Documentation technique de mon projet NodeJs :

Nom du projet : 3npm_project

Version : 1.0.0

Description : le projet met en place un serveur node.js capable de simuler et surveiller les positions des véhicules en temps réel

Auteur : moi-même, Mohamed Farouck LATOUNDJI

Le fichier principal du projet est : server.js

Dépendances utilisées :

L'application utilise les fonctionnalités suivantes :

- bcrypt : c'est une bibliothèque pour hasher les mots de passes
- cookie-parser : C'est un Middleware pour analyser les cookies
- events : C'est un emetteur d'événements nodeJs
- express : C'est un framework web pour nodeJs
- express-validator : C'est un Middleware pour valider et assainir les données
- fs : C'est un module de système de fichiers pour interagir avec les fichiers
- jsonwebtoken : Implémentation de JSON Web Token
- mongoose : outil de modélisation d'objet pour MongoDB
- path : C'est un utilitaire pour travailler avec les chemins de fichiers et de répertoires
- socket.io : Socket.IO est une bibliothèque événementielle pour les applications Web en temps réel. Il permet une communication bidirectionnelle en temps réel entre les clients Web et les serveurs.
- url : C'est un utilitaire pour la résolution et l'analyse des URL.

L'application est structurée comme suit :

- server.js : C'est le fichier principal ou l'application est initialisé et démarrée.
- node_modules/ : C'est le répertoire contenant toutes les dépendances installées
- database/index.js : C'est le fichier de connexion à la base de données mongoDB 3npm_project.
- api/ : Contient les routes et les services
- api/services/users.js : Contient les fonctions pour créer un utilisateur, authentifier un utilisateur, hasher les mots de passe des utilisateurs etc.
- api/services/vehiclePositions.js : contient les fonctions pourqu'un utilisateur puisse ajouter les données de position d'un véhicule et pourqu'il puisse aussi les récupérer
- api/routes/ : C'est le répertoire pour les définitions de chaque route.(inscription , connexion, déconnexion, récupérations des données de positions des véhicules)
- models/ : C'est le répertoire pour les modèles Mongoose qui permettent d'interargir avec la base de données
- middleware/authMiddleware.js : c'est le middleware d'authentification
- le fichier client.html teste la réception des mises à jour en temps réel via Socket.io

- le fichier private.key contient la clé privée qui est utilisée pour signer les tokens JWT, pour la générer on tape la commande : Elle garantit que les tokens émis par le serveur ne peuvent être falsifiés par des tiers
  openssl genpkey -algorithm RSA -out private.key
- le fichier public.key contient la clé publique qui est utilisée pour vérifier les signatures des tokens JWT. Elle est distribuée publiquement et peut être utilisée par toute application ayant besoin de vérifier l'authenticité et l'intégrité d'un token signé avec la clé privée correspondante. Pour la générer on tape la commande :
  openssl rsa -pubout -in private.key -out public.key

Fonctionnalités :

- ✓ **Authentification Utilisateur :**
  - Hachage des mots de passe utilisant bcrypt
  - Authentification basée sur JWT utilisant jwtwebtokens
- ✓ **Communication en temps réel**
  - Fonctionnalités en temps réel comme le Chat ou les notifications utilisant socket.io
- ✓ **Validation et Assainissement**
  - Validation et assainissement des données de requête utilisant express-validator
- ✓ **Interaction avec la base de données :**
  - Opérations sur la base de données MongoDB utilisant mongoose
- ✓ **Serveur Web**
  - Gestion des requêtes http utilisant express

Comment exécuter l'application :

1. **Installer les dépendances :**

Initialisez le projet Node.js : npm init -y

Installez Express : npm install express

Installez jsonwebtoken : npm install jsonwebtoken

Installez fs : npm install fs

Installez events : npm install events

Installez cookie-parser : npm install cookie-parser

Installez Socket.IO : npm install socket.io

Installez Mongoose : npm install mongoose

Installez express-validator : npm install express-validator

Installez bcrypt pour le hashage des mots de passe : npm install bcrypt

2. **Démarrer l'application :**

node server.js