# CORBEILLE D’EXERCICES

**NODEJS / Typescript**

# APPLICATION BACKEND + ROUTE

|  |  |
| --- | --- |
| **Objectif :**  Mettre en place une application backend, réaliser des appels vers les routes et récupérer les résultats.  Nous chercherons à interagir avec des capteurs.  Voici les informations de ceux-ci sous forme de tableau d’objets.   |  | | --- | | const sensors = [  {id:1, type:'alpha', datas:[1,2,3]},  {id:2, type:'beta', datas:[1,2,3], metrics:false},  {id:3, type:'omega', datas:{a:1,b:2}},  ] | |
| 1ére ÉTAPE [**Générer une application Express]**  Créer en CLI une application Express.  Ne pas lui appliquer de vue, le but est uniquement d’envoyer des réponses sous format JSON. |
| 2éme ÉTAPE [Créer sa première route] Créer votre première route nommée « sensors » qui aura pour réponse un status 200 et un JSON sous la forme {"response": "true"}  Vérifier que les informations envoyées depuis le serveur sont correctes, à l’aide d’un logiciel comme Postman |
| 3éme ÉTAPE [Récupérer les données d’un capteur spécifique] Créer une nouvelle route (selon les normes RESTFULL) pour récupérer les données d’un capteur spécifique. Nous souhaitons que la réponse nous envoie les données du capteur ainsi que l’état de la réponse. |
| 4éme ÉTAPE **[Personnalisation des erreurs]**  Vous devez modifier la fonction de récupération des données d’un capteur pour vous renvoyer un message personnalisé et un statut 404 si le capteur n’est pas trouvé. |
| 5éme ÉTAPE [**Ajouter les données d’un capteur]**  Ajouter une route, pour ajouter un produit (virtuellement pour le moment), l’ID est obligatoire et les datas sont optionnelles. Bien gérer si le capteur existe déjà. Code statut 201, pour la création de ressource. |
| 6éme ÉTAPE **[Supprimer un capteur]**  Mise en place d’une route de suppression avec ID obligatoire. |
| 7émé ÉTAPE **[Modifier les données d’un capteur]**  Mise en place d’une route pour modifier les données d’un capteur. Uniquement les données envoyées doivent être modifiées. |

# MOnGOOSE

|  |
| --- |
| **Objectifs :**  Se connecter à une base de données mongoDB, en lui appliquant un schéma au niveau applicatif. Nous allons utiliser la bibliothèque Mongoose pour cela. |
| 1ére ÉTAPE [**Créer la connexion**]  Vous devez créer un objet de connexion à la base de données. |
| 2ème ÉTAPE [**Appliquer un schéma**]  Vous devez mettre en place un schéma pour les capteurs, selon la même structure qui est fournis dans le premier exercice.  Pour cela, vous devez mettre en place un model Mongoose. |
| 3éme ÉTAPE [**manipulation des données**]   1. ADD SENSOR :   En modifiant la route précédemment créée, vous devez créer un capteur en base de données.   1. GET ALL SENSORS   Vous pouvez aussi récupérer toutes informations de tous les capteurs   1. GET SPECIFIC SENSOR   Vous pouvez récupérer les informations d’un capteur spécifique à partir de son \_id |

# TYPESCRIPT

|  |
| --- |
| **Objectifs :**  Utiliser TypeScript pour structurer son code |
| ÉTAPE 0 **[Installer TypeScript]**  Intégrer TypeScript dans votre projet à l’aide des commandes npm. |
| 1ére ÉTAPE **[Structurer son code]**  Mettre en place un « controller » qui va gérer la logique métier au niveau de la base de données. |
| 2éme ÉTAPE [**Le typage]**  Vous devez mettre en place un typage pour vos fonctions et paramètres. |
| 3Éme ÉTAPE [**Les interfaces**]  Mettre en place un schéma pour Mongoose en y intégrant une interface |