A local software company is busy with the creating text editors that are used by markets and they are decided that they wish to create one of their own. They chose the one who can write their text editor task and it was me. They do not intend to me that my software must to be featured completely but they want the text editor’s core functionality and incrementally build on it in the future because it will be necessary for the application to apply suitable features to aid developers, and to implement bracket matching to simplify debugging. It will be supported by the C# programming language but they want it to support with the other programming language in the future.

The software development company wants a simple text editor with the suitable features. This must have a suitable interface for the entering and manipulation of the text. It must have features such as file loading and saving file documents when entering is done, find the words and replace the highlight words, fixed line wrapping at a position identified by the user and to select and manipulate the text by column as well as the more traditional row.

When implementing the program, I have to made four stages. First, I will drawn the three rough designs of three forms of the program. Then, I will analyze them and start working on designation. After that, I will implement code in behind the form. The three forms I will make are the follows:

The User Register form is the form that enrolls the user’s information and store into the database. It contains seven textboxes called UserID, Username, Age, Phone Number, Email Address, Password and Address and two buttons named Register and Close. When the user put data into the form. It checks whether it has a null on a textbox or not. If there is a null, the message box will appear to enter at where the null existed. If the user registration has completed, it will show the message that the user register has successfully completed. Then, once the registration has finished, the user will press the button close to exit.

The user login form is the one that logs in the user’s account to connect the Simple Text Editor form. It consists of four textboxes named UserID, User Name, Email and Password and two buttons named Login and Register. When the user put the data into it, it checks not only the nulls in the textboxes but also the verification of the user’s information. If there is null, the message box will appear to make write on it. If there is something incorrect about the user’s account, the message box will appear that the the UserID or User Name or Email or Password has something incorrect. Once the login has finished and checked that there is no invalid data, the message box will appear that the login has successfully completed and connects to the Simple Text Editor form. If the user have not registered, he can proceed to the register form by pressing the Register button.

Once the user login has finished, it takes to the Simple Text Editor form. It has 3 menus named File, Edit and Format and a tab page at initial. But when the user pressed the file and new menu, it will show the another tool bar which contains the icons of new, open, save, cut, copy and paste and two combo boxes named Font Style Used and Font Size Used. When the user wants to use it, he can use these features by clicking the new, edit, and format menus or using the icons in the tool bar. The user can take a new page by clicking the new menu or icon, open the document using the open menu or icon, save the document using the save menu or icon, close the form using the close menu or icon, and close the form pressing the exit menu or icon. In the Edit menu, the user can use the following the features: Undo of the text, Redo of the text, copy, cut, paste, and find and replace. When the user press the find and replace, a group box will appear at the bottom and the user can find and replace the text the user wants to. Then, in the format menu, the user can write the code by using the use the “C# Code Format” and the user can also get the auto producing the pair of syntax brackets by ticking the Auto Generate checkbox on the right-top of the form. But as a difference, the tool bar will appear at the top of the rich text box when the user pressed the “C# Fomatting Code”. The user can change the font’s size and style by clicking and selecting on the font style combo box and font size combo box.

Once the designation and implementation has completed, I will create the Database of about Simple Text Editor Form. In it, I will implement the three tables named FileDetailInformation, Files, Users, and

When the database and program implementation has completed, I have to test the program. In testing, there are two types of testing: black-box testing and white-box testing. The black box testing tests the input against expected results without concerning about the internal function of the program. The white box testing tests the test input against expected results by making sure every flow of execution through a program is tested. When I test the black-box testing, I would also need to test the unit testing which accesses a function in isolation and so it can prove that the individual unit testing is correct in function. and the integration testing which allows the access of communication between units to isolate errors to the links between those units. After that, I will test the white-box testing. In white-box testing, it includes the regression testing which goes back over the code that has been fixed and checking for new errors that have been introduced as a result and the boundary testing. I will check both of them to make the program in great performance and in smooth conditions.

Once the testing has finished. I have to handle the error exceptions. After the exception handling is completed. I will give an explanation about these exceptions and how to handle it.

After all the exceptions were handled, I will make the class diagram and give the detail explanation of them. Then, I will make the description list of the class.

public partial class frmWelcome : Form

{

public frmWelcome()

{

InitializeComponent();

}

private void btnLogin\_Click(object sender, EventArgs e)

{

frmLogin lg = new frmLogin();

lg.Show();

this.Hide();

}

private void btnRegister\_Click(object sender, EventArgs e)

{

frmUser UR = new frmUser();

UR.Show();

}

}

public partial class frmUser : Form

{

dsTextEditiorTableAdapters.UsersTableAdapter udts = new dsTextEditiorTableAdapters.UsersTableAdapter();

public frmUser()

{

InitializeComponent();

}

private void AutoID()

{

DataTable dt = new DataTable();

dt = udts.GetData();

if (dt.Rows.Count == 0)

{

lblUserID.Text = "U\_00001";

}

else

{

int size = dt.Rows.Count - 1;

string oldid = dt.Rows[size][0].ToString();

int newid = Convert.ToInt16(oldid.Substring(2, 5));

if (newid >= 1 && newid <= 9)

{

lblUserID.Text = "U\_0000" + (newid + 1);

}

}

}

private void btnClose\_Click(object sender, EventArgs e)

{

Close();

}

private void btnRegister\_Click(object sender, EventArgs e)

{

clsUserRegister ur = new clsUserRegister();

ur.UID=lblUserID.Text;

ur.UName=txtUserName.Text;

ur.UAge=txtAge.Text;

ur.UAddress = txtaddress.Text;

ur.UEmail = txtEmail.Text;

ur.UPassword = txtPassword.Text;

ur.UPhone = txtPhoneNumber.Text;

if (txtUserName.Text=="")

{

MessageBox.Show("Please Enter User's Name","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtUserName.Focus();

}

else if (txtAge.Text=="")

{

MessageBox.Show("Please Enter User's Age","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtAge.Focus();

}

else if (txtPhoneNumber.Text=="")

{

MessageBox.Show("Please Enter User's Phone Number","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtPhoneNumber.Focus();

}

else if (txtEmail.Text=="")

{

MessageBox.Show("Please Enter User's Email Address","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtEmail.Focus();

}

else if (txtPassword.Text=="")

{

MessageBox.Show("Please Enter User's Password","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtPassword.Focus();

}

else if (txtaddress.Text == "")

{

MessageBox.Show("Please Enter User's Address", "User Register", MessageBoxButtons.OKCancel, MessageBoxIcon.Error);

txtaddress.Focus();

}

else

{

int result = udts.RegisterUserData(ur.UID, ur.UName, ur.UEmail, ur.UPassword, ur.UAddress, Convert.ToInt32(ur.UAge), ur.UPhone);

if (result > 0)

{

MessageBox.Show("User Registered Successfully", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

txtUserName.Focus();

txtUserName.Text = "";

txtAge.Text = "";

txtPhoneNumber.Text = "";

txtEmail.Text = "";

txtPassword.Text = "";

txtaddress.Text = "";

//frmSimpleTextEditior fste = new frmSimpleTextEditior();

//fste.Show();

}

}

}

private void frmUser\_Load(object sender, EventArgs e)

{

AutoID();

}

}

public partial class frmLogin : Form

{

dsTextEditiorTableAdapters.UsersTableAdapter udts = new dsTextEditiorTableAdapters.UsersTableAdapter();

public static string uid,uname;

public frmLogin()

{

InitializeComponent();

}

private void btnregister\_Click(object sender, EventArgs e)

{

frmUser fu = new frmUser();

fu.ShowDialog();

}

private void btnLogin\_Click(object sender, EventArgs e)

{

DataTable dt = new DataTable();

if (txtUserID.Text=="")

{

MessageBox.Show("Please Enter your User's ID", " User Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);

txtUserID.Focus();

}

else if (txtUsername.Text=="")

{

MessageBox.Show("Please enter your name", "User Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);

txtUsername.Focus();

}

else if (txtEmail.Text == "")

{

MessageBox.Show("Please enter your Email", "User Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);

txtEmail.Focus();

}

else if (txtPassword.Text == "")

{

MessageBox.Show("Please enter your password", "User Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);

txtPassword.Focus();

}

else

{

dt= udts.CheckUserLogin(txtEmail.Text, txtPassword.Text);

if (dt.Rows.Count > 0)

{

uid=dt.Rows[0][0].ToString();

uname = dt.Rows[0][1].ToString();

MessageBox.Show("User logged in successfully", "User login Form", MessageBoxButtons.OK, MessageBoxIcon.Information);

frmSimpleTextEditior fste = new frmSimpleTextEditior();

fste.Show();

this.Hide();

}

else

{

MessageBox.Show("Invalid User Login", "User Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

public partial class frmSimpleTextEditior : Form

{

public string FindID;

public string ReplaceID;

public string ID;

dsTextEditiorTableAdapters.FindAndReplaceTableAdapter fnr = new dsTextEditiorTableAdapters.FindAndReplaceTableAdapter();

dsTextEditiorTableAdapters.FilesTableAdapter ft = new dsTextEditiorTableAdapters.FilesTableAdapter();

dsTextEditiorTableAdapters.FileDetailInformationTableAdapter fdit = new dsTextEditiorTableAdapters.FileDetailInformationTableAdapter();

int NoofTags = 1;

public frmSimpleTextEditior()

{

InitializeComponent();

}

private RichTextBox GetOpenRichTextBox()

{

RichTextBox rtb = null;

TabPage tp = tabDisplay.SelectedTab;

if (tp != null)

{

rtb = tp.Controls[0] as RichTextBox;

}

return rtb;

}

private void mnuExit\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void mnuUndo\_Click(object sender, EventArgs e)

{

GetOpenRichTextBox().Undo();

rtbShow.Undo();

}

private void mnuRedo\_Click(object sender, EventArgs e)

{

GetOpenRichTextBox().Redo();

rtbShow.Redo();

}

private void mnuNew\_Click(object sender, EventArgs e)

{

try

{

tpmenu.Visible = true;

rtbShow.Hide();

RichTextBox rt = new RichTextBox();

NoofTags = NoofTags + 1;

TabPage newpage = new TabPage("Untitled" + NoofTags);

tabDisplay.TabPages.Add(newpage);

tabDisplay.SelectTab(NoofTags - 1);

tabDisplay.SelectedTab.Controls.Add(rt);

rt.AcceptsTab = true;

rt.Dock = DockStyle.Fill;

rt.BackColor = Color.White;

rt.ForeColor = Color.Black;

rt.Multiline = true;

rt.Font = new Font(this.Font.FontFamily, this.FontHeight + 2, FontStyle.Regular);

}

catch (ArgumentOutOfRangeException)

{

MessageBox.Show("Null Pages are not allowed","Null page is not accepted",MessageBoxButtons.OK,MessageBoxIcon.Error);

this.Hide();

frmSimpleTextEditior fste = new frmSimpleTextEditior();

fste.ShowDialog();

}

}

private void mnuOpen\_Click(object sender, EventArgs e)

{

Stream myStream;

RichTextBox rtb = new RichTextBox();

if (ofd.ShowDialog()==System.Windows.Forms.DialogResult.OK)

{

if ((myStream=ofd.OpenFile()) !=null)

{

string Filename = ofd.FileName;

string text = File.ReadAllText(Filename);

GetOpenRichTextBox().Text = text;

tabDisplay.SelectedTab.Text = Path.GetFileName(ofd.FileName);

}

}

}

private void AutoID()

{

DataTable dt = new DataTable();

dt = ft.GetData();

if (dt.Rows.Count == 0)

{

ID = "U\_00001";

}

else

{

int size = dt.Rows.Count - 1;

string oldid = dt.Rows[size][0].ToString();

int newid = Convert.ToInt16(oldid.Substring(2, 5));

if (newid >= 1 && newid <= 9)

{

ID = "U\_0000" + (newid + 1);

}

}

}

private void mnuSave\_Click(object sender, EventArgs e)

{

try

{

if (File.Exists(tabDisplay.SelectedTab.Text))

{

StreamWriter sw = new StreamWriter(tabDisplay.SelectedTab.Text);

sw.Write(tabDisplay.SelectedTab.Text);

sw.Close();

}

else

{

Save.Title = "Save File Page...";

Save.ShowDialog();

string filename,FileLocation;

filename = Save.FileName;

FileLocation = filename;

StreamWriter SaveAs = new StreamWriter(filename);

SaveAs.Write(tabDisplay.SelectedTab.Text);

SaveAs.Close();

tabDisplay.SelectedTab.Text = filename;

ft.Insert(ID,frmLogin.uid,filename,FileLocation);

fdit.Insert(ID,FindID,ReplaceID);

}

}

catch (ArgumentException)

{

MessageBox.Show("Do not leave with the Leave path. Please choose at least a path","Save Error for Empty Path",MessageBoxButtons.OK,MessageBoxIcon.Error);

}

}

private void mnuFind\_Click(object sender, EventArgs e)

{

lblFind.Visible = true;

lblReplace.Visible = true;

txtFind.Visible = true;

txtReplace.Visible = true;

btnFind.Visible = true;

btnReplace.Visible = true;

gpFindandReplace.Visible = true;

}

private void frmSimpleTextEditior\_Load(object sender, EventArgs e)

{

RichTextBox rt = new RichTextBox();

TabPage newpage = new TabPage("Untitled" + NoofTags);

tabDisplay.TabPages.Add(newpage);

tabDisplay.SelectTab(NoofTags - 1);

tabDisplay.SelectedTab.Controls.Add(rt);

rt.AcceptsTab = true;

rt.Dock = DockStyle.Fill;

rt.BackColor = Color.White;

rt.ForeColor = Color.Black;

rt.Multiline = true;

rt.Font = new Font(this.Font.FontFamily, this.FontHeight + 2, FontStyle.Regular);

tabDisplay.SelectedTab.BackColor = Color.LightGray;

rtbShow.Hide();

this.Text = "Welcome " + frmLogin.uname;

lblFind.Visible = false;

lblReplace.Visible = false;

btnFind.Visible = false;

btnReplace.Visible = false;

txtFind.Visible = false;

txtReplace.Visible = false;

chkGenerate.Visible = false;

gpFindandReplace.Visible = false;

tpmenu.Visible = false;

foreach (FontFamily Fonts in FontFamily.Families)

{

cboFontStyle.Items.Add(Fonts.Name.ToString());

}

}

private void closeToolStripMenuItem\_Click(object sender, EventArgs e)

{

MessageBox.Show("Do you want to close the form?", "Close form", MessageBoxButtons.OKCancel, MessageBoxIcon.Information);

TabPage ct = tabDisplay.SelectedTab;

tabDisplay.TabPages.Remove(ct);

}

private void rtbShow\_TextChanged(object sender, EventArgs e)

{

string pattern = "";

string[] keywords =

{

"bool","int","string","class","public","foreach","private","void"

};

foreach (var item in keywords)

{

pattern += item + "|";

}

Regex R = new Regex(pattern);

int index = rtbShow.SelectionStart;

foreach (Match m in R.Matches(rtbShow.Text))

{

rtbShow.Select(m.Index, m.Value.Length);

rtbShow.SelectionColor = Color.Blue;

rtbShow.SelectionStart = index;

}

rtbShow.SelectionColor = Color.Black;

}

private void cFormattingCodeToolStripMenuItem\_Click(object sender, EventArgs e)

{

tpmenu.Visible = true;

chkGenerate.Visible = true;

rtbShow.Show();

RichTextBox rt = new RichTextBox();

tabDisplay.SelectedTab.Controls.Add(rt);

rt.AcceptsTab = true;

rt.Dock = DockStyle.Fill;

rt.BackColor = Color.White;

rt.ForeColor = Color.Black;

rt.Multiline = true;

rt.Font = new Font(this.Font.FontFamily, this.FontHeight + 2, FontStyle.Regular);

}

private void mnuCopy\_Click(object sender, EventArgs e)

{

GetOpenRichTextBox().Copy();

rtbShow.Copy();

}

private void mnuCut\_Click(object sender, EventArgs e)

{

GetOpenRichTextBox().Cut();

rtbShow.Cut();

}

private void mnuPaste\_Click(object sender, EventArgs e)

{

GetOpenRichTextBox().Paste();

rtbShow.Paste();

}

private void rtbShow\_KeyPress(object sender, KeyPressEventArgs e)

{

string s = e.KeyChar.ToString();

int sel = rtbShow.SelectionStart;

if (chkGenerate.Checked==true)

{

switch (s)

{

case "(": rtbShow.Text = rtbShow.Text.Insert(sel, "()");

e.Handled = true;

rtbShow.SelectionStart = sel + 1;

break;

case "{" :

string t="{}";

rtbShow.Text=rtbShow.Text.Insert(sel,"{}");

e.Handled=true;

rtbShow.SelectionStart=sel + t.Length-1;

//conkey = true;

break;

case "[": rtbShow.Text = rtbShow.Text.Insert(sel, "[]");

e.Handled = true;

rtbShow.SelectionStart = sel + 1;

break;

case "<": rtbShow.Text = rtbShow.Text.Insert(sel, "<>");

e.Handled = true;

rtbShow.SelectionStart = sel + 1;

break;

case "\"": rtbShow.Text=rtbShow.Text.Insert(sel,"\"\"");

e.Handled=true;

rtbShow.SelectionStart=sel+1;

break;

}

}

}

private void btnFind\_Click(object sender, EventArgs e)

{

int start = 0;

int end = rtbShow.Text.LastIndexOf(txtFind.Text);

if (txtFind.Text=="")

{

MessageBox.Show("Please Enter Find Data", "Enter the word", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);

}

else if (rtbShow.Text=="")

{

MessageBox.Show("No Words are found in the box", "Find", MessageBoxButtons.OKCancel, MessageBoxIcon.Error);

}

else

{

while (start<end)

{

rtbShow.Find(txtFind.Text, start, rtbShow.TextLength, RichTextBoxFinds.MatchCase);

rtbShow.SelectionBackColor = Color.LightBlue;

start = rtbShow.Text.IndexOf(txtFind.Text, start) + 1;

rtbShow.SelectAll();

}

}

}

private void btnReplace\_Click(object sender, EventArgs e)

{

rtbShow.SelectAll();

rtbShow.SelectionBackColor = Color.Cyan;

try

{

string search = txtFind.Text.Trim();

string replace = txtReplace.Text.Trim();

if (search=="")

{

MessageBox.Show("Please Enter The Replace Word", "Type the Replace word", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

string newtext = rtbShow.Text.Replace(search, replace);

rtbShow.Text = newtext;

rtbShow.SelectAll();

fnr.Insert(ID, ID, search, replace);

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void cboFontStyle\_SelectedIndexChanged(object sender, EventArgs e)

{

try

{

rtbShow.Font = new Font(cboFontStyle.Text, rtbShow.Font.Size);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void tsnew\_Click(object sender, EventArgs e)

{

try

{

rtbShow.Hide();

RichTextBox rt = new RichTextBox();

NoofTags = NoofTags + 1;

TabPage newpage = new TabPage("Untitled" + NoofTags);

tabDisplay.TabPages.Add(newpage);

tabDisplay.SelectTab(NoofTags - 1);

tabDisplay.SelectedTab.Controls.Add(rt);

rt.AcceptsTab = true;

rt.Dock = DockStyle.Fill;

rt.BackColor = Color.Black;

rt.ForeColor = Color.Cyan;

rt.Multiline = true;

rt.Font = new Font(this.Font.FontFamily, this.FontHeight + 2, FontStyle.Regular);

}

catch (ArgumentOutOfRangeException)

{

MessageBox.Show("Null Pages are not allowed", "Null page is not accepted", MessageBoxButtons.OK, MessageBoxIcon.Error);

this.Hide();

frmSimpleTextEditior fste = new frmSimpleTextEditior();

fste.ShowDialog();

}

}

private void tsopen\_Click(object sender, EventArgs e)

{

Stream myStream;

RichTextBox rtb = new RichTextBox();

if (ofd.ShowDialog() == System.Windows.Forms.DialogResult.OK)

{

if ((myStream = ofd.OpenFile()) != null)

{

string Filename = ofd.FileName;

string text = File.ReadAllText(Filename);

GetOpenRichTextBox().Text = text;

tabDisplay.SelectedTab.Text = Path.GetFileName(ofd.FileName);

}

}

}

private void tsSave\_Click(object sender, EventArgs e)

{

try

{

if (File.Exists(tabDisplay.SelectedTab.Text))

{

StreamWriter sw = new StreamWriter(tabDisplay.SelectedTab.Text);

sw.Write(tabDisplay.SelectedTab.Text);

sw.Close();

}

else

{

Save.Title = "Save File Page...";

Save.ShowDialog();

string filename;

filename = Save.FileName;

StreamWriter SaveAs = new StreamWriter(filename);

SaveAs.Write(tabDisplay.SelectedTab.Text);

SaveAs.Close();

tabDisplay.SelectedTab.Text = filename;

}

}

catch (ArgumentException)

{

MessageBox.Show("Do not leave with the empty path. Please choose at least a path", "Save Error for Empty Path", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void cboFontSize\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void tscut\_Click(object sender, EventArgs e)

{

GetOpenRichTextBox().Cut();

rtbShow.Cut();

}

private void tscopy\_Click(object sender, EventArgs e)

{

GetOpenRichTextBox().Copy();

rtbShow.Copy();

}

private void tspaste\_Click(object sender, EventArgs e)

{

GetOpenRichTextBox().Paste();

rtbShow.Paste();

}

private void tsfontsize\_SelectedIndexChanged(object sender, EventArgs e)

{

try

{

rtbShow.Font = new Font(cboFontStyle.Font.FontFamily, float.Parse(tsfontsize.SelectedItem.ToString()));

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

}

No Test Case Form Used Date Time Purpose

1 1.1 Frmuser

/txtUserName 10/6/19 4:21PM To check null values in username textbox

2 1.2 frmuser/txtAge 10/6/19 4:21PM To check null values in the Age textbox

3 1.3 frmuser/txtPhoneNumber 10/6/19 4:24PM To check null values in the Phone Number textbox

4 1.4 frmuser/txtEmail 10/6/19 4:25PM To check null values in the Email Address textbox

5. 1.5 frmuser/txtPassword 10/6/19 4:25PM To check null values in the password textbox

6. 1.6 frmuser/txtaddress 10/6/19 4:26PM To check null values in the address textbox

7. 1.7 frmuser/btnRegister 10/6/19 4:27PM To insert the user’s data in the database

8. 1.8 Frmuser/btnClose 10/6/19 4:28PM To close the user registration form

No Test Case Form Used Date Time Purpose

1. 2.1 FrmLogin/

txtUserID 10/10/2019 2:07 PM To check the null values in the User ID textbox

2. 2.2 FrmLogin/

txtUsername 10/10/2019 2:08 PM To check the null values in the User Name textbox

3. 2.3 FrmLogin/

txtEmail 10/10/2019 2:09 PM To check the null values in the User Email textbox

4 2.4 FrmLogin/

txtPassword 10/10/2019 2:10 PM To check the null values in the User Password textbox

5. 2.5 FrmLogin/

btnLogin 10/10/2019 2:11 PM To check the user’s data entered and log into the text editor

6. 2.6 FrmLogin/btnClose 10/10/2019 2:13 PM To close the User Login form

No Test Case Form Used Date Time Purpose

1 3.1 FrmSimpleTextEditor/mnuNew 24/10/2019 11:37AM To add the new tab page

2. 3.2 FrmSimpleTextEditor/mnuOpen 24/10/2019 11:38AM To open the tab page that has been written

3 3.3 FrmSimpleTextEditor/mnuSave 24/10/2019 11:38AM To Save the document File

4 3.4 FrmSimpleTextEditor/mnuClose 24/10/2019 11:39AM To Close the document File

5 3.5 FrmSimpleTextEditor/mnuExit 24/10/2019 11:40AM To exit from the application

6. 3.6 FrmSimpleTextEditor/mnuUndo 24/10/2019 11:43AM To rewind the words and sentences that has been deleted or copied or pasted

7 3.7 FrmSimpleTextEditor/mnuRedo 24/10/2019 11:44AM To forward the words and sentences that has been deleted or copied or pasted before the undo

8 3.8 FrmSimpleTextEditor/mnuCopy 24/10/2019 11:45AM To Copy the words and sentences that has been written

9 3.9 FrmSimpleTextEditor/mnuCut 24/10/2019 11:46AM To cut the words and sentences that are not wronged or necessary

10 3.10 FrmSimpleTextEditor/mnuPaste 24/10/2019 11:48AM To paste onto the word and sentence with the new words or sentences

11 3.11 FrmSimpleTextEditor/mnuFind 24/10/2019 11:50AM To open the find and replace groupbox that finds and replaces words which wants to be changed

12 3.12 FrmSimpleTextEditor/cFormattingCodeToolStripMenuItem 24/10/2019 11:52AM To open the richtextbox that can be written with C# programming language

13 3.13 FrmSimpleTextEditor/tsnew 24/10/2019 11:54AM To add the new tab page by using the new toolbar

14 3.14 FrmSimpleTextEditor/tsopen 24/10/2019 11:55AM To open the tab page that has been written by using the open toolbar

15 3.15 FrmSimpleTextEditor/tsSave 24/10/2019 11:56AM To Save the document File by using the save toolbar

16 3.16 FrmSimpleTextEditor/tscut 24/10/2019 11:57AM To cut the words and sentences that are not wronged or necessary by using the cut toolbar

17 3.17 FrmSimpleTextEditor/tscopy 24/10/2019 11:58AM To Copy the words and sentences that has been written by using the copy toolbar

18 3.18 FrmSimpleTextEditor/tspaste 24/10/2019 11:59AM To paste onto the word and sentence with the new words or sentences by using the paste toolbar

19 3.19 FrmSimpleTextEditor/btnFind 24/10/2019 12:05PM To find and highlight the words typed in the textbox

20 3.20 FrmSimpleTextEditor/btnReplace 24/10/2019 12:09PM To replace the word with new words typed in the textbox

21 3.21 FrmSimpleTextEditor/cboFontStyle 24/10/2019 12:17PM To choose the words’ font format style

22 3.22 FrmSimpleTextEditor/tsfontsize 24/10/2019 12:18PM To adjust the words’ font size

23 3.23 FrmSimpleTextEditor/chkGenerate 24/10/2019 12:20PM To generate the various types of C# style syntax-brackets by writing to it

No. Form Functions Date Time

1 frmUser.cs btnRegister\_click(),btnClose\_Click() 6th Oct, 2019 4:21AM

2 frmLogin.cs btnLogin\_click(),btnRegister\_Click() 10th Oct,2019 3:33PM

3. frmSimpleTextEditor.cs mnuNew\_Click(),mnuOpen\_Click(),

mnuSave\_Click(),mnuClose\_Click(),

mnuExit\_Click(),mnuUndo\_Click(),

mnuRedo\_Click(),mnuCopy\_Click(),

mnuCut\_Click(),mnuPaste\_Click(),

mnuFind\_Click(),

cFormattingCodeToolStripMenuItem\_Click(),

tsnew\_Click(),tsopen\_Click(),tsSave\_Click(),

tscut\_Click(),tscopy\_Click(),tspaste\_Click(),

btnFind\_Click(),btnReplace\_Click() 24th Oct,2019 12:16AM

Test Case 1.1

Test Objective To check null values in username textbox

Procedure 1.Run frmUser

2.Click “Register” button

Test Data User ID=U\_00001, User name=null

Expected Result Show message box with error message

Actual Result Fig-1.1.1 , Fig-1.1.2

Test Case 1.2

Test Objective To check null values in the Age textbox

Procedure 1.Run frmUser

2.Type “Kaung Htet Kyaw” the “username”Textbox

3. Click “Register” button

Test Data User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=null

Expected result Show message box with error message

Actual Result Fig-1.2.1 Fig-1.2.2

Test Case 1.3

Test Objective To check null values in the Phone Number textbox

Procedure 1.Run frmUser

2.Type User Name “Kaung Htet Kyaw” in the “username” textbox

3.Type Age “17” in the “Age” textbox

4.Click “Register” button

Test Data User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=17, Phone Number= null

Expected result Show message box with error message

Actual Result Fig-1.3.1 Fig-1.3.2

Test Case 1.4

Test Objective To check null values in the Email Address textbox

Procedure 1.Run frmUser

2.Type User Name= “Kaung Htet Kyaw” in the “username” textbox

3.Type Age= “17” in the “Age” textbox

4.Type Phone Number= “0189532” in the “PhoneNumber” textbox

5. Click “Register” button

Test Data User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=17, Phone Number=0189532, Email Address=null

Expected result Show message box with error message

Actual Result Fig-1.4.1 Fig-1.4.2

Test Case 1.5

Test Objective To check null values in the Password textbox

Procedure 1.Run frmUser

2.Type User Name= “Kaung Htet Kyaw” in the username textbox

3.Type Age=”17” in the Age textbox

4.Type Phone Number=”0189532” in the “PhoneNumber” textbox

5.Type Email Address=”khk@gmail.com” in the “email” textbox

6. Click “Register” button

Test Data User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=17, Phone Number=0189532, Email Address= khk@gmail.com, password=null

Expected result Show message box with error message

Actual Result Fig-1.5.1 Fig-1.5.2

Test Case 1.6

Test Objective To check null values in the Address textbox

Procedure 1.Run frmUser

2.Type User Name= “Kaung Htet Kyaw” in the username textbox

3.Type Age=”17” in the Age textbox

4.Type Phone Number=”0189532” in the “PhoneNumber” textbox

5.Type Email Address=”khk@gmail.com” in the “email” textbox

6.Type Password= “khk2019” in the “Password” textbox

7. Click “Register” button

Test Data User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=17, Phone Number= 0189532, Email Address= khk@gmail.com, Password= khk2019, Address= null

Expected result Show message box with error message

Actual Result Fig-1.6.1 Fig-1.6.2

Test Case 1.7

Test Objective To check and insert into the database if there is no null

Procedure 1.Run frmUser

2.Type User Name= “Kaung Htet Kyaw” in the username textbox

3.Type Age=”17” in the Age textbox

4.Type Phone Number=”0189532” in the “PhoneNumber” textbox

5.Type Email Address=”khk@gmail.com” in the “email” textbox

6.Type Password= “khk2019” in the “Password” textbox

7.Type Address=”Yangon” in the “address” textbox

8. Click “Register” button

Test Data User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=17, Phone Number=0189532, Email Address= khk@gmail.com, Password= khk2019, Address = Yangon

Expected result Show message box to tell that the data is successfully registered with information message

Actual Result Fig-1.7.1 Fig-1.7.2

Test Case 2.1

Test Objective To check and log into the text editor form if there is no null and invalid data

Procedure 1.Run “frmLogin” form

2.Click “Login” button

Test Data UserID=null

Expected result Show Message box with error message

Actual Result Fig 2.1.1, Fig 2.1.2

Test Case 2.2

Test Objective To check and log into the text editor form if there is no null and invalid data

Procedure 1.Run “frmLogin” form

2.Type UserID=”U\_00001” in the UserID textbox

3.Click “Login” button

Test Data UserID=U\_00001, User Name=null

Expected result Show Message box with error message

Actual Result Fig 2.2.1, Fig 2.2.2

Test Case 2.3

Test Objective To check and log into the text editor form if there is no null and invalid data

Procedure 1.Run “frmLogin” form

2.Type UserID=”U\_00001” in the UserID textbox

3.Type User Name=”Kaung Htet Kyaw” in the User Name textbox

4.Click “Login” button

Test Data UserID=U\_00001, User Name=Kaung Htet Kyaw, Email=null

Expected result Show Message box with error message

Actual Result Fig 2.3.1, Fig 2.3.2

Test Case 2.4

Test Objective To check and log into the text editor form if there is no null and invalid data

Procedure 1.Run “frmLogin” form

2.Type UserID=”U\_00001” in the UserID textbox

3.Type User Name=”Kaung Htet Kyaw” in the User Name textbox

4.Type Email = “khk@gmail,com” in the Email textbox

5.Click “Login” button

Test Data UserID=U\_00001, User Name=Kaung Htet Kyaw, Email=khk@gmail,com, Password=null

Expected result Show Message box with error message

Actual Result Fig 2.4.1, Fig 2.4.2

Test Case 2.5

Test Objective To check and log into the text editor form if there is no null and invalid data

Procedure 1.Run “frmLogin” form

2.Type UserID=”U\_00001” in the UserID textbox

3.Type User Name=”Kaung Htet Kyaw” in the User Name textbox

4.Type Email = “khk@gmail,com” in the Email textbox

5.Type Password=”khk2019” in the Password textbox

6.Click “Login” button

Test Data UserID=U\_00001, User Name=Kaung Htet Kyaw, Email=khk@gmail,com, Password= khk2019

Expected result Show Message Box with information message that successfully logged into the form.

Actual Result Fig 2.5.1, Fig 2.5.2

Test Case 3.1

Test Objective To add the new tab page

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “File” menu strip

3.Click the “New” menu strip

Test Data Opening the new tab page

Expected result The new tab page will be added to the tab control form

Actual Result Fig-3.1.1, Fig-3.2.1

Test Case 3.2

Test Objective To open a tab page

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “File” menu strip

3.Click the “Open” menu strip

4.Type the file name that you wish to open in the text box and then click the “Open” button

Test Data Opening or loading the existing document file to the tab page

Expected result The existed document will be added to the tabpage

Actual Result Fig-3.2.1, Fig-3.2.2, Fig-3.2.3

Test Case 3.3

Test Objective To save the document file

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “File” menu strip

3.Click the “Save” menu strip

4.Type the file name if you want to save in the textbox and then click the “Save” Button

Test Data Saving the file

Expected result The document file will be saved

Actual Result Fig-3.3.1, Fig-3.3.2, Fig-3.3.3

Test Case 3.4

Test Objective To close the document file

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “File” menu strip

3.Click the “Close” menu strip

Test Data Showing the message box whether it would be closed or not and Closing the document file

Expected result The document file will be closed

Actual Result Fig-3.4.1, Fig-3.4.2, Fig-3.4.3

Test Case 3.5

Test Objective To close the Application

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “File” menu strip

3.Click the “Exit” menu strip

Test Data The application closes

Expected result The document file will be closed and also the application will be closed

Actual Result Fig-3.5.1, Fig-3.5.2

The testing was stopped due to the application was exited.

Test Case 3.6

Test Objective To rewind the words and sentences that has been deleted or copied or pasted

Procedure 1.Run “FrmSimpleTextEditor”

2.Type the words in the document and delete the word “World”

2.Click the “Edit” menu strip

3.Click the “Undo” menu strip

Test Data Word=Hello World, the word will be used to undo= World

Expected result The deleted or pasted words will be there After clicking the “Undo”

Actual Result Fig-3.6.1, Fig-3.6.2, Fig-3.6.3

Test Case 3.7

Test Objective To forward the words and sentences that has been deleted or copied or pasted before the undo

Procedure 1.Run “FrmSimpleTextEditor”

2.Type the words in the document and delete the word “World”

3.Click the “Edit” menu strip

4.Click the “Undo” menu strip

5.Click the “Redo” menu strip

Test Data Word=Hello World, the word will be used to undo and redo= World

Expected result The conditions will be reversed with the Undo state. The rewind word will be deleted like the time of before doing undo

Actual Result Fig-3.7.1, Fig-3.7.2, Fig-3.7.3, Fig-3.7.4

Test Case 3.8

Test Objective To Copy the words and sentences that has been written

Procedure 1.Run “FrmSimpleTextEditor”

2.Type the words in the document

3.Click the “Edit” menu strip

4.Click the “Copy” menu strip

Test Data Word=Hello World

Expected result The words will be copied

Actual Result Fig-3.8.1, Fig-3.8.2,

Test Case 3.9

Test Objective To cut the words and sentences that are not wronged or necessary

Procedure 1.Run “FrmSimpleTextEditor”

2.Type the words in the document

3.Select the words

4.Click the “Edit” menu strip

5.Click the “Cut” menu strip

Test Data Word=Hello World

Expected result The words will be cut

Actual Result Fig-3.9.1, Fig-3.9.2,

The word selected is cut.

Test Case 3.10

Test Objective To paste onto the word and sentence with the new words or sentences

Procedure 1.Run “FrmSimpleTextEditor”

2.Type the words in the document

3.Select the words

4.Click the “Edit” menu strip

5.Click the “Copy” menu strip

6.Select the cursor in the another blank and click the “Paste’ menu strip

Test Data Word=Hello World

Expected result The words will be pasted

Actual Result Fig-3.10.1, Fig-3.10.2,

Test Case 3.11

Test Objective To open the find and replace groupbox that finds and replaces words which wants to be changed

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “Edit” menu strip

3.Click the “Find and Replace” menu strip

Test Data Opening the group box

Expected result The group box of find and replace will appear at the bottom of the form

Actual Result Fig-3.11.1, Fig-3.11.2,

The group box has appeared at the button of the form.

Test Case 3.12

Test Objective To open the richtextbox that can be written with C# programming language

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “Format” menu strip

3.Click the “C# Formatting Code” menu strip

Test Data Open the richtextbox that can be written with C# programming language

Expected result The rich text box will appear and it can be written with C# programming language syntaxes

Actual Result Fig-3.12.1, Fig-3.12.2,

The rich text box has appeared and it can be written with C# Programming language.

Test Case 3.13

Test Objective To add the new tab page by using the new toolbar

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “New” tool strip

Test Data Adding the new document by using the new toolbar

Expected result The new document will be added

Actual Result Fig-3.13.1, Fig-3.13.2,

Test Case 3.14

Test Objective To open the tab page that has been written by using the open toolbar

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “Open” tool strip

Test Data Opening the open dialog box

Expected result The open file dialog box will be appeared

Actual Result Fig-3.14.1, Fig-3.14.2,

Test Case 3.15

Test Objective To Save the document File by using the save toolbar

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “Save” icon tool strip

Test Data Saving the document file

Expected result The selected word will be cut

Actual Result Fig-3.15.1, Fig-3.15.2,

Test Case 3.16

Test Objective To cut the words and sentences that are not wronged or necessary by using the cut toolbar

Procedure 1.Run “FrmSimpleTextEditor”

2.Type the word “Hello World”

3.Select the word

2.Click the “Cut” tool strip

Test Data Word=Hello World

Expected result The selected word will be cut

Actual Result Fig-3.15.1, Fig-3.15.2,

Test Case 3.17

Test Objective To Copy the words and sentences that has been written by using the copy toolbar

Procedure 1.Run “FrmSimpleTextEditor”

2.Type the word “Hello World” into the rich text box

3. Select the Word

4. Click the “Copy” icon tool strip

Test Data Word =” Hello World”

Expected result The Selected word will be copied

Actual Result Fig-3.17.1, Fig-3.17.2

The Selected text is copied.

Test Case 3.18

Test Objective To paste onto the word and sentence with the new words or sentences by using the paste toolbar

Procedure 1.Run “FrmSimpleTextEditor”

2.Place the cursor on the blank place

3.Click the “Paste” icon tool strip

Test Data Word = “Hello World”

Expected result The copied word will be pasted

Actual Result Fig-3.18.1, Fig-3.18.2

Test Case 3.19

Test Objective To find and highlight the words typed in the textbox

Procedure 1.Run “FrmSimpleTextEditor”

2.Open the “C# Formatting Code”

3.Type the Word “Hello World” and repeat to it

4.Click the “Edit” icon menu strip

5.Click the “Find and Replace” menu strip

6.Type the word that wish to search in in the text box

7.Click the “Find” button

Test Data Word=Hello World, Find word= Hello

Expected result The word that typed in the text box will be highlighted in the Rich Text Box

Actual Result Fig-3.19.1, Fig-3.19.2

Test Case 3.20

Test Objective To replace the word with new words typed in the textbox

Procedure 1.Run “FrmSimpleTextEditor”

2.Open the “C# Formatting Code”

3.Type the Word “Hello World” and repeat to it

4.Click the “Edit” icon menu strip

5.Click the “Find and Replace” menu strip

6.Type the word that you wish to search in the rich text box

7.Click the “Find” button

8. Type the word that you wish to replace in the rich text box

9. Click the “Replace” button

Test Data Word = Hello World, Find word= Hello , Replace Word= How are You?

Expected result The word typed in the Replace text box will replace the words existing in the rich text box

Actual Result Fig-3.20.1, Fig-3.20.2,

Test Case 3.21

Test Objective To choose the words’ font format style

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “C# Formatting Code” menu strip

3.Click onto the Font Style combo box

4.Select the style to “Castellar”

5.Type the words

Test Data Word= I am Kaung Htet Kyaw

Expected result The word’s font style will be changed

Actual Result Fig-3.21.1, Fig-3.21.2,Fig-3.21.3

It selects the font’s style.

Test Case 3.22

Test Objective To adjust the words’ font size

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “C# Formatting Code” menu strip

3.Click onto the Font Style combo box

4.Select the style to “Castellar”

5.Select the size to “12”

5.Type the words

Test Data Word= I am Kaung Htet Kyaw

Expected result The font’s size will be changed

Actual Result Fig-3.22.1, Fig-3.22.2, Fig-3.22.3

It selects the font’s size to 12.

Test Case 3.23

Test Objective To generate the various types of C# style syntax-brackets writing to it

Procedure 1.Run “FrmSimpleTextEditor”

2.Click the “C# Formatting Code” tool strip

3.Tick the “Auto Generate” check box

4.Type the code

Test Data generating the pair of C# style syntax-brackets automatically

Expected result Another pair of brackets will be generated once the one side of bracket is writing.

Actual Result Fig-3.23.1, Fig-3.23.2

1. try

2. {

3. if (File.Exists(tabDisplay.SelectedTab.Text))

4. {

5. StreamWriter sw = new StreamWriter(tabDisplay.SelectedTab.Text);

6. sw.Write(tabDisplay.SelectedTab.Text);

7. sw.Close();

8. }

9. else

10. {

11. Save.Title = "Save File Page...";

12. Save.ShowDialog();

13. string filename;

14. filename = Save.FileName;

15. StreamWriter SaveAs = new StreamWriter(filename);

16. SaveAs.Write(tabDisplay.SelectedTab.Text);

17. SaveAs.Close();

18. tabDisplay.SelectedTab.Text = filename;

19. }

20. }

21. catch (Exception ioe)

22. {

23. MessageBox.Show(ioe.Message);

24. }

No. Variable Condition Output

1 try Initial state and Works when the input taken the correct values

2 {

3 If(File.Exists(tabDisplay.SelectedTab.Text)) True Check the condition and works if the condition is true

4 {

5 StreamWriter sw = new StreamWriter(tabDisplay.SelectedTab.Text); Constructing Stream object

6 sw.Write(tabDisplay.SelectedTab.Text); Makes the Stream object to write text in the Tab Page

7 sw.Close(); Close the Stream

8 }

9 Else False Go to the other function when it is false

10 {

11 Save.Title=”Save File Page…” Name the save the dialog box’s name

12 Save.Showdialog() Show the dialog box to save if there is no saved files

13 string filename; Declaring the variable and its datatype as string

14 filename= save.filename; Assign the value into the variable

15 StreamWriter SaveAs = new StreamWriter(filename);

Declaring the variable to construct StreamWriter object

16 SaveAs.Write(tabDisplay.SelectedTab.Text);

Makes the Stream object to write text in the Tab Page

17 SaveAs.Close();

Close the Stream

18 tabDisplay.SelectedTab.Text = filename;

Assigning the value into the selected Tab Page

19 }

20 }

21 catch (Exception ioe)

Catch the exception when the input was encountered error

22 {

23 MessageBox.Show(ioe.Message);

Control the error with the error message

24 }

1. try

2. {

3. tpmenu.Visible = true;

4. rtbShow.Hide();

5. RichTextBox rt = new RichTextBox();

6. NoofTags = NoofTags + 1;

7. TabPage newpage = new TabPage("Untitled" + NoofTags);

8. tabDisplay.TabPages.Add(newpage);

9. tabDisplay.SelectTab(NoofTags - 1);

10. tabDisplay.SelectedTab.Controls.Add(rt);

11. rt.AcceptsTab = true;

12. rt.Dock = DockStyle.Fill;

13. rt.BackColor = Color.White;

14. rt.ForeColor = Color.Black;

15. rt.Multiline = true;

16. rt.Font = new Font(this.Font.FontFamily, this.FontHeight + 2, FontStyle.Regular);

17. }

18. catch (ArgumentOutOfRangeException)

19. {

20. MessageBox.Show("Null Pages are not allowed","Null page is not accepted",MessageBoxButtons.OK,MessageBoxIcon.Error);

21. this.Hide();

22. frmSimpleTextEditior fste = new frmSimpleTextEditior();

23. fste.ShowDialog();

24. }

No. Variable Condition Output

1 Try It is the initial stage and it makes the function once the input is correct

2 { Starts code

3 tpmenu.Visible = true;

Allows the toolbar menu in visible condition

4 rtbShow.Hide(); Hides the rich text box at first

5 RichTextBox rt = new RichTextBox(); Creates the rich text box object

6 NoofTags = NoofTags + 1; Creates the variable and increment the variable by 1

7 TabPage newpage = new TabPage("Untitled" + NoofTags); Creates the “newpage” TabPage object and adding parameters with Tab page name and the variable

8 tabDisplay.TabPages.Add(newpage); Adds the new pages to the tab pages when picking the new page

9 tabDisplay.SelectTab(NoofTags - 1);

10 tabDisplay.SelectedTab.Controls.Add(rt); Adds the rich text box into the Tab Control Page

11 rt.AcceptsTab = true; Accepts the tab to add the rich text box

12 rt.Dock = DockStyle.Fill; Makes the rich box styles to fill and depress the tab control page

13 rt.BackColor = Color.White; Defines the rich text box’s background blank color as white

14 rt.ForeColor = Color.Black; Defines the rich text box’s forecolor(words typed) as black

15 rt.Multiline = true; Accepts the amount of lines that will be typed

16 rt.Font = new Font(this.Font.FontFamily, this.FontHeight + 2, FontStyle.Regular); Makes the rich text box’s font by using object method and adding the parameters of fontfamily, font’s height’ and font’s style

17 }

18 catch (ArgumentOutOfRangeException) Catch the exception when the input data is incorrect

19 {

20 MessageBox.Show("Null Pages are not allowed","Null page is not accepted",MessageBoxButtons.OK,MessageBoxIcon.Error); Shows the error with the error message

21 this.Hide(); Hides the form to handle the exception

22 frmSimpleTextEditior fste = new frmSimpleTextEditior(); Create the object to open the form

23 fste.ShowDialog(); Makes the object to open the form

24 }

The exception is the problem that occurs during the execution of the program. It is also a response to an exceptional circumstance that happens during the program runtime. In this form, as the form has used the FileInfo, there are three exceptions that can happen in the Simple Text Editor:

The Argument exception is the exception when the one of the arguments provided to the method is invalid.

The argument out of range exception is the exception when a method is invoked and at least of one arguments passed to the method is not null and an invalid value that is not a member of the set of values which is expected for the argument.

The IO exception is the exception when there is a failure during reading, writing and searching files.

try

1. {

2. if (File.Exists(tabDisplay.SelectedTab.Text))

3. {

4. StreamWriter sw = new StreamWriter(tabDisplay.SelectedTab.Text);

5. sw.Write(tabDisplay.SelectedTab.Text);

6. sw.Close();

7. }

8. else

9. {

10. Save.Title = "Save File Page...";

11. Save.ShowDialog();

12. string filename;

13. filename = Save.FileName;

14. StreamWriter SaveAs = new StreamWriter(filename);

15. SaveAs.Write(tabDisplay.SelectedTab.Text);

16. SaveAs.Close();

17. tabDisplay.SelectedTab.Text = filename;

18. }

19. }

20. catch (Exception ioe)

21. {

22. MessageBox.Show(ioe.Message);

23. }

1. try

2. {

3. tpmenu.Visible = true;

4. rtbShow.Hide();

5. RichTextBox rt = new RichTextBox();

6. NoofTags = NoofTags + 1;

7. TabPage newpage = new TabPage("Untitled" + NoofTags);

8. tabDisplay.TabPages.Add(newpage);

9. tabDisplay.SelectTab(NoofTags - 1);

10. tabDisplay.SelectedTab.Controls.Add(rt);

11. rt.AcceptsTab = true;

12. rt.Dock = DockStyle.Fill;

13. rt.BackColor = Color.White;

14. rt.ForeColor = Color.Black;

15. rt.Multiline = true;

16. rt.Font = new Font(this.Font.FontFamily, this.FontHeight + 2, FontStyle.Regular);

17. }

18. catch (ArgumentOutOfRangeException)

19. {

20. MessageBox.Show("Null Pages are not allowed","Null page is not accepted",MessageBoxButtons.OK,MessageBoxIcon.Error);

21. this.Hide();

22. frmSimpleTextEditior fste = new frmSimpleTextEditior();

23. fste.ShowDialog();

24. }

public partial class frmUser : Form

{

dsTextEditiorTableAdapters.UsersTableAdapter udts = new dsTextEditiorTableAdapters.UsersTableAdapter();

public frmUser()

{

InitializeComponent();

}

private void AutoID()

{

DataTable dt = new DataTable();

dt = udts.GetData();

if (dt.Rows.Count == 0)

{

lblUserID.Text = "U\_00001";

}

else

{

int size = dt.Rows.Count - 1;

string oldid = dt.Rows[size][0].ToString();

int newid = Convert.ToInt16(oldid.Substring(2, 5));

if (newid >= 1 && newid <= 9)

{

lblUserID.Text = "U\_0000" + (newid + 1);

}

}

}

private void btnClose\_Click(object sender, EventArgs e)

{

Close();

}

private void btnRegister\_Click(object sender, EventArgs e)

{

clsUserRegister ur = new clsUserRegister();

ur.UID=lblUserID.Text;

ur.UName=txtUserName.Text;

ur.UAge=txtAge.Text;

ur.UAddress = txtaddress.Text;

ur.UEmail = txtEmail.Text;

ur.UPassword = txtPassword.Text;

ur.UPhone = txtPhoneNumber.Text;

if (txtUserName.Text=="")

{

MessageBox.Show("Please Enter User's Name","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtUserName.Focus();

}

else if (txtAge.Text=="")

{

MessageBox.Show("Please Enter User's Age","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtAge.Focus();

}

else if (txtPhoneNumber.Text=="")

{

MessageBox.Show("Please Enter User's Phone Number","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtPhoneNumber.Focus();

}

else if (txtEmail.Text=="")

{

MessageBox.Show("Please Enter User's Email Address","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtEmail.Focus();

}

else if (txtPassword.Text=="")

{

MessageBox.Show("Please Enter User's Password","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtPassword.Focus();

}

else if (txtaddress.Text=="")

{

MessageBox.Show("Please Enter User's Address","User Register",MessageBoxButtons.OKCancel,MessageBoxIcon.Error);

txtaddress.Focus();

}

else

{

try

{

int result=udts.RegisterUserData(ur.UID,ur.UName,ur.UEmail,ur.UPassword,ur.UAddress,Convert.ToInt32(ur.UAge),ur.UPhone);

if (result>0)

{

MessageBox.Show("User Registered Successfully", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

txtUserName.Focus();

txtUserName.Text = "";

txtAge.Text = "";

txtPhoneNumber.Text = "";

txtEmail.Text = "";

txtPassword.Text = "";

txtaddress.Text = "";

//frmSimpleTextEditior fste = new frmSimpleTextEditior();

//fste.Show();

}

}

catch

{

}

}

}

private void frmUser\_Load(object sender, EventArgs e)

{

AutoID();

}

}

Test Test Date Test Data Type of Data Expected Result Actual Result Meet excepted Result

Frmuser

/txtUserName 10/6/19 User ID=U\_00001, User name=null String Show message box with error message YES

frmuser/

txtAge 10/6/19 User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=null Integer Show message box with error message YES

frmuser/

txtPhoneNumber 10/6/19 User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=17, Phone Number= null String Show message box with error message YES

frmuser/

txtEmail 10/6/19 User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=17, Phone Number=0189532, Email Address=null String Show message box with error message YES

frmuser/

txtPassword 10/6/19 User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=17, Phone Number=0189532, Email Address= khk@gmail.com, password=null String Show message box with error message YES

frmuser/

txtaddress 10/6/19 User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=17, Phone Number= 0189532, Email Address= khk@gmail.com, Password= khk2019, Address= null String Show message box with error message YES

frmuser/

btnRegister 10/6/19 User ID=U\_00001, User Name=Kaung Htet Kyaw, Age=17, Phone Number=0189532, Email Address= khk@gmail.com, Password= khk2019, Address = Yangon String Show message box to tell that the data is successfully registered with information message YES

Frmuser/

btnClose 10/6/19 Closing the form - Close the form The form is closed when the “Close” button has been clicked YES

public partial class frmLogin : Form

{

dsTextEditiorTableAdapters.UsersTableAdapter udts = new dsTextEditiorTableAdapters.UsersTableAdapter();

public static string uid,uname;

public frmLogin()

{

InitializeComponent();

}

private void btnregister\_Click(object sender, EventArgs e)

{

frmUser fu = new frmUser();

fu.ShowDialog();

}

private void btnLogin\_Click(object sender, EventArgs e)

{

DataTable dt = new DataTable();

if (txtUserID.Text=="")

{

MessageBox.Show("Please Enter your User's ID", " User Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);

txtUserID.Focus();

}

else if (txtUsername.Text=="")

{

MessageBox.Show("Please enter your name", "User Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);

txtUsername.Focus();

}

else if (txtEmail.Text == "")

{

MessageBox.Show("Please enter your Email", "User Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);

txtEmail.Focus();

}

else if (txtPassword.Text == "")

{

MessageBox.Show("Please enter your password", "User Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);

txtPassword.Focus();

}

else

{

dt= udts.CheckUserLogin(txtEmail.Text, txtPassword.Text);

if (dt.Rows.Count > 0)

{

uid=dt.Rows[0][0].ToString();

uname = dt.Rows[0][1].ToString();

MessageBox.Show("User logged in successfully", "User login Form", MessageBoxButtons.OK, MessageBoxIcon.Information);

frmSimpleTextEditior fste = new frmSimpleTextEditior();

fste.Show();

this.Hide();

}

else

{

MessageBox.Show("Invalid User Login", "User Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

Test Test Date Test Data Type of Data Expected Result Actual Result Meet excepted Result

FrmLogin/

txtUserID 10/10/2019 UserID=null String Show Message box with error message YES

FrmLogin/

txtUsername 10/10/2019 UserID=U\_00001, User Name=null String Show Message box with error message YES

FrmLogin/

txtEmail 10/10/2019 UserID=U\_00001, User Name=Kaung Htet Kyaw, Email=null string Show Message box with error message YES

FrmLogin/

txtPassword 10/10/2019 UserID=U\_00001, User Name=Kaung Htet Kyaw, Email=khk@gmail,com, Password=null String Show Message box with error message YES

FrmLogin/

btnLogin 10/10/2019 UserID=U\_00001, User Name=Kaung Htet Kyaw, Email=khk@gmail,com, Password= khk2019 String Show message box to tell that the data is successfully registered with information message YES

FrmLogin/btnClose 10/10/2019 Closing the form - Closing the form The form is closed when the “Close” button has been clicked YES

Before the exception was handled, when the save file has not been written or chose the file path, it stops function and appears the error message that the exception of about Arguments exception was not handled.

After being handled the exception, it shows that the user do not have to leave the path with empty and choose at least a path as shown in the figure.

Before the exception was handled, when the document has been closed and opened with the new documents, it stops function and appears the error message that the exception of about Arguments being out of range was not handled.

After the exception was being handled, when the save file have not chosen, it shows the error message that null pages are not allowed as shown in the figure.

According to the diagram, there are four class diagrams. They are Users, Files, FileDetailInformation, and FindAndReplace. First, the “User” class have six attributes named UserID, Username, Email, Password, Address, and Age. It’s primary key is UserID. Its operations are the Fill and GetData which the data is filled in the table and supported for the dataset when it is extracted and shown the data the form is loaded, CheckUserLogin which checks the user’s data especially in email and password, and RegisterUserData which registers the user’s information account with the user’s User ID, User Name, Email, Password, Address, Age and Phone number.

The Second class is the “Files” class. It has four attributes named FileNumber, UserID, FileName, and FileLocation. It’s primary key is FileNumber. The operation is that the data is filled in the table and supported for the dataset when it is extracted and shown the data the form is loaded.

The third class is the “FileDetailInformation” class. It has three attributes named FileNo, FindNo, and EditNo. It’s foreign keys are the FIleNo and FindNo as it is existed as the dummy table between the two many-to-many relationship classes.Its operation is that that the data is filled in the table and supported for the dataset when it is extracted and shown the data the form is loaded.

The fourth class is the “FindAndReplace” class. It has four attributes named FindNo, ReplaceNo, FindWordName, and ReplaceWordName. It’s primary key is the FindNo. The operation is that the data is filled in the table and supported for the dataset when it is extracted and shown the data the form is loaded.

In the form, one user can use many files but they can be used by only one users. Therefore, there is a one-to-many relationship between the “User” and “Files” class. Then, When the files are used, they have many words to find and replace and the words are many words are existing through many files. Thus, the many-to-many relationship is existed between them. As they have many-to-many relationship, the dummy table named “FileDetailInformation” is appeared between them in order to prevent the duplicate of data.

Class Name: Users

Attributes : UserID, Username, Email, Password, Address, Age

Operation : Fill,GetData(),CheckUserLogin (@email, @password),RegisterUserData((@userid,@username,@email,@password,@address,@age,@phone)

Description : “Users” is used to record the User’s ID, name, Email Address, Password, Address, and Age.

Class Name: Files

Attributes : FileNumber, UserID, FileName, FileLocation

Operation : Fill,GetData()

Description : “Files” is used for saving and recording the document files.

Class Name: FileDetailInformation

Attributes : FileNo, FindNo, and EditNo

Operation : Fill,GetData()

Description : “FileDetailInformation” is used to record the file’s number, the find word’s number and the file edit number.

Class Name: FindAndReplace

Attributes : FindNo, ReplaceNo, FindWordName, and ReplaceWordName

Operation : Fill,GetData()

Description : “FindAndReplace” is used when recording the find word’s number, the word name that wants to find and the word name that wants to replace.