**PROJECT 3: MERN STACK IMPLEMENTATION**

**MERN** Web stack consists of following components:

1. [**M**ongoDB](https://www.mongodb.com): A document-based, No-SQL database used to store application data in a form of documents.
2. [**E**xpressJS](https://expressjs.com): A server side Web Application framework for Node.js.
3. [**R**eactJS](https://reactjs.org): A frontend framework developed by Facebook. It is based on [JavaScript](https://www.javascript.com), used to build User Interface (UI) components.
4. [**N**ode.js](https://nodejs.org/en/): A JavaScript runtime environment. It is used to run JavaScript on a machine rather than in a browser.

**Text

Description automatically generated with medium confidence**

As shown on the illustration above, a user interacts with the ReactJS UI components at the application front-end residing in the browser. This frontend is served by the application backend residing in a server, through ExpressJS running on top of NodeJS.

Any interaction that causes a data change request is sent to the NodeJS based Express server, which grabs data from the MongoDB database if required, and returns the data to the frontend of the application, which is then presented to the user.

#### Side Self Study

1. Make a research what types of [Database Management Systems (DBMS) exist and what each type is more suitable for](https://www.alooma.com/blog/types-of-modern-databases). Be able to explain the difference between Relational DBMS and NoSQL (of a different kind).

# From: <https://www.techtarget.com/searchdatamanagement/definition/database-management-system>

**What does a DBMS do?**

The DBMS manages the data; the database engine allows data to be accessed, locked and modified; and the database [schema](https://www.techtarget.com/searchdatamanagement/definition/schema) defines the database's logical structure. These three foundational elements help provide concurrency, security, data integrity and uniform data administration procedures. The DBMS supports many typical database administration tasks, including change management, performance monitoring and tuning, security, and backup and recovery. Most database management systems are also responsible for automated rollbacks and restarts as well as logging and auditing of activity in databases and the applications that access them.

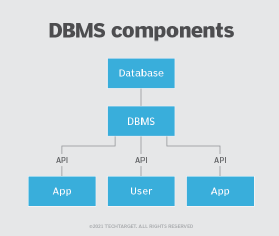
The DBMS provides a centralized view of data that can be accessed by multiple users from multiple locations in a controlled manner. A DBMS can limit what data end users see and how they view the data, providing many views of a single database schema. End users and software programs are free from having to understand where the data is physically located or on what type of storage medium it resides because the DBMS handles all requests.

The DBMS can offer both logical and physical data independence to protect users and applications from having to know where data is stored or from being concerned about changes to the physical structure of data. So long as programs use the application programming interface (API) for the database that the DBMS provides, developers won't have to modify programs just because changes have been made to the database.

In a relational database management system ([RDBMS](https://www.techtarget.com/searchdatamanagement/definition/RDBMS-relational-database-management-system)) -- the most widely used type of DBMS -- the API is SQL, a standard programming language for defining, protecting and accessing data.

### What are the components of a DBMS?

A DBMS is a sophisticated piece of system software consisting of multiple integrated components that deliver a consistent, managed environment for creating, accessing and modifying data in databases. These components include the following:



Structure of database management system

* **Storage engine.** This basic element of a DBMS is used to store data. The DBMS must interface with a file system at the operating system ([OS](https://www.techtarget.com/whatis/definition/operating-system-OS)) level to store data. It can use additional components to store data or interface with the actual data at the file system level.
* **Metadata catalog.** Sometimes called a system catalog or database dictionary, a metadata catalog functions as a repository for all the database objects that have been created. When databases and other objects are created, the DBMS automatically [registers information about them in the metadata catalog](https://www.techtarget.com/searchdatamanagement/feature/How-automated-metadata-management-improves-business-insights). The DBMS uses this catalog to verify user requests for data, and users can query the catalog for information about the database structures that exist in the DBMS. The metadata catalog can include information about database objects, schemas, programs, security, performance, communication and other environmental details about the databases it manages.
* **Database access language.** The DBMS also must provide an API to access the data, typically in the form of a database access language to access and modify data but may also be used to create database objects and [secure and authorize access to the data](https://www.techtarget.com/searchsecurity/tip/Evaluate-cloud-database-security-controls-best-practices). SQL is an example of a database access language and encompasses several sets of commands, including Data Control Language for authorizing data access, Data Definition Language for defining database structures and Data Manipulation Language for reading and modifying data.
* **Optimization engine.** A DBMS may also provide an optimization engine, which is used to parse database access language requests and turn them into actionable commands for accessing and modifying data.
* **Query processor.** After a query is optimized, the DBMS must provide a means for running the query and returning the results.
* **Lock manager.** This crucial component of the DBMS manages concurrent access to the same data. Locks are required to ensure multiple users aren't trying to modify the same data simultaneously.
* **Log manager.** The DBMS records all changes made to data managed by the DBMS. The record of changes is known as the log, and the [log manager](https://www.techtarget.com/searchitoperations/definition/log-management) component of the DBMS is used to ensure that log records are made efficiently and accurately. The DBMS uses the log manager during shutdown and startup to ensure data integrity, and it interfaces with database utilities to create backups and run recoveries.
* **Data utilities.** A DBMS also provides a set of utilities for managing and controlling database activities. Examples of database utilities include reorganization, runstats, backup and copy, recover, integrity check, load data, unload data and repair database.

### Popular types and examples of DBMS technologies

Popular database models and management systems include RDBMS, NoSQL DBMS, NewSQL DBMS, in-memory DBMS, columnar DBMS, multimodel DBMS and cloud DBMS.

**RDBMS.** Sometimes referred to as a SQL DBMS and adaptable to most use cases, RDBMS presents data as rows in tables with a fixed schema and relationships defined by values in key columns. RDBMS [Tier-1](https://www.techtarget.com/searchitchannel/definition/tier-1-vendor) products can be quite expensive, but there are high quality, open source options such as PostgreSQL that can be cost-effective. Other examples of popular RDBMS products include Oracle, [MySQL](https://www.techtarget.com/searchoracle/definition/MySQL), Microsoft [SQL Server](https://www.techtarget.com/searchdatamanagement/definition/SQL-Server) and IBM [Db2](https://www.techtarget.com/searchdatamanagement/definition/Db2).

**NoSQL DBMS.** Well-suited for loosely defined data structures that may evolve over time, [NoSQL DBMS](https://www.techtarget.com/searchdatamanagement/definition/NoSQL-Not-Only-SQL) may require more application involvement for schema management. There are four types of NoSQL database systems: document databases, graph databases, key-value stores and wide-column stores. Each uses a different type of data model, resulting in [significant differences between each NoSQL type](https://www.techtarget.com/searchcloudcomputing/tip/Compare-NoSQL-database-types-in-the-cloud).

* Document databases store semi-structured data and descriptions of that data in document format, usually JavaScript Object Notation ([JSON](https://www.theserverside.com/definition/JSON-Javascript-Object-Notation)). They're useful for flexible schema requirements such as those common with content management and mobile applications. Popular document databases include MongoDB and Couchbase.
* Graph databases organize data as nodes and relationships instead of tables or documents. Because it stores the relationship between nodes, the graph system can support richer representations of data relationships. The graph data model doesn't rely on a strict schema, and it can evolve over time. [Graph databases](https://www.techtarget.com/whatis/definition/graph-database) are useful for applications that map relationships, such as social media platforms, reservation systems or customer relationship management. Examples of popular graph databases include Neo4j and GraphDB.
* Key-value stores are based on a simple data model that pairs a unique key with an associated value. Due to this simplicity, key-value stores can be used to develop highly scalable and performant applications such as those for session management and caching in web applications or for managing shopping cart details for online buyers. Examples of popular key-value databases include Redis and Memcached.
* Wide-column stores use the familiar tables, columns and rows of [relational database systems](https://www.techtarget.com/searchdatamanagement/answer/Definition-of-primary-super-foreign-and-candidate-key-in-the-DBMS), but column names and formatting can differ from row to row in a single table. Each column is also stored separately on disk. As opposed to traditional row-orientated storage, a wide-column store is optimal when querying data by columns, such as in recommendation engines, catalogs, fraud detection and event logging. Cassandra and HBase are examples of wide-column stores.

**NewSQL DBMS.** Modern relational systems that use SQL, [NewSQL](https://www.techtarget.com/searchdatamanagement/definition/NewSQL) database systems offer the same scalable performance as NoSQL systems. But NewSQL systems also provide ACID (atomicity, consistency, isolation and durability) support for data consistency. A NewSQL DBMS is engineered as a relational, SQL database system with a distributed, fault-tolerant architecture. Other typical features of NewSQL system offerings include in-memory capability and clustered database services with the ability to be deployed in the cloud. Many NewSQL DBMS packages have fewer features and components and a smaller footprint than legacy relational offerings, making them easier to support and understand. Some vendors now eschew the NewSQL label and describe their technologies as distributed SQL databases. CockroachDB, Google Cloud Spanner, NuoDB, Volt Active Data and YugabyteDB are examples of database systems in this category.

**IMDBMS.** An [in-memory database management system](https://www.techtarget.com/searchdatamanagement/definition/in-memory-database-management-system-IMDBMS) predominantly relies on [main memory](https://www.techtarget.com/searchstorage/definition/primary-storage)for data [storage](https://www.techtarget.com/searchstorage/definition/storage), management and manipulation. By reducing the latency associated with reading from disk, an IMDBMS can provide faster response times and better performance but can consume more resources. Therefore, an in-memory database is ideal for applications that require high performance and rapid access to data, such as data stores that support real-time HTAP (hybrid transactional and analytical process). Any type of DBMS (relational, NoSQL, etc.) can also support in-memory processing. SAP HANA and Redis are examples of in-memory database systems.

**CDBMS.** A columnar database management system stores data in tables focused on columns instead of rows, resulting in more efficient data access when only a subset of columns is required. It's well-suited for [data warehouses](https://www.techtarget.com/searchdatamanagement/definition/data-warehouse) that have a large number of similar data items. Popular columnar database products include Snowflake and Amazon Redshift.

**Multimodel DBMS.** This system supports more than one database model. Users can choose the model most appropriate for their application requirements without having to switch to a different DBMS. For example, IBM Db2 is a relational DBMS, but it also offers a columnar option. Many of the most popular database systems similarly qualify as multimodel through add-ons, including [Oracle](https://www.techtarget.com/whatis/definition/Oracle-Cloud), PostgreSQL and MongoDB. Other products, such as Azure Cosmos DB and MarkLogic, were developed specifically as multimodel databases.

**Cloud DBMS.** Built in and accessed through the cloud**,** the DBMS may be any type (relational, NoSQL, etc.) and a conventional system that's deployed and managed by a user organization or a managed service provided by the database vendor. Popular cloud services that enable [cloud database](https://www.techtarget.com/searchcloudcomputing/definition/cloud-database) implementation include [Microsoft Azure, Google Cloud and AWS](https://www.techtarget.com/searchdatamanagement/tip/Cloud-database-comparison-AWS-Microsoft-Google-and-Oracle).

### Benefits of using a DBMS

One of the biggest advantages of using a DBMS is that it lets users and application programmers access and use the same data concurrently while managing data integrity. Data is better protected and maintained when it can be shared using a DBMS instead of creating new iterations of the same data stored in new files for every new application. The DBMS provides a central store of data that multiple users can access in a controlled manner.

Central storage and management of data within the DBMS provide the following:

* data abstraction and independence;
* [data security](https://www.techtarget.com/searchsecurity/feature/Top-7-types-of-data-security-technology);
* a locking mechanism for concurrent access;
* an efficient handler to balance the needs of multiple applications using the same data;
* the ability to swiftly recover from crashes and errors;
* strong [data integrity](https://www.techtarget.com/searchdatamanagement/feature/Maintaining-data-integrity-key-for-data-quality) capabilities;
* logging and auditing of activity;
* simple access using a standard API; and
* uniform administration procedures for data.

Another advantage of a DBMS is that database administrators ([DBAs](https://www.techtarget.com/searchdatamanagement/definition/database-administrator)) can use it to impose a logical, structured organization on the data. A DBMS delivers economy of scale for processing large amounts of data because it's optimized for such operations.

A DBMS can also provide many views of a single database schema. A view defines what data the user sees and how that user sees the data. The DBMS provides a level of abstraction between the conceptual schema that defines the logical structure of the database and the physical schema that describes the files, indexes and other physical mechanisms the database uses. A DBMS enables users to modify systems much more easily when business requirements change. A DBA can add new categories of data to the database without disrupting the existing system, thereby insulating applications from how data is structured and stored.

However, a DBMS must perform additional work to provide these advantages, thereby incurring overhead. A DBMS will use more memory and CPU than a simple file storage system, and different types of DBMSes will require different types and levels of system resources.

### Drawbacks of DBMSes

Perhaps the single biggest drawback is the cost of the hardware, software and personnel required to run an enterprise DBMS, such as SQL Server, Oracle or IBM Db2. The hardware is usually a high-end server with a significant amount of memory configured, coupled with large disk arrays to store the data. The software includes the DBMS itself, which is pricey, as well as tools for programming and testing and for DBAs to enable management, tuning and administration.

From a personnel perspective, using a DBMS requires hiring a DBA staff, training developers in the proper usage of the DBMS, and possibly hiring additional systems programmers for managing installation and [integrating the DBMS into the IT infrastructure](https://www.techtarget.com/searchdatamanagement/feature/5-data-integration-challenges-and-how-to-overcome-them). Dealing with additional complexity is also a concern when implementing a DBMS.

The DBMS software is complex and requires in-depth knowledge to properly implement and manage. But the DBMS interfaces with many other IT components, such as the OS, transaction processing systems, programming languages and networking software. Ensuring the proper configuration and efficiency of such a complicated setup can be difficult and cause performance slowdowns or even system outages.

Some of the cost and administrative overhead of running enterprise database systems can be [alleviated by the cloud computing model](https://www.techtarget.com/searchdatacenter/feature/Hybrid-cloud-strategy-blends-public-private-cloud-benefits). For example, the cloud service provider (CSP) installs and manages the hardware, which can be shared across cloud users. Furthermore, storage, memory and other resources can be scaled up and down as required based on usage needs. And basic DBA tasks like patching and simple backups become the responsibility of the CSP. Therefore, it can be easier and more cost-effective for some [databases to be deployed in the cloud instead of on-premises](https://www.techtarget.com/searchdatamanagement/feature/Should-you-host-your-database-on-site-or-in-the-cloud).

### DBMS use cases

Enterprises that need to store data and access it later to conduct business have a viable use case for [deploying a DBMS](https://www.techtarget.com/searchdatamanagement/feature/Building-a-database-application-the-DIY-way). Any application requiring a large amount of data that needs to be accessed by multiple users or customers is a candidate for using a DBMS. Most medium to large organizations can benefit from using a DBMS because they have more data-sharing and concurrency needs and are able to more readily overcome cost and complexity issues. Sample customer use cases for DBMS technology include the following:

* Applications can include storing customer and account information, tracking account transactions such as withdrawals and deposits, and tracking loan payments. ATMs are a good example of a banking system that relies on a DBMS to track and manage that activity.
* DBMSes manage sales for any type of business, including storing product, customer and salesperson information and recording the sale, tracking fulfillment and maintaining sales history information.
* Most commercial airlines rely on a DBMS for data-intensive applications such as scheduling flight plans and managing customer flight reservations.
* Manufacturing companies depend on a DBMS to [track and manage inventory in warehouses](https://www.techtarget.com/searchdatamanagement/feature/Evaluating-your-need-for-a-data-warehouse-platform). A DBMS can also be used to manage data for supply chain management applications that track the flow of goods and services, including the movement and storage of raw materials, work-in-process inventory and finished goods from the point of origin to the point of consumption.
* A DBMS also makes it easier for a company to track and manage employee information in a human resources management application, including managing employee data such as addresses, phone numbers, salary details, payroll and paycheck generation.

### Changes in how DBMSes are built, sold and serviced

By 2019, [open source DBMS technologies](https://www.techtarget.com/searchdatamanagement/feature/Open-source-database-comparison-to-choose-the-right-tool) were rapidly gaining traction. In fact, Gartner projected that open source databases would account for 10% of total spending on database software for that year due to increased enterprise adoption. By 2022, three of the top five databases [ranked](https://db-engines.com/en/) by DB-Engines were open source. Most mainstream IT organizations use open source software in some of their mission-critical operations. This trend complements two others: acquisitions of open source database vendors by bigger rivals and the expansion of the cloud-based database service market.

In 2019, Gartner also said that cloud databases were driving most of the growth in the DBMS market, describing the cloud as "the default platform for managing data." In 2021, Gartner concluded that "by 2022, cloud database management system revenue will account for 50% of the total DBMS market revenue." In connection with the increasing shift toward the cloud, numerous DBMS vendors have introduced managed cloud database services that offer to free IT and data management teams from many of the tasks required to deploy, configure and administer database systems.

Another growing trend is what Gartner refers to as HTAP -- using a single DBMS to deliver both transaction processing and analytics without requiring a separate DBMS for each operation. To support this trend, more DBMS vendors are creating hybrid database systems that deliver multiple database engines within a single DBMS. Most hybrid DBMSes provide a combination of relational and multiple NoSQL engines and APIs. Examples include Altibase, Microsoft's Azure Cosmos DB and DataStax Enterprise.

# From: <https://www.guru99.com/difference-dbms-vs-rdbms.html>

# Difference between DBMS and RDBMS

In this RDBMS vs DBMS tutorial, we will learn about main RDBMS and DBMS difference. But before that, let’s learn:

## What is DBMS?

A DBMS is a software used to store and manage data. The DBMS was introduced during 1960’s to store any data. It also offers manipulation of the data like insertion, deletion, and updating of the data.

DBMS system also performs the functions like defining, creating, revising and controlling the database. It is specially designed to create and maintain data and enable the individual business application to extract the desired data.

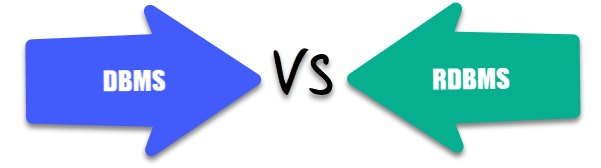
## What is RDBMS?

Relational Database Management System (RDBMS) is an advanced version of a DBMS system. It came into existence during 1970’s. RDBMS system also allows the organization to access data more efficiently then DBMS.

RDBMS is a software system which is used to store only data which need to be stored in the form of tables. In this kind of system, data is managed and stored in rows and columns which is known as tuples and attributes. RDBMS is a powerful data management system and is widely used across the world.

## KEY DIFFERENCE

* DBMS stores data as a file whereas in RDBMS, data is stored in the form of tables.
* DBMS supports single users, while RDBMS supports multiple users.
* DBMS does not support client-server architecture but RDBMS supports client-server architecture.
* DBMS has low software and hardware requirements whereas RDBMS has higher hardware and software requirements.
* In DBMS, data redundancy is common while in RDBMS, keys and indexes do not allow data redundancy.



Difference between DBMS and RDBMS

1. Get yourself familiar with a concept of [Web Application Frameworks](https://en.wikipedia.org/wiki/Web_framework). Get to know what server-side (backend) and client-side (forntend) frameworks exist and what they are used for.
2. [Practice basic JavaScript syntax just for fun](https://www.w3schools.com/js/js_intro.asp).
3. Explore what [RESTful API](https://restfulapi.net) is and what it is used for in Web development.
4. Read what [Cascading Style Sheets (CSS)](https://en.wikipedia.org/wiki/CSS) is used for and browse [basic syntax and properties](https://www.w3schools.com/css/css_intro.asp).

## DBMS vs RDBMS: Difference between DBMS and RDBMS

The below table demonstrates the main difference between RDBMS and DBMS:

| **Parameter** | **DBMS** | **RDBMS** |
| --- | --- | --- |
| Storage | [DBMS](https://www.guru99.com/dbms-tutorial.html) stores data as a file. | Data is stored in the form of tables. |
| Database structure | DBMS system, stores data in either a navigational or hierarchical form. | [RDBMS](https://www.guru99.com/difference-dbms-vs-rdbms.html) uses a tabular structure where the headers are the column names, and the rows contain corresponding values |
| Number of Users | DBMS supports single user only. | It supports multiple users. |
| ACID | In a regular database, the data may not be stored following the ACID model. This can develop inconsistencies in the database. | Relational databases are harder to construct, but they are consistent and well structured. They obey [ACID](https://www.guru99.com/dbms-transaction-management.html) (Atomicity, Consistency, Isolation, Durability). |
| Type of program | It is the program for managing the databases on the computer networks and the system hard disks. | It is the database systems which are used for maintaining the relationships among the tables. |
| Hardware and software needs. | Low software and hardware needs. | Higher hardware and software need. |
| Integrity constraints | DBMS does not support the integrity constants. The integrity constants are not imposed at the file level. | RDBMS supports the integrity constraints at the schema level. Values beyond a defined range cannot be stored into the particular RDMS column. |
| Normalization | DBMS does not support Normalization | RDBMS can be Normalized. |
| Distributed Databases | DBMS does not support distributed database. | RBMS offers support for distributed databases. |
| Ideally suited for | DBMS system mainly deals with small quantity of data. | RDMS is designed to handle a large amount of data. |
| Dr. E.F. Codd Rules | Dbms satisfy less than seven of Dr. E.F. Codd Rules | Dbms satisfy 8 to 10 Dr. E.F. Codd Rules |
| Client Server | DBMS does not support client-server architecture | RDBMS supports client-server architecture. |
| Data Fetching | Data fetching is slower for the complex and large amount of data. | Data fetching is rapid because of its relational approach. |
| Data Redundancy | Data redundancy is common in this model. | Keys and indexes do not allow Data redundancy. |
| Data Relationship | No relationship between data | Data is stored in the form of tables which are related to each other with the help of foreign keys. |
| Security | There is no security. | Multiple levels of security. Log files are created at OS, Command, and object level. |
| Data Access | Data elements need to access individually. | Data can be easily accessed using SQL query. Multiple data elements can be accessed at the same time. |
| Examples | Examples of DBMS are a file system, XML, Windows Registry, etc. | Example of RDBMS is MySQL, Oracle, SQL Server, etc. |

1. Get yourself familiar with concept of [Web Application Frameworks](https://en.wikipedia.org/wiki/Web_framework" \t "_blank). Get to know what server-side (backend) and client-side (forntend) frameworks exist and what they are used for.

A **web framework** (**WF**) or **web application framework** (**WAF**) is a [software framework](https://en.wikipedia.org/wiki/Software_framework) that is designed to support the development of [web applications](https://en.wikipedia.org/wiki/Web_application) including web services, [web resources](https://en.wikipedia.org/wiki/Web_resource), and web [APIs](https://en.wikipedia.org/wiki/API). Web frameworks provide a standard way to build and deploy web applications on the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web). Web frameworks aim to automate the overhead associated with common activities performed in [web development](https://en.wikipedia.org/wiki/Web_development). For example, many web frameworks provide [libraries](https://en.wikipedia.org/wiki/Library_(computing)) for [database](https://en.wikipedia.org/wiki/Database) access, [templating](https://en.wikipedia.org/wiki/Template_processor) frameworks, and [session](https://en.wikipedia.org/wiki/Session_(computer_science)) management, and they often promote [code reuse](https://en.wikipedia.org/wiki/Code_reuse).[[1]](https://en.wikipedia.org/wiki/Web_framework#cite_note-1) Although they often target development of [dynamic web sites](https://en.wikipedia.org/wiki/Dynamic_web_page), they are also applicable to [static websites](https://en.wikipedia.org/wiki/Static_web_page).

## Types of framework architectures

Most web frameworks are based on the [model–view–controller](https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller) (MVC) [pattern](https://en.wikipedia.org/wiki/Software_design_pattern).[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

### Model–view–controller (MVC)

Main article: [Model–view–controller](https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller)

Many frameworks follow the MVC [architectural pattern](https://en.wikipedia.org/wiki/Architectural_pattern_(computer_science)) to separate the [data model](https://en.wikipedia.org/wiki/Data_model) into [business rules](https://en.wikipedia.org/wiki/Business_rule) (the "controller") and the [user interface](https://en.wikipedia.org/wiki/User_interface) (the "view"). This is generally considered a good practice as it [modularizes code](https://en.wikipedia.org/wiki/Separation_of_concerns), promotes [code reuse](https://en.wikipedia.org/wiki/Code_reuse), and allows multiple interfaces to be applied. In web applications, this permits different views to be presented, for example serving different [web pages](https://en.wikipedia.org/wiki/Web_page) for mobile vs. desktop browsers, or providing machine-readable [web service](https://en.wikipedia.org/wiki/Web_service) interfaces.

#### Push-based vs. pull-based

Most MVC frameworks follow a push-based architecture also called "action-based". These frameworks use actions that do the required processing, and then "push" the data to the view layer to render the results.[[7]](https://en.wikipedia.org/wiki/Web_framework#cite_note-7) [Django](https://en.wikipedia.org/wiki/Django_(web_framework)), [Ruby on Rails](https://en.wikipedia.org/wiki/Ruby_on_Rails), [Symfony](https://en.wikipedia.org/wiki/Symfony), [Spring MVC](https://en.wikipedia.org/wiki/Spring_MVC), [Stripes](https://en.wikipedia.org/wiki/Stripes_(framework)), [Sails.js](https://en.wikipedia.org/wiki/Sails.js), [CodeIgniter](https://en.wikipedia.org/wiki/CodeIgniter)[[8]](https://en.wikipedia.org/wiki/Web_framework#cite_note-8) are good examples of this architecture. An alternative to this is pull-based architecture, sometimes also called "component-based". These frameworks start with the view layer, which can then "pull" results from multiple controllers as needed. In this architecture, multiple controllers can be involved with a single view. [Lift](https://en.wikipedia.org/wiki/Lift_(web_framework)), [Tapestry](https://en.wikipedia.org/wiki/Tapestry_(programming)), [JBoss Seam](https://en.wikipedia.org/wiki/JBoss_Seam), [Jakarta Server Faces](https://en.wikipedia.org/wiki/Jakarta_Server_Faces), and [Wicket](https://en.wikipedia.org/wiki/Apache_Wicket) are examples of pull-based architectures. [Play](https://en.wikipedia.org/wiki/Play_Framework), [Struts](https://en.wikipedia.org/wiki/Apache_Struts), RIFE, and [ZK](https://en.wikipedia.org/wiki/ZK_(framework)) have support for both push- and pull-based application controller calls.[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

### Three-tier organization

In [three-tier organization](https://en.wikipedia.org/wiki/Three-tier_architecture), applications are structured around three physical tiers: client, application, and database.[[9]](https://en.wikipedia.org/wiki/Web_framework#cite_note-9)[[10]](https://en.wikipedia.org/wiki/Web_framework#cite_note-10)[[11]](https://en.wikipedia.org/wiki/Web_framework#cite_note-11)[[12]](https://en.wikipedia.org/wiki/Web_framework#cite_note-12) The database is normally an [RDBMS](https://en.wikipedia.org/wiki/Relational_database_management_system). The application contains the business logic, running on a server and communicates with the client using [HTTP](https://en.wikipedia.org/wiki/HTTP).[[13]](https://en.wikipedia.org/wiki/Web_framework#cite_note-13) The client on web applications is a web browser that runs HTML generated by the application layer.[[14]](https://en.wikipedia.org/wiki/Web_framework#cite_note-14)[[15]](https://en.wikipedia.org/wiki/Web_framework#cite_note-15) The term should not be confused with MVC, where, unlike in three-tier architecture, it is considered a good practice to keep business logic away from the controller, the "middle layer".[[16]](https://en.wikipedia.org/wiki/Web_framework#cite_note-16)[[17]](https://en.wikipedia.org/wiki/Web_framework#cite_note-17)

## Framework applications

Frameworks are built to support the construction of internet applications based on a single programming language, ranging in focus from general purpose tools such as Zend Framework and Ruby on Rails, which augment the capabilities of a specific language, to native-language programmable packages built around a specific user application, such as [content management systems](https://en.wikipedia.org/wiki/Content_management_system), some mobile development tools and some portal tools.[[18]](https://en.wikipedia.org/wiki/Web_framework#cite_note-18)

### General-purpose website frameworks

Web frameworks must function according to the architectural rules of browsers and [protocols](https://en.wikipedia.org/wiki/Communication_protocol) such as [HTTP](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol), which is [stateless](https://en.wikipedia.org/wiki/Stateless_protocol). Webpages are served up by a [server](https://en.wikipedia.org/wiki/Server_(computing)) and can then be modified by the browser using [JavaScript](https://en.wikipedia.org/wiki/JavaScript). Either approach has its advantages and disadvantages.[*[citation needed](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed" \o "Wikipedia:Citation needed)*]

Server-side page changes typically require that the page be refreshed, but allow any language to be used and more computing power to be utilized. Client-side changes allow the page to be updated in small chunks which feels like a desktop application, but are limited to JavaScript and run in the user's browser, which may have limited computing power. Some mix of the two is typically used.[[19]](https://en.wikipedia.org/wiki/Web_framework#cite_note-19) Applications which make heavy use of JavaScript and only refresh parts of the page, are called [single-page applications](https://en.wikipedia.org/wiki/Single-page_application) and typically make use of a client-side JavaScript web framework to organize the code.[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

#### Server-side

* [Apache Wicket](https://en.wikipedia.org/wiki/Apache_Wicket)
* [ASP.NET Core](https://en.wikipedia.org/wiki/ASP.NET_Core)
* [CakePHP](https://en.wikipedia.org/wiki/CakePHP)
* [Catalyst](https://en.wikipedia.org/wiki/Catalyst_(software))
* [CodeIgniter](https://en.wikipedia.org/wiki/CodeIgniter)
* [CppCMS](https://en.wikipedia.org/wiki/CppCMS)
* [Django](https://en.wikipedia.org/wiki/Django_(web_framework))
* [Flask](https://en.wikipedia.org/wiki/Flask_(web_framework))
* [Jam.py](https://en.wikipedia.org/wiki/Jam.py_(web_framework))
* [Yii](https://en.wikipedia.org/wiki/Yii)
* [Laravel](https://en.wikipedia.org/wiki/Laravel)
* [Mojolicious](https://en.wikipedia.org/wiki/Mojolicious)
* [Ruby on Rails](https://en.wikipedia.org/wiki/Ruby_on_Rails)
* [Sails.js](https://en.wikipedia.org/wiki/Sails.js)
* [Symfony](https://en.wikipedia.org/wiki/Symfony)
* [Spring MVC](https://en.wikipedia.org/wiki/Spring_MVC)
* [VIEwoNLY](https://en.wikipedia.org/w/index.php?title=VIEwoNLY&action=edit&redlink=1)
* [Wt (web toolkit)](https://en.wikipedia.org/wiki/Wt_(web_toolkit))
* [Zend Framework](https://en.wikipedia.org/wiki/Zend_Framework)[[20]](https://en.wikipedia.org/wiki/Web_framework#cite_note-20)

#### Client-side

Main article: [Single-page application](https://en.wikipedia.org/wiki/Single-page_application)

Examples include [Backbone.js](https://en.wikipedia.org/wiki/Backbone.js), [AngularJS](https://en.wikipedia.org/wiki/AngularJS), [Angular](https://en.wikipedia.org/wiki/Angular_(application_platform)), [EmberJS](https://en.wikipedia.org/wiki/EmberJS), [ReactJS](https://en.wikipedia.org/wiki/React_(JavaScript_library)), [jQuery UI](https://en.wikipedia.org/wiki/JQuery_UI), [Svelte](https://en.wikipedia.org/wiki/Svelte), and [Vue.js](https://en.wikipedia.org/wiki/Vue.js)

 [Practice basic JavaScript syntax just for fun](https://www.w3schools.com/js/js_intro.asp).

 Explore what [RESTful API](https://restfulapi.net) is and what it is used for in Web development.

 Read what [Cascading Style Sheets (CSS)](https://en.wikipedia.org/wiki/CSS) is used for and browse [basic syntax and properties](https://www.w3schools.com/css/css_intro.asp).