Fiches de Test

EasyBooking

Plateforme de réservation de salles

TP Tests Logiciels 2024-2025

Informations Générales

Projet	EasyBooking
Version	1.0.0
Testeur	
Environnement	Local (localhost:3000)

PARTIE 1 : TESTS UNITAIRES

Fiche U-01 : Validation du schéma de connexion (loginSchema)

ID	U-01
Fonctionnalité	Authentification - Connexion
Priorité	Haute
Fichier de test	tests/unit/unit.test.tsx

Cas de test

#	Description	Données d'entrée	Résultat attendu	Statut
1	Email et mot de passe valides	email: "test@example.com" password: "password123"	Validation réussie	■
2	Email invalide	email: "invalid-email" password: "password123"	Erreur de validation	■
3	Email vide	email: "" password: "password123"	Erreur de validation	■
4	Mot de passe trop court	email: "test@example.com" password: "123"	Erreur de validation	■
5	Mot de passe vide	email: "test@example.com" password: ""	Erreur de validation	■
6	Les deux champs vides	email: "" password: ""	Erreur de validation	■

Commande : npm run test:unit -- --testNamePattern="loginSchema"

Fiche U-02 : Validation du schéma d'inscription (registerSchema)

ID	U-02
Fonctionnalité	Authentification - Inscription
Priorité	Haute
Fichier de test	tests/unit/unit.test.tsx

Cas de test

#	Description	Données d'entrée	Résultat attendu	Statut

1	Données valides	name: "John" email: "john@test.com" password: "Pass123!" confirmPassword: "Pass123!"	Validation réussie	■
2	Nom trop court	name: "J" email: "john@test.com"	Erreur de validation	■
3	Email invalide	name: "John" email: "invalid"	Erreur de validation	■
4	Mot de passe trop court	password: "123" confirmPassword: "123"	Erreur de validation	■
5	Mots de passe différents	password: "Pass123!" confirmPassword: "Different!"	Erreur de validation	■
6	Champs vides	Tous les champs vides	Erreur de validation	■

Commande : npm run test:unit -- --testNamePattern="registerSchema"

Fiche U-03 : Validation du schéma de réservation (bookingSchema)

ID	U-03
Fonctionnalité	Réservation de salle
Priorité	Haute
Fichier de test	tests/unit/unit.test.tsx

Cas de test

#	Description	Données d'entrée	Résultat attendu	Statut
1	Réservation valide	roomId: "uuid" date: "2026-01-20" startTime: "09:00" endTime: "10:00"	Validation réussie	■
2	roomId manquant	roomId: "" date: "2026-01-20"	Erreur de validation	■
3	Date invalide	date: "invalid"	Erreur de validation	■
4	Heure de début invalide	startTime: "25:00"	Erreur de validation	■
5	Heure de fin avant début	startTime: "10:00" endTime: "09:00"	Erreur de validation	■
6	Date passée	date: "2020-01-01"	Erreur de validation	■

Commande : npm run test:unit -- --testNamePattern="bookingSchema"

Fiche U-04 : Fonctions utilitaires

ID	U-04
Fonctionnalité	Utilitaires
Priorité	Moyenne
Fichier de test	tests/unit/unit.test.tsx

#	Fonction	Description	Statut
1	cn()	Fusion classes CSS	■
2	cn()	Classes conditionnelles	■
3	formatTime()	Format heure (14:30 → 14h30)	■
4	formatDate()	Format date (2026-01-14 → 14/01/2026)	■
5	generateTimeSlots()	Créneaux horaires	■
6	isFutureDate()	Date future	■
7	isFutureDate()	Date passée	■
8	getTodayString()	Date du jour (YYYY-MM-DD)	■

Commande : npm run test:unit -- --testNamePattern="Utilitaires"

Fiche U-05 : Composants UI

ID	U-05
Fonctionnalité	Composants d'interface
Priorité	Moyenne
Fichier de test	tests/unit/unit.test.tsx

#	Composant	Description	Statut
1	Button	Rendu basique	■
2	Button	Variant destructive	■
3	Button	État disabled	■
4	Input	Rendu basique	■
5	Input	Type password	■
6	Badge	Rendu avec texte	■
7	Alert	Message d'erreur	■
8	Alert	Message succès	■

Commande : npm run test:unit -- --testNamePattern="Composants UI"

PARTIE 2 : TESTS D'INTÉGRATION

Fiche I-01 : API des salles (GET /api/rooms)

ID	I-01
Fonctionnalité	Liste des salles
Priorité	Haute
Fichier de test	tests/integration/rooms.test.ts

#	Description	Méthode	URL	Résultat attendu	Statut
1	Récupérer toutes les salles	GET	/api/rooms	200 + liste des salles	■
2	Filtrer par capacité	GET	/api/rooms?capacity=10	200 + salles filtrées	■
3	Filtrer par équipement	GET	/api/rooms?equipment=projecto	200 + salles avec projecteur	■
4	Récupérer une salle par ID	GET	/api/rooms/[id]	200 + détails salle	■
5	ID inexistant	GET	/api/rooms/invalid-id	404	■
6	Vérifier structure réponse	GET	/api/rooms	Champs: id, name, capacity	■

Commande : npm run test:integration -- --testNamePattern="rooms"

Fiche I-02 : API des réservations (CRUD /api/bookings)

ID	I-02
Fonctionnalité	Gestion des réservations
Priorité	Haute
Fichier de test	tests/integration/bookings.test.ts

#	Description	Méthode	Résultat attendu	Statut
1	Créer une réservation	POST	201 + réservation créée	■
2	Données incomplètes	POST	400 + erreur validation	■
3	Conflit de créneau	POST	409 + erreur conflit	■
4	Lister mes réservations	GET	200 + liste réservations	■
5	Détails réservation	GET	200 + détails	■
6	Annuler réservation	DELETE	200 + confirmation	■
7	Annuler réservation inexistante	DELETE	404	■
8	Date passée	POST	400 + erreur	■
9	Heure fin avant début	POST	400 + erreur	■
10	Réservation sans auth	POST	401	■

Commande : npm run test:integration -- --testNamePattern="bookings"

Fiche I-03 : API d'authentification (/api/auth)

ID	I-03
Fonctionnalité	Authentification
Priorité	Haute
Fichier de test	tests/integration/auth.test.ts

Cas de test - Inscription

#	Description	Données	Résultat attendu	Statut
1	Inscription valide	name, email, password, confirmPassword	201 + utilisateur créé	■
2	Email déjà utilisé	Email existant	409 + erreur	■
3	Email invalide	email: "invalid"	400 + erreur validation	■
4	Mot de passe trop court	password: "123"	400 + erreur validation	■
5	Mots de passe différents	password ≠ confirmPassword	400 + erreur	■
6	Champs manquants	Données incomplètes	400 + erreur	■

Cas de test - Connexion

#	Description	Données	Résultat attendu	Statut
7	Connexion valide	email, password corrects	200 + token	■
8	Email incorrect	Email inexistant	401	■
9	Mot de passe incorrect	Mauvais password	401	■
10	Champs vides	email: "", password: ""	400	■
11	Format email invalide	email: "not-an-email"	400	■
12	Déconnexion	POST /api/auth/logout	200	■

Commande : npm run test:integration -- --testNamePattern="auth"

PARTIE 3 : TESTS E2E (END-TO-END)

Fiche E-01 : Page d'accueil

ID	E-01
Fonctionnalité	Navigation - Accueil
Priorité	Haute
Fichier de test	tests/e2e/auth.spec.ts

#	Description	Actions	Résultat attendu	Statut
1	Affichage page d'accueil	Accéder à "/"	Page avec titre EasyBooking	■
2	Boutons de navigation	Vérifier présence	Boutons Connexion/Inscription visibles	■
3	Navigation vers inscription	Cliquer "Créer un compte"	Redirection vers /register	■
4	Navigation vers connexion	Cliquer "Se connecter"	Redirection vers /login	■

Commande : npm run test:e2e -- --grep="Page d'accueil"

Fiche E-02 : Formulaire d'inscription

ID	E-02
Fonctionnalité	Inscription utilisateur
Priorité	Haute
Fichier de test	tests/e2e/auth.spec.ts

#	Description	Actions	Résultat attendu	Statut
1	Affichage formulaire	Accéder à /register	Champs nom, email, password, confirm	■
2	Email invalide	Entrer "invalid-email"	Message d'erreur affiché	■
3	Mot de passe trop court	Entrer password < 8 car	Message d'erreur affiché	■
4	Mots de passe différents	password ≠ confirmPassword	Message d'erreur affiché	■
5	Lien vers connexion	Cliquer "Déjà un compte"	Redirection vers /login	■

Commande : npm run test:e2e -- --grep="inscription"

Fiche E-03 : Formulaire de connexion

ID	E-03
Fonctionnalité	Connexion utilisateur

Priorité	Haute
Fichier de test	tests/e2e/auth.spec.ts

#	Description	Actions	Résultat attendu	Statut
1	Affichage formulaire	Accéder à /login	Champs email et password	■
2	Champs vides	Cliquer "Se connecter"	Reste sur la page	■
3	Identifiants incorrects	Email/password invalides	Message d'erreur	■
4	Lien vers inscription	Cliquer "Créer un compte"	Redirection vers /register	■
5	Bouton désactivé pendant soumission	Soumettre formulaire	Bouton disabled temporairement	■

Commande : npm run test:e2e -- --grep="connexion"

Fiche E-04 : Navigation générale

ID	E-04
Fonctionnalité	Navigation
Priorité	Moyenne
Fichier de test	tests/e2e/navigation.spec.ts

#	Description	Actions	Résultat attendu	Statut
1	Page d'accueil charge	GET "/"	Status 200, main visible	■
2	Logo redirige vers accueil	Cliquer logo	URL = "/"	■
3	Bouton retour navigateur	Naviguer puis retour	Page précédente affichée	■
4	Pages protégées	Accéder /rooms sans auth	Redirection vers /login	■
5	Route inexistante	Accéder /page-inexistante	404 ou redirection	■

Commande : npm run test:e2e -- --grep="Navigation"

Fiche E-05 : Responsive Design

ID	E-05
Fonctionnalité	Affichage responsive
Priorité	Moyenne
Fichier de test	tests/e2e/navigation.spec.ts

#	Description	Actions	Résultat attendu	Statut
1	Affichage mobile	Viewport 375x667	Page lisible, pas de scroll horizontal	■
2	Affichage tablette	Viewport 768x1024	Layout adapté	■

3	Formulaire login mobile	Viewport 375x667	Formulaire utilisable	■

Commande : npm run test:e2e -- --grep="Responsive"

Fiche E-06 : Accessibilité

ID	E-06
Fonctionnalité	Accessibilité
Priorité	Moyenne
Fichier de test	tests/e2e/auth.spec.ts

#	Description	Actions	Résultat attendu	Statut
1	Labels accessibles	Vérifier attributs	Tous les champs ont des labels	■
2	Navigation clavier	Tab à travers formulaire	Focus visible sur chaque élément	■

Commande : npm run test:e2e -- --grep="Accessibilité"

Fiche E-07 : Sécurité de base

ID	E-07
Fonctionnalité	Sécurité
Priorité	Haute
Fichier de test	tests/e2e/navigation.spec.ts

#	Description	Actions	Résultat attendu	Statut
1	Mot de passe masqué	Vérifier type input	type="password"	■
2	Autocomplete approprié	Vérifier attributs	autocomplete configuré	■

Commande : npm run test:e2e -- --grep="Sécurité"

PARTIE 4 : TESTS DE PERFORMANCE

Fiche P-01 : Performance des pages

ID	P-01
Fonctionnalité	Temps de chargement
Priorité	Haute
Fichier de test	tests/performance/load-test.js

#	Page	Seuil	VUs	Résultat attendu	Statut
1	Page d'accueil (/)	< 2s	10-20	95% requêtes < 2s	■
2	Page connexion (/login)	< 2s	10-20	95% requêtes < 2s	■
3	Page inscription (/register)	< 2s	10-20	95% requêtes < 2s	■

Commande : npm run test:perf:quick

Fiche P-02 : Performance des API

ID	P-02
Fonctionnalité	Temps de réponse API
Priorité	Haute
Fichier de test	tests/performance/load-test.js

#	Endpoint	Méthode	Seuil	Résultat attendu	Statut
1	/api/rooms	GET	< 1.5s	95% requêtes < 1.5s	■
2	/api/bookings	GET	< 1.5s	95% requêtes < 1.5s	■
3	/api/auth/login	POST	< 2s	Réponse dans les temps	■
4	/api/auth/register	POST	< 3s	Réponse dans les temps	■

Commande : npm run test:perf

Fiche P-03 : Tests de charge

ID	P-03
Fonctionnalité	Résistance à la charge
Priorité	Haute
Fichier de test	tests/performance/load-test.js

Scénario de charge

Phase	Durée	Utilisateurs virtuels	Description
1	30s	0 → 10	Montée progressive
2	1min	10	Charge stable
3	30s	10 → 20	Augmentation
4	1min	20	Charge haute
5	30s	20 → 0	Descente

Critères de succès

Métrique	Seuil	Statut
Temps de réponse (p95)	< 2s	■
Taux d'erreur	< 10%	■
Requêtes réussies	> 90%	■

Commande : npm run test:perf

Fiche P-04 : Tests de stabilité et résilience

ID	P-04
Fonctionnalité	Stabilité
Priorité	Moyenne
Fichier de test	tests/performance/load-test.js

#	Description	Condition	Résultat attendu	Statut
1	Requêtes répétées	3 requêtes consécutives	100% succès	■
2	Gestion timeout	Timeout 10s	Réponse avant timeout	■
3	Gestion erreurs 404	Route inexistante	Réponse < 2s	■
4	Taille réponse	Page d'accueil	< 5MB	■
5	Assets statiques	Logo.svg	< 500ms	■

Commande : npm run test:perf:quick

RÉCAPITULATIF D'EXÉCUTION

Résumé des tests

Type	Fiches	Cas de test	Passés	Échoués	Taux
Unitaires	U-01 à U-05	42	42	0	100%
Intégration	I-01 à I-03	58	58	0	100%
E2E	E-01 à E-07	32	32	0	100%
Performance	P-01 à P-04	15	15	0	100%
TOTAL	**19 fiches**	**147**	**147**	**0**	**100%**

Commandes d'exécution

- Tous les tests : **npm run test:all**
- Tests unitaires : **npm run test:unit**
- Tests intégration : **npm run test:integration**
- Tests E2E : **npm run test:e2e**
- Tests performance : **npm run test:perf:quick**
- Avec couverture : **npm run test:coverage**
- Rapport E2E : **npm run test:e2e:report**