# Vue d'Ensemble des Dépendances et Technologies du Projet Full-Stack avec Vue.js + Node.js + Express + PostgreSQL

## ***Dépendances Frontend***

### @kyvg/vue3-notification :

Utilisation : Fournit un système de notifications simple à utiliser dans Vue 3, ce qui permet d'informer l'utilisateur de manière interactive (ex : succès d'action, erreur).

### Axios :

Utilisation : Client HTTP pour envoyer des requêtes (GET, POST, PUT, DELETE) vers ton backend, récupérant ou envoyant des données (souvent utilisé pour la communication avec les API).

### bcryptjs :

Utilisation : Permet de hacher les mots de passe de manière sécurisée. Cela empêche de stocker les mots de passe en clair dans la base de données.

### body-parser :

Utilisation : Middleware pour parser les données JSON et URL encodées dans les requêtes HTTP, utilisé dans le backend avec Express.

### bootstrap :

Utilisation : Framework CSS populaire pour concevoir des interfaces réactives et modernes rapidement. Il fournit des composants prédéfinis comme des boutons, des cartes, etc.

## connect-pg-simple :

Utilisation : Permet de stocker les sessions utilisateurs dans une base de données PostgreSQL, garantissant ainsi que les sessions sont persistantes.

## cors :

Utilisation : Permet de configurer les en-têtes CORS pour autoriser les requêtes entre différents domaines, par exemple entre le frontend Vue.js et le backend Express.js.

## dotenv :

Utilisation : Charge les variables d'environnement à partir d'un fichier .env, permettant de configurer des informations sensibles (ex : base de données, clés API) sans les exposer dans le code source.

## express :

Utilisation : Framework backend populaire en Node.js, permettant de gérer les requêtes HTTP, les routes, la gestion des middlewares, etc.

## express-session :

Utilisation : Middleware pour gérer les sessions des utilisateurs, généralement utilisé en combinaison avec un store (comme PostgreSQL).

## express-validator :

Utilisation : Outil de validation des données dans les requêtes HTTP, permettant de vérifier que les données envoyées sont correctes avant d'être traitées.

## fs :

Utilisation : Module de Node.js pour interagir avec le système de fichiers (lecture, écriture de fichiers).

## jsonwebtoken :

Utilisation : Permet de créer et de vérifier des tokens JWT (JSON Web Tokens) utilisés pour l'authentification.

## nodemailer :

Utilisation : Utilisé pour envoyer des emails directement depuis l'application.

## pg :

Utilisation : Client PostgreSQL pour communiquer avec une base de données PostgreSQL.

## pg-hstore :

Utilisation : Permet de sérialiser et désérialiser les données JSON pour PostgreSQL.

## sequelize :

Utilisation : ORM (Object-Relational Mapping) pour interagir avec la base de données SQL, en transformant les requêtes SQL en objets JavaScript.

## vue (v3) :

Utilisation : Framework frontend pour construire des interfaces utilisateurs réactives et dynamiques.

## vue-router :

Utilisation : Gestion du routage dans Vue.js, permettant de définir et de gérer les différentes pages de ton application.

## vue-toastification :

Utilisation : Système de notifications toast pour afficher des messages courts à l'utilisateur (par exemple : confirmation de soumission de formulaire).

# Dépendances Backend

- axios :
- Même utilisation que pour le frontend : Permet d'envoyer des requêtes HTTP depuis le backend vers d'autres services API si nécessaire.
- bcrypt et bcryptjs :
- Hachage des mots de passe : Ces deux packages permettent de sécuriser les mots de passe des utilisateurs (il est possible d'utiliser l'un ou l'autre).
- body-parser :
- Même utilisation que pour le frontend : Permet de traiter les données envoyées dans les requêtes HTTP, comme les données JSON.
- cors :
- Même utilisation que pour le frontend : Permet de gérer les requêtes cross-origin et autoriser la communication entre le frontend et le backend.
- dotenv :
- Même utilisation que pour le frontend : Chargement des variables d'environnement pour ne pas exposer les informations sensibles dans le code source.
- express :
- Même utilisation que pour le frontend : Framework pour créer des serveurs HTTP et gérer les requêtes API, gérer la logique côté serveur.
- express-session :
- Gestion des sessions côté serveur : Comme le frontend, ce package permet de créer et gérer des sessions utilisateurs dans le backend.
- fs :
- Interaction avec le système de fichiers : Lecture et écriture de fichiers dans le backend.
- connect-pg-simple :
- Gestion des sessions PostgreSQL : Permet de stocker les sessions dans PostgreSQL pour une gestion sécurisée et persistante.
- jsonwebtoken :
- Authentification via JWT : Création et validation de tokens pour maintenir l'authentification des utilisateurs.
-

**child_process :**

Utilisation : Permet d'exécuter des processus système ou des commandes shell depuis le backend Node.js.

## multer :

Gestion des uploads de fichiers : Permet de gérer le téléchargement de fichiers sur le serveur (par exemple, pour les profils utilisateur ou les documents).

## pg et pg-hstore :

Interaction avec PostgreSQL : Utilisé pour communiquer avec une base de données PostgreSQL, récupérer et manipuler les données.

## sequelize :

Interaction avec la base de données : ORM pour faciliter les opérations SQL sur PostgreSQL sans avoir à écrire de SQL brut.

# Autres Technologies et Configurations

## Bundler : Vite

Vite : Outil de build et de développement pour les applications modernes, offrant un démarrage rapide et des fonctionnalités comme le hot-reloading.

---

- Toutes ces dépendances forment un projet full-stack moderne avec:
- Frontend en Vue.js 3
- Backend en Express.js
- Base de données PostgreSQL
- Système d'authentification complet
- Gestion de fichiers
- Styles avec Bootstrap et Tailwind CSS
- Système de routage
- Gestion des sessions et tokens JWT

## Configurations Environnement fichier (.env)

- Variables pour PostgreSQL:
  - DB_USER
  - DB_HOST
  - DB_NAME
  - DB_PASSWORD
  - DB_PORT
  - DB_DIALECT
- Sécurité:
  - SESSION_SECRET
  - JWT_SECRET

- Variables pour PostgreSQL : Contiennent des informations de connexion à la base de données comme l'utilisateur, l'hôte, le mot de passe, etc.
- Sécurité : Contient des clés secrètes pour la gestion des sessions (SESSION_SECRET) et des tokens JWT (JWT_SECRET).

---