Documentation technique
Pénibilité Ambiante

PTUT 3e année

Marie Cagnasso, Damien Evrard et Julie Hoarau

ISIS INGÉNIEURS CASTRES

UC

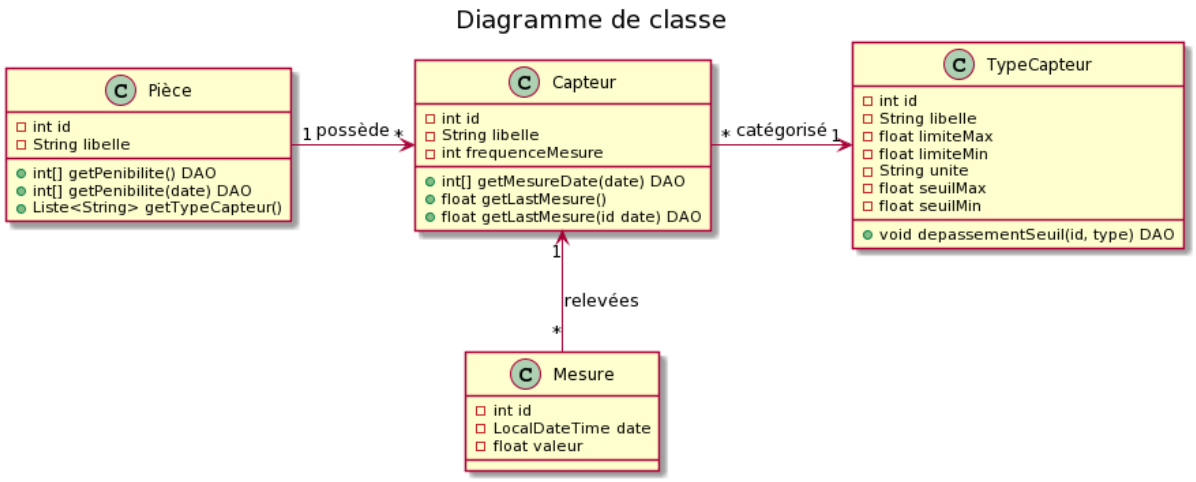
Présentation de l'application

Cette application est une interface d'administration, elle permet de gérer plusieurs pièces, capteurs et leurs types.

Son objectif est de surveiller la pénibilité ambiante subie sur le lieu de travail (température, humidité, bruit...) dans différente pièce via des capteurs de tout type. Ces capteurs communiquent leurs relevés de mesure à l'application à travers des requêtes HTTP à destination de notre APICapteur. L'application les stocke en base de données avant de les afficher sur l'interface.

Les relevés sont affichés dans des graphiques pour suivre l'évolution des mesures au cours du temps, cet intervalle est réglable par l'utilisateur.

Modèle de données



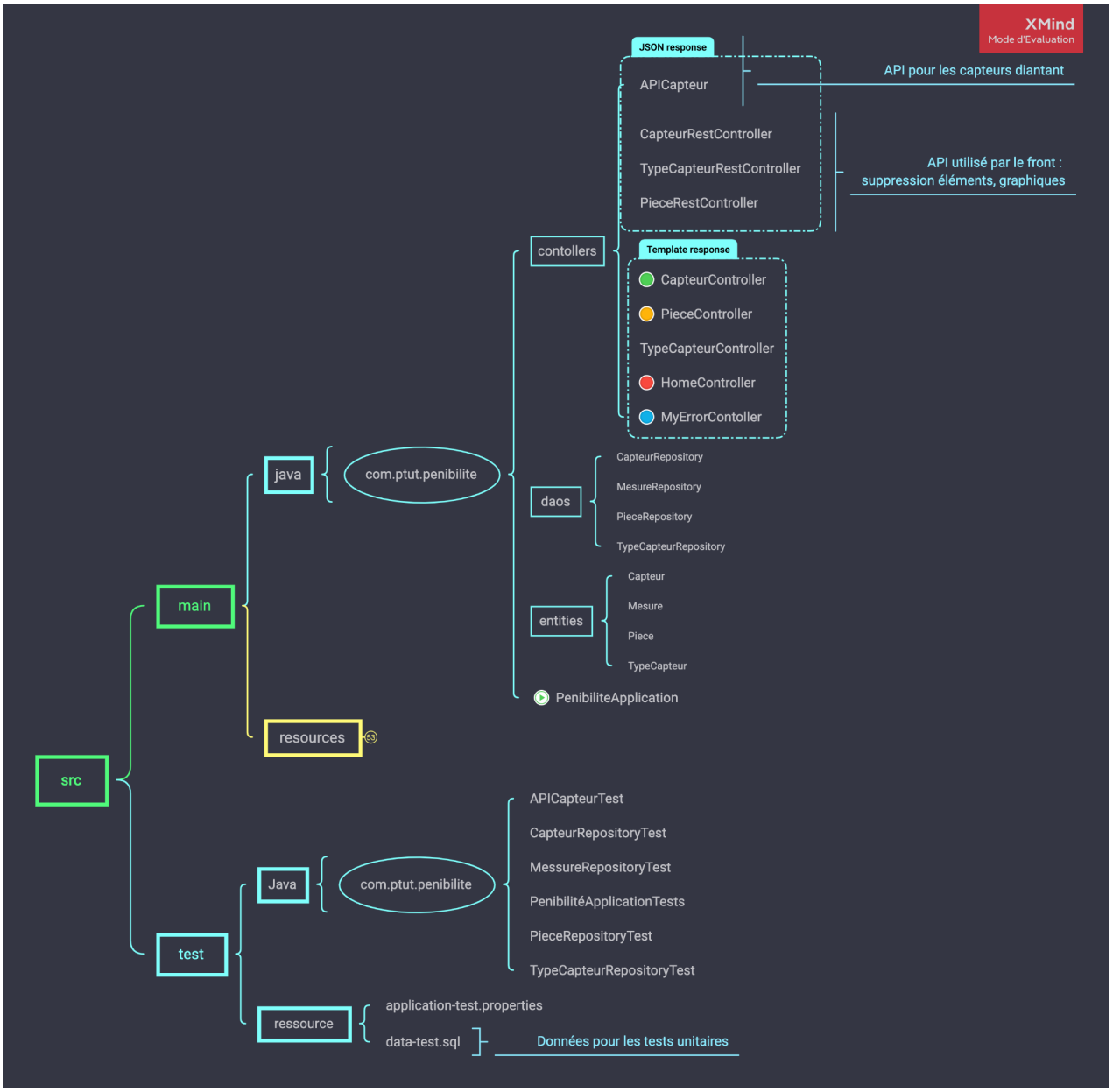
Dans notre diagramme de classe :

- ➔ Pour chaque entité, il nous faut déterminer un identifiant (integer id),
- ➔ Les entités ***Pièce***, ***Capteur*** et ***TypeCapteur*** ont chacune un libellé défini (String),
- ➔ Pour le calibrage, la fréquence de mesure (integer) peut être déterminée pour chacun des capteurs,
- ➔ Chaque capteur est obligatoirement relié à un type de capteur car c'est l'entité ***TypeCapteur*** qui donne les informations sur les limites et les seuils de mesures (floatant) et l'unité (String) de chaque capteur et les mesures qui lui sont associées,
- ➔ À un type de capteur peuvent être associés plusieurs capteurs,
- ➔ Un capteur est obligatoirement relié à une pièce, mais une pièce peut disposer de plusieurs capteurs,
- ➔ Les opérations définies dans l'entité ***Capteur*** permettent de récupérer les données de l'entité ***Mesure*** (soit les dates des mesures, la valeur de la dernière mesure effectuée, ainsi que les valeurs des mesures sur une période donnée),
- ➔ Dans l'entité ***Mesure***, il y a la date de la mesure ainsi que sa valeur.
- ➔ Une mesure est ainsi relevée par un unique capteur. Mais un capteur peut effectuer une infinité de mesures,
- ➔ Le dépassement de seuil est déterminé à partir de l'id de la pièce et du type du capteur,
- ➔ Pour une pièce donnée, il est possible de déterminer sa pénibilité ambiante (*getPenibilité* avec pour paramètre la date sélectionnée), ainsi que la liste des capteurs qui s'y trouvent (*getTypeCapteur*).

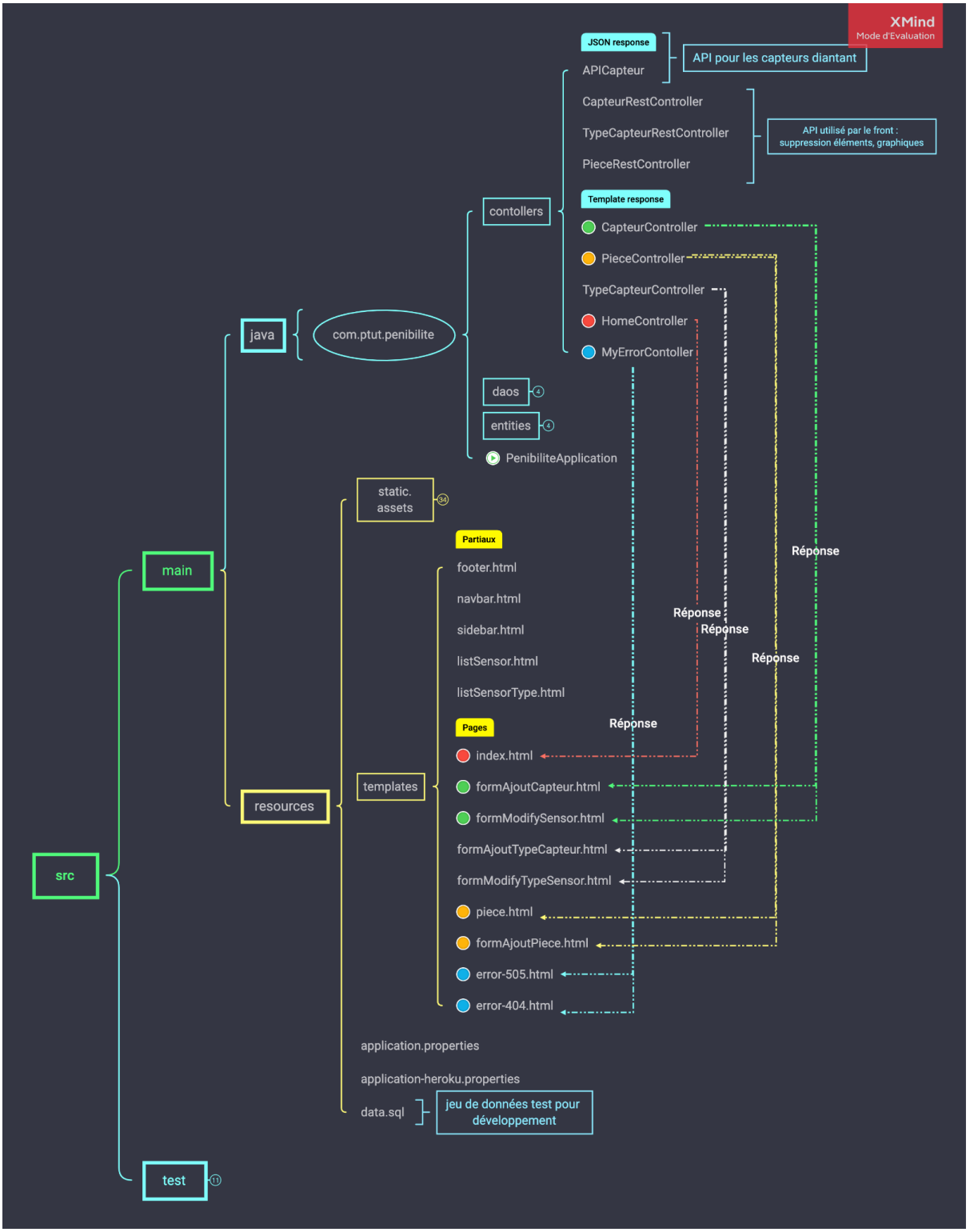
Structure de l'application

Architecture

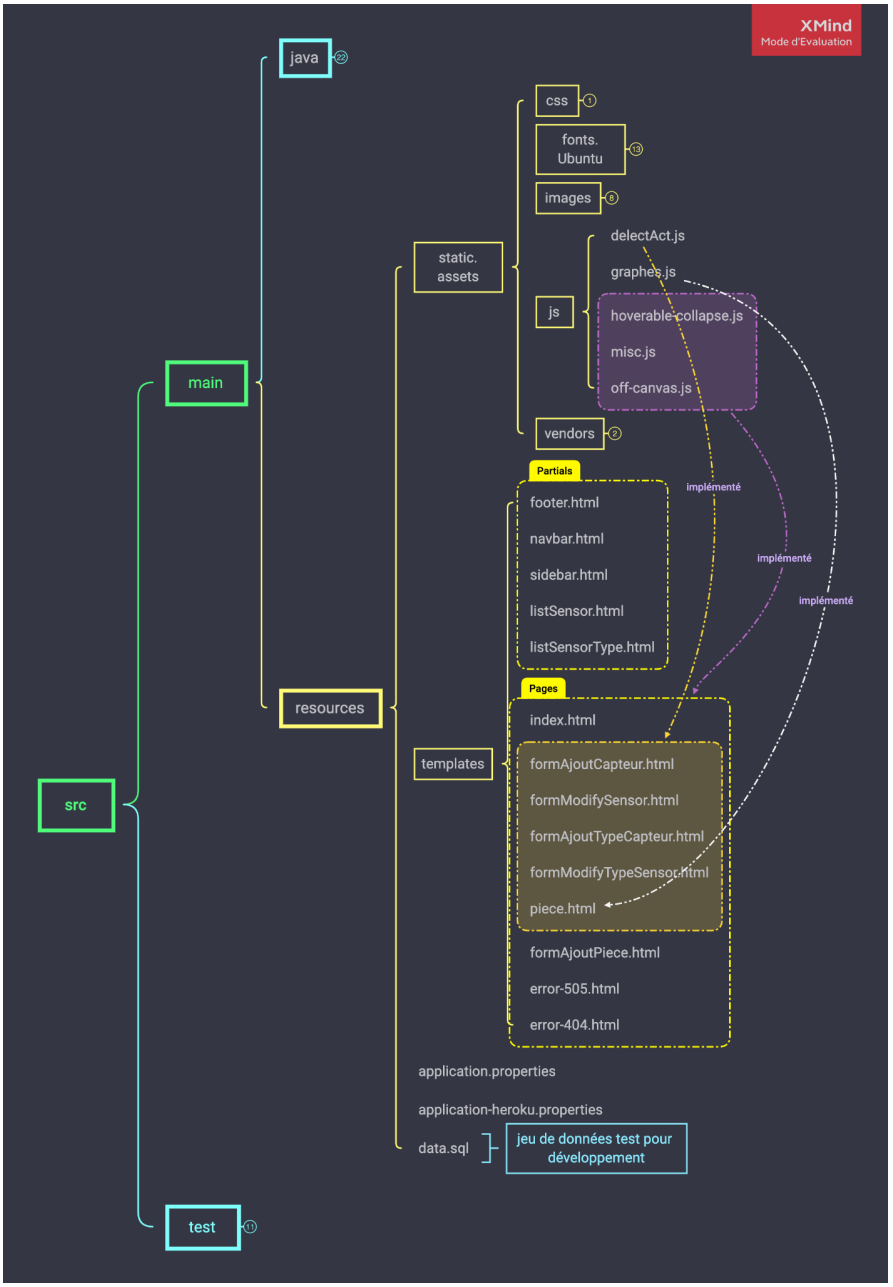
Back-end & Tests



Contrôle des vues

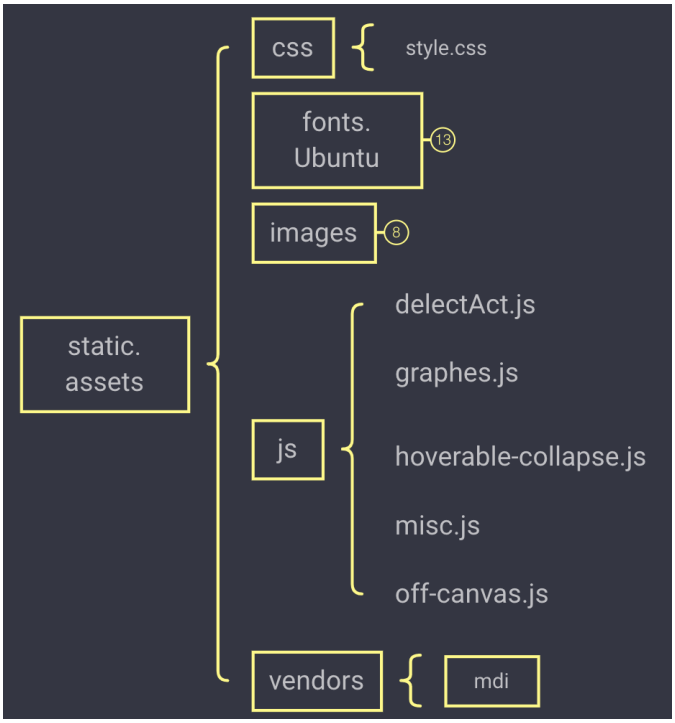


Front-end - JavaScript



- **deleteAct.js** : Fait des requêtes AJAX vers les contrôleurs REST pour supprimer les différents éléments et déclenche une alerte avant la suppression,
- **graphes.js** : Récupération des données et contrôle de leur l'affichage dans les graphiques,
- **hoverablecollapse.js** : animation du menu desktop plié,
- **off-canvas.js** : animation du menu mobile,
- **misc.js** : autres animations (fullscreen, apparition d'un form ...).

Front-end - CSS & vendors



Le répertoire "*vendors*" contient toutes les bibliothèques externes (elles ne doivent pas être modifiées), elles peuvent être remplacées par un CDN s'il en existe un.

➔ "*mdi*" : Material Design Icons

"style.css" : Ce fichier provient d'un template, il comprend notamment le CSS de Bootstrap. Il est possible de supprimer cette partie pour la remplacer par un CDN.

===== Table of Contents =====

* Bootstrap functions
* Template variables
* SCSS Compass Functions
* Bootstrap Main SCSS
* Template mixins
* Core Styles
* Components
* Landing screens
    + Auth

**Disponible plus en détail dans le fichier*

Utilisation de APICapteur

Cette API REST a pour but de récupérer les informations envoyées par les capteurs. Deux requêtes avec la méthode GET sont disponibles :

ajoutMesure (id, valeur)

Description :

Cette méthode ajoute dans la base de donnée la mesure relevée par un capteur dans une salle.

URL : /APIcapteur/ajout
Méthode : GET
Paramètre :
id = <id du capteur>
valeur = <mesure relevée par le capteur>

Réponse : Objet JSON

{succès exemple :

```
{
    "message" : "ok"
}
```

erreur exemple :

```
{
    "message" : "valeur erronée"
}
```

getFrequenceMesure (id)

Description :

Cette méthode communique au capteur la fréquence à laquelle il doit faire ses relevés.

URL : /APIcapteur/frequence

Méthode : GET

Paramètre :

id = <id du capteur>

Réponse : Objet JSON

succès exemple :

```
{
    "frequence" : 420
}
```

erreur exemple :

```
{
    "frequence" : null
}
```

Utilisation des contrôleurs REST

Ces contrôleurs renvoient des réponses au format JSON. Ils sont utilisés dans cette application pour communiquer avec le front (JS).

Supprimer un capteur

Description :
Cette requête supprime un capteur dans la base de données.

Contrôleur : CapteurRestController
URL : /api/capteur/delete
Méthode : DELETE
Paramètre :
id = <id du capteur>

Réponse : Objet JSON

succès exemple :

```
{
    "status" : "0"
}
```

erreur exemple :

```
{
    "status" : "1"
    "error" :  [erreur]
}
```

Supprimer un type de capteur

Description :
Cette méthode supprime un type de capteur.

Contrôleur : TypeCapteurRestController
URL : /api/type/delete
Méthode : DELETE
Paramètre :
id = <id du typeCapteur>

Réponse : Objet JSON

succès exemple :

```
{
    "status" : 0,
}
```

erreur exemple :

```
{
    "status" : 1,
    "error" : “exception”
}
```

Calcul des dépassements de seuil

Description :

Cette méthode calcule le nombre de valeurs qui dépassent les seuils sur la période sélectionnée.

Contrôleur : TypeCapteurRestController
URL : /api/type/depassement
Méthode : GET
Paramètre :

id = <id de la pièce>
dateMax = <date de fin de la période>
dateMin= <date de début de la période>

Réponse : Objet JSON

succès exemple :


```
{
    0 : {
        "libelle" : "Thermomètre",
        "depassement" : 5
    },
    1 : {
        "libelle" : "Vibromètre",
        "depassement" : 0
    }
}
```

Obtention des mesures d'une pièce

Description :

Cette requête renvoie les relevés faits par les capteurs d'une pièce sur une période donnée.

Contrôleur : PieceRestController
URL : /api/piece/getDonnees
Méthode : GET
Paramètre :

id = <id de la pièce>
dateMax = <date de fin de la période>
dateMin= <date de début de la période>

Réponse : Objet JSON

succès exemple :


```
{
    0 : {
        "libelle" : "Thermomètre",
        "type" : "Température,
        "dates" : ["27-03-2021T11:30:00", "28-03-2021T16:25:00"],
        "valeurs" : [25, 26],
        "unite" : "°C",
        "seuilMax" : 30,
        "seuilMin" : -5,
    },
    1 : {
        "libelle" : "Vibromètre",
        "type" : "Vibration,
        "dates" : ["27-03-2021T11:30:00" ,"28-03-2021T16:25:00"],
        "valeurs" : [1.05, 1.5],
        "unite" : "°C",
        "seuilMax" : 30,
        "seuilMin" : -5,
    }
}
```

Supprimer une pièce

Description :
Cette requête permet de supprimer une pièce.

Contrôleur : PieceRestController
URL : /api/piece/delete
Méthode : DELETE
Paramètre :
id = <id de la pièce>

Réponse : Objet JSON

succès exemple :

```
{
    "status" : 0,
}
```

erreur exemple :

```
{
    "status" : 1,
    "error" : "exception"
}
```

Obtention des types de capteur dans une pièce

Description :
Cette requête renvoie la liste des types de capteur présents dans une pièce sans doublon.

URL : /api/piece/getTypeCapteur
Méthode : GET
Paramètre :
id = <id de la pièce>

Réponse : Objet JSON

succès exemple :

```
{
    "types" : ["Température", "Vibration"],
}
```

Choix technologiques

Back-end

Langage : Java
Framework : Spring Boot

Object-Relational Mapping (ORM) : Java Persistence API (JPA)

Base de données de développement : h2
Base de données de production : postgres

Test unitaire : Junit & @dataJPATest

Contrôleurs

Moteur de template : Thymeleaf
Contrôleur REST : API REST avec Spring

Front-end

Bootstrap 4 : Site responsive
jQuery
Chart.js : Création de graphiques
Material Design Icons : Bibliothèque d'icônes
FreePurpleAdmin : Site inspiré de ce template

TODO

Fonctionnalités :

- Graphique : Ajouter les barres de seuil d'exposition qu'il ne faudrait pas dépasser. (début d'implémentation dans graphes.js)
- Liste administrable des mesures prises par les différents capteurs, possibilité d'en supprimer.
  Au vu des choix de conception de la base de données, et n'ayant pas la possibilité de supprimer des mesures depuis l'interface, si une pièce comporte un capteur ayant déjà réalisé des relevés de mesures, ni le capteur, ni la pièce ne pourront être supprimés.
- Implémenter une barre de recherche.

UX :

- Afficher un message d'état quand il n'y a pas de donnée affichée dans les graphes.
- Design des messages d'état sous forme de notification.
- Initialisation des inputs type number dans les formulaires de modification avec les données de l'objet. (réalisable en JavaScript).