It is my great pleasure to present this laboratory manual for S.Y (M.C.A) students for the subject of VB.NET (Visual Basic .NET). Subject name itself is expecting what are the advance tools that might be Visual Studio to develop and build rich .NET applications. In this manual we have dealt with VB.NET.

As a student, many of you may be wondering with some of the questions in your mind regarding the subject and exactly what has been tried is to answer through this manual.

As you may be aware that MGM has already been awarded with ISO 9000 certification and it is our endure to technically equip our students taking the advantage of the procedural aspects of ISO 9000 Certification.

Faculty members are also advised that covering these aspects in initial stage itself, will greatly relived them in future as much of the load will be taken care by the enthusiasm energies of the students once they are conceptually clear.

Dr. S.D. DESHMUKH
Principal
LABORATORY MANUAL CONTENTS

This manual is intended for the students S.Y (M.C.A) engineering for the subject of VB.NET. Subject name itself is expecting what are the advance tools that might be Visual Studio to develop and build rich .NET applications. In the subject of this manual typically contains practical/Lab Sessions we have dealt with VB.NET.

Students are advised to thoroughly go through this manual rather than only topics mentioned are the key to understanding and conceptual visualization of theoretical aspects covered in the books.

Good Luck for your Enjoyable Laboratory Sessions

Lect. V.V Shaga
SUBJECT INDEX:

1. Create and Validate Login Form.
2. Program to design an ‘ACCOUNT’ Class.
3. Program to demonstrate Inheritance, Polymorphism and Interfaces.
4. Advance Controls.
5. Common Dialog Controls.
6. ADO.NET Code to show records in DataGridView Control.
7. ADO.NET Code to perform Insert, Delete, Update and Select operations.
8. Crystal Reports
9. Web Application using ASP.NET that uses validation controls.
10. Web Application with ADO.NET to perform Insert, Delete, Update and Select Operations.

DOs and DON’T DOs in Laboratory:

1. Do not handle any equipment before reading the instructions/Instruction manuals

2. Read carefully the power ratings of the equipment before it is switched on whether ratings 230 V/50 Hz or 115V/60 Hz. For Indian equipments, the power ratings are normally 230V/50Hz. If you have equipment with 115/60 Hz ratings, do not insert power plug, as our normal supply is 230V/50 Hz, which will damage the equipment.

3. Observe type of sockets of equipment power to avoid mechanical damage

4. Do not forcefully place connectors to avoid the damage

5. Strictly observe the instructions given by the teacher/Lab Instructor

Instruction for Laboratory Teachers::

1. Submission related to whatever lab work has been completed should be done during the next lab session. The immediate arrangements for printouts related to submission on the day of practical assignments.

2. Students should be taught for taking the printouts under the observation of lab teacher.

3. The promptness of submission should be encouraged by way of marking and evaluation patterns that will benefit the sincere students.
**WARMUP EXERCISES:**

What is .NET?

Explain execution of .any .NET application.

What is difference between VB.NET and VB?

Explain .NET Framework?

What are CLR, CTS, and CLS?

What is Console?

What are Inheritance, Polymorphism and Interfaces?

Features of .NET

What is assembly? Explain different types of assemblies.

What is namespace?

What is ADO.NET? Explain ADO.NET architecture.

Explain ADO.NET objects.

What is Dataset? Difference between Dataset and DataReader object.

What are the methods of Command object?

What is connected and disconnected architecture?

What is stream? What is the use of StreamReader and StreamWriter?

What is Crystal report?

What is ASP.NET?

Compare .NET with other technologies.
1. Lab Exercises:

[Purpose of this exercise is to create and validate login form]

Steps to follow:
1) In FILE menu select “NEW PROJECT”.
2) Give the name for Project “LoginApplication” and click “OK”. Window with Form1.vb gets generated.
3) Select two textboxes, labels and buttons from Toolbox.
4) Set the properties of controls as below :

<table>
<thead>
<tr>
<th>Sr no</th>
<th>Controls</th>
<th>Properties</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TextBox1</td>
<td>Name</td>
<td>txtUsername</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MaxLength</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>TextBox2</td>
<td>Name</td>
<td>txtPassword</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MaxLength</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PasswordChar</td>
<td>*</td>
</tr>
<tr>
<td>3</td>
<td>Button1</td>
<td>Name</td>
<td>btnLogin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Login</td>
</tr>
<tr>
<td>4</td>
<td>Button2</td>
<td>Name</td>
<td>btnCancel</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Cancel</td>
</tr>
<tr>
<td>5</td>
<td>Form1</td>
<td>Name</td>
<td>frmLogin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Login Form</td>
</tr>
<tr>
<td>6</td>
<td>Label1</td>
<td>Name</td>
<td>lblUsername</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Username</td>
</tr>
<tr>
<td>7</td>
<td>Label2</td>
<td>Name</td>
<td>lblPassword</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Password</td>
</tr>
</tbody>
</table>

5) Double Click on Login Button and write code in view source.
6) Double Click on Cancel button and write code.
7) Build the Project and Execute by Start Debugging or by pressing F5.

Exercise No1: (2 Hours) - 1 Practical (code)

Public Class frmLogin
    Private Sub btnLogin_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnLogin.Click
        'Check for Valid Username or Password
        Select Case Trim(txtUsername.Text)
            Case "" Then
                MsgBox("Please Enter Username or Password")
                txtUsername.Focus()
            Case "vikrant" Then
                If Trim(txtPassword.Text) = "shaga" Then
                    MsgBox("Login Success..")
                    txtUsername.Focus()
                End If
            Case "MCA" Then
                If Trim(txtPassword.Text) = "MCA" Then
                    MsgBox("Login Success..")
                End If
        End Select
    End Sub
End Class
txtUsername.Focus()
End If
Case Else
MsgBox("Login fails..")
txtUsername.Text = ""
txtPassword.Text = ""
txtUsername.Focus()
End Select
End Sub

Private Sub btnCancel_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnCancel.Click
End
End Sub
End Class
2. Lab Exercises:

[Purpose of this exercise is to define a Class ‘ACCOUNT’. Include following
Data members: Name of depositor, Account no, type of Account, balance amount.
Member Functions: To Deposit an amount, to withdraw an amount after checking balance, to
show balance.
   Also provide proper validations wherever necessary. Write a main program to test above
class]

Steps to follow:
1) In FILE menu select “NEW PROJECT”. Then select “Console Application”.
2) Give the name for Project “AccountInformation” and click “OK”. Window with
   Module1.vb gets generated.
3) From Project Menu select “Add Class”. Give name to the class i.e. “ACCOUNT”.
4) Write the code and execute by pressing F5.

Exercise No2: (2 Hours) - 1 Practical
Public Class Account
   Private name_Of_Customer, accountNo, accountType As String
   Private balance_Amount As Double
   Public Sub New() 'Default constructor
      Console.WriteLine("Customers Account Information")
      End Sub
   'Parameterized constructor
   Public Sub New(ByVal name As String, ByVal acNo As String, ByVal acType As String, ByVal bal As Double)
      name_Of_Customer = name
      accountNo = acNo
      accountType = acType
      balance_Amount = bal
      End Sub
   'Deposite method
   Public Sub deposite(ByVal amt As Double)
      balance_Amount += amt
      End Sub
   'Withdraw method
   Public Sub withdraw(ByVal amt As Double)
      Select Case accountType
         Case "S"
            If (balance_Amount > amt) Then
               balance_Amount -= amt
            Else
               Console.WriteLine("Insufficient balance")
            End If
         Case "C"
            If (balance_Amount > 500) Then
               balance_Amount -= amt
      End Select
Else
    Console.WriteLine("Insufficient balance")
End If
Case Else
    Console.WriteLine("Invalid transaction")
End Select
End Sub
' Displays Customers information
Public Sub showDetails()
    Console.WriteLine(" Name : " & name_Of_Customer)
    Console.WriteLine(" Account No : " & accountNo)
    Console.WriteLine(" Account type : " & accountType)
    Console.WriteLine(" Balance Amount : " & balance_Amount)
End Sub
End Class

Module Module1
    Sub Main()
        Dim O1 As New Account ' Calling Default constructor
        Dim A1 As New Account("Harry Potter", "A101", "C", 1200.33) ' Calling parameterized constructors
        A1.showDetails()
        A1.deposite(200.34)
        A1.showDetails()
        A1.withdraw(700)
        A1.showDetails()
        Console.Read()
    End Sub
End Module
3. Lab Exercises:

[Purpose of this exercise is to implement Inheritance, Polymorphism and Interfaces.]

Steps to follow:
1) In FILE menu select “NEW PROJECT”. Then select “Console Application”.
2) Give the name for Project “EmployeeInformation” and click “OK”. Window with Module1.vb gets generated.
3) From Project Menu select “Add Class”. Give name to the class i.e. “Employee” and define its functions. This class will be BASE Class.
4) From Project Menu again select “Add Class”. Give name to the class i.e. “Manager”. This class should be derived class.
5) From Project Menu again select “Add Class”. Give name to the class i.e. “Engineer”. This class also should be a derived class.
6) Then Right Click on “EmployeeInformation” in solution explorer window and select “ADD”, then select “New Item”. Then select Interface and give name as “ITaxInterface”. Declare the methods in interface and implement interface methods in derived classes.
7) Write the code and execute by pressing F5.

Exercise No3: (2 Hours) – 1 Practical

[Base Class]
Public Class Employee
    Public empCode As String
    Public empName As String
    Public basic, netSalary As Double
    Public hra, da As Double

    Public Sub getEmpDetails()
        Console.WriteLine("Enter Code :")
        empCode = Console.ReadLine()
        Console.WriteLine("Enter Name :")
        empName = Console.ReadLine()
    End Sub

    Public Sub getSalaryDetails()
        Console.WriteLine("Enter Basic :")
        basic = Console.ReadLine()
        Console.WriteLine("Enter HRA :")
        hra = Console.ReadLine()
        Console.WriteLine("Enter DA:")
        da = Console.ReadLine()
    End Sub

    *Called by Manager Class
    Public Function calculateSalary() As Double
        Return basic + hra + da
    End Function
'Called by Engineer class
    Public Function calculateSalary(ByVal bonusAmt As Double) As Double
        Return basic + hra + da + bonusAmt
    End Function
End Class

[Interface]
Public Interface ITaxInterface
    Function calculateITax() As Double
End Interface

[Derived Class 1]
Public Class Manager
    Inherits Employee
    Implements ITaxInterface

    Public Function calculateITax() As Double Implements ITaxInterface.calculateITax
        Return (basic * 0.05)
    End Function
End Class

[Derived Class 2]
Public Class Engineer
    Inherits Employee
    Implements ITaxInterface

    Public Function calculateITax() As Double Implements ITaxInterface.calculateITax
        Return (basic * 0.02)
    End Function
End Class

[Main Program]
Module Module1
    Sub Main()
        Dim ch As Integer
        Console.WriteLine("1:Manager")
        Console.WriteLine("2:Engineer")
        Console.WriteLine("0:Exit")
        Console.WriteLine("Enter your choice:")
        ch = Convert.ToInt32(Console.ReadLine())
        Do While ch <> 0
            Select Case ch
                Case 1
                    Dim m As New Manager ' Creating an instance of Derived class
m.getEmpDetails() ' Calling members of Base class Employee
m.getSalaryDetails()
Console.WriteLine("Net Salary :" & m.calculateSalary()) ' Implementing Polymorphism
Console.WriteLine("Income Tax :" & m.calculateITax()) ' Calling Interface method

Case 2
Dim e As New Engineer ' Creating an instance of Derived class
e.getEmpDetails() ' Calling members of Base class Employee
e.getSalaryDetails()
Console.WriteLine("Net Salary :" & e.calculateSalary(4500)) ' Implementing Polymorphism
Console.WriteLine("Income Tax :" & e.calculateITax()) ' Calling Interface method
End Select
Console.WriteLine("1:Manager")
Console.WriteLine("2:Engineer")
Console.WriteLine("0:Exit")
Console.WriteLine("Enter your choice:")
ch = Convert.ToInt32(Console.ReadLine())
Loop
End Sub
End Module
4. Lab Exercises:

[Purpose of this exercise is to use advance controls]
Steps to follow:
1) In File menu select “NEW PROJECT”.
2) Give the name for Project “AdvanceControls” and click “OK”. Window with Form1.vb gets generated. Add second form Form2.vb also.
3) Add the advance controls from toolbox.
4) Set the properties of controls as below:

<table>
<thead>
<tr>
<th>Sr no</th>
<th>Controls</th>
<th>Properties</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Form1</td>
<td>Name</td>
<td>frmAdvanceControls</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Advance Controls</td>
</tr>
<tr>
<td>2</td>
<td>Form2</td>
<td>Name</td>
<td>frmCricket</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Cricket Information</td>
</tr>
<tr>
<td>3</td>
<td>TabControl1</td>
<td>Name</td>
<td>TabControl1</td>
</tr>
<tr>
<td>4</td>
<td>StatusStrip1</td>
<td>Name</td>
<td>StatusStrip1</td>
</tr>
<tr>
<td>5</td>
<td>Timer1</td>
<td>Name</td>
<td>Timer1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Interval</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enabled</td>
<td>False</td>
</tr>
<tr>
<td>6</td>
<td>TreeView1</td>
<td>Name</td>
<td>TreeView1</td>
</tr>
<tr>
<td>7</td>
<td>ProgressBar1</td>
<td>Name</td>
<td>ProgressBar1</td>
</tr>
</tbody>
</table>

5) Write code:
- To display date and time in status bar
- To show the progress in progress bar
- To show the full path of parent and child nodes of tree view.
6) Build the Project and Execute by Start Debugging or by pressing F5.

Exercise No4: (2 Hours) – 1 Practical
[Form 1]

Public Class frmAdvanceControls
    Private Sub CricketToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles CricketToolStripMenuItem.Click
        frmCricket.Show()
    End Sub

    Private Sub frmAdvanceControls_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Timer1.Enabled = True
    End Sub

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
        TSSLblDateTime.Text = System.DateTime.Now 'displays date and time on status bar
    End Sub
End Class
[Form 2]

Public Class frmCricket
    Private Sub TreeView1_AfterSelect(ByVal sender As System.Object, ByVal e As System.Windows.Forms.TreeViewEventArgs) Handles TreeView1.AfterSelect
        txtPath.Text = TreeView1.SelectedNode.FullPath 'Shows the full path of node
    End Sub

    Private Sub btnShow_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnShow.Click
        ProgressBar1.Value = 1
        ProgressBar1.Maximum = 10000
        While ProgressBar1.Value <= ProgressBar1.Maximum - 1
            ProgressBar1.Value += 1
        End While
        Panel1.Hide()
    End Sub

    Private Sub TabPage1_Enter(ByVal sender As Object, ByVal e As System.EventArgs) Handles TabPage1.Enter
        ProgressBar1.Value = 0
        Panel1.Show()
    End Sub
End Class
5. Lab Exercises:

[Purpose of this exercise is to use common dialog controls]
Steps to follow:
1) In file menu select “NEW PROJECT”.
2) Give the name for Project “CommonControls” and click “OK”. Window with Form1.vb gets generated.
3) Add Menu strip and Rich Textbox Controls from toolbox.
4) Add the common dialog controls from toolbox.
5) Set the properties of controls as below:

<table>
<thead>
<tr>
<th>Sr no</th>
<th>Controls</th>
<th>Properties</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Form1</td>
<td>Name</td>
<td>frmCommonControls</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Common dialog Controls</td>
</tr>
<tr>
<td>2</td>
<td>RichTextBox1</td>
<td>Name</td>
<td>rtbContents</td>
</tr>
</tbody>
</table>

6) Build the Project and Execute by Start Debugging or by pressing F5.

Exercise No5: (2 Hours) - 1 Practical
Imports System.IO

Public Class frmCommonControls
    Inherits System.Windows.Forms.Form
    Private filename As String
    Dim sr As StreamReader

    Private Sub OpenToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles OpenToolStripMenuItem.Click
        Try
            With OpenFileDialog1
                .Filter = "Text files(*.txt)|*.txt|" & "All files|*.*"
                If .ShowDialog() = DialogResult.OK Then
                    filename = .FileName
                    sr = New StreamReader(.OpenFile)
                    rtbContents.Text = sr.ReadToEnd()
                End If
            End With
        End Try
        Catch ex As Exception
            MsgBox(ex.Message)
        Finally
            If Not (sr Is Nothing) Then
                sr.Close()
            End If
        End Try
    End Sub

    Private Sub SaveAsToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles SaveAsToolStripMenuItem.Click
        Dim sw As StreamWriter

        Private Sub SaveAsToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles SaveAsToolStripMenuItem.Click
            Dim sw As StreamWriter

            Try
                With OpenFileDialog1
                    .Filter = "Text files(*.txt)|*.txt|" & "All files|*.*"
                    If .ShowDialog() = DialogResult.OK Then
                        filename = .FileName
                        sr = New StreamReader(.OpenFile)
                        rtbContents.Text = sr.ReadToEnd()
                    End If
                End With
            End Try
            Catch ex As Exception
                MsgBox(ex.Message)
            Finally
                If Not (sr Is Nothing) Then
                    sr.Close()
                End If
            End Try
        End Sub
Try
    With SaveFileDialog1
        .FileName = filename
        .Filter = "Text files(*.txt)|*.txt|" & "All files|*.*"
        If .ShowDialog() = DialogResult.OK Then
            filename = .FileName
            sw = New StreamWriter(filename)
            sw.Write(rtbContents.Text)
        End If
    End With
Catch ex As Exception
    MsgBox(ex.Message)
Finally
    If Not (sw Is Nothing) Then
data
    sw.Close()
End Try
End Sub

Private Sub SelectFontToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles SelectFontToolStripMenuItem.Click
    Try
        With FontDialog1
            .Font = rtbContents.Font
            If .ShowDialog() = Windows.Forms.DialogResult.OK Then
                rtbContents.Font = .Font
            End If
        End With
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub

Private Sub SelectColorToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles SelectColorToolStripMenuItem.Click
    Static CustomColors() As Integer = {RGB(255, 0, 0), RGB(0, 255, 0), RGB(0, 0, 255)}
    'initializing CustomColors with an array of integers and putting Red, Green, 
    'and Blue in the custom colors section
    Try
        With ColorDialog1
            .Color = rtbContents.ForeColor
            'initializing the selected color to match the color currently used 
            'by the richtextbox's foreground color
            .CustomColors = CustomColors
            'filling custom colors on the dialog box with the array declared above
            If .ShowDialog() = DialogResult.OK Then
                rtbContents.ForeColor = .Color
                CustomColors = .CustomColors
            End If
            'Storing the custom colors to use again
End If
ColorDialog1.Reset()
' resetting all colors in the dialog box
End With
Catch es As Exception
MessageBox.Show(es.Message)
End Try
End Sub
End Class
6. Lab Exercises:

[Purpose of this exercise is show records from database to datagridview control]
Steps to follow:

1) In FILE menu select “NEW PROJECT”.
2) Give the name for Project “DataGridViewApplication” and click “OK”. Window with Form1.vb gets generated.
3) Add DataGridView and command button control from toolbox.
4) Set the properties of controls as below:

<table>
<thead>
<tr>
<th>Sr no</th>
<th>Controls</th>
<th>Properties</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Form1</td>
<td>Name</td>
<td>frmDataGrid Application</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>DataGrid Application</td>
</tr>
<tr>
<td>2</td>
<td>DataGridView1</td>
<td>Name</td>
<td>dtgEmployee</td>
</tr>
<tr>
<td>3</td>
<td>Button1</td>
<td>Name</td>
<td>btnShow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Show</td>
</tr>
</tbody>
</table>

5) Build the Project and Execute by Start Debugging or by pressing F5.

Exercise No6: (2 Hours) - 1 Practical

Imports System.Data
Imports System.Data.OleDb

Public Class frmDataGrid
    Public con As OleDbConnection
    Public da As OleDbDataAdapter
    Public ds As New DataSet

    Private Sub btnShow_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnShow.Click
        Try
            'Establish connection
            con = New OleDbConnection("Provider=MSDAORA;Data Source=XE;User ID=vikrant;Password=shaga")
            'Execute SQL Query
            da = New OleDbDataAdapter("Select * from Emp", con)
            'Fill dataset
            da.Fill(ds, "Emp")
            'Put data from dataset to datagridview
            dtgEmployee.DataSource = ds.Tables(0)
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub
End Class
7. Lab Exercises:

[Purpose of this exercise is Inserting, deleting, updating and searching records from database]

Steps to follow:
1) In FILE menu select “NEW PROJECT”.
2) Give the name for Project “DatabaseApplication” and click “OK”. Window with Form1.vb gets generated.
3) Add Textboxes and command buttons from toolbox.
4) Set the properties of controls as below:

<table>
<thead>
<tr>
<th>Sr no</th>
<th>Controls</th>
<th>Properties</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Form1</td>
<td>Name</td>
<td>frmDepartment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Database application</td>
</tr>
<tr>
<td>2</td>
<td>TextBox1</td>
<td>Name</td>
<td>txtDeptId</td>
</tr>
<tr>
<td>3</td>
<td>TextBox2</td>
<td>Name</td>
<td>txtDeptName</td>
</tr>
<tr>
<td>4</td>
<td>Button1</td>
<td>Name</td>
<td>btnAdd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Add</td>
</tr>
<tr>
<td>5</td>
<td>Button2</td>
<td>Name</td>
<td>btnSave</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Save</td>
</tr>
<tr>
<td>6</td>
<td>Button3</td>
<td>Name</td>
<td>btnDelete</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Delete</td>
</tr>
<tr>
<td>7</td>
<td>Button4</td>
<td>Name</td>
<td>btnUpdate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Update</td>
</tr>
<tr>
<td>8</td>
<td>Button5</td>
<td>Name</td>
<td>btnSearch</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Search</td>
</tr>
</tbody>
</table>

5) Build the Project and Execute by Start Debugging or by pressing F5.

Exercise No7: (2 Hours) - 1 Practical

Imports System.Data
Imports System.Data.OleDb

Public Class frmDepartment
    Public con As OleDbConnection
    Public cmd As OleDbCommand
    Public dr As OleDbDataReader
    Dim count As Integer = 100
    Dim deptNo As String
    Private Sub frmDepartment_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        btnSave.Enabled = False
        btnDelete.Enabled = False
        btnUpdate.Enabled = False
        Try
            'Establish connection
            con = New OleDbConnection("Provider=MSDAORA;Data Source=XE;User ID=vikrant;Password=shaga")
            Catch ex As Exception
                MsgBox(ex.Message)
        End Try
Private Sub btnAdd_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnAdd.Click
    btnAdd.Enabled = False
    btnSave.Enabled = True
    Try
        con.Open()
        cmd = New OleDbCommand("Select Max(substr(dno,2,3)) from dept", con)
        count = cmd.ExecuteScalar()
        count += 1
        con.Close()
    Catch ex As Exception
        count += 1 'for adding first record Or to handle Null value exception
    End Try
    txtDeptId.Text = "D" & count
End Sub

Private Sub btnSave_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnSave.Click
    btnSave.Enabled = False
    btnAdd.Enabled = True
    Try
        con.Open() 'Opening the connection
        cmd = New OleDbCommand("Insert into dept values('" & txtDeptId.Text & ","," & txtDeptName.Text & ")", con) 'Execute INSERT Query
        cmd.ExecuteNonQuery()
        MsgBox("Data Saved Successfully...")
        con.Close() 'Close the connection
        txtDeptId.Text = ""
        txtDeptName.Text = ""
    Catch ex As Exception
        MsgBox(ex.Message)
    End Try
End Sub

Private Sub btnSearch_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnSearch.Click
    btnDelete.Enabled = True
    btnUpdate.Enabled = True
    btnAdd.Enabled = False
    btnSave.Enabled = False
    deptNo = InputBox("Enter DeptNo to Search ")
    Try
        con.Open() 'Opening the connection
        cmd = New OleDbCommand("Select * from dept where dno='" & deptNo & ")", con) 'Execute SELECT Query
        cmd.ExecuteReader()
        If dr.Read Then
Private Sub btnDelete_Click(byval sender as system.object, byval e as system.eventargs)
Handles btnDelete.Click
    btnDelete.Enabled = false
    btnUpdate.Enabled = false
    btnAdd.Enabled = true
    btnSave.Enabled = true
    try
        con.Open()
        cmd = new olecommand("Delete from dept where dno='" & txtDeptId.Text & "]'", con)
        'execute delete query
        cmd.executeNonQuery()
        messagebox("Record Deleted Successfully...")
    txtDeptId.Text = ""
    txtDeptName.Text = ""
    con.Close()
    catch ex as exception
        messagebox(ex.message)
    end try
end sub

Private Sub btnUpdate_Click(byval sender as system.object, byval e as system.eventargs)
Handles btnUpdate.Click
    btnDelete.Enabled = false
    btnUpdate.Enabled = false
    btnAdd.Enabled = true
    btnSave.Enabled = true
    try
        con.Open()
        cmd = new olecommand("Update dept set dname='" & txtDeptName.Text & "' where dno='" & txtDeptId.Text & "'", con)
        'execute update query
        cmd.executeNonQuery()
        messagebox("Record Updated Successfully...")
    con.Close()
    catch ex as exception
        messagebox(ex.message)
    end try
end sub
8. Lab Exercises:

[Purpose of this exercise is to show records from database to crystal reports]
Steps to follow:

1) In FILE menu select “NEW PROJECT”.
2) Give the name for Project “CrystalReportApplication” and click “OK”. Window with Form1.vb gets generated.
3) Take combo box from toolbox on Form1.
4) Fetch Department numbers from database in Combo box.
5) Add Form2 and add CrystalReportViewer from toolbox on Form2.
6) Choose the New Crystal report “CrystalReport1.rpt”.
7) Attach the data source with CrystalReport1.rpt.
8) Set the properties of controls as below:

<table>
<thead>
<tr>
<th>Sr no</th>
<th>Controls</th>
<th>Properties</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Form1</td>
<td>Name</td>
<td>frmDepartment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>CrystalReport application</td>
</tr>
<tr>
<td>2</td>
<td>ComboBox1</td>
<td>Name</td>
<td>cmbDept</td>
</tr>
<tr>
<td>3</td>
<td>Button1</td>
<td>Name</td>
<td>btnShow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Show</td>
</tr>
<tr>
<td>4</td>
<td>Form2</td>
<td>Name</td>
<td>frmDeptWiseReport</td>
</tr>
<tr>
<td>5</td>
<td>CrystalReport1</td>
<td>Name</td>
<td>deptWise</td>
</tr>
</tbody>
</table>

9) Build the Project and Execute by Start Debugging or by pressing F5.

Exercise No.8: (2 Hours) - 1 Practical

[form1]
Imports System.Data
Imports System.Data.OleDb

Public Class frmDepartment
    Public con As OleDbConnection
    Public cmd As OleDbCommand
    Public dr As OleDbDataReader
    Dim count As Integer = 100
    Dim deptNo As String
    Private Sub frmDepartment_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Try
            'Establish connection
            con = New OleDbConnection("Provider=MSDAORA;Data Source=XE;User ID=vikrant;Password=shaga")
            Catch ex As Exception
                MsgBox(ex.Message)
            End Try
            Try
                con.Open()
                cmd = New OleDbCommand("Select * from dept", con) ' Execute SELECT Query
                dr = cmd.ExecuteReader()
                While dr.Read
                    ' Process data...
            End Try
    End Sub

22
Public Class frmDeptWiseReport
    Public mreport As New ReportDocument
    Public str As String
    Private Sub frmDeptWiseReport_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        str = "{DEPT.DNO}='" & frmDepartment.cmbDept.SelectedItem & "'
        mreport.Load("C:\Documents and Settings\Vicky\My Documents\Visual Studio 2008\VBNET Lab\Practical8\Practical8\singleDept.rpt")
        mreport.RecordSelectionFormula = str
        CrystalReportViewer1.ReportSource = mreport
    End Sub
End Class
9. Lab Exercises:

[Purpose of this exercise is to design web application that uses validation controls]

Steps to follow:
1) In FILE menu select “NEW WEBSITE”.
2) Give the name for Project “RegistrationForm” and click “OK”. Web form with Default.aspx gets generated.
3) Take Server Controls from toolbox on WebForm1.
4) Take Validation Controls from toolbox
5) Add one more web form. Give name Default2.aspx for showing records of first form.
6) Set the properties of controls as below:

<table>
<thead>
<tr>
<th>Sr no</th>
<th>Controls</th>
<th>Properties</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WebForm1</td>
<td>Name</td>
<td>Default.aspx</td>
</tr>
<tr>
<td>3</td>
<td>Button1</td>
<td>Name, Text</td>
<td>btnSubmit, Submit</td>
</tr>
<tr>
<td>4</td>
<td>TextBox1</td>
<td>Name</td>
<td>txtUserName</td>
</tr>
<tr>
<td>5</td>
<td>TextBox2</td>
<td>Name</td>
<td>txtChoosePwd</td>
</tr>
<tr>
<td>6</td>
<td>TextBox3</td>
<td>Name</td>
<td>txtConfirmPwd</td>
</tr>
<tr>
<td>7</td>
<td>TextBox4</td>
<td>Name</td>
<td>txtEmail</td>
</tr>
<tr>
<td>8</td>
<td>TextBox5</td>
<td>Name</td>
<td>txtAge</td>
</tr>
<tr>
<td>9</td>
<td>DropDownList1</td>
<td>Name</td>
<td>Citydropdown</td>
</tr>
<tr>
<td>10</td>
<td>RequiredFieldValidator1</td>
<td>Error Message</td>
<td>*</td>
</tr>
<tr>
<td>11</td>
<td>CompareValidator1</td>
<td>Error Message</td>
<td>Mismatch Password</td>
</tr>
<tr>
<td></td>
<td>CustomValidator1</td>
<td>Error Message</td>
<td>Please Select city</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ClientValidationFunction</td>
<td>CityValidate</td>
</tr>
<tr>
<td>11</td>
<td>RangeValidator1</td>
<td>Error Message</td>
<td>18-40</td>
</tr>
<tr>
<td></td>
<td>MinimumValue</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MaximumValue</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Type</td>
<td>Integer</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>RegularExpressionValidator1</td>
<td>Error Message</td>
<td>Invalid Email</td>
</tr>
<tr>
<td></td>
<td>ValidationExpression</td>
<td>^\w+([-.<em>']\w+)*@\w+([-.</em>']\w+)<em>.[\w-]</em>$</td>
<td></td>
</tr>
</tbody>
</table>

7) Browse with right clicking on default.aspx and selecting “View in Browser”

Exercise No.9: (2 Hours) - 1 Practical

[default.aspx]

```html
<%@ Page Language="VB" AutoEventWireup="false" CodeFile="Default.aspx.vb" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Registration</title>
</head>
```
<script language="JavaScript" type="text/javascript">

function CityValidate(ctl, args)
{
    args.IsValid=(args.Value != "Select");
}

function window_onload()
{
    CityValidate();
}
</script>

<body onload="return window_onload()">
<form id="form1" runat="server">
<table>
<tr>
    <td>
        UserName :
    </td>
    <td>
        <asp:TextBox ID="txtUserName" runat="server"></asp:TextBox>
    </td>
</tr>
<tr>
    <td>
        Choose Password :
    </td>
    <td>
        <asp:TextBox ID="txtChoosePwd" runat="server"></asp:TextBox>
    </td>
</tr>
<tr>
    <td>
        Confirm Password :
    </td>
    <td>
        <asp:TextBox ID="txtConfirmPwd" runat="server"></asp:TextBox>
    </td>
</tr>
</table>
</form>
</body>
ControlToCompare="txtChoosePwd" ControlToValidate="txtConfirmPwd" ErrorMessage="Password Mismatch"></asp:CompareValidator>
</td>
</tr>

<tr>
<td>
    Age:
</td>
<td>
    <asp:TextBox ID="txtAge" runat="server"></asp:TextBox>
</td>
<td>
    <asp:RequiredFieldValidator ID="RequiredFieldValidator3" runat="server"
ControlToValidate="txtAge" ErrorMessage="*"></asp:RequiredFieldValidator>
<asp:RangeValidator ID="RangeValidator1" runat="server"
ControlToValidate="txtAge" ErrorMessage="18-40" MaximumValue="40" MinimumValue="18" Type="Integer"></asp:RangeValidator>
</td>
</tr>

<tr>
<td>
    Email Id:
</td>
<td>
    <asp:TextBox ID="txtEmail" runat="server"></asp:TextBox>
</td>
<td>
    <asp:RequiredFieldValidator ID="RequiredFieldValidator4" runat="server"
ControlToValidate="txtEmail" ErrorMessage="*"></asp:RequiredFieldValidator>
<asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server"
ControlToValidate="txtEmail" ErrorMessage="Invalid email"
</td>
</tr>

<tr>
<td>
    City:
</td>
<td>
    <asp:DropDownList ID="CityDropDown" runat="server"
CausesValidation="True">
<asp:ListItem>Select</asp:ListItem>
<asp:ListItem>Pune</asp:ListItem>
<asp:ListItem>Mumbai</asp:ListItem>
</asp:DropDownList>
</td>
</tr>
<asp:CustomValidator ID="CustomValidator1" runat="server"
ErrorMessage="Please select the city" ClientValidationFunction="CityValidate"
ControlToValidate="CityDropDown"></asp:CustomValidator>
</td>
</tr>
</table>
<asp:Button ID="Button1" runat="server" Text="Submit" Style="top: 211px; left: 125px;
position: absolute; height: 26px; width: 61px" />
</form>
</body>
</html>

'Inline code( Default.aspx.vb)
Partial Class _Default
    Inherits System.Web.UI.Page
    Protected Sub Button1_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles Button1.Click
        Response.Write("Submitted successfully...")
    End Sub
End Class
10. Lab Exercises:

[Purpose of this exercise is to design web application and insert, delete, update and search the data from database. Also provide proper validations]

Steps to follow:
1) In FILE menu select “NEW WEBSITE”.
2) Give the name for Project “RegistrationForm” and click “OK”. Web form with Default.aspx gets generated.
3) Take Server Controls from toolbox on WebForm1.
4) Set the properties of controls as below:

<table>
<thead>
<tr>
<th>Sr no</th>
<th>Controls</th>
<th>Properties</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WebForm1</td>
<td>Name</td>
<td>Default.aspx</td>
</tr>
<tr>
<td>2</td>
<td>WebForm2</td>
<td>Name</td>
<td>Default2.aspx</td>
</tr>
<tr>
<td>3</td>
<td>Button1</td>
<td>Name</td>
<td>btnSubmit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Text</td>
<td>Submit</td>
</tr>
<tr>
<td>4</td>
<td>TextBox1</td>
<td>Name</td>
<td>txtName</td>
</tr>
<tr>
<td>5</td>
<td>TextBox2</td>
<td>Name</td>
<td>txtChoosePwd</td>
</tr>
<tr>
<td>6</td>
<td>TextBox3</td>
<td>Name</td>
<td>txtConfirmPwd</td>
</tr>
<tr>
<td>7</td>
<td>TextBox4</td>
<td>Name</td>
<td>txtEmail</td>
</tr>
<tr>
<td>8</td>
<td>TextBox5</td>
<td>Name</td>
<td>txtPhno</td>
</tr>
</tbody>
</table>

5) Create a database and connect to the database.
6) Browse with right clicking on default.aspx and selecting “View in Browser”

Exercise No.9: (2 Hours) - 1 Practical
[default.aspx]

```vb
<%@ Page Language="VB" AutoEventWireup="false" CodeFile="Default.aspx.vb" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Student’s Online Registration Form</title>
</head>
<body>
<form id="form1" runat="server">
    <br />
    <table class="" style="height: 148px; width: 489px">
        <tr>
            <td class="">
                Candidate Name :
            </td>
            <td class="">
                <asp:TextBox ID="txtName" runat="server"></asp:TextBox>
            </td>
        </tr>
        <tr>
            <td class="">
                Choose password :
            </td>
        </tr>
    </table>
</form>
</body>
</html>
```
<td class="">
<asp:TextBox ID="txtChoosePwd" runat="server"></asp:TextBox>
</td>
</tr>
<tr>
<td class="">
Confirm Password :
</td>
<td class="">
<asp:TextBox ID="txtConfirmPwd" runat="server"></asp:TextBox>
</td>
</tr>
<tr>
<td class="">
Email Address :
</td>
<td class="">
<asp:TextBox ID="txtEmail" runat="server"></asp:TextBox>
</td>
</tr>
<tr>
<td class="">
Phone No :
</td>
<td class="">
<asp:TextBox ID="txtPhno" runat="server"></asp:TextBox>
</td>
</tr>
</table>
<br />
<p>
<asp:Button ID="btnSearch" runat="server"
style="top: 282px; left: 234px; position: absolute; height: 26px; width: 56px; right: 941px;"
Text="Search" />
<asp:Button ID="btnSubmit" runat="server"
style="top: 282px; left: 154px; position: absolute; height: 26px; width: 56px"
Text="Submit" />
</p>
<p>&nbsp;
<asp:Button ID="btnUpdate" runat="server"
style="top: 284px; left: 318px; position: absolute; height: 26px; width: 56px"
Text="Update" />
<p>
<asp:Button ID="btnDelete" runat="server"
style="top: 284px; left: 396px; position: absolute; height: 26px; width: 56px"
Text="Delete" />
</p>
</form>
</body>
</html>
Imports System.Data
Imports System.Data.OleDb

Partial Class _Default
  Inherits System.Web.UI.Page

  Dim con As OleDbConnection
  Dim cmd As OleDbCommand
  Dim dr As OleDbDataReader

  Public Sub getConnection()
    Try
      con = New OleDbConnection("Provider=MSDAORA;Data source=XE;UserID=vikrant;Password=shaga")
      Catch ex As Exception
        Response.Write("Error =>" & ex.Message)
    End Try
  End Sub

  Protected Sub btnSearch_Click(ByVal sender As Object, ByVal e As System.EventArgs)
    Handles btnSearch.Click
    Try
      getConnection()
      con.Open()
      cmd = New OleDbCommand("Select * from registration where username='" & txtName.Text & "'", con)
      dr = cmd.ExecuteReader()
      If dr.Read Then
        txtChoosePwd.Text = dr(1)
        txtConfirmPwd.Text = dr(1)
        txtEmail.Text = dr(2)
        txtPhno.Text = dr(3).ToString()
      Else
        Response.Write("No Records")
      End If
      con.Close()
    Catch ex As Exception
      Response.Write("Error =>" & ex.Message)
    End Try
  End Sub

  Protected Sub btnSubmit_Click1(ByVal sender As Object, ByVal e As System.EventArgs)
    Handles btnSubmit.Click
    Try
      getConnection()
      con.Open()
cmd = New OleDbCommand("Insert into registration values ('" & txtName.Text & "," & txtConfirmPwd.Text & "," & txtEmail.Text & "," & Convert.ToInt64(txtPhno.Text) & ")", con)
    cmd.ExecuteNonQuery()
    Response.Write("Inserted successfully...")
    con.Close()
Catch ex As Exception
    Response.Write("Error =>" & ex.Message)
End Try
End Sub

Protected Sub btnUpdate_Click(ByVal sender As Object, ByVal e As System.EventArgs)
Handles btnUpdate.Click
    Try
        getConnection()
        con.Open()
        cmd.ExecuteNonQuery()
        Response.Write("Updated successfully...")
        con.Close()
    Catch ex As Exception
        Response.Write("Error =>" & ex.Message)
    End Try
End Sub

Protected Sub btnDelete_Click(ByVal sender As Object, ByVal e As System.EventArgs)
Handles btnDelete.Click
    Try
        getConnection()
        con.Open()
        cmd = New OleDbCommand("Delete from registration where username='" & txtName.Text & "'", con)
        cmd.ExecuteNonQuery()
        Response.Write("Deleted successfully...")
        con.Close()
    Catch ex As Exception
        Response.Write("Error =>" & ex.Message)
    End Try
End Sub

txtName.Text = ""
txtConfirmPwd.Text = ""
txtChoosePwd.Text = ""
txtEmail.Text = ""
txtPhno.Text = ""
End Sub
End Class
4. **Quiz on the subject:**

Quiz should be conducted on tips in the laboratory, recent trends and subject knowledge of the subject. The quiz questions should be formulated such that questions are normally are from the scope outside of the books. However twisted questions and self formulated questions by the faculty can be asked but correctness of it is necessarily to be thoroughly checked before the conduction of the quiz.

5. **Conduction of Viva-Voce Examinations:**

Teacher should oral exams of the students with full preparation. Normally, the objective questions with guess are to be avoided. To make it meaningful, the questions should be such that depth of the students in the subject is tested. Oral examinations are to be conducted in co-cordial environment amongst the teachers taking the examination. Teachers taking such examinations should not have ill thoughts about each other and courtesies should be offered to each other in case of difference of opinion, which should be critically suppressed in front of the students.

6. **Submission:**

Document Standard:

A] Page Size              A4 Size
B] Running text          Justified text
C] Spacing               1 Line
D] Page Layout and Margins (Dimensions in Cms)

Normal Page

<table>
<thead>
<tr>
<th>Top</th>
<th>Right</th>
<th>Bottom</th>
<th>Left</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Top</th>
<th>Right</th>
<th>Bottom</th>
<th>Left</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5</td>
<td>2.5</td>
<td>2.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Top</th>
<th>Right</th>
<th>Bottom</th>
<th>Left</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.7&quot;</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>
7. Evaluation and marking system:

Basic honesty in the evaluation and marking system is absolutely essential and in the process impartial nature of the evaluator is required in the examination system to become popular amongst the students. It is a wrong approach or concept to award the students by way of easy marking to get cheap popularity among the students to which they do not deserve. It is a primary responsibility of the teacher that right students who are really putting up lot of hard work with right kind of intelligence are correctly awarded.

The marking patterns should be justifiable to the students without any ambiguity and teacher should see that students are faced with unjust circumstances.