

Atelier 2:

REST APIs

1. Creation de fichier 'packages.json':

```
houdaifa@houdaifa:~/everything/node/ateliers/atelier2$ npm init -y
Wrote to /home/houdaifa/everything/node/ateliers/atelier2/package.json:

{
  "name": "atelier2",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "type": "commonjs"
}
```

2. Installer les modules (express) :

```
● houdaifa@houdaifa:~/everything/node/ateliers/atelier2$ npm install express
added 65 packages, and audited 66 packages in 5s
22 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
```

3. Fichier functions (success et error) :

```
atelier2 > JS functions.js > ...
1  const success = (result) => {
2      return {
3          status: 'success',
4          result: result
5      }
6  }
7
8  const error = (message) => {
9      return {
10         status : 'error',
11         result: message
12     }
13 }
14
15 exports.success = success
16 exports.error = error
```

4. Fichier index.js :

a. get (read), post (update) et delete par id:

```
7 membersRouter.route('/:id')
8   .get((req,res) => {
9     let id = parseInt(req.params.id)
10    let member = members.find(m => m.id === id)
11
12    if (!member) {
13      return res.json(error("Member not found"))
14    }
15    res.json(success(member))
16  })
17
18  .post((req,res) => {
19    let id = parseInt(req.params.id)
20    let member = members.find(m => m.id === id)
21
22    if (!member) {
23      return res.json(error("Member not found"))
24    }
25
26    for (let i = 0; i < members.length; i++) {
27      if (req.body.name === members[i].name && members[i].id !== id) {
28        return res.json(error("Member name already exists"))
29      }
30    }
31
32    member.name = req.body.name
33    res.json(success(true))
34  })
35  .delete((req,res) => {
36    let id = parseInt(req.params.id)
37    let memberIndex = members.findIndex(m => m.id === id)
38
39    if (memberIndex === -1) {
40      return res.json(error("Member not found"))
41    }
42
43    members.splice(memberIndex, 1)
44    res.json(success(true))
45  })
46
```

b. get (read tous les members), post (create):

```
7   membersRouter.route('/')
8     .get((req, res) => {
9       res.json(success(members))
10    })
11    .post((req, res) => {
12      if (!req.body.name){
13        return res.json(error("Missing required parameter: name"))
14      }
15      for (let i = 0; i < members.length; i++) {
16        if (req.body.name === members[i].name) {
17          return res.json(error("Member name already exists"))
18        }
19      }
20      let newMember = {
21        id: currentId++,
22        name: req.body.name
23      }
24      members.push(newMember)
25      res.json(success(newMember))
26    })
27  
```

5. Tests:

a. Recuperer tous les members:

The screenshot shows a REST API testing interface with the following details:

- Method:** GET
- URL:** <http://localhost:8081/api/members>
- Headers:** Authorization, Headers (8), Body, Scripts, Settings
- Query Params:** Key: Value, Description
- Body:** JSON (empty)
- Response:**
 - Status: 200 OK
 - Time: 9 ms
 - Size: 497 B

```
1 {
2   "status": "success",
3   "result": [
4     {
5       "id": 1,
6       "name": "PHP"
7     },
8     {
9       "id": 2,
10      "name": "JavaScript"
11    },
12    {
13      "id": 3,
14      "name": "Python"
15    },
16    {
17      "id": 4,
18      "name": "Java"
19    },
20    {
21      "id": 5,
22      "name": "C#"
23    }
24  ]
25 }
```

b. Recuperer par ID:

GET | http://localhost:8081/api/members/2

Docs Params Authorization Headers (8) Body ● Scripts Settings

Query Params

| Key | Value | Description |
|-----|-------|-------------|
| Key | Value | Description |

Body Cookies Headers (9) Test Results

{ } JSON ▾ ▶ Preview Visualize

```
1 {  
2   "status": "success",  
3   "result": {  
4     "id": 2,  
5     "name": "JavaScript"  
6   }  
7 }
```

200 OK • 8 ms • 403 B • ⏱️ ⚙️ 🔍 🗃

c. Ajouter un member qui existe déjà (error) :

POST | http://localhost:8081/api/members/

Docs Params Authorization Headers (8) Body ● Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL **JSON** ▾

```
1 {  
2   "name": "JavaScript"  
3 }  
4 }
```

Body Cookies Headers (9) Test Results

{ } JSON ▾ ▶ Preview Visualize

```
1 {  
2   "status": "error",  
3   "result": "Member name already exists"  
4 }
```

d. Ajouter un nouveau membre (success) :

The screenshot shows the Postman interface with a POST request to `http://localhost:8081/api/members/`. The request body is a JSON object:

```
1 {  
2   "name": "C++"  
3 }  
4 }
```

The response body is also a JSON object:

```
1 {  
2   "status": "success",  
3   "result": {  
4     "id": 6,  
5     "name": "C++"  
6   }  
7 }  
8 }
```

e. Supprimer un membre (success) :

The screenshot shows the Postman interface with a DELETE request to `http://localhost:8081/api/members/2`. The request body is an empty JSON object:

```
{ } JSON
```

The response body is a JSON object:

```
1 {  
2   "status": "success",  
3   "result": true  
4 }
```

f. Mettre à jour un membre (success) :

The screenshot shows the Postman interface with the following details:

- Method:** POST
- URL:** <http://localhost:8081/api/members/3>
- Body (JSON):**

```
1 {  
2   "name": "JavaScript"  
3 }  
4 }
```
- Response Body (JSON):**

```
1 {  
2   "status": "success",  
3   "result": true  
4 }
```