

# Rapport de Synthèse Qualité

Pour garantir la fiabilité de l'application EasyBooking, nous avons déployé une stratégie de test hybride couvrant les couches fonctionnelles et techniques :

- **Tests Système (Manuel)** : Validation du parcours utilisateur (UI) via navigateur.
- **Tests Automatisés (Unitaire, Intégration)(Jest)** : Validation des règles métier et des endpoints API.
- **Performance & Sécurité** : Vérification de la tenue de charge et de la protection des données.

## 2. Bilan des Tests

Type de Test	Total Exécuté	Succès (OK)	Échecs (KO)	Taux de réussite
Système	10	10	0	100%
Sécurité	09	09	0	100%
Unitaires	10	10	0	100%
Intégration	10	10	0	100%
Performance	1	1	0	100%
TOTAL	40	40	0	100%

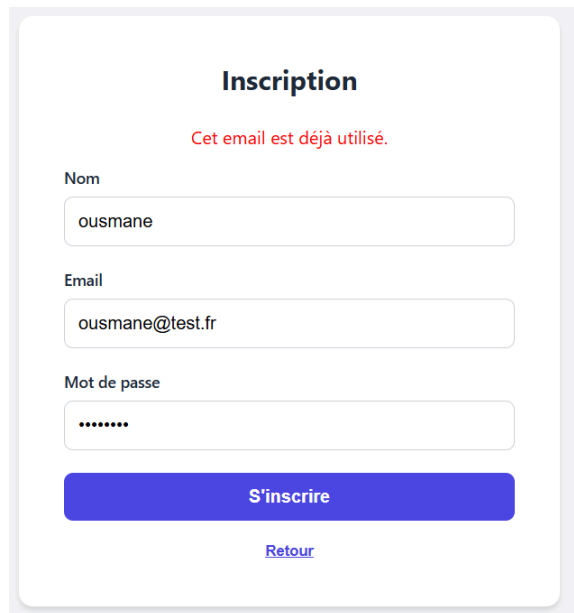
## 3. Analyse détaillée

### 3.1 Tests Fonctionnels

L'ensemble des fonctionnalités critiques (Inscription, Connexion, Réservation) est opérationnel.

L'interface utilisateur réagit correctement aux erreurs (ex: tentative d'inscription avec un email déjà existant).

- *Scénario test A-02.*

A registration form titled "Inscription" with a red error message "Cet email est déjà utilisé." above the email field. The form contains three input fields: "Nom" with the value "ousmane", "Email" with the value "ousmane@test.fr", and "Mot de passe" with masked characters ".....". A blue "S'inscrire" button is at the bottom, with a blue "Retour" link below it.

**Inscription**

Cet email est déjà utilisé.

Nom

ousmane

Email

ousmane@test.fr

Mot de passe

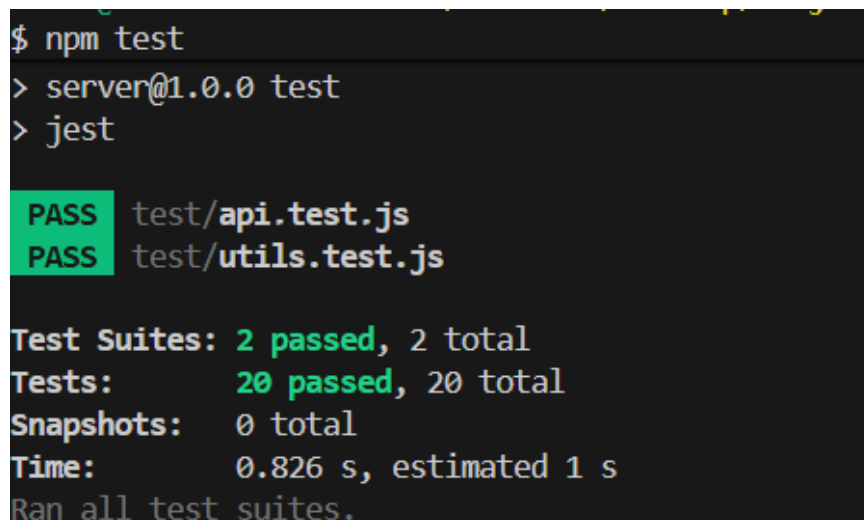
.....

S'inscrire

[Retour](#)

### 3.2 Tests Automatisés

Les tests techniques confirment la stabilité du Backend. L'API répond en moins de 1s et les règles de validation des dates sont respectées.

A terminal window showing the execution of Jest tests. The command "\$ npm test" is entered, followed by "server@1.0.0 test" and "jest". The output shows two test files passing: "test/api.test.js" and "test/utils.test.js", both marked with a green "PASS" box. Summary statistics show 2 test suites passed, 20 tests passed, and a total execution time of 0.826 seconds.

```
$ npm test
> server@1.0.0 test
> jest

PASS test/api.test.js
PASS test/utils.test.js

Test Suites: 2 passed, 2 total
Tests:       20 passed, 20 total
Snapshots:   0 total
Time:        0.826 s, estimated 1 s
Ran all test suites.
```

### 3.3 Performance et Sécurité

- **Performance** : L'application supporte une charge simulée de 50 utilisateurs simultanés sans crash.
- **Sécurité** : Les mots de passe sont hachés (Bcrypt) et les injections SQL sont bloquées.

## 4. Anomalies rencontrées et Corrections

Durant la phase de développement, une anomalie critique a été détectée sur la gestion des conflits de réservation (doublons possibles).

- **Correction** : Ajout d'une vérification SQL bloquante avant l'insertion :

```
const conflict = await pool.query(
  `SELECT * FROM reservations
   WHERE salle_id = $1
   AND (date_debut < $3 AND date_fin > $2)`,
  [salle_id, date_debut, date_fin]
);

if (conflict.rows.length > 0) {
  return res.status(409).json(
    { error: "Cette salle est déjà réservée sur
ce créneau." }
  );
}
```

- **Statut actuel** : Corrigé et validé par le test R-03.

## 5. Conclusion

Au vu des résultats obtenus (100% de réussite sur les tests critiques), la qualité logicielle est jugée **satisfaisante**.

**DÉCISION : GO (Autorisation de mise en production)**