

Année scolaire 2024-2025

RAPPORT DE TESTS DE LOGICIEL

POUR L’APPLICATION DE RESERVATION

DE COMPETTIONS SPORTIVES

**Réalisé par**: **Encadrant:**

N’GATTA KAMENAN ELVIS Prof ANANGAMAN SEDRICK

SOMMAIRE

**Introduction \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_1**

- projet \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 2

- exigences\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 3

- objectifs\_ \_ \_ \_ \_ \_ \_ \_\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 4

**Plan de test** \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_6  
**outils** \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_7

**Détails des tests**\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_9

**Génération et analyse des rapports** \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 11

**Conclusion** \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ 13

**INTRODUCTION**

Ce projet porte sur le développement d'une **application web de réservation de compétitions sportives**, réalisée avec le framework **Flask**. Elle vise à permettre aux membres de clubs de réserver des places pour des compétitions sportives, en fonction de leurs points disponibles, et à fournir un suivi transparent des points de chaque club. L'application est destinée aux clubs sportifs et aux administrateurs, les principaux utilisateurs, avec pour objectif de simplifier la gestion et l'accès aux réservations de compétitions.

Dès le début du développement, plusieurs **problèmes** ont été identifiés, notamment dans le lancement du serveur, la gestion des réservations, et la mise à jour en temps réel des points disponibles. L’identification de ces bugs a mis en lumière la nécessité de tests approfondis pour stabiliser l’application et garantir le bon fonctionnement des fonctionnalités. Parmi les fonctionnalités principales, on compte l’authentification utilisateur, la réservation de places, la consultation des points des clubs, et la sécurisation des sessions avec une option de déconnexion.

1

**EXIGENCES**

Les exigences de l'application couvrent les aspects **fonctionnels** et **non fonctionnels**:

**Exigences fonctionnelles :**

-Authentification sécurisée par email pour accéder à la plateforme.

-Consultation des compétitions disponibles et réservation de places en fonction des points de chaque club.

-Mise à jour des points après chaque réservation, avec un affichage en temps réel.

-Limite de places par club et fonctionnalité de suivi public des points.

-Déconnexion sécurisée pour chaque utilisateur.

**Exigences non fonctionnelles :**

-Performance : rapidité dans la réservation et la mise à jour des points en cas d’affluence.

-Sécurité : protection et sécurité des données des utilisateurs.

-Compatibilité : accessibilité sur différents navigateurs et dispositifs

2

Convivialité : interface intuitive et facile d’accès, permettant aux utilisateurs de naviguer et d’utiliser toutes les fonctionnalités sans difficulté.

3

**OBJECTIF**

Pour notre application de réservation de compétitions, ces tests poursuivent plusieurs objectifs essentiels :

**Assurer la conformité aux spécifications fonctionnelles** :

Vérifier que chaque fonctionnalité de l'authentification à la réservation en passant par la mise à jour des points est conforme aux spécifications initiales du projet.

S'assurer que les limites de réservation et les validations de points sont respectées, afin d’éviter les abus ou erreurs de gestion des réservations.

**Garantir une expérience utilisateur fluide** :

Identifier et corriger les bugs susceptibles d'affecter l'interaction utilisateur, tels que des messages d'erreur inappropriés ou une interface non intuitive.

Optimiser la rapidité et la réactivité de l'application, de manière à ce que les utilisateurs puissent réserver sans latence, même en cas d’afflux de connexions.

**Renforcer la robustesse du système de réservation** :

Tester la gestion des erreurs, comme les tentatives de réservation lorsque les points sont insuffisants ou les places limitées, pour éviter toute défaillance en production.

Vérifier que les utilisateurs ne peuvent pas contourner les

4

règles de réservation (nombre maximum de places par club,

contrôle des points disponibles) grâce à des tests d’intégrité et de sécurité.

**Assurer la stabilité du serveur** :

Effectuer des tests pour s'assurer que le serveur Flask démarre correctement et reste stable sous une charge d’utilisation.

Tester la résilience du serveur en simulant des scénarios d'erreurs et de charge, en veillant à ce que les données soient correctement gérées sans interruption de service.

**Maintenir la sécurité des données et la confidentialité** :

Vérifier que les données des utilisateurs (points, identifiants de clubs) sont correctement protégées et qu'il n'existe aucune fuite d’informations via l'interface.

Tester la déconnexion pour s’assurer qu’aucune session non autorisée ne persiste après qu’un utilisateur a quitté l’application.

5

PLAN DE TEST

**Identification des besoins de test :**

Les fonctionnalités clés de l'application ont été identifiées pour s'assurer de leur bon fonctionnement, notamment l'authentification, la réservation de places, le calcul des points, et la gestion de la capacité des compétitions.

Des cas de test spécifiques ont été créés pour vérifier que chaque composant respecte bien les règles métiers et les attentes des utilisateurs.

**Priorisation des tests :**

Les fonctionnalités critiques, telles que la gestion des réservations et la mise à jour des points, ont été priorisées afin d’éviter tout problème susceptible de perturber l'expérience utilisateur.

Les points de vulnérabilité potentiels, comme la manipulation des points de réservation ou la déconnexion des utilisateurs, ont été traités en priorité pour garantir la robustesse du système.

6

OUTILS

**Pytest:** Nous avions utilisé cet outil flexible pour écrire et exécuter les tests unitaires, fonctionnels et d'intégration.

**pytest-cov:** Intégré pour mesurer la couverture des tests.

**Flake8:** Utilisé pour la vérification de la conformité aux normes de codage, Flake8 identifie les erreurs de style et les lignes de code à optimiser.

**pytest-html:** Cet outil génère des rapports de test en HTML.

POURQUOI ?

**Pytest:** Car il facilite la création de rapports de test détaillés et est adapté pour gérer divers cas de test.

**pytest-cov:** cet outil aide à identifier les parties du code qui nécessitent des tests supplémentaires afin d'assurer une couverture complète des fonctionnalités.

**Flake8:** Pour renforcer la lisibilité et la maintenabilité

7

**pytest-html:** Pour faciliter la communication des progrès de test avec l’équipe de développement et les parties prenantes.

8

DETAILS DES TESTS

**Fichier test\_functional.py**

|  |  |  |
| --- | --- | --- |
| **TEST** | **DESCRIPTION** | **STATUT** |
| test\_show\_summary\_valid\_email | Vérifie l'accès à la page de résumé avec un email valide. | Passé |
| test\_show\_summary\_invalid\_email | Teste la redirection et le message flash en cas d'email invalide. | Passé |
| test\_purchase\_places\_valid | Vérifie qu'une réservation valide est possible. | Passé |
| test\_purchase\_places\_over\_limit | Teste la réservation lorsque le nombre de places dépasse la limite autorisée. .(plus de 12 places ) | Passé |
| test\_purchase\_places\_not\_enough\_points | Vérifie le refus de réservation en cas de points insuffisants. | Passé |
| test\_purchase\_places\_competition\_full | Vérifie le refus de réservation si la compétition est pleine. | Passé |
| test\_purchase\_places\_invalid\_club\_or\_competition | Vérifie la gestion des erreurs pour des entrées invalides de club ou compétition. | Passé |
| test\_overbooking | Teste le refus de réservation en cas de tentative de sur-réservation | Passé |

9

**Fichier test\_integration.py**

|  |  |  |
| --- | --- | --- |
| **TEST** | **DESCRIPTION** | **STATUT** |
| test\_valid\_booking | |  | | --- | | Vérifie qu'une réservation est bien enregistrée. |  |  | | --- | |  | | Passé |
| test\_max\_reserved\_places | Vérifie la limitation du nombre de places par club. | Passé |

**Fichier test\_server.py**

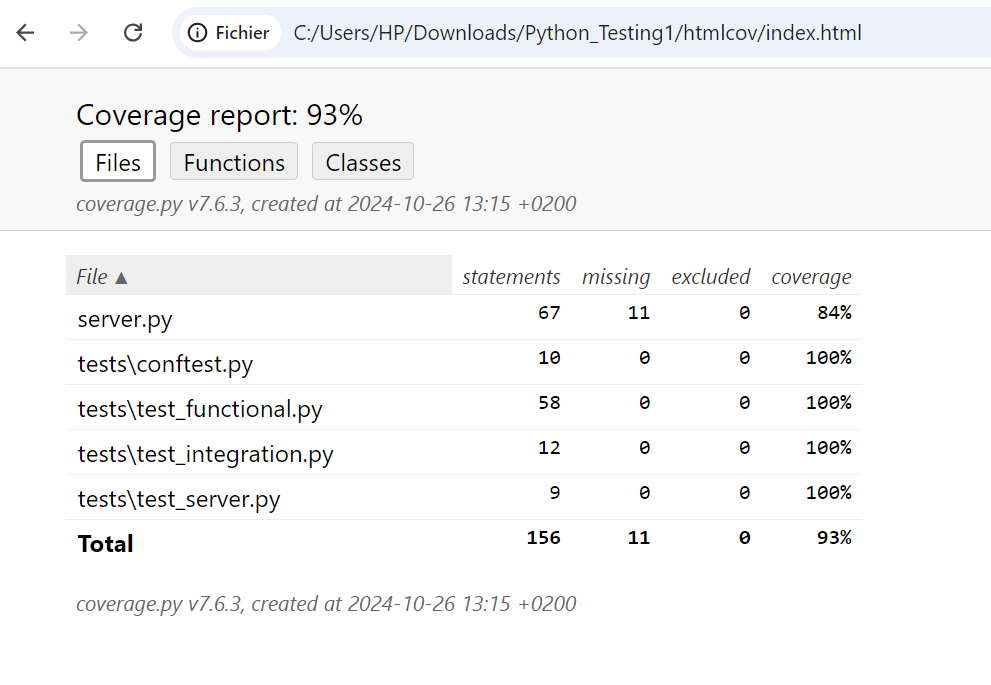
|  |  |  |
| --- | --- | --- |
| **TEST** | **DESCRIPTION** | **STATUT** |
| test\_loadClubs | Vérifie le chargement correct des clubs depuis le fichier JSON. | Passé |
| test\_loadCompetitions | Vérifie le chargement des compétitions depuis le fichier JSON. | Passé |

10

GENERATION ET ANALYSE DES RAPPORTS

**Rapport de Couverture de Code**

Pour la couverture de code nous avions utilisé “coverage”, et le résultat montre une couverture satisfaisante pour les fonctionnalités critiques

****

Le taux de couverture global des tests est de **93%**,

11

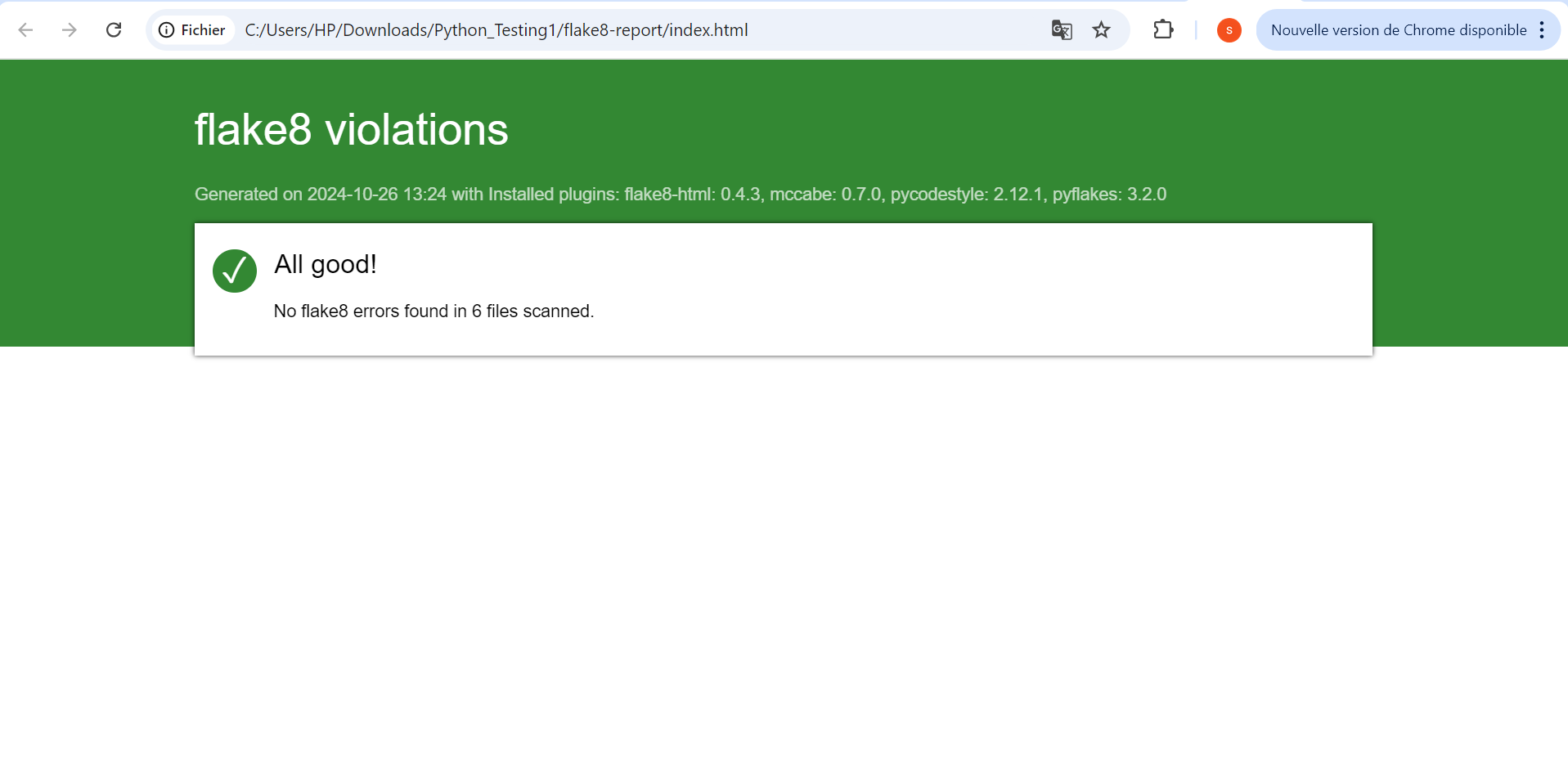
ce qui signifie que la majorité du code est couvert par des

tests. Cela indique que les tests couvrent la plupart des

fonctionnalités essentielles, bien que quelques portions du code ne soient pas encore testées

**Rapport de Flake8**

Ce rapport de flake8 indique que le code respecte toutes les normes de style et de qualité définies par cet outil. Voici une interprétation de l'image:

****

**Pas d'erreurs de style**: Le message "All good!" signifie qu'aucune erreur de style n'a été détectée dans les six fichiers analysés. Cela montre que le code est conforme

12

aux normes de style Python, assurant une meilleure lisibilité et une maintenabilité accrue.

CONCLUSION

Les tests effectués sur l'application ont permis de garantir la qualité et la fiabilité de ses fonctionnalités.

**Récapitulatif des types de tests réalisés et de leur apport**

**Tests unitaires** : Ils ont permis de vérifier le bon fonctionnement des fonctions individuelles, assurant que chaque composant de l'application fonctionne isolément comme prévu.

**Tests d'intégration** : Ils ont validé la cohérence et l’interaction entre les différents modules de l'application, notamment les communications entre le backend (Flask) et les données des compétitions et des clubs.

**Tests fonctionnels** : En simulant des scénarios utilisateur, ces tests ont permis de s’assurer que les fonctionnalités clés, telles que la réservation de places et la gestion des points, répondent bien aux attentes de l'utilisateur.

**Vérification du style de code** : L'utilisation de flake8 a permis d’assurer la conformité du code aux normes de style

13

PEP8, ce qui améliore la lisibilité et la maintenabilité du projet.

**Couverture de code** : Grâce à coverage, nous avons pu obtenir une vue d'ensemble de la couverture des tests, nous permettant d’identifier et de combler les parties du code non testées

15