Déployer une application React

* **1 - Comment déployer mon application React en front ?**

Pour déployer une application REACT en front, il est nécessaire de construire un ensemble de fichiers qui peuvent être interprétés par un navigateur. Les fichiers en questions sont le index.html, le css et le JS avec notre code React à l’intérieur.

Il est possible de gérer ses fichiers avec des bundlers comme par exemples Parcel qui va gérer les fichiers et les compresser en un seul fichier.

On peut aussi utiliser **create react app** qui est un environnement confortable et constitue la meilleure option pour une application web en React.

Il configure notre environnement de développement de façon à nous permettre d’utiliser les dernières fonctionnalités de Javascript, il propose aussi une expérience développeur agréable et optimise notre application pour la production.

Create react app ne prend pas en charge la logique côté serveur ni les bases de données, il crée simplement une chaîne de construction pour la partie frontale, de sorte que l’on peut utiliser le serveur de notre choix

pour utiliser create react app nous aurons besoin de node et de npm. Lorsque l’application est prête à être déployé en production il nous suffira d’exécuter la commande “npm run build” pour créer une version optimisée de notre application

Une fois nos fichiers optimisés, on peut placer notre projet dans un référentiel github et utiliser un outil de déploiement automatisé comme **DeployBot.**

* **2 - Comment déployer mon application disposant d'un back-office de type API avec une application front en React ?**

il y a plusieurs manières de procéder :

* On peut déployer le front-end sur une plateforme comme AWS qui est un ensemble d’outil et qui inclut un service d’hébergement web et l’API sur un serveur différents pour ensuite appeler l’API depuis l’application front-end
* On peut aussi configurer et exécuter le client et le serveur sur le même serveur à l’aide d’un fichier gulp, gulp est un task runner càd qu’il est capable de lancer des bouts de scripts tout seul, après il faut déployer le projet sur une même plateforme d’application

AWS permet de déployer son application plus simplement il faut créer un référentiel GitHub dans lequel on viendra déposer notre code

Il faut donc donc déployer une application React dans le Cloud AWS en l’intégrant à Github et en utilisant AWS Amplify, AWS Amplify permet de déployer en continu notre application dans le cloud et l’héberger sur un CDN

* **3 -Étant donné que React est du code généré,**

c-a-d que react permet via des systèmes de composants de générer une portion ou une page en html à chaque changement d'état. Le code généré doit être interprétable dans le navigateur. Cela se fait via la génération d’un DOM virtuel que react composera selon les syntaxes qu’il utilise.

*React utilise la syntaxe JSX pour le rendu de ces composants. JSX permet de décrire une structure DOM directement en javascript dans un langage proche du html. Cette extension syntaxique de Javascript permet de faciliter l’écriture et la lecture mais comporte quand même des différences avec le html (on doit donc avoir un compilateur JSX pour traduire n’importe quel code JSX en Javascript)*

Une page HTML Classique est représentée sous forme d’un arbre DOM dans le navigateur : des balises HTML qui s'emboîtent et se suivent. Chaque modif du DOM entraîne les opérations de reconstruction du nouveau dom (CSS, position et attributs HTML, affichage dans navigateur).

C’est la bibliothèque javascript ReactDOM qui contient ces méthodes spécifiques à React et qui traitent le DOM. Par exemple ReactDOM.render() est un moyen pour effectuer le rendu d’un élément JSX. Il prend l’expression JSX et crée un arbre correspondant aux nœuds du DOM puis il ajoute cet arbre au DOM.

Une interface React est donc découpée en composants assimilables à des fonctions. Un composant est une boite noire qui reçoit des arguments (props) en entrée et retourne du DOM virtuel en sortie. Une fois combinés, ces composants constituent une application complète.

Avec React.js, pour chaque objet DOM, il y a donc un “objet DOM virtuel” correspondant. Un objet DOM virtuel est une représentation d’un Objet DOM, comme une copie allégée. Il a les mêmes propriétés qu’un objet DOM réel, mais il n’a pas le pouvoir de changer directement ce qui s’affiche à l’écran. Entre deux mises à jour, React va donc comparer l’ancien et le nouveau DOM virtuel pour déterminer où effectuer d’éventuels changements. React ciblera les noeuds ou des modifs doivent être faites et les appliquera.

En gros React génère une représentation du DOM en Javascript grâce à laquelle on pré-calcule les changements nécessaires pour n’en faire que le minimum ce qui minimise les opérations et améliore grandement les performances de l’application.

Grâce à l’utilisation d’un serveur Node par exemple, le code va pouvoir être généré côté client et serveur à la différence des autres frameworks traditionnels. Le contenu généré par Javascript peut ainsi être lu et référencé par les moteurs de recherche :

* **4 - comment faire en sorte que mon application soit référencée sur les moteurs de recherche ?**

4- a - Qu’est-ce que le référencement ?

Le référencement naturel (SEO) désigne l’ensemble des techniques utilisées pour faire remonter son site sur la première page Google. Etant donné que la majorité des internautes ne vont pas plus loin que la 1ère page des résultats lors d’une recherche (91% [selon SEMJuice](https://www.semjuice.com/blog/viser-premiere-page-google-m7#:~:text=Les%20chiffres%20parlent%20d'eux,France%20se%20font%20sur%20Google.&text=Une%20fois%20la%20recherche%20des,pour%20trouver%20l'information%20recherch%C3%A9e.)), il faut se positionner le plus haut possible sur des mots-clés qui correspondent aux produits ou services vendus. Pour le référencement d’un site “classique”, on optimise techniquement le site en faisant notamment attention au chargement des pages, des images (le moins lourde possible), des scripts, à l’insertion des bonnes balises title, l’indexation des pages… On peut également s’appuyer sur des backlinks (liens d’un site x qui renvoie vers y). Depuis l’année dernière, Google s’appuye sur un principe d’EAT pour évaluer la qualité du lien :

* E pour Expertise : si le site x est reconnu par Google comme étant expert dans son domaine, la qualité du lien renvoyé vers y sera plus qualitative.
* A pour Autorité : si le site est déjà reconnu et bien classé par l’algorithme de Google, la qualité du lien sera meilleure.
* T pour trustworthiness ou Digne de confiance : tout ce qui montre que le site est sécurisé et que (notamment pour le e-commerce) les avis des consommateurs sont bons, par exemple.

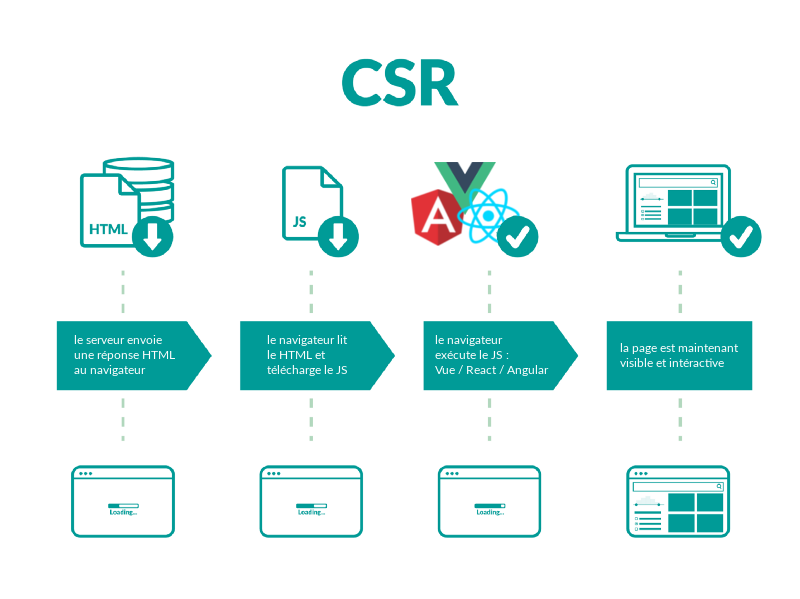
4- b - Pourquoi le référencement d’une application React pose problème ?

En React, on peut partir avec un désavantage côté référencement du fait de l’incompréhension des robots Google du JavaScript, même s’ils en sont de plus en plus capable. Trois facteurs sont à prendre en compte selon [openclassrooms](https://openclassrooms.com/fr/courses/5922626-optimisez-le-referencement-de-votre-site-seo-en-ameliorant-ses-performances-techniques/6055256-decouvrez-comment-google-interprete-le-javascript) :

* le **crawling**. Googlebot doit être capable d’explorer votre site en naviguant grâce aux liens ;
* le **rendering, ou *rendu***. Googlebot doit être capable de lire le contenu de votre page web ;
* le **budget de crawl**. Googlebot utilise beaucoup de ressources pour parcourir et indexer les sites. S’il a du mal à parcourir votre site et que cela lui demande des ressources trop importantes, il s’arrêtera avant d’avoir terminé.

En effet, une erreur en JS va bloquer l'exécution du code, et donc potentiellement ne pas montrer aux robots tout ce qu’il devrait voir. En React et via les API, les appels sont asynchrones (le code de la page continue de s'exécuter pendant que l'appel vers le serveur est effectué), ce qui implique qu’il faut s’assurer de la réception des réponses avant de rendre l’application pour récupérer les données lors des requêtes externes.

Normalement, le JS est exécuté par le navigateur et l’élément HTML n’est chargé que lorsque le JS est chargé également.



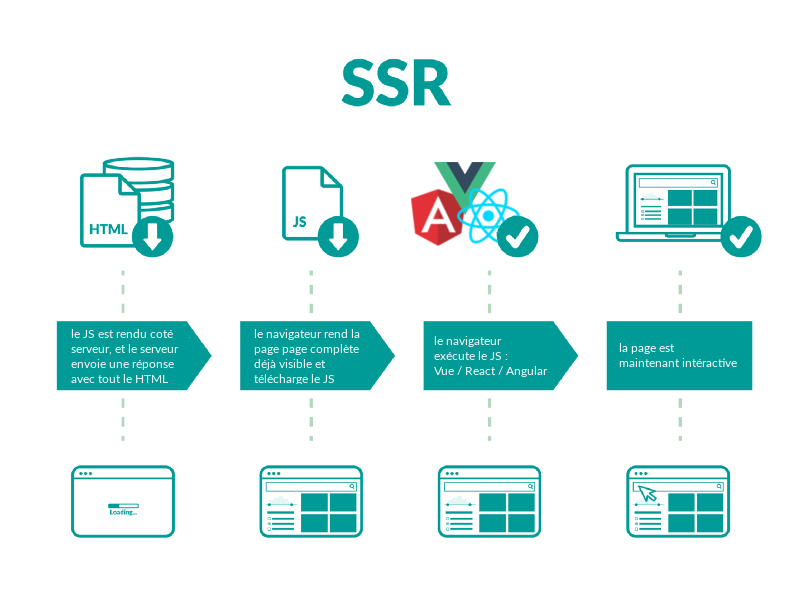
source de l’image : [theodo.fr](https://www.theodo.fr/digital-et-strategie/server-side-rendering-seo-performance)

Sans HTML pré-chargé, les robots peuvent avoir du mal à lire le contenu du site et donc à le référencer efficacement. Il faut ajouter à cela que les temps de chargement sont plus longs, ce qui ne plaît pas à Google qui doit dépenser plus de ressources pour lire le code.

4- c - Méthodes pour référencer correctement son site

Mettre à jour ses balises : en ajoutant la librairie [react-helmet](https://github.com/nfl/react-helmet) à l’application web, on peut automatiquement mettre à jour les balises méta, très utiles en référencement.

Le rendu serveur : en utilisant [Next JS](https://nextjs.org/) par exemple, l’application est exécutée côté serveur et retourne le code HTML au navigateur. Le rendu serveur avec React s’appelle Server Side Rendering (SSR), et le principe est de donner à l’utilisateur et au robot une page complète dans le navigateur avec un code HTML généré en JS par le serveur, pas par le navigateur.



En utilisant le SSR, la page est visible dès la première requête et le code HTML est affiché avant de devenir dynamique. Cette façon de procéder permet de résoudre le problème d’absence de code HTML dans l’application et permet aux robots de comprendre le contenu du site.

[Gatsby](https://www.gatsbyjs.com/docs/) : c’est un framework open-source basé sur React qui permet d’utiliser des composants React en générant du HTML et du CSS pré-rendus pour améliorer le temps de chargement. Gatsby va donc générer autant de pages statiques qu’il y a de pages dans l’application.

Avec ces deux méthodes, on résout les problèmes de lecture HTML par les robots et de chargement, ce qui permet un meilleur référencement de l’application React.

Bien sûr, il ne faut pas oublier d’optimiser les éléments traditionnels nécessaires à un bon référencement, comme l’indexation des url via la Google Search Console et/ou un sitemap, bien renseigner les balises titles, meta description, meta robots, un contenu de qualité etc…