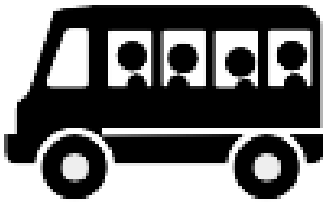


Bus Reservation System

TABLE OF CONTENTS

1. ABSTRACT

Traveling is a large growing business across all countries. Bus reservation system deals with maintenance of records of details of each passenger who had reserved a seat for a journey. It also includes maintenance of information like schedule and details of each bus.

We observed the working of the Bus reservation system and after going through it, we get to know that there are many operations, which they have to do manually. It takes a lot of time and causes many errors. Due to this, sometimes a lot of problems occur and they were facing many disputes with customers. To solve the above problem, and further maintaining records of items, seat availability for customers, price of per seat, bill generation and other things, we are offering this proposal of reservation system.

By using this software, we can reserve tickets from any part of the world, through telephone lines, via internet. This project provides and checks all sorts of constraints so that user does give only useful data and thus validation is done in an effective way.

2. INTRODUCTION

Our project is to computerize traveling company to manage data, so that all the transactions become fast and there should not be any error in transaction like calculation mistake, bill generation and other things. It replaces all the paper work. It keeps records of all bills also, giving to ensure 100% successful implementation of the computerized Bus reservation system.

Our reservation system has three modules. First module helps the customer to enquire the availability of seats in a particular bus at particular date. Second module helps him to reserve a ticket. Using third module he can cancel a reserved ticket.

- First module retrieves data from tables required for enquire.
- Second module inserts values into the tables on reservation.
- Third module deletes values into from the table on cancellation of tickets.

As our application is hosted using IIS Server onto internet it can be accessed by many number of people concurrently.

.

3. SYSTEM STUDY

System study aims at establishing requests for the system to be acquired, development and installed. It involves studying and analyzing the ways of an organization currently processing the data to produce information. Analyzing the problem thoroughly forms the vital part of the system study. In system analysis, prevailing situation of problem is carefully examined by breaking them into sub problems. Problematic areas are identified and information is collected. Data gathering is essential to any analysis of requests. It is necessary that this analysis familiarizes the designer with objectives, activities and the function of the organization in which the system is to be implemented.

3.1.Study of existing system

- ✓ Existing system is totally on book and thus a great amount of manual work has to be done. The amount of manual work increases exponentially with increase in bus services.
- ✓ Needs a lot of working staff and extra attention on all the records.
- ✓ In existing system, there are various problems like keeping records of items, seats available, prices of per/seat and fixing bill generation on each bill.
- ✓ Finding out details regarding any information is very difficult, as the user has to go through all the books manually.
- ✓ Major problem was lack of security.

3.2.Proposed system

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features

- ✓ Ensure data accuracy.
- ✓ Records are efficiently maintained by DBMS.
- ✓ DBMS also provides security for the information.
- ✓ Any person across the world, having internet can access this service.
- ✓ Availability of seats can be enquired very easily.
- ✓ Passengers can also cancel their tickets easily.
- ✓ Minimum time needed for the various processing
- ✓ Better Service
- ✓ Minimum time required
- ✓ This would help the corporation prepare and organize its schedules more efficiently on the basis of traffic demand.

3.3. Introduction about the front end(.NET)

This proposed software is going to be development using the latest technology from Microsoft called Microsoft .NET and it is the software that connects information, people, systems, and devices. It spans clients, servers, and developer tools, consists of:

- The .NET Framework 3.0 used for building and running all kinds of software, including Web-based applications, smart client applications, Mobile applications etc.
- Development tools, such as Microsoft Visual Studio .NET 2010, which provides an integrated development environment (IDE) for maximizing development productivity with the .NET Framework.

3.4. Introduction about the back end(SQL SERVER 2008)

Microsoft SQL Server 2008 is comprehensive, integrated data management and analysis software that enables organizations to reliably manage mission-critical information and confidently run today's increasingly complex business applications. SQL Server 2008 allows companies to gain greater insight from their business information and achieve faster results for a competitive advantage.

- SQL Server is the fastest growing Database and Business Intelligence vendor.
- SQL Server is more secure than Oracle. Since July 2003 more than 100 critical Oracle database security vulnerabilities have been identified, compared to ZERO for SQL Server for that period.

4. SYSTEM SPECIFICATION

4.1. Hardware Requirements:

- PC with Pentium IV processor.
- 512 MB RAM or above.
- 40 GB Hard Disk or above.

4.2.Software Requirements:

- Operating system : Windows XP (or latest).
- Front end : Visual C sharp
- Platform : ASP.NET (.NET web platform 3.0 or above)
- Integrated development environment(IDE): Visual Studio 10.0
- Back end : SQL SERVER 2008

5. DATABASE DESIGN

5.1. Conceptual design

a) **Requirement Analysis:**

What data is needed?

List of Entities:

- Bus
- Passenger
- Route
- Reserves

List of attributes:

Bus:

a. Bus id
b. Type
    a. AC (or) Non-AC
    b. Sleeper (or) Not
c. Maximum seats

Route information:

a. Route id
b. To location
c. From location
d. Depart date
e. Depart time
f. Fare

Reserves information:

a. Ticket number
b. Route id
c. Passenger id

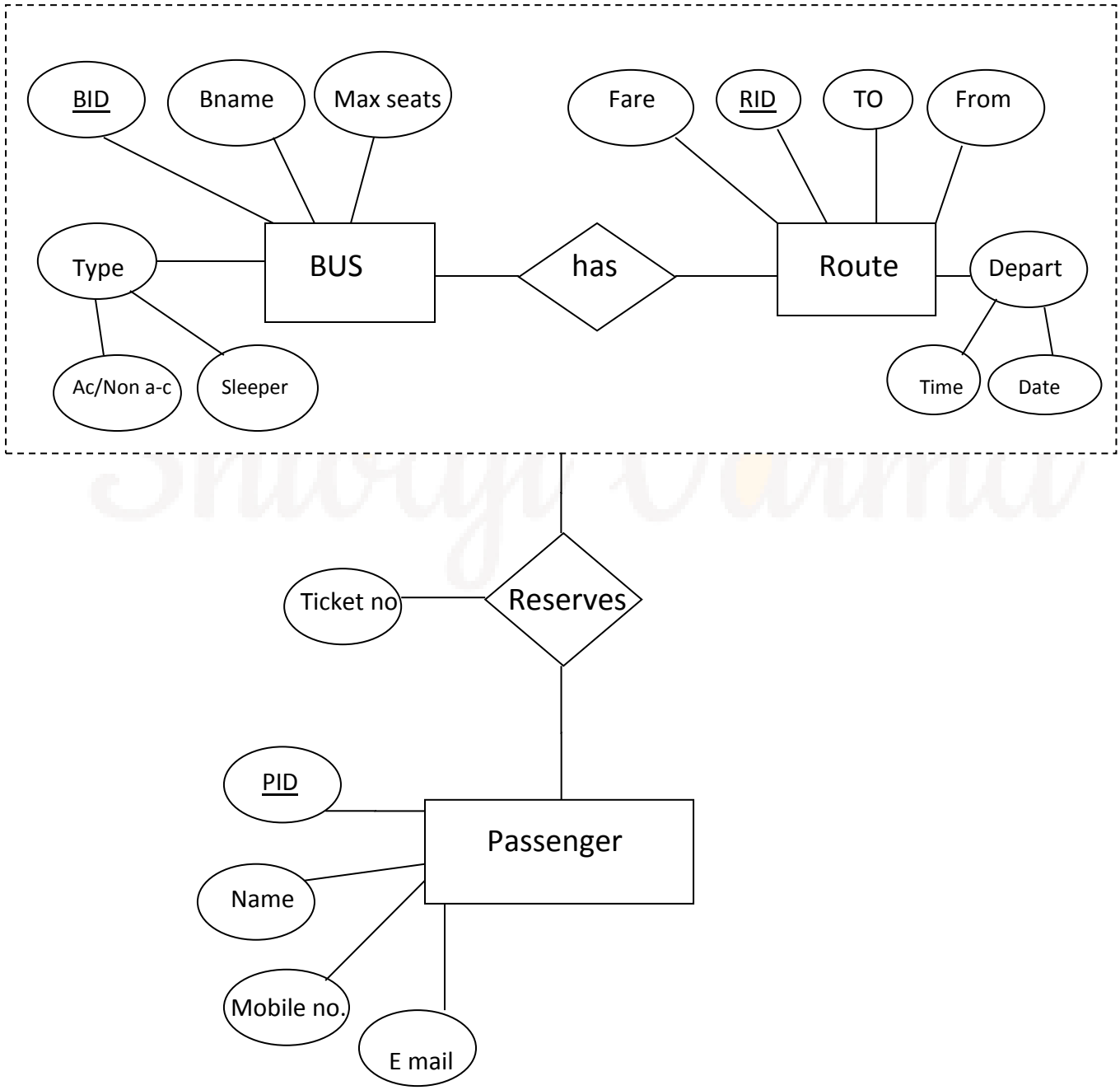
Passenger information:

a. Ticket number
b. Name
c. Mobile number
d. E-mail

List of Relations:

- Reserves
- Has

b) Er model:



5.2.Logical DB Design

1. PASSENGER: Keeps record of passengers.

S. No	Field name	Data Type	Description	Constraints
1	*PID*	*Integer*	*Passenger identity number.*	*Primary key*
2	*Name*	*Char(20)*	*Name of the passenger*	
3	*Mobile*	*Integer*	*Mobile number of passenger*	
4	*E_mail*	*Char(20)*	*Mail id of passenger*	

Schema definition:

CREATE TABLE passenger (PID integer primary key,

Name char(20),

Mobile integer,

Email char(20));

Table:

PID	Name	Mobile	E_mail
1	Shivaji	9999999999	shivaji@gmail.com
2	Pavan	9898989898	pavan@yahoo.com

2. BUS: Keeps records of bus information.

S. No	Field name	Data Type	Description	Constraints
1	*BID*	*Integer*	*Bus identity number*	*Primary key*
2	*Bname*	*Char(15)*	*Name of the bus*	
2	*Max_seats*	*Integer*	*Max no. of seats in a bus*	
3	*Type_ac*	*Binary*	*Mail id of passenger*	
4	*Type_sleeper*	*Binary*	*Whether bus is AC/non-AC*	

Schema definition:

```
CREATE TABLE bus (BID integer primary key,
                  Bname char(15),
                  Max_seats integer,
                  Type_ac bit,
                  Type_sleeper bit);
```

Table:

BID	Bname	Max_seats	Type_AC	Type_Sleeper
1	Ganga	28	Yes	No
2	Garuda	36	No	Yes

3. RESERVES: Keeps reservation information.

S. No	Field name	Data Type	Description	Constraints
1	*Ticket_no*	*Integer*	*Ticket number*	*Primary key*
2	*RID*	*Integer*	*Route identity number*	*Foreign key*
3	*PID*	*Integer*	*Passenger identity number*	*Foreign key*

```
CREATE TABLE reserves (Ticket_no integer primary key,
                       RID integer,
                       PID integer,
                       Foreign key(rid) references to route(rid),
                       Foreign key(pid) references to passenger(pid));
```

Table:

Ticket_no	RID	PID
315	1	3
316	3	6

4. ROUTE: Keep route information.

S. No	Field name	Data Type	Description	Constraints
1	*RID*	*integer*	*Route identity number*	*Primary key*
2	*BID*	*integer*	*Bus identity number*	*Foreign key*
3	*Fare*	*Real*	*Cost per seat*	
4	*From_location*	*char(30)*	*Starting point*	
5	*To_location*	*char(30)*	*Destination point*	
6	*Depart_date*	*Date*	*Date of departure*	
7	*Depart_time*	*Time*	*Time of departure*	

```
CREATE TABLE route (RID integer primary key,
                    BID integer,
                    Fare real,
                    From_location char(30),
                    To_location char(30),
                    Departure_date date,
                    Departure_time time,
                    Foreign key(bid) references to bus(bid));
```

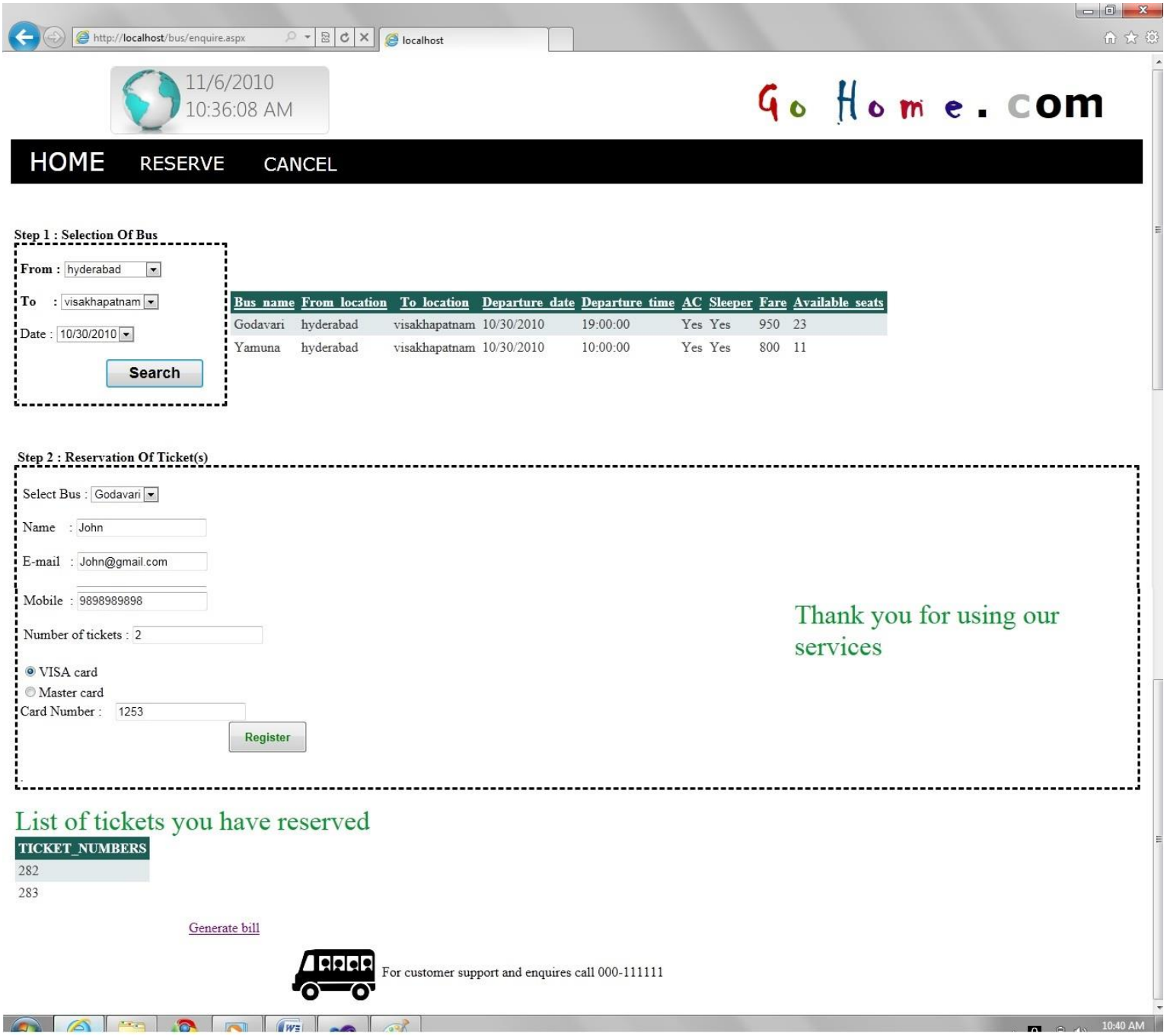
Table:

RID	BID	Fare	From_location	To_location	Departure_date	Departure_time
1	6	28	Hyderabad	Visakhapatnam	12/24/2010	04:50:00
2	5	36	Vijayawada	Visakhapatnam	11/21/2010	15:45:00

6. CODING AND SCREEN LAYOUTS

6.1.Reservation and enquire

6.1.1. Screen layout:



6.1.2. Programming code:

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


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

    }



    protected void Button2_Click(object sender, EventArgs e)
    {

        if (No_of_tickets.Text != "" && Name_textbox.Text != "" && Mobile_textbox.Text !=
"" && Email_textbox.Text != "" && DropDownList4.SelectedValue != "" &&
RadioButtonList1.SelectedValue != "" && Cardnumber.Text != "")
        {
            if (Int64.Parse(Mobile_textbox.Text) < 9999999999 &&
Int64.Parse(Mobile_textbox.Text) > 1000000000 && Email_textbox.Text.Contains("@"))
            {
                SqlConnection cn = new SqlConnection("Data Source=" +
CommonToALL.CommmonClass.server + ";Integrated Security=True");
                cn.Open();
                string cmd="SELECT (b.max_seats-(select count(rid) from reserves where
rid=r.rid)),r.fare from route r,bus b where r.bid=b.bid and r.from_location='" +
DropDownList1.SelectedValue + "' and r.to_location='" + DropDownList2.SelectedValue + "'
and r.departure_date='" + DropDownList3.SelectedValue + "' and r.rid='" +
DropDownList4.SelectedValue + "'";
                SqlCommand sqlcmd=new SqlCommand(cmd,cn);
                SqlDataReader DbReader= sqlcmd.ExecuteReader();

                DbReader.Read();

                int Max_seats = Int32.Parse(DbReader.GetValue(0).ToString());
                string Fare = DbReader.GetValue(1).ToString();
                sqlcmd.Dispose();
                cn.Close();

                if (Int32.Parse(No_of_tickets.Text) <= Max_seats &&
Int32.Parse(No_of_tickets.Text) > 0)
                {

CommonToALL.CommmonClass.put(Name_textbox.Text,No_of_tickets.Text,Mobile_textbox.Text,Ema
il_textbox.Text,Fare);
                    Bill.Visible = true;

                    Passenger_registration.Insert();
                    ThankBOX.Text = "Thank you for using our services";
                    ErrorBOX.Text = "";
                    TicketBOX.Text = "List of tickets you have reserved";
```

```
                    cmd = "SELECT max(pid) from passenger;";
                    cn.Open();
                    sqlcmd = new SqlCommand(cmd, cn);

                    DbReader = sqlcmd.ExecuteReader();

                    DbReader.Read();
                    string pid = DbReader.GetValue(0).ToString();
                    DbReader.Dispose();

                    for (int i = 0; i < Int32.Parse(No_of_tickets.Text); i++)
                    {
                        cmd = "INSERT INTO [Reserves] ([Rid], [Pid]) VALUES
('"+DropDownList4.SelectedValue+"','"+pid+"')";

                        sqlcmd = new SqlCommand(cmd, cn);

                        sqlcmd.ExecuteNonQuery();

                    }

                    sqlcmd.Dispose();
                    cn.Close();



                }

                else
                {
                    ErrorBOX.Text = "Only " + Max_seats + " seats are avilable";
                    ThankBOX.Text = "";
                    TicketBOX.Text = "";
                }
            }
            else
            {
                ErrorBOX.Text = "Invalid mobile number or email id";
                ThankBOX.Text = "";
                TicketBOX.Text = "";
            }
        }
        else
        {
            ErrorBOX.Text = "Fill the form to register completely";
            ThankBOX.Text = "";
            TicketBOX.Text = "";
        }

    }
}
```

//ASP CODE FOR INSERTION OF PASSENGER DETAILS:

```
<asp:SqlDataSource ID="Passenger_registration" runat="server"
                                    ConnectionString="<%$ ConnectionStrings:Sql2008 %>"

                                    InsertCommand="INSERT INTO [Passenger] ([Name],
[Mobile], [Email]) VALUES (@Name, @Mobile, @Email)">

                                    <InsertParameters>
                                        <asp:ControlParameter ControlID="Name_textbox"
PropertyName="Text" Name="Name" Type="String" />
                                        <asp:ControlParameter ControlID="Mobile_textbox"
PropertyName="Text" Name="Mobile" Type="Int64" />
                                        <asp:ControlParameter ControlID="Email_textbox"
PropertyName="Text" Name="Email" Type="String" />
                                    </InsertParameters>
                                </asp:SqlDataSource>
```

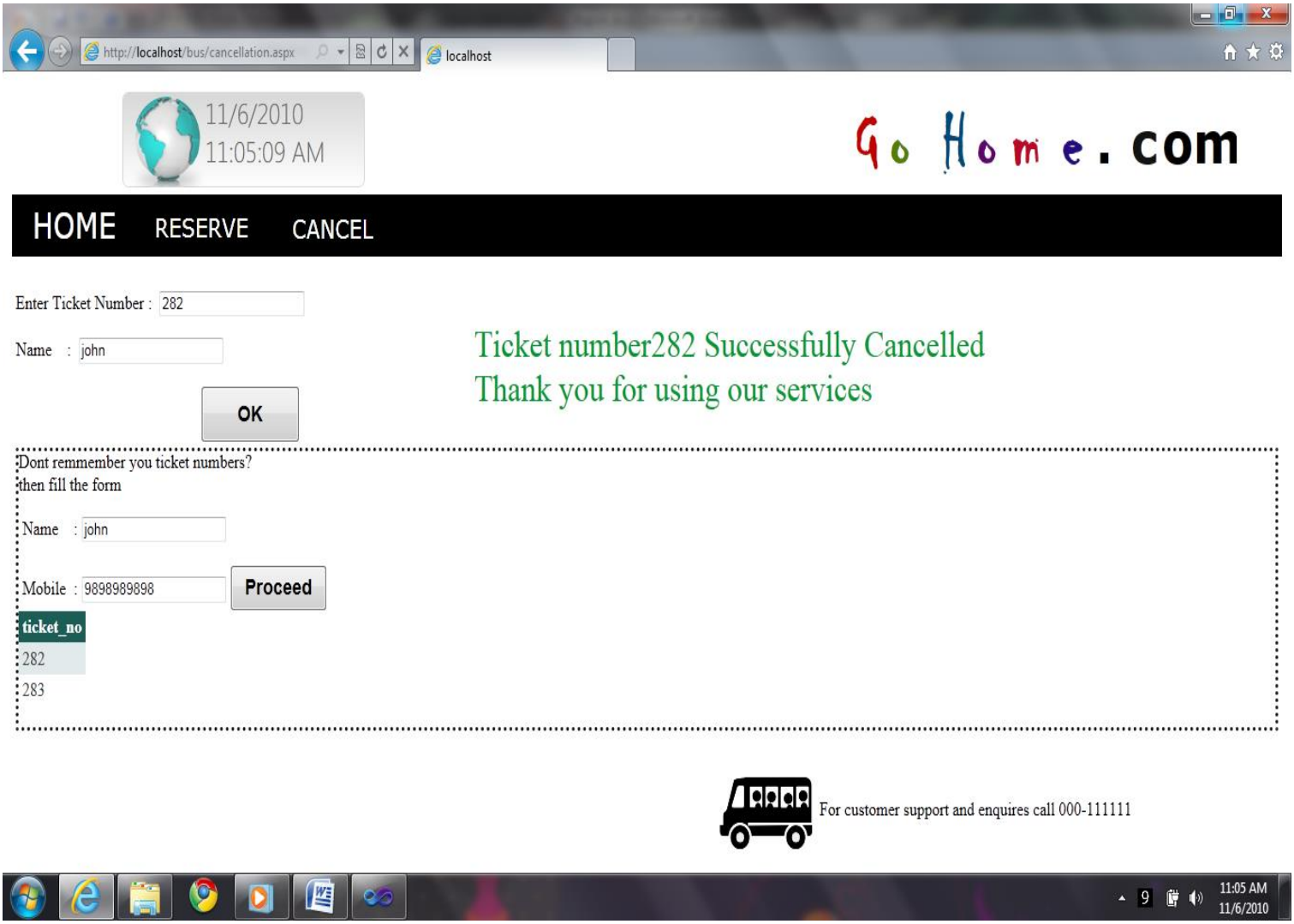
//ASP CODE FOR ENQUIRE DETAILS DISPLAY:

```
<asp:SqlDataSource ID="Bus_name" runat="server"
                                    ConnectionString="<%$ ConnectionStrings:Sql2008 %>"
                                    SelectCommand="SELECT b.bname,r.rid,(b.max_seats-
(select count(rid) from reserves where rid=r.rid)) as Available_seats
from route r,bus b

where r.bid=b.bid and r.from_location=@from_location and r.to_location=@to_location  and
r.departure_date=@departure_date">
                                    <SelectParameters>
                                        <asp:ControlParameter ControlID="DropDownList1"
Name="from_location"
                                            PropertyName="SelectedValue" />
                                        <asp:ControlParameter ControlID="DropDownList2"
Name="to_location"
                                            PropertyName="SelectedValue" />
                                        <asp:ControlParameter ControlID="DropDownList3"
Name="departure_date"
                                            PropertyName="SelectedValue" />
                                    </SelectParameters>
                                </asp:SqlDataSource>
```

6.2. Cancellation

6.2.1. Screen layout:



6.2.2. Programming code:

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

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


    }

    protected void Button1_Click(object sender, EventArgs e)
    {
        SqlConnection cn = new SqlConnection("Data Source=" +
CommonToALL.CommmonClass.server + ";Integrated Security=True");
        cn.Open();
        string cmd = "SELECT ticket_no from reserves where ticket_no='"+TextBox1.Text+"'
and pid in (select pid from passenger where name='"+Name_textbox.Text+"')";
        SqlCommand sqlcmd = new SqlCommand(cmd, cn);


        try
        {
        SqlDataReader DbReader = sqlcmd.ExecuteReader();

        DbReader.Read();

            int ticket = Int32.Parse(DbReader.GetValue(0).ToString());
            ThankBOX.Text = "Ticket number" + TextBox1.Text + " Successfully Cancelled
</br> Thank you for using our services";

            DbReader.Dispose();
            cmd = "delete from reserves where ticket_no='"+TextBox1.Text+"'";
            sqlcmd = new SqlCommand(cmd, cn);
            sqlcmd.ExecuteNonQuery();

        }
        catch (Exception ex)
        {
            ThankBOX.Text = "Invalid Ticket number or name";
        }
        sqlcmd.Dispose();
        cn.Close();
    }
}
```

6.3.Bill Generation:

6.3.1. Screen layout:

6.3.2. Programming code:

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

public partial class Ticket : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Name.Text = CommonToALL.CommmonClass.name;
        Mobile.Text = CommonToALL.CommmonClass.mobile;
        Notickets.Text = CommonToALL.CommmonClass.no_tickets;
        Fare.Text = CommonToALL.CommmonClass.Fare;
        Fare.Text = Fare.Text.TrimEnd('0');
        Fare.Text = Fare.Text.TrimEnd('.');
        Email.Text = CommonToALL.CommmonClass.email;
        Amount.Text =(Int32.Parse(Notickets.Text) * Int32.Parse(Fare.Text)).ToString();
    }
}
```

7. TESTING

Testing is an investigation to provide clients with information about the quality of the product. Testing method can be implemented at each phase in the development process, however the most test effort is employed after the requirements have been defined and coding process has been completed.

TEST CASES	DETAILS OF TEST	TEST RESULT	TEST CERTIFICATE
Mobile_textbox	Checking of numerals in phone no.	It is not taking alphabets and accept mobile with 10 digits only.	Test successful.
Email_textbox	Testing of Regular expression of email id	It was not accepting email id without '@'.	Test successful.
Form_fill	Testing of null values in entry columns.	Alarm is raised and data was not accepted if any item is null.	Test successful
Cancellation	Testing invalid ticket number cancellation.	Alarm is raised and modification is done to data base.	Test successful

8. CONCLUSION

This facility is helpful for the users and the organization as well. This is a simple yet effective technology which helps the users to access the service concurrently from different places.

Advantages:

1. It reduces the burden of the administrator to maintain numerous data of passengers.
2. It reduces wastage of time as it is fast and simple.
3. It is low cost.
4. Improves the efficiency of the organization.

9. BIBLIOGRAPHY

✓ Printed referential:

1. The Reference material for C# by **Srikanth Pragada.**

✓ Electronic Referential:

The following websites were used for information gathering about the project. These websites also gave adequate information for the project to be successfully completed.

1. http://www.srikanthtechnologies.com/
2. http://www.microsoft.com/NET