**Eastern Mediterranean University**



**Department of Computer Engineering**

**Software Engineering Program**

**Famagusta, North Cyprus**

**Term Project Report**

**CMSE 353**

**Secure Software System**

**Using Access Control Lists and DES Implementation**

**Students:**

1. Talal Hatem Mahdy (147139)
2. Abdoulgwad Hussien Elsheredi (147597)
3. Mohamed M. M. Balto (147697)

**Instructor:** Assoc. Prof. Dr. Alexander Chefranov

**Lab Coordinator:** Armin Mehri

**December 2016**

**Outline:**

1. Problem Definition 2  
2. Task Formulation 2  
3. Tools used in this project 3  
4. Description of the Algorithm used in this Application 3-5  
5. Description of the Developed Program 6-8  
6. User Guide 9-15  
7. Conclusion 16  
8. References 19  
9. Appendices 20-59

**1. Problem Definition:**

The aim of this project is to develop a secure software system which can securely store items such as notes, images with passwords for each user. The password and the notes for each user should be encrypted using the Data Encryption Standard (DES) Algorithm with the Counter Mode (CTR) Variant. The Access Model used to determine what each user can access is the Access Control Lists (ACL) Model along with a Capability Lists (CL) Model for some parts of the Database.

**2. Task Formulation:**

This is a complex project so various phases to develop the software had to be created.

1st Phase (11.11.2016 – 27.11.2016):  
Project Initiation; Team members meetings; Distribute responsibilities of each member; Understand the exact requirements of the project; Agree on the development environment to be used; Develop a basic human - software scenario of the software to be developed; Determine the various Modules to be used; Conduct a small presentation with a stakeholder of the project Dr. Alexander Chefranov to clear the doubts about the project.

2nd Phase (28.11.2016 – 4.12.2016):   
Write down the DES CTR Algorithm using the C++ Language; Implement this DES CTR Algorithm in the Visual Basic Language; Develop the login Module and Implement the DES CTR Algorithm to encrypt the Password; Develop the Notes Module to store the Notes of a user and Implement the DES CTR on this module.

3rd Phase (5.12.2016 – 16.12.2016):   
Develop the Database using ACL and CL to be used for the Users and Passwords Module; Connect this database to the modules created in the Visual Basic Application; Save this database on an online hosting service (MS Azure) so that we do not lose the contents of the database; Create the Images Module; Fix bugs in previously created modules; Develop the User Interface in Visual Basic.

4th Phase (17.12.2016 – 25.12.2016):   
Fix more bugs in the created modules; Improve the User Interface Design; Prepare the project for delivery; Prepare Final Report, Presentation and CD.

**3. Tools used in this project:**

1. Microsoft Visual Basic: The main parts of the application were developed using Visual Basic.  
2. Microsoft Visual C++: The DES CTR Algorithm was written using C++.  
3. Microsoft SQL Server Management: The Database for the Application was developed using MS SQL Server Management.  
4. Microsoft Azure Cloud Services: The Database was hosted on these servers.

**4. Description of the Algorithm used in this Application (See Appendix K):**The encryption algorithm used in the development of this application is the Data Encryption Standard (DES) along with the Counter (CTR) Mode of Operation. This Algorithm is used to encrypt the Password and Notes of a user.   
**4.1. How DES Works in the program:**

**Algorithm:**

**Step 1:** Start the program.

**Step 2:** Declare the necessary variables and functions.

**Step 3:** Get the plain text from the user

**Step 4:** Generate the key for encrypting the plain text.

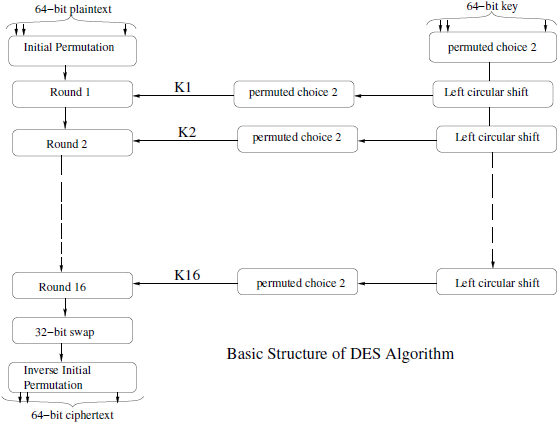
**Step 5:** Encrypt the plain text using the generated key.

**Step 6:** Display the encrypted text.

**Step 7:** Terminate the program.

**4.2. How DES Encryption works in detail:**

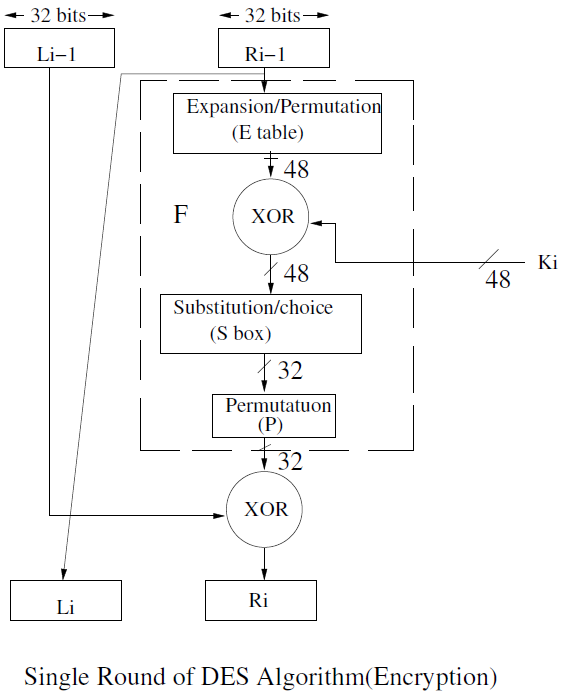
The 64-bit input is subjected to an initial permutation to obtain the 64-bit result (i.e. just the input with the bits shuffled). The 64-bit key is passed throw a permutation function to get the 56-bit key which is used to generate sixteen 48-bit per round keys, by taking a different 48-bit subset of the 56-bits for each of the keys. Each round takes as input the 64-bit output of the previous round, and the 48-bit per-round key, and produces a 64-bit output. After the 16th round, the 64-bit output has its halves swapped and is then subjected to another permutation, which happens to be the inverse of the initial permutation.



**4.2.1. Details of Single Round:**

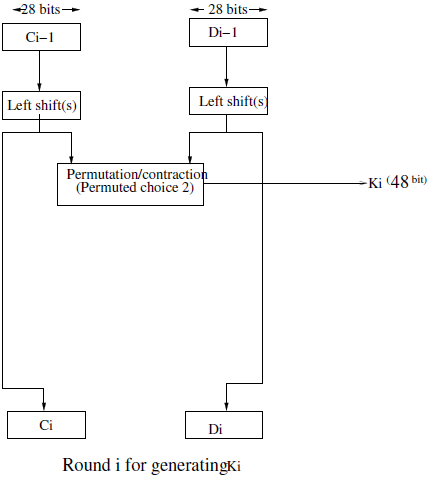
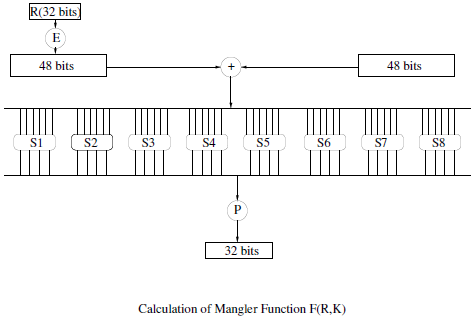
The left and right halves of each 64-bit intermediate value are treated as separate 32-bit quantities, labeled L (left) and R (right). The overall processing formulas at each round are as follows:

Li =Ri -1  
Ri =Li -1 XOR F ( Ri -1 , Ki )   
The round key Ki is 48-bits. The R input is 32-bits. This R input is first expanded to 48-bits using permutation plus an expansion table. The resulting 48-bits are XORed with Ki. This 48-bit result passes through a substitution function that produces a 32-bit output in each halves to get a 64-bit output. The substitution consists of a set of eight S-boxes, in the mangler function F, each of which accepts 6-bits as input and produces 4-bits as output.



**4.2.2. Key Generation:**

The bits of the 64-bit input key are numbered from 1 through 64 and every eighth bit is ignored. The key is first subjected to permutation choice one, the resulting 56-bit key is then treated as two 28-bit quantities, labeled C0 and D0. At each round, Ci -1 and Di -1 are separately subjected to a circular left shift or rotation of 1 or 2 bits that serve as input to next round and also to permuted choice two, which produces a 48-bit output that serves as input to the manglar function F(Ri-1 ,Ki ).

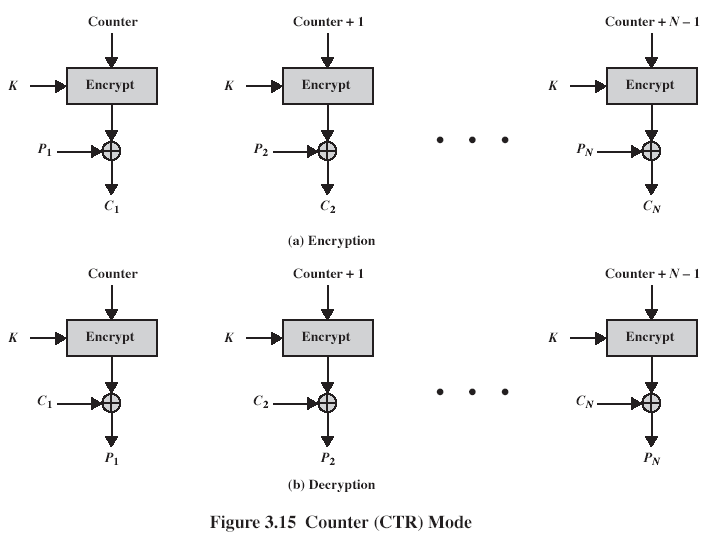


**4.2.3. DES Decryption:**

DES Decryption by essentially running DES Algorithm backwards.

**4.2.4. Description of Counter Mode:**

The DES Algorithm implemented in the program was modified to support Counter Mode of Operation. This Mode operates as follows:



**5. Description of the Developed Program (See Appendix A - J ):**

The program was developed using Microsoft Visual Basic. The program consists of 10 main parts/modules (Also known as forms in Visual Basic). These 10 modules were combined to form our Secure Software System.

**5.1. Description of Modules:**

The 10 Modules which form the Secure Software System are:

1. Add\_New\_user: This is the module that runs when we choose to Add/Delete a user to the program.   
2. Adding\_image: This is the module that runs when we choose to View/Add/Delete/Save an image for a user.  
3. images\_list: This is the module responsible for handling and displaying the images when clicking the images button. It also shows the images as a thumbnail to reduce loading time.  
4. Login: This module is run when logging in to the system either as a user or as an administrator.   
5. new form: This module is used to handle the encryption/decryption of notes.  
6. Notes: This module is run when we want to Add/Delete/Edit notes.   
7. Notes\_Access: This module runs when we want to give a specific user a Read/Write access to another user’s note.   
8. pass\_change: This module is run when we want to change the password of the user.   
9. User\_Options: This module controls the general display of the program and what happens when certain interactions take place.   
10. Users\_List: This module runs when we want to control what the users can access. It also updates these access values in their respective places in the Database.

**5.2. Description of the Database:**

The Database that interacts with the program was developed using the Microsoft SQL Server Management tool. This Database was linked with Visual Basic and we were able to modify the values in it using the Visual Basic code. The database contains 6 tables. The tables along with their attributes are:

**1. ImagesData** (Image\_ID, Image)   
**2. Images** (Image, Date\_Created, image\_name)  
**3. Users** (ID, User\_Name, Password, Date\_Created, Last\_Update, Admin)  
**4. Notes** (Note\_ID, ID, Note\_Title, Note, Date\_Created)  
**5. Notes\_Access** (Note\_id, ID, Edit)  
**6. ACCESS** (ID, Users\_e, Images, Notes)

The Access Control Model used in the Notes\_Access table is Access Control Lists while the Model used in the ACCESS table is the Capability Lists Model (See Appendix L).

Due to some limitations with the MS SQL Server Management tool, this database cannot be linked to the Visual Basic program when copying the program to another computer or running the program from a Flash Drive or a CD.  
Therefore, the program will not work on another computer unless we use an online Database hosting service such as Microsoft Azure Cloud Services or else MS SQL Server Management tool has to be installed on each and every computer planning to run the program which is not a convenient option. Therefore, we created a Microsoft Account and linked our Database to the Microsoft Azure Cloud Services. We were then given a host address (termproject2.database.windows.net) for our Microsoft Account to be used in the Visual Basic Program. The advantage to implementing this online hosting method in our program is that our database is always up to date, synchronized and backed up on a reliable online server. But a slight disadvantage to using this method is that the computer should have internet access when using the program. But today’s computers are almost always connected to the Internet so this is not a big problem. For example, when a user wants to add a new image, this image is automatically uploaded to Microsoft Azure online hosting service.   
Figures 1-4 include some pictures from the Database. The rest of the pictures can be found in the CD bundled with this report.

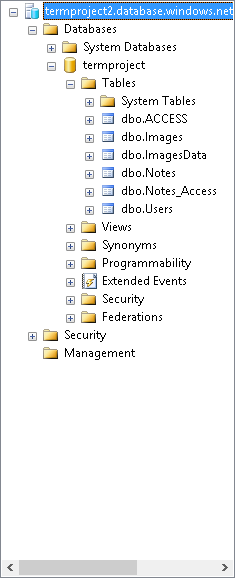


Figure 1. Overview of the Database

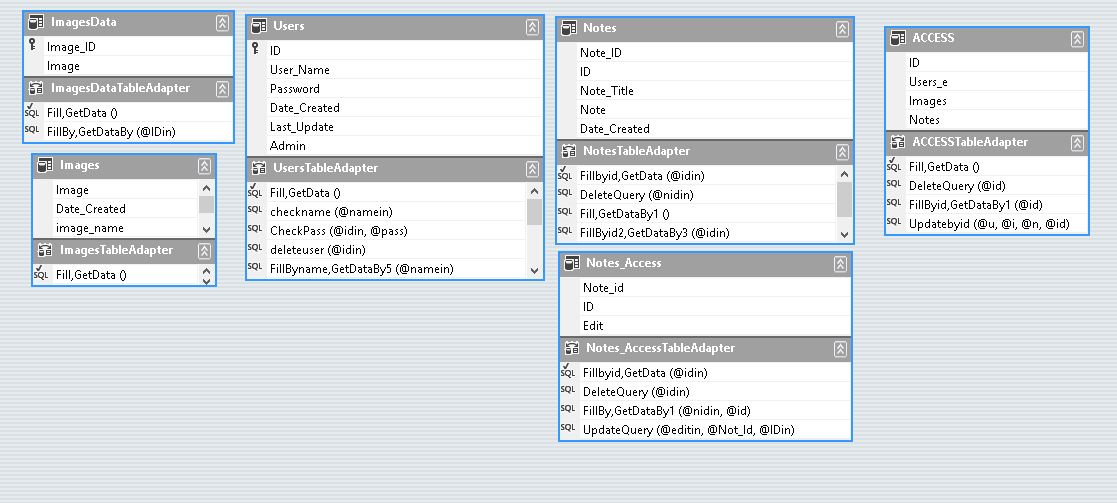


Figure 2. Database tables and their attributes

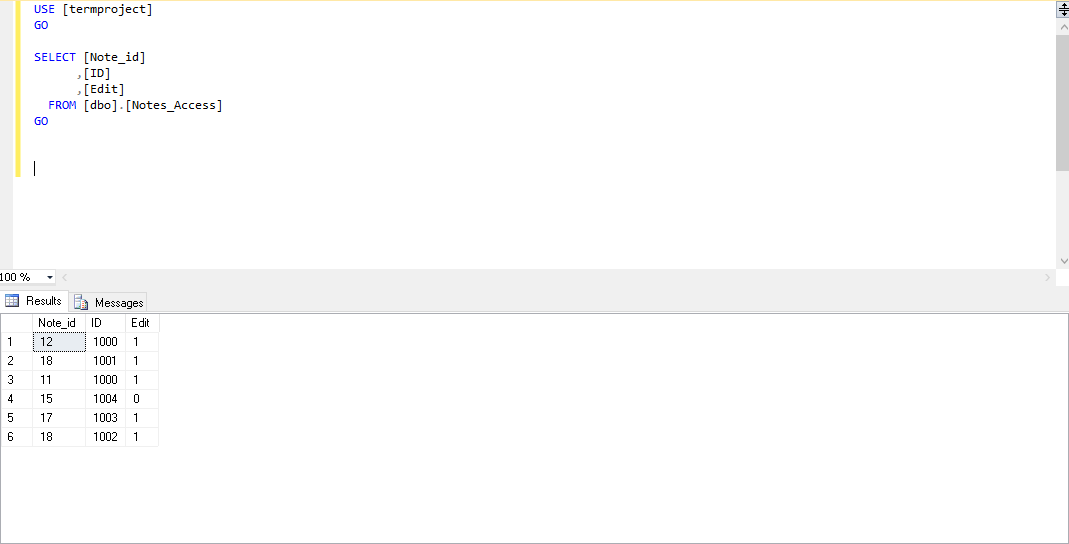


Figure 3. Table Notes\_Access which implements the Access Control Lists Model

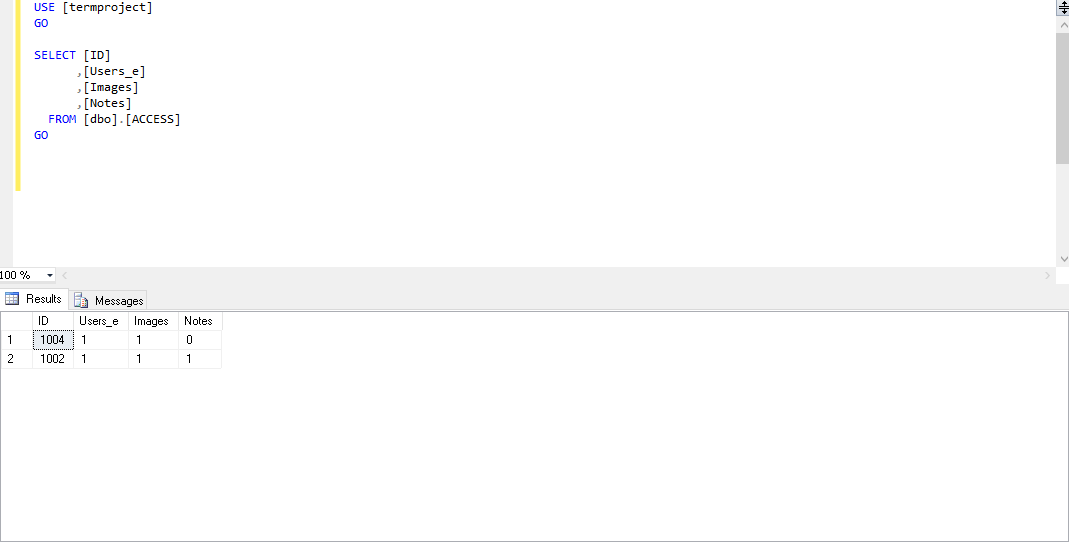


Figure 4. Table ACCESS which implements the Capability Lists Model

6. User Guide:

Please note that an Internet Connection is required when running the Program.  
This program can be simply opened by running the Secure Software System.exe   
When you run the program, it will ask you to enter a User Name and a Password. There are two types of accounts, an Administrator Account and a normal User Account. If you login with an Administrator Account, you will be given Administrator-only Privileges and if you login with a User account, you will be given User Privileges. The Default Administrator Account for this program is:   
User Name: admin  
Password: 123456

Upon logging in, you will be confronted with 4 options which are:   
1. Editing Users.  
2. Notes.  
3. Images.  
4. Change Password.

**6.1. Editing Users:**If you are an Administrator, you will be able to do the following functions:   
1. Add/Remove a User.   
2. Add/Remove Privileges from other users.  
3. Change the User Name/Password of other users.

If you are a User, you will only be able to remove your privileges but you will not be able to add new privileges in which an administrator did not give you access to. Note that you won’t be able to create a new administrator account if you are logged in as a user account.

**6.2. Notes:**

Clicking on the ‘+’ button allows you to add a new note. When you click it, you’ll be asked to enter a Note title and the Note itself. You will also be asked if you want to encrypt the note or not. If you choose to encrypt it, you will be asked to enter an encryption password (key). This key should consist of 8 characters. If you enter less than 8, an error will appear asking you to enter a different key. If you enter more than 8, only the first 8 characters will be used when decrypting your note. If you click Cancel, a message appears informing the user that he failed to enter a password. Clicking on the ‘-‘ sign deletes a note. You can edit a note and its title by clicking on the Edit button then click on the update button when you finish editing the note. If you find that a note is encrypted, you can decrypt it by clicking on the Decrypt button and entering the key. If you are administrator, you will also have access to the ‘Notes Access’ function. This option allows administrators to give a special privilege which allows a specific user to access a specific note mentioned by the administrator. The administrator can choose a user and a note to be given to that user. If the administrator clicks on the Edit checkbox, that user will also be able to edit that note.

**6.3. Images:**

A user can add a new image by clicking on the ‘+’ button. When you click it, you’ll be asked to enter a Picture Name and to open an image file from the computer. The program will only accept images of type JPG. You can then upload the image by clicking the Upload button. When you are viewing an image, you will notice that the image is a low quality small image. This is done to speed up the preview process of an image and to eliminate long waiting times for the image to download from the hosting server. To view the full size image in high resolution, click the High Resolution Button. When you click the High Resolution Button, you will also be given the option to save the image to your computer. Clicking on the ‘-‘ button deletes an image.

**6.4. Change Password:**

This option allows you the change the password for your account. You will be asked to enter your new password twice and if they are not matching, the New Password bar will start glowing in a red color to notify the user.

**7. Description of Conducted Tests with Screenshots**:



Figure 5. Running of the Program. Logging in with an Administrator Account.

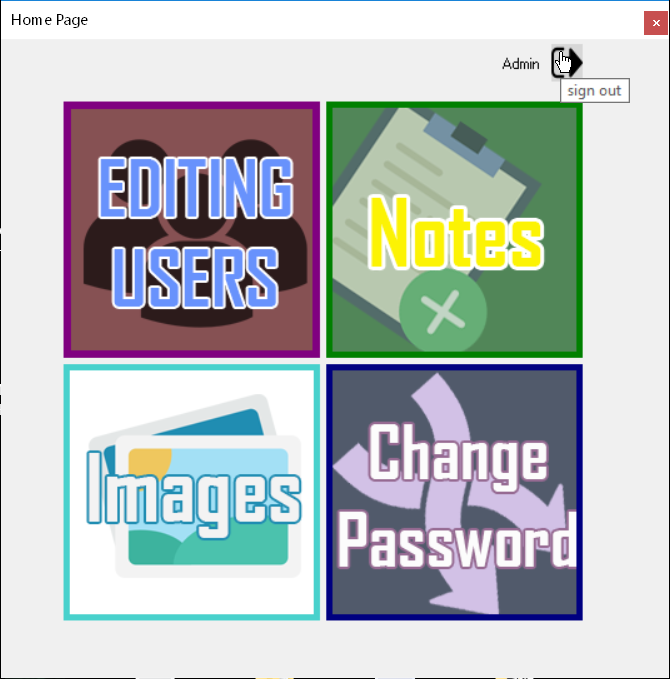


Figure 6. Main interface of the program.

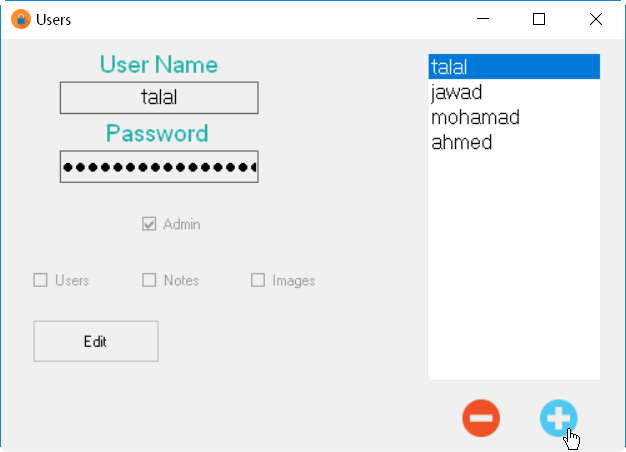


Figure 7. Editing Users Section.

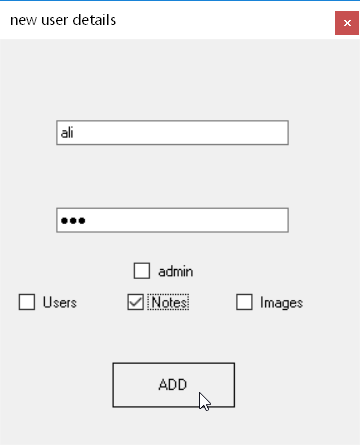


Figure 8. Adding a new user and granting him access to the Notes section.

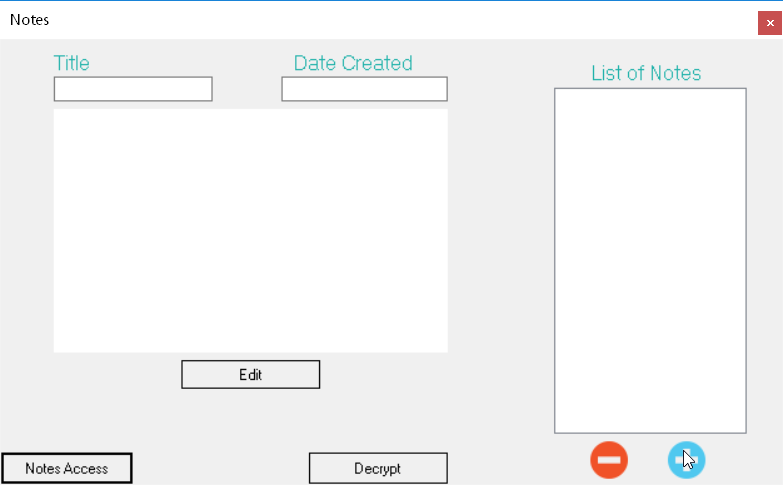


Figure 9. Notes Section.

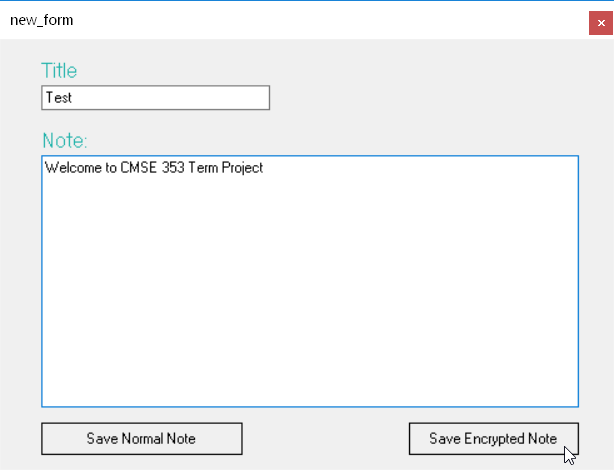


Figure 10. Create a new note and choose to encrypt it.

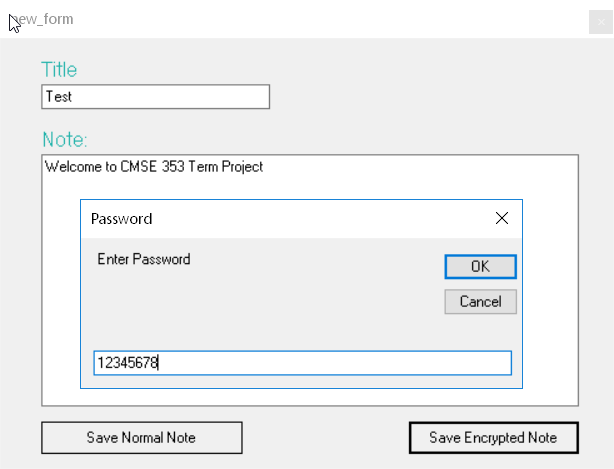


Figure 11. Enter the Encryption key.

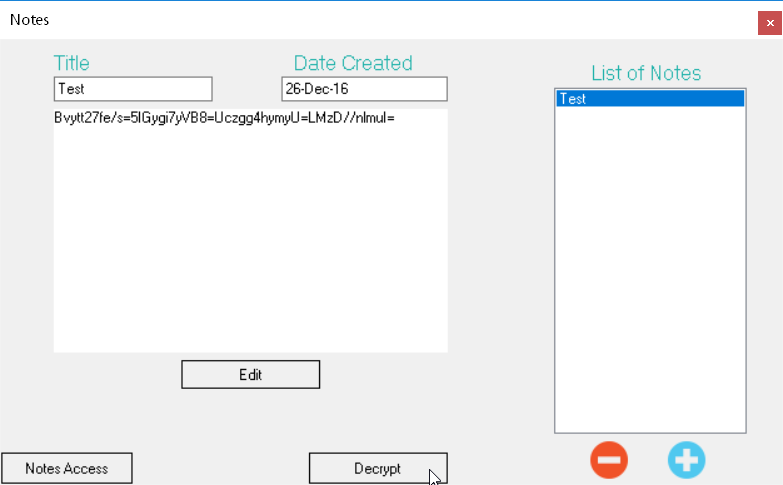


Figure 12. Note is saved and encrypted.

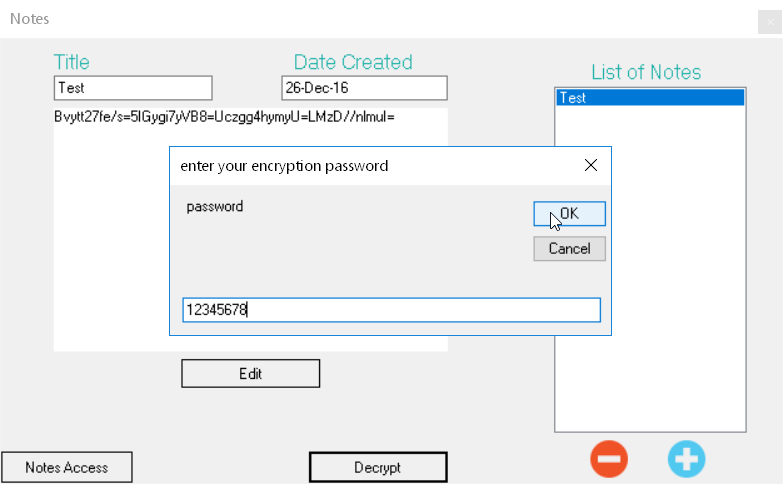


Figure 13. Decrypt the note. Enter the encryption key.

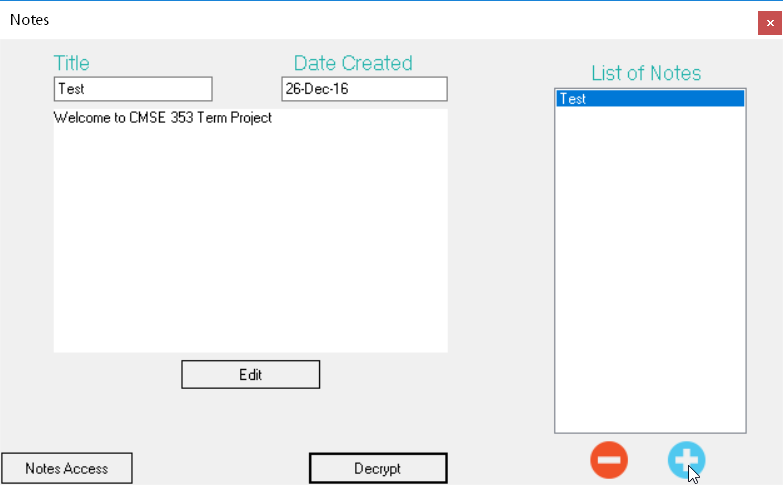


Figure 14. Note is decrypted.

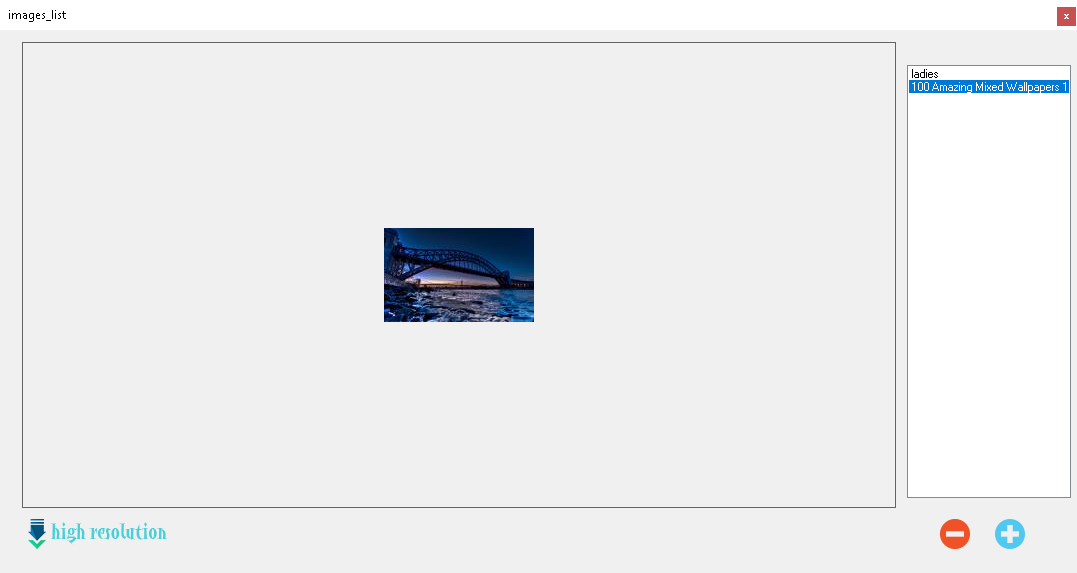


Figure 15. Images Section. Image is displayed in a small size and resolution.



Figure 16. Choose the high resolution option and choose to save it.



Figure 7. Click the ‘+’ button to upload a new image.

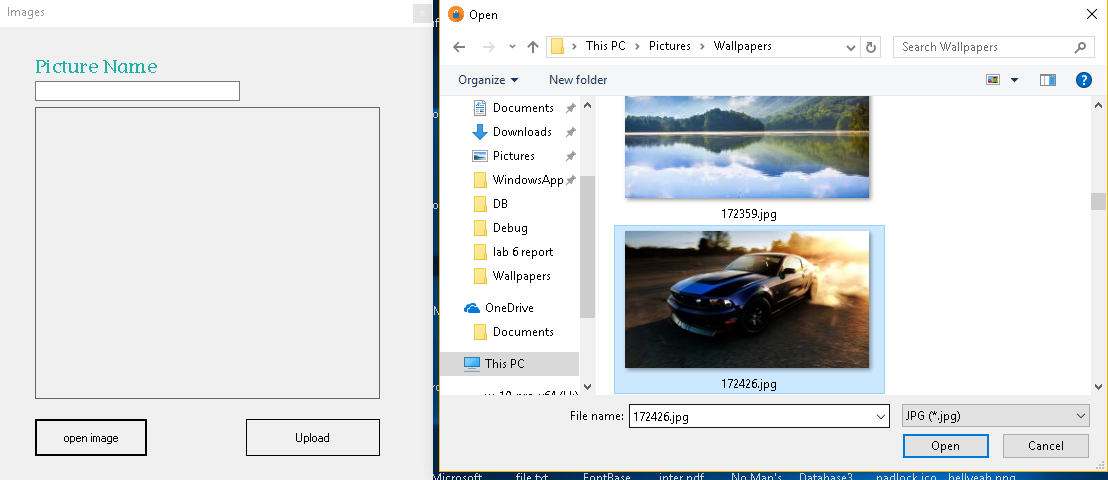
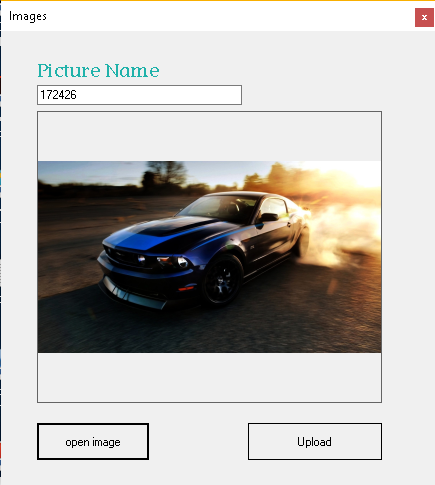


Figure . Select the Image from your computer.



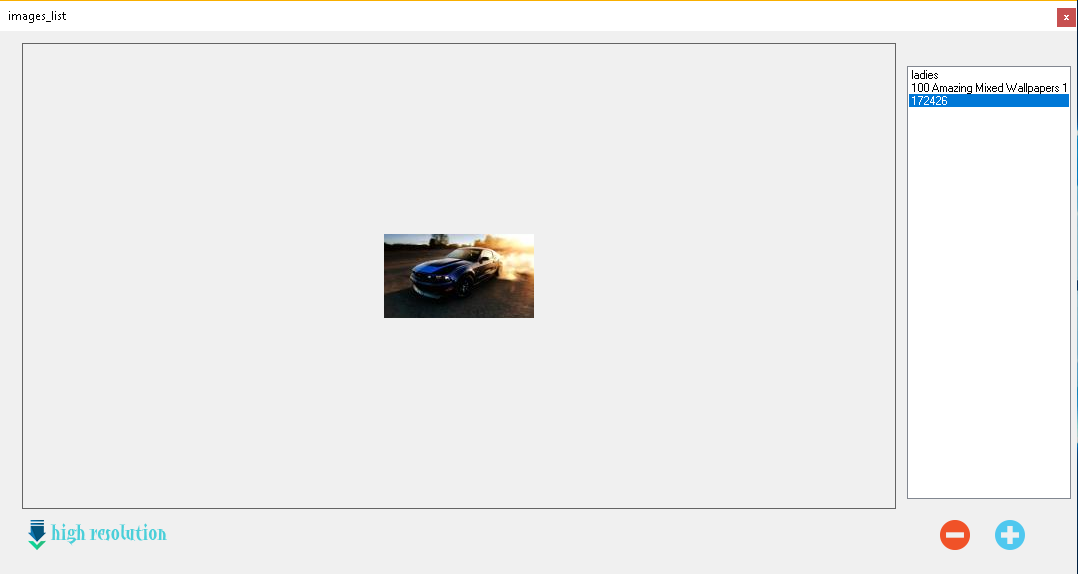


Figure 18-19. Click Upload. The image is uploaded and saved to the database.

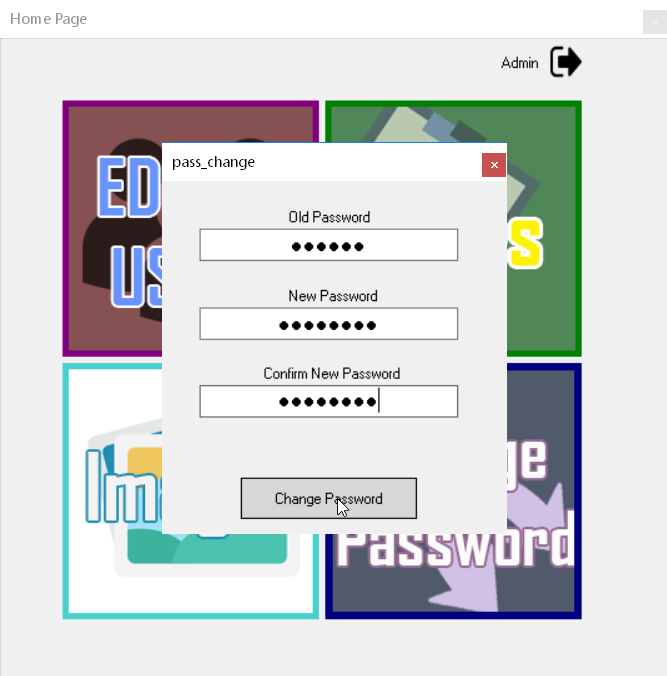


Figure 20. Change Password Section.

**8. Conclusion:**

After conducting numerous tests, we can finally say that this Secure Software System product is ready to start service. This project was a complex project and needed a lot of teamwork, organization and time to complete. The application was developed mainly using Visual Basic. The system consisted of 10 modules each of them interacting with a special part of the database associated with the program. The main features of the Application are to securely store notes and images for each user in the application and onto the database. The password and the notes section of the software is encrypted using DES Algorithm and saved to the database. Microsoft SQL Server Management was the database tool used with the application. This database was linked to Microsoft Azure which is an online hosting service for our database to stay synchronized with our application. New privileges to users can be given only by administrators. The Access Control Models used in the database are Access Control Lists and Capability Lists. The software was developed with an easy to use and a minimalistic design.

## *References*

*Adding Pictures to Your Application*. *Msdn.microsoft.com*. Retrieved 25 December 2016, from https://msdn.microsoft.com/en-us/library/aa261322(v=vs.60).aspx

Biswas, P. (2014). *Access Control List vs Capability List*. *prosuncsedu.wordpress.com*. Retrieved 25 December 2016, from https://prosuncsedu.wordpress.com/2014/08/21/comparing-object-centric-access-control-mechanisms-acl-capability-list-attribute-based-access-control/

*DES in C*. (2010). *GitHub*. Retrieved 25 December 2016, from https://github.com/tarequeh/DES

*Get started with Azure Cloud Services and ASP.NET*. *Docs.microsoft.com*. Retrieved 25 December 2016, from https://docs.microsoft.com/en-us/azure/cloud-services/cloud-services-dotnet-get-started

*How To Open a SQL Server Database by Using the SQL Server .NET Data Provider with Visual Basic .NET*. *Support.microsoft.com*. Retrieved 25 December 2016, from https://support.microsoft.com/en-us/kb/308656

*How to: Connect to a SQL Server Database Using the SqlDataSource Control*. *Msdn.microsoft.com*. Retrieved 25 December 2016, from https://msdn.microsoft.com/en-us/library/ms247242(v=vs.100).aspx

*Simple Login Tutorial - Visual Basic .NET*. *Visual-basic-tutorials.com*. Retrieved 25 December 2016, from http://www.visual-basic-tutorials.com/form/LoginT.htm

*System.Security.Cryptography Namespace*. *msdn.microsoft.com*. Retrieved 25 December 2016, from https://msdn.microsoft.com/en-us/library/system.security.cryptography(v=vs.110).aspx

*The DES Algorithm Illustrated*. *page.math.tu-berlin.de*. Retrieved 25 December 2016, from http://page.math.tu-berlin.de/~kant/teaching/hess/krypto-ws2006/des.htm

*Use SQL Server Management Studio*. *Msdn.microsoft.com*. Retrieved 25 December 2016, from https://msdn.microsoft.com/en-us/library/ms174173.aspx

*Windows Forms Application Basics (Visual Basic)*. *Msdn.microsoft.com*. Retrieved 25 December 2016, from https://msdn.microsoft.com/en-us/library/ms172749.aspx

**APPENDIX A**

**Add\_New\_user Module Source Code:**

Public Class Add\_New\_user

Dim userid As String

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

Dim adds As String = "INSERT INTO Users ( User\_Name, Password, Date\_Created, Last\_Update) VALUES(@UName,@Password,@DC,@LM)"

If (TextBox1.Text <> "" And TextBox2.Text <> "") Then

'UsersTableAdapter1.Insert(TextBox1.Text, TextBox2.Text, Date.Now, Date.Now)

If (UsersTableAdapter1.checkname(TextBox1.Text)) Then

MsgBox("name is already taken")

Exit Sub

End If

Dim now As Date = Date.Now

UsersTableAdapter1.Insert(TextBox1.Text, Login.seperatedenc(TextBox2.Text), now, now, CheckBox1.Checked)

If (CheckBox1.Checked <> True) Then

UsersTableAdapter1.Fill(ProjectDataSet1.Users)

Dim intid() As DataRow = ProjectDataSet1.Users.Select("User\_Name='" + TextBox1.Text + "'")

AccessTableAdapter1.Insert(intid(0).Item("ID"), CheckBox2.Checked, CheckBox3.Checked, CheckBox4.Checked)

End If

Users\_List.Show()

Me.Close()

End If

End Sub

Private Sub Form4\_Closed(sender As Object, e As EventArgs) Handles Me.Closed

Users\_List.Show()

End Sub

Private Sub CheckBox1\_CheckedChanged(sender As Object, e As EventArgs) Handles CheckBox1.CheckedChanged

If (CheckBox1.Checked) Then

CheckBox2.Enabled = False

CheckBox3.Enabled = False

CheckBox4.Enabled = False

Else

CheckBox2.Enabled = True

CheckBox3.Enabled = True

CheckBox4.Enabled = True

End If

End Sub

Private Sub Add\_New\_user\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

If (Login.admin) Then

CheckBox1.Enabled = True

CheckBox2.Enabled = True

Else

CheckBox1.Enabled = False

CheckBox2.Enabled = False

End If

End Sub

End Class

**APPENDIX B**

**Adding\_image Module Source Code:**

Imports System.IO

Public Class addingImages

Private Sub Images\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

End Sub

Public thumbimage As Image = Nothing

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

If OpenFileDialog1.ShowDialog() = DialogResult.OK Then

PictureBox1.Image = Image.FromFile(OpenFileDialog1.FileName)

TextBox1.Text = Path.GetFileNameWithoutExtension(OpenFileDialog1.FileName)

Dim image2 As Image = PictureBox1.Image

thumbimage = Nothing

thumbimage = image2.GetThumbnailImage(150, (PictureBox1.Image.Height / PictureBox1.Image.Width) \* 150, Nothing, New IntPtr())

PictureBox2.Image = thumbimage

End If

End Sub

Private Sub Images\_Closed(sender As Object, e As EventArgs) Handles Me.Closed

images\_list.Show()

End Sub

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

If OpenFileDialog1.FileName = "" Then

MsgBox("Choose a picture first")

Exit Sub

End If

If TextBox1.Text = "" Then

MsgBox("please enter a name for your image")

Exit Sub

End If

Dim ms As New MemoryStream

Dim thumbn As New MemoryStream

PictureBox2.Image.Save(thumbn, Imaging.ImageFormat.Jpeg)

PictureBox1.Image.Save(ms, PictureBox1.Image.RawFormat)

Dim data As Byte() = ms.GetBuffer

Dim thdata As Byte() = thumbn.GetBuffer

ImagesTableAdapter.InsertQuery(thdata, Date.Now, TextBox1.Text, Login.userid)

ImagesDataTableAdapter1.Insert(data)

ms.Flush()

ms.Close()

thumbn.Flush()

thumbn.Close()

Me.Close()

End Sub

Public Function ThumbnailCallback() As Boolean

Return True

End Function

Private Sub GetThumbnail(ByVal e As PaintEventArgs)

Dim callback As New Image.GetThumbnailImageAbort(AddressOf ThumbnailCallback)

Dim image As Image = New Bitmap("c:\FakePhoto.jpg")

Dim pThumbnail As Image = image.GetThumbnailImage(100, 100, callback, New IntPtr())

e.Graphics.DrawImage(pThumbnail, 10, 10, pThumbnail.Width, pThumbnail.Height)

End Sub

End Class

**APPENDIX C**

**images\_list Module Source Code:**

Imports System.IO

Public Class images\_list

Private Sub images\_list\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

Dim width As Integer = My.Computer.Screen.WorkingArea.Width

Dim height As Integer = My.Computer.Screen.WorkingArea.Height

If (Login.admin = True) Then

ImagesTableAdapter.Fill(PROJECTDataSet.Images)

Else

ImagesTableAdapter.FillByid(PROJECTDataSet.Images, Login.userid)

End If

Me.Size = New Size(0.8 \* width, 0.8 \* height)

PictureBox1.Location = New Point(Me.Size.Width \* 0.02, Me.Size.Height \* 0.02)

PictureBox1.Size = New Size(Me.Size.Width \* 0.8, Me.Size.Height \* 0.8)

ListBox1.Location = New Point(Me.Size.Width \* 0.83, Me.Size.Height \* 0.06)

Button1.Location = New Point(Me.Size.Width \* 0.02, Me.Size.Height \* 0.84)

ListBox1.Size = New Size(Me.Size.Width \* 0.15, Me.Size.Height \* 0.76)

PictureBox1.SizeMode = PictureBoxSizeMode.CenterImage

delete.Location = New Point(Me.Size.Width \* 0.86, Me.Size.Height \* 0.84)

addnew.Location = New Point(Me.Size.Width \* 0.91, Me.Size.Height \* 0.84)

Button2.Location = New Point(Me.Size.Width \* 0.68, Me.Size.Height \* 0.84)

End Sub

Private Sub ListBox1\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ListBox1.SelectedIndexChanged

If PictureBox1.Image.Width > 150 Then

PictureBox1.SizeMode = PictureBoxSizeMode.Zoom

Button2.Visible = True

Else

PictureBox1.SizeMode = PictureBoxSizeMode.CenterImage

Button2.Visible = False

End If

End Sub

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

ImagesDataTableAdapter1.FillBy(PROJECTDataSet.ImagesData, ListBox1.SelectedValue)

PictureBox1.SizeMode = PictureBoxSizeMode.Zoom

Dim pic() As Byte = PROJECTDataSet.ImagesData.Item(0).Image

Dim strm As New MemoryStream()

strm.Write(pic, 0, pic.Length())

PictureBox1.Image = Image.FromStream(strm)

Button2.Visible = True

End Sub

Private Sub images\_list\_Closed(sender As Object, e As EventArgs) Handles Me.Closed

If (addingImages.Visible = False) Then

User\_Options.WindowState = FormWindowState.Normal

End If

End Sub

Private Sub addnew\_Click(sender As Object, e As EventArgs) Handles addnew.Click

addingImages.Show()

Me.Close()

End Sub

Private Sub delete\_Click(sender As Object, e As EventArgs) Handles delete.Click

ImagesDataTableAdapter1.Delete(ListBox1.SelectedValue)

ImagesTableAdapter.Deletebyimageid(ListBox1.SelectedValue)

If (Login.admin = True) Then

ImagesTableAdapter.Fill(PROJECTDataSet.Images)

Else

ImagesTableAdapter.FillByid(PROJECTDataSet.Images, Login.userid)

End If

End Sub

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

If SaveFileDialog1.ShowDialog() = DialogResult.OK Then

PictureBox1.Image.Save(SaveFileDialog1.FileName)

End If

End Sub

End Class

**APPENDIX D**

**Login Module Source Code:**

Imports System.Data.SqlClient

Imports System.Security.Cryptography

Imports System.Security

Imports System.Text

Imports System.IO

Imports System.Math

Public Class Login

Public wrapper As New SimpleDes("57962384")

Public wrapperfornotes As New SimpleDes("57962384")

Public kn As Boolean = False

Public userid As Integer = 0

Public userName As String

Public passv As String

Public admin As Boolean = False

Private deswrap As Object

Public Function seperatedenc\_withpassword(ByVal sr As String, ByVal ps As String) As String

Dim n As Integer = sr.Length() Mod 8

Dim m As Integer = Truncate(sr.Length() / 8)

Dim st As String = ""

Dim inpass As String = ""

Dim full As String = ""

Dim ctrm As Int64 = 25698742

If (ps.Length() > 8) Then

For i As Integer = 0 To 7

inpass += ps(i)

Next

wrapperfornotes.newpass(inpass)

ElseIf (ps.Length() < 8) Then

inpass = ps

For i As Integer = (ps.Length() - 1) To 7

inpass += " "

Next

wrapperfornotes.newpass(inpass)

Else

wrapperfornotes.newpass(ps)

End If

If m > 0 Then

For i As Integer = 0 To (m - 1)

For j As Integer = 0 To 7

st += sr((i \* 8) + j)

Next

full += wrapperfornotes.EncryptData(((ctrm + i) Mod 99999999).ToString, System.Text.ASCIIEncoding.ASCII.GetBytes(st))

st = ""

Next

End If

Dim extra As String = ""

If n <> 0 Then

For i As Integer = 0 To n - 1

extra += sr((m \* 8) + i)

Next

For i As Integer = n To 7

extra += Nothing

Next

Dim tryit(7) As Byte

For i As Integer = 0 To n - 1

tryit(i) = System.Text.ASCIIEncoding.ASCII.GetBytes(extra).ToArray(i)

Next

full += wrapperfornotes.EncryptData(((ctrm + m) Mod 99999999).ToString, tryit)

End If

Return full

End Function

Public Function seperatedenc(ByVal sr As String) As String

Dim n As Integer = sr.Length() Mod 8

Dim m As Integer = Truncate(sr.Length() / 8)

Dim st As String = ""

Dim full As String = ""

Dim ctrm As Int64 = 25698742

If m > 0 Then

For i As Integer = 0 To (m - 1)

For j As Integer = 0 To 7

st += sr((i \* 8) + j)

Next

full += wrapper.EncryptData(((ctrm + i) Mod 99999999).ToString, System.Text.ASCIIEncoding.ASCII.GetBytes(st))

st = ""

Next

End If

Dim extra As String = ""

If n <> 0 Then

For i As Integer = 0 To n - 1

extra += sr((m \* 8) + i)

Next

For i As Integer = n To 7

extra += " "

Next

full += wrapper.EncryptData(((ctrm + m) Mod 99999999).ToString, System.Text.ASCIIEncoding.ASCII.GetBytes(extra))

End If

Return full

End Function

Public Function seperateddecwithpass(ByVal sr As String, ByVal ps As String) As String

Dim n As Integer = sr.Length() Mod 12

Dim m As Integer = sr.Length() / 12

Dim st As String = ""

Dim full As String = ""

Dim ctrm As Int64 = 25698742

wrapperfornotes.newpass(ps)

If m > 0 Then

For i As Integer = 0 To (m - 1)

For j As Integer = 0 To 11

st += sr((i \* 12) + j)

Next

full += wrapperfornotes.EncryptData2(((ctrm + i) Mod 99999999).ToString, Convert.FromBase64String(st))

st = ""

Next

End If

Dim extra As String = ""

If n <> 0 Then

For i As Integer = 0 To n - 1

extra += sr((m \* 12) + i)

Next

For i As Integer = n To 11

extra += " "

Next

full += wrapperfornotes.EncryptData2(((ctrm + m) Mod 99999999).ToString, Convert.FromBase64String(extra))

End If

Return full

End Function

Public Function seperateddec(ByVal sr As String) As String

Dim n As Integer = sr.Length() Mod 12

Dim m As Integer = sr.Length() / 12

Dim st As String = ""

Dim full As String = ""

Dim ctrm As Int64 = 25698742

If m > 0 Then

For i As Integer = 0 To (m - 1)

For j As Integer = 0 To 11

st += sr((i \* 12) + j)

Next

full += wrapper.EncryptData2(((ctrm + i) Mod 99999999).ToString, Convert.FromBase64String(st))

st = ""

Next

End If

Dim extra As String = ""

If n <> 0 Then

For i As Integer = 0 To n - 1

extra += sr((m \* 12) + i)

Next

For i As Integer = n To 11

extra += " "

Next

full += wrapper.EncryptData2(((ctrm + m) Mod 99999999).ToString, Convert.FromBase64String(extra))

End If

Return full

End Function

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles LogIn\_Button.Click

On Error Resume Next

If (User\_Name.Text = "") Then

MsgBox("enter username")

Exit Sub

ElseIf (Password.Text = "") Then

MsgBox("enter Password")

Exit Sub

End If

UsersTableAdapter1.FillBynamefullrow(ProjectDataSet1.Users, User\_Name.Text)

'Dim value As Integer = UsersTableAdapter.usernamecheck(User\_Name.Text)

Dim usersrows() As DataRow = ProjectDataSet1.Users.Select

If (usersrows.Count = 0) Then

MsgBox("Wrong UserName Or Password", MsgBoxStyle.OkOnly, "Error")

Exit Sub

ElseIf (usersrows.Count = 1) Then

passv = usersrows(0).Item("Password")

'passv = UsersTableAdapter.getPass(User\_Name.Text)

passv = passv.Replace(" ", "") 'should be fixed

Else

MsgBox("Wrong UserName Or Password", MsgBoxStyle.OkOnly, "Error")

Exit Sub

End If

If (seperatedenc(Password.Text) = passv) Then

'userid = UsersTableAdapter.GetID\_login(User\_Name.Text)

userid = usersrows(0).Item("ID")

userName = (usersrows(0).Item("User\_Name")).replace(" ", "")

admin = (usersrows(0).Item("Admin"))

Me.User\_Name.Text = ""

Me.Password.Text = ""

User\_Options.Show()

Me.Hide()

Else

MsgBox("Wrong UserName Or Password", MsgBoxStyle.OkOnly, "Error")

Exit Sub

End If

End Sub

Private Sub Form2\_Load(sender As Object, e As EventArgs) Handles Me.Activated

End Sub

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

Me.WindowState = FormWindowState.Minimized

End Sub

Private Sub Button1\_Click\_1(sender As Object, e As EventArgs) Handles Button1.Click

Me.Close()

End Sub

Private Sub Login\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

'UsersTableAdapter1.Insert("Admin", seperatedenc("123456"), Date.Now, Date.Now, 1)

End Sub

Public NotInheritable Class SimpleDes

Private myDes As New DESCryptoServiceProvider

Sub New(ByVal key As String)

' Initialize the crypto provider.

myDes.Key = System.Text.Encoding.ASCII.GetBytes(key)

myDes.IV = System.Text.Encoding.ASCII.GetBytes("12345678")

myDes.Mode = CipherMode.ECB

myDes.BlockSize = 64

myDes.Padding = PaddingMode.None

'myDes.Key = TruncateHash(key, myDes.KeySize \ 8)

'myDes.IV = TruncateHash("", myDes.BlockSize \ 8)

End Sub

Public Function newpass(ByVal passin As String)

myDes.Key = System.Text.Encoding.ASCII.GetBytes(passin)

End Function

Public Function DecryptData(

ByVal encryptedtext As String) As String

' Convert the encrypted text string to a byte array.

Dim encryptedBytes() As Byte = Convert.FromBase64String(encryptedtext)

' Create the stream.

Dim ms As New System.IO.MemoryStream

' Create the decoder to write to the stream.

Dim decStream As New CryptoStream(ms,

myDes.CreateDecryptor(),

System.Security.Cryptography.CryptoStreamMode.Write)

' Use the crypto stream to write the byte array to the stream.

decStream.Write(encryptedBytes, 0, encryptedBytes.Length)

decStream.FlushFinalBlock()

' Convert the plaintext stream to a string.

Return System.Text.Encoding.ASCII.GetString(ms.ToArray)

End Function

Public Function EncryptData(

ByVal plaintext As String, ByVal plaintext2() As Byte) As String

' Convert the plaintext string to a byte array.

Dim plaintextBytes() As Byte =

System.Text.Encoding.ASCII.GetBytes(plaintext)

' Create the stream.

Dim ms As New System.IO.MemoryStream

' Create the encoder to write to the stream.

Dim encStream As New CryptoStream(ms,

myDes.CreateEncryptor(),

System.Security.Cryptography.CryptoStreamMode.Write)

' Use the crypto stream to write the byte array to the stream.

encStream.Write(plaintextBytes, 0, plaintextBytes.Length)

encStream.FlushFinalBlock()

' Convert the encrypted stream to a printable string.

Dim bity(7) As Byte

For i As Integer = 0 To 7

bity(i) = ms.ToArray(i) Xor plaintext2(i)

Next

Return Convert.ToBase64String(bity)

End Function

Public Function EncryptData2(

ByVal plaintext As String, ByVal plaintext2() As Byte) As String

' Convert the plaintext string to a byte array.

Dim plaintextBytes() As Byte =

System.Text.Encoding.ASCII.GetBytes(plaintext)

' Create the stream.

Dim ms As New System.IO.MemoryStream

' Create the encoder to write to the stream.

Dim encStream As New CryptoStream(ms,

myDes.CreateEncryptor(),

System.Security.Cryptography.CryptoStreamMode.Write)

' Use the crypto stream to write the byte array to the stream.

encStream.Write(plaintextBytes, 0, plaintextBytes.Length)

encStream.FlushFinalBlock()

' Convert the encrypted stream to a printable string.

Dim bity(7) As Byte

For i As Integer = 0 To 7

bity(i) = ms.ToArray(i) Xor plaintext2(i)

Next

Return System.Text.ASCIIEncoding.ASCII.GetString(bity)

End Function

End Class

Private Sub Password\_TextChanged(sender As Object, e As EventArgs) Handles Password.TextChanged

End Sub

End Class

**APPENDIX E**

**new form Module Source Code:**

Public Class new\_form

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

Dim now As Date = Date.Now

NotesTableAdapter1.Insert(Login.userid, TextBox1.Text, TextBox2.Text, now)

If (Login.admin) Then

NotesTableAdapter1.Fill(ProjectDataSet1.Notes)

Else

NotesTableAdapter1.FillByid2(ProjectDataSet1.Notes, Login.userid)

End If

Dim row1() As DataRow = ProjectDataSet1.Notes.Select("ID='" + Login.userid.ToString + "' and Note\_Title='" + TextBox1.Text + "'")

Notes\_AccessTableAdapter1.Insert(row1(0).Item("Note\_ID"), Login.userid, True)

Me.Close()

End Sub

Private Sub new\_form\_Closed(sender As Object, e As EventArgs) Handles Me.Closed

Notes.Show()

End Sub

Private Sub new\_form\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

End Sub

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

Dim passy As String = InputBox("Enter Password", "Password", "")

If passy = "" Then

MsgBox("no password entered")

Exit Sub

End If

Dim now As Date = Date.Now

NotesTableAdapter1.Insert(Login.userid, TextBox1.Text, Login.seperatedenc\_withpassword(TextBox2.Text, passy), now)

If (Login.admin) Then

NotesTableAdapter1.Fill(ProjectDataSet1.Notes)

Else

NotesTableAdapter1.FillByid2(ProjectDataSet1.Notes, Login.userid)

End If

Dim row1() As DataRow = ProjectDataSet1.Notes.Select("ID='" + Login.userid.ToString + "' and Note\_Title='" + TextBox1.Text + "'")

Notes\_AccessTableAdapter1.Insert(row1(0).Item("Note\_ID"), Login.userid, True)

Me.Close()

End Sub

End Class

**APPENDIX F**

**Notes Module Source Code:**

Public Class Notes

Dim edit As New PROJECTDataSet.Notes\_AccessDataTable

Private Sub Notes\_Closed(sender As Object, e As EventArgs) Handles Me.Closed

If (new\_form.Visible = False) Then

User\_Options.WindowState = FormWindowState.Normal

User\_Options.Activate()

End If

End Sub

Private Sub Notes\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

If (Login.admin) Then

NotesTableAdapter1.Fill(ProjectDataSet1.Notes)

Button4.Visible = True

Button4.Enabled = True

Else

NotesTableAdapter1.Fillbyid(ProjectDataSet1.Notes, Login.userid)

End If

If (ListBox1.Items.Count = 0) Then

If (ListBox1.Items.Count = 0) Then

Button3.Enabled = False

End If

Else

Button3.Enabled = True

End If

Notes\_AccessTableAdapter1.FillBy(edit, ListBox1.SelectedValue, Login.userid)

If (edit.Count = 0) Then

Exit Sub

End If

If (edit(0).Item("edit")) Then

Button1.Enabled = True

Button3.Enabled = True

Else

Button1.Enabled = False

Button3.Enabled = False

End If

End Sub

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

If (Button1.Text = "Edit") Then

Button1.Text = "Update"

TextBox1.ReadOnly = False

TextBox2.ReadOnly = False

ElseIf (Button1.Text = "Update") Then

NotesTableAdapter1.Updatebynoteid(TextBox2.Text, TextBox1.Text, ListBox1.SelectedValue)

Button1.Text = "Edit"

TextBox1.ReadOnly = True

TextBox2.ReadOnly = True

If (Login.admin) Then

NotesTableAdapter1.Fill(ProjectDataSet1.Notes)

Else

NotesTableAdapter1.Fillbyid(ProjectDataSet1.Notes, Login.userid)

End If

End If

End Sub

Private Sub Button3\_Click(sender As Object, e As EventArgs) Handles Button3.Click

Dim result As Integer = MessageBox.Show("Are You sure you want to delete '" + TextBox2.SelectedText + "' Note?", "confirm", MessageBoxButtons.YesNo)

If result = DialogResult.No Then

Exit Sub

End If

Notes\_AccessTableAdapter1.DeleteQuery(ListBox1.SelectedValue)

NotesTableAdapter1.DeleteQuery(ListBox1.SelectedValue)

If (Login.admin) Then

NotesTableAdapter1.Fill(ProjectDataSet1.Notes)

Else

NotesTableAdapter1.Fillbyid(ProjectDataSet1.Notes, Login.userid)

End If

If (ListBox1.Items.Count = 0) Then

Button3.Enabled = False

End If

End Sub

Private Sub ListBox1\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles ListBox1.SelectedIndexChanged

If (Button1.Text = "Update") Then

Button1.Text = "Edit"

TextBox1.ReadOnly = True

TextBox2.ReadOnly = True

End If

Notes\_AccessTableAdapter1.FillBy(edit, ListBox1.SelectedValue, Login.userid)

If (edit.Count = 0) Then

Exit Sub

End If

If (edit(0).Item("edit")) Then

Button1.Enabled = True

Button3.Enabled = True

Else

Button1.Enabled = False

Button3.Enabled = False

End If

End Sub

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

new\_form.Show()

Me.Close()

End Sub

Private Sub Button4\_Click(sender As Object, e As EventArgs) Handles Button4.Click

Notes\_Access.Show()

End Sub

Private Sub Button5\_Click(sender As Object, e As EventArgs) Handles Button5.Click

Dim pass As String = InputBox("password", "enter your encryption password", "")

If pass = "" Then

Exit Sub

End If

TextBox1.Text = Login.seperateddecwithpass(TextBox1.Text, pass)

End Sub

End Class

**APPENDIX G**

**Notes\_Access Module Source Code:**

Public Class Notes\_Access

Private Sub Label1\_Click(sender As Object, e As EventArgs) Handles Label1.Click

End Sub

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

Dim usersh As New PROJECTDataSet.UsersDataTable

If (ComboBox2.SelectedIndex <> -1 And ComboBox1.SelectedIndex <> -1) Then

Notes\_AccessTableAdapter1.Fillbyid(ProjectDataSet1.Notes\_Access, ComboBox2.SelectedValue)

If (ProjectDataSet1.Notes\_Access.Count <> 0) Then

Notes\_AccessTableAdapter1.UpdateQuery(CheckBox1.Checked, ComboBox1.SelectedValue, ComboBox2.SelectedValue)

Else

Notes\_AccessTableAdapter1.Insert(ComboBox1.SelectedValue, ComboBox2.SelectedValue, CheckBox1.Checked)

End If

Else

MsgBox("check Your Inputs")

Exit Sub

End If

Me.Close()

End Sub

Private Sub Notes\_Access\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

NotesTableAdapter1.Fill(ProjectDataSet1.Notes)

ProjectDataSet1.Users.AdminColumn.AllowDBNull = True

ProjectDataSet1.Users.Date\_CreatedColumn.AllowDBNull = True

ProjectDataSet1.Users.Last\_UpdateColumn.AllowDBNull = True

ProjectDataSet1.Users.PasswordColumn.AllowDBNull = True

UsersTableAdapter1.Fillnameidexept(ProjectDataSet1.Users, Login.userid)

ToolTip1.SetToolTip(CheckBox1, "if checked user can modify note")

End Sub

End Class

**APPENDIX H**

**pass\_change Module Source Code:**

Public Class pass\_change

Public sqlcon As String = "Data Source=termproject2.database.windows.net;Initial Catalog=termproject;User ID=Talal;Password=15935736mM"

Dim here As New PROJECTDataSet.UsersDataTable

Public old\_password As String

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click

If (Old\_Pass.Text = "") Then

MsgBox("Please Enter your old password")

Exit Sub

ElseIf (New\_Pass.Text = "") Then

MsgBox("Please enter your new passwors")

Exit Sub

ElseIf (New\_Pass2.Text = "") Then

MsgBox("Please confirm your new passwors")

Exit Sub

Else

UsersTableAdapter1.Fillpassbyid(here, Login.userid)

If here.Count = 0 Then

MsgBox("Error")

Exit Sub

End If

If (here(0).Password.Replace(" ", "") = Login.seperatedenc(Old\_Pass.Text)) Then

If (New\_Pass.Text = New\_Pass2.Text) Then

UsersTableAdapter1.Updatepassbyid(Login.seperatedenc(New\_Pass.Text), Date.Now, Login.userid)

Else

MsgBox("your new password and confirm are not the same")

New\_Pass2.Text = ""

Exit Sub

End If

Else

MsgBox("Wrong old password")

Exit Sub

End If

Me.Close()

End If

End Sub

Private Sub New\_Pass2\_TextChanged(sender As Object, e As EventArgs) Handles New\_Pass2.TextChanged

If (New\_Pass2.Text <> New\_Pass.Text And New\_Pass2.Text <> "") Then

New\_Pass2.BackColor = Color.Red

Else

New\_Pass2.BackColor = Color.White

End If

End Sub

Private Sub pass\_change\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

UsersTableAdapter1.Fillpassbyid(here, Login.userid)

End Sub

End Class

**APPENDIX I**

**User\_Options Module Source Code:**

Public Class User\_Options

Dim state(4) As Integer

Private Sub Form1\_Load(sender As Object, e As EventArgs) Handles MyBase.Closed

If (Login.Visible = False) Then

Login.Close()

End If

End Sub

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles User\_List\_B.Click

Me.WindowState = FormWindowState.Minimized

Users\_List.Show()

Users\_List.Activate()

End Sub

Private Sub Form1\_Load\_1(sender As Object, e As EventArgs) Handles MyBase.Load

ToolTip2.SetToolTip(User\_List\_B, "creating Editing Users" + Environment.NewLine + "adding,removing previlige")

ToolTip3.SetToolTip(Button4, "change your account password")

ToolTip4.SetToolTip(Notes\_Button, "creating Editing Notes")

ToolTip1.SetToolTip(Button1, "sign out")

ToolTip5.SetToolTip(Images\_Button, "View add" + Environment.NewLine + " delete images")

AccessTableAdapter1.Fill(ProjectDataSet1.ACCESS)

If (Login.admin) Then

User\_List\_B.Enabled = True

User\_List\_B.Cursor = Cursors.Hand

Notes\_Button.Enabled = True

Notes\_Button.Cursor = Cursors.Hand

Images\_Button.Enabled = True

Images\_Button.Cursor = Cursors.Hand

Else

Dim accessvar() As DataRow = ProjectDataSet1.ACCESS.Select("ID='" + Login.userid.ToString + "'")

If (accessvar.Count) Then

If (accessvar(0).Item("Users\_e")) Then

User\_List\_B.Enabled = True

End If

If (accessvar(0).Item("Images")) Then

Notes\_Button.Enabled = True

End If

If (accessvar(0).Item("Notes")) Then

Images\_Button.Enabled = True

End If

End If

End If

Label1.Text = Login.userName

End Sub

Private Sub Button4\_Click(sender As Object, e As EventArgs) Handles Button4.Click

pass\_change.Show()

End Sub

Private Sub Notes\_Button\_Click(sender As Object, e As EventArgs) Handles Notes\_Button.Click

Me.WindowState = FormWindowState.Minimized

Notes.Show()

Notes.Activate()

End Sub

Private Sub LinkLabel1\_LinkClicked(sender As Object, e As LinkLabelLinkClickedEventArgs)

Login.Show()

Me.Close()

End Sub

Private Sub User\_List\_B\_MouseHover(sender As Object, e As EventArgs) Handles User\_List\_B.MouseHover

If User\_List\_B.Enabled Then

User\_List\_B.FlatAppearance.BorderColor = Color.Red

End If

End Sub

Private Sub User\_List\_B\_MouseLeave(sender As Object, e As EventArgs) Handles User\_List\_B.MouseLeave

User\_List\_B.FlatAppearance.BorderColor = Color.Purple

End Sub

Private Sub Notes\_Button\_MouseHover(sender As Object, e As EventArgs) Handles Notes\_Button.MouseHover

If Notes\_Button.Enabled Then

Notes\_Button.FlatAppearance.BorderColor = Color.Red

End If

End Sub

Private Sub Notes\_Button\_MouseLeave(sender As Object, e As EventArgs) Handles Notes\_Button.MouseLeave

Notes\_Button.FlatAppearance.BorderColor = Color.Green

End Sub

Private Sub Images\_Button\_Click(sender As Object, e As EventArgs) Handles Images\_Button.Click

images\_list.Show()

Me.WindowState = FormWindowState.Minimized

images\_list.Activate()

End Sub

Private Sub Images\_Button\_MouseHover(sender As Object, e As EventArgs) Handles Images\_Button.MouseHover

Images\_Button.FlatAppearance.BorderColor = Color.Red

End Sub

Private Sub Images\_Button\_MouseLeave(sender As Object, e As EventArgs) Handles Images\_Button.MouseLeave

Images\_Button.FlatAppearance.BorderColor = Color.MediumTurquoise

End Sub

Private Sub Button1\_Click\_1(sender As Object, e As EventArgs) Handles Button1.Click

Login.Show()

Me.Close()

End Sub

Private Sub Button4\_MouseHover(sender As Object, e As EventArgs) Handles Button4.MouseHover

Button4.FlatAppearance.BorderColor = Color.Red

End Sub

Private Sub Button4\_MouseLeave(sender As Object, e As EventArgs) Handles Button4.MouseLeave

Button4.FlatAppearance.BorderColor = Color.Navy

End Sub

End Class

**APPENDIX J**

**Users\_List Module Source Code:**

Public Class Users\_List

Dim password As String

Dim password2 As String

Private Sub Form3\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

'TODO: This line of code loads data into the 'TermprojectDataSet.Users' table. You can move, or remove it, as needed.

Me.AccessTableAdapter1.Fill(Me.ProjectDataSet1.ACCESS)

If (Login.admin) Then

delete.Enabled = True

Me.UsersTableAdapter1.Fillexcept(Me.ProjectDataSet1.Users, Login.userid)

Else

delete.Enabled = False

Me.UsersTableAdapter1.FillBynotadmin(Me.ProjectDataSet1.Users, Login.userid)

End If

If (UsersList.Items.Count = 0) Then

delete.Enabled = False

TextBox1.Enabled = False

TextBox2.Enabled = False

Else

If (Login.admin) Then

delete.Enabled = True

End If

TextBox1.Enabled = True

TextBox2.Enabled = True

End If

AccessTableAdapter1.Fill(ProjectDataSet1.ACCESS)

If (UsersList.Items.Count = 0) Then

Exit Sub

End If

If (CheckBox1.Checked = False) Then

Dim anyname() As DataRow = ProjectDataSet1.ACCESS.Select("ID=" + UsersList.SelectedValue.ToString)

If (anyname.Count <> 0) Then

CheckBox2.Checked = anyname(0).Item("Users\_e")

CheckBox3.Checked = anyname(0).Item("Images")

CheckBox4.Checked = anyname(0).Item("Notes")

End If

End If

TextBox1.Text = TextBox1.Text.Replace(" ", "")

password = TextBox2.Text

End Sub

Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles addnew.Click

Add\_New\_user.Show()

Me.Close()

End Sub

Private Sub Form3\_Closed(sender As Object, e As EventArgs) Handles Me.Closed

If (Add\_New\_user.Visible = False) Then

User\_Options.WindowState = FormWindowState.Normal

User\_Options.Activate()

End If

End Sub

Private Sub Button2\_Click(sender As Object, e As EventArgs) Handles Button2.Click

If (UsersList.Items.Count = 0) Then

TextBox1.Enabled = False

TextBox2.Enabled = False

Exit Sub

End If

'UsersBindingSource.EndEdit()

If (CheckBox1.Checked = False) Then

If (ProjectDataSet1.ACCESS.Select("ID='" + UsersList.SelectedValue.ToString + "'").Count = 0) Then

AccessTableAdapter1.Insert(UsersList.SelectedValue, CheckBox2.Checked, CheckBox3.Checked, CheckBox4.Checked)

Else

AccessTableAdapter1.Updatebyid(CheckBox2.Checked, CheckBox3.Checked, CheckBox4.Checked, UsersList.SelectedValue)

End If

AccessTableAdapter1.Fill(ProjectDataSet1.ACCESS)

Else

AccessTableAdapter1.DeleteQuery(UsersList.SelectedValue)

AccessTableAdapter1.Fill(ProjectDataSet1.ACCESS)

End If

If (TextBox2.Text = password2) Then

UsersTableAdapter1.UpdateQuery(TextBox1.Text, TextBox2.Text, Date.Now, CheckBox1.Checked, UsersList.SelectedValue)

Else

UsersTableAdapter1.UpdateQuery(TextBox1.Text, Login.seperatedenc(TextBox2.Text), Date.Now, CheckBox1.Checked, UsersList.SelectedValue)

End If

If (Login.admin) Then

UsersTableAdapter1.Fillexcept(ProjectDataSet1.Users, Login.userid)

Else

UsersTableAdapter1.FillBynotadmin(ProjectDataSet1.Users, Login.userid)

End If

Button2.Visible = False

Button1.Visible = True

CheckBox1.Enabled = False

CheckBox2.Enabled = False

CheckBox3.Enabled = False

CheckBox4.Enabled = False

TextBox1.ReadOnly = True

TextBox2.ReadOnly = True

End Sub

Private Sub Button3\_Click(sender As Object, e As EventArgs) Handles delete.Click

AccessTableAdapter1.DeleteQuery(UsersList.SelectedValue)

UsersTableAdapter1.deleteuser(UsersList.SelectedValue)

If (Login.admin) Then

UsersTableAdapter1.Fillexcept(ProjectDataSet1.Users, Login.userid)

Else

UsersTableAdapter1.FillBynotadmin(ProjectDataSet1.Users, Login.userid)

End If

If (UsersList.Items.Count = 0) Then

TextBox1.Enabled = False

TextBox2.Enabled = False

delete.Enabled = False

End If

TextBox1.Text = TextBox1.Text.Replace(" ", "")

End Sub

Private Sub CheckBox1\_CheckedChanged\_1(sender As Object, e As EventArgs) Handles CheckBox1.CheckedChanged

If (Login.admin) Then

If (CheckBox1.Checked = False And CheckBox1.Enabled) Then

CheckBox2.Enabled = True

CheckBox3.Enabled = True

CheckBox4.Enabled = True

Else

CheckBox2.Enabled = False

CheckBox3.Enabled = False

CheckBox4.Enabled = False

End If

Else

If (CheckBox1.Checked = False And CheckBox1.Enabled) Then

CheckBox2.Enabled = False

CheckBox3.Enabled = True

CheckBox4.Enabled = True

Else

CheckBox2.Enabled = False

CheckBox3.Enabled = False

CheckBox4.Enabled = False

End If

End If

End Sub

Private Sub UsersList\_SelectedIndexChanged(sender As Object, e As EventArgs) Handles UsersList.SelectedValueChanged

End Sub

Private Sub TextBox2\_TextChanged(sender As Object, e As EventArgs) Handles TextBox2.Click

If Not TextBox2.ReadOnly Then

TextBox2.Text = ""

End If

End Sub

Private Sub TextBox2\_Leave(sender As Object, e As EventArgs) Handles TextBox2.Leave

If (TextBox2.Text = "") Then

TextBox2.Text = password

Else

password = TextBox2.Text

End If

End Sub

Private Sub UsersList\_SelectedIndexChanged\_1(sender As Object, e As EventArgs) Handles UsersList.SelectedIndexChanged

On Error GoTo endchange

password = TextBox2.Text

Button2.Visible = False

Button1.Visible = True

If (Login.admin) Then

If (CheckBox1.Checked = False) Then

Dim anyname() As DataRow = ProjectDataSet1.ACCESS.Select("ID=" + UsersList.SelectedValue.ToString)

If (anyname.Count = 0) Then

Exit Sub

End If

If (anyname.Count <> 0) Then

CheckBox2.Checked = anyname(0).Item("Users\_e")

CheckBox3.Checked = anyname(0).Item("Images")

CheckBox4.Checked = anyname(0).Item("Notes")

End If

Else

CheckBox2.Checked = False

CheckBox3.Checked = False

CheckBox4.Checked = False

End If

Else

If (CheckBox1.Checked = False) Then

Dim anyname() As DataRow = ProjectDataSet1.ACCESS.Select("ID=" + UsersList.SelectedValue.ToString)

If (anyname.Count = 0) Then

Exit Sub

End If

If (anyname.Count <> 0) Then

CheckBox2.Checked = anyname(0).Item("Users\_e")

CheckBox3.Checked = anyname(0).Item("Images")

CheckBox4.Checked = anyname(0).Item("Notes")

End If

Else

CheckBox2.Checked = False

CheckBox3.Checked = False

CheckBox4.Checked = False

End If

End If

TextBox1.Text = TextBox1.Text.Replace(" ", "")

endchange:

End Sub

Private Sub Button1\_Click\_1(sender As Object, e As EventArgs) Handles Button1.Click

Button2.Visible = True

Button1.Visible = False

TextBox1.ReadOnly = False

TextBox2.ReadOnly = False

CheckBox3.Enabled = True

CheckBox4.Enabled = True

If (Login.admin) Then

CheckBox1.Enabled = True

CheckBox2.Enabled = True

Else

CheckBox1.Enabled = False

CheckBox2.Enabled = False

End If

password2 = TextBox2.Text

End Sub

End Class

**APPENDIX K**

**DES CTR Source Code:**

# include <stdio.h>

# include <fstream>

# include <string.h>

# include <limits>

# include <string>

# include <sstream>

# include <iostream>

# include <stdlib.h>

# include <bitset>

using namespace std;

int key[64]=

{

0,0,0,1,0,0,1,1,

0,0,1,1,0,1,0,0,

0,1,0,1,0,1,1,1,

0,1,1,1,1,0,0,1,

1,0,0,1,1,0,1,1,

1,0,1,1,1,1,0,0,

1,1,0,1,1,1,1,1,

1,1,1,1,0,0,0,1

},CTRstep=1;

void TextKey();

void Hexa\_Key();

class Des

{

public:

int keyi[16][48],

total[64],

total2[64],

left[32],

right[32],

ck[28],

dk[28],

expansion[48],

z[48],

xor1[48],

sub[32],

p[32],

xor2[32],

xorCTR[64],

temp[64],

pc1[56],

ip[64],

inv[8][8],

seperated[100][64],

CTRround[100][64];

char final[1000];

void IP();

void PermChoice1();

void PermChoice2();

void Expansion();

void inverse();

void xor\_two();

void xor\_CTR();

void xor\_oneE(int);

void xor\_oneD(int);

void substitution();

void permutation();

void keygen();

char \* Encrypt(int);

char \* EncryptCTR(int);

char \* Decrypt(int);

void seperating(string,int);

void CTR(int d2);

};

void Des::IP() //Initial Permutation

{

int k=58,i;

for(i=0; i<32; i++)

{

ip[i]=total[k-1];

if(k-8>0) k=k-8;

else k=k+58;

}

k=57;

for( i=32; i<64; i++)

{

ip[i]=total[k-1];

if(k-8>0) k=k-8;

else k=k+58;

}

}

void Des::PermChoice1() //Permutation Choice-1

{

int k=57,i;

for(i=0; i<28; i++)

{

pc1[i]=key[k-1];

if(k-8>0) k=k-8;

else k=k+57;

}

k=63;

for( i=28; i<52; i++)

{

pc1[i]=key[k-1];

if(k-8>0) k=k-8;

else k=k+55;

}

k=28;

for(i=52; i<56; i++)

{

pc1[i]=key[k-1];

k=k-8;

}

}

void Des::Expansion() //Expansion Function applied on `right' half

{

int exp[8][6],i,j,k;

for( i=0; i<8; i++)

{

for( j=0; j<6; j++)

{

if((j!=0)||(j!=5))

{

k=4\*i+j;

exp[i][j]=right[k-1];

}

if(j==0)

{

k=4\*i;

exp[i][j]=right[k-1];

}

if(j==5)

{

k=4\*i+j;

exp[i][j]=right[k-1];

}

}

}

exp[0][0]=right[31];

exp[7][5]=right[0];

k=0;

for(i=0; i<8; i++)

for(j=0; j<6; j++)

expansion[k++]=exp[i][j];

}

void Des::PermChoice2()

{

int per[56],i,k;

for(i=0; i<28; i++) per[i]=ck[i];

for(k=0,i=28; i<56; i++) per[i]=dk[k++];

z[0]=per[13];

z[1]=per[16];

z[2]=per[10];

z[3]=per[23];

z[4]=per[0];

z[5]=per[4];

z[6]=per[2];

z[7]=per[27];

z[8]=per[14];

z[9]=per[5];

z[10]=per[20];

z[11]=per[9];

z[12]=per[22];

z[13]=per[18];

z[14]=per[11];

z[15]=per[3];

z[16]=per[25];

z[17]=per[7];

z[18]=per[15];

z[19]=per[6];

z[20]=per[26];

z[21]=per[19];

z[22]=per[12];

z[23]=per[1];

z[24]=per[40];

z[25]=per[51];

z[26]=per[30];

z[27]=per[36];

z[28]=per[46];

z[29]=per[54];

z[30]=per[29];

z[31]=per[39];

z[32]=per[50];

z[33]=per[46];

z[34]=per[32];

z[35]=per[47];

z[36]=per[43];

z[37]=per[48];

z[38]=per[38];

z[39]=per[55];

z[40]=per[33];

z[41]=per[52];

z[42]=per[45];

z[43]=per[41];

z[44]=per[49];

z[45]=per[35];

z[46]=per[28];

z[47]=per[31];

}

void Des::xor\_oneE(int round) //for Encrypt

{

int i;

for(i=0; i<48; i++)

xor1[i]=expansion[i]^keyi[round-1][i];

}

void Des::xor\_oneD(int round) //for Decrypt

{

int i;

for(i=0; i<48; i++)

xor1[i]=expansion[i]^keyi[16-round][i];

}

void Des::substitution()

{

int s1[4][16]=

{

14,4,13,1,2,15,11,8,3,10,6,12,5,9,0,7,

0,15,7,4,14,2,13,1,10,6,12,11,9,5,3,8,

4,1,14,8,13,6,2,11,15,12,9,7,3,10,5,0,

15,12,8,2,4,9,1,7,5,11,3,14,10,0,6,13

};

int s2[4][16]=

{

15,1,8,14,6,11,3,4,9,7,2,13,12,0,5,10,

3,13,4,7,15,2,8,14,12,0,1,10,6,9,11,5,

0,14,7,11,10,4,13,1,5,8,12,6,9,3,2,15,

13,8,10,1,3,15,4,2,11,6,7,12,0,5,14,9

};

int s3[4][16]=

{

10,0,9,14,6,3,15,5,1,13,12,7,11,4,2,8,

13,7,0,9,3,4,6,10,2,8,5,14,12,11,15,1,

13,6,4,9,8,15,3,0,11,1,2,12,5,10,14,7,

1,10,13,0,6,9,8,7,4,15,14,3,11,5,2,12

};

int s4[4][16]=

{

7,13,14,3,0,6,9,10,1,2,8,5,11,12,4,15,

13,8,11,5,6,15,0,3,4,7,2,12,1,10,14,9,

10,6,9,0,12,11,7,13,15,1,3,14,5,2,8,4,

3,15,0,6,10,1,13,8,9,4,5,11,12,7,2,14

};

int s5[4][16]=

{

2,12,4,1,7,10,11,6,8,5,3,15,13,0,14,9,

14,11,2,12,4,7,13,1,5,0,15,10,3,9,8,6,

4,2,1,11,10,13,7,8,15,9,12,5,6,3,0,14,

11,8,12,7,1,14,2,13,6,15,0,9,10,4,5,3

};

int s6[4][16]=

{

12,1,10,15,9,2,6,8,0,13,3,4,14,7,5,11,

10,15,4,2,7,12,9,5,6,1,13,14,0,11,3,8,

9,14,15,5,2,8,12,3,7,0,4,10,1,13,11,6,

4,3,2,12,9,5,15,10,11,14,1,7,6,0,8,13

};

int s7[4][16]=

{

4,11,2,14,15,0,8,13,3,12,9,7,5,10,6,1,

13,0,11,7,4,9,1,10,14,3,5,12,2,15,8,6,

1,4,11,13,12,3,7,14,10,15,6,8,0,5,9,2,

6,11,13,8,1,4,10,7,9,5,0,15,14,2,3,12

};

int s8[4][16]=

{

13,2,8,4,6,15,11,1,10,9,3,14,5,0,12,7,

1,15,13,8,10,3,7,4,12,5,6,11,0,14,9,2,

7,11,4,1,9,12,14,2,0,6,10,13,15,3,5,8,

2,1,14,7,4,10,8,13,15,12,9,0,3,5,6,11

};

int a[8][6],k=0,i,j,p,q,count=0,g=0,v;

for(i=0; i<8; i++)

{

for(j=0; j<6; j++)

{

a[i][j]=xor1[k++];

}

}

for( i=0; i<8; i++)

{

p=1;

q=0;

k=(a[i][0]\*2)+(a[i][5]\*1);

j=4;

while(j>0)

{

q=q+(a[i][j]\*p);

p=p\*2;

j--;

}

count=i+1;

switch(count)

{

case 1:

v=s1[k][q];

break;

case 2:

v=s2[k][q];

break;

case 3:

v=s3[k][q];

break;

case 4:

v=s4[k][q];

break;

case 5:

v=s5[k][q];

break;

case 6:

v=s6[k][q];

break;

case 7:

v=s7[k][q];

break;

case 8:

v=s8[k][q];

break;

}

int d,i=3,a[4];

while(v>0)

{

d=v%2;

a[i--]=d;

v=v/2;

}

while(i>=0)

{

a[i--]=0;

}

for(i=0; i<4; i++)

sub[g++]=a[i];

}

}

void Des::permutation()

{

p[0]=sub[15];

p[1]=sub[6];

p[2]=sub[19];

p[3]=sub[20];

p[4]=sub[28];

p[5]=sub[11];

p[6]=sub[27];

p[7]=sub[16];

p[8]=sub[0];

p[9]=sub[14];

p[10]=sub[22];

p[11]=sub[25];

p[12]=sub[4];

p[13]=sub[17];

p[14]=sub[30];

p[15]=sub[9];

p[16]=sub[1];

p[17]=sub[7];

p[18]=sub[23];

p[19]=sub[13];

p[20]=sub[31];

p[21]=sub[26];

p[22]=sub[2];

p[23]=sub[8];

p[24]=sub[18];

p[25]=sub[12];

p[26]=sub[29];

p[27]=sub[5];

p[28]=sub[21];

p[29]=sub[10];

p[30]=sub[3];

p[31]=sub[24];

}

void Des::xor\_two()

{

int i;

for(i=0; i<32; i++)

{

xor2[i]=left[i]^p[i];

}

}

void Des::inverse()

{

int p=40,q=8,k1,k2,i,j;

for(i=0; i<8; i++)

{

k1=p;

k2=q;

for(j=0; j<8; j++)

{

if(j%2==0)

{

inv[i][j]=temp[k1-1];

k1=k1+8;

}

else if(j%2!=0)

{

inv[i][j]=temp[k2-1];

k2=k2+8;

}

}

p=p-1;

q=q-1;

}

}

char \* Des::Encrypt(int blkn)

{

int i,a1,j,nB,m,iB,k,K,B[8],n,t,d,round,mc=0;

for(i=0;i<64;i++){

total[i]=seperated[blkn][i];

}

keygen();

IP(); //Performing initial permutation on `total[64]'

for(i=0; i<64; i++) total[i]=ip[i]; //Store values of ip[64] into total[64]

for(i=0; i<32; i++) left[i]=total[i]; // +--> left[32]

// total[64]--|

for(; i<64; i++) right[i-32]=total[i]; // +--> right[32]

for(round=1; round<=16; round++)

{

Expansion(); //Performing expansion on `right[32]' to get `expansion[48]'

xor\_oneE(round); //Performing XOR operation on expansion[48],z[48] to get xor1[48]

substitution();//Perform substitution on xor1[48] to get sub[32]

permutation(); //Performing Permutation on sub[32] to get p[32]

xor\_two(); //Performing XOR operation on left[32],p[32] to get xor2[32]

for(i=0; i<32; i++) left[i]=right[i]; //Dumping right[32] into left[32]

for(i=0; i<32; i++) right[i]=xor2[i]; //Dumping xor2[32] into right[32]

}

for(i=0; i<32; i++) temp[i]=right[i]; // Dumping -->[ swap32bit ]

for(; i<64; i++) temp[i]=left[i-32]; // left[32],right[32] into temp[64]

inverse(); //Inversing the bits of temp[64] to get inv[8][8]

/\* Obtaining the Cypher-Text into final[1000]\*/

k=128;

d=0;

for(i=0; i<8; i++)

{

for(j=0; j<8; j++)

{

d=d+inv[i][j]\*k;

k=k/2;

}

final[mc++]=(char)d;

k=128;

d=0;

}

final[mc]='\0';

return(final);

}

char \* Des::Decrypt(int blkn)

{

int i,a1,j,nB,m,iB,k,K,B[8],n,t,d,round;

for(i=0;i<64;i++){

total[i]=seperated[blkn][i];

}

keygen();

int mc=0;

IP(); //Performing initial permutation on `total[64]'

for(i=0; i<64; i++) total[i]=ip[i]; //Store values of ip[64] into total[64]

for(i=0; i<32; i++) left[i]=total[i]; // +--> left[32]

// total[64]--|

for(; i<64; i++) right[i-32]=total[i]; // +--> right[32]

for(round=1; round<=16; round++)

{

Expansion(); //Performing expansion on `right[32]' to get `expansion[48]'

xor\_oneD(round);

substitution();//Perform substitution on xor1[48] to get sub[32]

permutation(); //Performing Permutation on sub[32] to get p[32]

xor\_two(); //Performing XOR operation on left[32],p[32] to get xor2[32]

for(i=0; i<32; i++) left[i]=right[i]; //Dumping right[32] into left[32]

for(i=0; i<32; i++) right[i]=xor2[i]; //Dumping xor2[32] into right[32]

} //rounds end here

for(i=0; i<32; i++) temp[i]=right[i]; // Dumping -->[ swap32bit ]

for(; i<64; i++) temp[i]=left[i-32]; // left[32],right[32] into temp[64]

inverse(); //Inversing the bits of temp[64] to get inv[8][8]

/\* Obtaining the Cypher-Text into final[1000]\*/

k=128;

d=0;

for(i=0; i<8; i++)

{

for(j=0; j<8; j++)

{

d=d+inv[i][j]\*k;

k=k/2;

}

final[mc++]=(char)d;

k=128;

d=0;

}

final[mc]='\0';

//char \*final1=new char[1000];

//for(i=0,j=16; i<16; i++,j++)

//final1[i]=final[j];

//final1[i]='\0';

return(final);

}

int main()

{

Des En,De;

string str;

stringstream ciph1,ciph2,ciph3,ciph4;

char \*str1=new char[1000],\*str2=new char[1000];

int choice;

//strcpy(str,"PHOENIX it & ece solutions.");

start:

cout<<" Group 1 Des C++ Implementation\n";

cout<<"\nSelect Your Key Type:\n1-Text 8 char Key\n2-Hex 16 char key (0x0000000000000000)\n3-binary(64 bit)\_O\_\n";

cin>>choice;

switch(choice){

case 1:

TextKey();

break;

case 2:

Hexa\_Key();

break;

case 3:

break;

default:

cout<<"\nWrong Input\n\nPlease Try again\n";

system("pause");

system("cls");

goto start;

}

cout<<"this program uses counter as follows:\n\t\t0xffffffff00000000 -> 0xfffffffE00000001\nEnter Counter step size (1 by default):";

cin>>CTRstep;

system("cls");

cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

cout<<"Enter Your Text : \n";

getline(cin,str);

En.seperating(str,str.length());

for(int kl=0;kl<((str.length()%8==0)?(str.length()/8):(str.length()/8+1));kl++)

ciph1<<En.Encrypt(kl);

cout<<"\ni/p Text: "<<str<<endl;

cout<<"\nNormal 8 bytes block Cypher : \n\""<<ciph1.str()<<"\""<<endl;

string str3=ciph1.str();

En.seperating(str3,str3.length());

// ofstream fout("out2\_fil.txt"); fout<<str1; fout.close();

for(int kl=0;kl<((str3.length()%8==0)?(str3.length()/8):(str3.length()/8+1));kl++)

ciph2<<En.Decrypt(kl);

cout<<"\nDecryption Text: "<<ciph2.str()<<endl;

if(str.length()%8==0){

De.CTR(str.length()/8);

}else

De.CTR(str.length()/8+1);

De.seperating(str,str.length());

for(int kl=0;kl<((str.length()%8==0)?(str.length()/8):(str.length()/8+1));kl++)

ciph3<<De.EncryptCTR(kl);

cout<<"\nCTR Cipher: "<<ciph3.str()<<endl;

De.CTR(ciph3.str().length()/8);

De.seperating(ciph3.str(),ciph3.str().length());

for(int kl=0;kl<((ciph3.str().length()%8==0)?(ciph3.str().length()/8):(ciph3.str().length()/8+1));kl++)

ciph4<<De.EncryptCTR(kl);

cout<<"\nCTR Decryption: "<<ciph4.str()<<endl;

system("pause");

}

void Des::keygen()

{

PermChoice1();

int i,j,k=0;

for(i=0; i<28; i++)

{

ck[i]=pc1[i];

}

for(i=28; i<56; i++)

{

dk[k]=pc1[i];

k++;

}

int noshift=0,round;

for(round=1; round<=16; round++)

{

if(round==1||round==2||round==9||round==16)

noshift=1;

else

noshift=2;

while(noshift>0)

{

int t;

t=ck[0];

for(i=0; i<28; i++)

ck[i]=ck[i+1];

ck[27]=t;

t=dk[0];

for(i=0; i<28; i++)

dk[i]=dk[i+1];

dk[27]=t;

noshift--;

}

PermChoice2();

for(i=0; i<48; i++)

keyi[round-1][i]=z[i];

}

}

void Des::seperating(string plaintext,int i){

int d,c,j,k=0,n,t=0;

stringstream temp;

string allbinary;

d=(i)/8;

c=(i)%8;

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

temp<<bitset<8> (plaintext[j]);

}

for(j=c;j<8;j++)

{

temp<<bitset<8> (' ');

}

allbinary=temp.str();

temp.str("");

t=0;

for(j=0;j<allbinary.length();j++){

if((j%64)==0 && j!=0)

t++;

seperated[t][j%64]=(int)allbinary[j]-48;

}

}

void Des::CTR(int d2){

stringstream keys;

unsigned long int counter=0x0000000000000000;

string temps;

for(int i=0;i<d2;i++)

{

counter+=CTRstep;

keys<<bitset<32> (0xffffffffffffffff-counter)<<bitset<32> (counter);

temps=keys.str();

keys.str(string());

for(int j=0;j<64;j++)

{

CTRround[i][j]=(int)temps[j]-48;

}

}

}

char \* Des::EncryptCTR(int blkn)

{

int i,a1,j,nB,m,iB,k,K,B[8],n,t,d,round,mc=0;

for(i=0;i<64;i++){

total[i]=CTRround[blkn][i];

total2[i]=seperated[blkn][i];

}

keygen();

IP(); //Performing initial permutation on `total[64]'

for(i=0; i<64; i++) total[i]=ip[i]; //Store values of ip[64] into total[64]

for(i=0; i<32; i++) left[i]=total[i]; // +--> left[32]

// total[64]--|

for(; i<64; i++) right[i-32]=total[i]; // +--> right[32]

for(round=1; round<=16; round++)

{

Expansion(); //Performing expansion on `right[32]' to get `expansion[48]'

xor\_oneE(round); //Performing XOR operation on expansion[48],z[48] to get xor1[48]

substitution();//Perform substitution on xor1[48] to get sub[32]

permutation(); //Performing Permutation on sub[32] to get p[32]

xor\_two(); //Performing XOR operation on left[32],p[32] to get xor2[32]

for(i=0; i<32; i++) left[i]=right[i]; //Dumping right[32] into left[32]

for(i=0; i<32; i++) right[i]=xor2[i]; //Dumping xor2[32] into right[32]

}

for(i=0; i<32; i++) temp[i]=right[i]; // Dumping -->[ swap32bit ]

for(; i<64; i++) temp[i]=left[i-32]; // left[32],right[32] into temp[64]

inverse(); //Inversing the bits of temp[64] to get inv[8][8]

/\* Obtaining the Cypher-Text into final[1000]\*/

for(i=0;i<8;i++)

for(j=0;j<8;j++)

total[i\*8+j]=inv[i][j];

xor\_CTR();

for(i=0;i<8;i++)

for(j=0;j<8;j++)

inv[i][j]=xorCTR[i\*8+j];

k=128;

d=0;

for(i=0; i<8; i++)

{

for(j=0; j<8; j++)

{

d=d+inv[i][j]\*k;

k=k/2;

}

final[mc++]=(char)d;

k=128;

d=0;

}

final[mc]='\0';

return(final);

}

void Des::xor\_CTR()

{

int i;

for(i=0; i<64; i++)

{

xorCTR[i]=total[i]^total2[i];

}

}

void TextKey(){

string textkey;

stringstream tkeyb;

system("cls");

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

cout<<"Please enter your 8 char key:\n";

getline(cin,textkey);

system("cls");

for(int i=0;i<8;i++)

{

tkeyb<<bitset<8>((int)textkey[i]);

}

string bkey=tkeyb.str();

for(int i=0;i<64;i++)

key[i]=(int)bkey[i]-48;

cout<<"your key is:\n\""<<textkey<<"\"\n";

system("pause");

system("cls");

}

void Hexa\_Key(){

//string hexaText;

stringstream hkeyb,hkeyb2;

//KeyMode=2;

int n,i,k,B[8],K,iB;

long unsigned int hkey=0x0000000000000000;

system("cls");

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

hexakeystart:

string hexaText,hexabinary;

cout<<"Enter your Key:";

getline( cin, hexaText);

for(i=0;i<16;i++){

switch(toupper(hexaText[i]))

{

case '0':

hkeyb2<<"0000";

break;

case '1':

hkeyb2<<"0001";

break;

case '2':

hkeyb2<<"0010";

break;

case '3':

hkeyb2<<"0011";

break;

case '4':

hkeyb2<<"0100";

break;

case '5':

hkeyb2<<"0101";

break;

case '6':

hkeyb2<<"0110";

break;

case '7':

hkeyb2<<"0111";

break;

case '8':

hkeyb2<<"1000";

break;

case '9':

hkeyb2<<"1001";

break;

case 'A':

hkeyb2<<"1010";

break;

case 'B':

hkeyb2<<"1011";

break;

case 'C':

hkeyb2<<"1100";

break;

case 'D':

hkeyb2<<"1101";

break;

case 'E':

hkeyb2<<"1110";

break;

case 'F':

hkeyb2<<"1111";

break;

default:

cout<<"\nWrong Input\nHexaDesimal shoud be only in '0-9' or 'A-F' characters\n";

system("pause");

hkeyb2.str("");

goto hexakeystart;

}

}

hexabinary=hkeyb2.str();

for(int i=0; i<64;i++)

key[i]=hexabinary[i];

system("cls");

cout<<"Your Key Is \"";

for(i=0;i<16;i++)cout<<(char)toupper(hexaText[i]);

cout<<"\"\n";

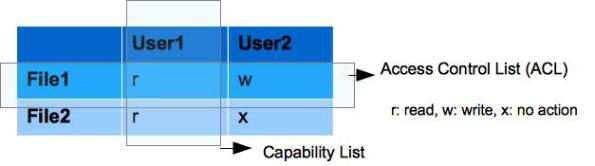
system("pause");

}

**APPENDIX L**

**Description of Access Control Lists and Capability Lists:**

In Access Control List (ACL) each object has a list of (action, user-list) tuple.

[](https://prosuncsedu.files.wordpress.com/2014/08/access-matric.jpg)

For example, in the above diagram File1 & File2 would have the following ACL:

File1: (         (read, {user1}),      (write, {user2})       )

File2: (     (read, {user1}),      (write, {})         )

On the other hand, in the capability list system which is a counter-part of ACL system, a user is associated with a list of (action, object-list) tuple.

For example, for user1, capability list would be:

User1: (       (read, {file1,file2}),           (write, {})          )

User2: (       (read, { }),           (write, {file1} )          )