INFOB318: Projet Individuelle

Documentation du programmeur du projet Clothed

VERSION 0.0.1

Author: Tchatchoua Bruge

May 8, 2024

Contents

1 Guide de Reprise du projet clothed **2**
1.1 Introduction . . . 2
1.2 Sources du Logiciel . . . 2
1.3 Environnement de Développement . . . 2
1.4 Installation et Configuration . . . 3
1.4.1 Instruction d'installation . . . 3
1.5 Architecture du Logiciel . . . 3
1.6 Développement . . . 7
1.7 Contribution au Projet . . . 7

2 Explication et documentation des composants **7**
2.1 Back-end(Node js) . . . 7
2.1.1 Architecture Détaillée et Motivations . . . 7
2.1.2 Upload . . . 8
2.2 Front-end(Vue js) . . . 9
2.2.1 Arborescence des méthodes et des packages les plus importantes . 9
2.2.2 Détails de l'architecture des fichiers . . . 10
2.3 comportement de l'application au run-time . . . 10

3 Guide de teste **11**
3.1 Test unitaire . . . 11

4 Licence **11**
4.1 Annexes . . . 12
4.1.1 Références . . . 12

1 Guide de Reprise du projet clothed

1.1 Introduction

Objectif du projet

Clothed est une application web conçue pour permettre le partage de vêtements et autres articles entre particuliers. L'objectif est de fournir une plateforme facile à utiliser pour aider les personnes à obtenir les biens dont elles ont besoin facilement pour éviter le gaspillage de ressources. l'application a particulièrement été pensée pour mieux s'adapter aux personnes âgées.

Pour être en mesure de poursuivre le projet, voici une description des ressources utilisés.

1.2 Sources du Logiciel

Développeurs

Le projet a été développé par [Tchatchoua Bruge]. Pour plus de détails sur les contributions, veuillez consulter le système de gestion de versions du répertoire sur GitHub

Cahier des Charges

Le cahier des charges est accessible dans la section documentation du projet sur github, fournissant un aperçu des exigences et des objectifs initiaux du projet.

1.3 Environnement de Développement

Technologies Utilisées

- **Langages utilisés:** [node.js, express.js,Vue.js, Mysql]
- **Éditeurs et outils :** Développé en utilisant [Éditeur/IDE : Visual Studio Code]

Configuration Requise

- Système d'exploitation : Linux Ubuntu / Windows / mac
- Node.js installé avec npm (Node Package Manager)
- MySQL installé et configuré
- Xampp doit être installé et configurer pour établir la connexion avec phpMyAdmin(base de données en ligne)
- IDE recommandé : Visual Studio Code (car elle est celle qui a permis la connexion)

1.4 Installation et Configuration

Téléchargement du Code Source

Le code source est accessible sur GitHub via le lien : `https://github.com/UNamurCSFaculty/2324_INFOB318_Clothed1.git`.

1.4.1 Instruction d'installation

- Cloner le dépôt Git : git clone
- Accéder au dossier du projet : **cd clothed-projet1**
- Installer les dépendances du projet : npm install.

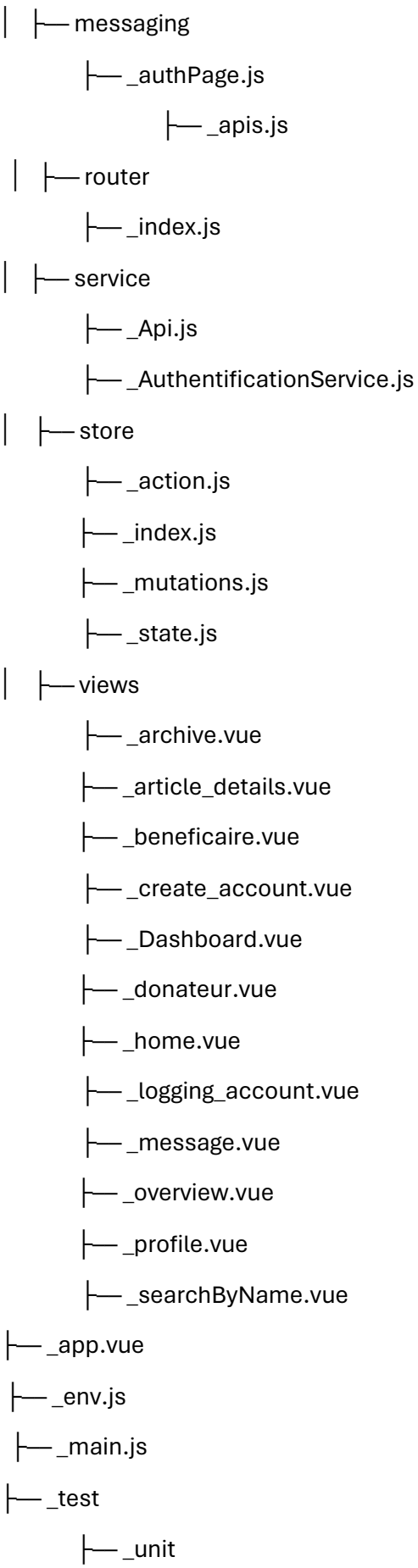
1.5 Architecture du Logiciel

Clothed1 est structuré en deux principales parties

- **Front-end:** interface utilisateur développée avec Vue.js
- **Back-end:** Serveur Node.js qui gère la logique métier et l'interaction avec la base de données MySQL.

Diagramme d'Architecture

```
Clothed/
│
├── node_app/
│   ├── __config
│           ├── _dbconfig.js
│   ├── controllers
│           ├── _imageController.js
│           ├── _personController.js
│   ├── les_test
│           ├── _imageApi.test.js
│           ├── _personApi.test.js
│   ├── models
│           ├── _imageModel.js
│           ├── _index.js
│           ├── _personModels.js
│   ├── route
│           ├── _personRouter.js
│           ├── _server.js
│   ├── upload
├── vue/
│    ├── models
│           ├── _cl-logo.png
│           ├── _global.css
│           ├── _unamur.png
│      ├── components
│           ├── _axiosClient.js
│           ├── _bouton.vue
│           ├── _CreateAccountForm.vue
│           ├── _fonction.js.js
│           ├── _HomeForm.vue
│           ├── _sidebar.vue
```


messaging
_authPage.js
_apis.js
router
_index.js
service
_Api.js
_AuthentificationService.js
store
_action.js
_index.js
_mutations.js
_state.js
views
_archive.vue
_article_details.vue
_beneficaire.vue
_create_account.vue
_Dashboard.vue
_donateur.vue
_home.vue
_logging_account.vue
_message.vue
_overview.vue
_profile.vue
_searchByName.vue
_app.vue
_env.js
_main.js
_test
_unit

├── _ArticleDetails.spec.js

├── _Beneficaire.spec.js

├── _Donateur.spec.js

├── _SearchView.spec.js

1.6 Développement

Déploiement Local

pour déployer l'application localement :

- Exécuter npm start dans le dossier du serveur back-end.
- Exécuter npm run serve dans le dossier front-end.

1.7 Contribution au Projet

Procédure de Contribution

Pour contribuer au projet Clothed1 :

- Forker le dépôt sur GitHub.
- Créer une nouvelle branche pour vos modifications
- Soumettre un pull request avec une description détaillée des changements.

2 Explication et documentation des composants

2.1 Back-end(Node js)

2.1.1 Architecture Détaillée et Motivations

les composants contenus dans cette section suivent une architecture de base utilisée pour le développement web d'application utilisant node js comme back-end. une explication détaillée de chaque composant est comme suit :

Config

Le dossier contient le fichier **dbconfig.js** utiliser pour la configuration de la base de données.

- **Explication :** Ce fichier contient toutes les informations nécessaires pour établir une connexion avec la base de données, comme les chaînes de connexion, les identifiants, et d'autres paramètres spécifiques à l'environnement. Cela centralise la gestion des configurations de base de données, facilitant la maintenance et la modification sans perturber le code du reste de l'application.

Controllers

La gestion des requêtes entre les données du front-end passe par les méthodes comprises dans ce dossier. Les détails concernant chaque fichier sont comme suit :

- **imageController.js :** Ce contrôleur traite toutes les actions HTTP (comme POST pour télécharger une image, GET pour afficher une image, DELETE pour supprimer une image) en appelant les méthodes appropriées des modèles d'image. Ce contrôleur peut également valider ou préparer les données pour s'assurer qu'elles respectent les critères requis.

- **personController.js :** ce fichier gère les actions HTTP pour les données de personnes, comme créer, mettre à jour, ou supprimer des informations sur les personnes. Il contient des logiques telles que la vérification de si les champs sont bien remplis avant de soumettre à la base donnée.

Models

Ici, nous avons la configuration des différents modèles dont nous nous sommes servis pour la base de données.

- **imageModel.js :** Définit la structure des données pour les images, incluant les champs nécessaires et le type de données. On peut également inclure la configuration des relations avec d'autres modèles si nécessaire.
- **index.js :** Sert à importer et à exporter tous les modèles définis dans le dossier pour faciliter l'accès depuis d'autres parties de l'application.
- **personModel.js :** Définit la structure des données pour les personnes, comme pour imageModel.js. Fournit des méthodes pour gérer les données de personnes, comme l'ajout, la mise à jour, et la suppression dans la base de données.

Route

le dossier contient toute la logique liée à la gestion des requêtes.

- **personRouter.js :** Mappe les routes HTTP aux méthodes spécifiques dans personController.js, facilitant ainsi la gestion des requêtes et la distribution des responsabilités.

  * l'intégration des middlewares et de l'authentification peut être ajoutées avant que les requêtes ne soient passées au contrôleur.

- **server.js :** Configure et lance le serveur web, en utilisant le framework Express. Définit également les middlewares globaux, comme le traitement des JSON, la gestion des erreurs,et les logs. Importe et utilise les fichiers route pour connecter les requêtes entrantes aux contrôleurs appropriés.

2.1.2 Upload

le dossier est destiné à stocker les fichiers téléchargés via l'application, séparant le stockage des données utilisateur du code de l'application pour des raisons de sécurité et d'organisation.

Ces éléments reflètent une structure typique d'une application web moderne où la séparation des préoccupations est clairement définie pour faciliter la maintenance et la mise à jour de l'application.

2.2 Front-end(Vue js)

2.2.1 Arborescence des méthodes et des packages les plus importantes

main.js

Ce fichier initialise l'application Vue en utilisant des importations de base comme Vue Router pour la gestion des routes, Vuex pour la gestion de l'état, et Axios pour les requêtes HTTP. Le fichier configure aussi Axios avec une URL de base pour toutes les requêtes sortantes.

Packages utilisés

- **vue :** Framework de base pour construire des interfaces utilisateur.
  mon choix s'est porté sur vue en raison de sa simplicité et de sa performance. le framework est facile à apprendre, car il a une documentation claire qui facilite l'apprentissage. Léger et rapide, Vue.js optimise les mises à jour de l'interface utilisateur, garantissant des performances élevées pour les applications web modernes. Avec une communauté active et un large choix de plugins, Vue.js est idéal, car il est évolutif.

- **bootstrap :** Framework CSS pour le style.
  Il a été utilisé dû à sa capacité à accélérer considérablement le processus de développement. Grâce à son vaste ensemble de composants prédéfinis et son système de grille responsive, Bootstrap permet de créer des sites web esthétiques et fonctionnels sans avoir à écrire beaucoup de code CSS ou JavaScript from scratch. Cela rend la mise en place de designs complexes beaucoup plus simple et plus rapide.

- **axios :** Utilisé pour les requêtes HTTP.
  Axios fonctionne à la fois dans le navigateur et avec Node.js, offrant une solution cohérente pour mes requêtes HTTP. Il permet d'écrire un code plus propre et plus lisible, notamment grâce à la possibilité d'utiliser async et await.

- **Vuex :** pour la gestion de l'état.
  Sachant que l'application Vue a plusieurs composants qui partagent un état, notament le nom ou l'email de l'utilisateur, Vuex offrir une solution robuste et bien intégrée pour maintenir la clarté et l'efficacité du code. les données stockées dans le vuex peuvent être accessible dans n'importe quel composant.

Motivation : La configuration de base dans ce fichier établit les dépendances essentielles pour l'application, gère l'état global et définit les styles et la structure de routage.

App.vue :

C'est le composant principal de l'application Vue qui utilise RouterView pour le rendu des composants basés sur l'URL courante. Ce fichier agit comme un point d'ancrage pour tous les autres composants et vues.

Motivation : Centraliser la gestion de l'interface utilisateur principale et faciliter la navigation.

2.2.2 Détails de l'architecture des fichiers

les dossiers et leurs composants sont expliqués de manière générale dans la section qui suit :

- **views :** Le dossier views contient des composants Vue qui représentent des pages entières et des layouts majeurs dans l'application. En d'autres termes, elle représente l'interface de l'application
- **Messaging :** Les fichiers à l'intérieur de ce dossier inclus des composants pour l'affichage et la gestion des messages. Ici, des interactions avec une API externe (chatengine.io) est utilisée pour les fonctionnalités liées à la messagerie. Il utilise Axios pour les opérations REST liées à l'authentification des utilisateurs.
- **Router :** le fichier définit les routes de l'application Vue.js. Certaines routes sont définies soient avec un chemin, un composant ou d'autre route (sorte de relation parent-enfant). Les routes configurées sont exportées pour être utilisés dans le fichier main de l'application.
  Cette configuration permet à l'application de gérer différentes URL et de les lier à des composants spécifiques.
- **store :** Utilisé pour la gestion de l'état. Vuex est utilisé ici pour gérer l'état global de l'application. Cela inclut les informations utilisateur, les préférences, et les données de session, qui doivent être accessibles de manière cohérente et réactive à travers l'application. La structure du store, avec ses actions, mutations, et états, assure que l'état peut être modifié de manière prévisible et traçable, celle-ci est cruciale pour le débogage et la maintenance de l'application.
- **service :** ce dossier était destiné à gérer le service d'authentification. il n'a pas été mis en œuvre et a été laissé à disposition en cas de continuité du projet.

  Chacun de ces éléments contribue à une architecture solide, facilitant le développement, l'extension, et la maintenance de l'application.

2.3 comportement de l'application au run-time

le diagramme d'activité suivant décrit les appels lorsque l'application est lancée :

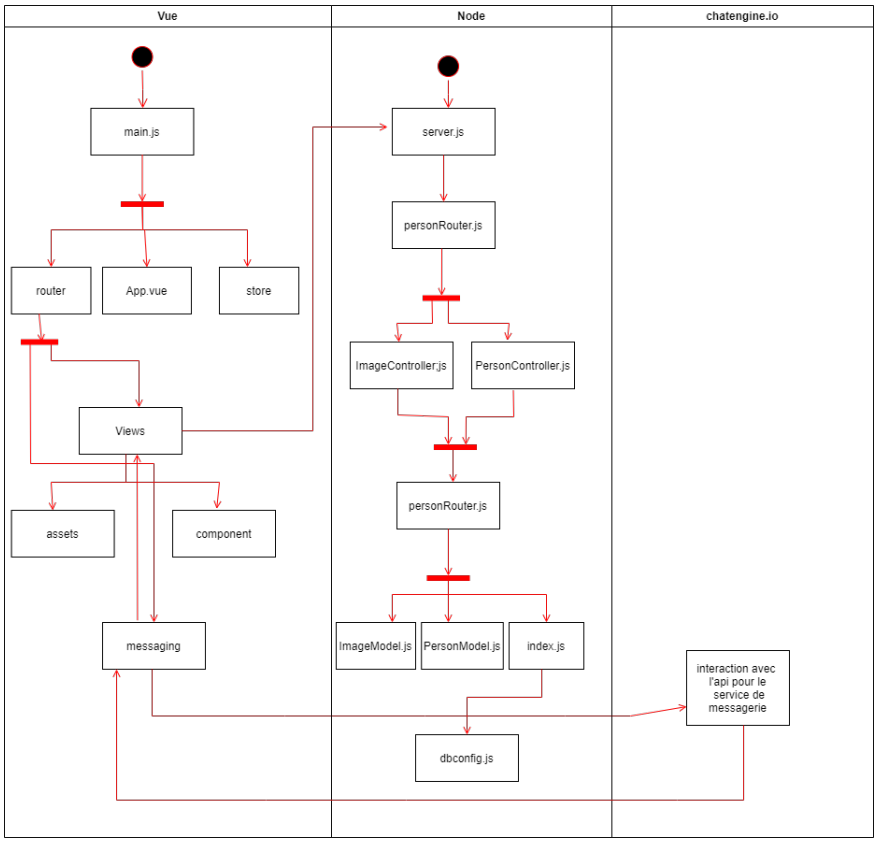


Figure 1: run-time routing

3 Guide de teste

Les testes effectuer sont fait en utilisant jest pour les tests unitaires et supertest pour simuler les requêtes HTTP.

3.1 Test unitaire

Pour faire des tests unitaire, il suffit tout simple d'entré la commande **npm test** . Cette commande exécutera tous les tests trouvés dans les fichiers .test.js ou .spec.js dans le projet. Vous verrez dans la console un rapport indiquant quels tests ont réussi et lesquels ont échoué, avec des détails sur les erreurs en cas d'échec.

4 Licence

La licence est sous MIT licence

Puis-je redistribuer l'application ?

Oui, vous pouvez redistribuer et modifier le projet. La licence MIT est une licence de logiciel libre qui permet une grande flexibilité d'utilisation. Voici quelques points clés à propos de cette licence :

- **Liberté de redistribution :** Vous pouvez redistribuer le logiciel, que ce soit gratuitement ou en le vendant.
- **Permission de modifier :** Vous avez le droit de modifier le code source du logiciel sous licence MIT selon vos besoins.
- **Utilisation commerciale :** Vous pouvez utiliser le logiciel dans des projets commerciaux.
- **Peu de restrictions :** La licence impose très peu de restrictions, principalement l'obligation de conserver le texte de la licence dans les copies ou versions substantielles du logiciel.

Cependant, même si la licence MIT est permissive, il est important de toujours conserver la mention de droit d'auteur et le texte de la licence avec le logiciel, même si vous le modifiez ou le redistribuez.

4.1 Annexes

4.1.1 Références

- Documentation Vue.js : `https://vuejs.org/v2/guide/`
- Documentation Node.js : `https://nodejs.org/en/docs/`