API REST

Better safe than sorry.

Francis Munabeno

METHODE GET

Comme vous remarquerez on a importé le système de router d'Express.

Ensuite, on a remplacé app par router et use() par get().

La méthode use() est appelée peu importe le type de requête à notre API.

En revanche, la méthode get() est appelée seulement si le type de notre requête à l'API est de type GET.

```
1   const express = require('express')
2   const router = express.Router()
3
4   router.get('/all', (req, res) => {
5     res.status(200).json({
6       response: "L'élément a bien été récupéré."
7     });
8   })
9
10  module.exports = router
```

Francis Munabeno

METHODE GET

Dans ce middleware nous envoyons ensuite une réponse sous forme de données JSON, avec un code 200 pour une demande réussie.

Si vous effectuez une requête de type GET vers cette route (aussi appelée **endpoint**) à partir de Postman.

Vous recevrez un objet JSON.
La route sera donc http://localhost:300/api/all

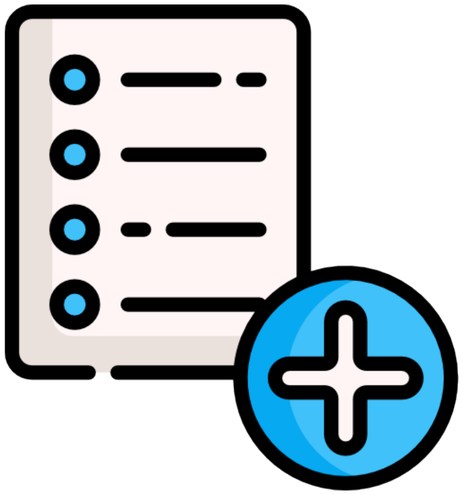
```
1   const express = require('express')
2   const router = express.Router()
3
4   router.get('/all', (req, res) => {
5     res.status(200).json({
6       response: "L'élément a bien été récupéré."
7     });
8   })
9
10  module.exports = router
```

Francis Munabeno

METHODE POST

Pour gérer les requête POST, on a besoin d'en extraire le corps JSON.

Pour cela, vous avez juste besoin d'un middleware très simple, mis à disposition par Express.

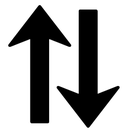


```
1   const express = require('express')
2   const router = express.Router()
3
4   router.get('/all', (req, res) => {
5     res.status(200).json({
6       response: "L'élément a bien été récupéré."
7     });
8   })
9
10  router.post('/add', (req, res) => {
11    console.log(req.body)
12    res.status(201).json({
13      response: "L'élément a bien été ajouté."
14    })
15  })
16
17  module.exports = router
```

Francis Munabeno

Middleware JSON

Il faudra ajouter le middleware qui permet t'intercepter toutes les requêtes :
app.use(express.json());



Ou bien importer le package body-parse, qui fait la même chose que le middleware Mis à disposition par Express :
app.use(bodyParser.json());

```
1   const express = require('express')
2   const app = express()
3
4   const PORT = process.env.PORT || 3000
5
6   // MIDDLEWARE
7   app.use(express.json());
```

```
1   const express = require('express')
2   const bodyParser = require('body-parser')
3
4   const app = express()
5
6   const PORT = process.env.PORT || 3000
7
8   // MIDDLEWARE
9   app.use(bodyParser.json());
```

Francis Munabeno

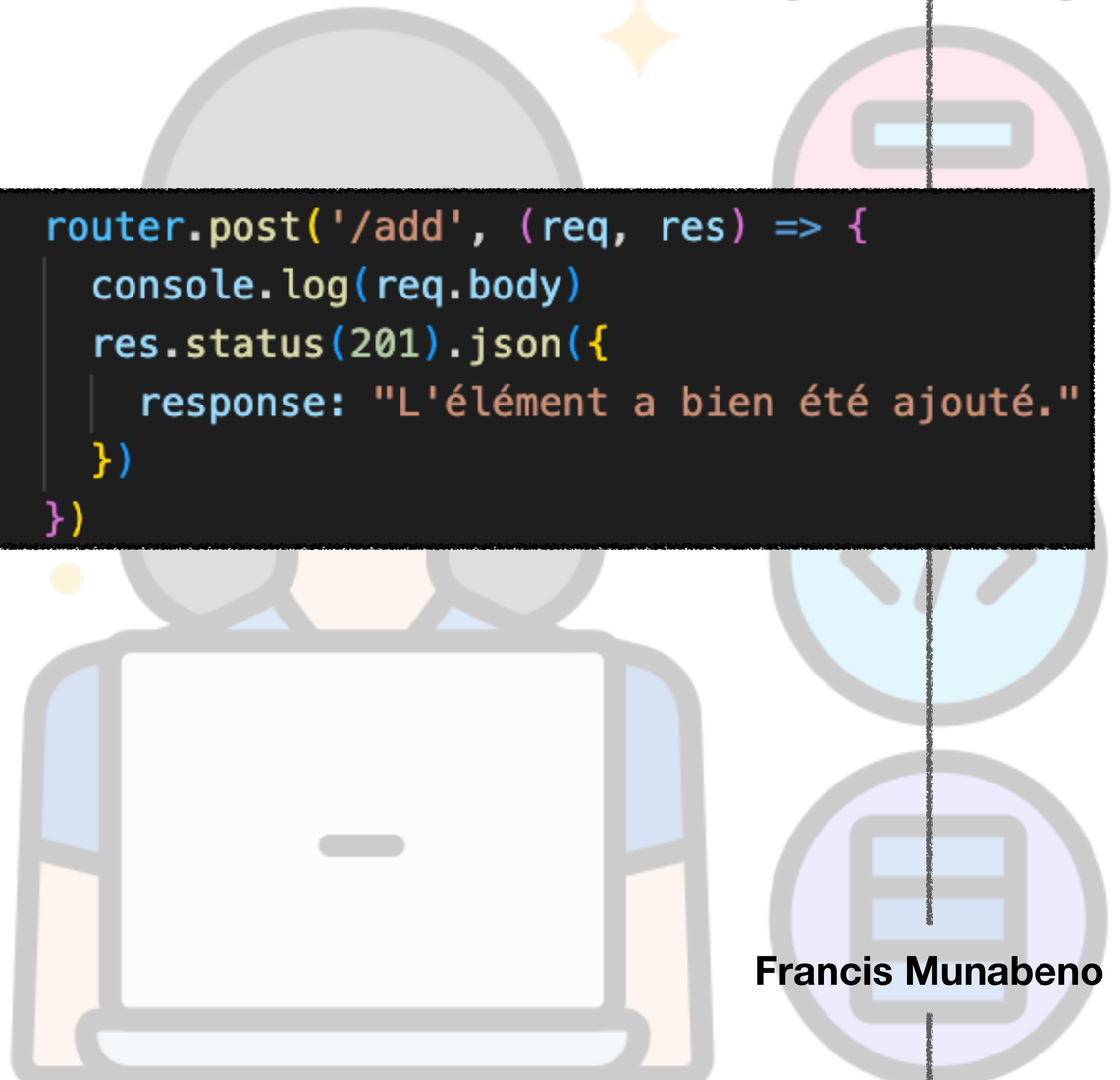
Middleware JSON

Grâce au middleware json(), Express intercepte toutes les requêtes qui ont comme Content-Type applciation/json.

Et met à disposition leur body directement sur l'objet req.

Ce qui nous permet de récupérer les données envoyé à cette url de notre API.

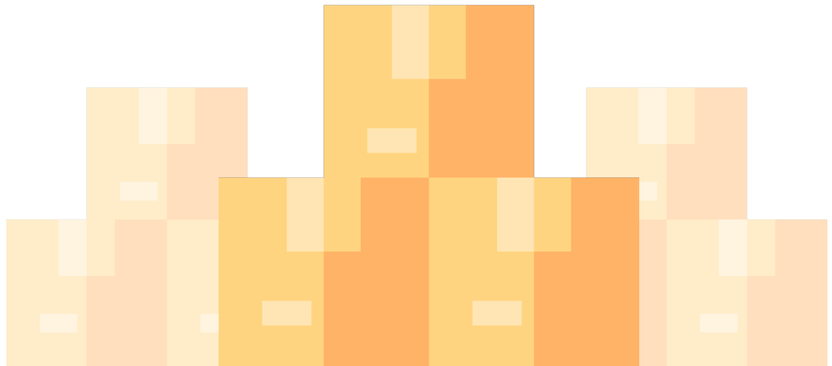
```
10    router.post('/add', (req, res) => {
11      console.log(req.body)
12      res.status(201).json({
13        response: "L'élément a bien été ajouté."
14      })
15    })
```

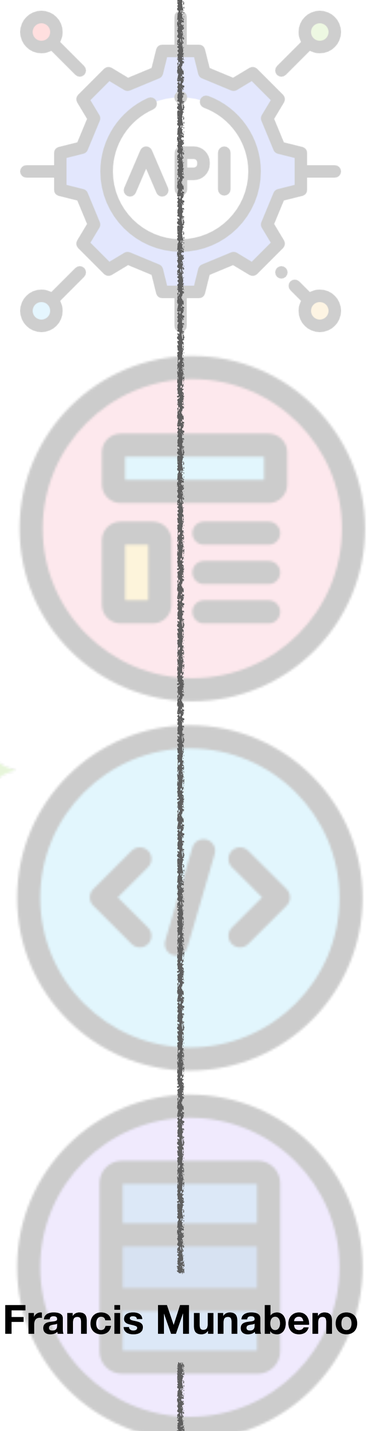


Postman

Postman est un outil permettant de manipuler une API depuis une interface graphique.

Pour plus d'infos : **https://practicalprogramming.fr/postman/**





Francis Munabeno

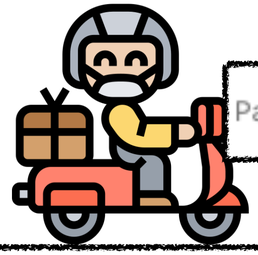
Méthode Post Postman

1 - Select méthode type

2 - Select Body.

3 - Select raw and JSON.

Puis créer un objet json et exécuter votre requête.



Francis Munabeno

Cross Origin Resource Sharing

Cors est un système de sécurité qui par défaut, bloque les appels HTTP entre des serveurs différents.

Ce qui empêche donc les requêtes malveillantes d'accéder à des ressources sensible.