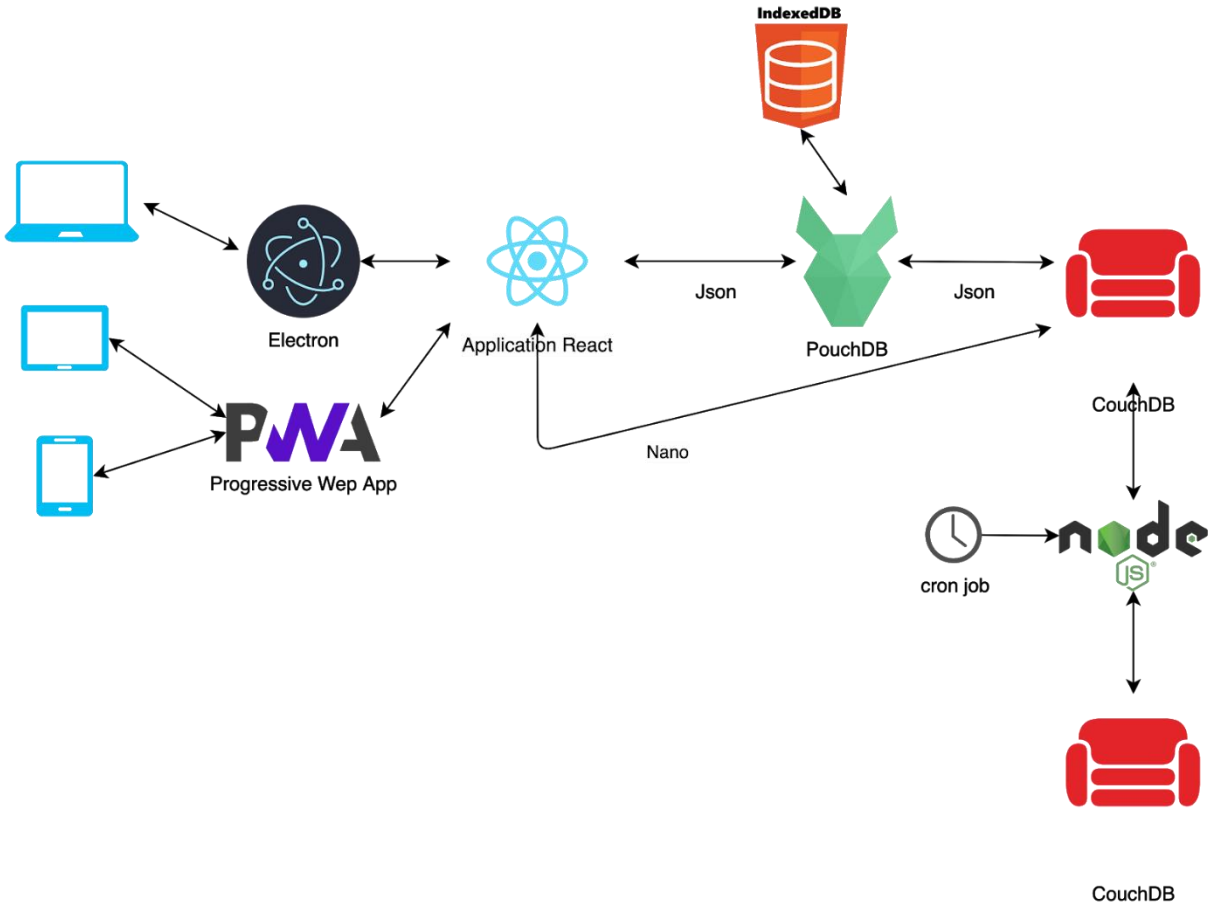
Store Manager

Architecture



Côté serveur, les données sont stockées au format Json dans une base CouchDB (NoSQL).

Côté client, l'application React utilise PouchDB pour synchroniser une base locale (utilisant IndexedDB) avec une partie de la base CouchDB. Le filtrage est fait uniquement sur les magasins du commercial grâce à une option de CouchDB.

Tous les stockages et échanges se font au format Json.

L'application est packagée avec Electron ou PWA pour pouvoir s'exécuter en mode hors-ligne.

Lors de la première connexion, l'application requête directement la base CouchDB grâce au connecteur Nano afin d'authentifier l'utilisateur et récupérer la liste de ses magasins.

PouchDB gère ensuite la synchronisation entre les différentes bases locales et la base CouchDB.

Lors de la saisie des informations par l'utilisateur, les données du formulaire sont enregistrées dans le local storage du navigateur afin de pouvoir reprendre la saisie ultérieurement. Ces données sont effacées à l'envoi du formulaire.

A l’enregistrement des données, un timestamp est ajouté au document.

La synchronisation est déclenchée automatiquement à chaque connexion au réseau, puis tant que la connexion est ouverte, la synchronisation s’effectue en mode “live”.

En cas de déconnexion lors de la synchronisation, celle-ci est retentée pendant maximum 10 minutes.

La gestion des conflits au niveau de PouchDB permet de garder uniquement la modification la plus récente (par rapport à l’heure de saisie et non celle de la synchronisation).

Dans PouchDB, c’est le document écrit en dernier qui l’emporte par défaut. En cas de conflit, on récupère donc toutes les versions en conflit du document, et on les supprime toutes afin de ne garder que celle dont la date d’écriture est la plus récente.

Une sauvegarde quotidienne de la base CouchDB permet d’éviter une perte majeure de données. Elle est effectuée grâce à un script Node JS, et automatisée avec un cron job.