Marathwada Shikshan Prasarak Mandal's
Deogiri Technical Campus for Engineering and Management Studies
Practical Experiment Instruction Sheet
Department: C.S.E. **Subject:** Software Development Lab-I
Master List of Experiments

Sr. No.	Name of Experiment
1	Introduction to Windows Application using C#.NET with basic controls.
2	Creating a calculator Application in C#.NET in windows application.
3	Connectivity with database in windows application.
4	Introduction to Web Applications using ASP.NET with AdRotator control.
5	Design sign up form and validate it in Web Application.
6	Design a web page to display, add, delete & edit information from database.
7	Create a simple web service i.e. Fahrenheit to Celsius conversion
8	Mini Project

Marathwada Shikshan Prasarak Mandal's

Deogiri Technical Campus for Engineering and Management Studies

Practical Experiment Instruction Sheet

Department: C.S.E. **Subject:** Software Development Lab-I **Experiment No. :** 1

Experiment Title: Introduction to Windows Applications using C#.NET with basic controls.

Aim: Design a form to demonstrate basic Windows Form controls using Picture Box.

Hardware Requirement: P-IV Processor, 40 GB HDD, 256MB RAM or above.

Software Requirement: Visual Studio 2010

Theory: To create this application we require following controls:

1. **ListBox**
   This control is typically used to display a list of items from which the user can choose. This ListBox control inherits directly from the ListControl class and is an ancestor to the CheckedListBox class.
2. **PictureBox**
   This control is used to display an image. The PictureBox control inherits directly from the Control class.
3. **Button**
   The most common control used to obtain a user response is a Button. Button press event let you place code in the **click** event handler to perform any action defined in the button event.

Stepwise Procedure

1. Select the new project option from file menu.
2. Select a new window application, give it specific name.
3. Design the window as shown in following Figure 5.1
4. Implement the code
5. Run the application using start icon or by pressing F5.

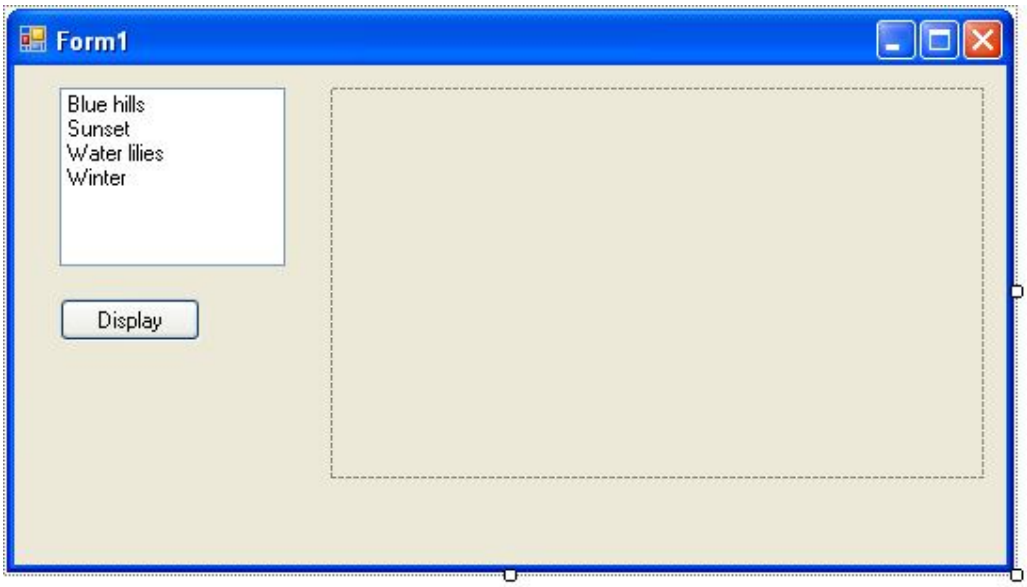

Figure 5.1 Form Design to select image form list and to display it in picture box

Lab Work

Write code to select image from list and display it in the picture box.

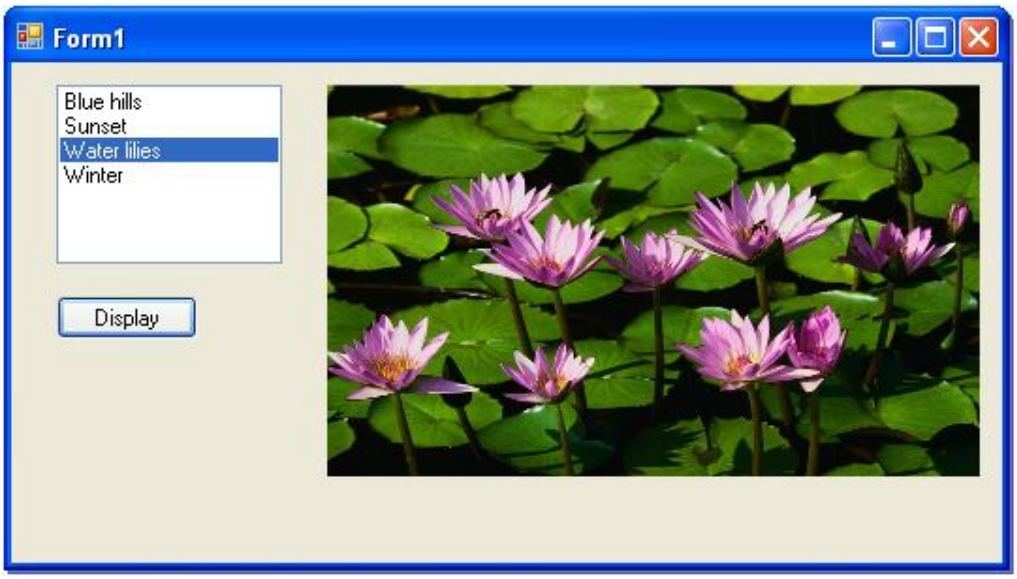

Figure 5.2 Output to display image in picture box

Conclusion:

Thus, we have studied introduction to basic windows forms controls by using PicturBox control.

Date of performance by Student	Date of Assessment by Staff	Staff Signature	Remark

Marathwada Shikshan Prasarak Mandal's

Deogiri Technical Campus for Engineering and Management Studies

Practical Experiment Instruction Sheet

Department: C.S.E. **Subject:** Software Development Lab-I **Experiment No. :** 2

Experiment Title: Creating a calculator Application in C#.NET in windows application.

Aim: Design form to create a calculator application.

Hardware Requirement: P-IV Processor, 40 GB HDD, 256MB RAM or above.

Software Requirement: Visual Studio 2010

Theory: To create a calculator application we require two controls:

1. **Button:** The most common control used to obtain a user response is a Button. Button press event let you place code in the **click** event handler to perform any action defined in the button event.
2. **TextBox:** The TextBox control lets your user provide text input to an application. The control provided by .NET includes additional functionality not in the standard Windows TextBox control. The TextBox is mainly used to display, or accept, a single line of text.

Stepwise Procedure

1. Select the new project option from file menu.
2. Select a new window application, give it specific name.
3. Design the window as shown in following Figure 2.1
4. Implement the code
5. Run the application using start icon or by pressing F5.

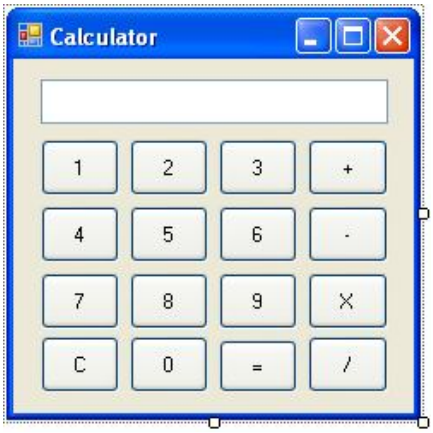

Figure 2.1 Form Design for Calculator Application

Lab Work

1. Write code for calculator application shown below.

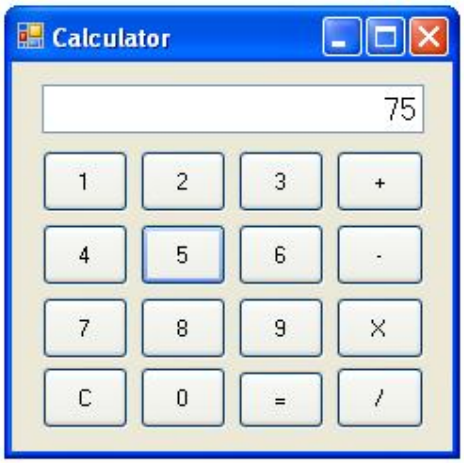

Figure 2.2 Output of calculator program.

Conclusion

Thus we have studied Button and TextBox controls, different arithmetic operators and we have implemented a calculator application.

Date of performance by Student	Date of Assessment by Staff	Staff Signature	Remark

Marathwada Shikshan Prasarak Mandal's

Deogiri Technical Campus for Engineering and Management Studies

Practical Experiment Instruction Sheet

Department: C.S.E. **Subject:** Software Development Lab-I **Experiment No. :** 3

Experiment Title: Connectivity with the database in windows application.

Aim: Design an application to demonstrate the use of database in your program.

Hardware Requirement: P-IV Processor, 40 GB HDD, 256MB RAM or above.

Software Requirement: Visual Studio 2010

Theory:

To create an application to demonstrate working with the database we require following controls:

1. **Label**
   This control is typically used to display descriptive text. The Label control inherits directly from the Control class.
2. **Button**
   The most common control used to obtain a user response is a Button. Button press event let you place code in the **click** event handler to perform any action defined in the button event.
3. **TextBox**
   The TextBox control lets your user provide text input to an application. The control provided by .NET includes additional functionality not in the standard Windows TextBox control. The TextBox is mainly used to display, or accept, a single line of text.

Here to create the application for working with database we have to use **ADO.NET,** to study the database connectivity we have to implement the following **ADO.NET** components:

1. Connection object:

The connection object is basically required to establish the connection between the front end i.e. the C#.NET application and whichever database we are going to use in our programs.

Connection object has some important properties such as Open(), Close() etc.

1. **Command object:**
   The command object is used to specify the required command which we want to execute on the database. It has some important properties such as CommandText, ExecuteNonQuery, ExecuteReader etc.

2. **DataReader:**
   The DataReader object is required to work mainly in disconnected mode. It is generally used to handle the data fetched from the database and required to display through the frontend.

Stepwise Procedure

1. Select the new project option from file menu.
2. Select a new window application, give it specific name.
3. Design the window as shown in following Figure 7.1
4. Implement the code
5. Run the application using start icon or by pressing F5.

Conclusion:

Thus we have studied the different operations with the database by using the C#.NET.

Deogiri Institute of Engineering and Management Studies

Practical Experiment Instruction Sheet

Department: C.S.E. **Subject:** Software Development Lab-II **Experiment No. : 4**

Experiment Title: Introduction to Web Applications using ASP.NET with AdRotator control

Aim: Design a web application to demonstrate AdRotator control and other basic controls to introduce the ASP.NET.

Hardware Requirement:

Processor: P4

Memory: 256 MB RAM or above

HDD Capacity: 80 GB or above

Software Requirements:

System Software: Windows Xp

Application Software: .Net Framework, Visual Studio

Theory :

AdRotator Control:

The AdRotator control is basically used to display the different adds with the different probabilities of display. It is generally used with the XML file which is to be embedded in the AdRotator control. Here we have to just set the property of AdRotator control i.e. *XML file* to the already created in XML.

XML File:

It is created for including the advertisement file into it. There are different tags available for creating the XML file:

<Advertisements> Tag:

This tag is used to create the different advertisements in the XML file. There is the important **<Ad>** tag which is embedded into the **<Advertisements>** tag. Any number of **<Ad>** tags can be embedded into the **<Advertisements>** tag.

<Ad> Tag:

The <**Ad**> tag is used to actually display the advertisement file into the XML file. There are some important sub tags of <**Ad>** tag which are as follows:

<ImageUrl> Tag:

This tag is used to embed the GIF image in the XML file. The syntax of this tag is:

<ImageUrl>~\FolderName\ImageName.gif<\ImageUrl>

<NavigationUrl> Tag:

This tag is used to navigate through the web page whose path is given in this tag when the user clicks on the image. The syntax of this tag is:

<NvigationUrl>http://www.google.com</NvigationUrl>

<AlternateText> Tag:

This tag is used to display the alternative text if the image is not properly displayed on the screen. The syntax for this tag is:

<AlternateText>Hiiiiiiii</AlternateText>

<Impressions> Tag:

This is the most important tag because it decides the probability by which the different images are going to be displayed on the AdRotator control. All the images should be specified in the <Ad> tag of XML file. The syntax of this tag:

<Impressions>10</Impressions>

Conclusion:

Date of performance by Student	Date of Assessment by Staff	Staff Signature	Remark

Deogiri Institute of Engineering and Management Studies

Practical Experiment Instruction Sheet

Department: C.S.E. **Subject:** Software Development Lab-I **Experiment No. :** 5

Experiment Title: Design Sign Up form and validate it in Web Application

Aim: Design Sign Up form and validate it.

Hardware Requirement:

Processor: P4

Memory: 256 MB RAM or above

HDD Capacity: 80 GB or above

Software Requirements:

System Software: Windows Xp

Application Software: .Net Framework, Visual Studio

Theory :

When using a TextBox for data entry a risk is that users may not enter necessary or appropriate data for script processing. For common types of data validation, ASP.NET provides a set of **validation controls**. These validators can test for missing values, comparison values, values within a range, and other forms of data to ensure that proper data is supplied to processing scripts. These controls are associated with TextBox controls and perform their tests automatically when Button, LinkButton, or ImageButton controls are clicked to call subprograms for processing. If a validation test is not met, the validator displays an error message to call attention to this fact, and the user is given the chance to reenter valid data in the associated TextBox.

Common properties for all validation control :-

An id is required only if the validator is referenced in a script. The ControlToValidate property gives the id of the TextBox to which the validation test is applied. ErrorMessage supplies a text and XHTML string to format an error message. SetFocusOnErrorMessage places a blinking text cursor in the associated TextBox for ease in entering a value. The space is preallocated on the page unless Display="Dynamic" is coded for the validator.

RequiredFieldValidator Control –

The <asp:RequiredFieldValidator> control tests a TextBox for a missing value and issues an error message if this is the case. The RequiredFieldValidator occupies horizontal space on the page equal to the length of the error message. Often, this error-message space appears along side the TextBox to which it applies.

RangeValidator Control –

The <asp:RangeValidator> control tests a TextBox for a value within the range of two values, inclusively. The test can be made for Currency (can include a dollar sign and commas), Date, Double (floating-point), Integer, or

String (default) data types. The entered value is treated as a string if a different type is not given by the Type property. An empty TextBox is evaluated as a valid data type; therefore, a RequiredFieldValidator normally needs to be paired with the RangeValidator to ensure that missing data is not treated as valid.

CompareValidator Control –

The <asp:CompareValidator> control tests a TextBox for a value within the range of two values, inclusively. The test can be made for Currency (can include a dollar sign and commas), Date, Double (floating-point), Integer, or String (default) data types.

The value against which the TextBox value is compared can be a literal value (given by the ValueToCompare property) or it can be the value contained in another control on the page (given

by the ControlToCompare property). By default, the comparison is for equality; however, other types of comparisons are made by coding the Operator property. By selecting the DataTypeCheck operator a test is made for a comparable data type to that given by the Type property. In this case, any ControlToCompare or ValueToCompare property setting is ignored.

CustomValidator Control –

Some validation tests that cannot be performed with a combination of RequiredFieldValidator, RangeValidator, and CompareValidator. The <asp:CustomValidator> is available for these additional tests.

The control's properties are similar to other validation controls except that it includes the OnServerValidate property to call a subprogram to perform explicitly coded tests. Again, an empty TextBox is considered valid, so a RequiredFieldValidator normally needs to be paired with the CustomValidator to test for a missing value prior to performing custom tests.

The called subprogram has the special ServerValidateEventArgs argument. Its IsValid property is set to False to indicate failure of a validation test. The argument's Value property is a reference to the value of the TextBox identified in the control's ControlToValidate property.

ValidationSummary Control –

By coding an <asp:ValidationSummary> control on the page individual error messages generated by separate controls can be displayed together. This summary control also can limit its error reporting to an identified set of grouped controls. The DisplayMode of the error summary is a bulleted list unless a simple List or SingleParagraph is specified. The report displays at the page location of the ValidationSummary control. Instead, it can be displayed as a pop-up message box by coding ShowMessageBox="True"; its on-page display is suppressed with ShowSummary="False".

Conclusion: Thus we have studied validation controls and we have validate sign u form.

Date of performance by Student	Date of Assessment by Staff	Staff Signature	Remark

Deogiri Institute of Engineering and Management Studies

Practical Experiment Instruction Sheet

Department: C.S.E. **Subject:** Software Development Lab-I **Experiment No. : 6**

Experiment Title: Design a web page to display, add, delete & edit information from database.

Aim: Design a web page to display, add, delete & edit information from database.

Hardware Requirement:

Processor: P4

Memory: 256 MB RAM or above

HDD Capacity: 80 GB or above

Software Requirements:

System Software: Windows Xp

Application Software: .Net Framework, Visual Studio

Theory :

ADO.NET is an object-oriented set of libraries that allows you to interact with data sources. Commonly, the data source is a database, but it could also be a text file, an Excel spreadsheet, or an XML file. For the purposes of this tutorial, we will look at ADO.NET as a way to interact with a data base.

As you are probably aware, there are many different types of databases available. For example, there is Microsoft SQL Server, Microsoft Access, Oracle, Borland Interbase, and IBM DB2, just to name a few. To further refine the scope of this tutorial, all of the examples will use SQL Server.

Data Providers

We know that ADO.NET allows us to interact with different types of data sources and different types of databases. However, there isn't a single set of classes that allow you to accomplish this universally. Since different data sources expose different protocols, we need a way to communicate with the right data source using the right protocol Some older data sources use the ODBC protocol, many newer data sources use the OleDb protocol, and there are more data sources every day that allow you to communicate with them directly through .NET ADO.NET class libraries.

ADO.NET provides a relatively common way to interact with data sources, but comes in different sets of libraries for each way you can talk to a data source. These libraries are called

Data Providers and are usually named for the protocol or data source type they allow you to interact with. Table 1 lists some well known data providers, the API prefix they use, and the type of data source they allow you to interact with.

Table 1. ADO.NET Data Providers are class libraries that allow a common way to interact with specific data sources or protocols. The library APIs have prefixes that indicate which provider they support.

Provider Name	API prefix	Data Source Description
ODBC Data Provider	Odbc	Data Sources with an ODBC interface. Normally older data bases.
OleDb Data Provider	OleDb	Data Sources that expose an OleDb interface, i.e. Access or Excel.
Oracle Data Provider	Oracle	For Oracle Databases.
SQL Data Provider	Sql	For interacting with Microsoft SQL Server.
Borland Data Provider	Bdp	Generic access to many databases such as Interbase, SQL Server, IBM DB2, and Oracle.

An example may help you to understand the meaning of the API prefix. One of the first ADO.NET objects you'll learn about is the connection object, which allows you to establish a connection to a data source. If we were using the OleDb Data Provider to connect to a data source that exposes an OleDb interface, we would use a connection object named OleDbConnection. Similarly, the connection object name would be prefixed with Odbc or Sql for an OdbcConnection object on an Odbc data source or a SqlConnection object on a SQL Server database, respectively. Since we are using MSDE in this tutorial (a scaled down version of SQL Server) all the API objects will have the Sql prefix. i.e. SqlConnection.

ADO.NET Objects

ADO.NET includes many objects you can use to work with data. This section introduces some of the primary objects you will use. Over the course of this tutorial, you'll be exposed to many more ADO.NET objects from the perspective of how they are used in a particular lesson. The objects below are the ones you must know. Learning about them will give you an idea of the types of things you can do with data when using ADO.NET.

The SqlConnection Object

To interact with a database, you must have a connection to it. The connection helps identify the database server, the database name, user name, password, and other parameters that are required for connecting to the data base. A connection object is used by command objects so they will know which database to execute the command on.

The SqlCommand Object

The process of interacting with a database means that you must specify the actions you want to occur. This is done with a command object. You use a command object to send SQL statements to the database. A command object uses a connection object to figure out which database to communicate with. You can use a command object alone, to execute a command directly, or assign a reference to a command object to an SqlDataAdapter, which holds a set of commands that work on a group of data as described below.

The SqlDataReader Object

Many data operations require that you only get a stream of data for reading. The data reader object allows you to obtain the results of a SELECT statement from a command object. For performance reasons, the data returned from a data reader is a fast forward-only stream of data. This means that you can only pull the data from the stream in a sequential manner This is good for speed, but if you need to manipulate data, then a DataSet is a better object to work with.

The DataSet Object

DataSet objects are in-memory representations of data. They contain multiple Datatable objects, which contain columns and rows, just like normal database tables. You can even define relations between tables to create parent-child relationships. The DataSet is specifically designed to help manage data in memory and to support disconnected operations on data, when such a scenario make sense. The DataSet is an object that is used by all of the Data Providers, which is why it does not have a Data Provider specific prefix.

The SqlDataAdapter Object

Sometimes the data you work with is primarily read-only and you rarely need to make changes to the underlying data source Some situations also call for caching data in memory to minimize the number of database calls for data that does not change. The data adapter makes it easy for you to accomplish these things by helping to manage data in a disconnected mode. The data adapter fills a DataSet object when reading the data and writes in a single batch when persisting changes back to the database. A data adapter contains a reference to the connection object and opens and closes the connection automatically when reading from or writing to the database. Additionally, the data adapter contains command object references for SELECT, INSERT, UPDATE, and DELETE operations on the data. You will have a data adapter defined for each table in a DataSet and it will take care of all communication with the database for you. All you need to do is tell the data adapter when to load from or write to the database.

Conclusion: Thus we have studied the database connectivity in ASP.NET and implemented the different operations.

Date of performance by Student	Date of Assessment by Staff	Staff Signature	Remark

Deogiri Institute of Engineering and Management Studies

Practical Experiment Instruction Sheet

Department: C.S.E. **Subject:** Software Development Lab-I **Experiment No. :** 7

Experiment Title: Create a simple web service i.e. Fahrenheit to Celsius conversion

Aim:

Hardware Requirement:

Processor: P4

Memory: 256 MB RAM or above

HDD Capacity: 80 GB or above

Software Requirements:

System Software: Windows Xp

Application Software: .Net Framework, Visual Studio

Thoery:

XML Web Services -

A Web Service (XML Web Service) is a unit of code that can be activated using HTTP requests. Stated another way, a Web Service is an application component that can be remotely callable using standard Internet Protocols such as HTTP and XML. One more definition can be, a Web Service is a programmable URL. Web Services came into existence to deliver distributed computing over the Internet. A major advantage of the Web services architecture is, it allows programs written in different languages on different platforms to communicate with each other in a standards-based way. Simply said, a Web service is a software service exposed on the Web through SOAP, described with a WSDL file and registered in UDDI.

Creating a Web Service File –

In order to convert the GetShoppingDays() function into a Web Service it must be recoded as a Visual Basic class and saved as a .asmx file.

For example –

```
<%@ WebService Class="ShoppingDays" %>

Imports System
Imports System.Web
Imports System.Web.Services
Imports System.Xml.Serialization

 Public Class ShoppingDays
 Inherits Web Service
 <WebMethod()> Public Function GetShoppingDays() As Integer
  Dim NoShoppingDays As Integer
  Dim ChristmasDay As String = "12/24/" & DatePart("yyyy", DateString())

  ShoppingDays = DateDiff("d", DateString(), ChristmasDay)
  Return NoShoppingDays
 End Function
End Class
```

The <%@ WebService Class="ShoppingDays" %> directive identifies this as a Web Service and supplies the name through which it is accessed. The System, System.Web, System.Web.Services, and System.XML.Serialization namespaces are imported to expose classes that ASP.NET requires to enable Web Services; the Microsoft.VisualBasic namespace is imported to expose intrinsic Visual Basic functions—DateDiff() and DateString() in the current example.

The Public Class named ShoppingDays is created as a wrapper for the function and the function is prefixed with <WebMethod()> to expose the function for access through a Web Service. Without the WebMethod() applied to the function it would be accessible only to local pages.

This file is saved with the .asmx extension inside a folder that is accessible from the Web (c:\MyWebServices).

Testing a Web Service-

A Web Service can be tested in the browser by entering its local URL.

Conclusion: Thus we have created a web service.

Date of performance by Student	**Date of Assessment by Staff**	**Staff Signature**	**Remark**

Marathwada Shikshan Prasarak Mandal's

Deogiri Technical Campus for Engineering and Management Studies

Practical Experiment Instruction Sheet

Department: C.S.E. **Subject:** Software Development Lab-I **Experiment No.:** 8

Experiment Title: Mini Project

Aim: Develop the mini project in ASP.NET i.e in web application using C#.NET.

Hardware Requirement: P-IV Processor, 40 GB HDD, 256MB RAM or above.

Software Requirement: Visual Studio 2010

Theory:

Mini Project shall follow the steps below:
1. Requirement Analysis
2. Design
3. Coding
4. Testing
5. Deployment

The report of this Mini project is to be submitted in typed form with Spiral Binding. The report should Have all the necessary diagrams, charts, printouts and source code. The work has to be done in groups.

Conclusion: Thus we have developed the project in web application as per the requirement.

Date of performance by Student	Date of Assessment by Staff	Staff Signature	Remark