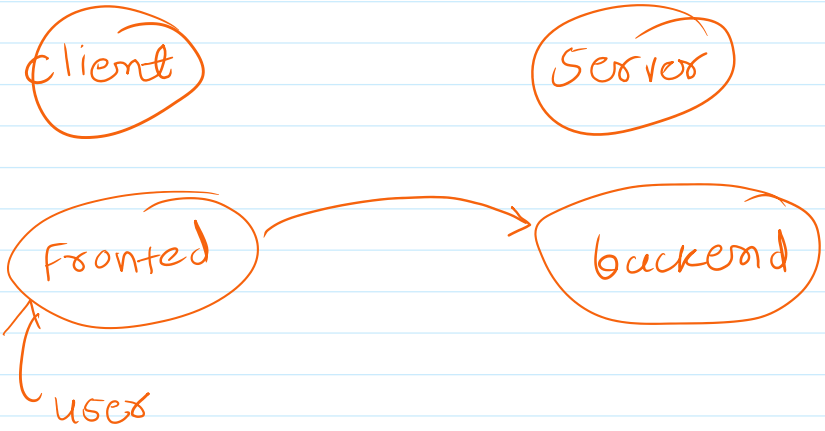
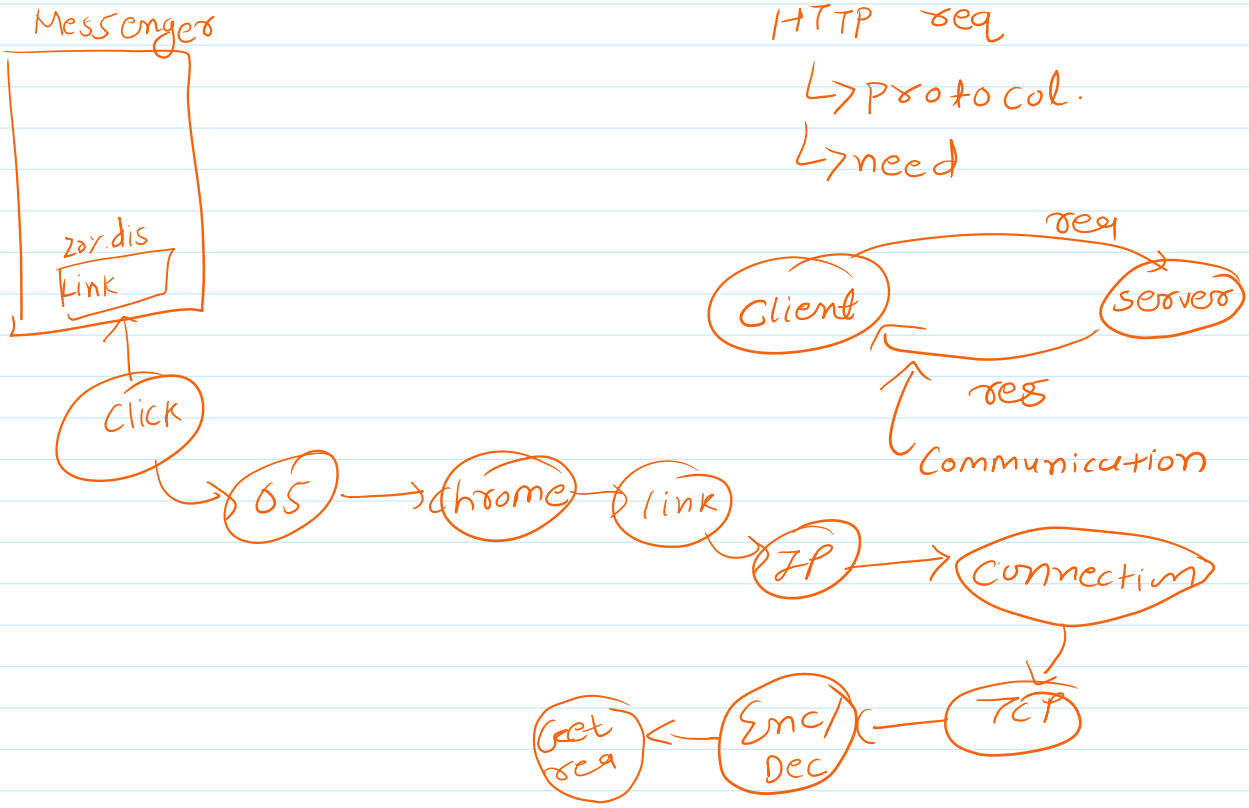
Backend

27 May 2024 13:08





GET Request :

HTTP
- GET → Fecth data (retrive/)
- POST → Submit (new data add)
- PUT → update data
- Delete → remove.

Server Create

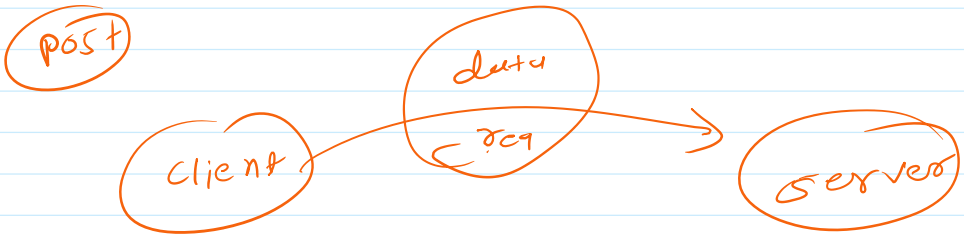
Server Create

↳ Express js

↳ Framework

↳ Server-side Application Create.

→ create a new folder
→ npm init -y
→ Move to flolder
→ npm i express
→ node file.name.js → run



postman throw test.

↳ new → HTTP → Url paste → post → body

↓

Json ← row.
Formed data.

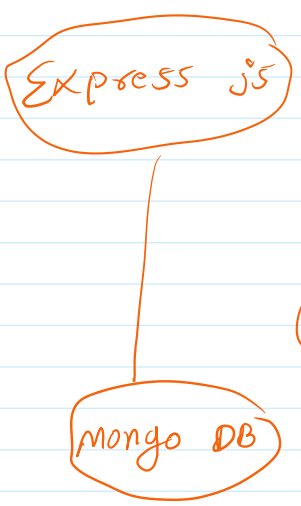
body-parser :- used to parese req.body in express
(passing - fatch / get)
Put or post case used.

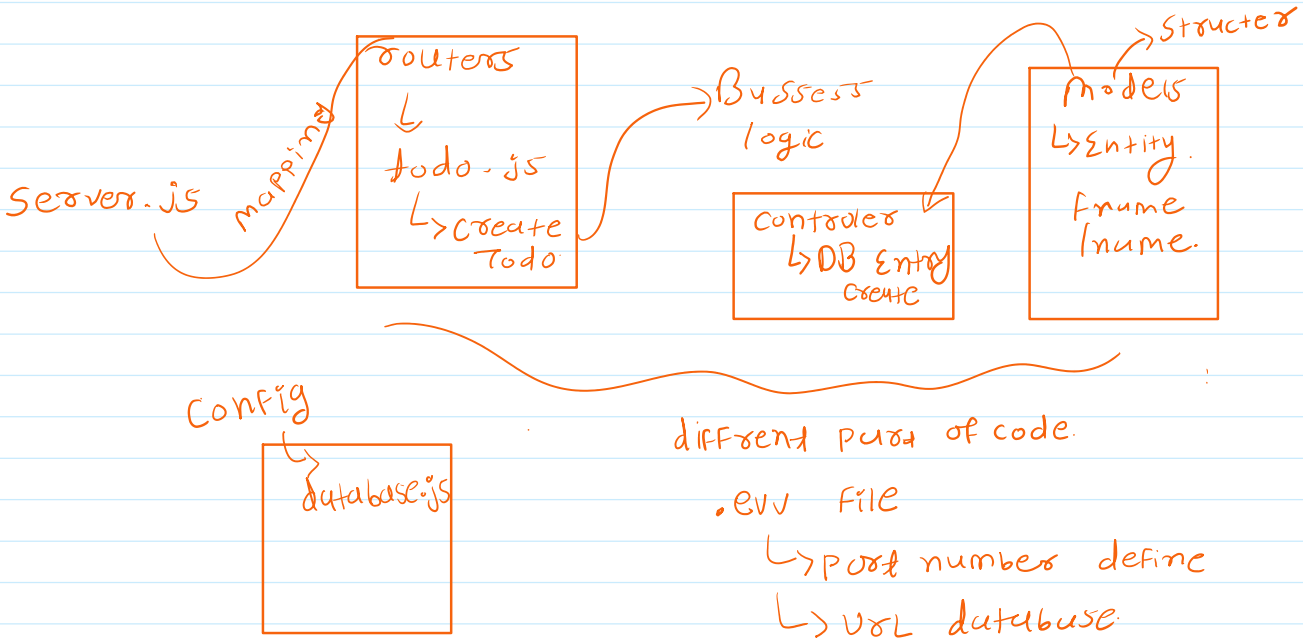
MongoDB :- No-sql database

↳ form of key-Value pair, document, graph

CRUD ← Delete
create
Read.
update.

create read update.





backend app automatic restart.

script → "start": "node index.js"

npm i nodemon

"dev": "nodemon index.js"

terminal → npm run dev

npm i dotenv → this used in the .env file to database URL in process object include.

include.

```
require("dotenv").config();
```

→ this line used in process objed include .env.

Mongoose provided function.

findC)
findbyid C)
DeleteC)

- Model.deleteMany()
- Model.deleteOne()
- Model.find()
- Model.findById()
- Model.findByIdAndDelete()
- Model.findByIdAndRemove()
- Model.findByIdAndUpdate()
- Model.findOne()
- Model.findOneAndDelete()
- Model.findOneAndReplace()
- Model.findOneAndUpdate()
- Model.replaceOne()
- Model.updateMany()
- Model.updateOne()

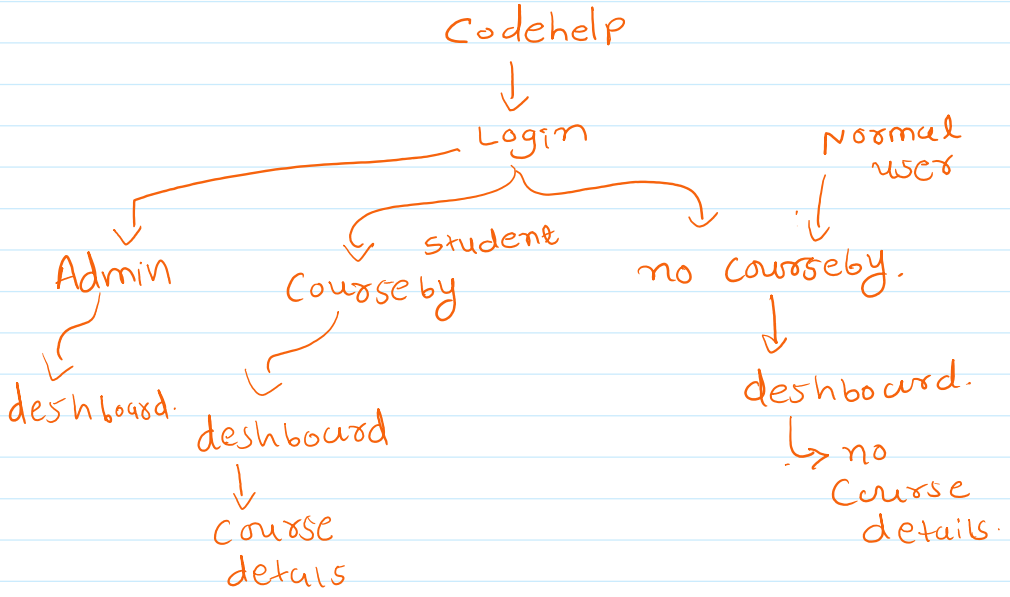
---

middleware : → app.use() → used in add a middleware.

↳ function

↳ adition work any http request

AuthN & AuthZ



Autication :- idatity Verification

Autorization :- access right
permission

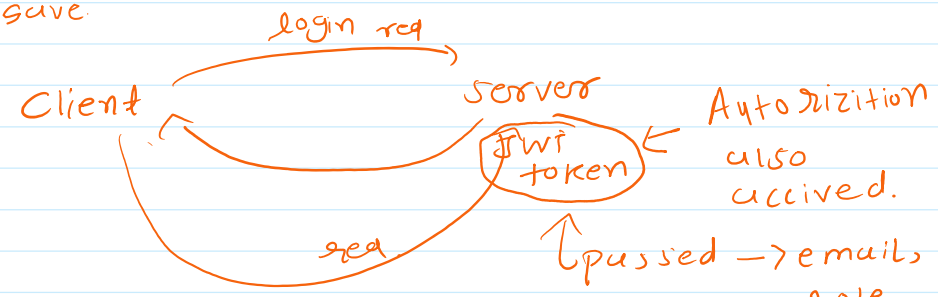
Signup → data fetch → name, email ---
→ validation
→ emailID → already register
↳ db call
→ pwd → hash bcrypty - this library used
→ Model / object → create / save → DB.

login :- email / pwd fetch
→ Validation
→ Check email
→ Password Match
No → Error
yes → sucess

Jwt & cookies → text file.
→ store data
→ local computer save.

Json web token
token genarat
→ not local computer save.

both create in server



login - JWT token

npm i jsonwebtoken

cookie(" '' , value , options)

↑ cookie name — ↑ cookie data — ↑ How many time expried.

login → (1) email / pwd Featch from req body

(2) validation. → email / pwd

(3) Check if user is registerd or (not)

↓ return response

(4) compre password

No → return res

yes → Jwt token create → singin

↳ user.token

→ user.password → hid

→ res.cookie(__ , __ , __)

middlware → function

req → auth

req → admin } authorzition.

req → sturdent.