1. Introduction

Project Title : BUGTRACKER

Project category : Web Based Application

The project named **BUGTRACKER** is developed using ASP.NET with C#

BUGTRACKER: Bugzilla is a "Defect Tracking System" or "Bug-Tracking System". Defect Tracking Systems allow individual or groups of developers to keep track of outstanding bugs in their product effectively. Most commercial defect-tracking software vendors charge enormous licensing fees. Despite being "free", Bugzilla has many features its expensive counterparts lack. Consequently, Bugzilla has quickly become a favorite of hundreds of organizations across the globe.

What Does Bugzilla Do?

Track bugs and code changes
Communicate with teammates
Submit and review patches
Manage quality assurance (QA)

Bugzilla can help you get a handle on the software development process. Successful projects often are the result of successful organization and communication. Bugzilla is a powerful tool that will help your team get organized and communicate effectively.

Bugzilla is Hard to Beat.

1.1 ABSTRACT OF PROJECT

Bugzilla is a "Defect Tracking System" or "Bug-Tracking System". Defect Tracking Systems allow individual or groups of developers to keep track of outstanding bugs in their product effectively. Most commercial defect-tracking software vendors charge enormous licensing fees. Despite being "free", Bugzilla has many features its expensive counterparts lack. Consequently, Bugzilla has quickly become a favorite of hundreds of organizations across the globe.

Bugzilla can help you get a handle on the software development process. Successful projects often are the result of successful organization and communication. Bugzilla is a powerful tool that will help your team get organized and communicate effectively.

Bugzilla is Hard to Beat

1.2 Objectives

1. Track bugs and code changes
2. Communicate with teammates
3. Submit and review patches
4. Manage quality assurance (QA)
5. Systems administration
6. Deployment management
7. Chip design and development problem tracking (both pre-and-post fabrication)
8. Software and hardware bug tracking
9. IT support queues

1.3 Problem specification/Need Of Project

Many companies are finding that Bugzilla helps reduce downtime, increase productivity, raise customer satisfaction, and improve communication Bugzilla can also help reduce costs by providing IT support accountability, telephone support knowledge bases, and by keeping tabs on unusual system or software issues. Bugzilla can do the same for your organization, regardless of its size.

Possible Uses

Systems administration
Deployment management
Chip design and development problem tracking (both pre-and-post fabrication)
Software and hardware bug tracking
IT support queues

2. Feasibility Study

Scope:

This document shall provide the requirement specification for the "BUGTRACKER" as per the scope defined.

Users:

This site can be used by 3 types of users:

1.1. Client : The users who is likely to form and add bug in the website.

1.2. Admin : The administrator of the system who will be responsible for add bug, add client, aad project and technology.

1.3. Employee : The employee can assign status, update status and can view assignment.

Assumptions:

The alerts will not be provided by the site, the user has to visit it to get the information.

Requirements:

Functional Requirements:

The portal will search for the details of the bug and project as per details entered by user.

Every user has to login.

Admin can add, modify or delete records related to bug and project.

The user can search for bug solution and can get solution from employee's throw admin.

The concept of Master Pages and Content Pages is used.

Non-functional Requirements:

Portability:

The system will be designed to be portable across popular Windows OS.

Extensibility:

The system should be extensible to add further information and users for more expansion.

Re-Usability:

The system's code could be reused to add further new features if need to be added in future.

Reliability and Availability:

System shall be able to deliver the required in reliable manner.

Software Upgradeability:

System is to be developed in phases, so it shall be easily upgradeable to include the new items in the database.

User Interface Requirements:

5.1 Log in screen:

Admin has to first log in to the site.

5.2 Home page:

Home page allows user to know about bugtracker , why it comes and main possible uses of it..

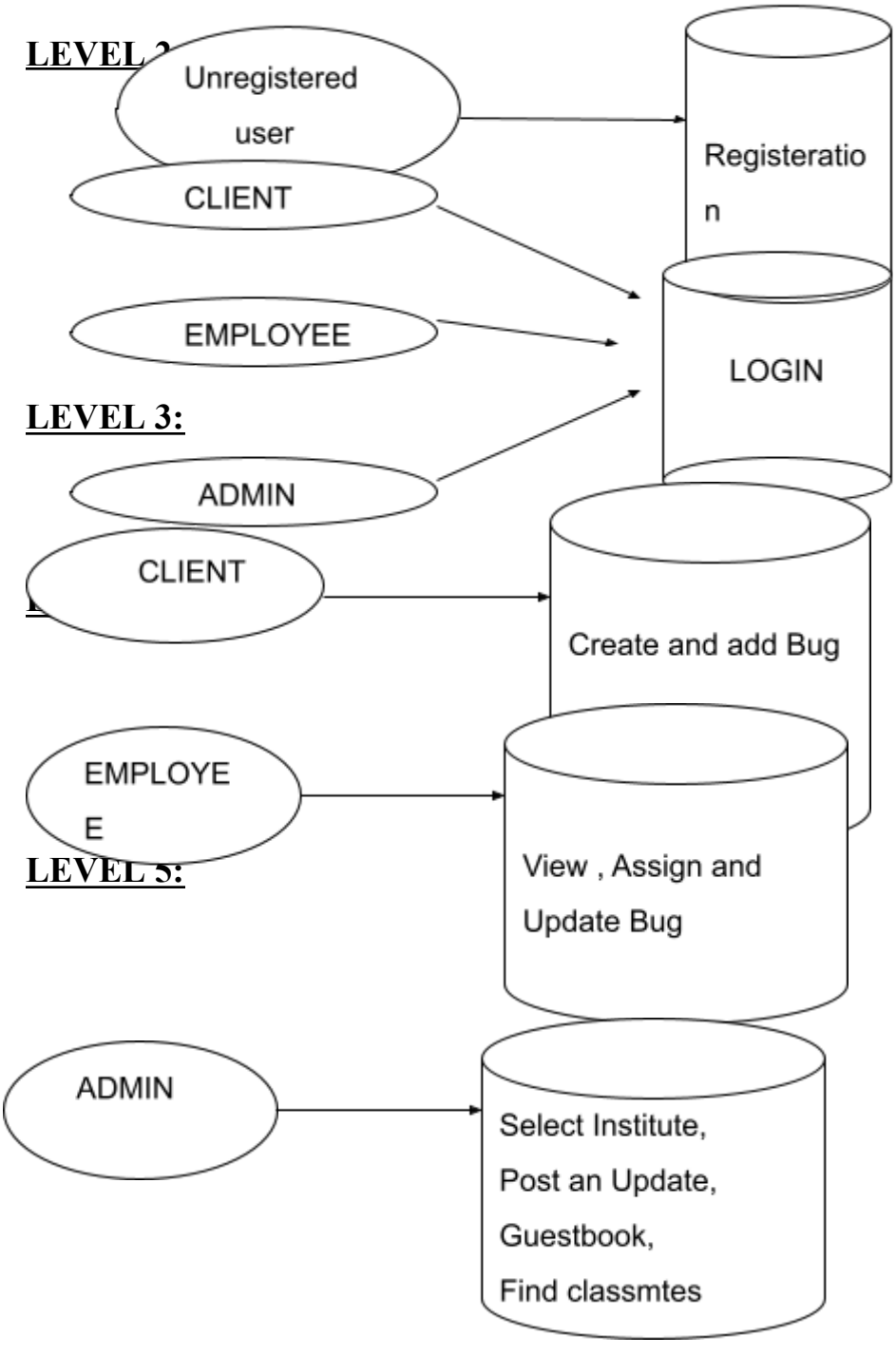
3. Software Requirement Specification

- ASP.NET 4.0 is being used for developing the web pages.
- C#. NET is being used for logic development for the project.
- SQL SERVER is being used for managing the database at server side.
- **OS Required** Microsoft Windows XP / higher version of Windows OS required.

4.1 ER Diagram

LEVEL 1:

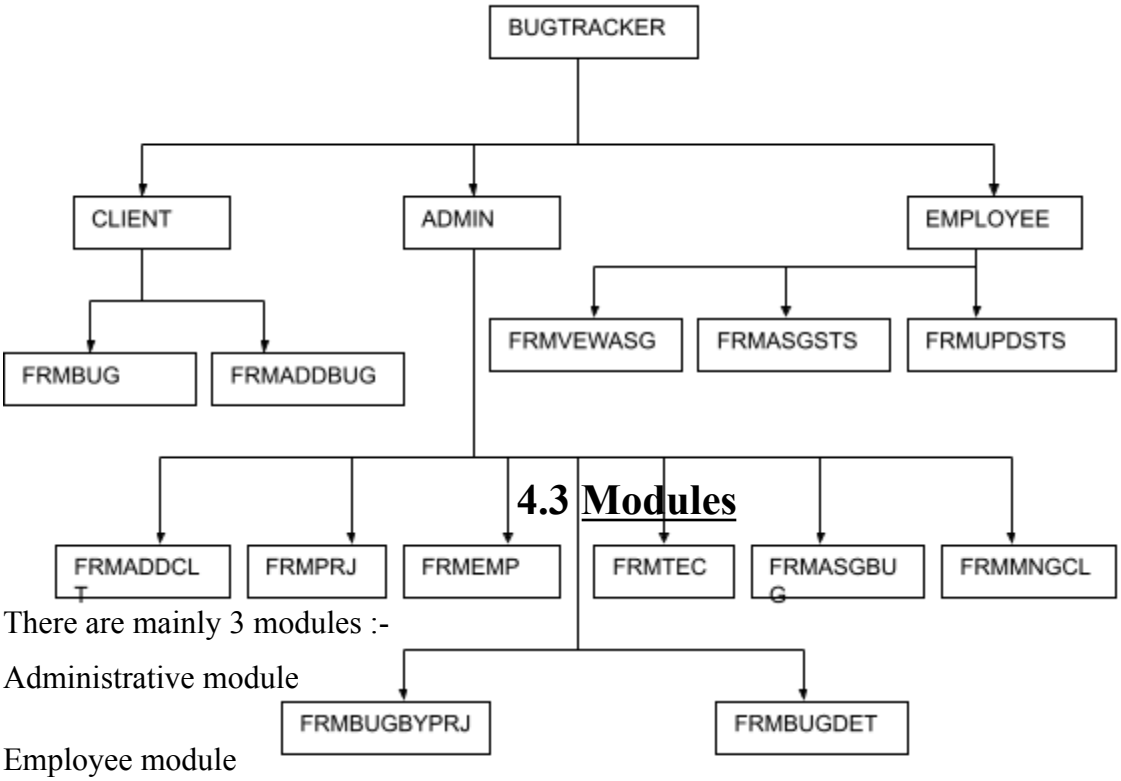
LEVEL 2



LEVEL 3:

LEVEL 5:

4.2 Data Flow Diagram



4.3 Modules

There are mainly 3 modules :-

- Administrative module
- Employee module
- Client Module

In the site we can contact to the admin for the partnership. Administrator has the power to add new Institute and can update and delete institute. First of all admin or user has to login for there further processes. User can login only after registrations. One user can not update or delete any other user and institute . He has no rights to do that work.

User can select institutes and post an update. The login user can also see the update's of another user but can not delete that one, he has the power to delete only his own update's. And user can see guestbook and can find his classmates.

4.4 DataBase

Sr. No	Table Name	Description
1.	Tbasg	Information about assign of Bug
2.	Tbbug	Information about the Bug
3.	Tbbugdep	Information about
4.	Tbbugres	Information about Bug Register
5.	Tbclt	Information about the
6.	Tbemp	Information about the Employee
7.	Tbmsg	Information about the
8.	Tbprj	Information about the

9.	Tbprjtec	Information about the
10.	Tbtec	Information about the
11.	Tbusr	Information about the User

RELATIONS IN THE DATABASE FOR BUGTRACKER

The following are the relations we have designed to manage the database. Here we have followed a convention of having the table names with tb as a prefix, and the remaining name of the table represent the description of the data inside that table.

Tbasg

Asgcod	Int	Null	Code
Asgbugcod	Int	Null	Code
Asgempcod	Int	Null	Code
Asgdat	Datetime	Null	Date
Asgesttim	Int	Null	Time
Aspaprsts	Char	Null	Status

Tbbug

Bugcod	Int	Null	Code
Bugprjcod	Int	Null	Code
Bugsevlel	Char(1)	Null	Level
Bugdsc	Varchar(500)	Null	
Bugscrsht	Varchar(50)	Null	
Bugurl	Varchar(50)	Null	
Bugpstdat	Datetime	Null	Date
Bugteccod	Int	Null	Code
Bugsts	Char(1)	Null	Status

Tbbugdep

Bugdepcod	Int	Null	Code
Bugdepbugcod	Int	Null	Code
Bugdepdepbugcod	Int	Null	Code
Bugdepaprsts	Char(1)	Null	Status

Tbbugres

Bugrescod	Int	Null	Code
Bugresasgcod	Int	Null	Code
Bugresdsc	Varchar(2000)	Null	
Bugresstrtim	Datetime	Null	Time
Bugresendtim	Datetime	Null	Time

Bugrescmt	Varchar(500)	Null	

Tbclt

Cltcod	Int	Null	Code
Cltnam	Varchar(50)	Null	Name
Clteml	Varchar(50)	Null	
Cltprf	Varchar(1000)	Null	Profile
Cltphn	Varchar(50)	Null	

Tbemp

Empcod	Int	Null	Code
Empnam	Varchar(50)	Null	Name
Empeml	Varchar(50)	Null	
Empteccod	Int	Null	Code

Tbmsg

Msgcod	Int	Null	Code
Msgsndusrcod	Int	Null	Code
Msgrecusrcod	Int	Null	Code
Msgdsc	Varchar(2000)	Null	
Msgdat	Datetime	Null	Date
Msgfil	Varchar(50)	Null	

Tbprj

Prjcod	Int	Null	Code
Prjnam	Varchar(100)	Null	Name
Prjdsc	Varchar(2000)	Null	
Prjcltcod	Int	Null	Code
Prjstrdat	Datetime	Null	Date
Prjenddat	Datetime	Null	Date

Tbprjtec

Prjteccod	Int	Null	Code
Prjtecprjcod	Int	Null	Code
Prjtecteccod	Int	Null	Code

Tbtec

Teccod	Int	Null	Code
Tecnam	Varchar(50)	Null	Name

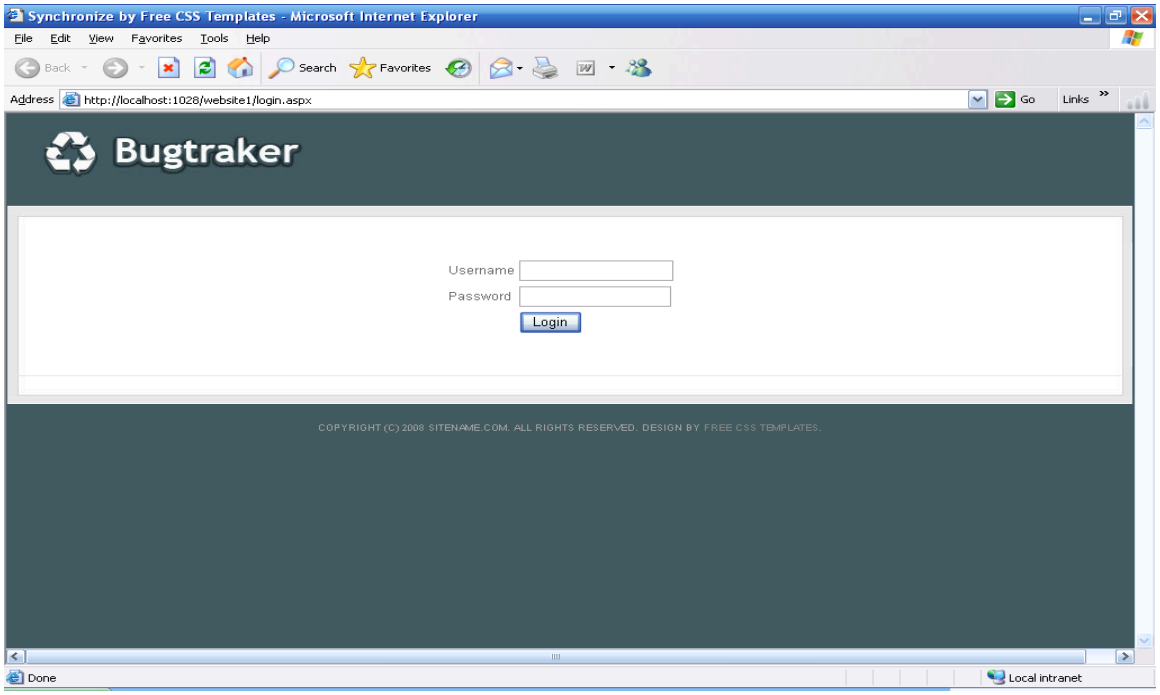
Tbusr

Usrcod	Int	Null	Code
Usrnam	Varchar(50)	Null	Name
Usrpwd	Varchar(50)	Null	Password
Usrcltempcod	Int	Null	Code

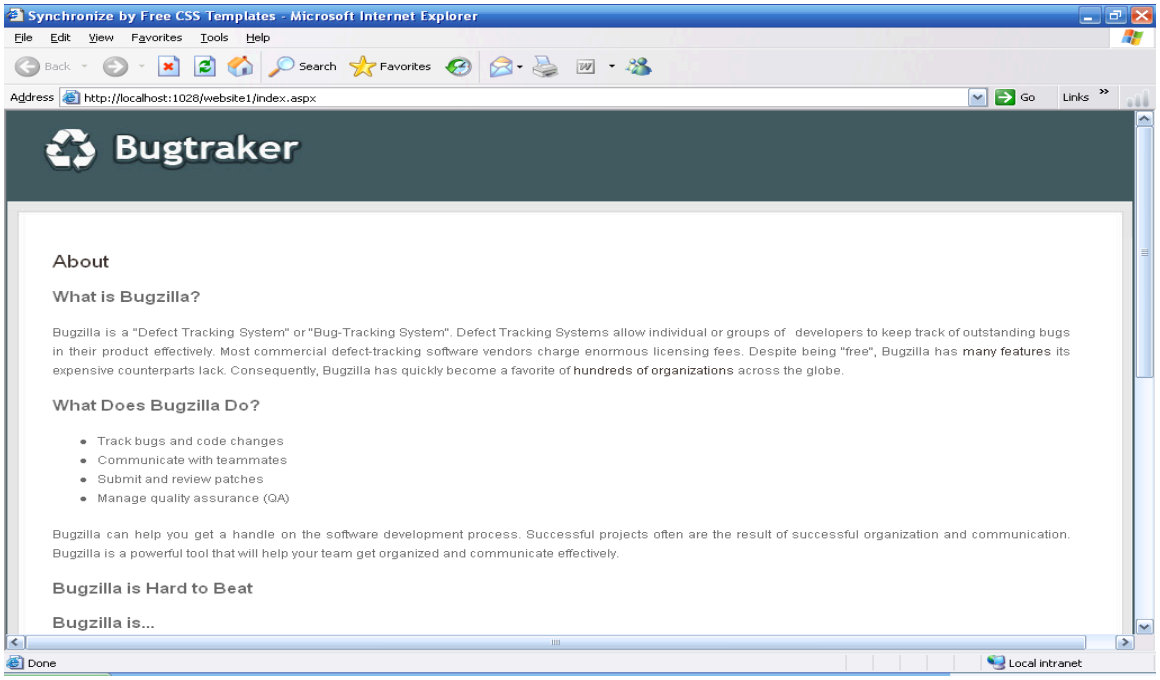
Usrrol	Char(1)	Null	

4.5 Input-Output Form(Screen Layout)

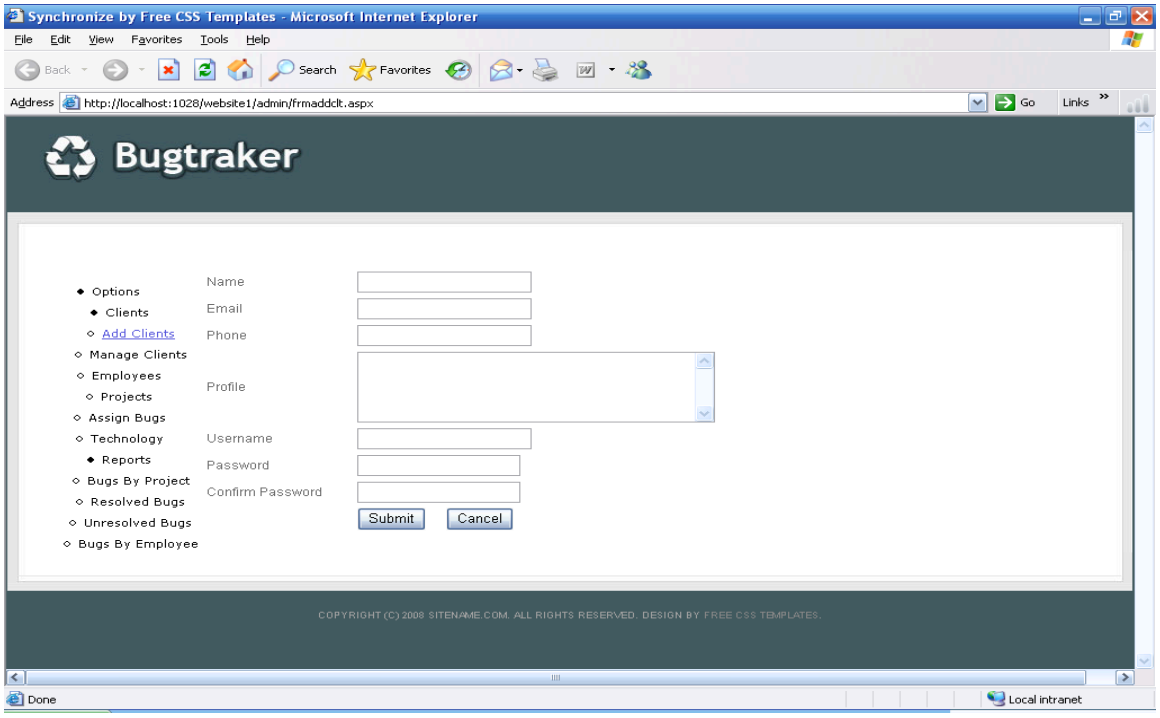
Login Page:



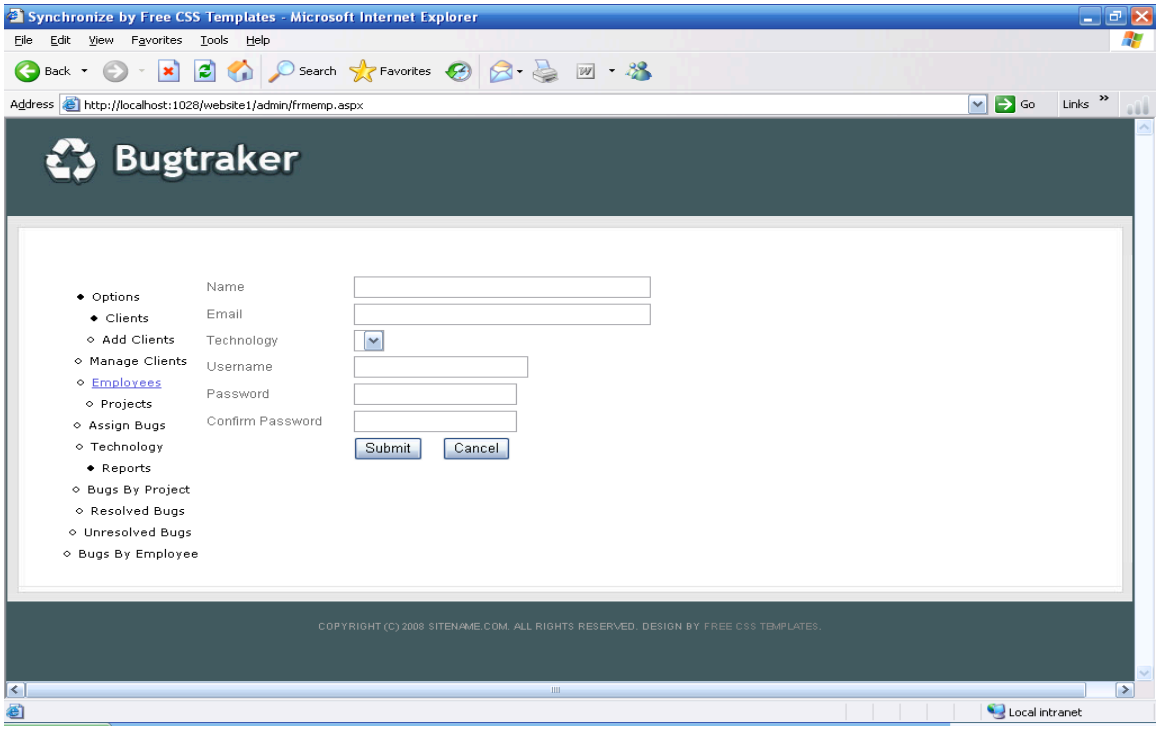
Index Page:



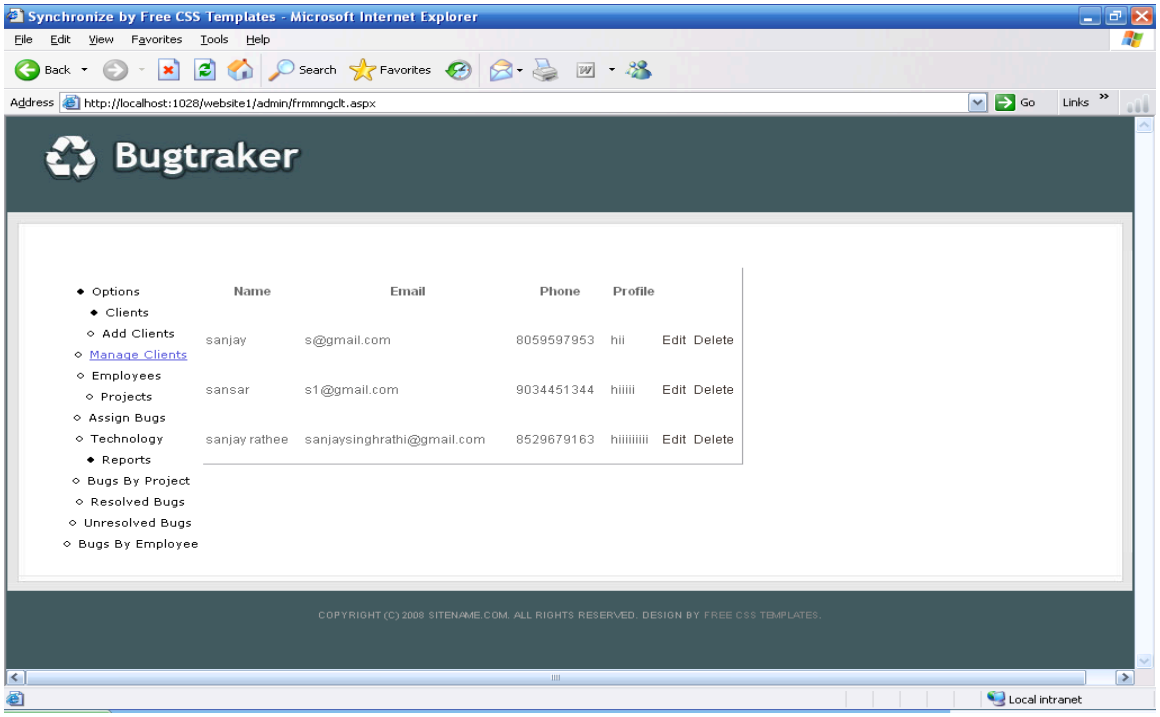
Admin:-
frmaddclt.aspx :



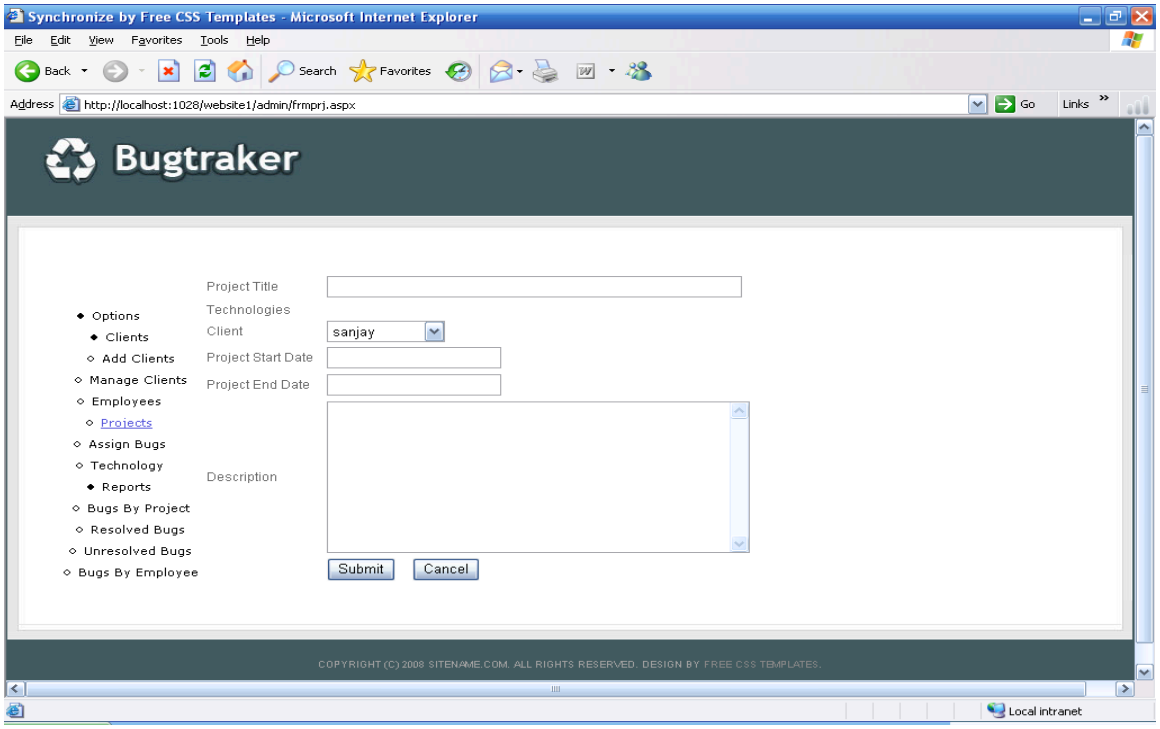
frmemp.aspx:



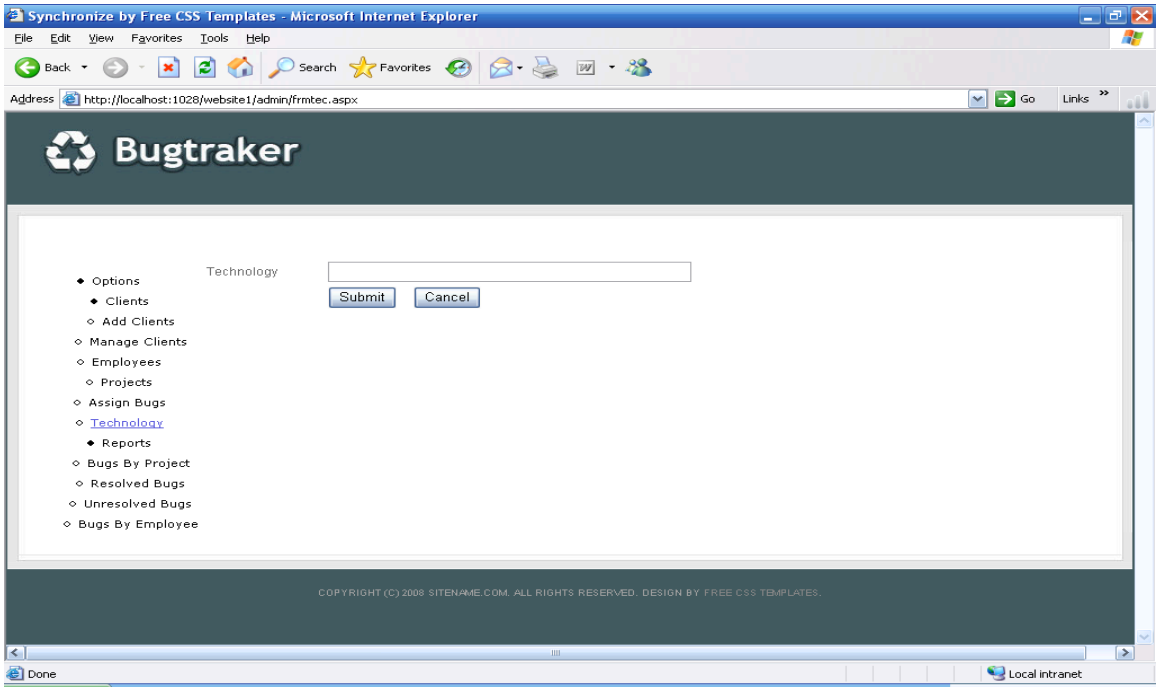
frmmngclt.aspx:



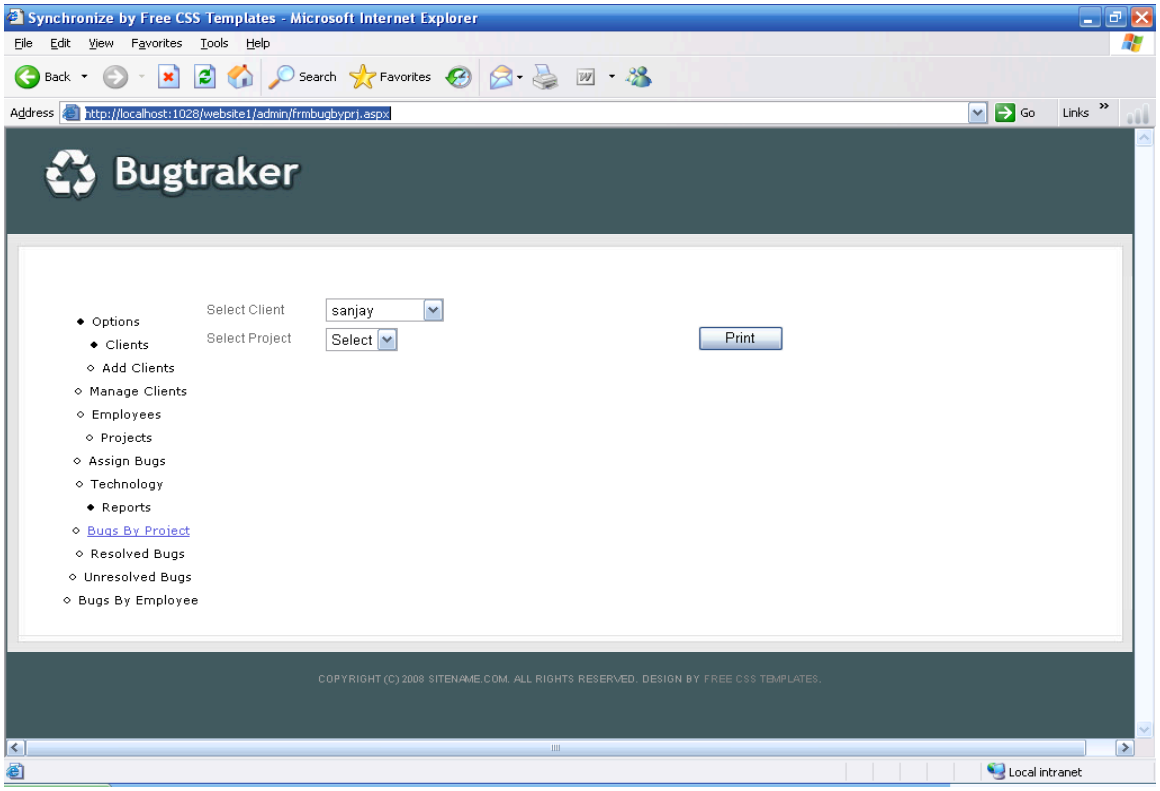
frmprj.aspx:



frmtec.aspx:

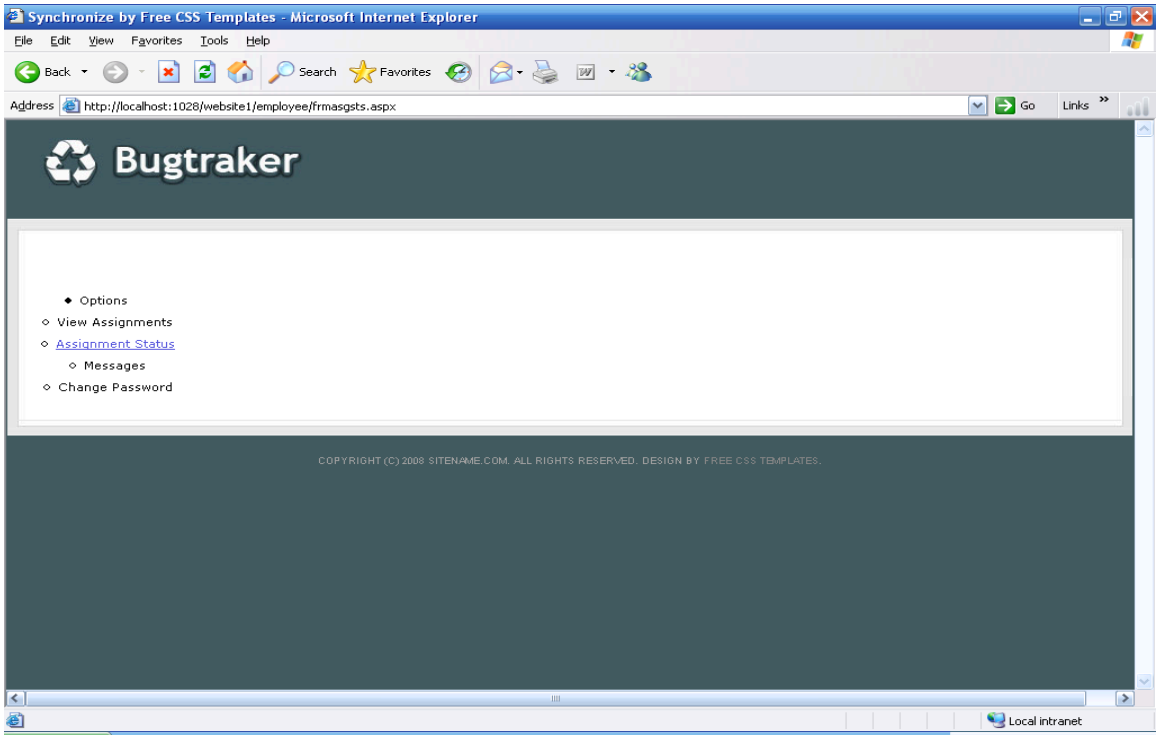


frmbugbyprj.aspx :-

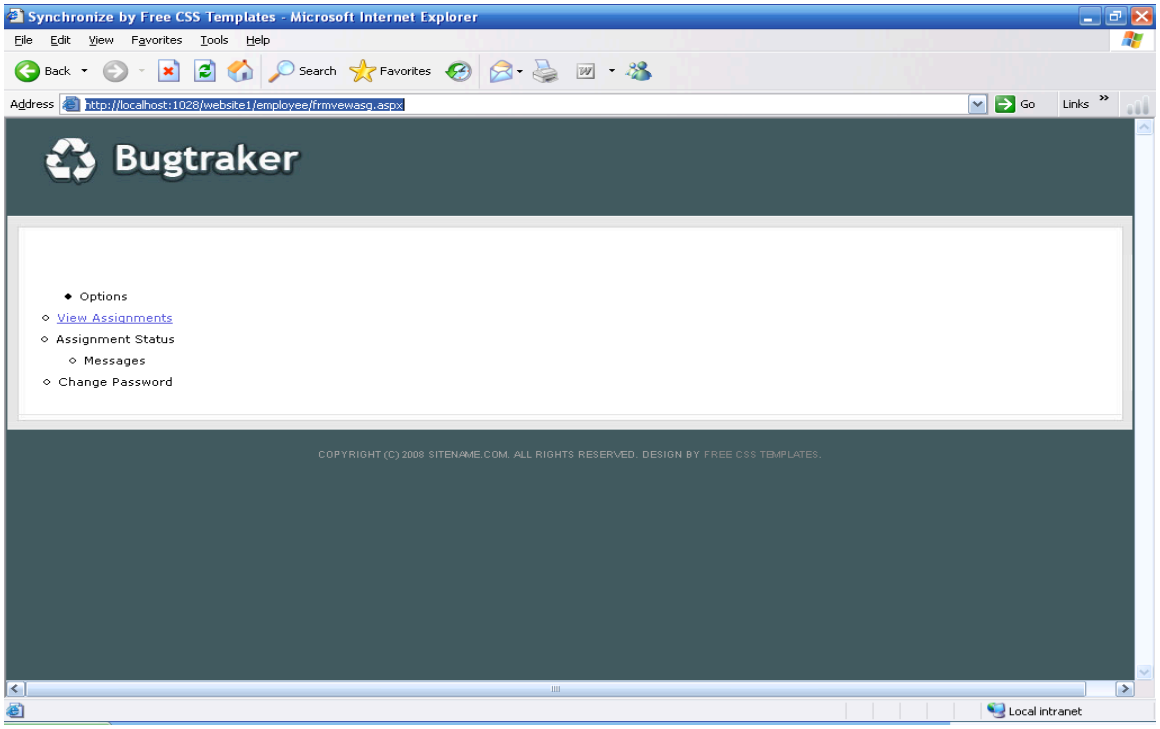


Employee page:-

Frmasgsts.aspx :-

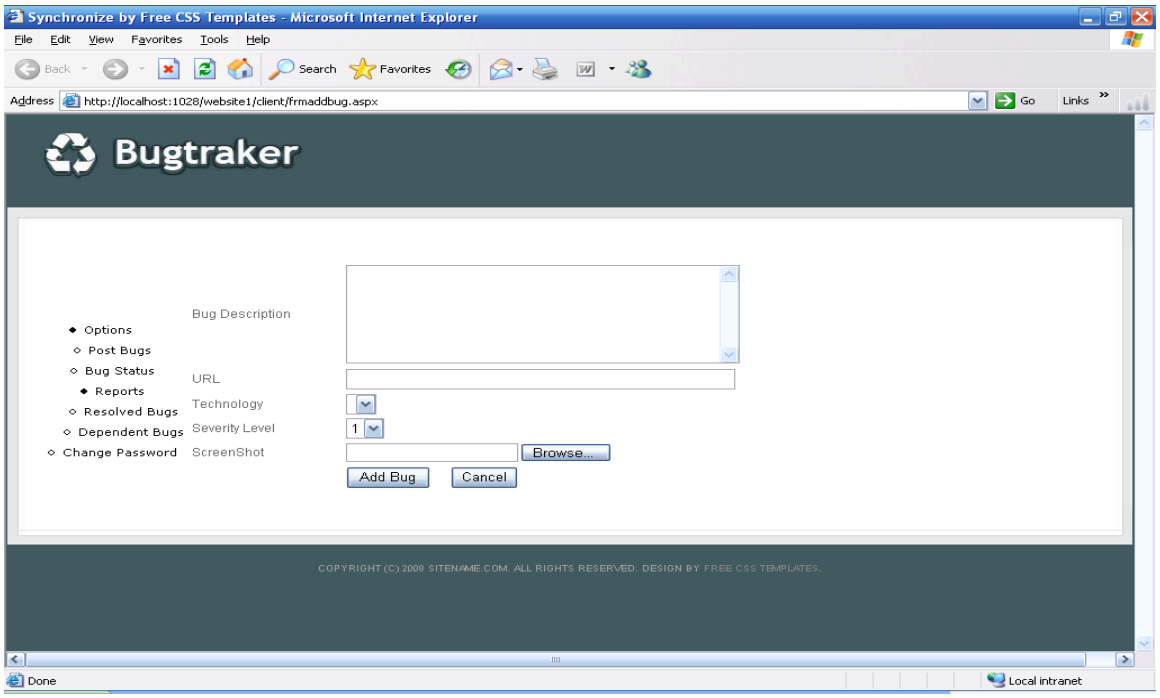


Frmvewasg.aspx :-

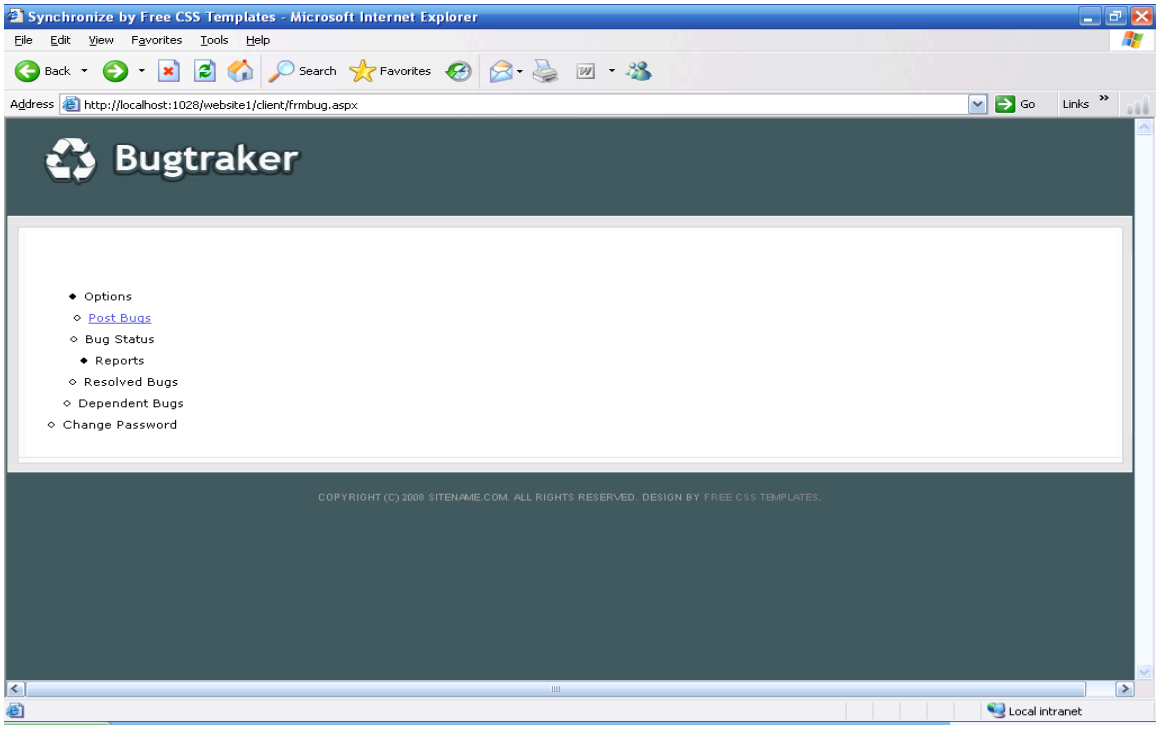


Client Page:-

Frmaddbug.aspx :-



Frmbug :-



5. Coding

Coding of web config file:-

```
<?xml version="1.0"?>
<configuration>
        <connectionStrings>
                <add name="cn" connectionString="Data
Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\dbbugtrk.mdf;Integrated
Security=True;User Instance=True" providerName="System.Data.SqlClient"/>
        </connectionStrings>
        <system.web>
                <compilation debug="true" targetFramework="4.0"/>
        </system.web>
</configuration>
```

Coding of class file:-

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;

namespace nsbugtrk                    //namespace BUGTRACKER
{
   public interface intclt            //interface for tbclt
   {
      Int32 cltcod
      {
         get;
         set;
      }
      String cltnam
      {
         get;
         set;
      }
      String clteml
      {
         get;
         set;
      }
      String cltprf
      {
         get;
         set;
      }
      String cltphn
      {
         get;
         set;
      }
   }
   public interface intprj            //interface for tbprj
   {
      Int32 prjcod
      {
         get;
         set;
      }
      String prjnam
      {
         get;
         set;
```

```
    }
    String prjdsc
    {
       get;
       set;
    }
    Int32 prjcltcod
    {
       get;
       set;
    }
    DateTime prjstrdat
    {
       get;
       set;
    }
    DateTime prjenddat
    {
       get;
       set;
    }
}
public interface inttec                //interface for tbtec
{
    Int32 teccod
    {
       get;
       set;
    }
    String tecnam
    {
       get;
       set;
    }
}
public interface intprjtec             //interface for tbprjtec
{
    Int32 prjteccod
    {
       get;
       set;
    }
    Int32 prjtecprjcod
    {
       get;
       set;
```

```
    }
    Int32 prjtecteccod
    {
        get;
        set;
    }
}
public interface intbug                 //interface for tbbug
{
    Int32 bugcod
    {
        get;
        set;
    }
    Int32 bugprjcod
    {
        get;
        set;
    }
    char bugsevlvl
    {
        get;
        set;
    }
    String bugdsc
    {
        get;
        set;
    }
    String bugscrsht
    {
        get;
        set;
    }
    String bugurl
    {
        get;
        set;
    }
    DateTime bugpstdat
    {
        get;
        set;
    }
    Int32 bugteccod
    {
```

```
        get;
        set;
    }
    char bugsts
    {
        get;
        set;
    }
}
public interface intemp                 //interface for tbemp
{
    Int32 empcod
    {
        get;
        set;
    }
    String empnam
    {
        get;
        set;
    }
    string empeml
    {
        get;
        set;
    }
    Int32 empteccod
    {
        get;
        set;
    }
}
public interface intasg                 //interface for tbasg
{
    Int32 asgcod
    {
        get;
        set;
    }
    Int32 asgbugcod
    {
        get;
        set;
    }
    Int32 asgempcod
    {
```

```
        get;
        set;
    }
    DateTime asgdat
    {
        get;
        set;
    }
    Int32 asgesttim
    {
        get;
        set;
    }
    char asgaprsts
    {
        get;
        set;
    }
}
public interface intbugres              //interface for tbbugres
{
    Int32 bugrescod
    {
        get;
        set;
    }
    Int32 bugresasgcod
    {
        get;
        set;
    }
    String bugresdsc
    {
        get;
        set;
    }
    DateTime bugresstrtim
    {
        get;
        set;
    }
    DateTime bugresendtim
    {
        get;
        set;
    }
```

```
    String bugrescmt
    {
       get;
       set;
    }
}
public interface intbugdep              //interface for tbbugdep
{
    Int32 bugdepcod
    {
       get;
       set;
    }
    Int32 bugdepbugcod
    {
       get;
       set;
    }
    string bugdepdsc
    {
       get;
       set;
    }
    char bugdepaprsts
    {
       get;
       set;
    }
}
public interface intusr                 //interface for tbusr
{
    Int32 usrcod
    {
       get;
       set;
    }
    String usrnam
    {
       get;
       set;
    }
    String usrpwd
    {
       get;
       set;
    }
```

```
    Int32 usrcltempcod
    {
       get;
       set;
    }
    char usrrol
    {
       get;
       set;
    }
}
public interface intmsg                //interface for tbmsg
{
    Int32 msgcod
    {
       get;
       set;
    }
    Int32 msgsndusrcod
    {
       get;
       set;
    }
    Int32 msgrecusrcod
    {
       get;
       set;
    }
    String msgdsc
    {
       get;
       set;
    }
    DateTime msgdat
    {
       get;
       set;
    }
    String msgfil
    {
       get;
       set;
    }
}
public class clscltprp : intclt        //property class of tbclt
{
```

```
private Int32 prvcltcod;
private String prvcltnam, prvclteml, prvcltprf, prvcltphn;
public int cltcod
{
   get
   {
      return prvcltcod;
   }
   set
   {
      prvcltcod = value;
   }
}
public string cltnam
{
   get
   {
      return prvcltnam;
   }
   set
   {
      prvcltnam = value;
   }
}
public string clteml
{
   get
   {
      return prvclteml;
   }
   set
   {
      prvclteml = value;
   }
}
public string cltprf
{
   get
   {
      return prvcltprf;
   }
   set
   {
      prvcltprf = value;
   }
}
```

```
    public string cltphn
    {
        get
        {
            return prvcltphn;
        }
        set
        {
            prvcltphn = value;
        }
    }
}
public class clsprjprp : intprj         //property class of tbprj
{
    private Int32 prvprjcod, prvprjcltcod;
    private String prvprjnam, prvprjdsc;
    private DateTime prvprjstrdat, prvprjenddat;
    public int prjcod
    {
        get
        {
            return prvprjcod;
        }
        set
        {
            prvprjcod = value;
        }
    }
    public string prjnam
    {
        get
        {
            return prvprjnam;
        }
        set
        {
            prvprjnam = value;
        }
    }
    public string prjdsc
    {
        get
        {
            return prvprjdsc;
        }
        set
```

```
        {
            prvprjdsc = value;
        }
    }
    public int prjcltcod
    {
        get
        {
            return prvprjcltcod;
        }
        set
        {
            prvprjcltcod = value;
        }
    }
    public DateTime prjstrdat
    {
        get
        {
            return prvprjstrdat;
        }
        set
        {
            prvprjstrdat = value;
        }
    }
    public DateTime prjenddat
    {
        get
        {
            return prvprjenddat;
        }
        set
        {
            prvprjenddat = value;
        }
    }
}
public class clstecprp : inttec        //property class of tbtec
{
    private Int32 prvteccod;
    private String prvtecnam;
    public int teccod
    {
        get
        {
```

```
            return prvteccod;
        }
        set
        {
            prvteccod = value;
        }
    }
    public string tecnam
    {
        get
        {
            return prvtecnam;
        }
        set
        {
            prvtecnam = value;
        }
    }
}
public class clsprjtecprp : intprjtec   //property class of tbprjtec
{
    private Int32 prvprjteccod, prvprjtecprjcod, prvprjtecteccod;
    public int prjteccod
    {
        get
        {
            return prvprjteccod;
        }
        set
        {
            prvprjteccod = value;
        }
    }
    public int prjtecprjcod
    {
        get
        {
            return prvprjtecprjcod;
        }
        set
        {
            prvprjtecprjcod = value;
        }
    }
    public int prjtecteccod
    {
```

```
        get
        {
            return prvprjtecteccod;
        }
        set
        {
            prvprjtecteccod = value;
        }
    }
}
public class clsbugprp : intbug          //property class of tbbug
{
    private Int32 prvbugcod, prvbugprjcod, prvbugteccod;
    private String prvbugdsc, prvbugscrsht, prvbugurl;
    private char prvbugsevlvl, prvbugsts;
    private DateTime prvbugpstdat;
    public int bugcod
    {
        get
        {
            return prvbugcod;
        }
        set
        {
            prvbugcod = value;
        }
    }
    public int bugprjcod
    {
        get
        {
            return prvbugprjcod;
        }
        set
        {
            prvbugprjcod = value;
        }
    }
    public char bugsevlvl
    {
        get
        {
            return prvbugsevlvl;
        }
        set
        {
```

```
        prvbugsevlvl = value;
    }
}
public string bugdsc
{
    get
    {
        return prvbugdsc;
    }
    set
    {
        prvbugdsc = value;
    }
}
public string bugscrsht
{
    get
    {
        return prvbugscrsht;
    }
    set
    {
        prvbugscrsht = value;
    }
}
public string bugurl
{
    get
    {
        return prvbugurl;
    }
    set
    {
        prvbugurl = value;
    }
}
public DateTime bugpstdat
{
    get
    {
        return prvbugpstdat;
    }
    set
    {
        prvbugpstdat = value;
    }
```

```
    }
    public int bugteccod
    {
       get
       {
          return prvbugteccod;
       }
       set
       {
          prvbugteccod = value;
       }
    }
    public char bugsts
    {
       get
       {
          return prvbugsts;
       }
       set
       {
          prvbugsts = value;
       }
    }
}
public class clsempprp : intemp          //property class of tbemp
{
    private Int32 prvempcod, prvempteccod;
    private String prvempnam, prvempeml;
    public int empcod
    {
       get
       {
          return prvempcod;
       }
       set
       {
          prvempcod = value;
       }
    }
    public string empnam
    {
       get
       {
          return prvempnam;
       }
       set
```

```
        {
            prvempnam = value;
        }
    }
    public string empeml
    {
        get
        {
            return prvempeml;
        }
        set
        {
            prvempeml = value;
        }
    }
    public int empteccod
    {
        get
        {
            return prvempteccod;
        }
        set
        {
            prvempteccod = value;
        }
    }
}
public class clsasgprp : intasg          //property class of tbasg
{
    private Int32 prvasgcod, prvasgbugcod, prvasgempcod, prvasgesttim;
    private DateTime prvasgdat;
    private char prvasgaprsts;
    public int asgcod
    {
        get
        {
            return prvasgcod;
        }
        set
        {
            prvasgcod = value;
        }
    }
    public int asgbugcod
    {
        get
```

```
    {
        return prvasgbugcod;
    }
    set
    {
        prvasgbugcod = value;
    }
}
public int asgempcod
{
    get
    {
        return prvasgempcod;
    }
    set
    {
        prvasgempcod = value;
    }
}
public DateTime asgdat
{
    get
    {
        return prvasgdat;
    }
    set
    {
        prvasgdat = value;
    }
}
public int asgesttim
{
    get
    {
        return prvasgesttim;
    }
    set
    {
        prvasgesttim = value;
    }
}
public char asgaprsts
{
    get
    {
        return prvasgaprsts;
```

```
        }
        set
        {
            prvasgaprsts = value;
        }
    }
}
public class clsbugresprp : intbugres   //property class of tbbugres
{
    private Int32 prvbugrescod, prvbugresasgcod;
    private String prvbugresdsc, prvbugrescmt;
    private DateTime prvbugresstrtim, prvbugresendtim;
    public int bugrescod
    {
        get
        {
            return prvbugrescod;
        }
        set
        {
            prvbugrescod = value;
        }
    }
    public int bugresasgcod
    {
        get
        {
            return prvbugresasgcod;
        }
        set
        {
            prvbugresasgcod = value;
        }
    }
    public string bugresdsc
    {
        get
        {
            return prvbugresdsc;
        }
        set
        {
            prvbugresdsc = value;
        }
    }
    public DateTime bugresstrtim
```

```
    {
        get
        {
            return prvbugresstrtim;
        }
        set
        {
            prvbugresstrtim = value;
        }
    }
    public DateTime bugresendtim
    {
        get
        {
            return prvbugresendtim;
        }
        set
        {
            prvbugresendtim = value;
        }
    }
    public String bugrescmt
    {
        get
        {
            return prvbugrescmt;
        }
        set
        {
            prvbugrescmt = value;
        }
    }
}
public class clsbugdepprp : intbugdep   //property class of tbbugdep
{
    private Int32 prvbugdepcod, prvbugdepbugcod;
    string prvbugdepdsc;
    private char prvbugdepaprsts;
    public int bugdepcod
    {
        get
        {
            return prvbugdepcod;
        }
        set
        {
```

```
            prvbugdepcod = value;
        }
    }
    public int bugdepbugcod
    {
        get
        {
            return prvbugdepbugcod;
        }
        set
        {
            prvbugdepbugcod = value;
        }
    }
    public string bugdepdsc
    {
        get
        {
            return prvbugdepdsc;
        }
        set
        {
            prvbugdepdsc = value;
        }
    }
    public char bugdepaprsts
    {
        get
        {
            return prvbugdepaprsts;
        }
        set
        {
            prvbugdepaprsts = value;
        }
    }
}
public class clsusrprp : intusr         //property class of tbusr
{
    private Int32 prvusrcod, prvusrcltempcod;
    private String prvusrnam, prvusrpwd;
    private char prvusrrol;
    public int usrcod
    {
        get
        {
```

```
        return prvusrcod;
    }
    set
    {
        prvusrcod = value;
    }
}
public string usrnam
{
    get
    {
        return prvusrnam;
    }
    set
    {
        prvusrnam = value;
    }
}
public string usrpwd
{
    get
    {
        return prvusrpwd;
    }
    set
    {
        prvusrpwd = value;
    }
}
public int usrcltempcod
{
    get
    {
        return prvusrcltempcod;
    }
    set
    {
        prvusrcltempcod = value;
    }
}
public char usrrol
{
    get
    {
        return prvusrrol;
    }
```

```
        set
        {
            prvusrrol = value;
        }
    }
}
public class clsmsgprp : intmsg          //property class of tbmsg
{
    private Int32 prvmsgcod, prvmsgsndusrcod, prvmsgrecusrcod;
    private String prvmsgdsc, prvmsgfil;
    private DateTime prvmsgdat;
    public int msgcod
    {
        get
        {
            return prvmsgcod;
        }
        set
        {
            prvmsgcod = value;
        }
    }
    public int msgsndusrcod
    {
        get
        {
            return prvmsgsndusrcod;
        }
        set
        {
            prvmsgsndusrcod = value;
        }
    }
    public int msgrecusrcod
    {
        get
        {
            return prvmsgrecusrcod;
        }
        set
        {
            prvmsgrecusrcod = value;
        }
    }
    public string msgdsc
    {
```

```
            get
            {
                return prvmsgdsc;
            }
            set
            {
                prvmsgdsc = value;
            }
        }
        public DateTime msgdat
        {
            get
            {
                return prvmsgdat;
            }
            set
            {
                prvmsgdat = value;
            }
        }
        public string msgfil
        {
            get
            {
                return prvmsgfil;
            }
            set
            {
                prvmsgfil = value;
            }
        }
    }
    public abstract class clscon            //connection class
    {
        protected SqlConnection con = new SqlConnection();
        public clscon()
        {
            con.ConnectionString =
ConfigurationManager.ConnectionStrings["cn"].ConnectionString;
        }
    }
    public class clsclt : clscon            //manipulations on tbclt
    {
        public Int32 getauto()
        {
```

```
        SqlDataAdapter adp = new SqlDataAdapter("select isnull(max(cltcod),0) from
tbclt", con);
        DataSet ds = new DataSet();
        adp.Fill(ds);
        Int32 a = Convert.ToInt32(ds.Tables[0].Rows[0][0]);
        return a + 1;
    }
    public void save_rec(clscltprp p)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("insclt", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@cltcod", SqlDbType.Int).Value = p.cltcod;
        cmd.Parameters.Add("@cltnam", SqlDbType.VarChar, 50).Value = p.cltnam;
        cmd.Parameters.Add("@clteml", SqlDbType.VarChar, 50).Value = p.clteml;
        cmd.Parameters.Add("@cltprf", SqlDbType.VarChar, 1000).Value = p.cltprf;
        cmd.Parameters.Add("@cltphn", SqlDbType.VarChar, 50).Value = p.cltphn;
        cmd.ExecuteNonQuery();
        cmd.Dispose();
        con.Close();
    }
    public void update_rec(clscltprp p)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("updclt", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@cltcod", SqlDbType.Int).Value = p.cltcod;
        cmd.Parameters.Add("@cltnam", SqlDbType.VarChar, 50).Value = p.cltnam;
        cmd.Parameters.Add("@clteml", SqlDbType.VarChar, 50).Value = p.clteml;
        cmd.Parameters.Add("@cltprf", SqlDbType.VarChar, 1000).Value = p.cltprf;
        cmd.Parameters.Add("@cltphn", SqlDbType.VarChar, 50).Value = p.cltphn;
        cmd.ExecuteNonQuery();
        cmd.Dispose();
        con.Close();
    }
    public void delete_rec(clscltprp p)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
```

```
    }
    SqlCommand cmd = new SqlCommand("delclt", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@cltcod", SqlDbType.Int).Value = p.cltcod;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public List<clscltprp> disp_rec()
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("dspclt", con);
    cmd.CommandType = CommandType.StoredProcedure;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clscltprp> obj = new List<clscltprp>();
    while (dr.Read())
    {
        clscltprp k = new clscltprp();
        k.cltcod = Convert.ToInt32(dr[0]);
        k.cltnam = dr[1].ToString();
        k.clteml = dr[2].ToString();
        k.cltprf = dr[3].ToString();
        k.cltphn = dr[4].ToString();
        obj.Add(k);
    }
    dr.Close();
    cmd.Dispose();
    con.Close();
    return obj;
}
public List<clscltprp> find_rec(int pk)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("fndclt", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@cltcod", SqlDbType.Int).Value = pk;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clscltprp> obj = new List<clscltprp>();
    if (dr.HasRows)
    {
```

```
            dr.Read();
            clscltprp k = new clscltprp();
            k.cltcod = Convert.ToInt32(dr[0]);
            k.cltnam = dr[1].ToString();
            k.clteml = dr[2].ToString();
            k.cltprf = dr[3].ToString();
            k.cltphn = dr[4].ToString();
            obj.Add(k);
         }
         dr.Close();
         cmd.Dispose();
         con.Close();
         return obj;
      }
   }
   public class clsprj : clscon          //manipulations on tbprj
   {
      public Int32 getauto()
      {
         SqlDataAdapter adp = new SqlDataAdapter("select isnull(max(prjcod),0) from
tbprj", con);
         DataSet ds = new DataSet();
         adp.Fill(ds);
         Int32 a = Convert.ToInt32(ds.Tables[0].Rows[0][0]);
         return a + 1;
      }
      public void save_rec(clsprjprp p)
      {
         if (con.State == ConnectionState.Closed)
         {
            con.Open();
         }
         SqlCommand cmd = new SqlCommand("insprj", con);
         cmd.CommandType = CommandType.StoredProcedure;
         cmd.Parameters.Add("@prjcod", SqlDbType.Int).Value = p.prjcod;
         cmd.Parameters.Add("@prjnam", SqlDbType.VarChar, 100).Value = p.prjnam;
         cmd.Parameters.Add("@prjdsc", SqlDbType.VarChar, 2000).Value = p.prjdsc;
         cmd.Parameters.Add("@prjcltcod", SqlDbType.Int).Value = p.prjcltcod;
         cmd.Parameters.Add("@prjstrdat", SqlDbType.DateTime).Value = p.prjstrdat;
         cmd.Parameters.Add("@prjenddat", SqlDbType.DateTime).Value = p.prjenddat;
         cmd.ExecuteNonQuery();
         cmd.Dispose();
         con.Close();
      }
      public void update_rec(clsprjprp p)
      {
```

```
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("updprj", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@prjcod", SqlDbType.Int).Value = p.prjcod;
    cmd.Parameters.Add("@prjnam", SqlDbType.VarChar, 100).Value = p.prjnam;
    cmd.Parameters.Add("@prjdsc", SqlDbType.VarChar, 2000).Value = p.prjdsc;
    cmd.Parameters.Add("@prjcltcod", SqlDbType.Int).Value = p.prjcltcod;
    cmd.Parameters.Add("@prjstrdat", SqlDbType.DateTime).Value = p.prjstrdat;
    cmd.Parameters.Add("@prjenddat", SqlDbType.DateTime).Value = p.prjenddat;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public void delete_rec(clsprjprp p)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("delprj", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@prjcod", SqlDbType.Int).Value = p.prjcod;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public List<clsprjprp> disp_rec(Int32 cltcod)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("dspprjbyclt", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@cltcod", SqlDbType.Int).Value = cltcod;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clsprjprp> obj = new List<clsprjprp>();
    while (dr.Read())
    {
        clsprjprp k = new clsprjprp();
        k.prjcod = Convert.ToInt32(dr[0]);
        k.prjnam = dr[1].ToString();
        k.prjdsc = dr[2].ToString();
```

```
            k.prjcltcod = Convert.ToInt32(dr[3]);
            k.prjstrdat = Convert.ToDateTime(dr[4]);
            k.prjenddat = Convert.ToDateTime(dr[5]);
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
    public List<clsprjprp> find_rec(int pk)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("fndprj", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@prjcod", SqlDbType.Int).Value = pk;
        SqlDataReader dr = cmd.ExecuteReader();
        List<clsprjprp> obj = new List<clsprjprp>();
        if (dr.HasRows)
        {
            dr.Read();
            clsprjprp k = new clsprjprp();
            k.prjcod = Convert.ToInt32(dr[0]);
            k.prjnam = dr[1].ToString();
            k.prjdsc = dr[2].ToString();
            k.prjcltcod = Convert.ToInt32(dr[3]);
            k.prjstrdat = Convert.ToDateTime(dr[4]);
            k.prjenddat = Convert.ToDateTime(dr[5]);
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
}
public class clstec : clscon            //manipulations on tbtec
{
    public void save_rec(clstecprp p)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
```

```
    }
    SqlCommand cmd = new SqlCommand("instec", con);
    cmd.CommandType = CommandType.StoredProcedure;
//    cmd.Parameters.Add("@teccod", SqlDbType.Int).Value = p.teccod;
    cmd.Parameters.Add("@tecnam", SqlDbType.VarChar, 50).Value = p.tecnam;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
 }
 public void update_rec(clstecprp p)
 {
    if (con.State == ConnectionState.Closed)
    {
       con.Open();
    }
    SqlCommand cmd = new SqlCommand("updtec", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@teccod", SqlDbType.Int).Value = p.teccod;
    cmd.Parameters.Add("@tecnam", SqlDbType.VarChar, 50).Value = p.tecnam;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
 }
 public void delete_rec(clstecprp p)
 {
    if (con.State == ConnectionState.Closed)
    {
       con.Open();
    }
    SqlCommand cmd = new SqlCommand("deltec", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@teccod", SqlDbType.Int).Value = p.teccod;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
 }
 public List<clstecprp> disp_rec()
 {
    if (con.State == ConnectionState.Closed)
    {
       con.Open();
    }
    SqlCommand cmd = new SqlCommand("dsptec", con);
    cmd.CommandType = CommandType.StoredProcedure;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clstecprp> obj = new List<clstecprp>();
```

```
        while (dr.Read())
        {
            clstecprp k = new clstecprp();
            k.teccod = Convert.ToInt32(dr[0]);
            k.tecnam = dr[1].ToString();
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
    public List<clstecprp> find_rec(int pk)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("fndtec", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@teccod", SqlDbType.Int).Value = pk;
        SqlDataReader dr = cmd.ExecuteReader();
        List<clstecprp> obj = new List<clstecprp>();
        if (dr.HasRows)
        {
            dr.Read();
            clstecprp k = new clstecprp();
            k.teccod = Convert.ToInt32(dr[0]);
            k.tecnam = dr[1].ToString();
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
}
public class clsprjtec : clscon          //manipulations on tbprjtec
{
    public void save_rec(clsprjtecprp p)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("insprjtec", con);
```

```
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@prjteccod", SqlDbType.Int).Value = p.prjteccod;
    cmd.Parameters.Add("@prjtecprjcod", SqlDbType.Int).Value = p.prjtecprjcod;
    cmd.Parameters.Add("@prjtecteccod", SqlDbType.Int).Value = p.prjtecteccod;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public void update_rec(clsprjtecprp p)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("updprjtec", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@prjteccod", SqlDbType.Int).Value = p.prjteccod;
    cmd.Parameters.Add("@prjtecprjcod", SqlDbType.Int).Value = p.prjtecprjcod;
    cmd.Parameters.Add("@prjtecteccod", SqlDbType.Int).Value = p.prjtecteccod;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public void delete_rec(clsprjtecprp p)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("delprjtec", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@prjteccod", SqlDbType.Int).Value = p.prjteccod;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public List<clsprjtecprp> disp_rec()
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("dspprjtec", con);
    cmd.CommandType = CommandType.StoredProcedure;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clsprjtecprp> obj = new List<clsprjtecprp>();
```

```
        while (dr.Read())
        {
            dr.Read();
            clsprjtecprp k = new clsprjtecprp();
            k.prjteccod = Convert.ToInt32(dr[0]);
            k.prjtecprjcod = Convert.ToInt32(dr[1]);
            k.prjtecteccod = Convert.ToInt32(dr[2]);
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
    public List<clsprjtecprp> find_rec(int pk)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("fndprjtec", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@prjteccod", SqlDbType.Int).Value = pk;
        SqlDataReader dr = cmd.ExecuteReader();
        List<clsprjtecprp> obj = new List<clsprjtecprp>();
        if (dr.HasRows)
        {
            dr.Read();
            clsprjtecprp k = new clsprjtecprp();
            k.prjteccod = Convert.ToInt32(dr[0]);
            k.prjtecprjcod = Convert.ToInt32(dr[1]);
            k.prjtecteccod = Convert.ToInt32(dr[2]);
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
}
public class clsbug : clscon            //manipulations on tbbug
{
    public void updbugsts(clsbugprp p)
    {
        if (con.State == ConnectionState.Closed)
            con.Open();
```

```
        SqlCommand cmd = new SqlCommand("updbugsts", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@bugcod", SqlDbType.Int).Value = p.bugcod;
        cmd.Parameters.Add("@bugsts", SqlDbType.Char, 1).Value = p.bugsts;
        cmd.ExecuteNonQuery();
        cmd.Dispose();
        con.Close();
    }
    public DataSet dspbugbyprj(Int32 prjcod)
    {
        SqlDataAdapter adp = new SqlDataAdapter("dspbugbyprj", con);
        adp.SelectCommand.CommandType = CommandType.StoredProcedure;
        adp.SelectCommand.Parameters.Add("@prjcod", SqlDbType.Int).Value = prjcod;
        DataSet ds = new DataSet();
        adp.Fill(ds);
        return ds;
    }
    public DataSet dspallbugbyprj(Int32 prjcod)
    {
        SqlDataAdapter adp = new SqlDataAdapter("dspallbugbyprj", con);
        adp.SelectCommand.CommandType = CommandType.StoredProcedure;
        adp.SelectCommand.Parameters.Add("@prjcod", SqlDbType.Int).Value = prjcod;
        DataSet ds = new DataSet();
        adp.Fill(ds);
        return ds;
    }
    public Int32 getauto()
    {
        SqlDataAdapter adp = new SqlDataAdapter("select isnull(max(bugcod),0) from
tbbug", con);
        DataSet ds = new DataSet();
        adp.Fill(ds);
        Int32 a = Convert.ToInt32(ds.Tables[0].Rows[0][0]);
        return a + 1;
    }
    public void save_rec(clsbugprp p)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("insbug", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@bugcod", SqlDbType.Int).Value = p.bugcod;
        cmd.Parameters.Add("@bugprjcod", SqlDbType.Int).Value = p.bugprjcod;
        cmd.Parameters.Add("@bugsevlvl", SqlDbType.Char, 1).Value = p.bugsevlvl;
```

```
            cmd.Parameters.Add("@bugdsc", SqlDbType.VarChar, 500).Value = p.bugdsc;
            cmd.Parameters.Add("@bugscrsht", SqlDbType.VarChar, 50).Value =
p.bugscrsht;
            cmd.Parameters.Add("@bugurl", SqlDbType.VarChar, 50).Value = p.bugurl;
            cmd.Parameters.Add("@bugpstdat", SqlDbType.DateTime).Value = p.bugpstdat;
            cmd.Parameters.Add("@bugteccod", SqlDbType.Int).Value = p.bugteccod;
            cmd.Parameters.Add("@bugsts", SqlDbType.Char, 1).Value = p.bugsts;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }
        public void update_rec(clsbugprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("updbug", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@bugcod", SqlDbType.Int).Value = p.bugcod;
            cmd.Parameters.Add("@bugprjcod", SqlDbType.Int).Value = p.bugprjcod;
            cmd.Parameters.Add("@bugsevlvl", SqlDbType.Char, 1).Value = p.bugsevlvl;
            cmd.Parameters.Add("@bugdsc", SqlDbType.VarChar, 500).Value = p.bugdsc;
            cmd.Parameters.Add("@bugscrsht", SqlDbType.VarChar, 50).Value =
p.bugscrsht;
            cmd.Parameters.Add("@bugurl", SqlDbType.VarChar, 50).Value = p.bugurl;
            cmd.Parameters.Add("@bugpstdat", SqlDbType.DateTime).Value = p.bugpstdat;
            cmd.Parameters.Add("@bugsts", SqlDbType.Char, 1).Value = p.bugsts;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }
        public void delete_rec(clsbugprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("delbug", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@bugcod", SqlDbType.Int).Value = p.bugcod;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }
        public List<clsbugprp> disp_rec()
```

```
{
   if (con.State == ConnectionState.Closed)
   {
      con.Open();
   }
   SqlCommand cmd = new SqlCommand("dspbug", con);
   cmd.CommandType = CommandType.StoredProcedure;
   SqlDataReader dr = cmd.ExecuteReader();
   List<clsbugprp> obj = new List<clsbugprp>();
   while (dr.Read())
   {
      clsbugprp k = new clsbugprp();
      k.bugcod = Convert.ToInt32(dr[0]);
      k.bugprjcod = Convert.ToInt32(dr[1]);
      k.bugsevlvl = Convert.ToChar(dr[2]);
      k.bugdsc = dr[3].ToString();
      k.bugscrsht = dr[4].ToString();
      k.bugurl = dr[4].ToString();
      k.bugpstdat = Convert.ToDateTime(dr[5]);
      k.bugteccod = Convert.ToInt32(dr[6]);
      k.bugsts = Convert.ToChar(dr[7]);
      obj.Add(k);
   }
   dr.Close();
   cmd.Dispose();
   con.Close();
   return obj;
}
public List<clsbugprp> find_rec(int pk)
{
   if (con.State == ConnectionState.Closed)
   {
      con.Open();
   }
   SqlCommand cmd = new SqlCommand("fndbug", con);
   cmd.CommandType = CommandType.StoredProcedure;
   cmd.Parameters.Add("@bugcod", SqlDbType.Int).Value = pk;
   SqlDataReader dr = cmd.ExecuteReader();
   List<clsbugprp> obj = new List<clsbugprp>();
   if (dr.HasRows)
   {
      dr.Read();
      clsbugprp k = new clsbugprp();
      k.bugcod = Convert.ToInt32(dr[0]);
      k.bugprjcod = Convert.ToInt32(dr[1]);
      k.bugsevlvl = Convert.ToChar(dr[2]);
```

```
            k.bugdsc = dr[3].ToString();
            k.bugscrsht = dr[4].ToString();
            k.bugurl = dr[5].ToString();
            k.bugpstdat = Convert.ToDateTime(dr[6]);
            k.bugteccod = Convert.ToInt32(dr[7]);
            k.bugsts = Convert.ToChar(dr[8]);
            obj.Add(k);
          }
          dr.Close();
          cmd.Dispose();
          con.Close();
          return obj;
        }
    }
    public class clsemp : clscon            //manipulations on tbemp
    {
        public DataSet dspempbytec(Int32 teccod)
        {
          SqlDataAdapter adp = new SqlDataAdapter("dspempbytec", con);
          adp.SelectCommand.CommandType = CommandType.StoredProcedure;
          adp.SelectCommand.Parameters.Add("@teccod", SqlDbType.Int).Value = teccod;
          DataSet ds = new DataSet();
          adp.Fill(ds);
          return ds;
        }
        public Int32 getauto()
        {
          SqlDataAdapter adp = new SqlDataAdapter("select isnull(max(empcod),0) from
tbemp", con);
          DataSet ds = new DataSet();
          adp.Fill(ds);
          Int32 a = Convert.ToInt32(ds.Tables[0].Rows[0][0]);
          return a + 1;
        }
        public void save_rec(clsempprp p)
        {
          if (con.State == ConnectionState.Closed)
          {
            con.Open();
          }
          SqlCommand cmd = new SqlCommand("insemp", con);
          cmd.CommandType = CommandType.StoredProcedure;
          cmd.Parameters.Add("@empcod", SqlDbType.Int).Value = p.empcod;
          cmd.Parameters.Add("@empnam", SqlDbType.VarChar, 50).Value = p.empnam;
          cmd.Parameters.Add("@empeml", SqlDbType.VarChar, 50).Value = p.empeml;
          cmd.Parameters.Add("@empteccod", SqlDbType.Int).Value = p.empteccod;
```

```
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public void update_rec(clsempprp p)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("updemp", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@empcod", SqlDbType.Int).Value = p.empcod;
    cmd.Parameters.Add("@empnam", SqlDbType.VarChar, 50).Value = p.empnam;
    cmd.Parameters.Add("@empeml", SqlDbType.VarChar, 50).Value = p.empeml;
    cmd.Parameters.Add("@empteccod", SqlDbType.Int).Value = p.empteccod;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public void delete_rec(clsempprp p)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("delemp", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@empcod", SqlDbType.Int).Value = p.empcod;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public List<clsempprp> disp_rec()
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("dspemp", con);
    cmd.CommandType = CommandType.StoredProcedure;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clsempprp> obj = new List<clsempprp>();
    while (dr.Read())
    {
        clsempprp k = new clsempprp();
```

```
            k.empcod = Convert.ToInt32(dr[0]);
            k.empnam = dr[1].ToString();
            k.empeml = dr[2].ToString();
            k.empteccod = Convert.ToInt32(dr[3]);
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
    public List<clsempprp> find_rec(int pk)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("fndemp", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@empcod", SqlDbType.Int).Value = pk;
        SqlDataReader dr = cmd.ExecuteReader();
        List<clsempprp> obj = new List<clsempprp>();
        if (dr.HasRows)
        {
            dr.Read();
            clsempprp k = new clsempprp();
            k.empcod = Convert.ToInt32(dr[0]);
            k.empnam = dr[1].ToString();
            k.empeml = dr[2].ToString();
            k.empteccod = Convert.ToInt32(dr[3]);
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
}
public class clsasg : clscon            //manipulations on tbasg
{
    public DataSet dspempasg(Int32 empcod)
    {
        SqlDataAdapter adp = new SqlDataAdapter("dspempasg", con);
        adp.SelectCommand.CommandType = CommandType.StoredProcedure;
        adp.SelectCommand.Parameters.Add("@empcod", SqlDbType.Int).Value =
empcod;
```

```
    DataSet ds = new DataSet();
    adp.Fill(ds);
    return ds;
}
public void save_rec(clsasgprp p)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("insasg", con);
    cmd.CommandType = CommandType.StoredProcedure;
//    cmd.Parameters.Add("@asgcod", SqlDbType.Int).Value = p.asgcod;
    cmd.Parameters.Add("@asgbugcod", SqlDbType.Int).Value = p.asgbugcod;
    cmd.Parameters.Add("@asgempcod", SqlDbType.Int).Value = p.asgempcod;
    cmd.Parameters.Add("@asgdat", SqlDbType.DateTime).Value = p.asgdat;
    cmd.Parameters.Add("@asgesttim", SqlDbType.Int).Value = p.asgesttim;
    cmd.Parameters.Add("@asgaprsts", SqlDbType.Char).Value = p.asgaprsts;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public void update_rec(clsasgprp p)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("updasg", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@asgcod", SqlDbType.Int).Value = p.asgcod;
    cmd.Parameters.Add("@asgbugcod", SqlDbType.Int).Value = p.asgbugcod;
    cmd.Parameters.Add("@asgempcod", SqlDbType.Int).Value = p.asgempcod;
    cmd.Parameters.Add("@asgdat", SqlDbType.DateTime).Value = p.asgdat;
    cmd.Parameters.Add("@asgesttim", SqlDbType.Int).Value = p.asgesttim;
    cmd.Parameters.Add("@asgaprsts", SqlDbType.Char).Value = p.asgaprsts;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public void delete_rec(clsasgprp p)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
```

```
    SqlCommand cmd = new SqlCommand("delasg", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@asgcod", SqlDbType.Int).Value = p.asgcod;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public List<clsasgprp> disp_rec()
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("dspemp", con);
    cmd.CommandType = CommandType.StoredProcedure;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clsasgprp> obj = new List<clsasgprp>();
    while (dr.Read())
    {
        clsasgprp k = new clsasgprp();
        k.asgcod = Convert.ToInt32(dr[0]);
        k.asgbugcod = Convert.ToInt32(dr[1]);
        k.asgempcod = Convert.ToInt32(dr[2]);
        k.asgdat = Convert.ToDateTime(dr[3]);
        k.asgesttim = Convert.ToInt32(dr[4]);
        k.asgaprsts = Convert.ToChar(dr[5]);
        obj.Add(k);
    }
    dr.Close();
    cmd.Dispose();
    con.Close();
    return obj;
}
public List<clsasgprp> find_rec(int pk)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("fndemp", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@asgcod", SqlDbType.Int).Value = pk;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clsasgprp> obj = new List<clsasgprp>();
    if (dr.HasRows)
    {
```

```
            dr.Read();
            clsasgprp k = new clsasgprp();
            k.asgcod = Convert.ToInt32(dr[0]);
            k.asgbugcod = Convert.ToInt32(dr[1]);
            k.asgempcod = Convert.ToInt32(dr[2]);
            k.asgdat = Convert.ToDateTime(dr[3]);
            k.asgesttim = Convert.ToInt32(dr[4]);
            k.asgaprsts = Convert.ToChar(dr[5]);
            obj.Add(k);
          }
          dr.Close();
          cmd.Dispose();
          con.Close();
          return obj;
        }
    }
    public class clsbugres : clscon          //manipulations on tbbugres
    {
        public void save_rec(clsbugresprp p)
        {
          if (con.State == ConnectionState.Closed)
          {
            con.Open();
          }
          SqlCommand cmd = new SqlCommand("insbugres", con);
          cmd.CommandType = CommandType.StoredProcedure;
         //  cmd.Parameters.Add("@bugrescod", SqlDbType.Int).Value = p.bugrescod;
          cmd.Parameters.Add("@bugresasgcod", SqlDbType.Int).Value = p.bugresasgcod;
          cmd.Parameters.Add("@bugresdsc", SqlDbType.VarChar, 2000).Value =
p.bugresdsc;
          cmd.Parameters.Add("@bugresstrtim", SqlDbType.DateTime).Value =
p.bugresstrtim;
          cmd.Parameters.Add("@bugresendtim", SqlDbType.DateTime).Value =
p.bugresendtim;
          cmd.Parameters.Add("@bugrescmt", SqlDbType.VarChar, 500).Value =
p.bugrescmt;
          cmd.ExecuteNonQuery();
          cmd.Dispose();
          con.Close();
        }
        public void update_rec(clsbugresprp p)
        {
          if (con.State == ConnectionState.Closed)
          {
            con.Open();
          }
```

```
        SqlCommand cmd = new SqlCommand("updbugres", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@bugrescod", SqlDbType.Int).Value = p.bugrescod;
        cmd.Parameters.Add("@bugresasgcod", SqlDbType.Int).Value = p.bugresasgcod;
        cmd.Parameters.Add("@bugresdsc", SqlDbType.VarChar, 2000).Value =
p.bugresdsc;
        cmd.Parameters.Add("@bugresstrtim", SqlDbType.DateTime).Value =
p.bugresstrtim;
        cmd.Parameters.Add("@bugresendtim", SqlDbType.DateTime).Value =
p.bugresendtim;
        cmd.Parameters.Add("@bugrescmt", SqlDbType.VarChar, 500).Value =
p.bugrescmt;
        cmd.ExecuteNonQuery();
        cmd.Dispose();
        con.Close();
    }
    public void delete_rec(clsbugresprp p)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("delbugres", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@bugrescod", SqlDbType.Int).Value = p.bugrescod;
        cmd.ExecuteNonQuery();
        cmd.Dispose();
        con.Close();
    }
    public List<clsbugresprp> disp_rec()
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("dspbugres", con);
        cmd.CommandType = CommandType.StoredProcedure;
        SqlDataReader dr = cmd.ExecuteReader();
        List<clsbugresprp> obj = new List<clsbugresprp>();
        while (dr.Read())
        {
            clsbugresprp k = new clsbugresprp();
            k.bugrescod = Convert.ToInt32(dr[0]);
            k.bugresasgcod = Convert.ToInt32(dr[1]);
            k.bugresdsc = dr[2].ToString();
            k.bugresstrtim = Convert.ToDateTime(dr[3]);
```

```
            k.bugresendtim = Convert.ToDateTime(dr[4]);
            k.bugrescmt = dr[5].ToString();
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
    public List<clsbugresprp> find_rec(int pk)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("fndbugres", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@bugrescod", SqlDbType.Int).Value = pk;
        SqlDataReader dr = cmd.ExecuteReader();
        List<clsbugresprp> obj = new List<clsbugresprp>();
        if (dr.HasRows)
        {
            dr.Read();
            clsbugresprp k = new clsbugresprp();
            k.bugrescod = Convert.ToInt32(dr[0]);
            k.bugresasgcod = Convert.ToInt32(dr[1]);
            k.bugresdsc = dr[2].ToString();
            k.bugresstrtim = Convert.ToDateTime(dr[3]);
            k.bugresendtim = Convert.ToDateTime(dr[4]);
            k.bugrescmt = dr[5].ToString();
            obj.Add(k);
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        return obj;
    }
}
public class clsbugdep : clscon          //manipulations on tbbugdep
{
    public void save_rec(clsbugdepprp p)
    {
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
```

```
        SqlCommand cmd = new SqlCommand("insbugdep", con);
        cmd.CommandType = CommandType.StoredProcedure;
    //      cmd.Parameters.Add("@bugdepcod", SqlDbType.Int).Value = p.bugdepcod;
        cmd.Parameters.Add("@bugdepbugcod", SqlDbType.Int).Value =
p.bugdepbugcod;
        cmd.Parameters.Add("@bugdepdsc", SqlDbType.VarChar ,1000).Value =
p.bugdepdsc;
        cmd.Parameters.Add("@bugdepapasts", SqlDbType.Char, 1).Value =
p.bugdepaprsts;
        cmd.ExecuteNonQuery();
        cmd.Dispose();
        con.Close();
      }
      public void update_rec(clsbugdepprp p)
      {
        if (con.State == ConnectionState.Closed)
        {
          con.Open();
        }
        SqlCommand cmd = new SqlCommand("updbugdep", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@bugdepcod", SqlDbType.Int).Value = p.bugdepcod;
        cmd.Parameters.Add("@bugdepbugcod", SqlDbType.Int).Value =
p.bugdepbugcod;
    //      cmd.Parameters.Add("@bugdepdepbugcod", SqlDbType.Int).Value =
p.bugdepdepbugcod;
        cmd.Parameters.Add("@bugdepapasts", SqlDbType.Char, 1).Value =
p.bugdepaprsts;
        cmd.ExecuteNonQuery();
        cmd.Dispose();
        con.Close();
      }
      public void delete_rec(clsbugdepprp p)
      {
        if (con.State == ConnectionState.Closed)
        {
          con.Open();
        }
        SqlCommand cmd = new SqlCommand("delbugdep", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.Add("@bugdepcod", SqlDbType.Int).Value = p.bugdepcod;
        cmd.ExecuteNonQuery();
        cmd.Dispose();
        con.Close();
      }
      public List<clsbugdepprp> disp_rec()
```

```
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("dspbugdep", con);
    cmd.CommandType = CommandType.StoredProcedure;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clsbugdepprp> obj = new List<clsbugdepprp>();
    while (dr.Read())
    {
        clsbugdepprp k = new clsbugdepprp();
        k.bugdepcod = Convert.ToInt32(dr[0]);
        k.bugdepbugcod = Convert.ToInt32(dr[1]);
//        k.bugdepdepbugcod = Convert.ToInt32(dr[2]);
        k.bugdepaprsts = Convert.ToChar(dr[3]);
        obj.Add(k);
    }
    dr.Close();
    cmd.Dispose();
    con.Close();
    return obj;
}
public List<clsbugdepprp> find_rec(int pk)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("fndbugdep", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@bugdepcod", SqlDbType.Int).Value = pk;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clsbugdepprp> obj = new List<clsbugdepprp>();
    if (dr.HasRows)
    {
        dr.Read();
        clsbugdepprp k = new clsbugdepprp();
        k.bugdepcod = Convert.ToInt32(dr[0]);
        k.bugdepbugcod = Convert.ToInt32(dr[1]);
  //      k.bugdepdepbugcod = Convert.ToInt32(dr[2]);
        k.bugdepaprsts = Convert.ToChar(dr[3]);
        obj.Add(k);
    }
    dr.Close();
    cmd.Dispose();
```

```
            con.Close();
            return obj;
        }
    }
    public class clsusr : clscon            //manipulations on tbusr
    {
        public Int32 checkuser(String unam, String upas, out char rol)
        {
            if (con.State == ConnectionState.Closed)
                con.Open();
            SqlCommand cmd = new SqlCommand("logincheck", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@unam", SqlDbType.VarChar, 50).Value = unam;
            cmd.Parameters.Add("@upas", SqlDbType.VarChar, 50).Value = upas;
            cmd.Parameters.Add("@ucod", SqlDbType.Int).Direction =
ParameterDirection.Output;
            cmd.Parameters.Add("@urol", SqlDbType.Char, 1).Direction =
ParameterDirection.Output;
            cmd.ExecuteNonQuery();
            Int32 a = Convert.ToInt32(cmd.Parameters["@ucod"].Value);
            rol = Convert.ToChar(cmd.Parameters["@urol"].Value);
            cmd.Dispose();
            con.Close();
            return a;
        }
        public void save_rec(clsusrprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("insusr", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@usrcod", SqlDbType.Int).Value = p.usrcod;
            cmd.Parameters.Add("@usrnam", SqlDbType.VarChar, 50).Value = p.usrnam;
            cmd.Parameters.Add("@usrpwd", SqlDbType.VarChar, 50).Value = p.usrpwd;
            cmd.Parameters.Add("@usrcltempcod", SqlDbType.Int).Value = p.usrcltempcod;
            cmd.Parameters.Add("@usrrol", SqlDbType.Char, 1).Value = p.usrrol;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }
        public void update_rec(clsusrprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
```

```
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("updusr", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@usrcod", SqlDbType.Int).Value = p.usrcod;
    cmd.Parameters.Add("@usrnam", SqlDbType.VarChar, 50).Value = p.usrnam;
    cmd.Parameters.Add("@usrpwd", SqlDbType.VarChar, 50).Value = p.usrpwd;
    cmd.Parameters.Add("@usrcltempcod", SqlDbType.Int).Value = p.usrcltempcod;
    cmd.Parameters.Add("@usrrol", SqlDbType.Char, 1).Value = p.usrrol;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public void delete_rec(clsusrprp p)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("delusr", con);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.Add("@usrcod", SqlDbType.Int).Value = p.usrcod;
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
}
public List<clsusrprp> disp_rec()
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("dspusr", con);
    cmd.CommandType = CommandType.StoredProcedure;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clsusrprp> obj = new List<clsusrprp>();
    while (dr.Read())
    {
        clsusrprp k = new clsusrprp();
        k.usrcod = Convert.ToInt32(dr[0]);
        k.usrnam = dr[1].ToString();
        k.usrpwd = dr[2].ToString();
        k.usrcltempcod = Convert.ToInt32(dr[3]);
        k.usrrol = Convert.ToChar(dr[4]);
        obj.Add(k);
    }
```

```
            dr.Close();
            cmd.Dispose();
            con.Close();
            return obj;
        }
        public List<clsusrprp> find_rec(int pk)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("fndusr", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@usrcod", SqlDbType.Int).Value = pk;
            SqlDataReader dr = cmd.ExecuteReader();
            List<clsusrprp> obj = new List<clsusrprp>();
            if (dr.HasRows)
            {
                dr.Read();
                clsusrprp k = new clsusrprp();
                k.usrcod = Convert.ToInt32(dr[0]);
                k.usrnam = dr[1].ToString();
                k.usrpwd = dr[2].ToString();
                k.usrcltempcod = Convert.ToInt32(dr[3]);
                k.usrrol = Convert.ToChar(dr[4]);
                obj.Add(k);
            }
            dr.Close();
            cmd.Dispose();
            con.Close();
            return obj;
        }
    }
    public class clsmsg : clscon            //manipulations on tbmsg
    {
        public void save_rec(clsmsgprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("insmsg", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@msgcod", SqlDbType.Int).Value = p.msgcod;
            cmd.Parameters.Add("@msgsndusrcod", SqlDbType.Int).Value =
p.msgsndusrcod;
```

```
            cmd.Parameters.Add("@msgrecusrcod", SqlDbType.Int).Value = p.msgrecusrcod;
            cmd.Parameters.Add("@msgdsc", SqlDbType.VarChar, 2000).Value = p.msgdsc;
            cmd.Parameters.Add("@msgdat", SqlDbType.DateTime).Value = p.msgdat;
            cmd.Parameters.Add("@msgfil", SqlDbType.VarChar, 50).Value = p.msgfil;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }
        public void update_rec(clsmsgprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("updmsg", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@msgcod", SqlDbType.Int).Value = p.msgcod;
            cmd.Parameters.Add("@msgsndusrcod", SqlDbType.Int).Value =
p.msgsndusrcod;
            cmd.Parameters.Add("@msgrecusrcod", SqlDbType.Int).Value = p.msgrecusrcod;
            cmd.Parameters.Add("@msgdsc", SqlDbType.VarChar, 2000).Value = p.msgdsc;
            cmd.Parameters.Add("@msgdat", SqlDbType.DateTime).Value = p.msgdat;
            cmd.Parameters.Add("@msgfil", SqlDbType.VarChar, 50).Value = p.msgfil;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }
        public void delete_rec(clsmsgprp p)
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
            SqlCommand cmd = new SqlCommand("delmsg", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@msgcod", SqlDbType.Int).Value = p.msgcod;
            cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
        }
        public List<clsmsgprp> disp_rec()
        {
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }
```

```
    SqlCommand cmd = new SqlCommand("dspmsg", con);
    cmd.CommandType = CommandType.StoredProcedure;
    SqlDataReader dr = cmd.ExecuteReader();
    List<clsmsgprp> obj = new List<clsmsgprp>();
    while (dr.Read())
    {
        clsmsgprp k = new clsmsgprp();
        k.msgcod = Convert.ToInt32(dr[0]);
        k.msgsndusrcod = Convert.ToInt32(dr[1]);
        k.msgrecusrcod = Convert.ToInt32(dr[2]);
        k.msgdsc = dr[3].ToString();
        k.msgdat = Convert.ToDateTime(dr[4]);
        k.msgfil = dr[5].ToString();
        obj.Add(k);
    }
    dr.Close();
    cmd.Dispose();
    con.Close();
    return obj;
}
public List<clsmsgprp> find_rec(int pk)
{
    if (con.State == ConnectionState.Closed)
    {
        con.Open();
    }
    SqlCommand cmd = new SqlCommand("fndmsg", con);
    cmd.CommandType = CommandType.StoredProcedure;
    SqlDataReader dr = cmd.ExecuteReader();
    cmd.Parameters.Add("@msgcod", SqlDbType.Int).Value = pk;
    List<clsmsgprp> obj = new List<clsmsgprp>();
    if (dr.HasRows)
    {
        dr.Read();
        clsmsgprp k = new clsmsgprp();
        k.msgcod = Convert.ToInt32(dr[0]);
        k.msgsndusrcod = Convert.ToInt32(dr[1]);
        k.msgrecusrcod = Convert.ToInt32(dr[2]);
        k.msgdsc = dr[3].ToString();
        k.msgdat = Convert.ToDateTime(dr[4]);
        k.msgfil = dr[5].ToString();
        obj.Add(k);
    }
    dr.Close();
    cmd.Dispose();
    con.Close();
```

```
            return obj;
        }
    }
}
```

Coding of Login page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        nsbugtrk.clsusr obj = new nsbugtrk.clsusr();
        Int32 a;
        char rol;
        a = obj.checkuser(txtusrnam.Text, txtpwd.Text, out rol);
        if (a == -1)
        {
            Label1.Text = "Username Password Incorrect";
        }
        else
        {
            Session["cod"] = a;
            if (rol == 'A')
                Response.Redirect("admin/frmprj.aspx");
            else if (rol == 'C')
                Response.Redirect("client/frmbug.aspx");
            else if (rol == 'E')
                Response.Redirect("employee/frmvewasg.aspx");
        }
    }
}
```

Coding of frmaddclt.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Net.Mail;
public partial class admin_Default : System.Web.UI.Page
{
   protected void Page_Load(object sender, EventArgs e)
   {

   }
   protected void Button1_Click(object sender, EventArgs e)
   {
      nsbugtrk.clsclt obj = new nsbugtrk.clsclt();
      nsbugtrk.clscltprp objprp = new nsbugtrk.clscltprp();
      Int32 a = obj.getauto();
      objprp.cltcod = a;
      objprp.cltnam = txtnam.Text;
      objprp.clteml = txteml.Text;
      objprp.cltphn = txtphn.Text;
      objprp.cltprf = txtprf.Text;
      obj.save_rec(objprp);
      //to store info in user table
      nsbugtrk.clsusr obj1 = new nsbugtrk.clsusr();
      nsbugtrk.clsusrprp objprp1 = new nsbugtrk.clsusrprp();
      objprp1.usrnam = txtusrnam.Text;
      objprp1.usrpwd = txtpwd.Text;
      objprp1.usrrol = 'C';
      objprp1.usrcltempcod = a;
      obj1.save_rec(objprp1);
      //send an email with usernam & password
   /*    MailMessage mm = new MailMessage("admin@bugtracker.com", txteml.Text,
"Bugtraker Login Credentials", " Username :" + txtusrnam.Text + " Password:" +
txtpwd.Text);
      SmtpClient c = new SmtpClient("mail.connectzone.in");
      c.Send(mm);  */
      txtnam.Text = "";
      txteml.Text = "";
      txtphn.Text = "";
      txtprf.Text = "";
      txtusrnam.Text = "";
   }
   protected void Button2_Click(object sender, EventArgs e)
   {
      txtnam.Text = "";
      txteml.Text = "";
      txtphn.Text = "";
```

```
        txtprf.Text = "";
        txtusrnam.Text = "";
    }
}
```

Coding of frmasgbug.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class admin_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
    {

    }
    public String getsts(char sts)
    {
        if (sts == 'N')
            return "Not Assigned";
        else if (sts == 'A')
            return "Assigned";
        else if (sts == 'D')
            return "Dependent";
        else
            return "";
    }
    protected void GridView1_RowEditing(object sender, GridViewEditEventArgs e)
    {
        Response.Redirect("frmbugdet.aspx?bcod=" +
GridView1.DataKeys[e.NewEditIndex][0].ToString());
    }
}
```

Coding of frmbugbyprj.aspx :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class admin_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Button1.Attributes.Add("onclick", "JavaScript:window.print(self);");
    }
    protected void Button1_Click(object sender, EventArgs e)
    {

    }
}
```

Coding of frmbugdet.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class admin_Default : System.Web.UI.Page
{
    public String getprjnam(Int32 pcod)
    {
        nsbugtrk.clsprj obj = new nsbugtrk.clsprj();
        List<nsbugtrk.clsprjprp> k = obj.find_rec(pcod);
        return k[0].prjnam;
    }
    public String gettecnam(Int32 tcod)
    {
        nsbugtrk.clstec obj = new nsbugtrk.clstec();
        List<nsbugtrk.clstecprp> k = obj.find_rec(tcod);
        return k[0].tecnam;
    }
    protected void Page_Load(object sender, EventArgs e)
    {
        if (Page.IsPostBack == false)
```

```
    {
        DetailsView1.DataBind();
        Int32 a = Convert.ToInt32(DetailsView1.DataKey[0]);
        nsbugtrk.clsemp obj = new nsbugtrk.clsemp();

        DropDownList1.DataSource = obj.dspempbytec(a);
        DropDownList1.DataTextField = "empnam";
        DropDownList1.DataValueField = "empcod";
        DropDownList1.DataBind();
        DropDownList1.Items.Insert(0,new ListItem("Select", "0"));
    }
}
protected void Button1_Click(object sender, EventArgs e)
{
    nsbugtrk.clsasg obj = new nsbugtrk.clsasg();
    nsbugtrk.clsasgprp objprp = new nsbugtrk.clsasgprp();
    objprp.asgdat = DateTime.Now;
    objprp.asgempcod = Convert.ToInt32(DropDownList1.SelectedValue);
    objprp.asgbugcod = Convert.ToInt32(Request.QueryString["bcod"]);
    objprp.asgesttim = 0;
    objprp.asgaprsts = 'A';
    obj.save_rec(objprp);
    nsbugtrk.clsbug obj1 = new nsbugtrk.clsbug();
    nsbugtrk.clsbugprp objprp1 = new nsbugtrk.clsbugprp();
    objprp1.bugcod = Convert.ToInt32(Request.QueryString["bcod"]);
    objprp1.bugsts = 'A';
    obj1.updbugsts(objprp1);
    Response.Redirect("frmasgbug.aspx");
}
protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
{
    nsbugtrk.clsasg obj = new nsbugtrk.clsasg();
    Int32 a = Convert.ToInt32(DropDownList1.SelectedValue);
    if (a != 0)
    {
        GridView1.DataSource = obj.dspempasg(a);
        GridView1.DataBind();
    }
}
protected void GridView1_RowEditing(object sender, GridViewEditEventArgs e)
{
    Int32 a = Convert.ToInt32(GridView1.DataKeys[e.NewEditIndex][0]);
    nsbugtrk.clsasg obj = new nsbugtrk.clsasg();
    nsbugtrk.clsasgprp objprp = new nsbugtrk.clsasgprp();
    objprp.asgcod = a;
    obj.delete_rec(objprp);
```

```
        nsbugtrk.clsbug obj1 = new nsbugtrk.clsbug();
        nsbugtrk.clsbugprp objprp1 = new nsbugtrk.clsbugprp();
        objprp1.bugcod = Convert.ToInt32(GridView1.DataKeys[e.NewEditIndex][1]);
        objprp1.bugsts = 'N';
         obj1.updbugsts(objprp1);
        GridView1.DataBind();
    }
    protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
    {

    }
    protected void DetailsView1_PageIndexChanging(object sender,
DetailsViewPageEventArgs e)
    {

    }
}
```

Coding of frmemp.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class admin_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        nsbugtrk.clsemp obj = new nsbugtrk.clsemp();
        nsbugtrk.clsempprp objprp = new nsbugtrk.clsempprp();
        objprp.empnam = TextBox1.Text;
        objprp.empeml = TextBox2.Text;
        objprp.empteccod = Convert.ToInt32(DropDownList1.SelectedValue);
        Int32 a;
        a = obj.getauto();
        objprp.empcod = a;
        obj.save_rec(objprp);
        nsbugtrk.clsusr obj1 = new nsbugtrk.clsusr();
        nsbugtrk.clsusrprp objprp1 = new nsbugtrk.clsusrprp();
```

```
        objprp1.usrnam = TextBox3.Text;
        objprp1.usrpwd = TextBox4.Text;
        objprp1.usrrol = 'E';
        objprp1.usrcltempcod = a;
        obj1.save_rec(objprp1);
        TextBox1.Text = "";
        TextBox2.Text = "";
        TextBox3.Text = "";
    }
}
```

Coding of frmmngclt.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class admin_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
    {

    }
    protected void GridView1_RowUpdating(object sender, GridViewUpdateEventArgs e)
    {
        nsbugtrk.clsclt obj = new nsbugtrk.clsclt();
        nsbugtrk.clscltprp objprp = new nsbugtrk.clscltprp();
        objprp.cltcod = Convert.ToInt32(GridView1.DataKeys[e.RowIndex][0]);
        objprp.cltnam =
((TextBox)(GridView1.Rows[e.RowIndex].Cells[0].Controls[0])).Text;
        objprp.clteml =
((TextBox)(GridView1.Rows[e.RowIndex].Cells[1].Controls[0])).Text;
        objprp.cltphn =
((TextBox)(GridView1.Rows[e.RowIndex].Cells[2].Controls[0])).Text;
        objprp.cltprf =
((TextBox)(GridView1.Rows[e.RowIndex].Cells[3].Controls[0])).Text;
        obj.update_rec(objprp);
        GridView1.EditIndex = -1;
        GridView1.DataBind();
        e.Cancel = true;
```

```
    }
    protected void GridView1_RowDeleting(object sender, GridViewDeleteEventArgs e)
    {
        nsbugtrk.clsclt obj = new nsbugtrk.clsclt();
        nsbugtrk.clscltprp objprp = new nsbugtrk.clscltprp();
        objprp.cltcod = Convert.ToInt32(GridView1.DataKeys[e.RowIndex][0]);
        obj.delete_rec(objprp);
        GridView1.EditIndex = -1;
        GridView1.DataBind();
        e.Cancel = true;
    }
}
```

Coding of frmprj.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class admin_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        nsbugtrk.clsprj obj = new nsbugtrk.clsprj();
        nsbugtrk.clsprjprp objprp = new nsbugtrk.clsprjprp();
        Int32 a = obj.getauto();
        objprp.prjcod = a;
        objprp.prjnam = TextBox1.Text;
        objprp.prjcltcod = Convert.ToInt32(DropDownList1.SelectedValue);
        objprp.prjstrdat = Convert.ToDateTime(TextBox2.Text);
        objprp.prjenddat = Convert.ToDateTime(TextBox3.Text);
        objprp.prjdsc = TextBox4.Text;
        obj.save_rec(objprp);
        nsbugtrk.clsprjtec obj1 = new nsbugtrk.clsprjtec();
        nsbugtrk.clsprjtecprp objprp1 = new nsbugtrk.clsprjtecprp();
        objprp1.prjtecprjcod = a;
        for (int i = 0; i < CheckBoxList1.Items.Count; i++)
        {
            if (CheckBoxList1.Items[i].Selected == true)
```

```
            {
                objprp1.prjtecteccod = Convert.ToInt32(CheckBoxList1.Items[i].Value);
                obj1.save_rec(objprp1);
            }
        }
    }
}
```

Coding of frmtec.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class admin_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        nsbugtrk.clstec obj = new nsbugtrk.clstec();
        nsbugtrk.clstecprp objprp = new nsbugtrk.clstecprp();
        objprp.tecnam = TextBox1.Text;
        obj.save_rec(objprp);
        TextBox1.Text = "";
    }
}
```

Coding of frmasgsts.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class employee_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
```

```
    protected void GridView1_RowUpdating(object sender, GridViewUpdateEventArgs e)
    {
        Response.Redirect("frmupdsts.aspx?bcod=" +
GridView1.DataKeys[e.RowIndex][0].ToString() + "&acod=" +
GridView1.DataKeys[e.RowIndex][1].ToString());
    }
}
```

Coding of frmupdsts.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class employee_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void RadioButtonList1_SelectedIndexChanged(object sender, EventArgs e)
    {
        if (RadioButtonList1.SelectedIndex == 0)
        {
            Panel1.Visible = true;
            Panel2.Visible = false;
        }
        else
        {
            Panel1.Visible = false;
            Panel2.Visible = true;
        }
    }
    protected void Button3_Click(object sender, EventArgs e)
    {
        nsbugtrk.clsbugres obj = new nsbugtrk.clsbugres();
        nsbugtrk.clsbugresprp objprp = new nsbugtrk.clsbugresprp();
        objprp.bugresasgcod = Convert.ToInt32(Request.QueryString["acod"]);
        objprp.bugresdsc = TextBox1.Text;
        objprp.bugresstrtim = Convert.ToDateTime(TextBox2.Text);
        objprp.bugresendtim = Convert.ToDateTime(TextBox3.Text);
        objprp.bugrescmt = TextBox4.Text;
        obj.save_rec(objprp);
```

```
        nsbugtrk.clsbug obj1 = new nsbugtrk.clsbug();
        nsbugtrk.clsbugprp objprp1 = new nsbugtrk.clsbugprp();
        objprp1.bugcod = Convert.ToInt32(Request.QueryString["bcod"]);
        objprp1.bugsts = 'R';
        obj1.updbugsts(objprp1);
        Response.Redirect("frmasgsts.aspx");
    }
    protected void Button4_Click(object sender, EventArgs e)
    {
        TextBox1.Text = "";
        TextBox2.Text = "";
        TextBox3.Text = "";
        TextBox4.Text = "";
        Panel2.Visible = false;
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        nsbugtrk.clsbugdep obj = new nsbugtrk.clsbugdep();
        nsbugtrk.clsbugdepprp objprp = new nsbugtrk.clsbugdepprp();
        objprp.bugdepbugcod = Convert.ToInt32(Request.QueryString["bcod"]);
        objprp.bugdepdsc = TextBox5.Text;
        objprp.bugdepaprsts = 'N';
        obj.save_rec(objprp);
        Response.Redirect("frmasgsts.aspx");
    }
}
```

Coding of frmaddbug.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class client_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        nsbugtrk.clsbug obj = new nsbugtrk.clsbug();
        nsbugtrk.clsbugprp objprp = new nsbugtrk.clsbugprp();
        objprp.bugdsc = TextBox1.Text;
```

```
        objprp.bugprjcod = Convert.ToInt32(Request.QueryString["pcod"]);
        objprp.bugpstdat = DateTime.Now;
        objprp.bugurl = TextBox2.Text = "";
        objprp.bugteccod = Convert.ToInt32(DropDownList1.SelectedValue);
        objprp.bugsts = 'N';
        objprp.bugsevlvl =Convert.ToChar(DropDownList2.SelectedValue );
        Int32 a = obj.getauto();
        objprp.bugcod = a;
        if (FileUpload1.FileName != "")
        {
           String s = FileUpload1.FileName;
           s = s.Substring(s.LastIndexOf('.'));
           s = a.ToString() + s;
           FileUpload1.PostedFile.SaveAs(Server.MapPath("../bugfiles") + "//" + s);
           objprp.bugscrsht = s;
        }
        else
           objprp.bugscrsht = "";
        obj.save_rec(objprp);
        TextBox1.Text = "";
        TextBox2.Text = "";
    }
}
```

Coding of frmbug.aspx page :-

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class client_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void GridView1_RowEditing(object sender, GridViewEditEventArgs e)
    {
        Response.Redirect("frmaddbug.aspx?pcod=" +
Convert.ToInt32(GridView1.DataKeys[e.NewEditIndex][0]));
    }
    protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
    {
```

```
    }
}
```

6. Implementation/Technological Environment

Implementation is the stage in the project where the theoretical design is turned into a working system. The implementation phase constructs, installs and operates the new system. The most crucial stage in achieving a new successful system is that it will work efficiently and effectively.

There are several activities involved while implementing a new project they are

- End user training
- End user Education
- Training on the application software
- System Design
- Parallel Run And To New System
- Post implementation Review

End user Training:

The successful implementation of the new system will purely upon the involvement of the officers working in that department. The officers will be imparted the necessary training on the new technology.

End User Education:

The education of the end user start after the implementation and testing is over. When the system is found to be more difficult to under stand and complex, more effort is put to educate the end used to make them aware of the system, giving them lectures about the new system and providing them necessary documents and materials about how the system can do this.

Training of application software:

After providing the necessary basic training on the computer awareness, the users will have to be trained upon the new system such as the screen flows and screen design type of help on the screen , type of errors while entering the data , the corresponding validation check at each entry and the way to correct the data entered. It should then cover information needed by the specific user or group to use the system.

Post Implementation View:

The department is planning a method to know the states of the past implementation process. For that regular meeting will be arranged by the concerned officers about the implementation problem and success

7. Testing And Result

Is the menu bar displayed in the appropriate contested some system related features included either in menus or tools? Do pull –Down menu operation and Tool-bars work properly? Are all menu function and pull down sub function properly listed ?; Is it possible to invoke each menu function using a logical assumptions that if all parts of the system are correct, the goal will be successfully achieved .? In adequate testing or non-testing will leads to errors that may appear few months later.

This create two problem

1. Time delay between the cause and appearance of the problem.
2. The effect of the system errors on files and records within the system

The purpose of the system testing is to consider all the likely variations to which it will be suggested and push the systems to limits.

The testing process focuses on the logical intervals of the software ensuring that all statements have been tested and on functional interval is conducting tests to uncover

errors and ensure that defined input will produce actual results that agree with the required results. Program level testing, modules level testing integrated and carried out.

There are two major type of testing they are

1) White Box Testing.
2) Black Box Testing.

White Box Testing

White box some times called “Glass box testing” is a test case design uses the control structure of the procedural design to drive test case.

Using white box testing methods, the following tests where made on the system

a) All independent paths within a module have been exercised once. In our system, ensuring that case was selected and executed checked all case structures. The bugs that were prevailing in some part of the code where fixed

b) All logical decisions were checked for the truth and falsity of the values.

Black box Testing

Black box testing focuses on the functional requirements of the software. This is black box testing enables the software engineering to derive a set of input conditions that will fully exercise all functional requirements for a program. Black box testing is not an alternative to white box testing rather it is complementary approach that is likely to uncover a different class of errors that white box methods like..

1) Interface errors
2) Performance in data structure
3) Performance errors
4) Initializing and termination errors

8. Enhancements

- **External user Access:-**

Here we can allow an external user to acess the bug information and use his skills to resolve the bugs easily so that we can save time of employees and clients. It will surely help to minimize the time requirement of project and improve the quality of project.

- **Enhanced Security:-**

In this project security features are very less. We can add some security features such as the bug information go to employee or clients which are related to that project in a encrypted form so that no one can change it in the midway.

- **Global Version:-**

Here this software project work only for a single organization . so we have to make a global version which will work for all organizations after some little modifications.

9. Limitations

There are some limitations in this project:

- **External user acess:-**

  Here an external user can not acess the bug information and use his skills to resolve the bugs easily so that we can save time of employees and clients. It will surely help to minimize the time requirement of project and improve the quality of project. So it is a big limitation.

- **Security:-**

  In this project security features are very less. The data about bug information is transmitted via network which is not secure. We can add some security

features such as the bug information go to employee or clients which are related to that project in an encrypted form so that no one can change it in the midway.

- **Organization Specific:-**

  Here this software project work only for a single organization .so we have to make a global version which will work for all organizations after some little modifi--cations.

10. Conclusion

The system has been developed for the given condition and is found working effectively. The developed system is flexible and changes can be made easily whenever required. Using the facilities and functionalities of .Net, the software has been developed in a neat and simple manner, thereby reducing the operator's work.

The speed and accuracy are maintained in proper way. The user-friendly nature of this software developed in .Net framework is very easy to work with both the higher management as well as other users with little knowledge of computer. The results obtained were fully satisfactory from the user point of view.

The system was verified with valid as well as invalid data in each manner. The system is run with an insight into the necessary modifications that may be required in the future. Hence the system can be maintained successfully.

11. Bibliography

BOOKS

Database management system	Vipin C. Desai
SQL Server	Microsoft Press
ASP.NET	Wrox publications
Javascript	Ivan Byross

WEBSITES

www.dotnetspider.com/home.aspx

www.gotdotnet.com/database.aspx

www.onlinetemplates.org/templates.aspx

www.webopedia.com/dotnet.aspx

www.msdn.microsoft.com/dotnet.aspx