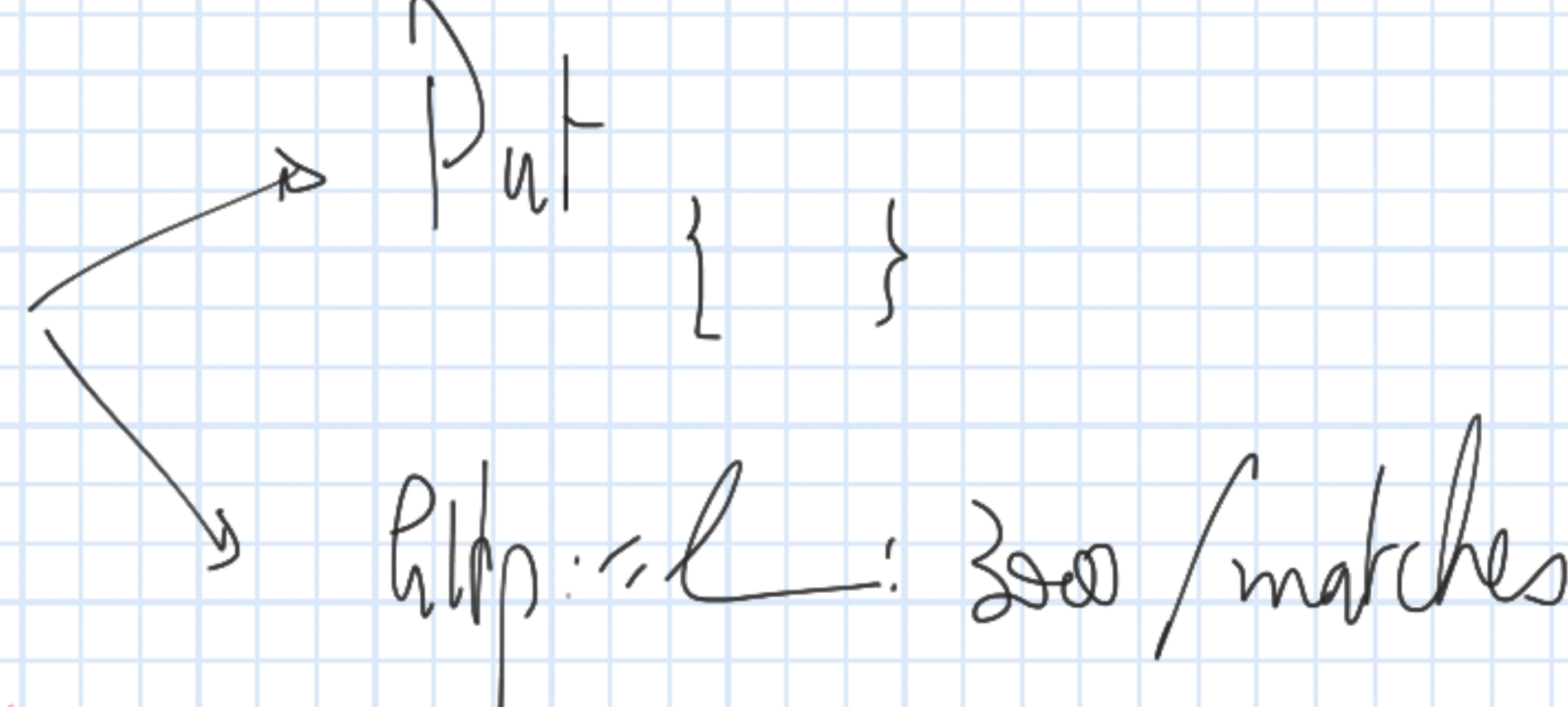


Ref → Delete
→ `http://localhost:3000/matches/id`

// Recupere `id` from `Ref` : `id = ref.player.id`


`Match.deleteOne({-id : id})` . Then

`() ⇒ { }`

Ref:  Put { }
http://localhost:3000/matches

app.put('/matches', (req, res)
let newMatch = req.body;

});

 {
score,
status,
home,
away

Match.updateOne ({ -id: ref.body._id }, newMatch) •

nom du module

méthode prédéfinie mongoose

attribut de la collection

condition de recherche

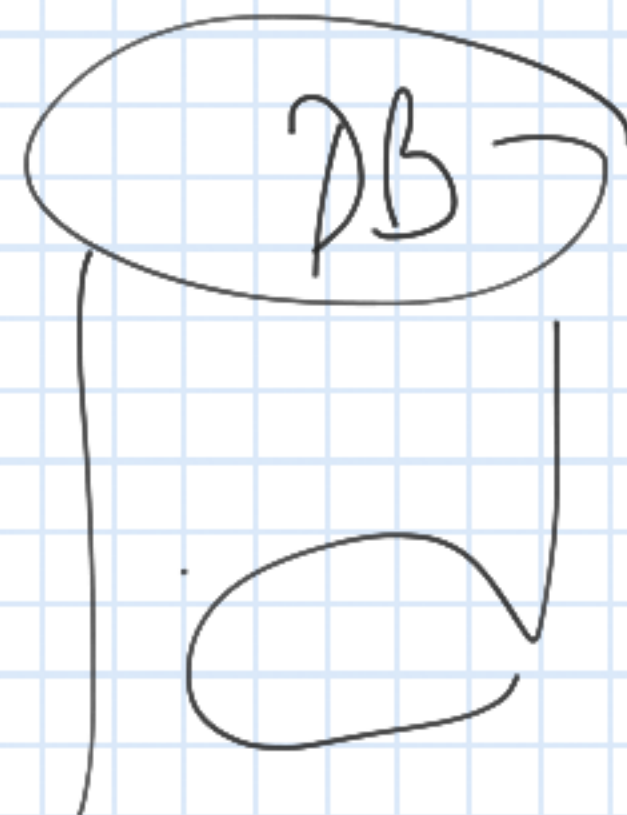
nouvelles valeurs
↓

then ((response) => { log(response) })

Le retour de la modification

App

mongoose . updateOne
deleteOne



response

{ deletedCount: 1

OK,

h

}

FE / Postman

Edit Match

S — 1 | 0

S — 2 | ~~1~~ 2

A — 1 | EST

T — 2 | CA

1 | Edit

$\left\{ \begin{array}{l} s_1: 0, s_2: 2, t_1: \text{EST}, \\ t_2: \text{CA}, \text{id}: \dots \end{array} \right\}$

o

$S1: 0$ / $S1: \underline{2}$

Match.updateOne($\{-id: ref.body.id\}$, $\{S1: ref.body.S1\}$)

att: valeur

(cherche ds la collect^o matches un objet
any \rightarrow un $= id == ref.body.id$)

①. Player

Model (player.js)

name, age, nbr, position
Number String

③ Tester
ls Refs

avec
Postman

@:

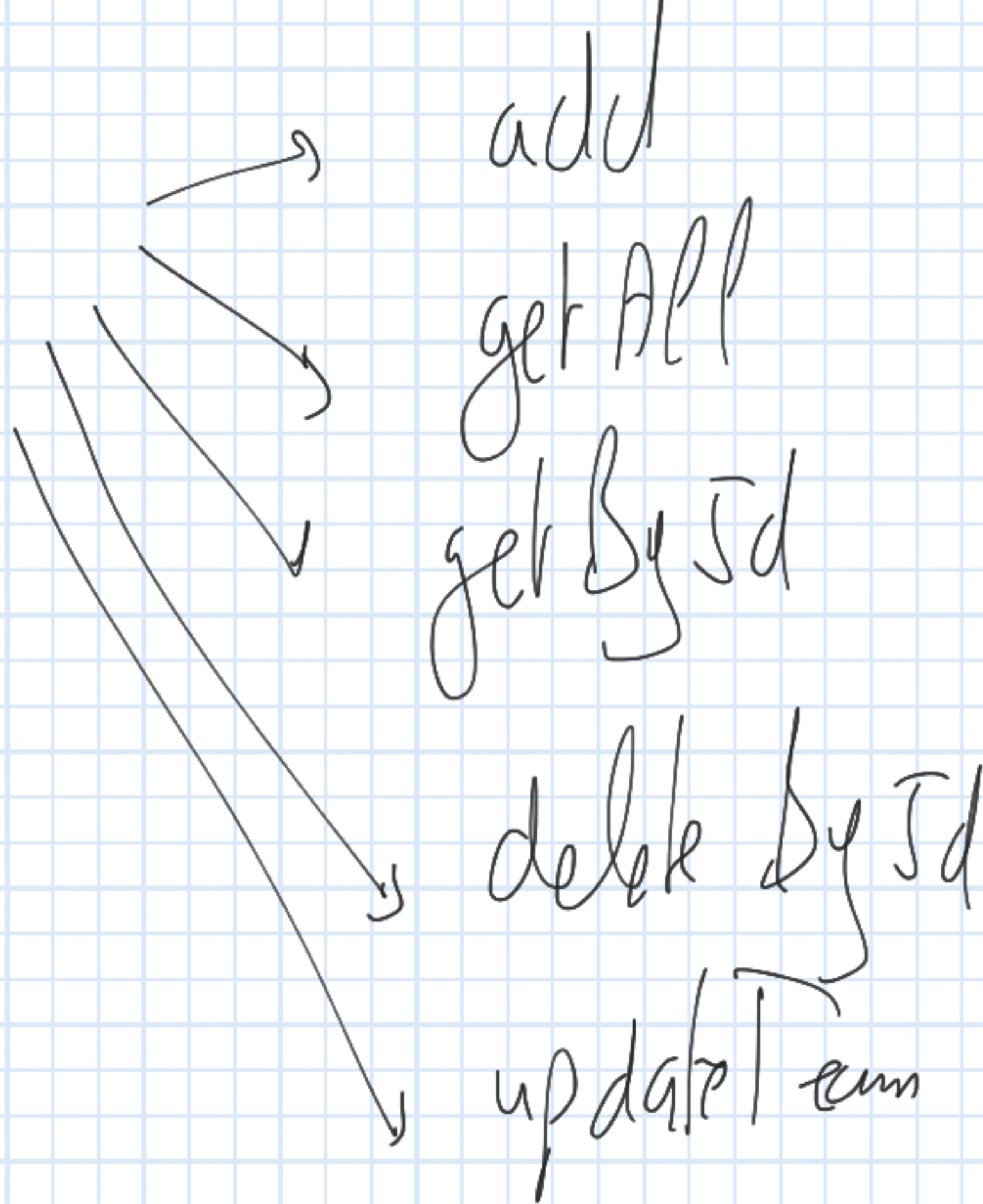
http://localhost:3000/
api/players

@de
Test

② Developper les Business Logics

• add Player, edit Player, get all Player,
get 1 Player by id, delete Player by id

Team



Homework

Resto :

Integrat^o

BE

resDB
Num Database

① Express
body-parser

② Models

Chef → name
→ experience
→ speciality

Tester
+ Postman

③ Business Logic

add, edit, get~~all~~, get by id,
delete by id
Plat, chef

flat → name
→ price
→ description