

# Plan de Tests - HabitatsConnect

---

## 1. Stratégie de Test

### 1.1 Objectifs

- Valider le fonctionnement correct de toutes les fonctionnalités
- Détecter les bugs et régressions
- Assurer la qualité et la fiabilité de la plateforme
- Valider les performances et la sécurité

### 1.2 Périmètre

- Tests unitaires : Composants et fonctions
- Tests d'intégration : API et interactions
- Tests end-to-end : Parcours utilisateur complets
- Tests de performance : Charge et scalabilité
- Tests de sécurité : Vulnérabilités et conformité

## 2. Tests Fonctionnels

### 2.1 Module Authentification

#### **TU-AUTH-001 : Inscription utilisateur**

**Préconditions** : Application accessible

**Actions** :

1. Accéder à la page d'inscription
2. Remplir le formulaire avec données valides
3. Soumettre le formulaire

**Résultat attendu** : Compte créé, redirection dashboard

#### **TU-AUTH-002 : Connexion utilisateur**

**Préconditions** : Compte existant validé

**Actions** :

1. Accéder à la page de connexion
2. Saisir identifiants corrects
3. Valider la connexion

**Résultat attendu** : Accès au dashboard approprié

### 2.2 Module Propriétés

#### **TU-PROP-001 : Création d'annonce**

**Préconditions** : Compte propriétaire

**Actions** :

1. Accéder à "Ajouter annonce"
2. Remplir toutes les étapes du formulaire
3. Upload au moins 3 photos
4. Publier l'annonce

**Résultat attendu** : Annonce visible dans les recherches

## **TU-PROP-002 : Recherche propriétés**

**Préconditions** : Propriétés existantes

**Actions** :

1. Accéder à la page d'accueil
2. Saisir critères de recherche
3. Appliquer filtres
4. Trier les résultats

**Résultat attendu** : Résultats pertinents affichés

## 3. Tests d'Intégration

### 3.1 API Tests

#### **TI-API-001 : Création réservation**

**Préconditions** : Propriété disponible, utilisateur authentifié

**Actions** :

1. Envoyer requête POST /api/reservations
2. Avec données valides
3. Vérifier réponse 201

**Résultat attendu** : Réservation créée en base

#### **TI-API-002 : Upload images**

**Préconditions** : Compte propriétaire

**Actions** :

1. Upload fichier image via API
2. Vérifier stockage Firebase
3. Vérifier URL générée

**Résultat attendu** : Image accessible via URL

## 4. Tests End-to-End

### 4.1 Parcours Client

#### **TE2E-CLIENT-001 : Réservation complète**

**Préconditions** : Compte client, propriétés disponibles

**Actions** :

1. Recherche propriété
2. Consultation détail
3. Sélection dates
4. Saisie informations
5. Paiement simulé
6. Confirmation

**Résultat attendu** : Réservation confirmée, emails envoyés

## 4.2 Parcours Propriétaire

### TE2E-OWNER-001 : Gestion annonce

**Préconditions** : Compte propriétaire

**Actions** :

1. Création annonce
2. Réception demande réservation
3. Acceptation réservation
4. Communication avec client

**Résultat attendu** : Réservation confirmée, paiement capturé

## 5. Tests de Performance

### 5.1 Métriques cibles

- **Temps de chargement** : < 3 secondes
- **First Contentful Paint** : < 1.5 secondes
- **Time to Interactive** : < 3 secondes
- **Core Web Vitals** : Tous "Good"

### 5.2 Tests de charge

- **Utilisateurs simultanés** : 1000
- **Requêtes/minute** : 10000
- **Temps de réponse moyen** : < 500ms

## 6. Tests de Sécurité

### 6.1 Authentification

- Tentatives de connexion invalides
- Sessions expirées
- Accès non autorisé

### 6.2 Injection SQL

- Paramètres malformés

- Scripts XSS
- CSRF attacks

## 6.3 Conformité RGPD

- Droit à l'oubli
- Portabilité données
- Consentement explicite

# 7. Environnements de Test

## 7.1 Développement

- Tests unitaires automatisés
- Tests d'intégration locaux
- Debugging actif

## 7.2 Staging

- Tests end-to-end complets
- Tests de performance
- Validation métier

## 7.3 Production

- Monitoring continu
- Tests de fumée post-déploiement
- Rollback automatisé

# 8. Outils de Test

## 8.1 Tests unitaires

- **Framework** : Jest
- **Bibliothèque** : React Testing Library
- **Coverage** : > 80%

## 8.2 Tests E2E

- **Framework** : Playwright
- **Navigateurs** : Chrome, Firefox, Safari
- **Mobile** : Responsive testing

## 8.3 Performance

- **Outil** : Lighthouse
- **Monitoring** : Vercel Analytics
- **Load testing** : k6

# 9. Critères d'Acceptation

## 9.1 Tests réussis

- 100% tests unitaires passent
- 100% tests d'intégration passent
- 100% tests E2E critiques passent
- Performance dans les cibles
- 0 vulnérabilités sécurité critiques

## 9.2 Métriques qualité

- Code coverage > 80%
- Complexité cyclomatique < 10
- Debt technique < 5%
- Performance score > 90/100

# 10. Rapport de Tests

## 10.1 Structure du rapport

- Résumé exécutif
- Couverture fonctionnelle
- Anomalies détectées
- Métriques performance
- Recommandations

## 10.2 Suivi des anomalies

- Classification (Critique/Majeur/Mineur)
- Priorisation et plan de correction
- Validation des correctifs
- Régression testing