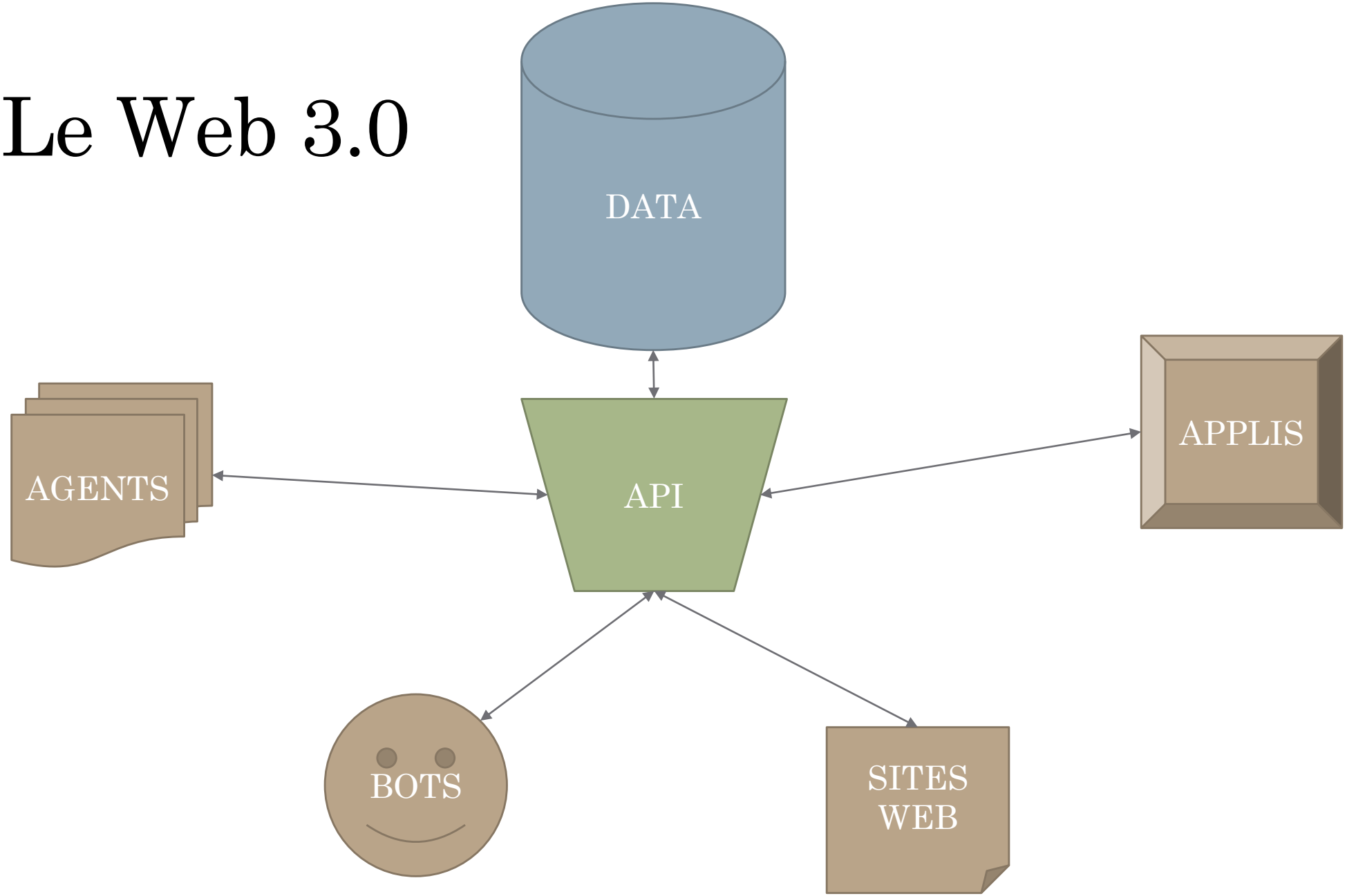
Développer des Applications Web

Et les déployer

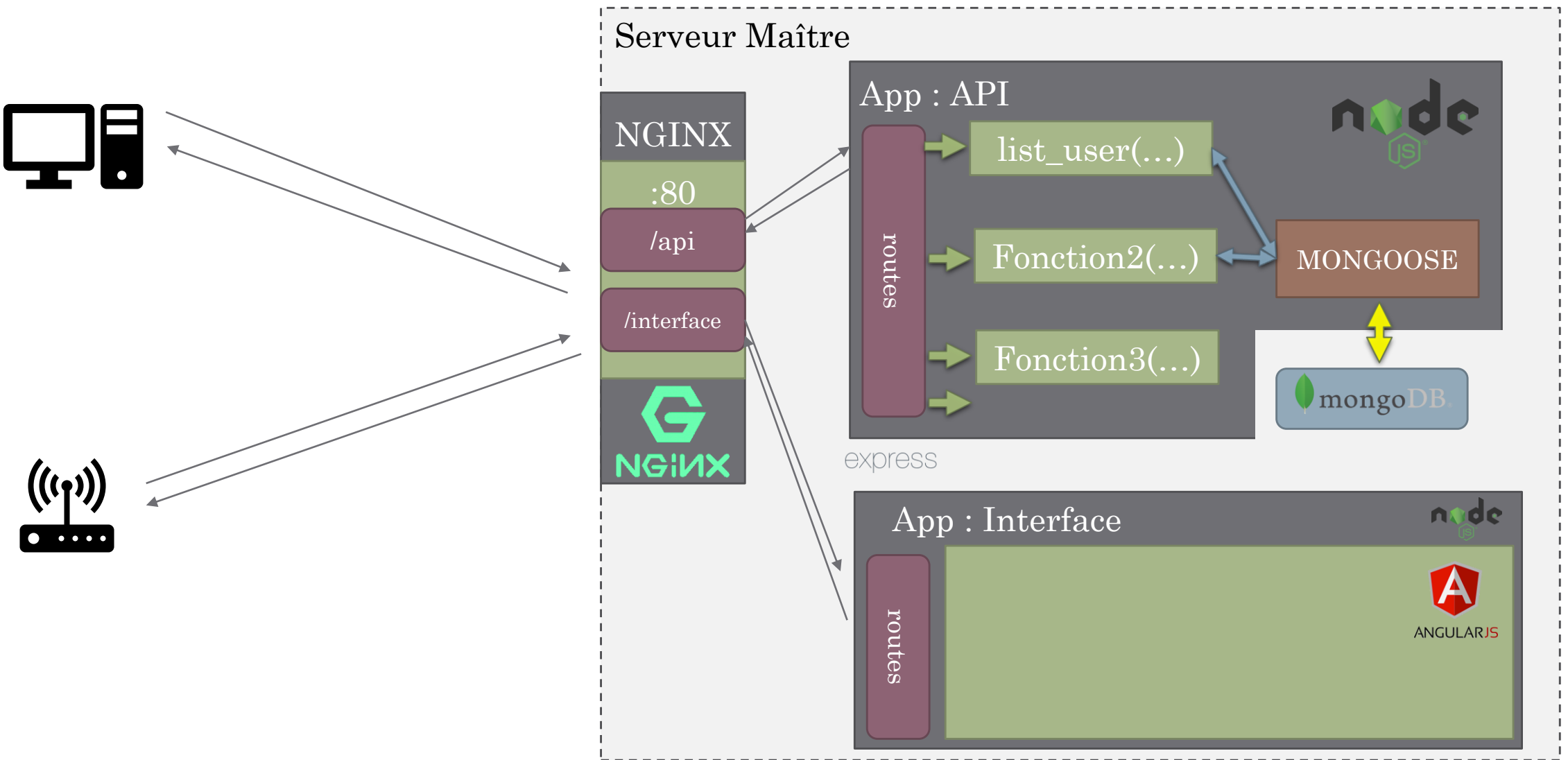
Kevin Thizy, I&S, prom 15

Le Web 3.0



express

Architecture



NODEJS

- Installation :
  - Site node.js (windows, recommandé pour linux)
- Démarrer un nouveau projet
  - `npm init`
- Nodemon permet de relancer automatiquement l'appli à chaque changement de code. L'installer via NPM (gestionnaire de paquets nodejs) :
  - `npm install nodemon –save`
- Le fichier *package.json* contient (entre autres) la liste des dépendances, ajouter la ligne pour démarrer le serveur dans « scripts » : "start": "nodemon server.js"
- Vous lancerez l'appli avec `npm run start`, y accéder à `http://localhost:3000`
- **Configurer Git :**
  - Ajouter un *.gitignore* par défaut pour **node** (via le site github).
  - Première chose à faire quand le projet est cloné : `npm install` (installe les dépendances définies dans le fichier *package.json*)

express

EXPRESS : Bases de l'App

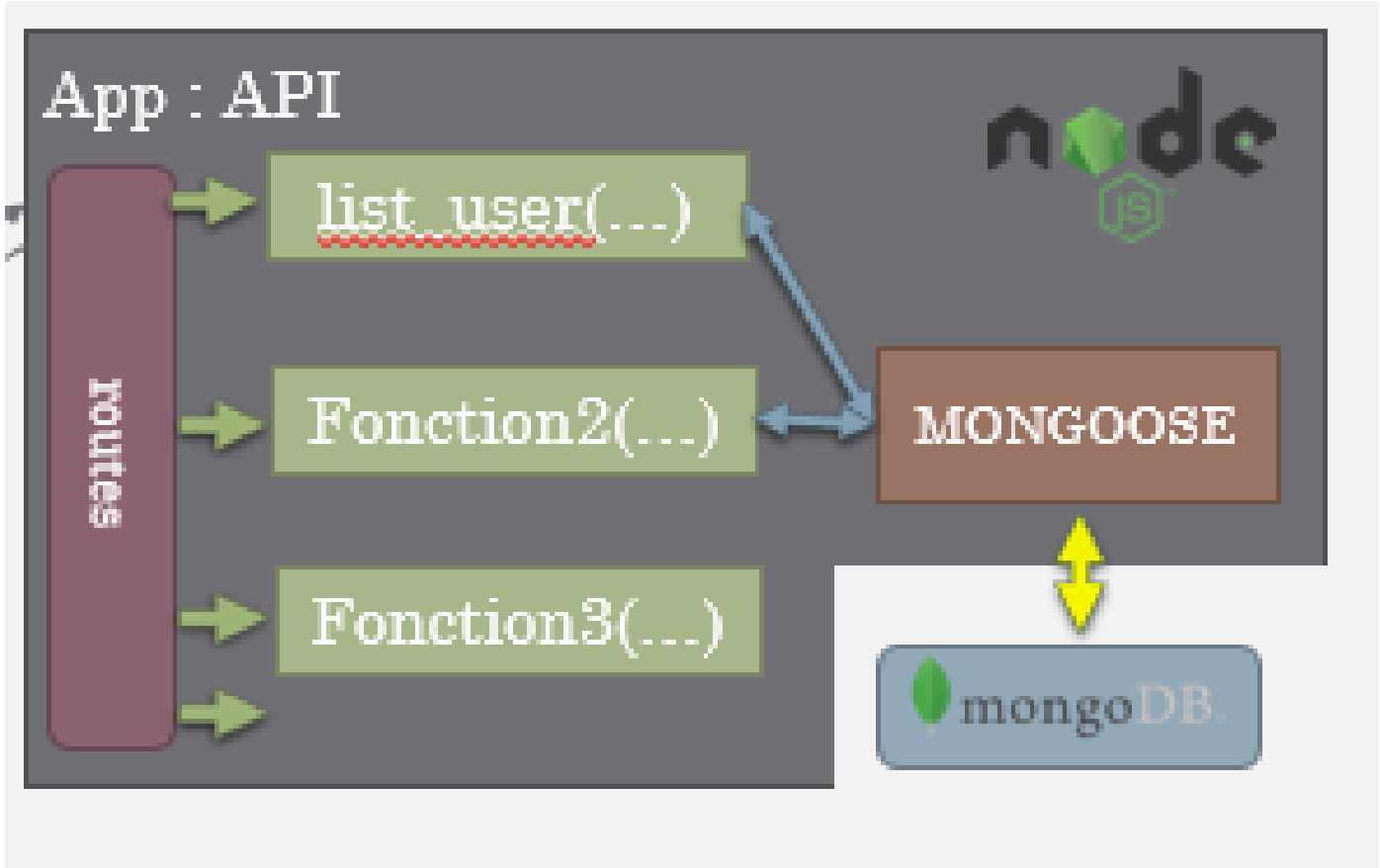
- Installation :
  - Ajouter Express en dépendance du projet avec NPM (gestionnaire de paquets) :
    - `npm install express --save`
- Code de base (créer *server.js*) :

```
var express = require('express'); // import modules
var app = express(); // create application
var port = process.env.PORT || 3000; // define port.

app.get('/', function(req, res) {
        return res.send(200, 'Welcome to my app !');
});

app.get('/json', function(req, res) {
        return res.json({ message: 'Hello, json data !');
});

app.listen(port); // start application on port.
```



Créer une API

Avec MongoDB

Mongoose (base de donnée)

- Installation :
  - Ajouter Express en dépendance du projet avec NPM (gestionnaire de paquets) :
    - `npm install mongoose --save`
- Si vous faites tourner la base mongodb en local, installer mongodb et démarrer le service (cf doc mongodb).
- Connecter la base dans le *server.js* :

```
var mongoose = require('mongoose');

mongoose.connect('mongodb://localhost/database_name);
```

- Créer les Schémas (modèles des données en base) et les Contrôleurs (liste des fonctions qui manipulent les données).
- Ajouter les routes pour appeler les Contrôleurs en fonction de l'adresse.

```
var user = require('./api/models/UserModel');
app.get('/users', user.list_users);
```

Modules supplémentaires

- Pour pouvoir traiter les données sous format json, ajouter body-parser.

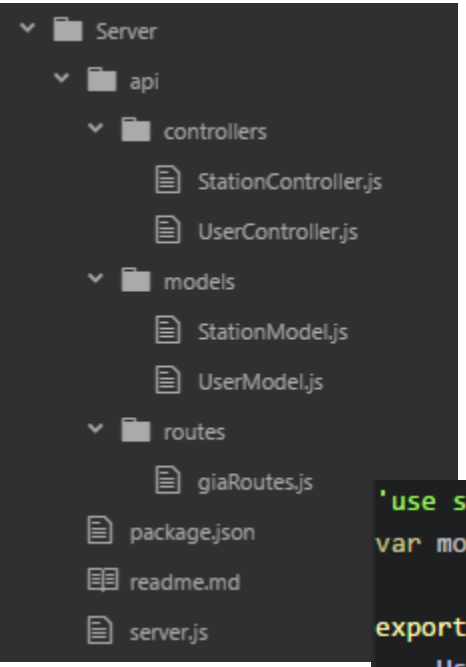
```
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ 'extended': true}));
app.use(bodyParser.json());
```

- Pour des raisons de sécurité, les serveurs refusent les requêtes de type PUT et POST depuis des serveurs externes. CORS permet d'autoriser toutes les entrées ou seulement certains domaines précisés :

```
var cors = require('cors');
app.use(cors());
```

  - Voir la documentation de CORS pour plus d'explication.

Exemple d'API




```
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ 'extended': true}));
app.use(bodyParser.json())

var cors = require('cors');
app.use(cors());

var user = require('./api/models/UserModel');
app.get('/users', user.list_users);
```

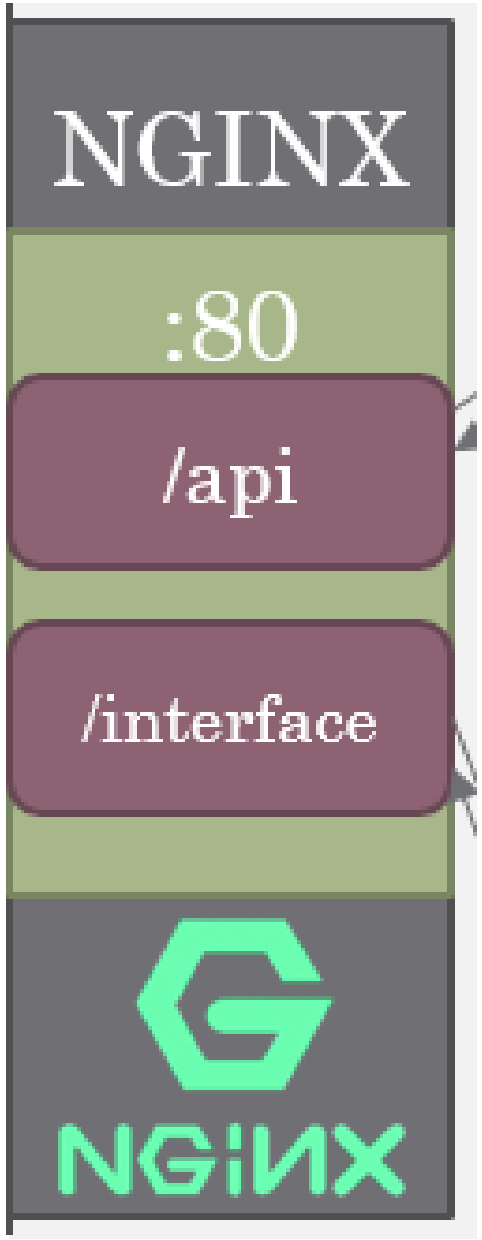
```
'use strict';
var mongoose = require('mongoose'), User = mongoose.model('Users');

exports.list_users = function(req, res) {
    User.find({}, function(err, users) {
        if (err) {
            res.send(err);
        }else {
            res.json(users);
        };
    });
};
```

```
'use strict';
var mongoose = require('mongoose');
var Schema = mongoose.Schema;

var UserSchema = new Schema({
    name: {
        type: String,
        required: 'user should have a name'
    },
    created_date: {
        type: Date,
        default: Date.now
    },
    bt_mac: {
        type: String,
        required: 'A user should have a mac address'
    },
    score: {
        type: String,
        default: '0'
    },
    role: {
      type: String,
      enum: ['mentor', 'master', 'ghosts', 'depinfo', 'others'],
      default: 'others'
    },
    class: {
        type: String,
        enum: ['vampire', 'peon'],
        default: 'peon'
    },
    connects: [{
        date: Date,
        station_id: String
    }]
});


module.exports = mongoose.model('Users', UserSchema);
```



Déployer son appli

Avec NGINX

NGINX

- Pas pratique de devoir entrer le port après l'adresse. Pour changer : NGINX.
- Installation :
  - Sous Linux : paquet nginx
- Mise en route rapide :
  - Effacer la configuration par défaut : `sudo rm /etc/nginx/sites-enabled/default`
  - Créer la configuration : `sudo nano /etc/nginx/sites-available/node` (ou éditeur préféré) :

```
server {
    listen 80;
    server_name mondomaine.fr;

    location / {
        proxy_set_header   X-Forwarded-For $remote_addr;
        proxy_set_header   Host $http_host;
        proxy_pass         "http://127.0.0.1:3000";
    }
}
```

  - Synchroniser les sites disponibles avec les sites actifs (simplification) : `sudo ln -s /etc/nginx/sites-available/node /etc/nginx/sites-enabled/node`

NGINX

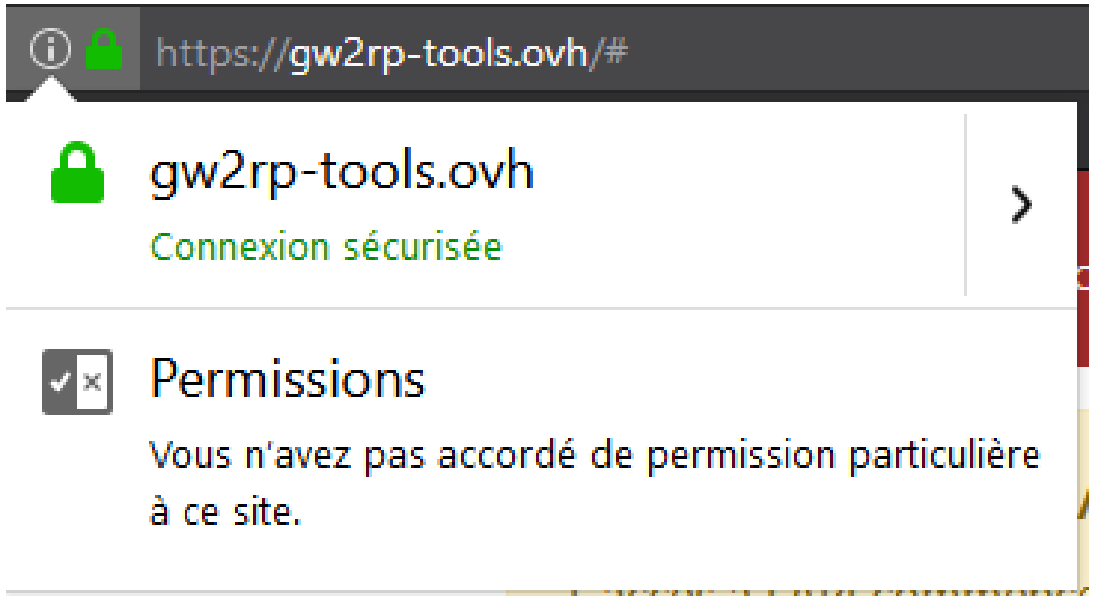
- NGINX démarre automatique au démarrage de la machine.
- Pour redémarrer le service après une modification de la configuration :
  `sudo service nginx restart`
- NGINX permet aussi de rediriger vers des sites statiques html :

```
location / {
        proxy_set_header   X-Forwaded-For $remote_addr;
        proxy_set_header   Host $http_host;
        root /path/to/site/folder/from/root;
        index index.html;
        rewrite ^/$ /index.html break;
}
```

Désigne le dossier racine du site

Désigne l'index du site

- Voir la documentation pour plus de fonctionnalités (pages d'erreurs, etc…)



Sécuriser les connexions

Via HTTPS

CERBOT

- Cerbot s'occupe de tout ! => https://certbot.eff.org/#ubuntuxenial-nginx
  - Il y a des scripts à installer sur son serveur, certbot s'occupe de renouveler les certificats de 90 jours sous Let's Encrypt.

- C'est **gratuit**, **facile**, et **indispensable** aujourd'hui ! Pas de site web sans https !