Comparison between ASP.NET and PHP – Implementation of a Real Estate Web Application

Master Thesis in Software Engineering
School of Innovation, Design and Engineering

Authors:
Sneha Prabha Chandran (scn10003@student.mdh.se)
Mridula Angepat (mat10001@student.mdh.se)

09 September 2011

Supervisor: Frank Lüders
frank.luders@mdh.se

 Examiner: Kristina Lundqvist
kristina.lundqvist@mdh.se
Abstract

The main aim of the thesis is to compare two web development technologies. For this purpose, a real estate application is developed using Active Server Pages (ASP.NET) with C-Sharp (C#) and Hypertext Preprocessor (PHP) as languages. The platforms used are Microsoft Visual Studio 2010 for ASP.NET and Eclipse Helios 2010 for PHP developers. Two different servers implemented in these platforms comprising the Sql server 2008 for ASP.NET and MySQL for PHP. A real estate web application is developed with three main panels which are the user panel, advertiser panel and admin panel. The same functional requirements are used in two web technologies. When comparing two technologies, ASP.NET is found to be more efficient and reliable than PHP. During the development process it was found that, PHP is a simple scripting language in comparison to the .NET language C#. However, ASP.NET provides built in tools, function and controls which helps in faster development of code as compared to PHP.
Table of Contents

1. Introduction ......................................................................................................................................... 6
2. Background .......................................................................................................................................... 6
3. Requirements ...................................................................................................................................... 6
4. Functionality of the Real Estate Web Application .............................................................................. 7
   4.1 Search Property option ................................................................................................................. 7
   4.2 Registration option ...................................................................................................................... 10
   4.3 Login option ................................................................................................................................. 11
   4.4 Admin panel ................................................................................................................................ 12
5. The Database Structure ..................................................................................................................... 14
6. Implementation in ASP.NET .............................................................................................................. 15
   6.1 ASP.NET ....................................................................................................................................... 15
   6.2 .NET Framework .......................................................................................................................... 16
   6.3 Sql Server 2008 ............................................................................................................................ 16
   6.4 Visual Studio 2010 ....................................................................................................................... 16
   6.5 Structure of files and Implementation of the classes ..................................................................... 17
   6.6 Master pages ................................................................................................................................ 18
   6.7 ASP.NET Content pages ............................................................................................................... 19
      6.7.1 LocationMaster.aspx.cs ........................................................................................................ 19
      6.7.2 Property_M.aspx.cs .............................................................................................................. 20
      6.7.3 AboutUs.aspx.cs ................................................................................................................... 21
      6.7.4 ContactUs.aspx.cs ................................................................................................................. 21
      6.7.5 Login.aspx.cs ......................................................................................................................... 21
      6.7.6 ChangePassword.aspx.cs ...................................................................................................... 21
      6.7.7 EditProfile.aspx.cs ................................................................................................................ 22
      6.7.8 Register.aspx.cs .................................................................................................................... 23
      6.7.9 Home.aspx.cs ........................................................................................................................ 23
      6.7.10 SearchResults.aspx.cs ....................................................................................................... 23
6.7.11 SearchDetails.aspx.cs .........................................................................................................24

7. Implementation in PHP ...................................................................................................................24
   7.1 Apache Server 2.5.9....................................................................................................................24
   7.2 MySQL Server and connectivity................................................................................................25
   7.3 PHP in Eclipse editor..................................................................................................................26
   7.4 Structure of files in PHP ............................................................................................................26
      7.4.1 Userproperty.php .............................................................................................................29
      7.4.2 TypeMaster.php ...............................................................................................................30
      7.4.3 Top.php ............................................................................................................................30
      7.4.4 Profile.php .......................................................................................................................30
      7.4.5 Change password.php ......................................................................................................30
      7.4.6 Location.php ....................................................................................................................30
      7.4.7 Login.php ........................................................................................................................31
      7.4.8 Logout.php .........................................................................................................................32
      7.4.9 Aboutus.php .....................................................................................................................32
      7.4.10 Contactus.php .................................................................................................................32
      7.4.11 Homepage.php .................................................................................................................32
      7.4.12 Registrationform.php ......................................................................................................33
      7.4.13 Search.php .......................................................................................................................33
      7.4.14 Viewproperty.php ............................................................................................................33

8. Google API in ASP.NET and PHP .....................................................................................................34

9. Comparison in ASP.NET and PHP ...................................................................................................35

10. Conclusion and future work ..........................................................................................................37

11. References ......................................................................................................................................37

12. Appendix.........................................................................................................................................39
   Appendix A. Visual studio 2010: ASP.NET ..................................................................................39
      A1. LocationMaster ......................................................................................................................39
1. Introduction
This thesis presents a comparison work performed between two web development technologies: ASP.NET and PHP. In order to perform the comparison, a web application in the form of a real estate website is developed in both languages with the same requirements. ASP and PHP were selected for comparison because these two languages are very popular web programming languages and provides support in the development of dynamic database driven websites.

In the real estate website, two different types of users are considered, a client who is interested in buying a property and an advertiser who wants to sell the property. There is an admin panel for the administrator to control the various activities. The background section, describes the platforms used for development and the technologies used in detail. Further, the implementation in both languages is explained. The conclusions obtained after comparing the two languages and future work in this area is also discussed towards the end.

2. Background
Nowadays most of the websites can be categorised as dynamic database driven websites. In a dynamic database driven website, the web page is connected to the database by means of programming and every time the information stored in the database changes, it is reflected in the web page. ASP.NET and PHP are considered as two competing technologies for developing dynamic websites. ASP.NET was developed by Microsoft and the version 1.0 of the .NET Framework was released in 2002. PHP was designed by Rasmus Lerdof in 1995 and later on developed by the PHP Group.

Other technologies used for developing dynamic websites are JavaServer Pages (JSP), PERL and Cold Fusion. JSP was released by Sun Microsystems in 1999. JSP technology is used in the development of web applications that are server and platform independent. PERL is considered as a general-purpose dynamic programming language which is used for graphic programming, network programming, finance and other applications. ColdFusion is used in the development of rich Internet applications and includes advanced features for enterprise integration.

3. Requirements
A set of basic requirements was considered by us before the development of the web application and performing the comparison between ASP.NET and PHP. These basic
requirements were gradually increased during the development phase. The various modules considered as a part of requirements for the development of the real estate web application are as follows,

- Login module for Admin, Users and Advertisers.
- Registration module for both the Users and Advertisers.
- A module to Search and View the property details.
- In the View property details section, more details about the property along with a property image are provided. The location of the property is indicated in the Google map with the help of Google API.
- In the Home page, a Latest Update section provides information about the recently advertised properties by the Advertiser.
- An Admin module where the administrator can control all the activities.
- An Advertiser module where the advertiser can publish the properties.
- About us and Contact us module.

Another part of the requirements were to use an IDE for development which renders almost similar features, so that it helps in the comparison process during development in ASP.NET and PHP. Thus Visual Studio was considered as IDE for development in ASP.NET and Eclipse PDT was considered for PHP. It was also a part of the requirement that servers most commonly used with these platforms need to be used and thus SQL Server 2008 was used for ASP.NET and MySQL for PHP.

4. Functionality of the Real Estate Web Application
The Home page is the welcome page of the web site. The Home page (Figure 1) has the Registration, Login, Search Property, About Us and the Contact Us options. The Latest Updates section includes a moving slider of the images of the recently advertised properties.

4.1 Search Property option
Search Property comprises of the price, rooms, type and location options. The client can specify the requirements in the corresponding fields and search accordingly by clicking on
the submit button. The client is then redirected to the Search Results page (Figure 2). The Search Results page displays the list of properties as per the requirements of the client. When client clicks on view details link, the client is redirected to Search Details page (Figure 3). In this page more details of the property is provided with the property images, a small description about the property, the location, the various features available and more. The exact location of the property is shown in the Google map with the help of Google API (Figure 3).

Figure 1. Home Page
Figure 2. Search Results Page
4.2 Registration option

When the client clicks on the registration option, the client is redirected to a registration form (Figure 4) where the client can register as the following:

- User
- Advertiser

User: If the client wants to buy a property from the website, the client should register as a user and fill in the details in the registration form. The client will be provided with a username and password after registration.
Advertiser: If the client wants to sell the property, client should register as an advertiser and fill in the details in the registration form. The advertiser will be provided with a username and password after registration.

![Registration Page](image)

**Figure 4. Registration Page**

### 4.3 Login option

The Login is provided for the admin, users and advertisers. When the user logs in (Figure 5), the user is redirected to the Home Page for searching the property. When advertiser logs in, advertiser is redirected to advertiser page having the Location Master, Type Master, Property Master, My Profile and Change Password. The admin panel pages are displayed when the administrator logs in.

Location Master specifies the location of the property; Type Master specifies the type of property like row house or apartment. In the Property Master, the advertiser fills in all the details of the property to be advertised. Images of the property can also be uploaded. In Change Password, the password can be changed.
When the admin logs in with the Login option, admin is redirected to the admin panel (Figure 6). This panel has the options: Location Master, Type Master, Property Master, My Profile and Change Password. In the Location Master the admin has the right to add a new location, update an existing location or delete a location. Type of the property can be updated or deleted by the admin in the Type Master. In the Property Master page the admin can view the properties registered by the different advertisers. The admin has the right to delete or edit the property advertisements. The right to delete different property is provided only to the admin. In My Profile and Change Password the admin can change the user information and password.
Figure 6. Admin panel (Property Master)
5. The Database Structure

The database of our web application has five tables (Figure 7). They are the Location table (tblLocation _M), the Login table (tblLogin _M), the Property details table (Property _M, tblProperty_ Md), the Type table (tblType _M) and the User table (tblUserType_M).

- **tblLocation _M** - This table stores the location name of the property.
- **tblLogin_M** - The Login table stores the login details of user and advertiser.
- **tblProperty_M** - The Property details table stores the property information such as name, price, address, zip code, and number of rooms, description, features and image ids related to the properties.
- **tblProperty_Md** - This table stores the image id related to each property id and the corresponding image path. One property can have more than one image. To maintain the integrity of the database the table name is given as tblProperty_Md.
- **tblType_M** – The type of the property is stored in this table as the properties can be of different types like row house, apartment, independent house etc.
- **tblUserType** - This table stores details of the different users. The users include the clients and the advertisers.

The relations between the tables are shown in Figure1. Five foreign keys are used in the database.  tblUserType_M has a foreign key in the table tblLogin_M. The tblProperty_M has a foreign key in tblProperty_Md, as the images of the property are stored in tblProperty_Md. The table tblLocation_M has a foreign key in tblProperty_M. tblType_M has a foreign key in tblProperty_M. tblLogin_M has a foreign key in tblProperty_M.
6. Implementation in ASP.NET

6.1 ASP.NET

ASP.NET is considered to be the next era of web development. The main purpose of ASP.NET is to create dynamic web pages. It allows C# and VB.NET for programming the web applications. One striking feature of ASP.NET is web services. Web services denote several pieces of application working in servers around the world. Using Extensible Markup Language (XML) is very simple in ASP.NET. The built in tools of ASP.NET supports XML for data storage, manipulation and configuration. The enormous class library in ASP.NET encapsulates many common functions. ASP.NET is built on the Common Language Runtime (CLR) [1] which allows programmers to write code using .NET supported languages.


6.2 .NET Framework
.NET Framework runs on Microsoft Windows. It eventually supports many programming languages there by allowing interoperability [2]. All programming languages in .NET are supported by .NET Library. Programs written for .NET Framework are executed in an environment called Common Language Runtime. The first beta version of .NET Framework was released in the year 2000 and the latest version released is .NET Framework 4 along with visual studio 2010. The .NET Framework 4 is used in our application. It is highly compatible and has an improved security model.

6.3 Sql Server 2008
Steps for creating database connection [3]:

1. Open the web application in Visual Studio 2010 and click view.
2. Select server explorer.
3. In Server explorer window, right click the database connection.
4. Select create new sql server database.
5. Give server name. Select sql server authentication and give user name and password.
6. Give server name and select ok.

We have used LINQ (Language Integrated Query) to access data from the database [4]. Classes representing the database are created and these classes are used to interact with data. These data classes are created under the App_code folder in solution explorer. Linq is easy to develop when comparing to stored procedures.

6.4 Visual Studio 2010
We have used Visual Studio 2010 ultimate version for developing our real estate web application. Before the start of the application we have referred other real estate websites to get an idea of its functions and working. Visual Studio is an IDE (Integrated Development Environment) [5] produced by Microsoft for .NET software. It is used to develop console and graphical user interface applications along with the windows forms application. The integrated debugger works as a source-level debugger and machine-level debugger. Application debugging is one of the more challenging tasks but Visual Studio 2010 debugging feature makes it easy to determine the state of the application and analyze the bugs [6].

Steps for creating the web application project:
2. Select the ASP.NET empty web site. In the left column select Visual C# under installed templates.
3. Give the name for the application and browse the path. Click OK. A new web application project is created.

### 6.5 Structure of files and Implementation of the classes

The Figure 8 displays the structure of files of ASP.NET in solution explorer. The Figure 9 displays the code of admin master page in Visual Studio 2010.

![Figure 8. Structure of files in ASP.NET](image)

**Figure 8. Structure of files in ASP.NET**
Master pages

Master page creates a template for the other pages of the web application. The look and layout needed for the other pages can be created in a single Master page. Design can be developed in the content place holder. The Master page and the content page merge to produce a combination, which has a template of the Master page and a design of the content page.

One main advantage of using Master page in ASP.NET is the layout changes can be made in the Master pages instead of making changes in other pages of the application. Multiple Master pages can be created for different purposes. User and the admin can have individual Master pages. Our application consists of two Master pages.

Steps to create Master pages:

1. Right click the web application in the solution explorer. Click add new item.
2. In the left panel select Visual C# from the installed templates.
3. Select the Master page and give name.
4. Give the path and then click the add button.
5. The Master page is created successfully.

The Adminmaster.master has the header. The header includes the Location Master, Type Master, Property Master, My Profile, and Change Password. These are created using the menu bar. A logout button is also placed in the Admin Master page.

The real estate web application has two Master pages. The Adminmaster.master is used to display the pages in the admin module. The UserMaster.master displays all the user panel pages in the user module.

Class1.cs - The Class1.cs is used to fetch the data from the database in the form of data table and to convert the resultant data of data row source to data table.

Web Config – The connection to the sql database is done in the Web Config file.

```xml
<appSettings>
  <add key="ConnectionString" value="server= server name; DataBase= database name; user id=sa; password= database password ;"/>
</appSettings>
```

**6.7 ASP.NET Content pages**

Steps to create the ASP.NET Content Pages:

1. Right click the web application in the solution explorer. Click add new item.
2. In the left panel select Visual C# from the installed templates.
3. Select web form and give name.
4. Select the Master page in the check box.
5. Click the add button.

After the creation of the content pages, two files with and extension .aspx and .aspx.cs are created by default in Visual Studio. The .aspx files are for meant for HTML design where ASP.NET controls are used. The .aspx.cs files are meant for coding. i.e. the content provided in the .aspx files is automatically handled and manipulated in aspx.cs files.

**6.7.1 LocationMaster.aspx.cs** - The location name of the property is entered. The location can be edited or deleted as per the needs of admin and advertiser.

```csharp
private void ShowData()
```
dbContext = new DataClassesDataContext();
var locData = from LocData1 in dbContext.tblLocation_M select LocData1;
grdLocation.Columns[0].Visible = true;
grdLocation.DataSource = locData;
grdLocation.DataBind();
grdLocation.Columns[0].Visible = false;
}

This file handles various aspects related to location. As mentioned above in the description of the real estate website, in the LocationMaster page, the user can add, update or delete the locations. The ShowData() function performs the main functionalities. The location is retrieved from the database and stored into the variable locData. Then visibility is made true and false accordingly.

6.7.2 Property_M.aspx.cs - contains the details of the property like name, price, address1, address2, zip code, no. of rooms, features, description, location and type. The aspx page has three buttons namely save, update and cancel. These buttons are provided for saving and updating the property details entered by the advertiser.

var objProp = from objProp1 in dbContext.tblProperty_M select new 
{
    objProp1.PropertyId, PropertyTitle = objProp1.PropertyTitle, Price = objProp1.Price,
    PriceType = objProp1.PriceType, Address1 = objProp1.Address1, Address2 =
    (objProp1.Features), Description = HttpUtility.HtmlDecode (objProp1.Description),
    LocationId = objProp1.LocationId, TypeId = objProp1.TypeId, Zipcode =
    objProp1.Zipcode
};

In the PropertyMaster page the all the property details are retrieved from the table Property_M and are stored in the variable objProp.

if (! IsPostBack)
{
    if (Session["Id"] != null)
    {
        FillDdl();
        ShowData();
        ClearData();
        ShowDataImage();
    }
}
ClearDataImage();

else
{
    Response.Redirect("~/Login.aspx")
}

In the above code, FillDdl() is used for filling the list of location and type in drop down list. ShowData() is used to display the uploaded property data. ClearData() is used to clear the values in the property page. ShowDataImage() displays the uploaded images in the property page. ClearDataImage() clears the images.

6.7.3 AboutUs.aspx.cs - This page displays the information about the real estate web application.

6.7.4 ContactUs.aspx.cs – This page displays the contact information of the administrator of the real estate web application.

6.7.5 Login.aspx.cs - The Login.aspx prompts the user to enter the username and the password. Registered users or registered advertisers can log into their existing account.

```csharp
var ChkLogin = (from ChkLogin1 in dbContext.tblLogin_M where ChkLogin1.UserName == txtUserName.Text && ChkLogin1.Password == txtPassword.Text && ChkLogin1.DeleteFlag == false select ChkLogin1).SingleOrDefault();
```

The Login details of the user, advertiser and the admin are retrieved from the database table Login_M. The Login details are stored in the variable ChkLogin.

6.7.6 ChangePassword.aspx.cs - is responsible for changing the password for registered users. The user is prompted to enter the old password and new password.

```csharp
if (!IsPostBack)
{
    if (Session["Id"] != null)
    {
        Label1.Text = ""; // Displays Password Changed Successfully
    }
}
```
else
{
    Response. Redirect("~/Login.aspx");
}

The IsPostBack gets a value which indicates the page is being rendered. The Response. Redirect method causes the browser to redirect the client to Login.aspx if the password is not changed.

6.7.7 EditProfile.aspx.cs - contains the user information like name, address, postal code, city, country, phone and mobile. The user can edit or change the details in his profile.

private void ShowData()
{
    if (Session["Id"] != null)
    {
        dbContext = new DataClassesDataContext();
        tblLogin_M locData = (from LocData1 in dbContext.tblLogin_Ms
                        where LocData1.UserId == Convert.ToInt32(Session["Id"]).ToString())
            select LocData1).SingleOrDefault();
        txtName.Text = locData.Name;
        txtAddress1.Text = locData.Address1;
        txtAddress2.Text = locData.Address2;
        txtCity.Text = locData.City;
        txtCounty.Text = locData.County;
        ddlCountry.Text = locData.Country;
        txtPhone.Text = locData.Phone;
        txtMobile.Text = locData.Mobile;
    }
    else
    {
        Response. Redirect("Login.aspx");
    }
}

The private ShowData() displays the user details of the logged in users. The details are stored in Login_M table.
6.7.8 **Register.aspx.cs** - This page displays registration details for the advertisers and the users. Two radio buttons are displayed for user and advertiser. An advertiser or the user clicks the required button for registering their details. The Register.aspx has the name, address, zip code, city, country, phone, mobile, user name and password fields for entering the data. The country is displayed using the drop down list control.

6.7.9 **Home.aspx.cs** – The home page has the search property of the real estate web application. The search property displays the price, rooms, type and location with the drop down list control. These options are provided for the user to search a property according to their requirement. When the user clicks the submit button, the page redirects to the SearchResults.aspx page displaying the results related to search.

```csharp
Var objLoc = from objLoc1 in dbContext.tblLocation_M select objLoc1;
ddlLocation.DataSource = objLoc;
ddlLocation.DataTextField = "Name";
ddlLocation.DataValueField = "LocationId";
ddlLocation.DataBind();
ddlLocation.Items.Insert(0, "All Areas");
ddlLocation.Items[0].Value = "0";
```

The variable objLoc fills the location details from the Location_M table in the drop down list of the search property of the home page.

6.7.10 **SearchResults.aspx.cs** - The search details page displays a list of properties in accordance with the search. To the right of the property listing is the view details link. When the client clicks the link, the user is redirected to Search Details page.

```csharp
if (!IsPostBack)
{
    ArrayList SearchData = new ArrayList();
    context = new DataClassesDataContext();
    string[] items = Request.QueryString["items"].ToString().Split(':');
    foreach (string item in items)
    {
        SearchData.Add(item);
    }
}
```
In the above code ArrayList is used for displaying the list of properties according to the filtration from home page.

6.7.11 SearchDetails.aspx.cs - The Search Details page has the property image along with the property details. Google API is used to locate the property address.

```csharp
tblProperty_M objProp = (from objProp1 in context.tblProperty_M
where objProp1.PropertyId == Convert.ToInt32(Request.QueryString["Id"].ToString())
select objProp1).SingleOrDefault();
```

The property details are retrieved from the table Property_M and stored in object objProp. When the user selects the SearchDetails page the property details are displayed.

```csharp
var objImage = from objImage1 in context.tblProperty_Md
where objImage1.PropertyId == Convert.ToInt32(Request.QueryString["Id"].ToString())
select objImage1;
```

The variable objImage displays the images from the Property_Md table. These images are displayed along with the property details.

7. Implementation in PHP

PHP is a general-purpose scripting language used to create dynamic web pages which contains dynamic content or images. This feature makes PHP as one of the best languages for server-side programming and is used on web servers. PHP can be deployed on most web servers and almost every operating system free of charge [7]. Thus, PHP is considered as competitor to Microsoft's Active Server Pages (ASP) in the area of server-side programming.

7.1 Apache Server 2.5.9

The Apache HTTP server project is a robust server which provides free availability of source code. Apache server is one of the popular web servers which support scripting languages like PHP to create dynamic pages. The initial release of Apache was made in the year 1995. PhpMyAdmin is a free software tool written in PHP intended to handle the administration of MySQL over the World Wide Web. It supports a wide range of operations with MySQL [8].

Steps for installing Apache Server 2.5.9 [9]:

1. Download apache server 2.5.9 and run the executable file.
2. The Appserv package has four components which includes
   - Apache HTTP Server
   - MySQL Database
   - PHP
   - PhpMyAdmin.

Select all the four components in the checkbox and click next.

3. In the next step of the installation, enter server name, administrator email address and default HTTP Port as 80.

4. In the next window enter root password for MySQL Server. Again re enter the root password. In MySQL server setting set the character sets and collations to UTF-8 unicode. Click install.

5. After installation click finish.

7.2 MySQL Server and connectivity
MySQL can be characterised as a reliable open source which is robust by nature. MySQL is also known as RDBMS (Relational Database Management System) [10]. MySQL is the most preferred database for web applications. Many third party applications (front ends) integrate with MySQL for developing database structure. One among the popular front end is PhpMyAdmin which is used in our application. It is a web based front end developed in PHP. MySQL function is used to connect to the MySQL database [11].

Our MySQL database is connected using the following code:

```php
<?
    $test_localhost = "localhost";
    $test_username = "root";
    $test_password = "admin";
    $test_database = "realestate";
    $con = mysql_connect ("localhost","root","admin");
    if (!$con)
    {
```
The root and the admin are the username and password which are given during the installation of MySQL server.

### 7.3 PHP in Eclipse editor

Eclipse provides an IDE (integrated development environment) which supports development in multiple languages. Mostly eclipse is used for developing applications in Java. With the help of its extensible plug-in system, it supports various programming languages such as C, C++, Perl, PHP etc. The Eclipse IDE used for PHP is known as Eclipse PDT, where PDT stands for PHP Development Tools. We have used the Helios version of the Eclipse PDT for PHP. PDT consists of specific modules related to PHP to customise Eclipse to work with PHP. Eclipse does not have any specific installation program [12]. Thus, it is very simple to install and use it.

The Eclipse PDT 2.2.0 was used by us. It can be installed from the below link,

http://www.eclipse.org/pdt/downloads/

After clicking on download, a file “eclipse-php-helios-win32.zip” is downloaded. Unzip the file and save it to a suitable location. The eclipse.exe file can be found inside the Eclipse folder. The eclipse.exe file can be even dragged to the start menu or desktop to make running the application easier.

After downloading the Helios version of eclipse for PHP developers, open the IDE, check if the perspective is PHP or else change it to PHP. This can be done by clicking on Windows and selecting PHP Perspective.

### 7.4 Structure of files in PHP

After running eclipse.exe, the IDE is opened with the welcome screen as the start page. The user is prompted to select a workspace. All the details related to the project are stored in this workspace. In the PHP Explorer, right click and select new PHP Project. A suitable name for the project is given and Finish option is selected. All the other files related to the project are created in this main project file. After right clicking on the project folder and then selecting
new option, different types of files such as PHP file, css file are selected and created. The file structure for this project in Eclipse is given in Figure 10. Figure 11 displays the change password page in Eclipse PDT.

Figure 10: structure of files in php
Admin folder Creation:

1. Right click the real estate folder and click new >> Folder.

2. Give the folder name and select the parent folder which is real estate in our application.

3. Click finish. A folder is created under the PHP project file.

To create a PHP file under the admin folder:

1. Right click the admin folder and click new >> PHP file.
2. Give file name and source folder as realestate/admin.

3. Click finish. A PHP file is created under the admin folder.

A short description of the PHP files included in the admin section is given below,

7.4.1 Userproperty.php - This file deals with the property details which include name, address 1, address 2, city, type, price, number of rooms, description, location and zip code. Predefined variable $POST is used to post the property details as given in the below portion of the code.

```php
if (isset($_POST['btnsubmit']))
{
    $str_name   = trim ($_POST['name']);
    $str_address1 = trim ($_POST['address1']);
    $str_address2   = trim ($_POST['address2']);
    $str_city     = trim($_POST['city']);
    $str_type    = trim($_POST['type']);
    $str_price   = trim ($_POST['price']);
    $str_room   = trim($POST['rooms']);
    $str_description  = trim($POST['description']);
    $str_location    = trim($POST['location']);
    $str_zipcode    = trim($POST['zipcode']);
}
```

In order to display the corresponding errors when any of the property detail is not entered, an array is created to store the list of error messages to be displayed.

```php
$error_list  = array();
if (strlen(trim($str_name))==0)
{
    $error_list['name'] = "Please, Enter Property Name";
}
if(strlen(trim($str_address1))==0)
```

```php
```
$error_list['address'] = "Please, enter Address"

if(strlen(trim($str_Bedroom)) == 0)
{
    $error_list['Bedroom'] = "Please, Enter Rooms";
}

7.4.2 TypeMaster.php – This file contains the code to update the table TypeMaster according to the changes performed which include deleting a type, updating an existing type and adding a new type.

7.4.3 Top.php – The main role of this file is to display the header section which contains the menus location master, type master, property master, my profile and change password in admin panel as well as the advertiser panel. A logout button is also placed in the header section.

7.4.4 Profile.php – This file displays the profile information of the advertiser or the admin which includes name, address, zip code, and city, country and contact number.

7.4.5 Change password.php – This file helps the advertiser or admin in changing the password entered by the advertiser or admin. Three options such as old password, new password and confirm password are required to be entered for changing the old password. For locating the particular user id in the UserMaster table, the following MySQL query is used and the result is stored.

```php
$wsql = ( "select * from Usermaster where userid = $userid" );
$result = mysql_query($sql);
```

7.4.6 Location.php – This file is responsible for the controls in the location master section. Advertisers and the admin can edit or delete the existing locations. Also new locations can be added and deleted. The priority is given to both the advertisers and the admin.
Grid control [13] is used in our application for the purpose of editing, selecting and deleting in tables. In our application grid is used for the editing and deleting of the locations. Data grid is used and is created in array CreateFromArray ($array).

```php
public static function CreateFromArray($array)
{
    if (isset($_GET['orderDir']) AND !empty($_GET['orderBy'])) {
        RGrid::$orderby['column']    = $_GET['orderBy'];
        RGrid::$orderby['direction'] = $_GET['orderDir'];
        uasort($array, array('RGrid', '_sortArray'));
    }
    $grid = new RGrid($array);
    return $grid;
}
```

7.4.7 Login.php – This file deals with the various login options available. The login option is common for the user, advertiser and the admin. When user logs in, user is redirected to the Home page to view the advertised properties. When the advertiser and admin logs in, the login page redirects to the Location Master in the admin.

```php
$sql="select * from usermaster where username="$_POST[username]";
$sql. = " and password ="$_POST[password]";
$rs=mysql_query($sql);
if(mysql_num_rows($rs)<=0)
{
    header("Location:login.php?msg=Invalid Username and Password");
}
else
{
    $r=mysql_fetch_row($rs);
    $_SESSION[user_id]= $r[0];
    $_SESSION[username]= $r[1];
    $_SESSION[password]= $r[2];
    $type = $r[3];
    if($type == "advertiser")
    {
        header("Location:profile.php");
    }
    else
    {
        if($type == "user")
        {
            header("Location:home.php");
        }
    }
}
```
7.4.8 Logout.php – A logout option is placed in all the pages and this file contains the code for logout function

7.4.9 Aboutus.php - The about us page provides more details of the website and this file is responsible for that.

7.4.10 Contactus.php – This file is responsible for the contact us page which provides contact details of the administrators of the website. This page is created for the users to contact the owners.

7.4.11 Homepage.php – This file is responsible for the welcome page of the website.

The links to the search property, registration, login, about us and contact us options are displayed. Latest updates section displays the recently advertised properties in a moving slider. The recent property images are extracted from the PropertyMaster table using the mysql query and displayed.

```php
<?php
$resultimages = mysql_query("SELECT img.image imagesname
FROM propertymaster AS a
INNER JOIN usermaster ON a.userid = usermaster.userid and usermaster.Type = 'advertiser'
INNER JOIN images As img ON img.prop_id = a.id and img.setdefault = 2
WHERE a.Deleted = 0");
//Code for moving slider with images
while($row = mysql_fetch_array($resultimages))
{
    echo "<img src="$row [imagesname]" width=100px height=100px>
"; &nbsp ;<?
```
7.4.12 Registrationform.php - This file is responsible for displaying the registration form the users and advertisers to register. Corresponding string variables are declared to store name, address, zip code, city, country, phone, mobile, username and password. The built-in $_POST function [14] is used to collect values in a form with method = post. Information sent using the POST method is invisible and has no limits on the amount of information sent.

```php
$str_name = trim($_POST['name']);
$str_address1 = trim($_POST['address']);
$str_zipcode = trim($_POST['zipcode']);
$str_city = trim($_POST['city']);
$str_country = trim($_POST['country']);
$str_phone = trim($_POST['phone']);
$str_mobile = trim($_POST['mobile']);
$str_username = trim($_POST['username']);
$str_password = trim($_POST['password']);
```

If any of the information is not provided by the user or the advertiser an array is used to display the error list. The information of the user and the advertiser are inserted in to the table UserMaster using the following query.

```sql
$sql="insert into usermaster
(name, address, zipcode, country, city, mobile, phone, username, password) values
("$name", "$Address", "$zipcode", "$country", "$city", "$mobile", "$phone", "$username", "$password")" ;
```

7.4.13 Search.php – This file displays the search options available under Search Property in the Home Page. The data entered by the user against price, rooms, type, location are passed in the mysql query to search in the tables and the results are displayed in the Search Results page.

7.4.14 Viewproperty.php - This file is responsible for displaying the property. When the user selects view details from the Search Results page, user is redirected to the property details page. The MySQL query with the help of inner joins extracts the details related to the property.
8. Google API in ASP.NET and PHP
We have used Google API in both the technologies to locate the location of the property address in the Google map. The Google maps API give an embedded Google map in the web page. We have used the places API to locate the location. A Google account is necessary to get a maps API key.

Once after logging in with our Gmail account in Google maps API Family, an API key is generated. The following example displays an API key [15].

Example:

```
“ABQIAAAAAd1q4qP02Z7sZOS8F6D2KyBSgCGX2UxBxfxJCVRAltWJJYKH4ZxQjYXe
NFQEqXFuhHL0r1em02yGhBg”
```

In ASP.NET, We have implemented the API key in our SearchDetails page. The design layout for implementing the Google map is done in the SearchDetails.aspx page where the API key is also implemented. The code is done in the SearchDetails.aspx.cs for the API to locate the address of the property. 90% of the location of the property address was located by the API. At some instant the API did not recognise the some of the special characters. The following code which contains the API key implementation is used in the SearchDetails.aspx page.

```
<cc2:GoogleMap ID="GoogleMap1" runat="server" Address="Fairmule House, Waterson Street, Shoreditch, E2 8HT" Zoom="15"
EnableScrollWheelZoom="true" EnableDragging="false"
EnableGoogleBar="false" EnableGoogleMapState="true"
Height="210px" Width="233px"
DefaultMapView="Normal">
</cc2:GoogleMap>
```

In PHP we have implemented Google API in view property page. Both the design layout and the API key are used in the same page. The location and the address are given in parameters.
9. Comparison in ASP.NET and PHP

The main idea behind this thesis is to find an efficient web development technology which is best suitable for developing web applications. The scope of the web application projects are increasing tremendously in the present generation. Web applications are popular due to the omnipresence of web browsers where web browsers are used as client. From our experience in working with both the web development technologies, we have measured certain factors and they are listed in a tabulated format (Figure 12). These factors are observed during the planning, development and execution phase.

When implementing a real estate website in two different web technologies, ASP.NET is found to have more advantages in comparison to PHP. The amount of code is less in ASP.NET, as the code and HTML are together which makes it more flexible and easy. The ASP.NET framework is well complemented with rich toolbox and designer. ActiveX Data Object for .NET (ADO.NET) [16] is a part of the .NET Framework and is used to access data from the database. ASP.NET works well with ADO.NET using data binding. Though PHP is easy to learn and code, it consumes more time for larger applications, as lines of code is required more. PHP supports, only partial encapsulation and polymorphism when compared to ASP.NET. PHP is easily customized to MySQL where the database access is very easy. One of the most striking features of ASP.NET is Master page. The design and the content for all the pages is done in a single Master page and in PHP the header, style and footer were coded in all the pages. The data grid containers in ASP.NET display the output more easily when compared to PHP grid. Customisable and reusable controls which are known as user controls [17] can be created in ASP.NET. User controls are substantially easier to create as they are reusable which makes development process easier.
One of the important aspects noted during the development was the memory usage. ASP.NET seems to be expensive with respect to the memory usage which can be a serious problem when developing a larger web application. The reason for the limitation in ASP.NET is due to the longer code path. When considering PHP in terms of memory usage it was more efficient than ASP.NET [18]. The reason is PHP has a small code path which means the server side code is less when compared to ASP.NET.

<table>
<thead>
<tr>
<th>Measures</th>
<th>ASP.NET</th>
<th>PHP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>ASP programs need IIS to be installed on Windows platform servers.</td>
<td>PHP programs runs on Apache on Linux and Unix servers which is free.</td>
</tr>
<tr>
<td></td>
<td>Database connectivity is expensive as MS-SQL is a Microsoft product.</td>
<td>MySQL is used as database, which is freely available.</td>
</tr>
<tr>
<td>Installation</td>
<td>ASP, C#, Visual Studio and .NET are a whole package where the installation is easy and is not dependent on third party frameworks.</td>
<td>More reliable on third-party frameworks and IDE for installation.</td>
</tr>
<tr>
<td>Memory Usage</td>
<td>Longer code path leads to expensive memory usage.</td>
<td>Smaller code path leads to efficient memory usage.</td>
</tr>
<tr>
<td>Support and Resources</td>
<td>Improvements and updates are made by available number of Microsoft developers alone. Less support available to solve new challenges.</td>
<td>More open source developers and resources available for PHP. More support available from the PHP forum.</td>
</tr>
</tbody>
</table>
Editors and tools | Mostly Microsoft Visual Studio is used to develop .NET Applications. | Editor independent. Have access to extensive number of editors.
---|---|---
Development and coding | Average development time is more for smaller websites. | Average development time is less for smaller websites.

Figure 12. Comparison table between ASP.NET and PHP

10. Conclusion and future work
From our experience with both technologies we conclude that ASP.NET has many advantages, as it is significant in its integration with windows server and programming tools. The web application in ASP.NET is easier to create, debug and deploy. Automatic management of server controls which behaves more like windows control are the key factors of ASP.NET. Even though PHP is considered to be the oldest scripting language it lacks in advancement of tools and controls. Few short comings of PHP which we observed during our development are its lack of exceptions, error handling techniques that interrupt the normal flow of the program. From our analysis on the two technologies ASP.NET in visual studio platform proves to be efficient and flexible for web applications.

When considering the real estate application in future perspective, more functionality and new features can be added. The present structure of the database can be increased with more requirements. Day to day there are new updates and new versions introduced in ASP.NET and PHP, the platform which is more compatible, reliable and which reduces the security issue can be considered for developing a web application.

11. References
http://en.wikipedia.org/wiki/Asp.net


http://www.quackit.com/sql_server/tutorial

http://en.wikipedia.org/wiki/Microsoft_Visual_Studio


http://en.wikipedia.org/wiki/PHP


http://www.appservnetwork.com/

http://en.wikipedia.org/wiki/Mysql


http://www.w3schools.com/Php/php_post.asp

http://code.google.com/apis/maps/index.html

http://en.wikipedia.org/wiki/ADO.NET

http://msdn.microsoft.com/en-us/library/y6wb1a0e.aspx

[18] *Migrating from PHP to ASP.NET*, Accessed on 29.08.2011
12. Appendix

Appendix A. Visual studio 2010: ASP.NET

A1. LocationMaster

```csharp
public partial class Admin_LocationMaster : System.Web.UI.Page
{
    DataClassesDataContext dbContext;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            if (Session["Id"] != null)
            {
                ShowData(); //To Display the location details from database
                ClearData(); // To Clear the values of controls in forms
            }
            else
            {
                Response.Redirect("~/Login.aspx");
            }
        }
    }
    private void ShowData()
    {
        //To Display the location details from database
        dbContext = new DataClassesDataContext();
        var locData = from LocData1 in dbContext.tblLocation_Ms
                      select LocData1;

        grdLocation.Columns[0].Visible = true;
        grdLocation.DataSource = locData;
        grdLocation.DataBind();
        grdLocation.Columns[0].Visible = false;
    }
    private void ClearData()
    {
        //To Clear the values of controls in a form
        divSave.Visible = true;
        divUpdate.Visible = false;
        txtLocation.Text = "";
        btnSave.Visible = true;
        btnUpdate.Visible = false;
    }
}```
lblld.Text = "";
}

protected void btnSave_Click(object sender, EventArgs e)
{
    try
    {
        //Save the new Location details

        lblMessage.Text = "";
        if (Page.IsValid)
        {
            dbContext = new DataClassesDataContext();

            tblLocation_M objLocation = new tblLocation_M()
            {
                Name = txtLocation.Text
            };
            dbContext.tblLocation_Ms.InsertOnSubmit(objLocation);
            dbContext.SubmitChanges();

            lblMessage.Text = "Data Saved Successfully";
            ShowData();
            ClearData();
        }
    }
    catch (Exception ex)
    {
        lblMessage.Text = "Data Not Saved";
    }
}

protected void grdLocation_RowCommand(object sender, GridViewCommandEventArgs e)
{
}

protected void grdLocation_RowDeleting(object sender, GridViewDeleteEventArgs e)
{
    //Used to Delete the Existing Selected Location

    lblMessage.Text = "";
    try
dbContext = new DataClassesDataContext();
tblLocation_M objLocation = (from objLocation1 in dbContext.tblLocation_M
    where objLocation1.LocationId == Convert.ToInt32(grdLocation.Rows[e.RowIndex].Cells[0].Text)
    select objLocation1).SingleOrDefault();
dbContext.tblLocation_Ms.DeleteOnSubmit(objLocation);
dbContext.SubmitChanges();

lblMessage.Text = "Data Deleted Successfully";
ShowData();
ClearData();

} 
catch (Exception ex)
{
    lblMessage.Text = "Data Not Deleted";
}

protected void grdLocation_SelectedIndexChanged(object sender, EventArgs e)
{
    //Used to Select Existing Location for Update or Delete

txtLocation.Text = grdLocation.SelectedRow.Cells[1].Text;
lblId.Text = grdLocation.SelectedRow.Cells[0].Text;
btnUpdate.Visible = true;
btnSave.Visible = false;
divUpdate.Visible = true;
divSave.Visible = false;

}
protected void btnUpdate_Click(object sender, EventArgs e)
{
try
{
    // Used to Update existing selected Location
    lblMessage.Text = "",
    if (Page.IsValid)
    {
        dbContext = new DataClassesDataContext();
        tblLocation_M objLocation = (from objLocation1 in dbContext.tblLocation_Ms
            where objLocation1.LocationId == Convert.ToInt32(grdLocation.Rows[e.RowIndex].Cells[0].Text)
            select objLocation1).SingleOrDefault();
        dbContext.tblLocation_Ms.DeleteOnSubmit(objLocation);
        dbContext.SubmitChanges();
        lblMessage.Text = "Data Deleted Successfully";
        ShowData();
        ClearData();
    }
}


catch (Exception ex)
{
    lblMessage.Text = "Data Not Deleted";
}
where objLocation1.LocationId == Convert.ToInt32(lblId.Text)
select objLocation1).SingleOrDefault();

objLocation.Name = txtLocation.Text;

dbContext.SubmitChanges();

lblMessage.Text = "Data Updated Successfully";
ShowData();
ClearData();
}
}
catch (Exception ex)
{
    lblMessage.Text = "Data Not Updated";
    WriteFile(ex);
}

protected void btnCancel_Click(object sender, EventArgs e)
{
    //Clear the Values
    lblMessage.Text = "";
    ClearData();
}

A2. TypeMaster

public partial class Admin_TypeMaster : System.Web.UI.Page
{
    DataClassesDataContext dbContext;

    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            if (Session["Id"] != null)
            {
                ClearData(); //Used to Clear form Values
                ShowData(); //Used to Display Type Information
            }
            else
            {
                //Response.Redirect("~/Login.aspx");
            }
        }
    }

    protected void btnSave_Click(object sender, EventArgs e)
    {
        lblMessage.Text = "";
    }
try
{
    //Used to Save the Type Information
    if (Page.IsValid)
    {
        dbContext = new DataClassesDataContext();

        tblType_M objType = new tblType_M()
        {
            Name = txtName.Text
        };
        dbContext.tblType_Ms.InsertOnSubmit(objType);
        dbContext.SubmitChanges();

        lblMessage.Text = "Data Saved Successfully";
        ShowData();
        ClearData();
    }
}
catch (Exception ex)
{
    lblMessage.Text = "Data Not Saved";
}

private void ShowData()
{
    //Used to Display the type Information
    dbContext = new DataClassesDataContext();
    var objType = from objType1 in dbContext.tblType_M
                   select objType1;

    grdType.Columns[0].Visible = true;
    grdType.DataSource = objType;
    grdType.DataBind();
    grdType.Columns[0].Visible = false;
}

private void ClearData()
{
    //Used to Clear the Form Values
    divSave.Visible = true;
    divUpdate.Visible = false;
    txtName.Text = "";
    lblId.Text = "";
    btnSave.Visible = true;
    btnUpdate.Visible = false;
}
protected void btnUpdate_Click(object sender, EventArgs e)
{
    //Used to Update the selected type Information

    lblMessage.Text = "";
    try
    {
        if (Page.IsValid)
        {
            dbContext = new DataClassesDataContext();
            tblType_M objType = (from objType1 in dbContext.tblType_Ms
                                 where objType1.TypeId == Convert.ToInt32(lblId.Text)
                                 select objType1).SingleOrDefault();

            objType.Name = txtName.Text;
            dbContext.SubmitChanges();

            lblMessage.Text = "Data Updated Successfully";
            ShowData();
            ClearData();
        }
    }
    catch (Exception ex)
    {
        lblMessage.Text = "Data Not Updated";
        WriteFile(ex);
    }
}

A3. SearchDetails

public partial class SearchDetails : System.Web.UI.Page
{
    DataClassesDataContext context;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            context = new DataClassesDataContext();

            tblProperty_M objProp = (from objProp1 in context.tblProperty_Ms
                                      where objProp1.PropertyId ==
                                      Convert.ToInt32(Request.QueryString["Id"].ToString())
                                      select objProp1).SingleOrDefault();
        }
    }
}
var objImage = from objImage1 in context.tblProperty_Mds
    where objImage1.PropertyId == Convert.ToInt32(Request.QueryString["Id"]).ToString()
    select objImage1;

Class1 objCls = new Class1();

DataTable dt = new DataTable();
dt = objCls.ToTable(context, objImage);

ViewState["dt"] = dt.Copy();
string oldSearch = "";
if (Session["SearchData"] != null)
{
    oldSearch = Session["SearchData"].ToString();
}
else
{
    //Menu1.InnerHtml = ""
    Menu2.InnerHtml = ""
    for (i = 1; i <= dt.Rows.Count - 1; i++)
    {
        div1.InnerHtml = div1.InnerHtml + "<a href=Admin" +
        dt.Rows[i]["ImagePath"].ToString() + " runat=server rel=lightbox[roadtrip] title=''>";
    }
    MainTitle.InnerHtml = objProp.PropertyTitle;
    plab_address1.InnerHtml = objProp.Address1.Replace("","" + objProp.Address2.Replace("","" + objProp.Zipcode;
    plab_price1.InnerHtml = "<span class=colour>Price : </span><strong>€ " +
    objProp.Rental + " per " + objProp.RentalType + "</strong>";
    plab_bedrooms1.InnerHtml = "<span class=colour>Number of bedrooms : </span>" +
    Convert.ToString(objProp.Bedroom);
    myFeatures.InnerHtml = HttpUtility.HtmlDecode(objProp.Features);
    myDetails.InnerHtml = HttpUtility.HtmlDecode(objProp.Description);
    int i;
    div2.InnerHtml = "";
    Menu1.InnerHtml = "";
    Menu2.InnerHtml = "";
    div2.InnerHtml = div2.InnerHtml + "<a href=Admin" +
    dt.Rows[0]["ImagePath"].ToString() + " runat=server rel=lightbox[roadtrip] title=''>";
    myDetails.InnerHtml = dt.Rows[0]["ImagePath"].ToString() + " width=485px height=350px"></a> ";
    for (i = 1; i <= dt.Rows.Count - 1; i++)
    {
        div1.InnerHtml = div1.InnerHtml + "<a href=Admin" +
        dt.Rows[i]["ImagePath"].ToString() + " runat=server rel=lightbox[roadtrip] title=''>";
div1.InnerHtml = div1.InnerHtml + "<img src=Admin" +
dt.Rows[i]["ImagePath").ToString() + " width=100 height=40"></a ";
}

GoogleMap1.Address = objProp.Address1.Replace("","" ) + ", " +
objProp.Address2.Replace("","" ) + ", " + objProp.Zipcode;

Artem.Web.UI.Controls.GoogleMarker abc=new
Artem.Web.UI.Controls.GoogleMarker();
    abc.Address=objProp.Address1.Replace("","" ) + ", " +
objProp.Address2.Replace("","" ) + ", " + objProp.Zipcode;

    GoogleMap1.Markers.Add(abc);

}

protected void Button1_Click(object sender, EventArgs e)
{
}

}

protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        lblMessage.Visible = false; //Used to display the Registration successful message
    }
}

protected void btnSubmit_Click(object sender, EventArgs e)
{
    try
    {
        //Used to Save the Registration Data
        if (Page.IsValid)
{ 
    lblMessage.Visible = false;
    dbContext = new DataClassesDataContext();

    var objType = (from objType1 in dbContext.tblLogin_M 
                   where objType1.UserName == txtUserName.Text 
                   select objType1).SingleOrDefault();

    if (objType == null)
    {
        int user;
        if (rbUser.Checked == true)
        {
            user = 2;
        }
        else
        {
            user = 3;
        }
        tblLogin_M objLogin = new tblLogin_M()
        {
            Name = txtName.Text,
            Address1 = txtAddress1.Text,
            Address2 = txtAddress2.Text,
            ZipCode = txtZipCode.Text,
            City = txtCity.Text,
            County = txtCounty.Text,
            Country = ddlCountry.Text,
            Phone = txtPhone.Text,
            Mobile = txtMobile.Text,
            UserName = txtUserName.Text,
            Password = txtPassword.Text,
            UserTypeId = user,
            DeleteFlag = false,
            SDate = System.DateTime.Now
        };
        dbContext.tblLogin_Ms.InsertOnSubmit(objLogin);
        dbContext.SubmitChanges();
        Session["Name"] = txtName.Text;
        Session["UserTypeId"] = user;
        Session["Id"] = objLogin.UserId;
    }
if (rbUser.Checked == true)
{
    Response.Redirect("Home.aspx");
}
else
{
    Response.Redirect("Admin/EditProfile.aspx");
}
else
{
    lblMessage.Visible = true;
}

A5. EditProfile

public partial class Admin_EditProfile : System.Web.UI.Page
{
    DataClassesDataContext dbContext;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            if (Session["Id"] != null)
            {
                ShowData(); //To Display the login User Information
            }
            else
            {
                //Response.Redirect("~/Login.aspx");
            }
        }
    }
    private void ShowData()
    {

//To Display the login User Information

dbContext = new DataClassesDataContext();
tblLogin_M locData = (from LocData1 in dbContext.tblLogin_Ms
    where LocData1.UserId ==
    Convert.ToInt32(Session["Id"].ToString())
    select LocData1).SingleOrDefault();

    txtName.Text = locData.Name;
    txtAddress1.Text = locData.Address1;
    txtAddress2.Text = locData.Address2;
    txtCity.Text = locData.City;
    txtCounty.Text = locData.County;
    ddlCountry.Text = locData.Country;
    txtPhone.Text = locData.Phone;
    txtMobile.Text = locData.Mobile;

protected void btnSubmit_Click(object sender, EventArgs e)
{
    //To Update the details of Logged in user

dbContext = new DataClassesDataContext();
    int user;

    tblLogin_M locData = (from LocData1 in dbContext.tblLogin_Ms
    where LocData1.UserID ==
    Convert.ToInt32(Session["Id"].ToString())
    select LocData1).SingleOrDefault();

    if (locData != null)
    {
        locData.Name = txtName.Text;
        locData.Address1 = txtAddress1.Text;
        locData.Address2 = txtAddress2.Text;
        locData.City = txtCity.Text;
        locData.County = txtCounty.Text;
        locData.Country = ddlCountry.Text;
        locData.Phone = txtPhone.Text;
        locData.Mobile = txtMobile.Text;
dbContext.SubmitChanges();

else
{


}


Appendix B. Eclipse PDT: PHP

B1. View Property

$sqldata = "SELECT a.id, a.name, a.Price, a.Address1, a.Bedroom, a.Features, a.Description, typemaster.name AS Propertytype, locationsmaster.seo AS location, a.Deleted, images.image as imagename, images.setdefault FROM propertymaster AS a INNER JOIN locationsmaster ON a.LocationId = locationsmaster.id INNER JOIN typemaster ON a.TypeId = typemaster.id INNER JOIN images ON a.id = images.prop_id WHERE a.id = '$proid' And setdefault= 2";
$sqlresult = mysql_query ($sqldata);
while ($row = mysql_fetch_array($result))
{


Room

ContentPlaceHolder1_plab_address1

Address1
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Price:** £\(?php echo $row['Price']; ?> per

**Number of bedrooms:** \(?php echo $row['Bedroom']; ?>

**Features:** \(?php echo $row['room']; ?> Room
<?php echo $row['Features']; ?>
</div>

<div id="plab_details">
  <span class="colour">Details : </span>
  <div id="Details">A lovely <?php echo $row['Bedroom']; ?> house <?php echo $row['Description']; ?>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</div>

</div>
</div>
</div>
</div>
</table>

<br />
<br />

<tr>
  <td>
    </td>
  <tr>
  </table>

</tbody>
</table>

</tbody>
</table>

</td>
</tr>
</tbody>
</table>

<br />
<br />

</td>
</tr>
</tbody>
</table>
<?php $resultimages = mysql_query("SELECT img.image imagesname FROM propertymaster AS a INNER JOIN usermaster ON a.userid = usermaster.userid and usermaster.Type = 'advertiser' INNER JOIN images As img ON img.prop_id = a.id and img.setdefault = 2 WHERE a.Deleted =0");

while($row = mysql_fetch_array($resultimages)) {
    echo "<img src=".$row[imagesname]." width=100px height=100px">";?&nbsp;<?
}
</marquee></div>
</tr>
</table>

B2. Home

</head>
<body>
<table class="style1">
<tr>
    <td width="200px">
        <img src="images/logo_RE.gif" id="Image1">
    </td>
    <td width="150px">
    </td>
    <td width="400px" align="right">
        <a href="regstationfrm.php">Registration</a> <a href="login.php">Login</a>
    </td>
</tr>
<tr align="top" rowspan="3" valign="top">
    <ul>
        <li><a href="home.php" class="first">Home</a></li>
        <li><a href="AboutUs.php">About Us</a></li>
        <li><a href="ContactUs.php">Contact Us</a></li>
    </ul>
</tr>
<tr colspan="2">
    <div>
        <table class="container" width="100%">
            <tr>
            </tr>
        </table>
    </div>
</tr>
<td colspan="2">
    <span id="hi" class="table1" style="color:#29291D;font-size:Medium;font-weight:bold;">
        Hi Welcome: <? echo $SESSION[username]; ?>
    </span>
</td>
</tr>
<tr>
    <td style="width: 50%">
        Search Property</td>
    <td style="width: 50%">&nbsp;</td>
</tr>
<tr>
    <td colspan="2">
        <table class="container" width="100%" style="font-size: medium">
            <form action="search.php" name="quicksearch" method="get">
            <tr>
                <td style="width: 25%">
                    Price
                </td>
                <td style="width: 30%">
                    <select class="style1" tabindex="4" id="Mini_Rent" name="Mini_Rent">
                        <option value="0">Minimum Price</option>
                        <option value="5000">100</option>
                        <option value="10000">200</option>
                        <option value="25000">300</option>
                        <option value="35000">500</option>
                        <option value="50000">600</option>
                        <option value="75000">800</option>
                        <option value="90000">1000</option>
                        <option value="100000">1200</option>
                    </select>
                </td>
                <td style="width: 10%" align="center">to</td>
            </tr>
            <tr>
                <td style="width: 30%">
                    <select class="style1" tabindex="4" id="Max_Rent" name="Max_Rent">
                        <option value="0">Maximum Rental</option>
                        <option value="5000">5000</option>
                    </select>
                </td>
            </tr>
        </form>
    </td>
</tr>
<table>
<thead>
<tr>
<th>Rooms</th>
<th>Option Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>$10,000</td>
</tr>
<tr>
<td>25000</td>
<td>$25,000</td>
</tr>
<tr>
<td>35000</td>
<td>$35,000</td>
</tr>
<tr>
<td>50000</td>
<td>$50,000</td>
</tr>
<tr>
<td>75000</td>
<td>$75,000</td>
</tr>
<tr>
<td>90000</td>
<td>$90,000</td>
</tr>
<tr>
<td>100000</td>
<td>$100,000</td>
</tr>
</tbody>
</table>

Minimum number of Bedroom:
- 0
- 1
- 2
- 3
- 4
- 5
- 6 or more

Maximum Number of Bedroom:
- 0
- 1
- 2
- 3
- 4
- 5
- 6 or more

Type

<table>
<thead>
<tr>
<th>Option Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum</td>
<td>Maximum</td>
</tr>
<tr>
<td>Bedroom</td>
<td>Bedroom</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6 or more</td>
<td>6 or more</td>
</tr>
</tbody>
</table>
Type

```php
<?php
$result = mysql_query("SELECT * FROM typemaster where deleted = 0 ORDER BY name");
while($row = mysql_fetch_array($result))
{
    echo "<option value=".$row[id].">$row[name]</option>";
}
?>
</select>
```

Location

```php
<? $result = mysql_query("SELECT * FROM locationsmaster Where deleted = 0 ORDER BY seo");
while($row = mysql_fetch_array($result))
{
    echo "<option value=".$row[id].">$row[seo]</option>";
}
?>
</select>
```

Search
<h2>
Latest Updates</h2>
<td>
<td>&nbsp;
</td>
</td>
</tr>
<tr>
<td colspan="2">
<marquee behavior="scroll" direction="up" scrollamount="1" width="100%" height="100px">
Latest Updates Latest Updates
Latest Updates Latest Updates
</marquee>
</td>
</tr>
</table>
</div>
</td>
</tr>
<tr>
<td colspan="2">
<div id="FooterSlide"><marquee><img src="images/ima1.jpg" width="100px" height="100px"></img>&nbsp;&nbsp;<img src="images/ima3.jpg" width="100px" height="100px"></img>&nbsp;&nbsp;<img src="images/ima10.jpg" width="100px" height="100px"></img>&nbsp;&nbsp;</marquee></div>
</td>
</tr>
</table>