Université de Mons
Faculté des sciences
Département d'Informatique

Energenius

Rapport de projet

Professeur :
Tom Mens
Assistants :
Pierre Hauweele
Sébastien Bonte

Auteurs :
Godwill Louhou
Gilles Jaunart
Jérémy Delnatte
Louis Dascotte

Année académique 2022-2023

Table des matières

1 Introduction

1.1 Préambule

Ce rapport présentera l'application client/serveur Energenius, ainsi que les fonctionnalités disponibles et implémentées. Il sera également présenté un petit mode d'emploi afin de permettre de lancer l'application localement.

1.2 Répartition des tâches

Pour réaliser ce projet, nous avons décidé de séparer les tâches de la manière suivante :

- **Godwill Louhou** : Développement de l'application client pour les clients.
- **Gilles Jaunart** : Développement de l'application client pour les fournisseurs.
- **Jérémy Delnatte** : Développement de l'application serveur/API.

De par cette séparation des tâches, il a été possible de se concentrer sur un aspect du projet à la fois et par personne, ce qui nous a permis d'améliorer nos compétences dans les domaines choisis. Nous étions à la base un groupe de 4 personnes, mais l'un d'entre nous ne donnait plus de nouvelles, et ne répondait plus à nos messages. Nous avons donc décidé de continuer le projet à 3. Après une réunion avec Mr. Tom MENS, il a été décidé que nous serions considérés et notés comme un groupe de 3.

1.3 Choix techniques

1.3.1 Langage

Pour le développement de l'application serveur, il nous a été imposé d'utiliser Java. Cependant, le choix était plutôt libre vis à vis de l'application client. Après avoir envisagé différentes technologiques, nous avons opté pour JavaScript couplé au framework ReactJs. Nous avons choisi ce framework, car en plus d'être très populaire, il permet de développer des applications web de manière très efficace et rapide. De plus, il permet de développer des applications web de manière modulaire, ce qui permet de séparer les différentes parties de l'application et de les réutiliser facilement. Le fait qu'il soit si populaire a également facilité la recherche de solutions aux divers problèmes rencontrés, ainsi que la recherche de documentation.

1.3.2 Base de données

Pour la base de données, nous avons choisi d'utiliser MongoDB Atlas, qui est une base de données NoSQL hébergée sur le Cloud. Cela nous a donc permis d'éviter de devoir héberger la base de données nous même, et de par l'application MongoDB Compass, il a été aisé de créer les différentes collections et de les gérer.

1.3.3 Outils

Durant le développement de ce projet, certains outils ont été indispensables pour nous permettre de travailler et collaborer de façon efficace. Ainsi, l'utilisation de GitHub nous a permis de travailler facilement sur le même code, et de pouvoir le partager sans avoir à faire de transferts de fichiers. Pour nous organiser, nous avons également créé un Trello afin de suivre l'avancée dans nos différentes tâches, mais également d'en ajouter pour les autres lorsque nécessaire. Nous avons majoritairement utilisé Discord pour communiquer entre nous. L'outil Postman nous a également été très utile pour tester les différentes routes de l'API, sans avoir à tout implémenter sur l'application client.

1.3.4 Structure des applications

Les deux applications client sont basées sur le même schéma de fonctionnement, avec comme majoritaire différence les pages qui sont disponibles.

1.3.5 Identifiants pour les tests

Application pour les clients :
Pour l'application pour les clients, il est possible de créer directement un compte depuis la page d'inscription, comme présenté dans la vidéo de démonstration. Il est recommandé d'utiliser une adresse mail valide pour pouvoir utiliser la fonction de réinitialisation de mot de passe. Cependant, il est également possible d'utiliser les identifiants suivants pour se connecter :

— **Adresse mail** : client@energenius.com
— **Mot de passe** : Client1234

Application pour les fournisseurs :
Pour l'application des fournisseurs, il faut utiliser les identifiants qui seront fournis. En effet, nous sommes partis du principe que dans le contexte où des entreprises utiliseraient cette application, ils ne s'enregistreraient pas via un simple formulaire d'inscription avec une adresse mail, et auraient une communication plus directe avec les administrateurs de l'application. Les identifiants sont donc :

— **Identifiant** : 36558965701
— **Mot de passe** : Fournisseur1234

2 Application client

2.1 Application pour les clients

Vidéo de présentation :
Il est possible de retrouver une playlist contenant des vidéos de présentation de l'application pour les clients via **ce lien**, et les vidéos de présentation de

l'application pour les fournisseurs via **ce lien**.

Fonctionnalités implémentées :
L'application pour les clients permet la création de portefeuilles, la visualisation des données de consommation, la gestion des contrats ainsi que l'entrée de données de consommation. Le développement de l'interface graphique suit la maquette proposée pendant la phase de modélisation, à quelques différences près, à cause de contraintes techniques ou de changements d'avis. L'utilisateur peut donc se rendre sur le site et se connecter à l'aide du compte qu'il a créé dans la page de création de compte. Une fois son compte créé, il est redirigé vers la page de connexion, où il peut alors se connecter et utiliser l'application.

Avantages et inconvénients
L'application est visuellement agréable et intuitive d'utilisation. Elle a été pensée de sorte à ce que n'importe quel utilisateur puisse l'utiliser sans soucis, peu importe son expérience sur Internet. Elle est très réactive, permettant de limiter le temps d'utilisation au maximum. Le fait d'utiliser React a permis de réutiliser certaines pages pour en créer d'autres, et donc accélérer la vitesse de développement, permettant également d'ajouter de nouvelles pages dans le futur sans trop de soucis. L'utilisation des JWT (JSON Web Tokens) permet de garder une sécurité sur le site, mais également de permettre au client de ne pas avoir à se connecter à chaque changement de page. Grâce au package **react_i18next**, nous avons pu traduire l'application pour les clients en anglais et en français. Il est très facile d'ajouter de nouvelles langues dans le futur, car il suffit de faire la traduction de chaque valeur et de les faire correspondre aux clés des langues déjà traduites. Malheureusement, l'application contient quelques défauts, non résolus par manque de temps.

Fonctionnalités manquantes
Il n'a pas été possible d'utiliser le protocole HTTPS pour sécuriser les échanges de données entre l'application client et l'application serveur. Ainsi, les requêtes sont envoyées en clair, ce qui peut poser problème si un attaquant arrive à intercepter les données. De même, la sélection d'échelle de temps sur les graphiques n'a pas pu être implémentée, mais peut se faire ultérieurement.

3 Serveur

3.1 API

L'api est séparée en trois parties : la partie Repository, la partie Service et la partie Controller.

- **Repository :** Cette partie permet d'interagir avec la base de donnée MongoDB, et de céer toutes les requêtes pour récuperer les documents dans la DB.
- **Service :** Cette partie sert à tous ce qui est de la logique de l'api. Elle

permet aussi la gestion des exceptions et des erreurs en envoyant des exceptions à la couche supérieur. Dans cette couche, des fonctions qui tourne tous les jours ont été aussi implémenter.
— **Controller :** Cette couche permet a l'application client d'interagir avec toutes la partie api de l'application. Elle gère aussi la gestion des exceptions envoyer par la couche service.

3.2 Base de données

La base de données ne respecte pas totalement la modélisation de l'ERD éffectuer dans la première partie du projet. Ces changements ont été effectués pour mieux fonctionner avec la base de données qui est du NoSQL.

3.3 Tests unitaires

Des tests unitaires ont été réalisés pour pratiquement toute la couche service de l'application. Malheuresement, par manque de temps certaines fonctions de la couche n'ont pas pu être tester. Pour la couche Repository, nous n'avons pas réussi à faire fonctionner une base de données embarqué pour tester la couche Repository de l'app.

4 JWT

Pour ne pas devoir faire des requêtes d'authentification pour chaque requêtes à l'api, nous avons choisi d'utiliser JWT qui va permet d'envoyer un token lors de la connexion d'un utilisateur. Ce token va être envoyer dans chaque requêtes et va permettre à l'api de vérifier que la requête viens bien de l'utilisateur connecté.

5 Extensions

5.1 Extension 4 - Analyse statistique de la consommation énergétique - Gilles Jaunart

5.1.1 Description de l'extension :

L'objectif de cette extension est de permettre au client de visualiser les données de consommation énergétique associées à son portefeuille. En plus de cela, l'extension offre une interface visuelle permettant d'analyser des statistiques relatives à la consommation énergétique. Les statistiques fournies au client comprennent le minimum, le maximum, la moyenne et l'écart-type de la consommation énergétique. D'autres données telles que la médiane et les quartiles ont été exclues, car elles ont été jugées soit inutiles, soit trop difficiles à interpréter pour un client non spécialisé. Il convient de noter que des infobulles sont disponibles pour expliquer la signification de la moyenne et de l'écart-type.

5.1.2 Vidéo de présentation des fonctionnalités

La vidéo de présentation des fonctionnalités de l'extension est disponible via ce lien.

5.1.3 Avantages et inconvénients

Beau graphique

5.1.4 Fonctionnalités manquantes

Malgré la découverte de méthodes permettant de détecter les problèmes, tels que l'utilisation de la méthode des outliers pour repérer des comportements inhabituels dans la consommation d'énergie, tels qu'une fuite d'eau ou de gaz, ces méthodes n'ont pas été mises en œuvre en raison d'une contrainte de temps. De plus, la comparaison de la consommation entre des habitations ayant des caractéristiques similaires n'a pas été réalisée conformément aux demandes. Au lieu de cela, une approche très basique a été utilisée, qui ne prend pas en compte les caractéristiques comparables des habitations. La comparaison est effectuée de manière globale et ne tient pas compte des spécificités de chaque logement.

5.2 Extension 5 - Auto-production d'énergie - Jérémy Delnatte

5.2.1 Introduction

L'extension d'auto-production d'énergie permet de surveiller la production d'électricité et de gérer les certificats verts.

Pour surveiller la production d'électricité, l'utilisateur peut ajouter un point de production d'électricité qui est associé a un compteur numérique. Pour fonctionner le point de production d'électricité doit être accepter par le fournisseur. Le point de production d'énergie fonctionne en toute transparence avec les fonctionnalités des points de fournitures de l'application de base.

Pour les certificats verts, l'utilisateur peut en demander lorsque le seuil de production d'électricité à depassser les 1000 kWh, dès lors une requête est envoyé au fournisseur et peut accepter le certificat vert ou non. L'application permet de voir tous les certificats verts demandés et acquis.

Malheuresement, par manque de temps, la partie sur l'analyse de rendement énergetique n'as pas pu être faite.

5.2.2 API

La base de données de l'application de base a dû légèrement être modifier pour pouvoir garder l'historique des certificats verts. Pour les points de productions, la base de données n'a pas dû être modifier.

Dans l'api, les méthodes des points de fournitures ont pu être utiliser pour les points de production d'énergie.

5.2.3 Application client

Pour la partie front-end de l'application, le graphique de la partie de base à été réutiliser pour pouvoir afficher la production d'énergie. Comme pour les points de fournitures, on peut aussi voir la production d'électricité en tableau. Un tableau a été créer pour afficher les informations des certificats verts.

5.3 Extension 7 - Facturation et paiement d'acomptes - Godwill LOUHOU

5.3.1 Introduction

Par manque de temps, et à cause de problèmes mentionnés dans l'introduction de ce rapport, je n'ai pas pu implémenter mon extension comme je le souhaitais. Quelques pages en front-end existent, mais la logique de l'API et du serveur ne sont pas implémentées, ce qui rend mon extension inutilisable. Je vais donc décrire ici ce que j'avais prévu de faire, et ce que j'ai pu faire.

5.3.2 Base de données

Pour cette partie, il était prévu d'ajouter une nouvelle collection **invoices**, qui contiendrait des documents représentant les factures, ainsi qu'une collection **payments**, qui contiendrait des documents représentant les différentes transactions effectuées. Enfin, une collection **advance_payments** aurait été ajoutée, pour stocker les acomptes versés et à verser par les clients.

6 Installation

6.1 Instructions d'installation

Pour tester l'application sur votre ordinateur, il est nécessaire de posséder les outils suivants :

— Nodejs (version 18 ou supérieure)
— Gradle (version 7.5.1 ou supérieure)
— Java (version 17 ou supérieure)

Pour lancer l'application, il faudra dans un premier temps se rendre dans le dossier *back_end*. Une fois dans ce dossier, il faudra lancer la commande suivante :

```
gradle bootRun
```

pour démarrer le serveur. Celui ci va démarrer sur le port 8080. Si une autre application utilise ce port, l'application ne pourra pas se lancer.

Une fois le serveur démarré, il faudra se rendre dans le dossier *front_end*. Une fois dans ce dossier, il faudra se rendre dans un des sous dossiers, parmi

lesquels : *client_ app_ commune*, *client_ app_ extension_ facturation* et toutes les autres extensions représentées par leurs noms respectifs.

Une fois dans un de ces sous dossiers, il faudra lancer la commande suivante :

```
npm install
```

Cette commander permettra d'installer toutes les dépendances nécessaires au bon fonctionnement de l'application client. Une fois les dépendances installées, il faudra lancer l'application avec la commande :

```
npm start
```

L'application se lancera sur le port 3000. Si une autre application utilise ce port, l'application ne pourra pas se lancer.

7 Conclusion

Ce projet a été pour nous l'occasion d'améliorer nos compétences dans le développement, que ce soit via l'apprentissage d'un nouveau langage de programmation ou de par une pratique plus approfondie de ceux que nous connaissions déjà. Malgré les différentes difficultés que nous avons rencontrés, nous pensons avoir rendu une application correcte. Bien sûr, de nombreuses fonctionnalités n'ont pas pu être implémentées, ou d'autres implémentées maladroitement. Nous pensons ainsi qu'avec un peu plus de temps, nous aurions pu fournir une application beaucoup plus complète et fonctionnelle. Cependant, nous sommes satisfaits du résultat obtenu, et nous espérons que ce projet vous plaira.