POLYTECH NANCY

Projet Interne 5A

Boutique de thé en ligne

Année : 5ème année

Auteur : Zixiao Zhong, SIR

Tuteur : Mario Valencia

UNIVERSITÉ DE LORRAINE

LORRAINE INP

Date du document

Sommaire

INTRODUCTION

Avec le développement de la technologie et les progrès de la société, Internet est devenu une partie intégrante de la vie, et les applications web/mobile jouent un rôle très important. Avec l'aide de diverses applications, les gens peuvent effectuer diverses activités, achat en ligne etc. Comme le potentiel et marché sont assez grands, je souhaite m'orienter dans le domaine, du développement, notamment full-stack. Donc dans cette année, avec l'aide de M.Valencia, je choisis de faire un projet interne.

Ce projet est un projet full-stack qui est une application web uniquement pour mobile. L'application est une boutique en ligne, mais il ne vend que des thés chinois. Bien que maintenant il y a déjà grandes entreprises, tels que Amazon, eBay, dans le domaine achat en ligne, l'achat en ligne a encore un grand potentiel. Cette application est développée pour le marché chinois.

L'application a les fonctionnalités principales de boutique en ligne :

- Inscription de client
- Login avec le numéro de mobile et le mot de passe
- Rechercher les produits
- Ajouter les produits au panier
- Gérer le lieu de livraison
- Payement
- Regarder les commandes

Je commencerai tout d'abord par présenter les outils utilises, les IDEs et les logiciels, et les technologies implémentes pendant ce développement, les langages de programmation, les frameworks et la base de données.

Ensuite, j'expliquerai précisément le travail que j'ai effectué, l'architecture de l'application et l'architecture à mettre en place. De plus, je ferai la description détaille du frontend et backend.

Enfin, je retracerai les difficultés rencontrées lors du développement et conclurai quels ont été les apports de cette expérience.

TECHNOLOGIES

I - IDE

WebStorm

WebStorm est un environnement de développement intégrédestiné au développement de logiciels informatiques reposant sur la technologie JavaScript et TypeScript. Il est développé par JetBrains, supporte les framworks principaux de frontend ReactJS, Angular, VueJS et le Framework backend de NodeJS, comme Express.

II - Logiciel

StarUML

StarUML est un outil de génie logiciel dédié à la modélisation UML et édité la société coréenne MKLabs.

WampServer

WampServer est une plateforme de développement Web de type WAMP. Il n'est pas en soi un logiciel, mais un environnement comprenant trois serveurs (Apache, MySQL et MariaDB), un interpréteur de script (PHP), ainsi que phpMyAdmin pour l'administration Web des bases MySQL. Dans ce projet, j'ai utilisé WampServer pour créer la base de données MySQL.

Navicat

Navicat est une suite logicielle graphique de gestion et de développement de bases de données produits par PremiumSoft CyberTech pour MySQL, MariaDB, Oracle, SQLite, PostgreSQL et Microsoft SQL Server. Dans ce projet, utiliser Navicat gère la base de données MySQL.

III - Langage

HTML

HyperText Markup Language est le langage de balisage conçu pour représenter les pages web. Il est utilisé pour créer la structure des pages. Dans ce projet, la version implémentée est HTML5.

CSS

Cascading Style Sheets est un langage de feuille de style utilisé pour décrire la présentation d'un document écrit dans un langage de balisage tel que HTML ou XML. Il est conçu pour permettre la séparation du contenu et de la présentation, y compris la mise en page, les couleurs et les polices.

JavaScript

JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives et à ce titre est une partie essentielle des applications web. Avec les langages HTML et CSS, JavaScript est au cœur des langages utilisés par les développeurs web. Dans ce projet, le frontend et le backend utilisent le langage JavaScript.

IV - Framework

Vue.js

Vue.js est un framework JavaScript open-source utilisé pour construire des interfaces utilisateur et des applications web monopages. Vue a été créé par Evan You et est maintenu par lui et le reste des membres actifs de l'équipe principale travaillant sur le projet et son écosystème. Vue a été publié pour la première fois en février 2014. Vue est actuellement l'un des trois frameworks frontend très populaire.

Node.js (Express)

Node.js est un environnement de serveur open source multiplateforme. Il permet aux développeurs d'utiliser JavaScript pour les scripts côté serveur. La fonctionnalité d'exécution de scripts côté serveur produit un contenu de page Web dynamique avant que la page ne soit envoyée au navigateur Web de l'utilisateur.

V - UI

Vant

Vant est une bibliothèque d'interface utilisateur Vue légère et personnalisable pour les applications Web mobiles. Il support vue2 et vue3, open-source en 2017. Il a ces composants comme le bouton, le formulaire, le calendrier, le toast, etc.

VI - Base de données

MySQL

MySQL est un système de gestion de bases de données relationnelles. Il est distribué sous une double licence GPL et propriétaire. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par des professionnels, en concurrence avec Oracle, PostgreSQL et Microsoft SQL Server.

ARCHITECTURE

I - Diagramme

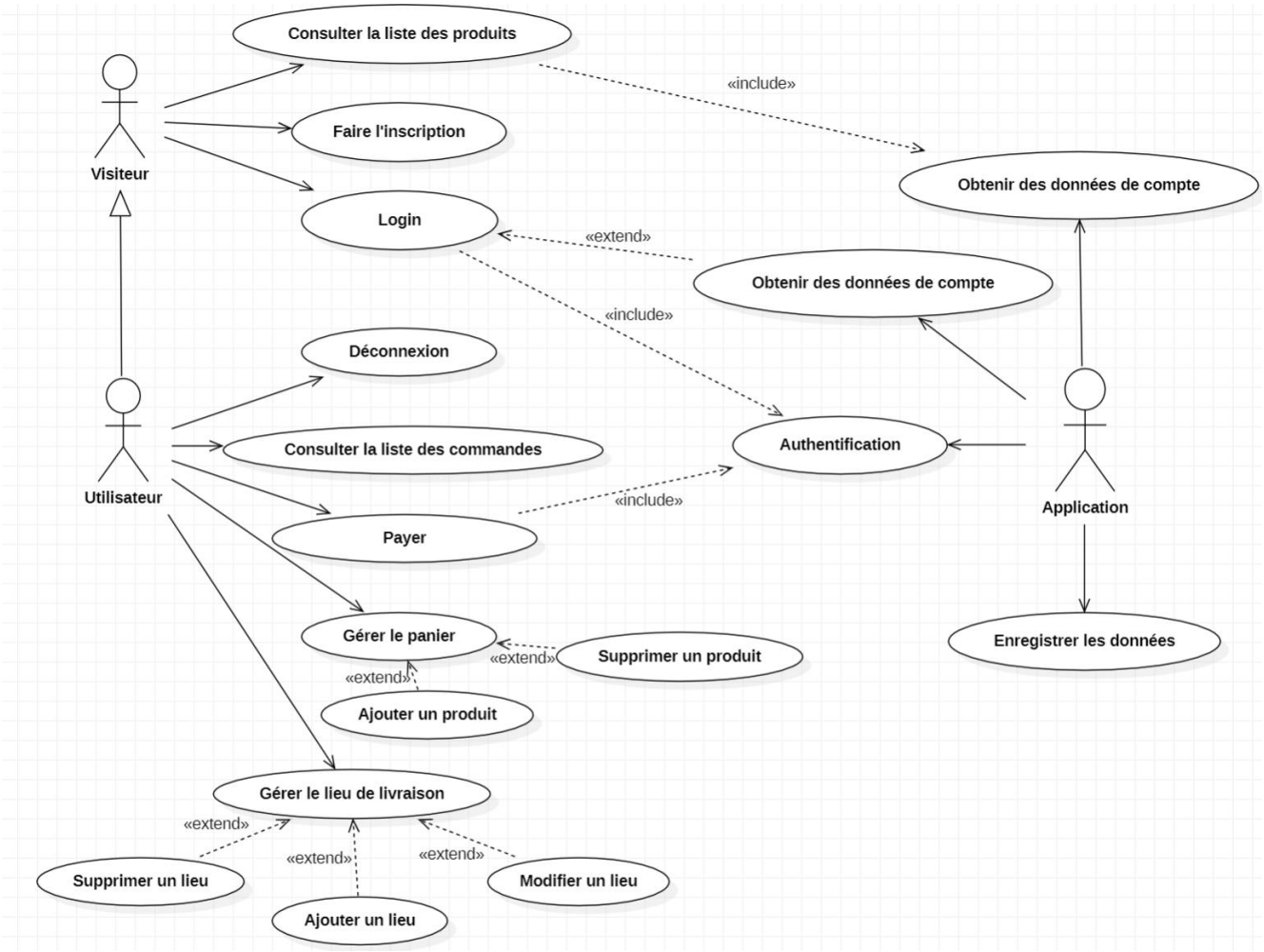
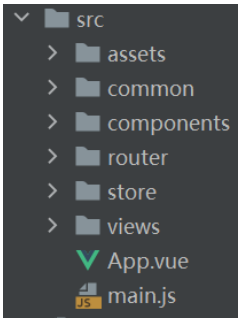


Figure 1 Diagramme de cas d'utilisation

II - Frontend

Le frontend est développé par le Framework Vue.js avec l'UI Vant. Pour réaliser cette application, il faut avoir les composants de Vue. Donc la prochaine étape, je vais expliquer l'architecture de projet Vue en détail et les dépendances principales.

II.a. - Architecture



- **views** : La partie views est utilisé pour stocker toutes les pages principales de l'application, l'accueil, le panier, le menu etc.

- **components** : Components est le dossier qui stocke les composants des pages principales. Par exemple la barre de recherche, car plusieurs pages ont la barre de recherche, on a juste besoin d'importer le

composant, pas besoin de créer la barre chaque fois. Sauf que les composants publics, il y a des composants spéciaux pour les pages.

- **router** : Router est la routage du projet, permettant aux développeurs d'écrire des applications d'une seule page qui basculent entre plusieurs vues. Pour réaliser le routage, installer la dépendance 'vue-router' est nécessaire. Chaque page dans le dossier views a une route unique.

- **store** : Une bibliothèque de gestion d'état pour Vue.js. Si une donnée est utilisée par diverses pages, la meilleure résolution est qu'enregistrer la donnée dans le store. Il y a deux types de bibliothèque, pinia et vuex. Dans ce projet, j'ai choisi le vuex.

II.b. - Dépendances

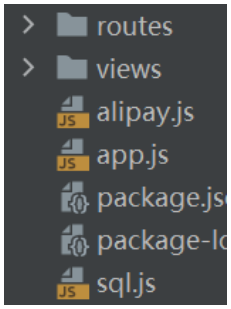
- **Axios** : Axios est une dépendance très importante. Il est un client HTTP basé sur des promesses pour node.js et le navigateur. Avec l'aide d'Axios, l'application peut envoyer les requêtes http au backend et obtenir les données retournées par backend.

- **node-sass/sass-loader/style-loader**: Les trois dépendances sont indispensables quand écrire la langage CSS en type SASS.

- **Vant** : La dépendance de l'UI de Vant.

III - Backend

Le backend est réalisé par node.js et le Framework Express. Parce que le frontend et le backend du projet sont indépendants. Les fonctions du backend sont que traiter les requetés et établir la connexion de base de données. Comparer avec le frontend, l'architecture est plus simple Architecture



- **routes** : Les APIs utilisés dans ce projet. Différente API réalise une fonction différente et renvoie les résultats correspondants au frontend.

- **sql.js** : Le fichier de configuration de la base de données.

- **alipay.js** : Le fichier de configuration de Alipay

III.a. - Dépendances

- jsonwebtoken : La dépendance utilisée pour créer token

- mysql : La dépendance de la base de données MySQL

- qcloudsms_js : Réaliser la fonction d'envoyer SMS, y compris le code de vérification. Mais dans ce projet il ne support que numéro tel de Chine.

DEVELOPPEMENT

I - Frontend

Tout d'abord, je présenterai quelques interfaces de l'application. Ensuite, Je vais expliquer comment je développe le frontend et réalise ces fonctions comme la connexion, le paiement, le panier dans cette partie. Car la limite de page, je vais présenter seulement les fonctions principales.

Figure 2 L'accueil

Figure 2 Le menu

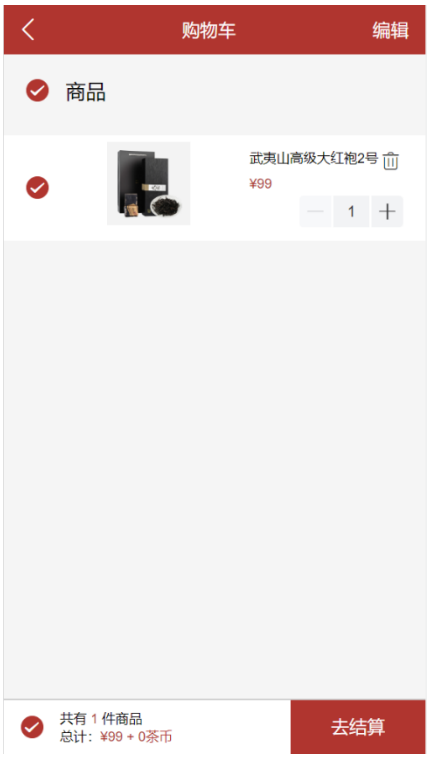


Figure 4 Le panier

Inscrire et Login

Login est la fonction plus importante dans ce projet. Parce que l'application a besoin de savoir quel client a acheté quel article, aussi chaque client a un lieu de livraison différent. Avant login, il faut avoir des données du client dans la base de données, sinon il faut faire l'inscription. Car la politique de Chine, maintenant toutes les applications en Chine nécessitent un numéro de mobile pour s'inscrire.

Figure 5 La formulaire d'inscription

Avant d'envoyer le code de vérification, le programme vérifiera d'abord le format du numéro de mobile avec l'aide de l'expression régulière. Si le format est correct, il envoie une requête http au backend, ensuite client reçoit un SMS, y compris le code. Après remplir ce formulaire et le code est correct, cliquer le bouton d'inscription, toutes les données sont enregistrées dans la base de données.

Pour login, il existe deux manières de se connecter, un est qu'utiliser le numéro tel et recevoir le code de vérification, l'autre est login avec le numéro tel et le mot de passe. Si connexion réussie, frontend reçoit un token créé par backend.

Rechercher

La recherche est une autre fonction très importante, notamment pour une boutique. Il peut y avoir des centaines ou des milliers d'articles dans une boutique. La fonction Rechercher est le moyen le plus rapide de trouver un produit spécifique. L'application permet à client d'utiliser un mot clé pour rechercher. Une fois que le client a cliqué sur le bouton de recherche, l'application envoie le contenu de la zone de recherche au backend à l'aide d'une requête http.

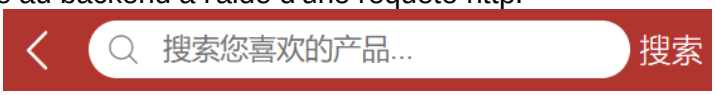


Figure 6 La barre de recherche Rechercher

Ajouter au panier

Une fois que le client a décidé d'acheter un article, il doit ajouter l'article au panier. Se connecter est requis avant d'utiliser le panier. Parce que l'application a besoin de savoir quel utilisateur a effectué l'action. Sinon, l'application redirige à la page de login. Déterminer si l'utilisateur est connecté en vérifiant s'il existe un token dans Vuex ou LocalStorage. Dans la page de détail du produit, il y a un bouton 'Ajouter au panier', une requête http est envoie dès que cliquer le bouton. Le token obtenu dans la connexion précédente est inclus dans l'en-tête de la requête. Le backend décode le token et obtient les infos du client.

Ajouter au panier

Gérer les lieux de livraison

Il est aussi obligatoire de login avant d'utiliser cette fonction. Bien sûr, l'utilisateur peut ajouter une nouvelle adresse, supprimer une adresse ou modifier une adresse. Il faut avoir une adresse par défaut. L'application sélectionnera l'adresse par défaut comme le lieu de livraison lorsque le paiement commencera.

Figure 7 L'interface d'ajoute d'adresse

L'image de droite est l'interface d'ajout d'adresse. J'ai utilisé le formulaire de Vant. Pour ajouter une nouvelle adresse, un nom complet, numéro de mobile et une adresse sont nécessaire. Après avoir rempli, cliquer sur le bouton enregistrer, toutes les informations seront enregistrées dans la base de données.

Payement

La fonction de payement est la dernière fonction majeure que j'introduis et aussi elle est une des plus complique fonctions dans ce projet. Parce que cette application est développée pour le marché chinois, il existe actuellement deux méthodes de payement principales en Chine, Alipay et WeChat pay. Ici, j'ai choisi Alipay. Alipay est une plateforme de paiement mobile lancée en Chine en 2004 par

le géant chinois du e-commerce Alibaba. En 2022, Alipay a atteint la barre des 1,3 milliard d'utilisateurs, dont plus de 320 millions d'utilisateurs quotidiens en Chine et hors de Chine.

Pour réaliser Alipay, il faut utiliser les APIs proposes par Alibaba. Mais c'est le travail du backend, frontend a besoin d'envoyer une requête http au backend et informe le backend qu'un client souhaite effectuer un payement. La requête contient l'identifiant client et l'identifiant du panier. Avoir les données envoyées par le frontend, le backend peut faire la demande de payement à l'API d'Alipay.

Proxy

Les navigateurs ne peuvent pas exécuter de scripts à partir d'autres sites Web. Lorsque vous demandez des ressources à partir d'une page Web d'un nom de domaine vers des ressources d'un autre nom de domaine, toute différence de nom de domaine, de port ou de protocole est CORS (Cross-Origin Resource Sharing). CORS est une restriction de sécurité imposée par le navigateur. Parce que le frontend et le backend sont indépendants dans mon projet. Ils sont exécutés sur des ports différents, le frontend est 8080, le backend est 3000. Si le frontend veut récupérer les données renvoyées par le backend, il est nécessaire de mettre en place un proxy. Dans ce projet, le proxy que j'ai met est : **'/api':http://localhost:3000,** les configurations sont écrits dans le script **vue.config.js**.

Keep-Alive

Keep-Alive est un composant intégré qui nous permet de mettre en cache de manière conditionnelle les instances de composants lors du basculement dynamique entre plusieurs composants. Avec keep-alive, il n'est pas nécessaire d'envoyer fréquemment des requêtes au backend lors de l'accès à certaines pages, ce qui réduit la pression sur le backend et améliore les performances.

II - Backend

Pour le backend, les APIs sont plus importantes. Différente API implémente une fonction différente et renvoie les données correspondantes au frontend. Donc la première chose sera que lister et expliquer la fonction des APIs importantes. Deuxièmement, je vais expliquer comment établir la connexion de la base de données et donner un exemple. Enfin, c'est la partie de la description de token.

APIs

- **/api/search/results**

Cette API sert à la fonction de recherche. Quand client cliquer le bouton 'Rechercher', une requête est envoyée avec le mot entré. Ensuite, le backend établi la connexion de MySQL et utilise la phrase de SQL pour chercher le produit. Car peut-être le mot entré n'est pas complet, ici j'ai choisi d'utiliser 'SELECT * FROM …. WHERE LIKE'. Après, le backend retourne les infos de produit au frontend. Si le produit n'existe pas, le backend retourne un code d'erreur au frontend.

- **/api/login**

L'API de login contrôle l'action de se connecter. L'événement de cliquer envoie une requête http, y compris le numéro de mobile et le mot de passe. Parce que le mot de passe est une donnée privée, la méthode choisie est POST.

Après recevoir la requête, le backend commence à chercher l'utilisateur dans la base de données. Si trouver, il crée un token et retourne au frontend. Le token est une chaîne spéciale obtenue après le chiffrement de l'ID utilisateur par un algorithme. Quand le frontend envoie des requêtes aux APIs privées, il est obligatoire d'avoir le token dans la tête de requête. Comme ça, le backend peut décoder le token et savoir quel utilisateur a envoyé la requête.

- **/api/smscode**

La fonction de cette API est d'envoyer un SMS avec un code de vérification au téléphone mobile de l'utilisateur lorsque l'utilisateur se connecte ou s'inscrit avec le numéro de téléphone mobile. Il faut avoir un AppID et AppKey pour réaliser cette fonction. AppID et AppKey doivent se rendre sur le site Web désigné pour postuler. Bien sûr, il est possible de définir le texte de SMS.

- **/api/register**

Cette API est utilisée pour implémenter la fonction d'inscription. Après recevoir la demande d'inscription, tout d'abord, le backend vérifie s'il existe ce client dans la base de données. Sinon, il crée un nouvel utilisateur avec les infos reçus et mise à jour la base de données.

- **/api/addCart**

Cette API réalise la fonction de 'ajouter au panier'. Dans la page de produit, le frontend envoie une requête avec l'identité de produit et le token de l'utilisateur lorsque l'utilisateur clique le bouton. Ensuite, le backend une nouvelle donnée avec les identités de produit et utilisateur. Il n'y a pas de grande différence pour supprimer.

- **/api/addOrEditAddress**

L'API pour la gestion des adresses. Ajouter une nouvelle adresse, modifier une adresse ou supprimer une adresse. Mettre en œuvre l'API est similaire aux APIs précédentes : établir la connexion de base de données et mettre à jour la base de données.

- **/api/alipay, /api/payResult**

La première API est chargée d'établir une connexion avec Alipay. Si la connexion est réussie, l'application effectuera un saut de page vers la page de paiement d'Alipay. Dans le même temps, l'API transmettra les données du panier, le prix total, l'identité de la commande et les noms de produits.

La deuxième API est définie pour obtenir le résultat du payement. Selon la situation de payement, différents codes seront retournés au frontend, par exemple la commande existe, mais non payé.

Base de données

Étant donné que la base de données utilisée dans ce projet est MySQL, le package correspondant doit être importé. De plus, si l'application souhaite établir une connexion avec succès, il faut configurer le nom d'utilisateur, le mot de passe, le nom de la base de données et le port. Le port utilisé est 3306. Le nom de la base de données est 'vue_shop'.

Exemple :

Pour la fonction d'inscription, la première étape est que vérifier s'il déjà existe l'utilisateur dans la base de données. Donc la phrase de SQL utilisée est 'SELECT * FROM user WHERE tel=**${tel}**'. La partie de ${tel} est le numéro de mobile envoyé par frontend. Si le backend trouve l'utilisateur, il enverra un message au frontend et s'arrête. Sinon, il effectue la deuxième étape : ajouter un nouvel utilisateur à la base de données.

La phrase de SQL pour ajouter un utilisateur est :

INSERT INTO user (……) VALUES (……)

User est le nom du tableau. Les choses entre les premières parenthèses sont les noms des champs du tableau, id, mot de passe etc. Les données entre les deuxièmes parenthèses sont les données envoyées par le frontend.

Token

Sauf que les APIs, il existe d'autres fonctions très importantes dans le backend. La fonction de token est la fonction plus importante d'entre elles. Le jeton (token) est une chaîne générée par le serveur en tant que jeton que le client peut demander. Après la première connexion, le serveur génère un token et le renvoie au client. À l'avenir, le client n'aura plus qu'à apporter ce token avant pour demander des données, pas besoin d'apporter à nom d'utilisateur et mot de passe. Le but de Token est de réduire la pression sur le serveur et de réduire les requêtes fréquentes sur la base de données.

Dans ce projet, j'ai utilisé le package JWT (JSON Web Token) pour créer les token. Tout d'abord, il faut déterminer le secret. Le secret est la règle de coder et décoder. Avec l'aide de la fonction ***sign,*** une token est généré. Pour décoder, il faut utiliser la fonction ***decode***.

CONCLUSION

Ce projet interne est un grand projet full-stack développé par moi seul. Durant le développement du projet, j'ai rencontré de nombreux problèmes, par exemple data rendering, proxy etc. Et j'ai dû chercher sur l'internet et proposer des solutions par moi-même ce qui m'a permis d'utiliser les connaissances apprises. Après avoir trouvé des solutions et corrigé des bugs, j'éprouve un sentiment de l'accomplissement inégalé. Le processus m'a aidé à mieux comprendre les langages HTML, CSS et JavaScript.

En plus de cela, chaque page de cette application nécessite de nombreuses fonctions différentes. Tout en réalisant la fonction, il est également nécessaire de prendre en compte l'esthétique de la page pour la conception Web. Donc juste le frontend, il y a beaucoup de choses à faire. Pour la partie de backend, il existe de nombreuses APIs, et chaque API implémente une fonction différente et renvoie des données différentes au frontend. J'ai besoin de bien réfléchir à la correspondance entre le frontend et le backend.

En réalisant ce projet, j'ai fait de grands progrès en programmation, notamment le framework Vue et Nodejs. Parce que c'est la première fois que je développe un projet full-stack avec le framework Vuejs et Nodejs. Le langage que j'utilisais habituellement avant est Java. Grâce à ce projet, j'ai appris plus de connaissances. C'est une expérience importante pour moi, je crois qu'elle jouera un rôle important dans ma future carrière.

ANNEXE

La documentation de Vue.js

https://vuejs.org//guide/introduction.html

La documentation de VantUI

https://vant-ui.github.io/vant/#/en-US

Swiper

https://swiperjs.com/get-started

fun-tab

https://github.com/ScoutYin/fun-tab

better-scroll :

http://ustbhuangyi.github.io/better-scroll/