React

- React est une bibliothèques js qui nous permet de créer des interfaces utilisateurs interactives et dynamiques
- La différences entre un bibliothèques est un frameworks est que avec un bibliothèques tu peux utiliser ces éléments de manière libre dans ton code alors qu'avec un framework on te donne des structures et des règles à respecter
- Un composant est comme une fonction ou class qui génère un interfaces en fonction des ses propriétés (props) et etat (state)
- Un props est une valeur ou un objet passé d'un composant parent à un composant enfant
- Un etat est une sorte de mémoire qui peut changer au fil du temps en affectant l'affichage d'un composant
- Le Dom est une arborescence, une représentation en mémoire de notre page html alors que le Virtual Dom est Version légère du Dom utiliser par react et qui affecte légèrement le dom en ne modifiant que les éléments nécessaires
- Une single-page application (SPA) est une application qui charge une unique page html et toute les ressources nécessaires pour le fonctionnement de l'application
- Le cycle de vie d'un composant fait référence aux différentes étapes qu'un composant traverse depuis sa création jusqu'à sa destruction
- Pourquoi utiliser un etat au lieu d'une variable parce que une variable nous permet pas de mettre à jour l'écran
- Il existe deux type de composant qui sont le composant fonctionnel et le class component
- React à été créée en … et publier 2013
- Un **composant fonctionnel** est une fonction qui prend des props en argument et retourne un élément React
- Les **composants de classe** sont plus détaillés et permettent une gestion plus complexe de l'état et des cycles de vie du composant
- Un **hook** en React est une fonction qui permet de "brancher" des fonctionnalités d'état ou de cycle de vie dans un **composant fonctionnel**.
- Un **hook** en React est une fonction qui permet d'ajouter des fonctionnalités (comme l'état local ou les effets secondaires) à un composant fonctionnel.
- `useState` : Permet de gérer l'état local dans un composant fonctionnel.
- `useEffect` : Permet de gérer les **effets secondaires** (par exemple, appels d'API, abonnements, timers, etc.) dans un composant fonctionnel.

- **`useContext`** : Permet de consommer un **contexte** React (une manière de passer des données globales dans l'arbre des composants sans avoir à les transmettre manuellement via des props).
- **`useReducer`** : Une alternative à `useState` pour gérer des états plus complexes, souvent utilisé lorsque l'état nécessite des mises à jour basées sur des actions (similaire à la gestion d'état avec Redux).
- DataFlow c'est à dire comment les données circulent en react les données circulent du composant parent a un composant enfant
- Pour permettre à deux composant de communiquer react utiliser le "lifting state up" est c'est le faite de remonter l'etat vers le composant le plus proche

  Important à savoir sur react
  - ☐ Gestion Formulaire
  - ☐ Gestion evenement
  - ☐ Mapper les listes (clés)
  - ☐ Stats et Props
  - ☐ Components reutilisable