



Jaypee University of Information Technology  
Solan (H.P.)

**LEARNING RESOURCE CENTER**

Acc. Num. SP5014 Call Num:

**General Guidelines:**

- ◆ Library books should be used with great care.
- ◆ Tearing, folding, cutting of library books or making any marks on them is not permitted and shall lead to disciplinary action.
- ◆ Any defect noticed at the time of borrowing books must be brought to the library staff immediately. Otherwise the borrower may be required to replace the book by a new copy.
- ◆ The loss of LRC book(s) must be immediately brought to the notice of the Librarian in writing.

Learning Resource Centre-JUIT



SP05014

**ONLINE STUDENT INTERACTION AND  
EVALUATION SYSTEM**

**By**

**ISHITA VERMA-051004  
SHUBHAM GUPTA-051014  
PRANAV BHASKER-051017**



**MAY-2009**

**Submitted in partial fulfillment of the Degree of  
Bachelor of Technology**


**DEPARTMENT OF  
ELECTRONICS AND COMMUNICATION ENGINEERING**



**JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY  
WAKNAGHAT, SOLAN**

**CERTIFICATE**

This is to certify that the work entitled, "**ONLINE STUDENT INTERACTION AND EVALUATION SYSTEM**" submitted by Ishita Verma, Shubham Gupta and Pranav Bhasker in partial fulfillment for the award of degree of Bachelor of Technology in Electronics and Communication engineering of Jaypee University of Information Technology has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

  
Rohit Sharma  
(Project Guide)

## ACKNOWLEDGEMENT

As we summarize our project in the form of this report, we express immense gratitude to our respected project guide and mentor Rohit Sharma without whose motivation, guidance and constant reviews it was impossible to complete the project undertaken. We wish to thank him for showing support and faith in us at each step we approached him.

We would also like to thank Chetna Gupta, Department of Computer Science, JUIT for her concern and valuable suggestions during various session of discussion that helped us to bring out our full potential towards successful completion and submission of project.

Last but not the least; we are thankful to almighty and all our family members and friends for their moral support and encouragement in this endeavor and throughout our studies.

*Ishita*

Ishita Verma

(051004)

*Shubham Gupta*

Shubham Gupta

(051014)

*Pranav Bhasker*

Pranav Bhasker

(051017)

## TABLE OF CONTENTS

CHAPTER	TOPIC	PAGE
	List of figures	vi
	List of abbreviations	viii
	Abstract	ix
<b>1</b>	Introduction	1
	1.1 Functional components of the project	2
	1.1.1 Technology Used	3
	1.2 Client-server model	4
	1.3 Approach	6
	1.3.1 System Analysis & Requirement Specification	6
	1.3.2 Software Architecture & Design	7
	1.3.3 Coding	7
	1.3.3 Testing Software Application	7
<b>2</b>	Web Server	8
	2.1 Introduction	9
	2.1.1 Apache	10
	2.1.2 Advantages of Apache	10
<b>3</b>	HTML	12
	3.1 Introduction	13
	3.2 Working of HTML	13
	3.3 Tags in HTML	13
	3.4 HTML with Cascading Stylesheet and JavaScript	14
	3.5 HTML Advantages	14
	3.6 HTML Disadvantages	15

4	Macromedia Dreamweaver	17
	4.1 Introduction	18
	4.2 Features	19
	4.2.1 Dreamweaver interface	20
	4.2.2 Syntax highlighting	21
	4.3 Language Availability	22
5	XAMPP	23
	5.1 About XAMPP	24
	5.2 XAMPP installation requirements	25
6	PHP and MySQL	28
	6.1 Web Database Application	29
	6.1.1 The Database	30
	6.1.2 The Application	31
	6.2 MySQL Database	31
	6.2.1 Advantages of MySQL	32
	6.2.2 How MySQL works	33
	6.3 PHP	34
	6.3.1 Advantages of PHP	35
	6.3.2 How PHP works	36
	6.4 MySQL and PHP, the Perfect Pair	37
	6.4.1 Advantages of the relationship	38
	6.4.2 How MySQL and PHP work together	38
	6.4.3 The PHP MySQL Architecture	40
7	AJAX	42
	7.1 What is AJAX?	43
	7.1.1 Where did it come from?	43
	7.2 What Makes AJAX Cool	44
	7.3 AJAX Technologies	46

	7.3.1	XML	46
	7.3.2	Document Object Model	47
	7.3.3	XMLHttpRequest	47
	7.3.4	JavaScript	47
	7.4	The AJAX framework – An overview	48
	7.5	AJAX based communication between a browser and a Web server	49
	7.6	Advantages of AJAX	50
	7.7	Various Sites Using AJAX	51
<b>8</b>		Project Modules	55
	8.1	User Registration	56
	8.2	Online Quiz	58
	8.3	Personal Problem Module	69
	8.4	Fee Management Module	71
<b>9</b>		Conclusion	72
		Bibliography	73

## LIST OF FIGURES

- |                   |  |
|-------------------|--|
| 1. Figure 1.2     | A typical client-server interaction  |
| 2. Figure 4.2.1   | Code View  |
| 3. Figure 4.2.2   | Design View  |
| 4. Figure 5.2.1   | XAMPP Installation   |
| 5. Figure 5.2.2   | XAMPP Control Center   |
| 6. Figure 6.4.3   | The PHP MySQL Architecture   |
| 7. Figure 7.2.1   | A traditional web app is synchronous system  |
| 8. Figure 7.2.2   | An Ajax application making asynchronous request to a web server                        |
| 9. Figure 7.4     | Initiating and sustaining Ajax based communications between a browser and a web server |
| 10. Figure 7.7.1  | Google Suggest   |
| 11. Figure 7.7.2  | Netflix.com  |
| 12. Figure 7.7.3  | instantdomainsearch.com  |
| 13. Figure 8.1.1  | Field-left empty   |
| 14. Figure 8.1.2  | User already exists  |
| 15. Figure 8.2.1  | Setting the Parameters   |
| 16. Figure 8.2.2  | Data Entered Successfully  |
| 17. Figure 8.2.3  | Preparing question bank  |
| 18. Figure 8.2.4  | Go to previous question  |
| 19. Figure 8.2.5  | Marked for review  |
| 20. Figure 8.2.6  | Indicate next question is last   |
| 21. Figure 8.2.7  | Display no. of attempted or not attempted questions                                    |
| 22. Figure 8.2.8  | Finish exam  |
| 23. Figure 8.2.9  | Time duration is over  |
| 24. Figure 8.2.10 | Result of quiz   |
| 25. Figure 8.3.1  | Student making a request through OSIES.  |
| 26. Figure 8.3.2  | Admin forwarding message to concerned department                                       |



27. Figure 8.3.3 Student receiving the status of his request

28. Figure 8.4.1 Fee Management

## LIST OF ABBREVIATIONS

PHP	HYPertext PREPROCESSOR
HTML	HYPertext MARKUP LANGUAGE
HTTP	HYPertext TRANSFER PROTOCOL
IIS	INTERNET INFORMATION SERVICES
DOM	DOCUMENT OBJECT MODEL
IP	INTERNET PROTOCOL
ISP	INTERNET SERVICE PROVIDER
DNS	DOMAIN NAME SERVER
AJAX	ASYNCHRONOUS JAVASCRIPT AND XML
GPL	GENERAL PUBLIC LICENCE
CSS	CASCADING STYLE SHEETS

## ABSTRACT

---

This project is aimed at developing an Online Student Interaction and Evaluation System (OSIES) for the students in the campus. This is an Intranet based application that can be accessed throughout the campus. This system can be used to automate the workflow of service requests for the various facilities in the campus as well as student's evaluation. Registered users will be able to log in a request for service for any of the supported facilities. These requests will be sent to the concerned people, who are also valid users of the system, to get them resolved. Also the student will be evaluated for their internal marks on the basis of performance in the online exam.

It is anticipated that this Web Application will form the base for a full management system for the university which includes strategic, academic and management components. Key objective of implementing the Online Student Interaction and Evaluation System is to have better communication between faculty, student and staff in achieving high performance outcome.

CHAPTER – 1

INTRODUCTION

## **Introduction**

---

In today's dynamic and expanding modern world, computers are at the forefront of the technology revolution and as aptly it can be, the given era is often referred to as the 'Era of Computers'. The wide ranges of applications of these smart machines just confirm their growing power in today's developing world.

Now one of their important applications includes the coordination between different members of an organization or an institution. An institution is based on the cooperation of different people who work together for its success. While working they need to interact with each other to exchange ideas or knowledge or discuss problems so as to get the work done within the time duration to achieve high performance outcome.

This project explores the impact of online evaluation and interaction of a student with the administration regarding issues within the university campus. This project basically aims to reduce the paper work in the university and provide an alternative method for that.

Here we present an implementation of a database driven Web Application using PHP and MySQL to design a system which can meet the following requirements:

- Evaluations of students for internal assessment through Online Quiz for each course in which they are registered.
- Allow student to make online request to administration for- Concession form, Bonafide certificate, Grade-sheet etc.
- Allow student to make a request for an available book in the library.
- Fee management of the students during the course

### **1.1 Functional components of the project**

There are registered people in the system (students, faculty, lab-assistants and others). Some of them are responsible for maintaining the facilities (like, the lab-assistant is responsible for keeping the lab ready with all the equipment in proper condition).

There are three kinds of users for this system:

- those who use the system to create a request (students)
- those who look at the created requests and assign them to the concerned people (facility-heads)
- those who work on the assigned requests

There is also an 'Administrator' for doing the Admin-level functions such as creating user accounts, adding new facilities to the system etc.

### **1.1.1 Technology Used**

The technology or software requirements of this project are mentioned below:

- Dreamweaver
- HTML
- Web Server (Apache or IIS)
- Database (MySQL or Oracle or SQLite)
- PHP (server side scripting)
- AJAX

## **1.2 Client-Server Model**

A client/server system consists of two different types of computers - client computers and server computers - connected by a network. The client computers are where users do their work. The server computer controls the data (information, files, or even computer programs) needed to do that work. Users on client computers use client software to request the data they need from the server computer. Server software on the server computer receives the requests for data and responds to them. For example, a client application might send a message to the server application asking for all the documents that match a set of criteria; the server application would then respond by locating the appropriate documents and then transferring them to the client computer over the network. Client/server computing is important because it centralizes the control of data.

There are some responsibilities that the client is responsible for, below are some of the responsibilities: -

The client's responsibility is usually to:

1. Handle the user interface.
2. Translate the user's request into the desired protocol.
3. Send the request to the server.
4. Wait for the server's response.
5. Translate the response into "human-readable" results.
6. Present the results to the user.

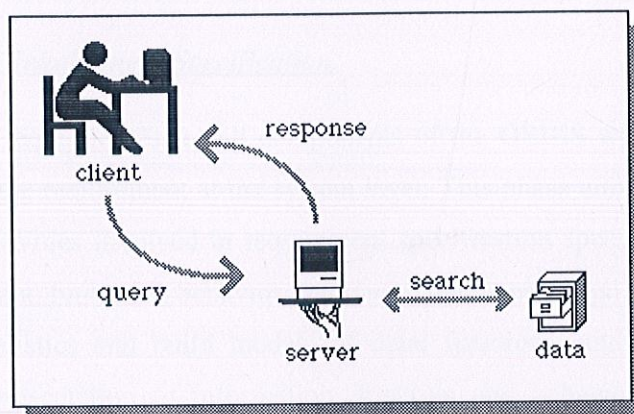
There are some functions that the server has to carry, and are very essential. Here are the functions that have to be carried out: -

1. Listen for a client's query.
2. Process that query.
3. Return the results back to the client.

There are interactions that are carried out by the client/server. Here are some examples of the interactions that are carried out: -

1. The user runs client software to create a query.
2. The client connects to the server.
3. The client sends the query to the server.
4. The server analyzes the query.
5. The server computes the results of the query.
6. The server sends the results to the client.
7. The client presents the results to the user.
8. Repeat as necessary.

Here is a simple diagram of the interaction: -



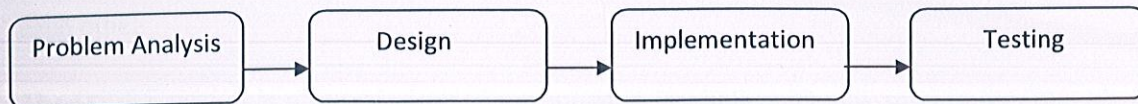
**Fig 1.2: A typical client-server interaction**

This client/server interaction is a lot like going to a French restaurant. At the restaurant, you (the user) are presented with a menu of choices by the waiter (the client). After making your selections, the waiter takes note of your choices, translates them into French, and presents them to the French chef (the server) in the kitchen. After the chef prepares your meal, the waiter returns with your dinner (the results). Hopefully, the waiter returns with the items you selected, but not always; sometimes things get "lost in the translation."



### 1.3 Approach

The diagrammatic representation of the approach we used for developing the system:



We adopted the above approach which consisted of the following phases:

- System Analysis & Requirement Specification (Problem Analysis Phase)
- Software Architecture & Design (Design Phase)
- Coding (Implementation Phase)
- Testing Software Application (Testing Phase)

#### 1.3.1 System Analysis & Requirement Specification

In this phase, firstly the requirements for all components of the **OSIES** were worked out. The system analysis denotes the requirement at the system level. This phase aimed at specifying and defining **OSIES**. The activities involved in requirement specification specified various **OSIES** characteristics: such as data, functions, behavior, interfaces, and constraints. Analysis allowed us to define these characteristics and build models of data, functions, and predefined **OSIES** behavior. Later, the representation of information, function and behavior was translated to **OSIES** design.

This phase helped us in gaining a clear understanding of the requirements to be met and the functions that the **OSIES** needed to perform these requirements.

### **1.3.2 Software Architecture & Design**

The **OSIES** architecture defines a framework to assemble the individual components of the project. It describes the manner in which the components should be integrated and interact with each other.

The **OSIES** design is representation of the software product to be delivered in the final phase of the development process. This design was prepared after identifying the **OSIES** architecture to be used. In this phase, the overall layout of the project was worked out. Therefore, this phase can be considered as a bridge between the analysis and the construction of **OSIES**.

### **1.3.3 Coding**

After high level designing of **OSIES**, with the help of design notations and pseudo code, programming was completed. The motive of coding phase was to convert the design of **OSIES** in a particular programming language i.e. **PHP**. In this phase we also laid great emphasis on the technology requirements for the implementing of system.

### **1.3.4 Testing Software Application**

The main objective of this phase was to make sure that **OSIES** work properly. The key objectives of this phase were:

- test should find as many errors as possible in the system.
- test was termed successful when it detected a previously undetected error.

## CHAPTER – 2

# Web Server

## 2.1 Introduction

---

Web servers are computers on the Internet that host websites, serving pages to viewers upon request from their browsers. When multiple web sites are available from a single web server the service provided is referred to as web (site) hosting. Every web server has a unique address so that other computers connected to the Internet know where to find it on this really immense network. The **IP** (Internet Protocol) address looks something like XXX.XXX.XXX.XXX. Three set of significant numbers with each set of three numbers having a unique meaning (i.e. 192.168.0.101). Finally, this address is mapped to a more human friendly address, such as [www.juit.ac.in](http://www.juit.ac.in) using **Domain Name Servers (DNS)**.

Web hosts rent out space on their web server's hard disk drive, to people or businesses, so that they set up their websites. The web server allocates a unique website address to each website it hosts. When connected to the internet, the computer on which the hard disk resides is assigned a unique IP address by the **Internet Service Provider (ISP)** i.e. the company that provides the connection (or pipeline) to the Internet. This address identifies the computer on the network. When a website's URL like [www.juit.ac.in](http://www.juit.ac.in) is keyed into a browser's address bar and **GO** is clicked the browser broadcasts a request for the website's **IP** address. The browser's request includes return information. The request passes through several computers on the way to [www.juit.ac.in](http://www.juit.ac.in), each routing it closer to its ultimate destination.

When the request reaches its destination, the web server that hosts the [www.juit.ac.in](http://www.juit.ac.in) website immediately returns an **HTML** page to the client machine that attempted contact. This return HTML page travels back through the network. The client computer receives the HTML page and its browser interprets the HTML code and then displays the page on the computer's **VDU** in graphic form.

The more powerful the web server, the faster it can serve website pages (i.e. the faster it can respond to client browser requests). Slower, less powerful web servers may result in a frustrating lag time for viewers. High traffic can also slow servers that are not powerful enough to handle high volumes of data being exchanged. This lag time should be a concern for e-commerce websites. Most web hosts have a page dedicated to sharing technical information about their web

server(s), which generally include processor speed, cache, capacity, network configuration, internet pipeline width and other details.

The web server used in this project is Apache.

### **2.1.1 APACHE**

The **Apache HTTP Server**, commonly referred to simply as **Apache** evolved from a patchy server, is a web server notable for playing a key role in the initial growth of the World Wide Web and in 2009 became the first web server to surpass the 100 million web site milestone. Apache was the first viable alternative to the Netscape Communications Corporation web server (currently known as Sun Java System Web Server), and has since evolved to rival other Unix-based web servers in terms of functionality and performance. The majority of all web servers using Apache are Linux web servers.

Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. The application is available for a wide variety of operating systems, including Unix, GNU, FreeBSD, Linux, Solaris, Novell NetWare, Mac OS X, Microsoft Windows, OS/2, TPF, and eComStation. Released under the Apache License, Apache is characterized as free software and open source software.

Since April 1996 Apache has been the most popular HTTP server on the World Wide Web. As of March 2009 Apache served over 46% of all websites and over 66% of the million busiest.

### **2.1.2 Advantages of Apache**

Apache offers various advantages to users, developers and Web administrators:

- ***It is Free.*** No licensing fee is required to use it. It is open source software.
- ***It runs on a variety of operating system.*** It is cross platform and can be run on windows, linux, Mac-Os etc.
- ***It is popular.*** Among 60% of the websites on the Internet use apache. Also it means that a large group of users can provide help.

- ***It is customizable.*** The open source license allows programmers to modify the apache software, adding or modifying modules as needed to fit their own environment.
- ***It is secured.*** Free software are available that runs with apache to make it into a secure SSL server. SSL is used to provide extra security for websites that need to protect important information. It means the information passed between the web server and the browser is encrypted so that no one can intercept and read it.
- ***It is reliable.*** Emergency problems with apache are extremely rare.

## CHAPTER – 3

# HTML

## HTML-HyperText Markup Language

---

### 3.1 INTRODUCTION

HTML is a computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively easy to learn, with the basics being accessible to most people in one sitting; and quite powerful in what it allows you to create. It is constantly undergoing revision and evolution to meet the demands and requirements of the growing Internet audience under the direction of the worldwide web, the organization charged with designing and maintaining the language.

The definition of HTML is **HyperText Markup Language**.

- **HyperText** is the method by which you move around on the web — by clicking on special text called **hyperlinks** which bring you to the next page. The fact that it is **hyper** just means it is not linear, i.e. you can go to any place on the Internet whenever you want by clicking on links, and there is no set order to do things in.
- **Markup** is what **HTML tags** do to the text inside them. They mark it as a certain type of text (*italicised* text, for example).
- HTML is a **Language**, as it has code-words and syntax like any other language.

### 3.2 Working of HTML

HTML consists of a series of short **codes** typed into a text-file by the site author — these are the tags. The text is then **saved as a html file**, and **viewed through a browser**, like **Internet Explorer** or **Netscape Navigator**. This browser reads the file and translates the text into a visible form, hopefully rendering the page as the author had intended. Writing your own HTML entails using tags correctly to create your vision. You can use anything from a rudimentary text-editor to a powerful graphical editor to create HTML pages.



### **3.3 Tags in HTML**

The tags are what separate normal text from HTML code. You might know them as the words between the `<angle-brackets>`. They allow all the cool stuff like images and tables and stuff, just by telling your browser what to render on the page. Different tags will perform different functions. The tags themselves don't appear when you view your page through a browser, but their effects do.

### **3.4 HTML with Cascading Stylesheet and JavaScript**

**Cascading Stylesheet** is used to control how your pages are presented, and make pages more accessible. Basic special effects and interaction is provided by **JavaScript**, which adds a lot of power to basic HTML. Most of this advanced stuff is for later down the road, but when using all of these technologies together, you have a lot of power at your disposal.

### **3.5 HTML Advantages**

If compatibility with user habits, expectations and multiple platforms is the goal, then HTML is the only approach to delivering a web application.

#### **➤ The term is browser**

After all, it is a browser, not a menu system. Many approaches to delivering terminal session on the Web use web technology as a menu only. Once the user has clicked on the link that connects to the host, a dedicated window appears containing yet another standard terminal emulator.

This approach fails in exploiting many of the benefits that have made the web such a success. When a user is launched into an environment not based on HTML, the user interface changes.

#### **➤ Consistent and effective**

Delivering host sessions using HTML provides the user with a consistent interface and provides developers a highly effective medium for presenting information. As millions of Web devotees have demonstrated, HTML works. Given that much of the flow of information is out of hosts, isn't it preferable to deliver it in the most effective medium?

➤ *Ideal for expanding cryptic screens*

One of the major usability problems with the fixed format host screen is that over time, many applications have sacrificed prompts and other textual user cues in favor of more data. As business and organizational processes add new informational requirements, the data expands and the "fluff" is pushed off of the limited screen real estate.

Unfortunately, that fluff is the difference between an easily understood display and a cryptic mass of data requiring training manuals and frequent help from co-workers.

➤ *The freedom of a text markup environment*

One of the first things an HTML author learns is that once a good information presentation pattern is established, the size of the page can expand well past the constraints of the user's screen. The dynamic formatted characteristics of browsers coupled with the dynamic scrolling of the page free developers from prior constraints which applied to graphical user interfaces as well.

For rapidly reformatting information screens for superior understanding and reduced training requirements, HTML offers the best medium to developers.

### 3.6 HTML Disadvantages

In formatting, HTML is weakened because of its complexity and incompatibility. Browsers don't all adhere to a single standard, where other formatting (.doc files, .txt files, etc) are more universal and/or are proprietary, so will always look the same. HTML isn't strict-- or at least, no browsers are in the least. You can type `<br>` without a terminating tag, or without specifying `<br/>`. Or you don't have to specify double quotes in your attribute values. That sort of thing. So many people's uses of HTML create a lot of misunderstanding about the language. If talking about usage on a website, HTML is static. It's not a programming language; it's a markup language, so you can't do things like save user input, let users log in, etc. You can fake it by inserting JavaScript and other applications in there, but HTML files themselves aren't dynamic. HTML is ugly to read. Because it's often so complex and in-depth, there aren't any adopted standards for formatting or composition. So interpreting what's going on is often very difficult. Everyone's got their hands in HTML, so HTML is often not "pure". It's often got Java-

script or CSS mixed in, and is integrated into templates like PHP or ColdFusion, or even made quasi-dynamic with SSI (Server-Side Includes). However, for all its faults (and they're not really all that bad), HTML is generally VERY well accepted across various display programs. Email readers, web browsers, word processors, spreadsheets, etc., etc. Many programs know how to read, display, or output HTML-- so it can be very handy for universality.

CHAPTER – 4

DREAMWEAVER

## MACROMEDIA DREAMWEAVER

---

### 4.1 Introduction

Macromedia Dreamweaver is a professional HTML editor for designing, coding, and developing websites, web pages, and web applications. Whether you enjoy the control of hand-coding HTML, or prefer to work in a visual editing environment, Dreamweaver provides you with helpful tools to enhance your web creation experience.

The visual editing features in Dreamweaver let you quickly create web pages without writing a line of code. You can view all your site elements or assets and drag them from an easy-to-use panel directly into a document. You can streamline your development workflow by creating and editing images in Macromedia Fireworks or another graphics application, and then import them directly into Dreamweaver. Dreamweaver also provides tools that make it easy to add Flash assets to web pages.

In addition to drag-and-drop features that help you build web pages, Dreamweaver provides a full-featured coding environment that includes code-editing tools (such as code coloring, tag completion, a coding toolbar, and code collapse) and language reference material on Cascading Style Sheets (CSS), JavaScript, ColdFusion Markup Language (CFML), and other languages. Macromedia Roundtrip HTML technology imports your hand-coded HTML documents without reformatting the code; you can then reformat code with your preferred

Dreamweaver also lets you build dynamic, database-driven web applications using server technologies such as CFML, ASP.NET, ASP, JSP, and PHP. If your preference is for working with XML data, Dreamweaver provides tools that let you easily create XSLT pages, attach XML files, and display XML data on your web pages.

Dreamweaver is fully customizable. You can create your own objects and commands, modify keyboard shortcuts, and even write JavaScript code to extend Dreamweaver capabilities with new behaviors, Property inspectors, and site reports.

## 4.2 Features

Although a hybrid WYSIWYG (What You See Is What You Get) and code-based web design and development application, Dreamweaver's WYSIWYG mode can hide the HTML code details of pages from the user, making it possible for non-coders to create web pages and sites. One criticism of this approach is that it has the potential to produce HTML pages whose file size and amount of HTML code is larger than an optimally hand-coded page would be, which can cause web browsers to perform poorly. This can be particularly true because the application makes it very easy to create table-based layouts. In addition, some web site developers have criticized Dreamweaver in the past for producing code that often does not comply with W3C standards, though recent versions have been more compliant. Dreamweaver 8.0 performed poorly on the Acid2 Test, developed by the Web Standards Project. However, Adobe has focused on support for standards-based layout in recent and current versions of the application, including the ability to convert tables to layers.

Dreamweaver allows users to preview websites in locally-installed web browsers. It also has site management tools, such as FTP/SFTP and WebDAV file transfer and synchronization features, the ability to find and replace lines of text or code by search terms and regular expressions across the entire site, and a templating feature that allows single-source update of shared code and layout across entire sites without server-side includes or scripting. The behaviors panel also enables use of basic JavaScript without any coding knowledge, and integration with Adobe's Spry AJAX framework offers easy access to dynamically-generated content and interfaces.

Dreamweaver can utilize third-party "Extensions" to enable and extend core functionality of the application, which any web developer can write (largely in HTML and JavaScript). Dreamweaver is supported by a large community of extension developers who make extensions available (both commercial and free) for most web development tasks from simple rollover effects to full-featured shopping carts.

Like other HTML editors, Dreamweaver edits files locally, then uploads all edited files to the remote web server using FTP, SFTP, or WebDAV. Dreamweaver CS4 now supports the Subversion (SVN) version control system.

## 4.2.1 The Dreamweaver Interface

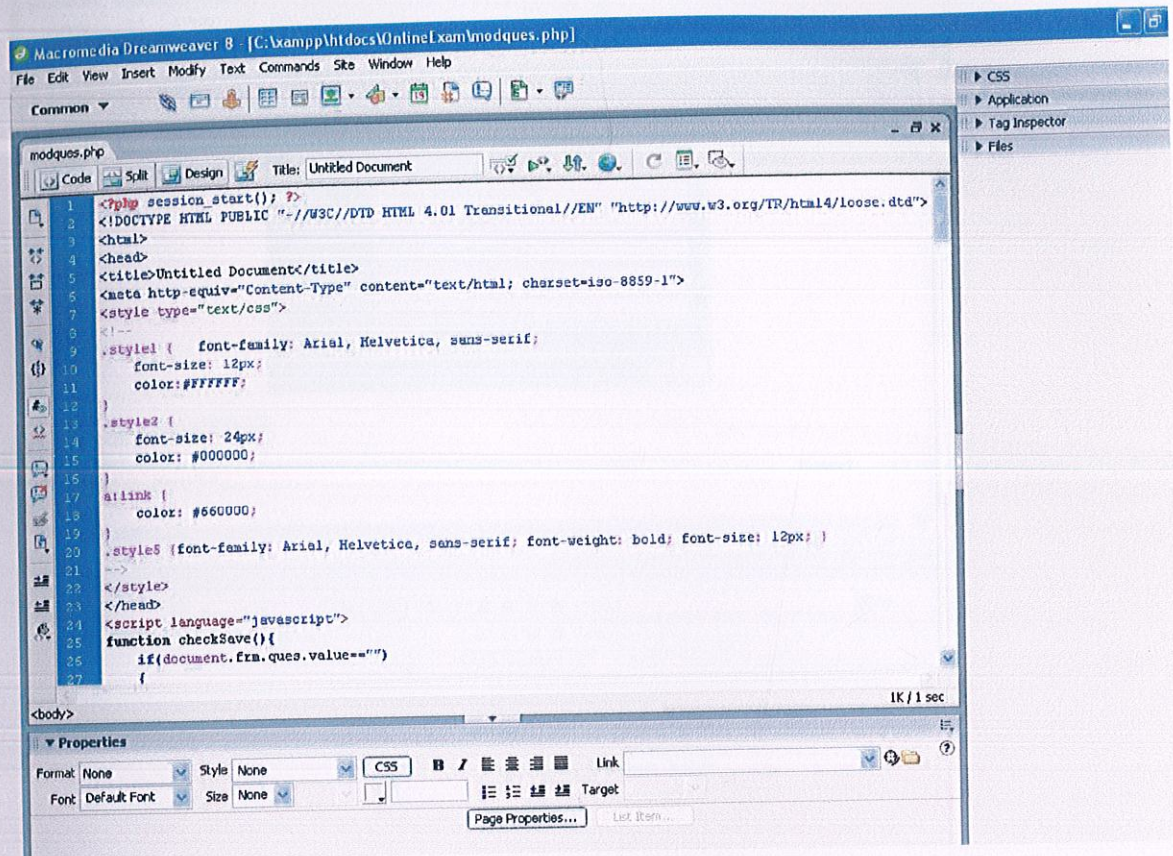


Fig 4.2.1: Code View

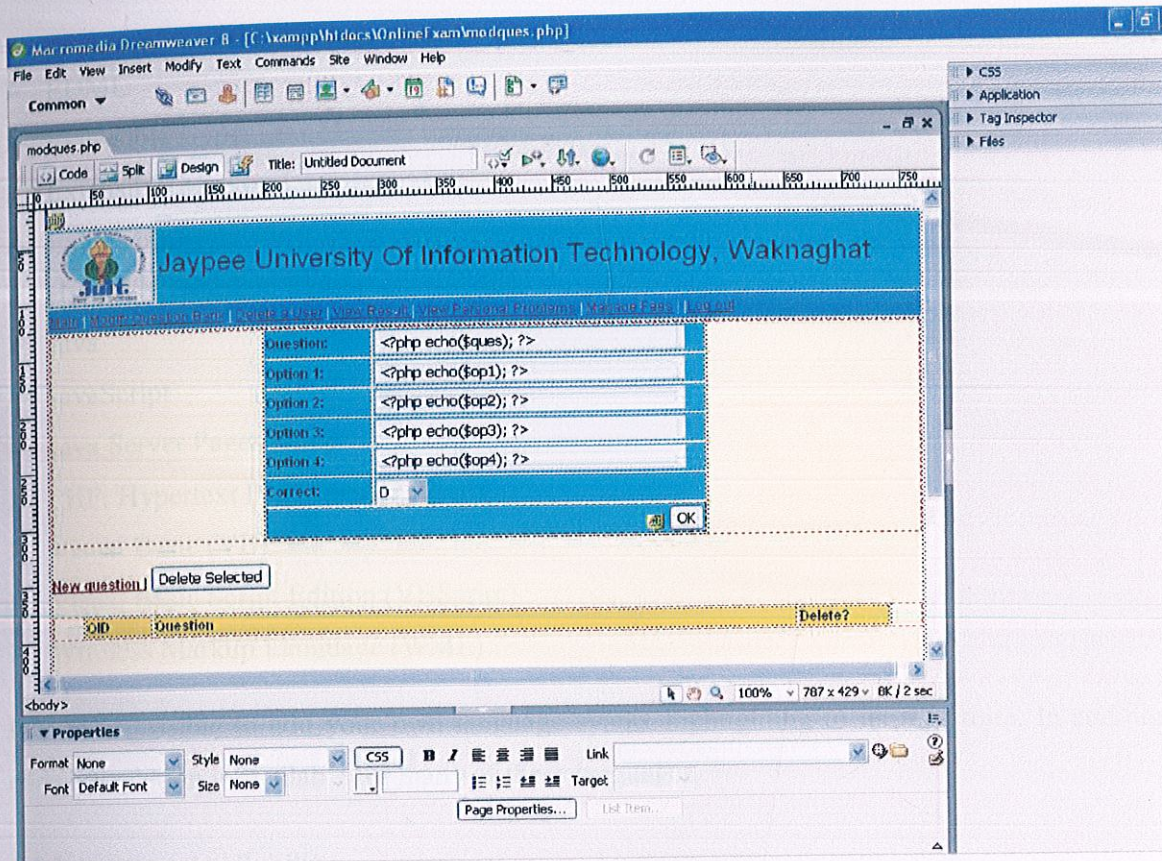


Fig 4.2.2: Design View

#### 4.2.2 Syntax highlighting

As of version 8, Dreamweaver supports syntax highlighting for the following languages out of the box:

- Action Scrip
- Active Server Pages (ASP).
- ASP.NET
- C#
- Cascading Style Sheets (CSS)
- ColdFusion



- EDML
- Extensible HyperText Markup Language (XHTML)
- Extensible Markup Language (XML)
- Extensible Style-sheet Language Transformations (XSLT)
- HyperText Markup Language (HTML)
- Java
- JavaScript
- Java Server Pages (JSP)
- PHP: Hypertext Preprocessor (PHP)
- Visual Basic (VB)
- Visual Basic Script Edition (VBScript)
- Wireless Markup Language (WML)

It is also possible to add your own language syntax highlighting to its repertoire. In addition, code completion is available for many of these languages.

#### **4.3 Language Availability**

Dreamweaver CS4 is available in the following languages: Brazilian Portuguese, Chinese Simplified (Windows only), Chinese Traditional (Windows only), Czech, Dutch, English, French, German, Italian, Japanese, Korean (Windows only), Polish, Russian, Spanish, Swedish and Turkish.

## CHAPTER – 5

# XAMPP

# XAMPP

---

## 5.1 Introduction

XAMPP is a free and open source cross-platform web server package, consisting mainly of the Apache HTTP Server, MySQL database, and interpreters for scripts written in the PHP and Perl programming languages.

XAMPP's name is an acronym for:

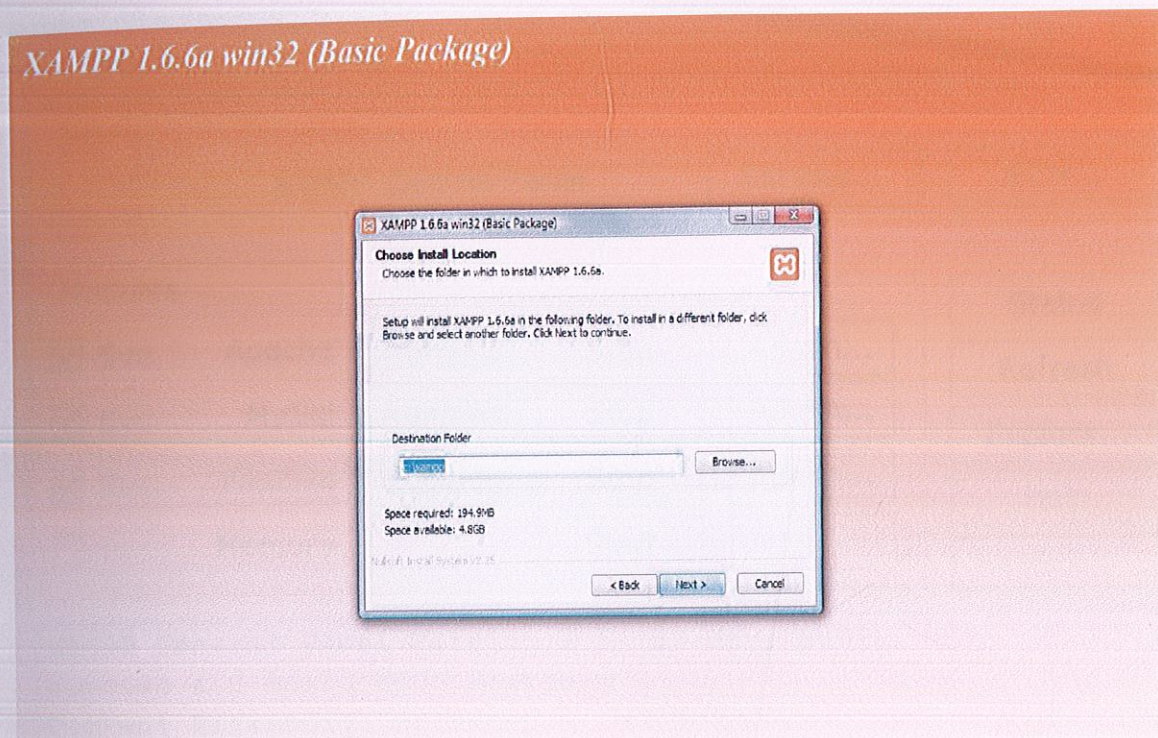
- X (meaning cross-platform)
- Apache HTTP Server
- MySQL
- PHP
- Perl

XAMPP is available for Microsoft Windows, Linux, Solaris, and Mac OS X, and is mainly used for web development projects. XAMPP is regularly updated to incorporate the latest releases of Apache, MySQL, PHP and Perl. Because its user interface is considered simple to use, it is sometimes called the "lazy man's WAMP/LAMP installation.

Officially, XAMPP's designers intended it for use only as a development tool, to allow website designers and programmers to test their work on their own computers without any access to the Internet. To make this as easy as possible, many important security features are disabled by default. In practice, however, XAMPP is sometimes used to actually serve web pages on the World Wide Web. A special tool is provided to password-protect the most important parts of the package.

XAMPP also provides support for creating and manipulating databases in MySQL and SQLite among others. XAMPP is a compilation of free software (comparable to a Linux distribution), it's free of charge and it's free to copy under the terms of the GNU General Public License.

## 5.2 XAMPP Installation Requirements



**Fig 5.2.1: Install XAMPP**

Installing XAMPP takes less time than installing each of its components separately. Self-contained, multiple instances of XAMPP can exist on a single computer, and any given instance can be copied from one computer to another.

XAMPP is a small and light Apache distribution containing the most common web development technologies in a single package. Its contents, small size, and portability make it the ideal tool for students developing and testing applications in PHP and MySQL. XAMPP is available as a free download in two specific packages: full and lite. While the full package download provides a wide array of development tools, this article will focus on using XAMPP Lite which contains the necessary technologies that meet the Ontario Skills Competition standards. As the name implies,

the light version is a small package containing Apache HTTP Server, PHP, MySQL, phpMyAdmin, Openssl, and SQLite.

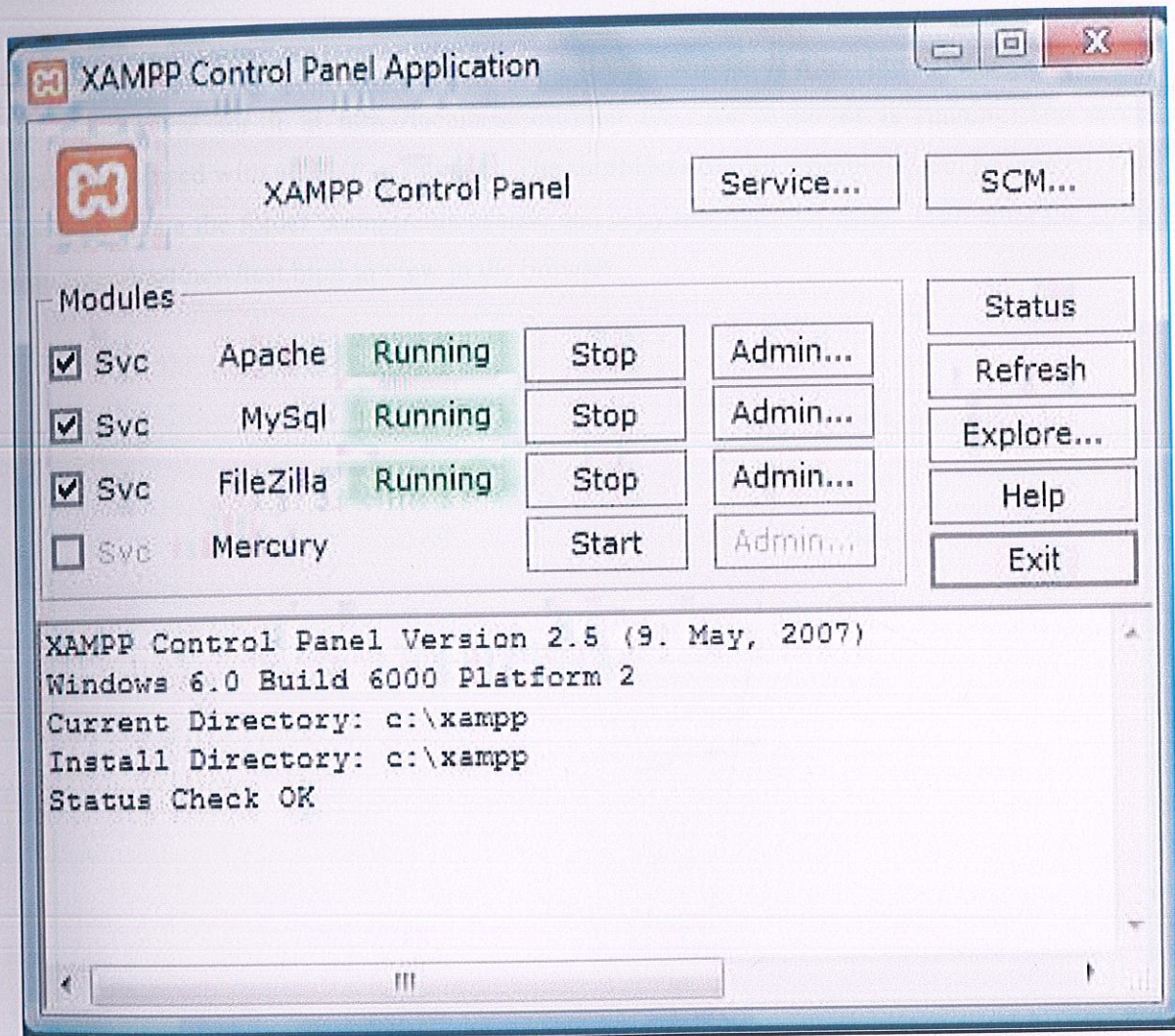


Fig 5.2.2: XAMPP Control Center appears at bottom right desktop tray

**Note:** After the installation is complete, we can use the XAMPP Control Panel to start/stop all server and also install/uninstall services.

The main folder for all WWW documents is at `\xampp\htdocs`. If a `test.html` file is placed here one can browse for it at `http://localhost/test.html` (if Apache server is running). The same procedure is used with all PHP or cgi files. The subfolders for the content too can be created. For example, create the folder `\xampp\htdocs\new` and copy `test.html` file there. Then enter the URL `http://localhost/new/test.html` to view in the browser.

## CHAPTER – 6

# PHP and MySQL

## PHP and MySQL

---

### 6.1 Web Database Application

An *application* is a program or a group of programs designed for use by an end user (for example, customers, members, circus acrobats, and so on). If the end user interacts with the application via a Web browser, the application is a *Web-based* or *Web application*. If the Web application requires the long-term storage of information, using a database, it is a *Web database application*.

A Web database application is designed to help a user accomplish a task. It can be a simple application that displays information in a browser window (for example, it displays current job openings when the user selects a job title) or a complicated program with extended functionality (for example, the book-ordering application at Amazon.com or the bidding application at eBay). Not surprisingly, a Web database application consists of a database and an application — just two pieces:

**Database:** The *database* is the long-term memory of the Web database application. The application can't fulfill its purpose without the database. However, the database alone is not enough.

**Application:** The *application* piece is the program or group of programs that performs the tasks. Programs create the display that the user sees in the browser window; they make the application interactive by accepting and processing information that the user types in the browser window and they store information in the database and get information out of the database. (The database is useless unless we can move data in and out.)

The Web pages can be *Static* or *Dynamic*. In HTML pages, all users see the same content. *Dynamic* Web pages, on the other hand, allow the user to interact with the Web page. Different users might see different Web pages. For instance, one user looking at a furniture store's online product catalog might choose to view information about the sofas, whereas another user might



choose to view information about coffee tables. To create dynamic Web pages, we must use another language in addition to HTML. One language widely used to make Web pages dynamic is JavaScript. JavaScript is useful for several purposes, such as mouse-over's (for example, to highlight a navigation button when the user moves the mouse pointer over it) or accepting and validating information that users type into a Web form. However, it's not useful for interacting with a database. We can't use JavaScript to move the information from the Web form into a database. PHP, however, is a language that is particularly well suited to interacting with databases. PHP can accept and validate the information that users type into a Web form and can also move the information into a database.

### **6.1.1 The Database**

The core of a Web database application is the *database*, which is the long term memory that stores information for the application. A database is an electronic file cabinet that stores information in an organized manner so that one can find it when he needs it. After all, storing information is pointless if we can't find it. A database can be small, with a simple structure — for example, a database containing the titles and authors' names of all the books that one owns. Or a database can be huge, with an extremely complex structure — such as the database that Google.com must have to hold all its information.

The information that we store in the database comes in many varieties. A company's online catalog requires a database to store information about all the company's products. A membership Web site requires a database to store information about members. An employment Web site requires a database (or perhaps two databases) to store information about job openings and information from résumés. The information that we plan to store could be similar to information that's stored by Web sites all over the Internet — or information that's unique to our application. Technically, the term *database* refers to the file or group of files that holds the actual data. The data is accessed by using a set of programs called a DBMS (Database Management System). Almost all DBMSs these days are RDBMSs (Relational Database Management Systems), in which data is organized and stored in a set of related tables.

### 6.1.2 *The application: Moving data in and out of the database*

For the database to be useful, we need to be able to move data into and out of it. Programs are our tools for this because they interact with the database to store and retrieve data. A program connects to the database and makes a request: "Take this data and store it in the specified location." Another program makes the request: "Find the specified data and give it to me." The application programs that interact with the database run when the user interacts with the Web page. For instance, when the user clicks the submit button after filling in a Web form, a program processes the information in the form and stores it in a database.

## 6.2 MySQL, My Database

MySQL is a fast, easy-to-use RDBMS used for databases on many Web sites. Speed was the developers' main focus from the beginning. In the interest of speed, they made the decision to offer fewer features than their major competitors (for instance, Oracle and Sybase). However, even though MySQL is less full featured than its commercial competitors, it has all the features needed by the large majority of database developers. It's easier to install and use than its commercial competitors, and the difference in price is strongly in MySQL's favor. MySQL is developed, marketed, and supported by MySQL AB, which is a Swedish company. The company licenses it two ways:

- **Open source software:** MySQL is available via the GNU GPL (General Public License) for no charge. Anyone who can meet the requirements of the GPL can use the software for free. If you're using MySQL as a database on a Web site (the subject of this book), you can use MySQL for free, even if you're making money with your Web site.
  
- **Commercial license:** MySQL is available with a commercial license for those who prefer it to the GPL. If a developer wants to use MySQL as part of a new software product and wants to sell the new product, rather than release it under the GPL, the developer needs to purchase a commercial license. The fee is very reasonable.

Finding technical support for MySQL is not a problem. One can join one of several e-mail discussion lists offered on the MySQL Web site at [www.mysql.com](http://www.mysql.com). one can even search the e-mail list archives, which contain a large knowledge base of MySQL questions and answers. If

users are more comfortable getting commercial support, MySQL AB offers technical support contracts — five support levels, ranging from direct e-mail support to phone support, at five price levels.

### 6.2.1 Advantages of MySQL

MySQL is a popular database with Web developers. Its speed and small size make it ideal for a Web site. Add to that the fact that it is open source, which means free. Here is a rundown of some of its advantages:

- ***It's fast.*** The main goal of the folks who developed MySQL was speed. Consequently, the software was designed from the beginning with speed in mind.
- ***It's inexpensive.*** MySQL is free under the open source GPL license, and the fee for a commercial license is very reasonable.
- ***It's easy to use.*** One can build and interact with a MySQL database by using a few simple statements in the SQL language, which is the standard language for communicating with RDBMSs.
- ***It can run on many operating systems.*** MySQL runs on a wide variety of operating systems — Windows, Linux, Mac OS, most varieties of UNIX (including Solaris, AIX, and DEC UNIX), FreeBSD, OS/2, Irix, and others.
- ***Technical support is widely available.*** A large base of users provides free support via mailing lists. The MySQL developers also participate in the e-mail lists. One can also purchase technical support from MySQL AB for a very small fee
- ***It's secure.*** MySQL's flexible system of authorization allows some or all database privileges (for example, the privilege to create a database or delete data) to specific users or groups of users. Passwords are encrypted.

- ***It supports large databases.*** MySQL handles databases up to 50 million rows or more. The default file size limit for a table is 4GB, but one can increase this (if the operating system can handle it) to a theoretical limit of 8 million terabytes (TB)
- ***It's customizable.*** The open source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

### **6.2.2 How MySQL works**

The MySQL software consists of the MySQL server, several utility programs that assist in the administration of MySQL databases, and some supporting software that the MySQL server needs. The heart of the system is the MySQL server.

The MySQL server is the manager of the database system. It handles all the database instructions. For instance, if we want to create a new database, we send a message to the MySQL server that says "create a new database and call it *newdata*." The MySQL server then creates a subdirectory in its data directory, names the new subdirectory *newdata*, and puts the necessary files with the required format into the *newdata* subdirectory. In the same manner, to add data to that database, we send a message to the MySQL server, giving it the data and telling it where we want the data to be added.

Before one can pass instructions to the MySQL server, it must be running and waiting for requests. The MySQL server is usually set up so that it starts when the computer starts and continues running all the time. This is the usual setup for a Web site. However, it's not necessary to set it up to start when the computer starts. If one needs to, it can be started manually whenever access to the database is required. When it's running, the MySQL server listens continuously for messages that are directed to it.

### **6.2.3 Communicating with the MySQL server**

All our interaction with the database is done by passing messages to the MySQL server. The PHP software has specific statements that we can use to send instructions to the MySQL server.

The MySQL server must be able to understand the instructions that is send to it. We communicate by using *SQL (Structured Query Language)*, which is a standard language understood by many RDBMSs. The MySQL server understands SQL. PHP doesn't understand SQL, but it doesn't need to: PHP just establishes a connection with the MySQL server and sends the SQL message over the connection. The MySQL server interprets the SQL message and follows the instructions. The MySQL server sends a return message, stating its status and what it did (or reporting an error if it was unable to understand or follow the instructions).

### **6.3 PHP, a Data Mover**

*PHP*, a scripting language designed specifically for use on the Web, is the tool for creating dynamic Web pages. Rich in features that make Web design and programming easier, PHP is in use on over 13 million domains (according to the Netcraft survey at [www.php.net/usage.php](http://www.php.net/usage.php)). Its popularity is growing, meaning that it must be fulfilling its function pretty well.

PHP stands for *PHP: HyperText Preprocessor*. In its early development by a guy named Rasmus Lerdorf, it was called *Personal Home Page tools*. When it developed into a full-blown language, the name was changed to be more in line with its expanded functionality.

The PHP language's syntax is similar to the syntax of C. PHP is actually simpler than C because it doesn't use some of the more difficult concepts of C. PHP also doesn't include the low-level programming capabilities of C because it is designed to program Web sites and doesn't require those capabilities.

PHP is particularly strong in its ability to interact with databases. PHP supports pretty much every database. PHP handles connecting to the database and communicating with it.

One does not need to know the technical details for connecting to a database or for exchanging messages with it. Just tell PHP the name of the database and where it is, and PHP handles the details. It connects to the database, passes the instructions to the database, and returns the database response to the user.

Technical support is available for PHP. User can join one of several e-mail discussion lists offered on the PHP Web site ([www.php.net](http://www.php.net)), including a list for *databases and PHP*. In addition, a Web interface to the discussion lists is available at [news.php.net](http://news.php.net), where he can browse or search the messages.

### 6.3.1 Advantages of PHP

The popularity of PHP is growing rapidly because of its many advantages:

- *It's fast.* Because it is embedded in HTML code, the response time is short.
- *It's inexpensive — free, in fact.* PHP is proof that free lunches do exist and that one can get more than that paid for.
- *It's easy to use.* PHP contains many special features and functions needed to create dynamic Web pages. The PHP language is designed to be included easily in an HTML file.
- *It can run on many operating systems.* It runs on a wide variety of operating systems — Windows, Linux, Mac OS, and most varieties of UNIX.
- *Technical support is widely available.* A large base of users provides free support via e-mail discussion lists.
- *It's secure.* The user does not see the PHP code.

- *It's designed to support databases.* PHP includes functionality designed to interact with specific databases. It relieves user of the need to know the technical details required to communicate with a database.
- *It's customizable.* The open source license allows programmers to modify the PHP software, adding or modifying features as needed to fit their own specific environments.

### 6.3.2 How PHP works

PHP is an embedded scripting language when used in Web pages. This means that PHP code is embedded in HTML code. We use HTML tags to enclose the PHP language that we embed in our HTML file — the same way that we would use other HTML tags. We can create and edit Web pages containing PHP the same way that we create and edit regular HTML pages.

The PHP software works in conjunction with the Web server. The Web server is the software that delivers Web pages to the world. When we type a URL into our Web browser, we are sending a message to the Web server at that URL, asking it to send us an HTML file. The Web server responds by sending the requested file. Our browser reads the HTML file and displays the Web page. We can also request the Web server to send us a file when we click a link in a Web page. In addition, the Web server processes a file when we click a Web page button that submits a form.

When PHP is installed, the Web server is configured to expect certain file extensions to contain PHP language statements. Often the extension is .php or .phtml, but any extension can be used. When the Web server gets a request for a file with the designated extension, it sends the HTML statements as-is, but PHP statements are processed by the PHP software before they're sent to the requester.

When PHP language statements are processed, only the output is sent by the Web server to the Web browser. The PHP language statements are not included in the output sent to the browser, so the PHP code is secure and transparent to the user. For instance, in this simple PHP statement:

```
<?php echo "<p>Hello World"; ?>
```

<?php is the PHP opening tag, and ?> is the closing tag. echo is a PHP instruction that tells PHP to output the upcoming text. The PHP software processes the PHP statement and outputs this:

```
<p>Hello World
```

which is a regular HTML statement. This HTML statement is delivered to the user's browser. The browser interprets the statement as HTML code and displays a Web page with one paragraph — Hello World. The PHP statement is not delivered to the browser, so the user never sees any PHP statements.

PHP and the Web server must work closely together. PHP is not integrated with all Web servers, but it does work with many of the most popular Web servers. PHP is developed as a project of the Apache Software Foundation — consequently, it works best with Apache. PHP also works with Microsoft IIS/PWS, iPlanet (formerly Netscape Enterprise Server), and others.

Although PHP works with several Web servers, it works best with Apache. Apache is a good choice. It is free, open source, stable, and popular. It currently powers over 60 percent of all Web sites, according to the Web server survey at [www.netcraft.com](http://www.netcraft.com). It runs on Windows, Linux, Mac OS, and most flavors of UNIX.

#### **6.4 MySQL and PHP, the Perfect Pair**

MySQL and PHP are frequently used together. They are often called the *dynamic duo*. MySQL provides the database part, and PHP provides the application part of your Web database application.



#### 6.4.1 Advantages of the relationship

MySQL and PHP as a pair have several advantages:

- *They're free.* It's hard to beat free for cost-effectiveness.
- *They're Web-oriented.* Both were designed specifically for use on Web sites. Both have a set of features that are focused on building dynamic Web sites.
- *They're easy to use.* Both were designed to get a Web site up quickly.
- *They're fast.* Both were designed with speed as a major goal. Together they provide one of the fastest ways to deliver dynamic Web pages to users.
- *They communicate well with one another.* PHP has built-in features for communicating with MySQL. Use does not need to know the technical details..
- *A wide base of support is available for both.* Both have large user bases. Because they are often used as a pair, they often have the same user base. Many people are available to help, including those on e-mail discussion lists who have experience using MySQL and PHP together.
- *They're customizable.* Both are open source, thus allowing programmers to modify the PHP and MySQL software to fit their own specific environments.

#### 6.4.2 How MySQL and PHP work together

PHP provides the application part, and MySQL provides the database part of a Web database application. We use the PHP language to write the programs that perform the application tasks. PHP is flexible enough to perform all the tasks that our application requires. It can be used for simple tasks (such as displaying a Web page) or for complicated tasks (such as accepting and verifying data that a user typed into an HTML form). One of the tasks that our application must

do is move data into and out of the database — and PHP has built-in features to use when writing programs that move data into and out of a MySQL database.

PHP statements are embedded in the HTML files with PHP tags. When the task to be performed by the application requires storing or retrieving data, we use specific PHP statements designed to interact with a MySQL database. We use one PHP statement to connect to the correct database, telling PHP where the database is located, its name, and the password needed to connect to it. The database doesn't need to be on the same machine as the Web site;

PHP can communicate with a database across a network. Another PHP statement is used to send instructions to MySQL. We send an SQL message across the connection, giving MySQL instructions for the task that we want done. MySQL returns a status message that shows whether it successfully performed the task. If there was a problem, it returns an error message. If our SQL message asked to retrieve some data, MySQL sends the data that we asked for, and PHP stores it in a temporary location where it is available to us.

We then use one or more PHP statements to complete the application task. For instance, we can use PHP statements to display data that we retrieved. Or we might use PHP statements to display a status message in the browser, informing the user that the data was saved.

As an RDBMS, MySQL can store very complex information. As a scripting language, PHP can perform very complicated manipulation of data, either data that we need to modify before saving it in the database or data that you retrieved from the database and need to modify before displaying or using it for another task. Together, PHP and MySQL can be used to build a Web database application that has a very sophisticated and complicated purpose.

### 6.4.3 The PHP MySQL Architecture

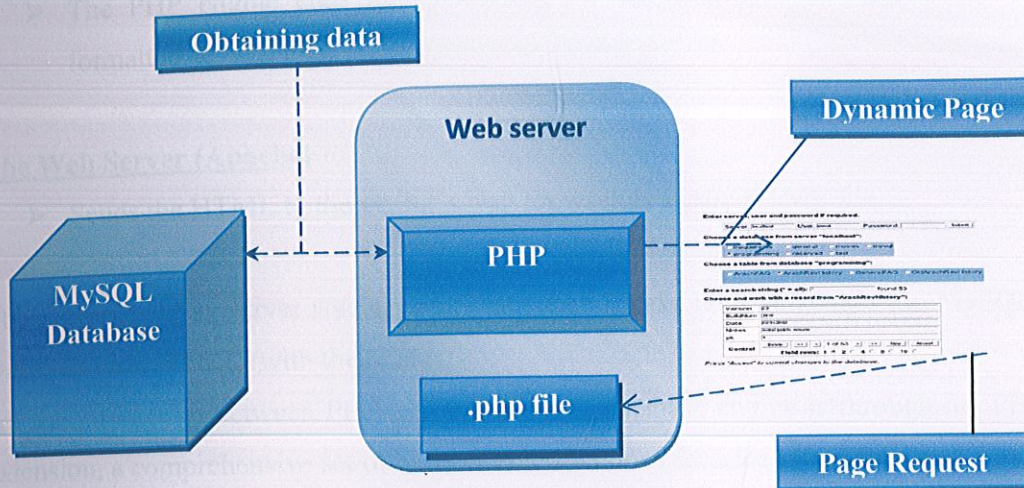


Figure 6.4.3

The Figure 6.4.3 depicts:

#### The Visitor's

- Web browser requests a web page (.php file) using a standard URL

#### The Web Server (Apache)

- Recognizes that the requested file is a PHP script
- The web server therefore interprets the file using its PHP engine.

#### PHP

- Processes the page request
- Fetches the data from the MySQL database

#### MySQL Database

- Responds by sending the requested content to the PHP script

## **PHP**

- The PHP script stores the content into one or more PHP variables
- The PHP engine ends up by spitting out the data fetched dynamically as a nicely formatted HTML page it has created to the web server.

## **The Web Server (Apache)**

- Sends the HTML to the visitor's web browser as a pure HTML file

This means PHP as server side scripting language speaks both languages i.e. MySQL (with the Database) and HTML (with the client).

All the interaction between PHP and the MySQL database engine is through the PHP-MySQL extension, a comprehensive set of built-in PHP functions for accessing MySQL.

## CHAPTER – 7

# AJAX

## AJAX

---

### **7.1 Introduction**

One of the problems with web applications is that typically every time a page gets data dynamically from the server, the data is displayed in a new page. The user has to wait for this new page to load before continuing. This can often take time, and users can get frustrated, and the experience is more disjointed than using a typical GUI application. The usability of such a web application is not good.

AJAX is Asynchronous JavaScript + XML, and is one solution to this problem. Like DHTML, LAMP or SPA, AJAX is not a technology in itself but a programming framework. It is a combination of several tried and tested technologies coming together in powerful ways. In an AJAX application, an HTML page makes asynchronous calls to the server using JavaScript and loads the data in bits and pieces as needed. The basis of AJAX is the *XMLHttpRequest*, which is a browser DOM object which sends an HTTP request to a server, and gets the response back as data only, as XML, rather than a whole HTML page.

#### **7.1.1 Where did it come from?**

While the term "Ajax" was coined in 2005, alternative techniques for the asynchronous loading of content date back to the mid 1990s. Java applets were introduced in the first version of the Java language in 1995. These allow compiled client-side code to load data asynchronously from the web server after a web page is loaded. In 1996, Internet Explorer introduced the I Frame element to HTML, which also enables this to be achieved. In 1999, Microsoft created the XMLHTTP ActiveX control in Internet Explorer 5. This is now supported by Mozilla, Safari and other browsers as the native XMLHttpRequest object. On April 5, 2006 the World Wide Web Consortium (W3C) released the first draft specification for the object in an attempt to create an official web standard.

It is now supported in most common web browsers. Though used by Microsoft it was popularized by Google in its stunning interface for Google maps and the excellent usability of Gmail.

## 7.2 What Makes AJAX Cool

Instead of having to send everything to the server in a single, huge mass, then wait for the server to send back a new page for rendering, web developers can communicate with the server in smaller chunks, and selectively update specific areas of the page based on the server's responses to those requests. This is where the word *asynchronous* in the AJAX acronym originated. It's probably easiest to understand the idea of an asynchronous system by considering its opposite—a synchronous system. Traditional web applications use a synchronous system, as shown in Figure 7.2.1

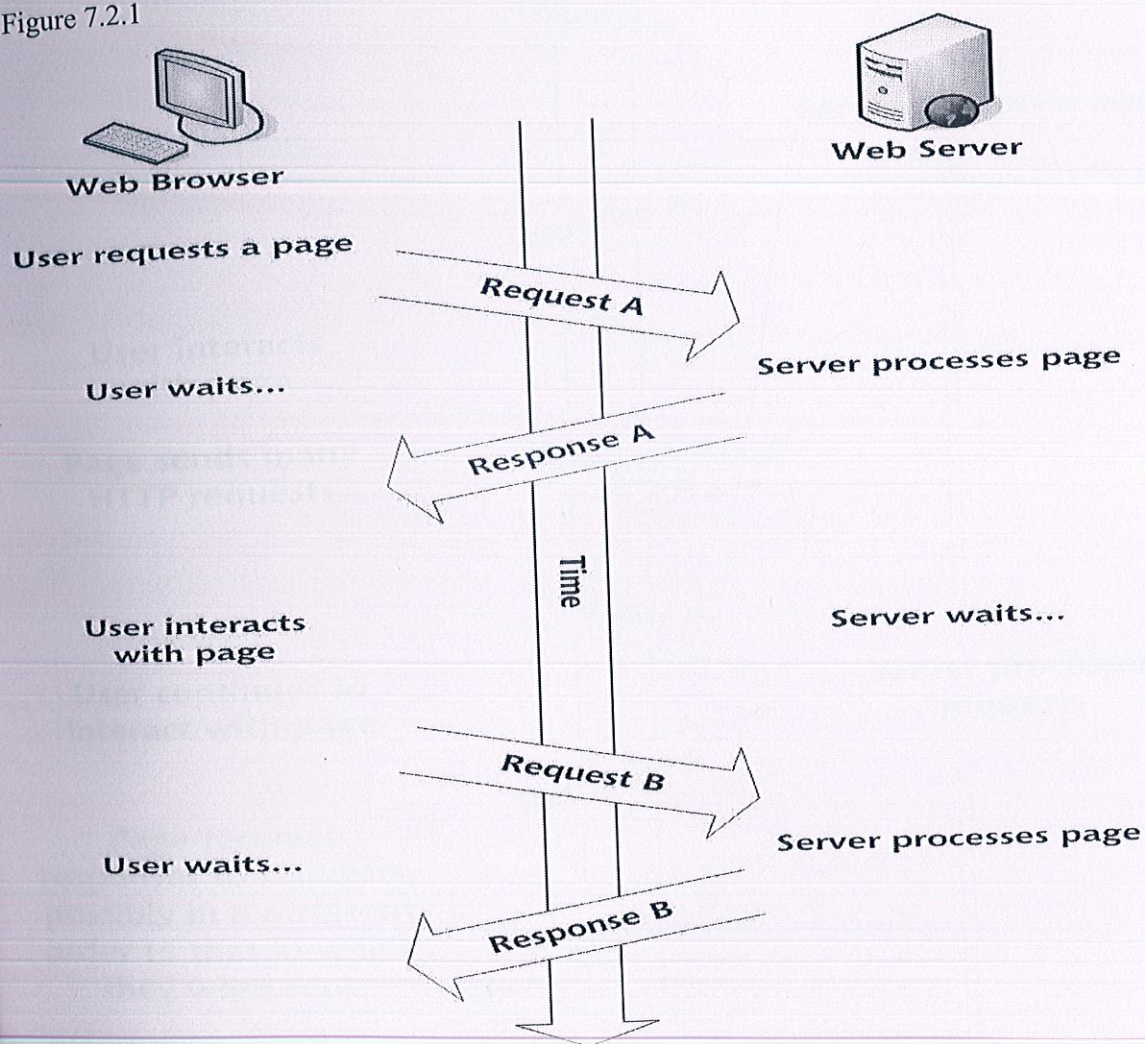


Fig 7.2.1: A traditional web app is a synchronous system

Figure 7.2.2 shows an AJAX application making asynchronous requests to a web server.

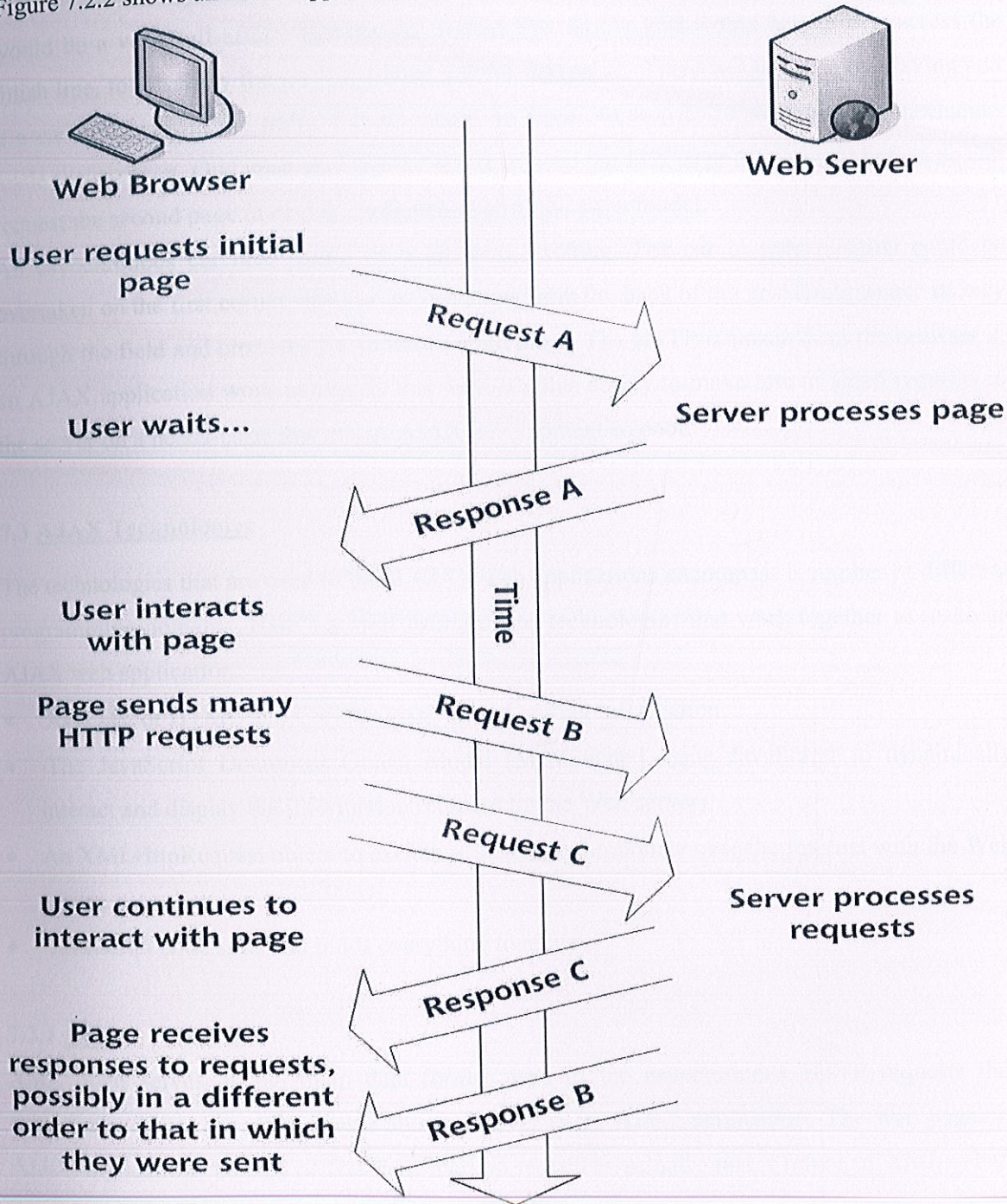


Fig 7.2.2



In a synchronous system, every thing occurs in order. If a car race was a synchronous system, it would be a very dull affair. The car that started first on the grid would be the first across the finish line, followed by the car that started second, and so on. There would be no overtaking and if a car broke down, the traffic behind would be forced to stop and wait while the mechanics made their repairs. One must wait for the server to send the first page of a system before he can request the second page in case of traditional web application model.

An asynchronous car race would be a lot more exciting. The car in pole position could be overtaken on the first corner, and the car that starts from the back of the grid could weave its way through the field and cross the finish line in third place. The HTTP requests from the browser in an AJAX application work in exactly this way. It's this ability to make lots of small requests to the server on a needs-basis that makes AJAX development so cool.

### **7.3 AJAX Technologies**

The technologies that are used to build AJAX web applications encompass a number of different programming domains. Here's a brief listing of the technologies that work together to make an AJAX web application:

- XHTML or HTML and CSS for mark up and styling information.
- The JavaScript Document Object Model (Manipulated using JavaScript to dynamically interact and display the information returned by the Web server)
- An XMLHttpRequest object to exchange data asynchronously over the Internet with the Web server
- JavaScript code spec that binds everything together

#### **7.3.1 XML**

XML often serves as the main data format used in the asynchronous HTTP requests that communicate between the browser and the server in an AJAX application. The web pages in AJAX applications consist of XHTML markup, which is actually just a flavor of XML. There are numerous advantages to using XHTML:

- It offers lots of standard tools and script libraries for viewing, editing, and validating XML.
- It's forward-compatible with newer, XML-compatible browsers.

- It works with either the HTML Document Object Model (DOM) or the XML DOM.
- It's more easily repurposed for viewing in non-browser agents.

### **7.3.2 Document Object Model**

The Document Object Model (DOM) is an object-oriented representation of XML and HTML documents, and provides an API for changing the content, structure, and style of those documents. The DOM represents the structure of an XML or HTML document as an object hierarchy, which is ideal for parsing by standard XML tools. JavaScript provides a large API for dealing with these DOM structures, in terms of both parsing and manipulating the document. This is one of the primary ways to accomplish the smaller, piece-by-piece changes to a web page that we see in an AJAX application.

### **7.3.3 XMLHttpRequest**

XMLHttpRequest, a JavaScript class with a very easy-to-use interface, sends and receives HTTP requests and responses to and from web servers. The XMLHttpRequest class is what makes true AJAX application development possible. The HTTP requests made with XMLHttpRequest work just as if the browser were making normal requests to load a page or submit a form, but without the user ever having to leave the currently loaded web page.

### **7.3.4 JavaScript**

JavaScript is the glue that holds the AJAX application together. It performs multiple roles in AJAX development:

- controlling HTTP requests that are made using XMLHttpRequest
- parsing the result that comes back from the server, using either DOM manipulation methods, XSLT, or custom methods, depending on the data exchange format used
- presenting the resulting data in the user interface, either by using DOM manipulation methods to insert content into the web page, by updating an element's inner HTML property, or by changing elements' CSS properties

## 7.4 The AJAX framework – An overview

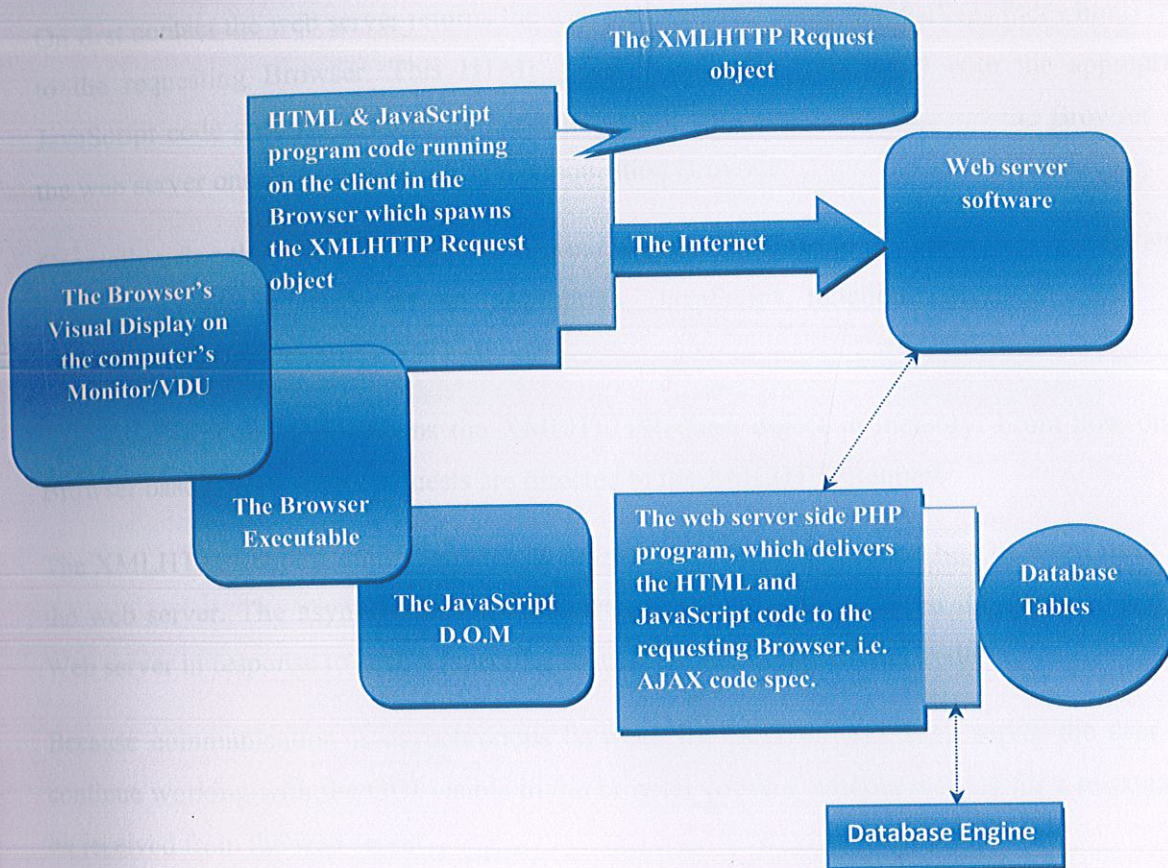


Fig 7.4: Initiating and sustaining Ajax based communications between a browser and a web server

### 7.5 AJAX based communication between a browser and a Web server

A browser makes its first call to a web server. This is when the user keys in a URL in the browser's address bar and clicks its *GO* button or presses the Enter key on the keyboard.

On first contact the web server returns the web site, gateway page, (i.e. perhaps index.html) back to the requesting Browser. This HTML page is delivered pre-loaded with the appropriate JavaScript code spec required to initiate Ajax based communication between the Browser and the web server once it begins to run in the requesting Browser

Generally using the <BODY> tag's Load event, which is mapped to the JavaScript onLoad event handler, a call is made to an appropriate JavaScript function described within the <HEAD></HEAD> tags of the HTML page.

This JavaScript function spawns the XMLHttpRequest object in memory. From now on all Browser based, Web server requests are directed to the XMLHttpRequest

The XMLHttpRequest object initiates an asynchronous communication link between itself and the web server. The asynchronous communication link is setup between the Browser and the Web server in response to such a (specific) request within the JavaScript code.

Because communication is asynchronous between the Browser and Web server the user can continue working with the GUI visible in the browser window without waiting for a response to be received from the web server.

When the XMLHttpRequest object sets up an asynchronous communication link between itself and the Web server a listener is also setup at the client Browser. This is standard behavior of the XMLHttpRequest object.

When the Web server completes processing the Browser's request (received via the XMLHttpRequest object spawned at the client browser) it returns the result of such processing back to the Browser.

The XMLHttpRequest listener, at the client's end, now responds appropriately to the Web server on receipt of such data.

The XMLHttpRequest object then invokes an appropriate client side, JavaScript function using one of its in-built methods and passes this function all the data returned by the Web server.

The JavaScript function's code spec processes the data returned and immediately updates the appropriate data array belonging to the JavaScript DOM.

The Browser, which keeps close tabs on the JavaScript DOM immediately, recognizes that data within the DOM arrays have been updated.

The Browser then immediately refreshes that section of the HTML page, (visible in the Browser window and displayed on the computer's VDU) which is bound to the array(s), which have been updated in its DOM.

This updation happens locally and is incredibly swift. So much so, that a user often does not notice even a moment delay in their user workflow experience.

This is the conceptual/actual framework on which all Ajax applications are based.

## **7.6 Advantages of AJAX**

The following are the advantages of Ajax:

### **➤ Continuous Feel**

Traditional web application force a user to submit a form back to the Web server, wait a few seconds, watch the page redraw and only then add some other information. If the area code in a phone number is not entered by mistake, then everything has to be entered all over again. This irritates a user, leading to an unhappy user work flow experience.

Ajax offers a smooth all the way. There are no major page reloads using Ajax, just the user, browser and Web server all doing their stuff in perfect harmony to ensure user delight.

➤ **Real Time Updates**

As part of the continuous user experience, Ajax applications can update HTML page sections in real time. Currently, new services on the web redraw the entire page at intervals, for example, once every 15 minutes. In contrast, it's feasible for a browser running an Ajax application to poll the server every few seconds. This ensure that browser is capable of updating any information directly in the parts of the HTML page that need changing almost in real time while the rest of the page remains unaffected.

➤ **Interactivity**

Ajax applications are mostly executed on the user's computer. They can perform a number of tasks without having their performance being hampered by Internet bandwidth. This permits the development of interactivity applications, (particularly reactive), with rich graphic user interfaces.

➤ **Portability Sample**

Ajax application is created using individual components that are well documented and implemented by all major browsers on most existing platforms. While it is uncertain that this compatibility will continue with the advent of the next generation of the browser such as Firefox2 and I.E.7, at the moment Ajax applications are effectively cross-platform.

### **7.7 Various Sites Using AJAX**

A number of commercial sites have currently switched to Ajax to improve their user work flow experience. These sites are more like web applications than the traditional brochureware sites that just display information as the user visits it to accomplish a specific goal.

Google is making a huge investment in developing the Ajax approach. All of the major products Google has introduced over the last year such as Orkut, Gmail, the latest beta version of Google Groups, Google Suggest and Google Maps are Ajax applications.



The following are some of the well-known sites that use Ajax:

➤ **Google Suggest**

Google Suggest (<http://www.google.com/webhp?complete=1&hl=en>) is the first example that comes into the mind when discussing Ajax. Google Suggest is a clone of the main Google site (<http://www.google.com>). As the words are typed in the text box, Google Suggest request suggestions from the server, showing a drop down list of search terms that may be of interest. Google Suggest is responsive beyond imagination. It updates itself no matter how fast the words are typed.

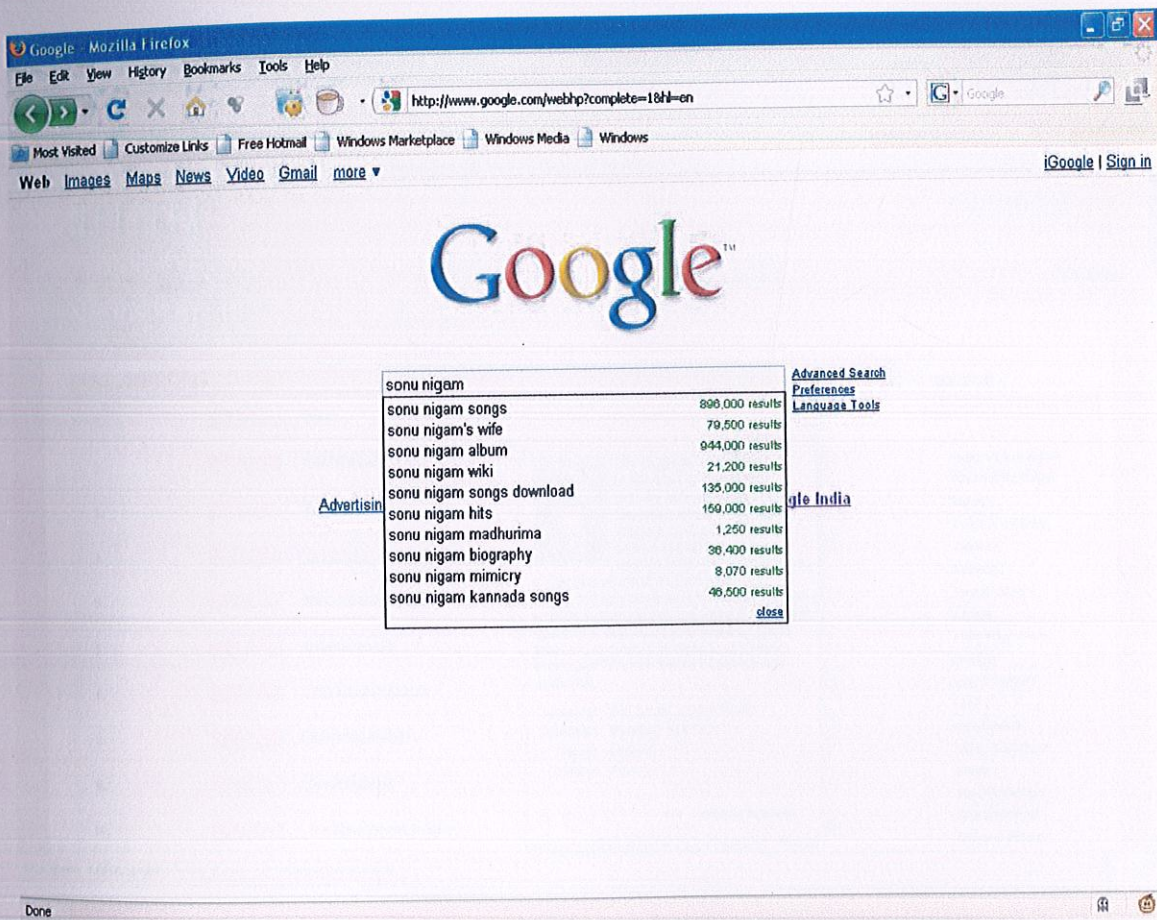


Figure 7.7.1: Google Suggest

➤ Netflix.com

Netflix top 100 is a very simple example of AJAX but it shows how even a simple idea like displaying a movie's details on mouse-over, which would be very impractical without AJAX, improves the usefulness of the page. For any given movie, displaying its description with a small image would not require much data to be pre-loaded onto the page, but x100 graphics text add up to a really long page download. AJAX makes this page more useful and interesting by pulling the data for each movie from the server and "thin streaming" it onto the page without pre-loading or re-loading.

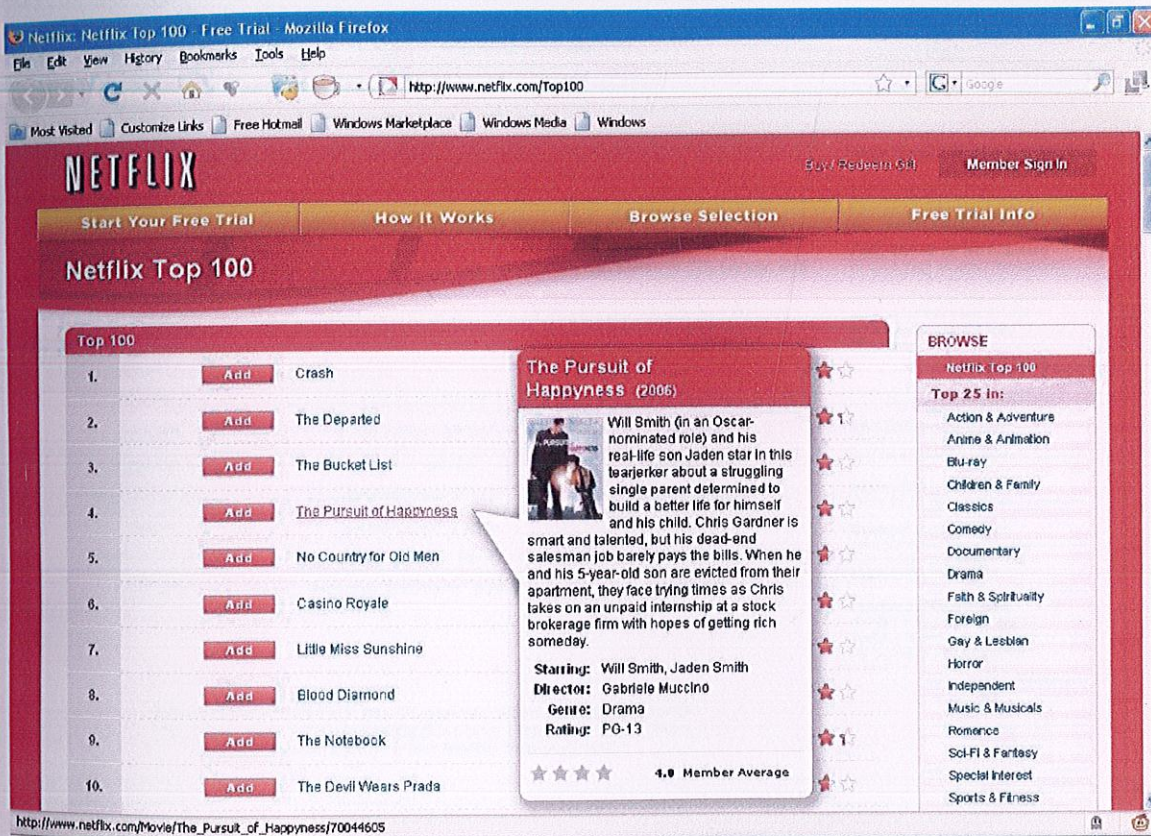


Figure 7.7.2: Netflix.com



➤ **Instant Domain Name Search**- This site is really helpful if you are trying to find a domain name for a new site. There are so few good names left, and it is extremely time consuming to enter an idea, click search, wait for the response, and try again. Instant Domain Name Search is similar to how Google suggest works. A domain's availability is dynamically determined as you type each letter. Try typing in "big..." and you'll see that big.com is, of course, taken. Continue typing "big toast" and you'll discover to your dismay that while bigtoa.com and bigtoas.com are indeed available, bigt.com, bigto.com, and bigtoast.com are not. Bigtoaster.com is also not available. But Bigtoasters.com is! Anyway, it sure makes finding variations and checking on the availability of potential domain names much easier.

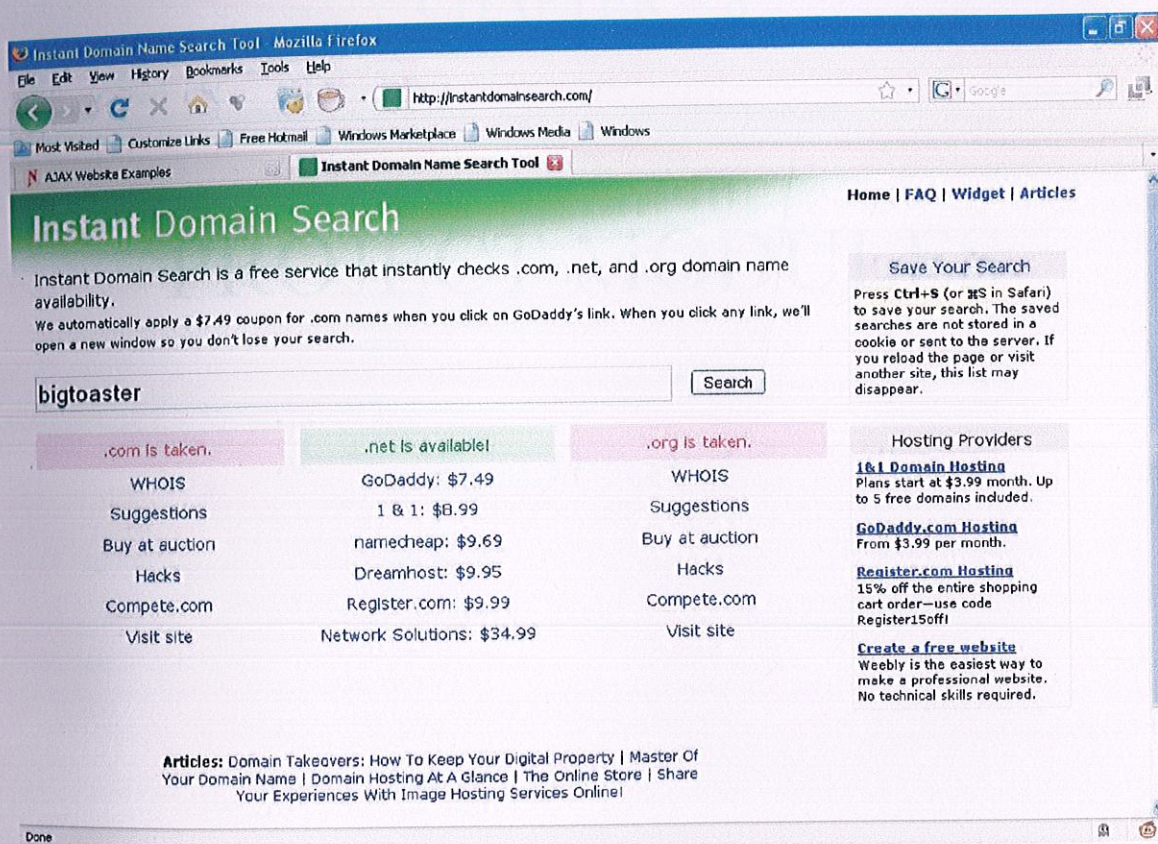


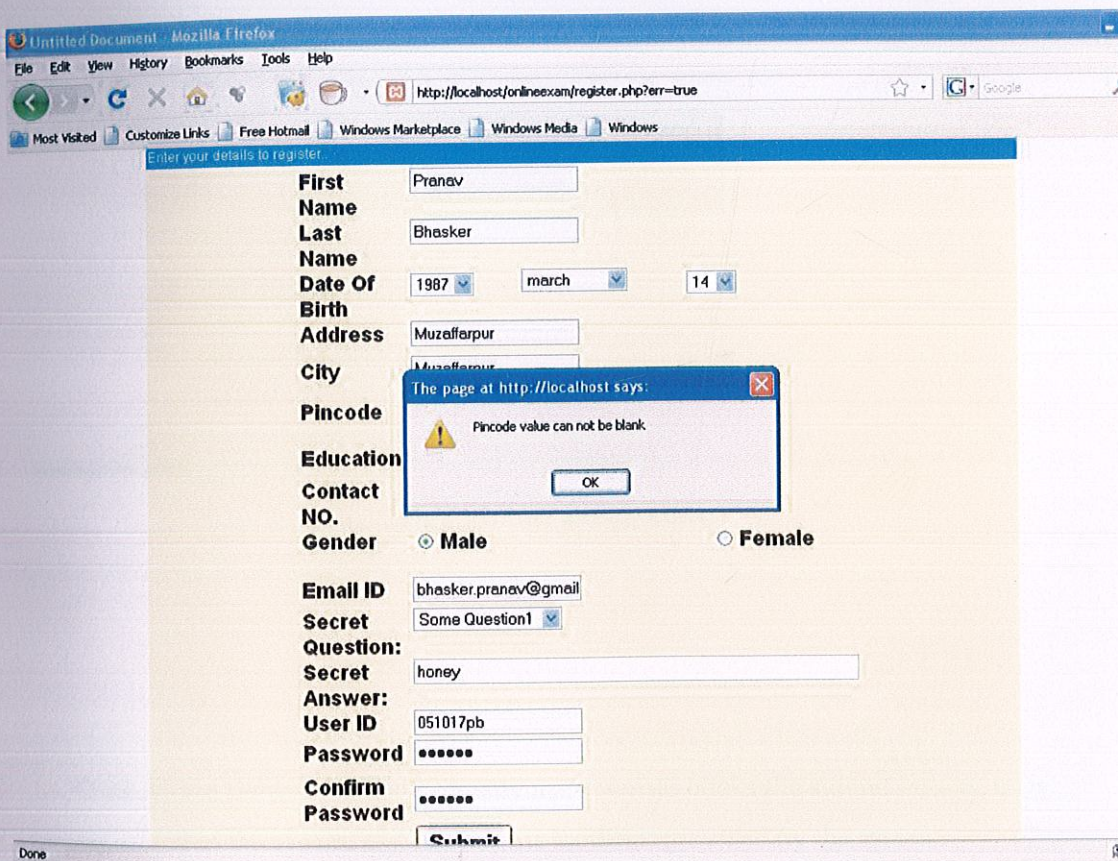
Fig 7.7.3: instantdomainsearch.com

CHAPTER – 8

PROJECT MODULES

## 8.1 User Registration

To enroll with the OHDMS one has to sign up the register form where user has to fill in the necessary details. User Id selected must be distinctive and chosen according to the requirement of the designer. The user id must contain roll number with initials. In case, any information field is left blank, a message will be displayed to fill that empty field.



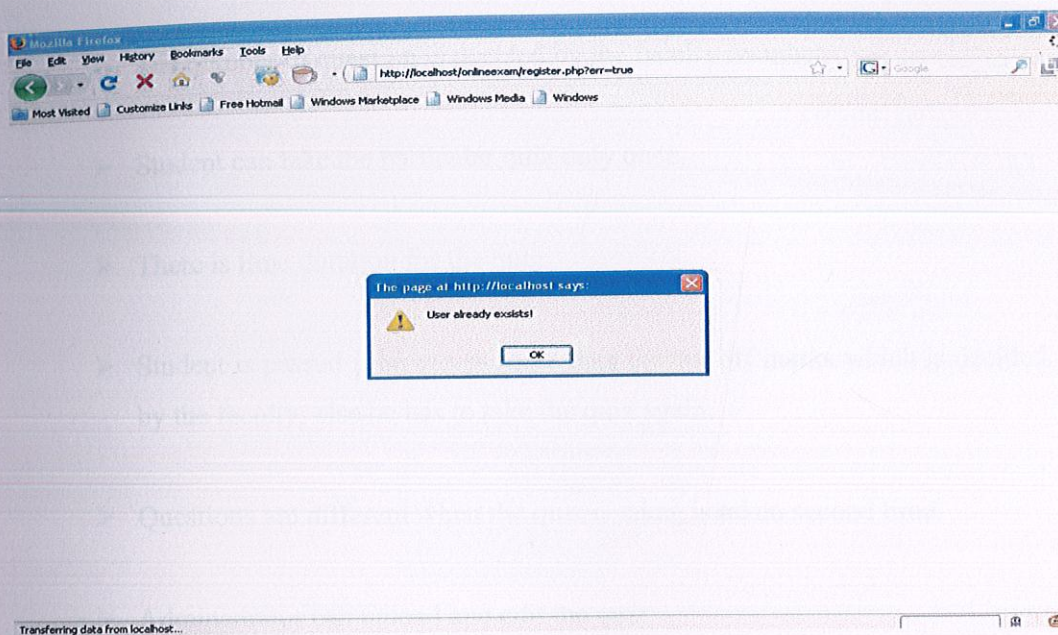
The screenshot shows a Mozilla Firefox browser window displaying a registration form titled "Enter your details to register". The form fields and their values are as follows:

First Name	Pranav
Last Name	Bhasker
Date Of Birth	1987   march   14
Address	Muzaffarpur
City	Muzaffarpur
Pincode	
Education	
Contact NO.	
Gender	<input checked="" type="radio"/> Male <input type="radio"/> Female
Email ID	bhasker.pranav@gmail
Secret Question:	Some Question1
Secret Answer:	honey
User ID	051017pb
Password	••••••
Confirm Password	••••••

A validation error message is displayed over the Pincode field, stating: "The page at http://localhost says: Pincode value can not be blank". The message box includes an "OK" button.

Fig 8.1.1: Field left empty

In case the user is already registered to the system, a message will be displayed showing that the user already exists.



**Fig 8.1.2: User already exists**

Once all the fields are filled correctly and the user clicks the submit button, he is registered to the system and can use the facilities provided by the system.

## 8.2 Online Quiz

---

This module is intended to do the evaluation of a student on the basis of his performance in the Online Quiz. The features covered in this module are as follows:

- Different departments can have their respective online quiz for the students.
- The number of question is decided by the faculty member.
- Student can take the particular quiz only once.
- There is time duration for the quiz.
- Student is passed if he scores more than the cut off marks which is decided by the faculty; else he has to take the quiz again.
- Questions are different when the quiz is taken is taken second time.
- Administrator can upload and edit the quiz.
- Result is displayed at the end of the quiz.

The administrator can set various parameters of the quiz. The parameters are:

- Total question per user
- Time(minutes)
- Cutoff marks
- Marks for correct answer
- Deduction for incorrect

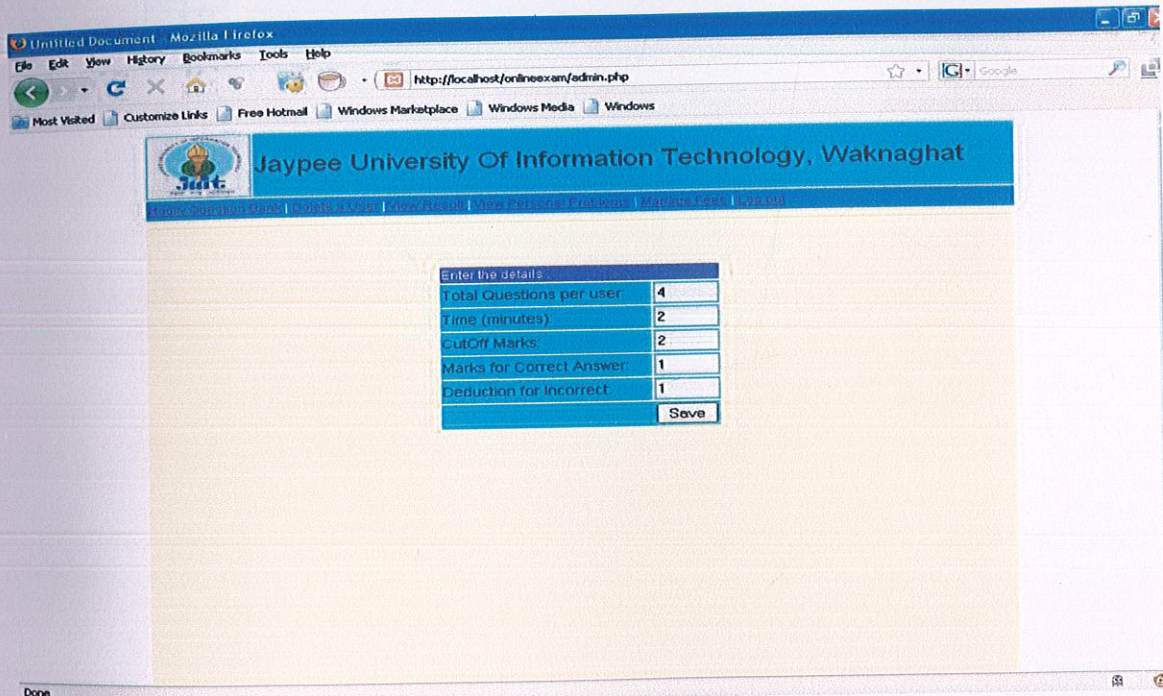


Fig 8.2.1: Setting the Parameters

Once all the fields are entered successfully and “Save” button is clicked, a dialogue box displaying “Entered Successfully” appears.

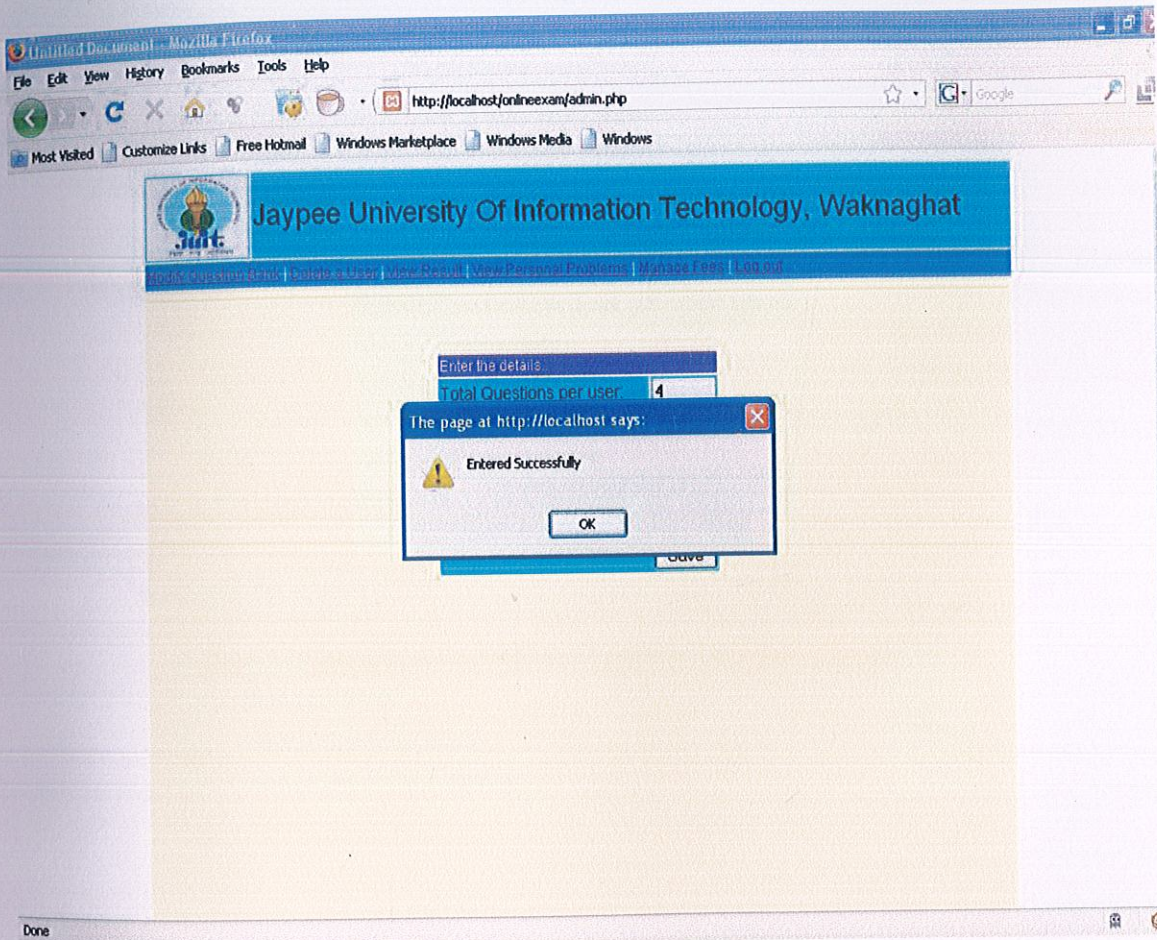


Fig 8.2.2

The administrator can modify the question bank in consultation with the faculty member.

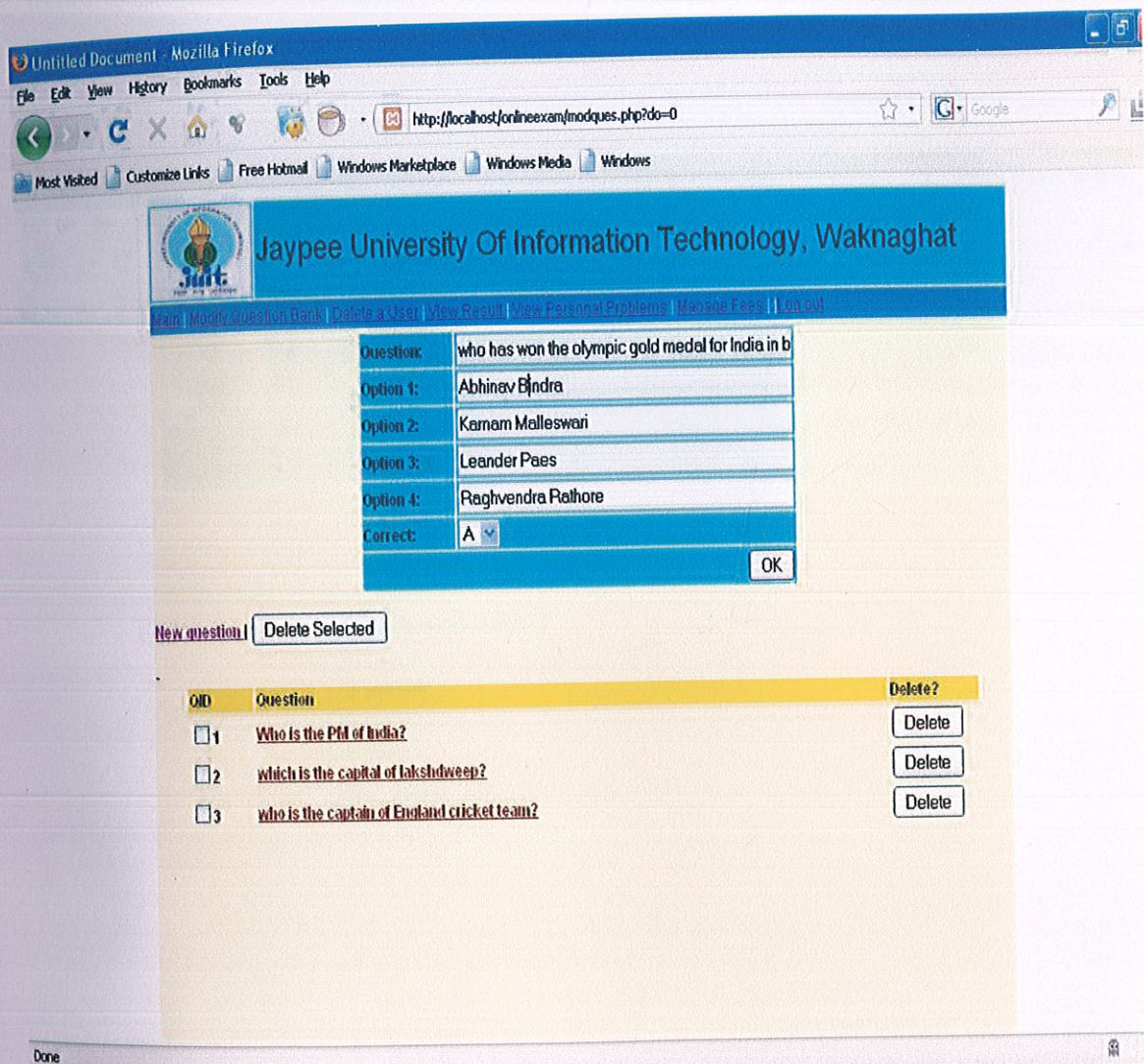


Fig 8.2.3: preparing question bank



The student has an option to go to the previous question or mark it for review. He can also change the answer before the time goes out.

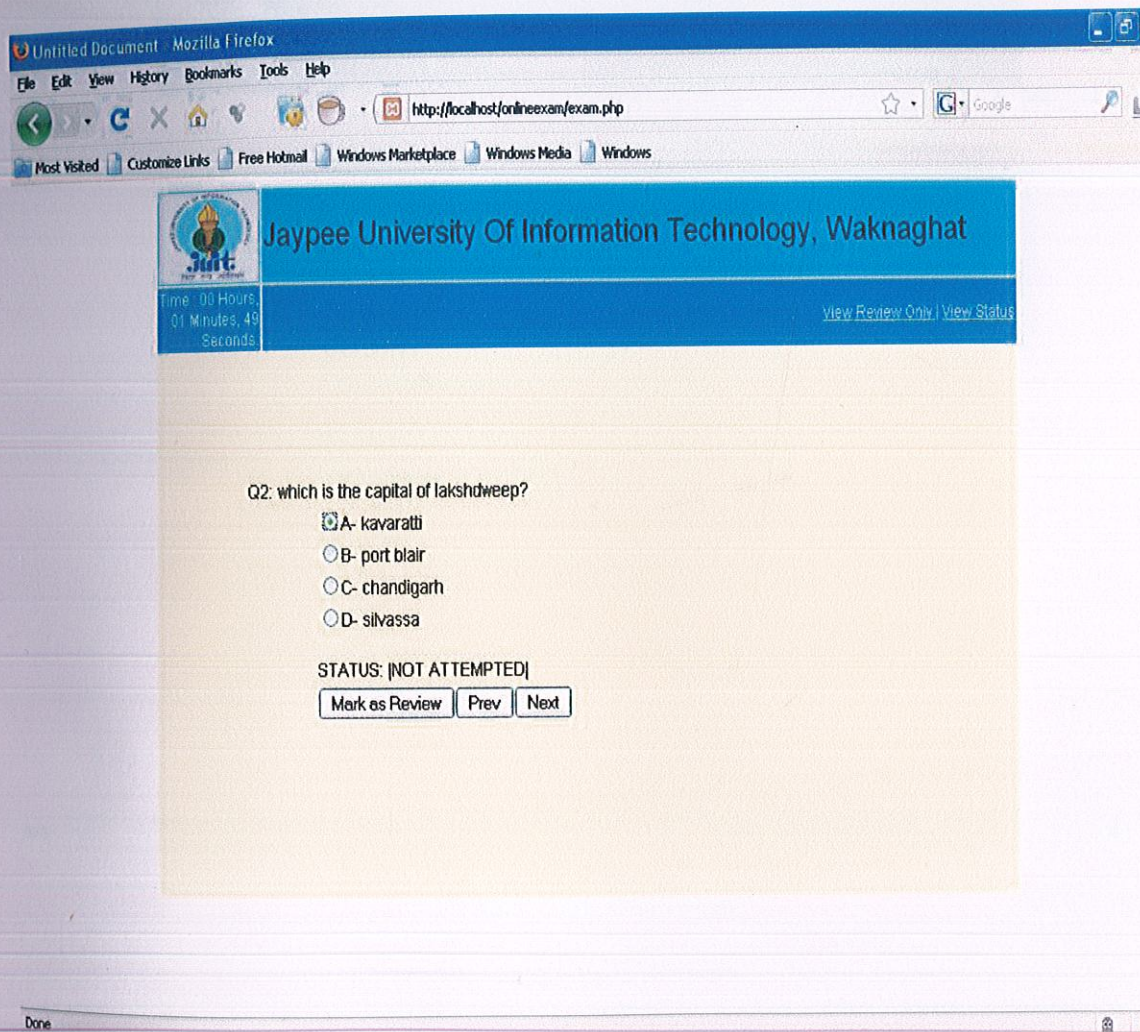


Fig 8.2.4

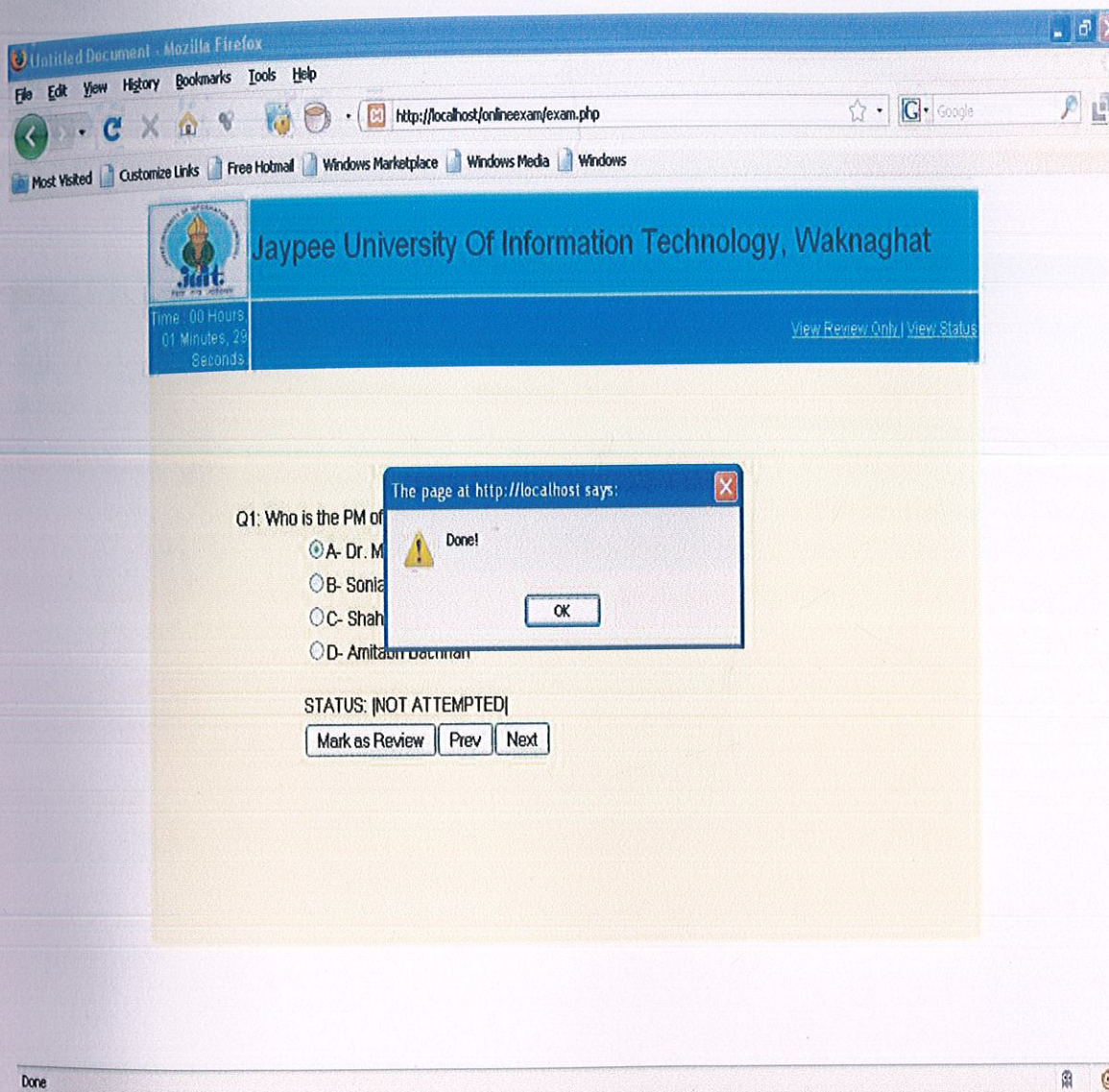


Fig 8.2.5: marked for review

When the student submits the answer of the penultimate question, a dialogue box appears indicating that the next question is the last question.

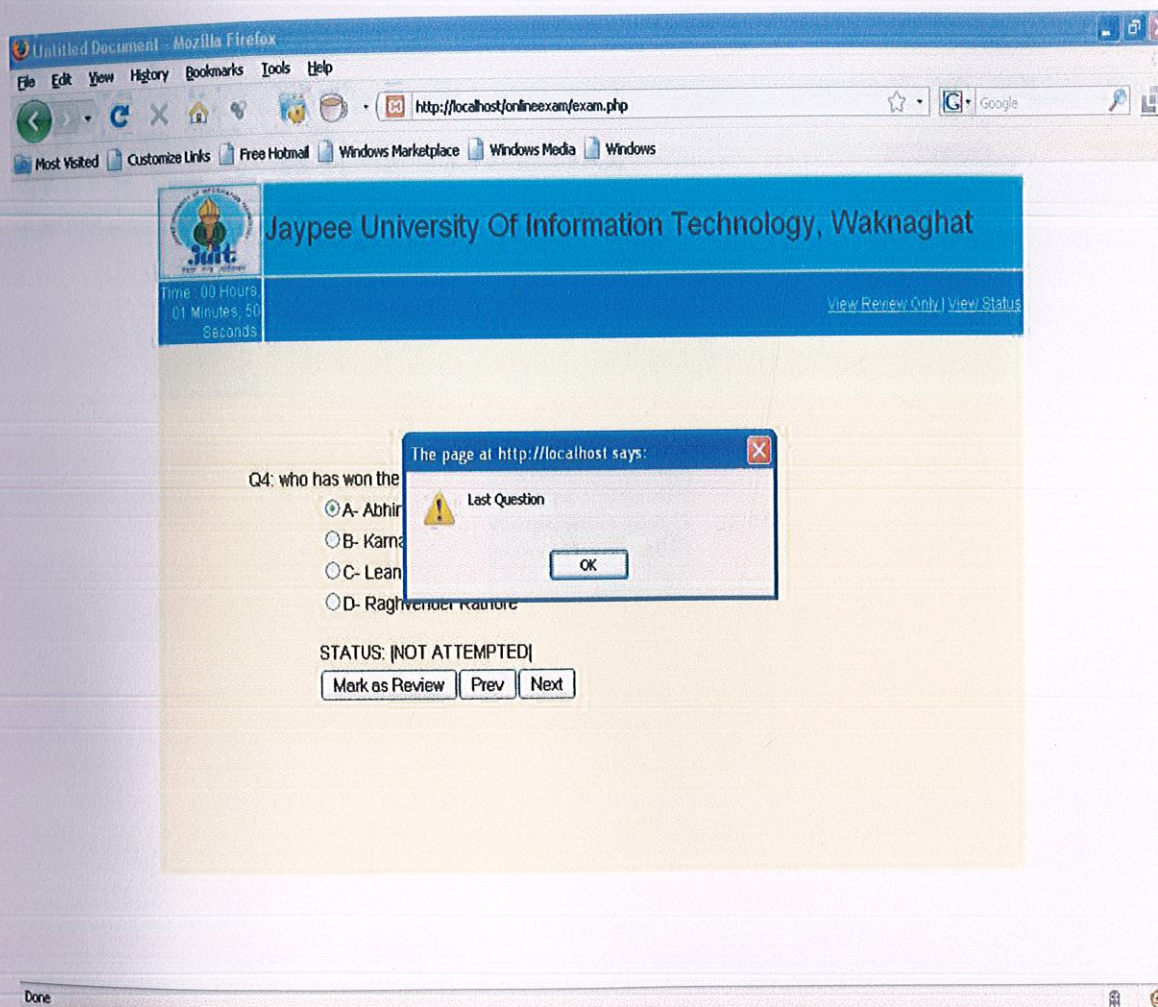


Fig 8.2.6

Once the quiz is finished, the total number of questions attempted or not attempted by the student is displayed.

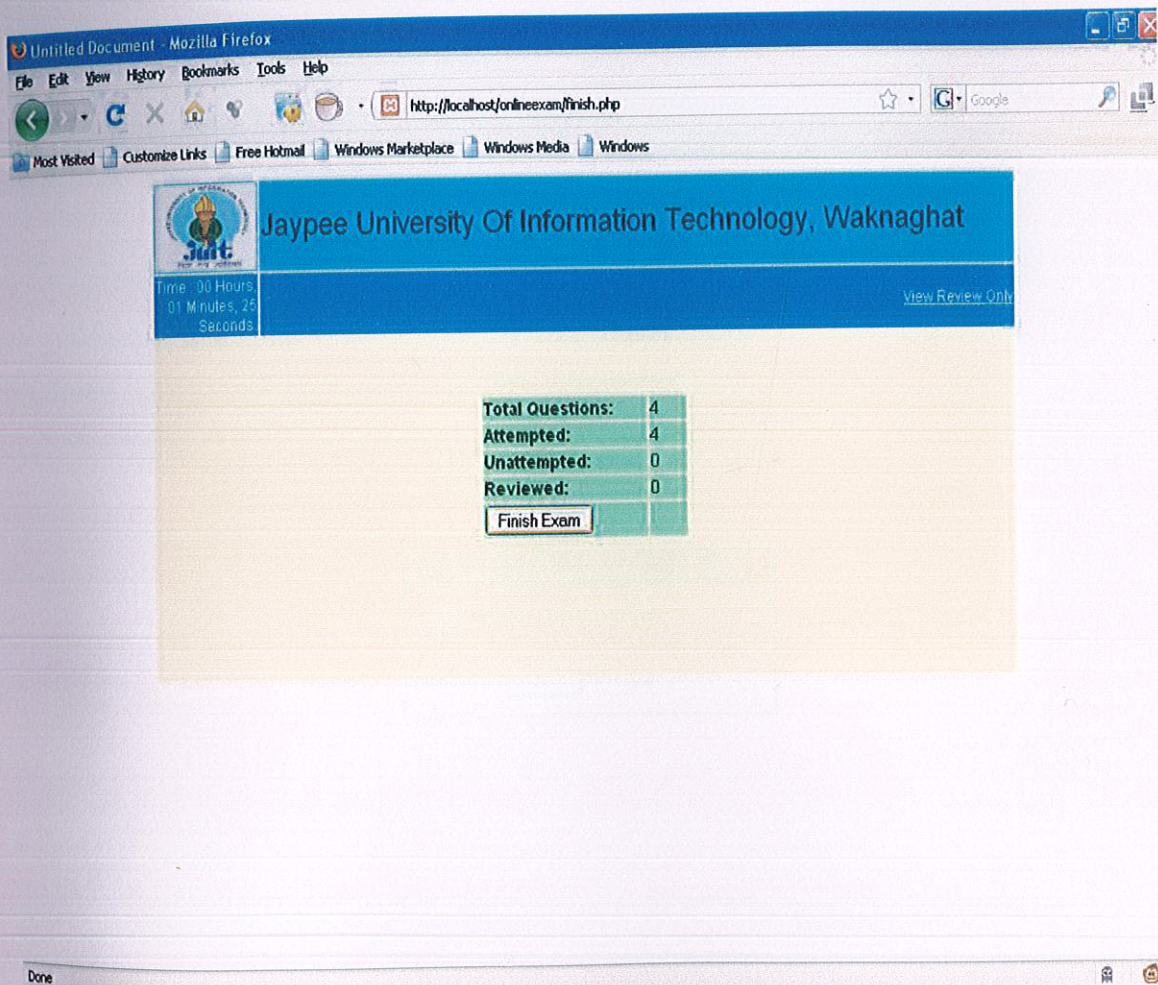


Fig 8.2.7

When the student clicks the "Finish Exam" button, a dialogue box appears asking whether he is sure to finish the exam

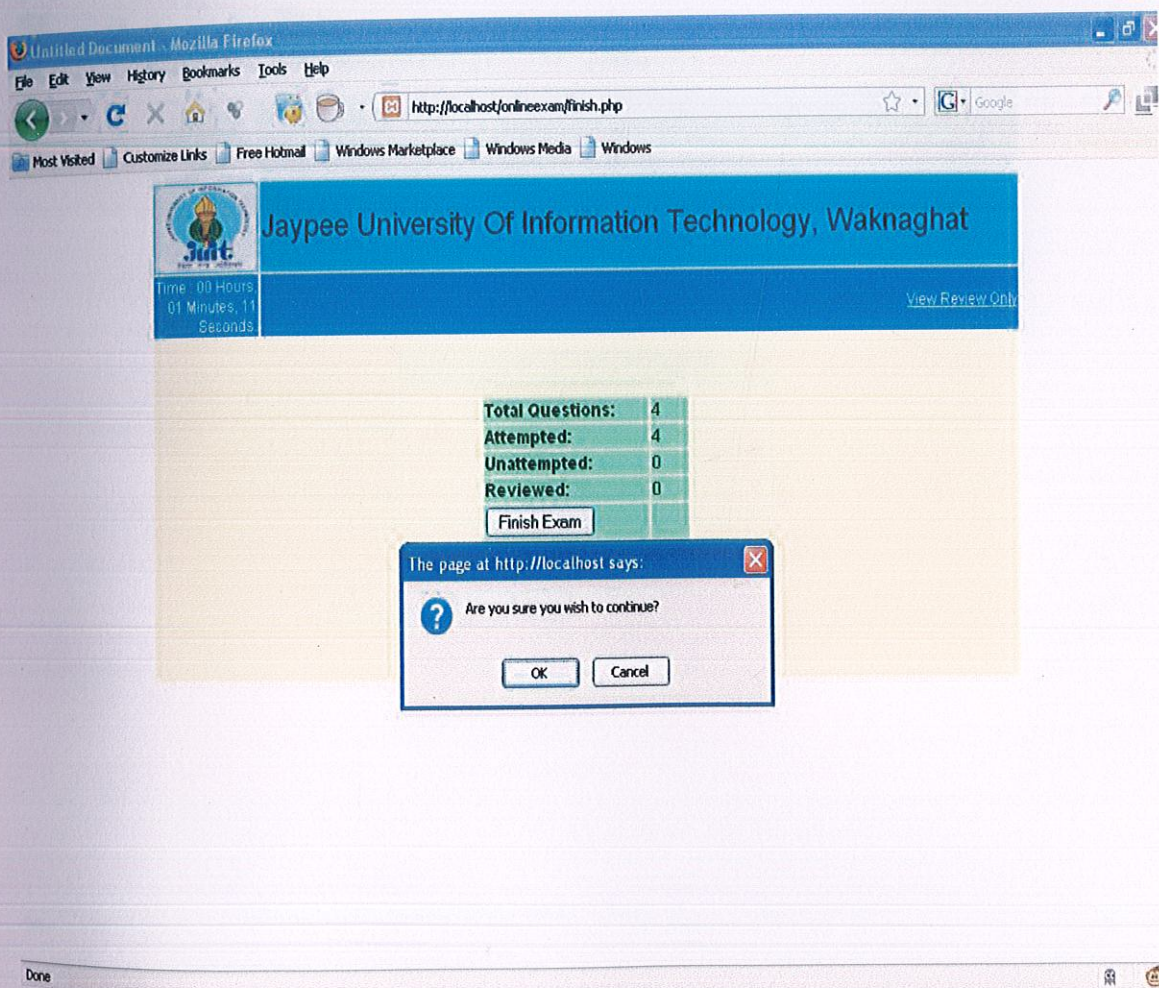


Fig 8.2.8

Once the time duration is over the quiz is terminated and the result is displayed.

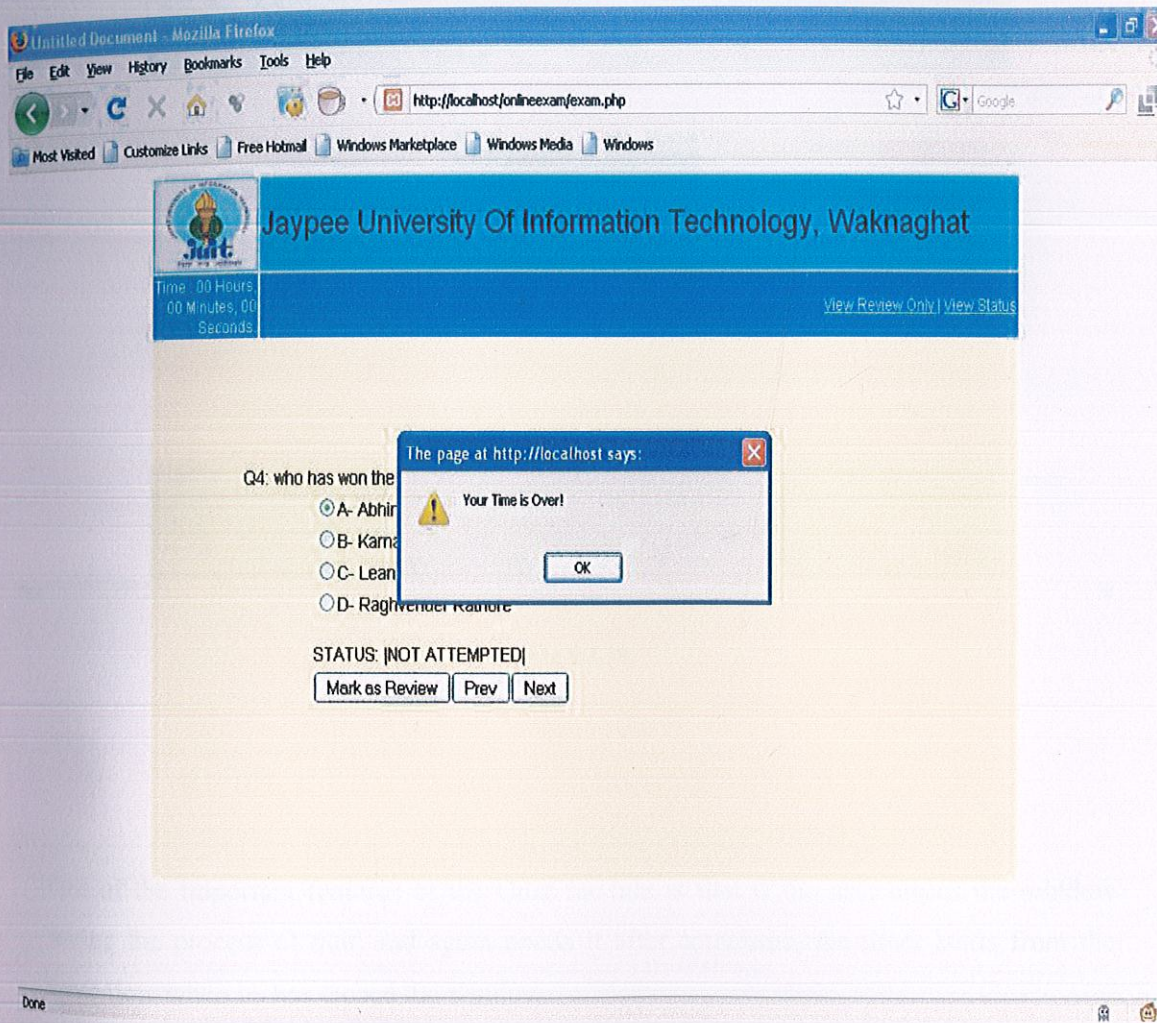


Fig 8.2.9

The result shows the number of correct and wrong answers marked by the student and the total marks obtained by him in the quiz.

The screenshot shows a Mozilla Firefox browser window displaying a quiz result page. The page header includes the university logo and name: "Jaypee University Of Information Technology, Waknaghat". Below the header, it says "QUIZ OVER" and "RESULTS". A table shows the following data:

Head	No.	Marks
Total Attempted:	4	-
Correct:	4	4
Incorrect:	0	-0
Total Marks:		4
Selected:	YES	
Log Out		
Retry Quiz		

Below the table, there is a "Result:" label and a question section with the following content:

Question	Status
1. Who is the PM of India? A: Dr. Manmohan Singh B: Sonia Gandhi C: Shahrukh Khan D: Amitabh Bachhan	

Fig 8.2.10

One of the important features of the Quiz module is that if the user closes the window during the process of quiz and again opens it after sometime, the timer starts from the same time when he has closed the window.

This module can be helpful in the evaluation of internal marks of a student.

### 8.3 Personal Problem Module

Through this module a registered user of the system can post his needs or problems to the administrator who can further forward the message to the concerned person in the university. The problems or needs can be personal or related to the academics or administration. For ex- suppose a student requires a bonafide certificate from the university for the purpose of making passport. He can make request to the administrator regarding this subject by sitting in his room in the hostel. He needs not to wait for the office hours and make that request. The administrator will forward the message to the concerned department through OSIES and respond back to the student about the status of

Student making a request through OSIES.

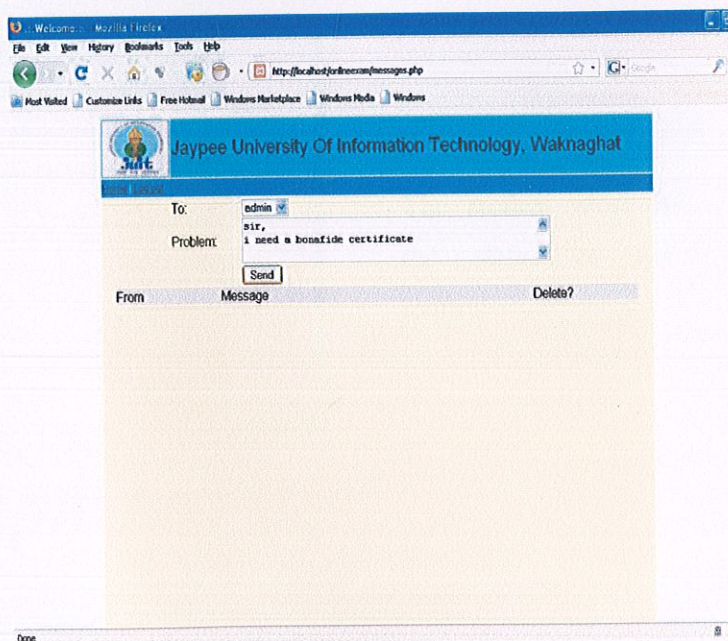


Fig 8.3.1



The administrator opens the message and forwards it to the concerned person/department

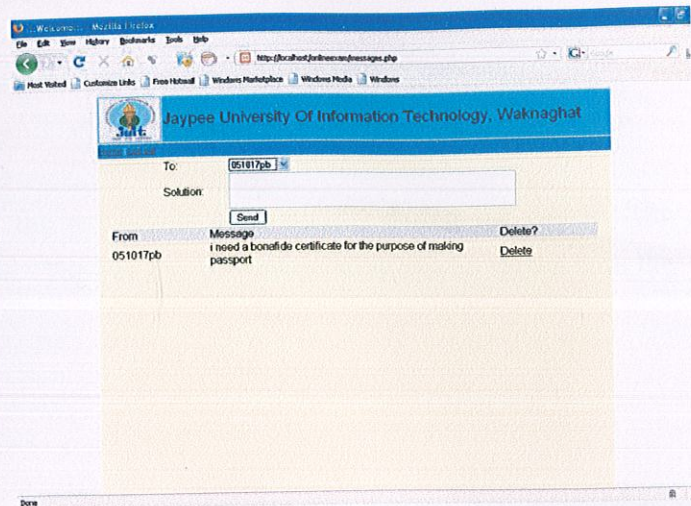
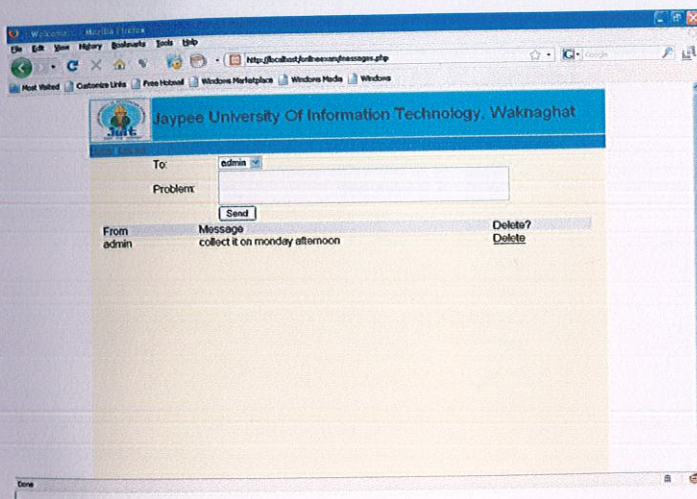


Fig 8.3.2



The student receives the status of his request

Fig 8.3.3

This module automates the workflow of services in the university campus and saves lot of time and energy.

## 8.4 Fee Management Module

This module helps in managing the fee details of a student during the course. The administrator manages this module. The Fee deposited/remaining can be shown with the help of this module.

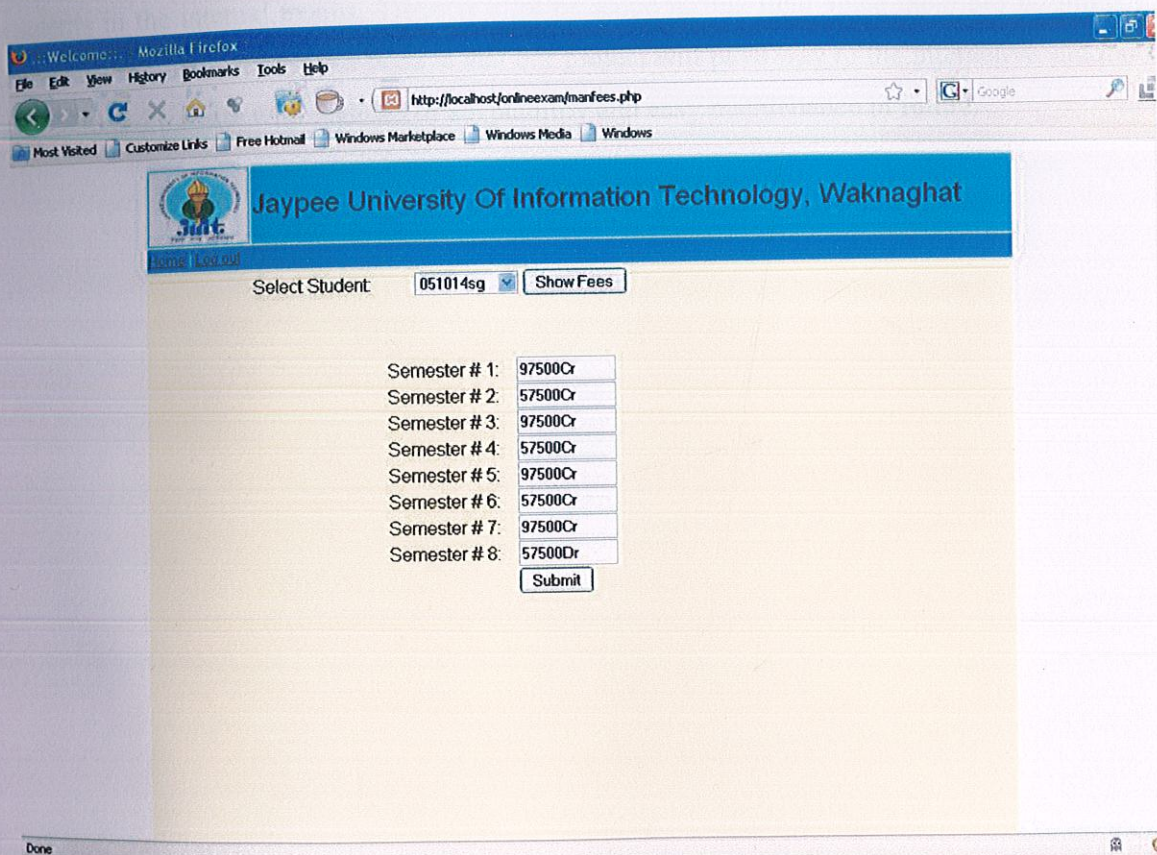


Fig 8.4.1: Fee Management-I

## CONCLUSION

The key aim of the original project was to introduce such a system in the JUIT campus that could benefit both the student and the faculty members. The basic motive was to automate the workflow of service requests for the various facilities in the campus so as to save time as well as energy. After studying various concepts regarding database driven web application i.e. client-server, database, scripting languages etc., we were finally able to develop the Online Student Interaction and Evaluation System (OSIES) for the JUIT Campus. This project not only automates the service request but also provides the method to evaluate the performance of students in the internal exams. There is a lot of scope in this field since there are no limits to creativity and innovations. We believe that our project will pave way to the interactive and more sophisticated website with features of adaptability and easy maintenance in future.

## BIBLIOGRAPHY

- <http://www.php.net/>
- <http://www.apache.org/>
- <http://www.mysql.com/>
- <http://www.wikipedia.org/>
- <http://www.w3schools.com/>
- <http://www.ajax.org/>
- <http://en.wikipedia.org/wiki/HTML/>
- <http://jcflowers1.iweb.bsu.edu/mod/rloadvantages.htm/>
- [http://www.inventu.com/guide/G\\_HtmlAdv.html/](http://www.inventu.com/guide/G_HtmlAdv.html/)
- <http://www.tizag.com/sqlTutorial/>
- <http://www.phpbuddy.com/>
- <http://en.wikipedia.org/wiki/Client-server/>
- <http://en.wikipedia.org/wiki/XAMPP/>
- [http://www.webdevelopersnotes.com/basics/client\\_server\\_architecture/](http://www.webdevelopersnotes.com/basics/client_server_architecture/)
- Leon Atkinson, *Core PHP Programming*, Pearson Education publication
- Davis & Philips, *Learning PHP and MySQL*, O'REILLY publication
- Sharanam Shah and Ivan Bayross, *PHP5.1 for beginners*
- Larry Ullman, *PHP and MySQL For Dynamic Web Sites*
- Janet Valede, *PHP and MySQL for Dummies*
- Vikram Vaswani, *MySQL: The complete Reference*

- Sharanam Shah and Ivan Bayross , *Ajax for Beginners*
- Thomas Powell, *HTML: The complete Reference*
- Eric Ray, *HTML 4 for Dummies Quick Reference*
- Wolf & Webster, *Web Designer's Guide to Dynamic HTML*
- Brad Halstead and Murray R. Summers, *Dreamweaver MX Templates*
- Bob Regan, *Dynamic Dreamweaver MX*

