Socket Programming & Intro to Express.

Because of NodeJS now we can run JS on our local machine & JS will be able to access OS resources. ex. file IO.

Also enables JS to make network requests.

Backend

→ network → scaling → business logic.

→ DB → security

Journey of a network request:

2 responsibilities of the client ① ②

Client is the layer with which our user interacts. Client is layer which is capable to initiating a network request.

Sometimes there are network requests which need to happen before user interaction. ex EMI mandate. (auto money deduction via server requests) no user interaction needed. (You may call them automated requests).

Task of server	own servers or public servers
→ collect requests	bare metal. aws, etc.
→ process requests	
→ serve responses.	

TCP network stack.

1. app layer
2. Transport layer
3. network layer
4. data link layer
5. Physical layer

Flow of data

OSI network stack.

flow of data

1. app layer
2. Presentation
3. session
4. transport
5. network
6. data link
7. Physical layer

data flows from layer to layer.
each layer processes the data.

conceptual architecture, easy representation of network stack

TCP stack combines app, presentation, session → app layer.

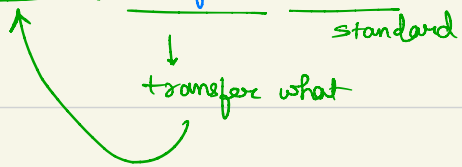
Why conceptual?

Different things that happen when a network requests happen.
the model above represents those things.

① Application layer - top layer, where the user is interacting.
collect data & interaction from user & initiate a request.

Requests follow a set of protocols / instructions / rules. These protocols define how a request should be done. Standardization are brought to us by protocols. → https, smtp, ftp. → app layer protocols.

HTTP — hyper text transfer protocol.



hyper text is a form of a document which is capable of containing hyper links.

html documents contains hyper text.

hypertext links connect one hypertext doc to another.
hyper links

markup language → does not contain logic.

To transfer hypertext docs we use http. which is the standard

---

2. transport layer – sits on the OS

transport layer protocols

① TCP → segments — sliding window mechanism

② UDP → data grams

→ breaks down data incoming from app layer into small components.

→ responsible for reliable or unreliable connection.

reliable → chat app connection. via TCP no data loss slow

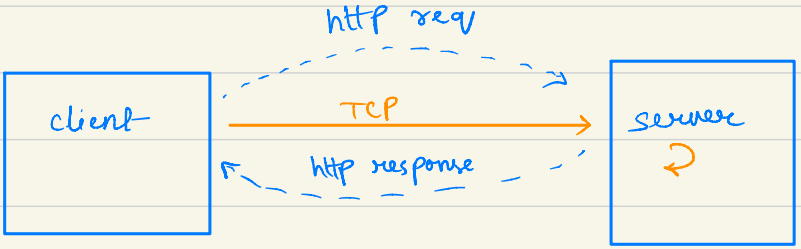
unreliable → streaming, voip, online gaming via UDP. supports data loss. fast.

http depends on TCP → no data loss.

wherever you need latest data.

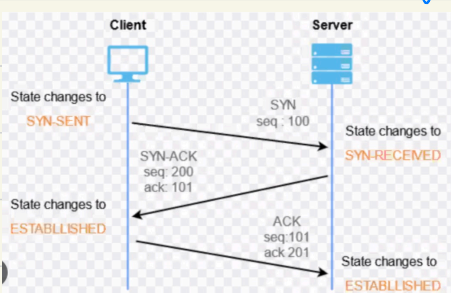
app layer protocols depend on Transport layer protocols.

sockets are hard to scale because of no. of socket connections.



TCP connection is established first then http works on top of it.

TCP is prepared via 3 way handshake



The three-step process of a TCP handshake is:

1. Computer A transmits a SYNchronize packet to computer B
2. Computer B sends back a SYNchronize-ACKnowledge packet to A
3. Computer A transmits an ACKnowledge packet to B, and the connection is established

3. Network layer / internet layer

contains IP & IP addresses etc. Also present in OS.

→ routing algorithms, shortest path algos.

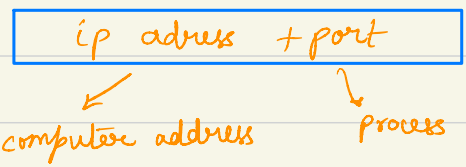
→ on the network layer routing logic for packets is present.

→ Network interface cards (NIC) PCIE. hardware.

Socket Programming (not web sockets)

socket → combination of ip address & port.

different process have network connections. Which process you want to communicate to is known from the port.



Mac address is hardware. NIC card. (data link layer) MAC-addr doesn't change

classless & classful ip address.

client prepares a socket object, server also prepare socke objects with use of socket object TCP connection is made. Frameworks abstract away these socket objects.

Frameworks use http module. This module contains logic for sending & receiving http requests. Http has net module which prepares the TCP connections. For UDP nodeJS has dgram module

socket object contains the logic for 3 way connection.