



Auteur : Christophe Pierrès	<b>Maison en place V2</b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		Page 1 / 15



## ***Plan de tests***

Diffusion : OpenClassrooms – Mentor : Cesare de Padua - Evalueur : Théo Lemoine


Version	Date	Auteur	Motif
1	22/08/2025	Christophe Pierrès	Création du document
2			

Auteur : Christophe Pierrès	<b>Maison en place V2</b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		Page 2 / 15

## Contenu

## Page

<b>1. Plan de Test .....</b>	<b>3</b>
1.1. Contexte .....	3
1.2. Fonctionnalités principales .....	3
1.3. Approche méthodologique pour les tests (TDD vs BDD).....	3
1.3.1. Behavior-Driven Development (BDD).....	3
1.3.2. Test-Driven Development (TDD) .....	3
1.3.3. Choix retenu : les deux ! .....	3
1.3.4. Quand les utiliser ? .....	4
1.3.5. Comment les combiner efficacement .....	4
1.4. Scénarios BDD avec Gherkin .....	6
1.4.1. Structure Gherkin .....	6
1.4.2. Fonctionnalité 1 : Authentification .....	6
1.4.2.1. Avantages du scénario générique pour l'authentification sociale avec `Exemples` .....	10
1.4.3. Fonctionnalité 2 : Simulateur de réalité augmentée .....	10
1.4.3.1. Fonctionnalité à ne pas tester : Algorithmes de rendu 3D internes.....	13
1.5. Décliner les tests BDD en TDD .....	13
1.5.1. Exemples de TDD pour scénarios d'authentification .....	14
1.5.1.1. Scénario : Connexion réussie (email/mot de passe).....	14
1.5.1.2. Scénario : Mot de passe incorrect .....	14
1.5.1.3. Scénario : Connexion sociale via OAuth .....	14

Auteur : Christophe Pierrès	<b>Maison en place V2</b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		Page 3 / 15

## 1. Plan de Test

### 1.1. Contexte

L'application MaisonEnPlace permet aux utilisateurs d'essayer virtuellement des meubles et objets de décoration dans leur propre environnement, grâce à la réalité augmentée. La V1 de l'application a été développée rapidement pour prouver le concept, mais manque de tests, ce qui a entraîné de nombreux bugs signalés par le service client.

La V2 vise à améliorer la fiabilité, la sécurité et la disponibilité de l'application.

### 1.2. Fonctionnalités principales

- **Authentification** : Connexion via login/mot de passe, Connexion sociale (Google ou Apple)
- **Catalogue produit** : Navigation, filtrage et recherche de produits
- **Simulateur de réalité augmentée** : Placement de meubles 3D dans l'environnement réel
- **Tunnel d'achat** : Ajout au panier et paiement

### 1.3. Approche méthodologique pour les tests (TDD vs BDD)

#### 1.3.1. Behavior-Driven Development (BDD)

Gherkin est un formalisme pour communiquer entre équipes techniques et fonctionnelles.

- Développement dirigé par le comportement utilisateur
- Langage naturel (Gherkin) compréhensible par tous
- Focus sur la valeur métier

Une fois validés au niveau fonctionnel, les scénarios peuvent être codés et automatisés via cucumber.

Le feedback du résultat des tests est compréhensible par tout le monde.


On peut alors facilement échanger lorsque les scénarios changent et que des bugs se produisent.

#### 1.3.2. Test-Driven Development (TDD)

- Développement dirigé par les tests techniques
- Cycle : Red → Green → Refactor
- Focus sur l'implémentation correcte du code (design testable)

#### 1.3.3. Choix retenu : les deux !

La V2 doit inclure de nouvelles fonctionnalités (authentification sociale, réalité augmentée, tunnel d'achat). Il est important de comprendre précisément les scénarios d'utilisation, leur articulation (BDD) et comment les implémenter pour les rendre testables facilement (TDD).

Auteur : Christophe Pierrès	<b>Maison en place V2</b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		Page 4 / 15

Gherkin (BDD) et TDD sont complémentaires - ils agissent à des niveaux différents et se renforcent mutuellement.

Gherkin (BDD) répond à « quoi » et « pourquoi » au niveau métier :

- Décrit le comportement attendu en langage naturel, partageable avec métiers/PO/design/QA.
- Sert de spécification vivante et de tests d'acceptation (exécutables via Cucumber, et combinable avec d'autres technos pour piloter l'IHM telles que Selenium, ou même Cypress).
- Pertinent pour parcours utilisateurs, règles métier, intégrations fonctionnelles.

TDD répond à « comment » au niveau technique :

- Guide la conception du code (unit tests) avec le cycle Red–Green–Refactor.
- Donne un feedback rapide, un design modulaire, et une base de tests robuste.
- Pertinent pour classes, fonctions, API internes, logique algorithmique.

A l'occasion du développement des évolutions, il est particulièrement important de bien décrire les scénarios (BDD) et de renforcer la fiabilité et la testabilité avec un bon design technique permis par le TDD.

#### **1.3.4. Quand les utiliser ?**

Utiliser Gherkin/BDD quand :

- Il faut aligner les parties prenantes sur le comportement attendu.
- Les exigences sont riches en règles métier ou en parcours utilisateur.
- On veut des tests d'acceptation lisibles et durables (niveau feature, pas détails d'UI).
- On valide les critères d'acceptation en amont (avant dev).

Utiliser TDD quand :

- On implémente de la logique applicative, des services, des adaptateurs, des validations.
- On a besoin de feedback rapide sur le design et la testabilité.
- On travaille sur des composants internes (peu visibles métier) ou algorithmiques.

Leur Maintenance est séparée :


- BDD évolue avec les besoins métier
- TDD évolue avec l'architecture technique
- Pas de couplage entre les deux approches

#### **1.3.5. Comment les combiner efficacement**

1) **Avant de coder** : écrire/valider 1–3 scénarios Gherkin par user story (niveau métier, stables).

2) **Pendant le dev** :


- Démarrer par TDD pour les unités nécessaires à satisfaire les scénarios.

Auteur : Christophe Pierrès	<b>Maison en place V2</b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		Page 5 / 15

- Écrire des tests d'intégration ciblés si besoin (ex : service).

3) **Après implémentation** : automatiser les scénarios Gherkin (Cucumber) pour l'acceptation et la non-régression.

4) **CI/CD**: exécuter d'abord unit/intégration (rapides), puis scénarios Gherkin (plus lents, e2e/semi-e2e).

Auteur : Christophe Pierrès	<b>Maison en place V2</b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		Page 6 / 15

## 1.4. Scénarios BDD avec Gherkin

### 1.4.1. Structure Gherkin

- **Feature** : Description de la fonctionnalité métier
- **Given** : Contexte initial (état du système)
- **When** : Action déclenchée par l'utilisateur
- **Then** : Résultat attendu/comportement observable

### 1.4.2. Fonctionnalité 1 : Authentification

Note : les mots clés et formalismes Gherkin ont leur équivalent pour de nombreuses langues.


**Feature** : Authentification des utilisateurs

**As/En tant qu'** utilisateur de MaisonEnPlace


**I want/Je veux** me connecter à l'application

**Then/Afin** d'accéder aux fonctionnalités personnalisées

Etape	Résultat attendu	Résultat obtenu (à remplir par testeur)	Conclusion (à remplir par testeur)
<b>Background</b> : initialisation application <b>Given</b> : L'application fonctionne <b>AND</b> L'écran de login s'affiche	Page de login visible avec formulaires de connexion + boutons : `Se connecter avec Google` et `Se connecter avec Apple`		
<u>Scénario 1</u> : Connexion réussie avec login/mot de passe <b>Given</b> L'utilisateur possède un compte valide <b>When</b> : Il saisit ses identifiants corrects <b>And</b> : clique sur bouton `Se connecter` <b>Then</b> : Il accède à l'écran d'accueil	Redirection vers l'écran d'accueil avec message de bienvenue		
<u>Scénario 2</u> : Échec de connexion - mot de passe incorrect <b>Given</b> : L'utilisateur possède un compte valide <b>When</b> : Il saisit un mot de passe incorrect <b>And</b> : clique sur bouton `Se connecter`	Message "Identifiants incorrects" + champs vidés		


Auteur : Christophe Pierrès	<b>Maison en place V2</b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		Page 7 / 15

Etape	Résultat attendu	Résultat obtenu (à remplir par testeur)	Conclusion (à remplir par testeur)
<b>Then</b> : Un message d'erreur s'affiche <b>And</b> : les champs saisis sont vidés			
<u>Scenario 3a</u> : Création de compte avec mot de passe non conforme <b>Given</b> l'utilisateur n'a pas de compte <b>And</b> il clique sur le bouton "S'inscrire" <b>When</b> il remplit le formulaire avec un email valide <b>And</b> il saisit le mot de passe "<password>" <b>Then</b> le système doit rejeter le mot de passe <b>And</b> un message d'erreur "<error_message>" doit s'afficher <b>And</b> le bouton de validation doit rester désactivé <b>Exemples:</b>   password   error_message     123456   au moins 8 caractères     password   une majuscule et un chiffre     PASSWORD123   au moins une minuscule     Password   au moins un chiffre     Password123   un caractère spécial	Compte créé + redirection vers l'accueil		
Scenario 3b : Validation temps réel de la complexité du mot de passe <b>Given</b> l'utilisateur est sur le formulaire d'inscription <b>When</b> il commence à taper son mot de passe <b>Then</b> un indicateur de force doit s'afficher <b>And</b> les critères de validation doivent être affichés <b>And</b> chaque critère respecté doit être coché visuellement			
Scenario 3c : Création de compte avec mot de passe sécurisé <b>Given</b> l'utilisateur n'a pas de compte <b>And</b> il clique sur le bouton "S'inscrire" <b>When</b> il remplit le formulaire avec un email valide "user@test.com"	Compte créé + redirection vers l'accueil		

Auteur : Christophe Pierrès	<b>Maison en place V2</b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		Page 8 / 15

Etape	Résultat attendu	Résultat obtenu (à remplir par testeur)	Conclusion (à remplir par testeur)
<p><b>And</b> il saisit un mot de passe conforme aux règles "SecurePass123!"</p> <p><b>And</b> il confirme son mot de passe "SecurePass123!"</p> <p><b>And</b> il accepte les conditions d'utilisation</p> <p><b>And</b> il clique sur "Créer mon compte"</p> <p><b>Then</b> un compte doit être créé avec succès</p> <p><b>And</b> il doit être connecté automatiquement</p> <p><b>And</b> il doit être redirigé vers l'écran d'accueil</p> <p><b>And</b> son mot de passe doit être haché avec un algorithme sécurisé</p>			
<p><u>Scénario Outline</u> : Connexion sociale via compte social existant (Google ou Apple selon bouton "Se connecter avec &lt;provider&gt;" cliqué)</p> <p><b>Given</b> : Compte social &lt;provider&gt; existant</p> <p><b>When</b> : Clic " Se connecter avec &lt;provider&gt;"</p> <p><b>And</b> : le processus OAuth "&lt;provider&gt;" se déroule avec succès</p> <p><b>And</b> : les informations OAuth autorisées sont transférées (email, nom, prénom)</p> <p><b>Then</b> : Lien compte MaisonEnPlace (lien ou création si nouveau)</p> <p><b>And</b> : Connexion directe</p> <p><b>And</b> : il doit être redirigé vers l'écran d'accueil</p> <p><b>Examples</b> :</p> <pre>   provider     Google       Apple      </pre>	Compte MaisonEnPlace créé/lié + redirection vers l'accueil		



Auteur : Christophe Pierrès	<b>Maison en place V2</b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		Page 9 / 15

Etape	Résultat attendu	Résultat obtenu (à remplir par testeur)	Conclusion (à remplir par testeur)
<p><u>Scénario Outline</u> : Connexion sociale avec création de compte "à la volée"</p> <p><b>Given</b> : l'utilisateur n'a pas de compte "&lt;provider&gt;"</p> <p><b>When</b> il clique sur le bouton "Se connecter avec &lt;provider&gt;"</p> <p><b>And</b> Pop-up/redirection vers OAuth du &lt;provider&gt;</p> <p><b>And</b> il complète le processus de création de compte dans le flux OAuth "&lt;provider&gt;" – <b>non testé car externe (transparent pour nous)</b></p> <p><b>And</b> il autorise l'accès à MaisonEnPlace avec les informations requises (email, nom, prénom)</p> <p><b>Then</b> un nouveau compte MaisonEnPlace doit être créé automatiquement avec lien infos social</p> <p><b>And</b> : Connexion directe</p> <p><b>And</b> : il doit être redirigé vers l'écran d'accueil</p> <p><b>Exemples</b> :</p> <pre>  provider     Google       Apple     </pre>	Compte MaisonEnPlace créé + redirection vers l'accueil		
<p><u>Scénario Outline</u> : Échec Connexion sociale OAuth &lt;provider&gt; (Google ou Apple selon bouton `Se connecter avec &lt;provider&gt;` cliqué)</p> <p><b>Given</b> : l'utilisateur n'a pas de compte "&lt;provider&gt;"</p> <p><b>When</b> il clique sur le bouton "Se connecter avec &lt;provider&gt;"</p> <p><b>And</b> Pop-up/redirection vers OAuth du &lt;provider&gt;</p> <p><b>And</b> il ne complète pas le processus de création de compte dans le flux OAuth "&lt;provider&gt;", et/ou ne donne pas autorisation à MaisonEnPlace – <b>non testé car externe (transparent pour nous)</b></p> <p><b>Then</b> : Analyse retour erreur OAuth</p> <p><b>And</b> : Retour page login</p>	Retour page connexion avec message d'erreur approprié « connexion sociale annulée »		

Etape	Résultat attendu	Résultat obtenu (à remplir par testeur)	Conclusion (à remplir par testeur)
<b>And</b> : afficher message erreur approprié  <b>Exemples</b> : <pre>  provider     Google       Apple     </pre>			

#### 1.4.2.1. Avantages du scénario générique pour l'authentification sociale avec `Exemples`

1. **Réutilisabilité** : Un seul scénario couvre Google ET Apple
2. **Maintenance facilitée** : Modifications en un seul endroit
3. **Cohérence** : Même logique de test pour tous les providers
4. **Évolutivité** : Facile d'ajouter d'autres providers (Facebook, Github, etc.)
5. **Lisibilité** : Structure claire et compréhensible

#### 1.4.3. Fonctionnalité 2 : Simulateur de réalité augmentée

**Feature**: Simulateur de réalité augmentée (RA) pour placement de meubles

**En tant qu'** utilisateur connecté

**Je veux** visualiser des meubles dans mon environnement réel

**Afin de** vérifier l'adéquation avant achat

Etape	résultat attendu	résultat obtenu (à remplir par testeur)	Conclusion (à remplir par testeur)
<u>Background</u> : Initialisation RA <b>Given</b> Utilisateur authentifié <b>And</b> permissions caméra <b>And</b> l'appareil supporte ARCore/ARKit <b>And</b> produit sélectionné	Prérequis techniques validés pour lancement RA		
Scénario 1 : Initialisation simulateur <b>Given</b> l'utilisateur a sélectionné un canapé "ModernSofa3D" <b>When</b> il clique sur "Essayer en RA" <b>Then</b> la caméra doit s'activer <b>And</b> l'interface RA doit s'afficher avec les contrôles <b>And</b> le message "Balayez lentement votre environnement" doit apparaître	Interface RA complète avec tous les éléments de guidage utilisateur		

Etape	résultat attendu	résultat obtenu (à remplir par testeur)	Conclusion (à remplir par testeur)
<b>And</b> les indicateurs de tracking doivent être visibles			
<p>Scénario 2 : Reconnaissance et mapping de l'environnement</p> <p><b>Given</b> le simulateur RA est lancé</p> <p><b>When</b> l'utilisateur effectue un balayage lent de son environnement</p> <p><b>And</b> la caméra détecte au moins 3 surfaces planes</p> <p><b>Then</b> les surfaces détectées doivent être visualisées par des grilles</p> <p><b>And</b> un indicateur de qualité de tracking doit passer au vert</p> <p><b>And</b> le message "Surfaces détectées - Tapez pour placer" doit s'afficher</p> <p><b>And</b> les points de tracking RA doivent être visibles dans l'interface</p>	Mapping environnemental réussi avec feedback visuel complet		
<p>Scénario 3: Chargement modèle 3D</p> <p>Chargement et prévisualisation du modèle 3D</p> <p><b>Given</b> l'environnement est mappé avec succès</p> <p><b>When</b> le modèle 3D commence à se charger</p> <p><b>Then</b> un indicateur de progression doit s'afficher</p> <p><b>And</b> une prévisualisation fantôme du modèle doit apparaître</p> <p><b>And</b> les dimensions réelles doivent être affichées (L x l x H)</p> <p><b>And</b> le modèle doit suivre les mouvements de la caméra avant placement</p>	Modèle 3D chargé avec prévisualisation interactive et informations techniques		
<p>Scénario 4: Placement précis du meuble avec validation spatiale</p> <p><b>Given</b> le modèle 3D est chargé et en prévisualisation</p> <p><b>And</b> une surface plane appropriée est détectée</p> <p><b>When</b> l'utilisateur tape sur une zone de la surface détectée</p> <p><b>Then</b> le système doit vérifier que l'espace est suffisant pour le meuble</p>	Placement réussi avec tous les effets de réalisme RA		

Etape	résultat attendu	résultat obtenu (à remplir par testeur)	Conclusion (à remplir par testeur)
<p><b>And</b> le meuble doit se placer avec ancrage spatial</p> <p><b>And</b> des ombres réalistes doivent être générées selon l'éclairage ambiant</p> <p><b>And</b> les dimensions doivent respecter l'échelle réelle</p>			
<p>Scénario 5: Détection d'espace insuffisant</p> <p><b>Given</b> le modèle 3D est en prévisualisation</p> <p><b>When</b> l'utilisateur tape sur une surface trop petite pour le meuble</p> <p><b>Then</b> un message "Espace insuffisant" doit s'afficher</p> <p><b>And</b> la zone inadéquate doit être surlignée en rouge</p> <p><b>And</b> des suggestions d'espaces alternatifs doivent être proposées</p> <p><b>And</b> le placement ne doit pas s'effectuer</p>	Gestion intelligente des contraintes spatiales		
<p>Scénario 6: Manipulation avancée du meuble placé</p> <p><b>Given</b> un meuble est placé et ancré dans l'environnement</p> <p><b>When</b> l'utilisateur utilise les gestes de manipulation (gestures touch screen)</p> <p><b>Then</b> la rotation doit fonctionner avec gestes à deux doigts</p> <p><b>And</b> le redimensionnement doit respecter les proportions réelles</p> <p><b>And</b> le repositionnement doit maintenir l'ancrage au sol</p> <p><b>And</b> l'éclairage et les ombres doivent s'adapter en temps réel</p> <p><b>And</b> les performances doivent rester fluides</p>	Interactions naturelles avec performances maintenues		
<p>Scénario 7 : Test de performance avec plusieurs meubles</p> <p><b>Given</b> l'environnement est mappé</p> <p><b>When</b> l'utilisateur place successivement 3 meubles différents</p> <p><b>Then</b> chaque placement doit rester fluide</p>	Performances optimales avec charge multiple		

Etape	résultat attendu	résultat obtenu (à remplir par testeur)	Conclusion (à remplir par testeur)
<b>And</b> aucun crash ne doit survenir			
Scénario 8 : Retour au catalogue avec sauvegarde de session <b>Given</b> l'utilisateur est dans le simulateur RA avec un meuble placé <b>When</b> il clique sur "Retour au catalogue" <b>Then</b> une popup "Sauvegarder cette disposition ?" doit apparaître <b>When</b> il confirme la sauvegarde <b>Then</b> il doit revenir au catalogue <b>And</b> les filtres précédents doivent être conservés <b>And</b> la disposition RA doit être sauvegardée pour consultation ultérieure	Gestion de session complète avec persistance des données		

#### 1.4.3.1. Fonctionnalité à ne pas tester : Algorithmes de rendu 3D internes

Les algorithmes internes de rendu 3D utilisés par le moteur de réalité augmentée ne seront pas testés directement dans notre plan de test pour les raisons suivantes :

1. Ces algorithmes sont généralement fournis par des bibliothèques tierces spécialisées qui ont leurs propres suites de tests.
2. Tester ces algorithmes nécessiterait des connaissances très spécifiques en graphisme 3D et en mathématiques.
3. Nous nous concentrerons plutôt sur les interactions utilisateur et les résultats visibles, plutôt que sur les calculs internes.

### 1.5. Décliner les tests BDD en TDD

Pour chaque scénario Gherkin (acceptation), on liste 3 à 6 tests unitaires couvrant :


- Règles métier (succès/échec, messages génériques).
- Intégrations techniques (hash, JWT, OAuth).
- Résilience/sécurité (rate limiting, tokens expirés, comptes verrouillés).

Bonnes pratiques :

- Garder les scénarios Gherkin stables et au niveau métier ; déduire les cas techniques (TDD) en « happy path + erreurs + sécurité ».
- Nommer les tests de manière explicite pour assurer la traçabilité directe entre un critère d'acceptation et 1–N tests unitaires.

En CI :

- Étape 1: tests unitaires (rapides, TDD).

Auteur : Christophe Pierrès	<b>Maison en place V2</b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		Page 14 / 15

- Étape 2: tests d'intégration ciblés (repo en mémoire, clocks/mocks).
- Étape 3: scénarios Gherkin automatisés (acceptation, parcours).

### 1.5.1. Exemples de TDD pour scénarios d'authentification

#### 1.5.1.1. Scénario : Connexion réussie (email/mot de passe)

**But Gherkin** : Given page de connexion, When email+mot de passe valides, Then redirection + message "Bienvenue".

**TDD ciblé (unit/integration) :**

- AuthenticationService: valide les identifiants, émet un jeton.
- PasswordHasher : vérifie le hachage.
- TokenService : émet un token avec exp court.
- UserRepository : retourne l'utilisateur par email.
- RateLimiter : non bloquant en cas de succès.

**Exemples de tests unitaires :**

```
void authenticate_success_whenCredentialsValid();
void token_contains_expected_claims_on_success();
void rateLimiter_notLocked_on_first_attempt();
```

#### 1.5.1.2. Scénario : Mot de passe incorrect

**But (Gherkin)** : Given page de connexion, When mot de passe incorrect, Then rester sur page + message d'erreur générique.

**TDD ciblé :**

- AuthenticationService : échec sans révéler la cause (user inconnu vs mauvais mot de passe).
- PasswordHasher : retourne false.
- RateLimiter : incrémente les échecs; verrouille après N tentatives.

**Exemples de tests unitaires :**

```
// Java
void authenticate_fails_whenPasswordIsWrong();
void failure_is_generic_for_unknown_or_wrongPassword();
void rateLimiter_locks_after_max_failures_within_window();
```

#### 1.5.1.3. Scénario : Connexion sociale via OAuth


**But (Gherkin)** : When "Se connecter avec Google" et succès Google, Then redirection + profil associé.

**TDD ciblé :**

- OAuthVerifier: valide le token/id\_token Google (signature, aud, exp).
- OAuthLinker : associe l'identité Google à l'utilisateur (création/liaison).
- TokenService : émet le JWT applicatif.
- UserRepository : findByOAuthProviderId, createIfAbsent.

**Exemples de tests unitaires :**

```
// Java
void oauth_authenticates_whenIdTokenValid_andKnownUser();
void oauth_createsAndLinksUser_whenFirstLogin();
void oauth_rejects_whenIdTokenInvalid_orAudienceMismatch();
```

Auteur : Christophe Pierrès	<b><i>Maison en place V2</i></b>	
Révision 1 - 25/08/2025 17:43	Plan de tests	
Référence : p12_1_test_202508.docx		<b>Page 15 / 15</b>