

Sensibilisation

Numérique Responsable

Travaux Pratiques

Table des matières

1.	Enoncé	2
2.	Présentation fonctionnelle de l'application	3
3.	Présentation technique de l'application	4
3.1.	Généralités.....	4
3.2.	Frontend	5
3.3.	Backend.....	5
3.4.	Accès à la base de données	6
4.	Outils à disposition	7
4.1.	Le navigateur web.....	7
4.1.1.	L'onglet « Réseau »	7
4.1.2.	L'onglet « Performances »	8
4.1.3.	Le plugin « GreenIT ».....	9
4.1.4.	L'onglet « Lighthouse » de Chrome	11
4.2.	La page « statistiques ».....	12
4.3.	SonarQube	13
4.4.	L'utilitaire « Chrono.java »	13
4.5.	Le code coverage	13
4.6.	Les logs et le mode debug	14
4.7.	Can I use ?	14

1. Enoncé

Vous êtes développeur de l'application « Aubay - NR », c'est une application récente en Java 22 et Angular 14, basée sur Spring Boot. Le client ne semble pas satisfait des performances de l'application, cela semble cacher des problèmes structurels, des mauvaises pratiques de développement, et des configurations douteuses.

Suite à la sensibilisation que vous venez de recevoir, vous devez prendre quelques heures pour identifier des problèmes, imaginer des solutions et pourquoi pas tenter de les corriger, vous avez carte blanche tant qu'il n'y a pas d'impact fonctionnel.

Chacun des problèmes identifié et éventuellement résolu vous rapportera un certain nombre de points (Une solution théorique rapporte moins de points qu'une solution fonctionnelle). Pensez à noter vos idées, une idée qui n'est pas écrite avant la fin du TP ne rapporte pas de point !

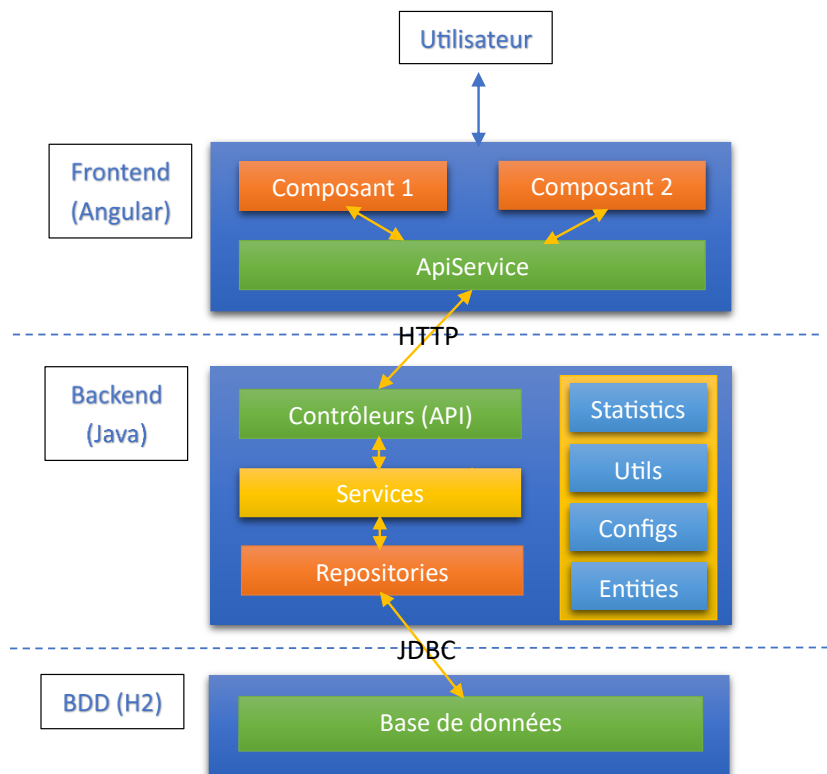
2. Présentation fonctionnelle de l'application

L'application est composée de 3 pages :

- Un module d'authentification, protégeant l'accès au contenu.
 - o Compte à utiliser : **user / password**
 - o Un bouton « Se déconnecter » est présenté dans le header une fois authentifié.
- Une page « Ressources humaines » :
 - o L'ensemble de la hiérarchie de l'entreprise est présenté. Il est alors possible de consulter les équipes de chacun des managers (Cliquer sur un manager « ouvre » ou « ferme » son équipe).
 - o Il est possible de modifier chacun des employés (Nom, prénom, pays et salaire) en cliquant sur une petite icône à côté de leur nom.
 - o Les managers ayant une équipe cachée sont en bleu, les managers ayant une équipe visible sont en vert, et les autres employés (non-managers) sont en blanc.
 - o Le nombre total d'employés apparaît sous le titre.
 - o Le nombre d'employés sous chacun des managers apparaît à côté de leur nom.
 - o Une petite fiche présente les données des employés à leur survol (Nom, prénom, salaire, date d'entrée et pays).
 - o Les employés ayant quittés l'entreprise ne doivent pas apparaître ici
- Une page « Statistiques » :
 - o Les performances de l'application sont présentées.
 - o Chacune des API est monitorée et présente :
 - Le nombre de requêtes SQL par appel API (Moyenne)
 - La durée moyenne d'un appel API
 - Le poids moyen de la réponse
 - Le nombre d'appels à cette API
 - o Les statistiques sont présentées de différentes couleurs en fonction de leur impact :
 - Normales : **en vert**
 - Non-négligeables : **en bleu**
 - Mauvaises : **en orange**
 - Très mauvaises : **en rouge**
 - Extrêmement mauvaises : **En rouge et en gras**
 - o Un graphique « bar » permet de consulter l'évolution dans le temps des statistiques :
 - Période d'analyse (depuis le dernier nettoyage des statistiques) découpées en 20 sous-périodes (Une sous-période = 1 barre, 20 barres au total, donc).
 - L'échelle est linéaire et automatique en fonction de la pire valeur.
 - Le survol d'une barre donne la valeur moyenne pour la sous-période donnée.
 - o Ce graphique est paramétré par une liste déroulante à côté, permettant de choisir le type de données souhaitée :
 - Durée d'exécution
 - Nombre de requêtes SQL
 - Poids des réponses
 - o Il est possible de choisir quelles API doivent être pris en compte dans le graphique (Case à cocher individuelle pour chaque API ou case à cocher « Tous »).
 - o Un bouton « Réinitialiser les statistiques » permet d'effacer les statistiques de la base de données et de recommencer une nouvelle période. Avec popup de confirmation.
 - o Un bouton « Rafraichir les données » est présenté en haut du tableau de statistiques.

3. Présentation technique de l'application

3.1. Généralités



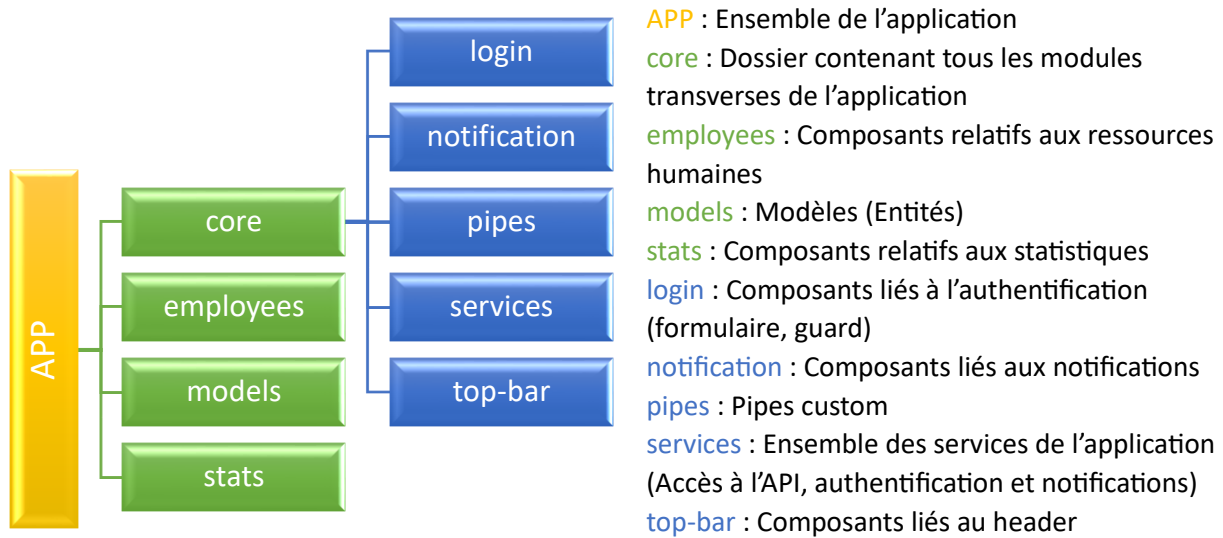
L'utilisateur utilise son navigateur web pour exécuter l'application Angular. Celle-ci construit une interface entre l'utilisateur et les services que propose le serveur backend et lui permet d'interagir.

Les communications entre le frontend et le backend se font en http, principalement via des flux JSON. Ces flux sont pour la plupart protégés par un moyen d'authentification et par protection CSRF (via cookie pour la mise en place, puis header).

La base de données est une base de données externe, hébergée sur un serveur distant et spécialisé, comme c'est habituellement le cas en entreprise.

3.2. Frontend

Le frontend est une application Angular, dont la structure est relativement simple :



Les composants liés à l'authentification sont donc dans « **app/core/login** ».

Les composants liés à la première page (Ressources humaines) sont donc dans « **app/employees** ».

Les composants liés à la seconde page (Statistiques) sont donc dans « **app/stats** ».

Installation du frontend : **npm install**

Démarrage du frontend : **ng serve**

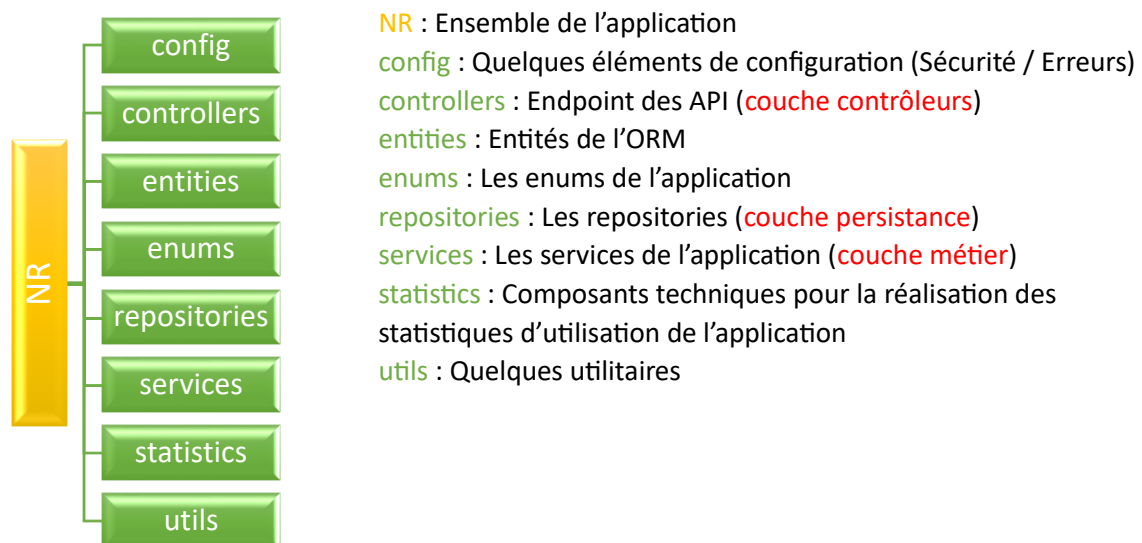
3.3. Backend

Le backend est basé sur Spring Boot, les contrôleurs REST sont dans le package « *controllers* ».

Les configurations relatives à la sécurité sont dans *WebSecurityConfig* du package « *config* ».

Les configurations relatives à l'application sont dans « *src/main/resources/application.properties* ».

Les configurations relatives aux logs sont dans « *src/main/resources/logback.xml* ».



Installation du backend : **mvn install** (Non obligatoire avec m2e)

Démarrage du frontend : Exécuter la classe **WebApplication** en tant qu'application Java

3.4. Accès à la base de données

La base de données est un SGBD H2 sur le réseau local. La configuration d'accès à cette base de données est dans le fichier `application.properties`, celui-ci devra donc être adapté selon votre environnement.

```
spring.datasource.url=jdbc:h2:tcp://192.168.3.84:9090/file:/data/app;Mode=Oracle
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=password
spring.datasource.hikari.connectionTimeout=1000
spring.jpa.properties.hibernate.hbm2ddl.auto=create
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.jpa.open-in-view=false
spring.jpa.show-sql=true
```

Il est possible d'accéder à la console d'administration de cette base de données via l'URL (A adapter selon l'IP du serveur) : <http://192.168.3.84:8080/h2-console/>

English Preferences Tools Help

Login

Saved Settings: Generic H2 (Embedded)

Setting Name: Generic H2 (Embedded) Save Remove

Driver Class: org.h2.Driver

JDBC URL: jdbc:h2:tcp://192.168.3.84:9090/file:/data/app;Mode=Ora

User Name: sa

Password: password

Connect Test Connection

Attention, cette base de données est partagée avec tous les participants à la formation.

4. Outils à disposition

4.1. Le navigateur web

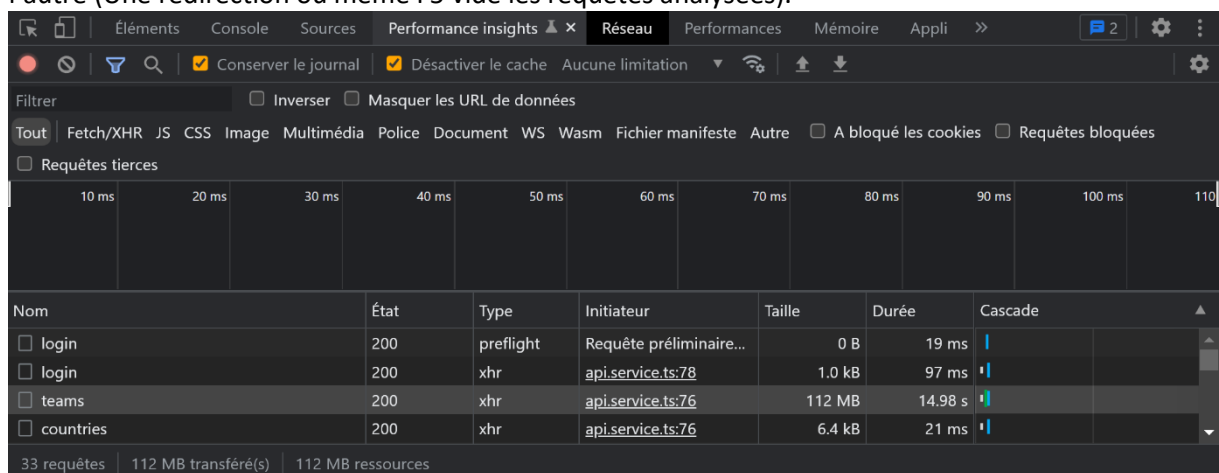
Le premier outil du développeur web, c'est le navigateur web et les outils développeurs accessibles via F12. Ces outils inclus entre-autres :

4.1.1. L'onglet « Réseau »

Cet onglet présente toutes les requêtes réalisées par le navigateur avec des serveurs distants. Il permet de consulter les données brutes échangées, et analyse leur poids, les délais de réponses et de traitement, leur type, leur compression, s'ils proviennent du cache, le temps nécessaire à charger le DOM, les erreurs rencontrées, les cookies et headers échangés, etc.

Cet onglet doit être parfaitement maîtrisé par les développeurs, y compris les plus débutants !

TIP : Pensez à cocher la case pour désactiver le cache, et celle pour conserver le journal d'une page à l'autre (Une redirection ou même F5 vide les requêtes analysées).



The screenshot shows the Chrome DevTools Network tab. The top bar includes filters for 'Tout', 'Fetch/XHR', 'JS', 'CSS', 'Image', 'Multimédia', 'Police', 'Document', 'WS', 'Wasm', 'Fichier manifeste', and 'Autre'. Below the filters, a timeline view shows request durations from 10 ms to 110 ms. The main table lists the following requests:

Nom	État	Type	Initiateur	Taille	Durée	Cascade
<input type="checkbox"/> login	200	preflight	Requête préliminaire...	0 B	19 ms	
<input type="checkbox"/> login	200	xhr	api.service.ts:78	1.0 kB	97 ms	
<input type="checkbox"/> teams	200	xhr	api.service.ts:76	112 MB	14.98 s	
<input type="checkbox"/> countries	200	xhr	api.service.ts:76	6.4 kB	21 ms	

At the bottom, a summary bar shows: 33 requêtes | 112 MB transféré(s) | 112 MB ressources.

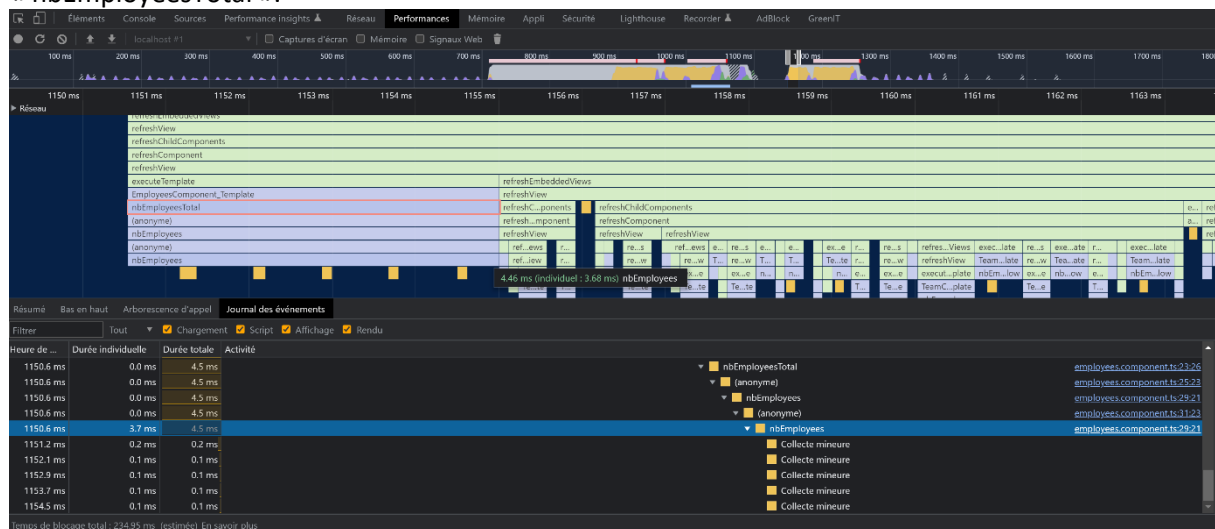
4.1.2. L'onglet « Performances »

Cet onglet permet de réaliser un profilage des performances JavaScript de l'application.

Une fois la page prête à être analysée, un clic sur le bouton « Enregistrer » (ou CTRL+E) permet de démarrer la capture. Un second clic sur « Arrêter » (ou CTRL+E encore) permet d'arrêter la capture et d'obtenir le résultat d'analyse.

Il convient ensuite de rechercher dans le graphique les tâches ayant retenue le plus de temps pour identifier les améliorations possibles. Essayez d'être le plus précis possible sur la capture (utilisez CTRL+E !) pour réduire la quantité d'informations à analyser.

À tout moment, l'ensemble de la stack est présentée à l'écran. On peut par exemple constater dans la capture ci-dessous qu'à 1151ms après le début de l'analyse, Angular est entré dans la fonction « nbEmployeeTotal » pendant 4.5ms et qu'à l'intérieur de cette fonction, c'est 4 niveaux plus profonds que la fonction « nbEmployees » a pris le plus de temps (3.7ms à elle seule, sans compter le contenu de sa propre stack, ayant pris le reste du temps). Le reste du temps d'exécution (celui de la fonction « nbEmployeesTotal » entre-autres, est négligeable. Il serait donc nécessaire d'améliorer les performances de la fonction « nbEmployees » si on souhaite améliorer les performances de la fonction « nbEmployeesTotal ».



Cet onglet permet aussi d'analyser l'état de la mémoire au cours du temps et des actions JS.

4.1.3. Le plugin « GreenIT »

GreenIT-Analysis est une extension pour navigateur qui vous permet de quantifier les impacts environnementaux d'un parcours utilisateur complet, même derrière un firewall et / ou une authentification applicative. L'outil vérifie également l'utilisation de bonnes pratiques visant à diminuer ces impacts.

Cette extension s'inspire fortement des fonctionnalités de EcoIndex et EcoMeter.

Elle est aussi supportée sur Firefox avec quelques bonnes pratiques non supportées (du fait de limitations des API Firefox). A noter que bien que fonctionnant sur la version ESR de Firefox, le plugin a des comportements particuliers, voir point Firefox ESR ci-après.

Fonctionnalités :

- Estimation des GES en gCO2eq
- Estimation de l'eau consommée
- Mesure du poids total de la page en ko
- Mesure du nombre de requêtes
- Mesure de la taille du DOM (à priori en lignes)
- Calcul d'un EcoIndex (meilleure valeur : 100)
- Vérification de 21 bonnes pratiques
- Historique des analyses pour constater une amélioration/détérioration

Installer l'extension :

Depuis Chrome, Chromium ou Edge :

<https://chrome.google.com/webstore/detail/greenit-analysis/mofbfhffeklkbebfclfaifefjflcpad>

Depuis Firefox ou Firefox ESR :

<https://addons.mozilla.org/fr/firefox/addon/greenit-analysis/>

Utiliser l'extension :

1. Ouvrir les outils de développement du navigateur (F12).
2. Aller dans l'onglet GreenIT.
3. Dans le navigateur, aller sur la page à analyser.
4. Dans l'onglet GreenIT des outils de développement, cliquer sur le bouton "Lancer l'analyse".
5. Les résultats s'affichent.
6. Vous pouvez sauvegarder ce résultat dans un historique (seuls les indicateurs sont enregistrés) via le bouton "Sauver l'analyse"
7. L'historique des résultats sauvegardés est disponible via le bouton "Historique"
8. Pour avoir une analyse des bonnes pratiques, il faut cocher la case "Activer l'analyse des bonnes pratiques".

Quelques points de vigilance :

- Si le nombre de requêtes est à zéro, c'est probablement parce que vous n'avez pas chargé la page avec les outils de développement démarrés. Il faut donc penser à faire un rechargement de la page.
- Pour avoir des mesures correctes, il faut préalablement vider le cache du navigateur (Dans le cas contraire, le volume transféré va être réduit si vous avez déjà consulté le site mesuré). Pour vous éviter d'aller dans les menus du navigateur, un bouton est prévu à cet effet dans l'extension.
- L'utilisation d'un bloqueur de publicité ou autre filtre a une influence sur le résultat.

ÉlémentsConsoleSourcesPerformance insightsRéseauPerformancesMémoireAppliGreenT>>12

Lancer l'analyseVider le cache navigateurSauver l'analyseHistoriqueAide

☒ Activer l'analyse des bonnes pratiques

EcolIndexE

EcolIndex	Eau (cl)	GES (gCO2e)
30.67	3.58	2.39

Nombre de requêtes	Taille de la page (Ko)	Taille du DOM
28	119906 (119905)	3586

Bonnes pratiques

Ajouter des expires ou cache-control headers

Compresser les ressources (>= 95%)

Limiter le nombre de domaines (<6)

Ne pas retailer les images dans le navigateur

Externaliser les css et les js

Eviter les requêtes en erreur

Limiter le nombre de requêtes HTTP (<27)

5 / 18 ressources cachées

0% ressources compressées

5 domaine(s) trouvé(s)

2 image(s) retailée(s) dans le navigateur

17 inline stylesheet(s) et inline javascript(s)

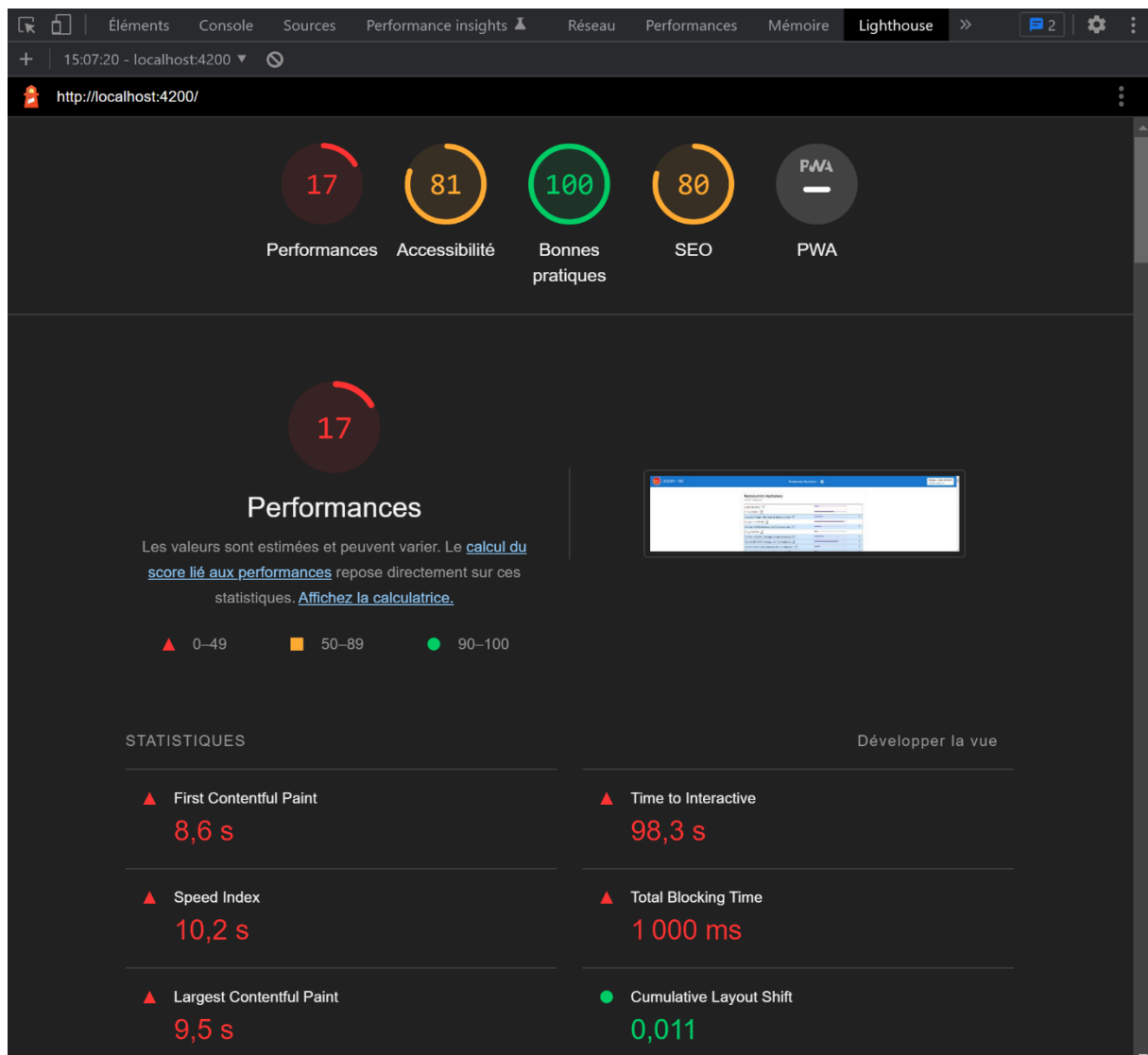
1 erreur(s) HTTP

28 requête(s) HTTP

4.1.4. L'onglet « Lighthouse » de Chrome

Cet onglet permet de faire un audit d'une application web en 1 clique et d'analyser aussi bien les performances, l'accessibilité, les bonnes pratiques (Sécurité, demande de droits...) et les SEO (conseils pour le référencement naturel).

De très nombreux cas sont testés et tous les conseils sont bons à suivre (Sauf ceux de SEO et PWA quand ils ne sont pas applicables à l'application).



4.2. La page « statistiques »

La page « Statistiques » de l'application est elle-même une mine d'or pour identifier les potentiels défauts de l'application, particulièrement sur le long terme grâce à l'historique.

Cette page permet d'afficher des statistiques sur les temps de traitements de chacune des API, le nombre de requêtes SQL moyen à chaque appel de ces API, le poids moyen des réponses et le nombre d'appel à chacune des API.

De quoi facilement détecter les services qui ne sont pas utilisés, ceux qui présentent des problèmes de type N+1 (Requêtes SQL), ceux qui ont une durée de traitement qui augmentent au fil du temps, etc.

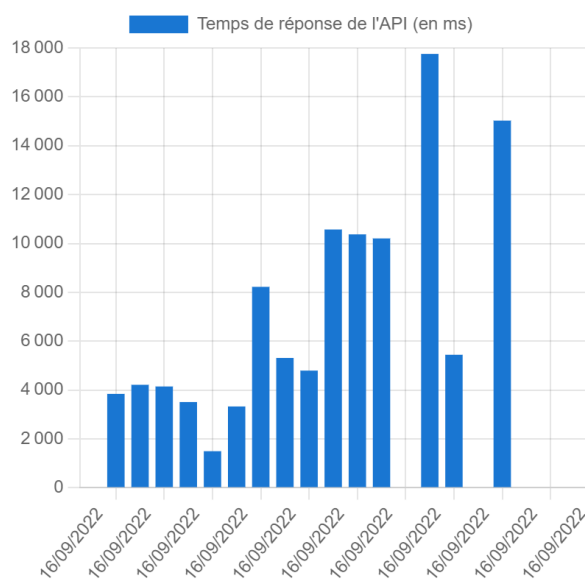
Toutes les API sont automatiquement scannées, y compris de potentielles nouvelles et présentées dans le tableau, avec ses caractéristiques mesurées.

Les valeurs apparaissent en couleur pour mettre en avant les valeurs les plus problématiques.

Statistiques



URI	Requêtes SQL par appel API	Durée moyenne appel API	Poids moyen de la réponse	Nombre d'appel API	
DELETE /stats	4	64 ms	0 o	1	<input type="checkbox"/>
GET /stats	1	72 ms	3 Ko	22	<input type="checkbox"/>
GET /countries	1	78 ms	6 Ko	15	<input type="checkbox"/>
GET /employees/top	1526	8 s	306 Ko	15	<input checked="" type="checkbox"/>
POST /employees	490	3 s	100 Ko	10	<input checked="" type="checkbox"/>
POST /init	1035	5 s	0 o	3	<input type="checkbox"/>



TYPE DE DONNÉES

Durée d'exécution

Injecter plus de données

Réinitialiser les statistiques

4.3. SonarQube

Sonar est capable nativement de détecter de nombreuses mauvaises pratiques en réalisant une analyse statique du code source. Beaucoup de ces pratiques ont un effet lié au numérique responsable, mais malheureusement il y a encore beaucoup d'autres règles à suivre qui ne sont pas encore pris en compte par Sonar.

De nombreuses entreprises, dont Aubay, se sont alliées pour développer le projet OpenSource Creedengo, un plugin Sonar, permettant d'ajouter à Sonar les règles aujourd'hui identifiées dans le cadre du numérique responsable.

<https://green-code-initiative.org>

<https://github.com/green-code-initiative/creedengo-rules-specifications>

Ce plugin est téléchargeable depuis le marketplace Sonar : **Administration > Marketplace**

4.4. L'utilitaire « Chrono.java »

Le petit utilitaire Chrono permet de facilement chronométrer une durée d'exécution d'un script.

Appelez **Chrono.start()**, puis **Chrono.trace(String)** pour tracer dans les logs le nombre de millisecondes s'étant écoulées depuis l'appel à la méthode start(), ou depuis le dernier appel à la méthode trace(String).

Si le Chrono est envoyé sur un serveur non-local, pensez à appeler **Chrono.stop()** pour prévenir des fuites de mémoire.

Par exemple :

```
Chrono.start();
employee.fired();
Chrono.trace("Employee fired");
employeeRepository.save(employee);
Chrono.trace("Employee saved");
Chrono.stop();
```

Dans les logs :

```
[CHRONO] Employee fired (67ms)
[CHRONO] Employee saved (24ms)
```

Le Chrono est Thread-safe et peut-être appelé y compris dans des sous-méthodes, sans avoir à transmettre d'information (Comme la date de début du Chrono).

4.5. Le code coverage

La couverture en tests unitaire peut être mesurée directement dans Eclipse (Via un Run Coverage) ou via des plugins à ajouter à la plateforme d'intégration continue, comme JaCoCo ou Corbertura.

Ces outils permettent de surligner en vert les instructions faisant l'objet de tests unitaires (Au moins un TU exécute cette instruction), en jaune les instructions testées, mais avec un ou plusieurs scénarios alternatifs non testés (Par exemple, les TU ne testent le code que si la condition est vraie, mais jamais

si elle est fausse : La ligne passe jaune). Et en rouge, toutes les instructions non testées par les tests unitaires.

Un taux de couverture élevé est un bon indicateur pour éviter les régressions dans le code, suite à des évolutions / corrections. Ces taux de couvertures sont calculés au global, ou par classe, ou par méthode, il est fortement recommandé d'avoir une couverture de plus de 80% du total du code.

4.6. Les logs et le mode debug

Hibernate propose une propriété `show_sql` (Avec Spring boot : `spring.jpa.show-sql=true`) pour afficher les requêtes SQL qui seront exécutées au niveau de la base de données. Cela fonctionne aussi bien avec JPA, JPA repositories, les criteria, etc.

Il est aussi possible de demander à hibernate de formater avec un peu d'indentation ces requêtes dans les logs : `spring.jpa.properties.hibernate.format_sql=true`.

Les bibliothèques et frameworks ont parfois des comportements étranges pour les développeurs. Pour les aider à comprendre, de très nombreux logs sont prêts à être activés.

4.7. Can I use ?

Les navigateurs web présentent chacun des limites aux fonctionnalités qu'ils proposent, et malheureusement tous ne sont pas égaux. Si les fonctionnalités de base en HTML 5, ECMAScript et CSS 3 sont assez bien maîtrisées aujourd'hui par les principaux navigateurs, le web continue d'évoluer et les navigateurs proposent encore des fonctionnalités expérimentales pas toujours bien standardisées.

Attention donc à ne pas exclure certains utilisateurs en utilisant certaines fonctionnalités implémentées que par un nombre réduit de navigateur, ou présentant des comportements différents d'un navigateur à l'autre.

En cas de doutes, <http://caniuse.com> permet de vérifier la compatibilité des fonctions JavaScript ou CSS avec les différents navigateurs.