Atelier 3 :

REST APIs avec db (mysql)

1. Installations des dépendances (mysqljs):

```
houdaifa@houdaifa:~/everything/node/ateliers/atelier3$ npm install mysql

added 10 packages in 5s

22 packages are looking for funding
  run `npm fund` for details
houdaifa@houdaifa:~/everything/node/ateliers/atelier3$
```

2. Configuration de la base de donnees :

```
const mysql = require('mysql')
const db = mysql.createConnection({
    host: 'mysql-houdaifa.alwaysdata.net',
    user: 'houdaifa',
    password: 'hodofozodo',
    database: 'houdaifa_nodejs_rest',

})
```

3. Connection a la base de donnees :

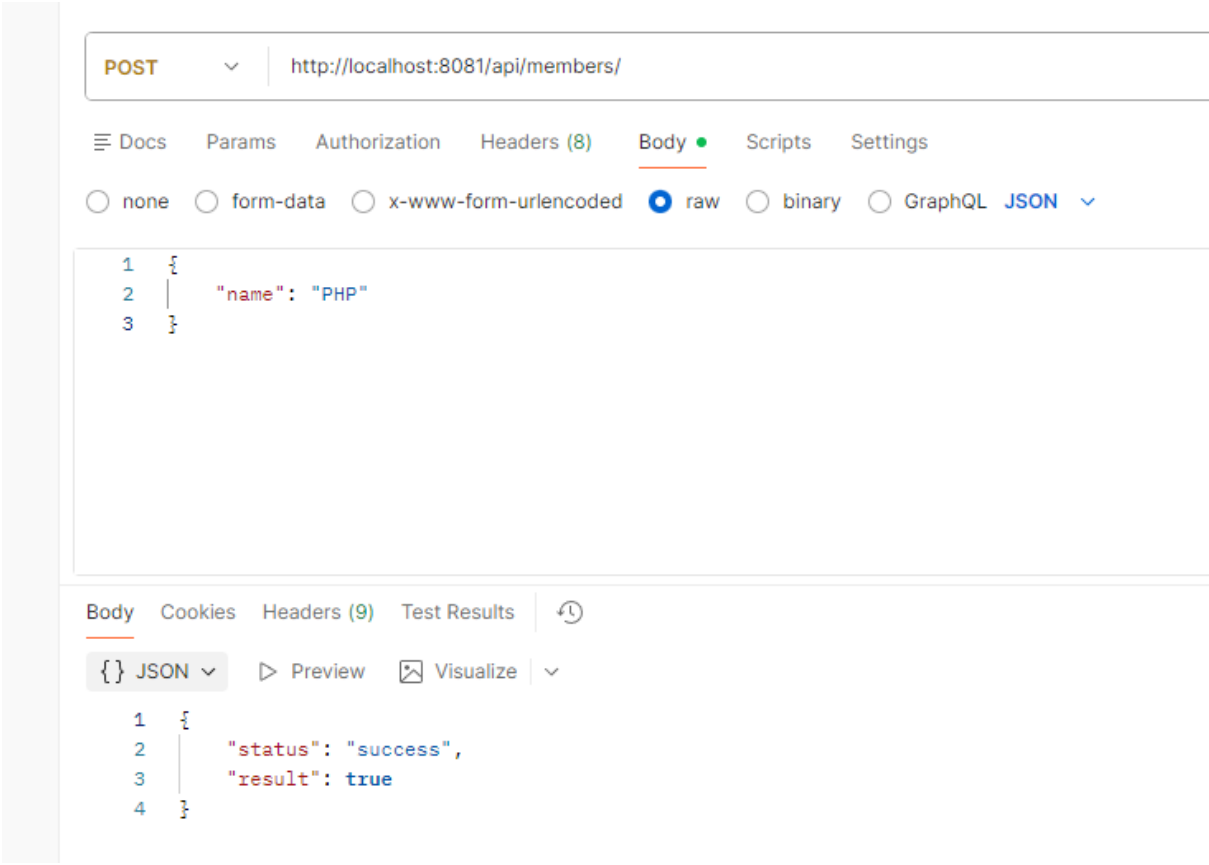
```
db.connect((err) => {
    if (err){
        console.log(err.message)
    }
    else {
        console.log('connected to database')
    }
})
```

4. Des routes :

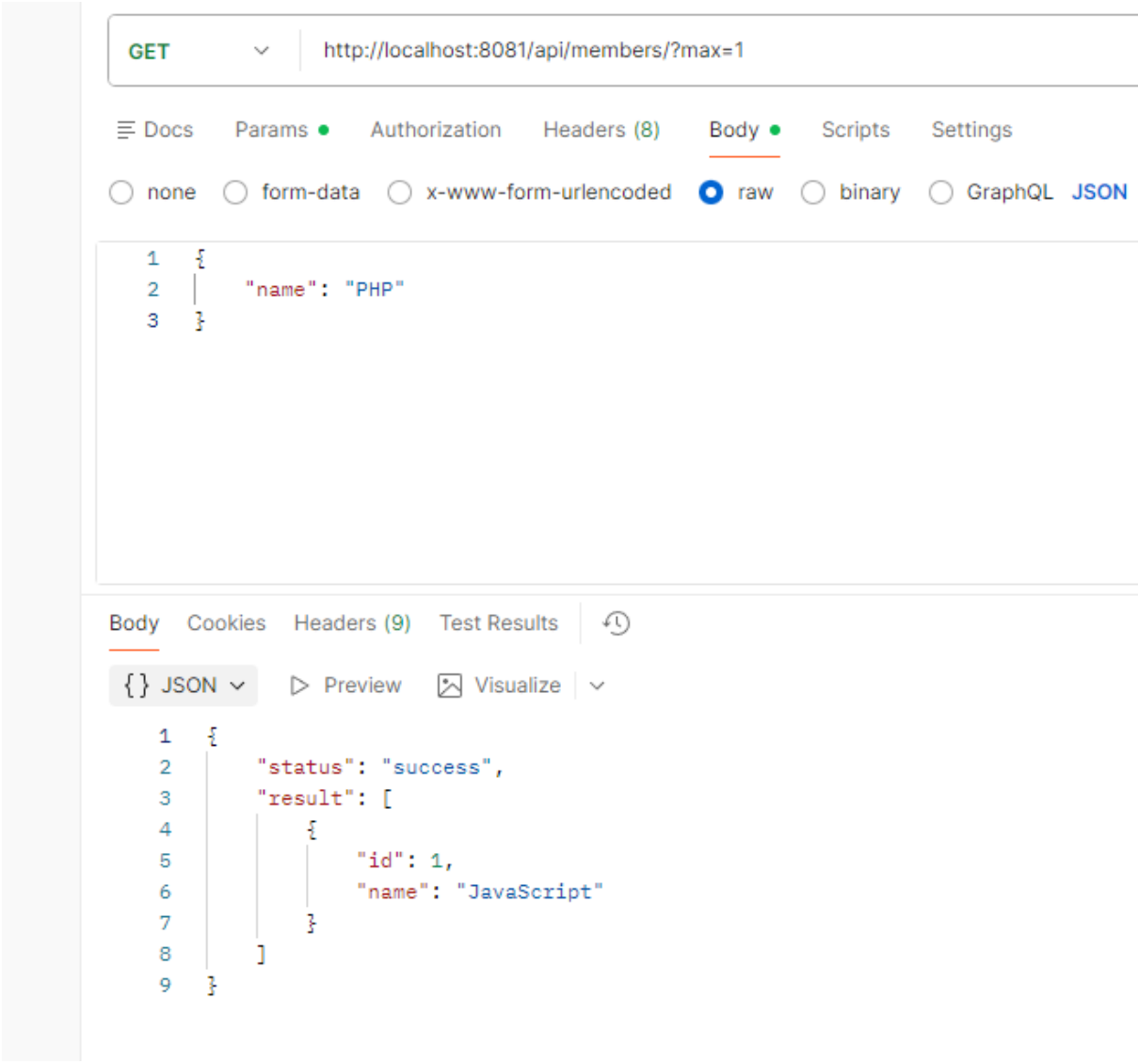
```
membersRouter.route('/:id')
            .get((req,res) => {
                db.query('select * from members where id = ?', [req.params.id], (err, results) => {
                    if (err) return res.json(error(err.message))
                    if (results[0] === undefined) return res.json(error("member not found"))
                    res.json(success(results[0]))
                })
            })

            .put((req,res) => {
                if (!req.body.name) return res.json(error("Missing required parameter: name"))
                db.query('select * from members where id = ?', [req.params.id], (err, results) => {
                    if (err) return res.json(error(err.message))
                    if (results[0] === undefined) return res.json(error("member not found"))

                    db.query('select * from members where name = ? and id != ?', [req.body.name, req.params.id], (err, results) => {
                        if (err) return res.json(error(err.message))
                        if (results[0] !== undefined) return res.json(error("Member name already exists"))

                        db.query('update members set name = ? where id = ?', [req.body.name, req.params.id], (err, results) => {
                            if (err) return res.json(error(err.message))
                            res.json(success(true))
                        })
                    })
                })
            })
            .delete((req,res) => {
                db.query('select * from members where id = ?', [req.params.id], (err, results) => {
                    if (err) return res.json(error(err.message))
                    if (results[0] === undefined) return res.json(error("member not found"))

                    db.query('delete from members where id = ?', [req.params.id], (err, results) => {
                        if (err) return res.json(error(err.message))
                        res.json(success(true))
                    })
                })
            })

        membersRouter.route('/')
            .get((req, res) => {
                if (req.query.max === undefined){
                    db.query('select * from members', (err, results) => {
                        if (err) return res.json(error(err.message))
                        return res.json(success(results))
                    })
                    return
                }
```

5. Tests :

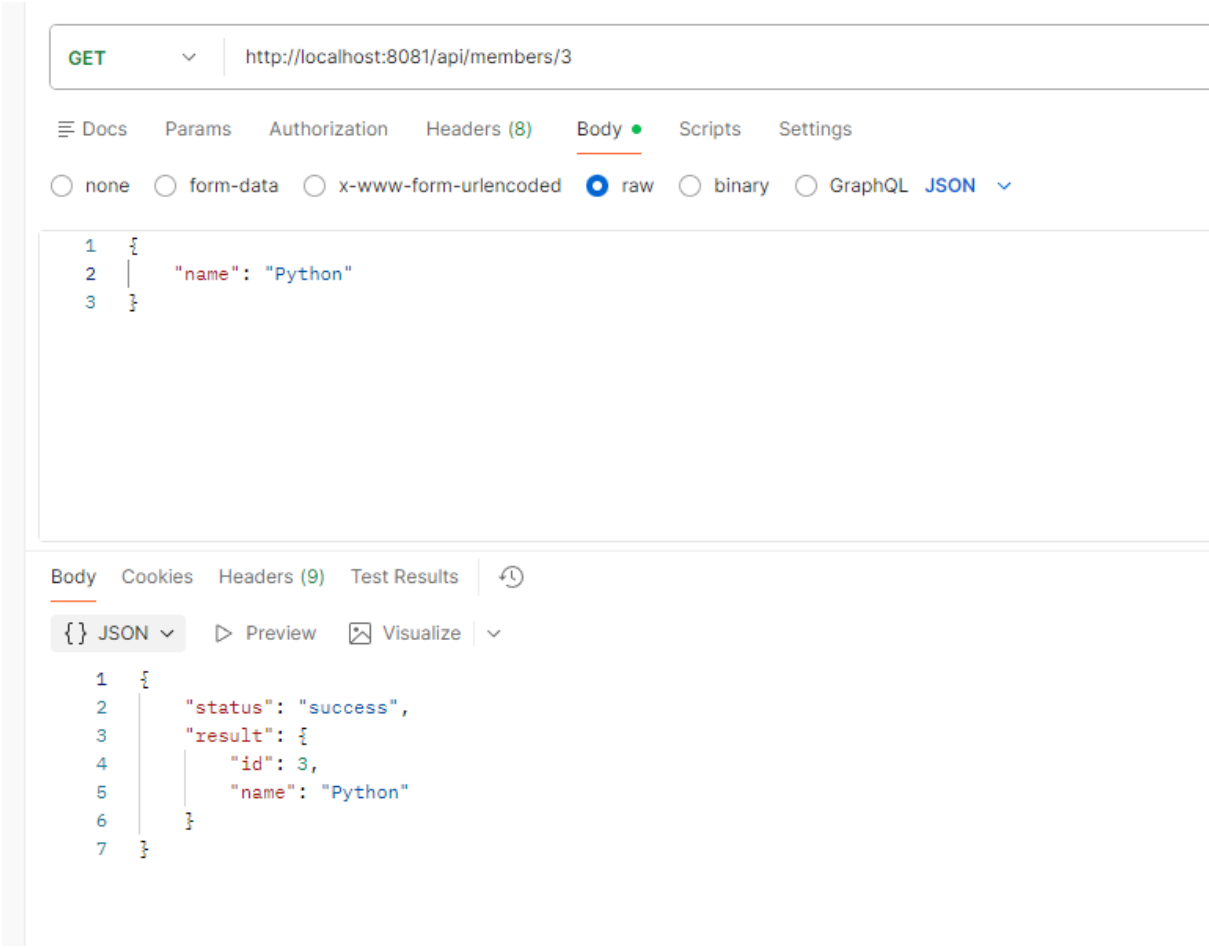
a. Ajouter un membre :



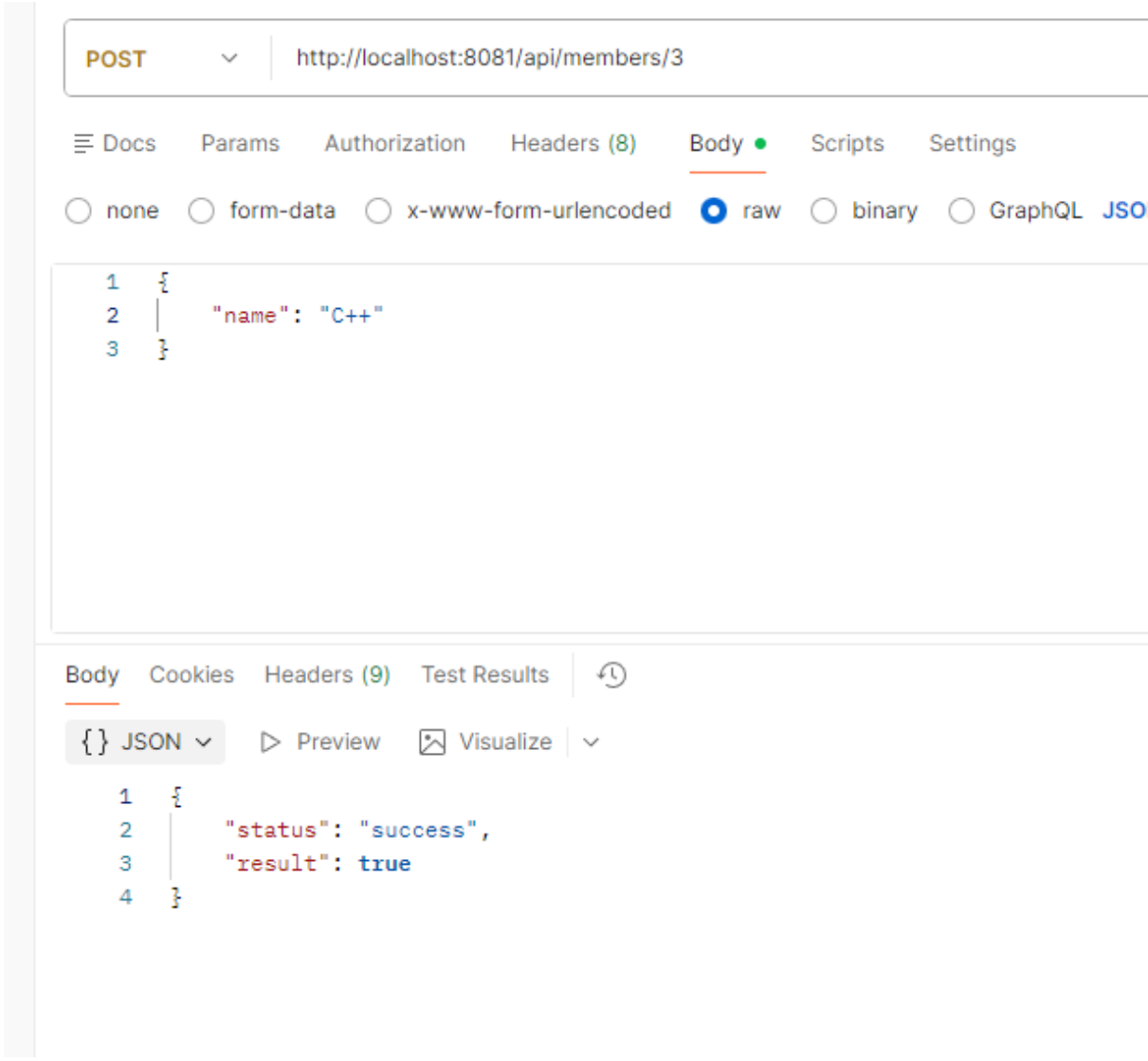
b. Recuperer les membres (avec max) :

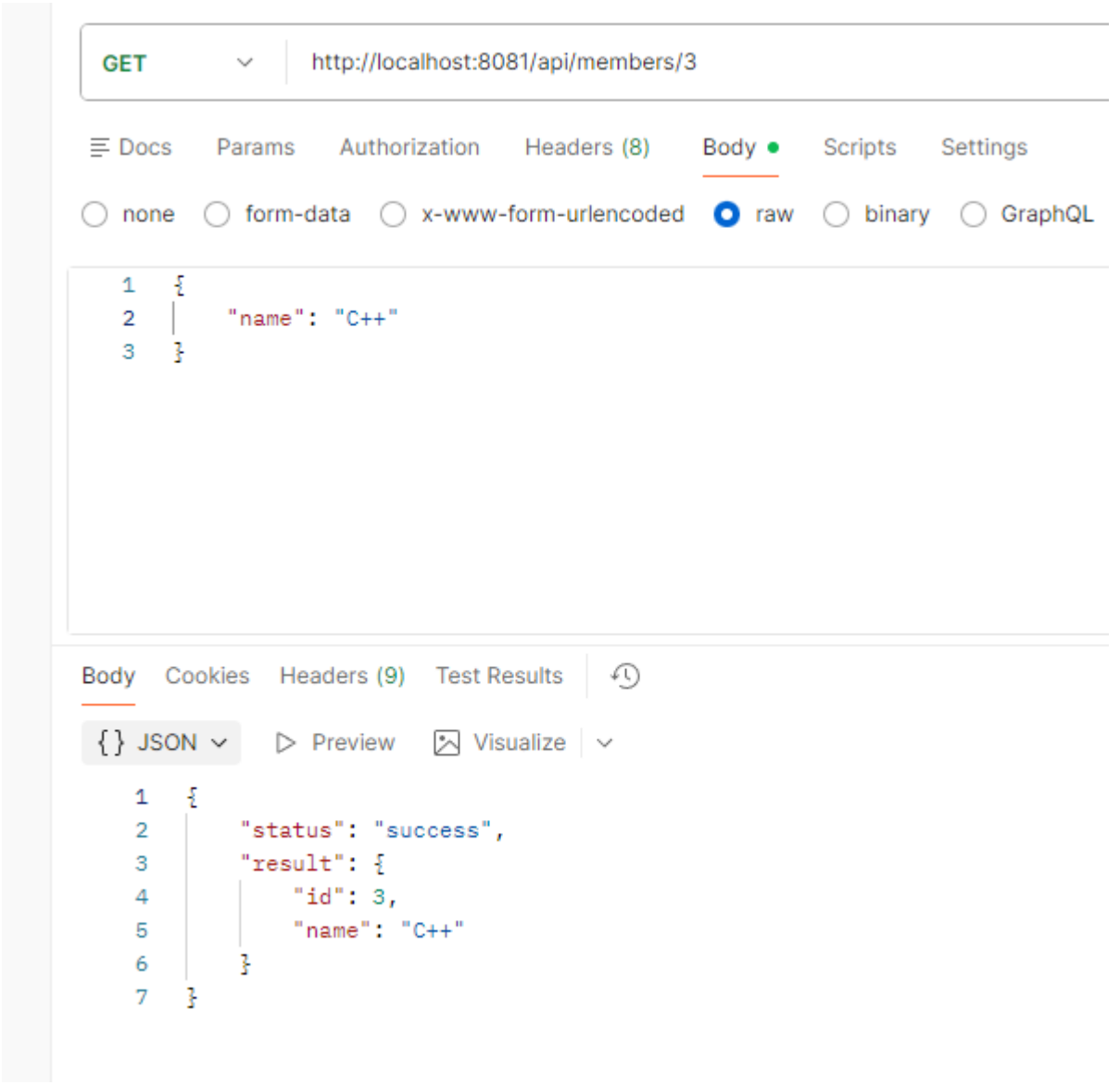


c. Recuperer a membre avec id :

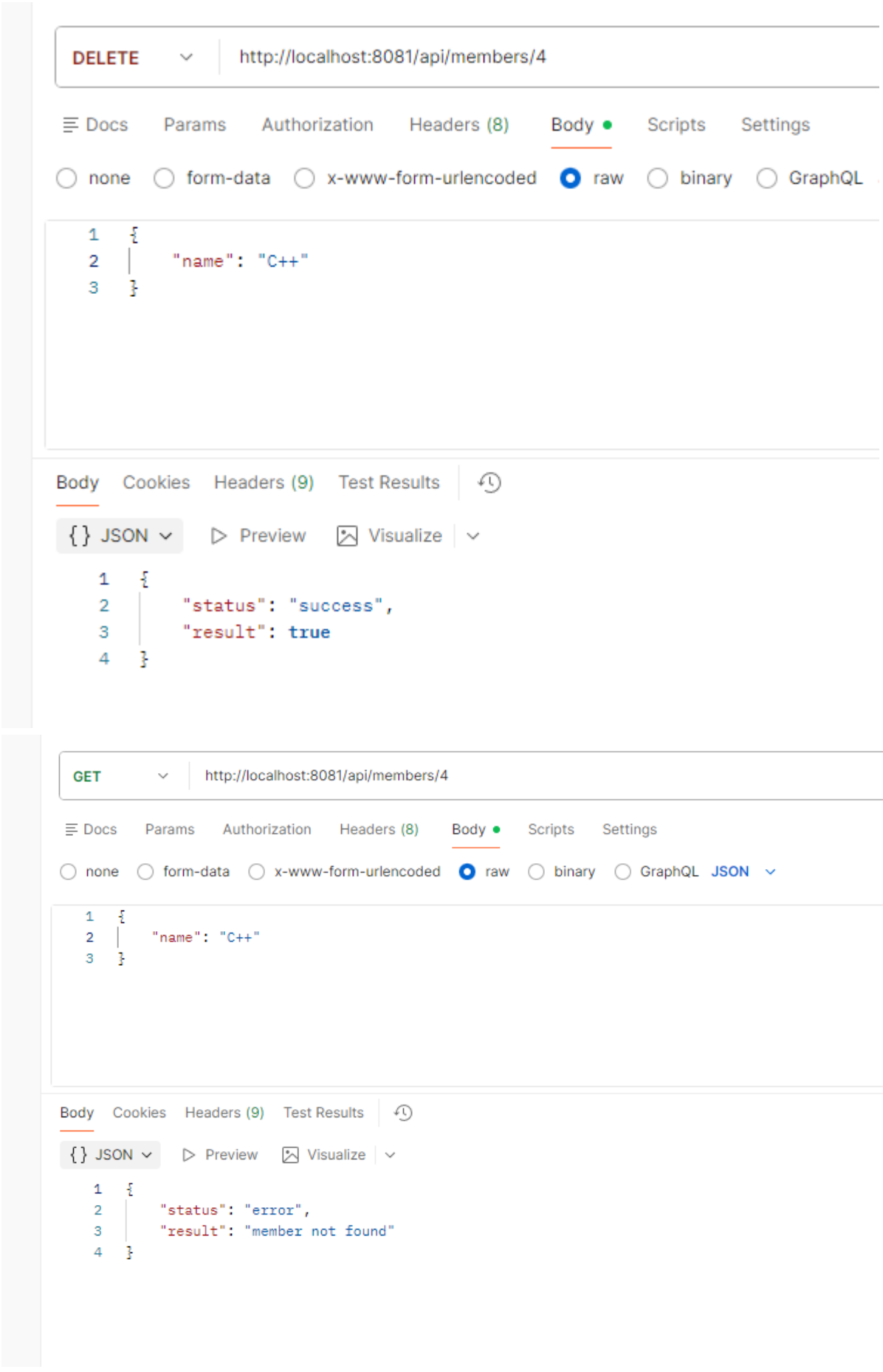


d. Metre a jour un membre :





e. Supprimer un membre :


DELETE
http://localhost:8081/api/members/4
Docs
Params
Authorization
Headers (8)
Body
Scripts
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
{
"name": "C++"
}
Body
Cookies
Headers (9)
Test Results
{ } JSON
Preview
Visualize
{
"status": "success",
"result": true
}
GET
http://localhost:8081/api/members/4
Docs
Params
Authorization
Headers (8)
Body
Scripts
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
{
"name": "C++"
}
Body
Cookies
Headers (9)
Test Results
{ } JSON
Preview
Visualize
{
"status": "error",
"result": "member not found"
}