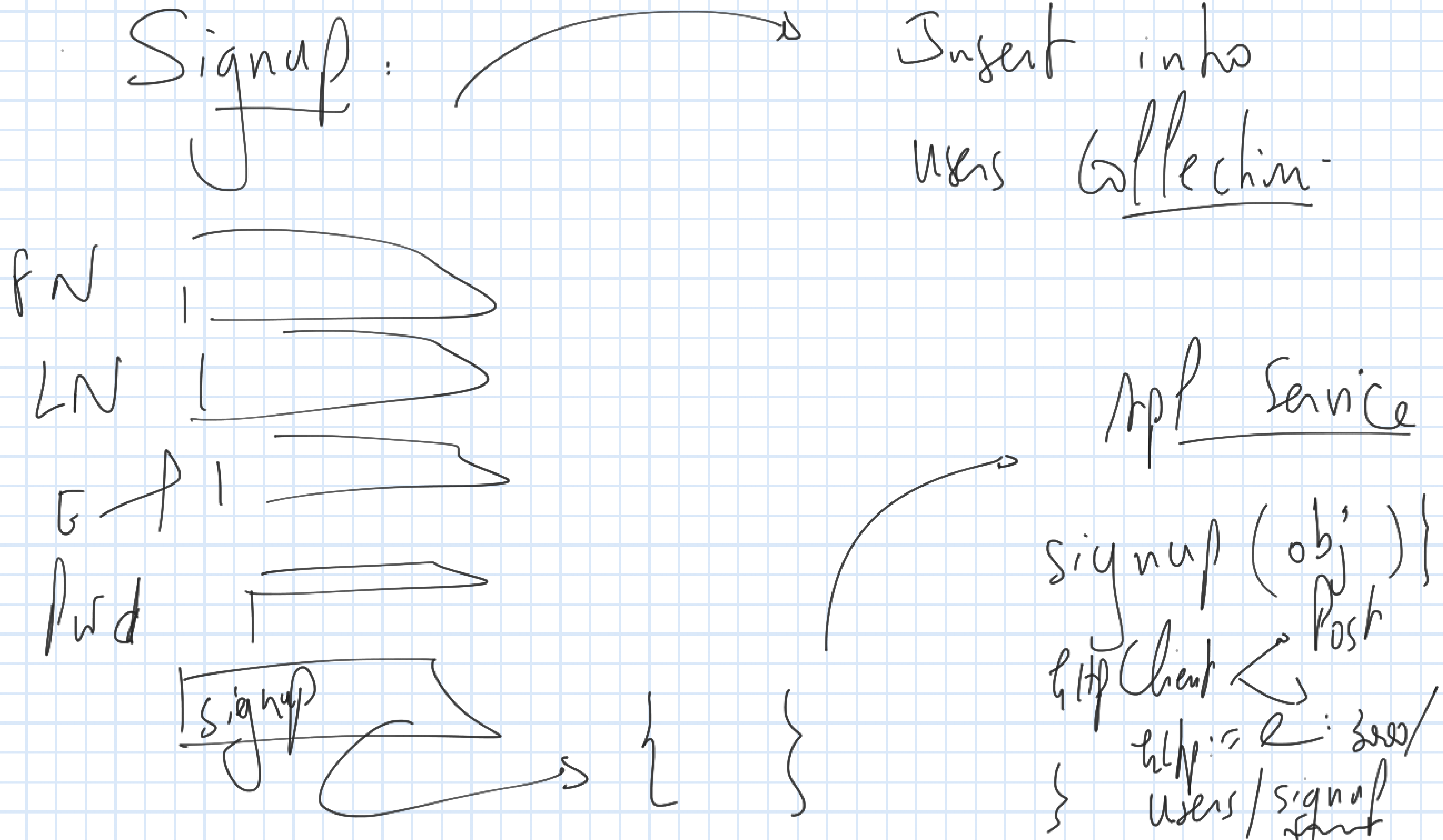
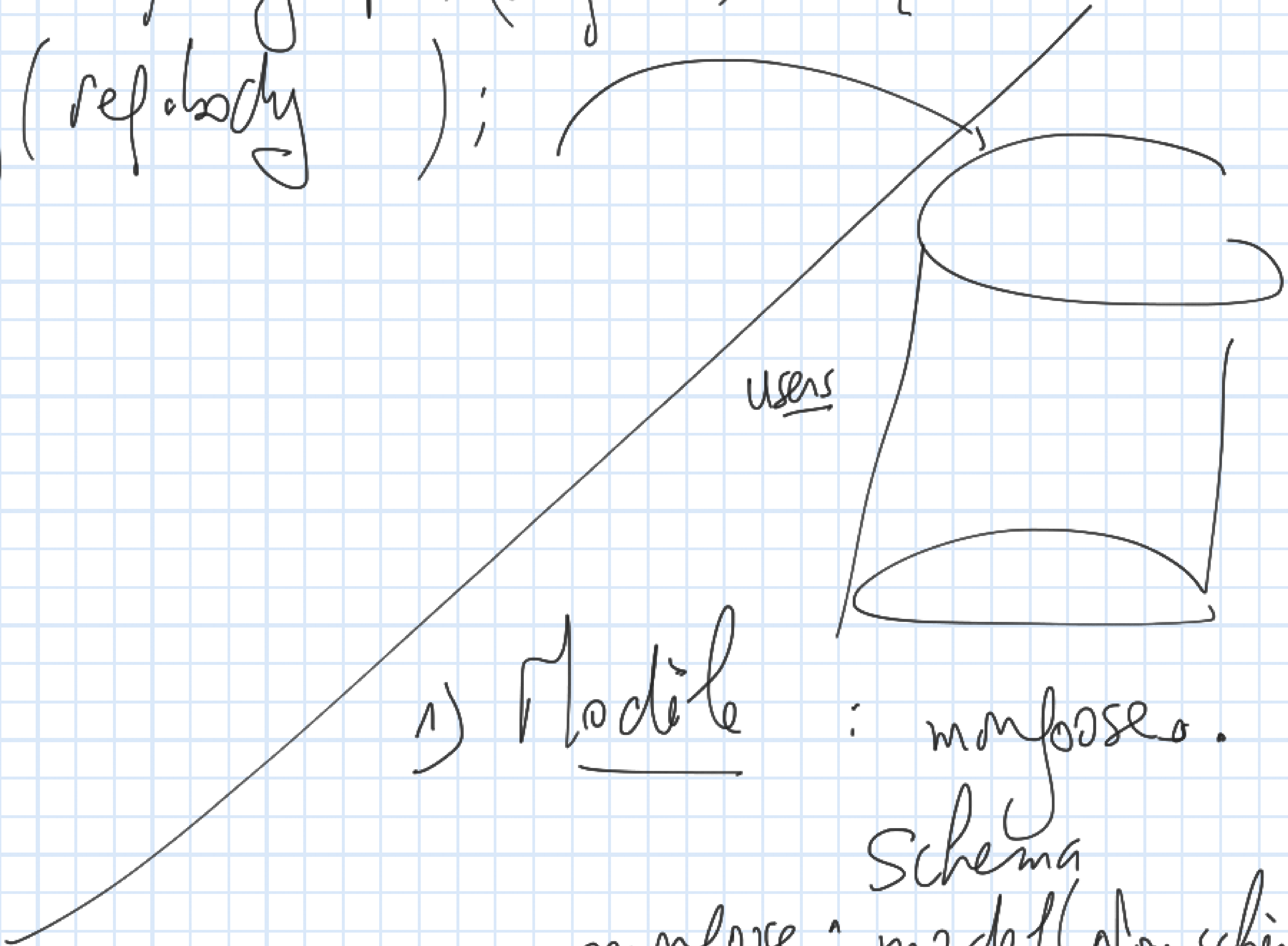


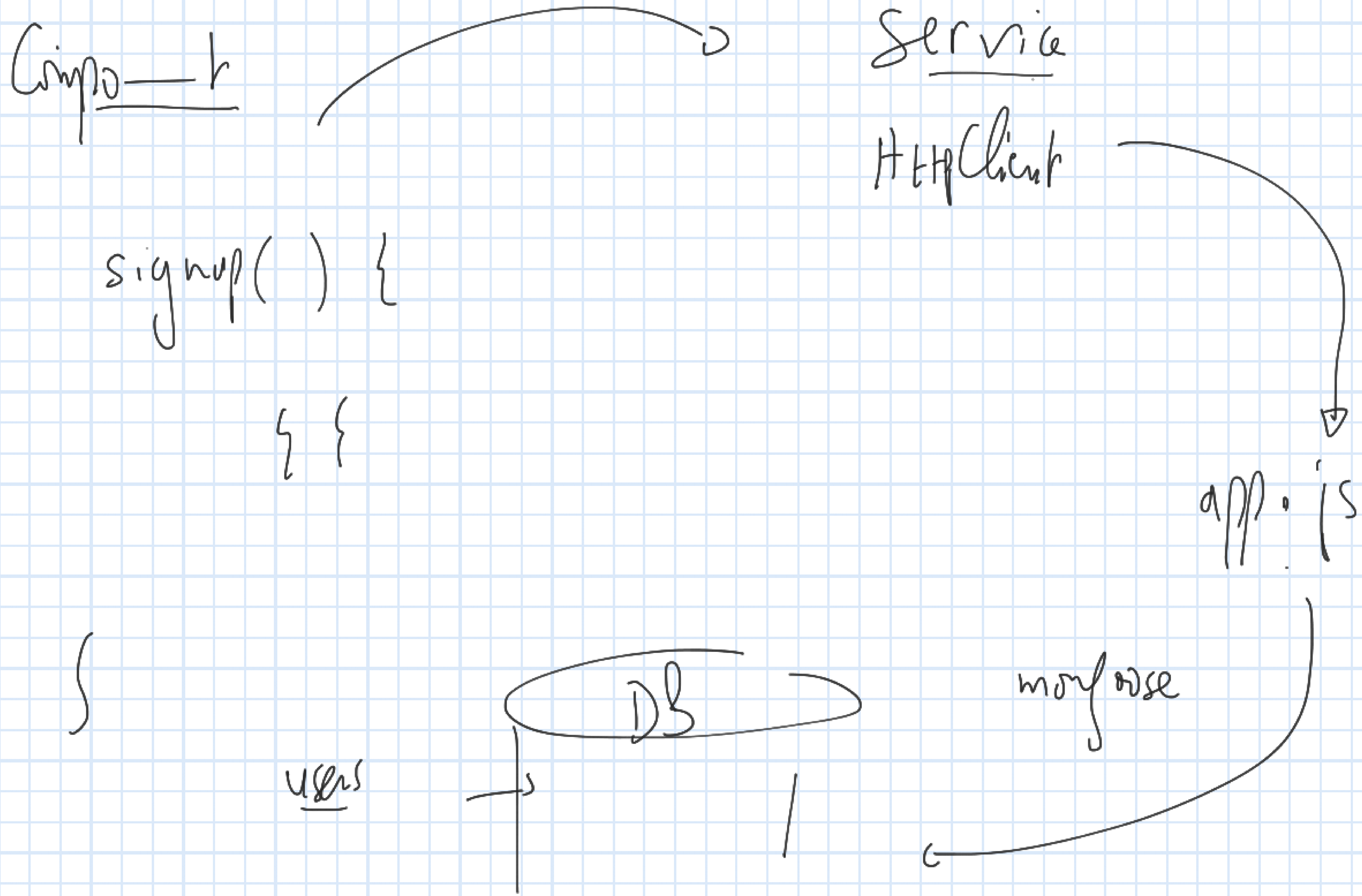
```
signup(obj){
  httpClient  Post
    http://localhost:3000/
    users/signup
}
```

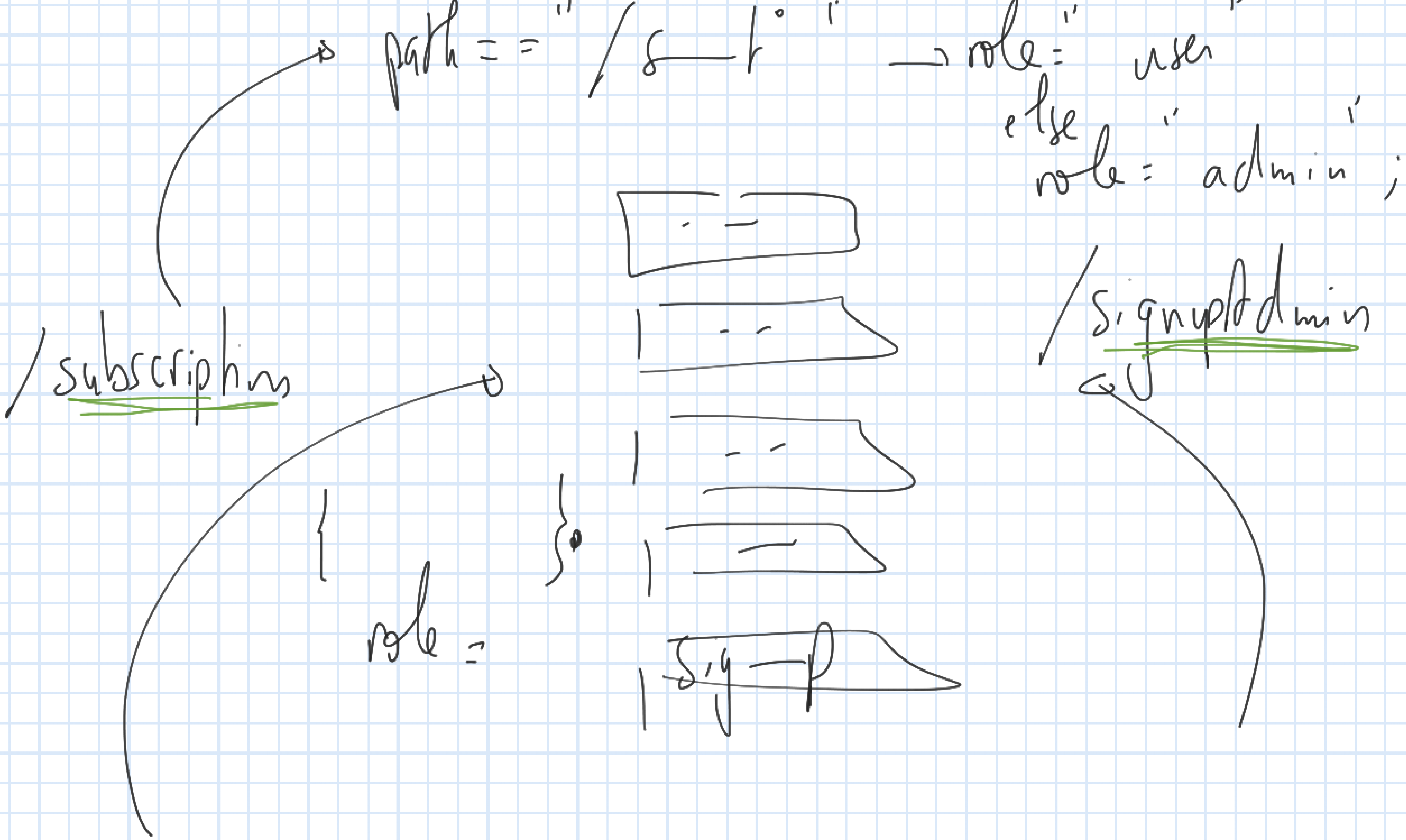
```
app.post("/users/signup", (req,res) => {
    = log(req.body);

});
```



1) Modèle : mongoose.Schema

mongoose.model(Nom, schém)


Compo—r
Service
HttpClient
signup( ) {
{ {
{
app.js
mongoose
DB
users



```
path == "/s—r" → role = "user"
else
role = "admin";
```

1) Compo~~——~~r :

- Déclarer le service ds le constructor

- Apl de la méthode

- .Subscribe (   )

$\downarrow$

Récupérer la réponse du service

$ Se pointer sur le dossier services

$ ng g s user

UserService

email: ---, pwd: -----

Cryptage du pwd

"abcde" ⟹ 1gaXz!edß?---

Bcrypt : un module de cryptage

$ npm i bcrypt | require( )

ch : " - - - "

bcrypt.hash(ch , 10).then()

méthode prédéfinie bcrypt

complexité 1 - 16

abderrahmen : a - z

abd ———— : a-z A-Z

a-z A-Z 0 - 9

db. users . drop()

$\Rightarrow$ delete users collection

$$\boxed{\texttt{db.CollName.drop()}}$$

Uniqueness

- E-mail
- CSN
- Tel
- N° Passport
- username

mongoose-unique-validator

```
checkUnigeEmail(          ){
    // Get All objects
    // for (                  ){
          if (           )
    }
}
```

1) npm i mongoose-unige-validator

2) import (require user.js)

Reactive form

Validators

email

minLength, required

maxLength

pattern

BE

Validators

req.body {f N, – } $\longrightarrow$ user.save( )

```
user.save((err, doc) => {
                         })
```

$\underbrace{\text{err}}$ → erreur

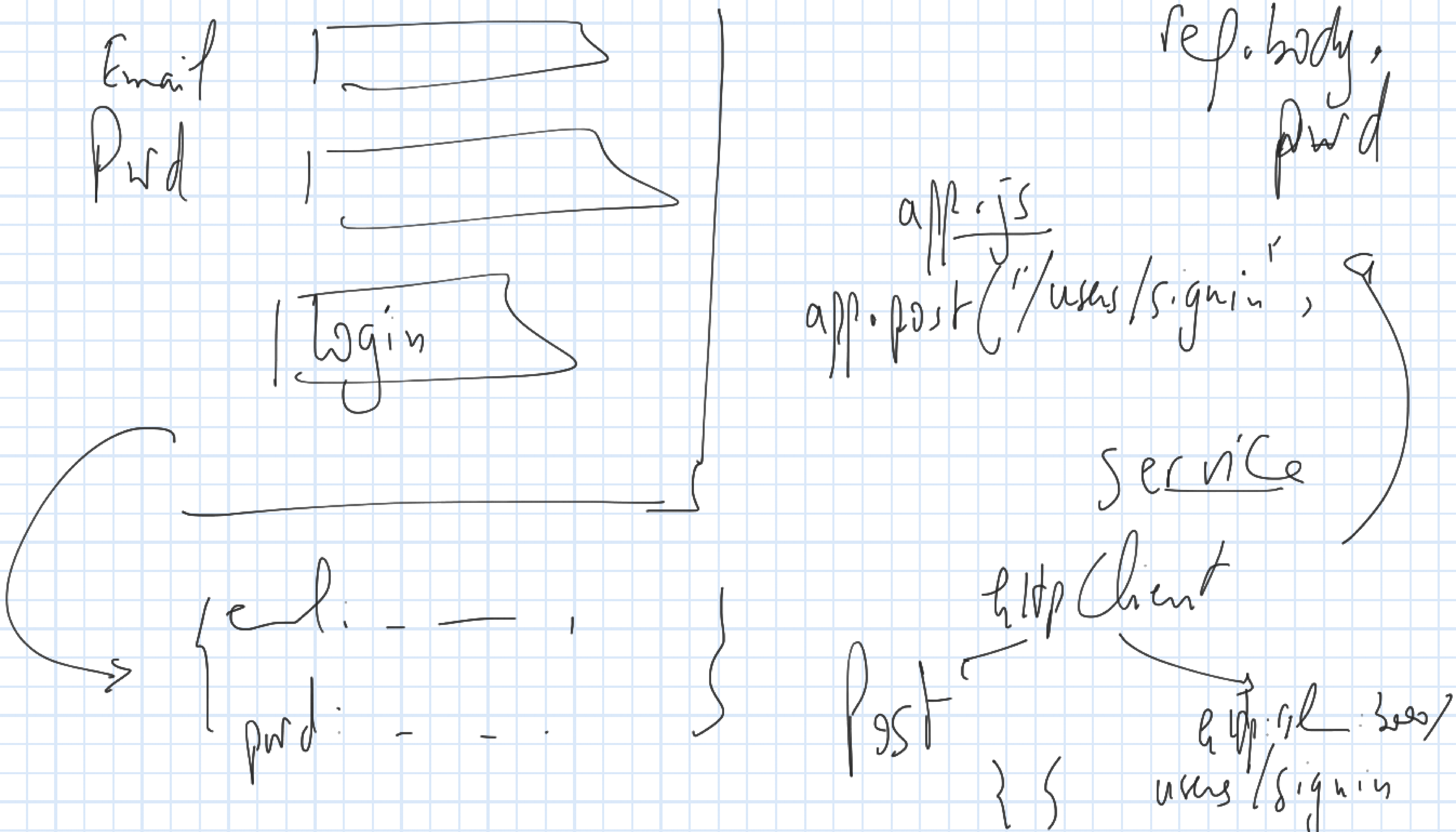
$\underbrace{\text{doc}}$ → document enregistré ds la collect°

$$\begin{tikzcd}
\text{mongoose} & \underline{\text{OK}} \arrow[r, Rightarrow] & \text{doc} \\
\underline{\text{Save}} \arrow[ur] \arrow[r] \arrow[d] & \underline{\text{NOK}} \arrow[r, Rightarrow] & \text{error} \\
\text{méthode prédéfinie mongoose} & &
\end{tikzcd}$$

displayError (cmd, msg, id)

— (true, "OK", "spīd")

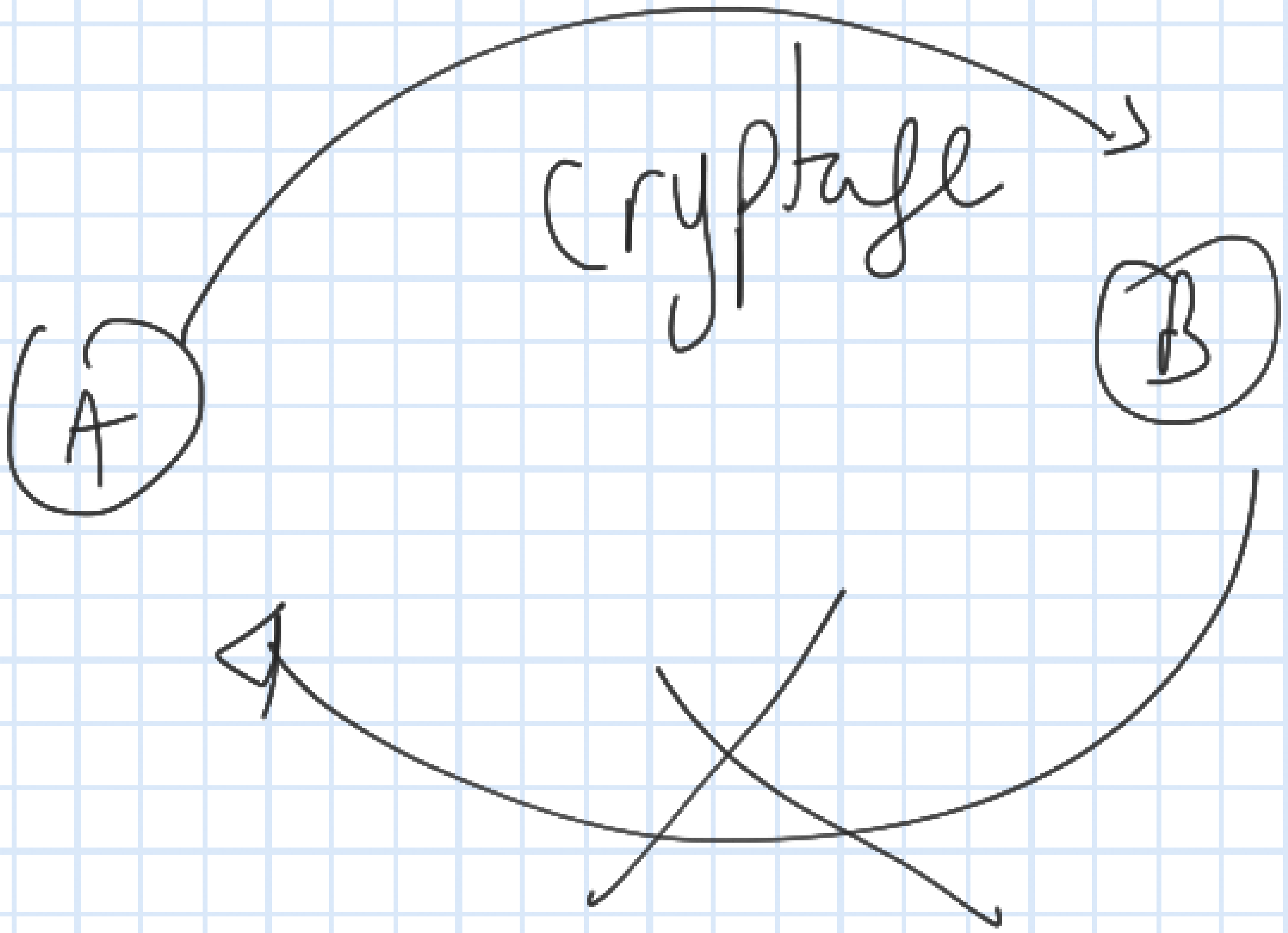
d— (msg, id, c—)


Email
Pwd
Login
{email: ___ ___ ,
pwd: ___ ___ }
req.body.pwd
app.js
app.post("/users/signin",
service
httpClient
Post
{ }
http://l— :3000/
users/signin

Eml | - - - - 1)

Pwd | aaaaaa

| login



.d_get |

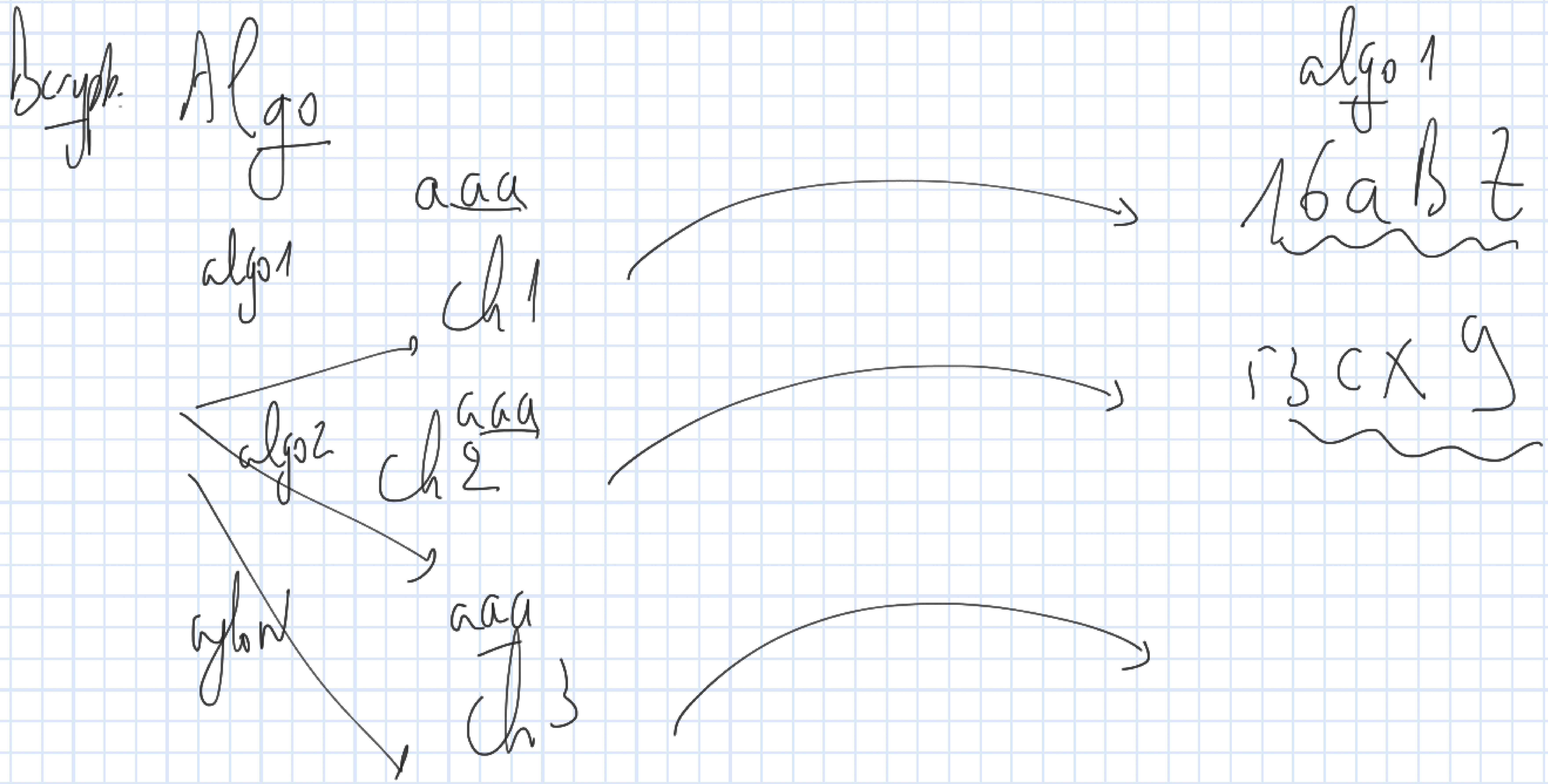
.d_get |

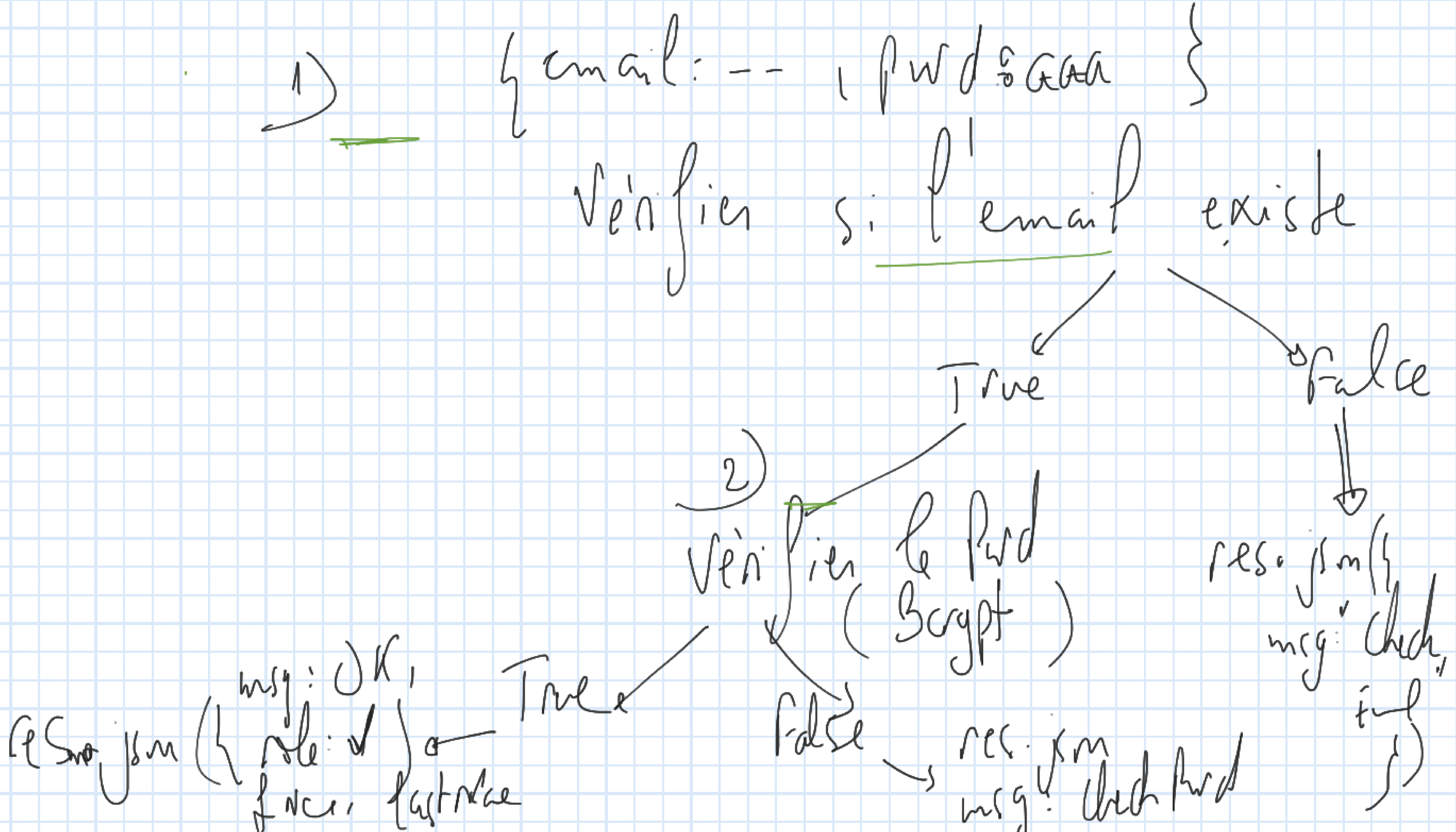
| login

// Get All users

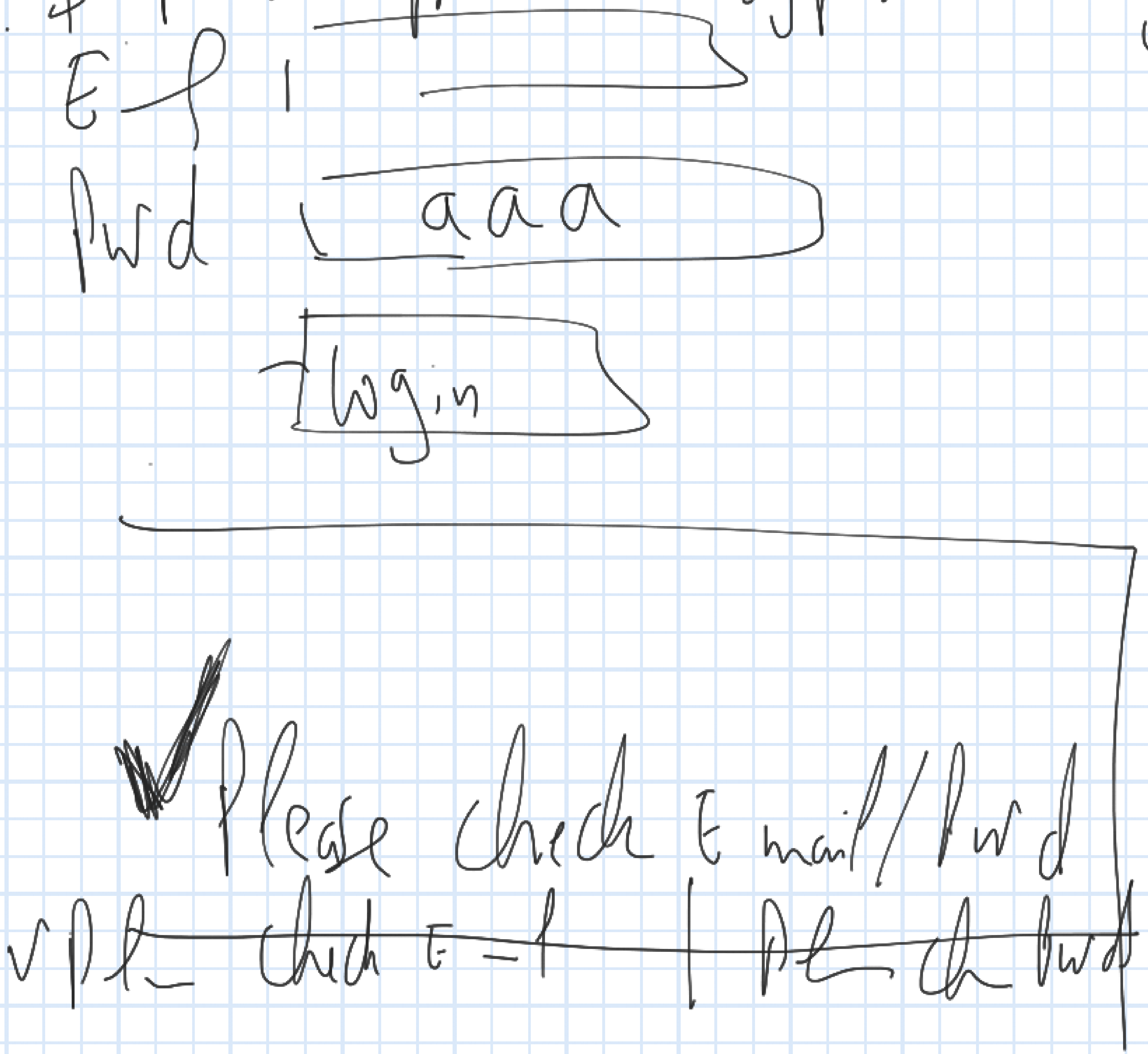
for ( )

if ( )


Bcrypt.
Algo
algo1
algo2
aaa
Ch1
aaa
Ch2
aaa
Ch3
algo1
16aBt
13cxy


1)
{ email: --- , pwd: aaa }
Vérifier si l'email existe
True
False
res.json({ msg: check, fal })
2)
Vérifier le Pwd (Bcrypt)
True
False
resSend.json({ msg: OK, role: ✓, fname, lastname })
res.json msg: check Pwd

```
$ npm i express ␣ body-parser ␣ mongoose@5.13.9 ␣ bcrypt ␣
                                          app.js
```



E-mail

Pwd aaa

login

✓ Please check Email/Pwd

✓ ~~Pl_ check E-mail | Pl_ ch Pwd~~

```
bcrypt.compare("aaa", req.body.pwd)
```

Mot de passe crypté