$ git fetch & $ git pull difference,

both will retrieve recent codes from a repo for you but git pull try/tries to pull the changes and merge them simultaneously.

$ git fetch origin (this would primarily fetch updates from the origin remote) unlike without it which fetches from all configured

NODE.JS :- (also written in C++)

① ↳ allows to run JS on serverside or directly on computer instead of just the web browser. & compiles.

V8 engine running in browsers that is made through C++ runs JS / compiles it into machine code.

High level programming lang
↑
Assembly code
↑
Machine Code

② Node.js also reads & writes files on computer } Backend ops.
③ " connects to a database
④ " acts as a server for content.

* Basically using JS as a backend.

Date

The Global Object: (check test.Js)

An object available in every Node.Js module that gives you access of global variables and functions.

(without needing to import

In browser we use 'Window' as the method of accessing global object. while in computer we write node global in the terminal.

Module & Requires. (Check module.js & people.js)

Making modular codes with multiple files.

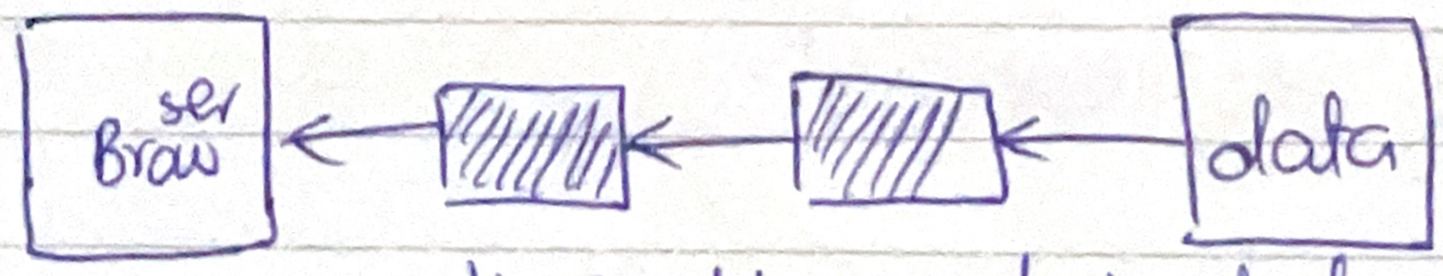
Node and the file System. (Check files.Js).

Concepts of writing / Reading data from files.
making and removing directories.
deleting pre-existing files.

Streams & Buffers. (Check stream.js)

↓

start using data before its fully loaded using the fs.
(Just like life streaming in twitch).



the buffer gets loaded and transfers data and thus buffer by buffer the data is continously / live transferred.

Clients & Server.

To and fro communication between website & server/backend is through sending & retriving request / HTTP; through functs like GET / POST; after the server identifies the right site the IPaddresses.

In Node we create a server (that lives in the backend of our website);

- it listens to requests from browser
- sends responses appriotately to the browser.

(check server.js)

LOCAL HOST & PORT NUMBER: ↳ are like doors to a computer. through which internet connection and communication can be made.

↳ localhost is like a DOMAIN NAME on the web, but it takes us to a very specific ip address i.e '127.0.0.1' which actually symbolizes our own computer.

∴ we are listening to requests coming to our own computer as host while developing a website.

hence localhost:3000 mean through this port our own computer is going to be listening to requests

Date

find
- sending responses (Basics)
- sending responses (html pages)

} at server.js

Status Codes.

describes the type of response sent to browser.

200 - ok
301 - Resource moved
404 - Not found
500 - Internal server error.

100 Range - informational responses
200 Range - success codes
300 Range - codes for redirects
400 Range - user or client error codes
500 Range - server error codes.

NPM (Node package manager).

It is the default Package manager for Node.js used to install, manage and share javascript packages (also called modules / libraries).

Package: any folder or module that contains reuseable JS code.

Example -g nodemon. //allows auto restart of server upon code change.

package.json file.

using
e.g lodash for randomno.
& once methods
etc.

contains project specific packages.

" npm init → to initialize it for our project.
↳ on terminal

Good for mentioning the packages we've used in our project, allowing others to download dependancies before using our project.

EXPRESS APPS: Assists in arranging code, pre
making it cleaner with self made
methods.

~ x ~

CALLBACK & PROMISES IN NODEJS.

JS is single threaded meaning it does one tasks once ata time but some tasks take time, here JS doesn't stops working ∴ the concept of callbacks and promises.

Callback ; it is a function that is passed as an argument in another function to be called backed / executed later.

Promise ; it is an object that represents a future value.
which will be a
- resolve (success)
- reject (failure).

if now ejs templates, same code is being repeated, we can put it in partials and reuse those partial templates. folder

```
let pizzaOrder = new Promise (function (resolve,
  reject) {
      console.log ("ordering Pizza")

  setTimeOut (() => {
  let success = true;
  if (success) {
      resolve("Pizza delivered");

  else { reject ("Pizza burned");

} }, 2000); });
```

~ x ~

VIEW - ENGINE :-

Also to inject dynamic datalike
- variables, - loops etc into the HTML template.

It is easily used with Express apps.
It also makes routing easier (somewhat)

* EJS Templates are processed through EJS view engine on the server.

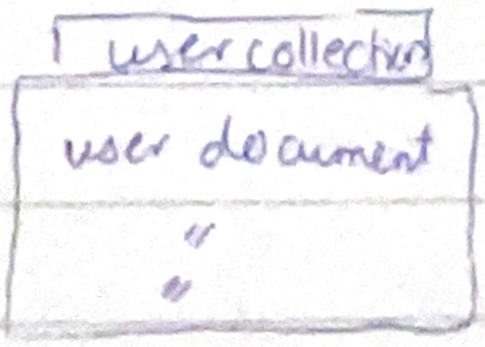
MIDDLE WARE :-

The code that runs inbetween the process of sending requests and receiving response is called as middle ware.

We can have multiple middleware, these codes runs on the server.

Use cases of middleware:

(i) Logger Malware to log details of every request.
(ii) Authentication checks middleware for protected routes
(iii) Middleware to parse JSON data from req
(iv) Return 404 pages.
(v) allows to make private files(by server) like static files to be accessed when placed in public folder.

SQL Databases	No SQL Database.
uses (i) table (ii) Rows (iii) Columns to arrange data	uses (i) Collections (ii) Documents to arrange data. ↓representation  ↓ used through MongoDB.

Date

Mongoose

ODM (Object Document Mapping) library that gives various methods to connect and communicate with MongoDB database.

① we make a schema first.
schema defines the structure of a type of data/ document.
- properties & property types.

② Then we use models that allow to communicate with database collection.

REQUEST TYPES:-

GET
POST (creating new data in DB)
DELETE (Delete Req, to delete data)
PUT (put request to update data)

Route Parameter: (adding avariable that is dynamic in our routing).

MVC → controller
↓ ↘
Model Views

} we can use controller files to perform certain functions in the views through models obv with schema.