"Web Scalability "

Project Report

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Engineering

by

Patil Amit Suresh Archana (12CO53)

Mirsinge Ibad Ibrahim Saba(12CO42)

Tulve Shabab Kasim Shagufta(12CO62)

Siddique Asma Abdul Wahab Zaibunnisa (12CO14)

Supervisor

Prof. P.S. Lokhande

Co-Supervisor

Prof. Salman Samshi

Department of Computer Engineering,

School of Engineering and Technology

Anjuman-I-Islam's Kalsekar Technical Campus

Plot No. 2 3, Sector -16, Near Thana Naka, Khanda Gaon,

New Panvel, Navi Mumbai. 410206

Academic Year : 2015-2016

CERTIFICATE

Department of Computer Engineering,
School of Engineering and Technology,
Anjuman-I-Islam's Kalsekar Technical Campus
Khanda Gaon,New Panvel, Navi Mumbai. 410206

This is to certify that the project entitled ***Web Scalability*** is a bonafide work of **Patil Amit Suresh Archana (12CO53),Mirsinge Ibad Ibrahim Saba (12CO42) , Tulve Shabab Kasim Shagufta (12CO62) ,Siddique Asma Abdul Wahab Zaibunnisa (12CO14) .** Submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **Bachelor of Engineering** in **Department of Computer Engineering**.

Prof. P.S Lokhande

Supervisor/Guide

Prof.Salman Samshi

Co-Supervisor/Guide

Prof. Tabrez Khan

Head of Department

Dr. Abdul Razak Honnutagi

Director

Project Approval for Bachelor of Engineering

This project entitled ***Web Scalability*** by ***Patil Amit Suresh Archana(12CO53) ,Mirsinge Ibad Ibrahim Saba(12CO42) ,Tulve Shabab Kasim Shagufta(12CO62) ,Siddique Asma Abdul Wahab Zaibunnisa(12CO14)*** is approved for the degree of ***Bachelor of Engineering in Department of Computer Engineering.***

Examiners

1. ....
2. ....

Supervisors

1. ....
2. ....

Chairman

....

Declaration

We declare that this written submission represents my ideas in my own words and where others ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Patil Amit Suresh Archana(12CO53).

Mirsinge Ibad Ibrahim Saba(12CO42).

Tulve Shabab Kasim Shagufta(12CO62).

Siddique Asma Abdul Wahab Zaibunnisa(12CO14).

Abstract

Tittle: Web Scalability

Web Scalability is the phenomenon when the incoming user requests on a server increases such that it exceeds the server capability to handle them and the system is able to somehow cope up with the increasing load. Since all servers can serve limited users if limit exceeds the server will either slow down or will crash. In today world all businesses are dependent on internet, as business grows the number of users accessing the web also increases which eventually grows the load on servers. Our project provides a mechanism for the businesses carried over internet to handle the icreasing amount of workload. The system provides an easy and handy solution to manage the user requests by mapping them on available server with the help of load balancing.

Patil Amit Suresh Archana (12CO53)

Mirsinge Ibad Ibrahim Saba (12CO42)

Tulve Shabab Kasim Shagufta(12CO62)

Siddique Asma Abdul Wahab Zainbunnisa (12CO14)

B.E. (Computer Engineering)
University of Mumbai.

Contents

List of Figures

List of Tables

Keywords And Glossary

Keywords :

Scaling, Web Server Scaling, Caching, Load Balancer, Application level Scaling, Scalability.

Glossary :

A

- **Attribute :**
  Column of the table which describes property of the table.

- **Application level Scaling :**
  Data aggregation is a type of data and information mining process where data is searched, gathered and presented in a report-base.

C

- **Caching :**A comparison predicate uses arithmetic operators to compare column data to a literal value.

G

- **GUI:** Graphical User Interface,is a type of interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces, typed command labels or text navigation.

L

- **Load Balancer:**Load balancing is a technique aiming at distributing workload in a computer network, in order to optimally utilize resources, avoid overload and maximize throughput.

M

- **MySQL:**SQL language.

P

- **Php:**Hypertext Preprocessor ,is a server-side scripting language designed for web development but also used as a general-purpose programming language.

S

- **Scalability:** The ease with which a system or component can be modified to fit the problem area.

- **SQL:** Structured Query Language is a special-purpose programming language designed for managing data held in a database management system .

- **Scaling:** he ease with which a system or component can be modified to fit the problem area.

Chapter 1

Introduction

1.1 Statement of Project

Web scalability is a technique which is used to scale the servers. There is a capacity of every server if limit exceeds the user wont get the requested site hence it is required to scale the server in such way that it will handle more number of users. Over last decade shopping sites have grown and so the users and it is needed to have the scaled system and hence we improved the system by scaling it by using load balancing and caching techniques.There are too many shopping sites present and they do not have a scaled system hence they cannot maintain the load on the servers.
The term Web scalability means without increasing the hardware of the servers and without increasing the number of the server we must handle the more request without degrading the response time.The available solutions are vertical scaling and horizontal scaling, vertical scaling is increasing number of servers and horizontal scaling is increasing the hardware of the server.[8]
We have designed the architecture which neither uses more number of servers nor hardware of server is improvised, we are using the load balancing and caching techniques which will handle the load more than current architecture.Scaling actually deals with limited resources you need to meet the peak requirements.we get errors many times saying that webpage not found its because the server has reached its limit and it wont serve further.
In our proposed system such errors will not happen since we have the load balancing and caching technique which will distribute the load such that there will not be such errors.Techniques used load balancing and caching.

1.2 Motivation

Now a days shopping is done hugely on internet. With gaining popularity of online shopping, the users accessing the ecommerce web application is also increasing. This growth in the number of users affects the efficiency of a system. Sometimes it is possible that server may not respond to each and every user because the incoming request may exceed the servers capacity to handle them. Every server possesses a limit to serve the user requests if this limit exceeds it cannot serve the requests, this may cause a great loss to a business website. Users cannot wait for long time so the response time should be as low as possible. If the response time is more then the user might terminate the purchase of product from the shopping website. Even on social networking sites there are huge number of users requesting the same web page at a same time so the servers may not serve all users it may either slow down or it wont serve each and every user. Hence it is neccessary to build a system that is scalable enough to overcome this issue. This motivated us to built a scalable server.

1.2.1 Advantage Over Current System

Current system has too may drawbacks such as high response time,serving limited users etc. Our proposed system is able to handle more no. of request than a current system and have less response time. Since in todays world where most of the buying and selling is done through website it is neccessary to have a scaled system. The current system is not scaled proporely our system is properly scaled and hence can handle more load than current system and also have the minimum response time than current system.Current system architecture has many drawbacks and hence we have the improvised system architecture.Our architecture consist enhanced load balancing and caching technique which is not present in current system.

1.2.2 Project Architecture

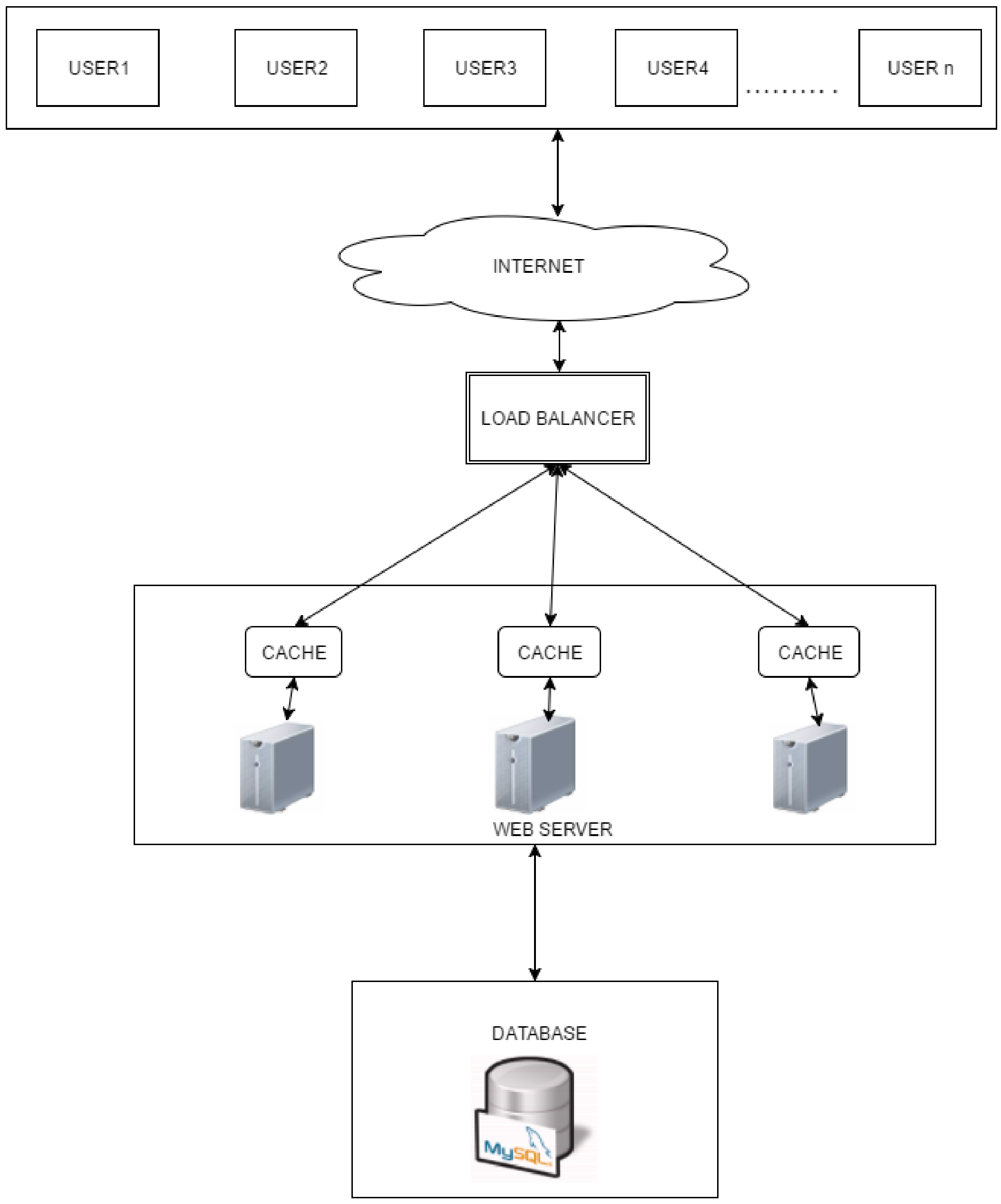


Figure 1.1: System Architecture

The System Architecture consists of three modules:

- Load Balancer.
- Web Server.
- Database Server.

Load Balancer: Computer clusters rely on load balancing to distribute workload across network links, CPUs, web servers, etc. A server farm is a common application of load balancing, where multiple servers seamlessly provide a single Internet service. In this case the load balancer accepts requests from external clients and forwards them to one of the available backend servers according to a scheduling algorithm (e.g. round robin, random choice, on a reported load basis, etc.) .Load balancers can be implmented using dedicated hardware or ad-hoc software.

Web Server: Main work of web server is process HTTP or any other protocol request. The primary function of a web server is to store, process and deliver web pages to clients. The communication between client and server takes place using the Hypertext Transfer Protocol (HTTP)[wiki]. In this Architecture the web server requests for web pages to the application server which are stored in database. The request to the web server is forwarded bt the load balancer, since load balancer calculates algorithms according to geographical areas, web page will also demand the page from application server according to the need.

Database Server: The database is of course stateful, after very definition and function of a database is to store application state. In fact, the state (application data) stored in a database must be a permanent record of actions performed by end users, otherwise the organization the application servers, or the application itself cannot function.

1.3 Objective and Scope

1.3.1 Objective

The objective of the proposed system is to scale a web server on application level that is to use minimum hardware and scale the requests on server side. The proposed system would have the capability to handle clients request more than its original limit. The main objective of our proposed system is that the server should serve more and more request without degrading the response time with minimum use of hardware and making the system scalable.The scalable system has the potential to serve the more number of request than the conventional system.To utilize the available resources in such a wa that the system is completely scaled and have advantages over a current system.

1.3.2 Scope

Proposed system is domain independent therefore can be used in any application. Where there is need to store the data and retrieving of data is done by non technical person or naive user, this system is useful.

- **Server will be capable of handling more concurrent user then conventional server :**The system can be used for many purpose in School and College .There is a need of storage of student,teacher,cleark data .The detail about student fees, result, document etc this data is usefull to student, teacher as well as to the administration department . Thus this data need to be stored in data base and native user find difficult to interact with the SQl software thus the system can be used in School and Colleges.

- **Caching gives Boost to handle more concurrent user:** In Hospital the patient, medicane are the important attribute there is the need that detail of this attribute need to be maintained so that one can get the information as required per their need this data are important for the user the day when the doctor are availabe the detail of the medicane present in the hosptial ,the day when the operation is to bee done are the important document for the user as well hosptial staff thus this data need to be stored in to the database thus interaction with this data form database need SQl command thus creating a system that can easily used to communicate with normal user in normal language that is English so that help to communicate with the system with ease.

- **Web security at client side of cookies:**The system can be used in Railway ,Air reservation purpose the detail of the user who have booked thier ticket the day the trail and flight will leave the place and the day it will reach the destation . The number of the customer which are in the wating list so on this detail can be maintained using the system and normal user can easily check for the detail like wise the system can bee used for below purpose.

Chapter 2

Literature Review

2.1 Scaling Web Sites Through Caching.

A powerful concept for reducing the delay caused in processing request and saving bandwidth is caching web resources. Here the resources refer to web pages. We find caches at Web browsers, organization proxy server caches, Internet service providers, content delivery networks (CDNs), and Web servers. A cache at the server side is mainly used to reduce the time required for processing a request[4]. Server side caching is the act of caching data on the server. Data can be cached anywhere and at any point on the server that makes sense. It is common to cache commonly used data from the database to prevent hitting the database every time the data is required. We cache the results from competition scores since the operation is expensive in terms of both processor and database usage. It is common to cache pages or page fragments so that they don't need to be generated for every visitor.

A cache-hit ratio is the number of times the database found something in cache divided by the number of times it looked for some object in the cache. The higher the ratio, the more effective the cache is at improving performance[4]. Taking into account the cache hit ratios, the frequency of reference f to Web documents is inversely proportional to the rank r, which is measured in terms of the document popularity. The most popular document has r = 1, the second most popular has r = 2, and so on. This relationship, called Zipf law, states that:

$f = k/r$ . . . where k is a constant.

With the advancing growth in technology solutions are available which helps solve the problem in smarter way. Use of cache is one of the way to scale such web applications at cheap cost. With Web site caching, Web content accelerates because documents likely to be requested are maintained in the cache[4]. This approach requires all incoming requests to pass through the cache first. Consequently, the cache server must have the capacity to handle all incoming traffic plus the cache update requests caused by cache misses.[4]

2.1.1 Pros

- Faster access to valid cache resources.
- Saving on costly use of bandwidth.
- Providing cached resources even when origin server is down.

2.1.2 Cons

- Its hard to maintain the cache data if changes are made in any of the data we are supposed to change the content from each and every cache server.
- If most of the request will be served by cache server the webserver will not be utilized at it fullest and it will be the wastage of money.
- Complex architecture and hence not implementable on every system.
- Too much of hardware is utilized.

2.1.3 How to Overcome

- Simplified architecture which can be implementable on every sytem.
- Hardware should be minimised

2.2 Web Scaling Frameworks for Web Services in the Cloud.

The concept of Web Scaling Frameworks (WSFs) in order to offload scaling to another layer of abstraction. this models improves the scaling ability of the current system.Web Application Frameworks (WAFs) focus on the creation of application logic and do not offer integrated cloud scaling concepts.[5]

WSFs take over the responsibilities of scaling by embedding existing WAFs in a larger system. The prototype they proposed in this work uses more components and a different composition than the normal version . It implements a Scaled Application Version that uses a WSF in combination with a WAF.[5]

2.2.1 Pros

Web scaling framework can triple the request throughput performance of single mchine.

Caching is highly used and hence maximised the scalability and performance of system.

2.2.2 Cons

– Complex architecture and hence not implementable on every system.

– Too much of hardware is utilized.

2.2.3 How to Overcome

– Simplified architecture which can be implementable on every sytem.

– Hardware should be minimised.

2.3 Scalability of Web-Based Electronic Commerce Systems

In a Web-based electronic commerce system, users browse product information offered by an online store and submit requests to purchase selected items. From the userâ$^{\mathrm{TM}}$s perspective, response time is a factor that could impact the acceptability of electronic commerce. This article is concerned with system architectures for online stores. Emphasis is placed on techniques to improve a systemâ$^{\mathrm{TM}}$s capacity to support more users without suffering a noticeable degradation in response time performance[3]. Such techniques have been investigated as part of a Canadian Institute for Telecommunications Research major project entitled Enabling Technology for Electronic Commerce Applications, developed in close collaboration with the IBMÂ® Centre for Advanced Studies[3]. The basic architecture of a Web-based electronic commerce system is first described along with the types of data that need to be managed. Next, an overview of existing techniques for improving system capacity is presented. Finally, highlights of research results obtained as part of the CITR electronic commerce major project are discussed.

A typical Web-based electronic commerce system has a three-tier architecture: the Web server, electronic commerce application server, and database system. The Web server is a process that handles requests from users and returns the requested Web pages[3]. The application server contains the business logic and accesses the database for information

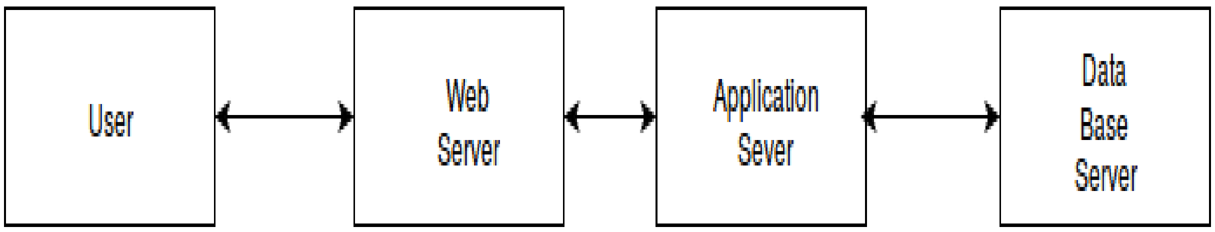


Figure 2.1: Basic system architecture.

such as catalogs, inventory level, and user information, such as registration data and shopping cart content. The three components may reside on the same machine or on different machines.

2.3.1 Pros

- Use of multiple server cluster means more availability of data.
- With mirror sites, information requested by users is made available at multiple server sites; each site has its own copy of the database.

2.3.2 Cons

- Each cluster has its own database, any updates to the catalogs must be made to all the databases.
- Within a server cluster, poor server node selection may lead to some nodes being saturated while other nodes have surplus capacity.

2.3.3 How to Overcome

- An appropriate selection algorithm should be used for selecting the best server node.
- Updation of database should be handled carefully

Chapter 3

Requirement Analysis

3.1 Platform Requirement :

3.1.1 Supportive Operating Systems :

The supported Operating Systems for client include:

- Linux Operating System (Ubuntu Server os 14.04 / 14.10 Utopic Unicorn) our.

Linux is the operating systems that will support comparative website. Since Linux is an open source operating system, The project we implemented is developed on Linux platform. The comparative website is tested on the same Linux OS.

Ubuntu 14.10 is used as server operating system. Apache 2.4 is used as Web server and for database server Mysql server is used.

3.2 Software Requirement :

The Software Requirements in this project include:

- Web Stress Tool

- Internet Explorer, Mozila Fire Fox, Google Chrome, etc.

For testing the website as well as implemented architecture Web Stress Tool is used. This tool provides a Ramp test on server by taking input as no of users and the complete url of the website including login sessions.

3.2.1 Back End Software Requirement

- VM ware Virtual Machine.
- Html.
- Css.
- Php.
- Mysql.
- Java Script.

Web Server Db server is the backbone of the entire project. Main purpose is to design and implement proposed architecture, VM ware is used to virtually distribute the hardware of system into different machine, by distributing the machine we gain throughput over the initial system.

Html, Css, Php is used to create the website for testing purpose of the architecture implemented. An E-Commerce website is built named Jewellery store. This website consists of all the necessary modules that basically exists on an e-commerce site.

In order to implement the same architecture an Administrator Gui is built using php language.

3.3 Hardware Requirement :

3.3.1 Hardware Required For Project Development:

- 2.5 GB Ram.
- 250 GB Hard Disk Minimum.
- Quad Core CPU.

Chapter 4

Project Design

4.1 Design Approach

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization. Once the software requirements have been analyzed and specified the software design involves three technical activities design, coding, implementation and testing that are required to build and verify the software. The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system. Design is the only way to accurately translate the customer requirements into finished software or a system. Design is the place where quality is fostered in development. Software design is a process through which requirements are translated into a representation of software. Software design is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data.

4.2 Software Architectural Designs

Our system follows the three tier architecture . First tier consist of GUI, Linguistic component and the Database.

1. User: A user sends a http request to access a particular website. The request is send to the website through the internet.

2. Internet: The internet receives the incoming user request and based on the url the user has provided, the internet forwards the request to that particular website.

3. Load Balancer: The load balancer residing at the website,to ensure load balancing forwards the user request to one of the available servers. Load balancer aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource.

4.Servers: The servers at the website to which the request is forwarded, answers the arriving user requests.

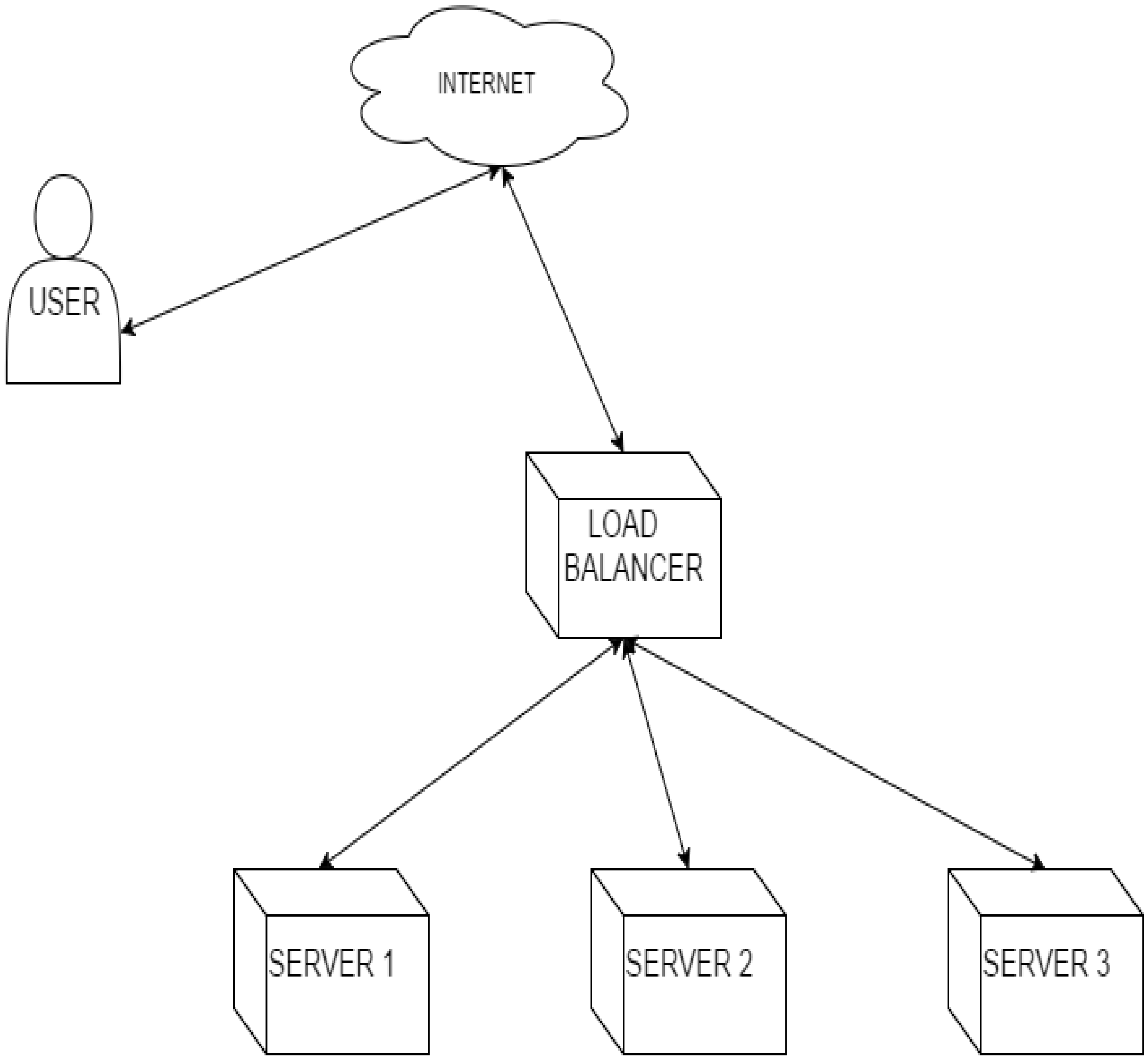
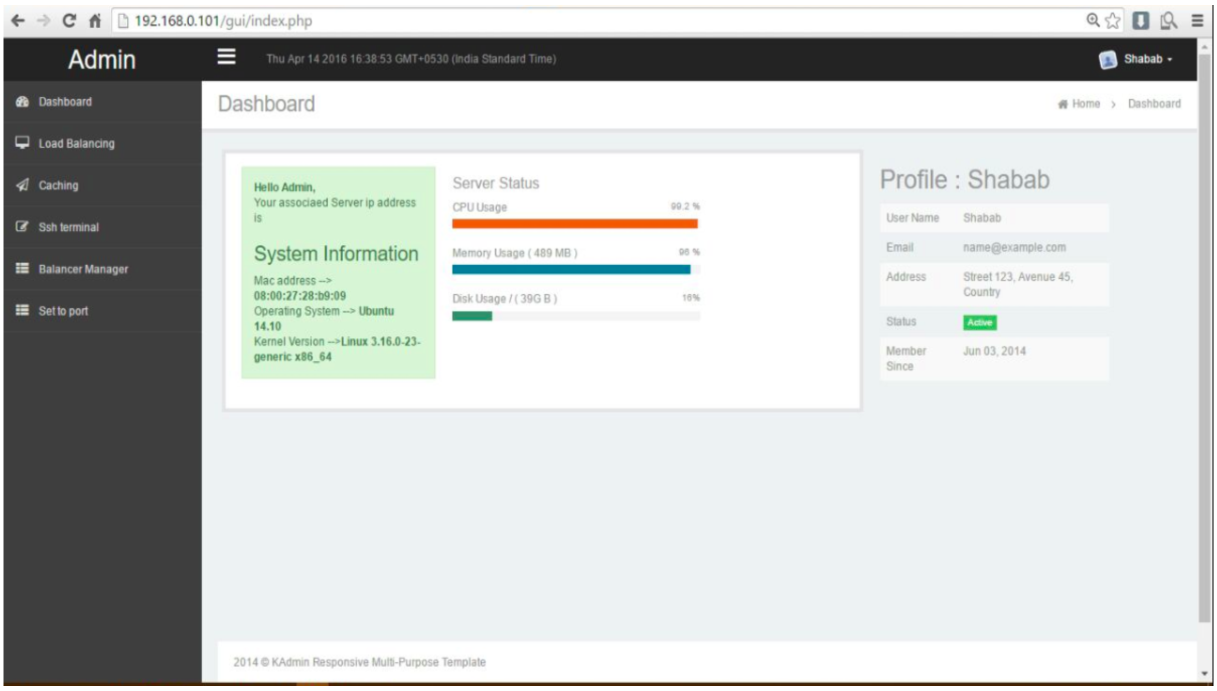


Figure 4.1: Software architecture Design

4.2.1 Front End Designs

Register/Login: Initially when the user accesses the webapp he gets a login/register page.


192.168.0.101/gui/index.php

Admin

Thu Apr 14 2016 16:38:53 GMT+0530 (India Standard Time)

Shabab

Dashboard

Load Balancing

Caching

Ssh terminal

Balancer Manager

Set to port

Dashboard

Home > Dashboard

Hello Admin,
Your associaed Server ip address is

System Information

Mac address --> 08:00:27:28:b9:09
Operating System --> Ubuntu 14.10
Kernel Version -->Linux 3.16.0-23-generic x86_64

Server Status

CPU Usage 99.2 %

Memory Usage (489 MB) 96 %

Disk Usage / (39G B) 16%

Profile : Shabab

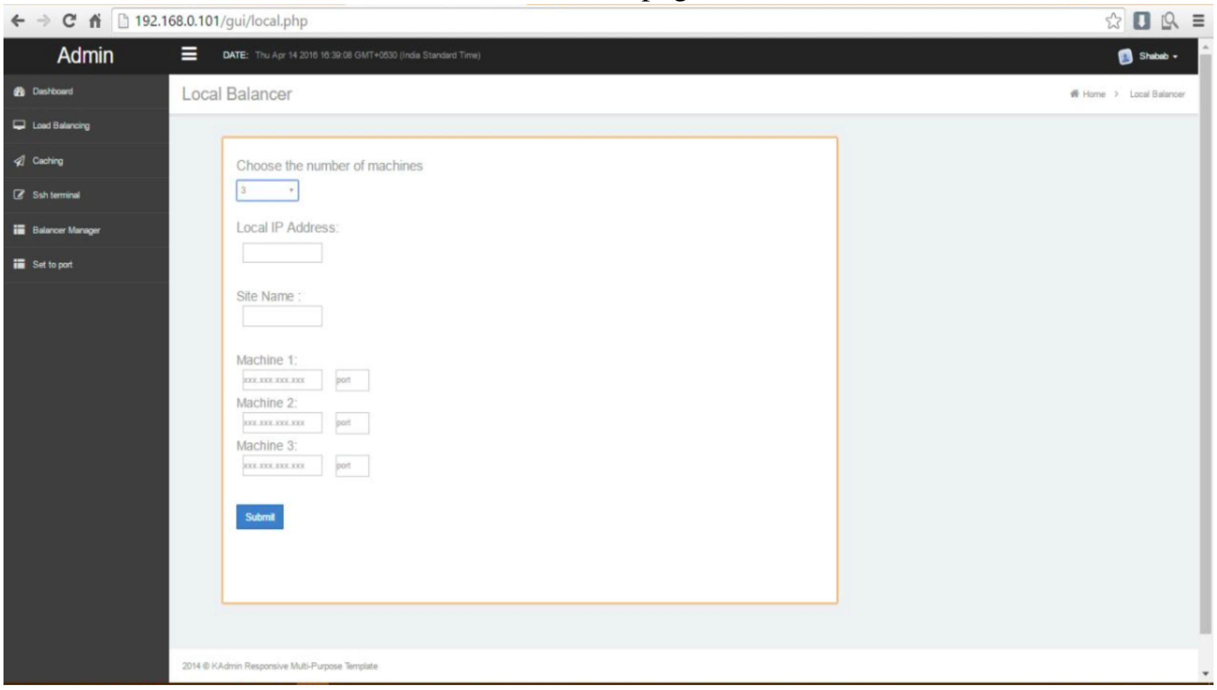
User Name	Shabab
Email	name@example.com
Address	Street 123, Avenue 45, Country
Status	Active
Member Since	Jun 03, 2014

2014 © KAdmin Responsive Multi-Purpose Template

Figure 4.2: Front End Design

Dashboard: The admin logins to the GUI and is directed to the dashboard, which displays the information of the CPU usage, Memory usage, Data usage and the information of the system the admin is using based on the IP address the admin has provided.

balancer.png


192.168.0.101/gui/local.php

Admin

DATE: Thu Apr 14 2016 16:39:08 GMT+0530 (India Standard Time)

Shabab

Dashboard

Load Balancing

Caching

Ssh terminal

Balancer Manager

Set to port

Local Balancer

Home > Local Balancer

Choose the number of machines: 3

Local IP Address:

Site Name :

Machine 1: xxx.xxx.xxx.xxx port

Machine 2: xxx.xxx.xxx.xxx port

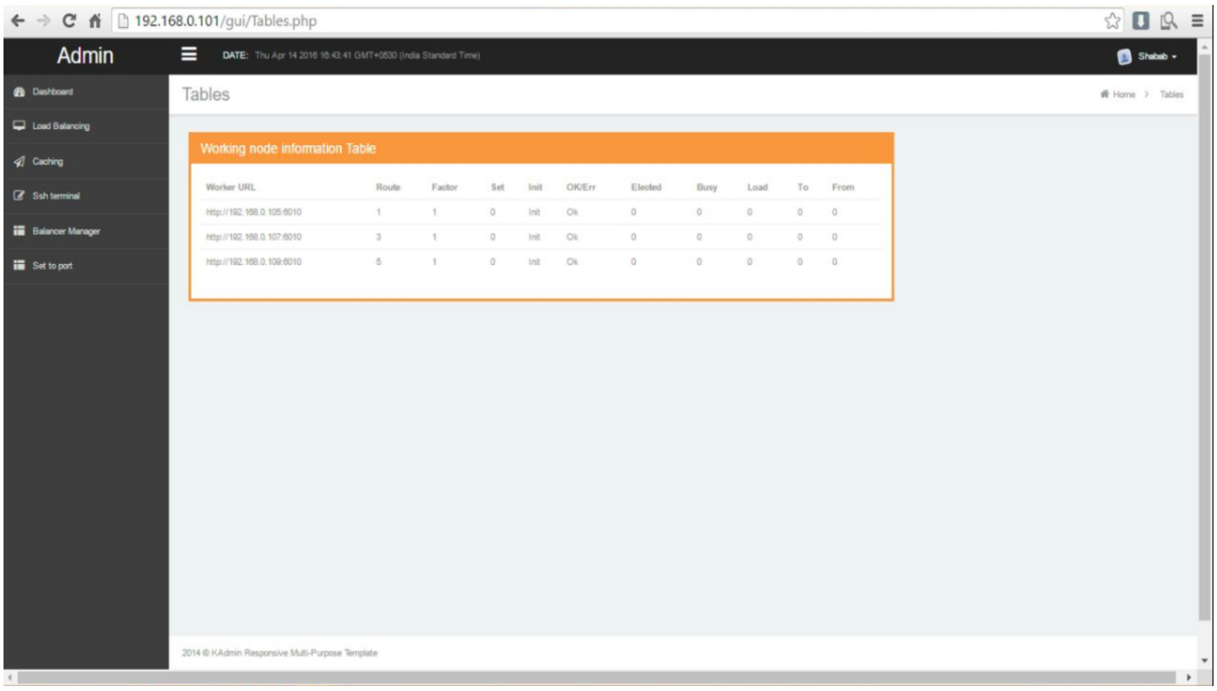
Machine 3: xxx.xxx.xxx.xxx port

Submit

2014 © KAdmin Responsive Multi-Purpose Template

Figure 4.3: Front End Design

Local Balancer: This functionality provided helps the admin to forward the incoming request to more than one machine and to which port of the machine acccording to his desire. But this features works only for local machine.


192.168.0.101/gui/Tables.php
Admin
DATE: Thu Apr 14 2016 16:43:41 GMT+0530 (India Standard Time)
Shabab
Dashboard
Load Balancing
Caching
Ssh terminal
Balancer Manager
Set to port
Tables
Home > Tables
2014 © KAdmin Responsive Multi-Purpose Template

Working node information Table

Worker URL	Route	Factor	Set	Init	OK/Err	Elected	Busy	Load	To	From
http://192.168.0.105:6010	1	1	0	Init	Ok	0	0	0	0	0
http://192.168.0.107:6010	3	1	0	Init	Ok	0	0	0	0	0
http://192.168.0.109:6010	5	1	0	Init	Ok	0	0	0	0	0

Figure 4.4: Front End Design

Balancer Manager: The manager provides the information about the working nodes in tabular form. This includes information about the load on a specific node, the number of requests on a node, the node elected, the data sent and received.

4.2.2 Component Diagram

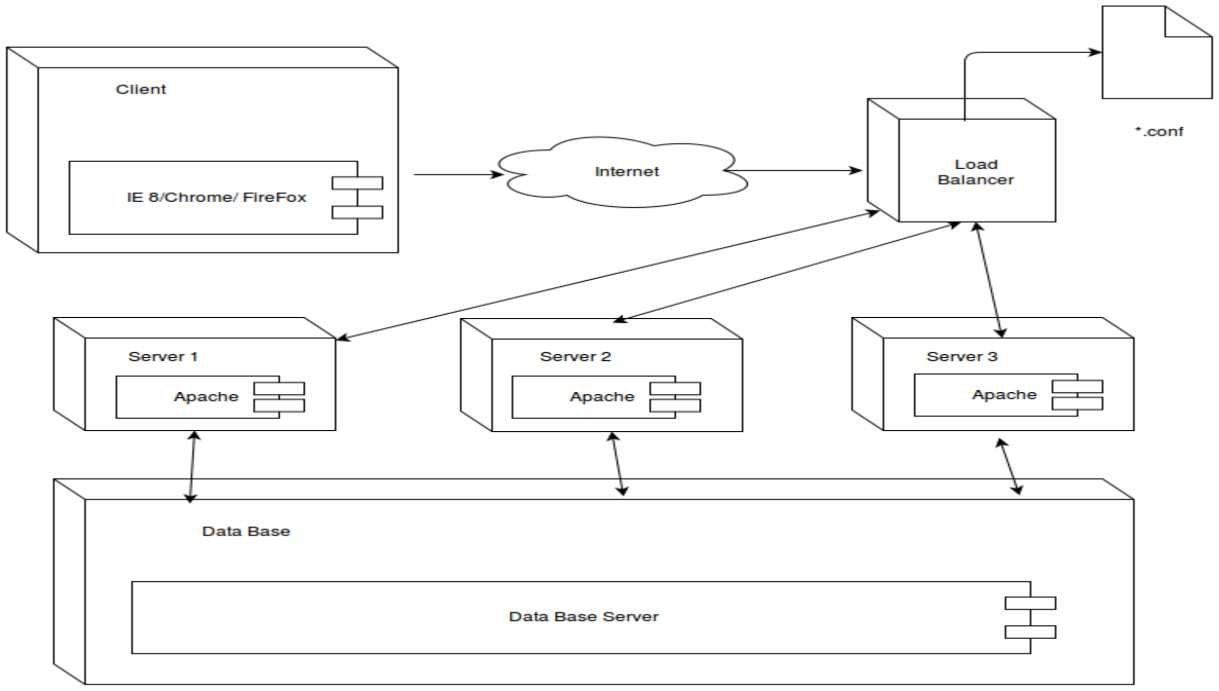


Figure 4.5: Component Diagram of Scalability Design

4.2.3 Deployment Diagram

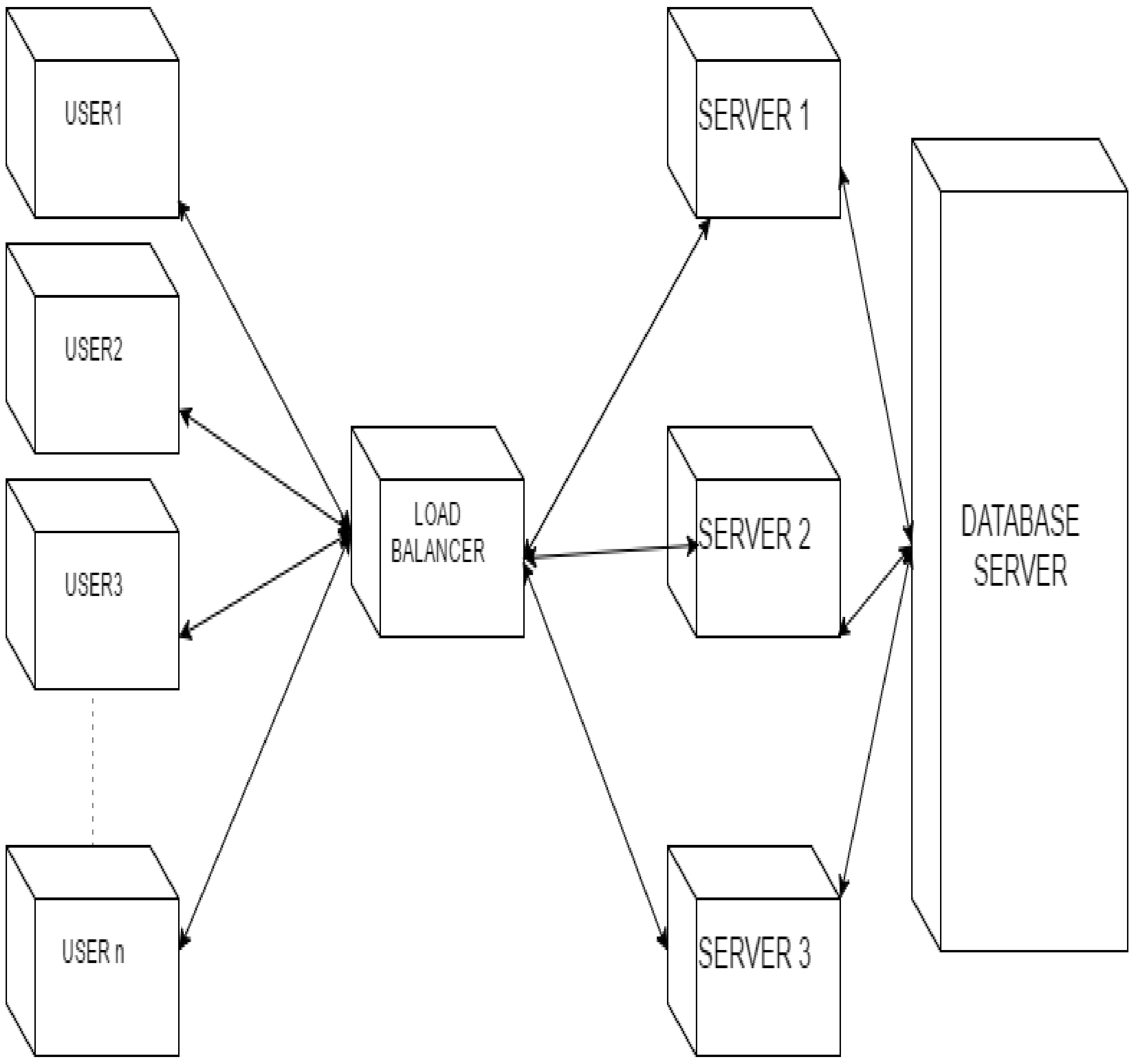


Figure 4.6: Deployment Diagram of Scalability Design

4.3 Database Design

4.3.1 E-R Diagram

drawing (1).jpg

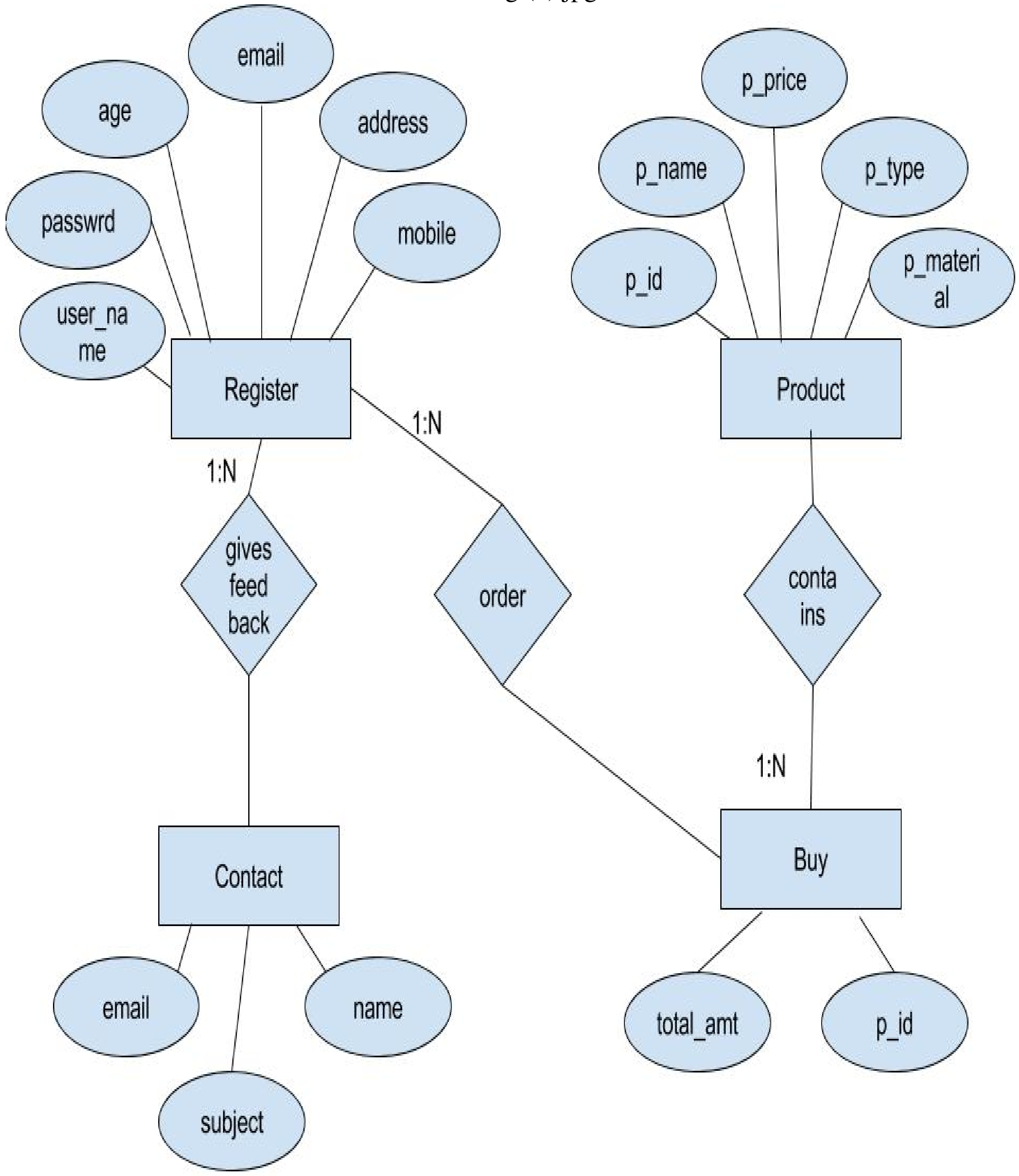


Figure 4.7: E-R Diagram of Scalability Design

Chapter 5

Implementation Details

5.1 Assumptions And Dependencies

5.1.1 Assumptions

The following Assumption was taken into consideration:

- The project is about web scalability, so main assumption is all the users will be served by their requests without increase in the response time, also there will be no error rate for any of the user once the system is scaled.

- There should not be any occurrence of bottleneck in Database with or without scaling the system, all queries should be processed for any no of users requesting for the db request.

5.1.2 Dependencies

The dependencies are as follows:

- For Web Server Apache 2.4 web server is used, this server is main backbone of the whole system, also for database MySQL server is used, all the systems scaled use this database server in order to process their db queries.

- VM ware virtual machine is used to divide the whole system into different machines, so the system is dependent on virtual machine software.

5.2 Implementation Methodologies

Different modules of the project are, Virtualization of the machine, installation of Server Operating System, installation of servers (Web server Database Server), installation of load balancer, Caching ,Creation of an E-Commerce website for testing of project, Installing and using web stress tool for testing of scaled and un scaled servers, creating an admin interface in order to make server configurations ease.

5.2.1 Modular Description of Project

Virtualization: In this module a single machine which was being used as Primary server is divided using VM ware virtual machine into four different machines for efficient work and coordination between divided machines.

Purpose for division is, a single machine has (in this case) has more processing power for different tasks, but when it comes about execution of project half of the processing power is being wasted i.e not used.

To use that remaining processing power,virtualization is used.

Installation of server Operating system: The operating system used for throughout project is Ubuntu 14.10 (Utopic Unicorn).

This is a command line interface based Server operating system. Also it is an open source operating system

Servers: For hosting and scaling the primary web server used is Apache 2.4 web server, and for database server MySQL server is used.

Load Balancer: An extension of apache web server named mod proxy is used to set the load balancer.

This load balancer balances the incoming request to the inner nodes for processing and returns the desired output to the user.

Caching: Caching on web server is used to reduce the server side operations, when a request arose, first it is being checked in the cache server, if caching is enabled, if found the cache server serves the request without any processing loads on actual web server.
The Cache server used is Squid server.

E-Commerce website: After implementation of the proposed architecture, to test all the servers capacity, an E-Commerce based Jewelry store website is being created. This website is full working website to buy any type of jewelry. This website is stored on the web servers and it is primary input to the test bed.

Web Stress tool: This is open source tool to test the capacity of the servers. This tool generates virtual no of users as input provided and shoots queries or request from all the created users, as an input it takes the url of the website which is being stored on scaled server.

In ouput it returns multiple graphs of total testing process, this includes main graph that is response time to no of users graph and error rate to no of users graph, through this graph we can understand the capacity of server.

Admin Interface: In order to make all the admin side configurations ease, an Admin dashboard Php based GUI is created. In this GUI you can access as well as configure your systems for load balancing, Caching, Setting a website to a particular port and many more.

5.3 Detailed Analysis and Description of Project

5.3.1 Activity Diagram

diagram(1).png

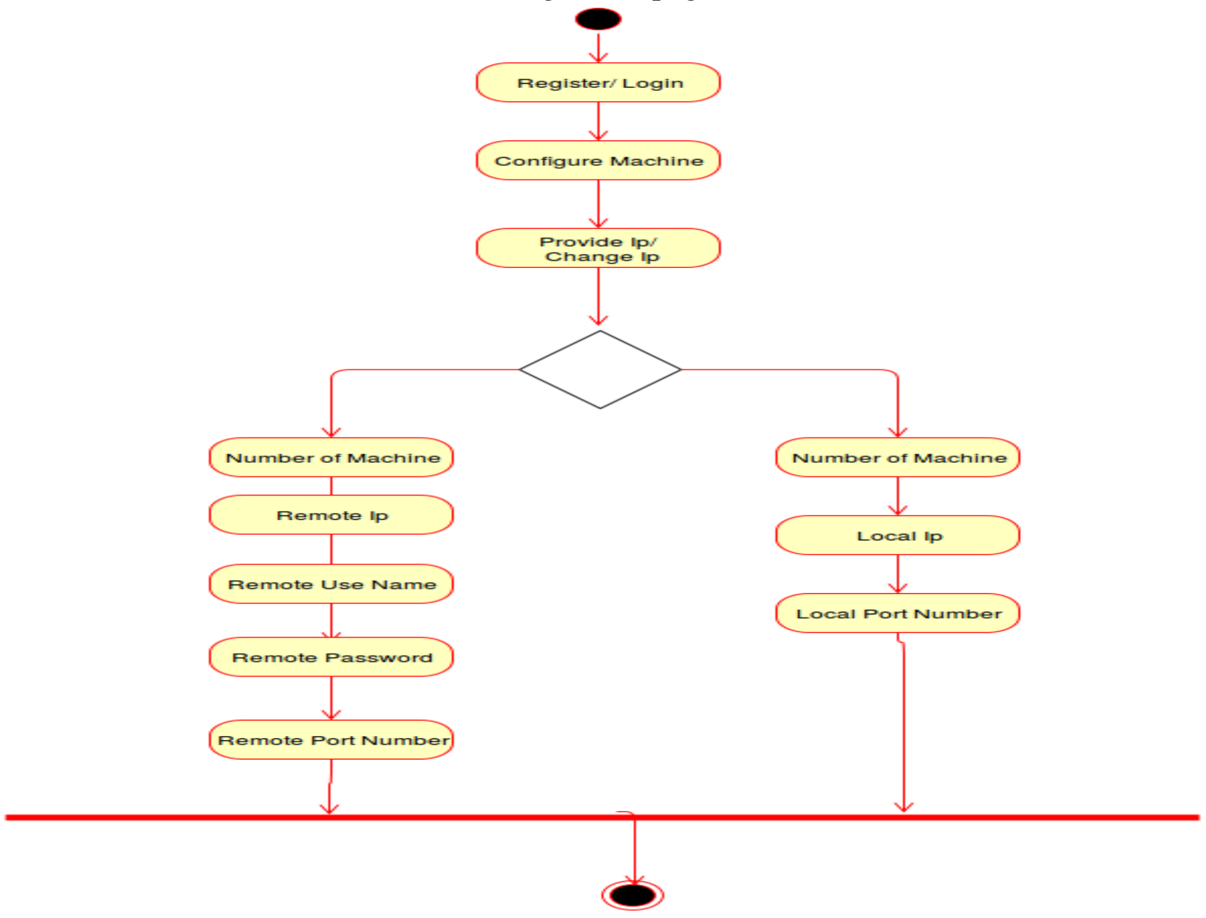


Figure 5.1: Activity Diagram

This work is been carried out in four stages:

Step1: The Admin registers himself and logs in to the interface.

Step2: The admin enters the IP address of the system he has logged in or he changes the previously entered IP if he logged in from a new system.

Step3: Admin configures the local or remote machines to map the request on available system.

Step4: a) Remote- To configure the requests on remote machine, he enters the remote machines IP, Username and Password and also the no of machines he desires to use.
b) Local- This process is same as remote process except that it does not asks for Username and Password as the admin is configuring local machine.

Registration of Administrator: When admin has a particular hosting server, admin can use the designed interface in order to set all the configurations for scaling as well as port settings of the website, admin is registered to use the interface.

Admin Dashboard: On this web page descriptions of server is displayed i.e what kind of operating system is used, what web server is being used, total configuration of the admin machine and network traffic.

Balancer: In this page a short description is displayed of what a balancer does and what kind of balancer is used and the configurations inputs is noted that how to provide input. Ip address of number of backend servers is taken as input as well as Ip address of the balancer server is taken, on just a click, all the configuration is executed and the admin server is now configured for scaling purpose.

Caching: If admin wants to enter some of websites or web pages into the cache memory of the server, here admin can configure for such setting. Input to be taken is Ip address of the server machine, notes will provide what and how to provide the inputs. And memory to be allocated for cache is asked on this page. After clicking on config button all the desired setting is executed on server side and caching is implemented for that server.

Set to Port: If the admin wants a particular website to be hosted on his server but on different port number, on this page admin can do such settings, inputs to be provided is ip address of the server, port number on which the site is being to set and the website files.

Admin Shell: An administrator shell is provided to the admin which is php based implemented.

Balancer Manager: When admin configures his system to load balancer, here admin can manage the balancer configuration, i.e increase or decrease the no of balancing nodes/machines and can choose the strategy on how the balancer should balance the load.

5.4 Usecase Report

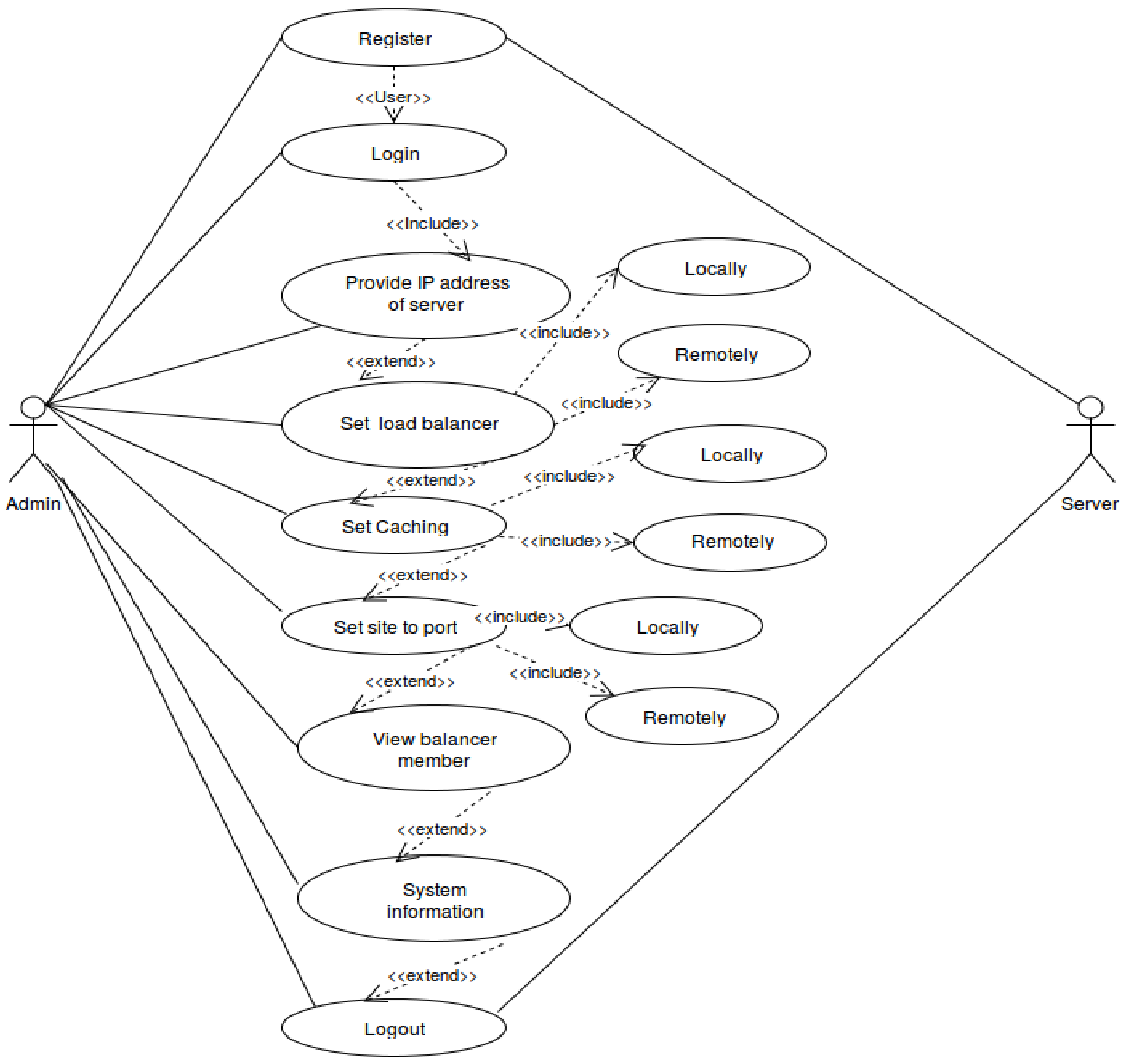


Figure 5.2: ER Diagram

5.4.1 Usecase Report

Title:	Web Scalability admin dashboard
Description:	Admin dashboard enables the user to configure the system into an scalable system, also it provides additional information of the server machine. Using this dashboard admin can make its system scalable by enabling load balancing and caching and setting a website to a port.
Primary Actor:	Administrator
Preconditions:	Well knowledge of the server machine
Post conditions:	By using simple Gui features, admin can make system scalable.
Frequency of Use:	Admin can use at any time.
System Requirement:	Admin server and web server.

Table 5.1: Usecase Report

5.4.2 Class Diagram Report

Title:	Web scalability admin dashboard
Description:	Admin dashboard enables the user to configure the system into an scalable system, also it provides additional information of the server machine. Using this dashboard admin can make its system scalable by enabling load balancing and caching and setting a website to a port.
Primary Actor:	Administrator. OR User
Preconditions:	Well knowledge of the server machine
Post conditions:	By using simple Gui features, admin can make system scalable.
GUI Interface:	First user will provide IP address and ports in back ground shell script will be executed and server will be started.
Database:	Databse is used for retrieving the result from. It is the main component.

Table 5.2: Class Diagram Report

Chapter 6

Results and Discussion

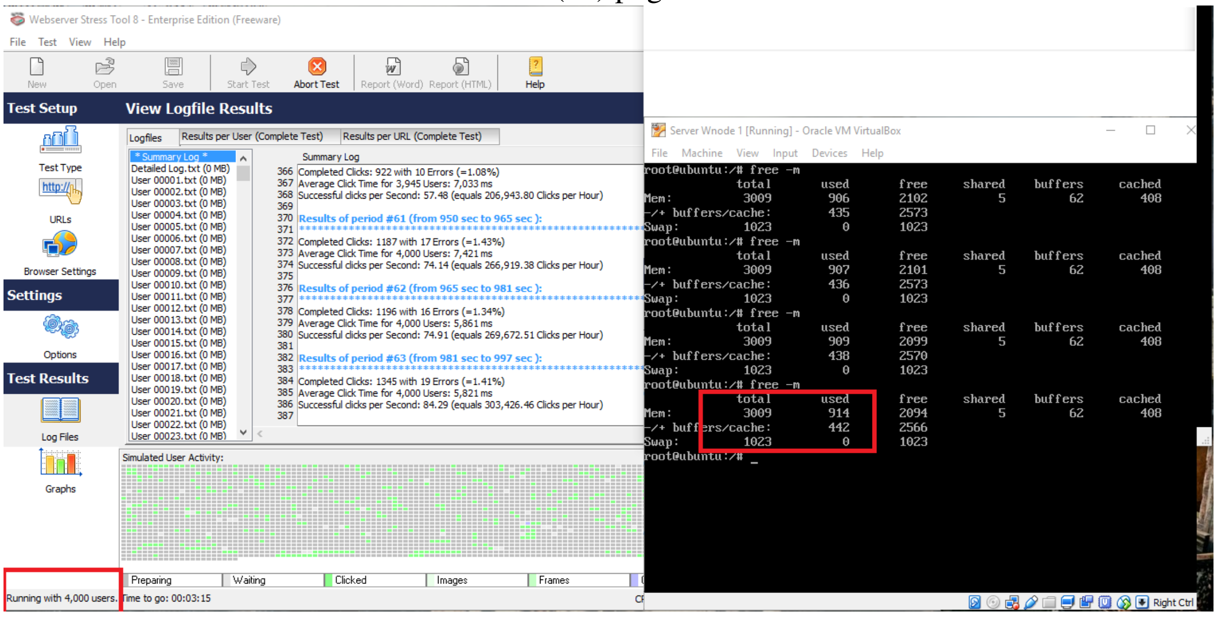
6.1 Test cases and Result

When the website is tested for load using the webserver stress tool, the results generated in both cases i.e with load balancing and without load balancing are generated by the stress tool in the form of graphs.

6.1.1 Unit Testing

Unit testing is a software testing method by which invidual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. The goal of unit testing is to isolate each part of the program and show that the invidual parts are correct. For the basic unit testing we configured a virtual machine with 3 GB RAM and Quad core CPU and installed the server operating system Ubuntu server 14.10 Utopic Unicorn, web server Apache and database server MySQL. After completely configuration, we created an E-Commerce based Jewellry website with database, Then we placed that website in this configured server. When we tested the website using Web Stress Tool for 4000 users, the server was unable to handle the load for such bulk. But an important point to be noted in this case is that even at full load it only used 945 MB RAM (approximately) out of 3 GB provided. Ideally system without load was occupying 450 MB of RAM i.e the system was only occupying 500 MB of ram and rest of the processing was getting wasted which is displayed in fig (6.1).

(39).png


Webserver Stress Tool 8 - Enterprise Edition (Freeware)
File Test View Help
New Open Save Start Test Abort Test Report (Word) Report (HTML) Help
Test Setup
Test Type
URLs
Browser Settings
Settings
Options
Test Results
Log Files
Graphs
View Logfile Results
Logfiles
Results per User (Complete Test)
Results per URL (Complete Test)
Simulated User Activity:
Preparing
Waiting
Clicked
Images
Frames
Server Wnode 1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Right Ctrl

```
* Summary Log *
Detailed Log.txt (0 MB)
User 00001.txt (0 MB)
User 00002.txt (0 MB)
User 00003.txt (0 MB)
User 00004.txt (0 MB)
User 00005.txt (0 MB)
User 00006.txt (0 MB)
User 00007.txt (0 MB)
User 00008.txt (0 MB)
User 00009.txt (0 MB)
User 00010.txt (0 MB)
User 00011.txt (0 MB)
User 00012.txt (0 MB)
User 00013.txt (0 MB)
User 00014.txt (0 MB)
User 00015.txt (0 MB)
User 00016.txt (0 MB)
User 00017.txt (0 MB)
User 00018.txt (0 MB)
User 00019.txt (0 MB)
User 00020.txt (0 MB)
User 00021.txt (0 MB)
User 00022.txt (0 MB)
User 00023.txt (0 MB)
```

```
Summary Log
Completed Clicks: 922 with 10 Errors (=1.08%)
Average Click Time for 3,945 Users: 7,033 ms
Successful clicks per Second: 57.48 (equals 206,943.80 Clicks per Hour)

Results of period #61 (from 950 sec to 965 sec ):
*****************************************************************
Completed Clicks: 1187 with 17 Errors (=1.43%)
Average Click Time for 4,000 Users: 7,421 ms
Successful clicks per Second: 74.14 (equals 266,919.38 Clicks per Hour)

Results of period #62 (from 965 sec to 981 sec ):
*****************************************************************
Completed Clicks: 1196 with 16 Errors (=1.34%)
Average Click Time for 4,000 Users: 5,861 ms
Successful clicks per Second: 74.91 (equals 269,672.51 Clicks per Hour)

Results of period #63 (from 981 sec to 997 sec ):
*****************************************************************
Completed Clicks: 1345 with 19 Errors (=1.41%)
Average Click Time for 4,000 Users: 5,821 ms
Successful clicks per Second: 84.29 (equals 303,426.46 Clicks per Hour)
```

```
Running with 4,000 users.  Time to go: 00:03:15
```

```
root@ubuntu:/# free -m
             total       used       free     shared    buffers     cached
Mem:          3009        906       2102          5         62        408
-/+ buffers/cache:        435       2573
Swap:         1023          0       1023
root@ubuntu:/# free -m
             total       used       free     shared    buffers     cached
Mem:          3009        907       2101          5         62        408
-/+ buffers/cache:        436       2573
Swap:         1023          0       1023
root@ubuntu:/# free -m
             total       used       free     shared    buffers     cached
Mem:          3009        909       2099          5         62        408
-/+ buffers/cache:        438       2570
Swap:         1023          0       1023
root@ubuntu:/# free -m
             total       used       free     shared    buffers     cached
Mem:          3009        914       2094          5         62        408
-/+ buffers/cache:        442       2566
Swap:         1023          0       1023
root@ubuntu:/# _
```

Figure 6.1: Click Time hits/s ,Users/s.

6.1.2 Functional Testing

Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered in this testing.

Ramp tests Ramp Tests are variations of Stress Tests in which the number of users is increased over the life of the test from a single user to hundreds of users. By reviewing the graphs of click times and errors, a Ramp Tests can help you determine what maximum load a server can handle while providing optimal access to web resources.

As discussed in unit testing the system RAM was not utilized fairly, we divided the machine that we created with 3 GB RAM into five virtual machines each with 512 MB RAM. The management of these machines required balancing the load. So one of the machines was choosed as a load balancer, the three of them as working nodes and one as a database server. These machines were tested for stress without using load balancing.

Result without LoadBalancer:

Without load balancing, for 4000 users the response time was 16000 ms and also the error rate shot to 100 percent at around 400 users.

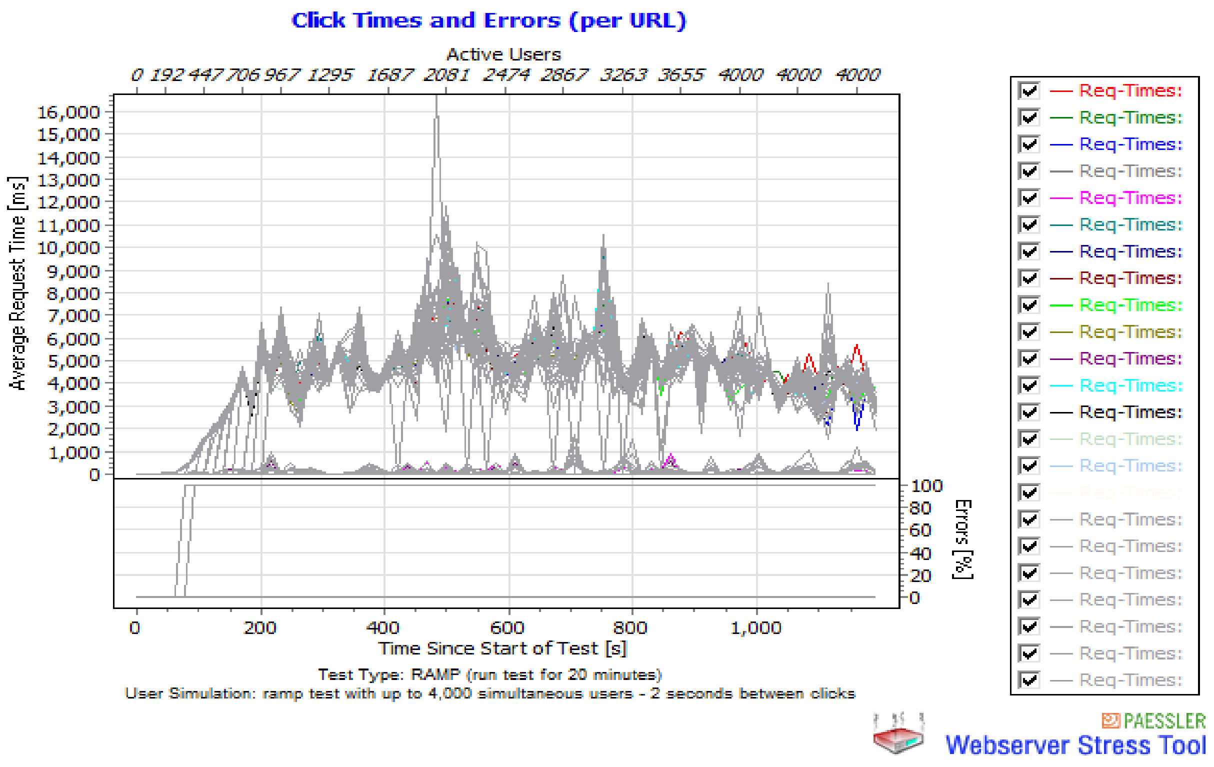


Figure 6.2: Click Times and Errors.

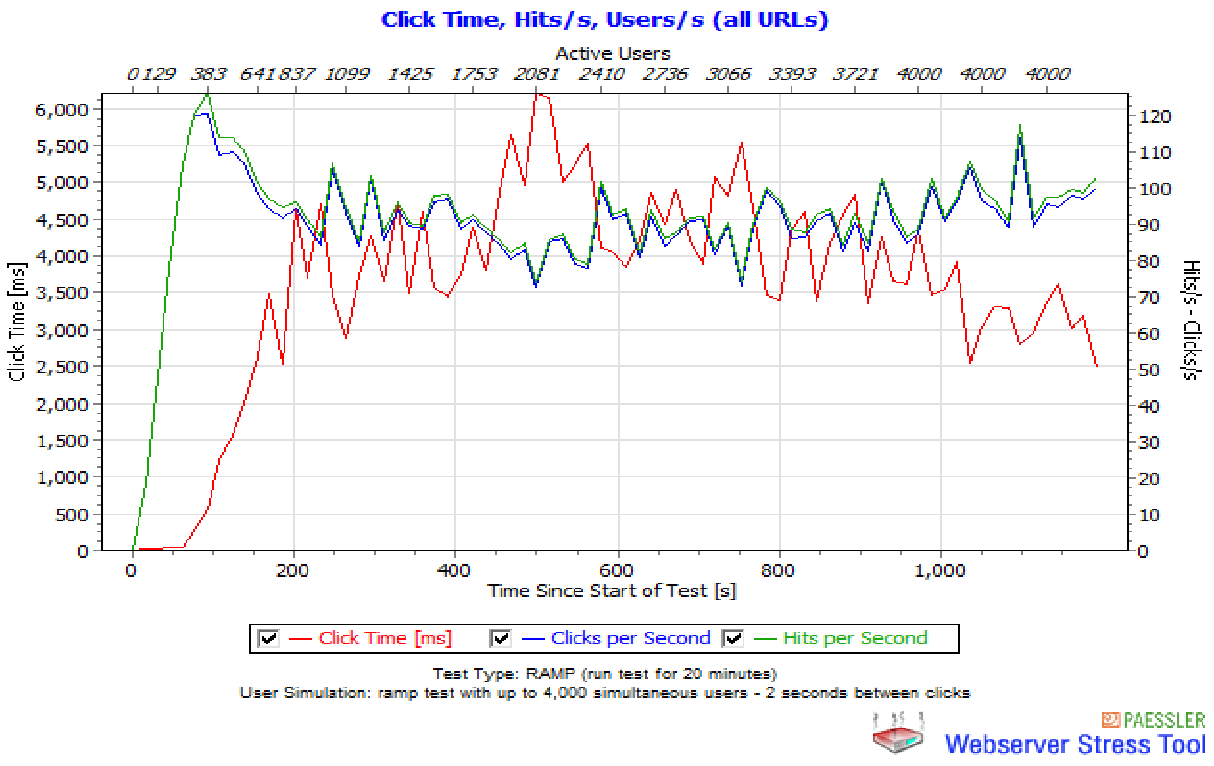


Figure 6.3: Click Time,Hits/s,Users/s.

The above graph shows the result of hits per second and clicks per second for 4000 users.

Reslt with Loadbalancer.

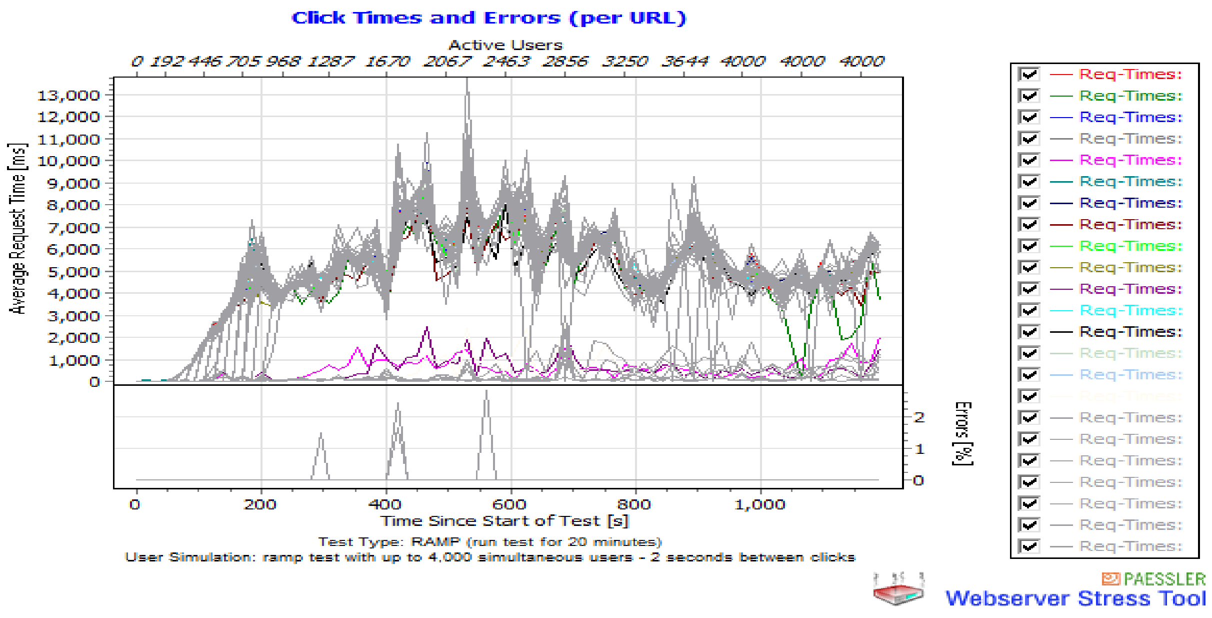


Figure 6.4: Click Time Error.

With load balancing the response time drop down to 13000 ms for 4000 users and also the error rate was comparatively very low.

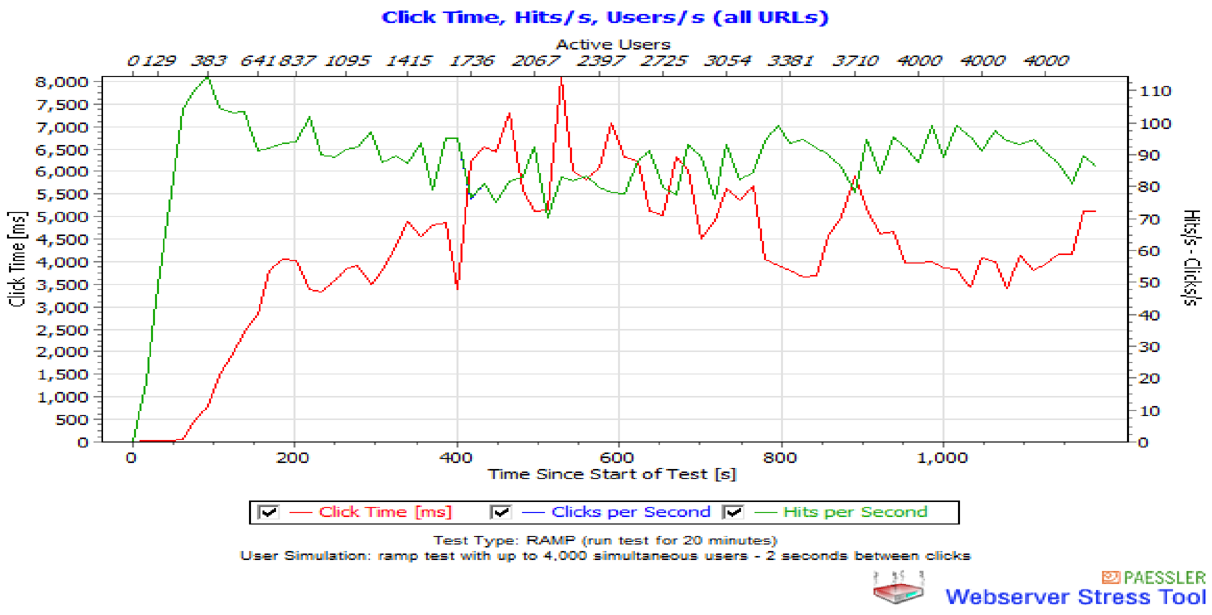


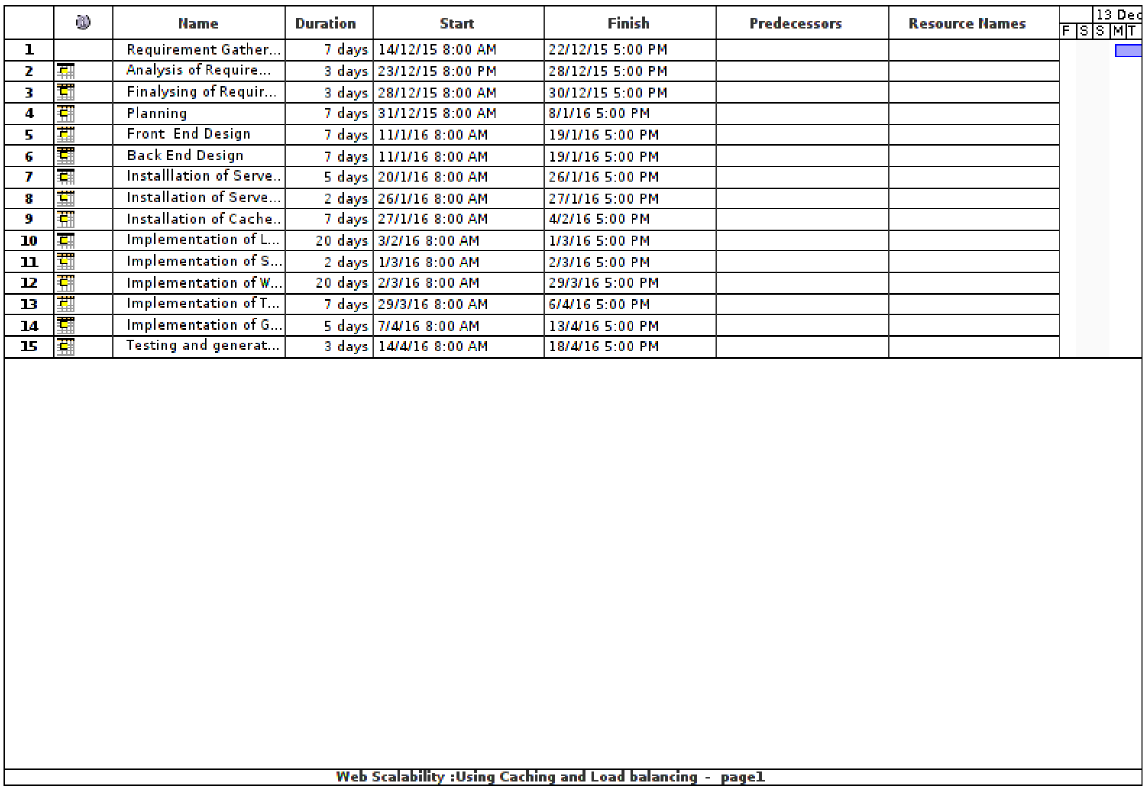
Figure 6.5: Click Time hits/s ,Users/s.

The results achieved using load balancing are more reliable than those obtained without load balancing technique. Load balancing ensures less reponse time and error rate and more utilization of available resources.

Chapter 7

Project Time Line

7.1 Project Time Line Matrix



		Name	Duration	Start	Finish	Predecessors	Resource Names
1		Requirement Gather...	7 days	14/12/15 8:00 AM	22/12/15 5:00 PM		
2		Analysis of Require...	3 days	23/12/15 8:00 PM	28/12/15 5:00 PM		
3		Finalysing of Requir...	3 days	28/12/15 8:00 AM	30/12/15 5:00 PM		
4		Planning	7 days	31/12/15 8:00 AM	8/1/16 5:00 PM		
5		Front End Design	7 days	11/1/16 8:00 AM	19/1/16 5:00 PM		
6		Back End Design	7 days	11/1/16 8:00 AM	19/1/16 5:00 PM		
7		Installlation of Serve..	5 days	20/1/16 8:00 AM	26/1/16 5:00 PM		
8		Installation of Serve...	2 days	26/1/16 8:00 AM	27/1/16 5:00 PM		
9		Installation of Cache..	7 days	27/1/16 8:00 AM	4/2/16 5:00 PM		
10		Implementation of L...	20 days	3/2/16 8:00 AM	1/3/16 5:00 PM		
11		Implementation of S...	2 days	1/3/16 8:00 AM	2/3/16 5:00 PM		
12		Implementation of W...	20 days	2/3/16 8:00 AM	29/3/16 5:00 PM		
13		Implementation of T...	7 days	29/3/16 8:00 AM	6/4/16 5:00 PM		
14		Implementation of G...	5 days	7/4/16 8:00 AM	13/4/16 5:00 PM		
15		Testing and generat...	3 days	14/4/16 8:00 AM	18/4/16 5:00 PM		

Web Scalability :Using Caching and Load balancing - page1

Figure 7.1: Time Line Matrix

7.2 Project Time Line Chart

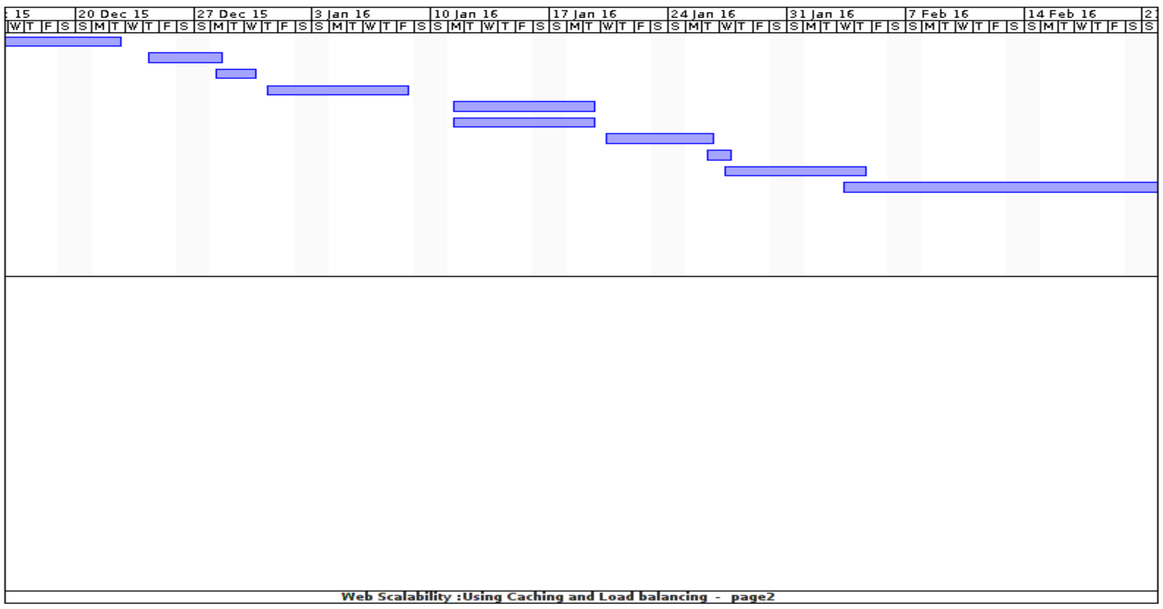


Figure 7.2: Time Line Chart

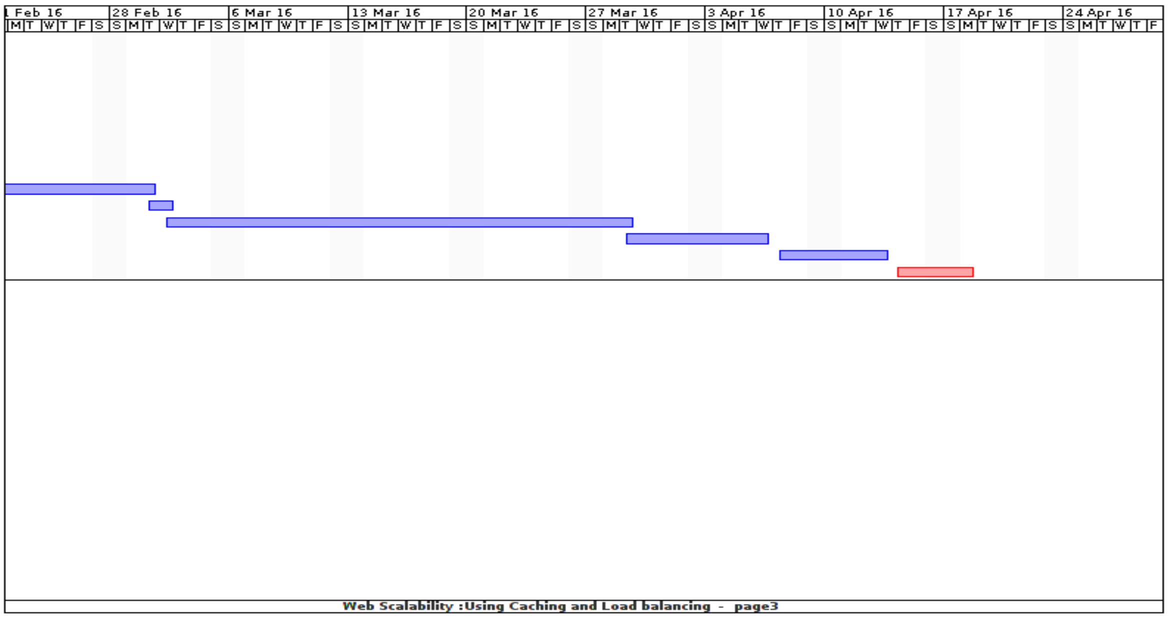


Figure 7.3: Time Line Chart

Chapter 8

Task Distribution

8.1 Distribution of Workload

8.1.1 Scheduled Working Activities

Activity	Time Period	Comment
Requirement Gathering	13 Days	Requirement gathering has took placed through searching on internet and taking the ideas, sharing the views among group members.
Planning	07 Days	Planning was done by reviewing of literature of IEEE papers and by taking the walkthrough.
Design	07 Days	Designing was accomplished by creating UML diagram, charts.
Implementation	70 days	Implementation has done First creating the backend and then front end module by module.
Testing	15 days	Testing has done by perfoming unit testing, alpha & Beta Testing, integrated testing and system testing.
Deployment	07 days	Deployment has done by installing project on the server.

Table 8.1: Scheduled Working Activities

8.1.2 Members actvities or task

Member	Activity	Time Period	Start Date	End Date	Comment
M1, M2, M3, M4	Requirement Gathering	07 Days	14/12/15	22/12/15	M1 and M2 has perfomed the seaching for project requirement on the internet by reviewing the related literature and by anlysing the related prject which is already available in the market. Regularly inform to the other member of team.
M1, M2, M3, M4	Analysing of the requirement	03 Days	23/12/16	28/12/16	M1, M2, M3, M4 done the requirement analysing of project by sharing the ideas, and by discussing on related information which is gather by the M1, And M2. M3 and M4 has created the list of requirement after every meeting
M1, M2, M3, M4	Finalysing the requirement	03 Day	28/12/15	30/12/15	Whole team finalize the requirement. M1 and M4 has created a list of finalise requirement.
M1, M2, M3, M4	Planning	07 Days	31/12/15	20/01/16	Planning has done by walkthrough and by analysing the available product. M2 and M3 creats a list of funtion which will be implement in the project. Each and every module were discuss in every group meeting and M1 and M2 creates a blue print for project .

M3, M2	Front End design	07 Days	11/01/16	19/01/16	M3 and M2 creates the front end of the system and data flow diagrams and informed to the whole team regularly.
M1, M4	Back End design	07 Days	11/01/16	19/01/16	M1 and M4 creates back end of the system and data flow diagrams and informed to the whole team regularly.
M2,M3	Installation of Server OS Ubuntu14.10	05 Days	20/01/16	26/01/16	M2 and M3 Installed Server OS Ubuntu14.10 discuss on it with other team membar
M4	Installation of Server (DB and Web)	02 Days	26/01/16	27/01/16	M4 installed Server (DB and Web) and discuss on it with other team member
M1 ,M4	Installation of Cache Server	07 Days	27/01/16	04/02/16	M4 and M1 installation of Cache Server. M4 and M1. Discuss the installation to the other member of team.

M2, M4	Implementation of Load Balancer	12 Days	03/02/16	18/02/16	M2 and M4 done the load balancer and discuss the method with other team member regularly.
M3, M4	Implementation of setting of port	02 Days	18/02/16	19/03/16	M3 and M4 implemented the setting of ports.
M1, M3	Implementation of Web-Site(Jewellery)	20 Days	02/03/16	29/03/16	M1 and M3 implemeted the website (Jewellery). And regularly giving update to other members.
M2 ,M3	Implemantation of testing software(Webstress)	07 Days	29/03/16	06/04/16	M2 and M3 implemented the testing software(Webstress) and explain the working of it to other members.
M1, M2,M3, M4	Implementation of GUI for Project	05 Days	07/04/16	13/04/16	M1, M2, M3 and M4 implemented the GUI for project.
M2 ,M3	Testing and Generating report	03 Days	14/04/16	18/04/16	M2 and M3 have tested and generated the report.

Table 8.2: Member Activities and Task

Chapter 9

Conclusion and Future Scope

9.1 Conclusion

Scalability is an important constraint in this century as several businesses are implemented on web in order to spread it throughout the globe. We have presented a new approach and several techniques for scaling web server. The new technique differs from existing technique in that with the use of minimum caches and improving load balancing technique we can achieve high request handling capacity of a server.

Cache operations are costly,so when the load on the server is less than its capacity this operations should be preserved, We divided the throughout architecture into two layers, this provides to reduce costly operations.

This solution we purposed in paper will be optimal as it uses a bit of hardware that is cache and software in load balancer will be used in optimal and efficient web server scaling.

9.2 Future Scope

1. In Our project we have virtualized our server by own but in future we can add monitoring machanism in which it will look toward traffic or load and automatically create number of virtual server by its own.

2. In future we can add one mechanism that continously check all the server functioning.In this it mainly check whether the server is functioning properly or not. If due to some reson any one of server crashes down we it will automatically discard it and the functionality will not be effected.

References

[1] E. Kirda, M. Jazayeri, C. Kerer, and M. Schranz, *Experiences in Engineering Flexible Web Services.*IEEE MultiMedia, vol. 8, no. 1, pp. 58-65, Jan. 2001.

[2] Sameena Naaz Load Balancing Algorithms for Peer to Peer and Client Server Distributed Environments,IEEE MultiMedia, June 2012

[3] Gregor v. Bochmann Scalability of Web-Based Electronic Commerce Systems,IEEE MultiMedia, July 2003.

[4] Daniel A. MenascÃ©website through caching George Mason University menasce@cs.gmu.ed June 2003.

[5] Thomas Fankhauser,Christos Grecos, Xinheng Wang, Qi Wang, Ansgar GerlicherWeb Scaling Frameworks for Web Services in the Cloud IEEE MultiMedia

[6] Xianjun Geng,Ram D Gopal, R. Ramesh, Andrew B. WhinstonScaling Web Services with Capacity Provision Networks IEEE MultiMedia ,July 2015

[7] Ken BirmanCan Web Server Scale Up? IEEE MultiMedia ,July 2015

[8] Damith C. RajapakseA Fresh Graduates Guide to Software Development Tool and Technologies University of Singapore,April 2011

Own Publications

Siddique Asma Abdul Wahab Zainbunnisa ,Patil Amit Suresh Archana,Mirsinge Ibad Ibrahim Saba ,Tulve Shabab Kasim Shagufta *Web Scalability :Using Server Vertiulization ,Load Balancing And Caching* IJSRD International Journal for Scientific Research Development.

Web Scalability: Using Server Virtualization, Caching and Load Balancing

Siddique Asma[1] Patil Amit[2] Mirsinge Ibad[3] Tulve Shabab[4] Prof. P.S Lokhande[5]
[1,2,3,4]Student [5]Assistant Professor
[1,2,3,4,5]Department of Computer Engineering
[1,2,3,4,5]AIKTC, University of Mumbai, Mumbai, India

Abstract— Web applications are the main source of information exchange of businesses over the globe. The main problem over the globe about the web applications is the service of the particular server. As a server has its own capacity to handle some amount of particular clients, there is always a possibility the number of Clients may get increased, at this time most of servers are unable to respond. This paper gives a purposed solution for Servers to handle more clients even when its capacity exceeds without scaling it horizontally that is (without increasing hardware). Management of clients requests on a web Server is managed In order to increase the capacity on a server and hence it gets capable to handle more clients. Architecture for scaling web servers is uses application level scaling which makes it flexible to be applied on any other servers.

Key words: Scaling, Web Server Scaling, Caching, Load Balancer, Application Level Scaling, Scalability

I. Introduction

Now-a-days E-commerce sites are increasing rapidly. Almost all selling and buying is done on E-commerce sites. Since these type of business is growing fast the system which is responsible for this type of business should also be improved so as to be compatible with increasing number of users. It is important to improve the system so as to maintain its performance. If the performance of this type of system is lowered then it will lead to the losing a customers. Losing customers means loss in business. In E-Commerce one hour of site failure results leads to loss of millions of dollars. Web applications typically undergo maintenance at a faster rate than other systems; this maintenance often consists of small incremental changes [1]

This influenced us to introduce scalability in these systems so as to improve the system and making system relevant to user. Scalability is essential term to achieve the best performance from the system. Scalability can be defined as "The potential or capability of a system to handle more loads without increasing the response time of request". In Web Scalability the capacity of Servers to handle a particular number of users request is increased, through this scalability is achieved. A Web application can be differentiated from a Web site based on the "ability of a user to affect the state of the business logic on the server"[2]

II. Literature review

For Web Scalability we have observed and studied following.

A. *Scaling using Load Balancer:*

When multiple web servers are present in a server group, the HTTP traffic needs to be evenly distributed among the servers. In the process these servers must appear as one web server to the web client, for example an internet browser. The load balancing mechanism used for spreading HTTP requests is known as IP spraying. The equipment use for IP spraying is also called the load dispatcher or network dispatcher or simply the load balancer. In this case the sprayer intercepts each HTTP request, and redirects them to a server in the server cluster.[5]

Depending on the type of the sprayer involved, the architecture can provide capability, load balancing and fail over requirements. Load balancing of servers by an IP sprayer can be implemented in different ways. These methods of load balancing can be set up in the load balancer based on available load balancing types. There are various algorithms used to distribute the load among the available servers. Algorithms: Random Allocation, Round Robin, Weighted Round robin.

B. *Scaling using Caching*

A powerful concept for reducing the delay caused in processing request and saving bandwidth is caching web resources. Here the resources refer to web pages. We find caches at Web browsers, organization proxy server caches, Internet service providers, content delivery networks (CDNs), and Web servers. A cache at the server side is mainly used to reduce the time required for processing a request. Server side caching is the act of caching data on the server. Data can be cached anywhere and at any point on the server that makes sense. It is common to cache commonly used data from the database to prevent hitting the database every time the data is required. We cache the results from competition scores since the operation is expensive in terms of both processor and database usage. It is common to cache pages or page fragments so that they don't need to be generated for every visitor.

1) *Cache Hit Ratio:*

A cache-hit ratio is the number of times the database found something in cache divided by the number of times it looked for some object in the cache. The higher the ratio, the more effective the cache is at improving performance. Taking into account the cache hit ratios; the frequency of reference f to Web documents is inversely proportional to the rank r, which is measured in terms of the document's popularity. The most popular document has $r = 1$, the second most popular has $r = 2$, and so on.

This relationship, called Zipf's law, states that: $f = k/r$ . . . where k is a constant.

Types of Caching Used:

- Object caches: are used to store objects for the application to reuse. These objects are either come from a database directly or are generated through data computation.
- Memcached: It is a high-performance, distributed memory object caching system, generic in nature, but

intended for use in speeding up dynamic web applications by alleviating database load.

– Reverse Proxy Cache: Reverse proxy caches is a strategy for web application caching. While object cache are usually used to cache database objects, reverse proxy cache are used to cache the result of web server e.g. web, DNS and other network lookups. Reverse proxy caches reduce loads on web servers and improve response time to user requests, facilitating scalability.

C. *Scaling Database:*

The database is the most common bottleneck in web applications, since a lot of reads and writes occur at the database level, and hence the primary focus in this book section is on scaling them. A scalable database is one that performs well under increasing traffic and dataset.

1) *Methods for Database Scaling*

a) Replication
- – Database replication using the master-slave model
- – Multi-master Replication Model
- – Replication Delay and Consistency

b) Partitioning
- – Round Robin Partitioning
- – Hash Partitioning
- – Range Partitioning
- – Vertical Partitioning
- – Horizontal Partitioning

2) *Short falls in Existing System:*

- – Multiple Web servers are assigned using horizontal scaling technique and appropriate load balancing technique is applied i.e DNS load balancer.
- – Cache operations are costly, hence existing system larger amount of cache is assigned for the repeated request of data.

3) *How to Overcome:*

- – In Proposed method we use server virtualization to create virtual server of single server and the applying load balancing technique in order to scale.
- – Using Zipf's Law and accurate calculations of the demanded page, It will be stored in the cache

III. PROBLEM STATEMENT

We are Scaling web servers on application level not by using horizontal or vertical scaling techniques.

IV. OBJECTIVE AND SCOPE

A. *Objective:*

The objective of the proposed system is to scale a web server on application level that is to use minimum hardware and scale the requests on server side. The proposed system would have the capability to handle clients request more than its original limit. Since available solutions in market are to increase the Hardware components another solution is to increasing the number of server machines.

B. *Project Scope:*

- – Server will be capable of handling more concurrent user then conventional server.
- – Caching gives Boost to handle more concurrent user.
- – Web security at client side of cookies.
- – Simultaneous updating of cache.
- – More space required for cache operation.

V. PROPOSED SYSTEM ARCHITECTURE

A. *Basic System Architecture:*

A typical Web-based electronic commerce system has a three-tier architecture: the Web server, electronic commerce application server, and database system[3]**.** Basic system follows three tier Architecture. Server is divided according to Web Server, Application Server and Database Server. The Web server is a process that handles requests from users and returns the requested web pages. The application server contains the business logic and accesses the database for information.

Basically the work of the Architecture is, A user asks any request to a server, in Server the Web server Process the request and forwards it to the application server. Application Server has access to database of the Server, through accessing the data it returns the information requested by the client

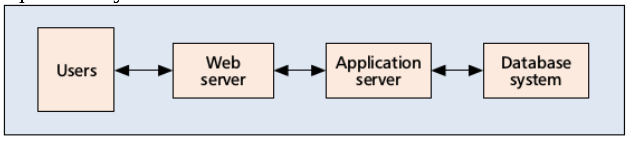


Fig. 1: Basic 3-tier Architecture

B. *Proposed Architecture:*

The System is divided into multiple components in which uses the three tier architecture along with some modifications.

1) *Load Balancer:*

On server side this is the first component to receive the users request. At a particular second it receives thousands of clients requests. The main purpose of this load balancer is to divide the requests among different web servers. Load balancer uses the technique such as DNS Load balancing. In this proposed system Architecture the load balancer will use this technique and the algorithm used will be Round Robin or Weighted Round Robin Along with Load balancer there is a counter which counts the number of concurrent users according to their respective ip address. Calculating these algorithms load balancer will forward the request to application server.

2) *Web Server:*

Main work of web server is process HTTP or any other protocol request, The primary function of a web server is to store, process and deliver web pages to clients. The communication between client and server takes place using the Hypertext Transfer Protocol (HTTP)[wiki]. In this Architecture the web server requests for web pages to the application server which are stored in database. If the request is found on the fragment it will retrieve from that fragment.

3) *Application Server:*

An application server, according to our definition, an application server exposes business logic to client applications through various protocols, possibly including HTTP [javaworld]. Here an application server will receive the request from the web server, It has all access to the database of the server. Application server will process the db request and will send reply to the web server.

4) Payment Gateway:

This is a third party gate way provided on internet for safer mode of payment.

5) Database Server:

A database server is a computer program that provides database services to other computer programs or computers, as defined by the client–server model[wiki]

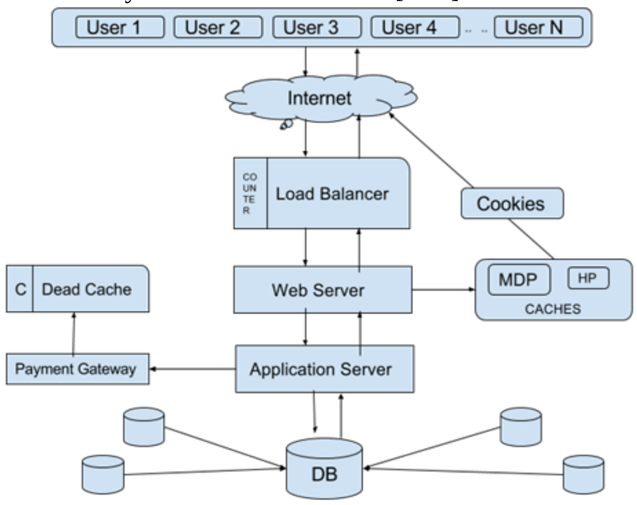


Fig. 2: Proposed System Architecture

6) Caches:

– MDP-Most Demanded Pages: In this cache recent most demanded page will be stored. Calculations will be done before assigning the caches to gain maximum hit ratio.
– HP- Home Page: This Home page cache will be created if the servers capacity and the capacity of MDP cache is exceeded. The user will be redirected to home page in such scenarios.
– Cookies: When the users request is redirected from the MDP cache, some information will be sent to users along with the page in form of cookies. This technique will allow reducing the calculation for next page on the server.
– Dead Cache: When user want to enter into the payment mode to payment gateway user goes into inactive state or dead state. In this protocol the server forwards the user session to the dead cache along with the counter. Advantage of this cache, the counter will count the users gone for payment and the server will take new users requests at mean time.

VI. TECHNICAL DETAILS

A. Methodology:

According to proposed architecture the system is divided into different components.

1) Scaling Method Flow:

– Distribute the server into multiple virtual servers to achieve clustering.
– Uses DNS load balancer to distribute the load on virtual servers.
– Appropriate Caching technique is used to create caches to reduce load.
– Data base is distributed into replica or fragments to avoid bottle neck and to reduce response time.

2) Description of Methodology

a) Load Balancing Algorithms used to handle requests on Servers.

DNS based Load Balancing Technique: DNS-based load balancing represents one of the early server load balancing approaches. The Internet’s domain name system (DNS) associates IP addresses with a host name. If you type a host name (as part of the URL) into your browser, the browser requests that the DNS server resolve the host name to an IP address. The DNS-based approach is based on the fact that DNS allows multiple IP addresses (real servers) to be assigned to one host name, as shown in the DNS lookup example in Listing. DNS is an efficient solution for global server load balancing, where load must be distributed between data centers at different locations. Often the DNS-based global server load balancing is combined with other server load balancing solutions to distribute the load within a dedicated data center. Algorithms: Randomized Distribution Round Robin Weighted Round Robin

3) A Local Server will be distributed into Virtual Servers.

a) Technique:
  – Set your hostnames or setup OS to recognize your local websites.
  – Create a folder for the website.
  – Setup Apache to serve multiple sites.

4) Creating Caches of Frequent used Data.

As stated in the book A Fresh Graduate’s Guide to Software Development Tools and Technologies caching of web paged is achieved by mathematical calculations by zipf’s principle.

a) Caching techniques:
  – Object Cache.
  – Memcached.
  – Reverse Proxy cache
  – Content Delivery Network.

5) Data fragmentation method is applied.

When particular piece of data is frequently accessed by the users at a particular site, it is more feasible to fragment that piece of information and store that copy of fragment at that site rather than storing the whole information including which is used very often. Thus the information can be retrieved more easily from that site, also the time required for retrieval decreases as there is less data stored to search for. This decreases the size of data store data each site and also speeds up the accessibility. Also it reduces the bottle neck attack.

a) Techniques:
  – Replication
    – Database replication using the master slave method.
    – Multi-master Replication model.
    – Replication delay and consistency.
  – Partitioning
    – Vertical
    – Horizontal

B. Project Requirements:

1) Software Requirements

– Technology	: PHP
– Web Technologies	: Html, JavaScript, CSS
– Database	: Mysql
– Web Server	: APACHE

2) Hardware Requirements

– Processor	: Intel
– RAM	: 1GB

VII. Market Potential

A. *Market Potential of Project:*

There is no proper solution in current market to overcome the scalability issue.

Other solutions are to scale vertical or horizontal which are time consuming, costly, and high maintenance operations

Another solution is cloud computing in which web servers are hired on timely basis.

Solution purposed in this paper is scaling web servers by managing the requests among the server in order to reduce the response time and thereby increasing the capacity of then server

B. *Competitive Advantage of Project-*

1) Previously web Scaling was to improve the hardware capacity or increasing the number of machines in web server.
2) This type of scaling is often costly and increases the maintenance when it is implemented.
3) This project provides software scaling in order to improve scalability
4) Software web Server scaling will provide reduce maintenance cost, reduce Hardware cost, easy maintainability.

VIII. Conclusion and Future Scope

A. *Conclusion:*

Scalability is an important constraint in this century as several businesses are implemented on web in order to spread it throughout the globe. We have presented a new approach and several techniques for scaling web server. The new technique differs from existing technique in that with the use of minimum caches and improving load balancing technique we can achieve high request handling capacity of a server. Cache operations are costly, so when the load on the server is less than its capacity this operations should be preserved, we divided the throughout architecture into two layers, this provides to reduce costly operations. This solution we purposed in paper will be optimal as it uses a bit of hardware that is cache and software in load balancer will be used in optimal and efficient web server scaling.

IX. Future Scope

– There is no proper solution in current market to overcome the scalability issue.
– Other solutions are to scale vertical or horizontal which are time consuming, costly, and high maintenance operations
– Another solution is cloud computing in which web servers are hired on timely basis.
– Solution purposed in this paper is scaling web servers by managing the requests among the server in order to reduce the response time and thereby increasing the capacity of then server

Acknowledgement

We would like to thanks our guide Prof. P.S. Lokhande for giving his valuable guidance who also provided expertise that greatly assisted the research.

References

[1] E. Kirda, M. Jazayeri, C. Kerer, and M. Schranz, “Experiences in Engineering Flexible Web Services,” IEEE MultiMedia, vol. 8, no. 1, pp. 58-65, Jan. 2001.
[2] J. Conallen, Building Web Applications with UML. Addison-Wesley, 2000.
[3] Gregor v. Bochmann,”Scalability of Web-Based Electronic Commerce Systems” July 2003.
[4] Daniel A. Menascé, “Scaling website through caching”, George Mason University menasce@cs.gmu.ed June 2003
[5] Sameena Naaz “Load Balancing Algorithms for Peer to Peer and Client Server Distributed Environments.” June 2012.
[6] [wiki] www.wikipedia.org
[7] [java world]www.javaworld.com

Chapter 10

Appendix I

10.1 Webserver Stress Tool

Most websites and web applications run smoothly and correctly as long as only one user (e.g. the original developer) or just a few users are visiting at a given time. But what happens if thousands of users access the website or web application at the same time?

Using Webserver Stress Tool you can simulate various load patterns for your webserver which will help you to find problems in your webserver set up. With steadily increasing loads (so called *ramp tests*) you are able to find out how much load you server can handle before serious problems arise.

10.1.1 Features of Web Stress Tool

- Webserver Stress Tool simulates anywhere from a few users to several hundred users accessing a website via HTTP/HTTPS at the same time.

- Based on a set of URLs or using a VBScript the software simulates independent users requesting webpages from that URL including images, frames etc.

- Each user is simulated by a separate thread with its own session information (e.g. cookies are stored individually for each user). URLs can be parameterized for each user and the sequence of URLs can be varied.

10.1.2 Webserver Stress Tool can be used for various tests

- Performance Tests are used to test each part of the webserver orthe web application to discover what parts, if any, are slow and how you can make them faster. Most often this is done by testing various implementations of single web pages/scripts to determine a configuration of code that is the fastest.
- Load Tests are performed by testing the website using the best estimate of the traffic your website must support. Consider this likea *real world* test of the website.
- Stress Tests are simulated âœbrute forceâ attacks that apply excessive load on your webserver. Real world situations like this can be created by a massive spike in users caused, innocently enough, by a new advertising campaign.
- Ramp Tests are used to determine the maximum threshold of users that can be served before error messages are produced.

10.1.3 Test result can be viewed as

Webserver webstress tool also provides serveral /ways to view results.

- Several eas.y to use graphs
- Summer Log.
- Detailed Log.
- Machine readable request Log(CSV).
- Raw graph data(CSV).

ACKNOWLEDGMENT

I would like to take the opportunity to express my sincere thanks to my guide **Prof. P.S. Lokhande**, Assistant Professor, Department of Computer Engineering, AIKTC, School of Engineering, Panvel for his invaluable support and guidance throughout my project research work. Without his kind guidance & support this was not possible.

I am grateful to him for his timely feedback which helped me track and schedule the process effectively. His time, ideas and encouragement that he gave is help me to complete my project efficiently.

I would also like to thank **Dr. Abdul Razak Honnutagi**, AIKTC, Panvel, for his encouragement and for providing an outstanding academic environment, also for providing the adequate facilities.

I am thankful to **Prof. Tabrez Khan**, HOD, Department of Computer Engineering, AIKTC, School of Engineering, Panvel and all my B.E. teachers for providing advice and valuable guidance.

I also extend my sincere thanks to all the faculty members and the non-teaching staff and friends for their cooperation.

Last but not the least, I am thankful to all my family members whose constant support and encouragement in every aspect helped me to complete my project.

Patil Amit Suresh Archana (12CO53)

Mirsinge Ibad Ibrahim Saba(12CO42)

Tulve Shabab Kasim Shagufta (12CO62.)

Siddique Asma Abdul Wahab Zaibunnisa (12CO14)
(Department of Computer Engineering)
University of Mumbai.