

REST API

REST (representational state transfer) is a very popular web API architecture. To be a REST API, an API must adhere to certain architectural constraints, or principles, including:

* **Client-server architecture :** the interface is separated from the backend and data storage. This allows for flexibility, and for different components to evolve independent of each other.
* **Statelessness:** no client context is stored on the server between requests.
* **Cacheability:** clients can cache responses, so a REST API response must explicitly state whether it can be cached or not.
* **Layered system:** the API will work whether it is communicating directly with a server, or through an intermediary such as a load balancer.

The REST architecture uses the original specifications of the HTTP protocol.

**Rule # 1:** the URI as a resource identifier  
**Rule # 2:** HTTP verbs as the identifier of operations  
**Rule # 3:** HTTP responses as a representation of resources  
**Rule # 4:** links as a relationship between resources  
**Rule # 5:** a parameter like authentication token

…………………………………………..

# HTTP Methods

Defines a set of query methods that indicate the action that we want to perform on the indicated resource. Although there are also names, these methods are often called HTTP verbs. Each of them implements a different semantics but some common functionalities can be shared by different methods

**GET:** The GET method requests a representation of the specified resource. GET requests should only be used to retrieve data.  
**HEAD:** The HEAD method requests a response identical to a GET request for which we have omitted the body of the response (we only have the header).  
**POST:** The POST method is used to send an entity to the indicated resource. This usually results in a change of state or side effects on the server.  
**PUT:** The PUT method replaces all current representations of the resource targeted by the content of the request  
**DELETE:** The DELETE method deletes the specified resource.  
**CONNECT:** The CONNECT method establishes a tunnel to the server identified by the target resource.  
**OPTIONS:** The OPTIONS method is used to describe the communication options with the targeted resource.  
**TRACE:** The TRACE method performs a round trip test message by following the path of the targeted resource.  
**PATCH:** The PATCH method is used to apply partial modifications to a resource.

…………………………………………………………………………………………………………..

# Definition of a database

Database is a systematic collection of data.  
Databases support storage and manipulation of data.  
Databases make data management easy.

Let's consider the Facebook as an example, it needs to store, manipulate and present data related to members, their friends, member activities, messages, advertisements and lot more. If we store all these data in files, it will be impossible to manipulate it.  
So here it comes the role of database

……………………………………………………………………………………………….

Why do we need a Database ?

Today, companies have to manage huge volumes of data, while ensuring data integrity, security and accuracy.  
Spreadsheets are great tools for making calculations. But if there is a lot of data, such as customers, employees, or inventory for a business, a more efficient data management tool could help speed up the business and be more efficient.  
Here's why we should consider replacing spreadsheets with databases to help your business grow.

Types of databases:

There are many different types of databases. The best database for a specific organization depends on how the organization intends to use the data.  
These are the database types:

* **Relational databases.**
* **Object-oriented databases.**
* **Distributed databases.**
* **Data warehouses.**
* **NoSQL databases.**
* **Graph databases.**
* **OLTP databases.**

Database history :

The history of databases dates back to the 1960s, with the appearance of network databases and hierarchical databases. In the 1980s, it was object-oriented databases that appeared. Today, the predominant databases are SQL, NoSQL and cloud databases.

In this skill, we’ll talk more about SQL and NoSQL databases.

If you want to read more  
<https://medium.com/@rpolding/databases-evolution-and-change-29b8abe9df3e>

…………………………………………………………………………………………….

# What is SQL?

SQL stands for "Structured Query Language".  
It's a language that allows communication with databases in order to manage all the data they contain.

### **Is SQL a Programming Language?**

* Yes, SQL is a language. It offers looping, logic directives, variables, and so on. However, it’s not a language in the same sense as say Java or C++.
* SQL may not be a language compared with Java or C#, but it is a language however.
* According to W3Schools: “SQL is a standard language for storing, manipulating and retrieving data in databases.”

# How to use SQL?

While an application might be programmed in a language like Python, PHP or Ruby, databases are not configured to understand these. Historically, databases understand only SQL (though this has changed significantly in recent years).  
Like other programming languages, SQL has its own markup. This makes it necessary for a programmer to learn SQL markup before using it effectively.  
Besides markup, another feature unique to database programming is the concept of tables. A database may be represented as a number of tables. Each table has its own number of columns and rows and represents a set of data.

…………………………………………………………………………………………..

Example of SQL database manipulation

Imagine a library. We could create a database that stores data about students in the GoMyCode. In this case, we would need only one table, this table would allow us to store all the information we need.  
There are a few frequently used SQL commands you should be familiar with for database work. When working with databases, a programmer might write commands such as:

* **CREATE DATABASE** to create a database.
* **CREATE TABLE** to create tables.
* **SELECT** to find/extract some data from a database.
* **UPDATE** to make adjustments and edit data.
* **DELETE** to delete some data.

These are just the most common commands. The more complicated the database is, the more commands you as the programmer will need to use.

* Example of database table :

Examples of NoSQL databases

Many NoSQL databases were designed by technology companies like Google, Amazon, Yahoo, and Facebook to provide more effective ways to store content or process data for huge websites. Some of the most popular NoSQL databases include the following:

* **Apache CouchDB**  
  An open source, JSON document-based database that uses JavaScript as its query language.
* **Apache Cassandra**  
  An open source, wide-column store database designed to manage large amounts of data across multiple servers and clustering that spans multiple data centers.
* **MongoDB**  
  An open source document-based database that uses JSON-like documents and schema, and is the database component of the MERN stack.
* **Redis**  
  A powerful in-memory key value store used for session caching, message queues, and other specific applications.
* **Elasticsearch**  
  A document-based database that includes a full-text search engine.

…………………………………………………………………………………………….

What’s the difference between SQL and NoSQL?

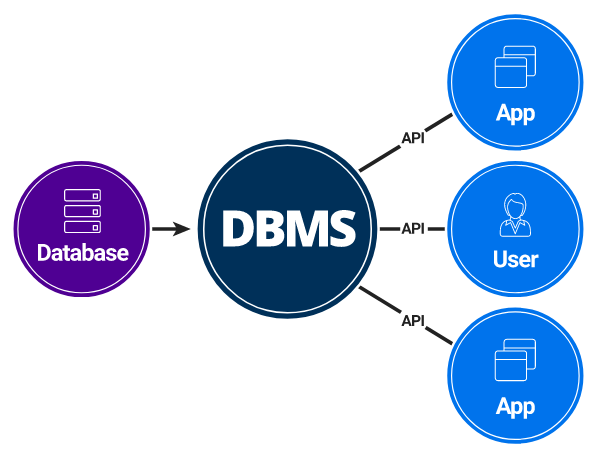
The fundamental difference between SQL and NoSQL is not all that complicated. Each has a different philosophy for how data should be stored and retrieved.

1. SQL databases are relational, NoSQL are non-relational.
2. SQL databases use structured query language and have a predefined schema. NoSQL databases have dynamic schemas for unstructured data.
3. SQL databases are vertically scalable, NoSQL databases are horizontally scalable.
4. SQL databases are table based, while NoSQL databases are document, key-value, graph or wide-column stores.
5. SQL databases are better for multi-row transactions, NoSQL are better for unstructured data like documents or JSON.

…………………………………………………………………………………

# Database Management System

A database typically requires a comprehensive database software program known as a database management system (DBMS).  
A DBMS serves as an interface between the database and its end users or programs, allowing users to retrieve, update, and manage how the information is organized and optimized. A DBMS also facilitates oversight and control of databases, enabling a variety of administrative operations such as performance monitoring, tuning, and backup and recovery.



## **Using DBMS allows us to:**

* Store a large volume of data for a long time and with security.
* Access data efficiently.
* Control access to data by multiple users at the same time. One user's action should not affect another.
* Create new databases as well as to specify their schemas.
* Support for a data model.
* Query and modify the database data.
* Have data redundancy.
* Have more flexibility than files.
* Have data consistency and integrity.

## **The most known DBMS:**

* **MySQL**  
  A free and open source DBMS. It is probably the best known DBMS. We will use it in this part.
* **PostgreSQL**  
  A free and open source like MySQL, with more functionality but a little less known.
* **SQLite**  
  A free and open source, very light but very limited in functionality;
* **Oracle**  
  A not free database, but it's used by very large companies; undoubtedly one of the most complete DBMS
* **Microsoft SQL Server**  
  Microsoft's DBMS. Microsoft markets at least a dozen different editions of Microsoft SQL Server, aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing applications with many concurrent users.

………………………………………………………………………………………………………………

The most common types of NoSQL databases are key-value, document, column and graph databases.

What is web development?

Web development is the process of building websites and applications for the Internet, or for a private network known as an Intranet.  
Web development is not concerned with the design of a website; rather, it’s all about the coding and programming that powers the website’s functionality.  
From the most simple, static web pages to social media platforms and applications, from eCommerce websites to content management systems (CMS); all the tools we use via the Internet on a daily basis have been built by web developers.

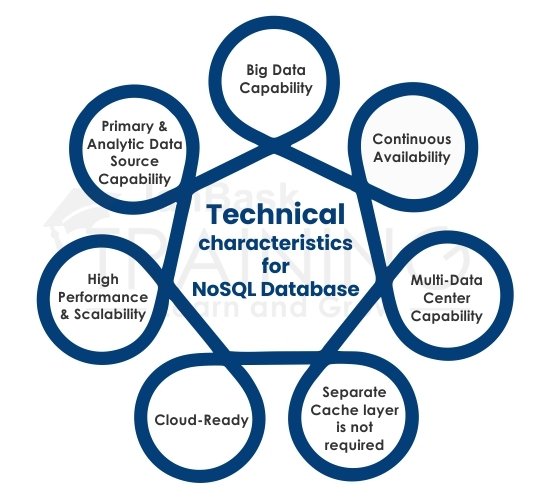
Web development can be broken down into three layers:

* Client-side coding (frontend)
* Cerver-side coding (backend)
* Database technology.

# Skills required to become a web developer?

What does it take to become a web developer? Essentially, just three things: HTML, CSS and JavaScript, the three pillars of the web, which we’ll be learning about over the next few weeks.  
Together, these three pillars make every website work, defining the content to be displayed (HTML), telling a browser how to display that content (CSS), and making the content interactive (JavaScript), respectively.  
A web developer is well versed in these three technologies. They can read other people’s code and make changes to it. They can find and debug bugs (shortcomings in existing code).  
A web developer might, at times, work on a new project (a new website) from scratch, or may have to work on an existing website and make it better. A typical day in the life of a web developer involves fixing bugs, developing new features (that is, enhancements) and webpages, and working with other developers to discuss ways to solve problems.

Don’t be overwhelmed by all of these details. You’ll soon see that they’re all very connected, and learning one of these automatically makes you good at a few others!



Examples of NoSQL databases

Many NoSQL databases were designed by technology companies like Google, Amazon, Yahoo, and Facebook to provide more effective ways to store content or process data for huge websites. Some of the most popular NoSQL databases include the following:

* **Apache CouchDB**  
  An open source, JSON document-based database that uses JavaScript as its query language.
* **Apache Cassandra**  
  An open source, wide-column store database designed to manage large amounts of data across multiple servers and clustering that spans multiple data centers.
* **MongoDB**  
  An open source document-based database that uses JSON-like documents and schema, and is the database component of the MERN stack.
* **Redis**  
  A powerful in-memory key value store used for session caching, message queues, and other specific applications.
* **Elasticsearch**  
  A document-based database that includes a full-text search engine.

VS : a popular code editor. A code editor is a very aptly named piece of software; it helps us edit text. It is similar to the default text editor that comes preinstalled on Mac or Windows, but has additional features like code highlighting to enhance our coding and debugging experience.

RECAP :

is a popular code editor. A code editor is a very aptly named piece of software; it helps us edit text. It is similar to the default text editor that comes preinstalled on Mac or Windows, but has additional features like code highlighting to enhance our coding and debugging experience.