المعهد الوطني للبريد والمواصلات
ⵓⵔⵣⵣⵓ ⵏⵓⵟⵟⴰⵏ ⵏ ⵜⵓⵙⵙⵓⵔⵜ ⴰ ⵜⵜⵓⵙⵙⵓⵔⵜ
Institut National des Postes et Télécommunications

2021/2022

PLATEFORMES & FRAMEWORKS DE DÉVELOPPEMENT WEB

Lab 2 (NodeJS - Typescript / ReactJS - Typescript)

Encadré par :

ALLAKI Driss

MOULINE Hatim

SUD (CLOUD-IOT) S4-P1

I-INTRODUCTION

TypeScript vs. JavaScript ?

- TypeScript est connu comme un langage de programmation orienté objet alors que JavaScript est un langage de script.
- TypeScript possède une fonctionnalité connue sous le nom de typage statique, mais JavaScript ne prend pas en charge cette fonctionnalité.
- TypeScript prend en charge les interfaces, mais pas JavaScript.

Avantages de l'utilisation de TypeScript par rapport à JavaScript

- TypeScript signale toujours les erreurs de compilation au moment du développement (pré-compilation). Pour cette raison, les erreurs d'exécution sont moins probables, alors que JavaScript est un langage interprété.
- TypeScript prend en charge le typage statique/fort. Cela signifie que l'exactitude du type peut être vérifiée au moment de la compilation. Cette fonctionnalité n'est pas disponible en JavaScript.
- TypeScript n'est rien d'autre que JavaScript et quelques fonctionnalités supplémentaires, à savoir les fonctionnalités ES6. Il peut ne pas être pris en charge dans un navigateur cible, mais le compilateur TypeScript peut également compiler les fichiers .ts dans ES3, ES4 et ES5.

L'inconvénient de l'utilisation de TypeScript sur JavaScript : généralement, TypeScript prend du temps pour compiler le code. C'est tout.

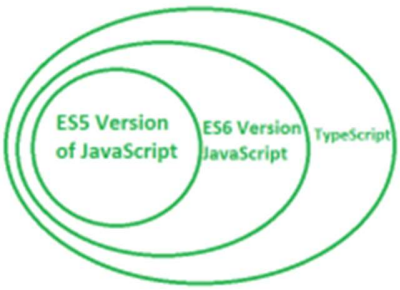
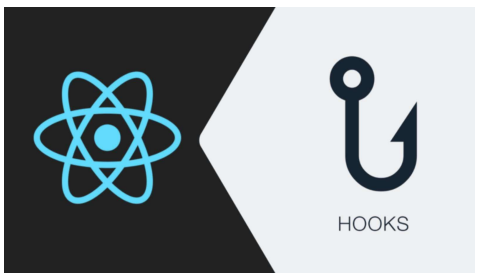


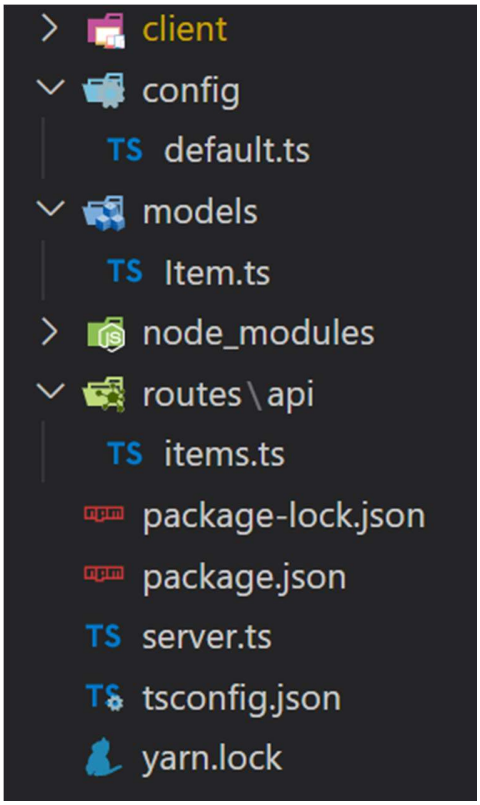
Fig: TypeScript is a SuperSet of JavaScript

Objectif de l'atelier ?

Ce qui est demandé à faire est de refaire l'atelier déjà fait dans les 2-3 dernières séances, en utilisant le langage de programmation TypeScript au lieu de JavaScript, en utilisant Yarn au lieu de NPM, en utilisant la base de données MongoDB de Atlas et enfin d'utiliser des Hooks.

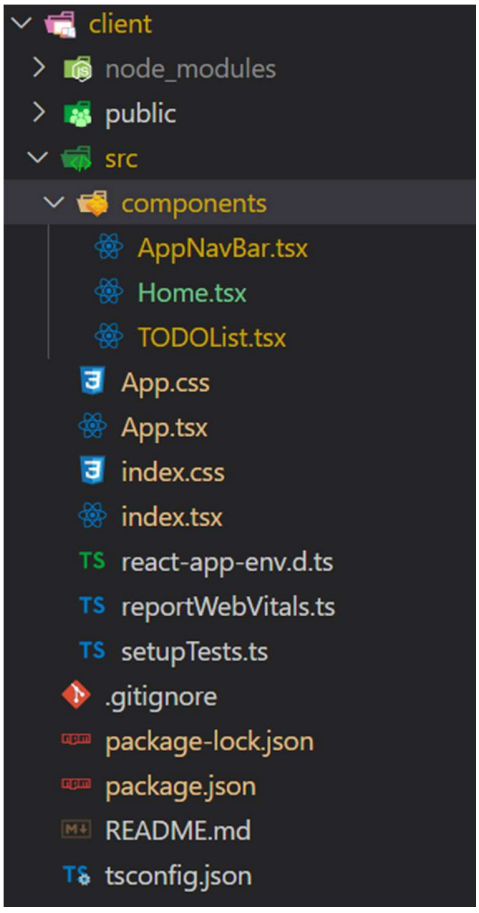


II- Réalisation de l'atelier



Commençons par la partie backend du projet MERN, NodeJS. On a proposé une architecture comme la suivante : un dossier ***models*** pour contenir les modèles qu'on va utiliser (un seul dans notre cas, celui de item), et puis un dossier routes qui contient les APIs (celle des items dans notre cas) et enfin le dossier config pour les fichiers de configuration qui peuvent contenir même des mots de passe et doivent être ajoutés à git ignore.

Passons maintenant à l'architecture de la partie frontend. Comme on peut le remarquer, le même dossier du backend contient un répertoire client. Ce dernier est dédié à la partie cliente avec un projet ReactJS.



Pour faire, index fait appel à App, qui fait appel à Home. Ce dernier fait appel à AppNavBar et TODOList. Ceci est dû au fait que Home est un composant principal qui englobe les deux autres. Dans un projet plus grand et complexe, on pourrait avoir plusieurs composants principaux, ainsi que des composants secondaires chacun dans un dossier séparé contenant son code typescript ou javascript et son propre css pour mieux s'organiser.

Le css utilisé dans notre cas est mis dans le App.css car il n'est pas assez large vue la taille de l'application qui ne contient qu'une seule page.

(Le lien vers GitHub : https://github.com/moulinehatim/MERN_LAB2)

On installe yarn.

```
'yarn' n'est pas reconnu en tant que commande interne
ou externe, un programme exécutable ou un fichier de commandes.

C:\Users\lenovo\Desktop\MERN APP LAB2>npm install -g yarn

added 1 package, and audited 2 packages in 3s

found 0 vulnerabilities

C:\Users\lenovo\Desktop\MERN APP LAB2>yarn create react-app my-app --template typescript
yarn create v1.22.18
[1/4] Resolving packages...
warning create-react-app > tar-pack > tar@2.2.2: This version of tar is no longer supported, and will not receive secur
ty updates. Please upgrade asap.
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...

success Installed "create-react-app@5.0.0" with binaries:
      - create-react-app
[########################################################################] 68/68
Creating a new React app in C:\Users\lenovo\Desktop\MERN APP LAB2\my-app.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template-typescript...

yarn add v1.22.18
info No lockfile found.
[1/4] Resolving packages...
 react
```

Et puis on crée le projet.

```
Microsoft Windows [version 10.0.19044.1586]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\lenovo\Desktop\MERN APP LAB2>yarn init
yarn init v1.22.18
question name (MERN APP LAB2): MERN APP LAB2
error Invalid package name.
question name (MERN APP LAB2): mern_app_lab2
question version (1.0.0):
question description:
question entry point (index.js):
question repository url:
question author:
question license (MIT):
question private:
success Saved package.json
Done in 63.30s.
```

On ajoute les paquets nécessaires…

```
C:\Users\lenovo\Desktop\MERN APP LAB2>yarn add @types/node typescript
yarn add v1.22.18
info No lockfile found.
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...

success Saved lockfile.
success Saved 2 new dependencies.
info Direct dependencies
├─ @types/node@17.0.22
└─ typescript@4.6.2
info All dependencies
├─ @types/node@17.0.22
└─ typescript@4.6.2
Done in 4.14s.
```

La totalité pour le node.js est la suivante :

On fait de même pour react js, on crée le projet client dans celui du backend, puis on installe le package ***concurrently***, pour lancer les deux serveurs et en parallèle avec facilité. On ajoute le ***cors*** dans projet node.js et le ***proxy*** dans le projet react, pour qu'ils puissent fonctionner.

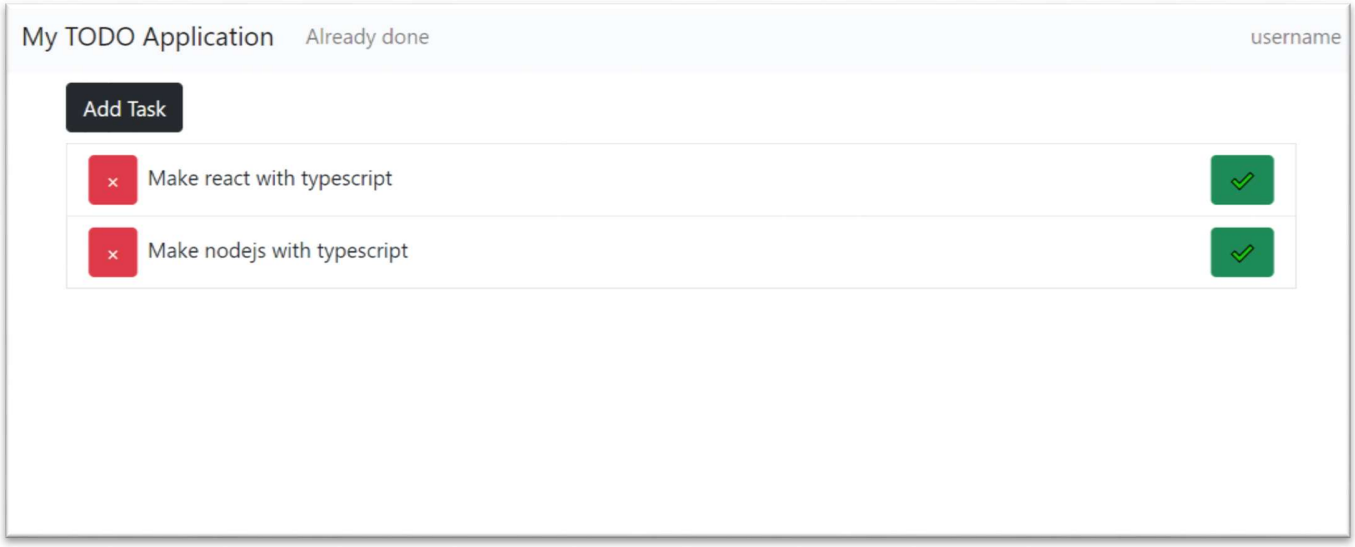
Le frontend se lance dans le port 3001, ainsi que le backend dans le port 5001.

Pour le frontend, au contraire du TP, on utilise les composants avec des fonctions, non pas des classes. Et donc on utilise les Hooks pour la gestion d'état des composants. Les Hooks sont des fonctions qui permettent de « se brancher » sur la gestion d'état local et de cycle de vie de React depuis des fonctions composants. Les Hooks ne fonctionnent pas dans des classes : ils nous permettent d'utiliser React sans classes.

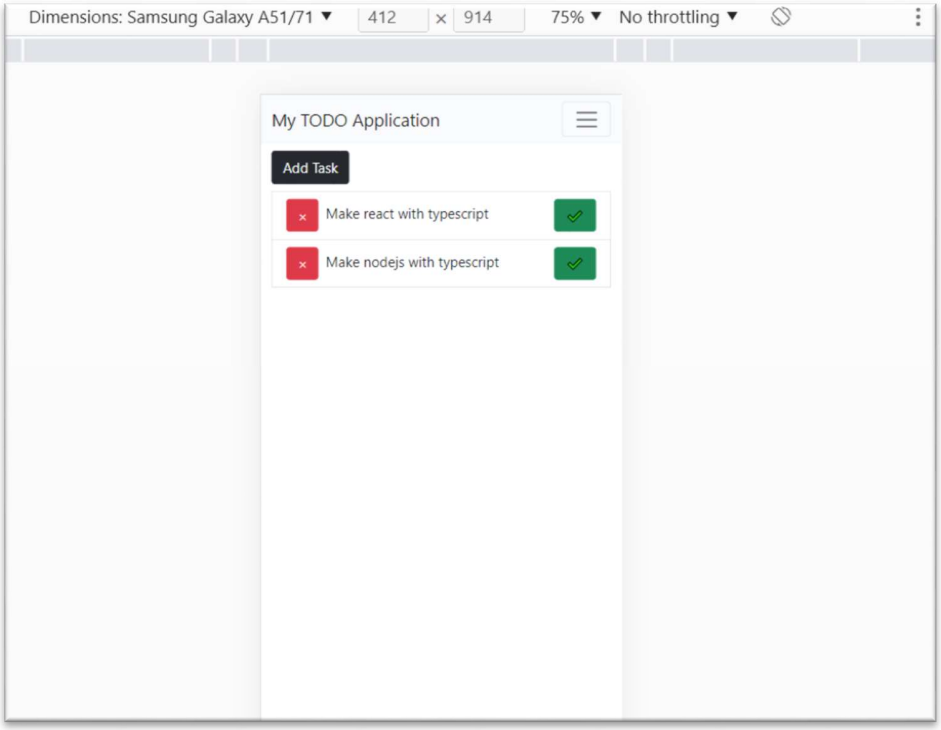
Les Hooks sont une manière très efficace pour le state management, et on va utiliser plus précisément le ***useState*** et ***useEffect*** pour cela dans cet atelier.

- La hook "useState" s'utilise comme tous les Hooks au sein d'un composant fonctionnel. Elle prend en paramètre l'état initial de votre composant et retourne un tuple composé de 2 éléments : l'état que vous manipulez et la fonction qui va mettre à jour cet état.
- Le hook useEffect est un hook qui va permettre de déclencher une fonction de manière asynchrone lorsque l'état du composant change. Cela peut permettre d'appliquer des effets de bords ou peut permettre de reproduire la logique que l'on mettait auparavant dans les méthodes componentDidMount et componentWillUnmount.

L’application web est une interface qui permet à son utilisateur de mentionner ses taches à faire sans qu’il ne les oublie pas.



Et, il est même responsive. Voici la capture d’écran pour une smartphone.



La responsivité est bien présente au niveau du NavBar, qui devient un menu glissant.

La fonctionnalité d’ajout, de suppression sont fonctionnel et le résultat s’affiche automatiquement (state management) quand il doit changer. Mais on ne peut pas visualiser cela dans le repport.

III- Conclusion

Deux choses qui n'était pas mentionné dans le rapport, et qui peuvent s'implémenter aussi mais sortent de ce qui est demandé, sont le bouton vert qui peut être considéré comme un check de la tâche. Et qu'est ce qui ce passe après ? Bon ici vient le rôle de la section dans le menu de '***already done***'. une fois cliqué sur le check vert, on ne supprime pas vraiment la tâche de la base de données mais on la supprime seulement de la page Home (cela peut se faire facilement avec un attribut boolean et une condition if). Là on affiche cette tâche dans l'autre section '***already done***' comme sorte d'historique.

Tous marchent comme prévu, le lien vers GitHub : https://github.com/moulinehatim/MERN_LAB2