**Client Server Architecture**

Server

Client

Request

Response

Presentation Logic & Data

(2 tier Architecture)

**Thick and Thin Client**

It’s always not necessary that logic sits on server. When logical processing happens on Client side, that client is called **Thick Client**.  
When the logical processing happens on server side, the client is called **thin client**.  
**Examples**  
Example of **thin** client are **streaming platforms** like Netflix, HotStar, Zee5, etc. where server does the majority processing.  
Examples of **thick** client are Microsoft **Outlook, Video Editing** Software, etc.

**3 Tier Architecture**When logical processing is immense and data is huge, the logic and data part are separated. This is 3 –tier architecture.

Logic

Data

Client

Presentation

**N Tier Architecture**  
When 3 tier architecture cannot serve the huge traffic and immense processing, we introduce **Caching** between Logic and Data layer and a **Load Balancer** between Client and Logic. Then the Architecture is termed as N-Tier.

**When to decide, which architecture to use?**

**Thick Client**

Gaming apps, Video Editing Software, Outlook, Ms-Excel, Ms-Paint.

**Thin Client**

Video streaming apps, E-Commerce Applications, Banking Applications, Web-browser.

**2-Tier Architecture**

Light weight website for small businesses

**3-Tier Architecture**

Basic library management system in school, inventory management system where logic (code) is on a separate machine and data servers are different.

**N-Tier Architecture**

Large scale, scalable systems like Gmail, Facebook, WhatsApp, etc.

**Pros and Cons of thick and thin client:**

Thick client:

**Pros :**

1) System can perform seamlessly without considering network latency.

2) More cost to maintain.

3) High responsive to the user

**Cons:**

1) No assurance of data availability as it has possibility of system crash and other difficulties.

2) High cost to maintain the app.

3) Effective usage of system resource is less.

Thin client:

**Pros:**

1) Easy to manage/maintain system infrastructure.

2) Highly scalable and easy maintainable app

3) Cost effective

**Cons:**

1) Highly dependent on network connectivity/availability.

2) Each components are managed by component professionals.

Yes, Server can be a client also, because in large enterprise applications not all the apps takes care of computations some may get the other processed/computation result and take care of responsibility to get back to presentation layer.