**P\_App WebStore**

**Une image contenant capture d’écran, Graphique, conception

Description générée automatiquement**

**Thibaud Racine – CID2A**

**ETML Sébeillon**

**2023 - 2024**

**40 P**

**Table des matières**

[Introduction 3](#_Toc166049271)

[1.1 Procédure : 3](#_Toc166049272)

[Explication code 3](#_Toc166049273)

[*1.2* CSHTLM 3](#_Toc166049274)

[*1.3* CSS 3](#_Toc166049275)

[1.4 DB 3](#_Toc166049276)

[Autres 3](#_Toc166049277)

[*1.5* Webographie 3](#_Toc166049278)

§

# Introduction

## Procédure :

• Le candidat réalise un travail personnel sur la base d'un cahier des charges reçu le premier jour de l'épreuve.

• Le cahier des charges est commenté et discuté avec l'élève. Par sa signature, l'élève accepte le travail proposé.

• L'élève a connaissance de la grille d'évaluation avant de débuter le travail.

• Il est entièrement responsable de la sécurité de ses données.

• En cas de problèmes graves, le candidat avertit son chef de projet au plus vite.

• Le candidat a la possibilité d’obtenir de l’aide externe et doit le mentionner dans son rapport final.

• Les projets sont réalisés seul

• Le projet est évalué selon la grille d’évaluation.

• Les apprentis ont de la semaine 1 à la semaine 5 pour travailler et finir le projet.

• A la semaine 5 du projet, les apprentis devront fournir leur projet et leur auto-évaluation.

• Lors des semaines 6 et 7, les apprentis seront en remédiation du projet ou en approfondissement et devront rendre leur présentation PowerPoint au chef de projet (semaine 7).

• Le trimestre 4 dure 9 semaines, mais le temps effectif du projet est de 7 semaines

(Vendredi Saint et Ascension). Le projet compte pour 32p.

Utilisation de deux containers :

|  |
| --- |
|  |

# Projet

## Phase 1

**Mise en place de l’environnement du projet.**

* Création journal de travail et rapport de projet
* Installation de tous les packages (npm etc.)
* Docker compose
* Update package
* Création du Github

## Phase 2

**Code.**

* Utilisation de Open SSL pour avoir des pages en **HTTPS** dans le **server.mjs** :

|  |
| --- |
|  |

Pour utiliser HTTPS, le serveur utilise un certificat SSL (**certificate.crt**) et une clé privée (**private.key**). Lorsqu'une requête arrive, elle est traitée de manière sécurisée grâce à HTTPS qui chiffre les données.

## Phase 3

Création d’un formulaire de login en html pour se login avec les différents user créés. J’ai créé une bête page html pour pouvoir se login.

|  |
| --- |
|  |

Grâce à ce code, Il est possible de se connecter avec les user stockés dans le **mockUsers.mjs** pour faire des tests. Si l’utilisateur existe, et le mot de passe est juste, l’utilisateur sera redirigé sur une page comme : *« https://localhost/users/bob »*, Sinon une boîte de dialogue s’ouvrera avec une erreur.

J’ai aussi créé un utilisateur « Admin » qui quand se connecte peut voir tous les utilisateurs de **mockUsers.mjs**

## Phase 4

**Problèmes**

Jusqu’à maintenant tout fonctionne. Mais le problème est la sécurité. Si on change le nom dans l’url *« https://localhost/users/bob »* par un autre nom comme « Alice » ou « Admin » on pourra avoir accès à leurs profils sans même savoir leur mot de passe. Problème connu sous le nom de « Broken Access Control ». Pour remédier à ce problème il faut utiliser des token d’identification.

Il faut aussi se débarrasser du « mockUsers.mjs » et avoir des utilisateurs dans une vraie base de données.

## Phase 5

# Autres

## Webographie