

**Développeur Web et Web Mobile**

**Prénom : Charly**

**Nom : Makhlouf**

**ATTENTION ! PENSEZ À RENSEIGNER VOS NOM ET PRÉNOM DANS LE TITRE DE VOS FICHIERS / PROJETS !**

Nom du projet : “API” Salle de sport

Lien Github du projet : <https://github.com/PaleGreenFILE/ECF-API-SPORT/tree/main>

**Attention ! Merci de bien classer vos documents dans votre Github ou votre drive.**

URL du site (si vous avez mis votre projet en ligne) : <https://ecf-2022.web.app/>

* ***Liste des compétences du référentiel qui sont couvertes par le projet***

***Front-End :***

1 - Maquetter une application

2 - Réaliser une interface utilisateur web statique et adaptable

3 - Développer une interface utilisateur web dynamique

4 - Réaliser une interface utilisateur avec une solution de gestion de contenu ou ecommerce

***Back-End:***

5 - Créer une base de données

6 - Développer les composants d’accès aux données

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

8 - Elaborer et mettre en œuvre des composants dans une application de gestion de

contenu ou e-commerce

------------------------------------------------------------------

* ***Résumé du projet en français d’une longueur d’environ 20 lignes soit 200 à 250 mots, ou environ 1200 caractères espaces non compris***

Dans le cadre de ma Formation Développeur Full stack en ligne via la plateforme STUDI, j'ai réalisé une application de salle de sport pour mettre en place un panel d'administration des partenaires et structure de la marque FitPark Fitness.

FitPark Fitness est une application WEB permettant une administration complète des Partenaires et Structures liés a la marque. Le back-office est accessible par les administrateurs, Partenaires et structures. Seuls les administrateurs peuvent modifier, créer, activer, désactiver , ajouter des permissions ou supprimer des Partenaires ou Structures. Les partenaires et structures on seulement un accès a la plateforme en mode lecture only.

Pour ce projet, nous avions que la partie administration a créé , mon choix pour la mise en place du projet s'est porté sur le framework REACT 18.2.0. Pour la partie back-end mon choix s'est porté sur Node.js, pour la base de données RGPD j'ai décidé d'utilisé POSTGRESQL hébergée sur ElephantSql .

Pour stylisé le tout j'ai utilise TAILWINDCSS.

Pour effectuer les relations entre la base de données et les requêtes entre le Front-End et le Back-end, j'ai mis en place une API avec Express qui est simple et rapide et qui fournit une performance robuste de fonctionalités pour les applications Web et mobiles . Enfin pour finir j'ai utilisé Firebase pour ça facilité de déploiement de mon application coté Front-End. Pour le back-end le service cloud-function de Firebase.

Dans un second temps, il m'as fallu mettre en place un système d'envoi d'email pour que le partenaire ou la structure puisse recevoir un lien qui leur permettrait de changer leur mot de passe avant leurs toutes premières connexions et qui sera mis à jour en temps réel dans la base de données.Il y avait aussi un email pour souhaiter la bienvenue avec un bouton de redirection vers le panel de connexion et un dernier email étaient envoyer aux partenaires et à la Structure liée lors de la création de celle-ci ainsi que pour toute modification des persmissions.

Enfin, pour sécuriser l'application, j'ai utilisé Bcrypt pour crypter les mots de passe, mis en place d'une connexion par rôle (Admin , Partenaires, Structure) ne pouvant voir que leur contenu ainsi que de la mise en place du JWT Token lors de l'authentification qui pourront donnés l'accès aux ressources.

Puis sur les formulaires une validation des inputs grâce a une libraire qui s'appelle React-Hook-Form

qui permet d'avoir des formulaires performants et une validation facile à utiliser permettant ainsi aussi d"eviter des attaques par injections en limitant le nombre de caractères et en utilisant des regex pour l'email et mot de passe.

--------------------------------------------------------------

* ***Cahier des charges, expression des besoins, ou spécifications fonctionnelles du projet***

**Besoin du client :**

- Le client souhaite avoir une application qui permettrait de gérer des droits d'accès pour les franchiser de la marque possède une ou plusieurs salle de sports.

**Les fonctionnalités:**

***- US1: Gérer les partenaires :***

Les utilisateur concérnés sont l'equipe technique de la marque.

Tous les partenaires qui auront signé un contrat devra être ajoutée dans l'application web et ne pourra s'effectuer que par l'administrateur (équipe technique).

Par contre nous voudrions pouvoir activer ou désactiver tout partenaire ou structures sans la supprimer de la base de données et si c'est une structure (franchise) qu'elle soit toujours répertoriée dans la liste du partenaire associé.

Pour tout enregistrement d'un partenaire ou d'une structure il sera demandé le nom ainsi qu'une adresse email qui par la suite permettra de créer son compte avec son adresse email et un mot de passe temporaire sécurisé.

***- US2: Gérer les structures :***

L'équipe technique(Administrateur) et le partenaire sont concernés.

Tous les partenaires de la marque pourront avoir 1 à plusieurs structures.

Il sera demandé une adresse postale et un email de contact du gérant pour chaque structure.

Elles pourront être désactivées à tout moment en un seul click.

Même chose que précédemment toute structure créée sera effectuée avec l'email du gérant et d'un mot de passe temporaire sécurisé.

***- US3: Les permissions :***

L'équipe technique(Administrateur), le partenaire et la structure seront concernés.

Toutes les permissions existantes qui seront disponibles pour une structure selon le contrat signé

pour être validée par une action booléenne à True pour dire oui vous avec accès à ce module ou à False pour dire non vous n'avez pas accès.

Toutes les permissions ne pourront être activées que par l'équipe technique, tout partenaire ou structure auront un accès en lecture seul et ne pourront modifier les permissions.

***- US4: Connexion :***

L'équipe technique(Administrateur), le partenaire et la structure seront concernés.

Pour se connecter à l'application le partenaire et la structure devront avoir un compte actif;

à ce moment il pourra se connecter avec son adresse email et son mot de passe sécurisé qu'il aura préalablement changé lors de sa première connexion.

Une fois la connexion réussie, il sera redirigé sur son dashboard ou il pourra voir ses permissions actives ainsi que ses structures et les permissions de celle ci .

De même que pour une structure active, si elle se connecte elle pourra aussi voir ses permissions accordées active ou non.

***US4.1: La notification par email***

L'équipe technique(Administrateur), le partenaire et la structure seront concernés.

Il existe plusieurs types d'emails qui seront envoyés par l'application .

1- lorsqu'un partenaire ou une structure sera créé. Ils recevront chacun un email de bienvenue avec un lien qui les redirigera vers une page pour changer leurs mots de passe avant toute première connexion.

2- Un email sera envoyé lorsque qu'une ou plusieurs permissions ont été changées

Si c'est une permission d'une structure qui est changée alors le partenaire recevra également un email pour prévenir d'un changement de permsission de l 'une de ces structures.

3- Un email de confirmation de mot de passe changer avec un lien de redirection vers la page de connexion.

***- US5: La confirmation de sécurité***

Seul l'équipe technique(Administrateur) est concernés.

Pour chaque modification ou suppression ou d'ajout d'un partenaire, structures un message s'affichera sur l'application demandant une confirmation pour l'action désirer. S'il ne souhaite pas continuer alors il pourra cliquer sur la croix ou le bouton annuler directement et l'action demandée sera aussitôt annulé.

***- US6: La recherche dynamique***

L'équipe technique(Administrateur), le partenaire.

En haut de l'application il y a une barre de recherche qui permettra à chaque administrateur et partenaire de faire une recherche simple et rapide sans rechargement de la page de trouver un partenaire si administrateur faisant la recherche de retrouver un partenaire en tapant les premières les de son nom, par son id, et par son rôle dans la liste qui presente tous les partenaires et structures, il pourra aussi trier la liste par partenaires, structures actif ou désactiver.

Pour le partenaire , il pourra effectuer aussi une recherche des ses structures avec le même procer que pour l'administrateur.

------------------------------------------------------------------------------

* ***Spécifications techniques du projet, élaborées par le candidat, y compris pour la sécurité et le web mobile***

- L'application sera hébergée par Firebase Hosting pour le coter Front-End et par Firebase-Function qui sera un serveur node pour héberger la partie Back-end, pour optimiser le tout et d'avoir une application stable , robuste et qui fournit un hébergement rapide et sécurisé pour notre application Web.

- Pour le coter back-end j'ai choisi Cloud Functions de Firebase permettant d'exécuter automatiquement du code backend en réponse aux requêtes HTTPS, le code est stocké dans le cloud de Google et s'exécute dans un environnement géré.

- Firebase permet une protection de notre domaine des menaces exterieures en utilisant une certification SSL gratuite , Firebase nous permet aussi grace à leur CLI de mettre en ligne et d'executer à l'aide de quelques lignes de commande . l'hebergement est soutenu par des SSD et des CDN qui permettent la diffusion du contenu trés rapidement.

- Pour la partie administration du site Web j'ai décidé d'utiliser Tailwind CSS pour une mise en place rapide et d'ecrire un code plus performant et rapide sans quitter mon code principal.

- Pour la validations, erreurs des Formulaires j'ai utilisé la libraries Reac-Hook-Form qui me permet a l'aide de quelque ligne de gagner du temps sur la rédaction du code aussi et pour de meilleure performance et de sécurité.

- Pour la base de données je suis parti sur PostGreSql qui est heberge et mis en ligne par ElephantSql

qui est parfaitement configurées et optimisées & prêtes en 2 minutes.

la plateforme nous permet d'avoir des sauvegardes automatisées qui sont effectuées tous les jours, qui sont stockées dans un stockage de fichiers cloud afin qu'elles soient toujours accessibles. Nous pouvez également utiliser la récupération ponctuelle pour restaurer notre base de données.

---------------------------------------------------------------------------

* ***Description de la veille, effectuée par le candidat durant le projet, sur les vulnérabilités de sécurité***

***Bcrypt :***

j'ai décidé d'utiliser Bcrypt pour hash le mot de passe lors de l'enregistrement d'un partenaire ou d'une structure avant d'être enregistré en base de données.

***React-Hook-Form:***

j'ai utilisé ceci pour la manipulation des formulaires. cela me permet d'écrire moins de code tout en obtenant de meilleures performances ainsi que d'avoir une validation des inputs, permettant ainsi d'ajouter des limitations dans les inputs de mot de passe en limitant le nombre de caractére, obligeant de mettre une majuscule , un chiffre etc et d'avoir des entrèes obligatoire pour valider l'envoi du formulaire sinon rien ne pourra être envoyé.

***Jwt-Web-Token:***

Ceci me permet de valider si la personne est bien authentifier et que toutes les fonctionalités du site comme Ajouter , supprimer , modifier un parteanire ou une structure ne pourra avoir lieu que si l'utilisateur est bien authentifier et le token est verifier.

***PostGresSql:***

Création d'un script sql pour la création des Tables pour le projet.

------------------------------------------------------------------

* ***Description d’une situation de travail ayant nécessité une recherche, effectuée par le candidat durant le projet, à partir de site anglophone + Extrait du site anglophone, utilisé dans le cadre de la recherche décrite précédemment, accompagné de la traduction en français effectuée par le candidat sans traducteur automatique (environ 750 lignes).***

- Je me suis documentée sur le principe du JSON Web Token (ou JWT) et de Bcrypt pour le hashage du mot de passe.

***Voici un extrait du site Bcrypt, ainsi que sa traduction.***

***Security considerations***

Besides incorporating a salt to protect against rainbow table attacks, bcrypt is an adaptive function: over time, the iteration count can be increased to make it slower, so it remains resistant to brute-force search attacks even with increasing computation power. (see)

While bcrypt.js is compatible to the C++ bcrypt binding, it is written in pure JavaScript and thus slower (about 30%), effectively reducing the number of iterations that can be processed in an equal time span.

The maximum input length is 72 bytes (note that UTF8 encoded characters use up to 4 bytes) and the length of generated hashes is 60 characters.

***Considérations de sécurité :***

En plus d'incorporer un sel pour se protéger contre les attaques de table en arc-en-ciel, bcrypt est une fonction adaptative : au fil du temps, le nombre d'itérations peut être augmenté pour le rendre plus lent, de sorte qu'il reste résistant aux attaques de recherche par force brute même avec une puissance de calcul croissante.

Bien que bcrypt js soit compatible avec la liaison bcrypt C++, il est écrit en Javascript pur et donc plus lent ( environ 30 % ), ce qui réduit efficacement le nombre d'itérations pouvant être traitées dans un laps de temps égal.

La longueur d'entrée maximum est de 72 octets (notez que les caractères codés UTF8 utilisent jusqu'à 4 octets) et la longueur de hachage générée est de 60 caractères.

***Voici un extrait du site officiel jwt.io, ainsi que la traduction.***

What is JSON Web Token?

JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA or ECDSA.

Although JWTs can be encrypted to also provide secrecy between parties, we will focus on signed tokens. Signed tokens can verify the integrity of the claims contained within it, while encrypted tokens hide those claims from other parties. When tokens are signed using public/private key pairs, the signature also certifies that only the party holding the private key is the one that signed it.

When should you use JSON Web Tokens?

Here are some scenarios where JSON Web Tokens are useful:

Authorization: This is the most common scenario for using JWT. Once the user is logged in, each subsequent request will include the JWT, allowing the user to access routes, services, and resources that are permitted with that token. Single Sign On is a feature that widely uses JWT nowadays, because of its small overhead and its ability to be easily used across different domains.

Information Exchange: JSON Web Tokens are a good way of securely transmitting information between parties. Because JWTs can be signed—for example, using public/private key pairs—you can be sure the senders are who they say they are. Additionally, as the signature is calculated using the header and the payload, you can also verify that the content hasn't been tampered with.

------------------

- En quoi consiste un jeton Web JSON? JSON Web Token (JWT) est une norme ouverte (RFC 7519) qui définit un moyen compact et autonome pour transmettre en toute sécurité des informations entre les parties en tant qu'objet JSON.

- Ces informations peuvent être vérifiées et approuvées car elles seront signées numériquement. Les JWT peuvent être signés à l'aide d'un secret (avec l'algorithme HMAC) ou d'une paire de clés publiques/privées utilisant RSA ou ECDSA. Bien que les JWT puissent être chiffrés pour assurer également le secret entre les parties, nous nous concentrerons sur les jetons signés. Les jetons signés peuvent vérifier l'intégrité des revendications qu'ils contiennent, tandis que les jetons chiffrés cachent ces revendications aux autres parties.

- Lorsque les jetons sont signés à l'aide de paires de clés publique/privée, la signature certifie également que seule la partie détenant la clé privée est celle qui l'a signée.

***Quand utiliser les jetons Web JSON ?***

- Voici quelques scénarios où les jetons Web JSON sont utiles :

- Autorisation : il s'agit du scénario le plus courant pour l'utilisation de JWT. Une fois l'utilisateur connecté, chaque demande ultérieure inclura le JWT, permettant à l'utilisateur d'accéder aux routes, services et ressources autorisés avec ce jeton.

- L'authentification unique est une fonctionnalité qui utilise largement JWT de nos jours, en raison de sa faible surcharge et de sa capacité à être facilement utilisée dans différents domaines.

Échange d'informations :

- les jetons Web JSON sont un bon moyen de transmettre en toute sécurité des informations entre les parties. Étant donné que les jetons JWT peuvent être signés, par exemple à l'aide de paires de clés publique/privée, vous pouvez être sûr que les expéditeurs sont bien ceux qu'ils prétendent être. De plus, comme la signature est calculée à l'aide de l'en-tête et de la charge utile, vous pouvez également vérifier que le contenu n'a pas été falsifié.