**Développer la partie back-end d’une application web ou web mobile sécurisée**

**(CCP2)**

**Portfolio**

**Sommaire**

**Introduction**

**Objectif du projet :**

Développer une application MERN (MongoDB, Express, React, Node.js) permettant de gérer et d'afficher des compétences via un portfolio dynamique. L'application doit être sécurisée, responsive (mobile-first) et conforme aux exigences RGPD, incluant un système de gestion des cookies (Tarteaucitron.js) et un Captcha (Google reCAPTCHA ou Tarteaucitron).

**Compétences évaluées et technologies utilisées :**

1. **Maquetter des interfaces utilisateur :**
   * **Outils :** Figma, Adobe XD
   * **Bonnes pratiques UX/UI :**
     + Design mobile-first
     + Navigation intuitive entre le portfolio et le Dashboard
     + Affichage des compétences sous forme de cartes interactives
   * **Astuce :** Utilisation de Flexbox ou Grid pour un design fluide.
2. **Base de données relationnelle (MongoDB + Mongoose) :**
   * **Collections MongoDB :**
     + users: nom, email, mot de passe (haché avec bcrypt), rôle (admin/user)
     + skills: titre, catégorie, niveau (débutant, intermédiaire, expert), image (Cloudinary)
     + settings: préférences utilisateur, choix des cookies
   * **Astuce :** Indexation des champs fréquents pour optimiser les requêtes.
3. **Composants d’accès aux données (NoSQL) :**
   * **Backend :** Express + MongoDB
   * **API REST (CRUD) :** /api/skills
   * **Upload d'images :** Cloudinary via Multer
   * **Authentification sécurisée :** JWT & bcrypt
   * **Astuce :** Utilisation de httpOnly cookies pour protéger le JWT des attaques XSS.
4. **Composants métier côté serveur :**
   * **Sécurité :**
     + helmet pour protéger les en-têtes HTTP
     + cors pour gérer les accès cross-origin
     + Tarteaucitron.js pour la gestion des cookies (RGPD)
     + reCAPTCHA ou Tarteaucitron CAPTCHA pour bloquer les bots
     + Middleware de rôles (admin, utilisateur) pour sécuriser le Dashboard
   * **Astuce :** Utilisation de Winston ou Morgan pour le logging des actions utilisateurs.
5. **Documentation du déploiement :**
   * **README.md** détaillant :
     + Installation et exécution du projet
     + Déploiement backend (Render, Railway, Heroku)
     + Déploiement frontend (Vercel, Netlify)
     + Collection Postman pour tester l’API
   * **Astuce :** Création d'un schéma d'architecture pour clarifier l'organisation du projet.
6. **Interfaces utilisateur statiques et dynamiques (React) :**
   * **Frontend :** React avec approche Mobile-First & Responsive
   * **Fonctionnalités :**
     + Portfolio dynamique pour afficher les compétences
     + Dashboard sécurisé pour gérer les compétences (ajout, modification, suppression)
     + Formulaire de connexion avec reCAPTCHA
     + Gestion des cookies avec Tarteaucitron.js
     + Utilisation de React Router et Axios pour la navigation et les requêtes
     + Mise en page responsive avec Tailwind CSS ou Bootstrap
   * **Astuce :** Utilisation de React Query ou Redux Toolkit pour optimiser les requêtes.

**Livrables attendus :**

* Code source hébergé sur GitHub avec un **README.md** clair
* **Rapport de projet** expliquant les choix techniques
* **Diaporama de présentation** pour la soutenance
* **Collection Postman** pour les tests API

**Oganisation et suivi**

Pour l’organisation des tâches je me suis aidé de l’outil en ligne Trello :

Screens screenshot of a computer

AI-generated content may be incorrect.

*Extrait du trello*

J’ai hébergé mon projet sur github et je l’ai envoyé via commande terminal git

A black cat in a circle

AI-generated content may be incorrect.*github & git logos* A black and orange sign

AI-generated content may be incorrect.

Github est un service d’hebergement cloud pour le code. Il sert à gérer le code et à faire du versionning

Git est un système de contrôle de version, il permet une collaboration fluide et un contrôle des versions et des contributeurs.

**Développement**

Pour réaliser ce projet j’ai eu besoin de plusieurs outils :

A blue ribbon with a black background

AI-generated content may be incorrect.

Pour développer j’ai utilisé Visual Studio Code car je l’utilise depuis plusieurs années et je me sens à l’aise avec.

Il y a un grand choix d’extensions qui m’aident beaucoup dans mes projets.

De plus VSCode simplifie pas mal le lien avec github pour ceux qui ont du mal avec les commandes git sur le terminal.

A green leaf on a black background

AI-generated content may be incorrect.

La base de donnée utilisée est MongoDB.

Nous l’avions déjà utilisé de nombreuses fois en cours et je me sentais assez à l’aise pour la réutiliser aisément.

A black background with a black square

AI-generated content may be incorrect.

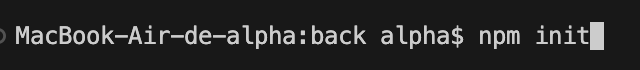
Pour le déploiement du backend j’ai utilisé Render, hébergeur que nous avions déjà utilisé lors de la formation et qui nous a été recommendé par notre formateur.

A black background with a black square

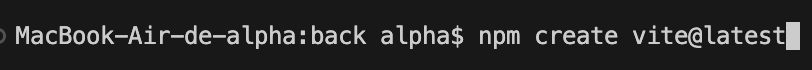
AI-generated content may be incorrect.

Quant au frontend il a été déployé sur Vercel. Idem que Render, nous l’avions déjà utilisé au cours de la formation.

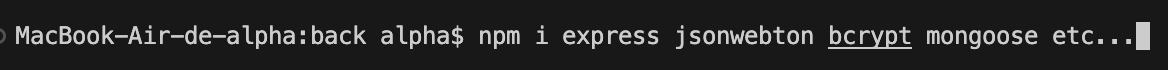
Tout d’abord il faut initier le projet node pour le backend avec la commande `npm init`



Ensuite il faut initier le projet vite react pour le frontend à la racine du projet avec la commande ` npm create vite@latest`



Après ceci il faut installer les dépendances néxessaires dans chaque parties du projet (front et back) avec la commande `npm i <le nom des dependances>`



Une fois les dépendances installées il faut créer l’arborescence du projet

A screenshot of a phone

AI-generated content may be incorrect..A screenshot of a phone

AI-generated content may be incorrect.

**Base de données**

Pour la base de donnée j’ai crée un compte sur Mongo Atlas puis j’ai récupéré le MONGO\_URI que j’ai ajouté dans le fichier .env qui sert à avoir des variables d’environnement qui peuvent être des données sensibles qui ne doivent pas êtres publiques.

A black and white sign with white text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Ensuite j’ai configuré la connexion à base de données avec mongoose dans un fichier db.js

A screen shot of a computer code

AI-generated content may be incorrect.

Pour finir j’ai lancé la connexion de la base de donnée

A black background with white text

AI-generated content may be incorrect.

A black background with white text

AI-generated content may be incorrect.

A black background with orange text

AI-generated content may be incorrect.

**CRUD** (create, read, update, delete)

Dans cette application il y a deux CRUD différents, un pour les utilisateurs et l’autre pour les compétences.

Dans un premier temps j’ai crée la gestion d’authentification avec :

* Création d’utilisateur avec hachage de mot de passe pour la sécurité (bcrypt)

A screen shot of a computer program

AI-generated content may be incorrect.

A computer code with numbers and letters

AI-generated content may be incorrect.

* Récupération de tous les utilisateurs pour les afficher

A screen shot of a computer code

AI-generated content may be incorrect.

* Modification d’un utilisateur via son id

A screen shot of a computer code

AI-generated content may be incorrect.

* Suppression d’un utlisateur via son id

A screen shot of a computer program

AI-generated content may be incorrect.

* Connexion d’utilisateur et génération d’un token pour vérifier son identité

A computer screen with text

AI-generated content may be incorrect.

**Documentation du projet**

Pour la documentation j’ai utilisé un Readme que j’ai placé à la racine du projet. Dans celui-ci j’y ai indiqué plusieurs informations qui sont les suivantes :

* Le sujet
* Les fonctionnalités
* Les technologies utilisées
* Les prérequis
* Comment l’installer, le configurer et le lancer
* Les dépendances installées
* Des identifiants d’un adminisrateur pour effectuer les tests
* L’arborescence du projet
* La collection postman pour les tests des routes
* Et les déploiements front-end et back-end

**Conclusion**

En résumé, ce projet incorpore des fonctionnalités robustes et des technologies modernes pour offrir une solution efficace et performante. Grâce à des prérequis bien définis et une documentation détaillée sur l’installation, la configuration et le lancement, les utilisateurs et administrateurs peuvent naviguer et tester aisément le système. L'intégration des dépendances essentielles et la fourniture d’identifiants administrateurs garantissent un testing complet et efficace. Avec une arborescence de projet bien structurée et une collection Postman dédiée, chaque route est testée avec précision. Enfin, les déploiements front-end et back-end assurent une flexibilité et une adaptabilité maximales. Ce projet promet donc une implémentation fluide et un fonctionnement optimal.