach lies in the characterization of users based on their interactions with their followers given that a user can evade features that are related to his/her own activities, but evading those based on the followers is difficult. Nineteen different features, including six newly defined features and two redefined features, are identified for learning three classifiers, namely, random forest, decision tree, and Bayesian network, on a real dataset that comprises benign users and spammers. The discrimination power of different feature categories is also analyzed, and interaction- and community-based features are determined to be the most effective for spam detection, whereas metadata-based features are proven to be the least effective.

Table of Contents

INTRODUCTION

TWITTER, a microblogging service, is considered a popular online social network (OSN) with a large user base and is attracting users from different walks of life and age groups. OSNs enable users to keep in touch with friends,relatives, family members, and people with similar interests, profession, and objectives. In addition, they allow users to interact with one another and form communities. A user can become a member of an OSN by registering and providing details, such as name, birthday, gender, and other contact information. Although a large number of OSNs exist on the web, Facebook and Twitter are among the most popular OSNs and are included in the list of the top 10 websites1 around the worldwide.

A. OSN and the Social Spam Problem

Twitter, which was founded in 2006, allows its users to post their views, express their thoughts, and share news and other information in the form of tweets that are restricted to280 characters. Twitter allows the users to follow their favourite politicians, athletes, celebrities, and news channels, and to subscribe to their content without any hindrance. Through *following* activity, a follower can receive status updates of subscribed account. Although Twitter and other OSNs are mainly used for various benign purposes, their open nature, huge user base, and real-time message proliferation have made them lucrative targets for cyber criminals and social bots. OSNs have been proven to be incubators for a new breed of complex and sophisticated attacks and threats, such as cyberbullying, misinformation diffusion, stalking, identity deception, radicalization, and other illicit activities, in addition to classical cyber attacks, such as spamming, phishing, and drive by download [1], [2]. Over the years, classical attacks have evolved into sophisticated attacks to evade detection mechanisms. A report2submitted to the US Securities and Exchange Commission in August 2014 indicates that approximately 14% of Twitter accounts are actually spambots and approximately 9.3% of all tweets are spam. In social networks, spambots are also known as socialbots that mimic human behaviour to gain trust in a network and then exploit it for malicious activities [3]. Such reports and findings demonstrate the extent of cyber crimes committed by spambots and how OSNs are proving to be a heaven for these bots. Although spammers are less than benign users, they are capable of affecting network structure and trust for various illicit purposes.

Modules used are user and an admin.

The main contributions of this study can be summarized as follows.

• A novel study that uses community-based features with other feature categories, including *metadata*, *content*, and *interaction*, for detecting automated spammers.

• Six new features are introduced and two existing features are redefined to design a feature set with improved discriminative power for segregating benign users and spammers. Among the six new features, one is content based, three are interaction-based, and the remaining two are community-based. Meanwhile, both redefined features are content-based. When defining interaction based features, focus should be on the *followers* of a user, rather than on the ones he/she is *followings*.
• A detailed analysis of the working behavior of automated spammers and benign users with respect to newly defined features. In addition, two-tailed *Z*-test statistical significance analysis is performed to answer the following question: “*is the difference between the working behavior of spammers and benign users in terms of newly defined features a random chance?*”
• A thorough analysis of the discriminating power of each feature category in segregating automated spammers from benign users.

SYSYEM ANALYSIS

Existing system

- Sahami et al. [14] proposed textual and non textual and domain-specific features and learned naive Bayes classifier to segregate spam emails from legitimate ones. Schafer [15] and [16] proposed metadata-based approaches to detect botnets based on compromised email accounts to diffuse mail spams. Spam campaigns on Facebook were analyzed by Gao et al. [10] using a similarity graph based on semantic similarity between posts and URLs that point to the same destination.
- Furthermore, they extracted clusters from a similarity graph, wherein each cluster represents a specific spam campaign. Upon analysis, they determined that most spam sources were hijacked accounts, which exploited the trust of users to redirect legitimate users to phishing sites. In [7] and [8], honey profiles were created and deployed on OSNs to observe the behavior of spammer. Both studies presented different sets of features to discriminate benign users from spammers and evaluated them on different sets of OSNs.
- Wang [17] used content and graph-based features to classify malicious and normal profiles on Twitter. In contrast to honey profiles, Wang used Twitter API to crawl the dataset. Yang et al. [12], Wang [17], and Ahmed and Abulaish [18], used content- and interaction based attributes for learning classifiers to segregate spammers from benign users on different OSNs.
- Yang et al. [12] and Ahmed and Abulaish [18] analyzed the contribution of each feature to spammer detection, whereas Yang et al. [19] conducted an in-depth empirical analysis of the evasive tactics practiced by spammers to bypass detection systems. They also tested the robustness of newly devised features. In [20], Zhu et al. used a matrix factorization technique to find the latent features from the sparse activity matrix and adopted social regularization to learn the spam discriminating power of the classifier on the Renren network, one of the most popular OSNs in China. Another spammer detection approach in social media was proposed by Tan et al. [21].

Proposed system

- In the proposed system, the system proposes a hybrid approach for detecting social spam bots in Twitter, which utilizes an amalgamation of metadata-, content-, interaction-, and community-based features. In the analysis of characterizing features of existing approaches, most network-based features are not defined using user followers and underlying community structures, thereby disregarding the fact that the reputation of user in a network is inherited from the followers (rather than from the ones user is following) and community members. Therefore, the system emphasizes the use of followers and community structures to define the network-based features of a user.

- ❖ The system classifies set of features into three broad categories, namely, metadata, content, and network, wherein the network category is further classified into interaction- and community based features. Metadata features are extracted from available additional information regarding the tweets of a user, whereas content-based features aim to observe the message posting behavior of a user and the quality of the text that the user uses in posts. Network-based features are extracted from user interaction network.

Preliminary investigation

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

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

Request clarification

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

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

Feasibility study

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

Three key considerations involved in the feasibility analysis are

- ♦ ECONOMICAL FEASIBILITY
- ♦ TECHNICAL FEASIBILITY
- ♦ SOCIAL FEASIBILITY

Economical feasibility

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

Technical feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

Social feasibility

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

Feasibility analysis

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

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

Operational Feasibility

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

Economic Feasibility

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

Technical Feasibility

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

Request approval

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

Advantages

- A novel study that uses community-based features with other feature categories, including metadata, content, and interaction, for detecting automated spammers.
- Used Hybrid technique to classify spammers such as random forest, decision tree, and Bayesian network.

Disadvantages

- There are no Hybrid techniques to classify different spam’s behaviours.
- There is no spambot detection techniques.

SYSTEM REQUIREMENTS

Hardware System Requirements :

➢ Processor	-	Pentium –IV
➢ RAM	-	4 GB (min)
➢ Hard Disk	-	20 GB
➢ Key Board	-	Standard Windows Keyboard
➢ Mouse	-	Two or Three Button Mouse
➢ Monitor	-	SVGA

Software Requirements :

➢ Operating System	-	Windows XP
➢ Coding Language	-	Java/J2EE(JSP,Servlet)
➢ Front End	-	J2EE
➢ Back End	-	MySQ

Process Model - WATERFALL MODEL

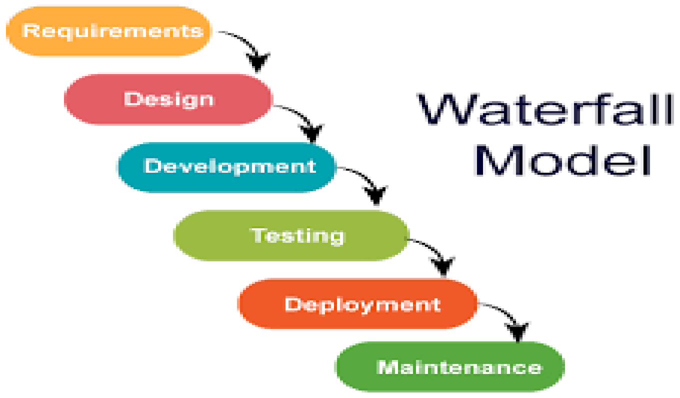
Waterfall model indicates that all the SDLC steps will be going on step by step. In this model once one stage completed then only it will start the another process. This will be implemented when the requirements well known and clearly defined.

In this model we can implement all the setps one by one. First we will do requirements gathering i.e in this step business analyst will collect the all functional requirements from the client and will discuss with the manager and development team. Second step is analysis where people can analyses the requirements so that they will do risk basement, risk mitigation, risk controlling steps. And also they will do the cost-time analysis to estimate the cost. They will also do the detail feasible study over the requirements of project.

In third phase called design page generally people will design the project i.e represent will prepare the uml diagrams and prototypes for project. And then developers will start programing or coding based on design . Finally testing team will test the content to identify and fix the bugs. After successful user acceptance testing application will be deployed and maintained in the client machine.

The waterfall model was selected as the SDLC model due to the following reasons:

- When the requirements are very clear and not changed.
- Available technology is enough to implement.
- Project is easy and simple to understand to finish.
- And if we don’t have any confusion in requirements.
- We can easily manage the review process and task management.
- Every stage defined clearly and with timelines



<u>SYSTEM DESIGN AND DEVELOPMENT</u>

Input design

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

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

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

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

Output design

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

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

TECHNOLOGIES USED

Front End:

Ellipse IDE for enterprise java and web developers 2022-03

Java SE Development Kit 18.0.1.1

HTML

CSS

Back End:

SQL yog community 13.1.6

Apache tomcat

MySQL installer and community1.6.2.0

MySQL server 8.0.29

WORKING ENIVRONMENT

Client Server

Over view:

With the varied topic in existence in the fields of computers, Client Server is one, which has generated more heat than light, and also more hype than reality. This technology has acquired a certain critical mass attention with its dedication conferences and magazines. Major computer vendors such as IBM and DEC, have declared that Client Servers is their main future market. A survey of DBMS magazine reveled that 76% of its readers were actively looking at the client server solution. The growth in the client server development tools from $200 million in 1992 to more than $1.2 billion in 1996.

Client server implementations are complex but the underlying concept is simple and powerful. A client is an application running with local resources but able to request the database and relate the services from separate remote server. The software mediating this client server interaction is often referred to as MIDDLEWARE.

The typical client either a PC or a Work Station connected through a network to a more powerful PC, Workstation, Midrange or Main Frames server usually capable of handling request from more than one client. However, with some configuration server may also act as client. A server may need to access other server in order to process the original client request.

The key client server idea is that client as user is essentially insulated from the physical location and formats of the data needs for their application. With the proper middleware, a client input from or report can transparently access and manipulate both local database on the client machine and remote databases on one or more servers. An added bonus is the client server opens the door to multi-vendor database access indulging heterogeneous table joins.

What is a Client Server

Two prominent systems in existence are client server and file server systems. It is essential to distinguish between client servers and file server systems. Both provide shared network access to data but the comparison dens there! The file server simply provides a remote disk drive that can be accessed by LAN applications on a file by file basis. The client server offers full relational database services such as SQL-Access, Record modifying, Insert, Delete with full relational integrity backup/

restore performance for high volume of transactions, etc. the client server middleware provides a flexible interface between client and server, who does what, when and to whom.

Why Client Server

Client server has evolved to solve a problem that has been around since the earliest days of computing: how best to distribute your computing, data generation and data storage resources in order to obtain efficient, cost effective departmental an enterprise wide data processing. During mainframe era choices were quite limited. A central machine housed both the CPU and DATA (cards, tapes, drums and later disks). Access to these resources was initially confined to batched runs that produced departmental reports at the appropriate intervals. A strong central information service department ruled the corporation. The role of the rest of the corporation limited to requesting new or more frequent reports and to provide hand written forms from which the central data banks were created and updated. The earliest client server solutions therefore could best be characterized as “SLAVE-MASTER”.

Time-sharing changed the picture. Remote terminal could view and even change the central data, subject to access permissions. And, as the central data banks evolved in to sophisticated relational database with non-programmer query languages, online users could formulate adhoc queries and produce local reports with out adding to the MIS applications software backlog. However remote access was through dumb terminals, and the client server remained subordinate to the Slave\Master.

Time-sharing changed the picture. Remote terminal could view and even change the central data, subject to access permissions. And, as the central data banks evolved in to sophisticated relational database with non-programmer query languages, online users could formulate adhoc queries and produce local reports with out adding to the MIS applications software backlog. However remote access was through dumb terminals, and the client server remained subordinate to the Slave\Master.

Front end or User Interface Design

Communication or Database Connectivity Tier

The Communication architecture is designed by concentrating on the Standards of Servlets and Enterprise Java Beans. The database connectivity is established by using the Java Data Base Connectivity.

The standards of three-tire architecture are given major concentration to keep the standards of higher cohesion and limited coupling for effectiveness of the operations.

Features of The Language Used

In my project, I have chosen Java language for developing the code.

About Java

Initially the language was called as “oak” but it was renamed as “Java” in 1995. The primary motivation of this language was the need for a platform-independent (i.e., architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

- Java is a programmer’s language.
- Java is cohesive and consistent.
- Except for those constraints imposed by the Internet environment, Java gives the programmer, full control.

Finally, Java is to Internet programming where C was to system programming.

Importance of Java to the Internet

Java has had a profound effect on the Internet. This is because; Java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the Server and the Personal computer. They are: Passive information and Dynamic active programs. The Dynamic, Self-executing programs cause serious problems in the areas of Security and probability. But, Java addresses those concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

Java can be used to create two types of programs

Applications and Applets: An application is a program that runs on our Computer under the operating system of that computer. It is more or less like one creating using C or C++. Java’s ability to create Applets makes it important. An Applet is an application designed to be transmitted over the Internet and executed by a Java –compatible web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can react to the user input and dynamically change.

Features Of Java

Security

Every time you that you download a “normal” program, you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scanned them for viruses prior to execution. Most users still worried about the possibility of infecting their systems with a virus. In addition, another type of malicious program exists that must be guarded against. This type of program can gather private information, such as credit card numbers, bank account balances, and passwords. Java answers both these concerns by providing a “firewall” between a network application and your computer.

When you use a Java-compatible Web browser, you can safely download Java applets without fear of virus infection or malicious intent.

Portability

For programs to be dynamically downloaded to all the various types of platforms connected to the Internet, some means of generating portable executable code is needed .As you will see, the same mechanism that helps ensure security also helps create portability. Indeed, Java’s solution to these two problems is both elegant and efficient.

The Byte code

The key that allows the Java to solve the security and portability problems is that the output of Java compiler is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreter for byte code.

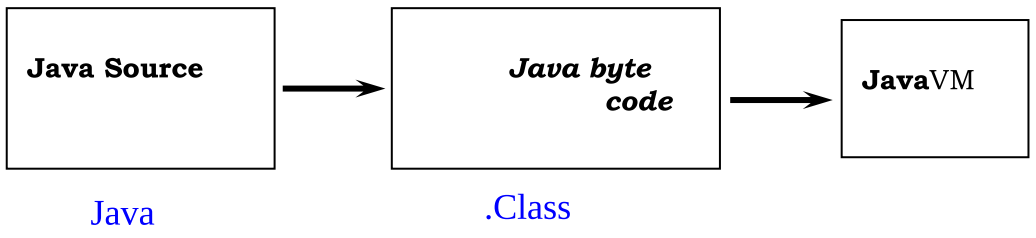
Translating a Java program into byte code helps makes it much easier to run a program in a wide variety of environments. The reason is, once the run-time package exists for a given system, any Java program can run on it.

Although Java was designed for interpretation, there is technically nothing about Java that prevents on-the-fly compilation of byte code into native code. Sun has just completed its Just In Time (JIT) compiler for byte code. When the JIT compiler is a part of JVM, it compiles byte code into executable code in real time, on a piece-by-piece, demand basis. It is not possible to compile an entire Java program into executable code all at once, because Java performs various run-time checks that can be done only at run time. The JIT compiles code, as it is needed, during execution.

Java, Virtual Machine (JVM)

Beyond the language, there is the Java virtual machine. The Java virtual machine is an important element of the Java technology. The virtual machine can be embedded within a web browser or an operating system. Once a piece of Java code is loaded onto a machine, it is verified. As part of the loading process, a class loader is invoked and does byte code verification makes sure that the code that’s has been generated by the compiler will not corrupt the machine that it’s loaded on. Byte code verification takes place at the end of the compilation process to make sure that is all accurate and correct. So byte code verification is integral to the compiling and executing of Java code.

Overall Description



Picture showing the development process of JAVA Program

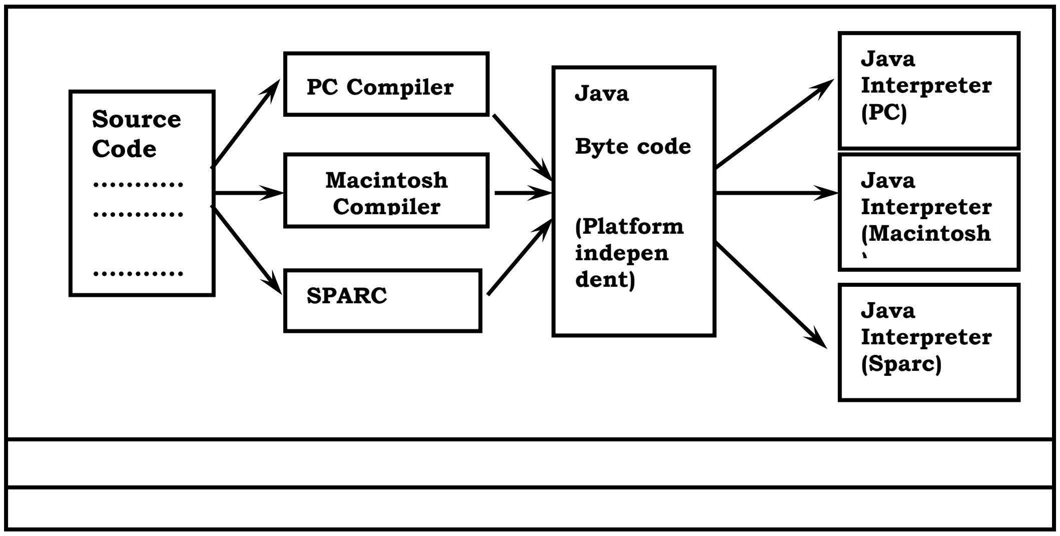
Java programming uses to produce byte codes and executes them. The first box indicates that the Java source code is located in a. Java file that is processed with a Java compiler called javac. The Java compiler produces a file called a. class file, which contains the byte code. The. Class file is then loaded across the network or loaded locally on your machine into the execution environment is the Java virtual machine, which interprets and executes the byte code.

Java Architecture

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

Compilation of code

When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of portability. The code is written and compiled for one machine and interpreted on all machines. This machine is called Java Virtual Machine.



Compiling and interpreting Java Source Code

During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be a Intel Pentium Windows 95 or Sun SARC station running Solaris or Apple Macintosh running system and all could receive code from any computer through Internet and run the Applets.

Simple

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ programmer, learning Java will be even easier. Because Java inherits the C/C++ syntax and many of the object oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

Object-Oriented

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

Robust

The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time.

Java virtually eliminates the problems of memory management and deallocation, which is completely automatic. In a well-written Java program, all run time errors can –and should –be managed by your program.

JAVASCRIPT

JavaScript is a script-based programming language that was developed by Netscape Communication Corporation. JavaScript was originally called Live Script and renamed as JavaScript to indicate its relationship with Java. JavaScript supports the development of both client and server components of Web-based applications. On the client side, it can be used to write programs that are executed by a Web browser within the context of a Web page. On the server side, it can be used to write Web server programs that can process information submitted by a Web browser and then updates the browser's display accordingly.

Even though JavaScript supports both client and server Web programming, we prefer JavaScript at Client side programming since most of the browsers supports it. JavaScript is almost as easy to learn as HTML, and JavaScript statements can be included in HTML documents by enclosing the statements between a pair of scripting tags

```
<SCRIPTS>..</SCRIPT>.

<SCRIPT LANGUAGE = “JavaScript”>

JavaScript statements

</SCRIPT>
```

Here are a few things we can do with JavaScript :

- Validate the contents of a form and make calculations.
- Add scrolling or changing messages to the Browser's status line.
- Animate images or rotate images that change when we move the mouse over them.
- Detect the browser in use and display different content for different browsers.
- Detect installed plug-ins and notify the user if a plug-in is required.

We can do much more with JavaScript, including creating entire application.

JAVASCRIPT vs JAVA

JavaScript and Java are entirely different languages. A few of the most glaring differences are:

- Java applets are generally displayed in a box within the web document; JavaScript can affect any part of the Web document itself.

- While JavaScript is best suited to simple applications and adding interactive features to Web pages; Java can be used for incredibly complex applications.

There are many other differences but the important thing to remember is that JavaScript and Java are separate languages. They are both useful for different things; in fact they can be used together to combine their advantages.

Advantages

- JavaScript can be used for Sever-side and Client-side scripting.
- It is more flexible than VBScript.
- JavaScript is the default scripting languages at Client-side since all the browsers supports it.

Hyper Text Markup Language

Hypertext Markup Language (HTML), the languages of the World Wide Web (WWW), allows users to produces Web pages that include text, graphics and pointer to other Web pages (Hyperlinks).

HTML is not a programming language but it is an application of ISO Standard 8879, SGML (Standard Generalized Markup Language), but specialized to hypertext and adapted to the Web. The idea behind Hypertext is that instead of reading text in rigid linear structure, we can easily jump from one point to another point. We can navigate through the information based on our interest and preference. A markup language is simply a series of elements, each delimited with special characters that define how text or other items enclosed within the elements should be displayed. Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same document.

HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop.

HTML provides tags (special codes) to make the document look attractive. HTML tags are not case-sensitive. Using graphics, fonts, different sizes, color, etc., can enhance the presentation of the document. Anything that is not a tag is part of the document itself.

Basic HTML Tags :

<!-- -->	Specifies comments
<A>……….</A>	Creates hypertext links
<B>……….</B>	Formats text as bold
<BIG>……….</BIG>	Formats text in large font.

<BODY>…</BODY>	Contains all tags and text in the HTML document
<CENTER>...</CENTER>	Creates text
<DD>…</DD>	Definition of a term
<DL>...</DL>	Creates definition list
<FONT>…</FONT>	Formats text with a particular font
<FORM>...</FORM>	Encloses a fill-out form
<FRAME>...</FRAME>	Defines a particular frame in a set of frames
<H#>…</H#>	Creates headings of different levels
<HEAD>...</HEAD>	Contains tags that specify information about a document
<HR>...</HR>	Creates a horizontal rule
<HTML>…</HTML>	Contains all other HTML tags
<META>...</META>	Provides meta-information about a document
<SCRIPT>…</SCRIPT>	Contains client-side or server-side script
<TABLE>…</TABLE>	Creates a table
<TD>…</TD>	Indicates table data in a table
<TR>…</TR>	Designates a table row
<TH>…</TH>	Creates a heading in a table

Advantages

- ➢ A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.
- ➢ HTML is platform independent.
- ➢ HTML tags are not case-sensitive.

Java Database Connectivity

What Is JDBC?

JDBC is a Java API for executing SQL statements. (As a point of interest, JDBC is a trademarked name and is not an acronym; nevertheless, JDBC is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API.

Using JDBC, it is easy to send SQL statements to virtually any relational database. One can write a single program using the JDBC API, and the program will be able to send SQL statements to the appropriate database. The combinations of Java and JDBC lets a programmer write it once and run it anywhere.

What Does JDBC Do?

Simply put, JDBC makes it possible to do three things:

- ➢ Establish a connection with a database
- ➢ Send SQL statements
- ➢ Process the results.

JDBC versus ODBC and other API's

At this point, Microsoft's ODBC (Open Database Connectivity) API is that probably the most widely used programming interface for accessing relational databases. It offers the ability to connect to almost all databases on almost all platforms.

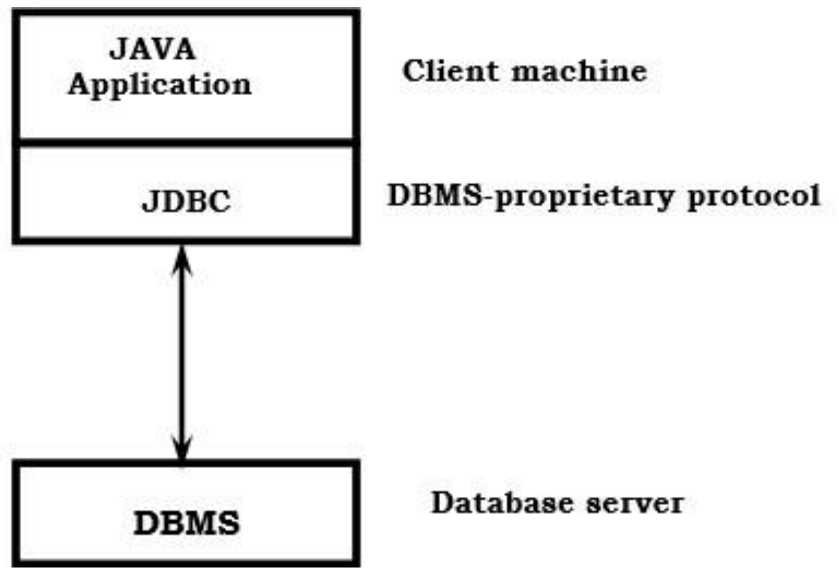
So why not just use ODBC from Java? The answer is that you can use ODBC from Java, but this is best done with the help of JDBC in the form of the JDBC-ODBC Bridge, which we will cover shortly. The question now becomes "Why do you need JDBC?" There are several answers to this question:

1. ODBC is not appropriate for direct use from Java because it uses a C interface. Calls from Java to native C code have a number of drawbacks in the security, implementation, robustness, and automatic portability of applications.
2. A literal translation of the ODBC C API into a Java API would not be desirable. For example, Java has no pointers, and ODBC makes copious use of them, including the notoriously error-prone generic pointer "void *". You can think of JDBC as ODBC translated into an object-oriented interface that is natural for Java programmers.
3. ODBC is hard to learn. It mixes simple and advanced features together, and it has complex options even for simple queries. JDBC, on the other hand, was designed to keep simple things simple while allowing more advanced capabilities where required.
4. A Java API like JDBC is needed in order to enable a "pure Java" solution. When ODBC is used, the ODBC driver manager and drivers must be manually installed on every client machine. When the JDBC driver is written completely in Java, however, JDBC code is automatically installable, portable, and secure on all Java platforms from network computers to mainframes.

Two-tier and Three-tier Models

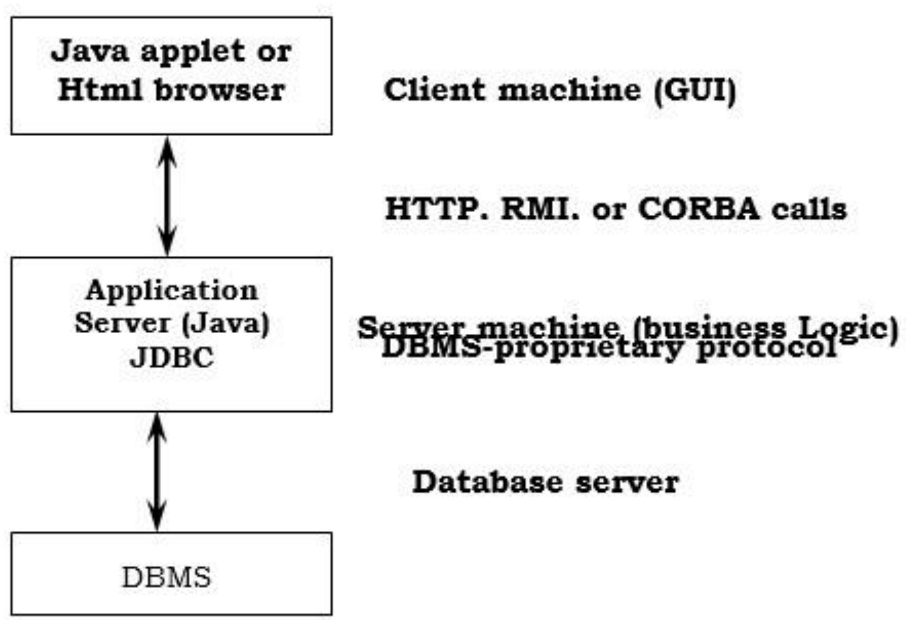
The JDBC API supports both two-tier and three-tier models for database access.

In the two-tier model, a Java applet or application talks directly to the database. This



requires a JDBC driver that can communicate with the particular database management system being accessed. A user's SQL statements are delivered to the database, and the results of those statements are sent back to the user. The database may be located on another machine to which the user is connected via a network. This is referred to as a client/server configuration, with the user's machine as the client, and the machine housing the database as the server. The network can be an Intranet, which, for example, connects employees within a corporation, or it can be the Internet.

In the three-tier model, commands are sent to a "middle tier" of services, which then send SQL statements to the database. The database processes the SQL statements and sends the results back to the middle tier, which then sends them to the user. MIS


Java applet or Html browser
Client machine (GUI)
HTTP. RMI. or CORBA calls
Application Server (Java) JDBC
Server machine (business Logic)
DBMS-proprietary protocol
Database server
DBMS

directors find the three-tier model very attractive because the middle tier makes it possible to maintain control over access and the kinds of updates that can be made to corporate data. Another advantage is that when there is a middle tier, the user can employ an easy-to-use higher-level API which is translated by the middle tier into the appropriate low-level calls. Finally, in many cases the three-tier architecture can provide performance advantages.

Until now the middle tier has typically been written in languages such as C or C++, which offer fast performance. However, with the introduction of optimizing compilers that translate Java byte code into efficient machine-specific code, it is becoming practical to implement the middle tier in Java. This is a big plus, making it possible to take advantage of Java's robustness, multithreading, and security features. JDBC is important to allow database access from a Java middle tier.

JDBC Driver Types

The JDBC drivers that we are aware of at this time fit into one of four categories:

- JDBC-ODBC bridge plus ODBC driver
- Native-API partly-Java driver
- JDBC-Net pure Java driver
- Native-protocol pure Java driver

JDBC-ODBC Bridge

If possible, use a Pure Java JDBC driver instead of the Bridge and an ODBC driver. This completely eliminates the client configuration required by ODBC. It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge (that is, the Bridge native library, the ODBC driver manager library, the ODBC driver library, and the database client library).

What Is the JDBC- ODBC Bridge?

The JDBC-ODBC Bridge is a JDBC driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge implements JDBC for any database for which an ODBC driver is available. The Bridge is implemented as the
sun.jdbc.odbc Java package and contains a native library used to access ODBC. The Bridge is a joint development of Intersolv and JavaSoft.

Java Server Pages (JSP)

Java server Pages is a simple, yet powerful technology for creating and maintaining dynamic-content web pages. Based on the Java programming language, Java Server Pages offers proven portability, open standards, and a mature re-usable component model .The Java Server Pages architecture enables the separation of content generation from content presentation. This separation not eases maintenance headaches, it also allows web team members to focus on their areas of expertise. Now, web page designer can concentrate on layout, and web application designers on programming, with minimal concern about impacting each other's work.

Features of JSP

Portability:

Java Server Pages files can be run on any web server or web-enabled application server that provides support for them. Dubbed the JSP engine, this support involves recognition, translation, and management of the Java Server Page lifecycle and its interaction components.

Components

It was mentioned earlier that the Java Server Pages architecture can include reusable Java components. The architecture also allows for the embedding of a scripting language directly into the Java Server Pages file. The components current supported include Java Beans, and Servlets.

Processing

A Java Server Pages file is essentially an HTML document with JSP scripting or tags. The Java Server Pages file has a JSP extension to the server as a Java Server Pages file. Before the page is served, the Java Server Pages syntax is parsed and processed into a Servlet on the server side. The Servlet that is generated outputs real content in straight HTML for responding to the client.

Access Models

A Java Server Pages file may be accessed in at least two different ways. A client's request comes directly into a Java Server Page. In this scenario, suppose the page accesses reusable Java Bean

components that perform particular well-defined computations like accessing a database. The result of the Beans computations, called result sets is stored within the Bean as properties. The page uses such Beans to generate dynamic content and present it back to the client.
In both of the above cases, the page could also contain any valid Java code. Java Server Pages architecture encourages separation of content from presentation.

Steps in the execution of a JSP Application:

1. The client sends a request to the web server for a JSP file by giving the name of the JSP file within the form tag of a HTML page.
2. This request is transferred to the JavaWebServer. At the server side JavaWebServer receives the request and if it is a request for a jsp file server gives this request to the JSP engine.
3. JSP engine is program which can understands the tags of the jsp and then it converts those tags into a Servlet program and it is stored at the server side. This Servlet is loaded in the memory and then it is executed and the result is given back to the JavaWebServer and then it is transferred back to the result is given back to the JavaWebServer and then it is transferred back to the client.

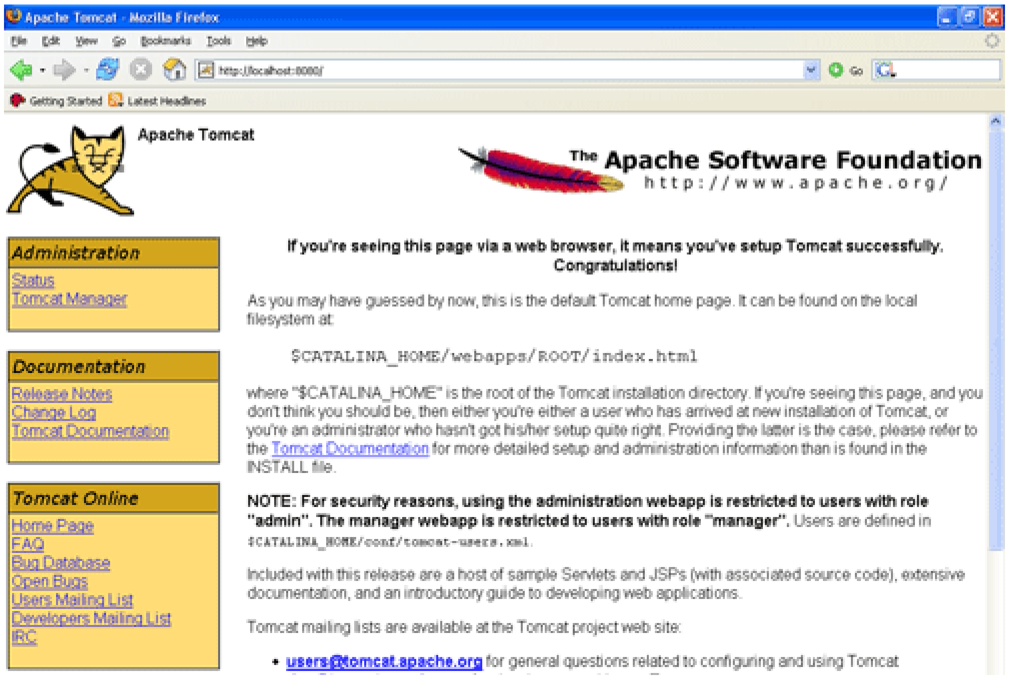
JDBC connectivity

The JDBC provides database-independent connectivity between the J2EE platform and a wide range of tabular data sources. JDBC technology allows an Application Component Provider to:

- Perform connection and authentication to a database server
- Manager transactions
- Move SQL statements to a database engine for preprocessing and execution
- Execute stored procedures
- Inspect and modify the results from Select statements.

Tomcat 6.0 web server

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Weblogic, is one of the popular application server).To develop a web application with jsp/servlet install any web server like JRun, Tomcat etc to run your application.



If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:

```
$CATALINA_HOME/webapps/ROOT/index.html
```

where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you don't think you should be, then either you're either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her setup quite right. Providing the latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration information than is found in the INSTALL file.

NOTE: For security reasons, using the administration webapp is restricted to users with role "admin". The manager webapp is restricted to users with role "manager". Users are defined in `$CATALINA_HOME/conf/tomcat-users.xml`.

Included with this release are a host of sample Servlets and JSPs (with associated source code), extensive documentation, and an introductory guide to developing web applications.

Tomcat mailing lists are available at the Tomcat project web site:

- users@tomcat.apache.org for general questions related to configuring and using Tomcat

Bibliography:

References for the Project Development were taken from the following Books and Web Sites.

Oracle

PL/SQL Programming by Scott Urman

SQL complete reference by Livion

JAVA Technologies

JAVA Complete Reference

Java Script Programming by Yehuda Shiran

Mastering JAVA Security

JAVA2 Networking by Pistoria

JAVA Security by Scotl oaks

Head First EJB Sierra Bates

J2EE Professional by Shadab siddiqui

JAVA server pages by Larne Pekowsley

JAVA Server pages by Nick Todd

HTML

HTML Black Book by Holzner

JDBC

Java Database Programming with JDBC by Patel moss.

Software Engineering by Roger Pressman.

SYSTEM MODEL AND DESIGN

SYSTEM MODEL

This section consists of the UML diagrams related to the modules developed such as Activity diagram, Sequence diagram and Use case diagram.

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

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

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

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

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.

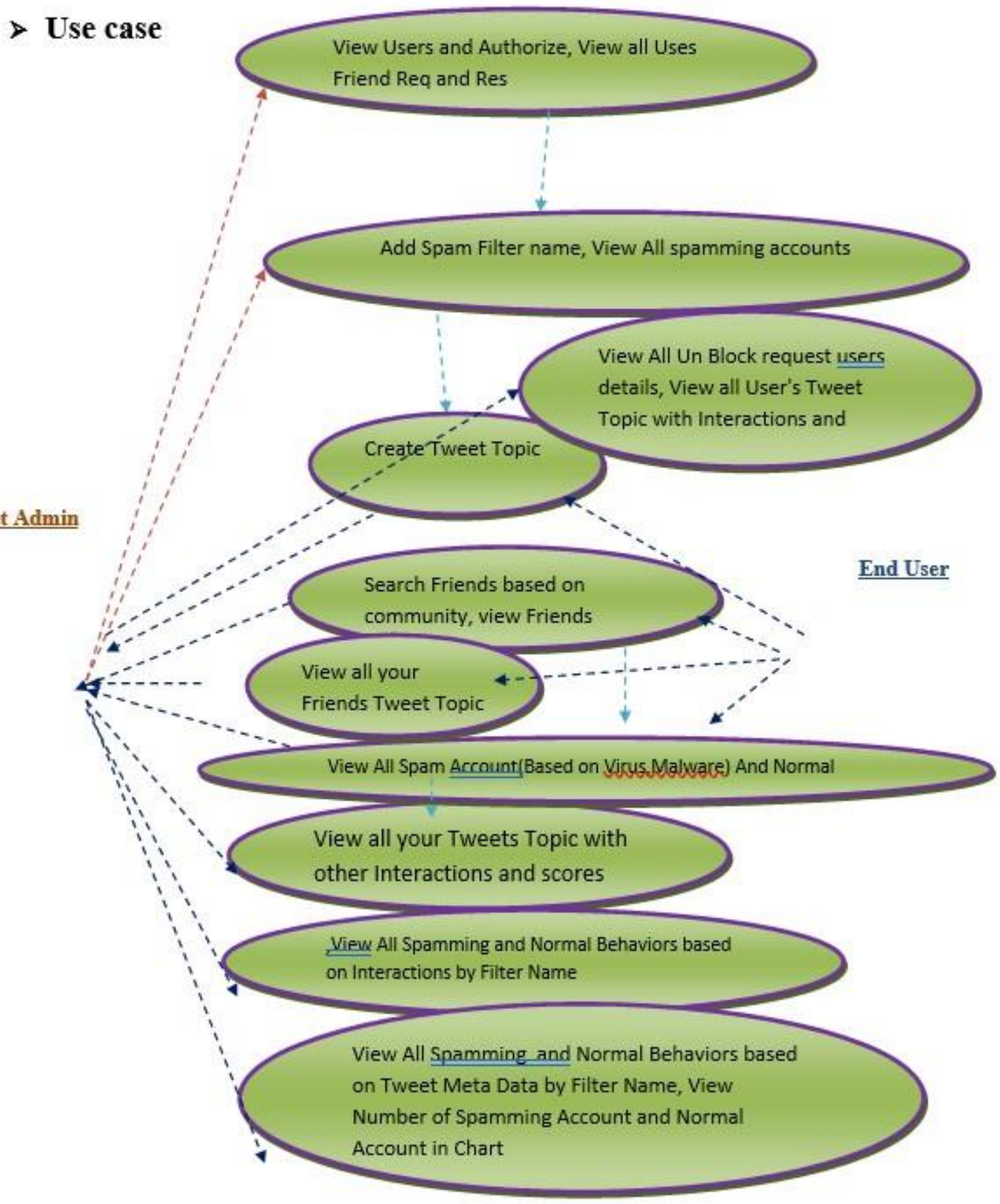
GOALS:

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

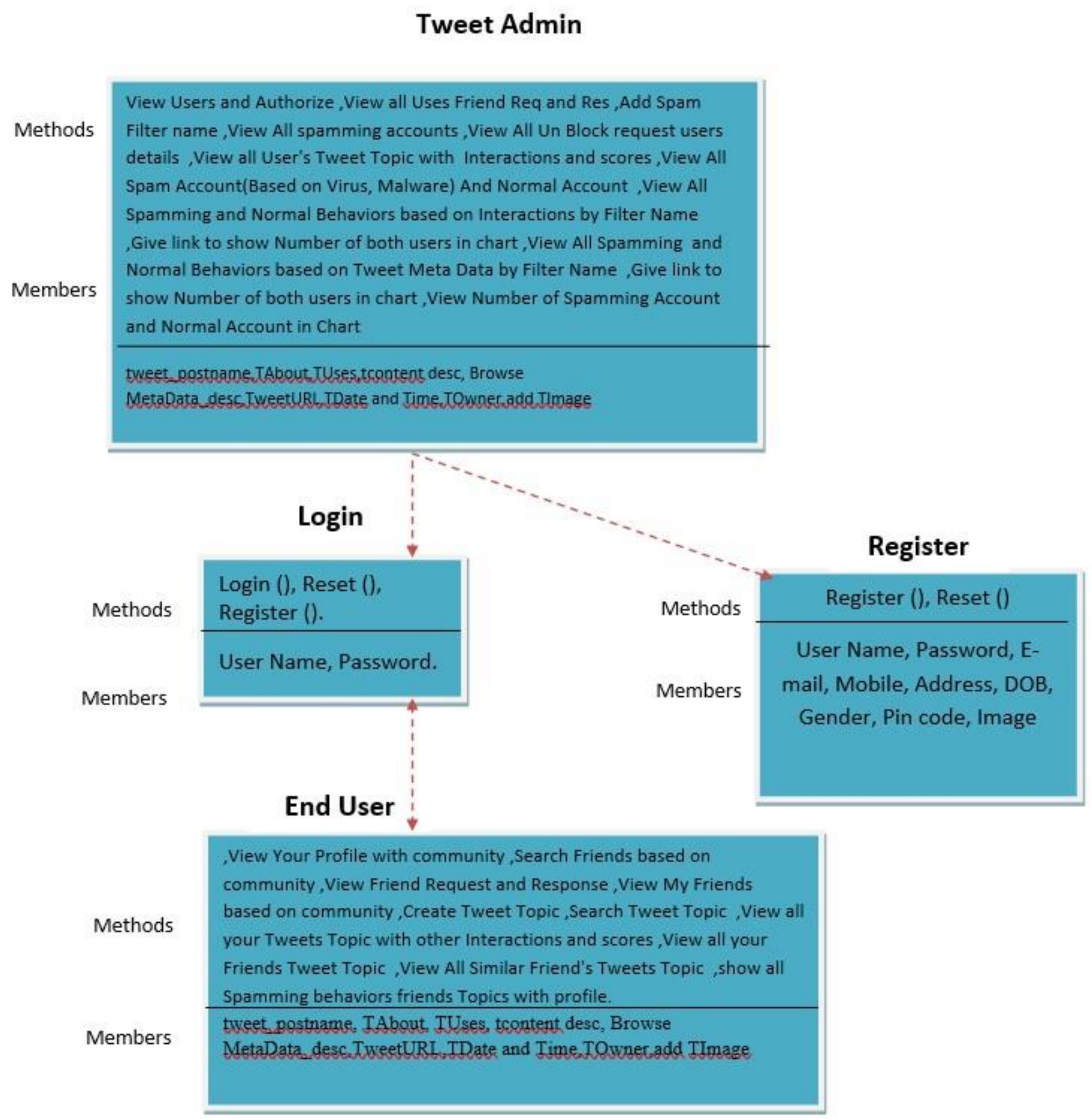
- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

DESIGN

Use case diagram for Tweet Admin and end user


Use case
View Users and Authorize, View all Uses Friend Req and Res
Add Spam Filter name, View All spamming accounts
View All Un Block request users details, View all User's Tweet Topic with Interactions and
Create Tweet Topic
Tweet Admin
End User
Search Friends based on community, view Friends
View all your Friends Tweet Topic
View All Spam Account(Based on Virus,Malware) And Normal
View all your Tweets Topic with other Interactions and scores
,View All Spamming and Normal Behaviors based on Interactions by Filter Name
View All Spamming and Normal Behaviors based on Tweet Meta Data by Filter Name, View Number of Spamming Account and Normal Account in Chart

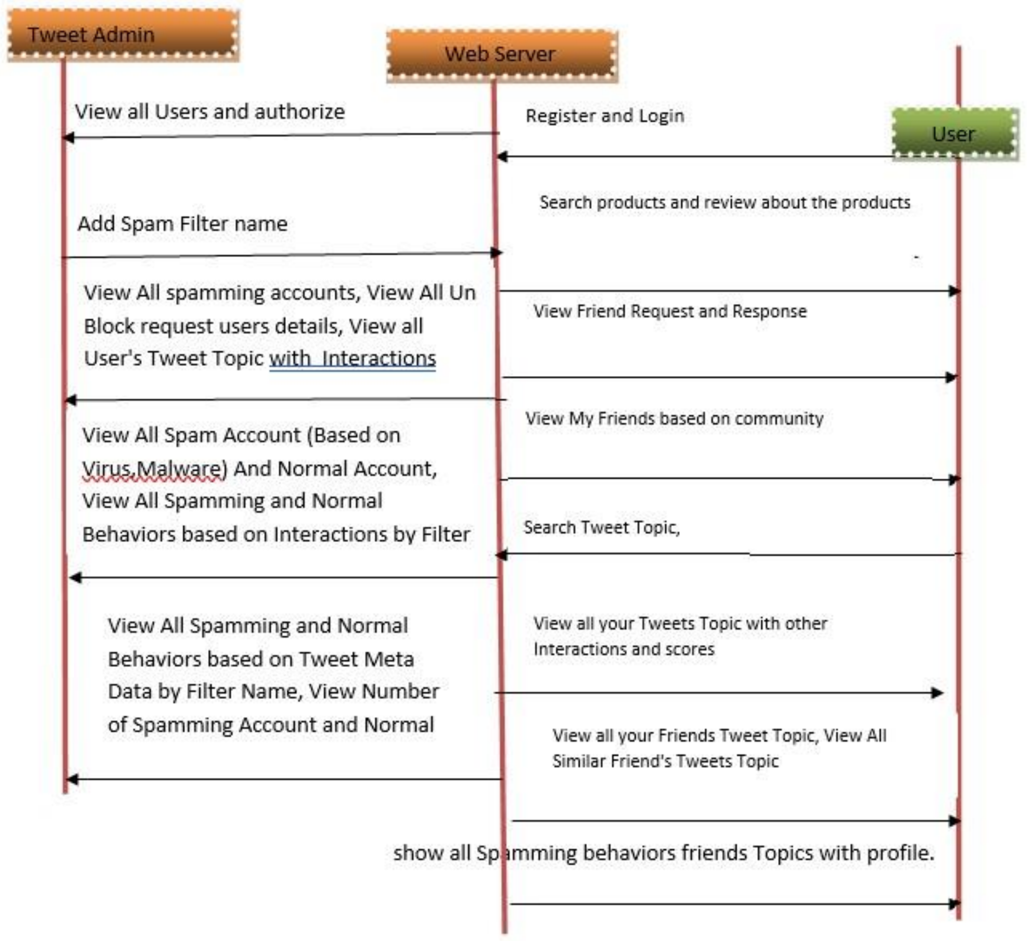
Class diagram for tweet Admin and End User



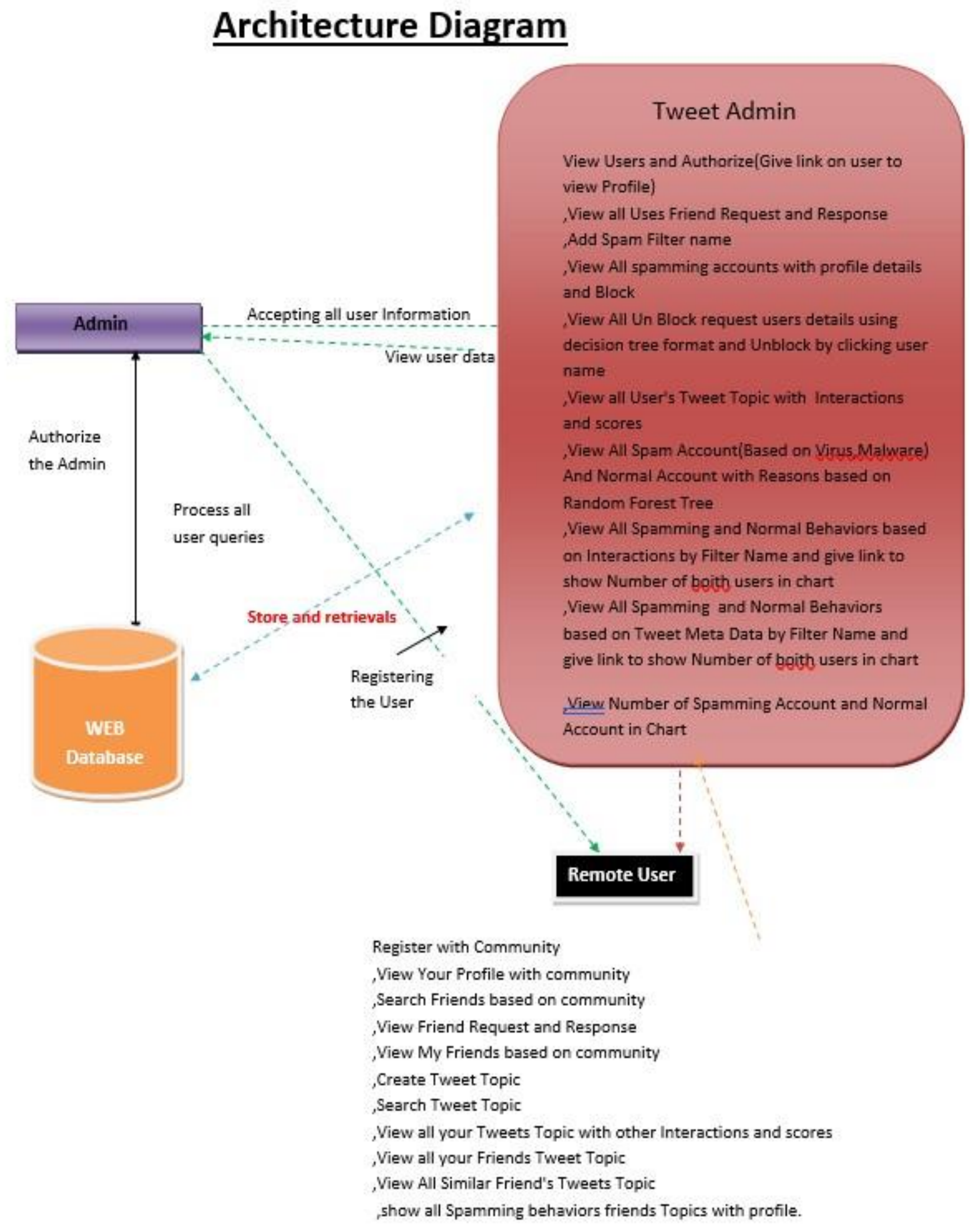
Tweet Admin
Methods
Members
View Users and Authorize ,View all Uses Friend Req and Res ,Add Spam Filter name ,View All spamming accounts ,View All Un Block request users details ,View all User's Tweet Topic with Interactions and scores ,View All Spam Account(Based on Virus, Malware) And Normal Account ,View All Spamming and Normal Behaviors based on Interactions by Filter Name ,Give link to show Number of both users in chart ,View All Spamming and Normal Behaviors based on Tweet Meta Data by Filter Name ,Give link to show Number of both users in chart ,View Number of Spamming Account and Normal Account in Chart
tweet_postname,TAbout,TUses,tcontent desc, Browse MetaData_desc,TweetURL,TDate and Time,TOwner,add TImage
Login
Methods
Members
Login (), Reset (), Register ().
User Name, Password.
Register
Methods
Members
Register (), Reset ()
User Name, Password, E-mail, Mobile, Address, DOB, Gender, Pin code, Image
End User
Methods
Members
,View Your Profile with community ,Search Friends based on community ,View Friend Request and Response ,View My Friends based on community ,Create Tweet Topic ,Search Tweet Topic ,View all your Tweets Topic with other Interactions and scores ,View all your Friends Tweet Topic ,View All Similar Friend's Tweets Topic ,show all Spamming behaviors friends Topics with profile.
tweet_postname, TAbout, TUses, tcontent desc, Browse MetaData_desc,TweetURL,TDate and Time,TOwner,add TImage

Sequence diagram for Tweet Admin and User

➢ **Sequence Diagram**

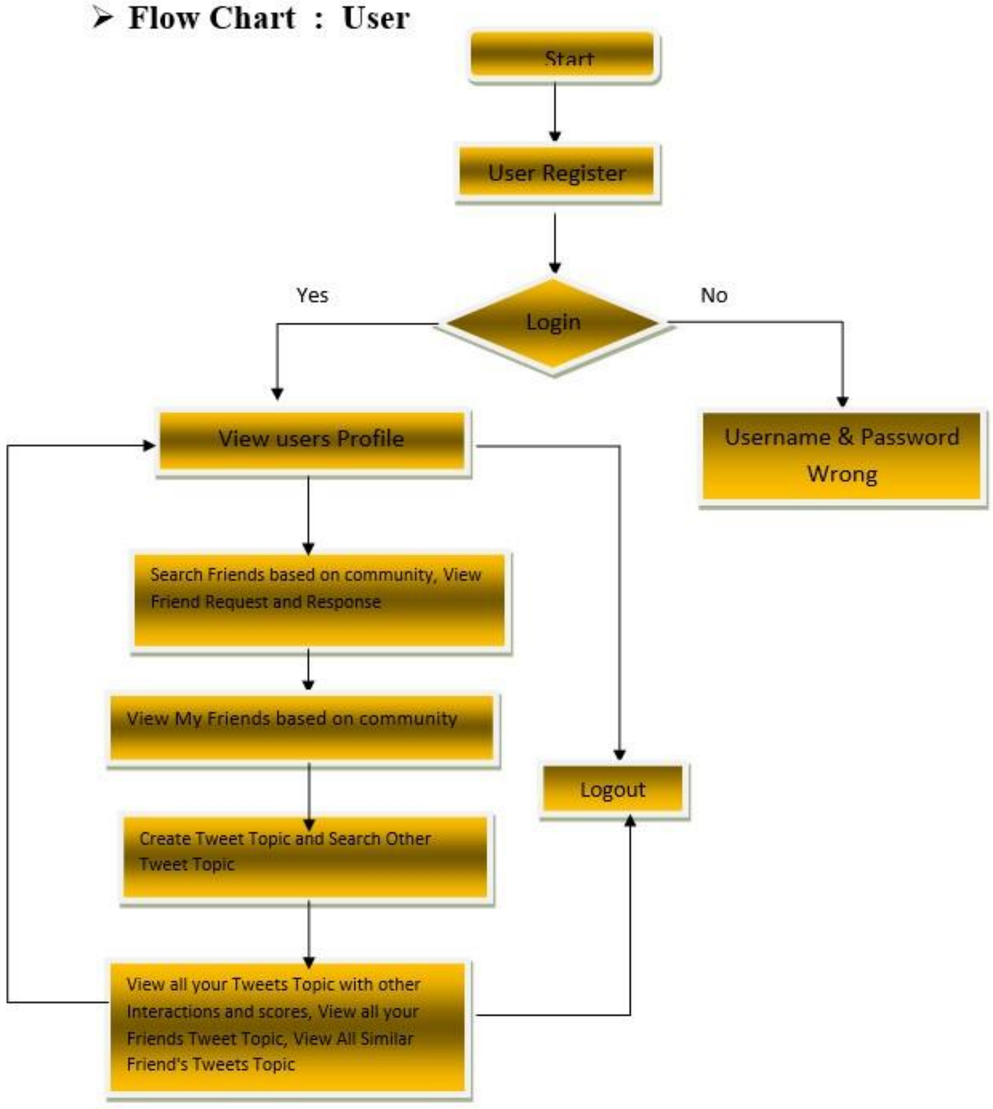

Tweet Admin
Web Server
User
View all Users and authorize
Register and Login
Search products and review about the products
Add Spam Filter name
View All spamming accounts, View All Un Block request users details, View all User's Tweet Topic with Interactions
View Friend Request and Response
View My Friends based on community
View All Spam Account (Based on Virus,Malware) And Normal Account, View All Spamming and Normal Behaviors based on Interactions by Filter
Search Tweet Topic,
View All Spamming and Normal Behaviors based on Tweet Meta Data by Filter Name, View Number of Spamming Account and Normal
View all your Tweets Topic with other Interactions and scores
View all your Friends Tweet Topic, View All Similar Friend's Tweets Topic
show all Spamming behaviors friends Topics with profile.

Architecture diagram for Tweet Admin

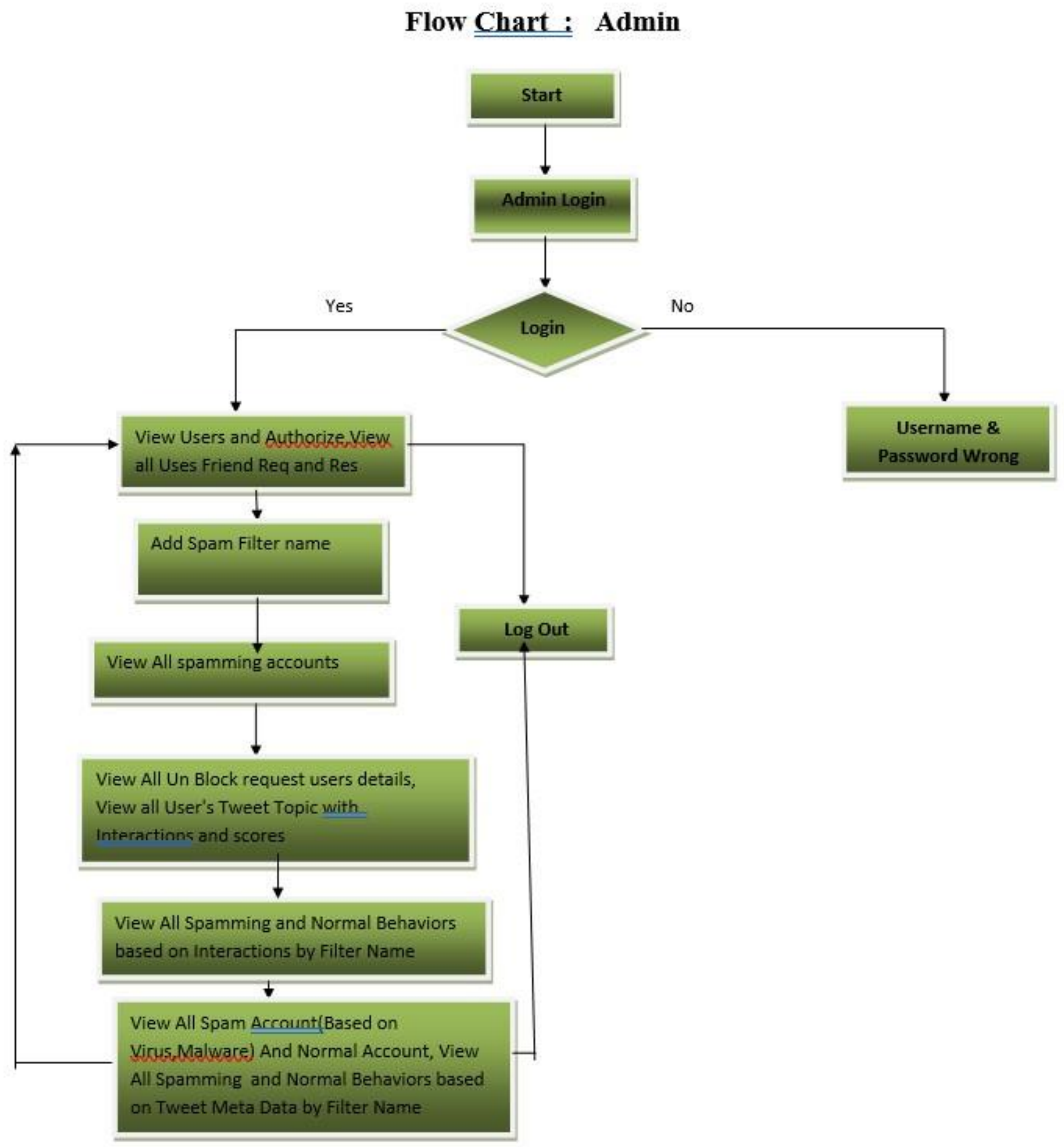

Architecture Diagram
Tweet Admin
View Users and Authorize(Give link on user to view Profile)
,View all Uses Friend Request and Response
,Add Spam Filter name
,View All spamming accounts with profile details and Block
,View All Un Block request users details using decision tree format and Unblock by clicking user name
,View all User's Tweet Topic with Interactions and scores
,View All Spam Account(Based on Virus,Malware) And Normal Account with Reasons based on Random Forest Tree
,View All Spamming and Normal Behaviors based on Interactions by Filter Name and give link to show Number of boith users in chart
,View All Spamming and Normal Behaviors based on Tweet Meta Data by Filter Name and give link to show Number of boith users in chart
,View Number of Spamming Account and Normal Account in Chart
Admin
Accepting all user Information
View user data
Authorize the Admin
Process all user queries
Store and retrievals
Registering the User
WEB Database
Remote User
Register with Community
,View Your Profile with community
,Search Friends based on community
,View Friend Request and Response
,View My Friends based on community
,Create Tweet Topic
,Search Tweet Topic
,View all your Tweets Topic with other Interactions and scores
,View all your Friends Tweet Topic
,View All Similar Friend's Tweets Topic
,show all Spamming behaviors friends Topics with profile.

Flow chart diagram for User

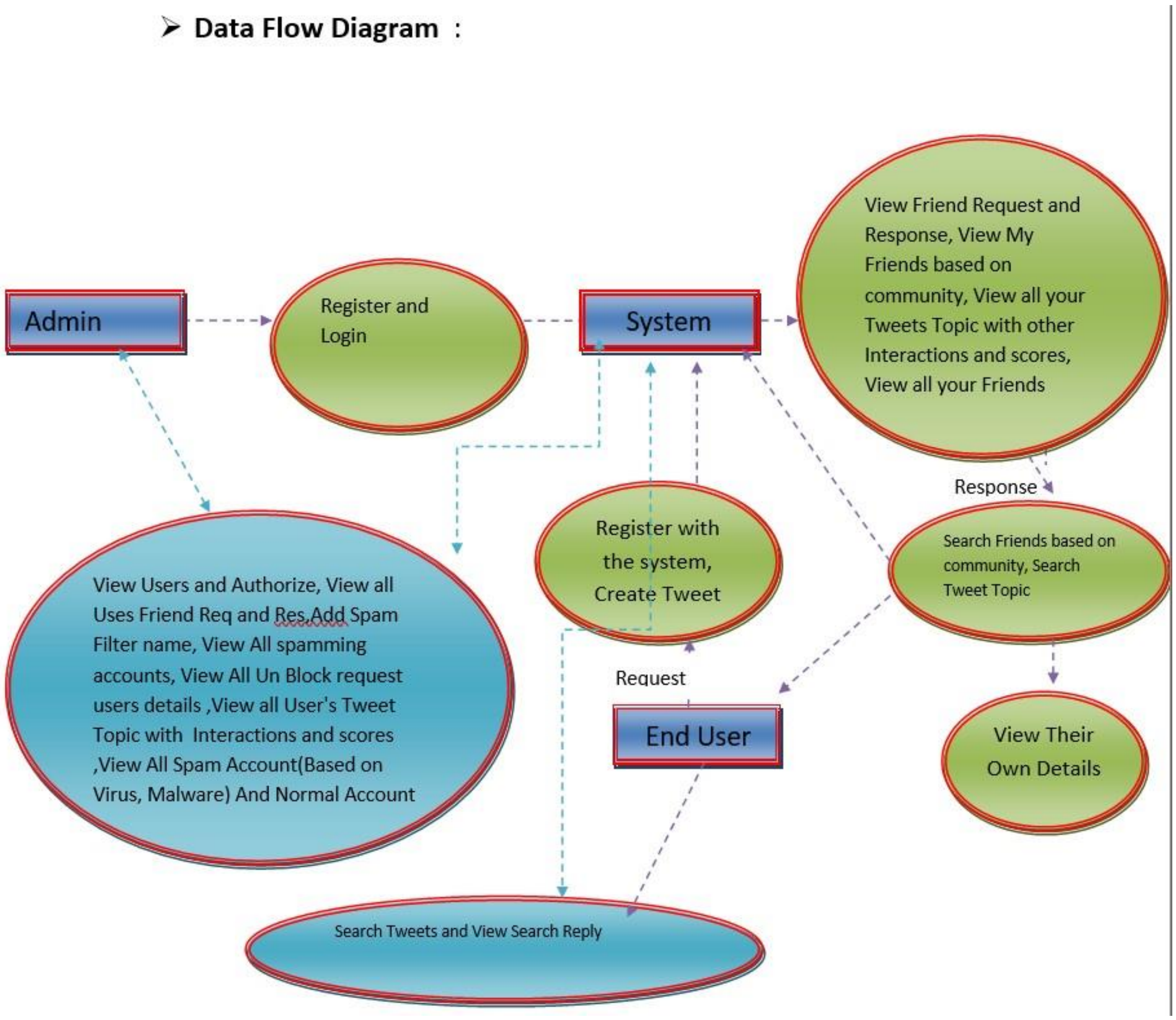
➢ **Flow Chart : User**


Start
User Register
Yes
Login
No
View users Profile
Username & Password Wrong
Search Friends based on community, View Friend Request and Response
View My Friends based on community
Logout
Create Tweet Topic and Search Other Tweet Topic
View all your Tweets Topic with other Interactions and scores, View all your Friends Tweet Topic, View All Similar Friend's Tweets Topic

Flow chart diagram for Admin


Flow Chart : Admin
Start
Admin Login
Yes
Login
No
View Users and Authorize View all Uses Friend Req and Res
Username & Password Wrong
Add Spam Filter name
Log Out
View All spamming accounts
View All Un Block request users details, View all User's Tweet Topic with Interactions and scores
View All Spamming and Normal Behaviors based on Interactions by Filter Name
View All Spam Account(Based on Virus,Malware) And Normal Account, View All Spamming and Normal Behaviors based on Tweet Meta Data by Filter Name

Data flow diagram for admin and end user


➢ Data Flow Diagram :
Admin
Register and Login
System
View Friend Request and Response, View My Friends based on community, View all your Tweets Topic with other Interactions and scores, View all your Friends
Response
Search Friends based on community, Search Tweet Topic
Register with the system, Create Tweet
View Users and Authorize, View all Uses Friend Req and Res,Add Spam Filter name, View All spamming accounts, View All Un Block request users details ,View all User's Tweet Topic with Interactions and scores ,View All Spam Account(Based on Virus, Malware) And Normal Account
Request
End User
View Their Own Details
Search Tweets and View Search Reply

IMPLEMENTATION

Modules

Tweet Admin

In this module, the Admin has to login by using valid user name and password. After login successful he can perform some operations such as View Users and Authorize(Give link on user to view Profile),View all Uses Friend Request and Response,Add Spam Filter name,View All spamming accounts with profile details and Block,View All Un Block request users details using decision tree format and Unblock by clicking user name ,View all User's Tweet Topic with Interactions and scores,View All Spam Account(Based on Virus,Malware) And Normal Account with Reasons based on Random Forest Tree,View All Spamming and Normal Behaviors based on Interactions by Filter Name and give link to show Number of boith users in chart,View All Spamming and Normal Behaviors based on Tweet Meta Data by Filter Name and give link to show Number of boith users in chart,View Number of Spamming Account and Normal Account in Chart

Friend Request & Response

In this module, the admin can view all the friend requests and responses. Here all the requests and responses will be displayed with their tags such as Id, requested user photo, requested user name, user name request to, status and time & date. If the user accepts the request then the status will be changed to accepted or else the status will remains as waiting.

User

In this module, there are n numbers of users are present. User should register before performing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user can perform some operations like View Your Profile with community, Search Friends based on community, View Friend Request and Response,View My Friends based on community, Create Tweet Topic with tweet_postname,TAbout,TUses,tcontent desc, Browse MetaData_desc,TweetURL,TDate and Time,TOwner,add TImage, Search Tweet Topic by keyword and give Your Interactions(increse score while viewing) and view URL to see web page,View all your Tweets Topic with other Interactions and scores,View all your Friends Tweet Topic with other Interactions and scores and give your Interactions,View All Similar Friend's Tweets Topic,show all Spamming behaviors friends Topics with profile.

Searching Users to make friends

In this module, the user searches for users in Same Network and in the Networks and sends friend requests to them. The user can search for users in other Networks to make friends only if they have permission.

TESTING AND RESULTS

TESTING METHODOLOGIES

The following are the Testing Methodologies:

- Unit Testing.
- Integration Testing.
- User Acceptance Testing.
- Output Testing.
- Validation Testing.

Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to
ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.
During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.
The following are the types of Integration Testing:

1.Top Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.
In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

2. Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

Validation Checking

Validation checks are performed on the following fields.

Text Field:

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field:

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data:

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

Using Artificial Test Data:

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package “Virtual Private Network” has satisfied all the requirements specified as per software requirement specification and was accepted.

User training

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

Maintenance

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user’s requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

TESTING STRATEGY :

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation .A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.
Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

System testing:

Software once validated must be combined with other system elements (e.g. Hardware, people, database). System testing verifies that all the elements are proper and that overall system function performance is achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation.

Unit testing:

In unit testing different are modules are tested against the specifications produced during the design for the modules. Unit testing is essential for verification of the code produced during the coding phase, and hence the goals to test the internal logic of the modules. Using the detailed design description as a guide, important Conrail paths are tested to uncover errors within the boundary of the modules. This testing is carried out during the programming stage itself. In this type of testing step, each module was found to be working satisfactorily as regards to the expected output from the module.

In Due Course, latest technology advancements will be taken into consideration. As part of technical build-up many components of the networking system will be generic in nature so that future projects can either use or interact with this. The future holds a lot to offer to the development and refinement of this project.

Users ---

Register with Community like Sports,Education,Cinema,Politics and Login,request to un block if it is blocked

--- View Your Profile with community

--- Search Friends based on community

--- View Friend Request and Response

--- View My Friends based on community

--- Create Tweet Topic with tweet_postname,TAbout,TUses,tcontent desc, Browse MetaData_desc,TweetURL,TDate and Time,TOwner,add TImage

-- Search Tweet Topic by keyword and give Your Interactions(increse score while viewing) and view URL to see web page

--- View all your Tweets Topic with other Interactions and scores

--- View all your Friends Tweet Topic with other Interactions and scores and give your Interactions

--- View All Similar Friend's Tweets Topic

--- show all Spamming behaviors friends Topics with profile.

Twitter Admin

--Login

--- View Users and Authorize(Give link on user to view Profile)

--- View all Uses Friend Request and Response(Give link on user to view Profile)

--- Add Spam Filter name

--- View All spamming accounts with profile details and Block

--- View All Un Block request users details using decision tree format and Unblock by clicking user name

--- View all User's Tweet Topic with Interactions and scores

--- View All Spam Account(Based on Virus,Malware) And Normal Account with Reasons based on Random Forest Tree

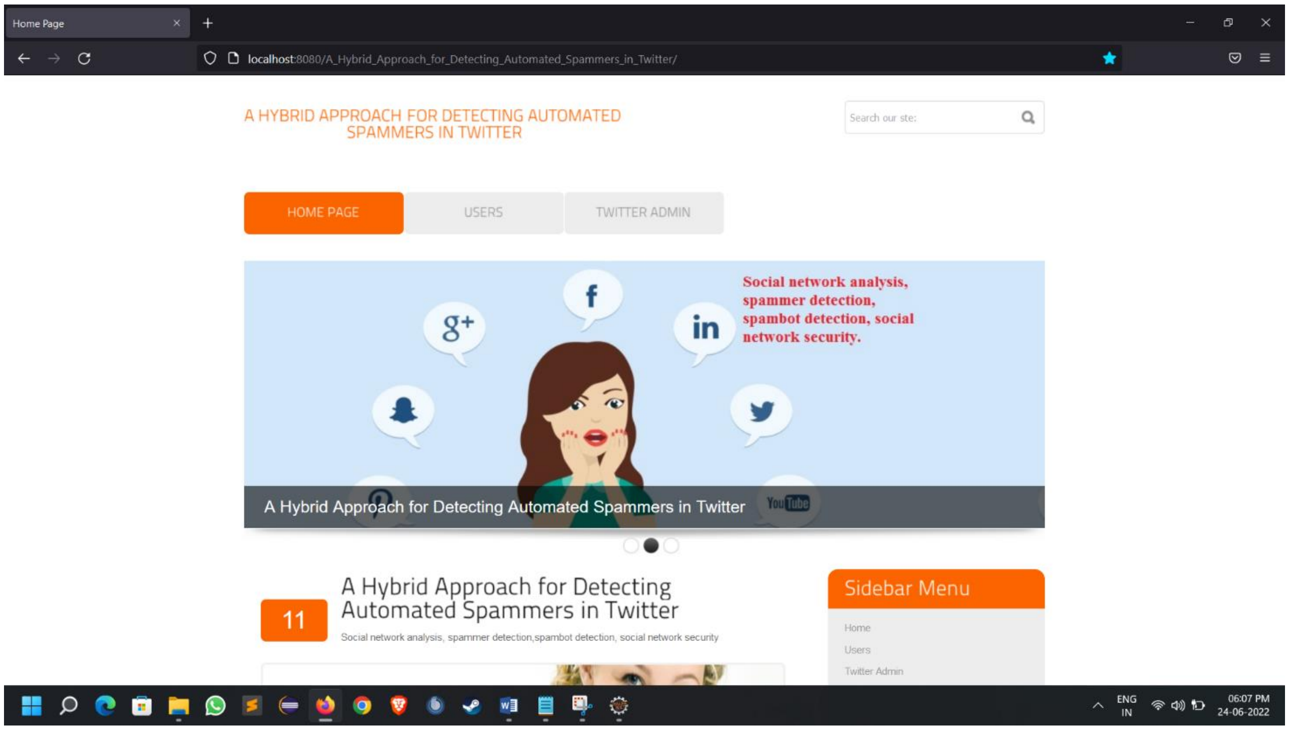
--- (Based ON Community)View All Spamming Behaviors and Normal Behaviors based on Interactions by Filter Name and give link to show Number of boith users in chart

--- (Based ON Community)View All Spamming Behaviors and Normal Behaviors based on Tweet Meta Data by Filter Name and give link to show Number of boith users in chart

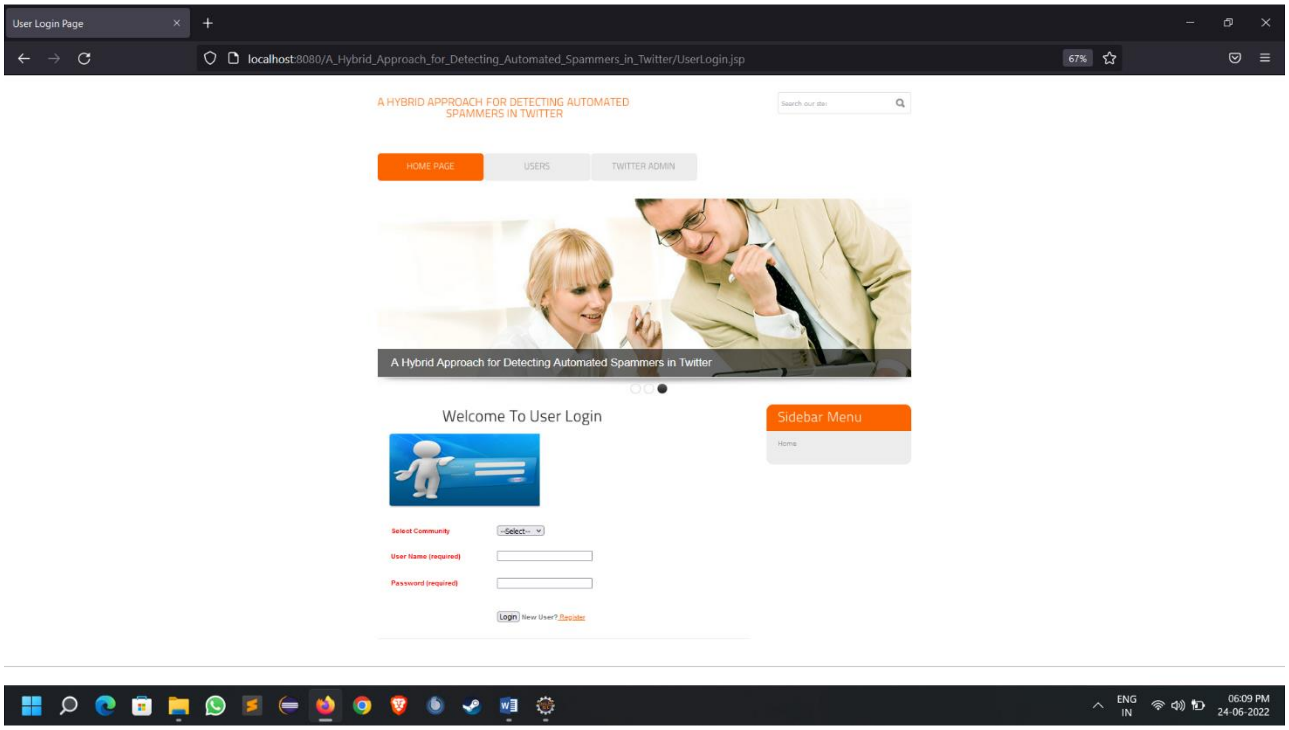
--- (Based ON Community)View Number of Spamming Account and Normal Account in Chart

Results and screenshots

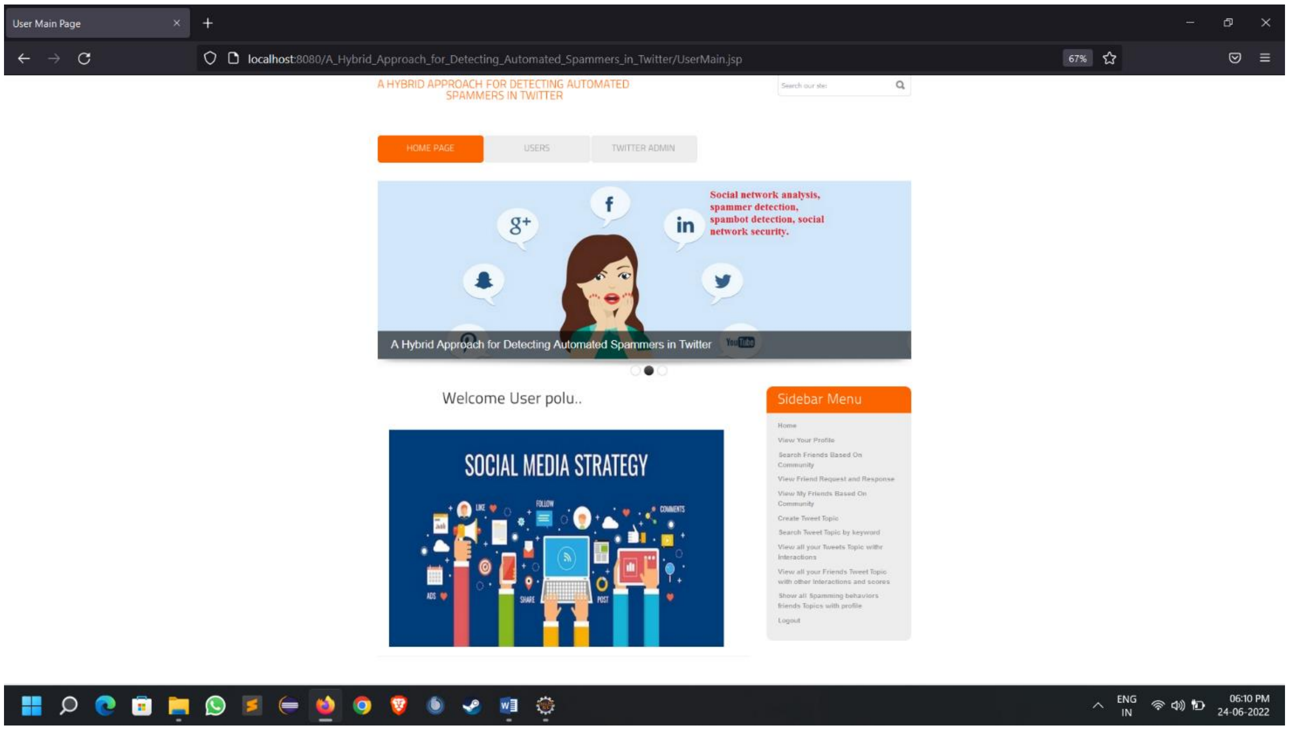
Here is the screenshot of home page:


Home Page
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/
A HYBRID APPROACH FOR DETECTING AUTOMATED SPAMMERS IN TWITTER
Search our site:
HOME PAGE
USERS
TWITTER ADMIN
Social network analysis, spammer detection, spambot detection, social network security.
A Hybrid Approach for Detecting Automated Spammers in Twitter
11
A Hybrid Approach for Detecting Automated Spammers in Twitter
Social network analysis, spammer detection,spambot detection, social network security
Sidebar Menu
Home
Users
Twitter Admin
06:07 PM 24-06-2022

Here is the screenshot of user login page:

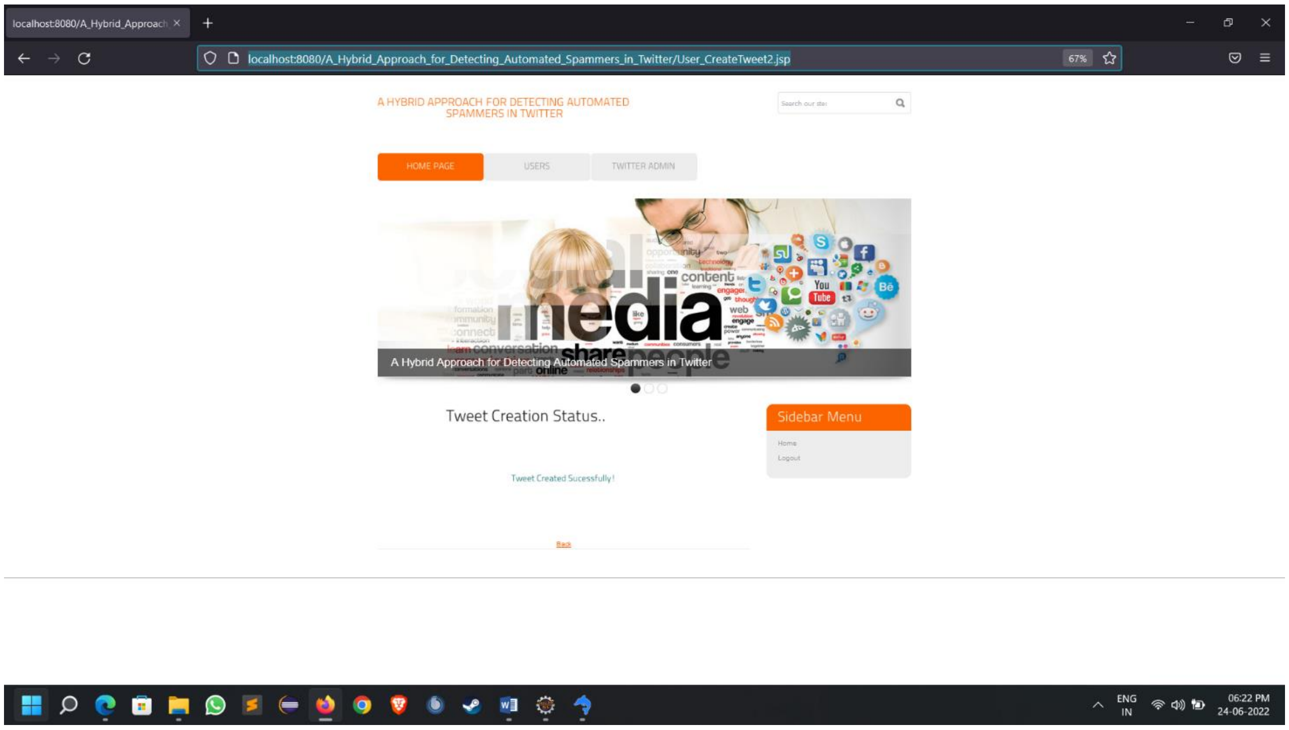

User Login Page
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/UserLogin.jsp
A HYBRID APPROACH FOR DETECTING AUTOMATED SPAMMERS IN TWITTER
HOME PAGE
USERS
TWITTER ADMIN
A Hybrid Approach for Detecting Automated Spammers in Twitter
Welcome To User Login
Sidebar Menu
Home
Select Community
--Select--
User Name (required)
Password (required)
Login
New User? Register
06:09 PM 24-06-2022

Here is the screenshot of successful user login or welcome page of user after logging in


User Main Page
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/UserMain.jsp
67%
A HYBRID APPROACH FOR DETECTING AUTOMATED SPAMMERS IN TWITTER
Search our site
HOME PAGE
USERS
TWITTER ADMIN
Social network analysis, spammer detection, spambot detection, social network security.
A Hybrid Approach for Detecting Automated Spammers in Twitter
Sidebar Menu
Home
View Your Profile
Search Friends Based On Community
View Friend Request and Response
View My Friends Based On Community
Create Tweet Topic
Search Tweet Topic by keyword
View all your Tweets Topic withr Interactions
View all your Friends Tweet Topic with other Interactions and scores
Show all Spamming behaviors friends Topics with profile
Logout
SOCIAL MEDIA STRATEGY
ENG IN
06:10 PM
24-06-2022

Welcome User polu..

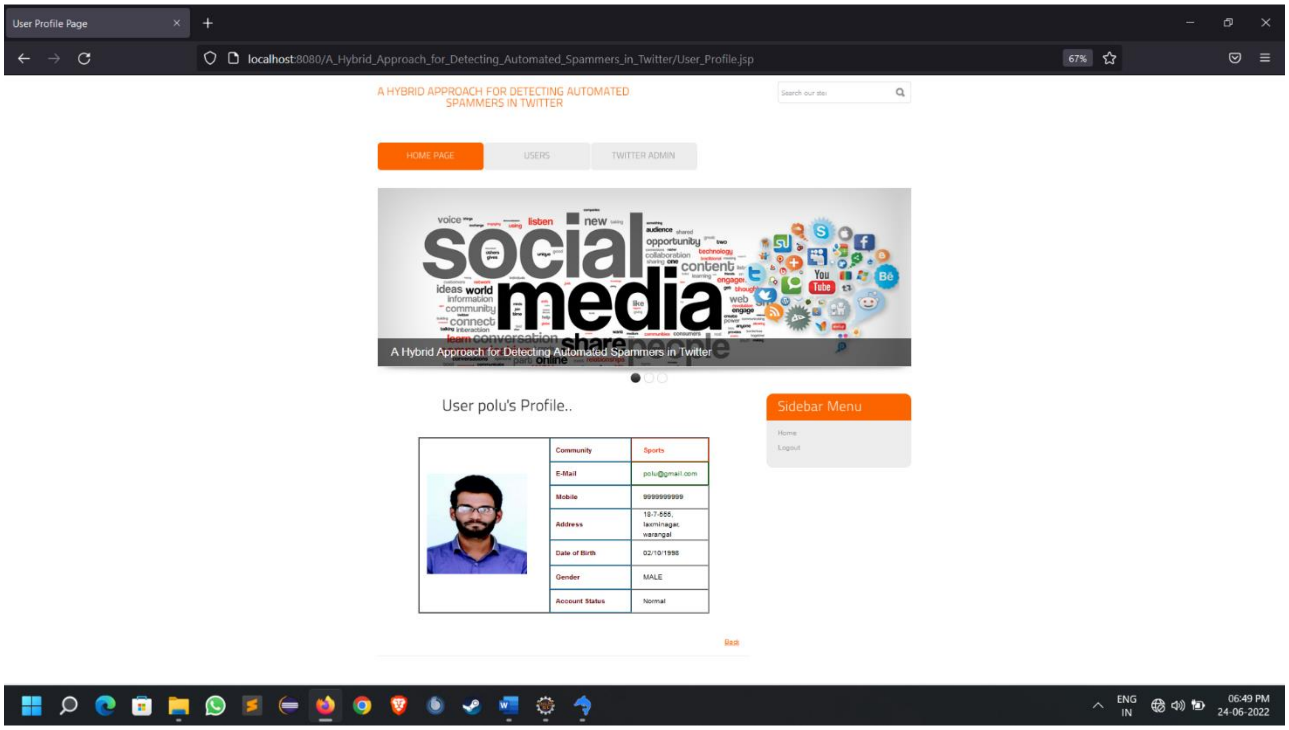
Here is the screenshot of user created a tweet successfully


localhost:8080/A_Hybrid_Approach
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/User_CreateTweet2.jsp
67%
A HYBRID APPROACH FOR DETECTING AUTOMATED SPAMMERS IN TWITTER
Search our site
HOME PAGE
USERS
TWITTER ADMIN
media
share
A Hybrid Approach for Detecting Automated Spammers in Twitter
Sidebar Menu
Home
Logout
Back
ENG IN
06:22 PM
24-06-2022

Tweet Creation Status..

Tweet Created Sucessfully!

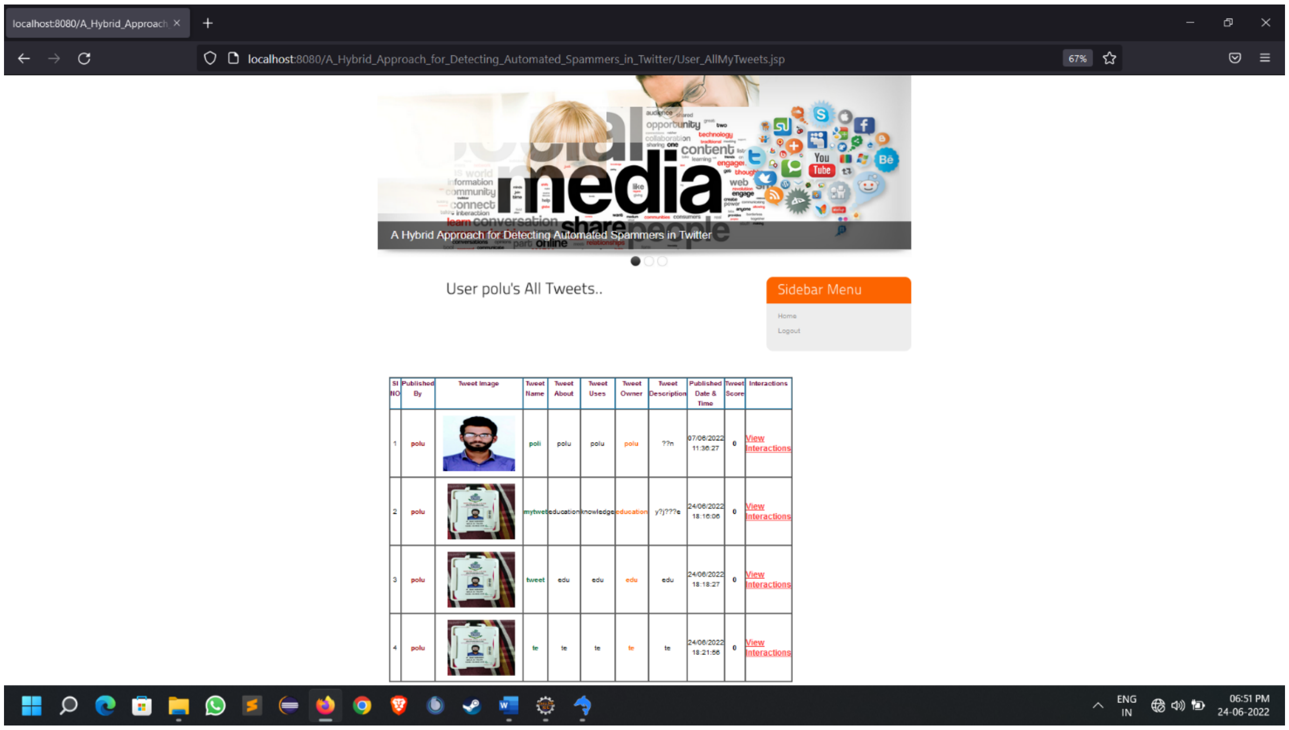
Here is the screenshot of view user profile option


User Profile Page
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/User_Profile.jsp
A HYBRID APPROACH FOR DETECTING AUTOMATED SPAMMERS IN TWITTER
HOME PAGE
USERS
TWITTER ADMIN
A Hybrid Approach for Detecting Automated Spammers in Twitter
Sidebar Menu
Home
Logout
Back
06:49 PM
24-06-2022

User polu's Profile..

Community	Sports
E-Mail	polu@gmail.com
Mobile	9999999999
Address	18-7-556, laxminagar, warangal
Date of Birth	02/10/1998
Gender	MALE
Account Status	Normal

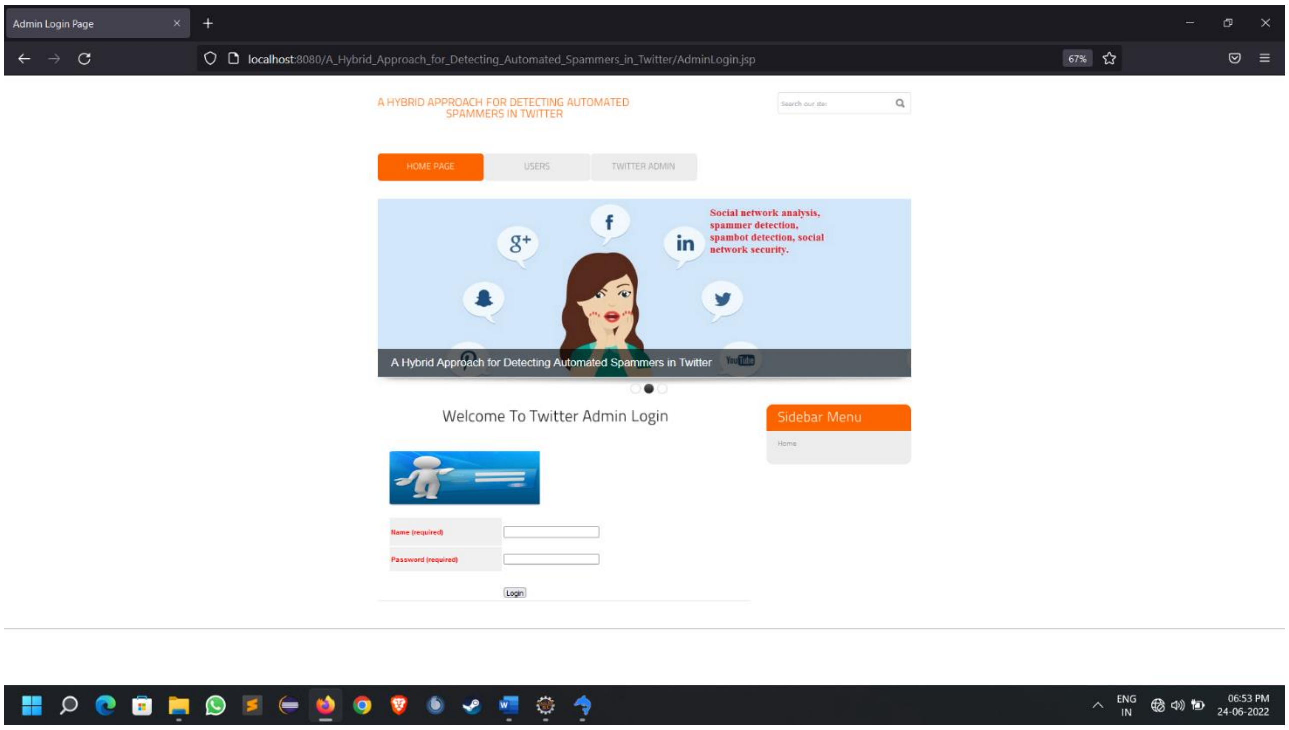
Screenshot viewing of user viewing his all own tweets


localhost:8080/A_Hybrid_Approach
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/User_AllMyTweets.jsp
A Hybrid Approach for Detecting Automated Spammers in Twitter
Sidebar Menu
Home
Logout
06:51 PM
24-06-2022

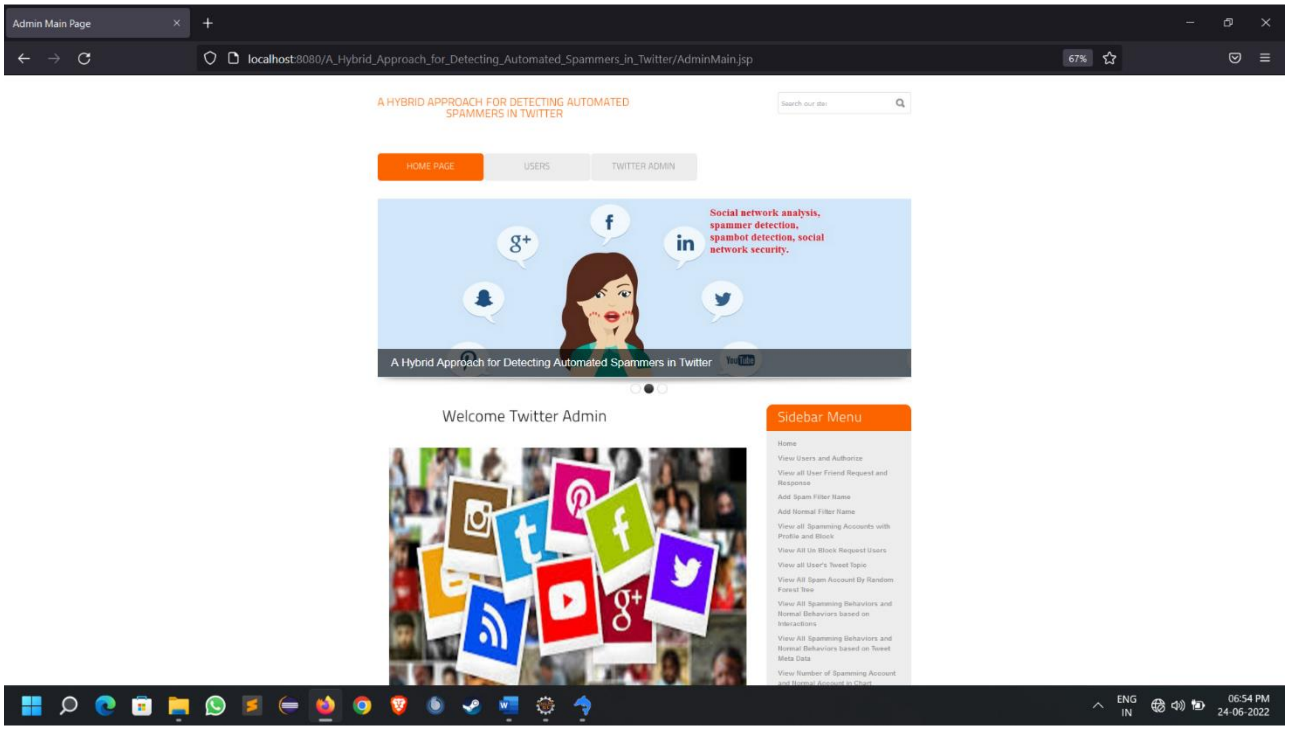
User polu's All Tweets..

Sl NO	Published By	Tweet Image	Tweet Name	Tweet About	Tweet Uses	Tweet Owner	Tweet Description	Published Date & Time	Tweet Score	Interactions
1	polu		poli	polu	polu	polu	??n	07/06/2022 11:36:27	0	View Interactions
2	polu		mytweet	education	knowledge	education	y?j???e	24/06/2022 18:16:06	0	View Interactions
3	polu		tweet	edu	edu	edu	edu	24/06/2022 18:18:27	0	View Interactions
4	polu		te	te	te	te	te	24/06/2022 18:21:56	0	View Interactions

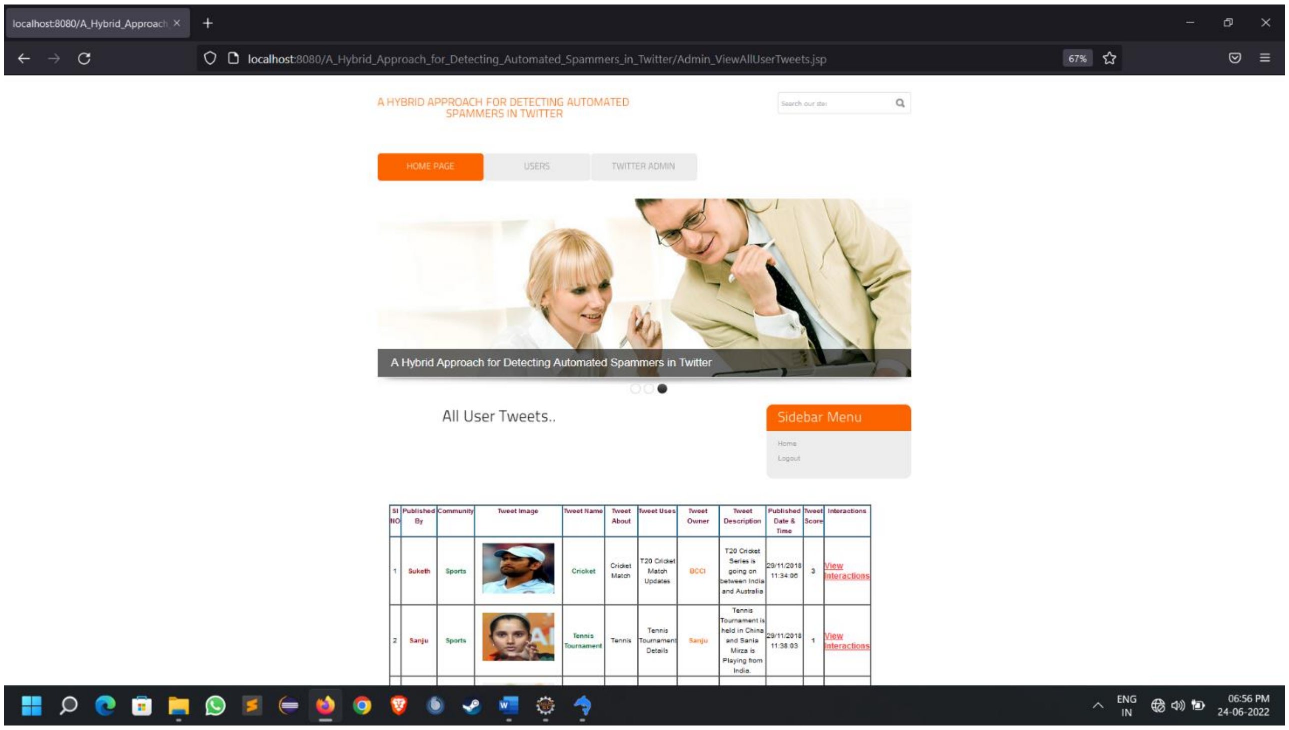
Screenshot of admin login page


Admin Login Page
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/AdminLogin.jsp
67%
A HYBRID APPROACH FOR DETECTING AUTOMATED SPAMMERS IN TWITTER
Search our site
HOME PAGE
USERS
TWITTER ADMIN
Social network analysis, spammer detection, spambot detection, social network security.
A Hybrid Approach for Detecting Automated Spammers in Twitter
Welcome To Twitter Admin Login
Sidebar Menu
Home
Name (required)
Password (required)
Login
ENG IN
06:53 PM
24-06-2022

Screenshot of welcome page of admin after successful login

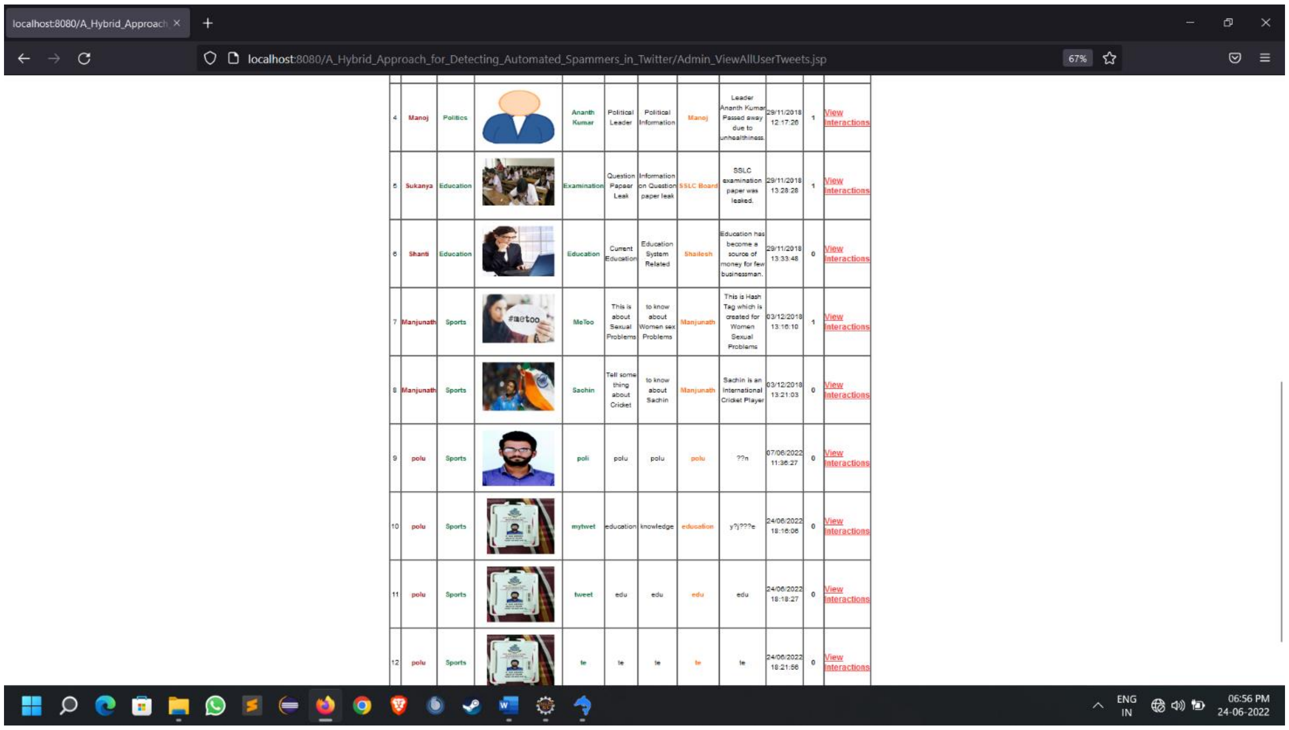

Admin Main Page
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/AdminMain.jsp
67%
A HYBRID APPROACH FOR DETECTING AUTOMATED SPAMMERS IN TWITTER
Search our site
HOME PAGE
USERS
TWITTER ADMIN
Social network analysis, spammer detection, spambot detection, social network security.
A Hybrid Approach for Detecting Automated Spammers in Twitter
Welcome Twitter Admin
Sidebar Menu
Home
View Users and Authorize
View all User Friend Request and Response
Add Spam Filter Name
Add Normal Filter Name
View all Spamming Accounts with Profile and Block
View All Un Block Request Users
View all User's Tweet Topic
View All Spam Account By Random Forest Tree
View All Spamming Behaviors and Normal Behaviors based on Interactions
View All Spamming Behaviors and Normal Behaviors based on Tweet Meta Data
View Number of Spamming Account and Normal Account in Chart
ENG IN
06:54 PM
24-06-2022

Screenshots of admin viewing all user created tweets


localhost:8080/A_Hybrid_Approach
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/Admin_ViewAllUserTweets.jsp
67%
A HYBRID APPROACH FOR DETECTING AUTOMATED SPAMMERS IN TWITTER
Search our site
HOME PAGE
USERS
TWITTER ADMIN
A Hybrid Approach for Detecting Automated Spammers in Twitter
Sidebar Menu
Home
Logout
ENG IN
06:56 PM 24-06-2022

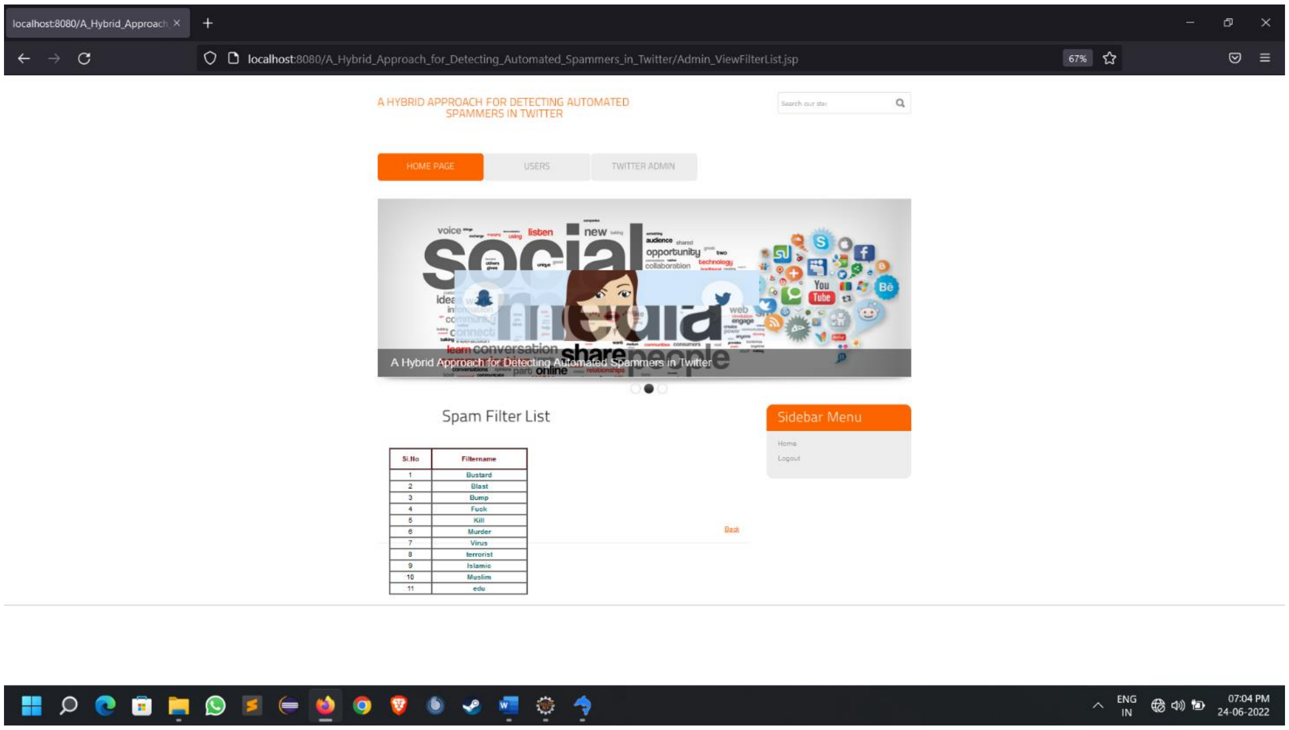
All User Tweets..

Sl NO	Published By	Community	Tweet Image	Tweet Name	Tweet About	Tweet Uses	Tweet Owner	Tweet Description	Published Date & Time	Tweet Score	Interactions
1	Suketh	Sports		Cricket	Cricket Match	T20 Cricket Match Updates	BCCI	T20 Cricket Series is going on between India and Australia	29/11/2018 11:34:06	3	View Interactions
2	Sanju	Sports		Tennis Tournament	Tennis	Tennis Tournament Details	Sanju	Tennis Tournament is held in China and Sania Mirza is Playing from India.	29/11/2018 11:38:03	1	View Interactions


localhost:8080/A_Hybrid_Approach
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/Admin_ViewAllUserTweets.jsp
67%
ENG IN
06:56 PM 24-06-2022

4	Manoj	Politics		Ananth Kumar	Political Leader	Political Information	Manoj	Leader Ananth Kumar Passed away due to unhealthiness.	29/11/2018 12:17:26	1	View Interactions
5	Sukanya	Education		Examination	Question Papaer Leak	Information on Question paper leak	SSLC Board	SSLC examination paper was leaked.	29/11/2018 13:28:28	1	View Interactions
6	Shanti	Education		Education	Current Education	Education System Related	Shailesh	Education has become a source of money for few businessman.	29/11/2018 13:33:48	0	View Interactions
7	Manjunath	Sports		MeToo	This is about Sexual Problems	to know about Women sex Problems	Manjunath	This is Hash Tag which is created for Women Sexual Problems	03/12/2018 13:16:10	1	View Interactions
8	Manjunath	Sports		Sachin	Tell some thing about Cricket	to know about Sachin	Manjunath	Sachin is an International Cricket Player	03/12/2018 13:21:03	0	View Interactions
9	polu	Sports		poli	polu	polu	polu	??n	07/06/2022 11:36:27	0	View Interactions
10	polu	Sports		mytwet	education	knowledge	education	y?j???e	24/06/2022 18:16:06	0	View Interactions
11	polu	Sports		tweet	edu	edu	edu	edu	24/06/2022 18:18:27	0	View Interactions
12	polu	Sports		te	te	te	te	te	24/06/2022 18:21:56	0	View Interactions

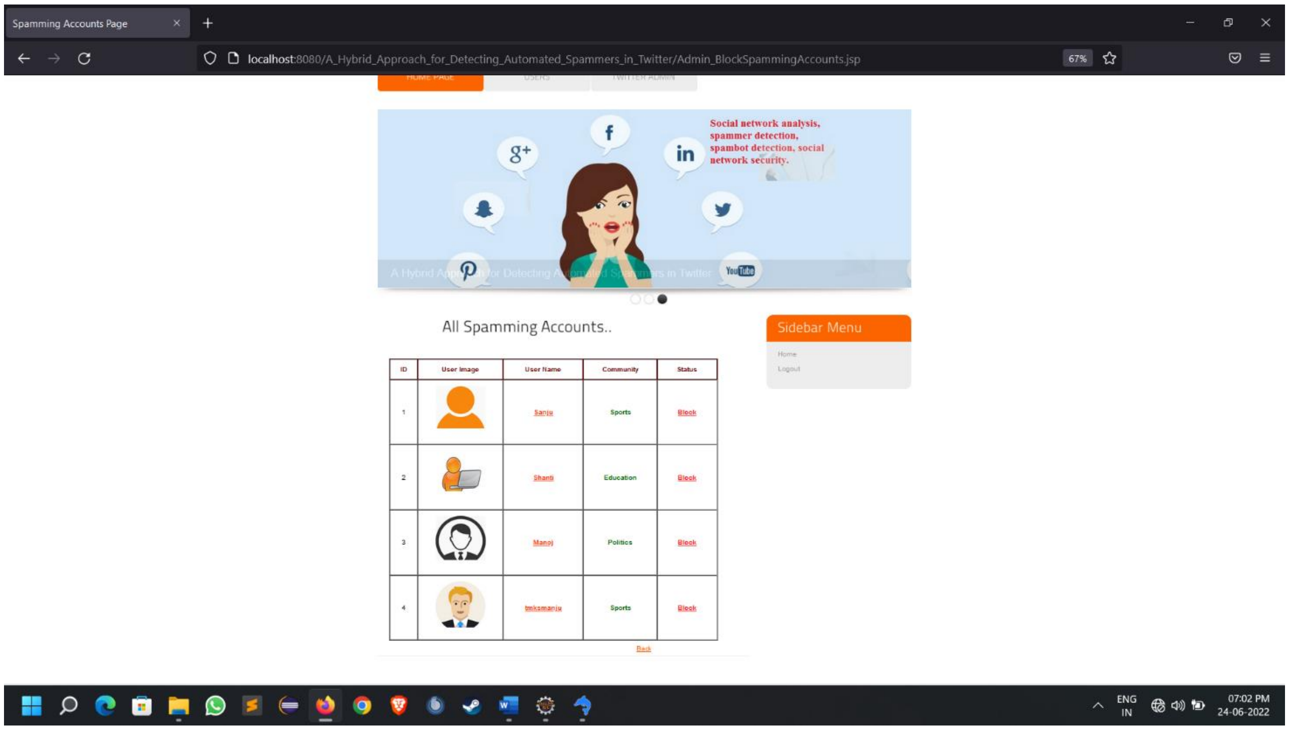
Screenshot of filters that used to detect spammers


localhost:8080/A_Hybrid_Approach
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/Admin_ViewFilterList.jsp
A HYBRID APPROACH FOR DETECTING AUTOMATED SPAMMERS IN TWITTER
Search our site
HOME PAGE
USERS
TWITTER ADMIN
A Hybrid Approach for Detecting Automated Spammers in Twitter
Sidebar Menu
Home
Logout
Back
07:04 PM
24-06-2022

Spam Filter List

Si.No	Filtername
1	Bustard
2	Blast
3	Bump
4	Fuck
5	Kill
6	Murder
7	Virus
8	terrorist
9	Islamic
10	Muslim
11	edu

Screenshot of people misusing the platform by spamming


Spamming Accounts Page
localhost:8080/A_Hybrid_Approach_for_Detecting_Automated_Spammers_in_Twitter/Admin_BlockSpammingAccounts.jsp
Social network analysis, spammer detection, spambot detection, social network security.
A Hybrid Approach for Detecting Automated Spammers in Twitter
Sidebar Menu
Home
Logout
Back
07:02 PM
24-06-2022

All Spamming Accounts..

ID	User Image	User Name	Community	Status
1		Sanju	Sports	Block
2		Shanti	Education	Block
3		Manoj	Politics	Block
4		tmksmanju	Sports	Block

CONCLUSION

In this paper, we have proposed a hybrid approach exploiting community-based features with *metadata*-, *content*-, and *interaction-based* features for detecting automated spammers in Twitter. Spammers are generally planted in OSNs for varied purposes, but absence of real-life identity hinders them to join the trust network of benign users. Therefore, spammers randomly follow a number of users, but rarely followed back by them, which results in low edge density among their *followers* and *followings*. This type of spammers interaction pattern can be exploited for the development of effective spammers detection systems. Unlike existing approaches of characterizing spammers based on their own profiles, the novelty of the proposed approach lies in the characterization of a spammer based on its neighboring nodes (especially, the followers) and their interaction network. This is mainly due to the fact that users can evade features that are related to their own activities,but it is difficult to evade those that are based on their followers. On analysis, metadata-based features are found to be least effective as they can be easily evaded by the sophisticated spammers by using random number generator algorithms. On the other hand, both interaction- and community-based features are found to be the most discriminative for spammers detection.

Attaining perfect accuracy in spammers detection is extremely difficult, and accordingly any feature set can never be considered as complete and sound, as spammers keep on changing their operating behavior to evade detection mechanism.

Therefore, in addition to profile-based characterization,complete logs of spammers starting from their entry in the network to their detection, need to be analyzed to model the evolutionary behavior and phases of the life-cycles of spammers. But, generally spammers are detected when they are at very advanced stage, and it is difficult to get their past logs data. Moreover, it may happen that a user is operative in the network as a benign user, and later on, it start sillicit activities due to whatsoever reasons, and considered as spammer. In this circumstance, even analyzing log data may lead to wrong characterization.

Analysis of spammers network to unearth different types of coordinated spam campaigns run by the spambots seems one of the promising future directions of research. Moreover, analyzing the temporal evolution of spammers' followers may reveal some interesting patterns that can be utilized for spammers characterization at different levels of granularity.

Future scope

It is not possible to develop a system that makes all the requirements of the user. User requirements keep changing as the system is being used. Some of the future enhancements that can be done to this system are:

- As the technology emerges, it is possible to upgrade the system and can be adaptable to desired environment.
- Because it is based on object-oriented design, any further changes can be easily adaptable.
- Based on the future security issues, security can be improved using emerging technologies.
- Attendance module can be added
- sub admin module can be added
- An in-built web browser can be added

Sample code:

Here are some sample codes that are used in the program.

Here is the sample code for admin user profile:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>User Profile Page</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style4 {color: #660000}
.style5 {
color: #fc6400;
font-size: 16px;
}
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html"><small>A Hybrid Approach for Detecting Automated <br />
Spammers in Twitter</small></a></h1>
```

```
</div>
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
<input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80"
value="Search our ste:" type="text" />
</span>
<input name="button_search" src="images/search.gif" class="button_search" type="image" />
</form>
</div>
<div class="clr"></div>
<div class="menu_nav">
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="UserLogin.jsp"><span>Users</span></a></li>
<li><a href="AdminLogin.jsp"><span>Twitter Admin</span></a></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="960" height="320" alt=""
/><span><big>A Hybrid Approach for Detecting Automated Spammers in Twitter</big></span></a>
<a href="#"><img src="images/slide2.jpg" width="960" height="320" alt="" /><span><big>A Hybrid
Approach for Detecting Automated Spammers in Twitter</big></span></a> <a href="#"><img
src="images/slide3.jpg" width="960" height="320" alt="" /><span><big>A Hybrid Approach for
Detecting Automated Spammers in Twitter</big></span></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
```

```
<div class="article">
<h2><span>User <%=request.getParameter("uname")%>'s Profile..</span></h2>
<div class="clr"></div>
<p>&nbsp;</p>
<table width="523" border="1.5" align="center"  cellpadding="0" cellspacing="0"  >
<%@ include file="connect.jsp" %>
<%
String user=request.getParameter("uname");
String type=request.getParameter("type");
String id=request.getParameter("id");
String tname=request.getParameter("tname");
String up_Name=request.getParameter("up_Name");
String sc=request.getParameter("score");
String community=request.getParameter("community");
String s1,s2,s3,s4,s5,s6,s7,s8;
int i=0;
try
{
String query="select * from user where username='"+user+"'";
Statement st=connection.createStatement();
ResultSet rs=st.executeQuery(query);
if ( rs.next() )
{
        i=rs.getInt(1);
        s1=rs.getString(9);
        s2=rs.getString(4);
        s3=rs.getString(5);
s4=rs.getString(6);
s5=rs.getString(7);
s6=rs.getString(8);
        %>
<tr>
<td width="226" rowspan="8" ><div class="style7 style26" style="margin:10px 13px 10px 13px;"
><strong><a class="#" id="img1" href="#" >
```

```
<input  name="image" type="image" src="user_Pic.jsp?id=<%=i%>" style="width:180px;
height:180px;" />
</a></strong></div></td>
</tr>
<tr>
<td  width="141" height="81" align="left" valign="middle" style="color: #2c83b0;"><div align="left"
class="style15 style42 style60 style61 style7 style8 style4" style="margin-
left:10px;"><strong>Community</strong></div></td>
<td  width="133" align="left" valign="middle" height="40" style="color: #FF3300;"><div align="left"
class="style9 style10 style22 style38" style="margin-left:20px;">
<strong><%out.println(s1);%></strong>
</div></td>
</tr>
<tr>
<td  width="141" height="37" valign="middle" style="color: #2c83b0;"><div align="left" class="style15
style42 style60 style61 style7 style8 style4" style="margin-left:10px;"><strong>E-
Mail</strong></div></td>
<td  width="133" valign="middle" height="40" style="color:#006600;"><div align="left" class="style9
style10 style22 style38" style="margin-left:20px;">
<%out.println(s2);%>
</div></td>
</tr>
<tr>
<td  width="141" height="40" valign="middle" style="color: #2c83b0;"><div align="left" class="style15
style42 style60 style61 style7 style8 style4" style="margin-
left:10px;"><strong>Mobile</strong></div></td>
<td  width="133" valign="middle" height="40" style="color: #000000;"><div align="left" class="style9
style10 style22 style38" style="margin-left:20px;">
<%out.println(s3);%>
</div></td>
</tr>
<tr>
```

```
<td height="43" align="left" valign="middle" style="color: #2c83b0;"><div align="left" class="style15
style42 style60 style61 style7 style8 style4" style="margin-
left:10px;"><strong>Address</strong></div></td>
<td  width="133" align="left" valign="middle" height="40" style="color: #000000;"><div align="left"
class="style9 style10 style22 style38" style="margin-left:20px;">
<%out.println(s4);%>
</div></td>
</tr>
<tr>
<td  width="141" height="43" align="left" valign="middle" style="color: #2c83b0;"><div align="left"
class="style15 style42 style60 style61 style7 style8 style4" style="margin-left:10px;"><strong>Date
of Birth</strong></div></td>
<td  width="133" align="left" valign="middle" height="40" style="color: #000000;"><div align="left"
class="style9 style10 style22 style38" style="margin-left:20px;">
<%out.println(s5);%>
</div></td>
</tr>
<tr>
<td  width="141" height="81" align="left" valign="middle" style="color: #2c83b0;"><div align="left"
class="style15 style42 style60 style61 style7 style8 style4" style="margin-
left:10px;"><strong>Gender</strong></div></td>
<td  width="133" align="left" valign="middle" height="40" style="color: #000000;"><div align="left"
class="style9 style10 style22 style38" style="margin-left:20px;">
<%out.println(s6);%>
</div></td>
</tr>
<%
}
connection.close();
}
catch(Exception e)
{
out.println(e);
}
```

```
%>
</table>
<h2 align="right">&nbsp;</h2>
<%
if(type.equalsIgnoreCase("Admin_Authorize")){%>
<h2 align="right"><a href="Admin_AuthorizeUsers.jsp" class="style5">Back</a></h2>
<%}
else if(type.equalsIgnoreCase("Admin_FriendRequests")){%>
<h2 align="right"><a href="Admin_ViewFriendRequests.jsp" class="style5">Back</a></h2>
<%}
else if(type.equalsIgnoreCase("Admin_BlockSpamAccount")){%>
<h2 align="right"><a href="Admin_BlockSpammingAccounts.jsp" class="style5">Back</a></h2>
<%}
else if(type.equalsIgnoreCase("Admin_UnBlockReq")){%>
<h2 align="right"><a href="Admin_ViewUnblockReq.jsp" class="style5">Back</a></h2>
<%}
else if(type.equalsIgnoreCase("Admin_Tweet")){%>
<h2 align="right"><a
href="Admin_ViewTweetInteractions.jsp?type=<%=type%>&up_Name=<%=up_Name%>&tname=<
%=tname%>&id=<%=id%>&score=<%=sc%>" class="style5">Back</a></h2>
<%}
else if(type.equalsIgnoreCase("Admin_ViewAccount")){%>
<h2 align="right"><a href="Admin_ViewSpam_NormalAccounts.jsp?community=<%=community%>"
class="style5">Back</a></h2>
<%}
%>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star"><span>Sidebar</span> Menu</h2>
<div class="clr"></div>
<ul class="sb_menu">
<li><a href="AdminMain.jsp">Home</a></li>
```

```
<li><a href="AdminLogin.jsp">Logout</a></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer"></div>
</div>
</body>
</html>
```

Here is the sample code for adding spam filter by admin:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Add Spam Filter Page</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style2 {color: #006666}
.style3 {color: #006600}
.style4 {font-size: 14px}
-->
</style>
<script language="javascript" type="text/javascript">      <!--Start Reg Validation Jai Siddalinga-->
```

```
function valid()
{

var na1=document.s.filtername.value;
if(na1=="")

{
alert("Please Enter Filter Name");
document.s.filtername.focus();
return false;
}
else
{
}
}
</script>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html"><small>A Hybrid Approach for Detecting Automated <br />
Spammers in Twitter</small></a></h1>
</div>
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
<input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80"
value="Search our ste:" type="text" />
</span>
<input name="button_search" src="images/search.gif" class="button_search" type="image" />
</form>
</div>
```

```
<div class="clr"></div>
<div class="menu_nav">
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="UserLogin.jsp"><span>Users</span></a></li>
<li><a href="AdminLogin.jsp"><span>Twitter Admin</span></a></li>

</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="960" height="320" alt=""
/><span><big>A Hybrid Approach for Detecting Automated Spammers in Twitter</big></span></a>
<a href="#"><img src="images/slide2.jpg" width="960" height="320" alt="" /><span><big>A Hybrid
Approach for Detecting Automated Spammers in Twitter</big></span></a> <a href="#"><img
src="images/slide3.jpg" width="960" height="320" alt="" /><span><big>A Hybrid Approach for
Detecting Automated Spammers in Twitter</big></span></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Add Spam Filter Name</span></h2>
<div class="clr"></div>
<p>&nbsp;</p>
<form  name="s" action="Admin_AddSpamFilter1.jsp" method="post" enctype="multipart/form-data"
onsubmit="return valid()">
<table width="410" border="0" align="center"  cellpadding="0" cellspacing="0" >
<tr>
```

```
<td  width="136" valign="middle" height="55" style="color: #2c83b0;"><div align="left" class="style5
style2 style4" style="margin-left:20px;"><b>Add Spam Filter</b></div></td>
<td  width="274" valign="middle" height="55" style="color:#000000;"><label>
<input type="text" name="filtername" />
<a href="Admin_ViewFilterList.jsp" class="style6 style3"><strong>View Filter List</strong></a>
</label></td>
</tr>
<div >
<tr>
<td height="30" colspan="2" id="learn_more" align="center"  style="color:#FFFFFF;"><input
name="submit" type="submit" style="width:100px; height:25px; background-color:#CC6600;
color:#FFFFFF;" value="Add"/></td>
</tr>
</div>
</table>
</form>
<p align="right"><a href="AdminMain.jsp">Back</a></p>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star"><span>Sidebar</span> Menu</h2>
<div class="clr"></div>
<ul class="sb_menu">
<li><a href="AdminMain.jsp">Home</a></li>
<li><a href="AdminLogin.jsp">Logout</a></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer"></div>
```

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

Here is the sample code for admin login page

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Admin Login Page</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style2 {color: #FF0000}
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html"><small>A Hybrid Approach for Detecting Automated <br />
Spammers in Twitter</small></a></h1>
</div>
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
```

```
<input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80"
value="Search our ste:" type="text" />
</span>
<input name="button_search" src="images/search.gif" class="button_search" type="image" />
</form>
</div>
<div class="clr"></div>
<div class="menu_nav">
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="UserLogin.jsp"><span>Users</span></a></li>
<li><a href="AdminLogin.jsp"><span>Twitter Admin</span></a></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="960" height="320" alt=""
/><span><big>A Hybrid Approach for Detecting Automated Spammers in Twitter</big></span></a>
<a href="#"><img src="images/slide2.jpg" width="960" height="320" alt="" /><span><big>A Hybrid
Approach for Detecting Automated Spammers in Twitter</big></span></a> <a href="#"><img
src="images/slide3.jpg" width="960" height="320" alt="" /><span><big>A Hybrid Approach for
Detecting Automated Spammers in Twitter</big></span></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Welcome To Twitter Admin Login</span></h2>
<div class="clr"></div>
<p>&nbsp;</p>
```

```
<p><img src="images/Login.png" width="271" height="108" /></p>
<form id="form1" name="form1" method="post" action="AdminAuthentication.jsp">
<table width="464" border="0" cellspacing="2" cellpadding="2">
<tr>
<td width="197" height="46" align="justify" bgcolor="#F0F0F0"><span class="style34
style2"><strong>
<label for="name">Name (required)</label>
</strong></span> </td>
<td width="253"><input id="name" name="userid" class="text" /></td>
</tr>
<tr>
<td height="40" align="justify" bgcolor="#F0F0F0"><span class="style2 style34"><strong>Password
(required)</strong></span></td>
<td><input type="password" id="pass" name="pass" class="text" /></td>
</tr>
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
<tr>
<td>&nbsp;</td>
<td>
<input name="imageField" type="submit"  class="LOGIN" id="imageField" value="Login" /></td>
</tr>
</table>
</form>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star"><span>Sidebar</span> Menu</h2>
<div class="clr"></div>
<ul class="sb_menu">
<li><a href="index.html">Home</a></li>
```

```
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer"></div>
</div>
</body>
</html>
```

Here is the code for admin main page after logging in:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Admin Main Page</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style1 {color: #FF0000}
-->
</style>
</head>
<body>
<div class="main">
```

```
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html"><small>A Hybrid Approach for Detecting Automated <br />
Spammers in Twitter</small></a></h1>
</div>
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
<input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80"
value="Search our ste:" type="text" />
</span>
<input name="button_search" src="images/search.gif" class="button_search" type="image" />
</form>
</div>
<div class="clr"></div>
<div class="menu_nav">
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="UserLogin.jsp"><span>Users</span></a></li>
<li><a href="AdminLogin.jsp"><span>Twitter Admin</span></a></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="960" height="320" alt=""
/><span><big>A Hybrid Approach for Detecting Automated Spammers in Twitter</big></span></a>
<a href="#"><img src="images/slide2.jpg" width="960" height="320" alt="" /><span><big>A Hybrid
Approach for Detecting Automated Spammers in Twitter</big></span></a> <a href="#"><img
src="images/slide3.jpg" width="960" height="320" alt="" /><span><big>A Hybrid Approach for
Detecting Automated Spammers in Twitter</big></span></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
```

```
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Welcome Twitter Admin </span></h2>
<div class="clr"></div>
<p>&nbsp;</p>
<p><img src="images/Admin.jpg" width="643" height="447" /></p>
<p>&nbsp;</p>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star"><span>Sidebar</span> Menu</h2>
<div class="clr"></div>
<ul class="sb_menu style1">
<li><strong><a href="AdminMain.jsp">Home</a></strong></li>
<li><strong><a href="Admin_AuthorizeUsers.jsp">View Users and Authorize</a></strong></li>
<li><strong><a href="Admin_ViewFriendRequests.jsp">View all User Friend Request and
Response</a></strong></li>
<li><strong><a href="Admin_AddSpamFilter.jsp">Add Spam Filter Name</a></strong></li>
<li><strong><a href="Admin_AddNormalFilter.jsp">Add Normal Filter Name</a></strong></li>
<li><strong><a href="Admin_BlockSpammingAccounts.jsp">View all Spamming Accounts with
Profile and Block</a></strong></li>
<li><strong><a href="Admin_ViewUnblockReq.jsp">View All Un Block Request
Users</a></strong></li>
<li><strong><a href="Admin_ViewAllUserTweets.jsp">View all User's Tweet
Topic</a></strong></li>
<li><strong><a href="Admin_ViewCommunity_Spam_NormalAccounts.jsp">View All Spam Account
By Random Forest Tree</a></strong></li>
<li><strong><a href="Admin_ViewCommunitySpamBehaviour.jsp">View All Spamming Behaviors
and Normal Behaviors based on Interactions</a></strong></li>
```

```
<li><strong><a href="Admin_ViewCommunitySpamMetaData.jsp">View All Spamming Behaviors
and Normal Behaviors based on Tweet Meta Data</a></strong></li>
<li><strong><a href="Admin_ViewCommunity_Spam_NormalAcInChart.jsp">View Number of
Spamming Account and Normal Account in Chart</a></strong></li>
<li><strong><a href="AdminLogin.jsp">Logout</a></strong></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer"></div>
</div>
</body>
</html>
```

Here is the sample code for user profile:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>User Profile Page</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style2 {font-size: 18px}
.style3 {
```

```
font-size: 24px;
color: #FF0000;
font-weight: bold;
}
.style4 {color: #660000}
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html"><small>A Hybrid Approach for Detecting Automated <br />
Spammers in Twitter</small></a></h1>
</div>
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
<input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80"
value="Search our ste:" type="text" />
</span>
<input name="button_search" src="images/search.gif" class="button_search" type="image" />
</form>
</div>
<div class="clr"></div>
<div class="menu_nav">
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="UserLogin.jsp"><span>Users</span></a></li>
<li><a href="AdminLogin.jsp"><span>Twitter Admin</span></a></li>

</ul>
</div>
```

```
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="960" height="320" alt=""
/><span><big>A Hybrid Approach for Detecting Automated Spammers in Twitter</big></span></a>
<a href="#"><img src="images/slide2.jpg" width="960" height="320" alt="" /><span><big>A Hybrid
Approach for Detecting Automated Spammers in Twitter</big></span></a> <a href="#"><img
src="images/slide3.jpg" width="960" height="320" alt="" /><span><big>A Hybrid Approach for
Detecting Automated Spammers in Twitter</big></span></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>User <%=(String)application.getAttribute("uname") %>'s Profile..</span></h2>
<div class="clr"></div>
<p>&nbsp;</p>
<table width="523" border="1.5" align="center"  cellpadding="0" cellspacing="0"  >
<%@ include file="connect.jsp" %>

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



String s1,s2,s3,s4,s5,s6,s7,s8;
int i=0;
try
{
String query="select * from user where username='"+user+"'";
Statement st=connection.createStatement();
ResultSet rs=st.executeQuery(query);
```

```
if ( rs.next() )
{
i=rs.getInt(1);
s1=rs.getString(9);
s2=rs.getString(4);
s3=rs.getString(5);
s4=rs.getString(6);
s5=rs.getString(7);
s6=rs.getString(8);
s7=rs.getString(12);


%>
<tr>
<td width="226" rowspan="8" ><div class="style7 style26" style="margin:10px 13px 10px 13px;"
><strong><a class="#" id="img1" href="#" >
<input  name="image" type="image" src="user_Pic.jsp?id=<%=i%>" style="width:180px;
height:180px;" />
</a></strong></div></td>
</tr>
<tr>
<td  width="141" height="40" align="left" valign="middle" style="color: #2c83b0;"><div align="left"
class="style15 style42 style60 style61 style7 style8 style4" style="margin-
left:10px;"><strong>Community</strong></div></td>
<td  width="133" align="left" valign="middle" height="40" style="color: #FF3300;"><div align="left"
class="style9 style10 style22 style38" style="margin-left:20px;">
<strong><%out.println(s1);%></strong>
</div></td>
</tr>
<tr>
<td  width="141" height="37" valign="middle" style="color: #2c83b0;"><div align="left" class="style15
style42 style60 style61 style7 style8 style4" style="margin-left:10px;"><strong>E-
Mail</strong></div></td>
```

```
<td  width="133" valign="middle" height="40" style="color:#006600;"><div align="left" class="style9
style10 style22 style38" style="margin-left:20px;">
<%out.println(s2);%>
</div></td>
</tr>
<tr>
<td  width="141" height="40" valign="middle" style="color: #2c83b0;"><div align="left" class="style15
style42 style60 style61 style7 style8 style4" style="margin-
left:10px;"><strong>Mobile</strong></div></td>
<td  width="133" valign="middle" height="40" style="color: #000000;"><div align="left" class="style9
style10 style22 style38" style="margin-left:20px;">
<%out.println(s3);%>
</div></td>
</tr>
<tr>
<td height="43" align="left" valign="middle" style="color: #2c83b0;"><div align="left" class="style15
style42 style60 style61 style7 style8 style4" style="margin-
left:10px;"><strong>Address</strong></div></td>
<td  width="133" align="left" valign="middle" height="40" style="color: #000000;"><div align="left"
class="style9 style10 style22 style38" style="margin-left:20px;">
<%out.println(s4);%>
</div></td>
</tr>
<tr>
<td  width="141" height="43" align="left" valign="middle" style="color: #2c83b0;"><div align="left"
class="style15 style42 style60 style61 style7 style8 style4" style="margin-left:10px;"><strong>Date
of Birth</strong></div></td>
<td  width="133" align="left" valign="middle" height="40" style="color: #000000;"><div align="left"
class="style9 style10 style22 style38" style="margin-left:20px;">
<%out.println(s5);%>
</div></td>
</tr>
<tr>
```

```
<td  width="141" height="40" align="left" valign="middle" style="color: #2c83b0;"><div align="left"
class="style15 style42 style60 style61 style7 style8 style4" style="margin-
left:10px;"><strong>Gender</strong></div></td>
<td  width="133" align="left" valign="middle" height="40" style="color: #000000;"><div align="left"
class="style9 style10 style22 style38" style="margin-left:20px;">
<%out.println(s6);%>
</div></td>
</tr>
<tr>
<td  width="141" height="40" align="left" valign="middle" style="color: #2c83b0;"><div align="left"
class="style15 style42 style60 style61 style7 style8 style4" style="margin-
left:10px;"><strong>Account Status</strong></div></td>
<td  width="133" align="left" valign="middle" height="40" style="color: #000000;"><div align="left"
class="style9 style10 style22 style38" style="margin-left:20px;">
<%out.println(s7);%>
</div></td>
</tr>
<%
}
connection.close();
}
catch(Exception e)
{
out.println(e);
}
%>
</table>
<p>&nbsp;</p>
<p align="right" class="style14"><a href="UserMain.jsp" class="style13">Back</a></p>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star"><span>Sidebar</span> Menu</h2>
```

```
<div class="clr"></div>
<ul class="sb_menu">
<li><a href="UserMain.jsp">Home</a></li>
<li><a href="UserLogin.jsp">Logout</a></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer"></div>
</div>
</body>
</html>
```

Here is the sample code for searching friends by user:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Tweet Creation Page</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style2 {font-size: 18px}
.style3 {
font-size: 24px;
```

```
color: #FF0000;
font-weight: bold;
}
.style7 {color:#000000}
-->
</style>
<script language="javascript" type='text/javascript'>
function loadFileAsText()
{
var fileToLoad = document.getElementById("file").files[0];
var l=document.getElementById("file").value;
document.getElementById("t42").value=document.getElementById("file").value.substring(l.lastIndex
Of("\\")+1,l.lastIndexOf(""));
var fileReader = new FileReader();
fileReader.onload = function(fileLoadedEvent)
{
var textFromFileLoaded = fileLoadedEvent.target.result;
document.getElementById("textarea").value = textFromFileLoaded;
};
fileReader.readAsText(fileToLoad, "UTF-8");
}
</script>
<script language="javascript" type="text/javascript">
function valid()
{
var na3=document.s.t42.value;
var na31=document.s.tname.value;
if(na31=="")
{
alert("Please Enter Tweet Post Name");
document.s.pname.focus();
return false;
}
var na4=document.s.tabout.value;
```

```
if(na4=="")
{
alert("Please Enter Tweet About");
document.s.tabout.focus();
return false;
}
var na5=document.s.tuses.value;
if(na5=="")
{
alert("Please Enter Tweet Uses");
document.s.tuses.focus();
return false;
}
var na6=document.s.mdesc.value;
if(na6=="")
{
alert("Please Enter Meta Data Description");
document.s.mdesc.focus();
return false;
}
var na7=document.s.turl.value;
if(na7=="")
{
alert("Please Enter Tweet URL");
document.s.turl.focus();
return false;
}
var na7=document.s.towner.value;
if(na7=="")
{
alert("Please Enter Tweet Owner");
document.s.towner.focus();
return false;
}
```

```
}
</script>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html"><small>A Hybrid Approach for Detecting Automated <br />
Spammers in Twitter</small></a></h1>
</div>
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
<input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80"
value="Search our ste:" type="text" />
</span>
<input name="button_search" src="images/search.gif" class="button_search" type="image" />
</form>
</div>
<div class="clr"></div>
<div class="menu_nav">
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="UserLogin.jsp"><span>Users</span></a></li>
<li><a href="AdminLogin.jsp"><span>Twitter Admin</span></a></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="960" height="320" alt=""
/><span><big>A Hybrid Approach for Detecting Automated Spammers in Twitter</big></span></a>
<a href="#"><img src="images/slide2.jpg" width="960" height="320" alt="" /><span><big>A Hybrid
Approach for Detecting Automated Spammers in Twitter</big></span></a> <a href="#"><img
```

```
src="images/slide3.jpg" width="960" height="320" alt="" /><span><big>A Hybrid Approach for
Detecting Automated Spammers in Twitter</big></span></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Create Tweet..</span></h2>
<div class="clr"></div>
<p>&nbsp;</p>
<form action="User_CreateTweet1.jsp" method="post" name="s" target="_top" id="s"
onsubmit="return valid()"  ons="ons">
<table width="553" border="0" align="center">
<tr>
<td width="157" height="37"><span class="style7"> <strong>Tweet Post Name
:</strong></span></td>
<td width="386"><input required="required" type="text" name="tname" id="tname" /></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>Tweet About :</strong></span></td>
<td width="386"><input required="required" type="text" name="tabout" id="tabout" /></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>Tweet Uses :</strong></span></td>
<td width="386"><input required="required" type="text" name="tuses" id="tuses" /></td>
</tr>
<tr>
<td height="42" ><span class="style7"> <strong>Tweet Description :</strong> </span></td>
<td><textarea name="tdesc" id="tdesc" cols="25" rows="4"></textarea></td>
</tr>
```

```
<tr>
<td width="157" height="37"><span class="style7"> <strong>Browse Meta Data Description
:</strong></span></td>
<td width="386"><input required="required" type="file" name="t42" id="file"
onchange="loadFileAsText()" /></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>File Name :</strong></span></td>
<td width="386"> <input required="required" name="tt" type="text" id="t42" readonly="readonly"
size="21"/></td>
</tr>
<tr>
<td height="37">&nbsp;</td>
<td width="386"><textarea name="text" id="textarea" cols="25" rows="4"></textarea></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>Tweet URL :</strong></span></td>
<td width="386"><input required="required" type="text" name="turl" id="turl" /></td>
</tr>
<tr>
<td height="42" ><span class="style7"> <strong>Tweet Owner :</strong> </span></td>
<td width="386"><input required="required" type="text" name="towner" id="towner" /></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>Tweet Date & Time
:</strong></span></td>
<td><p align="left" style="color:#000000"> <%=new java.util.Date()%></p></td>
</tr>
<tr>
<td><div align="right"><span class="style8"></span></div></td>
<td><input type="submit" name="Submit" value="Encrypt" /></td>
</tr>
</table>
```

```
</form>
<p>&nbsp;</p>
<div align="center" class="style22"><a href="UserMain.jsp" class="style11">Back</a></div>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star"><span>Sidebar</span> Menu</h2>
<div class="clr"></div>
<ul class="sb_menu">
<li><a href="UserMain.jsp">Home</a></li>
<li><a href="UserLogin.jsp">Logout</a></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer"></div>
</div>
</body>
</html>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<%@page
import="com.oreilly.servlet.*,java.sql.*,java.lang.*,java.text.SimpleDateFormat,java.util.*,java.io.*,jav
ax.servlet.*, javax.servlet.http.*" %>
<%@ page import="java.sql.*"%>
<%@ include file="connect.jsp" %>
<%@ page import="java.util.Date" %>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Search Friends Page</title>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style8 {font-weight: bold; color: #660000; }
.style9 {color: #FF3300}
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html"><small>A Hybrid Approach for Detecting Automated <br />
Spammers in Twitter</small></a></h1>
</div>
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
<input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80"
value="Search our ste:" type="text" />
</span>
<input name="button_search" src="images/search.gif" class="button_search" type="image" />
</form>
</div>
<div class="clr"></div>
<div class="menu_nav">
```

```
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="UserLogin.jsp"><span>Users</span></a></li>
<li><a href="AdminLogin.jsp"><span>Twitter Admin</span></a></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="960" height="320" alt=""
/><span><big>A Hybrid Approach for Detecting Automated Spammers in Twitter</big></span></a>
<a href="#"><img src="images/slide2.jpg" width="960" height="320" alt="" /><span><big>A Hybrid
Approach for Detecting Automated Spammers in Twitter</big></span></a> <a href="#"><img
src="images/slide3.jpg" width="960" height="320" alt="" /><span><big>A Hybrid Approach for
Detecting Automated Spammers in Twitter</big></span></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Search Friends..</span></h2>
<div class="clr"></div>
<p>&nbsp;</p>
<form id="form1" method="post" action="User_SearchFriends.jsp">
<table width="392" border="0" align="center" cellpadding="2" cellspacing="2">
<tr>
<td width="145"><span class="style49 style9">Enter Friend Name:</span></td>
<td width="214"><input type="text" name="keyword" /></td>
</tr>
<tr>
<td>&nbsp;</td>
```

```
<td>&nbsp;</td>
</tr>
<tr>
<td>&nbsp;</td>
<td><input class="art-button" name="submit" type="submit" value="Search" /></td>
</tr>
</table>
<p align="center">&nbsp;</p>
<p align="right"><a href="UserMain.jsp" class="style75">Back</a></p>
</form>
</div>
<p>&nbsp;</p>
<div class="article">
<h2><span>Results Found..</span></h2>
<p>&nbsp;</p>
<%
String
s1=null,ss1=null,s2=null,s3=null,s4=null,s5=null,s6=null,s7=null,s8=null,s9=null,s10=null,s11=
null,s12=null,s13=null;
String user = (String)application.getAttribute("uname");
String community=(String)application.getAttribute("ucommunity");
String keyword = request.getParameter("keyword");
int i=0;
try
{
SimpleDateFormat sdfDate = new SimpleDateFormat("dd/MM/yyyy");
SimpleDateFormat sdfTime = new SimpleDateFormat("HH:mm:ss");
Date now = new Date();
String strDate = sdfDate.format(now);
String strTime = sdfTime.format(now);
String dt = strDate + "   " + strTime;
if(!keyword.equals(""))
{
Statement st3 = connection.createStatement();
```

```
String query3 ="insert into fsearch(username,keyword,dt) values('"+user+"','"+keyword+"','"+dt+"')";
st3.executeUpdate (query3);
String query="select * from user where (username!='"+user+"' and community='"+community+"') and
status='"+"Authorized"+"' and username  LIKE '%"+keyword+"%'";
Statement st=connection.createStatement();
ResultSet rs=st.executeQuery(query);
while ( rs.next() )
{
i=rs.getInt(1);
s1=rs.getString(2);
s2=rs.getString(4);
s3=rs.getString(5);
s4=rs.getString(6);
s5=rs.getString(7);
s6=rs.getString(8);
s7=rs.getString(10);
s8=rs.getString(9);
%>
<table border="1" width="518" style="margin:5px 10px 10px 5px;">
<tr>
<td rowspan="8" width="148" ><input  name="image2" type="image" src="user_Pic.jsp?id=<%=i%>"
style="width:150px; height:150px;" class="image_wrapper" />
</td>
<td width="118" height="27" align="left"><div align="left" class="style30"><span class="style8"
style="margin-left:20px;">User Name :</span></div></td>
<td width="249" style="margin-left:20px; color:#CC0000;"><div align="left"
class="style75"><strong><%=s1%></strong></div></td>
</tr>
<tr>
<td height="25" align="left"><div align="left" class="style30"><span class="style8" style="margin-
left:20px;"><strong>E-Mail :</strong></span></div></td>
<td style="margin-left:20px; color:#006600;"><div align="left" class="style75"><%=s2%></div></td>
</tr>
<tr>
```

```
<td height="30" align="left"><div align="left" class="style30"><span class="style8" style="margin-
left:20px;"><strong>Mobile :</strong></span></div></td>
<td style="margin-left:20px; color:#000000;"><div align="left" class="style75"><%=s3%></div></td>
</tr>
<tr>
<td height="22" align="left"><div align="left" class="style30"><span class="style8" style="margin-
left:20px;"><strong>Address :</strong></span></div></td>
<td style="margin-left:20px; color:#000000;"><div align="left" class="style75"><%=s4%></div></td>
</tr>
<tr>
<td height="26" align="left"><div align="left" class="style30"><span class="style8" style="margin-
left:20px;"><strong>DOB :</strong></span></div></td>
<td style="margin-left:20px; color:#000000;"><div align="left" class="style75"><%=s5%></div></td>
</tr>
<tr>
<td height="24" align="left"><div align="left" class="style30"><span class="style8" style="margin-
left:20px;">Gender :</span></div></td>
<td style="margin-left:20px; color:#000000;"><div align="left" class="style75"><%=s6%></div></td>
</tr>
<tr>
<td height="24" align="left"><div align="left" class="style30"><span class="style8" style="margin-
left:20px;"><strong>Community :</strong></span></div></td>
<td style="margin-left:20px; color:#000000;"><div align="left" class="style75"><%=s8%></div></td>
</tr>
<tr>
<td height="46" align="left"><div align="left" class="style30"><span class="style8" style="margin-
left:20px;"><strong>Status :</strong></span></div></td>
<td style="margin-left:20px; color:#000000;"><p align="left"
class="style76"><strong><%=s7%></strong><span style="float:right"><a
href="User_updaterequest.jsp?rname=<%=s1%>&amp;comn=<%=s8%>">
<input class="art-button" name="button2" type="button" value="Request" />
</a></span></p></td>
</tr>
</table>
```

```
<p><span class="style72">
<%
}}
}catch(Exception e){
e.getMessage();
}
%>
</span></p>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star"><span>Sidebar</span> Menu</h2>
<div class="clr"></div>
<ul class="sb_menu">
<li><a href="UserMain.jsp">Home</a></li>
<li><a href="UserLogin.jsp">Logout</a></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer"></div>
</div>
</body>
</html>
```

Here is the sample code for creating tweet by user:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Tweet Creation Page</title>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style2 {font-size: 18px}
.style3 {
font-size: 24px;
color: #FF0000;
font-weight: bold;
}
.style7 {color:#000000}
-->
</style>
<script language="javascript" type='text/javascript'>
function loadFileAsText()
{
var fileToLoad = document.getElementById("file").files[0];
var l=document.getElementById("file").value;
document.getElementById("t42").value=document.getElementById("file").value.substring(l.lastIndex
Of("\\")+1,l.lastIndexOf(""));
var fileReader = new FileReader();
fileReader.onload = function(fileLoadedEvent)
{
var textFromFileLoaded = fileLoadedEvent.target.result;
document.getElementById("textarea").value = textFromFileLoaded;
};
fileReader.readAsText(fileToLoad, "UTF-8");
}
```

```
</script>
<script language="javascript" type="text/javascript">
function valid()
{
var na3=document.s.t42.value;
var na31=document.s.tname.value;
if(na31=="")
{
alert("Please Enter Tweet Post Name");
document.s.pname.focus();
return false;
}
var na4=document.s.tabout.value;
if(na4=="")
{
alert("Please Enter Tweet About");
document.s.tabout.focus();
return false;
}
var na5=document.s.tuses.value;
if(na5=="")
{
alert("Please Enter Tweet Uses");
document.s.tuses.focus();
return false;
}
var na6=document.s.mdesc.value;
if(na6=="")
{
alert("Please Enter Meta Data Description");
document.s.mdesc.focus();
return false;
}
var na7=document.s.turl.value;
if(na7=="")
{
alert("Please Enter Tweet URL");
document.s.turl.focus();
return false;
}
var na7=document.s.towner.value;
if(na7=="")
{
alert("Please Enter Tweet Owner");
document.s.towner.focus();
return false;
}
}
</script>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html"><small>A Hybrid Approach for Detecting Automated <br />
Spammers in Twitter</small></a></h1>
</div>
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
<input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80"
value="Search our ste:" type="text" />
</span>
<input name="button_search" src="images/search.gif" class="button_search" type="image" />
</form>
</div>
<div class="clr"></div>
```

```
<div class="menu_nav">
<ul>
<li class="active"><a href="index.html"><span>Home Page</span></a></li>
<li><a href="UserLogin.jsp"><span>Users</span></a></li>
<li><a href="AdminLogin.jsp"><span>Twitter Admin</span></a></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="960" height="320" alt=""
/><span><big>A Hybrid Approach for Detecting Automated Spammers in Twitter</big></span></a>
<a href="#"><img src="images/slide2.jpg" width="960" height="320" alt="" /><span><big>A Hybrid
Approach for Detecting Automated Spammers in Twitter</big></span></a> <a href="#"><img
src="images/slide3.jpg" width="960" height="320" alt="" /><span><big>A Hybrid Approach for
Detecting Automated Spammers in Twitter</big></span></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Create Tweet..</span></h2>
<div class="clr"></div>
<p>&nbsp;</p>
<form action="User_CreateTweet1.jsp" method="post" name="s" target="_top" id="s"
onsubmit="return valid()"  ons="ons">

<table width="553" border="0" align="center">
<tr>
<td width="157" height="37"><span class="style7"> <strong>Tweet Post Name
:</strong></span></td>
```

```
<td width="386"><input required="required" type="text" name="tname" id="tname" /></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>Tweet About :</strong></span></td>
<td width="386"><input required="required" type="text" name="tabout" id="tabout" /></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>Tweet Uses :</strong></span></td>
<td width="386"><input required="required" type="text" name="tuses" id="tuses" /></td>
</tr>
<tr>
<td height="42" ><span class="style7"> <strong>Tweet Description :</strong> </span></td>
<td><textarea name="tdesc" id="tdesc" cols="25" rows="4"></textarea></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>Browse Meta Data Description
:</strong></span></td>
<td width="386"><input required="required" type="file" name="t42" id="file"
onchange="loadFileAsText()" /></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>File Name :</strong></span></td>
<td width="386"> <input required="required" name="tt" type="text" id="t42" readonly="readonly"
size="21"/></td>
</tr>
<tr>
<td height="37">&nbsp;</td>
<td width="386"><textarea name="text" id="textarea" cols="25" rows="4"></textarea></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>Tweet URL :</strong></span></td>
<td width="386"><input required="required" type="text" name="turl" id="turl" /></td>
</tr>
<tr>
```

```
<td height="42" ><span class="style7"> <strong>Tweet Owner :</strong> </span></td>
<td width="386"><input required="required" type="text" name="towner" id="towner" /></td>
</tr>
<tr>
<td width="157" height="37"><span class="style7"> <strong>Tweet Date & Time
:</strong></span></td>
<td><p align="left" style="color:#000000"> <%=new java.util.Date()%></p></td>
</tr>
<tr>
<td><div align="right"><span class="style8"></span></div></td>
<td><input type="submit" name="Submit" value="Encrypt" /></td>
</tr>
</table>
</form>
<p>&nbsp;</p>
<div align="center" class="style22"><a href="UserMain.jsp" class="style11">Back</a></div>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star"><span>Sidebar</span> Menu</h2>
<div class="clr"></div>
<ul class="sb_menu">
<li><a href="UserMain.jsp">Home</a></li>
<li><a href="UserLogin.jsp">Logout</a></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg"></div>
<div class="footer"></div>
</div></body></html>
```

REFERENCES

[1] M. Tsikerdekis, “Identity deception prevention using common contribution network data,” *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 1,pp. 188–199, Jan. 2017.

[2] T. Anwar and M. Abulaish, “Ranking radically influential Web forum users,” *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 6,pp. 1289–1298, Jun. 2015.

[3] Y. Boshmaf, I. Muslukhov, K. Beznosov, and M. Ripeanu, “Design and analysis of social botnet,” *Comput. Netw.*, vol. 57, no. 2, pp. 556–578,2013.

[4] D. Fletcher, “A brief history of spam,” *TIME*, Nov. 2, 2009.[Online]. Available:http://www.time.com/time/business/article/0,8599,1933796,00.html

[5] Y. Boshmaf, M. Ripeanu, K. Beznosov, and E. Santos-Neto, “Thwarting fake OSN accounts by predicting their victims,” in *Proc. AISec*, Denver,CO, USA, 2015, pp. 81–89.

[6] A. A. Amleshwaram, N. Reddy, S. Yadav, G. Gu, andC. Yang, “CATS: Characterizing automation of Twitter spammers,” in *Proc. COMSNETS*, Bengaluru, India, Jan. 2013, pp. 1–10.

[7] K. Lee, J. C. Lee, and S. Webb, “Uncovering social spammers: Socialhoneypots + machine learning,” in *Proc. SIGIR*, Geneva, Switzerland,Jul. 2010, pp. 435–442.

[8] G. Stringhini, C. Kruegel, and G. Vigna, “Detecting spammers on socialnetworks,” in *Proc. ACSAC*, Austin, TX, USA, 2010, pp. 1–9.

[9] H. Yu, M. Kaminsky, P. B. Gibbons, and A. D. Flaxman, “SybilGuard:Defending against sybil attacks via social networks,” *IEEE/ACM Trans.Netw.*, vol. 16, no. 3, pp. 576–589, Jun. 2008.

[10] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Y. Zhao, “Detectingand characterizing social spam campaigns,” in *Proc. IMC*, Melbourne,VIC, Australia, 2001, pp. 35–47.

[11] W. Wei, F. Xu, C. C. Tan, and Q. Li “Sybildefender: Defend against sybil attacks in large social networks,” in *Proc. INFOCOM*, Orlando, FL, USA, Mar. 2012, pp. 1951–1959.

[12] C. Yang, R. C. Harkreader, and G. Gu, “Die free or live hard? Empirical evaluation and new design for fighting evolving Twitter spammers,” in *Proc. RAID*, Menlo Park, CA, USA, 2011, pp. 318–337.

[13] S. Lee and J. Kim, “WarningBird: A near real-time detection system forsuspicious URLs in Twitter stream,” *IEEE Trans. Depend. Sec. Comput.*,vol. 10, no. 3, pp. 183–195, May 2013.

[14] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, “A Bayesianapproach to filtering junk e-mail,” in *Proc. Workshop Learn. Text Categorization*, Madison, WI, USA, 1998, pp. 98–105.

[15] C. Schäfer, “Detection of compromised email accounts used by a spam botnet with country counting and theoretical geographical travelling speed extracted from metadata,” in *Proc. ISSREW*, Naples, Italy,Nov. 2014, pp. 329–334.

[16] C. Schäfer, “Detection of compromised email accounts used for spamming in correlation with origin-destination delivery notification extracted from metadata,” in *Proc. ISDFS*, Tirgu Mures, Romania, Apr. 2017,pp. 1–6.

[17] A. H. Wang, “Detecting spam bots in online social networking sites:A machine learning approach,” in *Proc. DBSec*, Rome, Italy, 2010,pp. 335–342.

[18] F. Ahmed and M. Abulaish, “A generic statistical approach for spam detection in online social networks,” *Comput. Commun.*, vol. 36,nos. 10–11, pp. 1120–1129, 2013.

[19] C. Yang, R. Harkreader, and G. Gu, “Empirical evaluation and new design for fighting evolving Twitter spammers,” *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 8, pp. 1280–1293, Aug. 2013.

[20] Y. Zhu, X. Wang, E. Zhong, N. N. Liu, H. Li, and Q. Yang, “Discovering spammers in social networks,” in *Proc. AAAI*, Toronto, ON, Canada,2012, pp. 52–58.

[21] E. Tan, L. Guo, S. Chen, X. Zhang, and Y. Zhao, “Spammer behavioranalysis and detection in user generated content on social networks,”in *Proc. ICDCS*, Macau, China, Jun. 2012, pp. 305–314.

[22] S. Y. Bhat and M. Abulaish, “Community-based features for identifying spammers in online social networks,” in *Proc. Int. Conf. Adv.Social Netw. Anal. Mining*, Niagara Falls, ON, Canada, Aug. 2013,pp. 100–107.

[23] L. M. Aiello, M. Deplano, R. Schifanella, and G. Ruffo, “People are strange when you’re a stranger: Impact and influence of bots on socialnetworks,” in *Proc. AAAI*, Dublin, Ireland, 2012, pp. 10–17.

[24] S. Cresci, R. D. Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, “The paradigm-shift of social spambots: Evidence, theories, and toolsfor the arms race,” in *Proc. WWW*, Perth, WA, Australia, 2017,pp.