Cahier des Charges - Application Web SnowBrain AI

SnowDev

17 septembre 2025

Table des matières

1 CAHIER DES CHARGES - APPLICATION WEB

2 PRÉSENTATION GÉNÉRALE DU PROJET

2.1 Contexte et enjeux business

L'application SnowBrain AI est une plateforme innovante d'apprentissage interactif basée sur l'intelligence artificielle. Dans un contexte où l'éducation en ligne et l'apprentissage personnalisé sont en pleine expansion, SnowBrain AI vise à démocratiser l'accès à des sessions de tutorat vocal en temps réel avec des agents IA. Les enjeux business incluent la monétisation via des abonnements et des paiements par session, la fidélisation des utilisateurs par une expérience immersive, et la concurrence avec des plateformes comme Duolingo ou Khan Academy, mais avec un focus sur l'interaction vocale. Le marché de l'EdTech est estimé à plus de 250 milliards de dollars d'ici 2025, offrant un potentiel de croissance significatif.

2.2 Objectifs de l'application web

Les objectifs principaux sont :

Permettre aux utilisateurs de créer des "Brains" personnalisés pour des sessions d'apprentissage vocal avec une IA. Offrir une interface intuitive pour sélectionner des sujets (ex. : mathématiques, codage, langue, sciences, histoire, economics), des voix (masculine/féminine), des détails spécifiques et la durée des sessions. Intégrer des appels vocaux en temps réel via l'API VAPI, avec possibilité d'interruption à tout moment(raccrocher). Gérer l'authentification, la facturation et les données via Clerk et Supabase. Assurer une scalabilité pour supporter un nombre croissant d'utilisateurs.

2.3 Cible utilisateurs (personas)

Étudiant adolescent/jeune adulte : Âge 15-25 ans, cherchant à apprendre des sujets scolaires comme les maths ou le codage de manière interactive. Professionnel en reconversion : Âge 25-40 ans, désirant acquérir de nouvelles compétences (ex. : programmation) via des sessions courtes et vocales. Apprenant autodidacte : Tout âge, motivé par l'apprentissage personnel, préférant l'interaction vocale pour une meilleure rétention.

2.4 Parties prenantes et sponsors

Développeurs : Équipe technique utilisant Next.js,React Router, VAPI, Clerk, Supabase et Tailwind v4. Utilisateurs finaux : Apprenants. Administrateurs : Gestion de la plateforme.

3 ANALYSE DE L'EXISTANT

3.1 Solutions actuelles utilisées

Actuellement, des outils comme Zoom avec bots IA ou des apps comme Replika existent, mais aucun ne combine spécifiquement l'apprentissage vocal personnalisé avec VAPI. Des plateformes comme Coursera offrent des cours, mais sans interaction vocale en temps réel.

3.2 Audit de l'écosystème web existant

L'écosystème actuel inclut des sites web éducatifs avec APIs pour IA (ex. : OpenAI), mais manque d'intégration vocale fluide. SnowBrain AI comblera ce vide en utilisant VAPI pour les appels.

3.3 Analyse concurrentielle

Concurrents directs : Apps comme Elsa Speak (apprentissage vocal) ou Tutor AI. Forces : Interaction en temps réel, personnalisation, assiatant qui vous apprend ce dont vous avez besoin a tous moment. Faiblesses : Dépendance à l'API VAPI pour la qualité vocale.

3.4 Retours d'expérience utilisateurs

Basé sur des retours hypothétiques : Les utilisateurs apprécient l'interaction vocale pour l'engagement, mais demandent une personnalisation plus fine et une facturation moins chère.

4 BESOINS FONCTIONNELS

4.1 Fonctionnalités core de l'application

Création de "Brains" (session d'apprentissage interactif) : Sélection de sujet, voix, détails du sujet d'apprentissage, durée, nom du Brain. Appel vocal en temps réel avec IA via VAPI. Interruption de l'appel à tout moment.

4.2 Parcours utilisateur (user journey)

Inscription/Connexion via Clerk. Création d'un Brain : Choix du sujet (maths, codage, etc.), voix, détails, durée. Lancement de l'appel vocal. Fin de session et feedback. Définit plus précisement dans le fichier FigJam

4.3 Gestion des comptes utilisateurs

Gestion des profils, historique des sessions, via Supabase.

4.4 Système d'authentification et autorisation

Utilisation de Clerk pour l'authentification sécurisée (utilisateur).

4.5 Fonctionnalités métier spécifiques

Intégration VAPI pour les appels IA, personnalisation des agents "Brain".

4.6 Modules d'administration

Dashboard pour gérer les utilisateurs, sessions, et facturation via Clerk.

4.7 Système de notifications

Notifications push/email pour rappels de sessions ou mises à jour.

5 INTERFACE UTILISATEUR (UI/UX)

5.1 Charte graphique et identité visuelle

Utilisation de Tailwind v4 pour un design moderne, ShadCn pour la librarie des composants, logos IA stylisés. Tout cela définit dans le fichier Figma.

5.2 Design responsive (mobile, tablette, desktop)

Tailwind V4 avec une utilisation maximum de tous ces "Layers" assure la responsivité pour tous les appareils.

5.3 Wireframes et maquettes

À fournir : Wireframes pour la page de création de Brain et l'interface d'appel, Acceuil et autres ...

5.4 Expérience utilisateur (UX)

Interface intuitive, flux linéaire pour minimiser les clics, loaders au moments de traitement pour une meilleur experience.

5.5 Accessibilité (WCAG, RGAA)

Respect des normes WCAG 2.1 : Contrastes, navigation clavier, 'alt' text pour images.

5.6 Navigation et arborescence

Menu principal : Home, Brains, My Profil, subscription.

6 ARCHITECTURE TECHNIQUE WEB

6.1 Architecture front-end

Next.js pour le rendering server-side et client-side, Typescript pour un code base robuste, Tailwind V4 pour une présentation responsive.

6.2 Architecture back-end (API)

Next.js API routes intégrées avec VAPI et Supabase.

6.3 Base de données

Supabase (PostgreSQL) pour stocker utilisateurs, Brains, historiques.

6.4 Technologies web recommandées

HTML5, CSS3, JavaScript ES6+.

6.5 Frameworks et librairies

Next.js, Tailwind v4, Typescript, Clerk, VAPI SDK, Supabase client, Sentry, Docker.

6.6 Gestion des assets (images, CSS, JS)

Assets statiques dans Next.js public folder, rendu optimisé avec les composants TSX core de NextJs, optimisation via Tailwind.

6.7 Performance et temps de chargement

Temps de chargement ¡ 2s, optimisation avec Next.js Image.

7 BESOINS NON FONCTIONNELS

7.1 Scalabilité et montée en charge

Supabase et Vercel ou AWS Ec2 (pour hébergement Next.js) pour scalabilité auto.

7.2 Compatibilité navigateurs

Chrome, Firefox, Safari, Edge (dernières versions).

7.3 SEO et référencement naturel

Next.js pour SSR, métadonnées dynamiques.

7.4 Sécurité web (HTTPS, OWASP)

HTTPS obligatoire, respect OWASP top 10 via Clerk et Supabase.

7.5 Disponibilité et uptime

99.9

8 INTÉGRATIONS ET API

8.1 API tierces à intégrer

VAPI pour les appels vocaux IA, Clerk pour auth/billing.

8.2 Systèmes d'information existants

Intégration avec Supabase pour stockage.

8.3 Services de paiement en ligne

Clerk intègre Stripe pour billing.

8.4 Réseaux sociaux

Partage de sessions via liens sociaux.

8.5 Services cloud (stockage, CDN)

Supabase pour stockage, Vercel CDN pour assets, AWS pour le déploiement.

9 HÉBERGEMENT ET INFRASTRUCTURE

9.1 Hébergement web (cloud, dédié, mutualisé)

Hébergement cloud via Vercel pour Next.js ou AWS Ec2, Supabase pour DB.

9.2 Nom de domaine et certificats SSL

Domaine snowbrain.ai ou gratuit avec SSL gratuit offert par vercel free trier, ou AWS Route 53.

9.3 CDN et optimisation de contenu

Vercel CDN intégré.

9.4 Monitoring et supervision

Sentry.

9.5 Environnements (dev, staging, prod)

Branches Git pour dev/staging/prod.

10 SÉCURITÉ WEB

10.1 Authentification

Clerk pour Auth sécurisées.

10.2 Chiffrement des données

Chiffrement at-rest via Supabase.

10.3 Conformité RGPD

Consentement utilisateur, droit à l'oubli.

10.4 Politique de mots de passe

Force minimale via Clerk et supabase.

11 RÉFÉRENCEMENT ET WEBMARKETING

11.1 Optimisation SEO on-page

Titres, descriptions optimisés.

11.2 Structure des URLs

URLs propres : /brains/[id] via 'File based Routing'.

11.3 Métadonnées et Open Graph

Métadonnées pour partage social.

11.4 Sitemap XML

Généré par Next.js.

12 TESTS ET QUALITÉ

12.1 Tests fonctionnels

Tests unitaires avec Jest.

12.2 Tests de compatibilité navigateurs

Cross-browser testing.

12.3 Tests de performance

Lighthouse audits.

12.4 Tests de sécurité

Scans OWASP.

12.5 Tests d'accessibilité

Outils comme WAVE.

12.6 Tests utilisateurs (UAT)

Beta testing avec utilisateurs.

13 DÉPLOIEMENT ET MISE EN LIGNE

13.1 Stratégie de déploiement

CI/CD via GitHub Actions et Vercel.

13.2 Migration des données

Migration initiale via Supabase scripts.

13.3 Mise en production

Déploiement automatique sur merge.

13.4 Plan de rollback

Versions Git pour rollback.

13.5 Formation des utilisateurs

Tutoriels in-app sur Youtube.

13.6 Communication du lancement

Emails, posts sociaux.

14 MAINTENANCE ET ÉVOLUTION

14.1 Maintenance corrective

Corrections bugs rapides avec Sentry, ce qui permet une haute Disponibilité toute en évitant un crash, ce qui a pour but de rassurer les clients.

14.2 Mises à jour de sécurité

Patches réguliers.

14.3 Évolutions fonctionnelles

Roadmap sur FigJam pour nouvelles features.

14.4 Support technique

Chat support in-app.

14.5 Sauvegarde et archivage

Sauvegardes quotidiennes et automatique.

15 BUDGET ET PLANNING

15.1 Estimation des coûts de développement

0 € pour développement initial en partant de free trier pour certains services tels que VAPI et sans embaucher de Developpeurs.

15.2 Coûts d'hébergement et infrastructure

Vercel/Supabase : 0 €/mois en partant de free trier.

15.3 Licences et outils

Clerk/VAPI : Abonnements mensuels.

15.4 Planning de développement

1 mois : Conception, dev, tests.

15.5 Jalons et livrables

Jalon 1 : MVP en 3 Semaines.

15.6 Ressources humaines nécessaires

2 devs full-stack, 1 designer mais tout sera fait par SnowDev.

16 CRITÈRES D'ACCEPTATION

16.1 Critères fonctionnels

Toutes features core fonctionnelles.

16.2 Critères de performance web

Temps de chargement ¡ 2s.

16.3 Critères d'accessibilité

WCAG AA compliant.

16.4 Critères de sécurité

Pas de vulnérabilités critiques.

17 ANNEXES

17.1 Annexe A : Maquettes et wireframes

(À joindre : Fichiers Figma et FigJam.)

17.2 Annexe B : Spécifications API

Détails endpoints VAPI et Clerk.

17.3 Annexe C : Charte graphique

Palette couleurs, fonts.

17.4 Annexe D : Matrice de compatibilité navigateurs

Tableau des browsers supportés.

17.5 Annexe E : Plan de tests détaillé

Liste des test cases.

17.6 Annexe F : Documentation technique

Docs Next.js, Supabase, Github (ReadMe) etc.