

Rapport de création de IntroduceYourEsiearque

Projet de 4ème Année à l'ESIEA - École d'Ingénieurs du Monde Numérique

Élèves : Adel ENKIRCHE et Jon CHERON
Professeur : Louis CHEREL | Année scolaire : 2019/2020

URL du site - <http://54.38.33.63:8080>

URL du site - http://54.38.33.63:8080	1
Introduction	2
Technologies utilisées	2
Présentation du projet	2
Phase de Conception	3
Modulabilité	3
Identifiants et données	3
Plan fonctionnel de l'application	3
Phase de Développement	5
Utilisation de l'application	5
Résultat côté back-end	6
Difficultés rencontrées	7
Phase de déploiement	8

Introduction

Cette application a été développée durant le cours de Monsieur Louis Cherel à l'ESIEA : elle permet aux étudiants de l'ESIEA d'avoir une plateforme sur laquelle se présenter avec leur âge, leur prénom ainsi que leur expérience scolaire pré-ESIEA.

Technologies utilisées

Front-End : VueJS

Back-End: Node.JS

Quelques modules complémentaires utilisés : Axios, Vuex, Vuetify, Express

Présentation du projet

Ce projet nous a été donné afin de nous permettre de maîtriser un framework moderne et son utilisation tel que VueJS avec du CRUD (Create Read Update Delete) effectué entre le back-end et le front-end du projet.

Nous verrons dans ce rapport que ces objectifs ont été atteint par notre projet avec succès.

Phase de Conception

Notre objectif au cours de ce projet est de faire un véritable travail d'ingénieur et donc de faire plus que seulement du développement, des critères de qualité logicielle doivent être intégrés dans notre application.

Modulabilité

Un critère important de qualité logicielle est la modulabilité : le fait de ne pas avoir une application composée d'un seul bloc permet une meilleure compréhension du code par un utilisateur qui serait amené à reprendre le code.

Dans cette optique, nos blocs front et back ont été séparés : la partie back-end de notre application se trouve à la racine du dossier contenant le projet, c'est un bloc indépendant pouvant être démarré sans l'utilisation de la partie front-end du projet, il est composé de son fichier `package.json` qui contient uniquement les dépendances de la partie serveur.

De même, la partie client de l'application dit *front-end* est entièrement présente dans le dossier *client* à la racine du dossier contenant le projet et peut être exécutée indépendamment de la partie back-end du projet.

Identifiants et données

Le professeur qui gère ce projet nous a dit que l'utilisation d'une base de données n'était pas nécessaire ici.

Ainsi nous n'utilisons pas ici de base de données, nous avons donc fait le choix pour garder les données des étudiants de les stocker dans un simple fichier JSON : il stocke les identifiants de connexion dans un premier tableau *users* et les données relatives aux différents utilisateurs dans un objet *usersData*.

Encore une fois, la modulabilité est importante, les données n'ont pas été intégrées au format JSON dans le fichier `server.js` mais dans un fichier JSON à part.

Plan fonctionnel de l'application

Notre idée au cours du projet est de permettre à l'utilisateur final d'avoir une interface simple d'utilisation et d'avoir quelque chose de très intuitif.

L'utilisateur doit pouvoir se présenter en quelques clics et très rapidement après s'être

connecté.

De même, voir les présentations des autres élèves doit être très facile et intuitif.

La présentation par *cards* (voir [figure 1](#)) semble bien convenir à notre idée d'intuitivité du résultat, l'organisation 1 carte = 1 élève est un choix qui nous plaît.

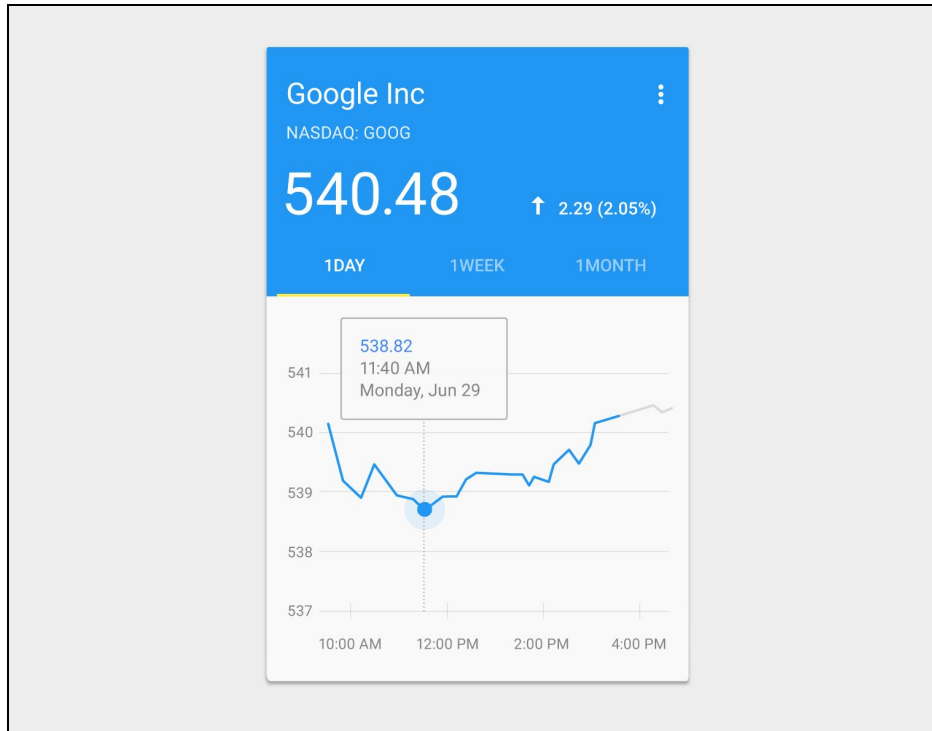


Figure 1 : card material design (source: google.com)

Phase de Développement

Utilisation de l'application

Lorsque l'utilisateur accède à l'URL du projet, une redirection est effectuée vers la page de connexion (URL + */login*), voir [figure 2](#).

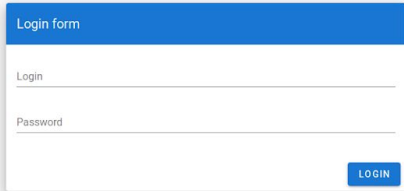
A screenshot of a web application's login form. The form is a white rectangular box with a blue header bar at the top containing the text "Login form". Below the header, there are two input fields: the first is labeled "Login" and the second is labeled "Password". Both labels are in a small, light gray font. To the right of the "Password" input field, there is a blue button with the word "LOGIN" in white capital letters. The entire form is centered on a light gray background.

Figure 2 : formulaire de connexion

Une fois connecté, l'utilisateur est redirigé vers la page d'accueil (URL + */home*), voir [figure 3](#).

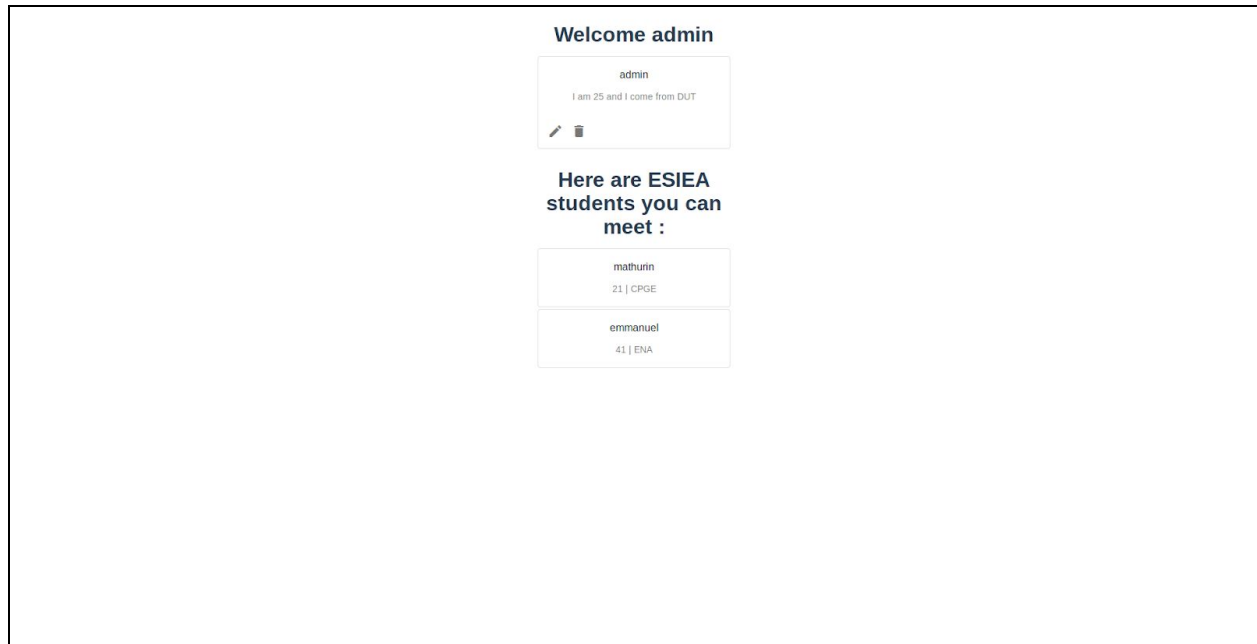


Figure 3 : page d'accueil de notre site web

Au-dessus de la page d'accueil se trouve la carte de présentation de l'étudiant connecté, qu'il peut modifier ou supprimer via les deux icônes présentes dans sa carte.

En dessous de sa carte se trouvent les cartes de présentation de tous les autres utilisateurs présents dans l'objet *usersData* dans le fichier *data.json* côté back-end.

Si l'utilisateur connecté supprime sa carte ou la modifie, il reçoit une confirmation que l'action qu'il a faite a bien été exécutée côté back via une alert Vuetify.

Si l'utilisateur connecté n'a jamais créé de carte de présentation, sa carte ne s'affichera pas et un formulaire lui permettant de créer sa carte sera affiché.

Résultat côté back-end

La connection s'effectue par sessions, comme demandé par le professeur gérant le projet : nous sommes partie de la base proposée par le professeur sur son compte GitHub.

Un utilisateur souhaitant accéder à notre page d'accueil via l'URL de la page d'accueil sans s'être connecté au préalable ne pourra pas le faire : la récupération des informations qui servent à l'affichage des cartes se fait durant l'authentification, ainsi, un aspect sécuritaire a été pris en

compte ici.

Difficultés rencontrées

Aucune difficulté majeure n'a été rencontrée lors du développement de notre application : chacun des membres de notre groupe ayant déjà travaillé par le passé sur des projets impliquant des technologies webs modernes, il a surtout été question de s'adapter à VueJS, le gros de notre travail ayant été fait jusqu'ici sur Angular ou React.

Phase de déploiement

Notre déploiement n'a pas fonctionné sur la plateforme Glitch comme recommandé par le professeur : la limite d'utilisation de mémoire vive par un compte gratuit Glitch ayant été atteinte rapidement lors de l'exécution de nos scripts.

Afin de déployer notre application, nous avons fait appel au serveur web d'un ami, nous avons eu à supprimer une simple ligne de code afin de faire fonctionner le tout : une ligne de paramétrage d'Axios qui provoquait un problème de requêtes multi-origines.

Voilà la ligne concernée :

```
axios.defaults.withCredentials = true
```