PPE2 - API NodeJS

Documentation

Sommaire

Présentation de l'API

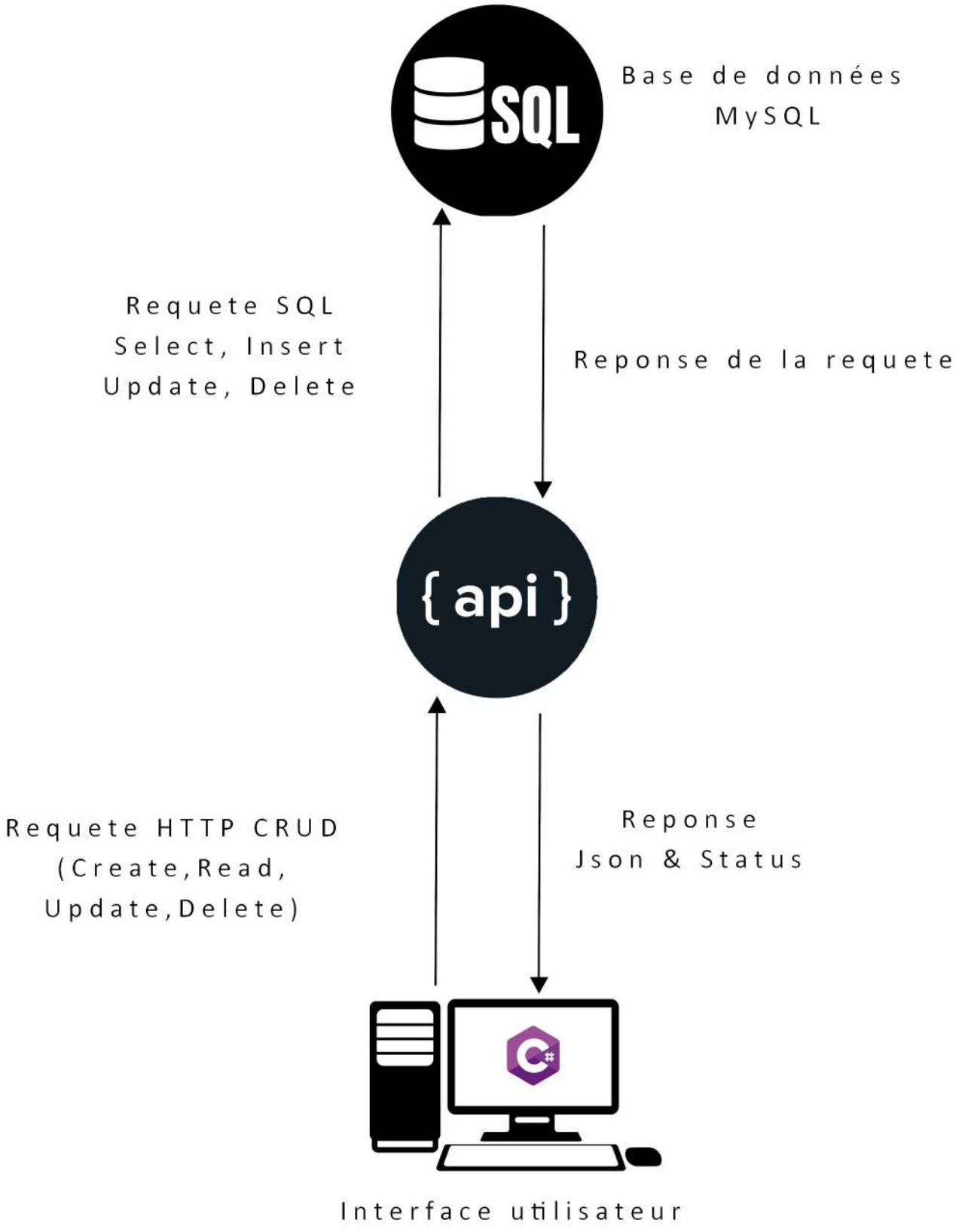
C'est une API REST développée en NodejS qui est une application qui permet de communiquer entre deux logiciel sans devoir utiliser la même architecture ou même systeme d'exploitation.

Cette API est reliée a une base de données MySql avec la quelle elle communique avec des requetes SQL et aussi a une application C# en recevant des requetes HTTP de cette derniere.

Le choix de l'API apporte plusieurs avantage comme une facilité d'utilisation puisqu'une fois les fonctions creer dans l'API, il est possible de les utiliser peu importe le language adopté qui est scalable sur tout les appareils, surtout a l'heure ou les objets connectés se generalise.

Mais aussi d'un point de vue securité car les identifiants de la base de données ne sont plus stocker en dur dans l'application mais plutot dans les fichiers de l'API sur le s Serveur.

Fonctionnement de L'API



Modules

SEQUELIZE :

Sequelize a été utilisé dans ce projet pour faciliter les échanges entre l'API et la base de données MySQL.
Ce module permet de gerer plus facilement les requetes ainsi que les creations de table.

Lien : https://sequelize.org

EXPRESS :

Express est une infrastructure d'application flexible qui permet la création d'une API robuste de maniere simple et rapide avec ses methodes utilitaires HTTP intégré et la gestion du serveur.
Il offre de nombre possibilité grace à ses plugins.

Lien : https://expressjs.com/fr/

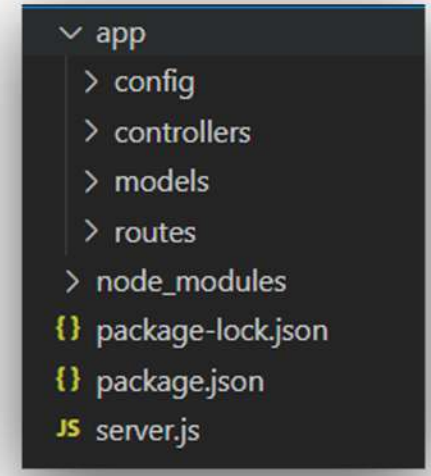
NODEMON :

Nodemon est un petit module qui permet de faciliter la modification du code et de gagner du temps puisqu'il redemarre l'application a chaque fois que le fichier est enregistré.

Lien : https://www.npmjs.com/package/nodemon

Architecture de l'API

Le projet a été classé par dossier pour plus de lisibilité, chaque dossier contient des fichiers particulier.



App :
Contient les dossiers qui constituent la logique de l'application.

Config :
Contient les identifiants de connexion a la base de données.

Controllers :
Definition des fonctions des differents objets.

Models :
Definition des modèles de table a creer dans la bdd.

Routes :
Contient les differentes routes accessible de l'API par requete.

Node_modules :
Dossier des dépendances de l'application.

Le fichier db.config

Le fichier db.config stock les identifiants de connexion a la base de données que va utiliser Sequelize pour envoyer des requetes SQL lors des appels de l'API. Ce qui permet en cas de besoin de modification, d'avoir un seul fichier a modifier.

```
// Information de connexion à la base de données
module.exports = {

    HOST: "localhost",

    USER: "root",

    PASSWORD: "mysql",

    DB: "library",

    dialect: "mysql",

    pool: {

    max: 5,

    min: 0,

    acquire: 30000,

    idle: 10000

    }

    };
```

db.config.js

Fonction de l'API

Le dossier controllers contient tous les fichiers qui regroupe toutes les fonctions de l'API.
Les f onctions sont utilisés en cas de besoin si une route de l'API reçoit une requête HTTP.

Sequelize permet de structurer les fonctions de manière simplifié.

Chaque définition de fonction commence par son nom, puis va stocker dans une variable la donnée transmise par la requête HTTP.

Une condition va être vérifiée puis une fonction de Sequelize va lancer la requête SQL sur la base de donnée.
L'API renvoie le résultat sous forme de Json au client avec un statut valide ou bien un message d'erreur avec un statut correspondant en cas de problème.

```
// Recuperer tout les livres de la bdd
exports.findAll = (req, res) => {
    const titre = req.query.titre;
    var condition = titre ? { titre: { [Op.like]: `%${titre}%` } } : null;
    Livre.findAll({ where: condition })
      .then(data => {
        res.send(data);
      })
      .catch(err => {
        res.status(500).send({
          message:
            err.message || "Un probleme est survenu lors de la requete pour recuperer tout les livres."
        });
      });
  };
```

Les modèles de table

Le dossier models contient tout les models de table de l'API ainsi que le fichier index qui paramètre Sequelize pour la création de ses dernieres.

Un modèle sert à creer automatiquement une table dans la base de donnée de maniere fiable.

Lors du lancement de l'API, si les tables ne sont pas déjà presente dans la base, elles seront creer une à une avec un "CREATE TABLE IF NOT EXIST" de requete SQL.

```
module.exports = (sequelize, Sequelize) => {
    const Livre = sequelize.define("livre", {
      titre: {
        type: Sequelize.STRING
      },
      auteur: {
        type: Sequelize.STRING
      },
      annee: {
        type: Sequelize.INTEGER
      },
      description: {
        type: Sequelize.STRING
      },
      categorie: {
        type: Sequelize.STRING
      },
      user: {
        type: Sequelize.STRING
      },
      disponible: {
        type: Sequelize.INTEGER
      }
    });
    return Livre;
  };
```

Modèle table "Livre"

Les routes de l'API

Le dossier routes contient les fichiers de routage de l'API, ce sont les chemins d'acces de l'API pour recevoir les requête HTTP.

Chaque route represente l'URL necessaire pour acceder a une fonction ainsi que pour recuperer une donnée ainsi que le type.

La route general de l'API est :
{nomDuServeur}/api/{categorie}.

Exemple pour les routes de "Livre"

```
app.use('/api/livres', router);
```

route de base fonctionel de livre

```
// Recuperer tout les livres.
router.get("/", livres.findAll);
```

Recuperation des livres a la racine de la route avec un Get:
Get.{nomDuServeur}/api/livres

```
// Recuperer les livres loué par un user
router.get("/location/:user", livres.findByUser);
```

Recuperer les livres d'un user avec un ainsi qu'une variable transmise dans la requête HTTP

Get.{nomDuServeur}/api/livres/nomUtilisateur

Requête HTTP & Test - 1

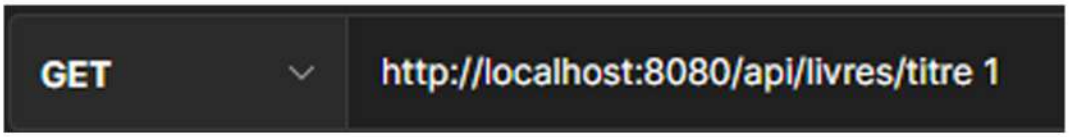
Nous allons réaliser des tests de l'API pour montrer son fonctionnement de manière concrète.

Pour cela je vais utiliser POSTMAN qui est un logiciel qui permet de simuler des requêtes HTTP sur une API en pouvant choisir le type (Post / Get / Put / Delete) mais aussi la posibilité de fournir un contenu et un retour du resultat.

Nous allons tenter de recuperer tout les livres de la base en utilisant la route correspondante :

```
// Recuperer un seul livre avec son titre
router.get("/:id", livres.findOne);
```

route en get qui appel la fonction findOne de livres



Requete émise dans Postman

La route de base est présente ainsi que le paramètre ":id" qui est dans l'exemple 'titre 1'

Requête HTTP & Test - 2

l'API reçoit la requête et execute findOne("titre1")

```
// Rechercher un titre de livre dans la bdd
exports.findOne = (req, res) => {
    const titre = req.params.id;
    Livre.findOne({ where: { titre: titre } })
      .then(data => {
        if (data) {
          res.send(data);
        } else {
          res.status(404).send({
            message: `Impossible de trouver un livre avec le titre =${titre}.`
          });
        }
      })
      .catch(err => {
        res.status(500).send({
          message: "Un probleme est survenu lors de la requete pour le titre=" + titre
        });
      });
  };
```

Requête SQL a la base de données

Retour de resultat sous forme de Json

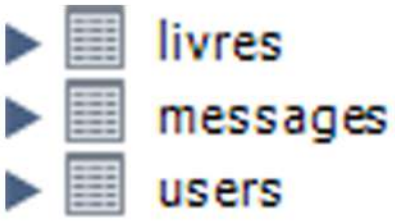

```
{
    "id": 1,
    "titre": "titre 1",
    "auteur": "jeff",
    "annee": 2019,
    "description": "Livre de la jungle",
    "categorie": "fantasy",
    "user": "",
    "disponible": 1,
    "createdAt": "2022-04-19T12:11:22.000Z",
    "updatedAt": "2022-04-24T22:45:16.000Z"
}
```

Base de données MySQL

La base de données de l'API est l'endroit ou sont stockés les données de l'application qui permet de faire fonctionner le tout.

Sequelize permet de verifier a chaque demarage du serveur si les tables du dossier models sont bien existante avec une requête SQL : "CREATE TABLE IF NOT EXIST".

L'essence même de l'API est de pouvoir communiquer avec la BDD de maniere simple et securisé a partir de n'importe quel client.



Les tables de la base de données

id	titre	auteur	annee	description	categorie	user	disponib
1	titre 1	jeff	2019	Une description "scientifique" des c...	fantasy		1
2	titre 2	victor	1998	Un soir d'hiver, a l'heure ou ..	fantasy		1
3	titre 3	phillipe	2001	Il est obligé de ramasser l'arme	fantasy	responsable@r...	0
4	titre 4	jean	2017	Demain, dès l'aube, à l'heure ou bla...	horreur	adminMail	0
5	titre 5	hugo	1980	Le chien etait heureux de revoir so...	horreur		1
6	titre 6	bertrand	1967	L'appartement de Thomas Clerc fait...	horreur		1
7	titre 7	robert	1990	Un coeur qui bat, les cellules de la ...	fiction		1
8	titre 8	nicolas	2021	Il est un fort, sur la lande, où l'on e...	fiction		1
9	titre 9	elise	2011	A l'époque oú commence l'histoire d...	fiction		1
10	titre 10	mehdi	2013	Parmi tous les animaux domestiqué...	romantique	NULL	1

Exemple de la table livre