*A project report on*

**AN OPERATING SYSTEM USING JAVA AND NETBEANS**

Submitted in partial fulfillment of the requirements for the degree of

**Master of Computer Application**

By

**UMANG PINCHA(18MCA0016)**

**ARPIT MISHRA(18MCA0039)**

**ABHISHEK RUSTAGI (18MCA0083)**

*Under the Guidance of*

**KUMAR P.J  
Assistant professor (Senior), SITE**



**SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING**

April, 2019

*A project report on*

**AN OPERATING SYSTEM USING JAVA AND NETBEANS**

Submitted in partial fulfillment of the requirements for the degree of

**Master of Computer Application**

By

**UMANG PINCHA(18MCA0016)**

**ARPIT MISHRA(18MCA0039)**

**ABHISHEK RUSTAGI (18MCA0083)**

*Under the Guidance of*

**KUMAR P.J  
Assistant professor (Senior), SITE**



**SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING**

April, 2019

Gorbachev Rd, Vellore, Tamil Nadu 632014

info@vit.ac.in, Contact: 91-416-2243091 / 93

**ABSTRACT**

***This paper presents a Distributed Operating system addressing the security and performance needs of network attached storage devices in which file management functions occur at a different location. In much storage system file managers issue capabilities to client machines that directly access files stored on the network attached distributed storage devices without intervention by a file server.***

***These capabilities may be reused by the client, so that interaction with the file manager is kept to a minimum. System optimizes performance and scalability while separating the granting capabilities and validating a capability.***

***Keywords- Distributed Data Processing and Security; Computer files management; Cryptography; Client server systems.***

**CONTENTS**

**CONTENTS................................................................................................... iv**

**CONSLUSION LIST..................................................................................... xxvi**

**REFERENCES LIST..................................................................................... xxvii**

**CHAPTER 1**

**INTRODUCTION**

**1.1 INTRODUCTION ...................................................................................................... 5**

**1.2 LITERATURE SURVEY………………................................................................... 5**

**1.3 CODE AND IMPLEMENTATION…...................................................................... 20**

**1.4 OUTPUT OF IMPLEMENTATION........................................................................ 25**

**1.5 SCOPE OF THE PROJECT...................................................................................... 26**

**CHAPTER 2**

**BACK GROUND**

**2.1 INTRODUCTION ................................................................................................... 27**

**2.2 SURVEY ON MANETS .......................................................................................... 27**

Chapter 1

**INTRODUCTION**

**1.1 INTRODUCTION**

We are implementing an operating system running on Java and Net Beans and developed in core JAVA with multiple featues like Notepad , task scheduling , calculator and contain multithreading features, Its Kernal runs on Java Virtual Machine.

It is very secure. Compared to mainstream operating systems JX has a very small trusted computing base (TCB). This is discussed in more detail in the paper "A Java Operating System as the Foundation of a Secure Network Operating System". The unique protection domain model allows to run code with different trustworthiness at the same time in the same system. It is free and open source. You can use JX as platform for your own open source or propriatary projects. We do not charge royalties or require you to disclose your source code. For details please see our license. It conforms to standards. The core of the JX system is a cleanroom implementation of a Java Virtual Machine. This implementation conforms to the "Java Virtual Machine Specification", which allows you to use all tools and IDEs that produce bytecode compliant to this specification. There are several JX components that implement industry standards, such as the TCP communication protocol and the Ext2 file system. It is flexible. You can configure the system according to your needs either as a very small embedded system with limited functionality or as a feature rich server system

**1.2 LITERATURE SURVEY**

# Equipment for authenticating a client to a root in computing systems which support various security techniques:

1. This review [1] includes customer server frameworks are outstanding in the field of information preparing. In a customer server framework, a "customer" process issues a solicitation to a "server" procedure to play out an administration for it. Accordingly, the server transmits an answer to the customer, informing it of the consequences of the administration. This may access to a database or other document framework, access to printers, or access to all the more dominant registering resources[2]. A customer framework makes demands for administration from a server framework and, in numerous occasions, the server requires "confirmation" of the client before the administration will be given and, now and again, the customer will necessitate that the server be verified, to ensure that somebody isn't acting like the server. Customer confirmation infers the nearness of a security instrument whereby the server can check that the customer is approved to get the mentioned administration.

venture [3] present development identifies with customer server correspondence systems utilizing a security convention, and specifically to a go-between gadget, strategy, and program for the security convention. In a customer server framework utilizing a security convention, server verification at a customer, customer validation at a server, and information encryption at both customer and server are performed to anticipate listening in, altering, or message fraud, accomplishing secure correspondence between two gatherings. A regular security giving framework will be depicted taking for instancThee the TLS Protocol giving association security. As indicated by the TLS Protocol, the Handshake technique is performed to trade parameters fundamental for session and association before beginning the session. At the point when a first handshake has been finished, the session and a solitary association included in that are simultaneously settled.

In return the client sends a ClientHello message to the server to inform the server of starting TLS Handshake procedure. The ClientHello message includes a list of encryption and compression systems supported by the client. When receiving the ClientHello message from the client, the server sends a ServerHello message back to the client. The ServerHello message includes a Session ID assigned to the present session and a selected system from the client-supported encryption and compression systems included in the ClientHello message. Accordingly, the ServerHello message informs the client of the encryption and compression system to be used between the server and the client. In addition, the server uses Certificate, ServerKeyExchange, CertificateRequest, and ServerHelloDone messages to inform the client of X.509 certification and the like including a server public key that is a parameter to be determined before exchanging data with the client. The client sends a CRL request to the CA and then downloads CRL data from the CA. The client determines whether the certification of the server is included in the CRL received from the CA. When found, it is determined that the certification of the server has been revoked and the client sends an alert message to the server. the client uses ClientKeyExchange, ChangeCipherSpec, and Finished messages to exchange a public key required for encryption and decryption and then send a test message encrypted using the public key to the server so that the server check whether encryption and decryption are successfully performed.

1. **Techniques for banking institution institution server and users :**

The proposition of the project[4] is that budgetary exchange handling framework incorporates somewhere around one monetary server associated through an open system to various clients related with customer PCs. Every client gets to the budgetary server through an internet browser. The internet browser is furnished with the abilities to create encryption keys, to scramble and decode HTML frames, and to carefully sign and timestamp HTML shapes. The money related server exchanges site pages including HTML shapes speaking to budgetary exchanges. The HTML frames contain expansions that determine the organization of an approaching configuration and the arrangement of a returned structure. The money related server tracks each handled exchange through a review trail including the client's record, the client's computerized mark, the timestamp of the exchange, and the content of the exchange. To restore the encoded structure information, the internet browser produces an irregular session key that is utilized to scramble the message containing the structure information. The session key is joined to the arrival message and encoded with the server's open key.

A header record is incorporated at the highest point of the message and incorporates a banner showing that the structure information is scrambled. On the off chance that a structure requires the client's advanced mark, the internet browser utilizes the client's private key to "sign" the returned structure information and to fasten a computerized timestamp. The web server verifies the client's computerized mark with the client's open key. For structures requiring a computerized mark and encryption, the internet browser signs the structure information with the client's private key and incorporates an advanced timestamp also. The irregular session key is joined to the arrival message and scrambled with the server's open key.

The web server peruses the header record related with a got message so as to decide the organization of the message. A banner related with the header record shows the specific arrangement. So as to check a client's advanced mark, a client enlistment process is performed at first by the web server at the time a client's record is set up. The enrollment procedure is encouraged by a client enlistment HTML structure that inspires money related information relating to a potential client. A client affixss its open key to the enrollment structure information which is come back to the web server. The web server stores the clients open key in a database and it is utilized from that point to confirm the client's advanced mark which might be inserted in consequent exchanges.

# Automatically detecting security risk in operating system applications:

# This undertaking favors [5] a strategy for consequently distinguishing security vulnerabilities in a customer server application where a customer is associated with a server. The strategy is actualized by a PC having a processor and a product program put away on a non-momentary PC clear medium. The technique incorporates consequently separating, with the product program at the customer, a portrayal of at least one approval keeps an eye on sources of info performed by the customer. The strategy additionally incorporates examining the server, with the product program by utilizing the at least one approval minds inputs performed by the customer, to decide if the server isn't performing approval watches that the server must perform. The strategy further incorporates establishing that security vulnerabilities in the customer server application exist when the server isn't performing approval watches that the server must perform. A technique further proposes avoiding parameter altering assaults on a running customer server application by implementing the at least one approval keeps an eye on data sources performed by the customer on each info that is submitted to the server.

# The invention[6] depicted in this application exhibits a novel methodology for consequently recognizing and distinguishing potential server-side security vulnerabilities in existing customer server applications through blackbox and whitebox examination of the server's code. The particular apparatuses (i.e., programming programs) used to identify the security vulnerabilities in web applications are additionally portrayed.

# In one exemplification, the development gives a strategy to naturally recognizing security vulnerabilities in a customer server application, where a customer is associated with a server. The technique is executed by a PC having a processor and a product program put away on a non-momentary PC comprehensible medium. The technique incorporates naturally removing, with the product program at the customer, a depiction of at least one approval keeps an eye on data sources performed by the customer.

# 

# Secure push and status communication between client and server

# This project prefers [7] Systems and methods of authentication and authorization between a client, a server, and a gateway to facilitate communicating a message between a client and a server through a gateway. The client has a trusted relationship with each of the gateway and the server. A method includes registering the client with the gateway. The client also constructs the address space identifying the gateway and the client. The client communicates the address space to the server. The client receives an identity identifying the server. If the client authorizes to receive a message from the server through the gateway, the client informs the authorization to the gateway. The client puts the identity identifying the server on a list of servers which are authorized to send messages to the client. In addition, the client communicates the list of servers to the gateway.

# 

1. **Technique for client delegation of security to a proxy:**

A project[8] empowering an intermediary to take an interest in a protected correspondence between a customer and a server. The technique starts by setting up a first secure session between the customer and the intermediary. After confirming the main secure session, the technique proceeds by building up a second secure session between the customer and the intermediary. In the second secure session, the customer demands the intermediary to go about as a channel to the server. From that point, the customer and the server arrange a session ace mystery. Utilizing the primary secure session, this session ace mystery is then given by the customer to the intermediary to empower the intermediary to partake in secure interchanges between the customer and the server. Subsequent to accepting the session ace mystery, the intermediary produces cryptographic data that empowers it to give a given administration (e.g., transcoding, checking, encryption/unscrambling, reserving, or something like that) on the customer's sake and without the server's learning or investment. The primary secure session is kept up between the customer and the intermediary amid such correspondences. System security conventions, for example, Netscape's Secure Sockets Layer convention (SSL) and the Internet Engineering Task Force (IETF) Transport Layer Security convention (TLS), give protection and information respectability between conveying applications.

Inescapable figuring customers, notwithstanding, normally don't bolster the full capacity set of a HTML Windows-based customer. Therefore, transcoding administrations normally are required to make an interpretation of data to be rendered on the unavoidable customer from one source markup language (e.g., HTML) to another (e.g., HDML or handheld gadget markup language). Confining outsider intercession in a safe session is additionally hazardous in different applications.

1. **Trustworthy client/server data processing network:**

The project[9] brings in customer/server figuring, particularly in the field of web based business, carefully marked accreditations are passed among customer and server to create trust between the gatherings. In any case, this necessitates one gathering uncover its accreditations (which could be viewed as delicate) to the next gathering before the revealing party knows anything about the accepting party (somebody needs to go first). To take care of this issue, the creation executes an exchange of accreditation divulgence called programmed steady certification exposure. Every accreditation held at a nearby site is related with an entrance strategy which depends on restricting site certifications. Approaching solicitations for accreditations are sensibly joined with the entrance arrangements to determine further exchange reactions. In such customer/server frameworks, it is imperative that the customer and the server build up an adequate dimension of trust in one another before they take part in an important association, on the grounds that the data that might be traded amid the customer's solicitation for server handling as well as the server's preparing result which is come back to the customer might be profoundly delicate data.

1. **Authentication with Client system:**

The venture portrays [10] Denial of administration by server asset depletion has turned into a noteworthy security danger in open correspondences systems. Open key validation does not totally secure against the assaults in light of the fact that the confirmation conventions frequently leave ways for an unauthenticated customer to devour a server's memory space and computational assets by starting an expansive number of convention runs and instigating the server to perform costly cryptographic calculations. We show how stateless validation conventions and the customer riddles of Juels and Brainard can be utilized to counteract such assaults

1. **Checking client/server availability:**

The project explains [11] Client/server availability is a complex issue due to the many possible configurations of client/server environments and failure modes of client, server, and network devices. Such complexity makes it difficult to properly account for availability in client/server architectural design. I present a methodology and data to help design engineers evaluate client/server availability. I evaluate client/server outage data, indicate the most important outage causes in a client/server environment, and discuss methods for improving client/server availability.

1. **Biometric client-server security mechanism:**

The present [13] development incorporates a customer server security framework. The customer server security framework incorporates a customer framework accepting first biometric information and having a first dimension security approval method. In one encapsulation, the first biometric information is discourse information and the principal level security approval strategy incorporates a first speaker acknowledgment calculation. A server framework is accommodated accepting second biometric information. The server framework incorporates a second dimension security approval system.

1. **Privacy and Security for Operating system:**

The venture [12] portrays online informal organizations, for example, Facebook, Myspace, and Twitter have encountered exponential development lately. These OSNs offer appealing methods for online social collaborations and interchanges, yet additionally raise protection and security concerns. In this article we talk about the structure issues for the security and protection of OSNs. We find there are innate structure clashes among these and the conventional plan objectives of OSNs, for example, ease of use and friendliness. We present the remarkable security and protection configuration challenges brought by the center functionalities of OSNs and feature a few chances of using interpersonal organization hypothesis to alleviate these plan clashes.

1. **Method and system for maintaining client server security associations in a distributed operating system:**

A technique [14] and framework for keeping up a protected relationship between a customer and a server in an appropriated processing framework by registering a session identifier as a component of a Kerberos-based confirmation ticket. The session identifier is freely determined or checked by the customer and the server upon a first solicitation by the customer to the server, and each ensuing solicitation by the customer to the server is labeled with this session identifier to give a dependable security affiliation.ID-based client authentication with key agreement protocol for mobile client–server system - ECC:

# Checking security threats in operating system:

# The paper [16] Recently 1 customer/server processing has turned into a genuine option in contrast to centralized computer figuring in industry. It offers a few advantages, however it likewise uncovered the processing condition to extra dangers: the adaptability that makes it appealing can likewise make it increasingly helpless against security ruptures. This paper reports the aftereffects of an examination that investigated how organizations that were moving from a centralized computer condition to one that included customer/server innovation, assessed and took measures to ensure against potential data security dangers. Obviously, in spite of the fact that safety efforts in the centralized server condition have been all around executed in respect to their apparent risk, the equivalent can't be said about the customer/server condition. Certain basic territories in the customer/server condition in which security introduction is likely are examined. Associations must wind up mindful of these basic territories and guarantee that fitting safety efforts are actualized to diminish the likelihood of misfortune.

## A fingerprint based bio‐cryptographic security protocol designed for client/server authentication in mobile computing environment:

## The paper presents [17] fast evolution of mobile devices and mobile network, the need of protecting user sensitive information locally and performing secure user authentication remotely become evermore increasing. Bio‐cryptography is emerging as a powerful solution which can combine the advantages of conventional cryptography and biometric security. In this paper, we present an efficient bio‐cryptographic security protocol designed for client/server authentication in current mobile computing environment, with a reasonable assumption that server is secure. In this protocol, fingerprint biometric is used in user verification, protected by a computationally efficient Public Key Infrastructure (PKI) scheme, Elliptic Curve Cryptography (ECC). The genuine fingerprint information is hidden in the feature vault which is the mixture of genuine and chaff features. Fingerprint features are not only used for biometric verification but also for cryptographic key generation. Our security analysis shows that the proposed protocol can provide a secure and trustworthy authentication of remote mobile users over insecure network. Experimental results on public domain database show an acceptable verification performance.

# Security between users and system which they are using:

# The paper [19] improved security between a customer and a server in a PC arrange is given by permitting either endpoint (the customer or the server) to start demand messages. Along these lines, it is conceivable to design the framework so the server dependably makes the opening move of exchange, enabling the area of the server to stay covered up until a legitimate session is set up. Dynamic movement of the server further shrouds the area of the server from unapproved clients. Furthermore, each message might be validated exclusively as it is gotten, with the endpoint making no reaction to an unauthentic message, in this manner anticipating assaults on its security. At long last, exchange of both the encryption technique and the key utilized in the encryption procedure takes into account the fast reconfiguration of encryption to secure against unapproved clients who may have broken the code.

# Protocol for proving authenticity:

# A convention [20] for building up the credibility of a customer to a server in an electronic exchange by encoding a testament with a key known just to the customer and the server. The trust of the server, if fundamental, can be set up by an open key convention. The customer creates and sends over a correspondences channel a message containing no less than a piece of a declaration encoded with the server's open key or a mystery session key. The server gets and forms the message to recoup in any event part of the authentication, confirms and acknowledges it as confirmation of the customer's credibility.

# Client-server system for maintaining a user desktop consistent with server application user access permissions:

# A framework [21] with a system interconnecting a server and a majority of client stations. The server stores a majority of client applications for downloading to client stations and further stores get to authorizations for the applications for every client. At the point when a client endeavors to sign onto the framework, the server utilizes the client's sign on identifier to manufacture a rundown of utilizations for which the client approaches authorization. The server downloads to the station a rundown of utilizations to which the client approaches authorization. The client station utilizes the rundown to assemble an envelope containing just the applications from the rundown to which the client approaches authorization. The framework further confirms from the rundown that the client approaches applications that are spoken to by articles that the client may have added to his or her work area at a prior time. For every client work area inclination indicated by the client at a prior time that compares to a client application, the entrance authorization for the client to the client application is checked from the rundown, and, if the application is excluded on the rundown, the work area object speaking to the application is expelled from the work area.

# Internet/network security method:

# The paper speaks to [22] techniques and contraption for system security frameworks, which are especially appropriate for discovering vulnerabilities to PC hacking and unapproved passage is unveiled. An utilization of the system security framework technique and mechanical assembly to PC systems is additionally uncovered for either an Internet-based framework or an inside PC organize framework.

# System and method for security in operating system:

# A system [23] of security for global computer transactions that provides a reverse-authentication authority system that enables a client to authenticate the server to the client first before the user of a client proceed to enter their user id and password or other personal data in the web page to assure them that the web page did indeed originate from the secure web server and not from an imposter. The system uses an authentication code that is given to the client user when they call a reverse-authentication server by an out-of-band method such as a telephone network, different than the global network and which then appears in the part of the web page for them to visually see and compare the one they have received in the telephone call to the server.

1. **Method for updating security in operating system:**

A method [24] for updating information on security , in which the client is connected with a server through a network, the server including a storage device that is managed by the client, the storage device storing security information, the method comprising updating the security information stored in the storage device that the client manages in the server.

1. **Security and software testing in operating system:**

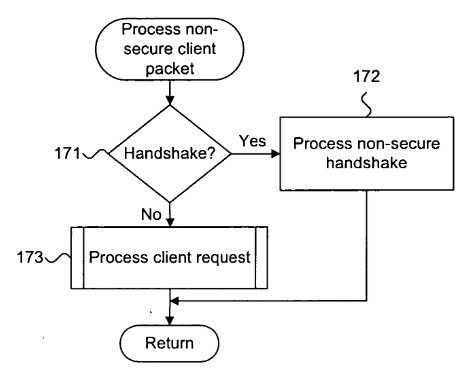
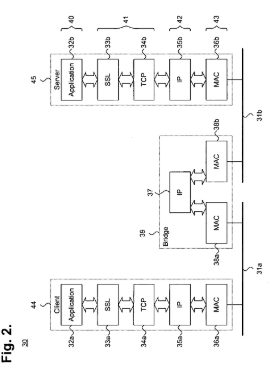
A framework [25] strategy and PC program item for testing scanner refreshes. At first, a full-discharge scanner update is conveyed from a server to a majority of PCs using a system. A pre-discharge scanner update is additionally appropriated from the server to the PCs using the system. The full-discharge scanner update is executed on the PCs for security filtering. Further, the pre-discharge scanner update is executed on the PCs for testing purposes. Aftereffects of the testing are transmitted from the PCs to the server using the system.

1. **Protecting client privacy:**

Framework [26] current believed registering activities for the most part include vast associations putting physically secure equipment on client machines, conceivably damaging client protection. However, it's conceivable to misuse strong server-side secure equipment to upgrade client protection Two contextual investigations exhibit utilizing secure coprocessors at the server.

1. **System and method for improving client response time:**

A framework [25] and strategy for giving incorporated verified and enhanced parcel informing is depicted. A majority of solicitation bundles arranged in a parcel line from a mentioning customer and determining content for recovery from a goal server are ordered. The substance is recovered from the goal server. The recovered substance is streamlined for no less than one such solicitation bundle. The recovered substance is traded as secure substance ensured utilizing a figure consulted with the mentioning customer for somewhere around one such solicitation bundle.

****

1. **Agents and security:**

The common sense [28] of portable specialists relies on practical security procedures. Portable specialist frameworks are blend customer/servers that vehicle, and furnish an interface with host PCs for, versatile operators. Transport of portable operators happens between versatile specialist frameworks, which are situated on heterogeneous stages, making up a foundation that can possibly scale to the span of any hidden system. Versatile specialists can be quickly sent, and can react to one another and their condition. These capacities uncover imperfections in current security innovation. This article studies the dangers associated with the utilization of portable specialists, and security strategies accessible to ensure versatile operators and their hosts. The insufficiencies of the security methods created from the data post show are recognized. They are the aftereffect of utilizing a decent model in a wrong setting (for example a shut framework demonstrate in an all around conveyed organizing processing base). Issues with industrially accessible methods include: (1) clashes between security strategies ensuring hosts and portable specialists, (2) powerlessness to deal with various community oriented versatile operators, and (3) accentuation on the qualifications of programming rather than on the honesty of programming to decide the dimension of trust.

1. **Communication in system:**

The common sense [28] of portable operators depends on reasonable security systems. Portable operator frameworks are blend customer/servers that vehicle, and furnish an interface with host PCs for, versatile specialists. Transport of portable specialists happens between versatile operator frameworks, which are situated on heterogeneous stages, making up a foundation that can possibly scale to the span of any fundamental system. Versatile operators can be quickly sent, and can react to one another and their condition. These capacities uncover imperfections in current security innovation. This article reviews the dangers associated with the utilization of versatile specialists, and security systems accessible to ensure portable operators and their hosts. The deficiencies of the security strategies created from the data stronghold display are recognized. They are the aftereffect of utilizing a decent model in a wrong setting (for example a shut framework display in an all inclusive dispersed systems administration registering base). Issues with financially accessible methods include: (1) clashes between security systems ensuring hosts and versatile specialists, (2) failure to deal with numerous community portable operators, and (3) accentuation on the accreditations of programming rather than on the respectability of programming to decide the dimension of trust.

1. **Strong secret from a password:**

A wandering client [30] who gets to a system front distinctive customer terminals, can be upheld by a qualifications server that verifies the client by secret phrase at that point helps with propelling a protected situation for the client. In any case, conventional accreditations server structures are helpless against comprehensive secret phrase speculating assault at the server. We depict an accreditations server model and supporting convention that beats that inadequacy. The convention accommodates safely creating a solid mystery from a frail mystery (secret word), in light of correspondences trades with at least two autonomous servers. The outcome can be utilized in different ways, for instance, the solid mystery can be utilized to unscramble an encoded private key or it tends to be utilized in unequivocally confirming to an application server. The convention has the properties that a future aggressor can't possibly total the solid mystery and has just a restricted chance to figure the secret key, regardless of whether the person approaches all messages and has authority over a few, however not all, of the servers.

1. **Security Protocol:**

The paper [31] Order-saving encryption - an encryption conspire where the sort request of ciphertexts matches the sort request of the comparing plaintexts - enables databases and different applications to process inquiries including request over encoded information proficiently. The perfect security ensure for request safeguarding encryption set forth in the writing is for the ciphertexts to uncover no data about the plaintexts other than request. Despite the fact that in excess of twelve plans were proposed, every one of these plans release more data than request. This paper exhibits the main request protecting plan that accomplishes perfect security. Our fundamental system is variable ciphertexts, implying that after some time, the ciphertexts for few plaintext values change, and we demonstrate that impermanent ciphertexts are required for perfect security. Our subsequent convention is intelligent, with few collaborations. We executed our plan and assessed it on microbenchmarks and with regards to an encoded MySQL database application. We demonstrate that notwithstanding giving perfect security, our plan accomplishes 1 - 2 requests of greatness higher execution than the best in class request saving encryption plot, which is less secure than our plan.

**1.3 CODE AND IMPLEMENTATION**

package osproject;

import java.beans.PropertyVetoException;

import java.util.logging.Level;

import java.util.logging.Logge;

public class Calculator extends javax.swing.JInternalFrame {

char o;

int ctr;

String value, cv, oBtn;

Double answer, v1, v2;

Double NumberConverted;

int count;

/\*\*

\* Creates new form Calculator

\*/

public Calculator(int cnt) {

initComponents();

ctr=0;

value="";cv="";oBtn="";

answer=0d;

v1=0d;v2=0d;NumberConverted=0d;

count=cnt;

//System.out.println(count+" Created");

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

tField = new javax.swing.JTextField();

bCE = new javax.swing.JButton();

clear = new javax.swing.JButton();

backspace = new javax.swing.JButton();

num7 = new javax.swing.JButton();

num8 = new javax.swing.JButton();

num9 = new javax.swing.JButton();

bDiv = new javax.swing.JButton();

bPer = new javax.swing.JButton();

bFrac = new javax.swing.JButton();

num5 = new javax.swing.JButton();

num6 = new javax.swing.JButton();

bMul = new javax.swing.JButton();

num4 = new javax.swing.JButton();

num1 = new javax.swing.JButton();

num2 = new javax.swing.JButton();

bSub = new javax.swing.JButton();

bSqrt = new javax.swing.JButton();

num3 = new javax.swing.JButton();

bDot = new javax.swing.JButton();

bAdd = new javax.swing.JButton();

equals = new javax.swing.JButton();

bInt = new javax.swing.JButton();

num0 = new javax.swing.JButton();

setClosable(true);

setIconifiable(true);

setNormalBounds(new java.awt.Rectangle(10, 10, 10, 10));

setPreferredSize(new java.awt.Dimension(300, 500));

addInternalFrameListener(new javax.swing.event.InternalFrameListener() {

public void internalFrameActivated(javax.swing.event.InternalFrameEvent evt) {

}

public void internalFrameClosed(javax.swing.event.InternalFrameEvent evt) {

formInternalFrameClosed(evt);

}

public void internalFrameClosing(javax.swing.event.InternalFrameEvent evt) {

}

public void internalFrameDeactivated(javax.swing.event.InternalFrameEvent evt) {

formInternalFrameDeactivated(evt);

}

public void internalFrameDeiconified(javax.swing.event.InternalFrameEvent evt) {

formInternalFrameDeiconified(evt);

}

public void internalFrameIconified(javax.swing.event.InternalFrameEvent evt) {

}

public void internalFrameOpened(javax.swing.event.InternalFrameEvent evt) {

}

});

tField.setEditable(false);

tField.setFont(new java.awt.Font("Tahoma", 0, 16)); // NOI18N

tField.setText("0");

tField.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(139, 69, 19), 2));

tField.setDisabledTextColor(new java.awt.Color(0, 0, 0));

tField.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

tFieldActionPerformed(evt);

}

});

bCE.setText("CE");

bCE.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

bCE.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

bCEActionPerformed(evt);

}

});

clear.setText("C");

clear.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

clear.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

clearActionPerformed(evt);

}

});

backspace.setText("Backspace");

backspace.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

backspace.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

backspaceActionPerformed(evt);

}

});

num7.setText("7");

num7.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

num7.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

num7ActionPerformed(evt);

}

});

num8.setText("8");

num8.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

num8.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

num8ActionPerformed(evt);

}

});

num9.setText("9");

num9.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

num9.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

num9ActionPerformed(evt);

}

});

bDiv.setText("/");

bDiv.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

bDiv.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

bDivActionPerformed(evt);

}

});

bPer.setText("%");

bPer.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

bPer.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

bPerActionPerformed(evt);

}

});

bFrac.setText("1/x");

bFrac.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

bFrac.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

bFracActionPerformed(evt);

}

});

num5.setText("5");

num5.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

num5.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

num5ActionPerformed(evt);

}

});

num6.setText("6");

num6.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

num6.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

num6ActionPerformed(evt);

}

});

bMul.setText("x");

bMul.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

bMul.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

bMulActionPerformed(evt);

}

});

num4.setText("4");

num4.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

num4.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

num4ActionPerformed(evt);

}

});

num1.setText("1");

num1.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

num1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

num1ActionPerformed(evt);

}

});

num2.setText("2");

num2.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

num2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

num2ActionPerformed(evt);

}

});

bSub.setText("-");

bSub.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

bSub.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

bSubActionPerformed(evt);

}

});

bSqrt.setText("sqrt");

bSqrt.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

bSqrt.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

bSqrtActionPerformed(evt);

}

});

num3.setText("3");

num3.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

num3.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

num3ActionPerformed(evt);

}

});

bDot.setText(".");

bDot.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

bDot.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

bDotActionPerformed(evt);

}

});

bAdd.setText("+");

bAdd.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

bAdd.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

bAddActionPerformed(evt);

}

});

equals.setText("=");

equals.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

equals.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

equalsActionPerformed(evt);

}

});

bInt.setText("+/-");

bInt.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

bInt.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

bIntActionPerformed(evt);

}

});

num0.setText("0");

num0.setBorder(new javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1, true));

num0.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

num0ActionPerformed(evt);

}

});

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

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(num0, javax.swing.GroupLayout.PREFERRED\_SIZE, 52, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(bInt, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(bDot, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(bAdd, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(equals, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addComponent(num4, javax.swing.GroupLayout.PREFERRED\_SIZE, 52, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(num5, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(num6, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(bMul, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(bFrac, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addComponent(num7, javax.swing.GroupLayout.PREFERRED\_SIZE, 52, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(num8, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(num9, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(bDiv, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(bPer, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addComponent(num1, javax.swing.GroupLayout.PREFERRED\_SIZE, 52, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(num2, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(num3, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(bSub, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(2, 2, 2)

.addComponent(bSqrt, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap())

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(tField)

.addGroup(layout.createSequentialGroup()

.addComponent(bCE, javax.swing.GroupLayout.PREFERRED\_SIZE, 82, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(clear, javax.swing.GroupLayout.PREFERRED\_SIZE, 82, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(backspace, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)))

.addGap(10, 10, 10))))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addContainerGap()

.addComponent(tField, javax.swing.GroupLayout.DEFAULT\_SIZE, 65, Short.MAX\_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(bCE, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(clear, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(backspace, javax.swing.GroupLayout.PREFERRED\_SIZE, 50, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(num7, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(num8, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(num9, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bDiv, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bPer, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(num4, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(num5, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(num6, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bMul, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bFrac, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(num1, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(num2, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(num3, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bSub, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bSqrt, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(num0, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bInt, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bDot, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(bAdd, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(equals, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)))

);

pack();

}// </editor-fold>

private void clearActionPerformed(java.awt.event.ActionEvent evt) {

ctr=0;

v1=0d;

v2=0d;

value="";

answer=0.;

tField.setText("0");

}

private void bCEActionPerformed(java.awt.event.ActionEvent evt) {

ctr=0;

value="";

tField.setText("0");

}

private void num9ActionPerformed(java.awt.event.ActionEvent evt) {

value+=9;

tField.setText(value);

}

private void num8ActionPerformed(java.awt.event.ActionEvent evt) {

value+=8;

tField.setText(value);

}

private void bDivActionPerformed(java.awt.event.ActionEvent evt) {

try{

v1 = Double.parseDouble( tField.getText() );

ctr=0;

o = '/';

value="";

tField.setText("" +value);

}

catch(StringIndexOutOfBoundsException | NumberFormatException | NullPointerException str){}

}

private void bPerActionPerformed(java.awt.event.ActionEvent evt) {

try{

v1 = Double.parseDouble( tField.getText() );

ctr=0;

value="";

answer=(v1/100);

tField.setText("" +answer);

}

catch(StringIndexOutOfBoundsException | NumberFormatException | NullPointerException str){}

}

private void bFracActionPerformed(java.awt.event.ActionEvent evt) {

try{

ctr=0;

value = "";

Double NumberContainer = ( 1 / Double.parseDouble(tField.getText() ) );

tField.setText("" +NumberContainer);

}

catch(StringIndexOutOfBoundsException | NumberFormatException | NullPointerException str){}

}

private void num5ActionPerformed(java.awt.event.ActionEvent evt) {

value+=5;

tField.setText(value);

}

private void num6ActionPerformed(java.awt.event.ActionEvent evt) {

value+=6;

tField.setText(value);

}

private void bMulActionPerformed(java.awt.event.ActionEvent evt) {

try{

v1 = Double.parseDouble( tField.getText() );

ctr=0;

o = 'x';

value="";

tField.setText("" +value);

}

catch(StringIndexOutOfBoundsException | NumberFormatException | NullPointerException str){}

}

private void num2ActionPerformed(java.awt.event.ActionEvent evt) {

value+=2;

tField.setText(value);

}

private void bSubActionPerformed(java.awt.event.ActionEvent evt) {

try{

v1 = Double.parseDouble( tField.getText() );

ctr=0;

o = '-';

value="";

tField.setText("" +value);

}

catch(StringIndexOutOfBoundsException | NumberFormatException | NullPointerException str){}

}

private void bSqrtActionPerformed(java.awt.event.ActionEvent evt) {

try{

ctr=0;

value = "";

v1 = Math.sqrt( Double.parseDouble( tField.getText() ) );

tField.setText("" +v1);

}

catch(StringIndexOutOfBoundsException | NumberFormatException | NullPointerException str){}

}

private void num3ActionPerformed(java.awt.event.ActionEvent evt) {

value+=3;

tField.setText(value);

}

private void bDotActionPerformed(java.awt.event.ActionEvent evt) {

if(ctr==0){

value+=".";

ctr+=1;

tField.setText("" +value);

}

}

private void bAddActionPerformed(java.awt.event.ActionEvent evt) {

try{

v1 = Double.parseDouble( tField.getText() );

ctr=0;

o = '+';

value="";

tField.setText("" +value);

}

catch(StringIndexOutOfBoundsException | NumberFormatException | NullPointerException str){}

}

private void equalsActionPerformed(java.awt.event.ActionEvent evt) {

try{

value="";

v2 = Double.parseDouble(tField.getText());

ctr=0;

switch (o) {

case '+':

answer = v1 + v2;

tField.setText("" +answer);

value="";

v1=null;

v2=null;

break;

case '-':

answer = v1 - v2;

tField.setText("" +answer);

value="";

v1=null;

v2=null;

break;

case 'x':

answer = v1 \* v2;

tField.setText("" +answer);

value="";

v1=null;

v2=null;

break;

case '/':

answer = v1 / v2;

tField.setText("" +answer);

value="";

v1=null;

v2=null;

break;

default:

break;

}

}

catch(StringIndexOutOfBoundsException | NumberFormatException | NullPointerException str){}

}

private void bIntActionPerformed(java.awt.event.ActionEvent evt) {

try{

ctr=0;

NumberConverted = ( Double.parseDouble(tField.getText()) \* -1 );

value = "";

tField.setText("" +NumberConverted);

}

catch(StringIndexOutOfBoundsException | NumberFormatException | NullPointerException str){}

}

private void tFieldActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void num0ActionPerformed(java.awt.event.ActionEvent evt) {

value+=0;

tField.setText(value);

//System.out.println(this.count+" button");

}

private void num1ActionPerformed(java.awt.event.ActionEvent evt) {

value+=1;

tField.setText(value);

}

private void num4ActionPerformed(java.awt.event.ActionEvent evt) {

value+=4;

tField.setText(value);

}

private void num7ActionPerformed(java.awt.event.ActionEvent evt) {

value+=7;

tField.setText(value);

}

private void backspaceActionPerformed(java.awt.event.ActionEvent evt) {

try{

value = value.substring(0, value.length()-1 );

tField.setText("" +value);

}

catch(StringIndexOutOfBoundsException | NumberFormatException | NullPointerException str){}

}

private void formInternalFrameDeactivated(javax.swing.event.InternalFrameEvent evt) {

}

private void formInternalFrameClosed(javax.swing.event.InternalFrameEvent evt) {

System.out.println("Closed!!!");

Home.processList.removeItem("Calculator");

Home.cal[count]=null;

//System.out.println(count+" Del");

}

private void formInternalFrameDeiconified(javax.swing.event.InternalFrameEvent evt) {

/\*for(int i=0;i<10;i++){

if(i!=count&&Home.cal[i]!=null)

try {

Home.cal[i].setIcon(true);

System.out.println("SetIcon");

} catch (PropertyVetoException ex) {

Logger.getLogger(Calculator.class.getName()).log(Level.SEVERE, null, ex);

}

}\*/

for(int j=0;j<10;j++){

System.out.println(j+" value of i");

if(Home.cal[j]!=null&&j!=count){

try {

Home.cal[j].setIcon(true);

} catch (PropertyVetoException ex) {

Logger.getLogger(Home.class.getName()).log(Level.SEVERE, null, ex);

}

}

if(Home.np[j]!=null){

try {

Home.np[j].setIcon(true);

} catch (PropertyVetoException ex) {

Logger.getLogger(Home.class.getName()).log(Level.SEVERE, null, ex);

}

}

if(Home.sc[j]!=null){

try {

Home.sc[j].setIcon(true);

} catch (PropertyVetoException ex) {

Logger.getLogger(Home.class.getName()).log(Level.SEVERE, null, ex);

}

}

if(Home.ter[j]!=null){

try {

Home.ter[j].setIcon(true);

} catch (PropertyVetoException ex) {

Logger.getLogger(Home.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

//Home.schedulingFunction('c', count);

}

// Variables declaration - do not modify

private javax.swing.JButton bAdd;

private javax.swing.JButton bCE;

private javax.swing.JButton bDiv;

private javax.swing.JButton bDot;

private javax.swing.JButton bFrac;

private javax.swing.JButton bInt;

private javax.swing.JButton bMul;

private javax.swing.JButton bPer;

private javax.swing.JButton bSqrt;

private javax.swing.JButton bSub;

private javax.swing.JButton backspace;

private javax.swing.JButton clear;

private javax.swing.JButton equals;

private javax.swing.JButton num0;

private javax.swing.JButton num1;

private javax.swing.JButton num2;

private javax.swing.JButton num3;

private javax.swing.JButton num4;

private javax.swing.JButton num5;

private javax.swing.JButton num6;

private javax.swing.JButton num7;

private javax.swing.JButton num8;

private javax.swing.JButton num9;

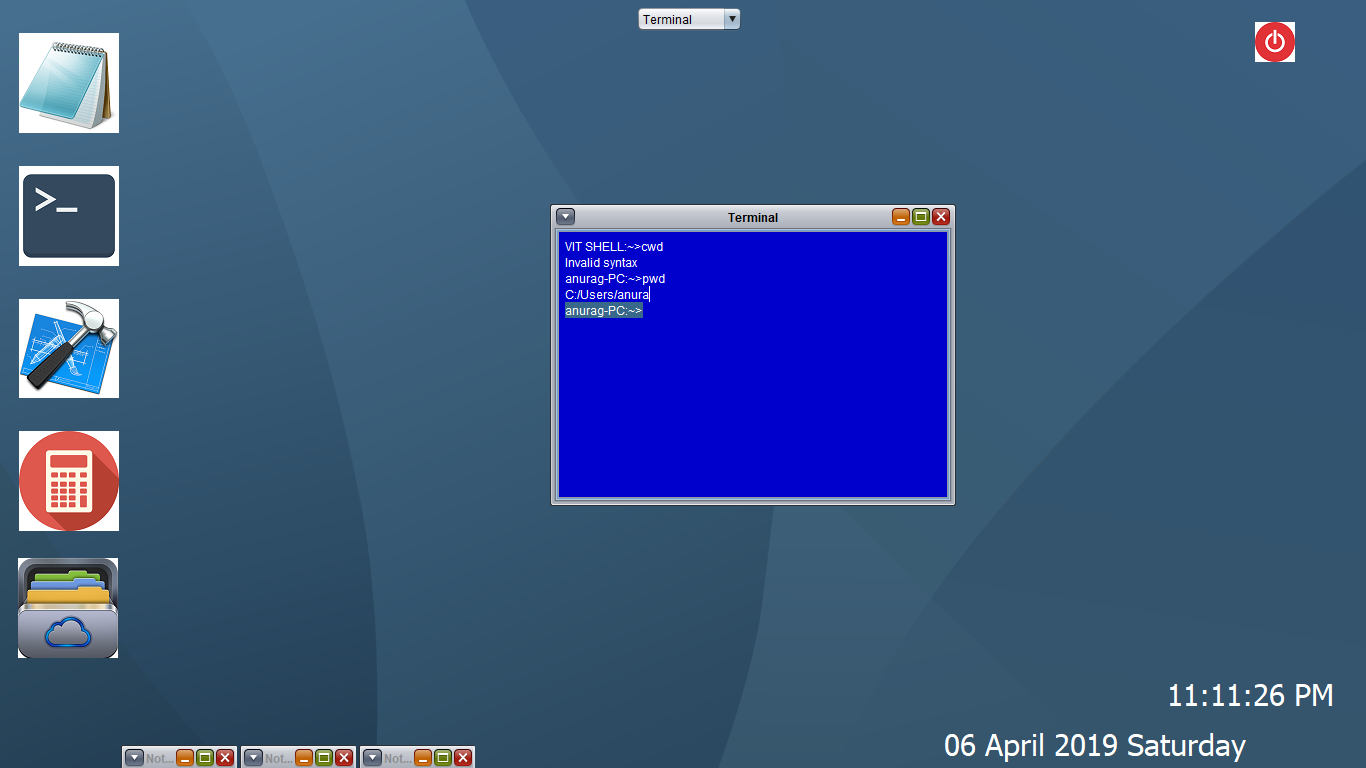
private javax.swing.JTextField tField;

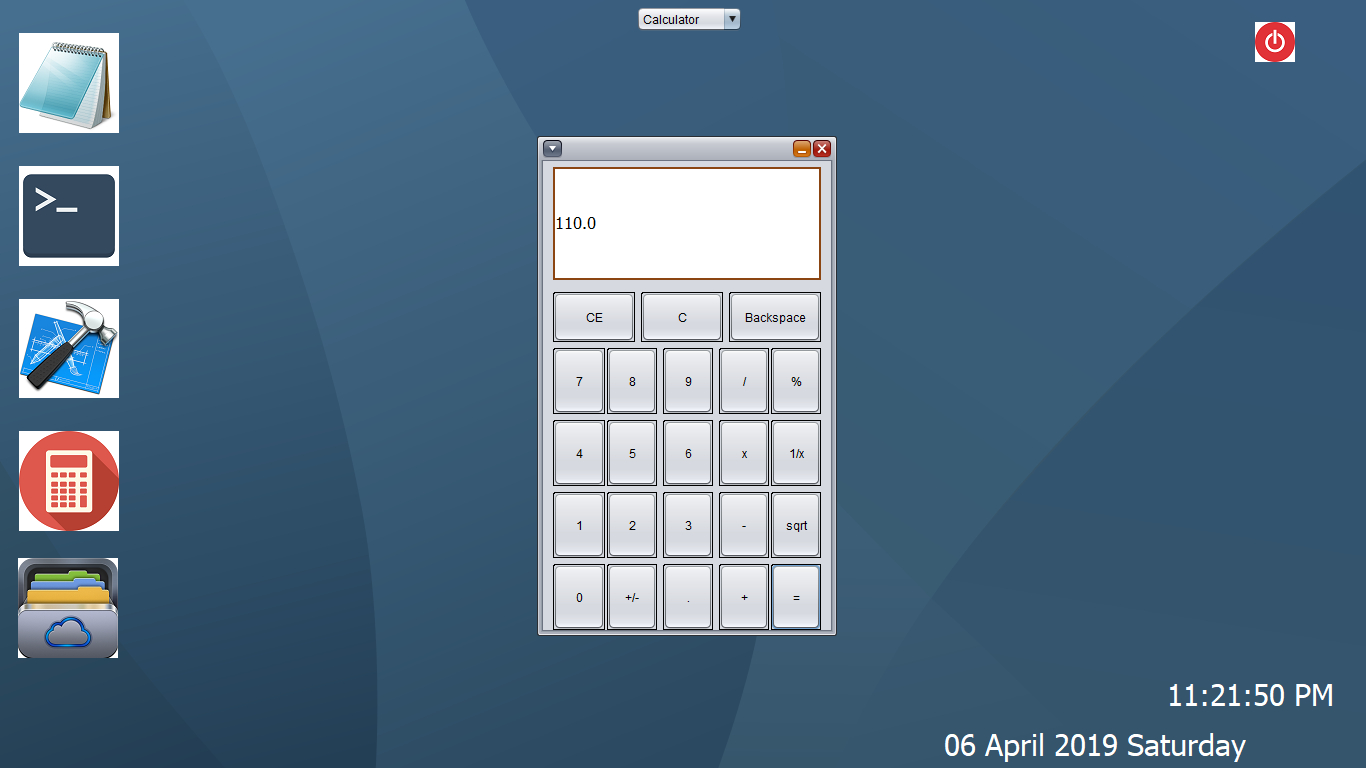
// End of variables declaration

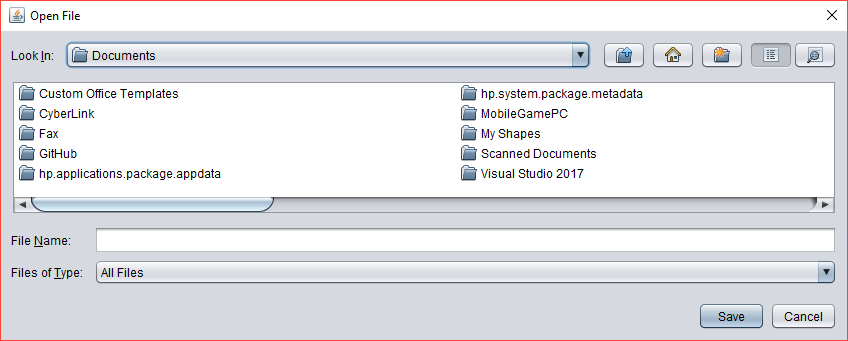
}

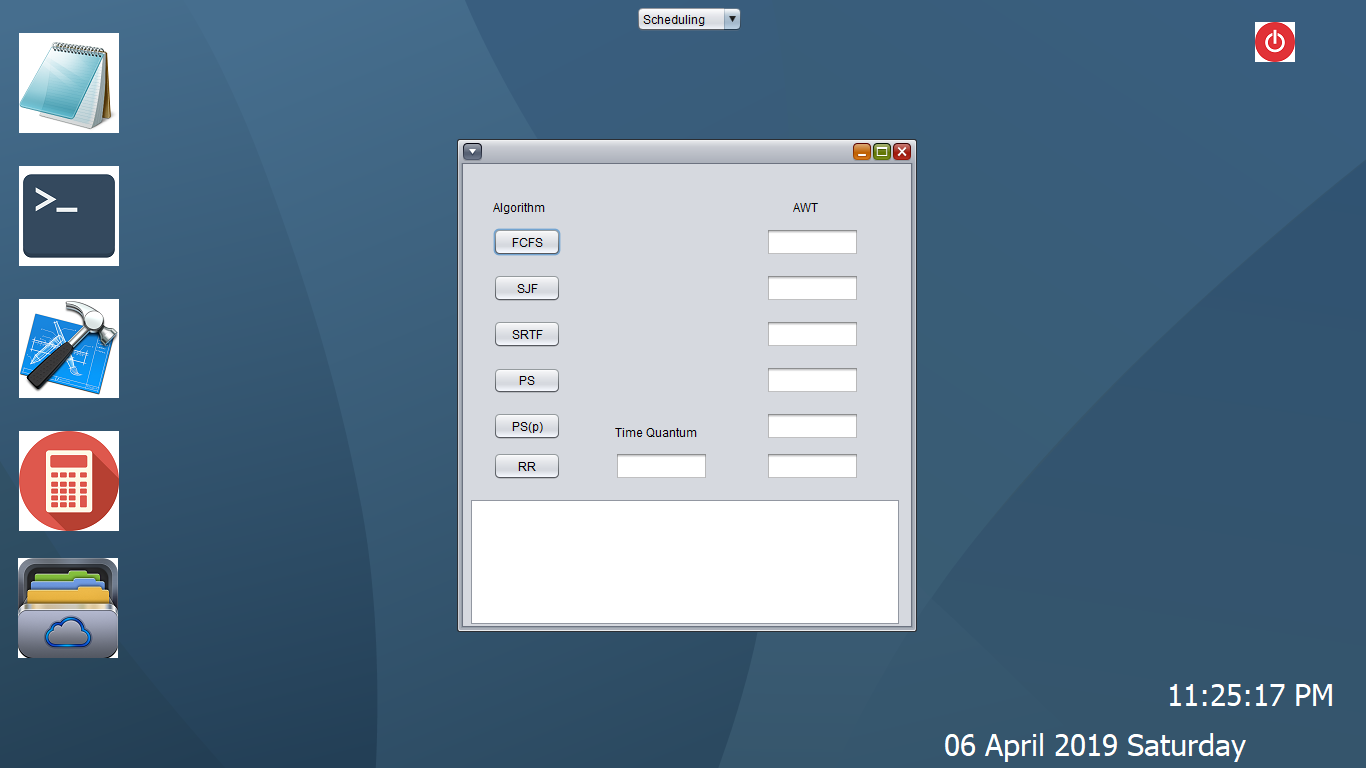
**1.4 OUTPUT OF IMPLEMENTATION**











**1.5 SCOPE OF THE PROJECT**

The Project many scopes like can help to connect with two or more separate machine and can transfer the data simontaneously encrypt message

The project is good for security purpose along with the compatibility of running the program. It is fast and convenient for both the user and customer.

In future we will develop more suffice and simplified along with more optimized features so that it can be ran in smooth way.

Chapter 2

**BACK GROUND**

* 1. **INTRODUCTION**

Operating System is well known in the field of data processing. In a operating syatem a “client” process issues a request to a “server” process to perform a service for it. In response, the server transmits a reply to the client, notifying it of the results of the service. A cryptographic capability of Distributed Operating system addressing the security and performance needs of network attached storage devices such as Client- server connection in which file management functions occur at many storage system file managers issue capabilities to client machines that directly access files stored on the network attached distributed storage devices without intervention by a file server.

This structured constructed our system that can work on client-server system along with network authentication service by which non-authorized devices or activities are prohibited.

Chapter n

**CONCLUSIONS AND FUTURE WORKS**

The Project many scopes like can help to connect with two or more separate machine and encrypt message with client and server end and middle tier will not be able to get the message cause of encrypted cipher text.

In future we will develop more suffice and simplified along with more optimized features so that it can be ran in smooth way and with more improvement.

**REFERENCES**

*[1] Hu, Wei-Ming. "Method and apparatus for authenticating a client to a server in computer systems which support different security mechanisms." U.S. Patent No. 5,586,260. 17 Dec. 1996.Proceedings of 1994 International Conference on Very Large Data Bases, pp. 487-499.*

*[2]Hu, W. M. (1996). U.S. Patent No. 5,586,260. Washington, DC: U.S. Patent and Trademark Office.*

*[1][2]Hu, Wei-Ming. "Method and apparatus for authenticating a client to a server in computer systems which support different security mechanisms." U.S. Patent 5,586,260, issued December 17, 1996.*

*[3] Mozer, Todd. "Client-server security system and method." U.S. Patent Application No. 09/875,261.*

*[3] Mozer, Todd. "Client-server security system and method." U.S. Patent Application 09/875,261, filed December 19, 2002.*

*[4]Chang, Sheueling, and Stuart Marks. "Security system and method for financial institution server and client web browser." U.S. Patent No. 6,105,012. 15 Aug. 2000.*

*[4] Chang, Sheueling, and Stuart Marks. "Security system and method for financial institution server and client web browser." U.S. Patent 6,105,012, issued August 15, 2000.*

*[4] Chang, S., & Marks, S. (2000). U.S. Patent No. 6,105,012. Washington, DC: U.S. Patent and Trademark Office.*

*[5] Bisht, Prithvi, Timothy Hinrichs, and Venkatesan Natarajan Venkatakrishnan. "System and a method for automatically detecting security vulnerabilities in client-server applications." U.S. Patent No. 9,118,713. 25 Aug. 2015.*

*[5] Bisht, P., Hinrichs, T., & Venkatakrishnan, V. N. (2015). U.S. Patent No. 9,118,713. Washington, DC: U.S. Patent and Trademark Office.*

*[6] Bisht, P., Hinrichs, T. and Venkatakrishnan,V.N.,University of Illinois, 2015. System and a method for automatically detecting security vulnerabilities in client-server applications. U.S. Patent 9,118,713.*

*[7]* *Herzog, Shai, Johannes Klein, and Alexandru Gavrilescu. "Secure push and status communication between client and server." U.S. Patent No. 8,099,764. 17 Jan. 2012.*

*[8]* *Lita, Christian, and Linas Vepstas. "Method for client delegation of security to a proxy." U.S. Patent No. 7,249,377. 24 Jul. 2007.*

*[9]* *Seamons, Kent Eldon, and William Hale Winsborough. "Trust negotiation in a client/server data processing network using automatic incremental credential disclosure." U.S. Patent No. 6,349,338. 19 Feb. 2002*.

*[10] Aura, Tuomas, Pekka Nikander, and Jussipekka Leiwo. "DOS-resistant authentication with client puzzles." International workshop on security protocols. Springer, Berlin, Heidelberg, 2000.*

*[11] Wood, Alan. "Predicting client/server availability." Computer28, no. 4 (1995): 41-48.*

*[12] Zhang, Chi, et al. "Privacy and security for online social networks: challenges and opportunities." IEEE network 24.4 (2010).*

*[13]* *Mozer, Todd F. "Biometric client-server security system and method." U.S. Patent No. 7,487,089. 3 Feb. 2009.*

*[14] Benantar, Messaoud, Robert Howard High Jr, and Mahesh Kumar Rathi. "Method and system for maintaining client server security associations in a distributed computing system." U.S. Patent No. 6,141,758. 31 Oct. 2000.*

*[15]* *Debiao, He, Chen Jianhua, and Hu Jin. "An ID-based client authentication with key agreement protocol for mobile client–server environment on ECC with provable security." Information Fusion 13.3 (2012): 223-230.*

*[16] Ryan, Sherry D., and Bijoy Bordoloi. "Evaluating security threats in mainframe and client/server environments." Information & Management 32.3 (1997): 137-146.*

*[17]* *Xi, Kai, et al. "A fingerprint based bio‐cryptographic security protocol designed for client/server authentication in mobile computing environment." Security and Communication Networks 4.5 (2011): 487-499.*

*[18]* *Su, Jin, Paul B. Hillyard, and Alan B. Butt. "PKI-based client/server authentication." U.S. Patent No. 7,032,110. 18 Apr. 2006.*

### *[19]* *Nguyen, Dan Linh. "Security Between client and server in a computer network." U.S. Patent No. 6,502,192. 31 Dec. 2002.*

### *[20] Kaliski Jr, Burton S. "Client/server protocol for proving authenticity." U.S. Patent No. 6,085,320. 4 Jul. 2000.*

### *[21] Hayes Jr, Kent Fillmore, and Brett Graham King. "Client-server system for maintaining a user desktop consistent with server application user access permissions." U.S. Patent No. 6,339,826. 15 Jan. 2002.*

### *[22] "Internet/network security method and system for checking security of a client from a remote facility." U.S. Patent Application 09/817,347, filed October 25, 2001.*

### *[23] Harris, Brendon, and Ray Hunt. "TCP/IP security threats and attack methods." Computer communications 22.10 (1999): 885-897.*

### *[24]* *Oshima, Satoshi, and Masahide Sato. "Method for updating security information, client, server and management computer therefor." U.S. Patent No. 7,225,461. 29 May 2007*

*[25] Ford, Warwick, and Burton S. Kaliski. "Server-assisted generation of a strong secret from a password." Enabling Technologies: Infrastructure for Collaborative Enterprises, 2000.(WET ICE 2000). Proeedings. IEEE 9th International Workshops on. IEEE, 2000.*

*[26] Iliev, A., & Smith, S. W. (2005). Protecting client privacy with trusted computing at the server. IEEE Security & Privacy, 3(2), 20-28.*

*[27] Mahaffey, K. P., Richardson, D. L., Salomon, A., Croy, R. T., Walker, S. A., Buck, B. J., ... & Golombek, D. (2016). U.S. Patent No. 9,374,369. Washington, DC: U.S. Patent and Trademark Office.*

*[28] Greenberg, Michael S., Jennifer C. Byington, and David G. Harper. "Mobile agents and security." IEEE Communications Magazine 36.7 (1998): 76-85.*

*[29] Linderman, Michael. "Communication security system." U.S. Patent Application No. 10/337,180.*

*[30] McEwan, William Alexander. "Security and software testing of pre-release anti-virus updates on client and transmitting the results to the server." U.S. Patent No. 7,231,637. 12 Jun. 2007.*

---------------------------------------------------------x----------------------------------------------------------