Detection of Flames in Computer Mediated Communication

by

Sachin Bhardwaj - 081222

Vidhushree Arya – 081223

Vishal Garg -081292

Under the supervision of

Dr. Nitin

MAY 2012

Submitted in partial fulfilment of the Degree of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING and INFORMATION TECHNOLOGY

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY
PO WAKNAGHAT,
DISTRICT SOLAN, HIMACHAL PRADESH
INDIA

TABLE OF CONTENTS

CERTIFICATE FROM SUPERVISOR

This is to certify that the work titled — **"Detection of Flames in Computer Mediated Communication".**submitted by **Sachin Bhardwaj ,Vidhushree Arya ,Vishal Garg** in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor	Nitin
Name of Supervisor	Dr. Nitin
Designation	Associate Professor
Date	28th May '12

ACKNOWLEDGEMENT

As we conclude our project with the God's grace, we have many people to thank for all the help, guidance and support they lent us, throughout the course of our endeavor.

First and foremost, we are sincerely thankful to Dr.Nitin, our Project Guide, who has always encouraged us to put in our best efforts and deliver a quality and professional output. His methodology of making the system strong from inside has taught us that output is not the END of project. We really thank him for his time & efforts.

We are deeply indebted to all those who provided reviews and suggestions for improving the materials and topics covered in our project,and we extend our apologies to anyone we may have failed to mention.

Sachin Bhardwaj
081222
CSE

Vidhushree Arya
081223
CSE

Vishal Garg
081292
CSE

SUMMARY

The growth in the use of global networks of computers, commonly labeled Internet, has led to questions about how the Internet can be understood and analyzed as an emergent communication tool.

In fact, computer-mediated communication (CMC) has helped in creating new opportunities for synchronous and asynchronous discussions among geographically dispersed individuals sitting at their computer keyboards. Everyday, thousands of people engage in "written" conversation using a novel linguistic genre, which combines characteristics of both written and oral language. They type their messages, and these messages appear on the screens of their interlocutors, preceded by the "speaker's" nickname or address.themselves as compared to Face-to-face communication.

The most common form of asynchronous CMC is e-mail, in which a sender leaves a message in a receiver's electronic letterbox, which the receiver has to open before he can read the message .Another more sophisticated type of asynchronous CMC is Usenet Newsgroup, an electronic notice board in which users can post messages referring to a specific topic or area of interest. Users read their messages by opening the notice board, and send their own messages in turn. As with e-mail, there is no real-time link between the computers of the interacting subjects.

Unlike asynchronous CMC, the most important feature of synchronous CMC is that it provides a real-time link between users' computers. Although the most frequent example is the videoconference, the most widespread system is the Internet Relay Chat, or IRC. IRC is a form of synchronous CMC allowing a group of users to “chat” by exchanging written messages. They can interact in two different ways: by sending message either to a specified user, or to all members of the group.

INTRODUCTION

The interaction among the users is not always clear: in a chat the conversation appears on the screen as a linear progression of lines of text, regardless of the conversation's dynamics. Finally, the temporal information found in oral conversations, such as turn-taking and the negotiation of conversational synchrony by the participants, is not captured by these sequential lines of text. As noted by Mabry [18], this leads to prevalence of argumentative exchanges that are in many cases characterized by a strong and destructive style.

In addition, CMC does not in any way guarantees that a user's declared identity is the real one. The use of false identities, often of a different sex, is widespread in electronic communities and in IRC.The anonymity of CMC, facilitating and facilitated by concealment in self presentation, allows distortions to be made through omission of information as well as through selective presentation.

In general four particular form of miscommunication are common in CMC : offensive conduct (flaming), listening without making his/her presence known (lurking), delivering a message to someone who would not otherwise choose to receive it (spamming and bombing) and identity deception (spoofing). In the next paragraphs we will try to deepen the analysis of these behaviors.

The Dark Side of CMC:-

1. FLAMING

Even though meta-analysis on existing studies found that the incidence of offensive conduct is overrated, different areas of CMC are characterized by intense language, swearing, negative or hostile communication. As experienced by many users of Usenet Newsgroup or Internet Relay Chat, the intensity of many communicative exchanges is usually heat.

To reduce the number of offensive messages, net groups have established a norms of network usage - that addresses specifically how the user can write and post messages. These norms stress obligations for group and self-monitoring to insure that members maintain a correct language, respect for the interlocutor and communicative relevance. The netiquette, however, is not the same everywhere. Some offenses are seen as more disturbing than others, and it is equally possible that what one group condemns, another condones. For example, while directing a particularly hostile message at another user is perfectly acceptable on some approach is usually censored by many socially oriented newsgroups.

The typical breach of netiquette involves the use of *flames*. With this word Siegel and colleagues defined "messages that are precipitate, often personally derogatory, ad hominem attacks directed towards someone due to the position taken in a message distributed
(posted) to the group"

However, the definition of flaming in CMC literature varies. For instance Rice describes flaming as "the tendency to react more critically or with greater hostility, leading to the rise ofconflict" Following this line Walther defines it as "insults, swearing, and hostile instances of behavior".

Flaming is composed by CMC behaviors that are interpreted to be inappropriately hostile.
This definition focuses on a very important point: for a flame to take place two separate actions must occur. First a behavior has to be created. Then someone else has to interpret the behavior as being offensive.

Some network groups have taken a step in this direction by establishing netiquette – rules of network usage – which specially tells users on how they can write and post their messages.

These rules emphasize on obligation for group and self-monitoring to ensure decorum to be maintained throughout the discussions by use of correct language, respect for other users and communicative relevance. These netiquette rules differ on different sites. Some offenses are likely to be considered more hostile than others and there is a possibility that what certain group condemns, others are indifferent to or condones. For example, while posting a certain hostile message directed to another user, it is acceptable on some newsgroups whereas this is usually not allowed and censored by social oriented newsgroups.

2. Lurking

The word "*Lurking*" defines the behavior of subscribers to electronic forums who
Rarely or never send contributions to the discussions

3. Spamming and Bombing

Spamming is currently described as an attempt to deliver a message, over the Internet, to someone who would not otherwise choose to receive it.

The term was originally used in Usenet newsgroups to describe identical commercial or off-topic posts made to multiple newsgroups. It has been expanded to include ordinary email messages since then. Most of the spams are composed by commercial advertising (unsolicited commercial e-mail or unsolicited bulk email). Potential target list is created with automated searches by scanning Usenet postings, stealing Internet mailing lists, or searching the Web for addresses.

A variant of email spamming is "bombing". Email "bombing" is characterized by repeatedly sending an identical message to a specific e-mail address. There are two
different forms of bombing:

- *Massmail bombing*: a subject sends to any specific e-mail address hundreds, or perhaps even thousands of pieces of email, usually by the means of a script and fakemail. It is relatively easy to defend against, since the messages will be coming from just a few e -mail addresses.

- *Mailing List bombing*: in this a subject subscribes a specific e-mail address to as many Mailing lists as possible. This is much worse than a mass mail because it requires to unsubscribe from each mailing list, a process that requires both information and time.

Be Discreet

A user should always keep a track of what he/she is typing into his/her profile. One should never type anything that is likely to attract unwanted users. User should think before posting something on his wall. Like if she/he posts something funny it is destined to attract other users and some fake users probably. So user should always be discreet in what she/he is doing.

Be Skeptical

People in general tend to click on all the posts by other users. It should be restricted to only those posts that are somehow useful or are of importance to the user. User should not click on something that is not useful to him/her as it may also attract fake users.

Be Thoughtful

User should never be a loud mouth and type anything that may come back and bite him/her. User should always think twice before typing anything because this is how flaming gets aroused and the problem starts.

Be Professional

If a user is to post a video or a picture then it should be something that presents him/her in the best possible light or is professional rather than giving a funny look because it is unprofessional. Funny uploads might give a bad look professionally. So the uploads must be chosen correctly.

Be Wary

A user should first verify the identity of the person whom (s)he is chatting with. The person might be using a fake account and leak some information.

Expressing emotions in CMC

CMC interlocutors are forced to find alternative way for reproducing the metacommunicative features (emotions, illocutionary force, etc.) of face-to-face conversation. According to Utz it is possible to identify three different forms of emotional expressions in CMC: emoticons, social verbs and emotes *Emoticons* (also called *smileys*) are the most used textual devices: ASCII glyphs designed to show an emotional state in plain text messages. They consist of various punctuation marks and are viewed by tilting one's head to the left or turning the page sideways.

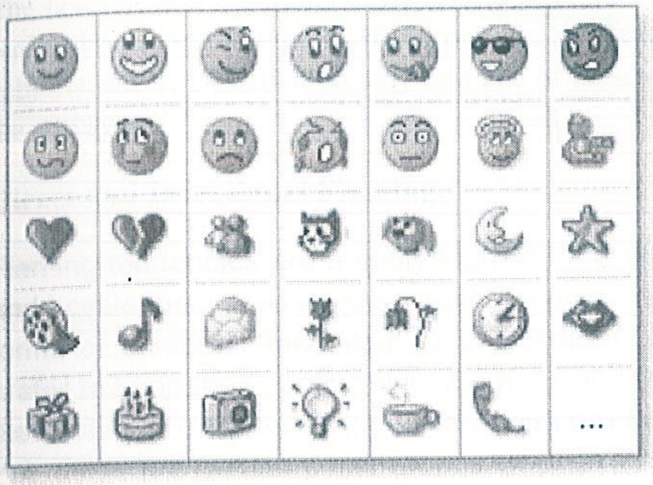
In particular, emoticons are described as "graphicrepresentations of facial expressions that many e-mail users embed in their messages. These symbols are widely known and commonly recognized among computer-mediated communication users, and are described by most observers as substituting for the nonverbal cues that are missing from the CMC in comparison to face-to-face communication"

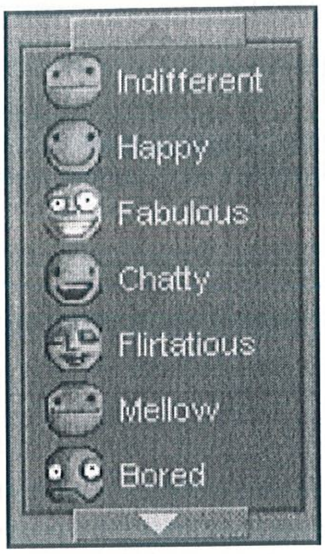
As noted by Wolf, many different emoticon collections exist online in several different languages. And for non verbal emotional expression, there is some confusion in emoticon interpretations: "In some instances the emoticon :-Q means *user smokes*; others define it as meaning *sticking out tongueor tongue hanging out in nausea.* There is a general acceptance in the interpretation of the basic smileys, frowneys, and winkeys emoticons their respective meaning is humor, sadness, and sarcasm. However, the more elaborate the emoticons become, the greater variation one finds in the interpretations.

THE MOST COMMON EMOTICONS

Symbol	*Meaning*	*Symbol*	*Meaning*
:-)	Basic smiley face; used for humor and sometimes sarcasm	/\/\/\	Laughter
:-(	Basic frowney face; used for sadness or anger	:-D	Laughing
;-)	Half-smiley or winkey face; used for sarcasm	!-(	Black eye
:-/	Wry face; used for wry humor	8-O	Astonished
#:-o	Shocked (1)	8-]	Surprise
8-o	Shocked (2)	: (	Sad
%-\	Hung over	: [	Bored
%-{	Ironic	:,(	Crying
>>:-<<	Furious	:-@	Screaming
>-	Female	(:-\	Very sad (1)
:-	Male	:-<	Very sad (2)
>:-<	Mad	:-,	Smirk
>:-(	Annoyed	:-6	Exhausted
<:-|	Dunce	:-e	Disappointed
(:&	Angry	%-(	Confused
(:-*	Kiss	8-#	Death

ADVANCED EMOTICONS





Emotional Expression	*Description*	*Example*
Emoticons	ASCII glyphs designed to show an emotional state in plain text messages	:-) humor :-(sadness ;-) sarcasm
Social verbs (feelings)	Small pre-programmed scripts to express actions and emotions by simply typing an abbreviation	*command*; laugh *appears on screen*: You fall down laughing *command*: smi Aron und *appears on screen*: You smile understandingly at Aron
Emotes	Narrative descriptions of conversational nonverbal behaviors	*command*: emote/say/pose is so happy that he could embrace the whole world *appears on screen*: <Name> is so happy that he could embrace the whole world.

Classification of Flames:

The research has showed a need to classify flames on SNS as it revealed some patterns of replies based on the type of flame posted by the users. All the statuses given to the survey subjects can be classified broadly into one of these categories. Some examples are taken from the survey.

Direct and Intentional Flaming:

Flaming tendencies are notably high when users intentionally use abusive, incendiary and hostile messages against another user or faction. This is pre-dominant on different forms of computer mediated communication but is less seen on where SNS users prefer to keep their confrontations private and not publicize them to all their friends on their list or to unnecessary people. Users are more conscious of their actions on SNS. Yet, there are certain small groups who take such steps and use venues like status messages, comments, etc for flaming.

This type of flaming can be defined as:

"Directing hostile, non-friendly, aggressive literature towards other users to show disagreement or oppose their statements or ideologies."

Such flaming patterns are seldom seen on status messages but are more predominant on discussions in groups or community venues. Such flames are reciprocated by other users by using direct flames.

Indirect Flaming:

This is generally seen on all forms of computer mediated communications including SNS. Indirect flaming is generally opted for to publicize disagreement or hostility but posted in a language which can only be understood by the people involved. Friends of the users who read such messages would note the state of disagreement but would seldom be able to track references or to whom the flame is intended towards.

"Use of hostile, non-friendly and aggressive literature towards users or situations not clearly mentioned to show disagreement, but with a subtlety that only the factions involved is capable of deriving the true intention of the statement."

Such flaming patterns are seen on status messages which are made public to all the friends. Such messages are posted to show disagreement.

Where Do People Flame?

It has already been analyzed and established that, "Participants might use the sites to interact with people they already know offline or to meet new people" (Ellison, N. B., Steinfield, C., & Lampe, C. (2007)). It has been observed that there are certain hotspots where flaming is more predominant than the others.

MySpace,Facebook and a few other SNSs have a feature which can be collectively called as "status updates" (also referred to as "status") which allows users to post messages or quotes for all people on the friend list to read. In turn, friends respond with their own personal comments with their reactions to the "status". A user's most recent status update usually appears at the top of his/her profile, while other statuses may be recorded on the profile for continuing any conversation on it.

Actually, the main purpose of the feature was to allow users to inform their friends of their current "status" (i.e. their current feelings, whereabouts or actions). As in the case of Facebook, users could do this by referring to themselves as the third person (e.g. "*user* is happy" or "*user* is having a party at his house"). However, on December 13, 2007, the requirement to start a status update with '*is*' was removed. The question "What are you doing right now?" was introduced. In March 2009, the status update question was changed from "What are you doing right now?" to "What's on your mind?"

With the introduction of the concept of "What's on your mind?" and other similar references, users were free to post almost anything, since such a concept is greatly universal. This also led to users posting messages of joy, sorrow, frustration,anger, quotations or philosophical statuses. Such messages are often reciprocated with appropriate situational replies. Flaming in statuses is seen when any user posts rude, incendiary or sarcastic messages of another user or a group of people.

Litereature Survey

Concept of Survey

Following was the method that was adopted for study of the general reaction and comment patterns when faced with hostile or incendiary status or messages when presented to different users. The research used a method of survey using the "Status hostility Scale" (Turnage A.K., 2007) – which is a semantic differential scale measuring responses on three different situations which consisted five corresponding status messages each. The participants were asked to rate the hostility of each of the status messages on a five-point scale. They were also asked their choice of status message that they would choose and put as their own status, out of the five provided with respect to each of the situations.

They were also asked that, if one of their 'Non-Met' friends were to put such a hostile status message, what would be their response and reaction to it, as this helps us in the better understanding of flaming patterns on SNSs.

The situations were selected from real life world incidents that would be invoke most contrasting and vehement of responses. Such situations were designed to study the response patterns of people, which also helps understand what online personalities of themselves would people like to present on SNSs.

Participants

Sample of study consists of 70 subjects with different cultural and social backgrounds who are members of at least one popular SNS.

Age Group(in yrs)	Number of Subjects	Percentage (%)
20 to 30	44	62.857
30 to 40	12	17.143
40 to 50	6	8.571
Above 50	8	11.428
	Total = 70	Total = 100

General Details of the Subjects

The sample consists of 58.57% males (number = 41) and 41.43% females (Number = 29) which are from the different age groups and their percentages are defined in the table given above.

All the 70 people are users of one or more SNSs for duration of 1 year and above and have mean of 266.67 friends in their respective friend list. All subjects were asked to give the number of 'Non Met' users in their friend list. This generated a mean percentage of 12.127% (Number = 32.34) friends whom the subjects had accepted as friends but never met in person ('Non-Met' Friends). The research also found that the number of Non-Met friends on female friend lists were less than of the male subjects in the survey. The number of Non-Met friends was around 13.32% of their total friend number while the number was only 8.72% for female subjects.

Procedures

The participants were asked to complete the survey subject matter, which consisted of three sections. The first being general personal details, such as age, gender, number of friends, Number of 'Non-Met' friends and Period of activity on at least one SNS.

The second section presented the subjects with three different scenarios. Each of the scenarios put forth five different status messages with varying level of hostility and aggressiveness. Every subject was asked to rate the hostility of each of the messages as per their personal discretion based on "Status Hostility Scale", where '*1*' represented '*not-hostile*' and '*5*' represented *'very hostile'*. One additional question was presented along with the situations where the user was asked to choose one of status messages, which they would be willing to put up as their own status message. The idea behind this was to know the pattern of the participants with respect to all other subjects; on what level of hostility would they choose in their own status messages.

The third section presented a hostile status message along with a hypothetical situation that such a status message was posted by one of their 'Non-Met' friends. The subjects were asked to consider this situation and present their reaction, if they would reply to such a message with any level of flaming.

Results

In second section, these were the situations presented to the survey subjects.

Situation 1: *During the final of the 2006 FIFA World cup (Soccer), Italian defender Marco Materazzi and French mid-fielder Zinedine Zidane briefly exchanged words. Moments later, Zidane suddenly stopped, turned around and rammed his head into Materazzi's chest, knocking him to the ground. Zidane was given a red card for this. The match later went to penalty shootouts, in which France lost the cup.*

These were the various statuses that were presented with the above situation:

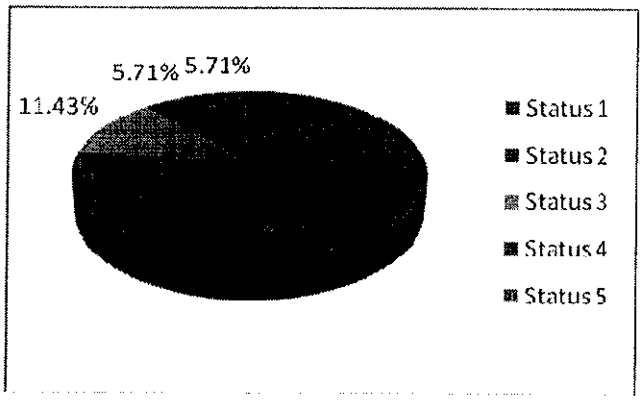
Status 1: "The only way Italians can win the cup is to get their opponents Red Carded. It's a team which has no spirit of the game."
Result: This message had an average rating of 3.182. Only 18.58% (Number =13) of the subjects selected this message.

Status 2: "The one who loses his cool loses the game in the end – the best example being the FIFA final."
Result: This message had an average rating of 1.633. Majority of subjects chose this as their status message. On statistical grounds, 58.57% (Number = 41) of the subjects selected this message.

Status 3: "Zidane is a fool to lose his cool like that, regardless of non sportsman like behaviour of the Italians; he should have focused in the game and played for his country."
Result: This message had an average rating of 2.152. Around 11.43% (Number=8) of subjects selected this message.

Status 4: "Materazzi and Zidane are big time @$$#%&@$"
Result This message had an average rating of 3.433. About 5.71% (Number=4) of the subjects selected this message.

Status 5: Materazzi and Zidane have disgraced the game. The Result was only pure chance."
Result: This message had an average rating of 2.32. Around 5.71% (Number=4) of the subjects selected this message.
The following chart summarizes the choices of the subjects:



Percentage of Subjects Choosing a Status (Situation 1)

3. OBJECTIVES DEFINED

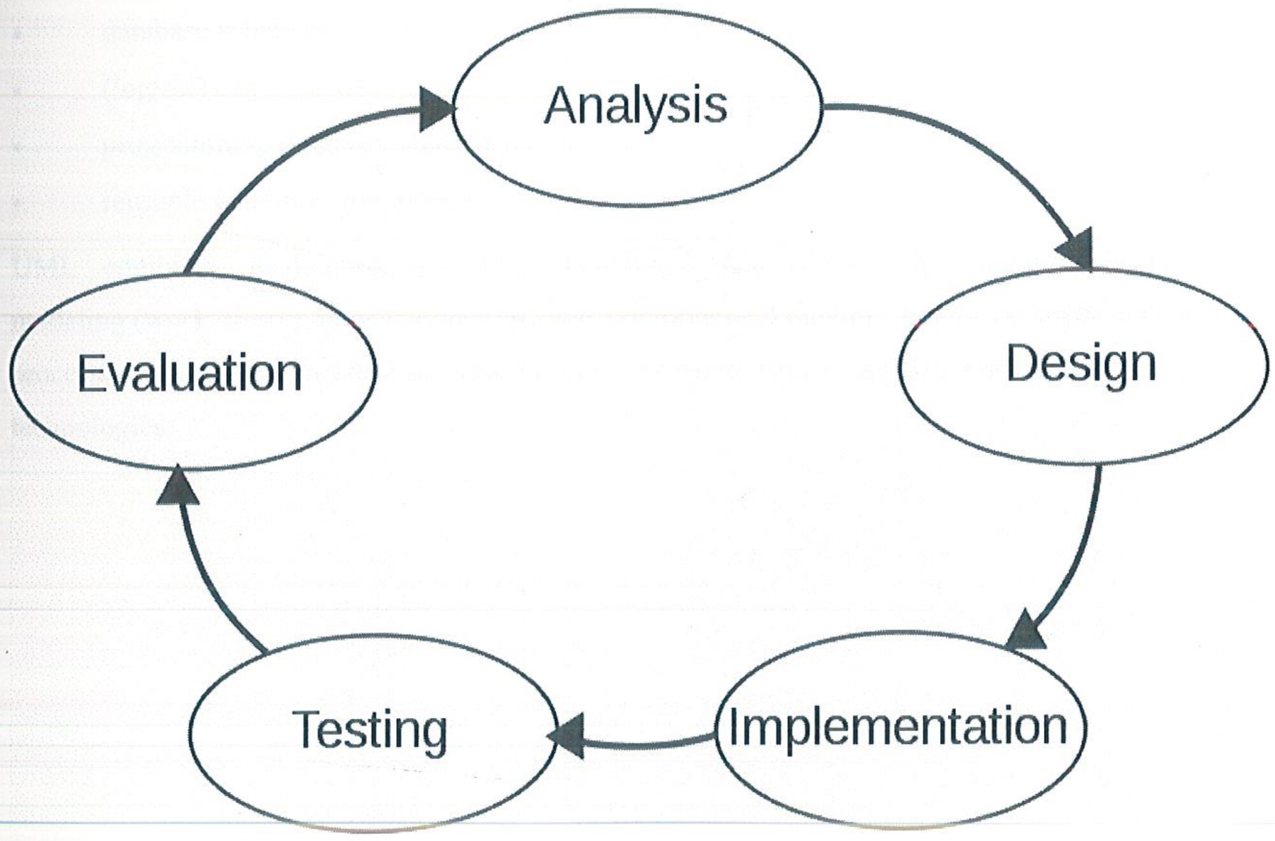
Making a flame detection algorithm based on the new user trends in social networking websites. We have introduced an algorithm for flamimg detection. Using this algorithm we aim to filter messages for a user according to the kind of language being used, that is this algorithm allows us to filter all kinds of ambiguous and vulgar messages being flashed on social networking websites like Facebook, Orkut, Twitter, Linkedin, etc.

Communication has always been an essential part of human life. In current lifestyle, Computer Mediated Communication has been of vital importance as it creates opportunities for communication between geographically dispersed individuals. With the growth of Internet, CMC also grew and is now part of almost every person. So the Social Networking sites like Facebook, LinkedIn, Twitter are also becoming more and more popular as they provide for CMC. As every coin has two sides, so is with our CMC. If it has been a providing for communication, also it is providing for flaming. So this paper aims at trying to stop Flaming in CMC and blocking users not abiding by the rules.

1. DESIGNING

SYSTEMS DEVELOPMENT LIFE CYCLE

The **systems development life cycle (SDLC)**, or ***software development life cycle*** in systems engineering, information systems and software engineering, is a process of creating or altering information systems, and the models and methodologies that we will use to develop our system.In software engineering the SDLC concept underpins many kinds of software development methodologies. These methodologies form the framework for planning and controlling the creation of an information system.



SDLC MODEL

1.1 DESIGN THEORY

The Unified Modeling Language (UML) is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software-intensive system under development. UML offers a standard way to visualize a system's architectural blueprints, including elements such as:

- activities
- actors
- business processes
- database schemas
- (logical) components
- programming language statements
- reusable software components.

UML combines techniques from data modeling (entity relationship diagrams), business modeling (work flows), object modeling, and component modeling. It can be used with all processes, throughout the software development life cycle, and across different implementation technologies.

CODING

Code for Main Window

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

/*
 * mainwindow.java
 *
 * Created on Sep 14, 2011, 10:35:39 AM
 */

/**
 *
 * @author 081221
 */
package avs;
public class mainwindow extends javax.swing.JFrame {

    /** Creates new form mainwindow */
    public mainwindow() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this
method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jMenuBar1 = new javax.swing.JMenuBar();
        jMenu1 = new javax.swing.JMenu();
        jMenuItem1 = new javax.swing.JMenuItem();
        jMenuItem2 = new javax.swing.JMenuItem();
        jMenu2 = new javax.swing.JMenu();
        jMenuItem3 = new javax.swing.JMenuItem();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CL
OSE);
        setTitle("Fl@me DeTector");
        setResizable(false);

        jLabel1.setFont(new java.awt.Font("Papyrus", 3, 48));
        jLabel1.setText("Fl@me DeTector.");

        jLabel2.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/avs/untitled.JPG
"))); // NOI18N

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLa
yout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()
                        .addGap(141, 141, 141)
                        .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 445,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(jPanel1Layout.createSequentialGroup()
                        .addGap(218, 218, 218)
                        .addComponent(jLabel2)))
                .addContainerGap(206, Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 96,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(43, 43, 43)
```

```
                    .addComponent(jLabel2)
                    .addContainerGap(82, Short.MAX_VALUE))
        );

        jMenu1.setText("Social Network");

jMenu1.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);


jMenuItem1.setAccelerator(javax.swing.KeyStroke.getKeyStroke(ja
va.awt.event.KeyEvent.VK_T,
java.awt.event.InputEvent.CTRL_MASK));
        jMenuItem1.setFont(new java.awt.Font("Comic Sans MS",
0, 14));
        jMenuItem1.setForeground(new java.awt.Color(0, 204,
255));
        jMenuItem1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/avs/twitter_logo
.jpg"))); // NOI18N
        jMenuItem1.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem1ActionPerformed(evt);
            }
        });
        jMenu1.add(jMenuItem1);


jMenuItem2.setAccelerator(javax.swing.KeyStroke.getKeyStroke(ja
va.awt.event.KeyEvent.VK_E,
java.awt.event.InputEvent.CTRL_MASK));
        jMenuItem2.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem2ActionPerformed(evt);
            }
        });
        jMenu1.add(jMenuItem2);

        jMenuBar1.add(jMenu1);

        jMenu2.setText("Help");


jMenuItem3.setAccelerator(javax.swing.KeyStroke.getKeyStroke(ja
va.awt.event.KeyEvent.VK_H,
java.awt.event.InputEvent.CTRL_MASK));
        jMenuItem3.setText("About");
        jMenuItem3.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem3ActionPerformed(evt);
            }
        });
        jMenu2.add(jMenuItem3);

        jMenuBar1.add(jMenu2);

        setJMenuBar(jMenuBar1);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void
jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jMenuItem1ActionPerformed
   Twitterwindow t=new Twitterwindow();
   t.setVisible(true);
   t.setSize(800,600);
   t.Operation();
    }//GEN-LAST:event_jMenuItem1ActionPerformed
    private void
jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jMenuItem2ActionPerformed
          System.exit(1);        // TODO add your handling code
here:
    }//GEN-LAST:event_jMenuItem2ActionPerformed

    private void
jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jMenuItem3ActionPerformed
     new Help().setVisible(true);        // TODO add your
handling code here:
    }//GEN-LAST:event_jMenuItem3ActionPerformed

    /**
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new mainwindow().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-
BEGIN:variables
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JMenu jMenu1;
    private javax.swing.JMenu jMenu2;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JMenuItem jMenuItem1;
    private javax.swing.JMenuItem jMenuItem2;
    private javax.swing.JMenuItem jMenuItem3;
    private javax.swing.JPanel jPanel1;
    // End of variables declaration//GEN-END:variables

}
```

Code For TwitterWindow

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

/*
 * Twitterwindow.java
 *
 * Created on Sep 14, 2011, 10:38:43 AM
 */

package avs;

import java.awt.Font;
import java.awt.geom.*;
import java.util.logging.Logger;
import org.openide.util.NbBundle;
import java.sql.*;
//import org.openide.util.ImageUtilities;
import org.netbeans.api.settings.ConvertAsProperties;
import avs.StatusType;
import avs.Statuses;
import java.awt.BorderLayout;
import java.awt.Canvas;
import java.awt.Color;
import java.awt.Frame;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Shape;
import javax.swing.JFrame;

/**
 *
 * @author 081221
 */
class draw{
    public void paint(Graphics g) {
      Graphics2D g2 = (Graphics2D) g;
      g2.drawRect(0, 0, 10, 10);}
}

public class Twitterwindow extends javax.swing.JFrame {
    Connection con;

    /** Creates new form Twitterwindow */
    public Twitterwindow() {
        initComponents();




    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this
method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jScrollPane1 = new javax.swing.JScrollPane();
        jFrame1 = new javax.swing.JFrame();
        jMenuBar1 = new javax.swing.JMenuBar();
        jMenu1 = new javax.swing.JMenu();
        jMenu2 = new javax.swing.JMenu();
        jPanel1 = new javax.swing.JPanel();
        jScrollPane2 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        jScrollPane3 = new javax.swing.JScrollPane();
        jTextArea2 = new javax.swing.JTextArea();
        loginButton = new javax.swing.JButton();
        jPanel2 = new javax.swing.JPanel();
        getStatusesButton = new javax.swing.JButton();
        jMenuBar2 = new javax.swing.JMenuBar();
        jMenu3 = new javax.swing.JMenu();
        jMenuItem1 = new javax.swing.JMenuItem();
        jMenuItem2 = new javax.swing.JMenuItem();

        javax.swing.GroupLayout jFrame1Layout = new
javax.swing.GroupLayout(jFrame1.getContentPane());
        jFrame1.getContentPane().setLayout(jFrame1Layout);
        jFrame1Layout.setHorizontalGroup(

jFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGap(0, 400, Short.MAX_VALUE)
        );
        jFrame1Layout.setVerticalGroup(
jFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                .addGap(0, 300, Short.MAX_VALUE)
        );

        jMenu1.setText("File");
        jMenuBar1.add(jMenu1);

        jMenu2.setText("Edit");
        jMenuBar1.add(jMenu2);


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CL
OSE);
        setTitle("Twitter");
        setBackground(new java.awt.Color(0, 51, 204));

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                .addGap(0, 196, Short.MAX_VALUE)
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                .addGap(0, 53, Short.MAX_VALUE)
        );

        jTextArea1.setColumns(20);
        jTextArea1.setRows(5);
        jScrollPane2.setViewportView(jTextArea1);

        jTextArea2.setColumns(20);
        jTextArea2.setRows(5);
        jScrollPane3.setViewportView(jTextArea2);

        loginButton.setFont(new java.awt.Font("Papyrus", 1,
14));
        loginButton.setText("Login");
        loginButton.addActionListener(new
java.awt.event.ActionListener() {
```

```
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                loginButtonActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGap(0, 405, Short.MAX_VALUE)
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGap(0, 0, Short.MAX_VALUE)
        );

        getStatusesButton.setFont(new java.awt.Font("Papyrus",
1, 14));
        getStatusesButton.setText("Get Status");
        getStatusesButton.setEnabled(false);
        getStatusesButton.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                getStatusesButtonActionPerformed(evt);
            }
        });

        jMenu3.setText("Add");


jMenuItem1.setAccelerator(javax.swing.KeyStroke.getKeyStroke(ja
va.awt.event.KeyEvent.VK_O, 0));
        jMenuItem1.setText("Offensive");
        jMenuItem1.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem1ActionPerformed(evt);
            }
        });
        jMenu3.add(jMenuItem1);
```

```
jMenuItem2.setAccelerator(javax.swing.KeyStroke.getKeyStroke(ja
va.awt.event.KeyEvent.VK_M, 0));
        jMenuItem2.setText("Medium");
        jMenuItem2.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem2ActionPerformed(evt);
            }
        });
        jMenu3.add(jMenuItem2);

        jMenuBar2.add(jMenu3);

        setJMenuBar(jMenuBar2);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addGap(482, 482, 482)
                .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
            .addComponent(jScrollPane2,
javax.swing.GroupLayout.DEFAULT_SIZE, 678, Short.MAX_VALUE)
            .addComponent(jScrollPane3,
javax.swing.GroupLayout.DEFAULT_SIZE, 678, Short.MAX_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addComponent(jPanel2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGap(273, 273, 273))
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(loginButton)
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.REL
ATED, 462, Short.MAX_VALUE)
                    .addComponent(getStatusesButton)
                    .addContainerGap())
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                    .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 203,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.REL
ATED)
                    .addComponent(jScrollPane3,
javax.swing.GroupLayout.PREFERRED_SIZE, 192,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(56, 56, 56)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                        .addComponent(loginButton)
                        .addComponent(getStatusesButton))
                    .addGap(18, 18, 18)
                    .addComponent(jPanel2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGap(52, 52, 52)
                    .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void
loginButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_loginButtonActionPerformed

    TwitterClient.login();
    getStatusesButton.setEnabled(true);
    loginButton.setEnabled(false);
         // TODO add your handling code here:
     }//GEN-LAST:event_loginButtonActionPerformed

public void Operation()
    {
        try
         {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            String sname;
            con = DriverManager.getConnection("jdbc:odbc:avs");
            /*Statement st= con.createStatement();
            ResultSet result=st.executeQuery("SELECT * FROM
offensive");
            while (result.next())
            {
               sname=result.getString("Flame");
               System.out.println(sname);
            }*/
         }
         catch(Exception e)
         {
             System.out.println("hello" + e.getMessage());
         }
    }

    private void
getStatusesButtonActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_getStatusesButtonActionPerformed
     String txt=this.offensiveText;
     ResultSet result;
     Statement st1;
     new Twitterwindow().Operation();

     TwitterClient client = new TwitterClient();

    client.initOAuth();
    Statuses response =
client.getFriendsTimeline(Statuses.class, null, null, null,
"10");
    response.getStatus().size();
    String s1= "";
    String text = "";
    for (StatusType st : response.getStatus()) {
        text += st.getUser().getName() + ":  " + st.getText() +
"\n";

        }
    String text2="";
    int i;
//"fuck yourself you m o th e r**fuck!!eR y o u ass}}hol#E ku
tT **&a";  //s1 for input

s1+=text;
String s2=s1.toLowerCase();  //s2 for lowercase

 int length=s2.length();  //length takes String length
char a1[]=s2.toCharArray();
char b[]=new char[length];

int j=0;

//removing extra symbols

for(i=0;i<length;i++)
{
if((a1[i]>=97)&&(a1[i]<=122))
b[j++]=a1[i];
}

String s3=new String(b);
//int length2=flaming.length;
//int length3=medium.length;
int as1=0,as2=0,as3=0,an=0;
//as1=as2=as3=an=0;
String chck;
//Agreesive Flaming
try
{
    st1= con.createStatement();
result=st1.executeQuery("SELECT * FROM offensive");
while(result.next())
{
    chck=result.getString("Flame");
j=-1;
j=s3.indexOf(chck);
if(j!=-1)
    {
        as3++;
        text2+="Aggresive Flaming Word   "+chck+"   found.\n";
    }
}


//Medium Flaming
```

```
result=st1.executeQuery("SELECT * FROM medium");
while(result.next())
{
    chck=result.getString("Flame");
j=-1;
j=s3.indexOf(chck);
if(j!=-1)
    {
        as2++;
        text2+="Medium Flmaing Word   "+chck+"   found.\n";
    }
}
}
catch(Exception e)
{

}
//Low-Level Flaming
char c[]=new char[length];
char a2[]=s1.toCharArray();
j=0;

for(i=0;i<length;i++)
{
if((a2[i]>=65)&&(a2[i]<=90))
    j++;

else if(a2[i]==32)
{
    if(j>2)
    as1++;
      j=0;
}
else
    while(a2[i]!=32)
    {
        i++;
        j=0;
        if(i>=length)
            break;
    }
}

j=0;
for(i=0;i<length;i++)
{
```

```
if((a2[i]>=65)&&(a2[i]<=90)||(a2[i]>=97)&&(a2[i]<=122)||(a2[i]>
=48)&&(a2[i]<=57))
{
if(j>=3)
    an++;
j=0;
}

else
    j++;

}
text2+="\nLow-Level Flaming1 is"+as1+"\nLow-Level Flaming2
is"+an+"\nMedium Level Flaming is"+as2+"\nAggressive Level
Flaming is"+as3;

         jTextArea1.setText(text);

          jTextArea2.setText(text2);
          draw d1= new draw();
         // jPanel2.add(d1, BorderLayout.CENTER);
    }//GEN-LAST:event_getStatusesButtonActionPerformed

    private void
jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jMenuItem1ActionPerformed
        // TODO add your handling code here:
        OffensiveDialog dialog=new OffensiveDialog();
        dialog.setTwitterwindow(this);
        dialog.setVisible(true);
        dialog.setSize(600, 200);

dialog.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    }//GEN-LAST:event_jMenuItem1ActionPerformed

    private void
jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jMenuItem2ActionPerformed
        // TODO add your handling code here:
        MedumDialog dialog=new MedumDialog();
        dialog.setTwitterwindow(this);
        dialog.setVisible(true);
        dialog.setSize(600, 200);

dialog.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    }//GEN-LAST:event_jMenuItem2ActionPerformed

    /**
    * @param args the command line arguments
    */

    public static void main(String args[]) {

         Twitterwindow tn=new Twitterwindow();
        tn.Operation();
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Twitterwindow().setVisible(true);

            }


        });


    }

    // Variables declaration - do not modify//GEN-
BEGIN:variables
    private javax.swing.JButton getStatusesButton;
    private javax.swing.JFrame jFrame1;
    private javax.swing.JMenu jMenu1;
    private javax.swing.JMenu jMenu2;
    private javax.swing.JMenu jMenu3;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JMenuBar jMenuBar2;
    private javax.swing.JMenuItem jMenuItem1;
    private javax.swing.JMenuItem jMenuItem2;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JScrollPane jScrollPane3;
    private javax.swing.JTextArea jTextArea1;
    private javax.swing.JTextArea jTextArea2;
    private javax.swing.JButton loginButton;
    // End of variables declaration//GEN-END:variables

 private String offensiveText;

 private String
 flaming[]={"suck","dick","pussy","asshole","motherfucker","sist
 erfucker","fucker","kutreya","rangbazi","dickface","bastdard"};
```

```
private String medium[]={"fuck",
"fck","fuk","fcuk","saala","sala","saale",
"kutta","kuttiya","sexy","sex"  };

    public String getOffensiveText() {
        return offensiveText;
    }

    public void setOffensiveText(String offensiveText) {
        this.offensiveText = offensiveText;
    }


}
```

Code For Twitter Client

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package avs;

import com.sun.jersey.api.client.Client;
import com.sun.jersey.api.client.UniformInterfaceException;
import com.sun.jersey.api.client.WebResource;
import com.sun.jersey.api.representation.Form;
import com.sun.jersey.oauth.client.OAuthClientFilter;
import com.sun.jersey.oauth.signature.OAuthParameters;
import com.sun.jersey.oauth.signature.OAuthSecrets;
import javax.ws.rs.core.MultivaluedMap;


/** Jersey REST client generated for REST resource:Twitter
OAuth [statuses/friends_timeline.{format}]<br>
 *  USAGE:<pre>
 *        TwitterClient client = new TwitterClient();
 *        Object response = client.XXX(...);
 *        // do whatever with response
 *        client.close();
 *  </pre>
 * @author Vishal
 */
public class TwitterClient {
    private WebResource webResource;
    private Client client;
    private static final String BASE_URI =
"http://twitter.com";
    private static final String OAUTH_BASE_URL =
"http://twitter.com/oauth";
    /**
     * Please, specify the consumer_key string obtained from
service API pages
     */
    private static final String CONSUMER_KEY =
"veW1PwOBaJ2i8DG93vMYw";
    /**
     * Please, specify the consumer_secret string obtained from
service API pages
     */
    private static final String CONSUMER_SECRET =
"U4nv95LtL2rAUTHSveGVGeGrPHteJ7rGFUOaES3a98";
    private OAuthParameters oauth_params;
    private OAuthSecrets oauth_secrets;
    private OAuthClientFilter oauth_filter;

    public TwitterClient() {
        com.sun.jersey.api.client.config.ClientConfig config =
new com.sun.jersey.api.client.config.DefaultClientConfig();
        client = Client.create(config);
        String resourcePath = "statuses";

        webResource =
client.resource(BASE_URI).path(resourcePath);
    }

   /* public void setResourcePath(String format) {
        String resourcePath =
java.text.MessageFormat.format("statuses/friends_timeline.{0}",
new Object[]{format});
        webResource =
client.resource(BASE_URI).path(resourcePath);
    */


    /**
     * @param responseType Class representing the response
     * @param since query parameter
     * @param since_id query parameter
     * @param page query parameter
     * @param count query parameter
     * @return response object (instance of responseType class)
     */
    public <T> T getFriendsTimeline(Class<T> responseType,
String since, String since_id, String page, String count)
throws UniformInterfaceException {
        String[] queryParamNames = new String[]{"since",
"since_id", "page", "count"};
        String[] queryParamValues = new String[]{since,
since_id, page, count};
        return
webResource.path("friends_timeline.xml").queryParams(getQueryOr
FormParams(queryParamNames,
queryParamValues)).accept(javax.ws.rs.core.MediaType.TEXT_XML).
get(responseType);
    }

    private MultivaluedMap getQueryOrFormParams(String[]
paramNames, String[] paramValues) {
        MultivaluedMap<String, String> qParams = new
com.sun.jersey.api.representation.Form();
        for (int i = 0; i < paramNames.length; i++) {
            if (paramValues[i] != null) {
                qParams.add(paramNames[i], paramValues[i]);
            }
        }
        return qParams;
    }

    public void close() {
        client.destroy();
    }

    /**
     * You need to call this method at the beginning to
authorize the application to work with user data.
     * The method obtains the OAuth access token string, that
is appended to each API request later.
     */
    public static void login() throws UniformInterfaceException
{
        Form requestTokenResponse = getOAuthRequestToken();
        String oauth_verifier =
authorizeConsumer(requestTokenResponse);
        Form accessTokenResponse =
getOAuthAccessToken(requestTokenResponse, oauth_verifier);
        java.util.prefs.Preferences prefs =
org.openide.util.NbPreferences.forModule(TwitterClient.class);
        prefs.put("oauth_token",
accessTokenResponse.getFirst("oauth_token"));
```

```
        prefs.put("oauth_token_secret",
accessTokenResponse.getFirst("oauth_token_secret"));
    }

    public static void logout() {
        java.util.prefs.Preferences prefs =
org.openide.util.NbPreferences.forModule(TwitterClient.class);
        prefs.remove("oauth_token");
        prefs.remove("oauth_token_secret");
    }

    private static Form getOAuthRequestToken() throws
UniformInterfaceException {
        Client reqTokenClient = new Client();
        WebResource resource =
reqTokenClient.resource(OAUTH_BASE_URL).path("request_token");
        OAuthParameters o_params = new
OAuthParameters().consumerKey(CONSUMER_KEY).signatureMethod(com
.sun.jersey.oauth.signature.HMAC_SHA1.NAME).version("1.0").nonc
e().timestamp();
        OAuthSecrets o_secrets = new
OAuthSecrets().consumerSecret(CONSUMER_SECRET);
        OAuthClientFilter o_filter = new
OAuthClientFilter(reqTokenClient.getProviders(), o_params,
o_secrets);
        resource.addFilter(o_filter);
        return resource.get(Form.class);
    }

    private static Form getOAuthAccessToken(Form
requestTokenResponse, String oauth_verifier) throws
UniformInterfaceException {
        Client accessTokenClient = new Client();
        WebResource resource =
accessTokenClient.resource(OAUTH_BASE_URL).path("access_token")
;
        OAuthParameters o_params = new
OAuthParameters().consumerKey(CONSUMER_KEY).token(requestTokenR
esponse.getFirst("oauth_token")).signatureMethod(com.sun.jersey
.oauth.signature.HMAC_SHA1.NAME).version("1.0").nonce().timesta
mp().verifier(oauth_verifier);
        OAuthSecrets o_secrets = new
OAuthSecrets().consumerSecret(CONSUMER_SECRET).tokenSecret(requ
estTokenResponse.getFirst("oauth_token_secret"));
        OAuthClientFilter o_filter = new
OAuthClientFilter(accessTokenClient.getProviders(), o_params,
o_secrets);
        resource.addFilter(o_filter);
        return resource.get(Form.class);
    }

    /**
     * The method sets the OAuth parameters for webResource.
     * The method needs to be called after login() method, or
when the webResource path is changed
     */
    public void initOAuth() {
        java.util.prefs.Preferences prefs =
org.openide.util.NbPreferences.forModule(this.getClass());
        String oauth_token = prefs.get("oauth_token", null);
        String oauth_token_secret =
prefs.get("oauth_token_secret", null);
        if (oauth_token == null || oauth_token_secret == null)
{
            org.openide.DialogDisplayer.getDefault().notify(new
org.openide.NotifyDescriptor.Message("You have to call the
login() method first to authorize the application to access
user data."));
        } else {
            oauth_params = new
OAuthParameters().consumerKey(CONSUMER_KEY).token(oauth_token).
signatureMethod(com.sun.jersey.oauth.signature.HMAC_SHA1.NAME).
version("1.0").nonce().timestamp();
            oauth_secrets = new
OAuthSecrets().consumerSecret(CONSUMER_SECRET).tokenSecret(oaut
h_token_secret);
            oauth_filter = new
OAuthClientFilter(client.getProviders(), oauth_params,
oauth_secrets);
            webResource.addFilter(oauth_filter);
        }
    }

    /**
     * The method increases OAuth nonce and timestamp
parameters to make each request unique.
     * The method should be called when repetitive requests are
sent to service API provider:
     * <pre>
     * client.initOauth();
     * client.getXXX(...);
     * client.makeOAuthRequestUnique();
     * client.getYYY(...);
     * client.makeOAuthRequestUnique();
     * client.getZZZ(...);
     * </pre>
    */
    public void makeOAuthRequestUnique() {
        if (oauth_params != null) {
            oauth_params.nonce().timestamp();
        }
    }

    private static java.lang.String authorizeConsumer(Form
requestTokenResponse) {
        class DialogPanel extends javax.swing.JPanel {

            private javax.swing.JTextField verifierTextField;

            DialogPanel(final String url) {
                java.awt.GridBagConstraints gridBagConstraints;
                javax.swing.JLabel topLabel = new
javax.swing.JLabel();
                verifierTextField = new
javax.swing.JTextField();
                javax.swing.JLabel verifierLabel = new
javax.swing.JLabel();
                javax.swing.JLabel bottomLabel = new
javax.swing.JLabel();
                javax.swing.JButton urlButton = new
javax.swing.JButton();
                setLayout(new java.awt.GridBagLayout());
                topLabel.setText("Click the URL link below to
open the browser, and authorize the application to access your
data:"); // NOI18N
                gridBagConstraints = new
java.awt.GridBagConstraints();
                gridBagConstraints.gridwidth = 2;
                gridBagConstraints.anchor =
java.awt.GridBagConstraints.WEST;
                add(topLabel, gridBagConstraints);
                urlButton.setForeground(java.awt.Color.BLUE);
                urlButton.setBorderPainted(false);
                urlButton.setContentAreaFilled(false);
                String urlText = url.toString();
                String text = "<html><b><u>" + urlText +
"</u></b></html>";
                urlButton.setText(text);

urlButton.setHorizontalAlignment(javax.swing.SwingConstants.LEF
T);
                urlButton.addActionListener(new
java.awt.event.ActionListener() {
```

```
                    @Override
                    public void
actionPerformed(java.awt.event.ActionEvent e) {
                        try {

org.openide.awt.HtmlBrowser.URLDisplayer.getDefault().showURLEx
ternal(new java.net.URL(url));
                        } catch (java.net.MalformedURLException
ex) {

java.util.logging.Logger.getLogger(DialogPanel.class.getName())
.log(java.util.logging.Level.WARNING, "incorrect URL string",
ex);
                        }
                    }
                });
                gridBagConstraints = new
java.awt.GridBagConstraints();
                gridBagConstraints.gridx = 0;
                gridBagConstraints.gridy = 1;
                gridBagConstraints.gridwidth = 2;
                gridBagConstraints.fill =
java.awt.GridBagConstraints.HORIZONTAL;
                gridBagConstraints.anchor =
java.awt.GridBagConstraints.WEST;
                add(urlButton, gridBagConstraints);
                verifierLabel.setText("Type oauth_verifier
string, taken from authorization(callback) page: "); // NOI18N
                gridBagConstraints = new
java.awt.GridBagConstraints();
                gridBagConstraints.gridx = 0;
                gridBagConstraints.gridy = 2;
                gridBagConstraints.anchor =
java.awt.GridBagConstraints.WEST;
                gridBagConstraints.insets = new
java.awt.Insets(10, 0, 10, 0);
                add(verifierLabel, gridBagConstraints);
                verifierTextField.setText(""); // NOI18N
                gridBagConstraints = new
java.awt.GridBagConstraints();
                gridBagConstraints.gridx = 1;
                gridBagConstraints.gridy = 2;
                gridBagConstraints.fill =
java.awt.GridBagConstraints.HORIZONTAL;
                gridBagConstraints.weightx = 1.0;
                gridBagConstraints.insets = new
java.awt.Insets(10, 0, 10, 0);
                add(verifierTextField, gridBagConstraints);
                bottomLabel.setText("After you allow the
application to access your data, press OK.");
                gridBagConstraints = new
java.awt.GridBagConstraints();
                gridBagConstraints.gridx = 0;
                gridBagConstraints.gridy = 3;
                gridBagConstraints.gridwidth = 2;
                gridBagConstraints.anchor =
java.awt.GridBagConstraints.WEST;
                add(bottomLabel, gridBagConstraints);
            }

            public String getVerifier() {
                return verifierTextField.getText().trim();
            }
        }
        String oauth_verifier = null;
        String loginUrl =
"http://twitter.com/oauth/authorize?oauth_token=" +
requestTokenResponse.getFirst("oauth_token");
        DialogPanel dialogPanel = new DialogPanel(loginUrl);
        org.openide.DialogDescriptor dd = new
org.openide.DialogDescriptor(dialogPanel, "OAuth Authentication
Dialog");
        org.openide.DialogDisplayer.getDefault().notify(dd);
        if (dd.getValue() ==
org.openide.DialogDescriptor.OK_OPTION) {
            oauth_verifier = dialogPanel.getVerifier();
        }
        return oauth_verifier;
    }

}
```

Code For Offensive Dialog

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package avs;
import java.sql.*;
/**
 *
 *
 */
public class OffensiveDialog extends javax.swing.JFrame {
Connection con;
```

```
    /**
     * Creates new form OffensiveDialog
     */
    public OffensiveDialog() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to
initialize the form.
     * WARNING: Do NOT modify this code. The content of this
method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        jButton1 = new javax.swing.JButton();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CL
OSE);

        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 14)); //
NOI18N

jLabel1.setText(org.openide.util.NbBundle.getMessage(OffensiveD
ialog.class, "OffensiveDialog.jLabel1.text")); // NOI18N

        jTextArea1.setColumns(20);
        jTextArea1.setRows(5);
        jScrollPane1.setViewportView(jTextArea1);


jButton1.setText(org.openide.util.NbBundle.getMessage(Offensive
Dialog.class, "OffensiveDialog.jButton1.text")); // NOI18N
        jButton1.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(18, 18, 18)
                .addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNR
ELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLa
yout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(jButton1)
                        .addContainerGap())

.addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 251, Short.MAX_VALUE)
                        .addGap(10, 10, 10))))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLa
yout.Alignment.LEADING)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 31,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 35,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(30, 30, 30)
                        .addComponent(jButton1)))
                .addContainerGap(28, Short.MAX_VALUE))
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_jButton1ActionPerformed
        // TODO add your handling code here:
        //twitterwindow.setOffensiveText(jTextArea1.getText());
        String txt=jTextArea1.getText();
        System.out.println(txt);
     try
     {
         Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:avs");
     Statement st1= con.createStatement();
     ResultSet result=st1.executeQuery("SELECT * FROM
offensive");
     String chck;
     int flag=0;
     while(result.next())
     {
         chck=result.getString("Flame");
         if(chck.equals(txt))
         {
             flag=1;
```

```
                break;
            }
        }
        if(flag==0)

        {
        st1.executeUpdate("insert into offensive
values('"+txt+"')");
        }
        }
        catch(Exception e)
        {
            System.out.println(e.getMessage());
        }
    }//GEN-LAST:event_jButton1ActionPerformed

    /**
     * @param args the command line arguments
     */

    // Variables declaration - do not modify//GEN-
BEGIN:variables
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextArea jTextArea1;
    // End of variables declaration//GEN-END:variables
   private Twitterwindow twitterwindow;

    public void setTwitterwindow(Twitterwindow twitterwindow) {
        this.twitterwindow = twitterwindow;
    }

}
```

Code For No internet Window

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

/*
 * NoInternet.java
 *
 * Created on Sep 16, 2011, 3:25:48 AM
 */
```

```
package avs;

/**
 *
 * @author Vishal
 */
public class NoInternet extends javax.swing.JFrame {

    /** Creates new form NoInternet */
    public NoInternet() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this
method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jButton1 = new javax.swing.JButton();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CL
OSE);

setTitle(org.openide.util.NbBundle.getMessage(NoInternet.class,
"NoInternet.title")); // NOI18N

        jLabel1.setFont(new java.awt.Font("Papyrus", 1, 24));

jLabel1.setText(org.openide.util.NbBundle.getMessage(NoInternet
.class, "NoInternet.jLabel1.text")); // NOI18N


jButton1.setText(org.openide.util.NbBundle.getMessage(NoInterne
t.class, "NoInternet.jButton1.text")); // NOI18N
        jButton1.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
```

```
                jButton1ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLa
yout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(jLabel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addContainerGap())

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                        .addComponent(jButton1)
                        .addGap(314, 314, 314))))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1,
javax.swing.GroupLayout.DEFAULT_SIZE, 48, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.REL
ATED, 14, Short.MAX_VALUE)
                .addComponent(jButton1)
                .addGap(38, 38, 38))
        );


jLabel1.getAccessibleContext().setAccessibleName(org.openide.ut
il.NbBundle.getMessage(NoInternet.class,
"NoInternet.jLabel1.AccessibleContext.accessibleName")); //
NOI18N
```

```
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_jButton1ActionPerformed
    System.exit(0);        // TODO add your handling code here:
    }//GEN-LAST:event_jButton1ActionPerformed

    /**
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new NoInternet().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-
BEGIN:variables
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    // End of variables declaration//GEN-END:variables
```

```
}
```

Code for medium dialog box

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

/*
 * MedumDialog.java
 *
 * Created on Mar 17, 2012, 11:49:51 PM
 */

package avs;
import java.sql.*;
/**
 *
 * @author Vishal
 */
public class MedumDialog extends javax.swing.JFrame {

    /** Creates new form MedumDialog */
    Connection con;
    public MedumDialog() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this
method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        jButton1 = new javax.swing.JButton();
```

```
setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CL
OSE);

        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 14));

jLabel1.setText(org.openide.util.NbBundle.getMessage(MedumDialo
g.class, "MedumDialog.jLabel1.text")); // NOI18N

        jTextArea1.setColumns(20);
        jTextArea1.setRows(5);
        jScrollPane1.setViewportView(jTextArea1);

        jButton1.setFont(new java.awt.Font("Tempus Sans ITC",
1, 14)); // NOI18N

jButton1.setText(org.openide.util.NbBundle.getMessage(MedumDial
og.class, "MedumDialog.jButton1.text")); // NOI18N
        jButton1.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLa
yout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()
                        .addGap(18, 18, 18)
                        .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 129,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 225,
javax.swing.GroupLayout.PREFERRED_SIZE))
.addGroup(jPanel1Layout.createSequentialGroup()
                        .addGap(133, 133, 133)
                        .addComponent(jButton1,
javax.swing.GroupLayout.PREFERRED_SIZE, 123,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(35, 35, 35)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLa
yout.Alignment.LEADING)
                    .addComponent(jScrollPane1, 0, 0,
Short.MAX_VALUE)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 36,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.REL
ATED)
                .addComponent(jButton1,
javax.swing.GroupLayout.PREFERRED_SIZE, 45,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(47, 47, 47))
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
```

```
            .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_jButton1ActionPerformed
        // TODO add your handling code here:
         String txt=jTextArea1.getText();
        System.out.println(txt);
     try
     {
         Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:avs");
     Statement st1= con.createStatement();
     ResultSet result=st1.executeQuery("SELECT * FROM medium");
     String chck;
     int flag=0;
     while(result.next())
     {
         chck=result.getString("Flame");
         if(chck.equals(txt))
         {
             flag=1;
             break;
         }
     }
     if(flag==0)

     {
     st1.executeUpdate("insert into medium values('"+txt+"')");
     }
     }
     catch(Exception e)
     {
         System.out.println(e.getMessage());
     }
    }//GEN-LAST:event_jButton1ActionPerformed

    /**
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new MedumDialog().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-
BEGIN:variables
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextArea jTextArea1;
    // End of variables declaration//GEN-END:variables
 private Twitterwindow twitterwindow;

    public void setTwitterwindow(Twitterwindow twitterwindow) {
        this.twitterwindow = twitterwindow;
    }
}
```

Code For Status Type

```
//
// This file was generated by the JavaTM Architecture for XML
Binding(JAXB) Reference Implementation, vhudson-jaxb-ri-2.2-147
// See <a
href="http://java.sun.com/xml/jaxb">http://java.sun.com/xml/jax
b</a>
// Any modifications to this file will be lost upon
recompilation of the source schema.
// Generated on: 2011.09.08 at 05:26:40 AM IST
//


package avs;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlType;


/**
 * <p>Java class for statusType complex type.
 *
 * <p>The following schema fragment specifies the expected
content contained within this class.
 *
 * <pre>
 * &lt;complexType name="statusType">
 *   &lt;complexContent>
 *     &lt;restriction
base="{http://www.w3.org/2001/XMLSchema}anyType">
 *       &lt;all>
 *         &lt;element name="created_at"
type="{http://www.w3.org/2001/XMLSchema}string" minOccurs="0"/>
 *         &lt;element name="id"
type="{http://www.w3.org/2001/XMLSchema}string" minOccurs="0"/>
 *         &lt;element name="text"
type="{http://www.w3.org/2001/XMLSchema}string" minOccurs="0"/>
 *         &lt;element name="source"
type="{http://www.w3.org/2001/XMLSchema}string" minOccurs="0"/>
 *         &lt;element name="truncated"
type="{http://www.w3.org/2001/XMLSchema}boolean"
minOccurs="0"/>
 *         &lt;element name="in_reply_to_user_id"
type="{http://www.w3.org/2001/XMLSchema}string" minOccurs="0"/>
 *         &lt;element name="in_reply_to_status_id"
type="{http://www.w3.org/2001/XMLSchema}string" minOccurs="0"/>
 *         &lt;element name="favorited"
type="{http://www.w3.org/2001/XMLSchema}string" minOccurs="0"/>
 *         &lt;element name="user" type="{}userType"
minOccurs="0"/>
 *       &lt;/all>
 *     &lt;/restriction>
 *   &lt;/complexContent>
 * &lt;/complexType>
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "statusType", propOrder = {

})
public class StatusType {

    @XmlElement(name = "created_at")
    protected String createdAt;
    protected String id;
    protected String text;
    protected String source;
    protected Boolean truncated;
```

```
@XmlElement(name = "in_reply_to_user_id")
protected String inReplyToUserId;
@XmlElement(name = "in_reply_to_status_id")
protected String inReplyToStatusId;
protected String favorited;
protected UserType user;

/**
 * Gets the value of the createdAt property.
 *
 * @return
 *     possible object is
 *     {@link String }
 *
 */
public String getCreatedAt() {
    return createdAt;
}

/**
 * Sets the value of the createdAt property.
 *
 * @param value
 *     allowed object is
 *     {@link String }
 *
 */
public void setCreatedAt(String value) {
    this.createdAt = value;
}

/**
 * Gets the value of the id property.
 *
 * @return
 *     possible object is
 *     {@link String }
 *
 */
public String getId() {
    return id;
}

/**
 * Sets the value of the id property.
 *
 * @param value
 *     allowed object is
```

```
     *     {@link String }
     *
     */
    public void setId(String value) {
        this.id = value;
    }

    /**
     * Gets the value of the text property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getText() {
        return text;
    }

    /**
     * Sets the value of the text property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setText(String value) {
        this.text = value;
    }

    /**
     * Gets the value of the source property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getSource() {
        return source;
    }

    /**
     * Sets the value of the source property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setSource(String value) {
        this.source = value;
    }

    /**
     * Gets the value of the truncated property.
     *
     * @return
     *     possible object is
     *     {@link Boolean }
     *
     */
    public Boolean isTruncated() {
        return truncated;
    }

    /**
     * Sets the value of the truncated property.
     *
     * @param value
     *     allowed object is
     *     {@link Boolean }
     *
     */
    public void setTruncated(Boolean value) {
        this.truncated = value;
    }

    /**
     * Gets the value of the inReplyToUserId property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getInReplyToUserId() {
        return inReplyToUserId;
    }

    /**
     * Sets the value of the inReplyToUserId property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setInReplyToUserId(String value) {
        this.inReplyToUserId = value;
    }

    /**
     * Gets the value of the inReplyToStatusId property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getInReplyToStatusId() {
        return inReplyToStatusId;
    }

    /**
     * Sets the value of the inReplyToStatusId property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setInReplyToStatusId(String value) {
        this.inReplyToStatusId = value;
    }

    /**
     * Gets the value of the favorited property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getFavorited() {
        return favorited;
    }

    /**
     * Sets the value of the favorited property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setFavorited(String value) {
        this.favorited = value;
    }

    /**
     * Gets the value of the user property.
     *
     * @return
     *     possible object is
     *     {@link UserType }
     *
     */
    public UserType getUser() {
        return user;
    }

    /**
     * Sets the value of the user property.
     *
     * @param value
     *     allowed object is
     *     {@link UserType }
     *
     */
    public void setUser(UserType value) {
        this.user = value;
    }

}
```

Code for Status Window

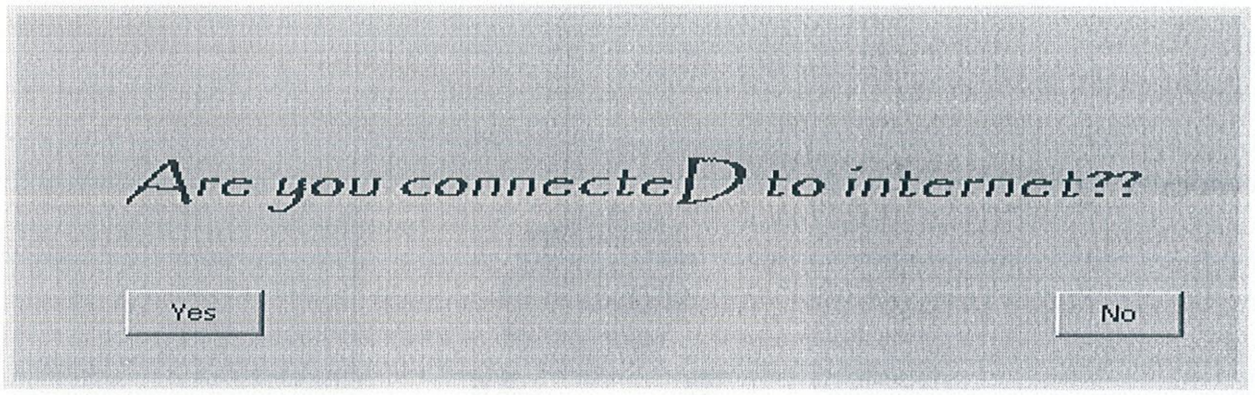
```
//
// This file was generated by the JavaTM Architecture for XML
Binding(JAXB) Reference Implementation, vhudson-jaxb-ri-2.2-147
// See <a
href="http://java.sun.com/xml/jaxb">http://java.sun.com/xml/jax
b</a>
// Any modifications to this file will be lost upon
recompilation of the source schema.
// Generated on: 2011.09.08 at 05:26:40 AM IST
//


package avs;
```

```
import java.util.ArrayList;
import java.util.List;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlType;


/**
 * <p>Java class for anonymous complex type.
 *
 * <p>The following schema fragment specifies the expected
content contained within this class.
 *
 * <pre>
 * &lt;complexType>
 *   &lt;complexContent>
 *     &lt;restriction
base="{http://www.w3.org/2001/XMLSchema}anyType">
 *       &lt;sequence>
 *         &lt;element name="status" type="{}statusType"
maxOccurs="unbounded" minOccurs="0"/>
 *       &lt;/sequence>
 *       &lt;attribute name="type"
type="{http://www.w3.org/2001/XMLSchema}string" />
 *     &lt;/restriction>
 *   &lt;/complexContent>
 * &lt;/complexType>
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "", propOrder = {
    "status"
})
@XmlRootElement(name = "statuses")
public class Statuses {

    protected List<StatusType> status;
    @XmlAttribute(name = "type")
    protected String type;

    /**
     * Gets the value of the status property.
     *
     * <p>
     * This accessor method returns a reference to the live
list,
     * not a snapshot. Therefore any modification you make to
the
     * returned list will be present inside the JAXB object.
     * This is why there is not a <CODE>set</CODE> method for
the status property.
     *
     * <p>
     * For example, to add a new item, do as follows:
     * <pre>
     *    getStatus().add(newItem);
     * </pre>
     *
     *
     * <p>
     * Objects of the following type(s) are allowed in the list
     * {@link StatusType }
     *
     *
     */
    public List<StatusType> getStatus() {
        if (status == null) {
            status = new ArrayList<StatusType>();
        }
        return this.status;
    }

    /**
     * Gets the value of the type property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getType() {
        return type;
    }

    /**
     * Sets the value of the type property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
```

```
         */
        public void setType(String value) {
            this.type = value;
        }

    }
```

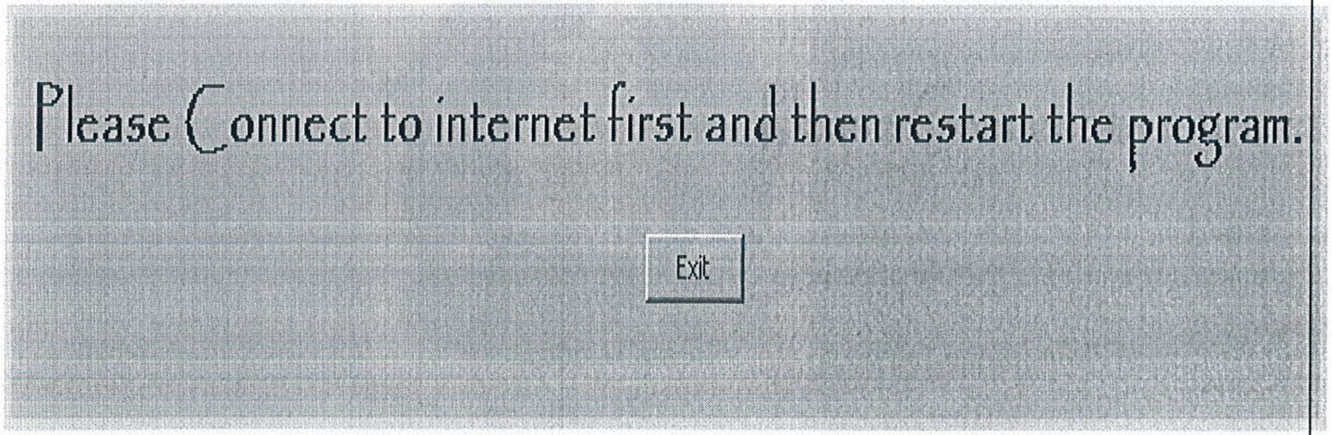
Results

Starting Window



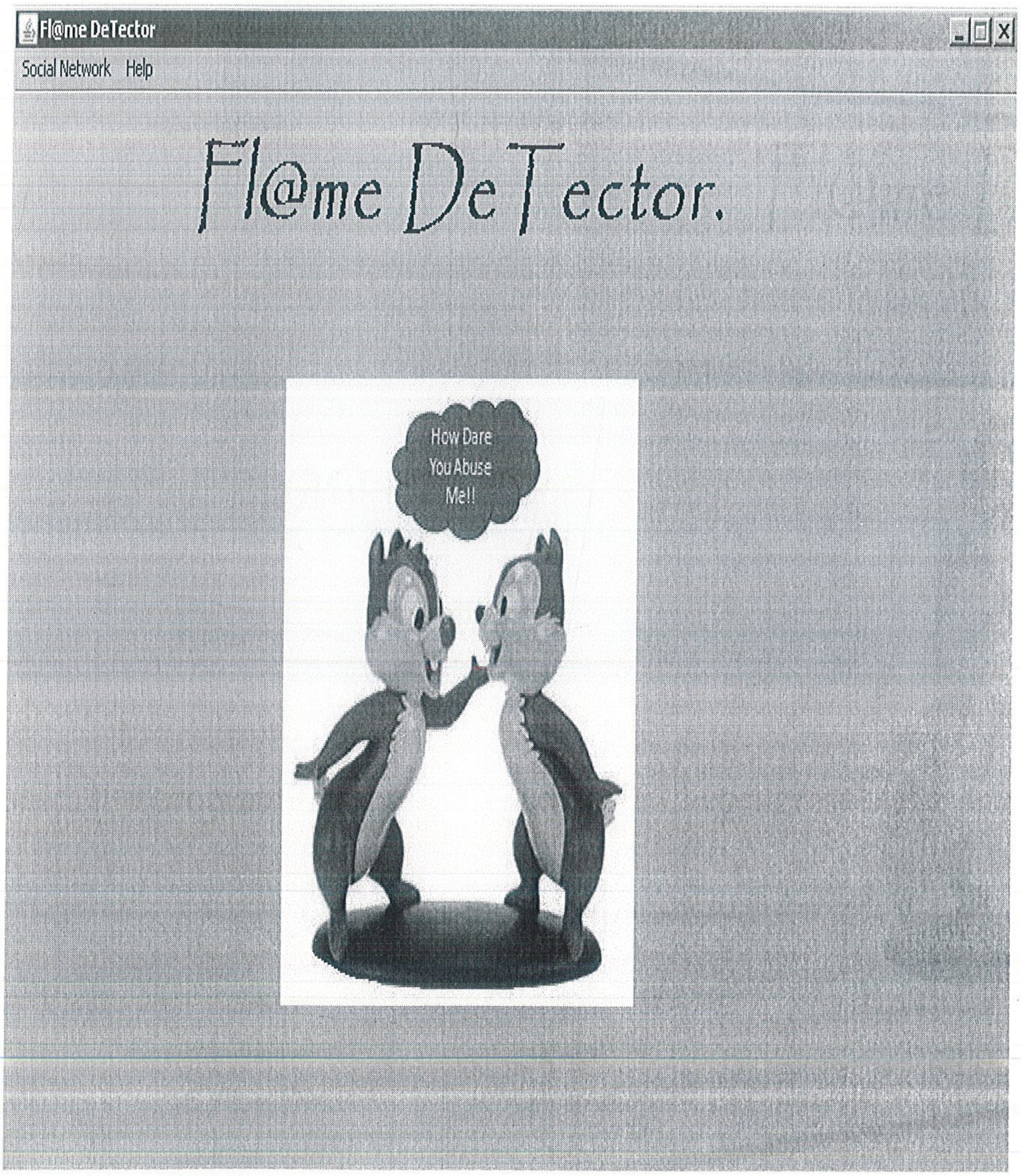
Are you connecteD to internet??

No Internet



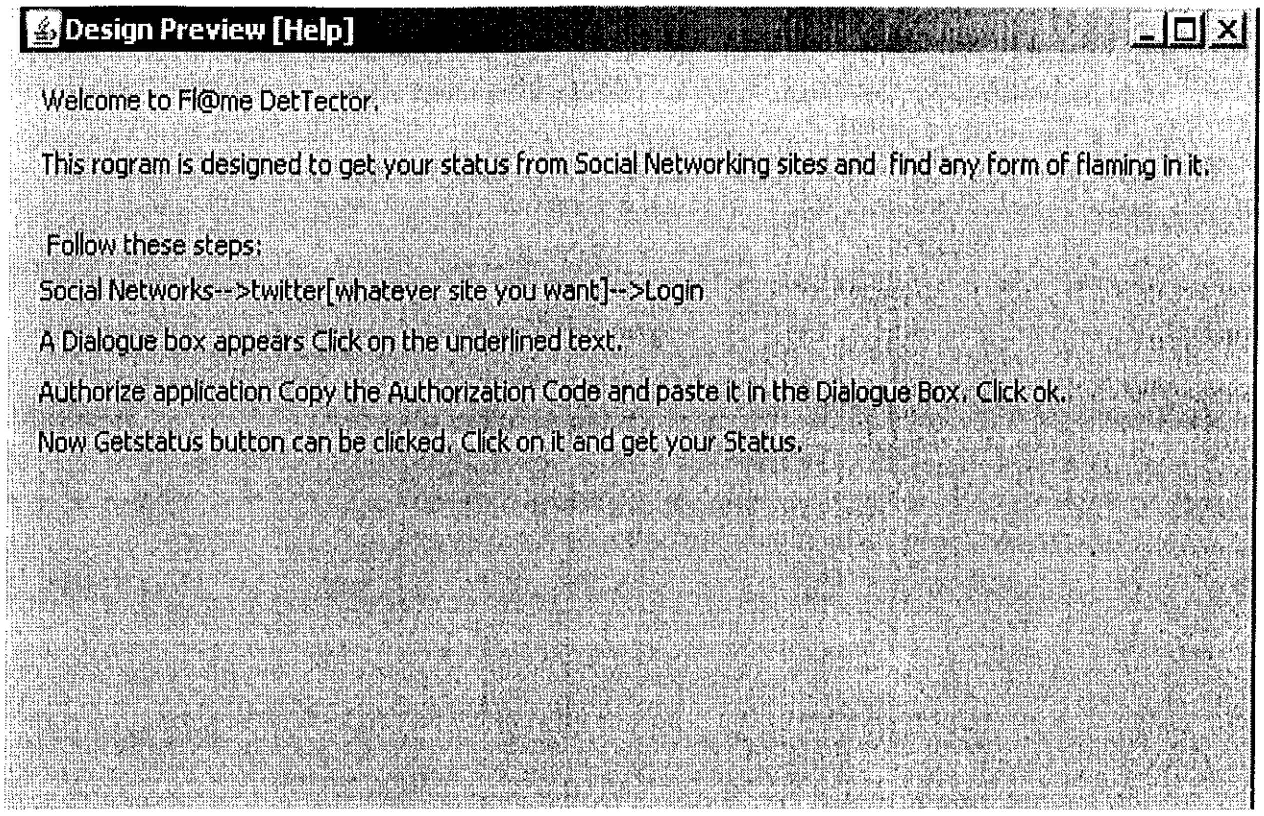
Please Connect to internet first and then restart the program.

Main Window


Fl@me DeTector
Social Network
Help
Fl@me DeTector.
How Dare You Abuse Me!!

Help Window

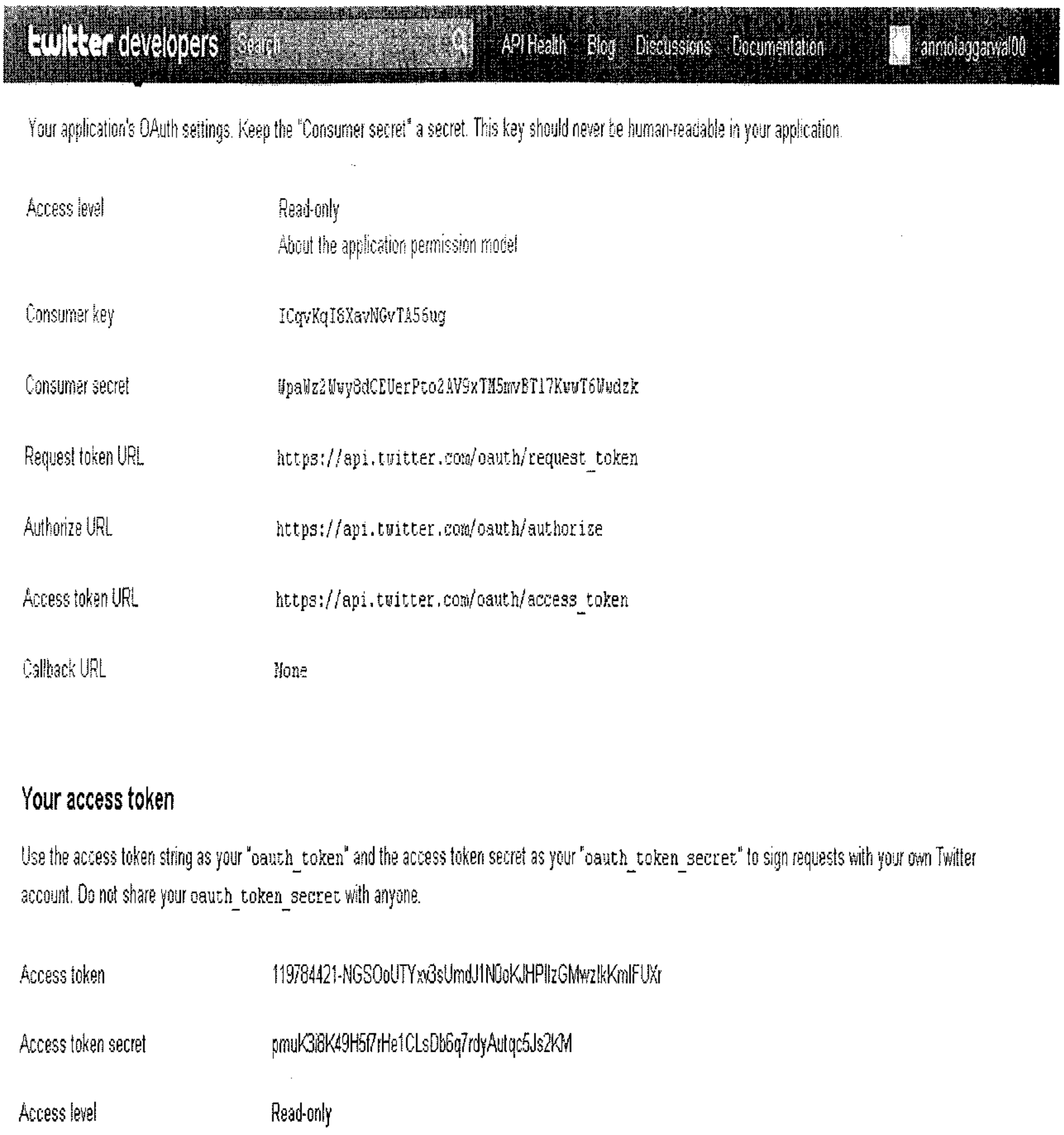
The help window comes when you click on the help menu in the main window



Welcome to Fl@me DetTector.

This rogram is designed to get your status from Social Networking sites and find any form of flaming in it.

Follow these steps:
Social Networks-->twitter[whatever site you want]-->Login
A Dialogue box appears Click on the underlined text.
Authorize application Copy the Authorization Code and paste it in the Dialogue Box. Click ok.
Now Getstatus button can be clicked. Click on it and get your Status.



Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

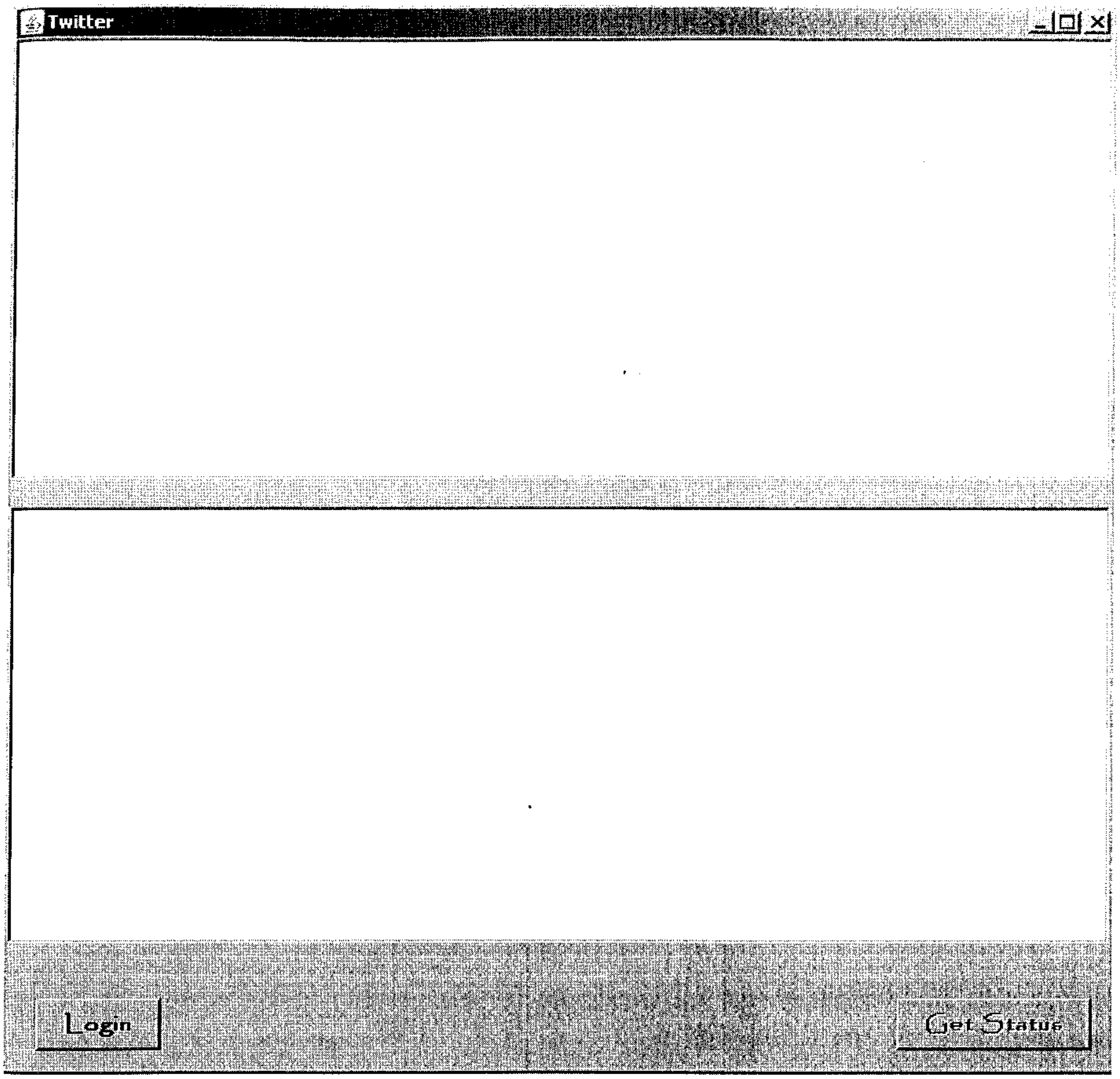
Access level	Read-only About the application permission model
Consumer key	`ICqvKqI8XavNGvTA56ug`
Consumer secret	`WpaWz2Wwy8dCEUerPto2AV9xTM5mvBT17KwwT6Wwdzk`
Request token URL	`https://api.twitter.com/oauth/request_token`
Authorize URL	`https://api.twitter.com/oauth/authorize`
Access token URL	`https://api.twitter.com/oauth/access_token`
Callback URL	`None`

Your access token

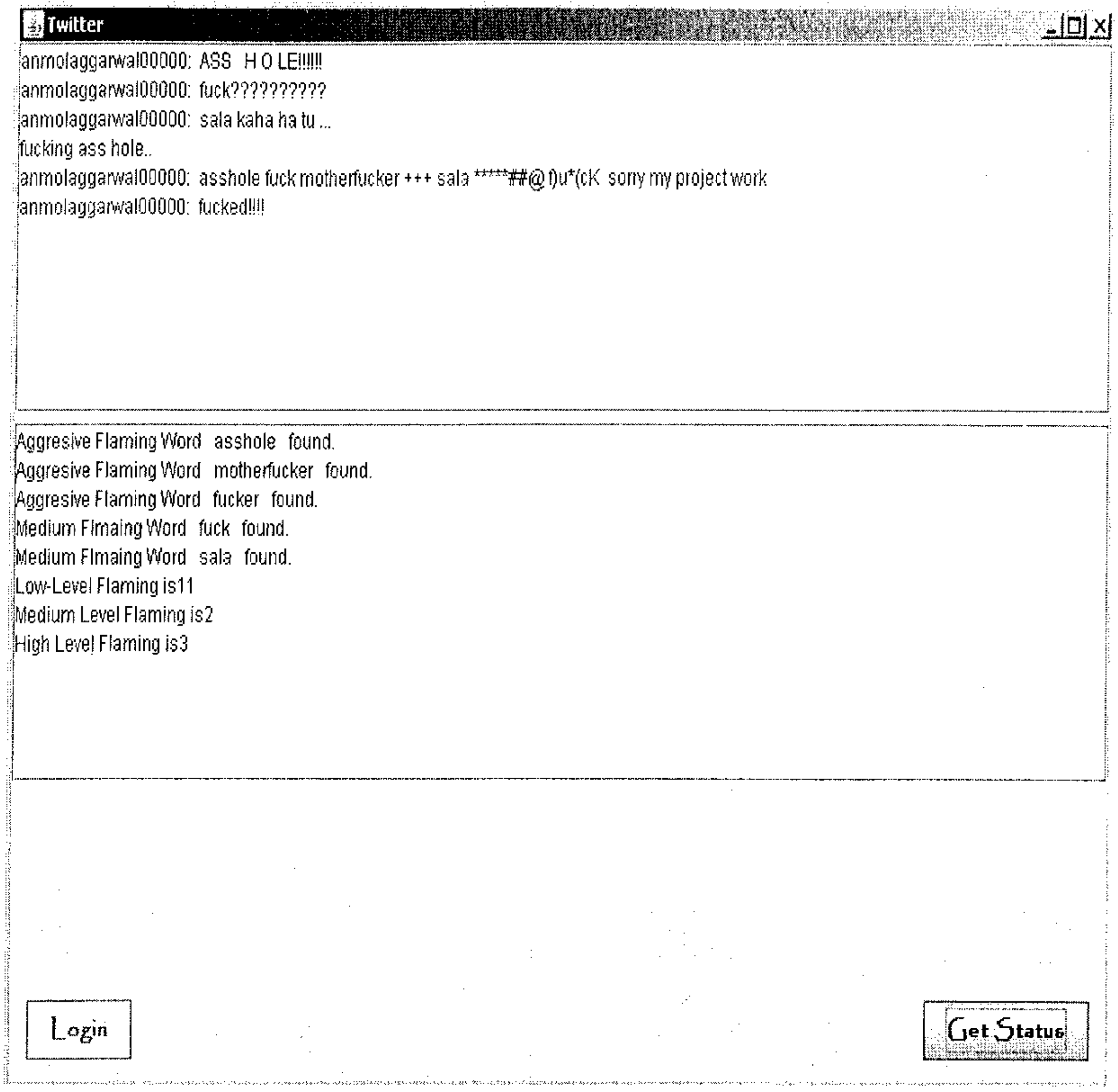
Use the access token string as your "`oauth_token`" and the access token secret as your "`oauth_token_secret`" to sign requests with your own Twitter account. Do not share your `oauth_token_secret` with anyone.

Access token	119784421-NGSOoUTYxv3sUmdJ1N0oKJHPllzGMwzlkKmlFUXr
Access token secret	pmuK3i8K49H5f7rHe1CLsDb6q7rdyAutqc5Js2KM
Access level	Read-only

Twitter Window



Result Window



```
anmolaggarwal00000:  ASS   H O LE!!!!!!
anmolaggarwal00000:  fuck??????????
anmolaggarwal00000:  sala kaha ha tu ...
fucking ass hole..
anmolaggarwal00000:  asshole fuck motherfucker +++ sala *****##@ f)u*(cK  sorry my project work
anmolaggarwal00000:  fucked!!!!
```

```
Aggresive Flaming Word   asshole   found.
Aggresive Flaming Word   motherfucker   found.
Aggresive Flaming Word   fucker   found.
Medium Flmaing Word   fuck   found.
Medium Flmaing Word   sala   found.
Low-Level Flaming is11
Medium Level Flaming is2
High Level Flaming is3
```

CONCLUSION

We have successfully completed the task of developing a software tool that detects flaming on a SNS(Social Networking Site). We choose Twitter as our SNS and picked up statuses from the site and then with the help of our software tool we analyzed the statuses and detected the level of flaming in each status. In our software tool we have classified the flaming as low level, medium level and high level. We have made our tool user friendly by allowing user to add words according to his/her will, which was achieved by making a database. Making our software user friendly also made our software lingually diverse.

Bibliography

1. MCKENNA, K., & BARGH, J.A. (2000). Plan 9 from cyberspace: the implications of the Internet for personality and social psychology. Personality and Social Psychology Review, 4, 57–75.
2. MOORE, D.A., KURTZBERG, T.R., THOMPSON, L.L., & MORRIS, M.W. (1999). Long and short routes to success in electronically mediated negotiations: Group affiliations and good vibrations. Organizational Behavior and Human Decision Processes, 77 (1), 22-43.
3. Dr. Nitin, Ankush Bansal, Chirag Bogra, Jodhbir Singh Sehmiand Siddhartha .M. SarmaCollege of Information Science and Technology, Peter Kiewit Institute, University of Nebraska at Omaha, Nebraska United States of America
4. www.netbeans.org
5. www.webservices.com
6. www.twitter.com