Cahier des charges — MirrorMind

1. Contexte

MirrorMind est une application web visant à améliorer le bien-être mental via des exercices guidés, capture vidéo et suivi utilisateur. Le projet est développé en React (Vite) avec intégration de modèles ML légers (TFJS/MediaPipe) et stockage/authentification via Supabase.

2. Objectif général (MVP)

Livrer une application web complète, responsive et déployable, permettant :

- Authentification utilisateur (inscription/connexion).
- Dashboard avec historique et indicateurs de session.
- Module d'exercice guidé (respiration).
- Module caméra + détection basique (TFJS/MediaPipe) fournissant feedback en temps réel.
- Persistance de sessions (Supabase).

3. Périmètre fonctionnel

Fonctions obligatoires (MVP) :

- Auth : inscription, connexion, réinitialisation mot de passe.
- Dashboard : affichage sessions, graphiques simples (recharts).
- Exercice respiration : guide visuel / temporisateur / progression.
- Caméra : accès navigateur, détection visage/pose minimale, feedback visuel.
- Stockage : enregistrement métadonnées sessions dans Supabase.
- UI : thèmes (clair/sombre), responsive, accessibilité de base.

Fonctions optionnelles (phase ultérieure) :

- Recommandations personnalisées, analytics avancées, export données.

4. Contraintes non-fonctionnelles

- Conformité RGPD (consentement audio/vidéo, suppression données).
- Latence temps-réel pour feedback ≤ 200 ms dans conditions normales.
- Sécurité : HTTPS, stockage sécurisé des tokens, protection endpoints.
- Qualité : tests unitaires et d'intégration, CI (lint + tests + build).
- Accessibilité : navigation clavier, labels ARIA.

5. Technologies retenues

- Frontend : React 18, TypeScript, Vite, Tailwind CSS, Radix UI.
- Vision/ML client : @tensorflow/tfjs, MediaPipe.
- Backend / Auth / Storage : Supabase.
- Tests : React Testing Library / Vitest ou Jest.
- CI/CD : GitHub Actions → déploiement Vercel/Netlify.

6. Architecture (haut niveau)

- Client web (React) :
  - Pages : Auth, Dashboard, Exercice, Caméra.
  - Services : API Supabase, ML Service (TFJS), stockage local (IndexedDB).
- Backend Hébergé : Supabase (auth, base sessions, storage).
- Pipelines CI : build + tests + déploiement.

7. Critères d'acceptation

- Inscription/connexion testées et fonctionnelles.
- Dashboard affichant des données (réelles ou factices) et graphiques.
- Exercice respiration interactif et mesurable.
- Caméra détecte visage/pose pour la plupart des cas simples (≥80%).
- Tests unitaires critiques passent et build de production validé en CI.
- Documentation d’installation et URL de démo en production.

8. Livraison & planning (suggestion 8 semaines)

- Semaine 1–2 : Setup projet, architecture, auth.
- Semaine 3–4 : Dashboard, tests unitaires.
- Semaine 5–6 : Module respiration (UI + tests).
- Semaine 7–8 : Module caméra + ML, QA, CI, déploiement.

9. Livrables

- Repo Git avec code, README et CAHIER_DES_CHARGES.md.
- URL de démonstration (Vercel/Netlify).
- Documentation d’installation et tests.

- Rapport technique + capture vidéo de la démo.

10. Risques et atténuations

- Problèmes permission caméra → tests multi-navigateurs, messages clairs.

- Performance ML → modèles légers et optimisation frames.

- Données personnelles → consentement explicite et anonymisation.