WEALTH HEALTH

PRÉSENTATION

Le contrat financiers Wealth Health utilise une application web interne pour gérer les souches de contrats.

L'application est archivée et utilise jQuery (v3x) Vue3 and, ce qui entraîne des bugs considérables et des augmentation des plaintes de clients.

PROJET

Notre équipe de développeurs a pour mission de :

1.Convertir l'existante à un projet Vite/ Vue et React.

2.Remplacer les fonctionnalités existantes par des projets React. Remplacer les 3 pages jQuery restante par des composants Vue.

3.Préconiser des tests de performance Lighthouse et comparer l'ancienne et la nouvelle application.

STRUCTURE

Le fichier **index.**h tourné les différents scènes utilisé via l'importation de la librairie Fontawesome.

- Le fichier **vite.**h
- Les **tests** de l'application
- Les **variables** et **mixins** pour gérer les couleurs et la gestion des différentes résolutions (écran)

STORE

React-Hook-useContext: sera utilisé pour le choix du manager. L'application restant jqu-portée, un état manager lié au contrat reste plus facilitant.

Création d'un **context** et placement de celui-ci dans un **Custom-Hook-Context**. Le **Provider** sera placé au plus haut niveau de l'application.

Les différentes actions qui seront utilisées et leurs qui impliquent.

Contrat

Utilisation

Provider

Data et Fonctions

Custom Hook

En exportant le **EmployeeProvider** ainsi que le **Custom-Hook** **useEmployeeStore**, nous pouvons importer dans les différentes pages du client pour les **composants** qui les requièrent.

CONVERSION Plugin-jQuery

Le plugin **Dropzone-Menu** sera le plugin choisi pour être converti en **Dropzone** **Vue**.

Nous utilisons create-component-lib afin de refactoriser.

Il contient 6 arguments empaquetés.

- **options** : Menu-vue
- **setup** : Menu-vue
- **setupSnippet** : Menu-vue
- **setup** : Menu-vue
- **plugin**:Menu-vue
- **log** :Menu-vue

L'utilisation de **PropTypes** permettra de vérifier les arguments employés, afin de ne pas dépasser les paramètres.

Le fichier **Routes** est placé dans le dossier **Pages**. Nous utilisons le parameter **to** plugin.

CONVERSION Modal

Le plugin **Modale**: offre un message de confirmation de création d'employé. Une fonction de fermeture de celui-ci permet de fermeture du site. Nous

CONVERSION Table

Le fichier **DataTable.js** contient une table de présentation des employés.

Il importe 3 fichiers de configuration :

- Le **array** contenant les employés
- Le **numbers** de l'index selon le "tri" choisit affiché.

La fonction **sortBy**(b) permet d'indiquer au composant parent (Vue) le sens du tri (ascendant ou descendant).

La fonction **tri** et la fonction **sortBy** sont implémentées dans Vue.

La fonction **tri** permet d'indiquer au composant parent (Vue) le sens du tri (ascendant ou descendant).

La fonction **sortBy** sera sélectionnée au composant parent

Le contenu de data peut aussi être à la modifié de la source, ou par le JS par l'intermédiaire du **store**.

Lorsque l'utilisateur ajoute la liste de selection, le **display: none** présent dans le **CSS** permettant de l'inspecter à nouveau.

CREATE

Le fichier **CreateEmployee.js** contient le composant permettant la création d'employé.

Les fonctions suivantes y sont présentes :

- la **formulaire** contrôlé par React

- la **vérification** et la **validation** des données de l'employé

Les différentes **erreurs** sont sur les **divers** correspondantes

- le **display** de la modale

- l'ajout de l'employé créé à la base de données
- le **reset** du formulaire et de ses valeurs

VIEW

Le fichier **ViewEmployees.js** contient le composant permettant la visualisation de la base de données.

Les fonctionnalités suivantes y sont présentes :

- **numéros** de pages et le **nombre** de résultat
- **affichage** de la **table** selectionnée

- Fonction de **recherche**

Les variables de pagination et de tri sont à la différence des autres composants, stockées dans le store global.

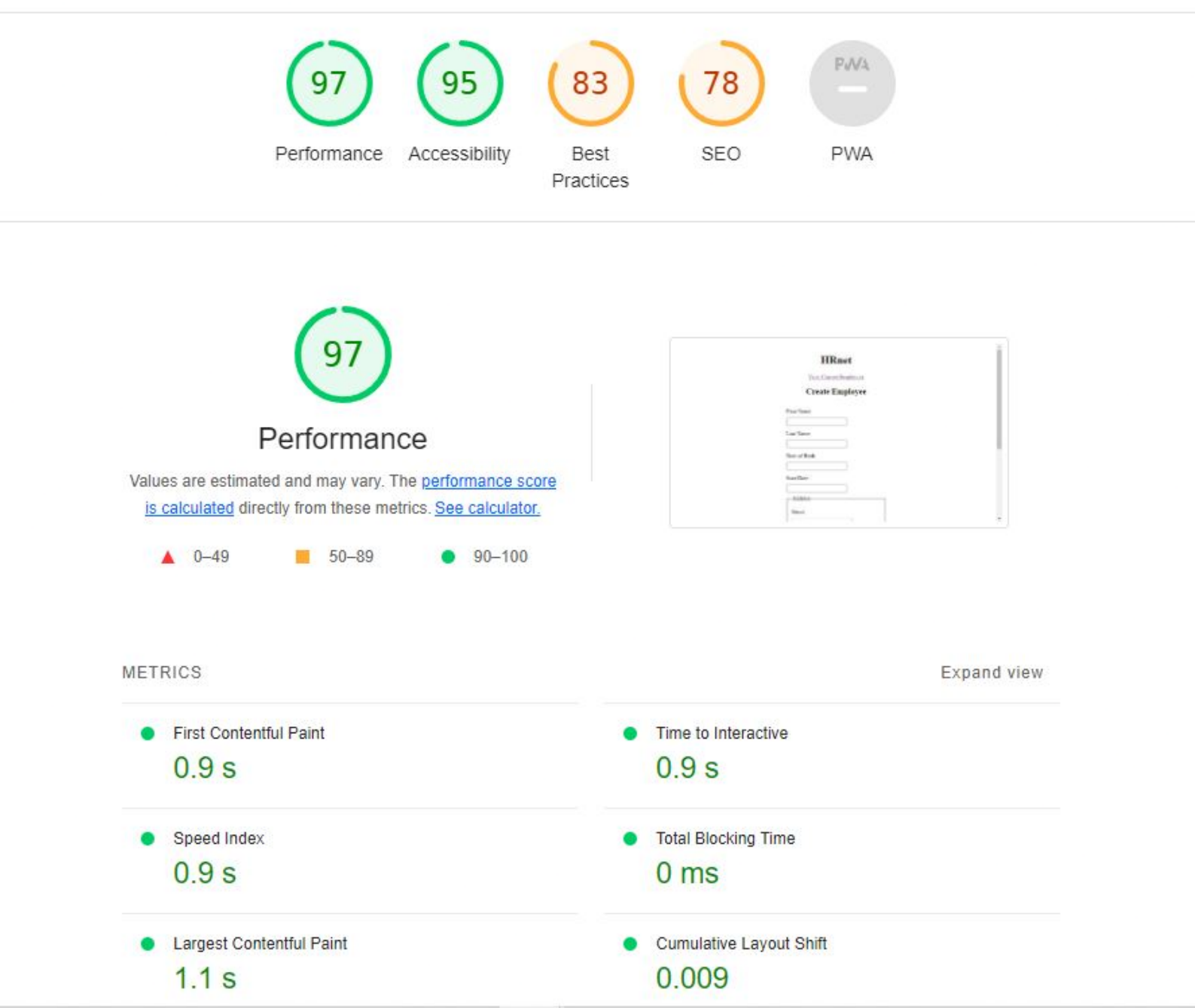
TEST LIGHTHOUSE

Le test Lighthouse est réalisé sur le serveur production. Il réalise des performances supérieures au test de la version comprenant les plugins jQuery.

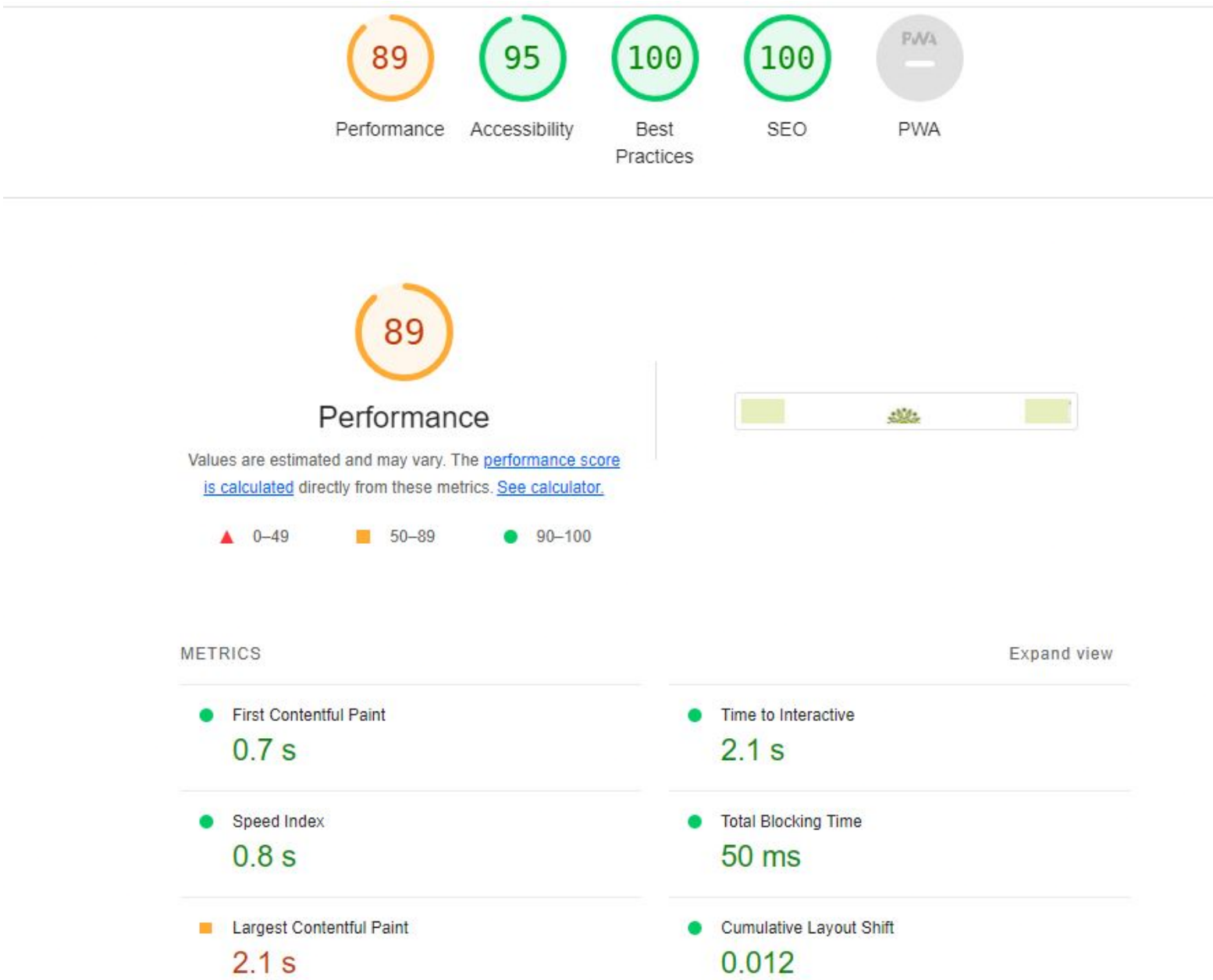
Rapport de performance LightHouse

Un test LightHouse est réalisé sur la version production. Il révèle des performances supérieures au test de la version comprenant les plugins jQuery.

Test LightHouse **sans** l'import du plugin jQuery converti:



Test LightHouse **avec** la version production:



Le composant converti est disponible sur npmjs à l'adresse suivante:
https://www.npmjs.com/package/dropdown-menu-select-plugin

Ainsi que sur le github suivant:
https://github.com/Cdrohm/Dropdown-menu