1. **INTRODUCTION**

Data permeates everything we do – that’s why we often hear data referred to as the lifeblood of businesses and organizations. However, it’s true that not all data is equal, but it’s also true that data is so critical we couldn’t live without it. That’s why numerous experts have put a dollar value on Data.

In this Era of Information Technology, when Data is in such an abundant form, when each and every day we are generating Millions of GBs of Data, this much data has to be managed very carefully and analyzed very critically so that this millions of GBs of data can be used crucially, Organizations are investing millions of dollars, if not billions of dollars to create, store, maintain and protect data. So, if we combine the value of data, the necessity of data, and the amount of money being spent on data, it’s easy to see how important data is, but if you can’t access that data within an appropriate timeframe based on the need, then organizations are spending a lot of money without gaining any real value. After all, what do we really gain when data is just sitting on storage? Nothing.

The real value of data is derived when it’s used. Furthermore, the speed and efficiency in accessing and obtaining that data is even more important as it significantly impacts the value of the data and one of the way to use it crucially is Data Repository System. There is no denying that data is invaluable. It underpins everything we do, especially being in the digital age. If anything happens to any of our servers we use backup copies to restore lost data.

We use data to make decisions; some simple and others life changing e.g. using data to develop cures for illnesses. We use data to bring to market new applications. We use data to enable us to move forward.  Some businesses are literally built on data – e.g. Facebook (worth $28 billion) and Twitter (worth $10 billion). Other businesses can easily attribute how data is adding to their bottom line.

Data Repository System is an Automised Dashboard System to analyze and visualize the Data and here specifically Human Development Indicators, which can be used by various organizations such as UNO (United Nations Organisation), WHO (World Health Organisation) and various local organisations to prepare their results out of these Human Development Indicators accordingly which is provided by the various Countries on this Dashboard System in an automised way. This system is used to create an automized flow of Data between the Data Provider and Data Consumer, and forming a Data Repository out of this Data so that Consumer can also analyze this Data via text or via a more systematic visual form which is also get notified and delivered on the System with the help of Web Services so that, that respective system can also manage that data accordingly.

This is a 3-way process in which there are 3-Entities: Data Provider, Data Repository Maintainer, and Data Consumer, with which the Data flows is the System.

1. Provider: This is the Data Provider which provides the Data to the Data Repository System according the measures set by the Data Maintainer. Data provider, to provide the data has to register with the System so to provide the Data to the system so that Consumer can consume the Data. Moreover, System will generate a notification on the Registered Provider’s System when the Data Consumer generates a request for the Data.
2. Maintainer: This is the Data Maintainer which maintains the flow of Data from the Provider to the Consumer. This Maintainer Entity sets the measures with which the Data provider provides the Data and the Data Consumer can consume the Data with the help of the Data Repository System. And also, this Maintainer takes the request from the Data Consumer to consume the Data and notify the Provider for the same. Similarly, when a Provider provides the Data to the System, this Maintainer will generate a notification on the Consumer’s System so that he can consume the provided Information.
3. Consumer: This is the Data Consumer consumes the Data, provided by the Data Provider to the System. This Entity constitutes various organizations which require Analyzed Data in the Visual form so as to work on it. For this, these various organizations have to register with the System so as to get the Analyzed, visualized Data from the Provider.
4. **DATA ENTITIES**
   1. **Data Provider**

Data Provider, the first entity in the three-entity communication system. From this data provider, the system communication and sharing of data starts taking place. This provides the data onto the system. Data provider, as an entity itself, firstly gets the input data from the providers (i.e. countries in this case) on the following parameters:

1. Indicators

2. Areas

3. Units

4. Subgroups

For inputting the data each country has a specific login with which it has to login onto the system and maintain the session, this has been done with node and express middleware, where after login a session is created for a specific country for a specific time, where no activity results in session termination. Then after login into the country’s portal one creates the data against the parameters, which will be available into the Datasets tab, where the country entity combines the data according to the data stored with respective parameters. With data input into the respective parameters, data provider also provides the option to edit and delete the input data in those respective parameters. When all the data is uploaded according to the parameters then dataset is created. Then that dataset in the datasets table gets shared across the Repository to the subscribed Consumers which consumes and analyze that dataset.

* 1. **Data Registry**

Data Registry, it is the second entity in the three-entity communication system. This entity is responsible for managing the system and the entities. It controls the communication between these entities. It uses a **Pub-Sub** design pattern, where the providers and consumers both registers themselves onto this entity, and with-it they also state their specifications. Consumers, after stating their specifications subscribes to the registered providers from where they need the data according to the specifications they stated.

After both the providers and subscribers are registered then, this middle entity send a request to both the entities i.e. provider and consumer so as to establish the link between the two so that communication and data can be shared between them. With this communication link, data is shared with the consumers (i.e. organizations) which has been put onto this system by the data providers. When the data has been put in by the providers on to the respective systems, data Registry fetches that data from the provider (or data is provided by the provider) and then that data is shared with the data consumer with a notification, that data has been put into the system by its subscribed provider. In this way consumers gets their subscribed providers data. This sharing via pub-sub, linking and the maintenance of the entities is the responsibility of the Data Registry.

* 1. **Data Consumer:**

This is the third entity in the three-entity Data communication system. This entity is responsible for the consuming of the data supplied onto the system by the subscribed registered data providers and then finally to analyze the system via the bar graphs. This entity is the last step of this whole process. In this it collects the data in the form of a Datasheet, where all the data is collected in a tabular form. And at the same there is this another option(charting) to visualize the data graphically by the means of bar graphs. Charting by the Bar graphs has been done by the D3.js which stands for basically Data Driven Documents which is basically used to represent the textual data into the visual and more of a graphical form as data in the graphical form is more easily readable and understandable.

For consumption of the data and finally analyzing of it, starts from where the consumer has to go onto the Data Registry entity and register itself onto the system and state it data specifications for which it needs data and also its URL to where it needs that data. And at the same time with specification and URL, consumer also has to subscribe to list of providers which have registered onto the Data Registry system. This completes the Subscription part of the Pub-Sub pattern.

1. **APPLICATIONS**

Data Repository technology can be used where there is a communication need between these or like these entities, to share data or to communicate with each other. This project links these three entities in the simplest way for the purpose of their communication and sharing of the data and to analyze the data further by the consuming entity. Thus, this can be applied to various applications from the simple to the complex ones.

Data Repository for Human Development Indicators provides easy and convenient access to real-time data generated by censuses conducted by governments in various nations, this data ranges from a country’s population to its literacy rate. These indicators are then used by organizations like United Nations(UN) or World Health Organization(WHO) to rank nations for lifestyle quality.

One such example is Human Development Index which is calculated on the basis of a country’s life expectancy for health, expected years of schooling, mean of years of schooling for education and Gross National Income per capita for standard of living by United Nations Development Program. Every year UNDP ranks countries based on the HDI report released in their annual report. HDI is one of the best tools to keep track of the level of development of a country, as it combines all major social and economic indicators that are responsible for economic development.

Access to this data is paramount and is made easily possible through Data Repository for Human Development Indicators.

Other than for Human Development Indices, it (Data Repository) can also be used to implement simpler system like to maintain the stock of goods and services at the consumer (i.e. shop owner) by the provider (i.e. supplier) with which both the entities are updated simultaneously, when something is consumed or in the same way when something is produced. With the help of the Data Repository, both the entities get to be updated all the time of their goods and services in the simplest manner with which they can fill their inventories timely.

**Uses in the Public Sector**

Groups in the public sector include all levels of government (those in it and those trying to get in it), as well as police, military, transportation agencies, and educational and healthcare facilities. Just a few folks, right? Oh, and let’s add philanthropy and philanthropic projects, a.k.a. charities, into the mix, just for fun.

All these organizations have a key interest in discovering what is happening (the data) and then conveying that information internally to others in their own group and/or externally to the broader public (the visual). Many such efforts are mandated and essential to the organization’s existence. Take, for example, the U.S. census. The data is collected on a massive scale every 10 year—by law—and then impacts multiple facets of American life such as state and regional funding and, of course, congressional representation. The U.S. Census Bureau maintains a treasure trove of the aggregate data, now visually accessible to everyone through its online presence at www.census.gov. Not only are there government-sanctioned representations of the collected census information but the site also makes APIs available (api.census.gov) for public web developer access.

**Business-to-Business and Intrabusiness Uses**

If the business of business is business, how do you do business? Mostly through marketing, whether you’re a vendor targeting another company or one department lobbying internally for increased resources. And the heart of marketing is persuasion—which is often bolstered, if not solely accomplished, by making your case through the compelling presentation of data.

As with the public sector, many such presentations are required. Look through any annual report to see the latest encapsulation of the company’s standing, graphically depicted in quickly graspable charts. Today, creating an online report is standard practice. Similar data visualizations are undertaken daily in department and division meetings to plot sales progress, reveal public reaction to products, and adjust business direction.

1. **TECHNOLOGIES**

* **D3.js:** D3.js is a JavaScript library for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG and CSS. D3’s emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.

D3.js helps you attach your data to DOM (Document Object Model) elements. Then you can use CSS3, HTML, and/or SVG showcase this data. Finally, you can make the data interactive through the use of D3.js data-driven transformations and transitions. This D3.js Framework will be used to generate visual representation to the Analysed Data on the Data Repository Dashboard which is provided by the Provider so that the consumer can analyse and consume the Data accordingly.

**Who develops D3.js**

[Mike Bostock](http://bost.ocks.org/mike/) wrote D3.js based on his work during his PhD studies at the [Stanford Visualization Group](http://vis.stanford.edu/). Mike worked at the [The New York Times](http://www.nytimes.com/) for a while and is now independently working on D3.js.

**Why use D3.js**

You should use D3.js because it lets you build the data visualization framework that you want. Graphic / Data Visualization frameworks make a great deal of decisions to make the framework easy to use. D3.js focuses on binding data to DOM elements.

D3.js is written in JavaScript and uses a functional style which means you can reuse code and add specific functions to your heart's content. Which means it is as powerful as you want to make it. How you chose to style, manipulate, and make interactive the data is up to you.

**When you should use D3.js**

You should use D3.js when your webpage is interacting with data. D3 stands for **D**ata **D**riven **D**ocuments. We will explore D3.js for it's graphing capabilities. We will not explore the myriad ways you can use data to in your webpages in non-graphical ways.

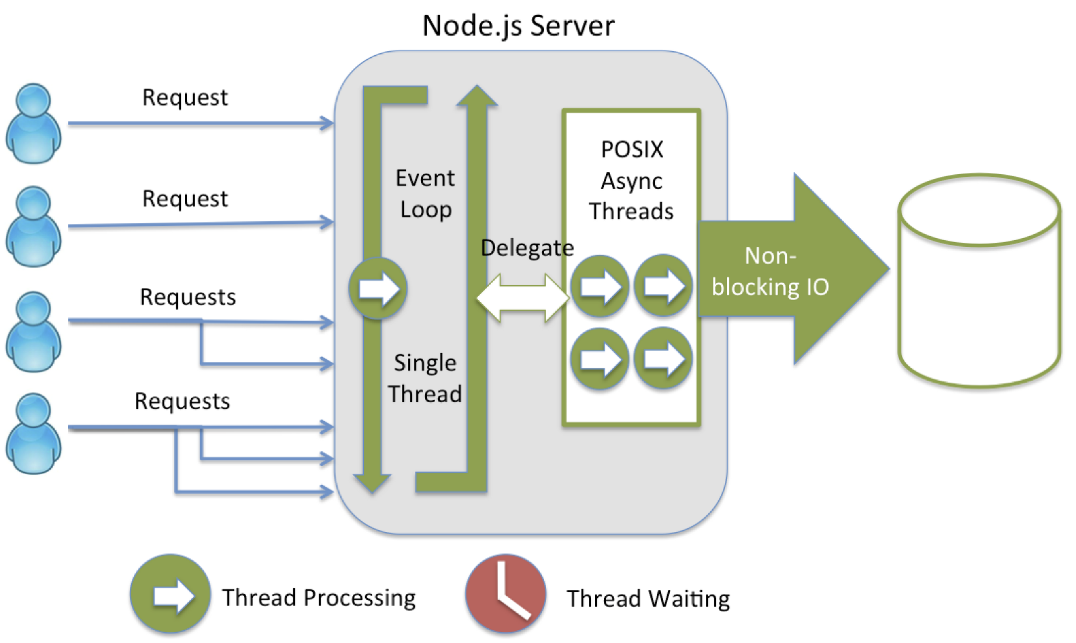
**Where is D3.js used**

D3.js is a javascript library added to the front-end of your web application. Your back-end (the server) will generate the necessary data. The part of the application the users interact with (the front-end) will use D3.js.

* **Node.js:** **Node.js** is an [open-source](https://en.wikipedia.org/wiki/Open-source_software), [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [JavaScript](https://en.wikipedia.org/wiki/JavaScript) [run-time environment](https://en.wikipedia.org/wiki/Runtime_system) that executes JavaScript code [server-side](https://en.wikipedia.org/wiki/Server-side). Historically, JavaScript was used primarily for [client-side scripting](https://en.wikipedia.org/wiki/Client-side_scripting), in which scripts written in JavaScript are embedded in a webpage's HTML and run client-side by a JavaScript engine in the user's web browser. Node.js lets developers use JavaScript for [server-side scripting](https://en.wikipedia.org/wiki/Server-side_scripting)—running scripts server-side to produce [dynamic web page](https://en.wikipedia.org/wiki/Dynamic_web_page) content *before* the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm

Node.js is a platform built on Chrome’s JavaScript Runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications.



Following are some of the prominent highlights that make Node.js the first choice of software developers specially backend.

#### ****1. It is asynchronous and event Driven**** −

All APIs of Node.js library are asynchronous, that is, **non-blocking**. It essentially means a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.

#### 2. Super Fast :

Being built on Google Chrome’s V8 JavaScript Engine, Node.js is super efficient and quick in code execution.

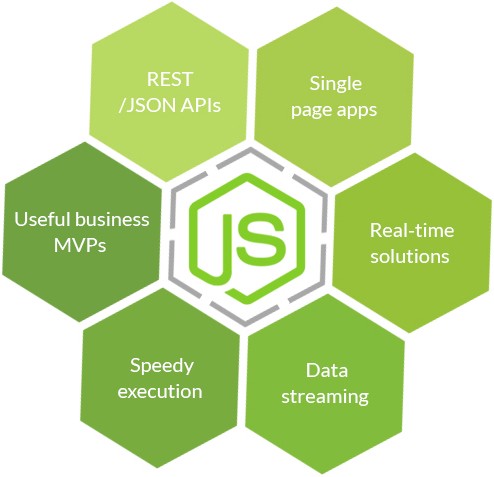
#### 3. Single Threaded but Highly Scalable :

Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers like **Apache** which create limited threads to handle requests.

#### 4. No Buffering :

Node.js applications never buffer any data. These applications simply output the data in chunks.

Because of above specifications some of the biggies like eBay, Paypal, Godaddy, General electronic , Yahoo etc.. relies on Node for their server side programming needs.



### Application of Node.js :

It is being employed in

* I/O bound Applications
* Data Streaming Applications
* Data Intensive Real-time Applications (DIRT)
* JSON APIs based Applications
* Single Page Applications
* **Express.js:** Express is a minimal and flexible Node.js web application framework that provides a robust set of features to develop web and mobile applications. It facilitates the rapid development of Node based Web applications. Following are some of the core features of Express framework

1. Allows to set up middleware to respond to HTTP Requests.
2. Defines a routing table which is used to perform different actions based on HTTP Method and URL.
3. Allows to dynamically rendering HTML Pages based on passing arguments to templates.

This Node framework with the help of Node is used to make the server side of the Dashboard, which is used to generate the notifications to the provider and the consumer’s system.

### EXPRESS.JS FRAMEWORK BASICS

* **Server-side web and mobile application framework**
* **Language**: written in JavaScript
* **Express builds**:
  + Single-page, multi-page, and hybrid mobile and web apps
  + Common back-end functions for web applications
  + APIs (application programming interfaces)
* **Templating engines**: Express comes with two templating engines, Jade and EJS, which facilitate the flow of data into a website structure.
* **MVC pattern**: Express supports the Model-View-Controller architecture, a really helpful way to build websites in a model-driven format.
* **Platform**: Node.js
* **Operating system**: It’s cross-platform, so it’s not limited to one OS.
* **The Express Generator** lets you create complex applications quickly.
* **MySQL:** A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.

Other kinds of data stores can also be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those type of systems.

Nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as **Foreign Keys**.

A **Relational DataBase Management System (RDBMS)** is a software that −

* Enables you to implement a database with tables, columns and indexes.
* Guarantees the Referential Integrity between rows of various tables.
* Updates the indexes automatically.
* Interprets an SQL query and combines information from various tables.
* MySQL is a fast, easy-to-use RDBMS being used for many small and

big businesses.

1. MySQL is released under an open-source license.
2. MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
3. MySQL uses a standard form of the well-known SQL data language.
4. MySQL is very friendly to PHP, the most appreciated language for web development.
5. MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).
6. MySQL is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

**Storage Engine Used:**

**InnoDB**is a general-purpose storage engine that balances high reliability and high performance. In MySQL 8.0, InnoDB is the default MySQL storage engine.

1. Its [DML](https://dev.mysql.com/doc/refman/8.0/en/glossary.html#glos_dml) operations follow the [ACID](https://dev.mysql.com/doc/refman/8.0/en/glossary.html#glos_acid) model, with [transactions](https://dev.mysql.com/doc/refman/8.0/en/glossary.html#glos_transaction) featuring [commit](https://dev.mysql.com/doc/refman/8.0/en/glossary.html#glos_commit), [rollback](https://dev.mysql.com/doc/refman/8.0/en/glossary.html#glos_rollback), and [crash-recovery](https://dev.mysql.com/doc/refman/8.0/en/glossary.html#glos_crash_recovery) capabilities to protect user data.
2. Row-level [locking](https://dev.mysql.com/doc/refman/8.0/en/glossary.html#glos_locking) and Oracle-style [consistent reads](https://dev.mysql.com/doc/refman/8.0/en/glossary.html#glos_consistent_read) increase multi-user concurrency and performance
3. InnoDB tables arrange your data on disk to optimize queries based on [primary keys](https://dev.mysql.com/doc/refman/8.0/en/glossary.html#glos_primary_key). Each InnoDB table has a primary key index called the [clustered index](https://dev.mysql.com/doc/refman/8.0/en/glossary.html#glos_clustered_index) that organizes the data to minimize I/O for primary key lookups

* **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**](http://www.yourhtmlsource.com/starthere/glossary.html#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.

The tags are what separate normal text from HTML code. You might know them as the words between the <angle-brackets>.Different tags will perform different functions. The tags themselves don’t appear when you view your page through a browser, but their effects do. The simplest tags do nothing more than apply formatting to some text, like this:

**<b>**These words will be bold**</b>**, and these will not.

In the example above, the <b> tags were wrapped around some text, and their effect will be that the contained text will be bolded when viewed through an ordinary web browser.

1. HTML stands for Hyper Text Mark-up Language
2. HTML describes the structure of Web pages using mark-up
3. HTML elements are the building blocks of HTML pages
4. HTML elements are represented by tags
5. HTML tags label pieces of content such as "heading", "paragraph", "table", and so on
6. Browsers do not display the HTML tags, but use them to render the content of the page

**CSS:** CSS stands for Cascading Style Sheets. CSS helps Web developers create a uniform look across several pages of a Web site. Instead of defining the style of each table and each block of text within a page's HTML, commonly used styles need to be defined only once in a CSS document. Once the style is defined in cascading style sheet, it can be used by any page that references the CSS file. Plus, CSS makes it easy to change styles across several pages at once. For example, a Web developer may want to increase the default text size from 10pt to 12pt for fifty pages of a Web site. If the pages all reference the same style sheet, the text size only needs to be changed on the style sheet and all the pages will show the larger text.

While CSS is great for creating text styles, it is helpful for formatting other aspects of Web page layout as well. For example, CSS can be used to define the cell padding of table cells, the style, thickness, and color of a table's border, and the padding around images or other objects. CSS gives Web developers more exact control over how Web pages will look than HTML does. This is why most Web pages today incorporate cascading style sheets.

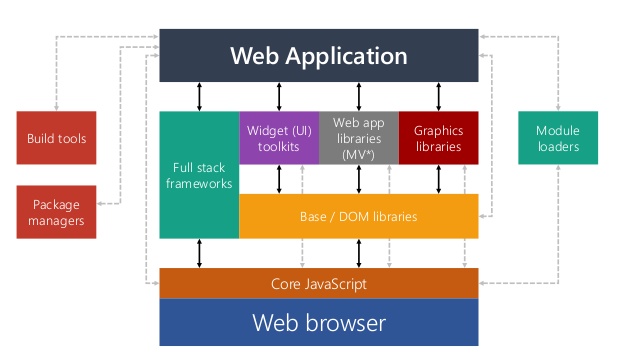
1. CSS describes how HTML elements are to be displayed on screen, paper, or in other media

1. CSS saves a lot of work. It can control the layout of multiple web pages all at once
2. External stylesheets are stored in CSS files

* **JAVASCRIPT:** **JavaScript**® (often shortened to **JS**) is a lightweight, interpreted, object-oriented language with [first-class functions](https://en.wikipedia.org/wiki/First-class_function), and is best known as the scripting language for Web pages, but it's [used in many non-browser environments](https://en.wikipedia.org/wiki/JavaScript#Uses_outside_web_pages) as well. It is a [prototype-based](https://en.wikipedia.org/wiki/Prototype-based_programming), multi-paradigm scripting language that is dynamic, and supports object-oriented, imperative, and functional programming styles.

JavaScript runs on the client side of the web, which can be used to design / program how the web pages behave on the occurrence of an event. JavaScript is an easy to learn and also powerful scripting language, widely used for controlling web page behaviour.

JavaScript can function as both a [procedural](https://en.wikipedia.org/wiki/Procedural_programming) and an [object oriented language](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript). Objects are created programmatically in JavaScript, by attaching methods and properties to otherwise empty objects **at run time**, as opposed to the syntactic class definitions common in compiled languages like C++ and Java. Once an object has been constructed it can be used as a blueprint (or prototype) for creating similar objects.



* **AJAX:** Ajax is not a programming language or a tool, but a concept. Ajax is a [client-side script](https://www.seguetech.com/client-side-scripting-performance/) that communicates to and from a server/database without the need for a [postback](http://www.c-sharpcorner.com/uploadfile/2f73dd/what-is-postback-in-Asp-Net/) or a complete page refresh. The best definition I’ve read for Ajax is “the method of exchanging data with a server, and updating parts of a web page – without reloading the entire page.” Ajax itself is mostly a generic term for various [JavaScript](https://www.seguetech.com/blog/2013/02/15/java-vs-javascript) techniques used to connect to a web server dynamically without necessarily loading multiple pages. In a more narrowly-defined sense, it refers to the use of [XmlHttpRequest](http://en.wikipedia.org/wiki/XMLHttpRequest) objects to interact with a web server dynamically via JavaScript.

## **Benefits of Ajax**

There are 4 main benefits of using Ajax in web applications:

1. **Callbacks:** Ajax is used to perform a callback, making a quick round trip to and from the server to retrieve and/or save data without posting the entire page back to the server. By not performing a full postback and sending all form data to the server, network utilization is minimized and quicker operations occur. In sites and locations with restricted bandwidth, this can greatly improve network performance. Most of the time, the data being sent to and from the server is minimal. By using callbacks, the server is not required to process all form elements. By sending only the necessary data, there is limited processing on the server. There is no need to process all form elements, process the ViewState, send images back to the client, or send a full page back to the client.
2. **Making Asynchronous Calls**: Ajax allows you to make asynchronous calls to a web server. This allows the client browser to avoid waiting for all data to arrive before allowing the user to act once more.
3. **User-Friendly:**Because a page postback is being eliminated, Ajax enabled applications will always be more responsive, faster and more user-friendly.
4. **Increased Speed:**The main purpose of Ajax is to improve the speed, performance and usability of a web application. A great example of Ajax is the movie rating feature on [Netflix](https://signup.netflix.com/MediaCenter). The user rates a movie and their personal rating for that movie will be saved to their database without waiting for the page to refresh or reload. These movie ratings are being saved to their database without posting the entire page back to the server.

