

Université Paris Nanterre (Paris 10)

UFR Sciences Economiques ,Gestion,
Mathématiques, Informatique

Rapport de Projet

En vue de l'obtention du Diplôme de Licence MIAGE
(Méthodes Informatiques Appliquées à la Gestion d'Entreprise)

Thème du Projet :
**Développer une webapp de gestion
de listes de tâches en REACTJS**

Réalisé par **Aimed BOUGUERRA**
Encadré par **M. François DELBOT**
Edition Juin 2021

RESUME

Ce document est vise à présenter le projet ReactJS réalisé au cours des mois d'avril, mai et juin 2021 dans le cadre de ma Licence MIAGE au sein de l'Université Paris-Nanterre. J'y développerais les éléments qui composent ledit projet.

Mots-clés : MIAGE, React, developpement, projet, résumé.

ABSTRACT

This document aims to present the React project achieved in april, may and june as part of my MIAGE Licence in Paris-Nanterre University. I will explain all the elements of this project.

Key words: MIAGE, React, development, project, abstract.

Table des matières

RESUME	2
ABSTRACT	2
REACT, UN PEU D'HISTOIRE	5
CADRE DE DEVELOPPEMENT	5
Environnement de travail	5
Langage de développement et Packages	5
Méthodologie de développement	6
DESCRIPTIF DU PROJET	7
MODELE DE DONNEES	7
INTERFACE GRAPHIQUE	8
DESCRIPTION DU PRODUIT FINAL	10
Module Login	10
Module Liste de tâches	10
Détail des fonctions MyToDoListApp.js	11
Menu de Navigation et Gestion des espaces	12
Module Liste de listes	12
Notification par mail	13
LIMITES DU PRODUIT	13
REGARD SUR LE TRAVAIL EFFECTUE	14
Bilan	14
Difficultés et problématiques rencontrées	14

Introduction

L'année 2020-2021 aura été encore fortement marquée par une crise sanitaire sans précédent, malgré ma détermination et ma motivation, je n'ai pu trouver une structure d'accueil pour mon stage de fin de Licence comme de mes nombreux camarades.

Pourtant j'avais mis beaucoup d'énergie, pris beaucoup de contacts, mais le contexte sanitaire a eu raison de mes tentatives. Ainsi, j'ai vu dans cet obstacle une opportunité d'acquérir de nouvelles compétences.

Fort de mes recherches de stage, j'avais constaté que le langage ReactJS est une vive demande sur le marché du travail, aussi lorsque Monsieur Delbot m'a proposé de réaliser mon projet dans ce langage, il m'a semblé intéressant d'ajouter cette corde à mon arc.

Débrouillard et motivé, il ne me semblait pas impossible d'approfondir et perfectionner mes aptitudes en JavaScript en m'engageant dans la voie d'une de ses bibliothèques logicielles (library) les plus populaires, à savoir le ReactJS. J'ai ainsi découvert la simplicité avec laquelle celle-ci permet de développer rapidement des éléments d'application web sans avoir besoin de les recompiler. C'est donc ce projet, ses objectifs et ses développements que je présenterais dans les pages suivantes.

REACT, un peu d'histoire

Avant de se plonger dans le code du projet, il serait intéressant de donner quelques éléments sur cette librairie, ce qui peut éclairer sur son succès. ReactJS naît en 2013, grâce à Jordan Walke, ingénieur chez Facebook, aidé par Peter Hunt, lui-même ingénieur chez Instagram, ils vont travailler à rendre indépendantes de Facebook certaines portions de la bibliothèque.

Contrairement à nombre de ses concurrents, ReactJS n'est pas conçu comme un framework MVC (Model-View-Controller), mais bien comme une simple bibliothèque JavaScript ce qui favorise la création de composants réutilisables. La bibliothèque n'utilise pas de système de templates et ne fonctionne qu'avec JavaScript, ainsi plutôt que de dialoguer avec les APIs du navigateur, comme avec un langage de templating, ReactJS génère une arborescence d'objets JavaScript en mémoire. De ce fait, il permet de créer de très grandes applications qui peuvent charger des données sans avoir à recharger la page.

A partir de 2015, la logique de simplicité est poussée un cran plus loin, puisque naît React Native, qui vise à rendre « agnostique » les applications ainsi exécutables aussi bien sur iOS qu'Android.

CADRE DE DEVELOPPEMENT

ReactJS ne nécessite pas d'environnement ou d'IDE particulier pour son développement. Aussi, dans les premiers temps du projet, j'ai travaillé sur le site [Codepen.io](https://codepen.io) pour développer mon code et le tester en live (Bouguerra, s.d.).

Environnement de travail

J'ai par la suite utilisé une installation locale, puisque le projet a évolué par rapport aux consignes initialement reçues. Ainsi, j'ai installé **Node.js 16.0.4** pour stabiliser mon environnement de travail, *Node* embarque *npm* qui me sera très utile par la suite. J'utiliserais **SublimeText 3** comme éditeur de texte pour développer mon code, dans lequel j'ai importé les packages React, ReactJS et Babel. J'ai choisi SublimeText pour sa rapidité même avec des gros fichiers et sa légèreté d'installation.

Dans *npm*, j'ai installé le package `creat-react-app` (`npm install -g create-react-app`), qui aide à démarrer un projet ReactJS en créant le dossier d'application et tous les éléments de départ nécessaire à celle-ci. Ainsi, après avoir installé le package, j'ai lancé la commande `create-react-app mytodolist` qui a créé mon application **MyTodoList**. Il suffit de lancer ensuite `npm start` pour accéder au serveur local (`http://localhost :3000`). Je m'attacherais ainsi à éditer le fichier `src/App.js` pour réaliser mon application. (ReactJs with create-react-app and Sublime Text, s.d.).

Langage de développement et Packages

ReactJS comme toute librairie Javascript ne peut fonctionner totalement seule. Aussi, je développerais une partie de mon application en **HTML**, langage composé de balises, permettant de créer des pages web statiques, il permet ainsi de décrire la structure et la

représentation de documents hypertextes. Associé au ReactJS et à l'HTML, j'utiliserais également le **CSS**, ces feuilles de styles en cascade (comme indiquent leur nom) permettent de mettre en forme visuellement les pages HTML.

Ainsi, pour disposer d'une interface graphique facile d'utilisation et agréable à l'œil, j'ai choisi d'utiliser **Bootstrap**. Ce framework JS et CSS, conçu par des développeurs de Twitter, est une mine d'or de collection d'outils préconçus pour la création de design web. Une version spécifique de Bootstrap dédiée à ReactJS a été développée, n'embarquant pas jQuery. C'est cette version que j'ai choisi d'utiliser.

J'ai utilisé `npm` pour l'installer (commande : `npm install react-bootstrap bootstrap@4.6.0`). Dans mes fichiers JS, il me suffira par la suite d'importer simplement chaque composant qui me sera nécessaire de la façon suivante : `import Button from 'react-bootstrap/Button';` (React-Bootstrap installation, s.d.).

Pour faire tourner le projet en local, voici la liste des packages installés :

```
- react 17.0.2,  
- react-dom 17.0.2,  
- react-scripts 4.0.3,  
- react-router-dom 5.2.0  
- react-bootstrap 1.6.1,  
- bootstrap 4.6.0,  
- cors 2.8.5,  
- express 4.17.1  
- nano 9.0.3  
- nanoid 3.1.23  
- emailjs-com 3.1.0
```

Méthodologie de développement

Afin de répondre au mieux aux exigences attendues du projet, j'ai fait le choix d'exécuter et de réadapter plusieurs tutoriels trouvés sur des sites de référence concernant ReactJS. J'ai ainsi pu rapidement apprendre à maîtriser les spécificités du langage tout en composant l'application la plus proche du fonctionnement recherché. Je ne manquerais pas de détailler les sites qui m'ont servi de source d'apprentissage et de développer les éléments que j'ai adaptés ou moi-même créés.

Dans un premier temps, j'ai analysé au mieux le besoin de mon « client ». Ce que je détaillerais dans la prochaine partie Puis, je me suis interrogé sur un modèle de données et sur l'interface graphique à envisager.

Dans la suite de ma démarche, j'ai choisi d'aller par étape pour développer mon produit. Ainsi, certaines parties peuvent être visibles dans la version du projet remise et pourtant non encore créées. Malgré tout, je compte achever ce projet même après la remise du présent rapport car le sujet m'a passionné et je pense pouvoir ainsi encore progresser dans mes apprentissages en vue de mon Master l'an prochain.

DESCRIPTIF DU PROJET

La présente description de notre produit est basée sur les consignes de mon « commanditaire », Monsieur François Delbot.

Langage de développement imposé : ReactJS

Objectif principal : Permettre à des utilisateurs connectés de gérer des listes de tâches.

Description des Objets :

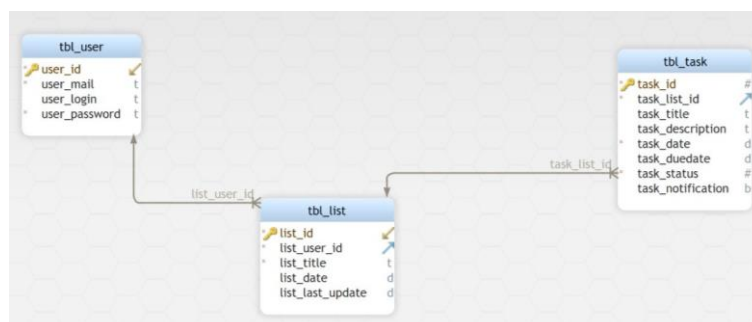
- Objet « **Liste de tâches** » : Une liste de tâches est un groupe d'objets « tâches » associée à un titre et une description.
- Objet « **Tâche** » : Une tâche est composée d'un titre, une description (texte court), une date de création, une date d'échéance, un statut (à faire, en cours, fait), une possibilité de faire un rappel par mail.
- Objet « **Utilisateur** » : un utilisateur est un compte d'un individu pour se connecter à l'application. Il est composé d'un login, mot de passe, et adresse mail. Un utilisateur pourra créer une ou plusieurs listes, elles-mêmes composées d'une ou plusieurs tâches.

Description des Fonctionnalités :

- **Accueil/Login/Création de compte** : Un utilisateur qui lance l'application arrivera sur la page de connexion. Si il n'a pas de compte, l'utilisateur pourra se créer un compte.
- **Page de listes** : Une fois connecté, l'utilisateur accèdera à une page de listes. Si il n'a pas de liste, il aura un bouton de création de liste. Il pourra ouvrir ses listes pour accéder aux tâches de la liste. Il pourra supprimer une liste, ou la renommer.
- **Page de tâches** : lorsqu'un utilisateur crée une liste ou en sélectionne une existante, il accède à la page de gestion des tâches. Il peut ajouter ou supprimer des tâches. Il peut également renommer une tâche, et modifier son statut.
- **Notification par mail** : l'utilisateur pourra choisir d'être notifié par mail pour lui rappeler l'échéance d'une tâche.

MODELE DE DONNEES

Concernant la gestion des données dans l'application, je me suis beaucoup interrogé. Initialement, j'ai pensé le modèle de données comme proposé ci-dessous, avec dans l'idée d'utiliser une base de données serveur. Cependant, suivant les divers tutoriels trouvés sur internet, les préconisations étaient variées, aussi j'ai choisi d'utiliser les éléments « locaux » du navigateur via le localStorage entre autre pour pouvoir avancer sur la question du fonctionnement lui-même de l'application.



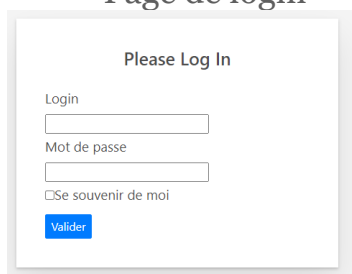
J'ai bien conscience qu'un tel choix, à cet étape du projet rend compliqué l'usage pour un utilisateur, puisqu'en quittant son navigateur il perd les données qu'il a créé. Mais, même si le délai qui m'était imparti ne m'a pas permis de mener cela vers une version optimal en termes de fonctionnement, je compte perfectionner cette question par la suite, en adossant cela à une base de données.

INTERFACE GRAPHIQUE

Comme précisé plus tôt, j'ai utilisé Bootstrap qui offre de nombreux modules pré-construits et facilite l'intégration graphique. Il offre aussi un avantage non négligeable, sa capacité à être responsive-design et donc à s'adapter au device qui l'affiche. Ainsi, qu'on soit sur mobile, tablette ou pc on aura toujours un affichage optimal de notre application adaptée à l'écran qui l'affiche.

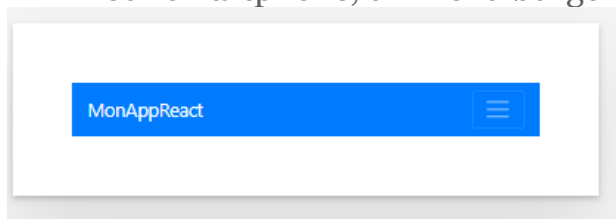
Les différentes vues à prévoir associées aux captures du résultat final :

- Page de login

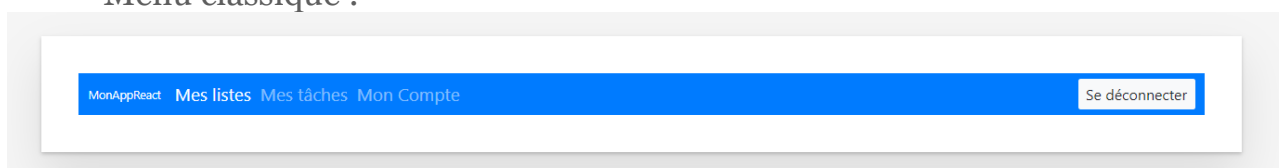
A screenshot of a login form titled "Please Log In". It contains two input fields labeled "Login" and "Mot de passe", a checkbox labeled "Se souvenir de moi", and a blue button labeled "Valider".

- Page de Création de compte (à faire)

- Sur smartphone, un menu burger s'affiche (plutôt que le menu classique) :



- Menu classique :



- Page de listes de l'utilisateur :

Mes listes

Voulez-vous ajouter une liste de tâches ?
Pour créer une nouvelle liste, saisissez un titre, une description et choisissez une couleur :

Titre

Description

Couleur de la liste

Importer une image

Aucun fichier choisi

286x180

Le titre de ma liste
Une courte description de ma liste de tâches LOL

286x180

Le titre de ma liste2
Une courte description de ma liste de tâches LOL

286x180

Le titre de ma liste3
Une courte description de ma liste de tâches LOL

286x180

Le titre de ma liste4
Une courte description de ma liste de tâches LOL

- Page d'une liste et de ses taches associées

MonAppReact Mes listes Mes tâches Mon Compte

Se déconnecter

MyToDoList

Voulez-vous ajouter une tâche ?
Pour créer une nouvelle tâche, saisissez un nom :

Nom de la tâche

Toutes

Actives

Finies

5 tâches restantes

☒ Manger
list-0

☐ Dormir
list-1

- Mon compte (à faire)

DESCRIPTION DU PRODUIT FINAL

Dans cette partie, je détaillerais plus précisément chaque module de l'application, je citerais mes éventuelles sources et je présenterais le fonctionnement de celle-ci. J'ai choisi de détailler chaque module plutôt dans l'ordre dans lequel je les ai développés, ce qui peut expliquer parfois les impacts sur les autres parties.

Module Login

Le premier module auquel l'utilisateur est confronté en ouvrant l'application est le module de connexion. Ce module est une mire de connexion, où l'utilisateur doit saisir son login (son adresse mail) et son mot de passe, il peut également cocher une checkbox pour que l'application se connecte automatiquement.

Pour créer ce module, je me suis inspiré d'un tutoriel proposé sur le site DigitalOcean.com (Morgan, 2020). Le module fait appel au package `react-router-dom` qui permet de créer plusieurs « chemins » dans une même page en fonction de certains paramètres.

Ainsi, dans le cas où l'utilisateur n'est pas connecté, ce que l'on vérifie à l'aide d'un token, il affiche le contenu de `Login()` (fonction de `Login.js`). Ce module nécessite le démarrage d'un serveur Node pour fonctionner.

L'utilisateur aura aussi, dans un second temps, la possibilité de créer un compte ou de redemander son mot de passe (non fait).

Module Liste de tâches

Le deuxième module que j'ai développé est destiné à la gestion des tâches. Pour ce module, je me suis librement inspiré du tutoriel ToDoMatic (Débuter notre React todo list, s.d.), que j'ai en partie réadapté pour mon contexte.

Ainsi, j'ai créé un composant appelé `<Todo>` (`components\todo.js`) qui permet de gérer la vue ou l'édition d'une tâche. Cette balise une fois créée peut ainsi être appelée pour plusieurs objets du même type dont il reçoit les éléments en argument. Afin de gérer l'édition et l'affichage d'une tâche dans le même document, j'ai utilisé les éléments `{ useEffect, useRef, useState }` de ReactJS, en effet `useEffect` me permet de connaître si un élément est en cours d'édition ou non, `useRef` me permet de récupérer la référence de l'objet en cours de modification et `useState` me permet d'affecter les valeurs aux objets. Cela me permettra notamment d'afficher des boutons d'actions différents selon le mode retenu (Affichage ou Edition). Cliquer sur le bouton Modifier déclenche le mode Edition, les boutons d'action (Annuler / Enregistrer à la place de Modifier / Supprimer) changent et le titre devient modifiable.

Ce composant `<Todo>` est appelé au niveau du fichier `MyTodoListApp.js` qui détaille le composant `<MyTodoListApp tasks={props.tasks}/>`. Dans cette partie, on détaille les fonctions de la liste de tâches : `toggleTaskCompleted(id)`, `deleteTask(id)`, `editTask(id, newName)`, `addTask(name)`. Ces 4 fonctions permettent d'ajouter, supprimer ou éditer des tâches, ou bien de choisir d'afficher les tâches en fonction de leur état (Active ou Terminée). On utilise par ailleurs pour gérer l'affichage de ces tâches, stockées sous forme de tableau, la fonction `array.filter` qui permet de filtrer sur un critère précis. Il est important de noter qu'on utilisera également `array.map` pour mapper

justement les données d'une tâche stockés dans le tableau de données (ou d'une liste de tâches) avec leur affichage dans la page.

Détail des fonctions *MyTodoListApp.js*

```
function toggleTaskCompleted(id) {
  const updatedTasks = tasks.map(task => {
    if (id === task.id) {
      return {...task, completed: !task.completed}
    }
    return task;
  });
  setTasks(updatedTasks);
}
```

Dans **toggleTaskCompleted(id)**, on filtre les tâches en fonction de leur état. Cependant on récupère l'id de la tâche éventuellement en cours de modification pour ne pas la perdre et ne pas changer garder son statut à 'Terminé'.

```
function deleteTask(id) {
  const remainingTasks = tasks.filter(task => id !== task.id);
  setTasks(remainingTasks);
}
```

Dans **deleteTask(id)**, on filtre les tâches du tableau en supprimant la tâche ayant le même id en argument.

```
function editTask(id, newName) {
  const editedTaskList = tasks.map(task => {
    // if this task has the same ID as the edited task
    if (id === task.id) {
      //
      return {...task, name: newName}
    }
    return task;
  });
  setTasks(editedTaskList);
}
```

Dans **editTask(id, newName)**, on filtre les tâches pour trouver celle ayant le même id que reçu en argument. Puis on « remet ensemble » la tâche modifiée avec le reste du groupe.

```
function addTask(name) {
  const newTask = { id: "todo-" + nanoid(), name: name, completed: false };
  setTasks([...tasks, newTask]);
}
```

Dans **addTask(name)**, on crée une nouvelle tâche. On compose l'id à l'aide de nanoid qui incrémente l'id à partir des id des autres éléments, puis on l'ajoute à notre tableau de données. La fonction addTask est appelée dans le formulaire dédié à la création de tâches, module **<FormTask>**.

Cette dernière fonctionnalité est incomplète puisque nous ne pouvons pas pour le moment saisir une date d'échéance, ou plusieurs types de statuts. Cela fait partie des prochaines features à venir.

Menu de Navigation et Gestion des espaces

Dans cette partie, je vais détailler deux éléments : le menu de navigation, mais dans un premier temps, je m'attacherai à présenter le choix fait pour définir ces différents espaces du site.

Lorsqu'on ouvre le fichier App.js qui est le cœur de mon application, on constate que j'ai importé { BrowserRouter, Route, Switch } de react-router-dom, afin de pouvoir gérer dans un même fichier l'affichage de composants différents. Aussi, mon site est composé de 3 espaces à ce jour : Dashboard (qui permet d'afficher les tâches d'une liste), TaskList (que je détaillerai juste après) et Préférence, qui vise à gérer le compte de l'utilisateur (non fait). Ainsi en cas de token présent, l'utilisateur sera dirigé vers la page correspondante à sa demande (par défaut à la connexion vers la liste des listes).

Concernant le menu de navigation, il s'agit ici d'un import de Bootstrap. L'avantage est que le module est pré-conçu, il m'a suffi de l'adapter à mon contexte et de variabiliser les éléments pour simplifier la mise à jour. Autre élément considérable ce menu est responsive-design ainsi il change en fonction de la taille de l'écran qui l'affiche. J'ai ajouté à ce module préconçu la notion de lien actif, qui met en gras l'endroit où on se situe dans l'application.

Module Liste de listes

Ce module permet d'afficher les « cartes » de chaque liste créée par l'utilisateur connecté. Le concept de « carte » est issu là encore de Bootstrap, il rend la page visuellement agréable, et un peu à l'image de Trello, qui pourrait être une application dans la même veine de ce que je tente de créer, les tableaux de tâches de Trello sont symbolisés par des cartes dans leur fonctionnement.

En termes de fonctionnement et de développement, je me suis largement inspiré du fonctionnement déjà acquis sur la partie liste de tâches. Ainsi, le module se présente en 2 parties : dans la partie supérieure, un formulaire permet de créer une nouvelle liste, en la nommant, lui donnant une description et ajoutant une couleur voire une image. La seconde partie du module, liste toutes les listes de l'utilisateur. Il peut ensuite cliquer sur l'une d'elles pour afficher les tâches rattachées à cette liste.

```
function viewList(listid) {  
  localStorage.setItem('listid', listid);  
  window.location.href = '/dashboard';  
}
```

Lorsqu'on clique sur le bouton d'une liste on déclenche la fonction viewList, qui récupère l'id et la stocke dans le localStorage pour la réutiliser ensuite, puis on redirige vers le dashboard.

Afin de ne pas encombrer l'affichage à l'œil, j'ai prévu qu'on ne puisse pas afficher plus de 4 listes par ligne. Aussi j'ai conçu la fonctionnalité suivante qui me permet de réduire mon tableau d'objets, à un tableau de tableaux d'objet :

```
const rowsoflist = lists.reduce(function (rows, key, index) {  
  return (index % 4 === 0 ? rows.push([key])  
    : rows[rows.length-1].push(key)) && rows;  
}, []);
```

Notification par mail

La notification par mail était une demande de mon client. Cependant à ce jour je n'ai pas trouvé le temps de pouvoir l'implémenter et j'ai eu des difficultés à faire fonctionner les packages de routage de mail.

J'ai notamment essayé d'utiliser (entre autre dans mon premier développement sur codepen.io) emailJS, sans grand succès jusqu'ici, mais j'espère trouver prochainement un solution.

LIMITES DU PRODUIT

En l'état, mon produit demande encore quelques améliorations importantes.

D'un point de vue technique, il me manque un certain nombre d'éléments que je manquerais pas de creuser même après la remise de ce rapport, A savoir :

- La liaison entre les listes et les tâches, le filtre que j'ai tenté de manier pour ce faire ne semble pas fonctionner
- La gestion du compte utilisateur et la création de compte (qui sont liés)
- L'utilisation du vraie base de données pour permettre la persistance de celle-ci
- L'ajout d'éléments demandés sur les tâches (date, rappel, statut, etc.)
- Les notifications par email, notamment avec rappel des tâches urgentes

D'un point de vue fonctionnel, le produit pour être vraiment utile et user-centric devrait comprendre également quelques éléments qui pourraient être ajoutés :

- Design :
 - o Un fonctionnement plus User-friendly avec un Drag&Drop des tâches en fonction de leur statut
 - o Une page d'accueil plus professionnelle présentant l'application et invitant à s'inscrire
- Connectivité & Collaboration
 - o La possibilité d'ajouter les tâches à son calendrier
 - o Le partage des listes de tâches avec d'autres utilisateurs pour favoriser la collaboration
- Gestion des tâches :
 - o Ajouter des catégories sur les tâches pour pouvoir les classer
 - o Pouvoir uploader des pièces-jointes sur les tâches
 - o Pouvoir créer des checklist dans les tâches si celle-ci se réalise en plusieurs étapes.
 - o Pouvoir faire vivre la tâche en la commentant par exemple

C'est une liste d'idées non-exhaustive bien entendu, dont je m'inspirerais peut-être pendant l'été pour prolonger ce travail.

REGARD SUR LE TRAVAIL EFFECTUE

Bilan

Comme pour le projet Graphe & Open Data, je pourrais dire que réaliser un tel projet seul n'a pas été d'une grande facilité. En effet, l'année en distanciel a rendu les apprentissages plus difficiles. La difficulté supplémentaire ici consistait à apprendre seul un nouveau langage, en maîtriser ses rudiments, pour répondre au mieux à la demande initiale. J'ai dû déployer beaucoup d'abnégation et de motivation pour m'investir dans ce projet, surtout quand on a un emploi étudiant prenant en parallèle et des examens à passer.

Cependant, réaliser ce projet sur un nouveau langage m'a donné l'occasion d'ajouter une corde à mon arc de développeur en construction. Et je réalise que peut-être en stage je n'aurais pas eu autant d'occasions d'accroître mes compétences techniques. Cela a en tout cas confirmé, mon goût pour le développement et la recherche d'algorithmes toujours plus efficaces, tout en réalisant des choses directement visuelles.

Je l'ai plusieurs fois écrit tout au long de ce rapport, pour moi ce projet n'est pas achevé, et j'ai à cœur d'aller au bout pour continuer d'apprendre et mettre à profit mon été, à défaut justement d'avoir eu un stage jusqu'à la rentrée. Malgré tout, je suis plutôt fier du résultat proposé, même inachevé, avec l'étendue de mes connaissances en React en début de projet (quasi proche de zéro), je peux avoir la fierté d'avoir réalisé un produit fonctionnel, visuellement agréable, et pour lequel j'ai pensé à des améliorations futures. Je crois que c'est le propre de tout développeur professionnel : ne pas s'arrêter à la ligne de code une fois qu'elle est en production mais toujours penser à ce qu'elle pourrait devenir demain...

Difficultés et problématiques rencontrées

Concernant les difficultés rencontrées, j'en dénombre au moins 4.

Tout d'abord, ma disponibilité et ma communication difficile avec mon commanditaire. Mon emploi étudiant a pris beaucoup de place dans mon quotidien et je me suis souvent retrouvé indisponible lors des sessions prévues à destination des étudiants sans stage. La communication au sein du groupe de Licence n'a pas toujours été optimale et les informations ne me sont pas toujours parvenues. Ce faisant j'ai commencé mon projet tardivement, puis j'ai eu du mal à trouver le bon horaire et le bon canal de communication pour échanger sur le projet. Cependant, je n'ai pas baissé les bras et je remercie Monsieur Delbot de m'avoir permis de réaliser un tel projet porteur pour ma carrière future.

Ensuite, bien sûr je l'ai déjà évoqué, ma méconnaissance du langage principal de l'application a été une source d'inquiétudes forte. Pourrais-je acquérir les bases du langage et réaliser un projet à la hauteur des attentes. J'ai la fierté de penser qu'aujourd'hui cette difficulté est devenu un point fort sur mon Curriculum Vitae.

De plus, viennent avec cela les aspects techniques. Initialement j'avais choisi codepen.io parce que cela me semblait plus simple, mais j'ai vite réalisé que pour un projet conséquent comme le mien il fallait envisager un projet local. Ne pas savoir comment gérer les données, comment router des mails avec ce langage spécifique, ne pas connaître non plus node mais devoir l'utiliser ont été des éléments qui m'ont ralenti dans ma production concrète de résultat. Malgré tout, j'ai là aussi conscience que ce sont autant d'acquis que je pourrais mettre à profit dans les années à venir.

Dernière difficulté et non des moindres, il a fallu réaliser l'ensemble de ce travail en parallèle des sessions de rattrapages des examens de l'année, et je remercie à nouveau Monsieur Delbot pour le délai de 24h supplémentaire qu'il m'a accordé pour perfectionner ce projet et ce rapport.

REMERCIEMENTS

Au terme de ce travail de fin d'études et de cette dernière année de licence MIAGE, c'est un devoir agréable d'exprimer en quelques lignes la reconnaissance que je dois à toutes celles et tous ceux qui ont contribué de près ou de loin à l'élaboration de ce travail et à mon parcours universitaire au sein de l'Université Paris-Nanterre. Ainsi qu'ils se voient ici recevoir mon vif respect et ma profonde gratitude quant à ce que chacune et chacun a pu me transmettre.

En premier lieu, donc, je tiens à exprimer toute ma sympathie à l'ensemble des enseignant.e.s et encadrant.e.s qui ont contribué à l'ensemble de mon parcours universitaire au sein de l'Université Paris-Nanterre .

Je tiens à remercier tout particulièrement mon encadrant pour ce projet, Monsieur François DELBOT, le temps qu'il m'a consacré et ses précieux conseils et encouragements, tout au long de l'année, ont été un élément de motivation pour maintenir mon engagement malgré le contexte particulier de cet année universitaire.

Je salue chaleureusement également mes camarades de promotion qui ont toujours su répondre présents pour partager leurs connaissances et leurs découvertes, le contexte de l'année nous aura, en tout cas, appris à être solidaires et collectifs.

Je ne laisserais pas passer cette occasion sans remercier tous les nombreux "anonymes" qui partagent sur Internet leurs connaissances et leurs façons diverses et variées de coder, qui m'ont été de précieuses sources pour mieux intégrer et comprendre comment donner vie à ce projet.

Enfin, je tiens à remercier toutes les personnes qui ont été à mes côtés, dans ce parcours, mes proches, ma famille, mes amis, qui ont su maintenir ma motivation et m'encourager à donner le meilleur de moi-même pour réussir cette année et ce projet. Merci à elles et eux d'avoir toujours été présents.

En conclusion, je peux me féliciter d'avoir su aller vers l'objectif que je m'étais fixé même si il n'est que partiellement atteint.

Conclusion

Ce projet m'a permis d'acquérir de nouvelles compétences (des soft skills qui valoriseront sans nul doute mon CV), de découvrir un nouveau langage (une nouvelle hard skill à mon actif), mais aussi de renforcer mes fondations techniques en HTML/CSS/JS.

Il m'a permis également d'imaginer et de mettre en œuvre toute la partie qui entoure le développement réel d'un produit. Que ce soit en amont avec la définition du produit, la création de maquettes, la réflexion à mener sur les données, etc ou en aval avec le packaging d'un projet.

Mais surtout ce projet m'a donné l'opportunité de pousser plus loin que l'édition de simples lignes de codes. J'ai eu à réfléchir et penser mon produit dans sa globalité, un peu à la manière d'un Product Owner qui le ferait pour son équipe Scrum et ses clients. C'est assurément parce que je me suis totalement investi dans ce projet et ses implications que j'ai envie d'aller au bout de ce que j'ai commencé.

*Ainsi, j'espère publier sur mon github régulièrement les avancées dans ce projet qui n'est que le début de **MyToDoList**. Stay tuned !*

Bibliographie & Webographie

(s.d.). Récupéré sur React-Bootstrap installation: <https://react-bootstrap.github.io/getting-started/introduction>

Bouguerra, A. (s.d.). *Module Login*. Récupéré sur CodePen.io: <https://codepen.io/aimedbouguerra/pen/qBRJRZz>

Débuter notre React todo list. (s.d.). Récupéré sur https://developer.mozilla.org/fr/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/React_todo_list_beginning

Morgan, J. (2020, Décembre 2). *How To Add Login Authentication to React Applications*. Récupéré sur DigitalOcean: <https://www.digitalocean.com/community/tutorials/how-to-add-login-authentication-to-react-applications>

ReactJs with create-react-app and Sublime Text. (s.d.). Récupéré sur <https://medium.com/web-tutorials-club/reactjs-with-create-react-app-and-sublime-text-984e7fb46455>

Documents joints au projet :

- Le présent rapport : RAPPORT_PROJET_REACT_AimedBouguerra.docx
- Le dossier **build** du projet MyToDoList : build.zip
- Le dossier **source** du projet MyToDoList : src.zip
- Le dossier **public** du projet MyToDoList : public.zip
- Le fichier listant les packages installés : package.json

**Ayez le courage de suivre
votre cœur et votre
intuition. L'un et l'autre
savent ce que vous voulez
réellement devenir. Le
reste est secondaire.**

Steve Jobs