

XSEURE PROJECT

SECTION 373



Rawan Aljibreen

Razan Bajaaman

Noor Elmasry

Renad Al hussain

MAY 8, 2022
T. AL-JAWHARA AL-YOUSSEF

Table of Contents

- Overview of the project design..... 2
- Approach and steps to implementation..... 3
- Code 10
- Challenges 30
- References..... 31

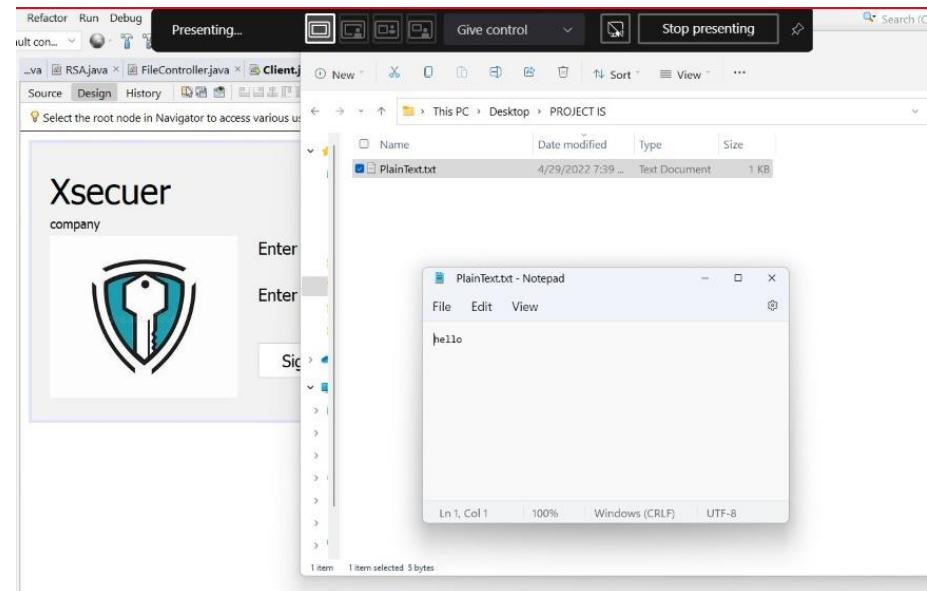
- Overview of the project design

Our project was designed to help users communicate and send files to each other securely. In our project we used java language to implement and design the interface. Our application provides secure registration and login services and we make sure that each user has their special username and password. We are following the public and private key approach by using both AES and RSA algorithms to encrypt and decrypt the files. Our system also has the ability to work via network.

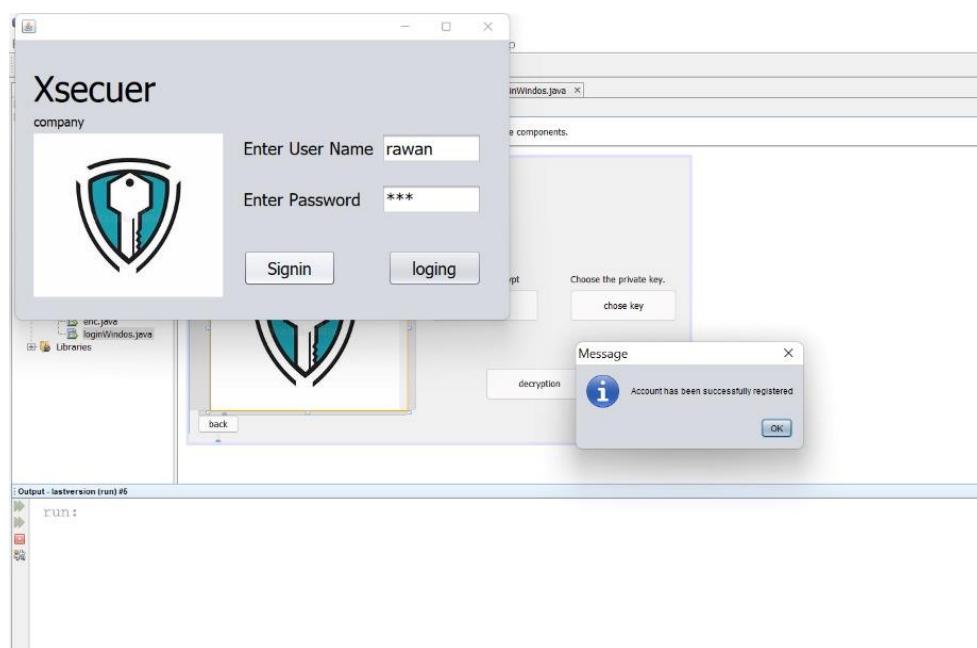


- Approach and steps to implementation

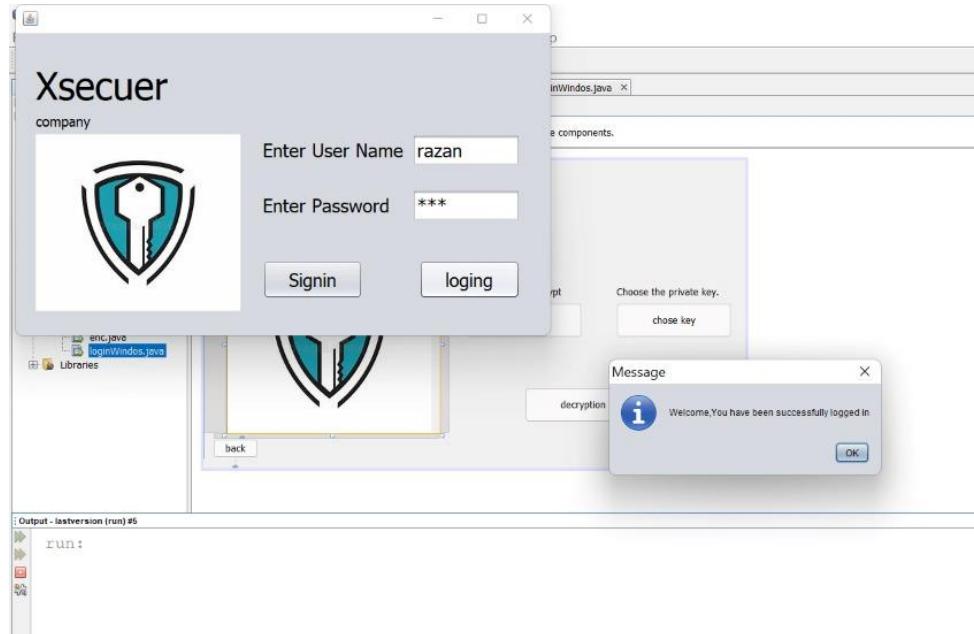
Step 1: write the text we want to send.



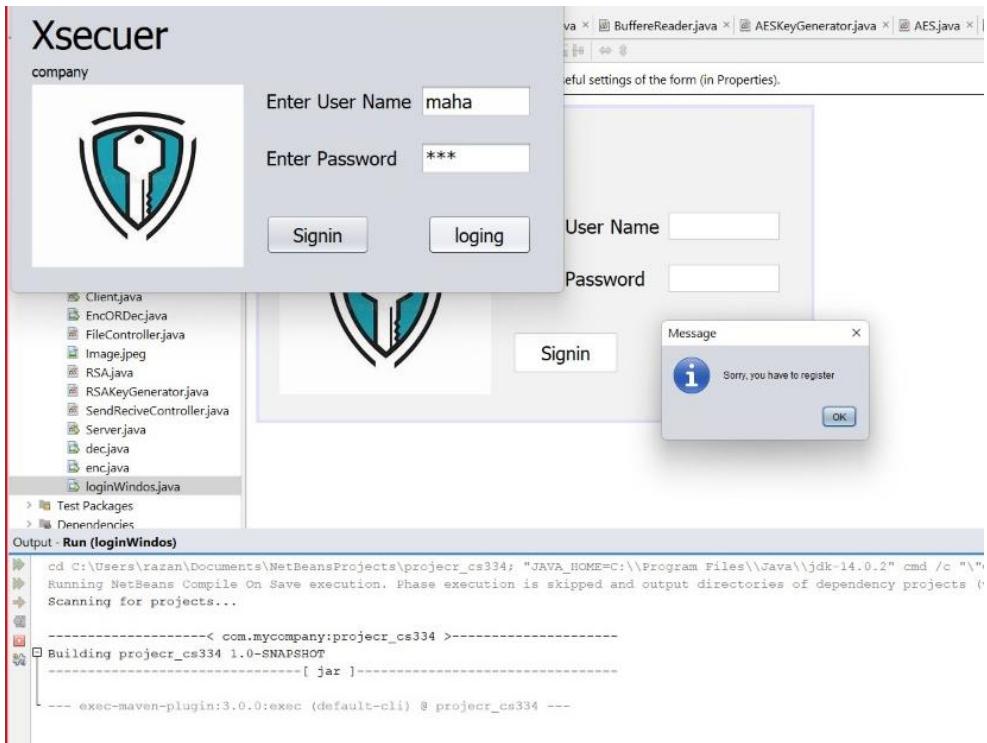
Step 2: user must register to the application.



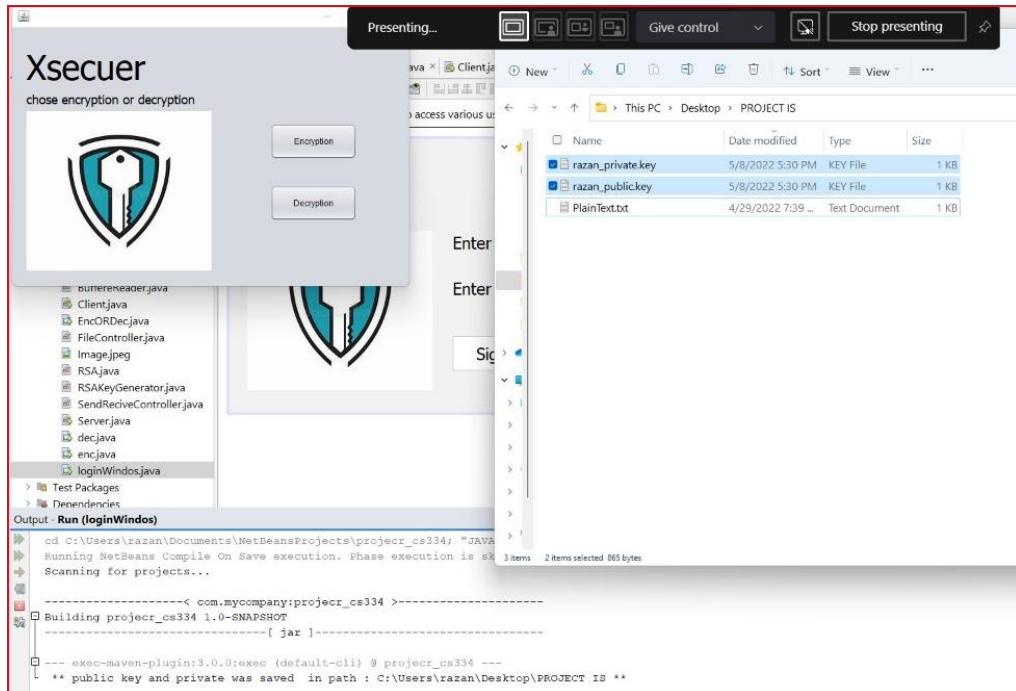
Step 3: after registration user can login to the application.



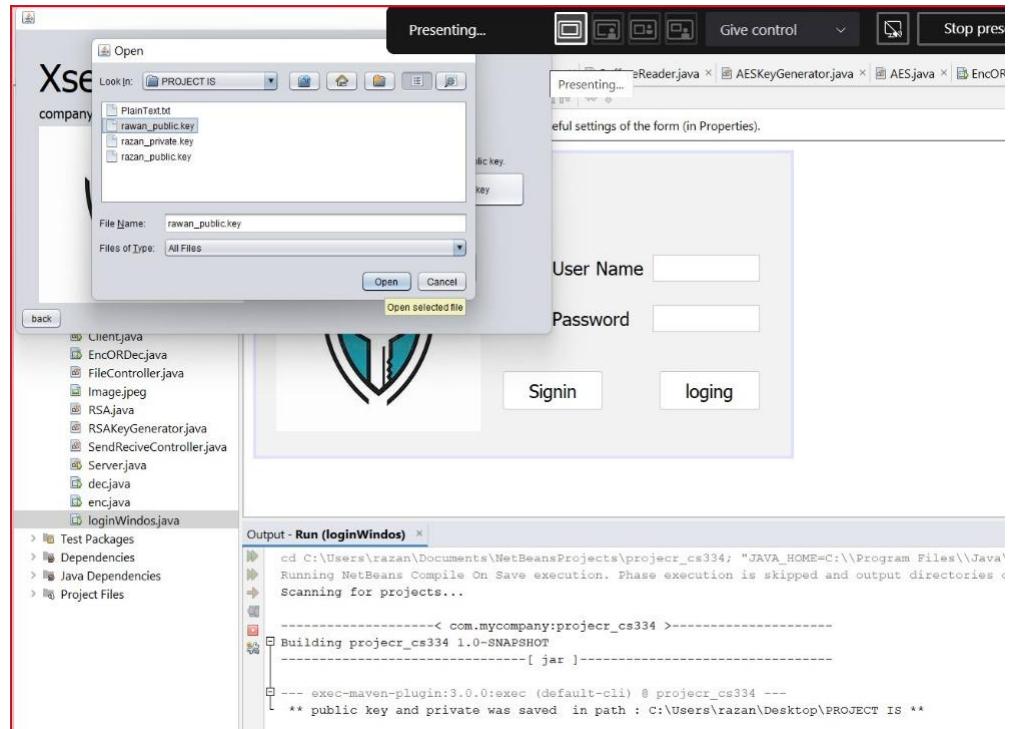
Step 4: if the user tries to login without registration, he will get an error message.



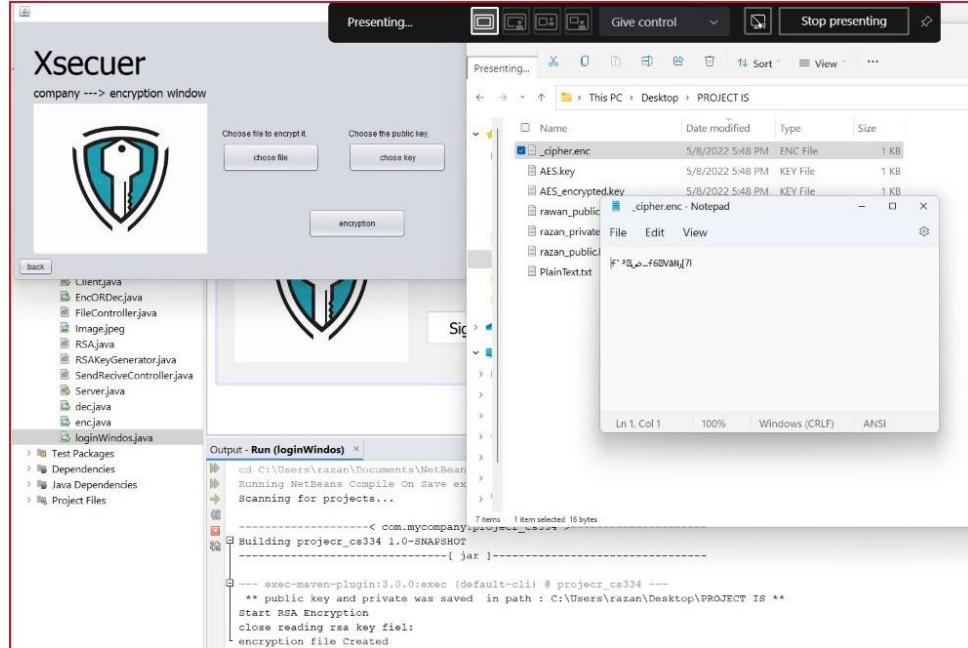
Step 5: the system have both public and private key techniques.



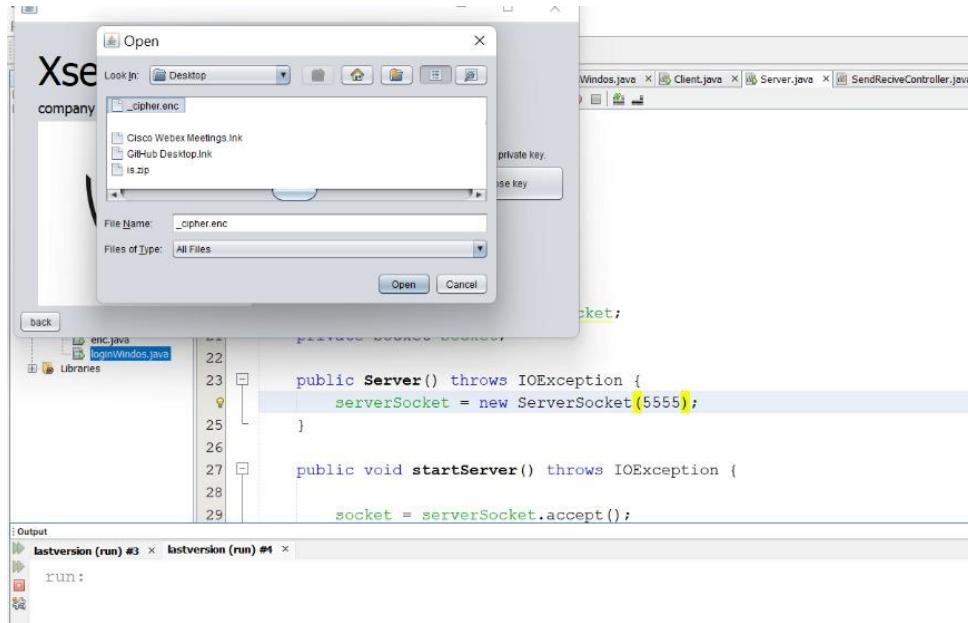
Step 6: the sender chose the public key to encrypt the file.



Step 7: the encrypted file.



Step 8: decrypting the file.



Step 9: after decrypting the file the system saved it for us.

The screenshot shows an IDE interface with two main panes. The left pane displays the Java code for a `Client` class:

```

public class Client {
    private Socket socket;
    public void startClient() {
        socket = new Socket();
    }
    public static void main(String[] args) {
        File file = new File("C:/Users/.../Desktop/PROJECT IS/CipherAfterDec.txt");
        Client client = new Client();
        client.startClient();
    }
}

```

The right pane shows a file explorer window titled "PROJECT IS" displaying files in the "This PC > Desktop > PROJECT IS" directory. The files listed are:

Name	Date modified	Type	Size
CipherAfterDec.txt	5/8/2022 6:38 PM	Text Document	1 KB
AES_decrypted.key	5/8/2022 6:38 PM	KEY File	1 KB
AES_encrypted.key	5/8/2022 6:38 PM	KEY File	0 KB
rawan_private.key	5/8/2022 6:38 PM	KEY File	1 KB
_cipher.enc	5/8/2022 5:48 PM	ENC File	1 KB
AES.key	5/8/2022 5:48 PM	KEY File	1 KB
rawan_publickey	5/8/2022 5:47 PM	KEY File	1 KB
razan_privatekey	5/8/2022 5:30 PM	KEY File	1 KB
razan_publickey	5/8/2022 5:30 PM	KEY File	1 KB
PlainText.txt	4/29/2022 7:39 PM	Text Document	1 KB

Below the file list, the output window shows the command executed and its results:

```

Building project_c334 1.0-SNAPSHOT
[jar] jar -cvf project_c334.jar .
[exec-maven-plugin:3.0.0:exec (default-cli) @ project_c334]
--- exec-maven-plugin:3.0.0:exec (default-cli) @ project_c334 --
Start RSA Decryption
close reading rsa key file:
decryption file created
AES file content:16
file content: F'0*fL..f6GvàNç?J8
The plaintext is :hello

```

server and client connection:

1. server

lastversion - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects lastversion Source Packages Source History Start Page Lastversion.java dec.java enc.java loginWindows.java Client.java Server.java SendRecive.java

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java
4   */
5  package pkg_is;
6
7  import java.io.File;
8  import java.io.IOException;
9  import java.net.ServerSocket;
10 import java.net.Socket;
11 import java.util.logging.Level;
12 import java.util.logging.Logger;
13
14 /**
15  *
16  * @author razan
17  */
18 public class Server {
```

Output lastversion (run) #6 × lastversion (run) #7 ×

```
run:
receiveFile method start ..
fileName : _cipher.enc
getFileExtention : enc
```

lastversion - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Projects lastversion Source Packages Source History Start Page Lastversion.java dec.java enc.java loginWindows.java Client.java Server.java SendRecive.java

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java
4   */
5  package pkg_is;
6
7  import java.util.logging.Level;
8  import java.util.logging.Logger;
9
10 import java.io.*;
11 import java.net.*;
12 import java.util.*;
13
14 /**
15  *
16  * @author razan
17  */
18 public class Server {
19
20     private ServerSocket serverSocket;
21     private Socket socket;
22
23     public Server() throws IOException {
24         serverSocket = new ServerSocket();
25     }
26
27     public void startServer() throws IOException {
28         socket = serverSocket.accept();
29     }
30 }
```

Output - lastversion (run) #3

```
run:
receiveFile method start ..
fileName : _cipher.enc
getFileExtention : enc
```

Desktop

This PC > Desktop >

Name	Date modified	Type	Size
_cipher.enc	11/11/2017 10:15 PM	ENC File	1 KB

22 items

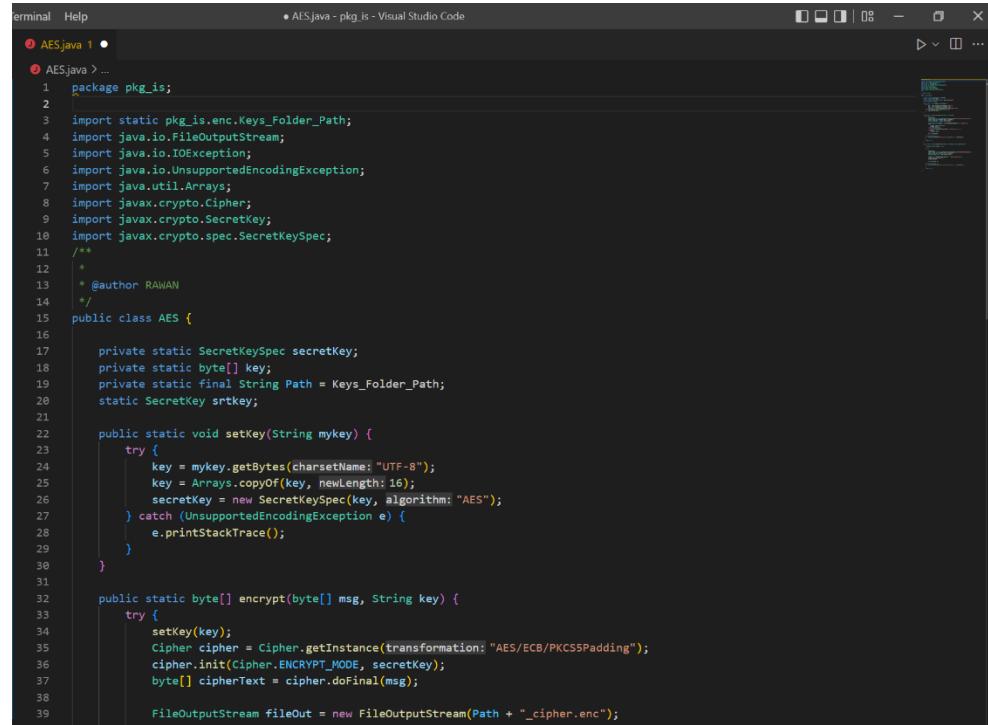
2. client

The screenshot shows an IDE interface with the following details:

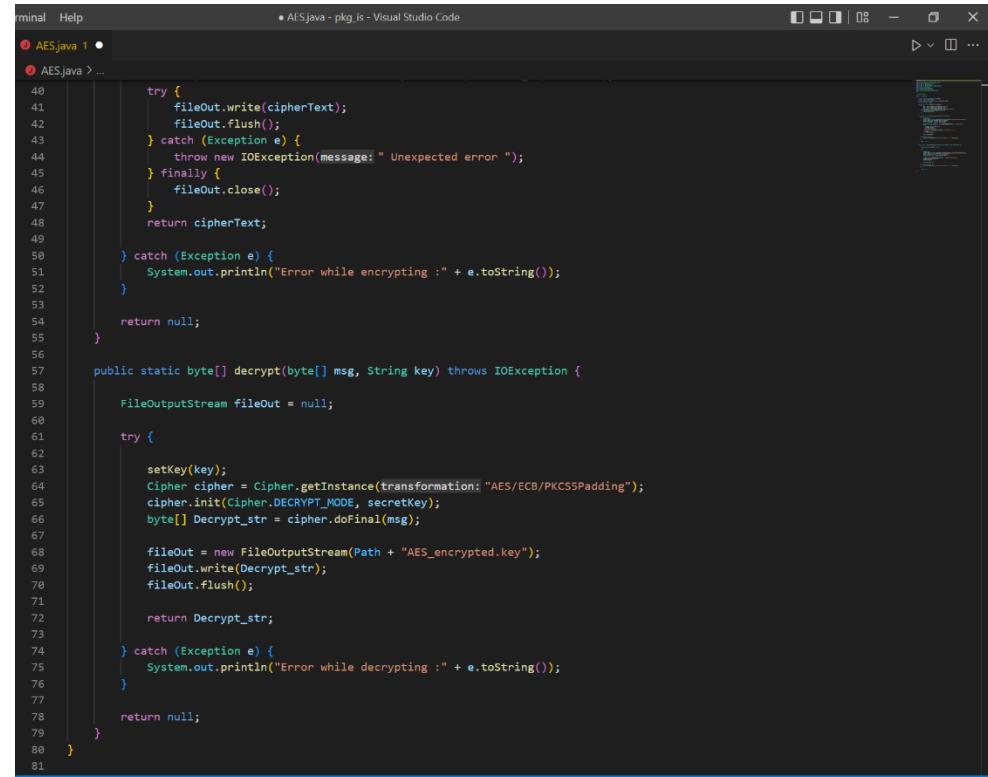
- Projects:** A tree view showing various Java files and packages, including `Client.java`, `SendReceiveController.java`, `RSAKeyGenerator.java`, `RSA.java`, `FileController.java`, and `Client.java`.
- Code Editor:** The `Client.java` file is open, displaying Java code for a `Client` class. The code includes methods `startClient()` and `main(String[] args)`. The `main` method attempts to connect to a socket at "192.168.1.26", port 8888, and reads from a file named `Desktop_cipher.enc`.
- Output - Run (Client) :** This panel shows the Maven build logs for the project `project_cs334`. It indicates a successful build with a total time of 0.722 seconds and a finished date of 2022-05-08T17:50:46+03:00.

○ Code

1. AES

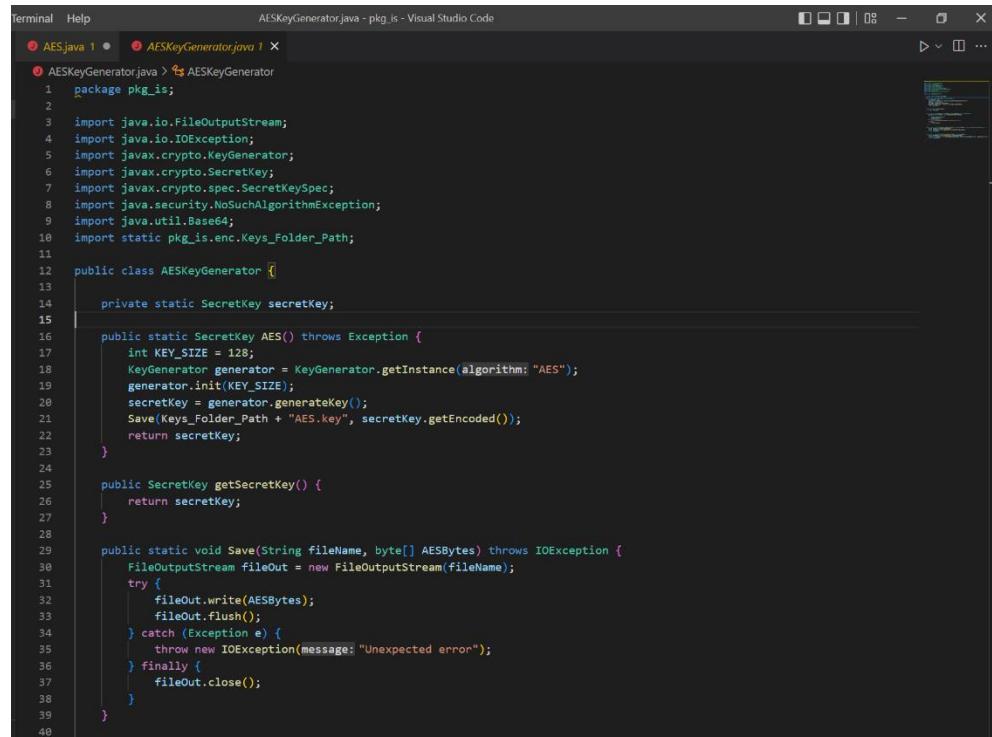


```
terminal Help • AES.java - pkg_is - Visual Studio Code
AES.java 1 ●
AES.java > ...
1 package pkg_is;
2
3 import static pkg_is.Keys_Folder_Path;
4 import java.io.FileOutputStream;
5 import java.io.IOException;
6 import java.io.UnsupportedEncodingException;
7 import java.util.Arrays;
8 import javax.crypto.Cipher;
9 import javax.crypto.SecretKey;
10 import javax.crypto.spec.SecretKeySpec;
11 /**
12 * 
13 * @author RAWAN
14 */
15 public class AES {
16
17     private static SecretKeySpec secretKey;
18     private static byte[] key;
19     private static final String Path = Keys_Folder_Path;
20     static SecretKey srkey;
21
22     public static void setKey(String mykey) {
23         try {
24             key = mykey.getBytes(charsetName: "UTF-8");
25             key = Arrays.copyOf(key, newLength: 16);
26             secretKey = new SecretKeySpec(key, algorithm: "AES");
27         } catch (UnsupportedEncodingException e) {
28             e.printStackTrace();
29         }
30     }
31
32     public static byte[] encrypt(byte[] msg, String key) {
33         try {
34             setKey(key);
35             Cipher cipher = Cipher.getInstance(transformation: "AES/ECB/PKCS5Padding");
36             cipher.init(Cipher.ENCRYPT_MODE, secretKey);
37             byte[] cipherText = cipher.doFinal(msg);
38
39             FileOutputStream fileOut = new FileOutputStream(Path + "_cipher.enc");
40
41             fileOut.write(cipherText);
42             fileOut.flush();
43         } catch (new IOException(message: " Unexpected error "));
44         finally {
45             fileOut.close();
46         }
47         return cipherText;
48     }
49
50     catch (Exception e) {
51         System.out.println("Error while encrypting :" + e.toString());
52     }
53
54     return null;
55 }
56
57     public static byte[] decrypt(byte[] msg, String key) throws IOException {
58
59     FileOutputStream fileOut = null;
60
61     try {
62
63         setKey(key);
64         Cipher cipher = Cipher.getInstance(transformation: "AES/ECB/PKCS5Padding");
65         cipher.init(Cipher.DECRYPT_MODE, secretKey);
66         byte[] Decrypt_str = cipher.doFinal(msg);
67
68         fileOut = new FileOutputStream(Path + "AES_encrypted.key");
69         fileOut.write(Decrypt_str);
70         fileOut.flush();
71
72         return Decrypt_str;
73
74     } catch (Exception e) {
75         System.out.println("Error while decrypting :" + e.toString());
76     }
77
78     return null;
79 }
80 }
```

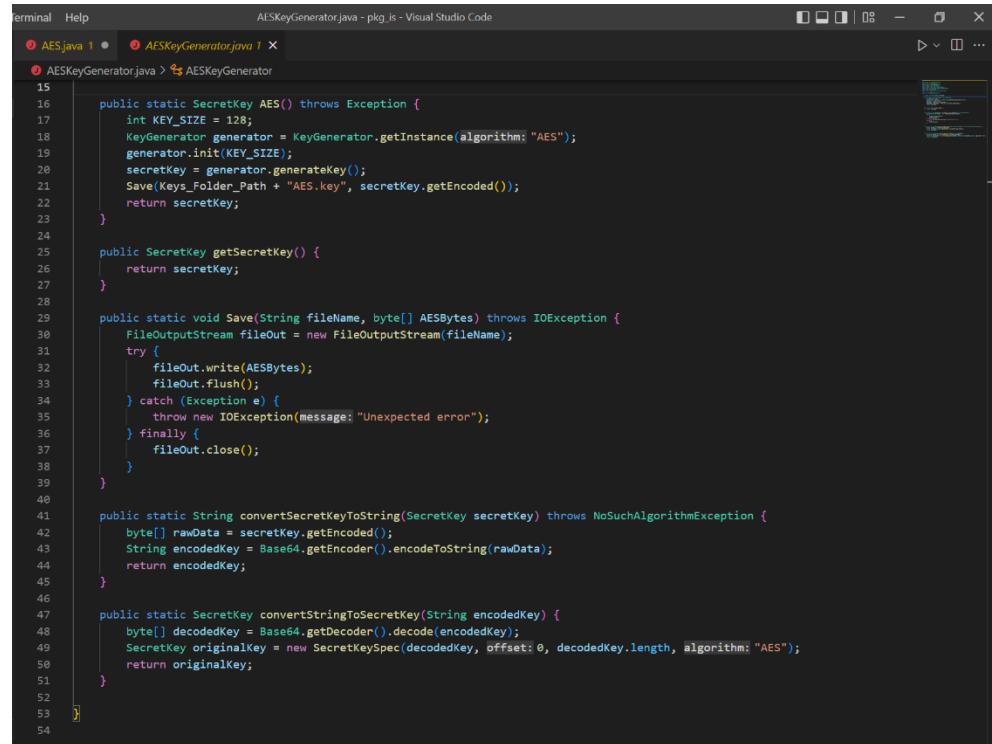


```
terminal Help • AES.java - pkg_is - Visual Studio Code
AES.java 1 ●
AES.java > ...
40
41         fileOut.write(cipherText);
42         fileOut.flush();
43     } catch (new IOException(message: " Unexpected error "));
44     finally {
45         fileOut.close();
46     }
47     return cipherText;
48 }
49
50     catch (Exception e) {
51         System.out.println("Error while encrypting :" + e.toString());
52     }
53
54     return null;
55 }
56
57     public static byte[] decrypt(byte[] msg, String key) throws IOException {
58
59     FileOutputStream fileOut = null;
60
61     try {
62
63         setKey(key);
64         Cipher cipher = Cipher.getInstance(transformation: "AES/ECB/PKCS5Padding");
65         cipher.init(Cipher.DECRYPT_MODE, secretKey);
66         byte[] Decrypt_str = cipher.doFinal(msg);
67
68         fileOut = new FileOutputStream(Path + "AES_encrypted.key");
69         fileOut.write(Decrypt_str);
70         fileOut.flush();
71
72         return Decrypt_str;
73
74     } catch (Exception e) {
75         System.out.println("Error while decrypting :" + e.toString());
76     }
77
78     return null;
79 }
80 }
```

2. AESKeyGenerator

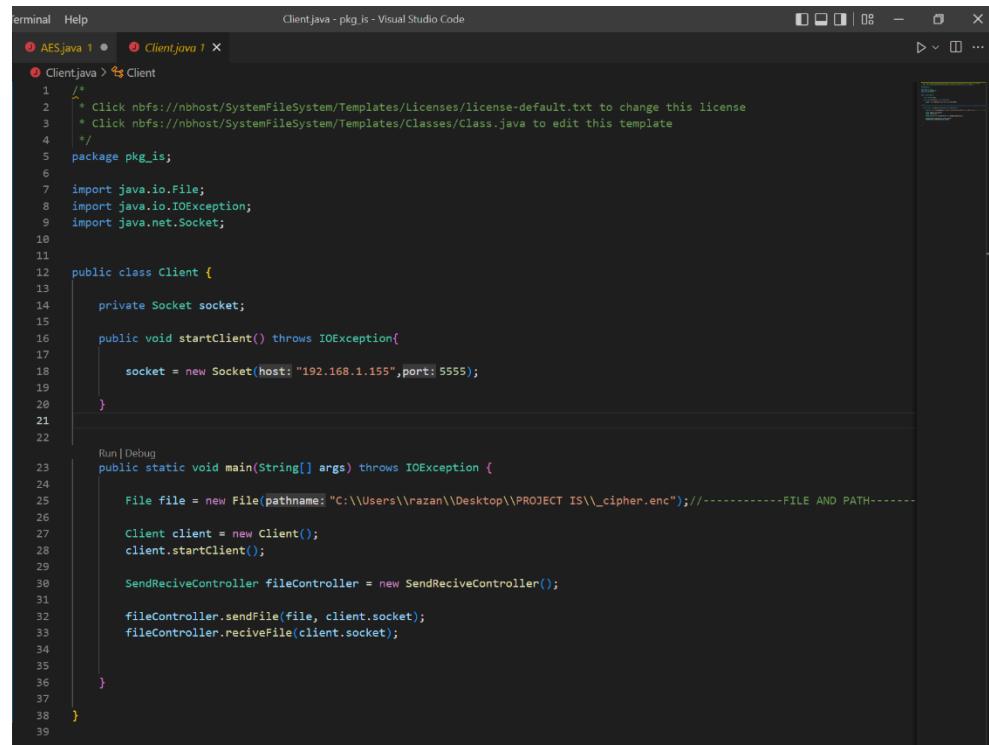


```
Terminal Help AESKeyGenerator.java - pkg_is - Visual Studio Code
AES.java 1 AESKeyGenerator.java 1
AESKeyGenerator.java > AESKeyGenerator
1 package pkg_is;
2
3 import java.io.FileOutputStream;
4 import java.io.IOException;
5 import javax.crypto.KeyGenerator;
6 import javax.crypto.SecretKey;
7 import javax.crypto.spec.SecretKeySpec;
8 import java.security.NoSuchAlgorithmException;
9 import java.util.Base64;
10 import static pkg_is.enc.Keys_Folder_Path;
11
12 public class AESKeyGenerator {
13
14     private static SecretKey secretKey;
15
16     public static SecretKey AES() throws Exception {
17         int KEY_SIZE = 128;
18         KeyGenerator generator = KeyGenerator.getInstance(algorithm: "AES");
19         generator.init(KEY_SIZE);
20         secretKey = generator.generateKey();
21         Save(Keys_Folder_Path + "AES.key", secretKey.getEncoded());
22         return secretKey;
23     }
24
25     public SecretKey getSecretKey() {
26         return secretKey;
27     }
28
29     public static void Save(String fileName, byte[] AESBytes) throws IOException {
30         FileOutputStream fileOut = new FileOutputStream(fileName);
31         try {
32             fileOut.write(AESBytes);
33             fileOut.flush();
34         } catch (Exception e) {
35             throw new IOException(message: "Unexpected error");
36         } finally {
37             fileOut.close();
38         }
39     }
40
41 }
```



```
Terminal Help AESKeyGenerator.java - pkg_is - Visual Studio Code
AES.java 1 AESKeyGenerator.java 1
AESKeyGenerator.java > AESKeyGenerator
15
16     public static SecretKey AES() throws Exception {
17         int KEY_SIZE = 128;
18         KeyGenerator generator = KeyGenerator.getInstance(algorithm: "AES");
19         generator.init(KEY_SIZE);
20         secretKey = generator.generateKey();
21         Save(Keys_Folder_Path + "AES.key", secretKey.getEncoded());
22         return secretKey;
23     }
24
25     public SecretKey getSecretKey() {
26         return secretKey;
27     }
28
29     public static void Save(String fileName, byte[] AESBytes) throws IOException {
30         FileOutputStream fileOut = new FileOutputStream(fileName);
31         try {
32             fileOut.write(AESBytes);
33             fileOut.flush();
34         } catch (Exception e) {
35             throw new IOException(message: "Unexpected error");
36         } finally {
37             fileOut.close();
38         }
39     }
40
41     public static String convertSecretKeyToString(SecretKey secretKey) throws NoSuchAlgorithmException {
42         byte[] rawData = secretKey.getEncoded();
43         String encodedKey = Base64.getEncoder().encodeToString(rawData);
44         return encodedKey;
45     }
46
47     public static SecretKey convertStringToSecretKey(String encodedKey) {
48         byte[] decodedKey = Base64.getDecoder().decode(encodedKey);
49         SecretKey originalKey = new SecretKeySpec(decodedKey, offset: 0, decodedKey.length, algorithm: "AES");
50         return originalKey;
51     }
52
53 }
54
```

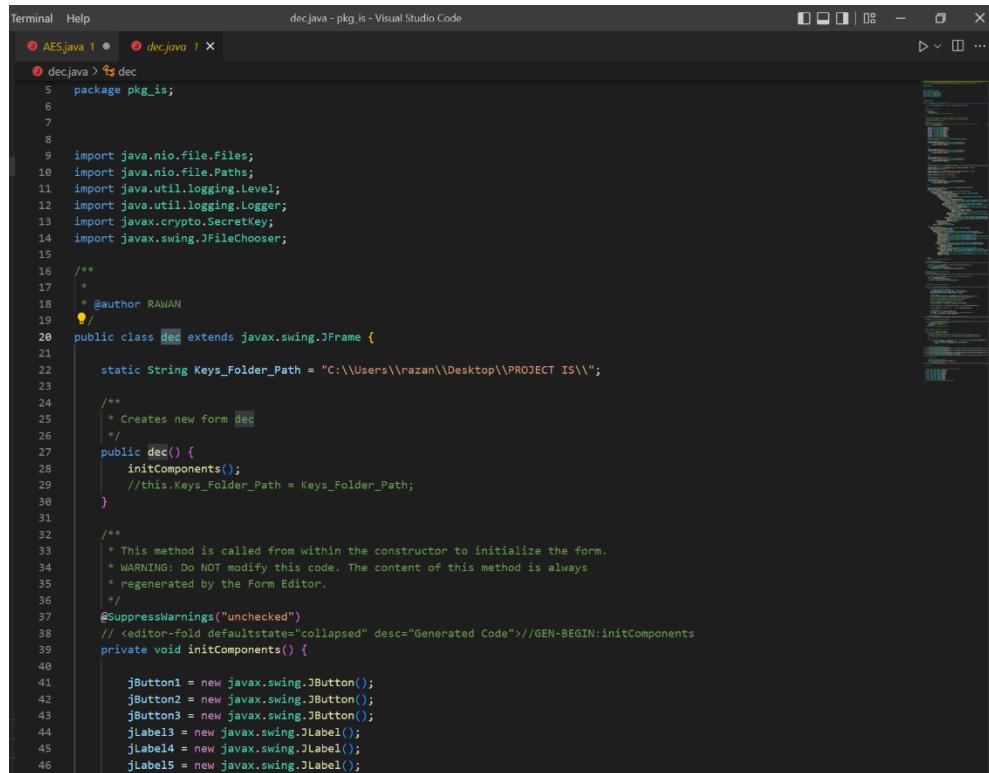
3. Client



A screenshot of the Visual Studio Code interface showing a Java file named `Client.java`. The code implements a client socket to connect to a host at port 5555. It also includes a main method to handle file transfer between the client and server.

```
terminal Help
① AES.java 1 ● ② Client.java 1 ✘
Client.java > 🏃 Client
1 /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5 package pkg_is;
6
7 import java.io.File;
8 import java.io.IOException;
9 import java.net.Socket;
10
11 public class Client {
12
13     private Socket socket;
14
15     public void startClient() throws IOException{
16
17         socket = new Socket(host: "192.168.1.155",port: 5555);
18     }
19
20
21
22
Run|Debug
23 public static void main(String[] args) throws IOException {
24
25     File file = new File(pathname: "C:\\\\Users\\\\razan\\\\Desktop\\\\PROJECT IS\\\\cipher.enc");//-----FILE AND PATH-----
26
27     Client client = new Client();
28     client.startClient();
29
30     SendReceiveController fileController = new SendReceiveController();
31
32     fileController.sendFile(file, client.socket);
33     fileController.receiveFile(client.socket);
34
35
36
37
38
}
39
```

4. Dec



A screenshot of the Visual Studio Code interface showing a Java file named `dec.java`. The code defines a `JFrame` window titled "dec" with several buttons and labels. It includes annotations for file paths and logging levels.

```
terminal Help
① AES.java 1 ● ② dec.java 1 ✘
dec.java > 🏃 dec
5 package pkg_is;
6
7
8
9 import java.nio.file.Files;
10 import java.nio.file.Paths;
11 import java.util.logging.Level;
12 import java.util.logging.Logger;
13 import javax.crypto.SecretKey;
14 import javax.swing.JFileChooser;
15
16 /**
17 *
18 * @author RAWAN
19 */
20 public class dec extends javax.swing.JFrame {
21
22     static String Keys_Folder_Path = "C:\\\\Users\\\\razan\\\\Desktop\\\\PROJECT IS\\\\";
23
24     /**
25      * Creates new form dec
26      */
27     public dec() {
28         initComponents();
29         //this.Keys_Folder_Path = Keys_Folder_Path;
30     }
31
32     /**
33      * This method is called from within the constructor to initialize the form.
34      * WARNING: Do NOT modify this code. The content of this method is always
35      * regenerated by the Form Editor.
36      */
37     @SuppressWarnings("unchecked")
38     // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
39     private void initComponents() {
40
41         jButton1 = new javax.swing.JButton();
42         jButton2 = new javax.swing.JButton();
43         jButton3 = new javax.swing.JButton();
44         jLabel13 = new javax.swing.JLabel();
45         jLabel14 = new javax.swing.JLabel();
46         jLabel15 = new javax.swing.JLabel();
47
48     }
49
50 }
```



```

119         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
120             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 150, javax.swing.GroupLayout.PREFERRED_SIZE)
121             .addGap(min: 119, pref: 119, max: 119)))
122     .addGroup(layout.createSequentialGroup()
123         .addContainerGap()
124         .addComponent(jButton4))
125     .addContainerGap(pref: 18, Short.MAX_VALUE)
126 );
127 layout.setVerticalGroup(
128     layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
129         .addGroup(layout.createSequentialGroup()
130             .addContainerGap(pref: 29, Short.MAX_VALUE)
131             .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
132                 .addGroup(layout.createSequentialGroup()
133                     .addComponent(jLabel4)
134                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
135                     .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 22, javax.swing.GroupLayout.PREFERRED_SIZE)
136                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
137                     .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 230, javax.swing.GroupLayout.PREFERRED_SIZE)
138                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
139                     .addComponent(jButton4)
140                     .addContainerGap())
141                 .addGroup(layout.createSequentialGroup()
142                     .addComponent(jLabel1)
143                     .addComponent(jLabel2)
144                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
145                     .addComponent(jLabel11)
146                     .addComponent(jLabel12)
147                     .addPreferredGap(javax.swing.GroupLayout.Alignment.BASELINE)
148                     .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 45, javax.swing.GroupLayout.PREFERRED_SIZE)
149                     .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 45, javax.swing.GroupLayout.PREFERRED_SIZE)
150                     .addGap(min: 65, pref: 65, max: 65)
151                     .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 45, javax.swing.GroupLayout.PREFERRED_SIZE)
152                     .addGap(min: 60, pref: 60, max: 60)))
153             )
154         )
155     )
156 // </editor-fold>GEN-END:initComponents
157 String Encrypted_File_Path = "";
158 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {GEN-FIRST:event_jButton1ActionPerformed
159 JFileChooser j = new JFileChooser(currentDirectoryPath: "C:\\\\Users\\\\razaan\\\\Desktop\\\\PROJECT IS\\\\");
160

```

```

160 JFileChooser j = new JFileChooser(currentDirectoryPath: "C:\\\\Users\\\\razaan\\\\Desktop\\\\PROJECT IS\\\\");
161 int r = j.showOpenDialog(this);
162 if (r == JFileChooser.APPROVE_OPTION) {
163     Encrypted_File_Path = j.getSelectedFile().getAbsolutePath();
164 }
165 }GEN-LAST:event_jButton1ActionPerformed
166 String Key_Path = "";
167 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {GEN-FIRST:event_jButton2ActionPerformed
168 JFileChooser j = new JFileChooser(currentDirectoryPath: "C:\\\\Users\\\\razaan\\\\Desktop\\\\PROJECT IS\\\\");
169 int r = j.showOpenDialog(this);
170 if (r == JFileChooser.APPROVE_OPTION) {
171     Key_Path = j.getSelectedFile().getAbsolutePath();
172 }
173
174 }GEN-LAST:event_jButton2ActionPerformed
175
176 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {GEN-FIRST:event_jButton3ActionPerformed
177 try {
178     // TODO add your handling code here:
179     String AESKEY_encrypted = Keys_Folder_Path + "AES_encrypted.key";
180     String AESKEY_decrypted = Keys_Folder_Path + "AES_decrypted.key";
181     RSA.Decrypt(AESKEY_encrypted, AESKEY_decrypted, Key_Path);
182
183     //read aes key file
184     SecretKey AESKey = FileController.readAESKey(Keys_Folder_Path + "AES_decrypted.key");
185     //String AE = FileController.read(Keys_Folder_Path + "AES_decrypted.key");
186     //Secretkey AESKey = AESKeyGenerator.convertStringToSecretKey(AE);
187     //now we want to decrypt cipher text we read encrypted file
188     //String read=FileController.read(Encrypted_File_Path);
189     //byte[] encryptedFile = read.getBytes();
190     byte[] encryptedFile = FileController.read(Encrypted_File_Path);
191
192     byte[] decryptedString = AES.decrypt(encryptedFile, new String(AESKey.getEncoded()));
193
194     System.out.println("The plaintext is : " + new String(decryptedString));
195
196     // save the cipher
197     String content = new String(decryptedString);
198     String path = "C:\\\\Users\\\\razaan\\\\Desktop\\\\PROJECT IS\\\\ CipherAfterDec.txt";
199
200 }
```

```
Terminal Help decjava - pkg_is - Visual Studio Code
AES.java 1 decjava 1
decjava > dec
281     String path = "C:\\\\Users\\\\rakan\\\\Desktop\\\\PROJECT IS\\\\ CipherAfterDec.txt";
282     Files.write( Paths.get(path), content.getBytes());
283
284 } catch (Exception ex) {
285     Logger.getLogger(dec.class.getName()).log(Level.SEVERE, msg: null, ex);
286 }
287 } //GEN-LAST:event_jButton3ActionPerformed
288
289 private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_jButton4ActionPerformed
290     // TODO add your handling code here:
291     new EncORDec().setVisible(true);
292     dispose();
293 } //GEN-LAST:event_jButton4ActionPerformed
294
295 /**
296 * @param args the command line arguments
297 */
298 Run | Debug
299 public static void main(String args[]) {
300     /* Set the Nimbus look and feel */
301     // <editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
302     /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
303      * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html 
```

```
Terminal Help decjava - pkg_is - Visual Studio Code
AES.java 1 decjava 1
decjava > dec
224 try {
225     for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
226         if ("Nimbus".equals(info.getName())) {
227             javax.swing.UIManager.setLookAndFeel(info.getClassName());
228             break;
229         }
230     }
231 } catch (ClassNotFoundException ex) {
232     java.util.logging.Logger.getLogger(dec.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
233 } catch (InstantiationException ex) {
234     java.util.logging.Logger.getLogger(dec.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
235 } catch (IllegalAccessException ex) {
236     java.util.logging.Logger.getLogger(dec.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
237 } catch (javaw.swing.UnsupportedLookAndFeelException ex) {
238     java.util.logging.Logger.getLogger(dec.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
239 }
240 //
```

5. Enc

The screenshot shows two tabs open in Visual Studio Code: 'AES.java 2' and 'enc.java 1'. The 'enc.java' tab is active, displaying Java code for a Swing application. The code includes imports for java.util.logging, javax.crypto, javax.swing, and javax.swing.filechooser. It defines a class 'enc' that extends javax.swing.JFrame. The constructor initializes the form and calls 'initComponents()'. The 'initComponents()' method creates various UI components like JButton, JLabel, and JFileChooser. A note in the code states: '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.' The code is annotated with Javadoc-style comments and includes suppression annotations for unchecked warnings.

```
package pkg_is;

import java.util.logging.Level;
import java.util.logging.Logger;
import javax.crypto.SecretKey;
import javax.swing.JFileChooser;
import javax.swing.filechooser.FileSystemView;

/**
 * 
 * @author RAWAN
 */
public class enc extends javax.swing.JFrame {

    static String Keys_Folder_Path = "C:\\Users\\razan\\Desktop\\PROJECT IS\\";

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

    // this.Keys_Folder_Path = Keys_Folder_Path;
    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() {

    jButton1 = new javax.swing.JButton();
    jButton2 = new javax.swing.JButton();
    jButton3 = new javax.swing.JButton();
    jLabel13 = new javax.swing.JLabel();
    jLabel14 = new javax.swing.JLabel();
    jLabel15 = new javax.swing.JLabel();
    jLabel11 = new javax.swing.JLabel();
    jLabel12 = new javax.swing.JLabel();
}
```

The screenshot shows a Java application titled "enc.java" open in Visual Studio Code. The code implements a GUI with several buttons and labels. It includes logic for file selection, key entry, and encryption. The code uses Java's Swing library for the UI and AWT for event handling.

```
46 jLabel2 = new javax.swing.JLabel();
47 jButton4 = new javax.swing.JButton();
48 
49 setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
50 
51 jButton1.setText("choose file");
52 jButton1.addActionListener(new java.awt.event.ActionListener() {
53     public void actionPerformed(java.awt.event.ActionEvent evt) {
54         jButton1ActionPerformed(evt);
55     }
56 });
57 
58 jButton2.setText("choose key");
59 jButton2.addActionListener(new java.awt.event.ActionListener() {
60     public void actionPerformed(java.awt.event.ActionEvent evt) {
61         jButton2ActionPerformed(evt);
62     }
63 });
64 
65 jButton3.setText("encryption");
66 jButton3.addActionListener(new java.awt.event.ActionListener() {
67     public void actionPerformed(java.awt.event.ActionEvent evt) {
68         jButton3ActionPerformed(evt);
69     }
70 });
71 
72 jLabel3.setIcon(new javax.swing.ImageIcon(filename: "C:\\Users\\razan\\Desktop\\WhatsApp Image 2022-05-08 at 4.16.47 PM"));
73 
74 jLabel4.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 50)); // NOI18N
75 jLabel4.setText("Xsecuer");
76 
77 jLabel5.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 18)); // NOI18N
78 jLabel5.setText("company --> encryption window");
79 
80 jLabel1.setText("Choose file to encrypt it.");
81 
82 jLabel2.setText("Choose the public key.");
83 
84 jButton4.setText("back");
85 jButton4.addActionListener(new java.awt.event.ActionListener() {
86     public void actionPerformed(java.awt.event.ActionEvent evt) {
87         jButton4ActionPerformed(evt);
88     }
89 });

The right side of the screen shows the standard Visual Studio Code interface with a sidebar, status bar, and a floating terminal window.
```

```
Terminal Help encjava - pkg is - Visual Studio Code

AES.java 2  encjava 1 x
encjava > enc

128     .addGap(min: 31, pref: 31, max: 31)
129     .addComponent(jLabel14)
130     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
131     .addComponent(jLabel15, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 22, javax.swing.GroupLayout.PREFERRED_SIZE)
132     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
133     .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
134         .addGroup(layout.createSequentialGroup()
135             .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
136                 .addComponent(jLabel12)
137                 .addComponent(jLabel11))
138             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
139             .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
140                 .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 46, javax.swing.GroupLayout.PREFERRED_SIZE)
141                 .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 45, javax.swing.GroupLayout.PREFERRED_SIZE)
142             .addGap(min: 56, pref: 56, max: 56)
143             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 45, javax.swing.GroupLayout.PREFERRED_SIZE)
144             .addGap(min: 24, pref: 24, max: 24)
145             .addComponent(jLabel13, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 230, javax.swing.GroupLayout.PREFERRED_SIZE)
146             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
147             .addComponent(jButton4)
148             .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
149     );
150
151     pack();
152 } // </editor-fold>/GEN-END:initComponents
153 String FilePath = "";
154 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {GEN-FIRST:event_jButton1ActionPerformed
155
156     JFileChooser j = new JFileChooser(currentDirectoryPath: "C:\\\\Users\\\\razan\\\\Desktop\\\\PROJECT IS\\\\");
157     int r = j.showOpenDialog(this);
158     if (r == JFileChooser.APPROVE_OPTION) {
159         FilePath = j.getSelectedFile().getAbsolutePath();
160     }
161 }
162 }//GEN-LAST:event_jButton1ActionPerformed
163 String KeyPath = "";
164 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {GEN-FIRST:event_jButton2ActionPerformed
165
166     JFileChooser j = new JFileChooser(currentDirectoryPath: "C:\\\\Users\\\\razan\\\\Desktop\\\\PROJECT IS\\\\");
167     int r = j.showOpenDialog(this);
168     if (r == JFileChooser.APPROVE_OPTION) {
169         KeyPath = j.getSelectedFile().getAbsolutePath();
170     }
171 }
```

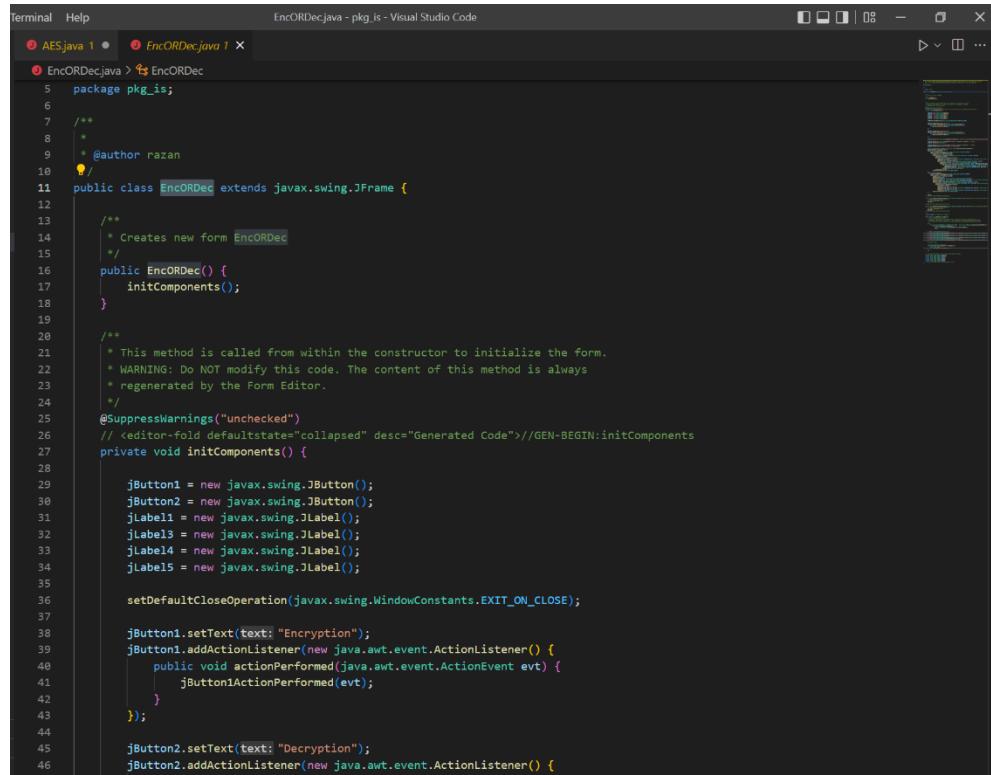
A screenshot of the Visual Studio Code interface. The title bar shows "AES.java 2" and "enc.java 1". The main editor area displays Java code for AES encryption. The code includes methods for generating AES keys, reading files, and encrypting data using RSA. It also handles exceptions and logs errors using Java's logging framework. The code is annotated with comments and TODO items.

```
169     KeyPath = j.getSelectedFile().getAbsolutePath();
170 }
171 } //GEN-LAST:event_jButton2ActionPerformed
172
173 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_jButton3ActionPerformed
174     try {
175
176         SecretKey SecretKey = AESKeyGenerator.AES();
177
178         String read=fileController.read(filePath);
179         byte[] plainFile =read.getBytes();
180
181         AES.encrypt(plainFile, new String(SecretKey.getEncoded()));
182
183
184         // byte[] publicKey= readPublicKey(File_KeyPath);
185         String AESKey_=keys_Folder_Path + "AES.key";
186         String AESKey_encryptedkey = Keys_Folder_Path + "AES_encrypted.key";
187         RSA.Encrypt( AESKey, AESKey_encryptedkey, KeyPath);
188
189
190     } catch (Exception ex) {
191         Logger.getLogger(enc.class.getName()).log(Level.SEVERE, null, ex);
192     }
193
194
195 } //GEN-LAST:event_jButton3ActionPerformed
196
197 private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_jButton4ActionPerformed
198     // TODO add your handling code here
199     new EncORDec().setVisible(true);
200     dispose();
201 } //GEN-LAST:event_jButton4ActionPerformed
202
203 /**
204 * @param args the command line arguments
205 */
206 Run | Debug
207 public static void main(String args[]) {
208     /* Set the Nimbus look and feel */
209 }
```

A screenshot of the Visual Studio Code interface. The title bar shows "AES.java 2" and "enc.java 7". The main editor area displays Java code for setting up a Swing UI. It includes logic to determine the installed LookAndFeel and set it accordingly. It also creates and displays a form using Java's EventQueue.invokeLater. The code is annotated with comments and TODO items.

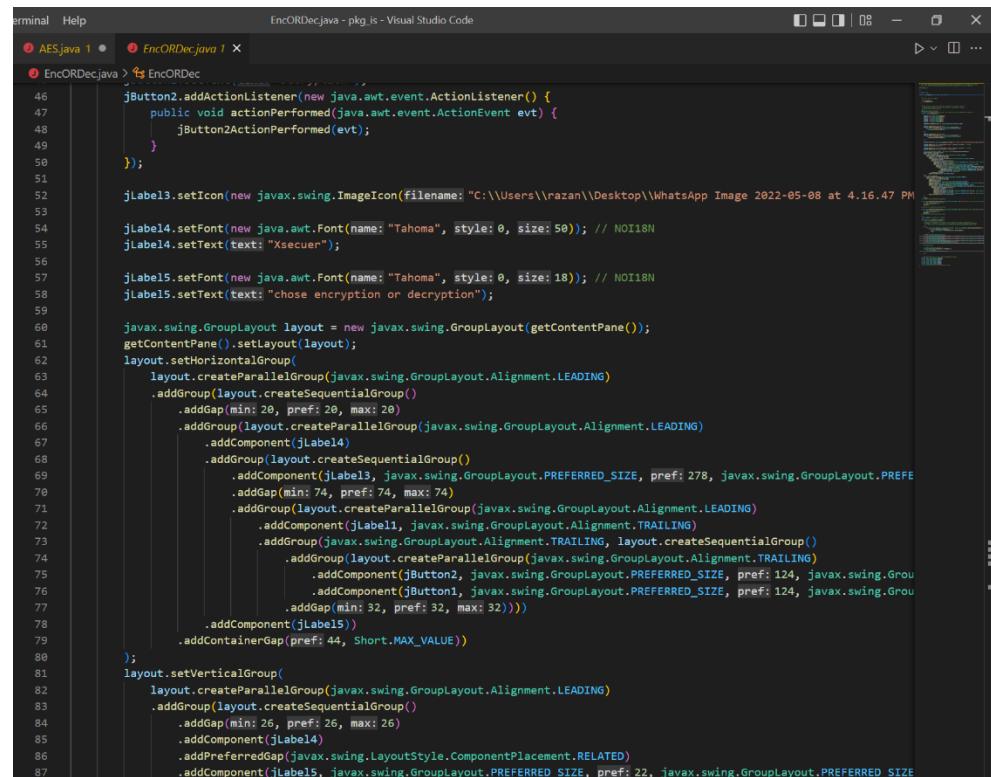
```
213     /*
214     try {
215         for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
216             if ("Nimbus".equals(info.getName())) {
217                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
218                 break;
219             }
220         }
221     } catch (ClassNotFoundException ex) {
222         java.util.logging.Logger.getLogger(enc.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
223     } catch (InstantiationException ex) {
224         java.util.logging.Logger.getLogger(enc.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
225     } catch (IllegalAccessException ex) {
226         java.util.logging.Logger.getLogger(enc.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
227     } catch (javax.swing.UnsupportedLookAndFeelException ex) {
228         java.util.logging.Logger.getLogger(enc.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
229     }
230     //
231
232     /* Create and display the form */
233     java.awt.EventQueue.invokeLater(new Runnable() {
234
235         public void run() {
236
237             new enc().setVisible(true);
238         }
239     });
240
241     // Variables declaration - do not modify//GEN-BEGIN:variables
242     private javax.swing.JButton jButton1;
243     private javax.swing.JButton jButton2;
244     private javax.swing.JButton jButton3;
245     private javax.swing.JButton jButton4;
246     private javax.swing.JLabel jLabel1;
247     private javax.swing.JLabel jLabel2;
248     private javax.swing.JLabel jLabel3;
249     private javax.swing.JLabel jLabel4;
250     private javax.swing.JLabel jLabel5;
251     // End of variables declaration//GEN-END:variables
252 }
253 }
```

6. EncORDec



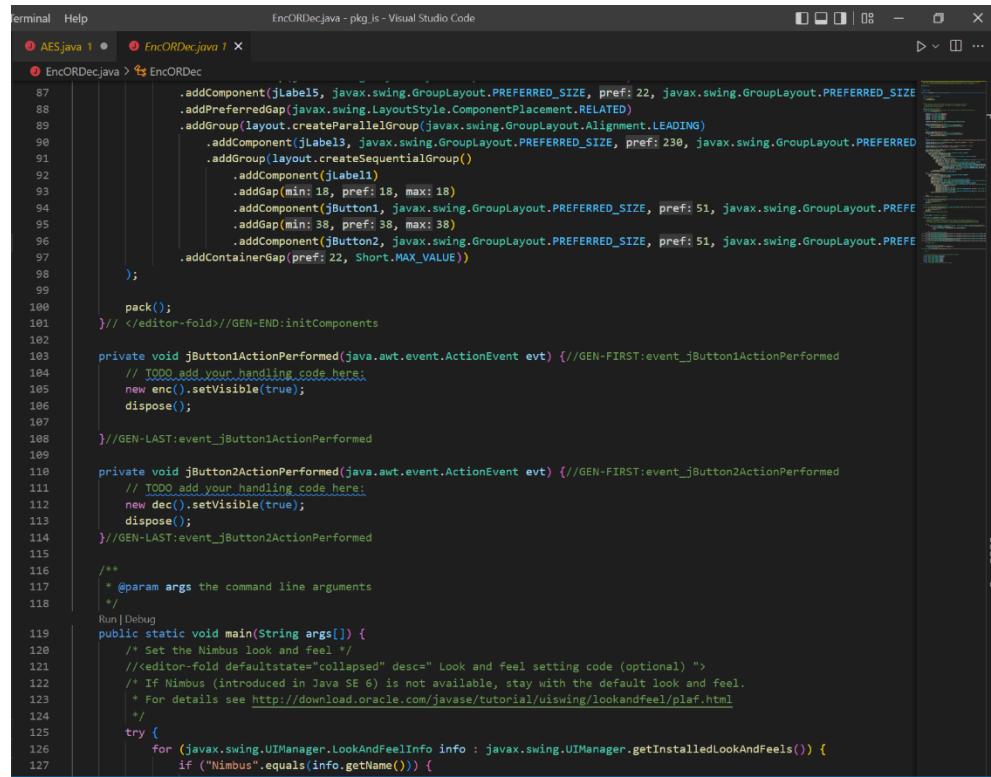
A screenshot of the Visual Studio Code interface showing the EncORDec.java file. The code is a Java Swing application for encryption and decryption. It includes imports for javax.swing and java.awt.event, and defines a class EncORDec that extends javax.swing.JFrame. The constructor initializes components like JButton1 and JButton2, and adds action listeners for each button. The actionPerformed method for JButton1 calls jButton1ActionPerformed, and the actionPerformed method for JButton2 calls jButton2ActionPerformed.

```
Terminal Help
AES.java 1 EncORDec.java 1
EncORDec.java > EncORDec
5 package pkg_1;
6
7 /**
8 *
9 * @author razan
10 */
11 public class EncORDec extends javax.swing.JFrame {
12
13     /**
14      * Creates new form EncORDec
15     */
16     public EncORDec() {
17         initComponents();
18     }
19
20     /**
21      * This method is called from within the constructor to initialize the form.
22      * WARNING: Do NOT modify this code. The content of this method is always
23      * regenerated by the Form Editor.
24      */
25     @SuppressWarnings("unchecked")
26 // <editor-fold defaultstate="collapsed" desc="Generated Code">/GEN-BEGIN:initComponents
27     private void initComponents() {
28
29         jButton1 = new javax.swing.JButton();
30         jButton2 = new javax.swing.JButton();
31         jLabel11 = new javax.swing.JLabel();
32         jLabel13 = new javax.swing.JLabel();
33         jLabel14 = new javax.swing.JLabel();
34         jLabel15 = new javax.swing.JLabel();
35
36         setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
37
38         jButton1.setText("Encryption");
39         jButton1.addActionListener(new java.awt.event.ActionListener() {
40             public void actionPerformed(java.awt.event.ActionEvent evt) {
41                 jButton1ActionPerformed(evt);
42             }
43         });
44
45         jButton2.setText("Decryption");
46         jButton2.addActionListener(new java.awt.event.ActionListener() {
```

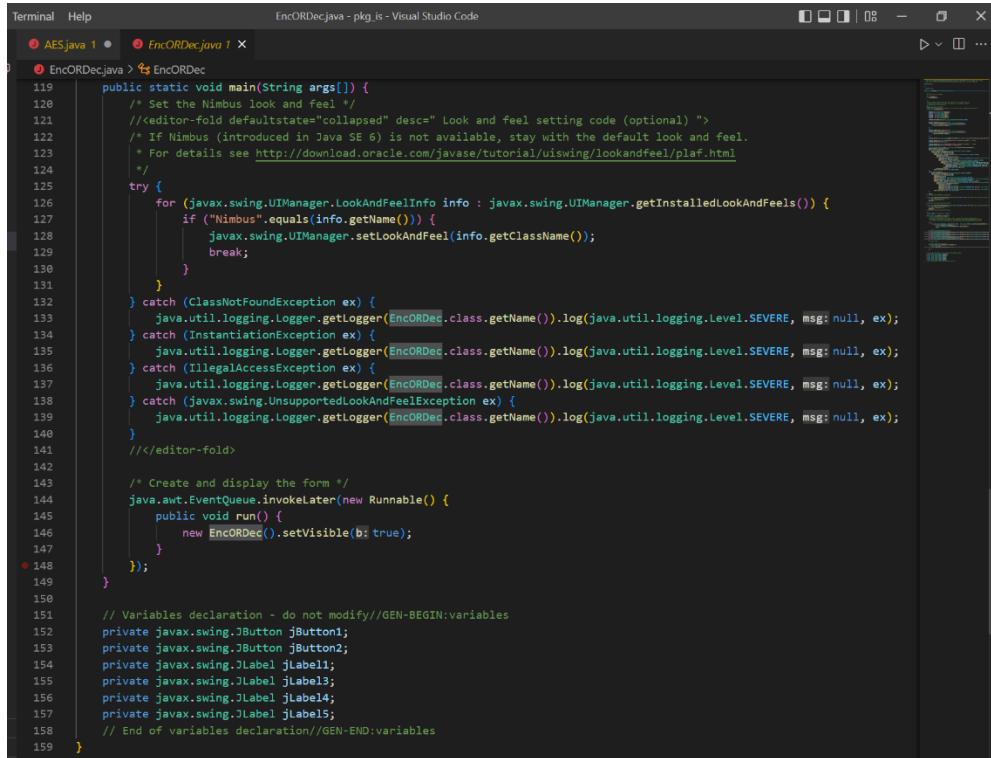


A screenshot of the Visual Studio Code interface showing the continuation of the EncORDec.java file. The code continues from the previous snippet, defining the jButton1ActionPerformed and jButton2ActionPerformed methods. It also defines jLabel11, jLabel13, jLabel14, and jLabel15. The layout is managed using javax.swing.GroupLayout, with various alignment, gap, and preferred size settings. The jLabel15 component is specifically noted to have a preferred size of 32x32 pixels.

```
        jButton1ActionPerformed(evt);
    }
});
jButton2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton2ActionPerformed(evt);
    }
});
jLabel3.setIcon(new javax.swing.ImageIcon(filename: "C:\\Users\\razan\\Desktop\\WhatsApp Image 2022-05-08 at 4.16.47 PM"));
jLabel4.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 50)); // NOI18N
jLabel4.setText(text: "Xsecuer");
jLabel5.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 18)); // NOI18N
jLabel5.setText(text: "choose encryption or decryption");
javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(20, 20, 20)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addComponent(jLabel14)
                    .addComponent(jLabel13, javax.swing.GroupLayout.PREFERRED_SIZE, 278, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel15, javax.swing.GroupLayout.PREFERRED_SIZE, 32, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(layout.createSequentialGroup()
                    .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 124, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 124, javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addGap(32, 32, 32)
            .addComponent(jLabel11, javax.swing.GroupLayout.PREFERRED_SIZE, 32, javax.swing.GroupLayout.PREFERRED_SIZE)
        )
        .addGroup(layout.createSequentialGroup()
            .addGap(26, 26, 26)
            .addComponent(jLabel14)
            .addGap(26, 26, 26)
            .addComponent(jLabel13, javax.swing.GroupLayout.PREFERRED_SIZE, 22, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(26, 26, 26)
            .addComponent(jLabel15, javax.swing.GroupLayout.PREFERRED_SIZE, 22, javax.swing.GroupLayout.PREFERRED_SIZE)
        )
    )
);
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(20, 20, 20)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jLabel14)
                .addComponent(jLabel13, javax.swing.GroupLayout.PREFERRED_SIZE, 278, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(jLabel15, javax.swing.GroupLayout.PREFERRED_SIZE, 32, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(32, 32, 32)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 124, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, 124, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(32, 32, 32)
            .addComponent(jLabel11, javax.swing.GroupLayout.PREFERRED_SIZE, 32, javax.swing.GroupLayout.PREFERRED_SIZE)
        )
        .addGroup(layout.createSequentialGroup()
            .addGap(26, 26, 26)
            .addComponent(jLabel14)
            .addGap(26, 26, 26)
            .addComponent(jLabel13, javax.swing.GroupLayout.PREFERRED_SIZE, 22, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(26, 26, 26)
            .addComponent(jLabel15, javax.swing.GroupLayout.PREFERRED_SIZE, 22, javax.swing.GroupLayout.PREFERRED_SIZE)
        )
    )
);
```

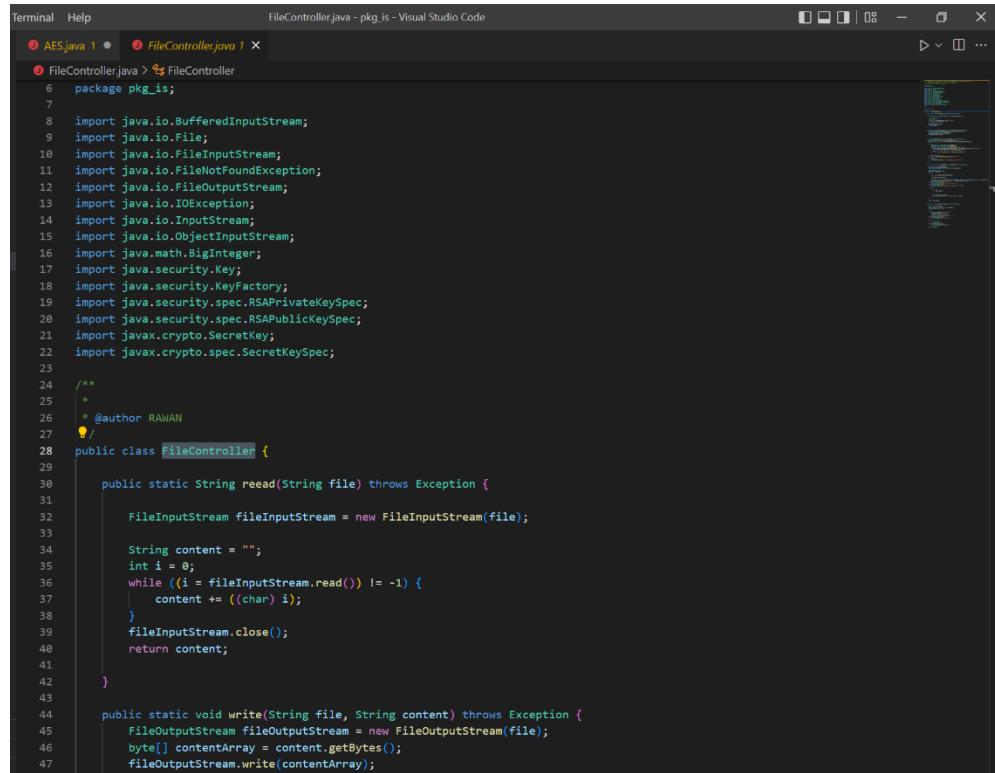


```
87     .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 22, javax.swing.GroupLayout.PREFERRED_SIZE)
88     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
89     .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
90         .addGroup(layout.createSequentialGroup()
91             .addComponent(jLabel1)
92             .addGap(min: 18, pref: 18, max: 18)
93             .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 51, javax.swing.GroupLayout.PREFERRED_SIZE)
94             .addGap(min: 38, pref: 38, max: 38)
95             .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 51, javax.swing.GroupLayout.PREFERRED_SIZE)
96             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 51, javax.swing.GroupLayout.PREFERRED_SIZE)
97         ).addGap(22, Short.MAX_VALUE)
98     );
99
100    pack();
101 // </editor-fold>//GEN-END:initComponents
102
103    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {GEN-FIRST:event_jButton1ActionPerformed
104        // TODO add your handling code here:
105        new enc().setVisible(true);
106        dispose();
107    }GEN-LAST:event_jButton1ActionPerformed
108
109    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {GEN-FIRST:event_jButton2ActionPerformed
110        // TODO add your handling code here:
111        new dec().setVisible(true);
112        dispose();
113    }GEN-LAST:event_jButton2ActionPerformed
114
115    /**
116     * @param args the command line arguments
117     */
118    Run|Debug
119    public static void main(String args[]) {
120        /* Set the Nimbus look and feel */
121        // <editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
122        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
123         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html 
```



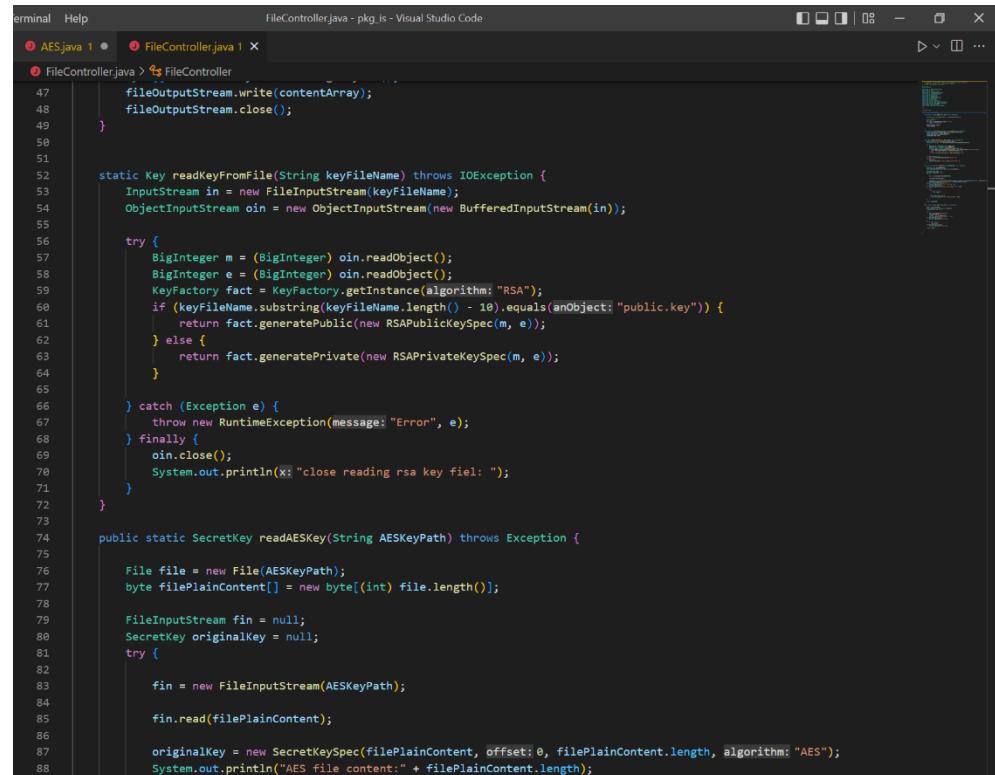
```
119        /* Set the Nimbus look and feel */
120        // <editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
121        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
122         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html 
```

7. FileController



A screenshot of the Visual Studio Code interface showing the `FileController.java` file. The code implements two static methods: `read(String file)` which reads the content of a file into a string, and `write(String file, String content)` which writes a string to a file. It also includes a static method `readKeyFromFile(String keyFileName)` for reading RSA keys from a file.

```
6 package pkg_is;
7
8 import java.io.BufferedReader;
9 import java.io.File;
10 import java.io.FileInputStream;
11 import java.io.FileNotFoundException;
12 import java.io.FileOutputStream;
13 import java.io.IOException;
14 import java.io.InputStream;
15 import java.io.ObjectInputStream;
16 import java.math.BigInteger;
17 import java.security.Key;
18 import java.security.KeyFactory;
19 import java.security.spec.RSAPrivateKeySpec;
20 import java.security.spec.RSAPublicKeySpec;
21 import javax.crypto.SecretKey;
22 import javax.crypto.spec.SecretKeySpec;
23
24 /**
25 * @author RAWAN
26 */
27
28 public class FileController {
29
30     public static String read(String file) throws Exception {
31
32         FileInputStream fileInputStream = new FileInputStream(file);
33
34         String content = "";
35         int i = 0;
36         while ((i = fileInputStream.read()) != -1) {
37             content += ((char) i);
38         }
39         fileInputStream.close();
40         return content;
41     }
42
43
44     public static void write(String file, String content) throws Exception {
45         FileOutputStream fileOutputStream = new FileOutputStream(file);
46         byte[] contentArray = content.getBytes();
47         fileOutputStream.write(contentArray);
48     }
49
50
51     static Key readKeyFromFile(String keyFileName) throws IOException {
52         InputStream in = new FileInputStream(keyFileName);
53         ObjectInputStream oin = new ObjectInputStream(new BufferedInputStream(in));
54
55         try {
56             BigInteger m = (BigInteger) oin.readObject();
57             BigInteger e = (BigInteger) oin.readObject();
58             KeyFactory fact = KeyFactory.getInstance(algorithm: "RSA");
59             if (keyFileName.substring(keyFileName.length() - 10).equals(anObject: "public.key")) {
60                 return fact.generatePublic(new RSAPublicKeySpec(m, e));
61             } else {
62                 return fact.generatePrivate(new RSAPrivateKeySpec(m, e));
63             }
64         } catch (Exception e) {
65             throw new RuntimeException(message: "Error", e);
66         } finally {
67             oin.close();
68             System.out.println(x: "close reading rsa key file: ");
69         }
70     }
71
72
73     public static SecretKey readAESKey(String AESKeyPath) throws Exception {
74
75         File file = new File(AESKeyPath);
76         byte filePlainContent[] = new byte[(int) file.length()];
77
78         FileInputStream fin = null;
79         SecretKey originalKey = null;
80         try {
81
82             fin = new FileInputStream(AESKeyPath);
83
84             fin.read(filePlainContent);
85
86             originalKey = new SecretKeySpec(filePlainContent, offset: 0, filePlainContent.length, algorithm: "AES");
87             System.out.println("AES file content: " + filePlainContent.length);
88         } finally {
89             if (fin != null) fin.close();
90         }
91     }
92 }
```



A screenshot of the Visual Studio Code interface showing the `AES.java` file. This file contains the implementation of the `FileController` class, including the `read`, `write`, and `readKeyFromFile` methods, as well as the `readAESKey` method which reads the AES key from a file.

```
47         fileOutputStream.write(contentArray);
48         fileOutputStream.close();
49     }
50
51
52     static Key readKeyFromFile(String keyFileName) throws IOException {
53         InputStream in = new FileInputStream(keyFileName);
54         ObjectInputStream oin = new ObjectInputStream(new BufferedInputStream(in));
55
56         try {
57             BigInteger m = (BigInteger) oin.readObject();
58             BigInteger e = (BigInteger) oin.readObject();
59             KeyFactory fact = KeyFactory.getInstance(algorithm: "RSA");
60             if (keyFileName.substring(keyFileName.length() - 10).equals(anObject: "public.key")) {
61                 return fact.generatePublic(new RSAPublicKeySpec(m, e));
62             } else {
63                 return fact.generatePrivate(new RSAPrivateKeySpec(m, e));
64             }
65         } catch (Exception e) {
66             throw new RuntimeException(message: "Error", e);
67         } finally {
68             oin.close();
69             System.out.println(x: "close reading rsa key file: ");
70         }
71     }
72
73
74     public static SecretKey readAESKey(String AESKeyPath) throws Exception {
75
76         File file = new File(AESKeyPath);
77         byte filePlainContent[] = new byte[(int) file.length()];
78
79         FileInputStream fin = null;
80         SecretKey originalKey = null;
81         try {
82
83             fin = new FileInputStream(AESKeyPath);
84
85             fin.read(filePlainContent);
86
87             originalKey = new SecretKeySpec(filePlainContent, offset: 0, filePlainContent.length, algorithm: "AES");
88             System.out.println("AES file content: " + filePlainContent.length);
89         } finally {
90             if (fin != null) fin.close();
91         }
92     }
93 }
```

A screenshot of the Visual Studio Code interface. The title bar says "FileController.java - pkg_is - Visual Studio Code". There are two tabs open: "AES.java 1" and "FileController.java 1". The "FileController.java" tab is active, showing Java code for file reading and exception handling. The code includes try-catch blocks for `FileNotFoundException` and `IOException`, and a finally block for closing streams. It also includes a static method `read(String file)` that reads the content of a file into a byte array.

```
89     } catch (FileNotFoundException e) {
90         System.out.println("AES file not found" + e);
91     } catch (IOException ioe) {
92         System.out.println("Exception while reading file" + ioe);
93     } finally {
94
95         try {
96             if (fin != null) {
97                 fin.close();
98             }
99
100        } catch (IOException ioe) {
101            System.out.println("Error closing stream:" + ioe);
102        }
103    }
104
105    return originalKey;
106}
107
108 public static byte[] read(String file) throws Exception {
109
110     File f = new File(file);
111     byte Content[] = new byte[(int) f.length()];
112     FileInputStream fin = null;
113
114     try {
115         fin = new FileInputStream(file);
116         fin.read(Content);
117         String s = new String(Content);
118         System.out.println("file content: " + s);
119     } catch (IOException ioe) {
120         System.out.println("Erzon");
121     }
122
123     try {
124         if (fin != null)
125             fin.close();
126     } catch (IOException ioe) {
127         System.out.println("Erzon");
128     }
129
130     return Content;
131 }
```

8. loginWindos

A screenshot of the Visual Studio Code interface. The title bar says "loginWindos.java - pkg_is - Visual Studio Code". There are three tabs open: "AES.java 1", "FileController.java 1", and "loginWindos.java 3". The "loginWindos.java" tab is active, showing Java code for a Swing application. It defines a class `loginWindos` that extends `JFrame`. The constructor initializes components, and there is a private method `initComponents()` for setting up UI elements like labels, text fields, and buttons.

```
5 package pkg_is;
6
7 import java.io.File;
8 import java.nio.charset.StandardCharsets;
9 import java.security.MessageDigest;
10 import java.util.logging.Level;
11 import java.util.logging.Logger;
12 import javax.swing.JFileChooser;
13 import javax.swing.JOptionPane;
14
15 /**
16 *
17 * @author RAWAN
18 */
19 public class loginWindos extends javax.swing.JFrame {
20
21
22     /*
23      * Creates new form loginWindos
24      */
25     public loginWindos() {
26         initComponents();
27     }
28
29
30     /**
31      * This method is called from within the constructor to initialize the form.
32      * WARNING: Do NOT modify this code. The content of this method is always
33      * regenerated by the Form Editor.
34      */
35     @SuppressWarnings("unchecked")
36 // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
37     private void initComponents() {
38
39         jLabel1 = new javax.swing.JLabel();
40         jLabel2 = new javax.swing.JLabel();
41         jLabel3 = new javax.swing.JLabel();
42         jLabel4 = new javax.swing.JLabel();
43         jLabel5 = new javax.swing.JLabel();
44         jLabel6 = new javax.swing.JLabel();
45         jLabel7 = new javax.swing.JLabel();
46
47         setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
48
49         jLabel1.setText("JLabel");
50         jLabel2.setText("JLabel");
51         jLabel3.setText("JLabel");
52         jLabel4.setText("JLabel");
53         jLabel5.setText("JLabel");
54         jLabel6.setText("JLabel");
55         jLabel7.setText("JLabel");
56
57         javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
58         getContentPane().setLayout(layout);
59         layout.setHorizontalGroup(
60             layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
61             .addGroup(layout.createSequentialGroup()
62                 .addContainerGap()
63                 .addComponent(jLabel1)
64                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
65                 .addComponent(jLabel2)
66                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
67                 .addComponent(jLabel3)
68                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
69                 .addComponent(jLabel4)
70                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
71                 .addComponent(jLabel5)
72                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
73                 .addComponent(jLabel6)
74                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
75                 .addComponent(jLabel7)
76                 .addContainerGap())
77         );
78
79         pack();
80     }//GEN-END:initComponents
81
82     /**
83      * @param args the command line arguments
84      */
85     public static void main(String args[]) {
86
87         java.awt.EventQueue.invokeLater(new Runnable() {
88             public void run() {
89                 new loginWindos().setVisible(true);
90             }
91         });
92     }
93
94 }
```

```
terminal Help loginWindows.java - pkg is - Visual Studio Code
● AES.java 1 ● ● FileController.java 1 ● ● loginWindows.java 3 ×
● loginWindows.java > ⌂ loginWindows

46
47     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
48
49     jLabel1.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 24)); // NOI18N
50     jLabel1.setText(text: "Enter User Name");
51
52     jLabel2.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 24)); // NOI18N
53     jLabel2.setText(text: "Enter Password");
54
55     username.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 24)); // NOI18N
56
57     password.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 24)); // NOI18N
58     password.addActionListener(new java.awt.event.ActionListener() {
59         public void actionPerformed(java.awt.event.ActionEvent evt) {
60             passwordActionPerformed(evt);
61         }
62     });
63
64     jButton2.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 24)); // NOI18N
65     jButton2.setText(text: "logging");
66     jButton2.addActionListener(new java.awt.event.ActionListener() {
67         public void actionPerformed(java.awt.event.ActionEvent evt) {
68             jButton2ActionPerformed(evt);
69         }
70     });
71
72     jButton3.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 24)); // NOI18N
73     jButton3.setText(text: "Signin");
74     jButton3.addActionListener(new java.awt.event.ActionListener() {
75         public void actionPerformed(java.awt.event.ActionEvent evt) {
76             jButton3ActionPerformed(evt);
77         }
78     });
79
80     jLabel3.setIcon(new javax.swing.ImageIcon(filename: "C:\\Users\\razaan\\Desktop\\WhatsApp Image 2022-05-08 at 4.16.47 PM"));
81
82     jLabel4.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 50)); // NOI18N
83     jLabel4.setText(text: "Xsecurer");
84
85     jLabel5.setFont(new java.awt.Font(name: "Tahoma", style: 0, size: 18)); // NOI18N
86     jLabel5.setText(text: "company");
87
```

```
77
78     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
79     getContentPane().setLayout(layout);
80     layout.setHorizontalGroup(
81         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
82             .addGroup(layout.createSequentialGroup()
83                 .addGap(18, 18, 18)
84                 .addComponent(jLabel1)
85                 .addGap(18, 18, 18)
86                 .addComponent(jLabel2)
87                 .addGap(18, 18, 18)
88                 .addComponent(jLabel3)
89                 .addGap(18, 18, 18)
90                 .addComponent(jButton1)
91                 .addGap(18, 18, 18)
92                 .addComponent(jButton2)
93                 .addGap(18, 18, 18)
94                 .addComponent(jButton3)
95                 .addGap(18, 18, 18)
96                 .addComponent(jButton4)
97                 .addGap(18, 18, 18)
98                 .addComponent(jButton5)
99                 .addGap(18, 18, 18)
100                .addComponent(jButton6)
101                .addGap(18, 18, 18)
102                .addComponent(jButton7)
103                .addGap(18, 18, 18)
104                .addComponent(jButton8)
105                .addGap(18, 18, 18)
106                .addComponent(jButton9)
107                .addGap(18, 18, 18)
108                .addComponent(jButton10)
109                .addGap(18, 18, 18)
110            )
111            .setVerticalGroup(
112                layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
113                    .addGroup(layout.createSequentialGroup()
114                        .addGap(18, 18, 18)
115                        .addComponent(jLabel4)
116                        .addGap(18, 18, 18)
117                        .addComponent(jLabel5)
118                        .addGap(18, 18, 18)
119                        .addComponent(jLabel6)
120                        .addGap(18, 18, 18)
121                        .addComponent(jLabel7)
122                        .addGap(18, 18, 18)
123                        .addComponent(jLabel8)
124                        .addGap(18, 18, 18)
125                        .addComponent(jLabel9)
126                        .addGap(18, 18, 18)
127                        .addComponent(jLabel10)
128                        .addGap(18, 18, 18)
129                    )
130            )
131        )
132    );
133    layout.setVerticalGroup(
134        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
135            .addGroup(layout.createSequentialGroup()
136                .addGap(18, 18, 18)
137                .addComponent(jLabel1)
138                .addGap(18, 18, 18)
139                .addComponent(jLabel2)
140                .addGap(18, 18, 18)
141                .addComponent(jLabel3)
142                .addGap(18, 18, 18)
143                .addComponent(jButton1)
144                .addGap(18, 18, 18)
145                .addComponent(jButton2)
146                .addGap(18, 18, 18)
147                .addComponent(jButton3)
148                .addGap(18, 18, 18)
149                .addComponent(jButton4)
150                .addGap(18, 18, 18)
151                .addComponent(jButton5)
152                .addGap(18, 18, 18)
153                .addComponent(jButton6)
154                .addGap(18, 18, 18)
155                .addComponent(jButton7)
156                .addGap(18, 18, 18)
157                .addComponent(jButton8)
158                .addGap(18, 18, 18)
159                .addComponent(jButton9)
160                .addGap(18, 18, 18)
161                .addComponent(jButton10)
162                .addGap(18, 18, 18)
163                .addComponent(jLabel4)
164                .addGap(18, 18, 18)
165                .addComponent(jLabel5)
166                .addGap(18, 18, 18)
167                .addComponent(jLabel6)
168                .addGap(18, 18, 18)
169                .addComponent(jLabel7)
170                .addGap(18, 18, 18)
171                .addComponent(jLabel8)
172                .addGap(18, 18, 18)
173                .addComponent(jLabel9)
174                .addGap(18, 18, 18)
175                .addComponent(jLabel10)
176                .addGap(18, 18, 18)
177            )
178        )
179    );
180
```

```
128         .addGap(min: 51, pref: 51, max: 51)
129         .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
130             .addComponent(jButton3, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 50, javax.swing.GroupLayout.PREFERRED_SIZE)
131             .addComponent(jButton2, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 50, javax.swing.GroupLayout.PREFERRED_SIZE)
132             .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, pref: 230, javax.swing.GroupLayout.PREFERRED_SIZE)
133         ).addGap(min: 32, pref: 32, max: 32)
134     );
135
136     pack();
137 } // </editor-fold>//GEN-END:initComponents
138
139 private void passwordActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_passwordActionPerformed
140     // TODO add your handling code here:
141 } //GEN-LAST:event_passwordActionPerformed
142 static String save_keys_path;
143 byte[] hashPassword;
144 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_jButton3ActionPerformed
145
146     JOptionPane.showMessageDialog(parentComponent: null, message: "Account has been successfully registered");
147     JFileChooser fileChooser = new JFileChooser();
148     fileChooser.setCurrentDirectory(new java.io.File(pathname: "C:\\\\Users\\\\ravan\\\\Desktop\\\\PROJECT IS\\\\"));
149     fileChooser.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
150     int option = fileChooser.showOpenDialog(this);
151     if (option == JFileChooser.APPROVE_OPTION) {
152         File file = fileChooser.getSelectedFile();
153         save_keys_path = file.getAbsolutePath();
154
155         System.out.println(" ** public key and private was saved in path : "+save_keys_path+" **");
156     }
157     try {
158         hashPassword = generateHash(username.getText(), password.getText());
159         RSAKeyGenerator.RSA();
160     } catch (Exception ex) {
161         Logger.getLogger(loginWindos.class.getName()).log(Level.SEVERE, msg: null, ex);
162     }
163 }
164 //new enc().setVisible(true);
165 new EncORDec().setVisible(true);
166 dispose();
167 } //GEN-LAST:event_jButton3ActionPerformed
168
169 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_jButton2ActionPerformed
170     // TODO add your handling code here:
171     try {
172
173         String user = username.getText();
174         String pass = password.getText();
175
176         byte hashed_pass[] = generateHash(user, pass);
177         //if (user.equals("ravan") && Arrays.equals(hashed_pass, hashPassword)) {
178         if (user.equals("ravan") && (pass.equals("123"))){
179             JOptionPane.showMessageDialog(parentComponent: null, message: "Welcome,You have been successfully logged in");
180             //new enc(save_keys_path).setVisible(true);
181             new EncORDec().setVisible(true);
182             dispose();
183         }
184
185         else if (user.equals(anObject: "razan") && (pass.equals(anObject: "123"))){
186             JOptionPane.showMessageDialog(parentComponent: null, message: "Welcome,You have been successfully logged in");
187             new EncORDec().setVisible(true);
188             dispose();
189         }
190         else if (user.equals(anObject: "renad") && (pass.equals(anObject: "123"))){
191             JOptionPane.showMessageDialog(parentComponent: null, message: "Welcome,You have been successfully logged in");
192             //new enc(save_keys_path).setVisible(true);
193             new EncORDec().setVisible(true);
194             dispose();
195         }
196         else if (user.equals(anObject: "nor") && (pass.equals(anObject: "123"))){
197             JOptionPane.showMessageDialog(parentComponent: null, message: "Welcome,You have been successfully logged in");
198             //new enc(save_keys_path).setVisible(true);
199             new EncORDec().setVisible(true);
200             dispose();
201         }
202         else
203             JOptionPane.showMessageDialog(parentComponent: null, message: "Sorry, you have to register");
204
205     } catch (Exception ex) {
206         ex.printStackTrace();
207     }
208 } //GEN-LAST:event_jButton2ActionPerformed
209
210 public byte[] generateHash(String user_text, String password_text) throws Exception {
```

```
169     private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_jButton2ActionPerformed
170     // TODO add your handling code here:
171     try {
172
173         String user = username.getText();
174         String pass = password.getText();
175
176         byte hashed_pass[] = generateHash(user, pass);
177         //if (user.equals("ravan") && Arrays.equals(hashed_pass, hashPassword)) {
178         if (user.equals("ravan") && (pass.equals("123"))){
179             JOptionPane.showMessageDialog(parentComponent: null, message: "Welcome,You have been successfully logged in");
180             //new enc(save_keys_path).setVisible(true);
181             new EncORDec().setVisible(true);
182             dispose();
183         }
184
185         else if (user.equals(anObject: "razan") && (pass.equals(anObject: "123"))){
186             JOptionPane.showMessageDialog(parentComponent: null, message: "Welcome,You have been successfully logged in");
187             new EncORDec().setVisible(true);
188             dispose();
189         }
190         else if (user.equals(anObject: "renad") && (pass.equals(anObject: "123"))){
191             JOptionPane.showMessageDialog(parentComponent: null, message: "Welcome,You have been successfully logged in");
192             //new enc(save_keys_path).setVisible(true);
193             new EncORDec().setVisible(true);
194             dispose();
195         }
196         else if (user.equals(anObject: "nor") && (pass.equals(anObject: "123"))){
197             JOptionPane.showMessageDialog(parentComponent: null, message: "Welcome,You have been successfully logged in");
198             //new enc(save_keys_path).setVisible(true);
199             new EncORDec().setVisible(true);
200             dispose();
201         }
202         else
203             JOptionPane.showMessageDialog(parentComponent: null, message: "Sorry, you have to register");
204
205     } catch (Exception ex) {
206         ex.printStackTrace();
207     }
208 } //GEN-LAST:event_jButton2ActionPerformed
209
210 public byte[] generateHash(String user_text, String password_text) throws Exception {
```

```
terminal Help
AES.java 1 ● FileController.java 1 ● loginWindos.java 3
loginWindos.java > loginWindos

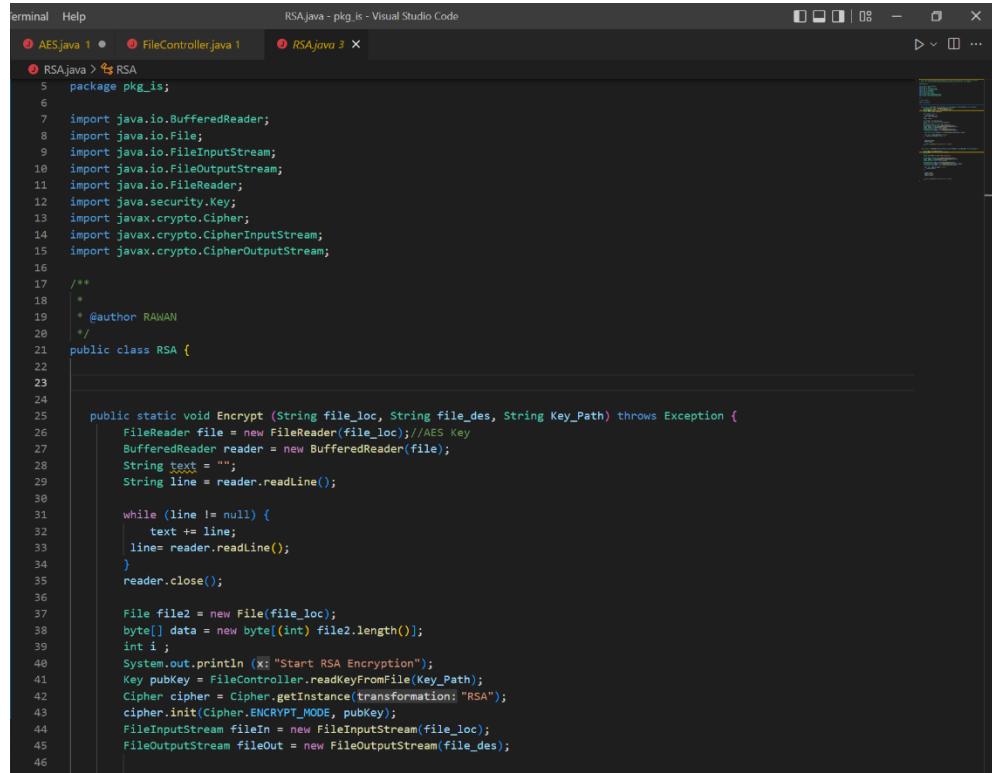
210     public byte[] generateHash(String user_text, String password_text) throws Exception {
211
212         MessageDigest md = MessageDigest.getInstance(algorithm: "SHA-512");
213         String hash = user_text + password_text;
214         byte[] hashedPassword = md.digest(hash.getBytes(StandardCharsets.UTF_8));
215         return hashedPassword;
216
217     /**
218      * @param args the command line arguments
219     */
220     Run|Debug
221     public static void main(String args[]) {
222
223         /* Set the Nimbus look and feel */
224         //editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) "
225         /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
226         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
227         */
228         try {
229             for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
230                 if ("Nimbus".equals(info.getName())) {
231                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
232                     break;
233                 }
234             }
235         } catch (ClassNotFoundException ex) {
236             java.util.logging.Logger.getLogger(loginWindos.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
237         } catch (InstantiationException ex) {
238             java.util.logging.Logger.getLogger(loginWindos.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
239         } catch (IllegalAccessException ex) {
240             java.util.logging.Logger.getLogger(loginWindos.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
241         } catch (javax.swing.UnsupportedLookAndFeelException ex) {
242             java.util.logging.Logger.getLogger(loginWindos.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
243         }
244         //
245
246         /* Create and display the form */
247         java.awt.EventQueue.invokeLater(new Runnable() {
248             public void run() {
249                 new loginWindos().setVisible(b: true);
250             }
251         });
252     }
253
254 }
```

```
terminal Help
AES.java 1 ● FileController.java 1 ● loginWindos.java 3
loginWindos.java > loginWindos

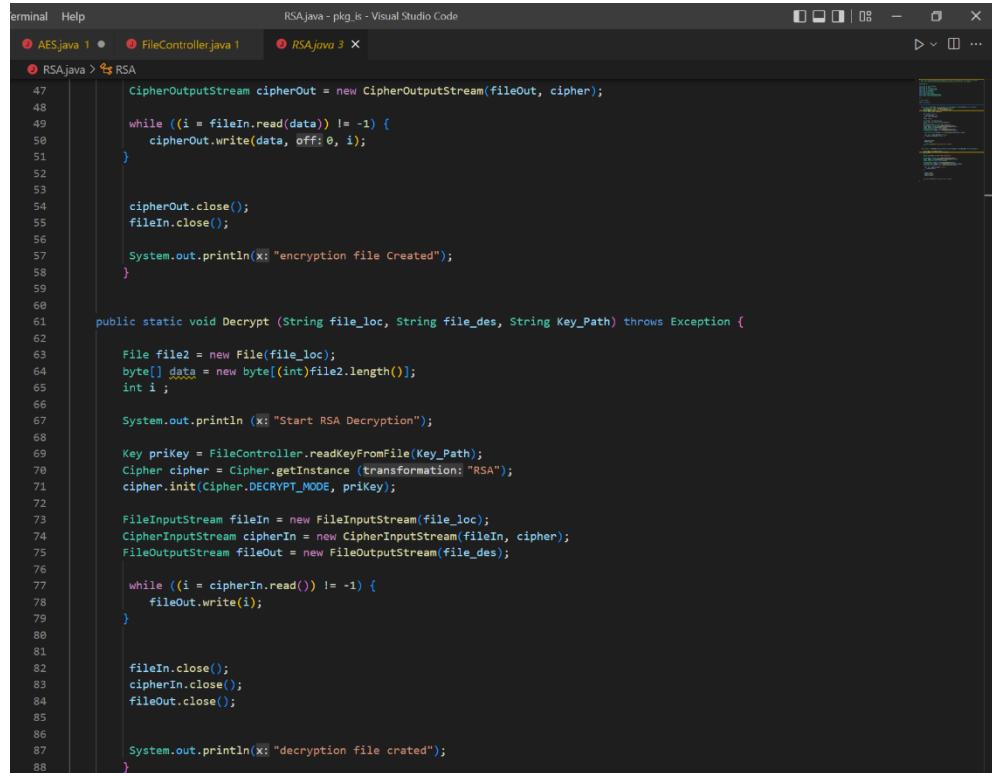
245         */
246         try {
247             for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
248                 if ("Nimbus".equals(info.getName())) {
249                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
250                     break;
251                 }
252             }
253         } catch (ClassNotFoundException ex) {
254             java.util.logging.Logger.getLogger(loginWindos.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
255         } catch (InstantiationException ex) {
256             java.util.logging.Logger.getLogger(loginWindos.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
257         } catch (IllegalAccessException ex) {
258             java.util.logging.Logger.getLogger(loginWindos.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
259         } catch (javax.swing.UnsupportedLookAndFeelException ex) {
260             java.util.logging.Logger.getLogger(loginWindos.class.getName()).log(java.util.logging.Level.SEVERE, msg: null, ex);
261         }
262         //
```

```
// Variables declaration - do not modify//GEN-BEGIN:variables
263     private javax.swing.JButton jButton2;
264     private javax.swing.JButton jButton3;
265     private javax.swing.JLabel jLabel1;
266     private javax.swing.JLabel jLabel2;
267     private javax.swing.JLabel jLabel3;
268     private javax.swing.JLabel jLabel4;
269     private javax.swing.JLabel jLabel5;
270     private javax.swing.JPasswordField password;
271     public static javax.swing.JTextField username;
272     // End of variables declaration//GEN-END:variables
273 }
```

9. RSA

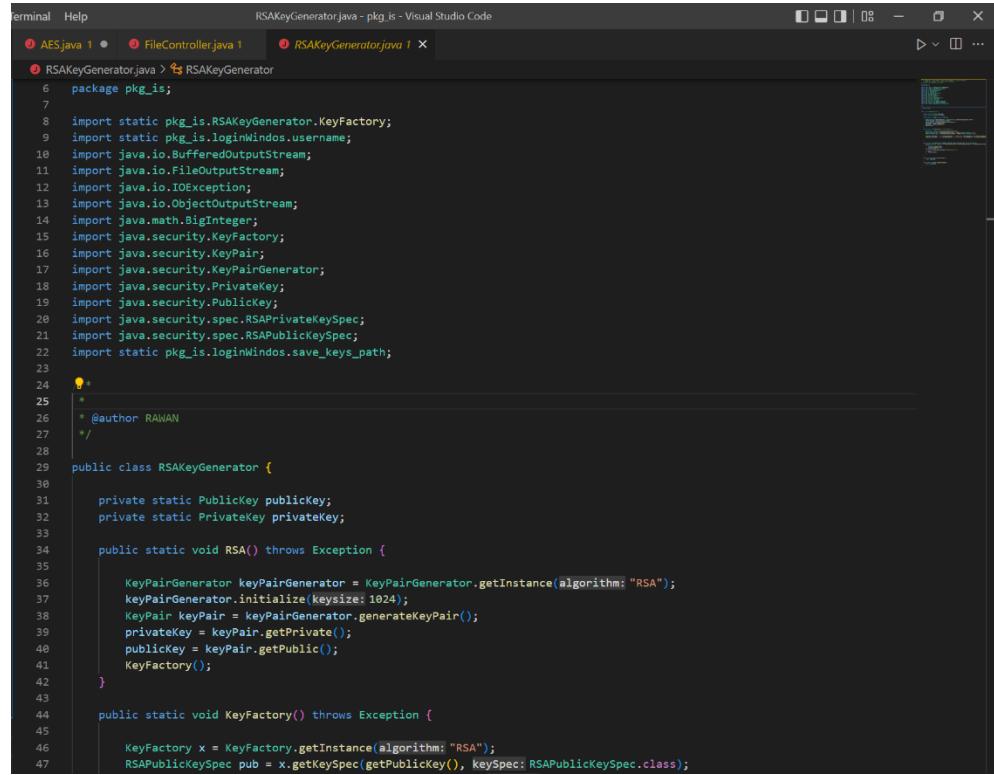


```
terminal Help RSA.java - pkg_is - Visual Studio Code
① AESjava 1 ② FileController.java 1 ③ RSAjava 3 X
④ RSA.java > RSA
5 package pkg_is;
6
7 import java.io.BufferedReader;
8 import java.io.File;
9 import java.io.FileInputStream;
10 import java.io.FileOutputStream;
11 import java.io.FileReader;
12 import java.security.Key;
13 import javax.crypto.Cipher;
14 import javax.crypto.CipherInputStream;
15 import javax.crypto.CipherOutputStream;
16
17 /**
18 *
19 * @author RAWAN
20 */
21 public class RSA {
22
23
24
25     public static void Encrypt (String file_loc, String file_des, String Key_Path) throws Exception {
26         FileReader file = new FileReader(file_loc); //AES Key
27         BufferedReader reader = new BufferedReader(file);
28         String text = "";
29         String line = reader.readLine();
30
31         while (line != null) {
32             text += line;
33             line= reader.readLine();
34         }
35         reader.close();
36
37         File file2 = new File(file_loc);
38         byte[] data = new byte[(int) file2.length()];
39         int i ;
40         System.out.println (x: "Start RSA Encryption");
41         Key pubKey = FileController.readKeyFromFile(Key_Path);
42         Cipher cipher = Cipher.getInstance(transformation: "RSA");
43         cipher.init(Cipher.ENCRYPT_MODE, pubKey);
44         FileInputStream fileIn = new FileInputStream(file_loc);
45         FileOutputStream fileOut = new FileOutputStream(file_des);
46     }
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61     public static void Decrypt (String file_loc, String file_des, String Key_Path) throws Exception {
62
63         File file2 = new File(file_loc);
64         byte[] data = new byte[(int)file2.length()];
65         int i ;
66
67         System.out.println (x: "Start RSA Decryption");
68
69         Key priKey = FileController.readKeyFromFile(Key_Path);
70         Cipher cipher = Cipher.getInstance (transformation: "RSA");
71         cipher.init(Cipher.DECRYPT_MODE, priKey);
72
73         FileInputStream fileIn = new FileInputStream(file_loc);
74         CipherInputStream cipherIn = new CipherInputStream(fileIn, cipher);
75         FileOutputStream fileOut = new FileOutputStream(file_des);
76
77         while ((i = cipherIn.read()) != -1) {
78             fileOut.write(i);
79         }
80
81
82
83
84
85
86
87
88     }
89 }
```

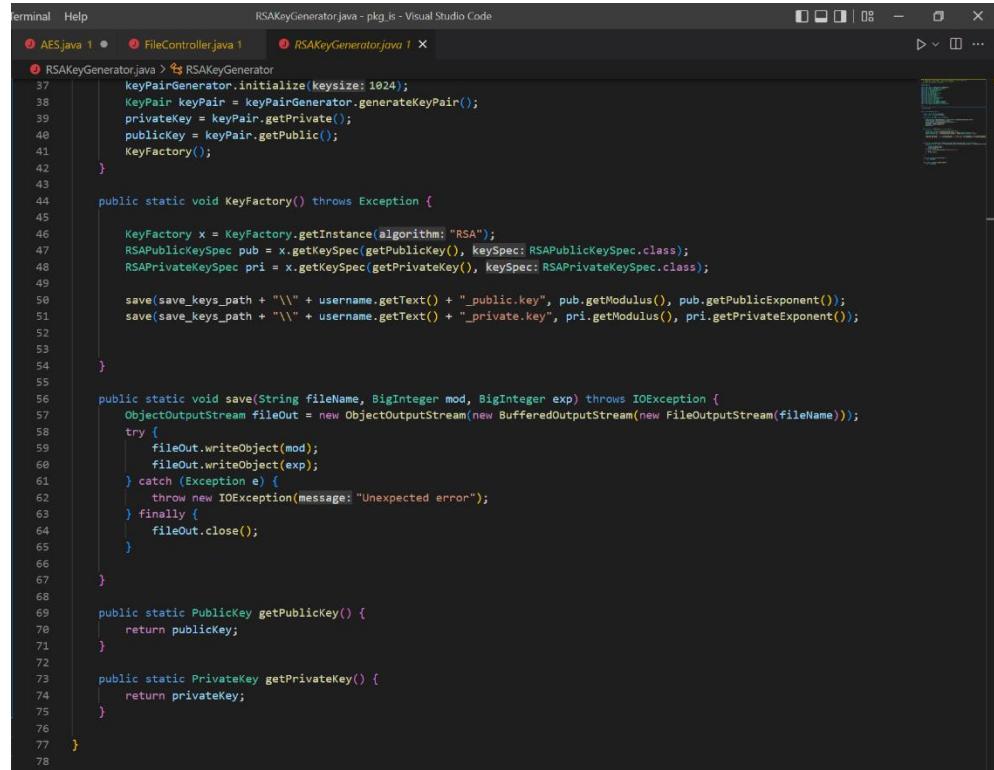


```
terminal Help RSA.java - pkg_is - Visual Studio Code
① AESjava 1 ② FileController.java 1 ③ RSAjava 3 X
④ RSA.java > RSA
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61     public static void Decrypt (String file_loc, String file_des, String Key_Path) throws Exception {
62
63         File file2 = new File(file_loc);
64         byte[] data = new byte[(int)file2.length()];
65         int i ;
66
67         System.out.println (x: "Start RSA Decryption");
68
69         Key priKey = FileController.readKeyFromFile(Key_Path);
70         Cipher cipher = Cipher.getInstance (transformation: "RSA");
71         cipher.init(Cipher.DECRYPT_MODE, priKey);
72
73         FileInputStream fileIn = new FileInputStream(file_loc);
74         CipherInputStream cipherIn = new CipherInputStream(fileIn, cipher);
75         FileOutputStream fileOut = new FileOutputStream(file_des);
76
77         while ((i = cipherIn.read()) != -1) {
78             fileOut.write(i);
79         }
80
81
82
83
84
85
86
87
88     }
89 }
```

10. RSAKeyGenerator

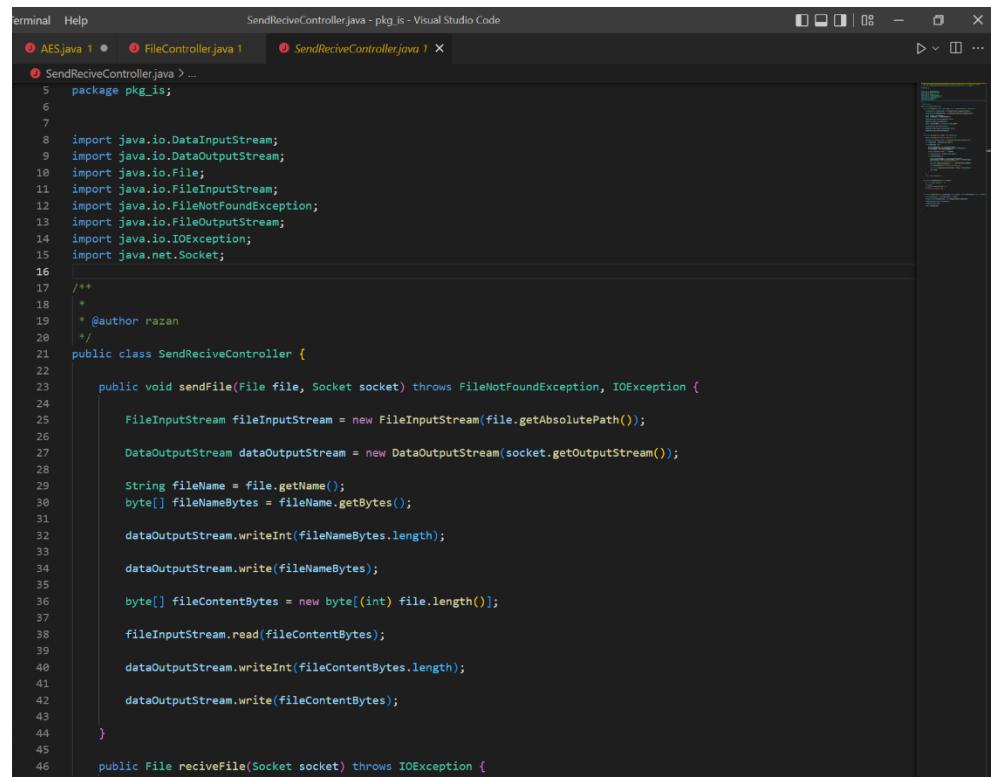


```
terminal Help RSAKeyGenerator.java - pkg_is - Visual Studio Code
① AES.java 1 ② FileController.java 1 ③ RSAKeyGenerator.java 1
④ RSAKeyGenerator.java > RSAKeyGenerator
6 package pkg_is;
7
8 import static pkg_is.RSAKeyGenerator.KeyFactory;
9 import static pkg_is.loginWindos.username;
10 import java.io.BufferedOutputStream;
11 import java.io.FileOutputStream;
12 import java.io.IOException;
13 import java.io.ObjectOutputStream;
14 import java.math.BigInteger;
15 import java.security.KeyFactory;
16 import java.security.KeyPair;
17 import java.security.KeyPairGenerator;
18 import java.security.PrivateKey;
19 import java.security.PublicKey;
20 import java.security.spec.RSAPrivateKeySpec;
21 import java.security.spec.RSAPublicKeySpec;
22 import static pkg_is.loginWindos.save_keys_path;
23
24 /**
25 * @author RAWAN
26 */
27
28 public class RSAKeyGenerator {
29
30     private static PublicKey publicKey;
31     private static PrivateKey privateKey;
32
33     public static void RSA() throws Exception {
34
35         KeyPairGenerator keyPairGenerator = KeyPairGenerator.getInstance(algorithm: "RSA");
36         keyPairGenerator.initialize(keysize: 1024);
37         KeyPair keyPair = keyPairGenerator.generateKeyPair();
38         privateKey = keyPair.getPrivate();
39         publicKey = keyPair.getPublic();
40         KeyFactory();
41     }
42
43
44     public static void KeyFactory() throws Exception {
45
46         KeyFactory x = KeyFactory.getInstance(algorithm: "RSA");
47         RSAPublicKeySpec pub = x.getKeySpec(getPublicKey(), keySpec: RSAPublicKeySpec.class);
48
49     }
50 }
```

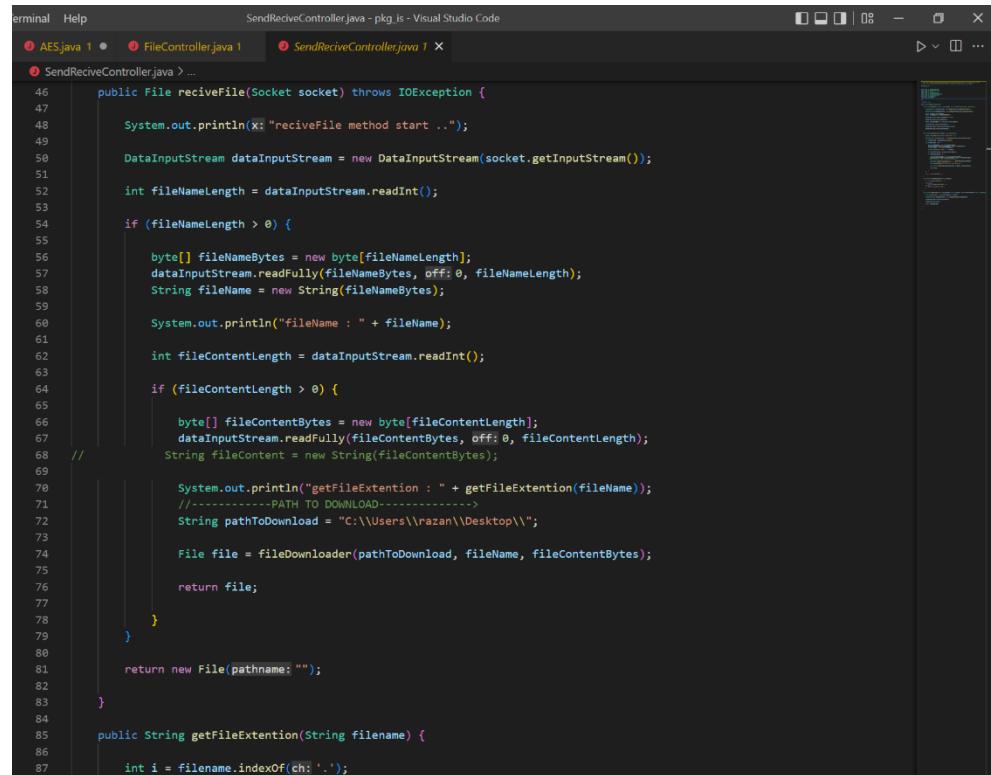


```
terminal Help RSAKeyGenerator.java - pkg_is - Visual Studio Code
① AES.java 1 ② FileController.java 1 ③ RSAKeyGenerator.java 1
④ RSAKeyGenerator.java > RSAKeyGenerator
37     keyPairGenerator.initialize(keysize: 1024);
38     KeyPair keyPair = keyPairGenerator.generateKeyPair();
39     privateKey = keyPair.getPrivate();
40     publicKey = keyPair.getPublic();
41     KeyFactory();
42
43
44     public static void KeyFactory() throws Exception {
45
46         KeyFactory x = KeyFactory.getInstance(algorithm: "RSA");
47         RSAPublicKeySpec pub = x.getKeySpec(getPublicKey(), keySpec: RSAPublicKeySpec.class);
48         RSAPrivateKeySpec pri = x.getKeySpec(getPrivateKey(), keySpec: RSAPrivateKeySpec.class);
49
50         save(save_keys_path + "\\" + username.getText() + "_public.key", pub.getModulus(), pub.getPublicExponent());
51         save(save_keys_path + "\\" + username.getText() + "_private.key", pri.getModulus(), pri.getPrivateExponent());
52
53     }
54
55
56     public static void save(String fileName, BigInteger mod, BigInteger exp) throws IOException {
57         ObjectOutputStream fileOut = new ObjectOutputStream(new BufferedOutputStream(new FileOutputStream(fileName)));
58         try {
59             fileOut.writeObject(mod);
60             fileOut.writeObject(exp);
61         } catch (Exception e) {
62             throw new IOException(message: "Unexpected error");
63         } finally {
64             fileOut.close();
65         }
66     }
67
68
69     public static PublicKey getPublicKey() {
70         return publicKey;
71     }
72
73     public static PrivateKey getPrivateKey() {
74         return privateKey;
75     }
76
77 }
```

11. SendReceiveController



```
terminal Help SendReceiveController.java - pkg_is - Visual Studio Code
① AES.java 1 ② FileController.java 1 ③ SendReceiveController.java 1
④ SendReceiveController.java > ...
5 package pkg_is;
6
7
8 import java.io.DataInputStream;
9 import java.io.DataOutputStream;
10 import java.io.File;
11 import java.io.FileInputStream;
12 import java.io.FileNotFoundException;
13 import java.io.FileOutputStream;
14 import java.io.IOException;
15 import java.net.Socket;
16
17 /**
18 *
19 * @author razan
20 */
21 public class SendReceiveController {
22
23     public void sendFile(File file, Socket socket) throws FileNotFoundException, IOException {
24
25         FileInputStream fileInputStream = new FileInputStream(file.getAbsolutePath());
26
27         DataOutputStream dataOutputStream = new DataOutputStream(socket.getOutputStream());
28
29         String fileName = file.getName();
30         byte[] fileNameBytes = fileName.getBytes();
31
32         dataOutputStream.writeInt(fileNameBytes.length);
33
34         dataOutputStream.write(fileNameBytes);
35
36         byte[] fileContentBytes = new byte[(int) file.length()];
37
38         fileInputStream.read(fileContentBytes);
39
40         dataOutputStream.writeInt(fileContentBytes.length);
41
42         dataOutputStream.write(fileContentBytes);
43
44     }
45
46     public File receiveFile(Socket socket) throws IOException {
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87 }
```



```
terminal Help SendReceiveController.java - pkg_is - Visual Studio Code
① AES.java 1 ② FileController.java 1 ③ SendReceiveController.java 1
④ SendReceiveController.java > ...
46     public File receiveFile(Socket socket) throws IOException {
47
48         System.out.println("receivefile method start ..");
49
50         DataInputStream dataInputStream = new DataInputStream(socket.getInputStream());
51
52         int fileNameLength = dataInputStream.readInt();
53
54         if (fileNameLength > 0) {
55
56             byte[] fileNameBytes = new byte[fileNameLength];
57             dataInputStream.readFully(fileNameBytes, 0, fileNameLength);
58             String fileName = new String(fileNameBytes);
59
60             System.out.println("fileName : " + fileName);
61
62             int fileContentLength = dataInputStream.readInt();
63
64             if (fileContentLength > 0) {
65
66                 byte[] fileContentBytes = new byte[fileContentLength];
67                 dataInputStream.readFully(fileContentBytes, 0, fileContentLength);
68                 String fileContent = new String(fileContentBytes);
69
70                 System.out.println("getFileExtention : " + getFileExtention(fileName));
71                 //-----PATH TO DOWNLOAD-----
72                 String pathToDownload = "C:\\Users\\razan\\Desktop\\";
73
74                 File file = fileDownloader(pathToDownload, fileName, fileContentBytes);
75
76                 return file;
77             }
78         }
79
80         return new File(pathname: "");
81     }
82
83     public String getFileExtention(String filename) {
84
85         int i = filename.indexOf('.');
86     }
```

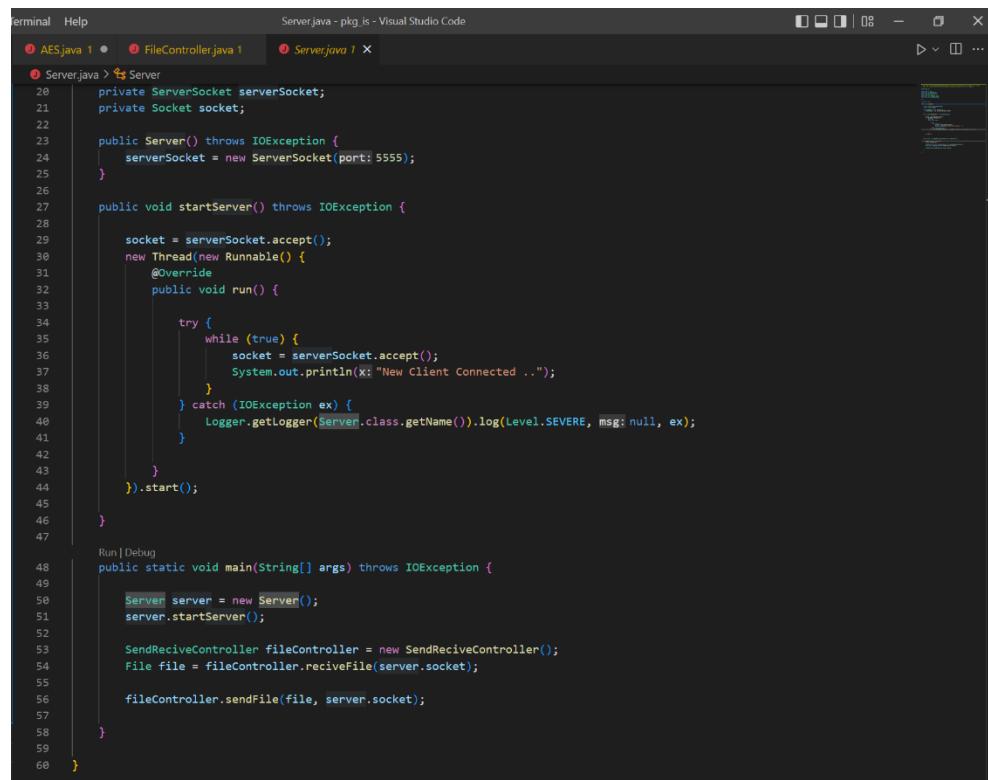
A screenshot of the Visual Studio Code interface showing the `SendReceiveController.java` file. The code implements a file download functionality. It includes methods for getting file extensions and downloading files from a specified path. The code uses Java's `File` and `FileOutputStream` classes.

```
terminal Help
SendReceiveController.java - pkg_is - Visual Studio Code
AES.java 1 ● FileController.java 1 ● SendReceiveController.java 1 X
SendReceiveController.java > ...
71     //-----PATH TO DOWNLOAD----->
72     String pathToDownload = "C:\\Users\\razan\\Desktop\\";
73
74     File file = fileDownloader(pathToDownload, fileName, fileContentBytes);
75
76     return file;
77 }
78 }
79
80 return new File(pathname: "");
81 }
82 }
83 }
84
85 public String getFileExtention(String filename) {
86
87     int i = filename.indexOf(ch: '.');
88
89     if (i > e) {
90         return filename.substring(i + 1);
91     } else {
92         return "no extention found.";
93     }
94 }
95
96
97 public File fileDownloader(String downloadPath, String fileName, byte[] fileContentBytes) throws FileNotFoundException, IOException {
98
99     File fileToDownload = new File(downloadPath + fileName);
100
101    FileOutputStream fileOutputStream = new FileOutputStream(fileToDownload);
102
103    fileOutputStream.write(fileContentBytes);
104
105    fileOutputStream.close();
106
107    return fileToDownload;
108
109 }
110
111 }
112 }
```

12. Server

A screenshot of the Visual Studio Code interface showing the `Server.java` file. The code defines a `Server` class that extends `Runnable`. It uses `ServerSocket` and `Socket` to handle client connections, printing a message when a new client connects. The server port is set to 5555.

```
terminal Help
Server.java - pkg_is - Visual Studio Code
AES.java 1 ● FileController.java 1 ● Server.java 1 X
Server.java > Server
Server.java > ↗ Server
5 package pkg_is;
6
7 import java.io.File;
8 import java.io.IOException;
9 import java.net.ServerSocket;
10 import java.net.Socket;
11 import java.util.logging.Level;
12 import java.util.logging.Logger;
13
14 /**
15 * @author razan
16 */
17 public class Server {
18
19     private ServerSocket serverSocket;
20     private Socket socket;
21
22     public Server() throws IOException {
23         serverSocket = new ServerSocket(port: 5555);
24     }
25
26     public void startServer() throws IOException {
27
28         socket = serverSocket.accept();
29         new Thread(new Runnable() {
30             @Override
31             public void run() {
32
33                 try {
34                     while (true) {
35                         socket = serverSocket.accept();
36                         System.out.println(x: "New Client Connected ..");
37                     }
38                 } catch (IOException ex) {
39                     Logger.getLogger(Server.class.getName()).log(Level.SEVERE, msg: null, ex);
40                 }
41
42             }
43         }).start();
44     }
45
46 }
```



```
Terminal Help
AESjava 1 • FileController.java Server.java 1 ×
Serverjava > Server
20     private ServerSocket serverSocket;
21     private Socket socket;
22
23     public Server() throws IOException {
24         serverSocket = new ServerSocket(port: 5555);
25     }
26
27     public void startServer() throws IOException {
28
29         socket = serverSocket.accept();
30         new Thread(new Runnable() {
31             @Override
32             public void run() {
33
34                 try {
35                     while (true) {
36                         socket = serverSocket.accept();
37                         System.out.println("New Client Connected ..");
38                     }
39                 } catch (IOException ex) {
40                     Logger.getLogger(Server.class.getName()).log(Level.SEVERE, msg: null, ex);
41                 }
42             }
43         }).start();
44     }
45
46 }
47
Run | Debug
48 public static void main(String[] args) throws IOException {
49
50     Server server = new Server();
51     server.startServer();
52
53     SendReceiveController fileController = new SendReceiveController();
54     File file = fileController.receiveFile(server.socket);
55
56     fileController.sendFile(file, server.socket);
57
58 }
59
60 }
```

○ Challenges

- how to Read the plaintext file
- Save the encrypted file as a file
- Create a TCP connection between sender and receiver
- How to connect the algorithm RSA and AES with our code
- how to protect password

○ Note

You must change the path in the code to the path in your device so it can work correctly with you.

○ References

- Stack Overflow, “Stack Overflow - Where Developers Learn, Share, & Build Careers,” Stack Overflow. <https://stackoverflow.com/>
- “Baeldung,” www.baeldung.com, Oct. 30, 2020. <https://www.baeldung.com/>
- “Novixys.com: Convert & Import XML,” www.novixys.com.
<https://www.novixys.com/> (accessed May 07, 2022).
- W3Schools, “W3Schools Online Web Tutorials,” W3schools.com, 2019.
<https://www.w3schools.com/>
- “Wiki Forum at JavaRanch,” coderanch.com.
<https://coderanch.com/f/153/Wiki> (accessed May 07, 2022).
- “Java Socket Programming - Send and Download Files Between Client and Server,” www.youtube.com.
<https://www.youtube.com/watch?v=GLrlwwyd1gY&feature=youtu.be> (accessed May 07, 2022).
- “Java select a file,” www.youtube.com.
<https://www.youtube.com/watch?v=A6sA9KltwpY> (accessed May 08, 2022).
- baeldung, “Generating a Secure AES Key in Java | Baeldung,” www.baeldung.com, Jan. 22, 2022.
<https://www.baeldung.com/java-secure-aes-key> (accessed May 08, 2022).
- “Java Examples & Tutorials of KeyGenerator.init (javax.crypto) | Tabnine,” www.tabnine.com.
<https://www.tabnine.com/code/java/methods/javax.crypto.KeyGenerator/init> (accessed May 08, 2022).