1. **Create a Windows application having two forms.** The first form should allow an administrator to login by accepting user name and password. The password should not be readable. The form should check if the user name is "Admin" as well as the password is "Admin" and only then allow to proceed further. If username and password is not matching, appropriate error message should be displayed. The form can have a user interface as shown in Figure 4.1

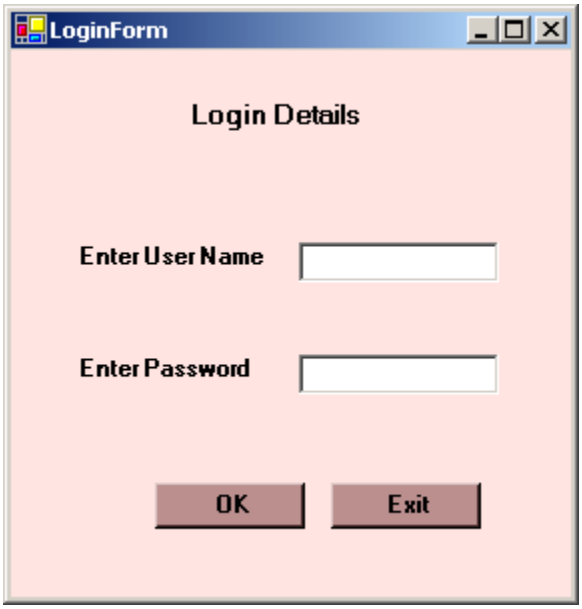

Figure 4.1: User interface for Login

On successful login, a new form should be displayed. The new form can have a user interface similar to Figure 4.2. Change the foreground color, background color, Font type of the controls as per your choice.

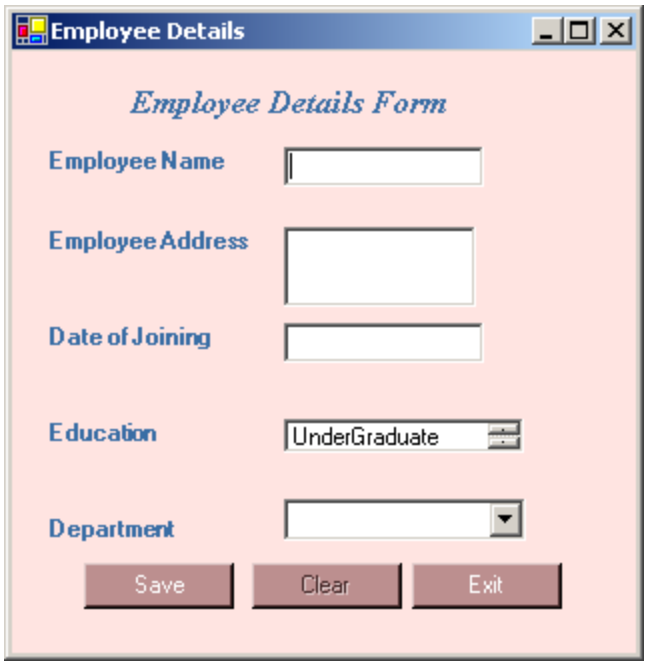

Figure 4.2: User Interface of Employee Details

This new form should allow user to enter Employee Name, Employee Address and Date of Joining. It should also allow the user to select the Education from the list provided and allow either selecting the department from a list of departments or entering a new department name. The list of departments should be in sorted order. Further features of this form are:

a. The Employee address should be more than 1 line
b. The Save button should confirm the details entered by the user.
c. The Clear button should clear the contents of the controls. Initially the Clear button should be disabled. It should be enabled after details have been confirmed through the Save button.
d. The Exit button should allow the user to exit from the application.

Hints:

a. To display the Employee Details form from the Login form, create an object of Employee Details form and invoke its "Show" method.
b. Set MultiLine property of textbox for address.
c. Set PasswordChar property of the Password textbox for the login form to appropriate value
d. Use the ListBox control to accept the Education.
e. Use the ComboBox control to accept the Department. Set the Sorted property of the ComboBox.

Solution:

1. **Create a Windows application named Login Form. Name the form as LoginForm. Add controls to it to create user interface as shown in the Lab Guide 1.**

2. **Set the `PasswordChar` property of Password `TextBox` to *.**

3. **Write codes for the `Click` event of the buttons OK and Exit.**

```
   private void btnOK_Click(object sender, System.EventArgs e)
   {
      if (this.txtUsrName.Text.Equals("Admin"))
      {
         if (this.txtPassword.Text.Equals("Admin"))
         {
            frmEmployee objEmp = new frmEmployee ();
            objEmp.Show();
            this.Hide();
         }
         else
            MessageBox.Show ("Password is invalid");
      }
      else
         MessageBox.Show ("User Name is invalid");
   }

   private void btnExit_Click(object sender, System.EventArgs e)
   {
      Application.Exit();
   }
```

4. Add a new form. Name the form as EmployeeDetails.cs. Add controls to it to create user interface as shown in the Lab Guide 1.

5. Set the `MultiLine` property of employee address `TextBox` to true.

6. Set the `Sorted` property of `ComboBox` to true.

7. Declare two variables to store the Education and Department information.

```
   private string edu, dept;
```

8. Add the following code to `Load` event of the Form

```
   private void EmployeeDetails_Load(object sender, System.EventArgs e)
   {
      btnClear.Enabled = false;
   }
```

3. Add the following code to the `Click` event of the Save button

```
   private void btnSave_Click(object sender, System.EventArgs e)
   {
      MessageBox.Show("Employee Name is " +txtName.Text);
      MessageBox.Show("Employee Addr is " + txtAddr.Text);
      MessageBox.Show(" Joined on " + txtDOJ.Text + " has education " +  edu
+ " and belongs to dept " + dept);
      btnClear.Enabled= true;
   }
```

4. Add the following code to the `SelectedIndexChanged` event of the Education `List Box`

```
   private void lstEdu_SelectedIndexChanged(object sender,
System.EventArgs e)
   {
      edu = lstEdu.Text;
   }
```

5. Add the following code to the `SelectedIndexChanged` event of the Department Combo box

```
   private void cboDept_SelectedIndexChanged(object sender,
System.EventArgs e)
   {
      dept = cboDept.Text;
   }
```

6. Add the following code to the `Click` event of the Clear button and the Exit button.

```
   private void btnClear_Click(object sender, System.EventArgs e)
   {
      txtName.Text = "";
      txtAddr.Text = "";
      txtDOJ.Text = "";
      cboDept.Text = "";
      lstEdu.ClearSelected();
      btnClear.Enabled= false;
   }

   private void btnExit_Click(object sender, System.EventArgs e)
      {
         this.Close();
      }
```

7. Save, build and execute the application.

2. **Create a Windows application, which will allow the user to verify the details entered for reserving a Flight ticket. The application should have the following features:**

   a. Allows the user to enter the ticket number, name of the passenger, passport number and flight date.
   b. Allow the user to select the source, destination and the class.
   c. Allows the user to check the respective service / services from the list provided
   d. The Verify button should display an error message if the source and the destination are the same.
   e. The Clear button should clear the contents of the controls.

   The Form can have the following user interface:

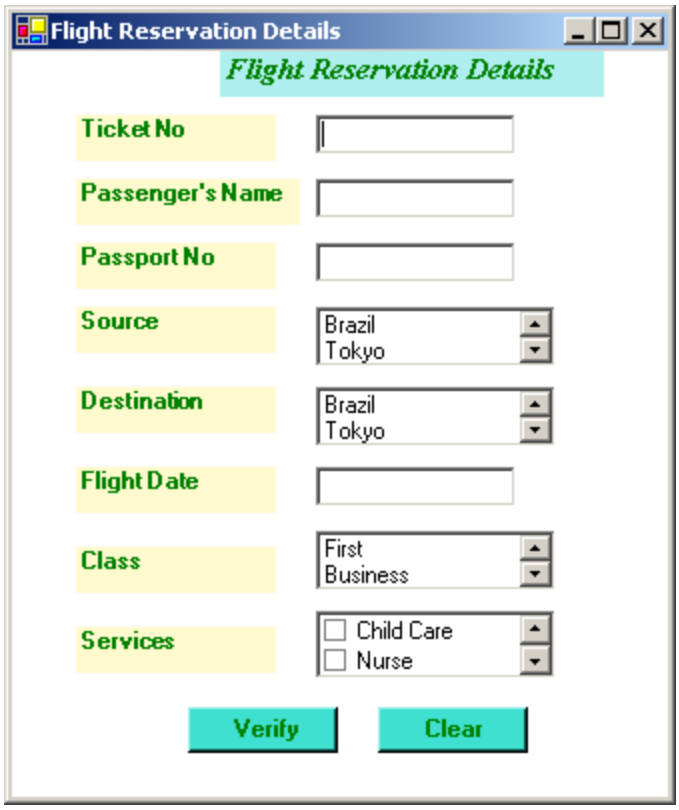

Figure 4.3: User Interface

Change the foreground color, background color, Font type of the controls as per your choice.

Solution:

Create a Windows application named Flight. Add controls to it to create user interface as shown in the Lab Guide 1.

Declare two variables to store the Source and Destination information.

```
    private string source, destn;
```

Add code for the `SelectedIndexChanged` **event of the lstSource control**

```
   private void lstSource_SelectedIndexChanged(object sender,
System.EventArgs e)
   {
      source = lstSource.Text;
   }
```

3. Add code for the `SelectedIndexChanged` **event of the lstDestination control**

```
   private void lstDestination_SelectedIndexChanged(object sender,
System.EventArgs e)
   {
```

```
      destn = lstDestination.Text;
   }
```

4. Add code for the `Click` event of each button respectively

```
   private void btnVerify_Click(object sender, System.EventArgs e)
   {
      if (source == dest)
      {
         MessageBox.Show("Source and Destination cannot be the
   same");
         lstSource.Focus();
      }
      else
         MessageBox.Show("Data Verified");
   }

   private void btnClear_Click(object sender, System.EventArgs e)
   {
      txtTicketNo.Text = "";
      txtName.Text = "";
      txtPassportNo.Text = "";
      lstSource.ClearSelected();
      lstDestination.ClearSelected();
      txtFlightDate.Text="";
      lstClass.ClearSelected();
      chklstServices.Items.Clear();
      this.chklstServices.Items.AddRange(new object[]
      {
          "Child Care",
          "Nurse",
          "Wheel Chair"
      });
   }
```

3. **Create a Windows application, which will allow the user to do some arithmetic calculations on two given numbers such as addition, subtraction, multiplication and division.** The application should have two forms. The first should be an MDI form with the following menu

Calculator
Start

The **Start** menu item should open the second form. The second form should have the following menu:

Arithmetic Calculations	**Exit**
Add	Quit
Subtract	
Divide	
Multiply	

The second form should accept two numbers from the user. On clicking any of the options, arithmetic operations based on the option selected should be performed and the result should be displayed in the third textbox. The user must not be allowed to type into the third textbox. If the user clicks the menu options without entering data in the first two textboxes appropriate error message should be displayed. Clicking Quit on the Exit menu will terminate the application.

The second form can have the following interface:

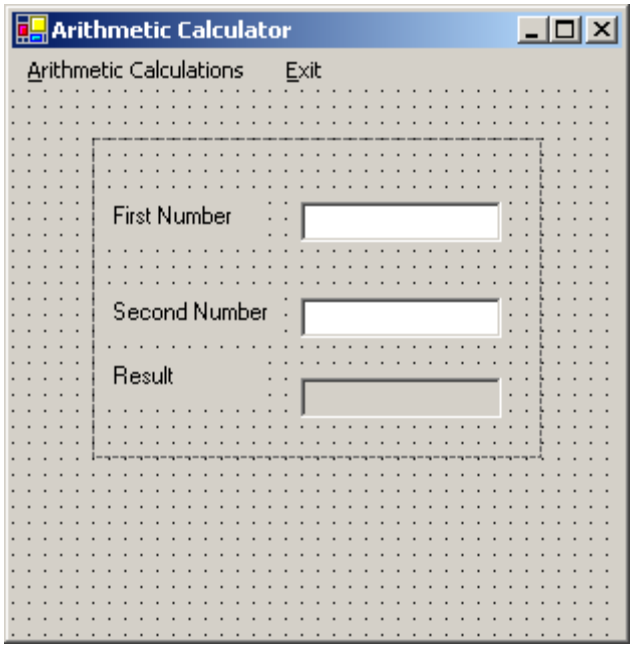

Figure 4.4: User Interface for Arithmetic Calculator

Hints:

- Set IsMDIContainer to true for the first form
- Set the Readonly Property to true for third textbox on second form
- Use validation function to validate the data in the textboxes
- Use Convert.ToInt32() function to perform text to integer conversion for arithmetic operations.

Solution:

1. **Create a blank Windows solution named LD4.**

2. **Add a project to it named Calculator. Rename Form1.cs to `frmMDI.cs`. Change the `Text` property of the form to '`MDI Form`'. Set its `IsMdiContainer` property to true. Add a `MainMenu` control to it and submenu items. Name the controls appropriately.**

3. **Add a new form and rename it to ArithCalc.cs. Create a user interface for this form as shown in the lab guide. Add a `MainMenu` control to it and sub menu items as shown in the lab guide. Change the class name in the code to `frmArithCalc`.**

4. **Change the properties of the various controls added as per the Lab Guide using Table 8.1 given here**

Control	Property	Value
TextBox	Name Text	txtFirstNo
TextBox	Name Text	txtSecondNo
TextBox	Name Text ReadOnly	txtResult True
Label	Name Text	lblFirstNo First Number
Label	Name Text	lblFirstNo Second Number
Label	Name Text	lblResult Result
Panel	Name	pnlCalc
MainMenu	Name	mnuArithCalc
MenuItem	Name Text	mnuItemArith Arithmetic Calculations
MenuItem	Name Text	mnuItemExit Exit
MenuItem	Name Text	mnuItemAdd Add
MenuItem	Name Text	mnuItemSub Subtract
MenuItem	Name Text	mnuItemMul Multiply
MenuItem	Name Text	mnuItemDiv Divide
MenuItem	Name Text	mnuItemQuit Quit

Table 4.1: Properties

5. Add code to the `Click` event of the `MenuItem` Start in frmMDI as follows:

```
private void mnuItemStart_Click(object sender, System.EventArgs e)
{
   frmArithCalc frm = new frmArithCalc();
   frm.MdiParent = this;
   frm.Show();
}
```

6. Next add code to the ArithCalc.cs for the `Click` events of the various menu items

```
   private void mnuItemAdd_Click(object sender, System.EventArgs e)
   {
   if ((validateText(txtFirstNo)) && (validateText(txtSecondNo)))
```

```
        {
           int result = Convert.ToInt32 (txtFirstNo.Text)+
Convert.ToInt32 (txtSecondNo.Text);
                 txtResult.Text  = result.ToString ();
        }
        }

     private void mnuItemSub_Click(object sender, System.EventArgs e)
     {

        if ((validateText(txtFirstNo)) && (validateText(txtSecondNo)))
        {
           int result = Convert.ToInt32 (txtFirstNo.Text) -
Convert.ToInt32 (txtSecondNo.Text);
           txtResult.Text  = result.ToString ();
        }

     }

     private void mnuItemMul_Click(object sender, System.EventArgs e)
     {

        if ((validateText(txtFirstNo)) && (validateText(txtSecondNo)))
        {
           int result = Convert.ToInt32 (txtFirstNo.Text)*
Convert.ToInt32 (txtSecondNo.Text);
           txtResult.Text  = result.ToString ();
        }

     }

     private void mnuItemDiv_Click(object sender, System.EventArgs e)
     {

        if ((validateText(txtFirstNo)) && (validateText(txtSecondNo)))
        {
           decimal result = Convert.ToInt32 (txtFirstNo.Text)/
Convert.ToInt32 (txtSecondNo.Text);
           txtResult.Text  = result.ToString ();
        }

     }

     private void mnuItemQuit_Click(object sender, System.EventArgs e)
     {
     Application.Exit ();
     }
```

7. Add code to perform validation of the `TextBox` **controls as follows:**

```
     private bool validateText(object txt)
```

```
      {
         TextBox txtbox=(TextBox) txt;
         if ((txtbox.Text.Length==0) || (txtbox.Text.Trim()==""))
         {
            MessageBox.Show ("Field cannot be empty. Please enter some
value.");
            txtbox.Focus();
            return false;
         }
         else
            return true;
      }
```

4. **Create a Windows application which simulates a text file editor.** It should allow a user to create new text files, save and open text files. It should also have options to set background color as well as close. Close should not close the application but only close the open document. The interface can look as shown in Figure 4.5. On clicking Set Color a new menu option Color should be added at runtime between File and Exit menu options. This menu option when clicked should display options to choose desired color. On clicking right mouse button a popup menu should be displayed in the child window which allows to set color or close the document.

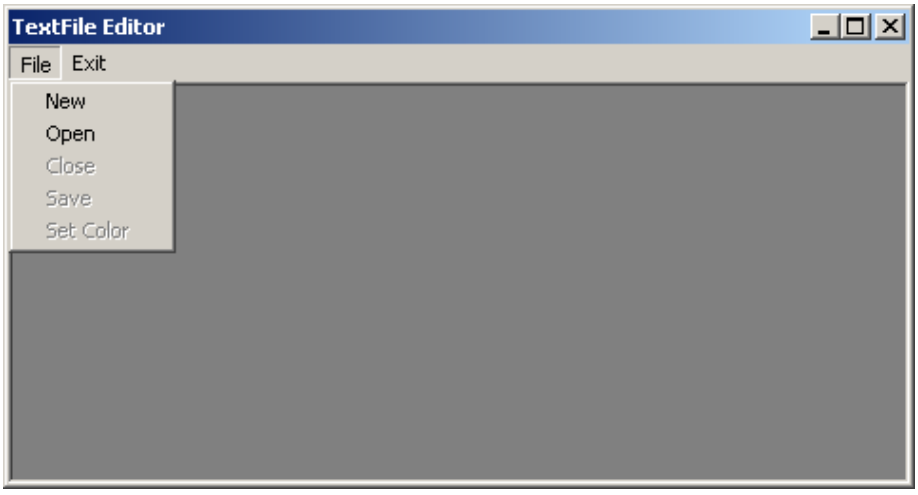

Figure 4.5: Text File Editor Application

The second form should look like Figure 4.6:

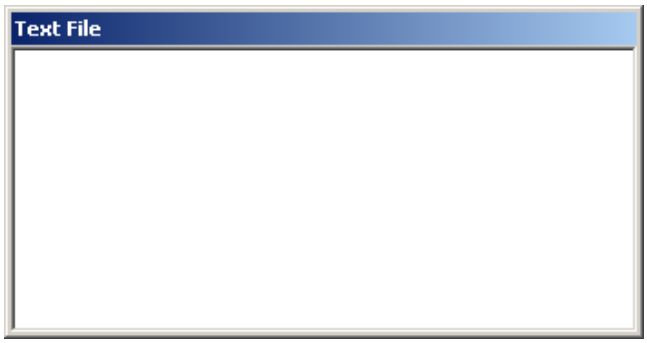

Figure 4.6: Text File Window

Hints:

- Make use of MenuItem to create new items at runtime.
- Make use of color dialog, open dialog and save dialog.
- Use a RichTextBox control to create the interface for the second form.
- Make use of ContextMenu to create the popup menu.

1. **Create a project named TextFileEditor in the existing solution. Rename Form1.cs to `frmMDI.cs`. Change the `Text` property of the form to '`TextFile Editor`'. Set its `IsMdiContainer` property to true. Add a `MainMenu` control to it and submenu items to create a form interface as shown in the lab guide. Name the controls appropriately.**

2. **Add a new form and rename it to TextFile.cs. Change the `Text` property of the form to 'Text File'. Create a user interface for this form as shown in the lab guide by adding a RichTextBox control and setting its `Dock` property to Fill.**

3. **Add a SaveFileDialog control and an OpenFileDialog control to MDIForm.**

4. **Add the following code to the `Click` events of the various menu items on MDIForm**

```
    private void mnuFileNew_Click(object sender, System.EventArgs e)
    {
       fTxtFile = new frmTextFile();
       fTxtFile.MdiParent= this;
       fTxtFile.Text = "Text File " + this.MdiChildren.Length;
       fTxtFile.Show();


       if(this.MdiChildren.Length >=1)
       {
          cmnuText = new ContextMenu();
cmnuText.MenuItems.Add("BackColor",new
EventHandler(mnuItmColor_Click));
          cmnuText.MenuItems.Add("Close",new
EventHandler(mnuFileClose_Click));
          this.ActiveMdiChild.ActiveControl.ContextMenu = cmnuText;
          arrangeMenuItems();
```

```
         }
      }

      private void mnuFileOpen_Click(object sender, System.EventArgs e)
      {
         StreamReader sr;
         ofdlgOpen.Filter= "Text Files|*.txt";
         DialogResult dr=ofdlgOpen.ShowDialog();

         if (dr.Equals (DialogResult.OK))
         {
            fTxtFile = new frmTextFile();
            fTxtFile.MdiParent= this;
            fTxtFile.Text = "Text File " + this.MdiChildren.Length;
            fTxtFile.Show();

            if (this.ActiveMdiChild.Name == "frmTextFile")
            {
               sr = new StreamReader(this.ofdlgOpen.FileName);
               this.ActiveMdiChild.ActiveControl.Text = sr.ReadToEnd();
               sr.Close();

            }
            cmnuText = new ContextMenu();
            cmnuText.MenuItems.Add("BackColor",new
EventHandler(mnuItmColor_Click));
            cmnuText.MenuItems.Add("Close",new
EventHandler(mnuFileClose_Click));
            this.ActiveMdiChild.ActiveControl.ContextMenu = cmnuText;
            arrangeMenuItems();

         }
      }

      private void mnuFileSave_Click(object sender, System.EventArgs e)
      {
         if(this.MdiChildren.Length > 0)
         {
            // for file save
            this.sdlgSave.Filter = "Text Files|*.txt";

            DialogResult dr=this.sdlgSave.ShowDialog();

            if (dr.Equals(DialogResult.OK))
               if (this.ActiveMdiChild.Name == "frmTextFile")
               {

                  StreamWriter sw = new
StreamWriter(this.sdlgSave.FileName);
                  sw.Write(this.ActiveMdiChild.ActiveControl.Text);
                  sw.Close();
```

```
            }
        }

    }

    private void arrangeMenuItems()
    {
        if(this.MdiChildren.Length == 0 )
        {
            this.mnuMain.MenuItems[0].MenuItems[2].Enabled = false;
            this.mnuMain.MenuItems[0].MenuItems[4].Enabled = false;
            this.mnuMain.MenuItems[0].MenuItems[3].Enabled = false;
            status = true;

            if(this.mnuMain.MenuItems[1].Text == "Set Color")
            {
                this.mnuMain.MenuItems.Remove(mnuItmColor);

            }

        }
        else
        {
            this.mnuMain.MenuItems[0].MenuItems[2].Enabled = true;
            this.mnuMain.MenuItems[0].MenuItems[4].Enabled = true;
            this.mnuMain.MenuItems[0].MenuItems[3].Enabled = true;
            if(this.mnuMain.MenuItems[1].Text != "BackColor")
            {
                this.mnuMain.MenuItems[0].MenuItems[4].Enabled = true;
            }
        }
    }

    private void mnuFileClose_Click(object sender, System.EventArgs
e)
    {
        if(this.MdiChildren.Length > 0 )
        {
            this.ActiveMdiChild.Close();
            arrangeMenuItems();
        }
    }

    private void mnuFileExit_Click(object sender, System.EventArgs e)
    {
        Application.Exit();
    }

    private void mnuColor_Click(object sender, System.EventArgs e)
    {
        // Creating the color menu at run time
```

```
         if(status)
         {
            status=false;
            mnuItmColor= new MenuItem("BackColor",new
EventHandler(mnuItmColor_Click));
            this.mnuMain.MenuItems.Add(1,mnuItmColor);
            this.mnuMain.MenuItems[0].MenuItems[4].Enabled = false;
         }
      }

      public void mnuItmColor_Click(object sender, System.EventArgs e)
      {
         if(this.MdiChildren.Length>0)
         {
            nColor = new ColorDialog();
            nColor.ShowDialog();
            this.ActiveMdiChild.ActiveControl.BackColor = nColor.Color;
         }
      }
      private void mnuItemQuit_Click(object sender, System.EventArgs e)
      {
         Application.Exit();
      }
```

5. Build and execute the application.