

Cahier de recettes - Penpal AI

Ce cahier de recettes présente l'ensemble des scénarios de tests à exécuter pour valider le bon fonctionnement de la plateforme Penpal AI. Il couvre les tests fonctionnels, techniques, de sécurité et de performance avec les résultats attendus pour chaque scénario.

Vue d'ensemble des tests

Environnements de test

- **Staging** : staging.app.penpal-ai.maksou.dev - Tests complets avant mise en production
- **Production** : app.penpal-ai.maksou.dev - Tests de smoke et validation finale

Types de tests couverts

1. **Tests fonctionnels** : Validation des fonctionnalités utilisateur
2. **Tests techniques** : Validation des APIs et intégrations
3. **Tests de sécurité** : Validation des mesures de protection
4. **Tests de performance** : Validation de la charge et des temps de réponse
5. **Tests de compatibilité** : Validation navigateurs et appareils

Critères de validation

- **Réussite** : Scénario exécuté avec succès selon les critères attendus
- **Attention** : Fonctionnalité opérationnelle avec remarques mineures
- **Échec** : Scénario non conforme aux critères attendus

Tests fonctionnels

TF01 - Accès à l'application

TF01.1 - Accès page d'accueil

Objectif : Vérifier l'accessibilité de l'application et l'affichage correct de la page d'accueil.

Prérequis : Navigateur web moderne

Étapes :

1. Ouvrir un navigateur web
2. Naviguer vers <https://app.penpal-ai.maksou.dev>
3. Attendre le chargement complet de la page

Résultats attendus :

- Page d'accueil chargée en < 3 secondes
- Titre affiché : "PenPal - Your conversational AI to learn languages"
- Présence des boutons : "Démo AI Gratuite", "Commencer gratuitement", "Se connecter"

- Interface responsive adaptée à la taille d'écran
- Absence d'erreurs JavaScript dans la console
- Favicon et métadonnées correctement chargés

TF01.2 - Navigation multilingue

Objectif : Vérifier le changement de langue de l'interface.

Prérequis : Page d'accueil chargée

Étapes :

1. Localiser le sélecteur de langue (généralement en haut à droite)
2. Cliquer sur "Français" si l'interface est en anglais
3. Observer le changement de langue
4. Cliquer sur "English" pour revenir à l'anglais

Résultats attendus :

- Sélecteur de langue visible et accessible
 - Changement immédiat de langue sans rechargement de page
 - Tous les textes traduits correctement
 - URL mise à jour avec le locale approprié
 - Préférence de langue persistante lors de la navigation
-

TF02 - Démonstration gratuite

TF02.1 - Accès à la démo

Objectif : Tester l'accès à la démonstration IA sans inscription.

Prérequis : Page d'accueil chargée

Étapes :

1. Cliquer sur le bouton "Démo AI Gratuite"
2. Attendre le chargement de la page démo
3. Observer l'interface de configuration

Résultats attendus :

- Redirection vers [/demo](#) en < 2 secondes
- Titre affiché : "Discover artificial intelligence for language learning"
- Instructions d'utilisation visibles et claires
- Zone de configuration avec 3 sections : Langue, Niveau, Mode
- Interface de chat initialisée et prête à l'emploi

TF02.2 - Configuration de la démo

Objectif : Valider la configuration des paramètres de démonstration.

Prérequis : Page démo chargée

Étapes :

1. Configuration langue :

- Sélectionner "English" dans le menu déroulant langue
- Vérifier la mise à jour de l'interface

2. Configuration niveau :

- Sélectionner "Beginner" dans les options de niveau
- Observer le feedback visuel

3. Configuration mode :

- Sélectionner "Tutor" (mode avec corrections)
- Vérifier l'activation du mode

Résultats attendus :

- Sélection de langue parmi 7 options disponibles (Anglais, Français, Espagnol, Italien, Allemand, Portugais, etc.)
- Feedback visuel immédiat pour chaque sélection
- Niveaux disponibles : Débutant, Intermédiaire, Avancé
- Modes disponibles : Tuteur (corrections) et Partenaire (conversation libre)
- Configuration sauvegardée automatiquement
- Interface IA adaptée aux paramètres sélectionnés

TF02.3 - Conversation démo

Objectif : Tester l'interaction avec l'IA en mode démonstration.

Prérequis : Démo configurée (Anglais, Débutant, Mode Tuteur)

Étapes :

1. Message simple :

- Taper "Hello, how are you?" dans la zone de texte
- Cliquer sur "Envoyer" ou presser Entrée
- Attendre la réponse de l'IA

2. Message avec erreur :

- Taper "I are very good" (erreur grammaticale volontaire)
- Envoyer le message
- Observer les corrections proposées

3. Test suggestions :

- Cliquer sur une suggestion de conversation prédéfinie
- Vérifier la réponse de l'IA

Résultats attendus :

- Messages utilisateur affichés à droite avec fond coloré
- Réponse IA générée en < 5 secondes
- Messages IA affichés à gauche avec fond neutre
- Erreurs grammaticales détectées et surlignées

- Corrections expliquées en contexte : "I **am** very good" avec explication
 - Suggestions de conversation cliquables et fonctionnelles
 - Indicateur "Penpal est en train d'écrire..." pendant génération
 - Historique des messages conservé pendant la session
-

TF03 - Authentification et inscription

TF03.1 - Inscription par email

Objectif : Valider le processus d'inscription avec email et mot de passe.

Prérequis : Page d'accueil chargée

Étapes :

1. Cliquer sur "Commencer gratuitement"
2. Attendre le chargement du formulaire d'inscription
3. **Remplir le formulaire** :
 - Nom complet : "Jean Dupont"
 - Email : "jean.dupont.test@example.com"
 - Mot de passe : "TestPassword123!"
 - Cocher "J'accepte les conditions d'utilisation"
4. Cliquer sur "S'inscrire"
5. Attendre la validation et redirection

Résultats attendus :

- Redirection vers formulaire d'inscription en < 2 secondes
- Formulaire avec 4 champs obligatoires clairement identifiés
- Validation en temps réel des champs :
 - Nom : minimum 2 caractères, lettres uniquement
 - Email : format email valide requis
 - Mot de passe : 8+ caractères, majuscule, minuscule, chiffre, caractère spécial
- Case à cocher conditions obligatoire pour activation du bouton
- Bouton "S'inscrire" désactivé tant que formulaire invalide
- Soumission réussie en < 3 secondes
- Redirection automatique vers processus d'onboarding
- Compte créé avec période d'essai de 30 jours activée

TF03.2 - Inscription via Google OAuth

Objectif : Tester l'inscription rapide avec compte Google.

Prérequis : Page d'inscription chargée, compte Google valide

Étapes :

1. Cliquer sur le bouton "Continuer avec Google"
2. **Si redirection vers Google** :

- Sélectionner le compte Google à utiliser
 - Autoriser l'accès aux informations demandées
3. Attendre la redirection vers l'application
 4. Vérifier la création automatique du compte

Résultats attendus :

- Bouton Google OAuth visible et stylisé
- Redirection sécurisée vers accounts.google.com
- Scope demandé limité : email, profile, openid
- Retour automatique vers l'application après autorisation
- Compte créé automatiquement avec informations Google
- Pas de demande de mot de passe supplémentaire
- Email de bienvenue envoyé automatiquement
- Redirection vers onboarding pour finaliser le profil
- Période d'essai 30 jours activée automatiquement

TF03.3 - Connexion utilisateur existant

Objectif : Valider la connexion d'un utilisateur déjà inscrit.

Prérequis : Compte utilisateur existant créé précédemment

Étapes :

1. Depuis la page d'accueil, cliquer sur "Se connecter"
2. **Remplir le formulaire de connexion :**
 - Email : "jean.dupont.test@example.com"
 - Mot de passe : "TestPassword123!"
3. Cliquer sur "Se connecter"
4. Attendre l'authentification et la redirection

Résultats attendus :

- Formulaire de connexion avec 2 champs + bouton
- Validation côté client des champs obligatoires
- Authentification réussie en < 2 secondes
- Redirection vers [/chat](#) si onboarding terminé
- Redirection vers [/onboarding](#) si profil incomplet
- Menu utilisateur visible avec nom/avatar
- Session sécurisée avec JWT stocké en httpOnly cookie
- Pas d'affichage du mot de passe en clair

TF03.4 - Gestion des erreurs d'authentification

Objectif : Valider la gestion des erreurs de connexion.

Prérequis : Page de connexion chargée

Étapes :

1. Test email invalide :

- Email : "email-inexistant@example.com"
- Mot de passe : "MotDePasseValide123!"
- Cliquer sur "Se connecter"

2. Test mot de passe incorrect :

- Email : "jean.dupont.test@example.com"
- Mot de passe : "MauvaisMotDePasse"
- Cliquer sur "Se connecter"

3. Test champs vides :

- Laisser les champs vides et tenter de se connecter

Résultats attendus :

- Message d'erreur générique : "Email ou mot de passe incorrect"
 - Pas d'indication sur quel champ est erroné (sécurité)
 - Formulaire réinitialisé après erreur
 - Rate limiting appliqué après 5 tentatives échouées
 - Compte temporairement bloqué après tentatives excessives
 - Validation côté client pour champs obligatoires
 - Messages d'erreur clairs et en français/anglais selon la langue
-

TF04 - Processus d'onboarding**TF04.1 - Étape 1 : Nom préféré**

Objectif : Valider la première étape du processus d'onboarding.

Prérequis : Utilisateur connecté, redirection vers [/onboarding](#)

Étapes :

1. Observer l'interface de l'étape 1/4
2. **Saisir le nom préféré :**
 - Entrer "Jean" dans le champ
 - Observer la validation en temps réel
3. Cliquer sur "Suivant" pour passer à l'étape suivante

Résultats attendus :

- Indicateur de progression affiché : "Étape 1 sur 4"
- Titre clair : "Comment souhaitez-vous être appelé ?"
- Champ de saisie avec placeholder explicite
- Exemples interactifs fournis pour inspiration
- Validation en temps réel :
 - Minimum 2 caractères
 - Lettres, espaces, traits d'union et apostrophes uniquement
- Bouton "Suivant" activé seulement si validation OK
- Sauvegarde automatique du progrès
- Possibilité de revenir en arrière avec bouton "Précédent"

TF04.2 - Étape 2 : Langue d'apprentissage







Objectif : Tester la sélection de la langue d'apprentissage.

Prérequis : Onboarding étape 1 complétée

Étapes :

1. Observer l'interface de sélection des langues
2. **Sélectionner une langue** :
 - Cliquer sur "English" avec le drapeau britannique
 - Observer le feedback de sélection
3. Cliquer sur "Suivant"

Résultats attendus :

- Progression mise à jour : "Étape 2 sur 4"
- Titre : "Quelle langue souhaitez-vous apprendre ?"
- 7 langues disponibles avec drapeaux :
 - Anglais () , Français () , Espagnol ()
 - Italien () , Allemand () , Portugais () , etc.
- Interface visuelle claire avec cartes cliquables
- Feedback immédiat de sélection (bordure colorée)
- Une seule langue sélectionnable à la fois
- Bouton "Suivant" activé après sélection
- Sauvegarde automatique du choix

TF04.3 - Étape 3 : Niveau de compétence

Objectif : Valider la sélection du niveau de langue.

Prérequis : Onboarding étape 2 complétée

Étapes :

1. Observer les options de niveau disponibles
2. **Sélectionner "Intermédiaire"** :
 - Cliquer sur la carte "Intermédiaire"
 - Lire la description détaillée
3. Cliquer sur "Suivant"

Résultats attendus :

- Progression : "Étape 3 sur 4"
- Titre : "Quel est votre niveau actuel ?"
- 3 niveaux avec descriptions claires :
 - **Débutant** : "Vous commencez l'apprentissage de la langue"
 - **Intermédiaire** : "Vous avez des bases solides"
 - **Avancé** : "Vous maîtrisez bien la langue mais voulez perfectionner"
- Icônes visuelles pour chaque niveau

- Descriptions détaillées visibles au survol ou sélection
- Feedback visuel de la sélection
- Une seule option sélectionnable
- Bouton "Suivant" activé après sélection

TF04.4 - Étape 4 : Récapitulatif et finalisation

Objectif : Valider le récapitulatif et la finalisation de l'onboarding.

Prérequis : Onboarding étape 3 complétée

Étapes :

1. Observer le récapitulatif des informations saisies
2. **Vérifier les données** :
 - Nom : "Jean"
 - Langue : "English"
 - Niveau : "Intermédiaire"
3. Cliquer sur "Finaliser" pour terminer l'onboarding
4. Attendre la redirection vers le chat

Résultats attendus :

- Progression : "Étape 4 sur 4"
 - Titre : "Récapitulatif de votre profil"
 - Affichage correct de toutes les informations saisies
 - Possibilité de modification avec boutons "Modifier"
 - Bouton "Finaliser" clairement visible
 - Processus de finalisation en < 2 secondes
 - Redirection automatique vers [/chat](#)
 - Profil utilisateur complet et sauvegardé
 - Onboarding marqué comme terminé (pas de re-redirection)
-

TF05 - Interface de chat conversationnel

TF05.1 - Accès et initialisation du chat

Objectif : Vérifier l'accès au chat et l'initialisation de l'interface.

Prérequis : Utilisateur authentifié avec onboarding terminé

Étapes :

1. Naviguer vers [/chat](#) ou cliquer sur "Commencer à chatter"
2. Attendre le chargement de l'interface de chat
3. Observer les éléments de l'interface

Résultats attendus :

- Chargement de l'interface chat en < 3 secondes

- Zone de conversation vide et prête à recevoir des messages
- Zone de saisie avec placeholder : "Écrivez votre message..."
- Bouton d'envoi visible (icône ou texte "Envoyer")
- Paramètres IA visibles (langue, niveau, mode)
- Suggestions de conversation disponibles
- Menu utilisateur accessible
- Pas d'erreurs JavaScript dans la console
- Interface responsive adaptée à l'appareil

TF05.2 - Configuration des paramètres IA

Objectif : Tester la modification des paramètres de conversation IA.

Prérequis : Interface de chat chargée

Étapes :

1. **Localiser les paramètres IA** (généralement panneau latéral ou modal)
2. **Modifier la langue** :
 - Changer de "English" vers "Français"
 - Appliquer le changement
3. **Modifier le niveau** :
 - Passer de "Intermédiaire" vers "Avancé"
4. **Changer le mode** :
 - Basculer entre "Tuteur" et "Partenaire"
5. Sauvegarder les modifications

Résultats attendus :

- Panneau de paramètres facilement accessible
- Modification de langue prise en compte immédiatement
- Changement de niveau reflété dans les interactions
- Modes disponibles :
 - **Tuteur** : Corrections et explications pédagogiques
 - **Partenaire** : Conversation naturelle sans corrections systématiques
- Feedback visuel des changements appliqués
- Sauvegarde automatique des préférences
- Paramètres persistants lors des sessions suivantes

TF05.3 - Conversation mode Tuteur

Objectif : Valider les fonctionnalités du mode Tuteur avec corrections.

Prérequis : Chat configuré en mode "Tuteur", langue "English", niveau "Intermédiaire"

Étapes :

1. **Message correct** :
 - Taper : "Hello! I would like to practice English with you."
 - Envoyer le message

- Observer la réponse de l'IA

2. Message avec erreurs :

- Taper : "I are very happy today and I want learn more english"
- Envoyer le message
- Observer les corrections proposées

3. Suivi des corrections :

- Observer les explications fournies
- Vérifier l'intégration dans la conversation

Résultats attendus :

- Messages utilisateur affichés à droite, fond coloré
- Messages IA affichés à gauche, fond neutre
- Réponse IA générée en < 5 secondes
- **Corrections automatiques détectées :**
 - "I **am** very happy today" (correction surlignée)
 - "I want **to** learn more **English**" (corrections multiples)
- **Explications pédagogiques :**
 - "I are" → "I am" : accord sujet-verbe
 - "want learn" → "want to learn" : infinitif requis
 - "english" → "English" : nom propre avec majuscule
- IA continue la conversation de manière naturelle
- Ton encourageant et bienveillant
- Adaptation au niveau déclaré (intermédiaire)

TF05.4 - Conversation mode Partenaire

Objectif : Tester le mode Partenaire pour conversation naturelle.

Prérequis : Chat configuré en mode "Partenaire"

Étapes :

1. **Changer le mode** vers "Partenaire" dans les paramètres
2. **Conversation naturelle :**
 - Taper : "What's your favorite hobby and why?"
 - Envoyer et attendre la réponse
3. **Message avec erreur mineure :**
 - Taper : "That's interesting! I likes reading books too."
 - Observer si les erreurs sont corrigées ou ignorées
4. **Conversation continue :**
 - Poursuivre la discussion naturellement

Résultats attendus :

- Mode Partenaire activé sans corrections systématiques
- IA répond comme un ami ou partenaire de conversation
- Conversation fluide et naturelle
- Erreurs mineures ignorées pour privilégier le flux

- Ton décontracté et engageant
- Questions de relance pour maintenir la conversation
- Adaptation au contexte et aux intérêts exprimés
- Pas de corrections grammaticales sauf si demandées explicitement

TF05.5 - Fonctionnalités avancées du chat

Objectif : Tester les fonctionnalités supplémentaires de l'interface.

Prérequis : Conversation en cours avec plusieurs messages

Étapes :

1. **Test suggestions** :
 - Cliquer sur une suggestion de conversation prédéfinie
 - Vérifier l'envoi automatique
2. **Test historique** :
 - Actualiser la page
 - Vérifier la persistance des messages
3. **Test indicateurs d'état** :
 - Envoyer un message
 - Observer l'indicateur "Penpal est en train d'écrire..."
4. **Test responsive** :
 - Redimensionner la fenêtre
 - Vérifier l'adaptation mobile

Résultats attendus :

- Suggestions cliquables et contextuelles
 - Envoi automatique des suggestions sélectionnées
 - Historique complet conservé après rafraîchissement
 - Indicateur de frappe visible pendant génération IA
 - Interface parfaitement responsive :
 - Mobile : conversation pleine largeur
 - Desktop : panneau latéral pour paramètres
 - Scroll automatique vers le dernier message
 - Performance fluide même avec historique long
-

TF06 - Gestion du profil utilisateur

TF06.1 - Accès au profil

Objectif : Valider l'accès aux informations du profil utilisateur.

Prérequis : Utilisateur authentifié

Étapes :

1. Cliquer sur le menu utilisateur (avatar/nom en haut à droite)

2. Sélectionner "Profil" ou "Paramètres"
3. Observer les informations affichées

Résultats attendus :

- Menu utilisateur facilement accessible
- Redirection vers page profil en < 2 secondes
- Informations personnelles affichées :
 - Nom complet et nom préféré
 - Adresse email
 - Date d'inscription
 - Méthode de connexion (email ou OAuth)
- Préférences d'apprentissage :
 - Langue d'apprentissage actuelle
 - Niveau de compétence
 - Mode de conversation préféré
- Interface claire et organisée par sections

TF06.2 - Modification des informations

Objectif : Tester la modification des informations du profil.

Prérequis : Page profil chargée

Étapes :

1. **Modifier le nom préféré :**
 - Cliquer sur "Modifier" à côté du nom
 - Changer "Jean" vers "Johnny"
 - Sauvegarder les modifications
2. **Changer la langue d'apprentissage :**
 - Modifier de "English" vers "Español"
 - Confirmer le changement
3. **Ajuster le niveau :**
 - Passer de "Intermédiaire" vers "Avancé"
 - Valider la modification

Résultats attendus :

- Formulaire de modification accessibles et intuitifs
- Validation en temps réel des champs modifiés
- Sauvegarde réussie avec confirmation visuelle
- Modifications reflétées immédiatement dans l'interface
- Nouveau nom affiché dans le menu utilisateur
- Paramètres de langue/niveau appliqués au chat
- Aucune perte de données lors des modifications

TF06.3 - Historique et statistiques

Objectif : Valider l'affichage des statistiques d'utilisation.

Prérequis : Utilisateur avec historique de conversations

Étapes :

1. Naviguer vers la section "Statistiques" ou "Progression"
2. Observer les métriques affichées
3. Vérifier la cohérence des données

Résultats attendus :

- Section statistiques clairement organisée
 - **Métriques d'activité** :
 - Nombre de conversations
 - Messages échangés
 - Temps de session moyen
 - Dernière activité
 - **Progression pédagogique** :
 - Erreurs corrigées
 - Vocabulaire acquis
 - Points d'amélioration identifiés
 - **Graphiques visuels** :
 - Activité quotidienne/hebdomadaire
 - Évolution du niveau
 - Objectifs atteints
 - Données cohérentes avec l'utilisation réelle
 - Mise à jour en temps réel des métriques
-

TF07 - Gestion des abonnements

TF07.1 - Affichage du statut d'abonnement

Objectif : Vérifier l'affichage correct du statut d'abonnement.

Prérequis : Utilisateur avec période d'essai active

Étapes :

1. Depuis le profil, localiser la section "Abonnement"
2. Observer les informations d'abonnement affichées
3. Naviguer vers [/pricing](#) pour voir les options

Résultats attendus :

- **Statut clairement affiché** :
 - "Essai gratuit" avec durée restante
 - Date de fin d'essai précise
 - "X jours restants" bien visible
- **Informations complètes** :

- Type d'abonnement actuel
- Date de début et fin
- Statut de paiement
- Prochaine facturation (si applicable)
- **Actions disponibles :**
 - Bouton "Mettre à niveau" vers plan payant
 - Lien vers page tarification
 - Option d'annulation si applicable
- Page tarification accessible et cohérente

TF07.2 - Processus de souscription

Objectif : Tester le processus de souscription à un plan payant.

Prérequis : Compte en période d'essai, carte de test Stripe

Étapes :

1. **Accès à la tarification :**
 - Naviguer vers [/pricing](#)
 - Observer les plans disponibles
2. **Sélection plan mensuel :**
 - Cliquer sur "Choisir" pour le plan mensuel
 - Vérifier la redirection vers Stripe
3. **Processus de paiement** (avec carte de test) :
 - Remplir les informations de paiement
 - Numéro de carte test : [4242 4242 4242 4242](#)
 - Date d'expiration : toute date future
 - CVC : tout code à 3 chiffres
 - Confirmer le paiement

Résultats attendus :

- **Page tarification claire :**
 - Plan Mensuel : 20€/mois
 - Plan Annuel : 200€/an (économie de 2 mois)
 - Comparaison détaillée des fonctionnalités
- **Fonctionnalités listées :**
 - IA conversationnelle avancée
 - Génération de contenu
 - Support email/prioritaire
 - Accès mobile et desktop
- **Redirection Stripe sécurisée** vers checkout.stripe.com
- **Formulaire de paiement sécurisé :**
 - Champs carte bancaire validés
 - Informations chiffrées côté Stripe
 - Pas de stockage local des données carte
- **Confirmation de paiement** et retour vers l'application

- **Mise à jour immédiate** du statut d'abonnement

TF07.3 - Gestion de l'abonnement actif

Objectif : Valider la gestion d'un abonnement payant actif.

Prérequis : Abonnement payant actif

Étapes :

1. **Vérification du statut** :
 - Retourner au profil
 - Observer le nouveau statut d'abonnement
2. **Test des fonctionnalités premium** :
 - Accéder au chat
 - Vérifier l'accès illimité aux conversations
3. **Options de gestion** :
 - Localiser les options de modification d'abonnement
 - Tester la possibilité d'annulation

Résultats attendus :

- **Statut mis à jour** :
 - "Abonnement Premium Actif"
 - Date de prochaine facturation
 - Montant du prochain paiement
- **Accès illimité confirmé** :
 - Pas de limitations sur le chat
 - Toutes les fonctionnalités disponibles
 - Pas de messages de restriction
- **Options de gestion disponibles** :
 - Modifier le plan (mensuel ↔ annuel)
 - Mettre à jour les informations de paiement
 - Annuler l'abonnement
 - Télécharger les factures

TF07.4 - Processus d'annulation

Objectif : Tester le processus d'annulation d'abonnement.

Prérequis : Abonnement actif

Étapes :

1. **Initiation de l'annulation** :
 - Cliquer sur "Annuler l'abonnement"
 - Observer le processus de confirmation
2. **Confirmation d'annulation** :
 - Lire les informations sur les conséquences
 - Confirmer l'annulation si désiré

3. Vérification post-annulation :

- Observer le nouveau statut
- Vérifier l'accès aux fonctionnalités

Résultats attendus :

- **Processus d'annulation clair :**
 - Explication des conséquences
 - Date d'effet de l'annulation
 - Confirmation double requise
 - **Information transparente :**
 - Accès maintenu jusqu'à la fin de la période payée
 - Perte d'accès à partir de date X
 - Possibilité de réactivation
 - **Statut post-annulation :**
 - "Abonnement annulé" avec date d'effet
 - Fonctionnalités accessibles jusqu'à expiration
 - Option de réactivation disponible
-

Tests techniques

TT01 - APIs Backend

TT01.1 - Service d'authentification

Objectif : Valider le fonctionnement des APIs d'authentification.

Prérequis : Accès aux APIs, outil de test (Postman/curl)

Étapes :

1. Test d'inscription :

```
POST /api/v1/auth/register
Content-Type: application/json

{
  "name": "Test User",
  "email": "test.api@example.com",
  "password": "TestPassword123!"
}
```

2. Test de connexion :

```
POST /api/v1/auth/login
Content-Type: application/json
```



```
{  
  "email": "test.api@example.com",  
  "password": "TestPassword123!"  
}
```

3. Test route protégée :

```
GET /api/v1/users/me  
Authorization: Bearer {JWT_TOKEN}
```

Résultats attendus :

- **Inscription réussie :**
 - Status Code : 201 Created
 - Réponse contient userId et confirmation
 - Email de bienvenue envoyé
 - Période d'essai initialisée
- **Connexion réussie :**
 - Status Code : 200 OK
 - JWT token valide retourné
 - Cookie httpOnly défini
 - Informations utilisateur incluses
- **Route protégée accessible :**
 - Status Code : 200 OK avec token valide
 - Status Code : 401 Unauthorized sans token
 - Données utilisateur complètes retournées

TT01.2 - Service IA (Asimov)

Objectif : Tester les APIs de génération de contenu IA.

Prérequis : Token d'authentification valide

Étapes :

1. Test génération chat :

```
POST /api/v1/ai/chat  
Authorization: Bearer {JWT_TOKEN}  
Content-Type: application/json  
  
{  
  "message": "Hello, how are you?",  
  "conversationId": "conv_123",  
  "language": "english",  
  "level": "intermediate",  
}
```

```
"mode": "tutor"
}
```

2. Test analyse de texte :

```
POST /api/v1/ai/analyze
Authorization: Bearer {JWT_TOKEN}
Content-Type: application/json

{
  "text": "I are very happy today",
  "language": "english",
  "analysisType": "grammar"
}
```

3. Test suggestions de conversation :

```
GET /api/v1/ai/conversation-starters?
language=english&level=intermediate
Authorization: Bearer {JWT_TOKEN}
```

Résultats attendus :

- **Chat IA fonctionnel :**
 - Status Code : 200 OK
 - Réponse générée en < 5 secondes
 - Contenu adapté au niveau et mode
 - Format de réponse structuré et cohérent
- **Analyse grammaticale :**
 - Status Code : 200 OK
 - Erreurs détectées : "I are" → "I am"
 - Explications pédagogiques fournies
 - Suggestions d'amélioration pertinentes
- **Suggestions contextuelles :**
 - Status Code : 200 OK
 - Liste de 5-10 suggestions
 - Adaptées au niveau spécifié
 - Variété thématique (salutations, hobbies, actualité)

TT01.3 - Service de base de données

Objectif : Valider les opérations CRUD sur les données utilisateur.

Prérequis : Token d'authentification, utilisateur existant

Étapes :

1. Test lecture utilisateur :

```
GET /api/v1/users/{userId}
Authorization: Bearer {JWT_TOKEN}
```

2. Test mise à jour profil :

```
PATCH /api/v1/users/{userId}
Authorization: Bearer {JWT_TOKEN}
Content-Type: application/json

{
  "preferredName": "Johnny",
  "learningLanguage": "spanish",
  "proficiencyLevel": "advanced"
}
```

3. Test gestion conversations :

```
GET /api/v1/conversations/user/{userId}
Authorization: Bearer {JWT_TOKEN}
```

Résultats attendus :

- **Lecture utilisateur :**
 - Status Code : 200 OK
 - Données complètes et à jour
 - Informations sensibles masquées (mot de passe)
 - Métadonnées d'abonnement incluses
- **Mise à jour réussie :**
 - Status Code : 200 OK
 - Modifications appliquées immédiatement
 - Validation des données entrantes
 - Audit log de la modification
- **Gestion conversations :**
 - Status Code : 200 OK
 - Liste paginée des conversations
 - Métadonnées complètes (date, nombre de messages)
 - Tri chronologique correct

TT01.4 - Service de paiement

Objectif : Tester les APIs de gestion des abonnements.

Prérequis : Compte Stripe test configuré

Étapes :**1. Test création abonnement :**

```
POST /api/v1/subscriptions
Authorization: Bearer {JWT_TOKEN}
Content-Type: application/json

{
  "userId": "{userId}",
  "plan": "monthly",
  "paymentMethodId": "pm_card_visa"
}
```

2. Test statut abonnement :

```
GET /api/v1/subscriptions/user/{userId}/status
Authorization: Bearer {JWT_TOKEN}
```

3. Test webhook Stripe :

```
POST /api/v1/webhooks/stripe
Stripe-Signature: {signature}
Content-Type: application/json

{
  "type": "invoice.payment_succeeded",
  "data": { ... }
}
```

Résultats attendus :

- **Création abonnement :**
 - Status Code : 200 OK
 - Abonnement créé dans Stripe
 - Statut utilisateur mis à jour
 - Période d'essai prolongée ou terminée
- **Statut correct :**
 - Status Code : 200 OK
 - Informations d'abonnement exactes
 - Dates de facturation précises
 - Statut de paiement à jour
- **Webhook traité :**
 - Status Code : 200 OK
 - Événement Stripe validé et traité

- Base de données synchronisée
 - Logs d'audit créés
-

TT02 - Intégrations externes

TT02.1 - Intégration OpenAI

Objectif : Valider l'intégration avec l'API OpenAI.

Prérequis : Clé API OpenAI configurée

Étapes :

1. Test appel API direct :

- Vérifier la configuration de la clé API
- Tester un appel simple à l'API OpenAI
- Mesurer les temps de réponse

2. Test gestion des erreurs :

- Simuler une clé API invalide
- Tester la limitation de rate (si applicable)
- Vérifier les mécanismes de fallback

Résultats attendus :

- **Connexion OpenAI fonctionnelle :**
 - Réponses GPT-4 générées correctement
 - Temps de réponse < 5 secondes
 - Qualité des réponses adaptée au contexte
- **Gestion des erreurs robuste :**
 - Erreurs API capturées et loggées
 - Messages d'erreur utilisateur appropriés
 - Mécanismes de retry configurés
- **Monitoring des coûts :**
 - Tracking des tokens consommés
 - Limites de coûts respectées
 - Alertes en cas de dépassement

TT02.2 - Intégration Stripe

Objectif : Valider l'intégration complète avec Stripe.

Prérequis : Compte Stripe test, webhooks configurés

Étapes :

1. Test création customer :

- Créer un customer Stripe via API

- Vérifier la synchronisation en base

2. Test cycle de paiement complet :

- Créer une subscription
- Simuler un paiement réussi
- Vérifier la réception du webhook

3. Test échec de paiement :

- Simuler un échec de paiement
- Vérifier la gestion de l'échec

Résultats attendus :

- **Customer management :**
 - Customers créés correctement dans Stripe
 - Synchronisation bidirectionnelle fonctionnelle
 - Métadonnées utilisateur associées
- **Cycle de paiement :**
 - Subscriptions créées et activées
 - Webhooks reçus et traités en < 30 secondes
 - Statuts utilisateurs mis à jour automatiquement
- **Gestion des échecs :**
 - Notifications d'échec envoyées
 - Tentatives de recouvrement configurées
 - Suspension d'accès après échec persistant

TT02.3 - Intégration SendGrid

Objectif : Valider l'envoi d'emails via SendGrid.

Prérequis : Clé API SendGrid configurée

Étapes :

1. Test email de bienvenue :

- Créer un nouveau compte
- Vérifier l'envoi automatique de l'email
- Contrôler le contenu et le design

2. Test emails transactionnels :

- Reset de mot de passe
- Confirmations d'abonnement
- Notifications de facturation

Résultats attendus :

- **Emails de bienvenue :**
 - Envoi automatique lors de l'inscription

- Template responsive et professionnel
 - Personnalisation avec nom utilisateur
 - Liens fonctionnels vers l'application
 - **Emails transactionnels :**
 - Templates appropriés pour chaque type
 - Envoi rapide (< 2 minutes)
 - Tracking des ouvertures et clics
 - Taux de délivrabilité > 95%
-

TT03 - Performance et scalabilité

TT03.1 - Tests de charge API

Objectif : Valider la performance des APIs sous charge.

Prérequis : Outil de test de charge (Artillery, k6)

Étapes :

1. Test de charge authentification :

```
# Configuration Artillery
config:
  target: 'https://staging.auth-service.penpal-ai.maksou.dev'
  phases:
    - duration: 60
      arrivalRate: 10
  scenarios:
    - name: "Login test"
      requests:
        - post:
            url: "/api/v1/auth/login"
            json:
              email: "test@example.com"
              password: "password"
```

2. Test de charge IA :

- 50 requêtes simultanées vers l'API chat
- Mesure des temps de réponse P95
- Vérification de la stabilité

Résultats attendus :

- **Performance authentification :**
 - 100 logins/seconde supportés
 - Temps de réponse P95 < 500ms
 - Taux d'erreur < 1%
 - Pas de memory leaks détectés

- **Performance IA :**

- 20 requêtes IA simultanées gérées
- Temps de réponse P95 < 8 secondes
- Queue management efficace
- Dégradation gracieuse sous charge

TT03.2 - Tests de performance base de données

Objectif : Valider les performances MongoDB sous charge.

Prérequis : Accès direct à MongoDB, dataset de test

Étapes :

1. Test requêtes utilisateurs :

- 1000 lectures simultanées de profils
- Mesure des temps d'exécution
- Vérification des index

2. Test requêtes conversations :

- Chargement de conversations avec historique
- Pagination de gros volumes
- Agrégations statistiques

Résultats attendus :

- **Requêtes utilisateurs :**

- Lectures < 50ms en moyenne
- Index optimisés et utilisés
- Connections pool bien dimensionné

- **Requêtes conversations :**

- Chargement historique < 200ms
- Pagination efficace (offset/limit)
- Agrégations < 1 seconde

TT03.3 - Tests de performance frontend

Objectif : Valider les performances côté client.

Prérequis : Lighthouse, WebPageTest

Étapes :

1. Audit Lighthouse :

- Performance
- Accessibilité
- Bonnes pratiques
- SEO

2. Test sur différents appareils :

- Desktop haut de gamme
- Mobile moyen de gamme
- Connexion 3G lente

Résultats attendus :

- **Scores Lighthouse :**
 - Performance : > 90
 - Accessibilité : > 95
 - Bonnes pratiques : > 90
 - SEO : > 85
 - **Performance multi-device :**
 - First Contentful Paint < 1.5s
 - Largest Contentful Paint < 2.5s
 - Cumulative Layout Shift < 0.1
 - First Input Delay < 100ms
-

Tests de sécurité

TS01 - Authentification et autorisation

TS01.1 - Sécurité des mots de passe

Objectif : Valider la robustesse de la gestion des mots de passe.

Prérequis : Accès au formulaire d'inscription

Étapes :

1. Test mots de passe faibles :

- Essayer "123456"
- Essayer "password"
- Essayer "test" (trop court)

2. Test politique de mots de passe :

- Vérifier les exigences de complexité
- Tester la validation côté client et serveur

3. Test hashage :

- Créer un compte et vérifier en base
- S'assurer que le mot de passe n'est pas stocké en clair

Résultats attendus :

- **Validation robuste :**
 - Rejet des mots de passe < 8 caractères

- Exigence majuscule + minuscule + chiffre + spécial
- Messages d'erreur clairs et pédagogiques
- **Stockage sécurisé :**
 - Hashage bcrypt avec salt (rounds ≥ 12)
 - Aucun mot de passe en clair en base
 - Temps de hashage approprié ($< 500\text{ms}$)
- **Protection contre attaques :**
 - Rate limiting sur tentatives de connexion
 - Blocage temporaire après échecs répétés
 - Pas d'énumération d'utilisateurs possible

TS01.2 - Sécurité JWT

Objectif : Valider la sécurité des tokens JWT.

Prérequis : Token JWT obtenu après connexion

Étapes :

1. Analyse du token :

- Décoder le JWT et examiner les claims
- Vérifier la durée d'expiration
- Tester avec token expiré

2. Test manipulation :

- Modifier le payload du token
- Tester avec signature invalide
- Essayer d'accéder à des ressources non autorisées

3. Test révocation :

- Se déconnecter et tester le token
- Vérifier l'invalidation côté serveur

Résultats attendus :

- **Token bien formé :**
 - Structure JWT standard (header.payload.signature)
 - Claims appropriés : sub, exp, iat, roles
 - Expiration raisonnable (1h pour access token)
- **Sécurité robuste :**
 - Signature HMAC-SHA256 ou RS256
 - Rejet des tokens manipulés (401 Unauthorized)
 - Validation de l'expiration côté serveur
- **Gestion des sessions :**
 - Refresh token avec rotation
 - Révocation effective à la déconnexion
 - Pas de stockage persistant côté client vulnérable

TS01.3 - OAuth 2.0 sécurisé

Objectif : Valider la sécurité de l'intégration OAuth.

Prérequis : Intégration Google OAuth configurée

Étapes :

1. Test flow OAuth :

- Initier l'authentification Google
- Vérifier les paramètres PKCE
- Contrôler la validation du state

2. Test sécurité :

- Tenter de replay d'un code d'autorisation
- Vérifier la validation de l'ID token
- Tester avec un state parameter modifié

Résultats attendus :

- **PKCE implémenté :**
 - Code challenge généré côté client
 - Code verifier validé côté serveur
 - Protection contre interception du code
 - **State parameter :**
 - State aléatoire généré pour chaque flow
 - Validation stricte au retour
 - Protection contre CSRF
 - **ID Token validation :**
 - Signature Google vérifiée
 - Audience et issuer validés
 - Expiration contrôlée
-

TS02 - Protection contre les injections

TS02.1 - Injection SQL/NoSQL

Objectif : Tester la résistance aux attaques d'injection.

Prérequis : Accès aux formulaires de l'application

Étapes :

1. Test injection dans login :

- Email : `admin@test.com' || '1'=='1`
- Email : `{"$ne": null}`
- Mot de passe : `{"$gt": ""}`

2. Test injection dans recherche :

- Paramètres d'URL avec payloads NoSQL
- Headers avec tentatives d'injection

3. Test paramètres de requête :

- Injection dans filtres de conversation
- Manipulation des IDs d'utilisateur

Résultats attendus :

- **Protection NoSQL :**
 - Toutes les injections NoSQL bloquées
 - Utilisation de requêtes paramétrées
 - Validation stricte des types de données
- **Validation des entrées :**
 - Sanitisation automatique des inputs
 - Rejet des caractères spéciaux suspects
 - Logging des tentatives d'injection
- **Responses sécurisées :**
 - Pas de leak d'informations techniques
 - Messages d'erreur génériques
 - Status codes appropriés (400 Bad Request)

TS02.2 - Injection XSS

Objectif : Valider la protection contre les attaques XSS.

Prérequis : Interface de chat ou formulaires utilisateur

Étapes :

1. Test XSS stocké :

- Message chat : `<script>alert('XSS')</script>`
- Nom utilisateur : ``

2. Test XSS réfléchi :

- URL : `?search=<script>alert('XSS')</script>`
- Headers : `User-Agent: <script>...`

3. Test échappement :

- Caractères spéciaux : `&<>"'`
- Encodages alternatifs : `%3Cscript%3E`

Résultats attendus :

- **Échappement automatique :**
 - Tous les scripts bloqués ou échappés

- Balises HTML neutralisées
- Attributs dangereux supprimés
- **Content Security Policy :**
 - CSP header présent et configuré
 - Inline scripts bloqués
 - Sources externes restreintes
- **Validation côté serveur :**
 - Double validation frontend/backend
 - Sanitisation des données persistées
 - Audit des tentatives d'injection

TS02.3 - Injection de commandes

Objectif : Tester la résistance aux injections de commandes.

Prérequis : Fonctionnalités avec traitement de fichiers ou paramètres

Étapes :

1. Test injection dans paramètres :

- Nom de fichier : `test.txt; rm -rf /`
- Paramètres URL : `?file=../../../../etc/passwd`

2. Test directory traversal :

- Chemin : `../../../../etc/hosts`
- Encodage : `..%2F..%2F..%2Fetc%2Fpasswd`

Résultats attendus :

- **Validation des chemins :**
 - Directory traversal bloqué
 - Whitelist des caractères autorisés
 - Sandbox pour fichiers utilisateur
 - **Exécution sécurisée :**
 - Pas d'exécution de commandes système
 - Isolation des processus
 - Permissions minimales
-

TS03 - Sécurité des communications

TS03.1 - HTTPS et TLS

Objectif : Valider la configuration SSL/TLS.

Prérequis : Accès à l'application en production

Étapes :

1. Test configuration SSL :

- Utiliser SSL Labs pour tester app.penpal-ai.maksou.dev
- Vérifier les protocoles supportés
- Contrôler les cipher suites

2. Test redirection HTTP :

- Accéder via <http://app.penpal-ai.maksou.dev>
- Vérifier la redirection automatique

3. Test headers sécurité :

- Strict-Transport-Security
- X-Content-Type-Options
- X-Frame-Options

Résultats attendus :

- **Configuration SSL optimale :**
 - Score SSL Labs : A ou A+
 - TLS 1.2+ uniquement
 - Cipher suites sécurisés
- **Redirection HTTPS :**
 - Redirection 301 automatique
 - HSTS header présent (max-age \geq 31536000)
 - Certificat valide et à jour
- **Headers de sécurité :**
 - CSP configuré et restrictif
 - X-Frame-Options: DENY
 - X-Content-Type-Options: nosniff

TS03.2 - Protection CSRF

Objectif : Valider la protection contre les attaques CSRF.

Prérequis : Formulaires avec actions sensibles

Étapes :

1. Test sans protection :

- Créer un formulaire externe qui POST vers l'API
- Tenter de modifier des données utilisateur

2. Test SameSite cookies :

- Vérifier la configuration des cookies de session
- Tester depuis un domaine externe

Résultats attendus :

- **Protection CSRF active :**
 - Cookies SameSite=Strict ou Lax
 - Token CSRF pour actions sensibles
 - Validation Origin/Referer headers
 - **Requêtes cross-origin bloquées :**
 - Modifications de données impossibles
 - Lecture limitée par CORS
 - Logging des tentatives suspectes
-

TS04 - Sécurité des données

TS04.1 - Chiffrement des données

Objectif : Valider le chiffrement des données sensibles.

Prérequis : Accès aux bases de données (avec permissions admin)

Étapes :

1. Vérification stockage :

- Examiner les mots de passe en base
- Contrôler les données sensibles
- Vérifier les clés API stockées

2. Test chiffrement en transit :

- Capturer le trafic réseau
- Vérifier l'absence de données en clair

Résultats attendus :

- **Chiffrement au repos :**
 - Mots de passe hashés avec bcrypt (rounds \geq 12)
 - Données sensibles chiffrées si nécessaire
 - Clés API sécurisées dans Coolify
- **Chiffrement en transit :**
 - Toutes communications en HTTPS
 - APIs internes en TLS
 - Connexions base de données chiffrées

TS04.2 - Gestion des secrets

Objectif : Valider la gestion sécurisée des secrets.

Prérequis : Accès à la configuration Coolify

Étapes :

1. Audit des variables d'environnement :

- Vérifier l'absence de secrets dans le code
- Contrôler la configuration Coolify
- Examiner les logs pour fuites

2. Test rotation des secrets :

- Changer une clé API
- Vérifier la mise à jour automatique

Résultats attendus :

- **Secrets sécurisés :**
 - Aucun secret dans le code source
 - Variables d'environnement chiffrées
 - Accès restreint aux secrets
 - **Bonnes pratiques :**
 - Secrets différents par environnement
 - Rotation périodique possible
 - Audit trail des accès
-

Tests de performance

TP01 - Performance applicative

TP01.1 - Temps de chargement

Objectif : Valider les temps de chargement des pages principales.

Prérequis : Connexion internet stable, outils de mesure

Étapes :

1. Test page d'accueil :

- Mesurer le First Contentful Paint
- Mesurer le Largest Contentful Paint
- Analyser les ressources chargées

2. Test interface de chat :

- Temps d'initialisation du chat
- Chargement de l'historique
- Réactivité de l'interface

Résultats attendus :

- **Page d'accueil :**
 - First Contentful Paint < 1.5 secondes
 - Largest Contentful Paint < 2.5 secondes
 - Time to Interactive < 3 secondes
- **Interface chat :**

- Initialisation < 2 secondes
- Chargement historique < 1 seconde
- Réactivité fluide (60 FPS)

TP01.2 - Performance IA

Objectif : Mesurer les performances de génération IA.

Prérequis : Accès au chat, messages de test variés

Étapes :

1. Messages simples :

- "Hello" (message court)
- Mesurer le temps de réponse

2. Messages complexes :

- Texte long avec erreurs multiples
- Demande d'analyse grammaticale

3. Charge multiple :

- Plusieurs utilisateurs simultanés
- Messages en série rapide

Résultats attendus :

- **Réponses rapides** :

- Messages simples : < 3 secondes
- Messages complexes : < 8 secondes
- Variabilité faible (écart-type < 2s)

- **Gestion de la charge** :

- 20+ utilisateurs simultanés supportés
- Queue management transparent
- Pas de dégradation significative

TP01.3 - Performance mobile

Objectif : Valider l'expérience mobile.

Prérequis : Appareil mobile ou émulation, connexion 4G

Étapes :

1. Test sur mobile réel :

- iPhone/Android moyen de gamme
- Connexion 4G standard
- Mesurer les métriques Core Web Vitals

2. Test responsive :

- Différentes tailles d'écran
- Orientation portrait/paysage
- Fonctionnalités tactiles

Résultats attendus :

- **Performance mobile :**
 - First Input Delay < 100ms
 - Cumulative Layout Shift < 0.1
 - Interface fluide et responsive
 - **Expérience utilisateur :**
 - Adaptation parfaite aux écrans
 - Boutons tactiles appropriés (44px min)
 - Navigation intuitive
-

TP02 - Performance réseau

TP02.1 - Optimisation des ressources

Objectif : Valider l'optimisation des ressources statiques.

Prérequis : Outils d'analyse réseau (DevTools)

Étapes :

1. Analyse des ressources :

- Taille des bundles JavaScript
- Compression des images
- Mise en cache des ressources

2. Test CDN :

- Temps de réponse des assets
- Distribution géographique
- Headers de cache

Résultats attendus :

- **Optimisation ressources :**
 - Bundle JS principal < 500KB gzippé
 - Images optimisées (WebP/AVIF si supporté)
 - CSS critique inliné
- **Mise en cache efficace :**
 - Cache-Control headers appropriés
 - Versioning des assets (hashing)
 - Service Worker pour cache offline

TP02.2 - APIs et latence

Objectif : Mesurer la latence des APIs backend.

Prérequis : Accès aux APIs, outil de monitoring

Étapes :

1. Test latence standard :

- Appels API depuis différentes localisations
- Mesure des temps de réponse P50, P95, P99

2. Test sous charge :

- Montée en charge progressive
- Identification du point de saturation

Résultats attendus :

- **Latence acceptable** :
 - APIs standard : P95 < 500ms
 - API IA : P95 < 8 secondes
 - Health checks : < 100ms
 - **Scalabilité** :
 - Dégradation gracieuse sous charge
 - Auto-scaling fonctionnel
 - Circuit breakers actifs
-

Tests de compatibilité

TC01 - Navigateurs

TC01.1 - Navigateurs modernes

Objectif : Valider la compatibilité avec les navigateurs principaux.

Prérequis : Accès aux différents navigateurs

Étapes :

1. Chrome (dernière version) :

- Fonctionnalités complètes
- Performance optimale
- DevTools sans erreurs

2. Firefox (dernière version) :

- Interface identique
- Fonctionnalités IA opérationnelles

- Compatibilité CSS

3. Safari (si disponible) :

- Rendu correct
- WebKit spécificités gérées

4. Edge (dernière version) :

- Fonctionnement complet
- Performances acceptables

Résultats attendus :

- **Chrome** : Expérience de référence optimale
- **Firefox** : Fonctionnalités 100% compatibles
- **Safari** : Adaptation WebKit correcte
- **Edge** : Compatibilité complète
- **Performance homogène** entre navigateurs
- **Aucune fonctionnalité dégradée**

TC01.2 - Versions antérieures

Objectif : Tester la rétrocompatibilité.

Prérequis : Navigateurs en versions N-1, N-2

Étapes :

1. Test fonctionnalités critiques :

- Authentification
- Chat de base
- Navigation principale

2. Test graceful degradation :

- Fonctionnalités avancées
- Polyfills activés
- Messages informatifs

Résultats attendus :

- **Fonctionnalités essentielles** préservées
- **Dégradation gracieuse** pour features avancées
- **Messages d'information** si limitations
- **Pas de crash** ou erreurs bloquantes

TC02 - Appareils et résolutions

TC02.1 - Appareils mobiles

Objectif : Valider l'expérience sur différents appareils mobiles.

Prérequis : Émulateurs ou appareils réels

Étapes :

1. **Smartphones** :

- iPhone 12/13/14 (iOS Safari)
- Samsung Galaxy S21+ (Chrome Android)
- Pixel 6 (Chrome Android)

2. **Tablettes** :

- iPad (Safari)
- Android tablet (Chrome)

Résultats attendus :

- **Interface adaptée** à chaque taille d'écran
- **Navigation tactile** intuitive et fluide
- **Performance** acceptable sur hardware moyen
- **Fonctionnalités** complètes préservées
- **Orientation** portrait/paysage gérée

TC02.2 - Résolutions d'écran

Objectif : Tester différentes résolutions et densités.

Prérequis : Outils de test responsive

Étapes :

1. **Résolutions courantes** :

- 320px (mobile portrait)
- 768px (tablette)
- 1024px (desktop small)
- 1920px+ (desktop large)

2. **Densités d'écran** :

- Standard (1x)
- Retina/HiDPI (2x, 3x)

Résultats attendus :

- **Breakpoints** CSS bien définis
- **Images** adaptées aux densités
- **Texte** lisible à toutes les tailles
- **Interface** utilisable sans scroll horizontal
- **Éléments interactifs** suffisamment grands

Validation et critères d'acceptation

Critères de réussite globaux

Fonctionnalités critiques

Obligatoires pour validation :

- **Authentification** : 100% des scénarios passent
- **Chat IA** : Réponses générées avec qualité
- **Abonnements** : Cycle complet fonctionnel
- **Sécurité** : Aucune faille majeure détectée
- **Performance** : Métriques dans les seuils définis

Critères qualité

Seuils minimum acceptables :

- **Disponibilité** : > 99.5% sur 7 jours
- **Disponibilité** : > 99.5% sur 7 jours
- **Performance** : P95 < seuils définis
- **Sécurité** : Score OWASP > 95%
- **Compatibilité** : 95% navigateurs modernes
- **Accessibilité** : Score Lighthouse > 90%

Processus de validation

Étapes de validation

1. **Tests automatisés** : CI/CD pipeline vert
2. **Tests manuels** : Exécution des scénarios critiques
3. **Tests utilisateurs** : Validation UX avec panel test
4. **Tests sécurité** : Audit complet OWASP
5. **Tests performance** : Validation sous charge réelle

Critères de blocage

Blocants pour mise en production :

- Faille de sécurité critique (OWASP High/Critical)
- Perte de données utilisateur
- Impossibilité d'authentification
- Chat IA non fonctionnel
- Système de paiement défaillant

Critères d'amélioration

À corriger en priorité :

- Performance dégradée (hors seuils)

- Problème d'accessibilité majeur
 - Bug UX critique
 - Compatibilité navigateur manquante
-

Conclusion

Ce cahier de recettes couvre l'ensemble des aspects fonctionnels, techniques et qualité de la plateforme Penpal AI. L'exécution complète de ces tests garantit :

Qualité fonctionnelle

- Toutes les fonctionnalités utilisateur validées
- Expérience utilisateur fluide et intuitive
- Parcours d'onboarding optimal

Sécurité robuste

- Protection contre les vulnérabilités OWASP
- Authentification et autorisation sécurisées
- Données utilisateur protégées

Performance optimale

- Temps de réponse acceptables
- Scalabilité validée sous charge
- Expérience mobile fluide

Compatibilité étendue

- Support navigateurs modernes
- Adaptation multi-devices
- Accessibilité RGAA/WCAG

La validation réussie de ces tests constitue le **feu vert pour la mise en production** de nouvelles versions et garantit une expérience utilisateur de qualité entreprise pour la plateforme d'apprentissage linguistique Penpal AI.