REAL ESTATE WEB APPLICATION

by

RASHI CHOPRA

B.E., Medicaps Institute of Technology and Management, India, 2006

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2008

Approved by:

Major Professor
Dr. Daniel Andresen

Abstract

The Real Estate Web Application is an interactive, effective and revenue-generating website designed for the Real Estate Industry. The main objective of this application is to help the Real Estate Company to display unlimited number of property listings on the website.

The primary focus is to get familiar to .NET framework and code with ASP.NET and C# .NET to provide a featured GUI which contains sophisticated search engine for buyer's to search for property listings specific to their needs. The search engine not only provides an easy and convenient way to search for listings but also display the entire list of properties in a customized grid format. The buyer can then view the complete specification of each property listing with its features, description and photographs. The application also provides a drag and drop control to save a list of selected property listings while browsing other options on the Real Estate Website.

There are hundreds of Real Estate Websites on the World Wide Web but the intention of designing this application is to develop something new, innovative and efficient using latest technologies like AJAX, Java Script, etc which not only enhances the already existing search features available on the internet but also gets rid of their annoying and unessential features. The main emphasis lies in providing a user-friendly search engine for effectively showing the desired results on the GUI.

Table of Contents

List of Figures

List of Tables

Acknowledgement

I would like to thank my Major Professor Dr. Daniel Andresen for his constant help, encouragement and guidance throughout the project. I specially acknowledge his patience to help me figure out the right project to work on, providing me flexibility to implement my new ideas and provide valuable inputs for the project.

I would also like to thank Dr. Gurdip Singh and Dr. Mitchell L. Neilsen for their support and for graciously accepting to serve on my committee.

Dedication

I would like to dedicate this project to my parents Mr. Rajesh Chopra and Mrs. Deepti Chopra for their words of encouragement and helping me to get through the difficult times.

CHAPTER 1 - Introduction

1.1 Motivation

The motivation to develop Real Estate Web Application comes from my urge to learn Visual Studio .NET 2005 for building the business logic of the application, SQL server 2005 for database designing and using new web technologies like AJAX, Java Script for website designing. The most influential factor for selecting this application is to add some innovative features to the search engine of a Real Estate Website which can make the task of a property buyer easy to search for property listings.

This application provides a wide scope to include number of web controls to develop the search engine and allow them to implement using AJAX functionality. Performance of the search engine is the main factor which allows the buyers to search for listings with different combination features. Growing buyer needs should be taken in to concern with the new features to be included. Another challenge in this application is the effective storage / retrieval of property images from the SQL database.

Thus, the development of powerful search engine, displaying detailed summary of listings on the customized grid and drag - drop tool to select the properties are the main motivations for the project.

1.2 Project Overview

1.2.1 Project Introduction

There are large numbers of commercial real estate online information service providers offering a suite of commercial properties and services tailored to the national and local needs of the commercial investments industry. These online marketplaces have thousands of commercial real estate properties for sale and lease under various categories including commercial office space, industrial, single - family, multi - family, land, etc both for sale and lease as well. There purpose it to attract community of industry professionals including investors, property managers, landlords, appraisers, local and national buyers to select the properties with desired features.

In this increasingly demanding scenario for a platform which could help buyers to have a look at the available property listings with all its photographs, necessary specification and description, this application is an effort to provide such a web interface with some attractive and innovative features using the latest technologies. Real Estate Web Application is an AJAX enabled website with the latest AJAX controls giving attractive and interactive look to the web pages.

The most significant feature in these websites is the interactive search criterion which lets the buyer specify their requirements to get the correct set of records from the database. The search tool should be strong enough to include all the required features which a buyer may desire. Also, each search requires a post back call to the database to retrieve some set of records but making the website AJAX enabled prevents the annoying post backs. Thus, each call refreshes just the data grid to display the listings rather the whole web page. Some other new features like drag and drop tool to save the listings, retrieving related results for each search, etc make this application more feature rich.

1.2.2 Problems with existing systems

The purpose of creating this Real Estate Web Application is to outcast the discrepancies in hundreds of such existing systems on the World Wide Web. One of the basic problems with the existing systems is the non-interactive environment they provide to the users. Most of the applications involved in Real Estate business use some web template to put the content specific to their company and make it communicate with the database to search the listings. These templates simply use basic web controls to do this task making the web page non-interactive. On the other hand, the motive of this Real Estate Web Application is to allow the user to play with the search tool and create different combinatorial search criterion to perform exhaustive search.

Another problem in such applications designed so far is the use of traditional user interfaces which make continuous post backs to the server; each post back makes a call to the server, gets the response and then refreshes the entire web form to display the result. This scenario adds an extra trade off causing a delay in displaying the results. Making such applications AJAX enabled gets rid of these unnecessary delays letting the user to perform exhaustive search. The users of this application can easily feel the difference between the Ajax-empowered user interfaces vs. traditional user interfaces.

Scrutinizing the features of the existing systems reveals another problem, when the user tries to save some property listings; the user is forced to login as a buyer/user of this website. Once the user logs-in, he can then view this saved list. In contrast, this application uses the session state to maintain the list of saved property listings rather than making the user register first. As soon as the user performs a search, a new session is created and then lets the user select a listing, drag and drop it to the "Saved Search" tool. This tool then keeps track of this list until the session expires.

1.2.3 Objective

The foremost objective of this project was to give a different visualization styles to the Real Estate Website which has more features, attractive animations and all together a new look in contrast to the already existing websites. Usually in a real estate website, the property search page consist of traditional search style i.e. a set of textboxes / drop-downs to select a particular county or MLS# for the property to search along with other web controls to specify the number of beds / baths or any additional features they are looking in the property. In contrast, this website provides altogether a new visualization of the search page i.e. AJAX enabled controls with no post backs and additional functionalities like sliding bars to select the number of beds, baths, footage and price, drag and drop functionalities, animation enabled collapsible styles, etc.

The next objective of importance was to build an AJAX enabled real estate website which not only reduces the annoying post backs and loss of control focus, but also gives a faster and more interactive user interface. Moreover to make the website more features rich, features like customized grid, drag and drop tool, accordion panels and sliding bars were added to the website.

The third objective was to include a new feature of saving the selected property listing during a session in a drag – drop tool which similar websites don't have. Usually a real estate website forces a user to login as a buyer to save the selected listings so that when the user logs in the next time, those saved listings are accessible. But this site simply provides a toolbox to drag a selected number of listings and save them during the entire session.

1.2.4 Architecture

The Real Estate Web Application is built using a layered architecture where the total functionality can be divided into layers having different functionalities. The main layers include the database access layer, business logic layer and the presentation layer. Thus, this application follows the 3-Tier Architecture. The System Architecture is explained in detail in the further sections of this document.

1.3 Requirement Specification

1.3.1 Scope

The scope of MS project "Real Estate Web Application" is to enable the buyers to search for property listings online. The motive of developing this application is to design a feature rich search engine which can make the search of commercial land/properties an easy task.

1.3.2 Goal

Goal is to gain a good knowledge of the complete life cycle of the project development starting from the Requirement Gathering Phase to the Testing Phase. The implementation of this application also gives a hands-on experience in ASP.NET, C# .NET programming and also design efficient databases in SQL Server.

1.3.3 Assumptions

- User will have Internet Connection while using Real Estate Web Application.
- User will have Internet Explorer 5.0 or high IE version while using Real Estate Web Application.
- User will upload valid images for the Real Estate Web Application.

1.3.4 Environment

- The Real Estate Web Application will be written in C# .NET language.
- The development environment will be Microsoft Visual Studio .NET.
- The Real Estate Web Application will be tested on Windows XP platform.

CHAPTER 2 - Developer Platform

The Real Estate Web Application is developed on the .NET Platform using the .NET framework together with Microsoft SQL Server 2005. It is developed in the Visual Studio .NET 2005 integrated development environment. The goal of this chapter is to give an overview of the .NET Framework to show how this platform is architectured.

2.1 Microsoft .NET Framework

Microsoft .NET Framework is a software technology that is available with several Microsoft Windows Operating Systems. This computing platform simplifies application development in the highly distributed environment of the Internet. This framework not only provides an environment for building windows applications but also building, deploying and running web-applications and web-services. It includes a huge library of pre-coded solutions to common programming problems, a runtime or virtual machine that manages the execution of programs written specifically for the framework, and a set of tools for configuring and building applications. The .NET architecture is shown in Figure 2.1.

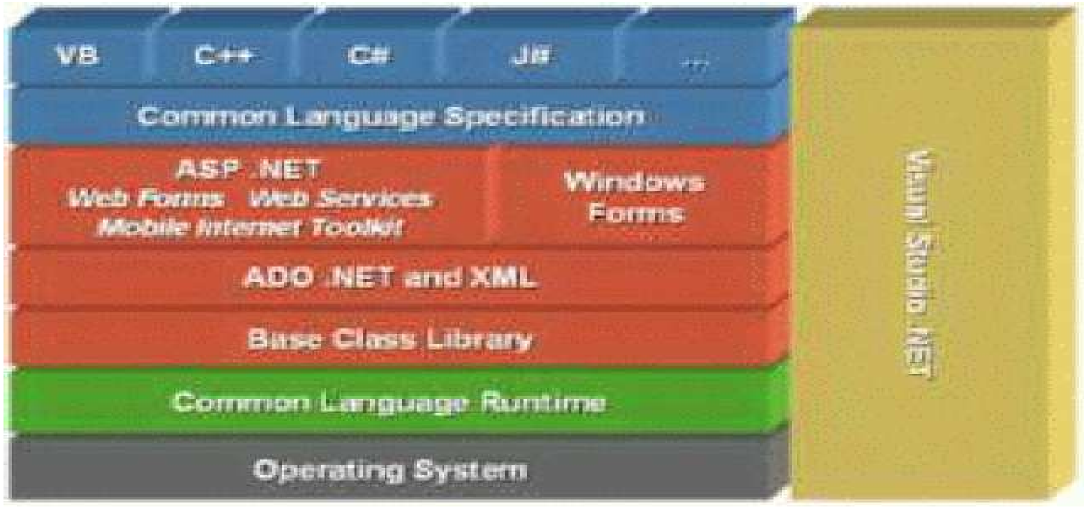


Figure 2.1 Microsoft .NET Architecture [7]

The core aspects of the .NET Framework lie within the Common Language Infrastructure, or CLI. CLI enables applications written in multiple .NET languages to operate in various environments without requiring program modifications. Its powerful features include application development and execution including exception handling, garbage collection, security and interoperability. Microsoft's implementation of the CLI is called the Common Language Runtime or **CLR**.CLR comprises of four major components: Common Type System (CTS), metadata system, base class library (FCL), file format (PE), intermediate language (CIL), and access to the underlying Windows operating system via the Win32 API. An overview of the CLR is shown in Figure 2.2.

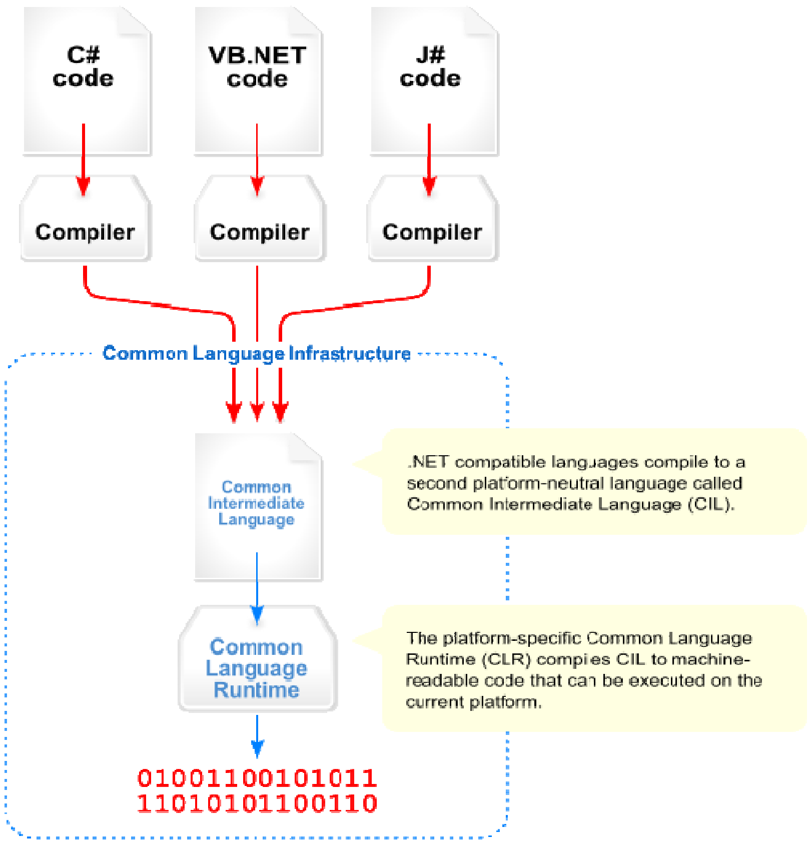

C# code
VB.NET code
J# code
Compiler
Compiler
Compiler
Common Language Infrastructure
Common Intermediate Language
.NET compatible languages compile to a second platform-neutral language called Common Intermediate Language (CIL).
Common Language Runtime
The platform-specific Common Language Runtime (CLR) compiles CIL to machine-readable code that can be executed on the current platform.
01001100101011
11010101100110

Figure 2.2 Overview of Common Language Infrastructure (CLI) [8]

.NET Common Type System [CTS] provides basic value types, type composition, type safety, objects, interfaces, and delegates. The Common Language Specification [CLS] is the subset of the Common Type System that all first class .Net languages need to share. The Common Language Runtime [CLR] is the managed code environment that everything else is built on. .Net is a garbage-collected environment but never interpreted - while .Net uses byte codes like Java, the Common Intermediate Language [CIL] code is always compiled, usually Just In Time [JIT] to be executed.

.NET metadata, in CLR, refers to certain data structures embedded within the Common Intermediate Language [CIL] code that describes the high-level structure of the code. It describes all classes and class members that are defined in the assembly, and the classes and class members that the current assembly will call from another assembly. The metadata for a method contains the complete description of the method, including the class, the return type and all of the method parameters.

.NET Virtual Execution System [VES] provides an environment for executing managed code. Its purpose is to provide the support required to execute the CIL instruction set. It also provides direct support for a set of built-in data types, defines a hypothetical machine with an associated machine model and state, a set of control flow constructs, and an exception handling model.

In .NET the intermediate language is complied **Just In Time** [JIT] into native code when the application or component is run instead of compiling the application at development time. It is a technique for improving the runtime performance of a computer program. In JIT environment, the source code is first translated to an intermediate representation, when this code is executed the runtime environment translates it into the native machine code i.e. the code is compiled when it is just to be executed thereby improving the runtime performance.

CHAPTER 3 - Technologies

This chapter includes the details of the latest technologies and tools used to build this application, their benefits and implementation details. The chapter also describes the interaction of these tools with the .NET Framework.

3.1 Tools and Technologies

The latest tools and technologies involved in building this website are: ASP.NET 2.0, Microsoft Visual Studio 2005, ADO.NET, SQL Server 2005, AJAX and Java Script.

3.1.1 ASP.NET 2.0

ASP.NET 2.0 is the next generation of Microsoft's Active Server Page [ASP] technology which provides a web application framework that allows programmers to dynamically build web pages, web applications and create web services. ASP.NET supports code written in compiled languages such as Visual Basic, C++, C# and Perl, and it features server from the content. The basis for ASP.NET 2.0 is built on the Common Language Runtime [CLR] described above. The dynamic content supported by ASP.NET is contained in the ASPX web forms which contain the static XHTML markup, as well as server-side Web Controls and User Controls where the developers place all the required static and dynamic content for the web page. ASP.NET 2.0 is an extension of ASP.NET 1.1 making it much easier for developers to develop dynamic Web Applications [1]. The major aspects of this technology include ASP.NET Compiler which includes strong typing, performance optimization and early binding. The task of this compiler is to compile all application components including pages and controls into an assembly which can then be used to service user requests. The Page and Controls Framework runs on the web server which then dynamically produces and render ASP.NET web pages which are completely object-oriented. Other services of ASP.NET include Security infrastructure, State-management facilities, Application configuration, Debugging support and XML Web Services framework.

Some of the new features of ASP.NET 2.0 are the introduction of Master Pages and Themes which allow the developer to define a common structure and interface elements, to be shared by many pages in the website. Standard controls for navigation, security, support for XML standards and interoperability with AJAX make web development easier and quicker.

3.1.2 Microsoft Visual Studio 2005

Microsoft Visual Studio provides an Integrated Development Environment (IDE) for programming that enables developers to quickly create data-driven and distributed applications using C# techniques and reusable controls. It can be used to develop both console and Graphical User Interface applications. The GUI applications include Windows Form applications, web applications and web services. Visual Studio comprises of a Code Editor, Debugger, Designer and other tools making the environment very interactive and easy to use. VS 2005 was upgraded to support all new features introduced in .NET Framework 2.0 as described above. The new features like intelligence and code refactoring supported by code editor, support for SQL Server 2005 databases, an improved environment for web publishing and load testing to see application performance under various sorts of user loads [2].

3.1.3 Microsoft SQL Server 2005

Microsoft SQL 2005 is a relational database management system from Microsoft. Its primary components are SQLOS implementing basic services required fir its server, including thread scheduling, I/O and Memory management, Relational Engine implementing relational database components having support for databases, tables, query and stored procedures and finally Protocol layer which exposes all these functionalities [3]. Being Transact-SQL as its primary query language a high performance data access is provided. This SQL Server version is Microsoft's next generation data management and analysis software delivering increased scalability, availability and security for enterprise data. The next major enhancement in SQL Server 2005 which the SQL Server 2000 lacks is the integration of a .NET compliant language such as C#, ASP.NET or VB.NET to build objects (stored procedures, triggers, functions, etc.), which enables the application to execute .NET code in the DBMS to take advantage of the .NET functionality. SQL Server 2005 has native capabilities to support encryption of data stored in user-defined databases which SQL Server 2000 lacks. Real Estate Web Application utilizes these features of Microsoft SQL Server 2005 to store the entire details of the property listings along with its images.

3.1.4 ADO.NET

.NET provides databases access through the set of tools and namespaces collectively referred to as Microsoft ADO.NET. There are three layers: the physical data store which could be a SQL database or an XML file, the Data Provider which interacts between the program and the database, the DataSet which stores disconnected data on the local memory. Figure 3.1 shows the ADO.NET model in action:

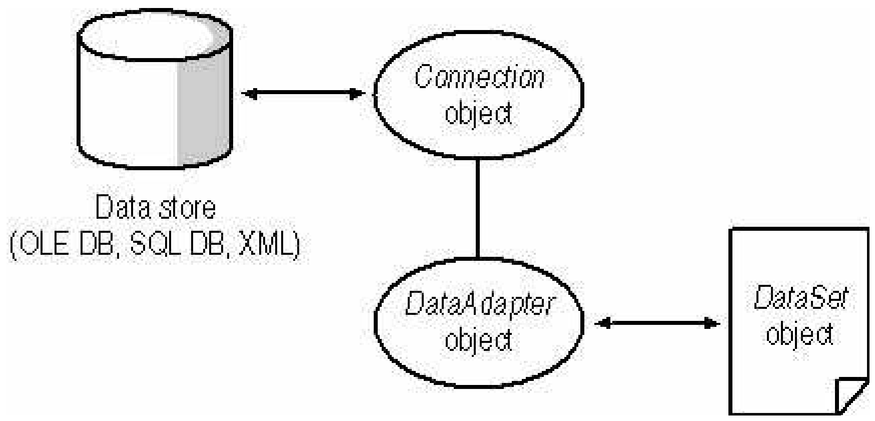


Figure 3.1 Overview of ADO.NET Model

3.1.5 AJAX / ASP.NET 2.0 support for AJAX

AJAX Asynchronous JavaScript and XML is a way of web development technique used for creating interactive and rich web applications. It provides a mechanism for the applications to interact with the Web Server asynchronously and retrieve the data in the background keeping the behavior of the application intact in the foreground. AJAX combines XHTML and CSS standards based presentation, interaction with the page using DOM model, data interchange with XML and XSLT and JavaScript altogether to incorporate dynamic behavior to the websites.

In contrast to the AJAX applications, the classic web application model works like this: Most user actions in the interface trigger an HTTP request back to a web server. The server does some processing — retrieving data, crunching numbers, talking to various legacy systems — and then returns an HTML page to the client. This approach does not make a great user experience as it makes the user wait for a long time. But AJAX makes all the difference by eliminating the start-stop-start-stop nature of interaction on the Web by introducing an intermediary — an Ajax engine — between the user and the server. Instead of loading a webpage, at the start of the session, the browser loads an Ajax engine — written in JavaScript and usually tucked away in a hidden frame. This engine is responsible for both rendering the interface the user sees and communicating with the server on the user's behalf. The Ajax engine allows the user's interaction with the application to happen asynchronously — independent of communication with the server. Thus, the user wait time is negligible. This scenario is evident in the Figure 3.4.

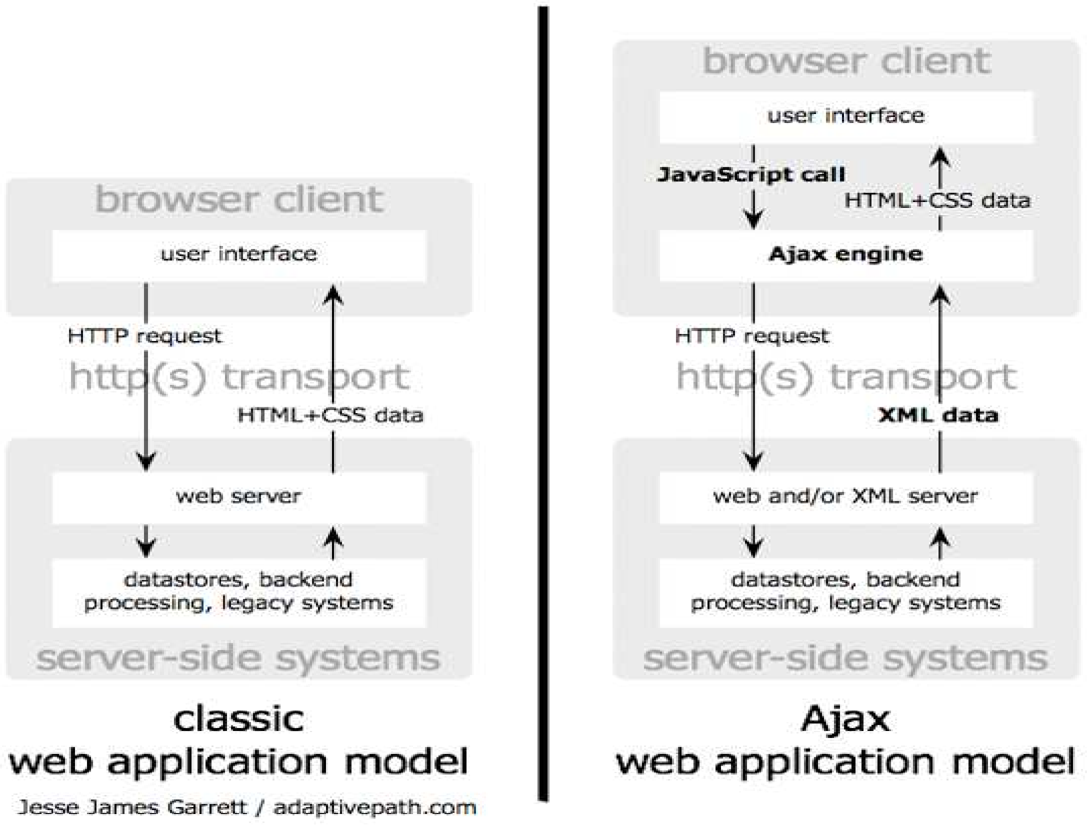


Figure 3.2 Comparisons of Classic Web Application Model and AJAX Web Application Model

In many applications the sections of pages need to be reloaded rather than reloading the whole application, AJAX fulfills this requirement allowing for much more responsive web applications. Also, the use of Ajax can reduce connections to the server by its extensive use of since scripts and style sheets which have to be requested once.

CHAPTER 4 - System Architecture

This chapter provides an Architectural Design for the Real Estate Web Application which represents a three tier architecture comprising of the Presentation Tier: list of web pages planned to be implemented in the application, the Middle Tier: class diagrams and description of each class and the Data Tier: comprising of the ER diagram and database schema.

4.1 System Architecture

The architecture of the system is based on the three-tier architecture. There are three logical layers: the Presentation Tier, the Business Tier, and the Data Tier. Each layer in the system is a reusable portion of code that performs a specific function. These layers are not only responsible for performing these functions but also interact with other layers to perform specific goals [4]. Like, in this application when the presentation layer needs to extract information from the backend database, it would utilize a series of layers to retrieve the data, rather than having the database calls embedded directly within it. The ASP.NET web site/web form is called the presentation layer because the content within it is viewable to the users of the site.

The middle tier called the Business Tier communicates between the presentation tier and the data tier. As mentioned above, the Presentation Layer could communicate with the data access layer directly, it usually goes through the business layer. This layer then validates the input conditions before calling a method from the data layer. This ensures the data input is correct before proceeding, and can often ensure that the outputs are correct as well. Most of the business logic of the application lies within the business layer, which makes this logic reusable across applications. This layer helps move logic to a central layer for “maximum reusability.”

The Data Tier is the key component for this application as it is responsible for storing the data corresponding to each property listing. This layer is a separate component whose sole purpose is to serve up the data from the database and return it to the caller. Through this approach, data can be logically reused, meaning that a portion of an application reusing the same query can make a call to one data layer method, instead of embedding the query multiple times. This is generally more maintainable. Now this data is returned using the ADO.NET technology which is built into the .NET framework, ADO.NET contains a mechanism to query data out of the database and return it to the caller in a connected or disconnected fashion [5]. This scenario can be explained as, when the buyer makes a request to search for a particular property listing in a particular area (specified by the zip code/city/state/MLS#); the business layer passes this information to the database in the form of a query. The database retrieves the data corresponding to the query and submits the results back to the presentation tier for display. The Figure 4.1 shows the three-tier architecture of the Real Estate Web Application.

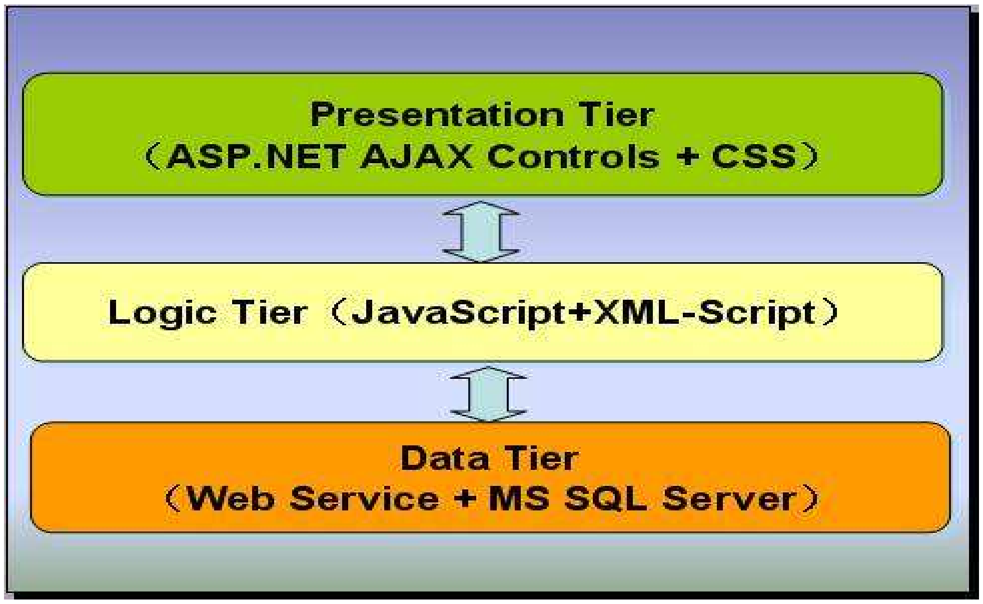

Presentation Tier
（ASP.NET AJAX Controls + CSS）
Logic Tier （JavaScript+XML-Script）
Data Tier
（Web Service + MS SQL Server）

Figure 4.1 3-Tier Architecture [10]

In the above figure, the Presentation Tier comprises of ASP.NET AJAX Controls which interact with the Logic Tier. The Presentation Layer can be elaborated to show what happens when the user requests something and calls to the server are made behind the scenes. This ASP.NET AJAX Framework uses the client-centric programming model. In this model, the large quantities of business logic from the server side are moved to the client side by introducing AJAX characteristics at the client-side controls. This asynchronous communication is shown in the Figure 4.2:

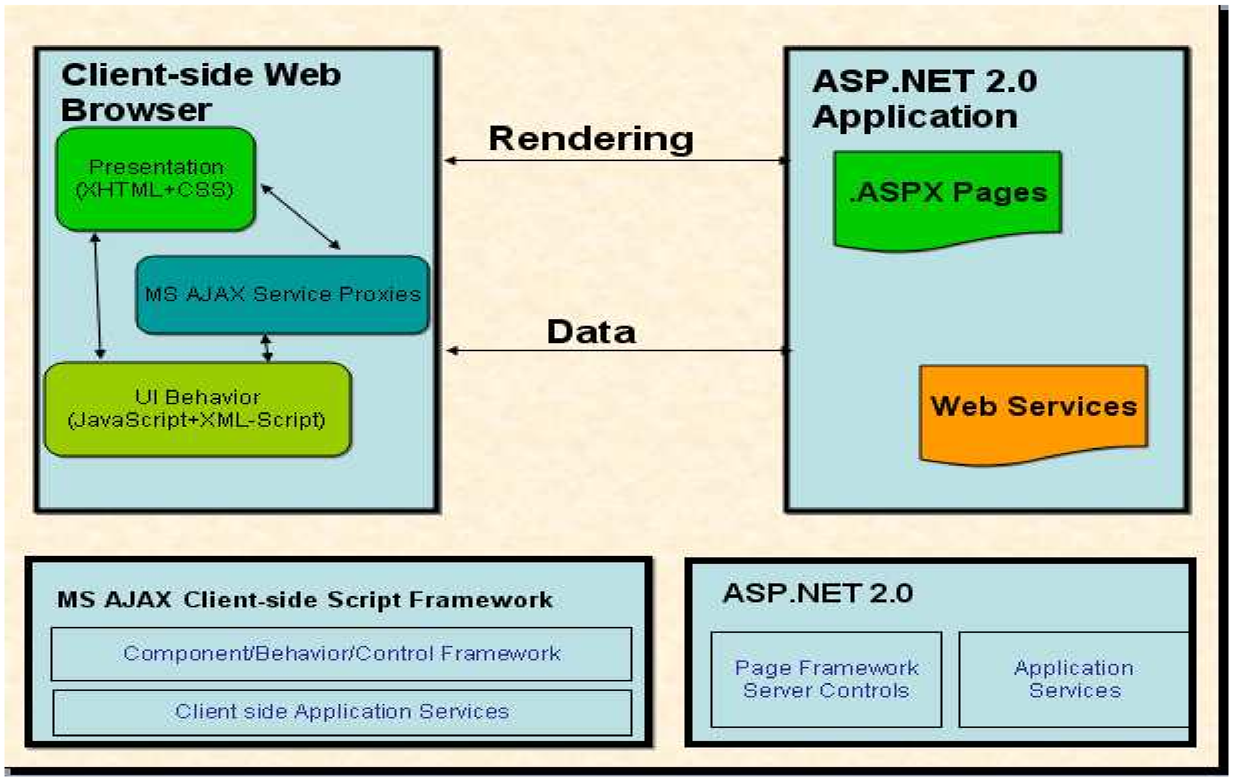

Client-side Web Browser
Presentation (XHTML+CSS)
MS AJAX Service Proxies
UI Behavior (JavaScript+XML-Script)
Rendering
Data
ASP.NET 2.0 Application
.ASPX Pages
Web Services
MS AJAX Client-side Script Framework
Component/Behavior/Control Framework
Client side Application Services
ASP.NET 2.0
Page Framework Server Controls
Application Services

Figure 4.2 ASP.NET AJAX Frameworks

4.2 Architecture of Real Estate Web Application

The Real Estate Web Application is based on the three tier architecture namely the presentation tier, the middle tier, and the data tier.

4.2.1 Presentation Tier

The Presentation Tier of this application consists of ASP.NET web forms which contain ASP web controls, user controls and AJAX controls. The web pages are designed to make the website look attractive and interactive. The objective of this layer was to take the full advantage of the latest AJAX animated controls like Collapsible panels, sliding bars, watermark textboxes, etc. Visual Studio .NET 2005 version is being used to design and code the ASP.NET pages.

ASP.NET 2.0 has an extra support of master pages and themes making the site pages look uniform and have common structure throughout. The banners and footers are included in the master page; the Tab Container loads the rest of the web pages. The content is added by the user in the search engine using the AJAX controls which causes the data grid present in the

Update Panel being refreshed. The application constitutes some of the basic pages needed by a Real Estate Application. This layer is responsible for the results/output from the business layer and transforms the results into a readable format for the end user. The following Page Flow Diagram presents the navigation flow in the application:

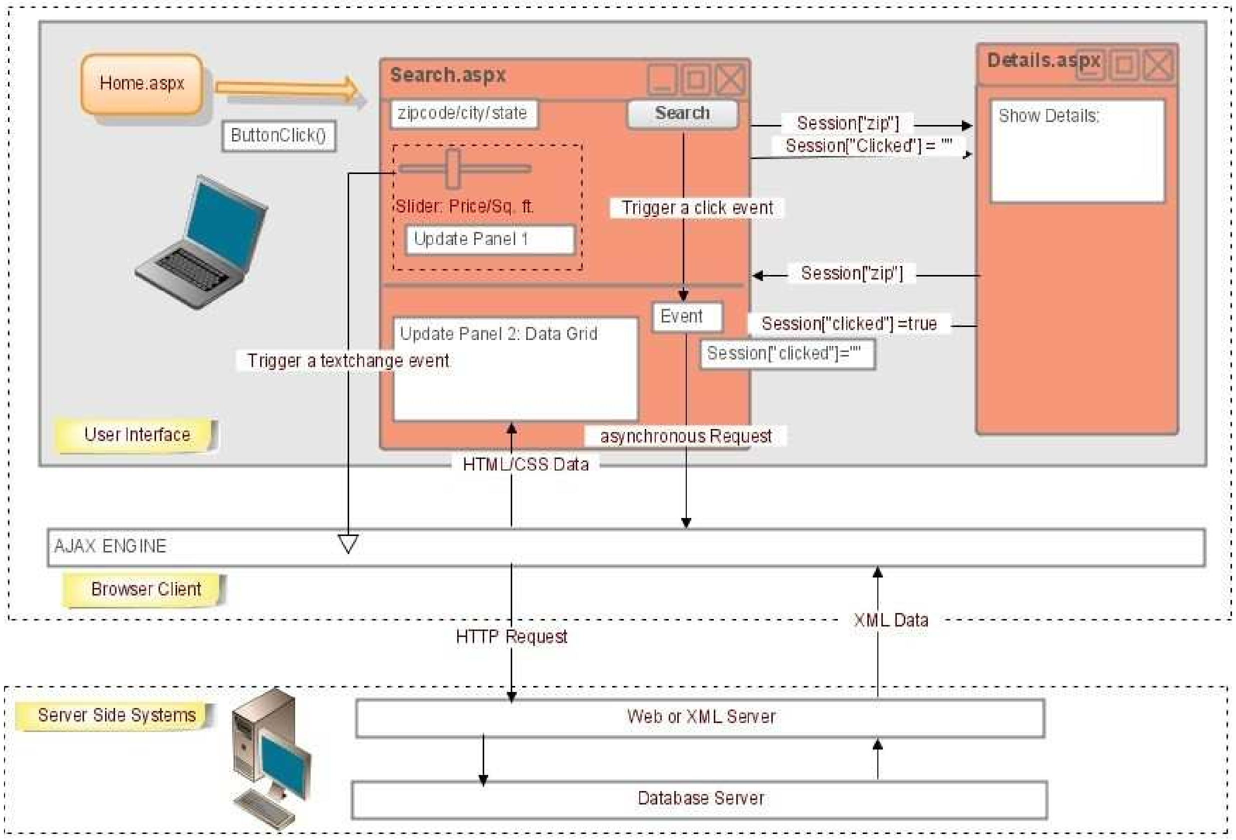


Figure 4.3 Page Flow Diagrams

Table 4.1 represents the list of ASP.NET Pages implemented in the Real Estate Web Application and the purpose of each page.

ASP.NET Pages	Purpose
HomePage.aspx	Information about the Real Estate Company.
Search.aspx	Provides a search engine to the buyer, results are displayed on the data grid.
UploadListing.aspx	Uploads the photographs of the property.
Details.aspx	Displays the property details.

Table 4.1 List of pages being planned for the Page Flow Diagram

4.2.2 Middle Tier

The Middle Tier called as the Business Logic Layer allows the users to share and control the business logic by isolating it from the other layers of the application. This tier increases the code transparency and supports changes in the data layer and is also responsible for altering the database. This tier contains classes for Property Listings, Property Photo, Property Features, Property Type and Zip codes and accesses the databases. A better understanding of the middle tier can be obtained with the help of the Use Case Diagram and Class Diagram.

4.2.2.1 Use Case Diagram

Figure 4.4 shows the Use Case Diagram for the Real Estate Web Application. The buyer can search the property listings, view property details, enter the selected listings in the buyer cart and can delete the listings from the buyer cart.

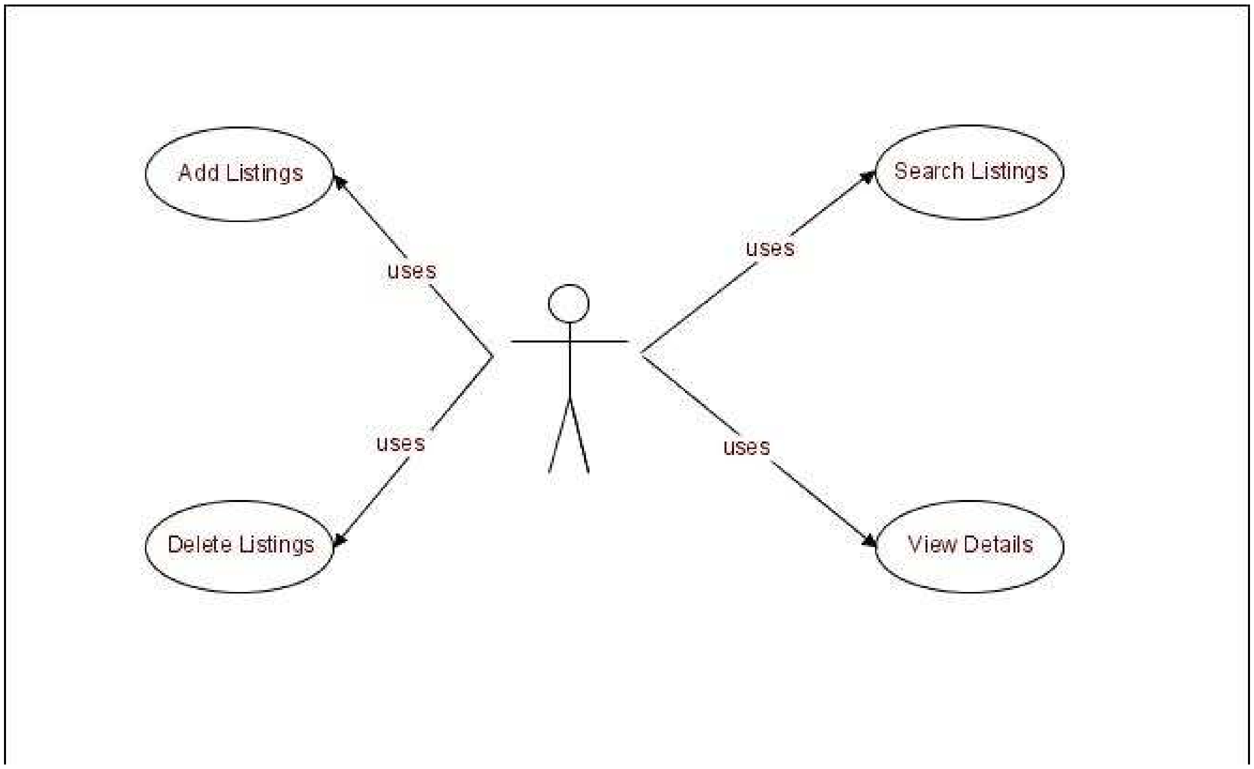


Figure 4.4 Use Case Diagram

4.2.2.2 Class Diagram

Figure 4.5 shows the Class Diagram representing the relations between the different classes.

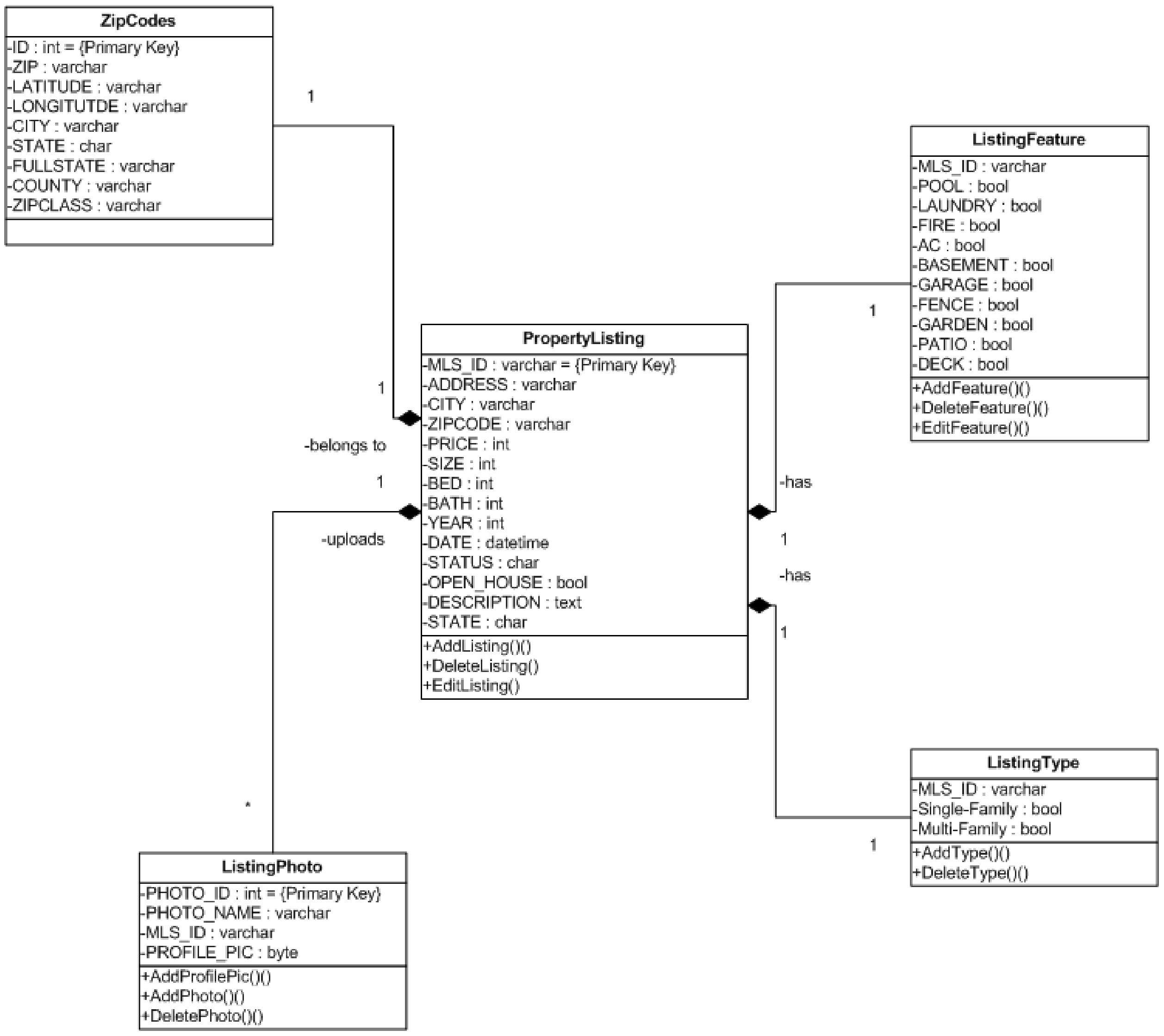


Figure 4.5 Class Diagram

The description for each of the classes in the Class Diagram is given below:

PropertyListing Class:

The PropertyListing class is the basic class for this application having attributes and description for each property listing which help the buyer to select the property desired for his need. This class includes the basic attributes to describe the listing like MLS#, listing_name, listing_address, listing_description, etc. The methods included in this class are:

Adding a new property listing to the database.

Editing the existing property listing.

Uploading photographs for each property listing.

Deleting the photographs for each property listing.

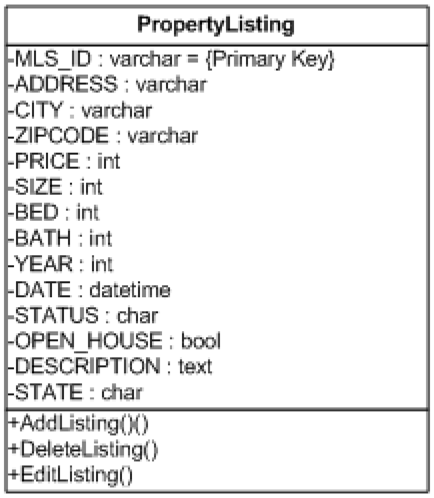


Figure 4.6 PropertyListing Class

ListingPhoto Class:

The ListingPhoto class associates itself with the PropertyListing. It is used to define the attributes of the photo that belongs to each Property Listing. The Real Estate Company can upload any number of photos but can have only one picture that acts as its profile picture. This profile picture of the listing is viewed whenever that particular listing pops up on the data grid by entering a particular search scenario. The rest of the pictures corresponding to the listing can be viewed in the photo album. The methods corresponding to this class are AddPhoto and DeletePhoto. The Class Diagram for ListingPhoto is shown below:

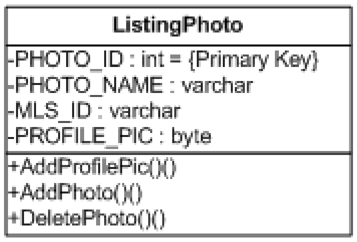


Figure 4.7 ListingPhoto Class

ListingFeature Class:

The ListingFeature Class corresponds to the super class PropertyListing. This class contains the attributes which describe the extra features of the properties present in the PropertyListing Class. The methods AddFeatures and DeleteFeatures add information corresponding to the property present in the PropertyListing Class. Thus, all the attributes combined from the PropertyListing and PropertyPhoto Classes completely describe the property which a buyer is looking for. The Class Diagram for ListingFeature is shown below:

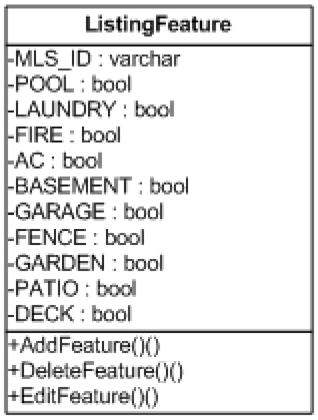


Figure 4.8 ListingFeature Class

PropertyType Class

PropertyType also corresponds to the super class PropertyListing. This class contains the attributes which describe the type of the property the buyer is looking for. The type of the property can be single-family or multi-family. The operations on the class named AddType and DeleteType determine the type of the property listing. Class Diagram for PropertyType is shown below:

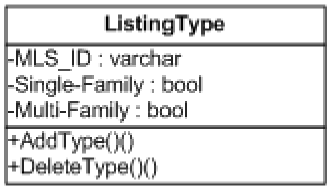


Figure 4.9 ListingType Class

ZipCodes Class

The ZipCode class supports the PropertyListing class. One of the attribute of the PropertyListing class is zip code which identifies the place where the property is located. ZipCode class contains all the US zip codes and supports the PropertyListing class by locating all the zip codes in the vicinity of a particular zip code. Class Diagram for ZipCode is shown below:

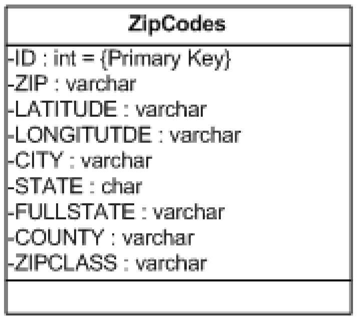


Figure 4.10 ZipCodes Class

4.2.3 Database Tier

The Real Estate Web Application is supported by the SQL Server 2005 and its database. SQL Server provides a good response time of the data being stored making the search effective, convenient way for storing the photographs of the properties and storing the entire description and features of the Property Listings.

The database schema for this application consists of five tables out of which the Listing is the main table to store the primary details of the property. Listing_type and Listing_feature reference the MLS# in the Listing table to describe more features of the listings. Lisitng_photo stores the photographs of the properties in the form of binary data. The MLS_ID column name is the attribute in the Listing table which is referenced by all the other tables.

4.2.3.1 ER Diagram

The Entity-Relationship model of the database being displayed is shown below:

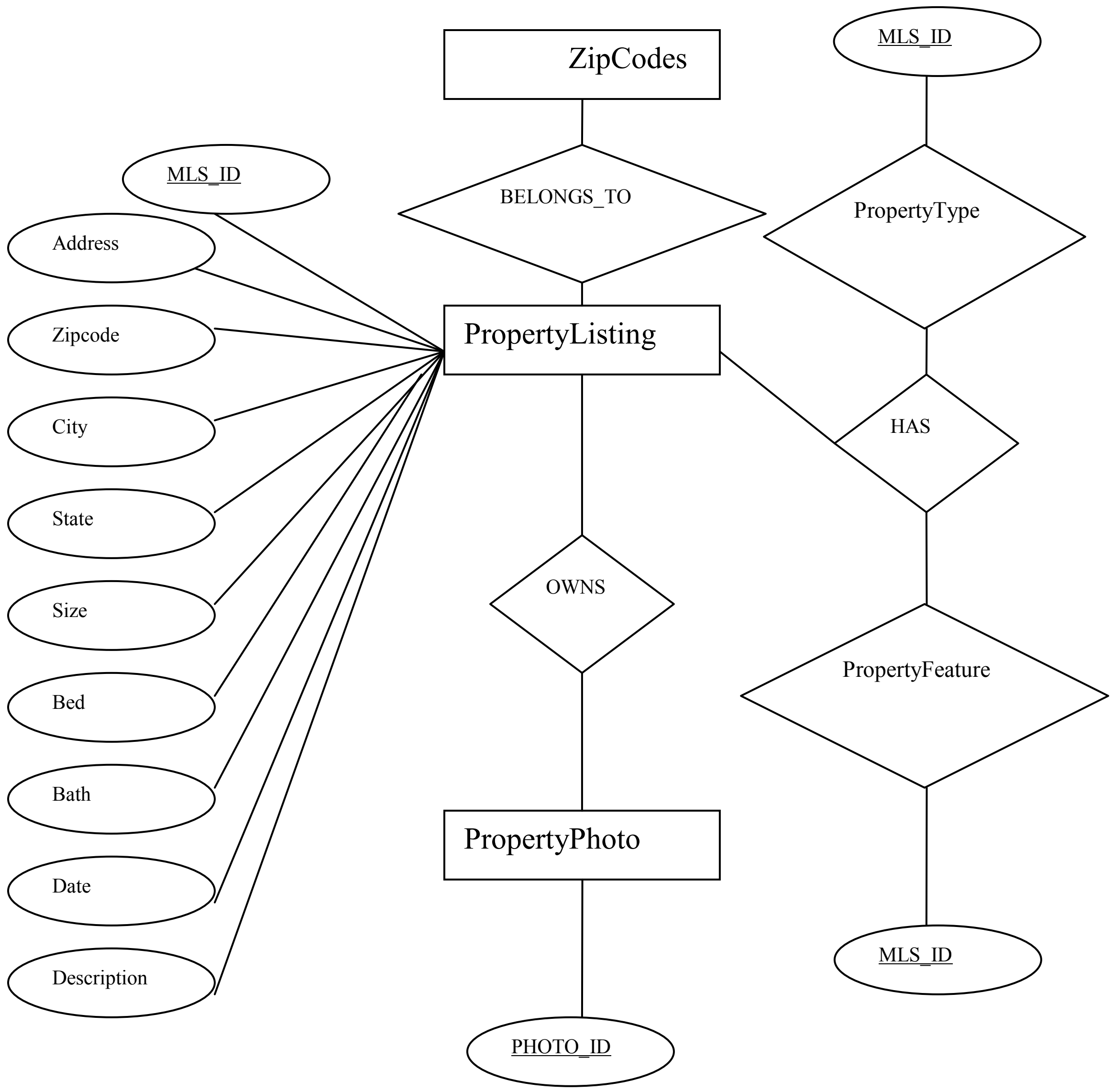


Figure 4.11 E-R Diagram

4.2.3.2 Database Schema

Following is the database schema diagram for the Real Estate Web Application, the associations between various database classes can be easily seen. The associations are also consistent with the class diagram's associations presented above. Member and Owns relationships are changed to strong entities in implemented.

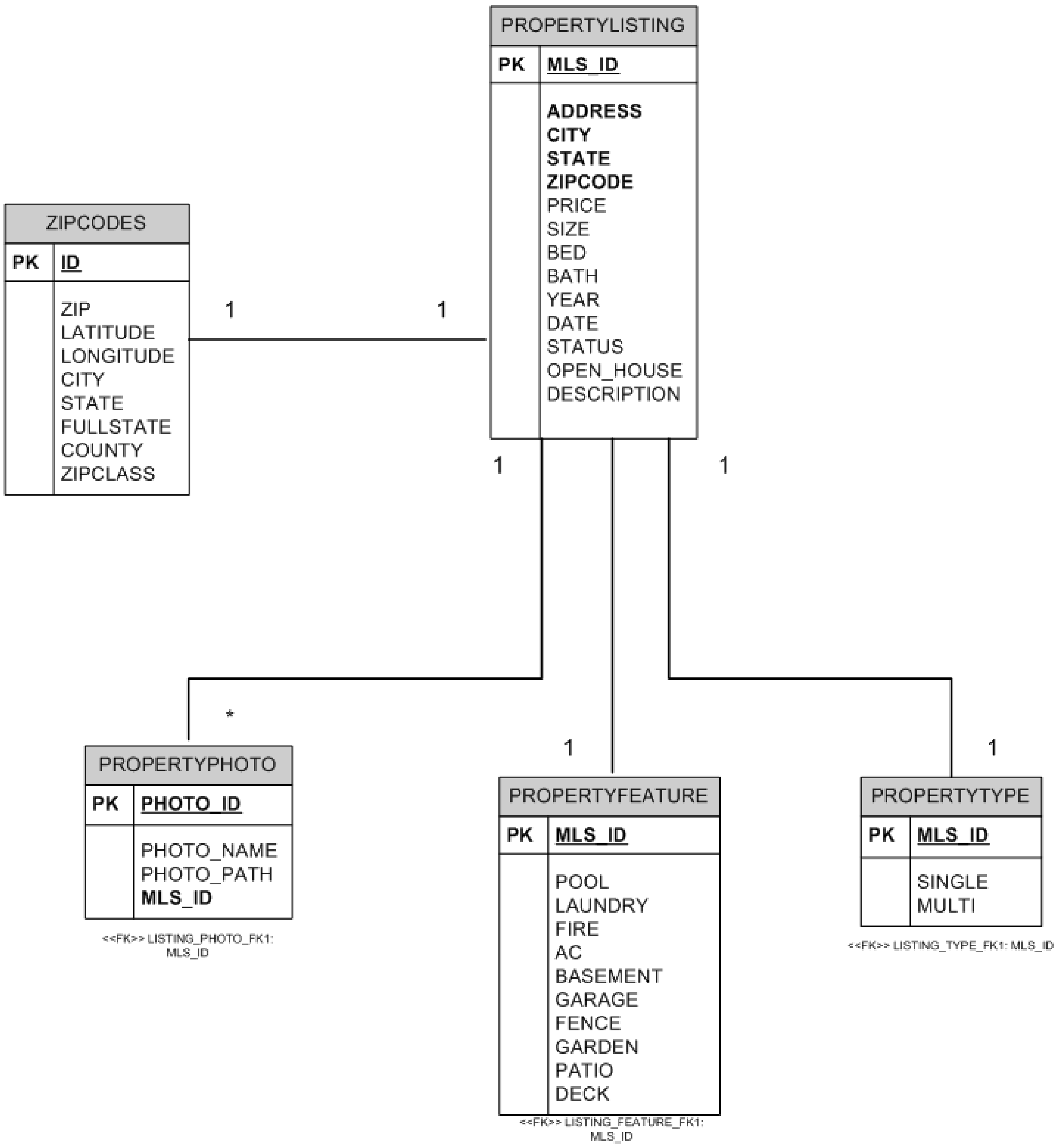


Figure 4.12 Database Schema

CHAPTER 5 - Testing

This chapter discusses the various tests performed on the web-application with a set of test scenarios. Both unit and load testing is performed on this application. Testing is done on a local machine with the following system configuration.

Processor	Intel Pentium®
Processor Speed	2.20GHz
Physical Memory	3.00 GB of RAM
Operating System	Windows XP Professional

Table 5.1: System Configuration

5.1 Unit Testing

Unit test is performed to test and validate the individual units of source code. It is the code wrapper around the application code that permits test tools to execute them for fail-pass conditions. Unit testing is important as it gives the code authors and reviewers' confidence that patches produce the correct results. The test cases are a good impetus for developers to discover edge cases.

The Real Estate Web Application uses NUnit Framework to perform Unit Testing. NUnit Framework is port of JUnit framework from java and Extreme Programming (XP) and is an open source product. This framework is developed to make use of .NET framework functionalities and uses an Attribute based programming model. The nunit.exe program is a graphical runner that shows the test cases in an explorer-like browser window and provides visual indication of success or failure of the tests.

Following are the test cases executed against the web-application using NUnit:

Bed_BathTest: This test case validates the functional unit in the application that is responsible for querying the database to retrieve the number of beds and baths from the database corresponding to the number entered by the user in the search criterion. This functional test checks for the valid beds and baths combination that can let the user retrieve the results from the database.

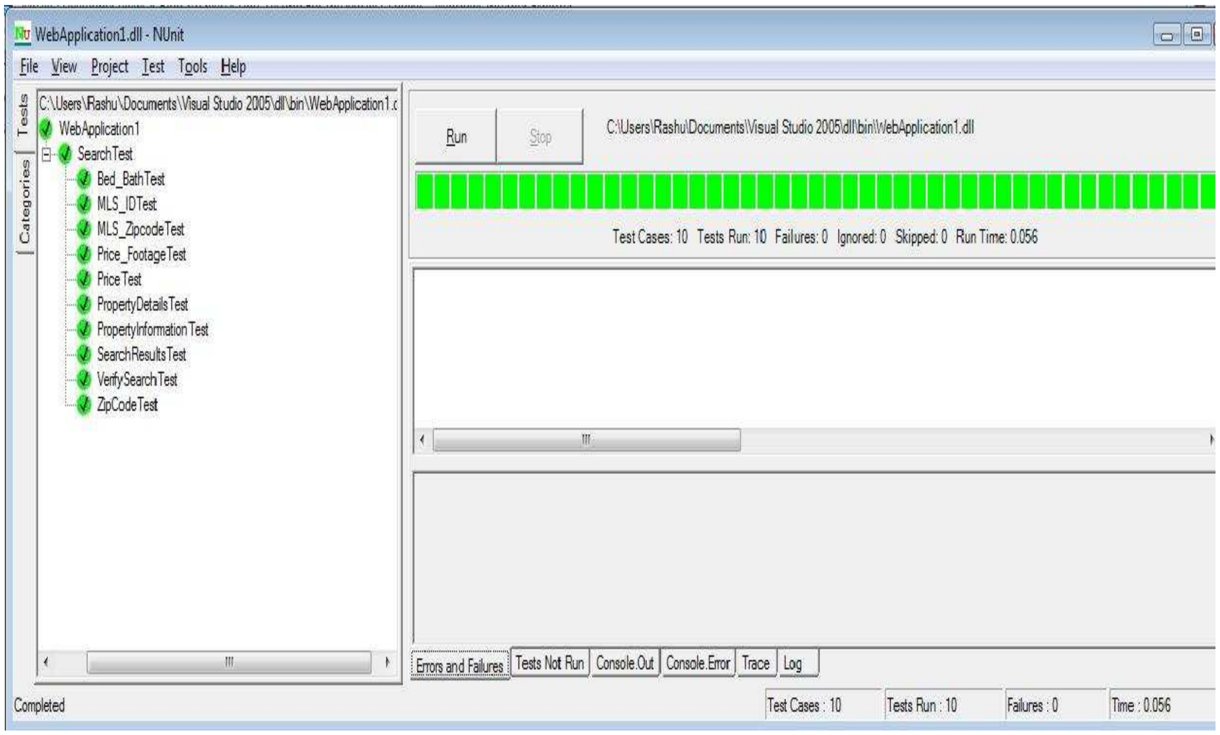
MLS_IDTest: This test case tests the functional unit that is responsible for fetching the details of the property listing such as city, state, zipcode, beds, baths, etc and displays it to the user in the data grid. This function ensures whether the search entered in the search criterion.

MLS_ZipCodeTest: This test case validates the functional unit in the application that is responsible for querying the database to retrieve the property listing details from the database based on the combinatorial search for the MLS_ID and ZipCode. This functional test checks for the valid MLS_ID and ZipCode combination that can let the user retrieve the results from the database. The result of this combinatorial search results in one row corresponding to the unique MLS_ID entry in the database.

PriceFootageTest: This test case validates the functional unit in the application that is responsible for querying the database to retrieve the property listing details where the price is greater than the price entered by the user and the footage is greater than the square feet miles entered by the user. This query may result in many rows satisfying the search criterion.

PriceTest: This test case tests the functional unit that is responsible for fetching the details of the property listing such as city, state, zipcode, beds, baths, etc and displays it to the user in the data grid where price is greater than the price entered by the user. This function ensures whether the search entered in the search criterion is correct.

PropertyDetailsTest, PropertyInformationTest, SearchResultsTest, VerifySearchTest and ZipCodeTest are other functional unit tests that validate the correctness of the results corresponding to different combinatorial search criteria. The screenshot of NUnit test cases execution is shown in the Figure 5.1.



```
C:\Users\Rashu\Documents\Visual Studio 2005\dll\bin\WebApplication1.c
WebApplication1
  SearchTest
    Bed_BathTest
    MLS_IDTest
    MLS_ZipcodeTest
    Price_FootageTest
    PriceTest
    PropertyDetailsTest
    PropertyInformationTest
    SearchResultsTest
    VerifySearchTest
    ZipCodeTest
```

```
C:\Users\Rashu\Documents\Visual Studio 2005\dll\bin\WebApplication1.dll
Test Cases: 10  Tests Run: 10  Failures: 0  Ignored: 0  Skipped: 0  Run Time: 0.056
```

```
Completed          Test Cases : 10    Tests Run : 10    Failures : 0    Time : 0.056
```

Figure 5.1 Screenshot of NUnit test case execution.

5.2 Performance Testing

Performance of a web-application can be tested done by applying the Load and Stress Testing on it. Load testing is subjecting the application to a statistically representative load. This testing is done in support of the application reliability testing and in performance testing. In performance testing, the load is varied from minimum (zero) to the maximum level the application can handle without application-specific excessive delay. In this testing the realistic workloads are characterized, simulated and submitted to the system under test. Load Testing verifies the acceptability of the target-of-test's performance behavior under varying operational conditions (such as number of users/threads, number of loop counts, etc.) while the configuration remains constant. While the Stress Testing verifies the acceptability of the target-of-test's performance behavior when abnormal or extreme conditions are encountered, such as diminished resources or extremely high number of users. Extreme loads are used in **load stress testing** - to find the breaking point and bottlenecks of tested system. In load stress testing we have tried breaking the application by an extreme load and expose bugs that are likely to appear under stress, such as data corruption, buffer overflows, large number of users, etc.

Load analysis needs to proceed by having a special purpose browser act like a human user. This assures that the performance checking experiment indicates true performance - not performance on simulated but unrealistic conditions. The testing tools try to perform Load Simulation to re-create a realistic scenario and so the results obtained assures that they can be used to analyze the performance characteristics of the web site under high load conditions.

The Real Estate Web Application uses the Jakarta JMeter as the Website load testing tool as it can simulate thousands of users, came from the same quantity of IPs and analyze, how many concurrent users can handle this web site. Apache JMeter is a 100% pure Java Desktop application designed to load test functional behavior and measure performance of a web application. This tool is strong enough to test perform both on static and dynamic resources such as files, Servlets, Java Objects, Data Bases and Queries, FTP Servers and more. JMeter is used to simulate a heavy load on a server, network or object to test its strength or to analyze overall performance under different load types. It can be used to obtain a graphical analysis of performance behavior under heavy concurrent load [6].

Using the JMeter, I have performed several tests taking different readings and found out the peak number of users that can run effectively with this tool. The hardware capabilities of my machine limit the number of users to 1000. The testing results are shown in a tabular form containing the testing figures for various web pages. The testing is done keeping the loop count constant i.e. 16000 and Ramp-up period as 5 seconds and varying the number of users.

Users	Ramp-Up Period (sec)	Loop Count
100	5	16000
500	5	16000
1000	5	16000

Table 5.2: Test Cases Summary

Testing is performed on the following pages:

- HTML Page (Home Page)
- Database Intensive Page (Search Page)
- Business Logic Page (Details Page)

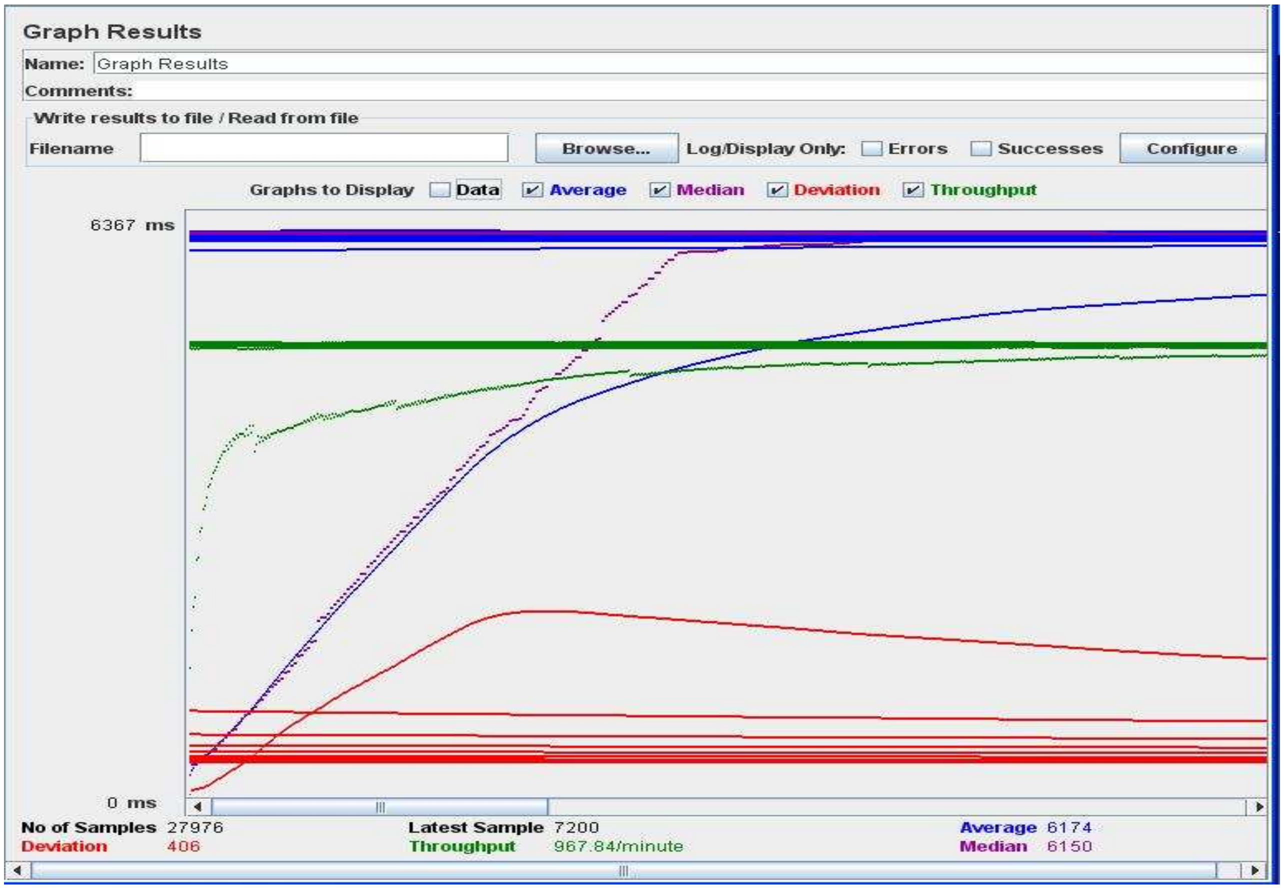
5.2 Testing Samples

5.2.1 Search Web Page

5.2.1.1 Test Case 1

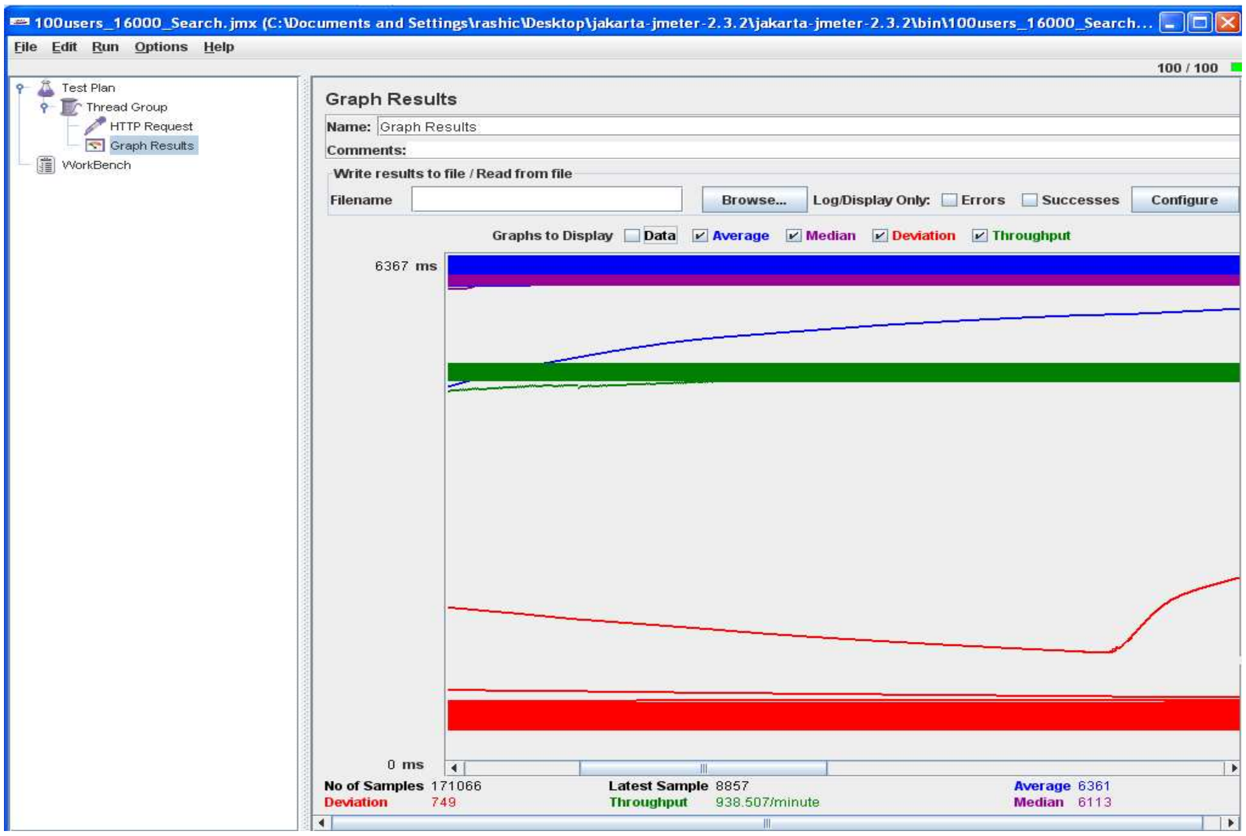
Users	Loop Count	Ramp-up period (sec)	Average Response Time (ms)	Throughput
100	16000	5	6174	967.84/min

Table 5.3: Search Page Test Case 1


Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Graphs to Display Data ✓ Average ✓ Median ✓ Deviation ✓ Throughput
6367 ms
0 ms

No of Samples 27976
Deviation 406
Latest Sample 7200
Throughput 967.84/minute
Average 6174
Median 6150

Figure 5.2 Search Page Test Case 1.1 - Graph Results


100users_16000_Search.jmx (C:\Documents and Settings\rashic\Desktop\jakarta-jmeter-2.3.2\jakarta-jmeter-2.3.2\bin\100users_16000_Search...
File Edit Run Options Help
100 / 100
Test Plan
Thread Group
HTTP Request
Graph Results
WorkBench
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Graphs to Display Data Average Median Deviation Throughput
6367 ms
0 ms

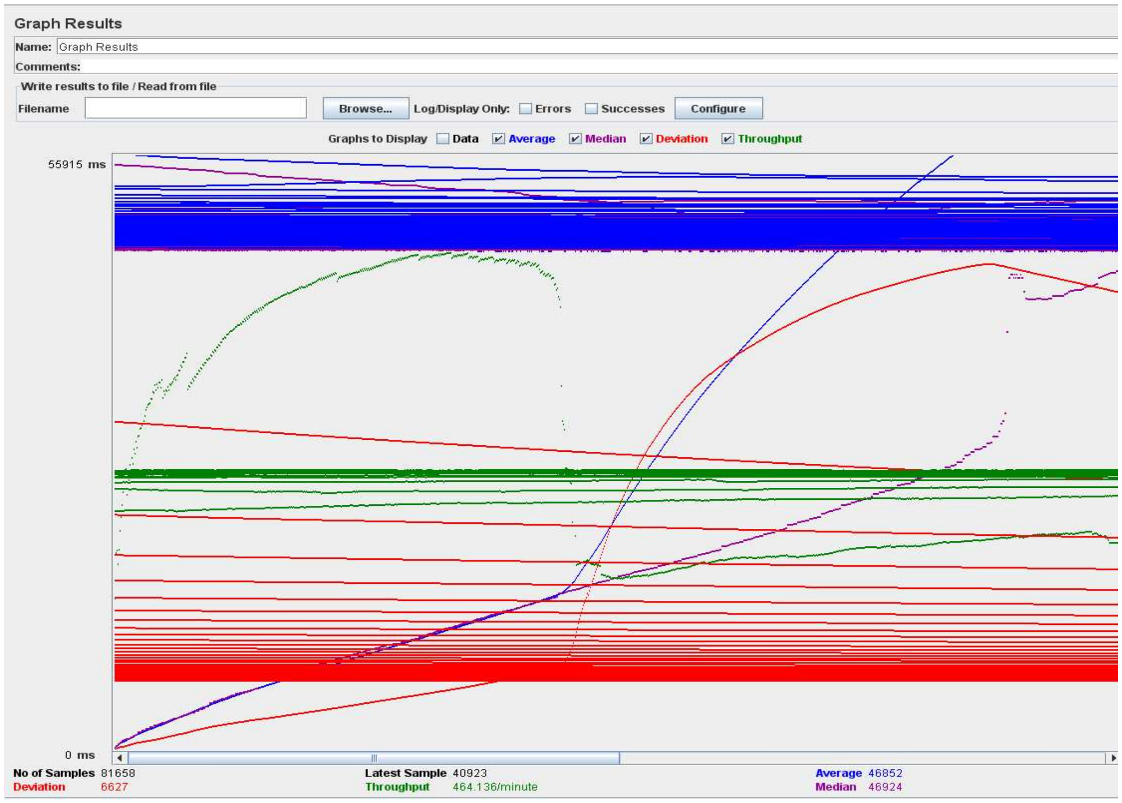
No of Samples 171066 Latest Sample 8857 Average 6361
Deviation 749 Throughput 938.507/minute Median 6113

Figure 5.3 Search Page Test Case 1.2 - Graph Results

5.2.1.2 Test Case 2

Users	Loop Count	Ramp-up period (sec)	Average Response Time (ms)	Throughput
500	16000	5	55915	967.84/min

Table 5.4: Test Case 2



Name: Graph Results

Graphs to Display: Data ☐ Average ✓ Median ✓ Deviation ✓ Throughput ✓

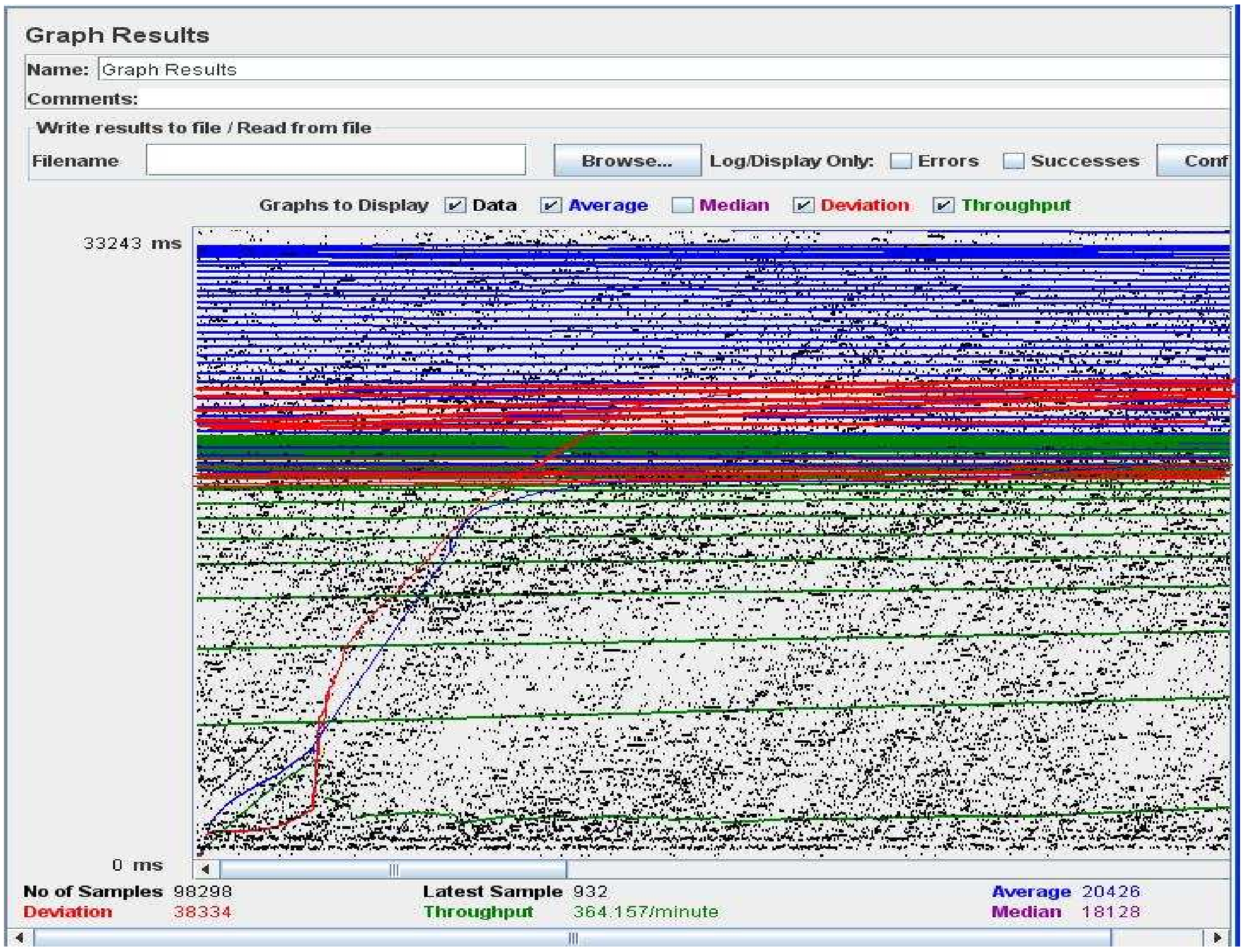
No of Samples 81658
Deviation 6627
Latest Sample 40923
Throughput 464.136/minute
Average 46852
Median 46924

Figure 5.4 Search Page Test Case 2 - Graph Results

5.2.1.3 Test Case 3

Users	Loop Count	Ramp-up period (sec)	Average Response Time (ms)	Throughput
1000	16000	5	33243	364.157/min

Table 5.5: Search Page Test Case 3



No of Samples 98298
Deviation 38334
Latest Sample 932
Throughput 364.157/minute
Average 20426
Median 18128

Figure 5.5 Search Page Test Case 3.1 - Graph Results

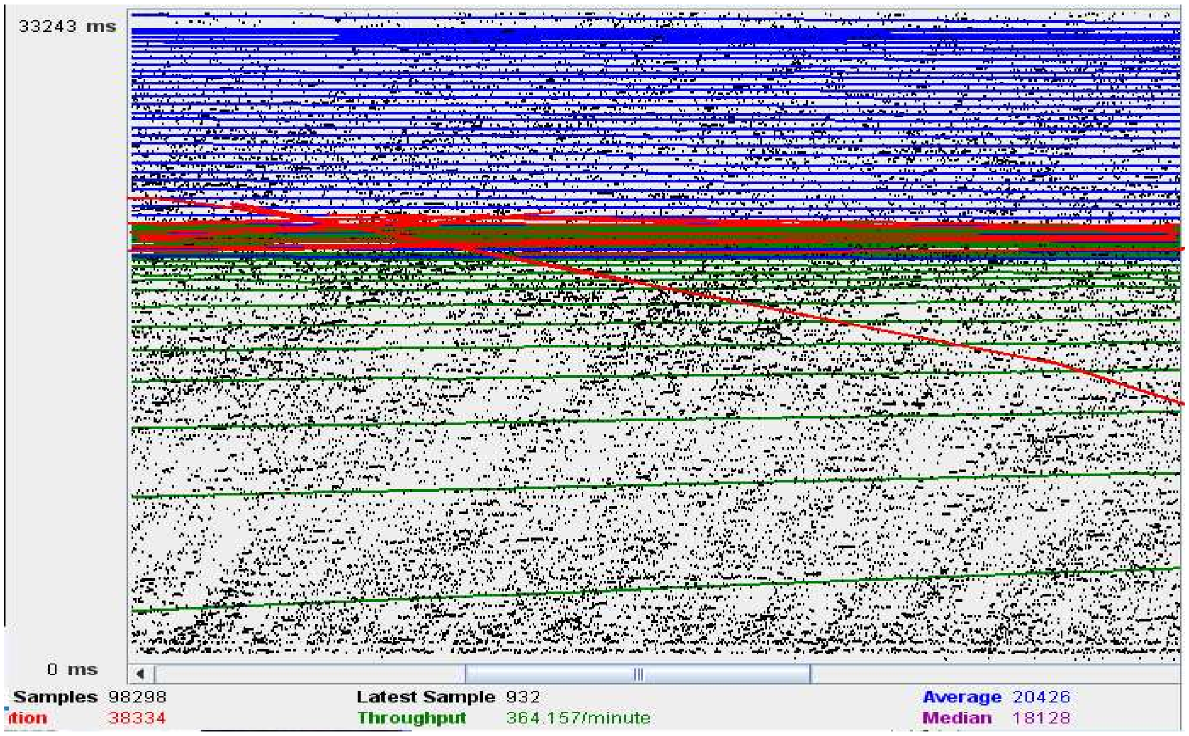


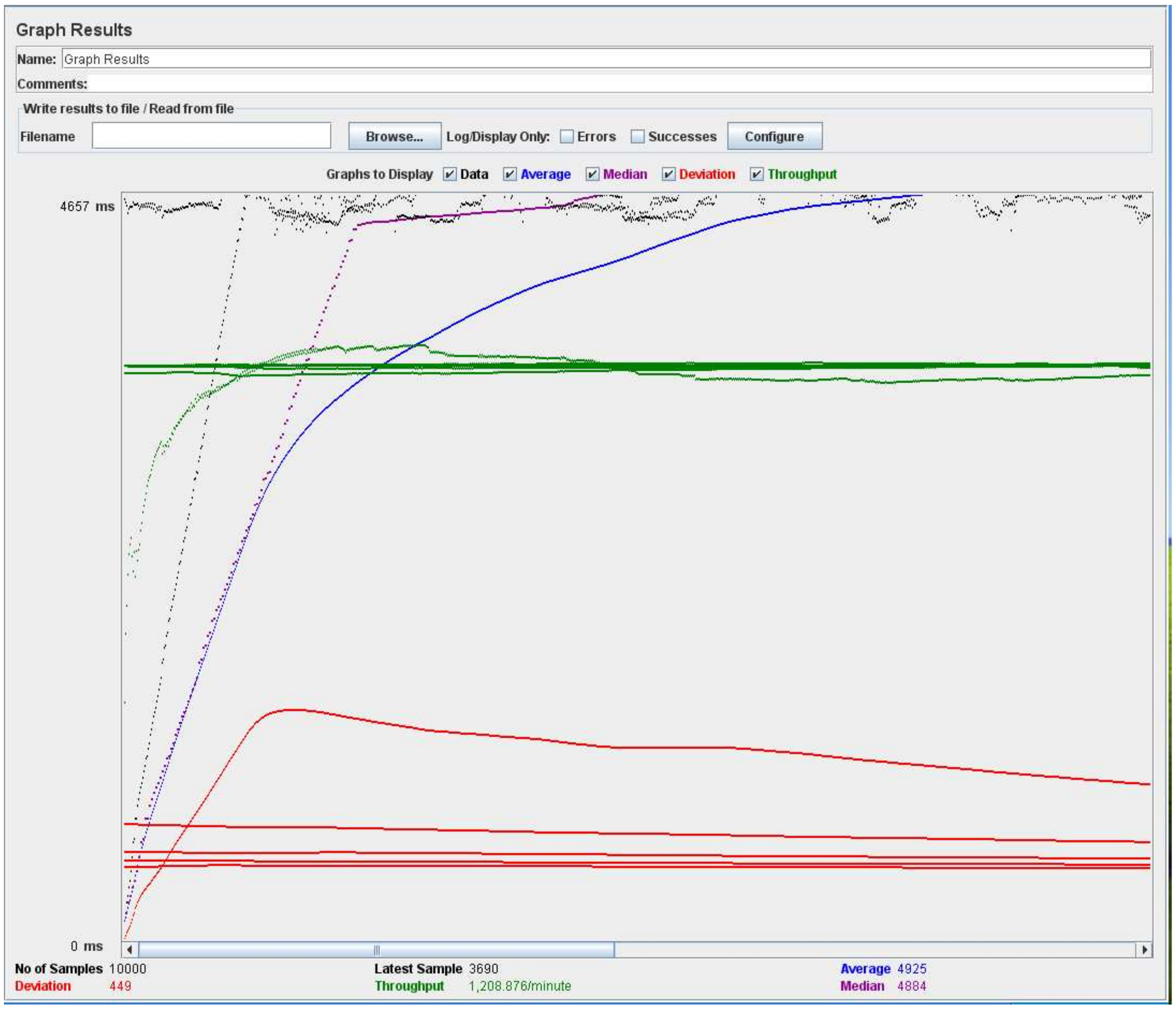
Figure 5.6 Search Page Test Case 3.2 - Graph Results

5.2.2 Details Web Page

5.2.2.1 Test Case 1

Users	LoopCount	Ramp-up period (sec)	Average Response Time (ms)
100	16000	5	4657

Table 5.6: Details Page Test Case 1



Graph Results

Name: Graph Results

Comments:

Write results to file / Read from file

Filename

Log/Display Only: ☐ Errors ☐ Successes

Graphs to Display ✓ Data ✓ Average ✓ Median ✓ Deviation ✓ Throughput

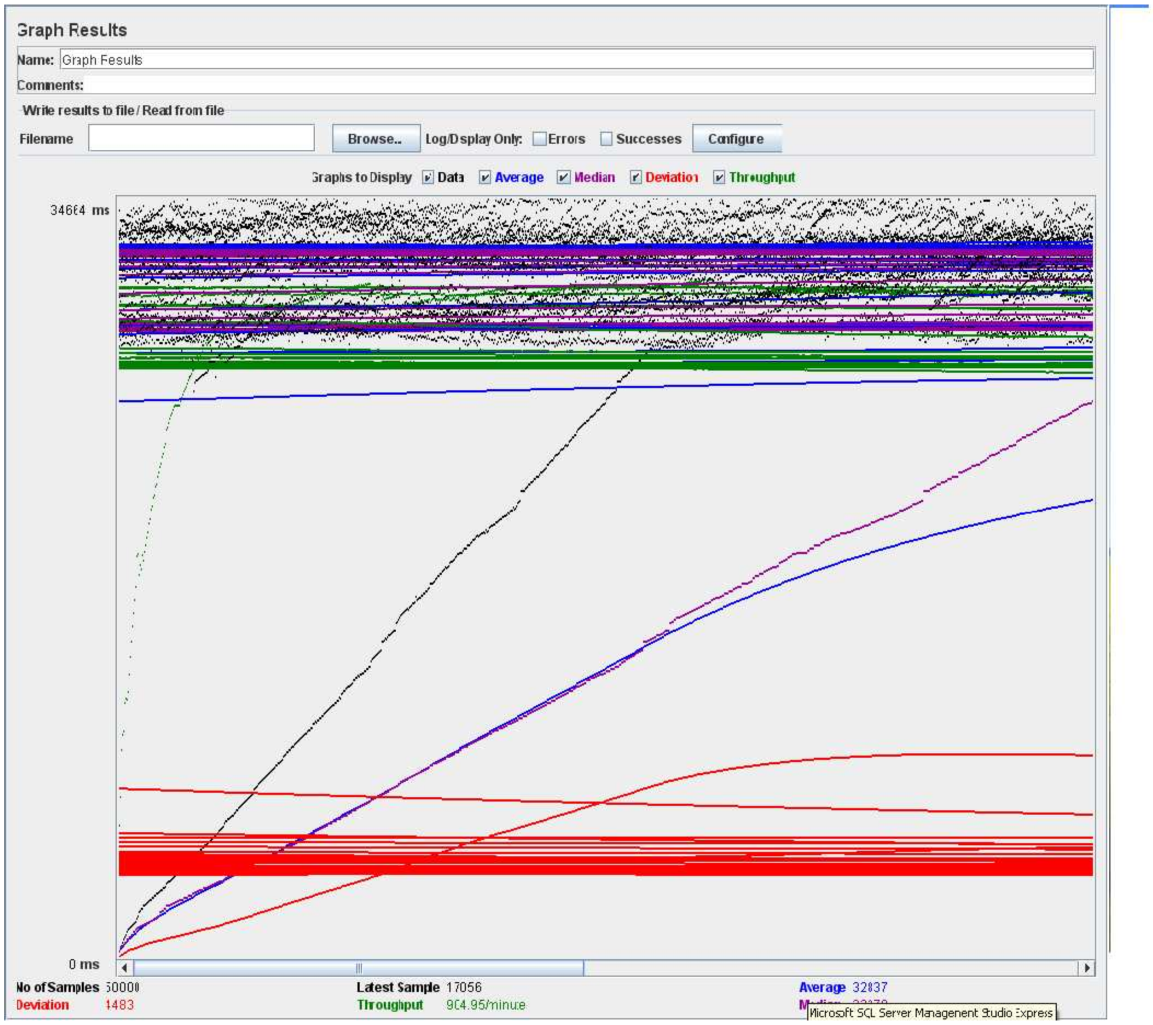
No of Samples 10000
Deviation 449
Latest Sample 3690
Throughput 1,208.876/minute
Average 4925
Median 4884

Figure 5.7 Details Page Test Case 1 - Graph Results

5.2.2.2 Test Case 2

Users	LoopCount	Ramp-up period (sec)	Average Response Time (ms)
500	16000	5	34664

Table 5.7: Details Page Test Case 2



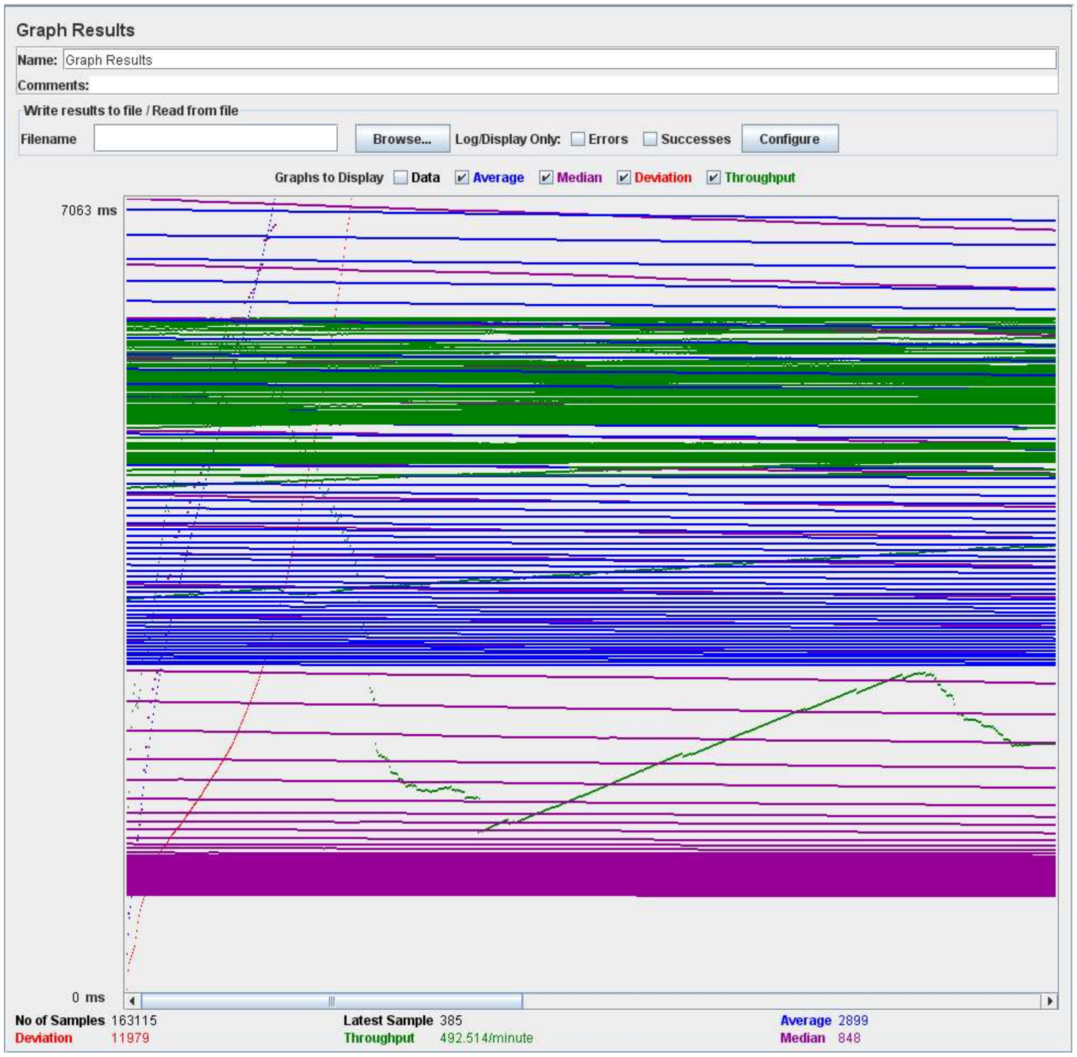
No of Samples 50000
Deviation 1483
Latest Sample 17056
Throughput 904.95/minute
Average 32837

Figure 5.8 Details Page Test Case 2 - Graph Results

5.2.2.3 Test Case 3

Users	LoopCount	Ramp-up period (sec)	Average Response Time (ms)
1000	16000	5	70063

Table 5.8: Details Page Test Case 3



No of Samples 163115
Deviation 11979
Latest Sample 385
Throughput 492.514/minute
Average 2899
Median 848

Figure 5.9 Details Page Test Case 3 - Graph Results

5.4 Test Analysis

The main purpose of performing JMeter Testing is analyzing the response time requirements of the web applications. In context of web applications, response time refers to the time elapsed between the submission of a request and the receipt of the resulting HTML. While calculating the response time from JMeter, it does not takes into account the rendering time required by the browser to load the HTML. Thus, Response Time is just the exact time between sending the request and receiving it. The average response time is determined by the Blue Line in the above graphs. The Red Line represents the Deviation which is the deviation from the average response times which can take place if some percent of the samples have less response times while others have high. The main statistics for the analysis of the load on the web application comes from the Average Response Time and the Deviation readings.

Analyzing the search page results we find that, when the number of users are too less i.e. 10 the Response time comes up to approx .5 seconds which is reasonable enough. When the number of users are increased to 100 the Response Time increases to approx 6 seconds indicating more load on the application. Initially when the application starts, the number of users are less for the first few seconds so the Average and Deviation increase parallel but as the number of users are increased the deviation starts decreasing to a constant line and at the same time the Average increases to a constant line. Thus, the Average Time comes up to 6 seconds and Deviation to .4 seconds. Gradually, when the load on the application is too high the Deviation shoots up and the readings come close to Average line. Increasing more number of users i.e. 500 the Average Time further increases to 5 seconds over 80,000 samples. The same scenario can be scene with this case with Average and Deviation moving parallel and then deviation reduces when number of samples increase and finally the deviation shoots upcoming parallel to Average. Finally, incrementing the number of users to 1000 takes the maximum Average Response Time and puts more load on the application. Analyzing the Throughput i.e. number of requests per minute we figure out that it is increases and then comes to a constant and a reasonable value.

The results can be explained for the Details page. The figures and Tables above indicate the Response Time and Throughputs for the same number of users i.e. 100, 500, 1000.

Looking at the results and the analysis it is evident that for this application and the above mentioned system configuration the application works fine with 100 users but as we increase the number of users the application starts hanging up indicating the increase of load at the server. The main reason for these performance results is the fault of the system level and the probable reason for it is its configuration. If we have to increase the performance of the application we need to work on the system configuration and upgrade it but before that it is important to figure out which part of the system would shoot up the application's performance. The main factors which might be cause the application to have low performance are: CPU, RAM and Hard Disk. Out of the three if the performance degradation takes place because of the low CPU processing then it is because of Thrashing and so the better CPU configuration is required to handle more users. In order to confirm this assumption we need to look for the evidence and figure out which one needs to be improved.

Since the application was not able to handle the load properly for 100 users, I performed the tests for 10, 40 and 50 users. To confirm if the RAM was degrading the performance, I performed the same tests simultaneously at two machines with the same configuration but different RAM's. Both these machines have the application deployed in it and could make out that the performance was still the same. I tested it for CPU by running this application on my laptop and then taking the results in JMeter, when the number of users were increased from 40 to 50 and so on the CPU started blinking indicating the increase of load put on it and also the Task Manager gave a good indication about the amount of time the CPU is taking for testing the application. Table 5.9 show the increase in Response Times with the increase in load i.e. number of users.

Users	Average Response Times (ms)
10	532
40	2543
50	3120

Table 5.9: Average Response Times Summary.

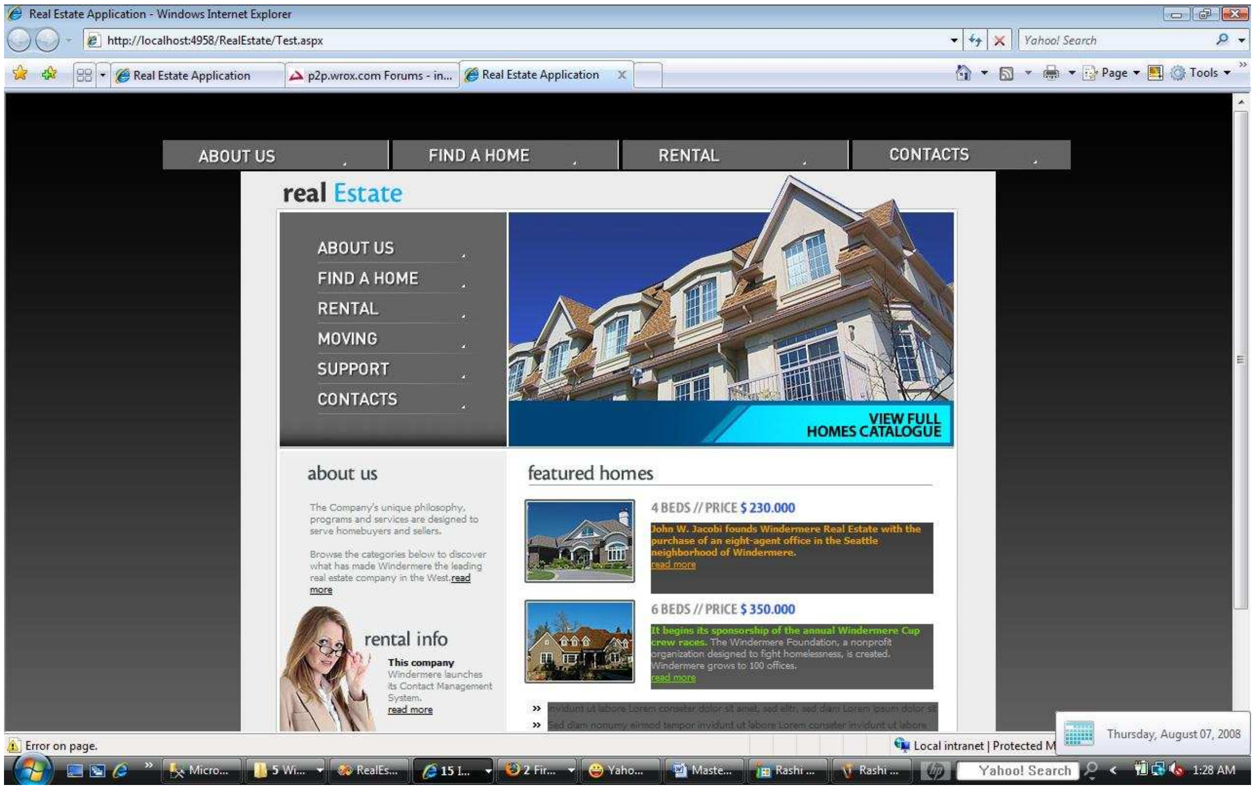
This confirms that the CPU needs to get upgraded with high configuration in order to get good performance results for more users. The CPU can be upgraded with a better processor speed i.e. more than 2 GHz in order to increase the performance load. Table 5.10 shows the results of the system configuration testing.

System Part	CPU	Hard Disk	RAM	Reason
Performance	Low	High	High	Low Processor Speed

Table 5.10: System configuration testing for Performance Upgrade.

5.3 Screen Shots of Tested Web Pages

5.3.1 About Us Webpage


Real Estate Application - Windows Internet Explorer
http://localhost:4958/RealEstate/Test.aspx
Yahoo! Search
Real Estate Application
p2p.wrox.com Forums - in...
Real Estate Application
Page
Tools
ABOUT US
FIND A HOME
RENTAL
CONTACTS
real Estate
ABOUT US
FIND A HOME
RENTAL
MOVING
SUPPORT
CONTACTS
VIEW FULL HOMES CATALOGUE
Error on page.
Local intranet | Protected M
Thursday, August 07, 2008
1:28 AM

about us

The Company's unique philosophy, programs and services are designed to serve homebuyers and sellers.

Browse the categories below to discover what has made Windermere the leading real estate company in the West.read more

rental info

This company Windermere launches its Contact Management System.
read more

featured homes

4 BEDS // PRICE $ 230.000

John W. Jacobi founds Windermere Real Estate with the purchase of an eight-agent office in the Seattle neighborhood of Windermere.
read more

6 BEDS // PRICE $ 350.000

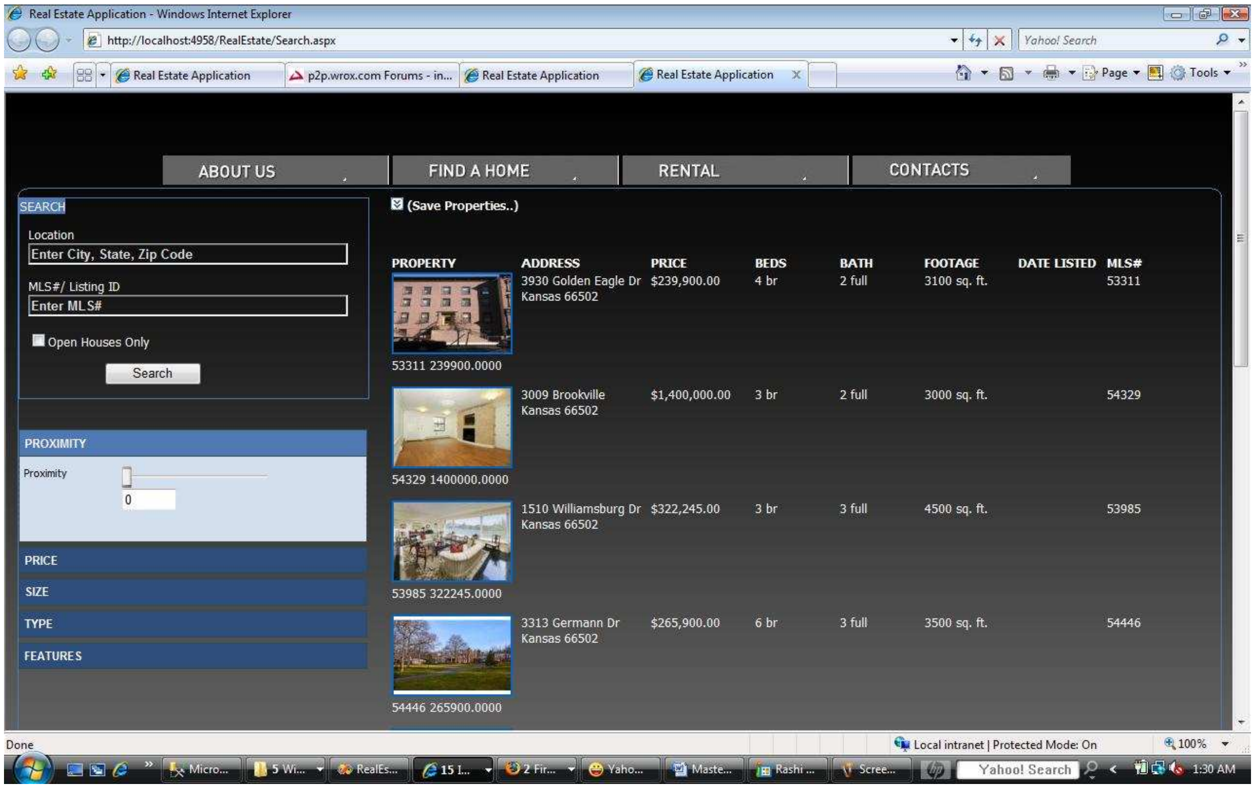
It begins its sponsorship of the annual Windermere Cup crew races. The Windermere Foundation, a nonprofit organization designed to fight homelessness, is created. Windermere grows to 100 offices.
read more

» invidunt ut labore Lorem conseter dolor sit amet, sed elitr, sed diam Lorem ipsum dolor sit

» Sed diam nonumy eirmod tempor invidunt ut labore Lorem conseter invidunt ut labore

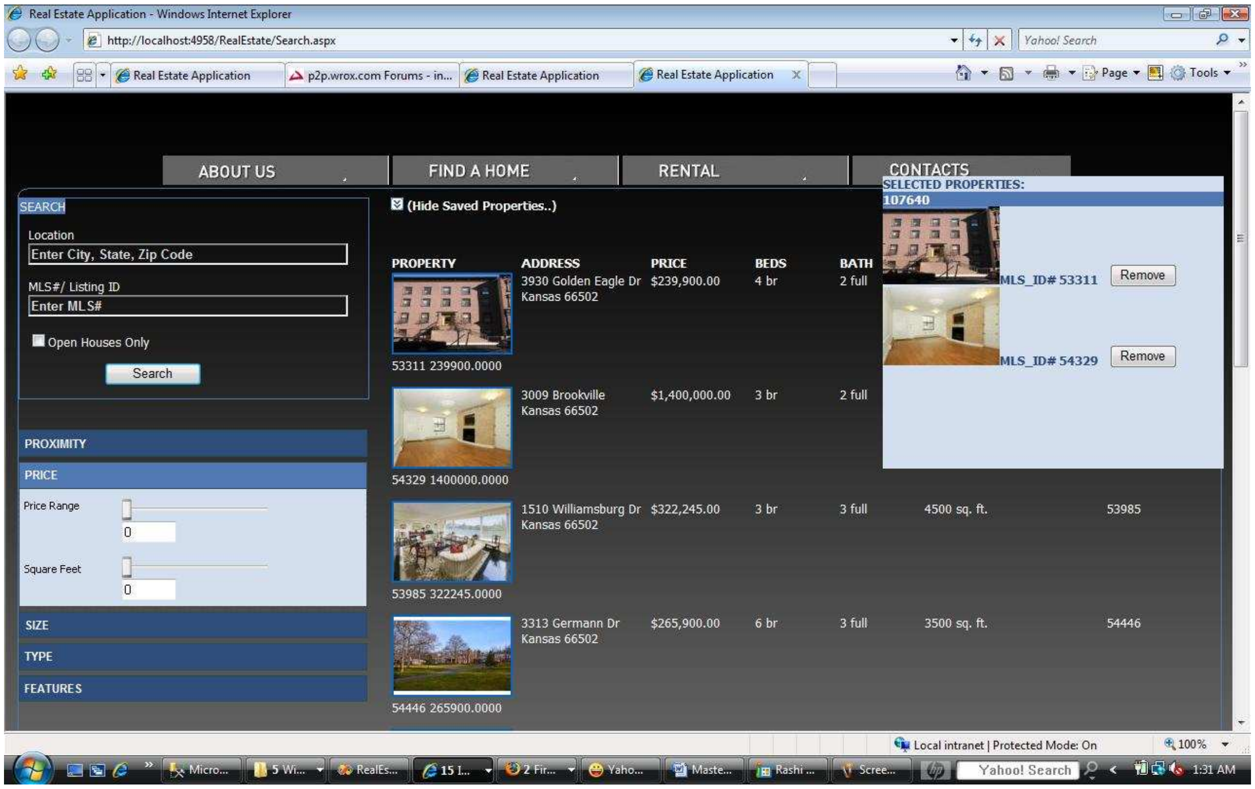
Figure 5.10 Screenshot of About Us Webpage

5.3.2 Search Webpage



PROPERTY	ADDRESS	PRICE	BEDS	BATH	FOOTAGE	DATE LISTED	MLS#
53311 239900.0000	3930 Golden Eagle Dr Kansas 66502	$239,900.00	4 br	2 full	3100 sq. ft.		53311
54329 1400000.0000	3009 Brookville Kansas 66502	$1,400,000.00	3 br	2 full	3000 sq. ft.		54329
53985 322245.0000	1510 Williamsburg Dr Kansas 66502	$322,245.00	3 br	3 full	4500 sq. ft.		53985
54446 265900.0000	3313 Germann Dr Kansas 66502	$265,900.00	6 br	3 full	3500 sq. ft.		54446

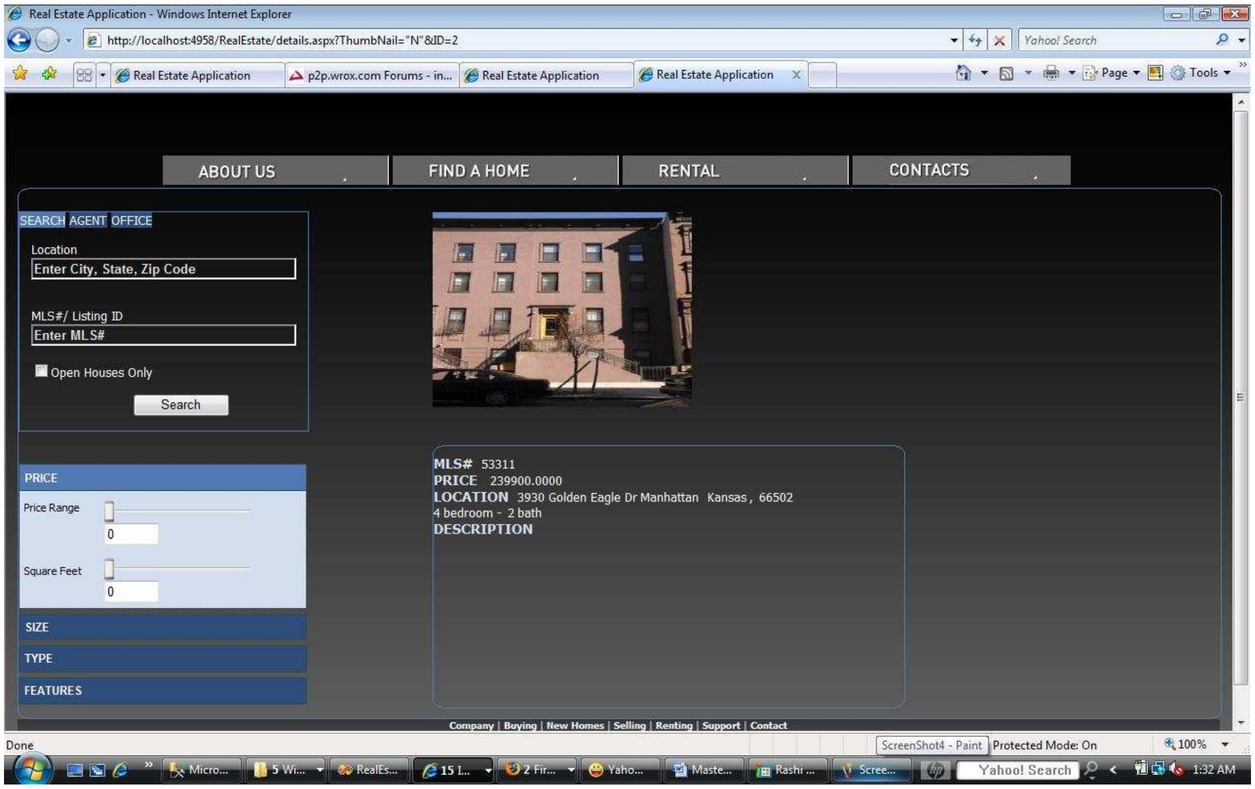
Figure 5.11 Screenshot of Search Webpage



PROPERTY	ADDRESS	PRICE	BEDS	BATH	FOOTAGE	DATE LISTED	MLS#
53311 239900.0000	3930 Golden Eagle Dr Kansas 66502	$239,900.00	4 br	2 full			
54329 1400000.0000	3009 Brookville Kansas 66502	$1,400,000.00	3 br	2 full			
53985 322245.0000	1510 Williamsburg Dr Kansas 66502	$322,245.00	3 br	3 full	4500 sq. ft.		53985
54446 265900.0000	3313 Germann Dr Kansas 66502	$265,900.00	6 br	3 full	3500 sq. ft.		54446

Figure 5.12 Screenshot of Search Webpage with Drag and Drop Tool

5.3.3 Details Webpage


Real Estate Application - Windows Internet Explorer
http://localhost:4958/RealEstate/details.aspx?ThumbNail="N"&ID=2
ABOUT US
FIND A HOME
RENTAL
CONTACTS
SEARCH AGENT OFFICE
Location
Enter City, State, Zip Code
MLS#/ Listing ID
Enter MLS#
Open Houses Only
Search
PRICE
Price Range
Square Feet
SIZE
TYPE
FEATURES
Company | Buying | New Homes | Selling | Renting | Support | Contact

MLS# 53311
PRICE 239900.0000
LOCATION 3930 Golden Eagle Dr Manhattan Kansas , 66502
4 bedroom - 2 bath
DESCRIPTION

Figure 5.13 Screenshot of Details Webpage

CHAPTER 6 - Project Metrics and Experience

This chapter presents the project metrics showing the number of hours spent completing each phase of the project. It also summarizes the experiences gained during the entire life-cycle of the project.

6.1 Project Metrics

Project Metrics are the indicators that track the ongoing project progress. The Project Metrics discussed in this document are source lines of code and the amount of time spent during the entire project span. Table 1 and Table 2 represent the project metrics: source lines of code and project phases and their duration respectively.

ASP.NET & C# Server Side Code	Handwritten C# Code – 1180 lines approx. Auto-generated C# Code – 750 lines approx.
SQL Code	300 lines
CSS Code	500 lines

Table 6.1: Project Lines of Code

Learning Project Technologies	1 Week
Requirement Gathering and Design	2 Week
Implementation	4 Weeks
Testing	1 Week
Documentation	2 Weeks

Table 6.2: Project Planning Phase

6.2 Overall Experience

The idea of developing Real Estate Web Application originated when an immediate requirement aroused from local Real Estate Company to develop a new website for them. This company had its own website in production but wanted to re-design it with some attractive features which could help them fetch more business. The project started with an intention to develop an application specific to the needs of this company but later due to the lack of funds the project could not be continued. But with the continuous support of Dr. Andresen this application came into development. There are hundreds of real estate websites on the World Wide Web with same features but the intention of building this application was to design something new and innovative and include some cool features which have not been incorporated in these websites so far. The biggest challenge involved in this project was to gather the requirements and design its structure so that it can altogether have a new look. It also involved thinking about new features which can be incorporated in this application and could make the search of listings much easier for the real estate buyers. Understanding the structure of the application was the biggest problem on the start of the project. Finally, the scope of this application was defined which greatly helped in understanding what all features have to be included in the project. The whole emphasis in this application is given on the search criteria to help buyers to search for new property listings.

After the Requirement Specification, learning .NET 2005 and new AJAX controls was the second phase which gave me a great learning experience. From my past knowledge of .NET, working with the new version of Microsoft Visual Studio was not a big problem but incorporating AJAX features to it was a deal. I spent around two weeks of time learning these technologies with the help of online tutorials and sample applications. This learning brought me in a very comfortable position to think about what new AJAX features I can put up in the application.

Another problem I faced in my project was the implementation of the drag and drop tool which was required to store the saved listings by the buyer. Traditionally, to save the listings the buyer has to register to the website so that the selected listings can be saved corresponding to the account of the user but this project implements this using the session. As soon as the user starts a search a new session is created and during this session the buyer can store the listings in the tool by dragging and dropping it on the listing cart. Once the user resets the search or starts a new search, a new session is created and the saved listings are lost. This session based application does not force the buyer to log in or register to just select the property listings.

I have learnt also of new things while developing dynamic web-applications using Microsoft ASP.NET and AJAX. This project gave me an opportunity to learn a lot of new concepts in programming and web development such as Java Script, AJAX controls and ASP.NET 2.0 which would help me throughout my career.

CHAPTER 7 - Conclusion and Future Work

This chapter describes the future scope and extensions for the project. There is still a huge scope of implementing something new and more to the project which can make it to the level of a commercial product. This section also concludes stating the advantages and applications of this Real Estate Web Application.

7.1 Conclusion

This Real Estate Web Application is a typical .NET web application using ASP.NET 2.0 and SQL 20005 in the C# programming language. It uses a client/server architecture based on the HTTP protocol. It is developed in Microsoft's Visual Studio .NET programming environment. The buyer performs a search for the property listings by putting either Zip code/City/State or MLS# in the search textbox. The business logic tier communicates with the database tier requesting the results of the query sent by it. The results obtained by the database are displayed on the data grid, by refreshing the grid rather than refreshing the entire web page. Efficiency of the application is improved by the use of web methods that help in separating Application Tier from the Presentation Tier. The performance of this application is evaluated by rigorously testing it against various test scenarios. Efficiency and correctness of the application is evaluated with the help of various test cases. Some ways in which this system could be enhanced with additional functionalities are discussed in the section.

7.2 Future Extensions to the Project

This project is developed as a master's project and still gives lot of scope for its extension which could be made to the project if it is going to be developed as commercial product. We can use pure object-oriented domain model to deal the database access tier using Visual Studio 2005 and SQL Server 2005. In doing this, we can get a better architecture design which will improve code efficiency and running performance. Besides, we can build XML web service programming model that enables other applications to consume real estate web services built by us using standard protocol such as HTTP, XML, XSD, SOAP and web services description language (WSDL). This project just deals with the Home page and Search page to search for property listings, more functionality can be added for searching the agents and offices making it a complete application. The feature of providing Google Maps within this application adds up to the functionality of the website. With the advancement of technology, dynamic maps can be generated using AJAX which can help the buyer locate a particular area where the property is located in the Google Map. Inclusion of all these features would make the application feature rich. The advantages of putting these functionalities in the project are described in detail in the following sections.

7.2.1. Using Object-Oriented Domain Model

Visual Studio .NET 2005 and SQL Server 2005 add a significant feature of object-oriented domain model. Instead of using traditional Relational Database Model, the database access layer treat each table as class model and each row as instance object. It gives a friendly development thought to the developer and makes the business logic more convenient when interacting with the back end. Currently, there is a third-party tool called NEO which could generate the domain model for database automatically. In using object-oriented domain model, the database is transported to the developer and the application using these domain classes to handle data processing instead of using disconnected Dataset or typed Dataset.

7.2.1.1 Providing Web Services

XML and web services are used to build highly scalable, loosely coupled, distributed application using standard web protocols such as HTTP, XML, and SOA. An XML web service is a component that implements the program logic and provides functionality to other applications. There are some examples such as Amazon.com, Google.com. WSDL is used to describe XML web service and XML-based messaging is used to send and receive data.

Visual Studio .NET IDE has a powerful support to develop XML web services. We can create a real estate web application which provides application logic to a client application. For example, I can create a search function to search the houses based on the feature. The client can call my web service and provide parameters to the search function and the web service will return dataset containing the result house records list back to the client. We can create other convenient functions and the client can easy build their own custom web application while still using the programming logic from the server web services.

7.2.1.2 Functionality Extensions

If we are going to develop commercial real estate web application based on this project, we can add some more feature as the following:

1. We can provide a login page for buyers and store their username and password in the database and can save the search criteria for the buyers. This will help the buyers to use the same search criteria rather than creating a new criterion every time the buyer performs the search.
2. By using the AdRotator web control we could add more advertisement on the web site or links to other web site.
3. Add agent and company search functionality.
4. Displaying the search results on Google Maps locating the area where the property listing is located.

4. Refine this web site and make it friendly and pretty.

References

[1] ASP.NET and Web Development Overview,
http://msdn.microsoft.com/en-us/library/4w3ex9c2.aspx

[2] Microsoft Visual Studio Overview,
http://en.wikipedia.org/wiki/Microsoft_Visual_Studio#Visual_Studio_2005

[3] Microsoft SQL Server 2005,
http://en.wikipedia.org/wiki/Microsoft_SQL_Server

[4] 3-Tier System Architecture,
http://en.wikipedia.org/wiki/Multitier_architecture

[5] Introduction to ADO.NET,
http://en.wikipedia.org/wiki/ADO.NET

[6] Load Testing using Apache JMeter Testing Tool,
http://jakarta.apache.org/jmeter/

[7] Introduction to .NET Architecture,
http://www.devtopics.com/what-is-net/

[8] Common Language Infrastructure Definition,
http://en.wikipedia.org/wiki/Common_Language_Infrastructure

[9] AJAX – Bridging the Thin-Client Performance Gap,
http://www.ironspeed.com/articles/ajax-bridging%20the%20thin-client%20performance%20gap/article.aspx

[10] 3-Tier Architecture

http://www.c-sharpcorner.com/

.