Projet P\_Sec  
Site d’e-commerce sécurisé (webstore)



De Pina Correia Ryan – Grp2C

ETML

01.02.2024

Gaël Sonney

Table des matières

[1 Spécifications 3](#_Toc192502915)

[1.1 Titre 3](#_Toc192502916)

[1.2 Description 3](#_Toc192502917)

[1.3 Matériel et logiciels à disposition 3](#_Toc192502918)

[1.4 Cahier des charges 3](#_Toc192502919)

[1.4.1 Authentification et sécurité 3](#_Toc192502920)

[1.4.2 Administration 3](#_Toc192502921)

[1.4.3 Protection des données 3](#_Toc192502922)

[1.4.4 Documentation 3](#_Toc192502923)

[1.4.5 Technologies utilisées 4](#_Toc192502924)

[1.5 Les points suivants seront évalués 4](#_Toc192502925)

[1.6 Validation et conditions de réussite 4](#_Toc192502926)

[2 Analyse 4](#_Toc192502927)

[2.1 Opportunités 4](#_Toc192502928)

[3 Réalisation 4](#_Toc192502929)

[3.1 Dossier de Réalisation 4](#_Toc192502930)

[3.1.1 Page d’inscription (Signup) 4](#_Toc192502931)

[3.1.2 Page de connexion (Login) 5](#_Toc192502932)

[3.1.3 Page d’Administration (Admin) 6](#_Toc192502933)

[3.1.4 Page de profile (Profile) 7](#_Toc192502934)

[4 Tests 8](#_Toc192502935)

[4.1 Dossier des tests 8](#_Toc192502936)

[4.1.1 Lancement 8](#_Toc192502937)

[4.1.2 Base de données : 8](#_Toc192502938)

[4.1.3 Authentification : 8](#_Toc192502939)

[5 Conclusion 8](#_Toc192502940)

[5.1 Bilan des fonctionnalités demandées 8](#_Toc192502941)

[5.2 Bilan de la planification 9](#_Toc192502942)

[5.3 Bilan personnel 9](#_Toc192502943)

[6 Divers 9](#_Toc192502944)

[6.1 Webographie 9](#_Toc192502945)

# Spécifications

## Titre

Un site d’e-commerce très simple pour des fonctionnalités simple

## Description

Ce projet vise à concevoir et développer un site d’e-commerce sécurisé en Node.js. L'objectif principal est de mettre en place un système d’authentification sécurisé pour les clients et les administrateurs du site, avec un accent sur la protection des données.

## Matériel et logiciels à disposition

• Un ordinateur équipé de Docker Desktop pour la conteneurisation.

## Cahier des charges

### Authentification et sécurité

• Mise en place d’un système d’authentification basé sur JWT.

• Protection des mots de passe avec un mécanisme personnalisé puis bcrypt.

• Utilisation de certificats HTTPS auto-signés via OpenSSL.

### Administration

• Création d’une page d’administration réservée aux administrateurs.

• Recherche et gestion des utilisateurs.

### Protection des données

• Sécurisation contre les injections SQL en limitant l’accès aux bases de données.

• Restriction de l’accès aux profils utilisateurs.

### Documentation

• Versioning du code sur GitHub avec un fichier. gitignore.

• Rapport détaillant la conceptualisation et l’implémentation.

### Technologies utilisées

• Node.js et Express pour le backend.

• Mysql pour la gestion de la base de données.

• JSON Web Token (JWT) pour l’authentification sécurisée.

• OpenSSL pour la gestion des certificats HTTPS.

• bcrypt pour le hachage des mots de passe.

• Docker pour la conteneurisation des services.

## Les points suivants seront évalués

* Le rapport
* Les planifications (initiale et détaillée)
* Le journal de travail
* Le code et les commentaires
* Les documentations de mise en œuvre et d’utilisation

## Validation et conditions de réussite

* Compréhension du travail
* Possibilité de transmettre le travail à une personne extérieure pour le terminer, le corriger ou le compléter
* Etat de fonctionnement du produit livré

# Analyse

## Opportunités

L'implémentation du token JWT a présenté plusieurs complications au cours du développement :

* **Sécurisation des tokens** : Il a fallu s'assurer que les tokens générés ne puissent pas être interceptés et réutilisés par des tiers. L'utilisation de HTTPS a été une solution.
* **Stockage sécurisé des tokens** : Initialement stockés dans le localStorage, les tokens ont finalement été déplacés vers des cookies HTTPOnly.
* **Expiration et renouvellement des tokens** : La gestion de l'expiration des tokens a posé des problèmes, nécessitant la mise en place d'un mécanisme de rafraîchissement via des tokens de rafraîchissement (refresh tokens). (Pas réussi à implémenter)
* **Déconnexion sécurisée** : Assurer que les tokens invalidés ne puissent pas être réutilisés après la déconnexion ou après leur expiration.

# Réalisation

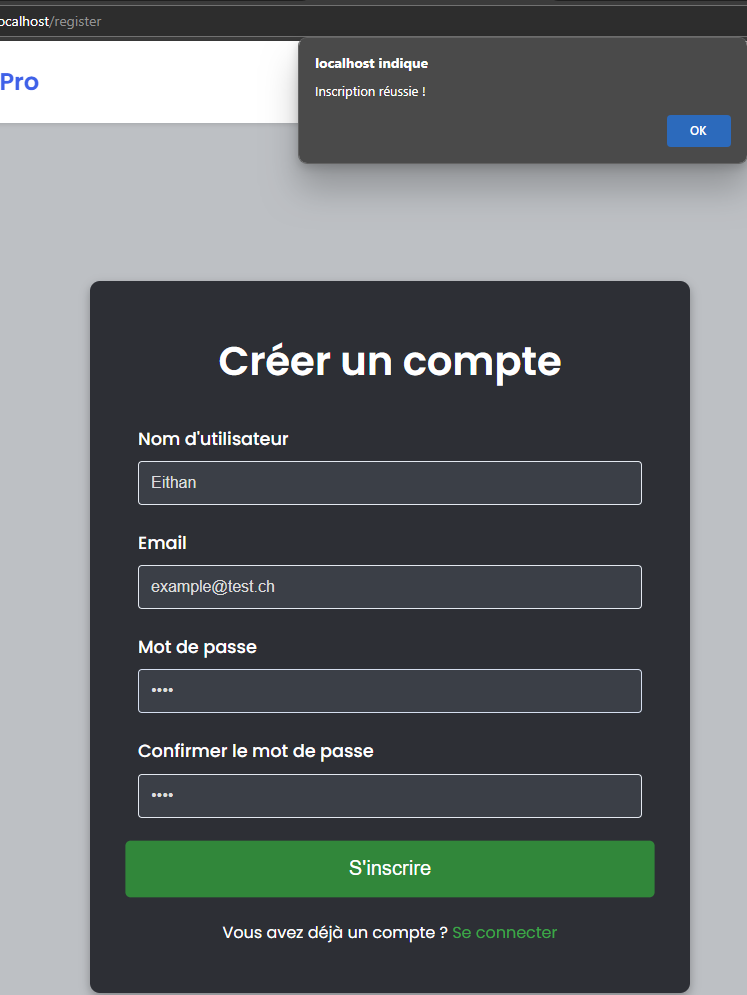
## Dossier de Réalisation

La réalisation du projet s'est développée autour de plusieurs pages clés permettant aux utilisateurs de naviguer de manière fluide et sécurisée sur le site d'e-commerce. L’implémentation s’est faite en utilisant Node.js avec Express pour le backend, MySQL pour la base de données et JSON Web Token (JWT) pour la gestion des sessions et authentification (sécurisé).

### Page d’inscription (Signup)

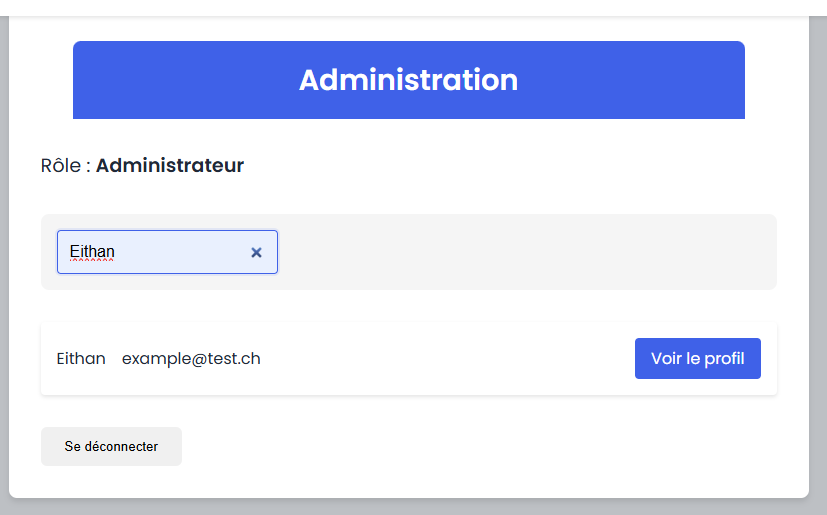
* Formulaire permettant aux utilisateurs de créer un compte.
* Les champs obligatoires incluent : nom, prénom, email, mot de passe.
* Validation des entrées pour éviter les erreurs (format email, etc).
* Hashage des mots de passe avec bcrypt avant stockage en base de données.
* Envoi d’un token JWT après validation pour authentifier l'utilisateur. (Pas eu le temps de faire la connexion immédiate après l’inscription réussie).

### Page de connexion (Login)

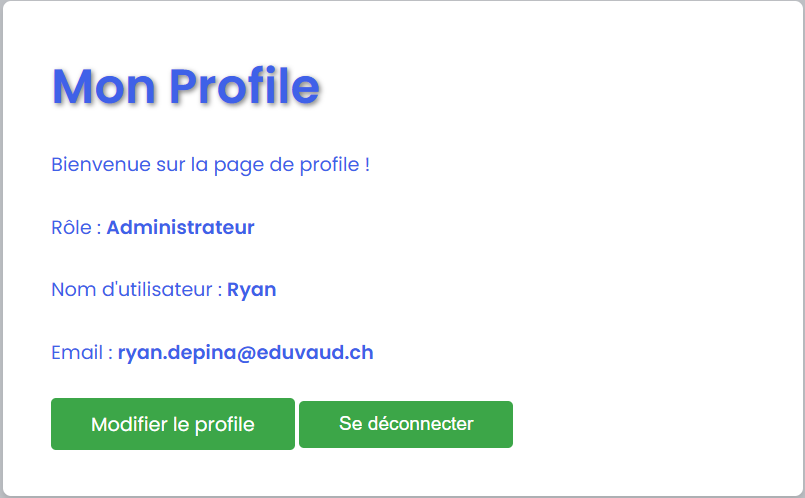
* Formulaire avec nom d’utilisateur et mot de passe pour l’authentification.
* Vérification du mot de passe haché avec bcrypt.
* Génération d’un token JWT en cas de succès.
* Redirection vers la page de profile.
* Gestion des erreurs avec messages d’avertissement en cas d’identifiants incorrects**.**



### Page d’Administration (Admin)

* Accès réservé aux administrateurs authentifiés.
* Tableau de bord permettant de voir et de gérer les utilisateurs.
* Fonctionnalité de recherche des utilisateurs et de pouvoir voir leur page profile.
* Accès limité par rôle avec vérification du token JWT.

### Page de profile (Profile)

* Affichage des informations personnelles de l'utilisateur.
* Possibilité de modifier certains champs (adresse, mot de passe) en changeant dans la db.
* Mise à jour sécurisée avec vérification d'identité.
* Gestion de la déconnexion en supprimant le token JWT.

### Implémentation de SSL pour https

Une image contenant texte, capture d’écran, affichage, logiciel

Description générée automatiquementDans ce code nous pouvons apercevoir comment je me suis pris pour insérer et donner mes keys pour mon serveur https. J’ai récupéré mes deux keys pour ensuite les mettre dans la variable https Server avec comme option pour celui-ci le ssl et toute mon application qui se trouve sous app.js

### Implémentation du token Jwt

Une image contenant texte, Appareils électroniques, capture d’écran, affichage

Description générée automatiquementLe token jwt qui me sert de jeton d’authentification mais aussi de stockeur de valeur pour par exemple is\_admin pour vérifier si la personne est admin ou non pour que le code sache comment afficher le front-end. J’utilise deux clé secrètes qui se trouvent dans le fichier .env et je les utilise pour comme dit pour quand nous nous logons nous puissions naviguer entre les pages tout en restant connecter et quand le token est devenue invalide nous devons nous reconnecter pour en avoir un nouveau.

Une image contenant texte, Appareils électroniques, capture d’écran, affichage

Description générée automatiquementDans ce code qui se trouve sous la fonction Login. La création des tokens « access et refresh » token se font ensuite nous les transmettant dans les cookies avec comme payload l’objet user avec certaine donnée. J’ai fait l’access et le refresh pour apprendre comment refresh le token automatiquement. (Toutefois il y a encore quelque bug et donc n’set pas au point).

# Tests

## Dossier des tests

### Lancement

### Base de données :

### Authentification :

Pour que tout soit implémenter correctement et avec sureté, j’ai mis des condition et fonction pour que chaque chose soit lié à des besoins spécifiques. Par exemple, pour regarder la page de profile, il faut être connecter au site et donc avoir un token valide, sinon il nous renvoie à la page de login. Pour aller voir la page administration qui se trouve sur le header.ejs il faut tout d’abord que dans ton token la valeur « is\_admin = true » sinon il ne l’affiche pas et n’est donc pas disponible.

# Conclusion

## Bilan des fonctionnalités demandées

Le site de e-commerce présente plusieurs fonctionnalités clés côté client, visant à offrir une expérience utilisateur fluide, sécurisée et personnalisée. L'utilisateur a accès à son profil via une interface dédiée, accessible uniquement après connexion, garantissant que ses informations personnelles (nom, adresse, préférences) restent privées et sécurisées. L'authentification se fait par une page de connexion sécurisée via HTTPS, où le mot de passe est haché et salé pour être vérifié en toute sécurité. Un token JWT est généré après la connexion pour maintenir l'utilisateur authentifié lors de ses interactions avec le site, assurant ainsi la sécurité des sessions. L'interface d'administration permet aux administrateurs de rechercher et de gérer les utilisateurs, tout en garantissant la protection contre les injections SQL. Enfin, le mot de passe utilisateur est haché à l'aide de bcrypt avant d'être stocké dans la base de données, offrant ainsi une couche supplémentaire de sécurité. Ce système garantit une expérience utilisateur simple, intuitive et conforme aux meilleures pratiques de sécurité web.

## Bilan de la planification

Pour un projet de cette envergure nous avions très largement le temps si nous avions les compétences nécessaires pour le faire dès le départ.

## Bilan personnel

Il manque dans ce projet, une partie théorie car finalement nous ne savions pas comment commencer et avancer le projet dans sa globalité. Sans les outils comme les IA, les vidéos YouTube durant le projet nous n’aurions pas vraiment le temps de finir le projet à temps.

# Divers

## Webographie

JWT : [jsonwebtoken - npm](https://www.npmjs.com/package/jsonwebtoken)

Chat GPT : [ChatGPT](https://chatgpt.com/) (Prompt : (tu arrives à me faire chaque chapitre pour un rapport mais côté client donc frontend ce qu’il voit des fonctionnalités etc.))