System Design

Basics

What is a System?

It is an architecture or collection of software / technologies that interact with each other to serve a certain set of users to fulfill a set of requirements.

3 things are crucial to a system:

1. Users of that system
2. Set of requirements that it fulfills
3. Components it is made up of

As an engineer, you must know which components are apt for specific use case, the pros and cons of each component, the concerns related to each component etc.

Components of a System

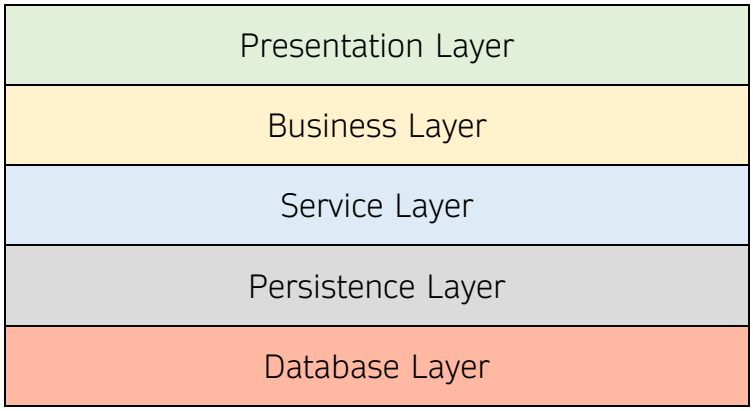
These are classified into 2 categories-

1. Logical Entities
2. Tangible Entities

Logical Entities	Tangible Entities
Data	Text, images, videos
Database	MongoDB, MySQL, Cassandra
Applications	Java, Python, Amber, React, Angular
Cache	Redis, MemeCache
Message Queues	Kafka, RabbitMQ
Infrastructure	AWS, GCP, Azure
Communication	APIs, RPCs, Messages

Layers in a System

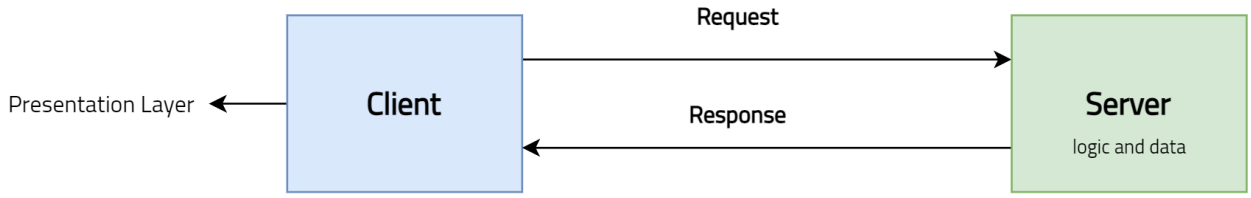
There can be any number of layers in a system, some of the important ones are listed below-



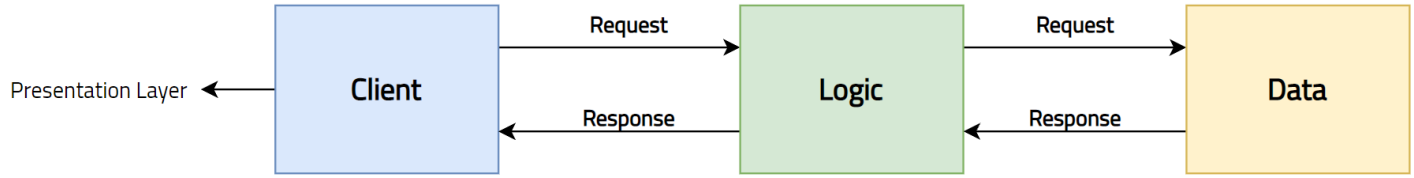
The colors don't mean anything, I just did that because it looks cool

Client-Server Architecture

2-Tier Architecture

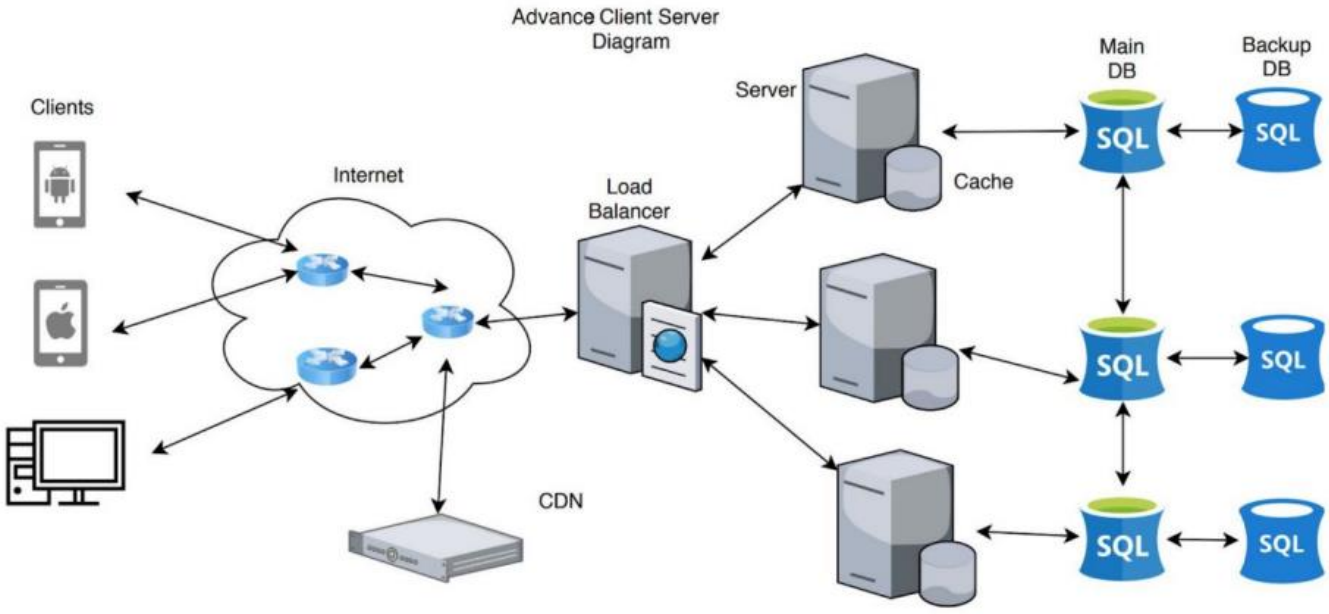


3-Tier Architecture



There can be any number of tiers in an architecture.

Detailed diagram –



Flow-

1. Client requests data from server
2. Load Balancer routes the request to the appropriate server
3. Server processes the request client
4. Server queries appropriate database for some data
5. Database returns the queried data back to the server
6. The server processes the data and send the data back to the client
7. This process repeats

Thin Clients – software that is primarily designed to communicate with a server. Its features are produced by servers such as a cloud platform. Ex. Chrome, Netflix etc.

Thick Clients – software that implements its own features. It may connect to server, but it remains mostly functional when disconnected. Ex. Valorant, multiplayer PC games, video editing software etc.

Negative behaviors of Client-Server architecture

1. Greater potential for failure because of centralized control
    a. Servers help in administering the whole set-up
    b. If the servers go down, then entire service goes down
2. Vulnerable to DOS (Denial of Service) attacks because the number of servers is considerably smaller than the number of clients
    a. DOS attack – flooding the resource with requests in an attempt to overload the system, results in legitimate requests not being fulfilled
    b. DDOS attack – Distributed Denial of Service, the incoming traffic comes from multiple clients
3. Expensive to install and manage the network – Server machines are powerful and expensive, also it requires hiring employees with networking and infrastructure knowledge, in order to manage the system

Supported non-functional properties

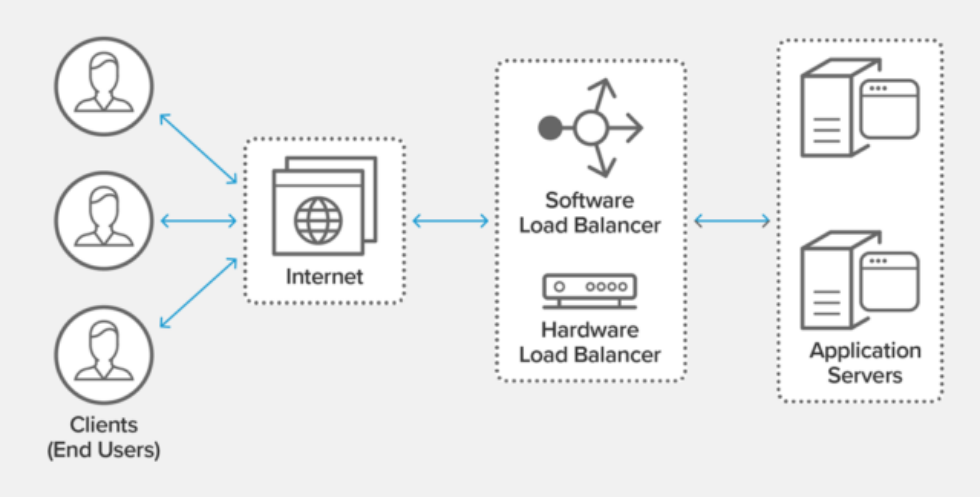
1. Scalability – capable of horizontal and vertical scaling of the system
    a. Horizontal scaling – adding or removing servers in the network
    b. Vertical scaling – migrating to larger and faster server machines
2. Availability – server uptime is possible during maintenance, with server duplications
3. Dependability – processing and resource allocation is done by a dedicated set of machines, these servers can be optimized to complete a certain task quickly and efficiently
4. Heterogeneity – clients are consumers of services and servers are providers of services
    a. Clear separation of concerns results in the system being composed of disparate parts
    b. For ex, one or more servers may fail, but the system can still function as long as the other servers offer the same services

Load Balancer

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

A load balancer 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. The main functions are summarized below –

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



Load balancing algorithms

Every algorithm has its own pros and cons, which one to choose, depends on the requirement

1. Round Robin – Requests are distributed across the group of servers sequentially
2. Least Connections – A new request is sent to the server with the fewest current connections to clients
3. Least Time – Sends request to the server selected by a formula that combines the fastest response time and fewest connections
4. Hash – Distributes the requests based on a key you define, such as the client IP address or the request URL
5. Random with two choices – Picks 2 servers at random and sends the requests to the one that has least connections

Benefits of Load Balancing

1. Reduced downtime
2. Scalable
3. Redundancy
4. Flexibility
5. Efficiency

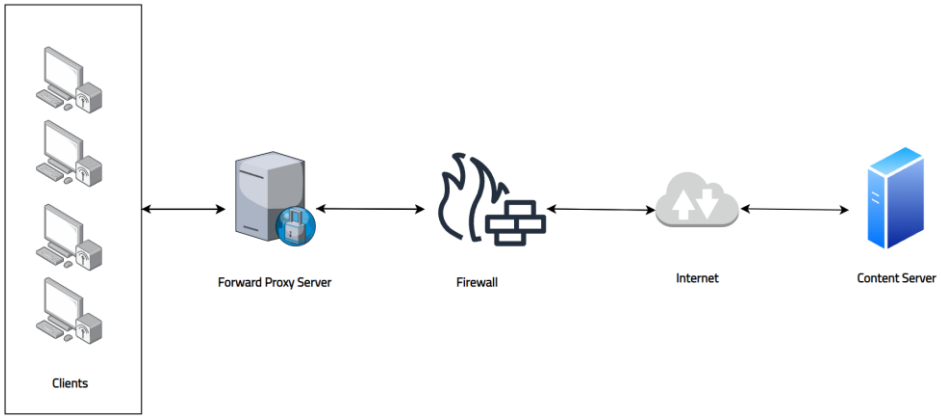
Proxies

What is a proxy server?

A proxy server is an intermediary piece of hardware / software sitting between the client and the backend server.

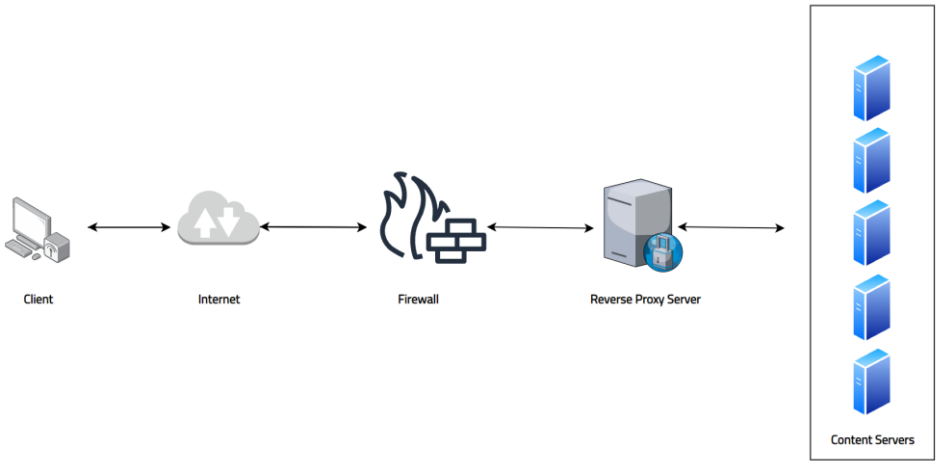
Forward Proxy

It is used to pass requests from an isolated, private network to the Internet through a firewall. Requests from an intranet can be rejected or allowed to pass through a firewall. Requests may also be fulfilled by serving from cache rather than passing through the Internet. This allows a level of network security and lessens network traffic. A forward proxy proxies on behalf of the clients.



Reverse Proxy

It is used to pass requests from the Internet, through a firewall to isolated, private networks. It is used to prevent Internet clients from having direct, unmonitored access to sensitive data residing on content servers on an intranet. If caching is enabled, a reverse proxy can also lessen network traffic by serving cached information rather than passing all requests to actual content servers. A reverse proxy proxies on behalf of the servers. A reverse proxy server also acts as a load balancer for the servers behind them. When a reverse proxy performs load balancing, it distributes incoming requests to a cluster of servers, all providing the same kind of service.



What functions can it perform?

1. Filter requests – runs every request through a filter, looking up each address in its database of allowed or disallowed sites, and it allows or blocks each request based on its internal database.
2. Log requests
3. Transform requests (encryption, compression etc.)
4. Cache – caching frequently used pages so the user request doesn't have to go all the way out to the Internet
5. Batch requests
6. Collapsed forwarding – enable multiple client requests for the same URI to be processed as one request to the backend server
7. Security – It can provide network address translation, which makes the individual users and computers on the network anonymous when they are using the Internet.