AMECA

Symmetric Cryptography Encryptor & Decryptor

Group 8 | 4 ISA 3 | CRUD project

TEAM

Leader

Riki Awal Syahputra
(2120010136)

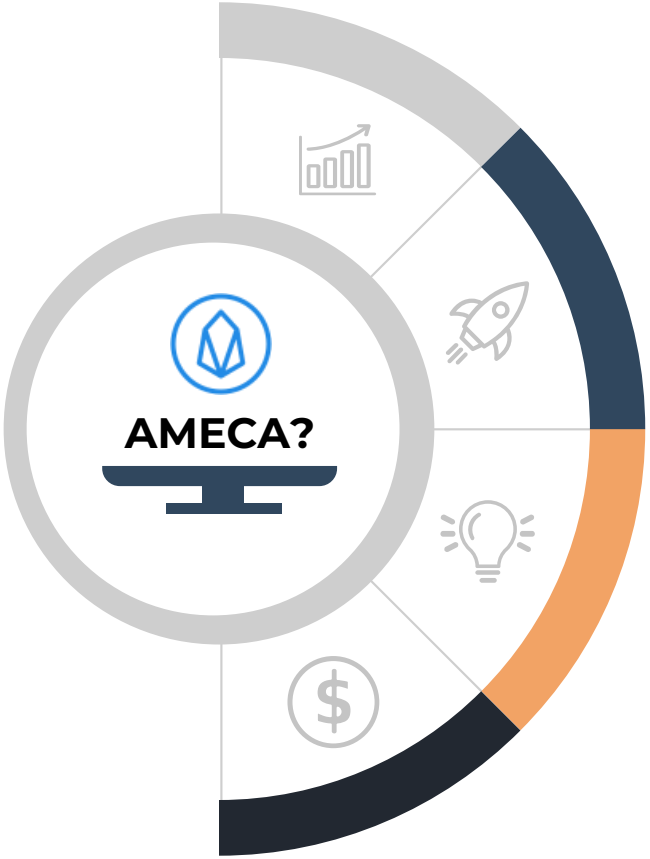
Developer

Faza Rama Nugraha
(2120010291)

TABLE OF CONTENT

- System Analysis
- Design Interface
- Database Design
- Table Design
- Flow Diagram
- Code
- Configuration

System Analysis



01

Java-based Dekstop Application

02

To introduce the Symmetric Encryption

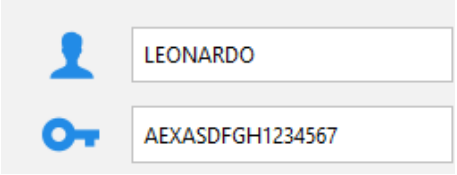
03

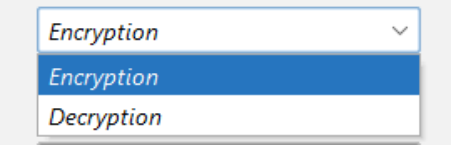
Using 1 key to encrypt & Decrypt

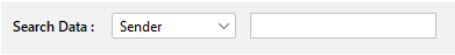
04

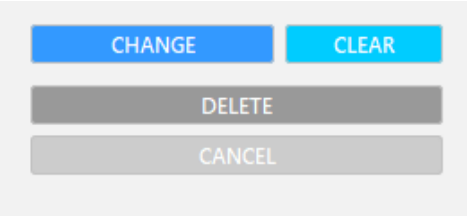
Powered by AES (Advanced Encryption Standard)

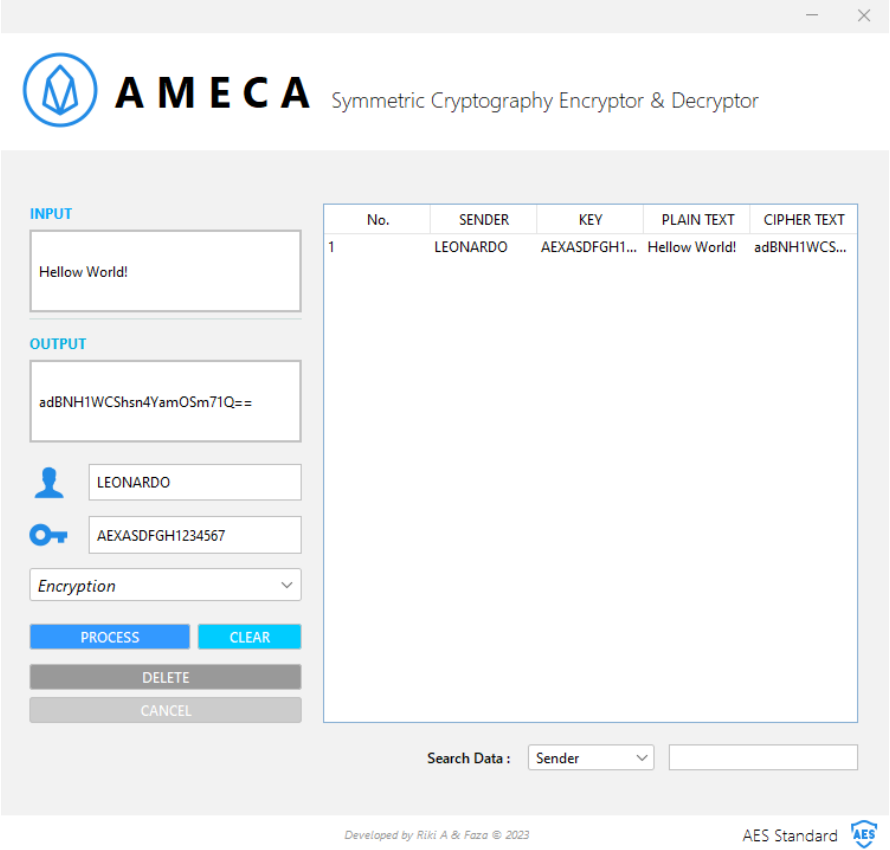
Design Interface

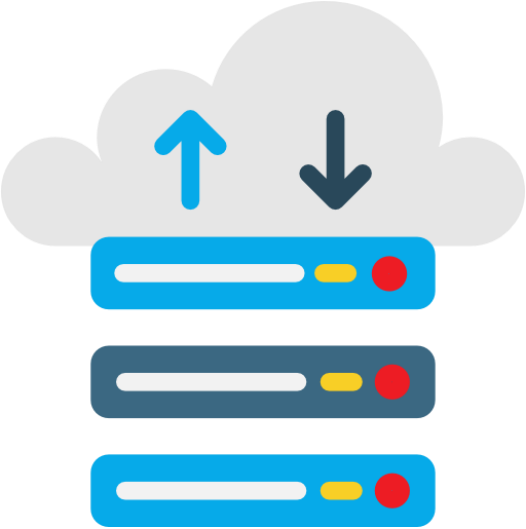

LEONARDO
AEXASDFGH1234567


Encryption
Encryption
Decryption


Search Data :
Sender


CHANGE
CLEAR
DELETE
CANCEL


AMECA
Symmetric Cryptography Encryptor & Decryptor
INPUT
Hellow World!
OUTPUT
adBNH1WCShsn4YamOSm71Q==
LEONARDO
AEXASDFGH1234567
Encryption
PROCESS
CLEAR
DELETE
CANCEL
No.
SENDER
KEY
PLAIN TEXT
CIPHER TEXT
1
LEONARDO
AEXASDFGH1...
Hellow World!
adBNH1WCS...
Search Data :
Sender
Developed by Riki A & Faza © 2023
AES Standard



Database Design

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
☐	1	**sender**	varchar(50)	utf8mb4_general_ci		No	*None*			Change	Drop	More
☐	2	**enKey**	varchar(32)	utf8mb4_general_ci		No	*None*			Change	Drop	More
☐	3	**plain**	varchar(2000)	utf8mb4_general_ci		No	*None*			Change	Drop	More
☐	4	**cipher**	varchar(2000)	utf8mb4_general_ci		No	*None*			Change	Drop	More

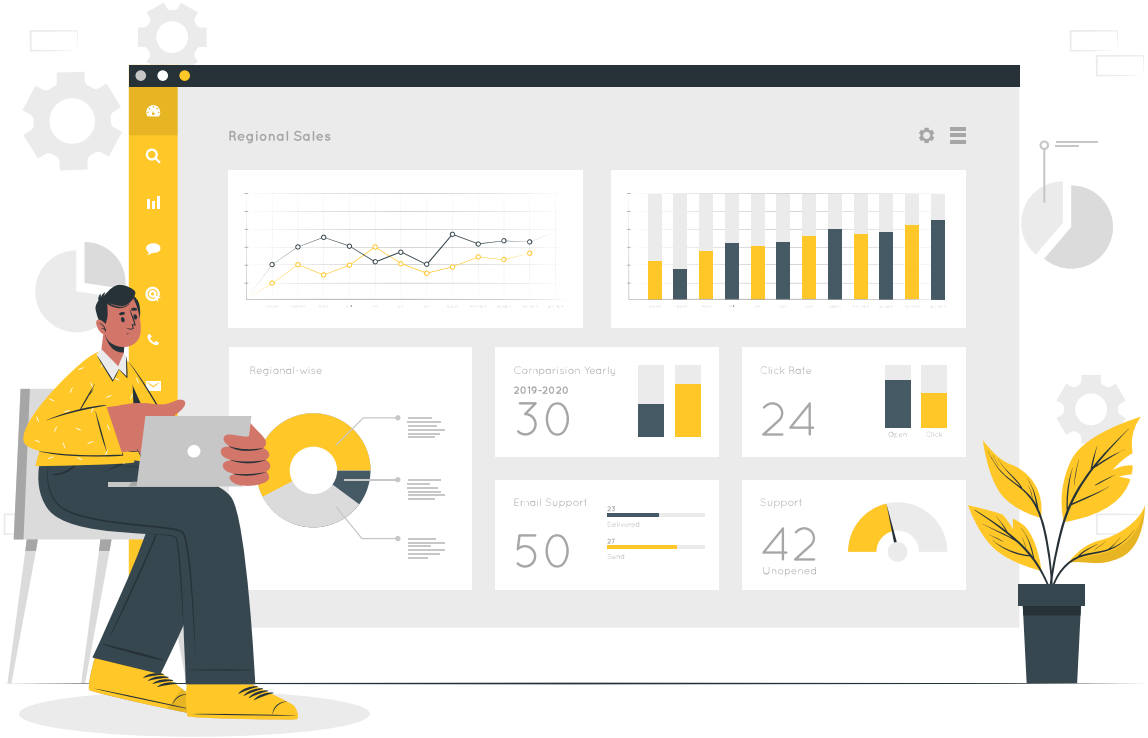
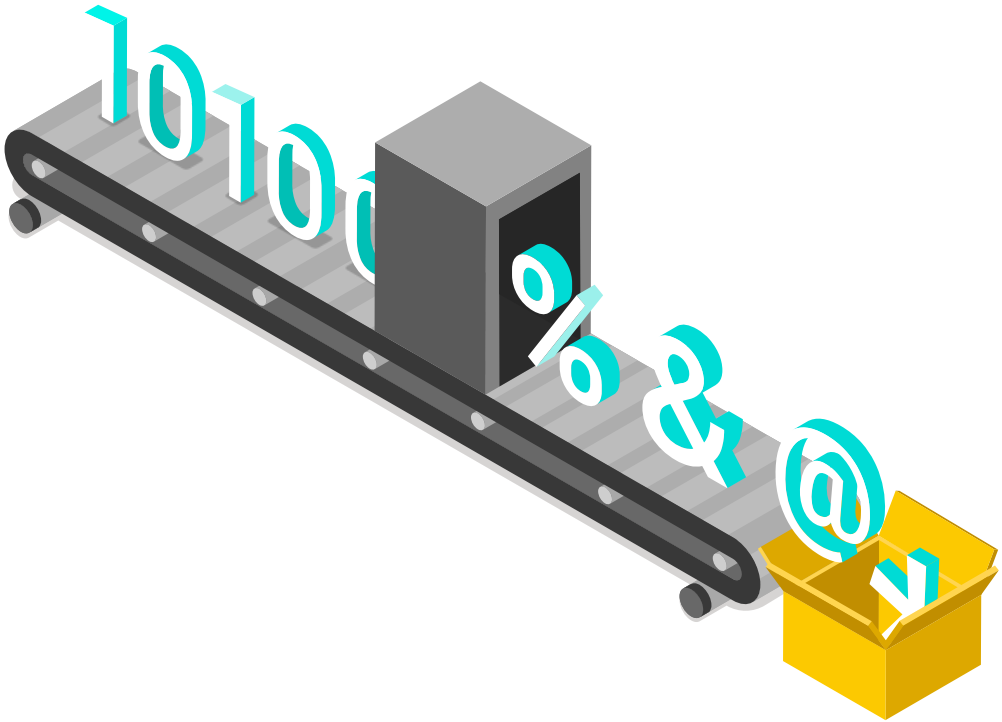


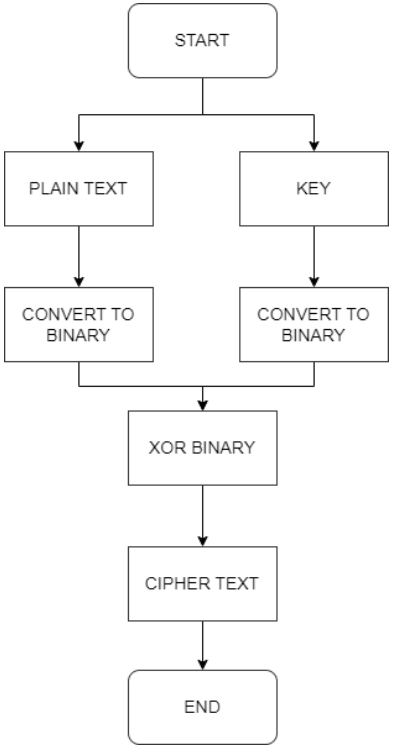
Table Design

NO	Field Name	Data Type	Length	Description
1	sender	varchar	50	Sender name (cryptographer)
2	enKey	varchar	32	Encryption key (to encrypt & decrypt)
3	plain	varchar	2000	plain text, the original text
4	cipher	varchar	2000	cipher text, the encrypted text

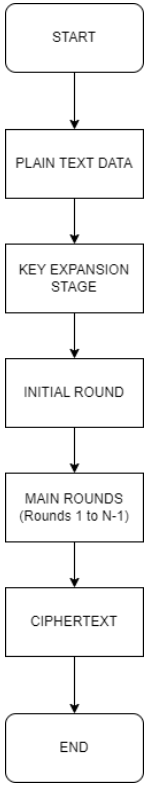


Flow Diagram

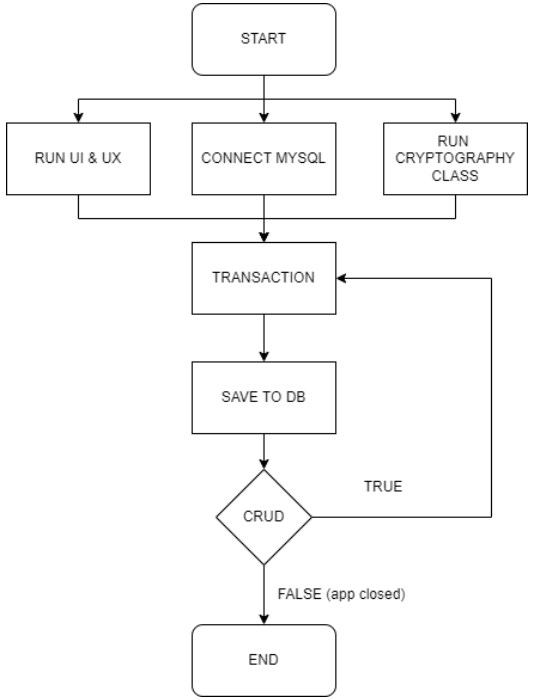
AMECA (Basic Symmetric Encryption)



AMECA Symmetric Encryption (AES based)



AMECA app flow diagram (basic)



Code

Set UI Style

(FlatLightLaf UI)

frmDashboard

```
public static void main(String args[]) {
    try{
        UIManager.setLookAndFeel(new FlatLightLaf());
    }catch (Exception e){
        e.printStackTrace();
    }

    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new frmDashboard().setVisible(true);
        }
    });
```

Koneksi Package

(package to connect the local database)

koneksiDatabase

```java
package koneksi;

import java.sql.*;
import javax.swing.JOptionPane;
public class koneksiDatabase {
    Connection cn;
    public static Connection BukaKoneksi(){
        try{
            Class.forName("com.mysql.jdbc.Driver");
            Connection cn =
DriverManager.getConnection("jdbc:mysql://localhost/latihan_crud","root","");
            return cn;
        }catch (Exception e){
            JOptionPane.showMessageDialog(null, e);
            return null;
        }
    }
}
```

Make a connection

(snippet program to make a connection)

frmDashboard

```java
//-----------------------------------------------------
public Statement st;
public ResultSet rs;
Connection cn = connect.dbConnection.openConnection();
//-----------------------------------------------------
```

Cryptography Class Code

frmDashboard

```
//-----------------------CRYPTOGRAPHY CLASS-----------------------------------------------

public class SymmetricEncryption {
private SecretKeySpec secretKey;
private byte[] key;


public SymmetricEncryption(String myKey) {
    key = myKey.getBytes(StandardCharsets.UTF_8);
    secretKey = new SecretKeySpec(key, "AES");
}

public String encrypt(String strToEncrypt) {
    try {
        Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
        cipher.init(Cipher.ENCRYPT_MODE, secretKey);
        byte[] encryptedBytes =
cipher.doFinal(strToEncrypt.getBytes(StandardCharsets.UTF_8));
        return Base64.getEncoder().encodeToString(encryptedBytes);
    } catch (Exception e) {
        e.printStackTrace();
        JOptionPane.showMessageDialog(null, "There's a problem in the key or encryption");
    }
    return null;
}

public String decrypt(String strToDecrypt) {
    try {
        Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5PADDING");
        cipher.init(Cipher.DECRYPT_MODE, secretKey);
        byte[] decryptedBytes = cipher.doFinal(Base64.getDecoder().decode(strToDecrypt));
        return new String(decryptedBytes);
    } catch (Exception e) {
        e.printStackTrace();
        JOptionPane.showMessageDialog(null, "It is not the encrypted text");
    }
    return null;
}
}
//-----------------------------------------------------------------------------------------
```

Check key function

frmDashboard

```
private void checkKey(){
        if (txtKey.getText().equals("")){
    JOptionPane.showMessageDialog(null, "Set the keys first");
    }else{

        int keyLength = txtKey.getText().length();
        if(keyLength == 16 || keyLength == 24 || keyLength == 32){
            key_ameca = txtKey.getText();
            keyStat = true;

        }else{
            JOptionPane.showMessageDialog(null, "The key must be 16, 24 or 32 characters,
your key is "+keyLength+" characters");
            keyStat = false;
        }
    }
    return;
    }

private void Clear(){
txtInput.setText("");
txtOutput.setText("");
txtSender.setText("");
txtKey.setText("");


btnProcess.setText("PROCESS");
txtSender.setEditable(true);
txtKey.setEditable(true);
txtInput.setEditable(true);
txtOutput.setEditable(true);
}
```

Show Data function

```
private void ShowData(){
try{
    st = cn.createStatement();
    rs = st.executeQuery("SELECT * FROM dashboard");

    DefaultTableModel model = new DefaultTableModel();
    model.addColumn("No.");
    model.addColumn("SENDER");
    model.addColumn("KEY");
    model.addColumn("PLAIN TEXT");
    model.addColumn("CIPHER TEXT");

    int no = 1;

    model.getDataVector().removeAllElements();
    model.fireTableDataChanged();
    model.setRowCount(0);

    while (rs.next()){
        Object[] data = {
            no++,
            rs.getString("sender"),
            rs.getString("enKey"),
            rs.getString("plain"),
            rs.getString("cipher")
        };

        model.addRow(data);
        tblData.setModel(model);
    }

}catch (Exception e){

}
}
```

Find Data Function

```
private void FindData(){

    try{


        if (cmbSearch.getSelectedItem().toString().equals("Sender")){
            searchData = "sender";
        }else if (cmbSearch.getSelectedItem().toString().equals("Key")){
            searchData = "enKey";
        }else if (cmbSearch.getSelectedItem().toString().equals("Plain Text")){
            searchData = "plain";
        }

        st = cn.createStatement();
        rs = st.executeQuery("SELECT * FROM dashboard WHERE "
                +searchData+
                " LIKE '%"+ txtSearch.getText() + "%'");

        DefaultTableModel model = new DefaultTableModel();
        model.addColumn("No.");
        model.addColumn("SENDER");
        model.addColumn("KEY");
        model.addColumn("PLAIN TEXT");
        model.addColumn("CIPHER TEXT");

        int no = 1;

        model.getDataVector().removeAllElements();
        model.fireTableDataChanged();
        model.setRowCount(0);

        while (rs.next()){
            Object[] data = {
                no++,
                rs.getString("sender"),
                rs.getString("enKey"),
                rs.getString("plain"),
                rs.getString("cipher")
            };

            model.addRow(data);
            tblData.setModel(model);
        }

    }catch (Exception e){

}
}
```

Main Process

(create, read, use/change)

```
frmDashboard

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

        try{
            st = cn.createStatement();
            if (key_ameca.equals("")){
                checkKey();
            }else{
                if (txtSender.getText().equals("")){
                    checkKey();
                JOptionPane.showMessageDialog(null, "Sender data cannot be empty", "Sender
Data",JOptionPane.INFORMATION_MESSAGE);
                }else{

                    if (cmbMode.getSelectedItem().toString().equals("Encryption")){

                        checkKey();
                        if (txtInput.getText().equals("")){
                        JOptionPane.showMessageDialog(null, "Please input text", "Input
Text",JOptionPane.INFORMATION_MESSAGE);
                        return;
                        }else{
                            //aksi simpan data
                            if (btnProcess.getText() == "PROCESS"){

                                String key = key_ameca;
                                SymmetricEncryption encryption = new
SymmetricEncryption(key_ameca);
                                String encryptedText = encryption.encrypt(txtInput.getText());
                                txtOutput.setText(encryptedText);


                                String sql = "INSERT INTO dashboard VALUES ('" +
txtSender.getText() +
                                        "','" + txtKey.getText()+
                                        "','" + txtInput.getText()+
                                        "','" + txtOutput.getText()+ "')";
                                st.executeUpdate(sql);
                                JOptionPane.showMessageDialog(null, "Text Encrypted");

                                ShowData();
                            }else{
                                //aksi ubah data
                                String update = "UPDATE dashboard SET sender = '"+
txtSender.getText()+ "'"+" WHERE cipher = '"+txtOutput.getText()+"'";

                                st.executeUpdate(update);
                                JOptionPane.showMessageDialog(null, "Data changed");
                                Clear();
                                ShowData();

                            }

                        }
                    }
```

```
                else if (cmbMode.getSelectedItem().toString().equals("Decryption")){

                    checkKey();
                    if (txtInput.getText().equals("")){
                    JOptionPane.showMessageDialog(null, "Please input text", "Input
Text",JOptionPane.INFORMATION_MESSAGE);
                    return;
                    }else{
                        //aksi simpan data
                        if (btnProcess.getText() == "PROCESS"){

                            String key = key_ameca;
                            SymmetricEncryption encryption = new
SymmetricEncryption(key_ameca);
                            String decryptedText = encryption.decrypt(txtInput.getText());
                            txtOutput.setText(decryptedText);


                            String sql = "INSERT INTO dashboard VALUES ('" +
txtSender.getText() +
                                    "','" + txtKey.getText()+
                                    "','" + txtOutput.getText()+
                                    "','" + txtInput.getText()+ "')";
                            st.executeUpdate(sql);


                            ShowData();
                        }else{
                            //aksi ubah data
                            String update = "UPDATE dashboard SET sender = '"+
txtSender.getText()+ "'"+" WHERE cipher = '"+txtOutput.getText()+"'";

                            st.executeUpdate(update);
                            JOptionPane.showMessageDialog(null, "Data changed");
                            Clear();

                            ShowData();

                        }

                    }
                }

            }

        }catch (Exception e){
            JOptionPane.showMessageDialog(null, e);
        }
    }
```

Delete Process

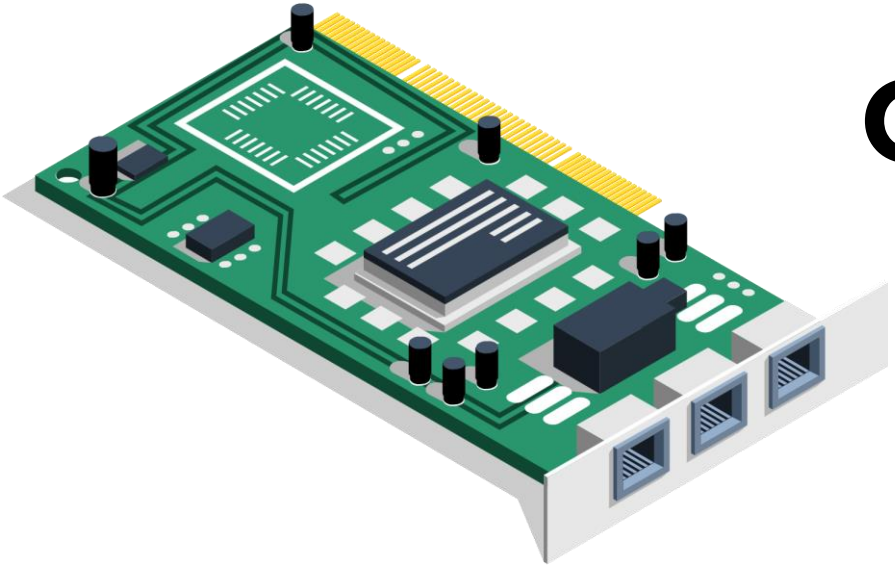
(delete)

frmDashboard

```
    private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:

        if(txtSender.getText().equals("")){
        JOptionPane.showMessageDialog(this, "Please select the data to be deleted");
        }else{
            int jawab = JOptionPane.showConfirmDialog(null, "This data will be deleted,
continue?", "CONFIRMATION", JOptionPane.YES_NO_OPTION);
            if (jawab == 0){
                try{
                    st = cn.createStatement();
                    String sql = "DELETE FROM dashboard WHERE cipher = '" +
txtOutput.getText()+ "'";
                    st.executeUpdate(sql);
                    JOptionPane.showMessageDialog(null, "Data deleted successfully");
                    ShowData();
                    Clear();

                }catch (Exception e){
                    JOptionPane.showMessageDialog(null, e);
                }
            }
        }
    }
```



Configuration

Minimum Configuration	
Hardware	Computer ; RAM: 8GB , ROM: 512, CPU: Intel i5
Operating System	Windows 10
Software	XAMPP V3.3.0,NetBeans IDE 13 (optional)

Thanks!