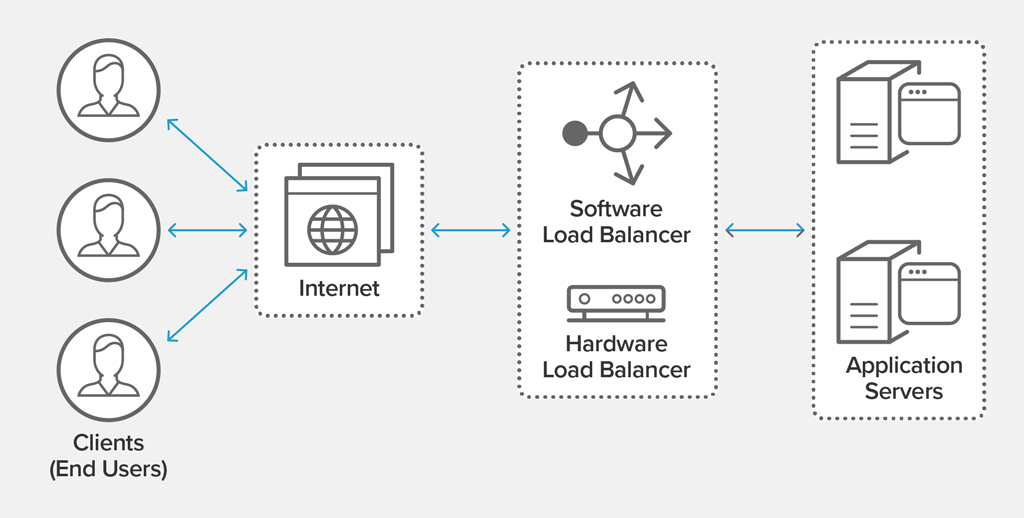
**Load balancing** refers to efficiently distributing incoming network traffic across a group of backend servers, also known as a *server farm* or *server pool*.

Modern high‑traffic websites must serve hundreds of thousands, if not millions, of concurrent requests from users or clients and return the correct text, images, video, or application data, all in a fast and reliable manner. To cost‑effectively scale to meet these high volumes, modern computing best practice generally requires adding more servers.

A [**load balancer**](https://www.nginx.com/solutions/adc) acts as the “traffic cop” sitting in front of your servers and routing client requests across all servers capable of fulfilling those requests in a manner that maximizes speed and capacity utilization and ensures that no one server is overworked, which could degrade performance. If a single server goes down, the load balancer redirects traffic to the remaining online servers. When a new server is added to the server group, the load balancer automatically starts to send requests to it.

In this manner, a load balancer performs the following functions:

* Distributes client requests or network load efficiently across multiple servers
* Ensures high availability and reliability by sending requests only to servers that are online
* Provides the flexibility to add or subtract servers as demand dictates



**Load Balancing Algorithms**

Different load balancing algorithms provide different benefits; the choice of load balancing method depends on your needs:

* **Round Robin** – Requests are distributed across the group of servers sequentially.
* **Least Connections** – A new request is sent to the server with the fewest current connections to clients. The relative computing capacity of each server is factored into determining which one has the least connections.
* **Least Time** – Sends requests to the server selected by a formula that combines the  
  fastest response time and fewest active connections. Exclusive to NGINX Plus.
* **Hash** – Distributes requests based on a key you define, such as the client IP address or  
  the request URL. NGINX Plus can optionally apply a consistent hash to minimize redistribution  
  of loads if the set of upstream servers changes.
* **IP Hash** – The IP address of the client is used to determine which server receives the request.
* **Random with Two Choices** – Picks two servers at random and sends the request to the  
  one that is selected by then applying the Least Connections algorithm (or for NGINX Plus  
  the Least Time algorithm, if so configured).

**Benefits of Load Balancing**

* Reduced downtime
* Scalable
* Redundancy
* Flexibility
* Efficiency

<https://www.tecmint.com/deploy-html-website-on-lamp-in-ubuntu/>

***XML AJAX:***

***https://www.w3schools.com/php/php\_ajax\_xml.asp***

<!DOCTYPE html>

<html>

<style>

table,th,td {

border : 1px solid black;

border-collapse: collapse;

}

th,td {

padding: 5px;

}

</style>

<body>

<h1>The XMLHttpRequest Object</h1>

<button type="button" onclick="loadDoc()">Get my CD collection</button>

<br><br>

<table id="demo"></table>

<script>

function loadDoc() {

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() {

if (this.readyState == 4 && this.status == 200) {

myFunction(this);

}

};

xhttp.open("GET", "cd\_catalog.xml", true);

xhttp.send();

}

function myFunction(xml) {

var i;

var xmlDoc = xml.responseXML;

var table="<tr><th>Artist</th><th>Title</th></tr>";

var x = xmlDoc.getElementsByTagName("CD");

for (i = 0; i <x.length; i++) {

table += "<tr><td>" +

x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +

"</td><td>" +

x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +

"</td></tr>";

}

document.getElementById("demo").innerHTML = table;

}

</script>

</body>

</html>