CAHIER DES CHARGES BACKEND

Projet : Plateforme de recherche et de mise en relation de profils

---

1 Présentation générale du projet

1.1 Contexte

De nombreux profils compétents, dans la tech comme hors tech, ne sont pas visibles sur les réseaux professionnels classiques. Pourtant, ils sont disponibles, motivés, et ouverts à des opportunités de collaboration, d'emploi ou de projets.

Le projet vise à créer une **base de données centralisée de profils**, consultable via une API backend, permettant une **recherche avancée et ciblée** selon des critères précis.

1.2 Objectif du backend

Développer une **API robuste, évolutive et professionnelle**, permettant :

- la gestion d'une base de profils structurés
- la recherche multicritère avancée
- l'accès direct à un moyen de contact
- la consultation du statut de disponibilité des profils

Aucune interface utilisateur n'est prévue dans la première phase.

---

2 Périmètre du projet (scope)

2.1 Inclus

- API REST backend
- gestion des profils
- recherche avancée avec filtres combinables
- stockage et structuration des données
- documentation technique de l'API
- architecture propre et maintenable
- projet open source (GitHub)

2.2 Exclus

- interface frontend
- système d'authentification utilisateur
- création ou modification de profil par l'utilisateur final
- messagerie interne
- notifications

---

3 Description fonctionnelle (backend)

3.1 Concept de "profil"

Un profil représente une **entrée de données**, et non un compte utilisateur.

Chaque profil est créé, modifié ou supprimé via l'API (usage interne ou administré).

3.2 Données d'un profil

Champs obligatoires

- identifiant unique
- prénom ou pseudo
- pays
- ville
- domaine (tech / non-tech)
- statut de disponibilité (disponible / non disponible)
- moyen de contact (email ou WhatsApp)

Champs optionnels

- tranche d'âge
- genre
- technologies / compétences
- description courte
- date de création
- date de mise à jour

3.3 Gestion des profils

Fonctionnalités backend :

- création d'un profil
- consultation d'un profil
- mise à jour d'un profil
- suppression d'un profil
- listing paginé des profils

3.4 Recherche avancée

Fonction centrale du système.

L'API doit permettre de :

- filtrer par pays
- filtrer par ville
- filtrer par domaine
- filtrer par technologies
- filtrer par tranche d'âge
- filtrer par genre
- filtrer par statut de disponibilité
- combiner plusieurs filtres simultanément

Résultat attendu :

- liste structurée de profils
- pagination
- ordre cohérent (ex: date, pertinence)

---

4 Architecture technique

4.1 Stack backend (proposée)

- Langage : JavaScript
- Runtime : Node.js
- Framework : Express ou NestJS
- Base de données : PostgreSQL (évolutive)
- ORM : Prisma ou équivalent
- Format d'échange : JSON
- Gestion de versions : Git
- Hébergement : non défini à ce stade

4.2 Architecture logique

Organisation recommandée :

- controllers: gestion HTTP
- services: logique métier
- repositories: accès aux données
- middlewares: validations, erreurs
- routes: définition des endpoints

Objectif :

- séparation claire des responsabilités
- lisibilité
- maintenabilité
- évolutivité

---

5 API REST – Spécifications générales

5.1 Principes

- respect des conventions REST
- endpoints clairs et cohérents
- réponses HTTP normalisées
- gestion centralisée des erreurs
- validation des données en entrée

5.2 Exemples d’actions backend

- créer un profil
- récupérer un profil par identifiant
- rechercher des profils selon critères
- mettre à jour le statut de disponibilité
- supprimer un profil

---

6 Contraintes techniques

- API sans interface graphique
- aucune dépendance frontend
- code documenté et lisible

- gestion propre des erreurs
- données cohérentes et validées
- projet prêt à accueillir de futures extensions

---

7 Sécurité (niveau initial)

Dans cette première version :

- validation stricte des données entrantes
- protection contre les données incohérentes
- pas d'authentification utilisateur complexe
- possibilité future d'ajouter des rôles (admin)

---

8 Évolutivité prévue

Le backend doit pouvoir évoluer vers :

- ajout d'authentification
- interface frontend
- gestion des contributions externes
- indexation avancée
- moteur de recherche plus performant
- statistiques et analytics

---

9 Livrables attendus

- dépôt GitHub open source
- code backend structuré
- documentation API (README)
- exemples de requêtes
- instructions de lancement local

Conclusion

Ce projet backend a pour vocation de servir :

- de **produit réel**
- de **support pédagogique**

- de **base communautaire évolutive**

Il doit refléter les bonnes pratiques du métier de développeur backend JavaScript, tout en restant accessible et pragmatique.