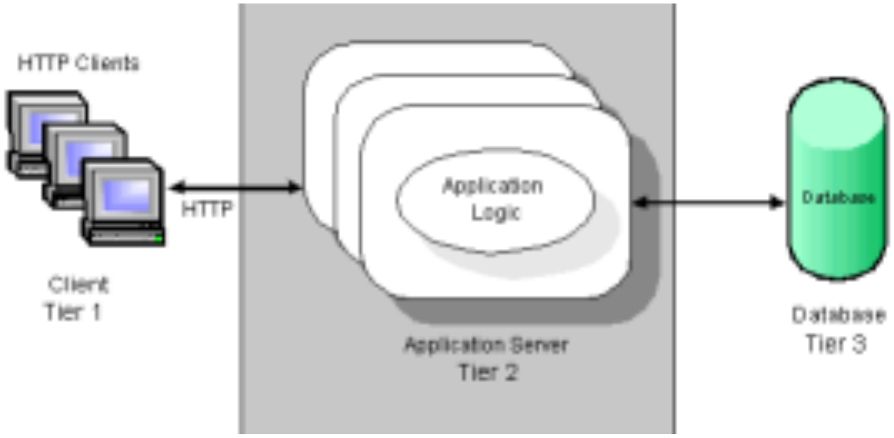
Web Based Database Application

Multi-tier Architecture



Multitier architecture - Wikipedia

In software engineering, **multi-tier architecture** (often referred to as **n-tier architecture**) is a client–server architecture in which presentation, application processing, and data management functions are logically separated. For example, an application that uses middleware to service data requests between a user and a database employs multi-tier architecture. The most widespread use of multi-tier architecture is the **three-tier architecture**.

N-tier application architecture provides a model by which developers can create flexible and reusable applications. By segregating an application into tiers, developers acquire the option of modifying or adding a specific layer, instead of reworking the entire application. **Three-tier** architectures typically comprise a *presentation* tier, a *business* or *data access [logic]* tier, and a *data* tier.

(*Wikipedia : Multitier Architecture*)

Why N-tier Architecture

- Increased performance, flexibility, maintainability, reusability, and scalability, while hiding the complexity of distributed processing from the user.

Significance of “Tiers”

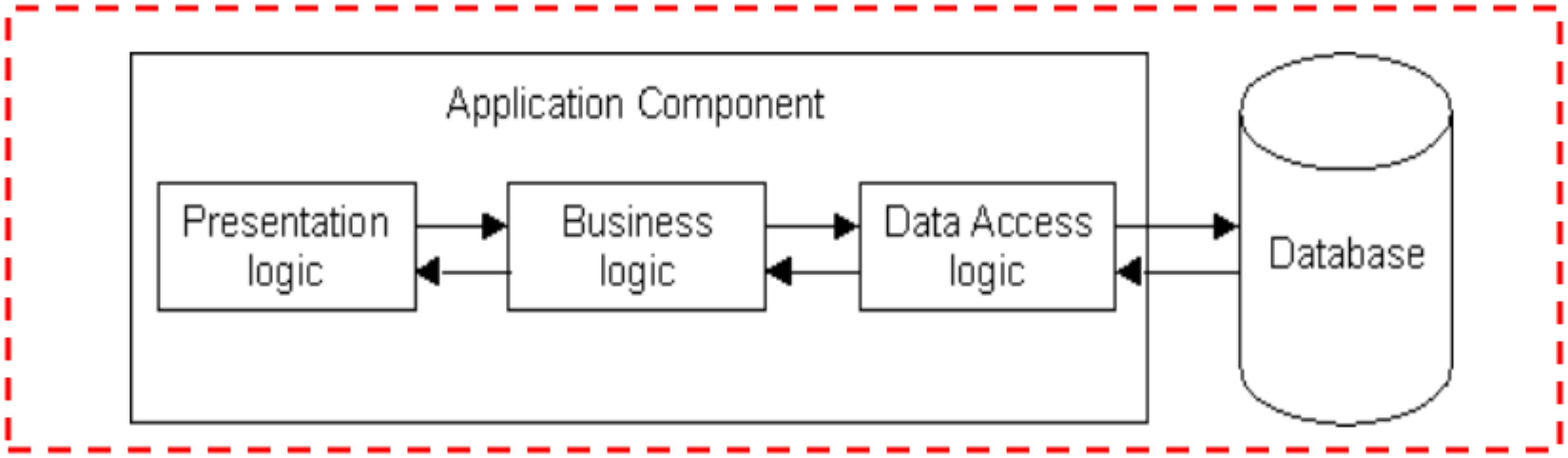
N-tier architectures have the same components

- Presentation
- Business/Logic
- Data

N-tier architectures try to separate the components into different tiers/layers

- Tier: physical separation
- Layer: logical separation

Significance of “Tiers”



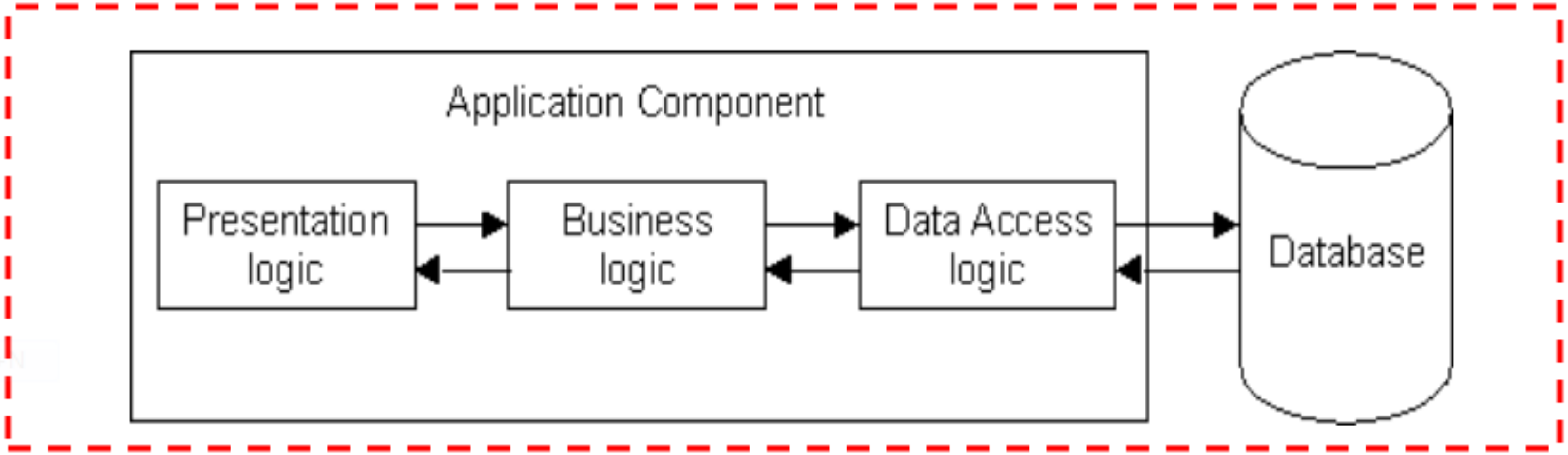
Database runs on Server

- Separated from client
- Easy to switch to a different database

Presentation and logic layers still tightly connected

- Heavy load on server
- Potential congestion on network
- Presentation still tied to business logic

1-Tier Architecture



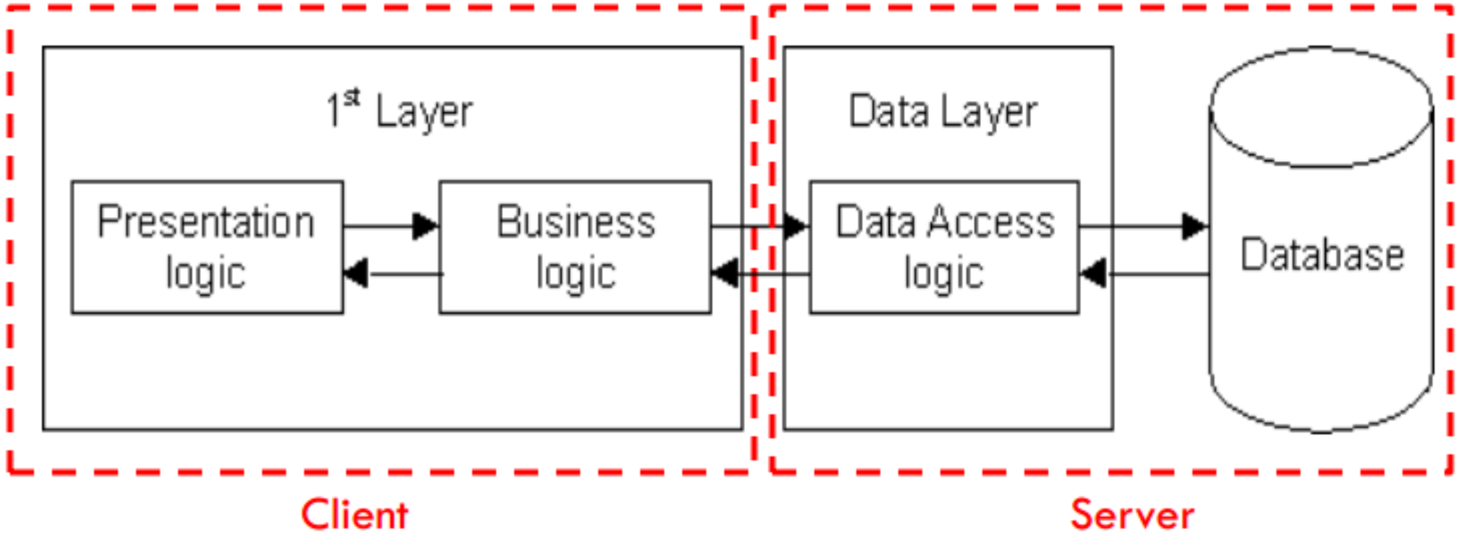
All 3 layers are on the same machine

- All code and processing kept on a single machine

Presentation, Logic, Data layers are tightly connected

- Scalability: Single processor means hard to increase volume of processing
- Portability: Moving to a new machine may mean rewriting everything
- Maintenance: Changing one layer requires changing other layers

2-Tier Architecture



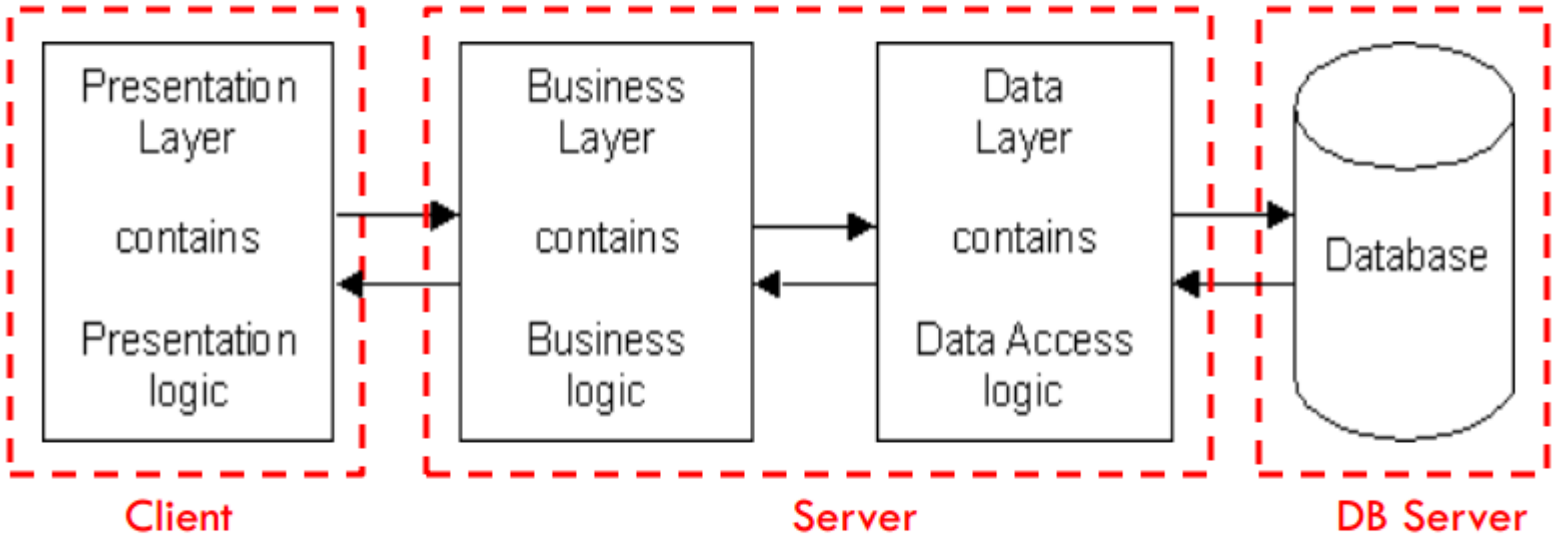
Database runs on Server

- Separated from client
- Easy to switch to a different database

Presentation and logic layers still tightly connected (coupled)

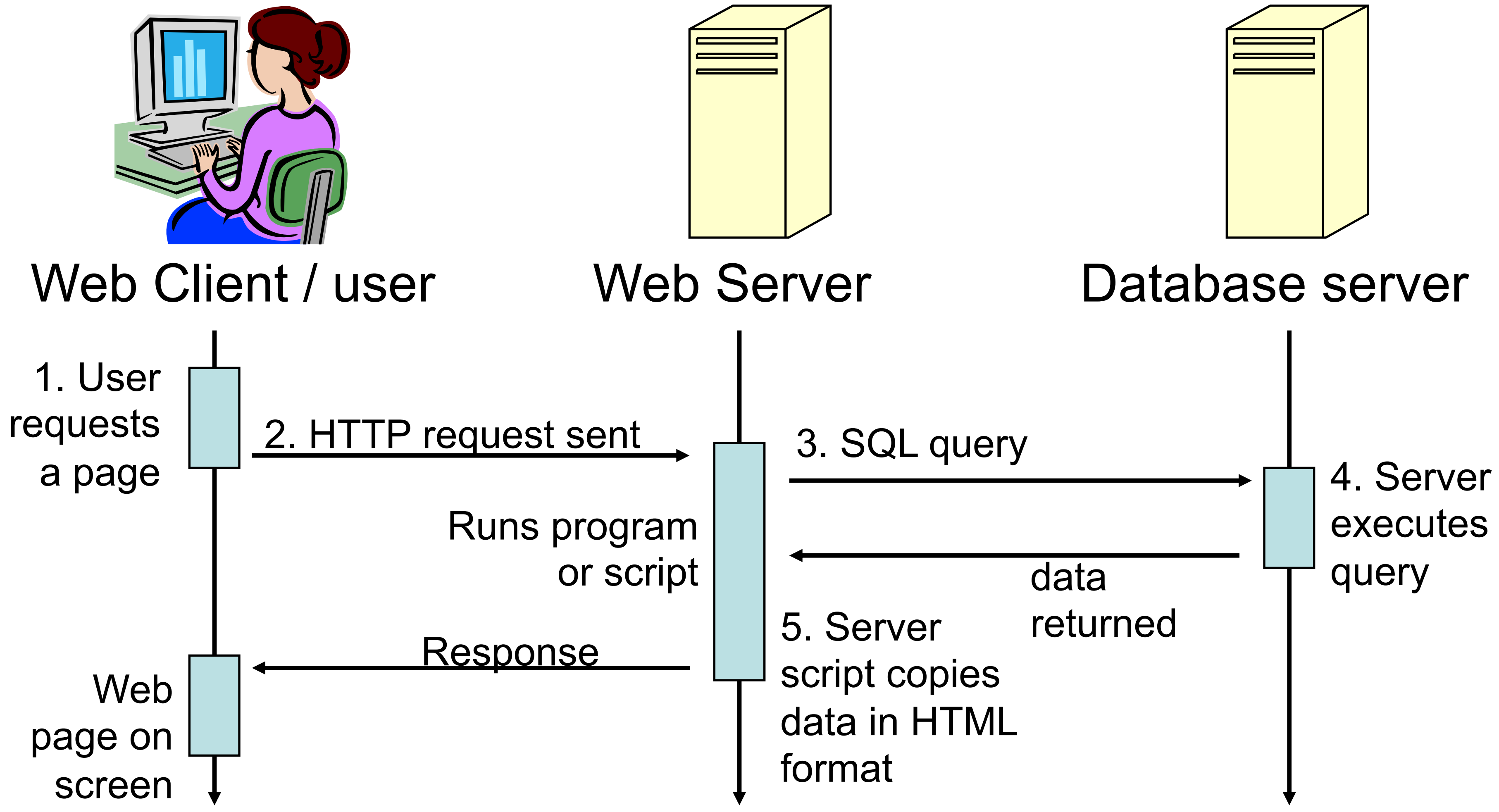
- Heavy load on server
- Potential congestion on network
- Presentation still tied to business logic

3-Tier Architecture



- Each layer can potentially run on a different machine
- Presentation, logic, data layers disconnected

Web sites based on data



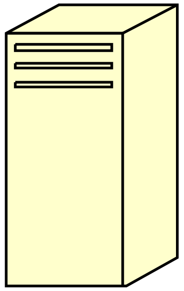
The "three tier architecture"

Some technologies to use

Web Client / user

Any Web browser

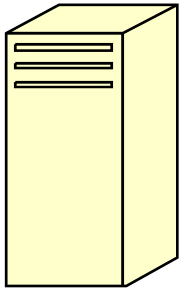
Client languages:
HTML, CSS,
JavaScript



Web Server

Apache (most popular)
(Tomcat-Java based logic)

Server language:
Java, Perl, Python,
Tcl, PHP, C, C#, etc.



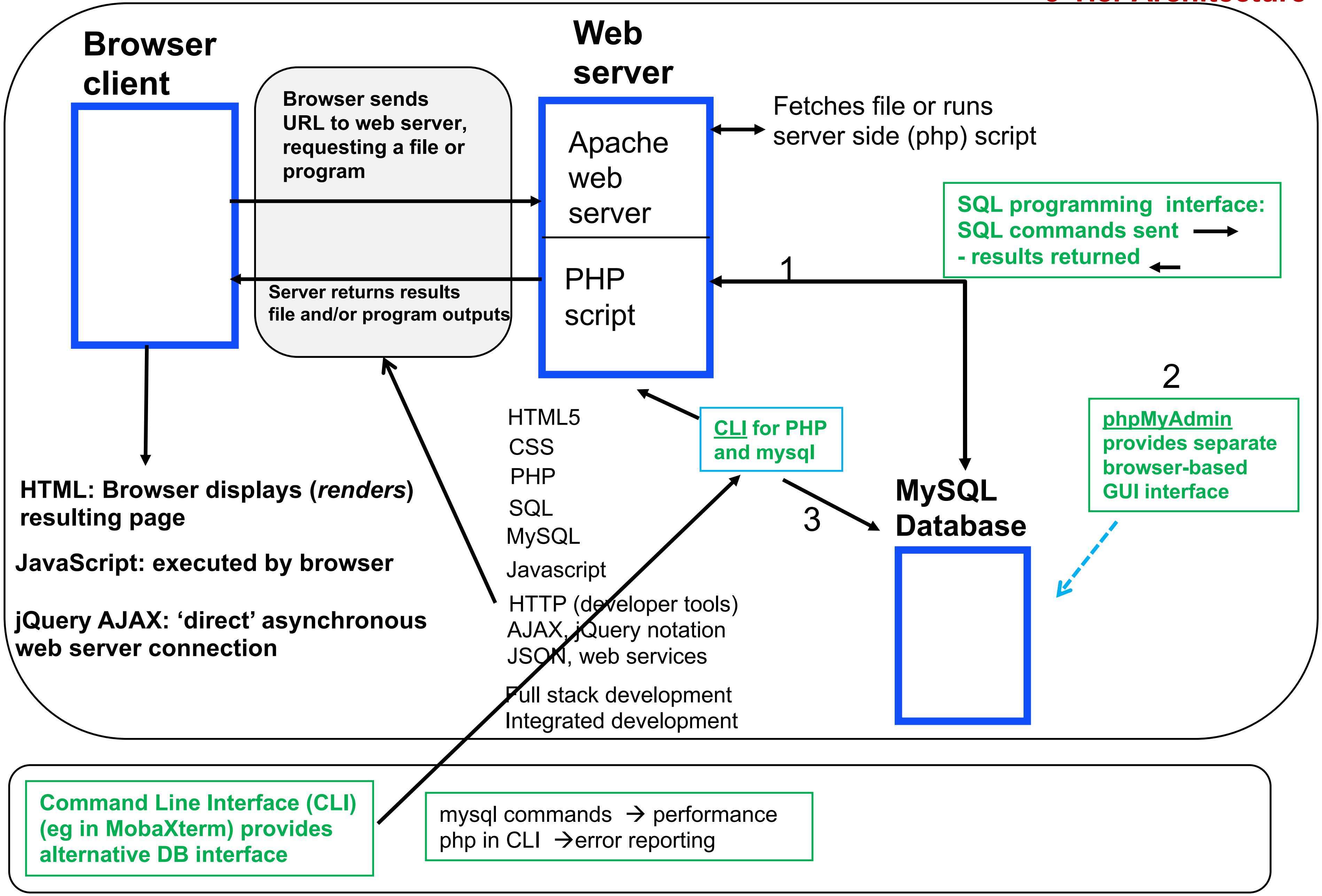
Database server

MySQL

Query language:
SQL

Run as slide show to see animated display.

3-Tier Architecture



Browser sends URL to web server, requesting a file or program

Server returns results file and/or program outputs

Fetches file or runs server side (php) script

SQL programming interface: SQL commands sent → - results returned ←

CLI for PHP and mysql

phpMyAdmin provides separate browser-based GUI interface

HTML: Browser displays (*renders*) resulting page

JavaScript: executed by browser

jQuery AJAX: 'direct' asynchronous web server connection

Command Line Interface (CLI) (eg in MobaXterm) provides alternative DB interface

mysql commands → performance
php in CLI →error reporting

The 3-Tier Architecture for Web Apps

- **Presentation Layer**

  Static or dynamically generated content rendered by the browser (front-end)

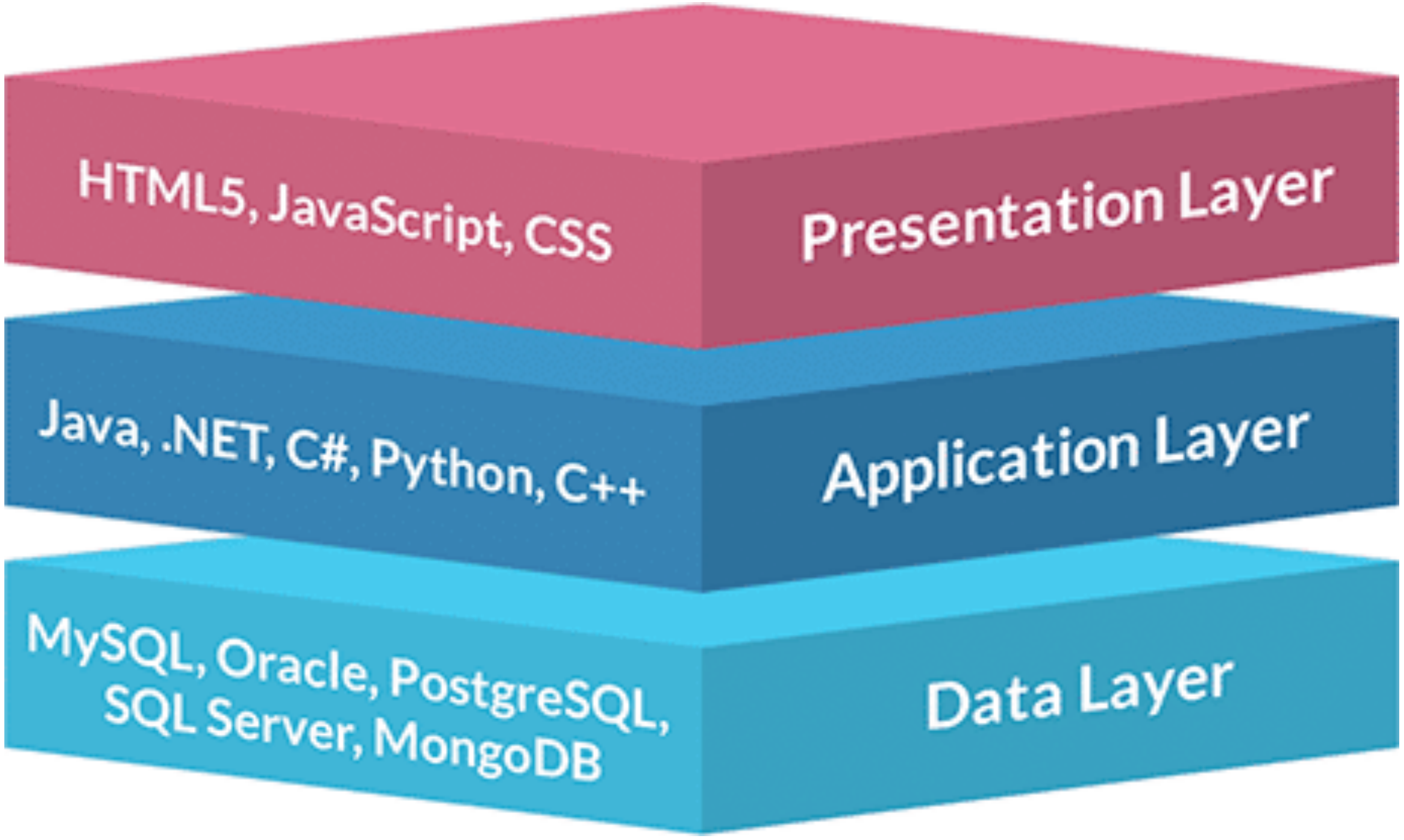
- **Logic Layer**

  A dynamic content processing and generation level application server, e.g., Java EE, ASP.NET, PHP, ColdFusion platform (middleware)

- **Data Layer**

  A database, comprising both data sets and the database management system or RDBMS software that manages and provides access to the data (back-end)

A Typical 3-tier Architecture


HTML5, JavaScript, CSS
Presentation Layer
Java, .NET, C#, Python, C++
Application Layer
MySQL, Oracle, PostgreSQL, SQL Server, MongoDB
Data Layer