**Plan de test**

*« Paris sportifs de la Superbowl »*

**Table des Matières**

1. Objectif du Plan de Test (Voir Page 3)
2. Outils de Test (Voir Page 3)
3. Cas de Test (Voir Page 3, 4, 5)
4. Exécution et Suivi (Voir Page 5)
5. Rapport des résultats (Voir Page 6)
6. Conclusions (Voir Page 7)
7. Captures d’écran (Voir Page 8, 9, 10, 11, 12)
8. **Objectif du Plan de Test**

L'objectif de ce plan est de vérifier la fonctionnalité et la robustesse du processus d'authentification dans l’application web développée avec Flask. Cela inclut la capacité de gérer correctement les identifiants de connexion, la redirection vers des pages spécifiques en fonction du rôle de l'utilisateur, et la gestion adéquate des erreurs.

1. **Outils de Test**

* **Selenium WebDriver** :

Utilisé pour simuler l'interaction de l'utilisateur avec l'interface de l'application web. Cela inclut la navigation, l'entrée de données dans des formulaires et les clics sur des boutons.

* **Tests Unittest** :

Pour des tests unitaires qui invoquent directement les routes du serveur web et évaluent les réponses sans nécessiter un navigateur.

1. **Cas de Test**

***Tests d'Intégration avec Selenium***

**Test de Connexion Réussie** **Utilisateur :**

**Description** : Vérifie qu'un utilisateur avec des identifiants valides peut se connecter et est redirigé vers son espace personnel.

**Étapes** :

* Naviguer vers la page de connexion.
* Entrer un email et un mot de passe valides.
* Cliquer sur le bouton de ‘Se connecter’.
* Attendre le chargement de la page de l'utilisateur et vérifier la présence d'éléments spécifiques.

**Résultat attendu** : L'utilisateur est redirigé vers son espace personnel et le contenu approprié est affiché.

**Test d'Échec de Connexion** :

**Description** : Évalue le comportement de l'application lors de la saisie d'identifiants invalides.

**Étapes** :

* Répéter les étapes du cas précédent en utilisant un email ou un mot de passe incorrect.

**Résultat attendu** : Le système reste sur la page de connexion et affiche un message d'erreur.

**Test de Connexion Réussie** **Administrateur :**

**Description** : Vérifie qu'un administrateur avec des identifiants valides peut se connecter et est redirigé vers son espace administrateur.

**Étapes** :

* Naviguer vers la page de connexion.
* Entrer un email et un mot de passe valides.
* Cliquer sur le bouton de ‘Se connecter’.
* Attendre le chargement de la page de l'administrateur et vérifier la présence d'éléments spécifiques.

**Résultat attendu** : L'administrateur est redirigé vers son espace administrateur et le contenu approprié est affiché.

**Test d'Échec de Connexion** :

**Description** : Évalue le comportement de l'application lors de la saisie d'identifiants invalides.

**Étapes** :

* Répéter les étapes du cas précédent en utilisant un email ou un mot de passe incorrect.

**Résultat attendu** : Le système reste sur la page de connexion et affiche un message d'erreur.

***Tests Unitaires***

**Connexion Réussie (Utilisateur et Administrateur)** :

**Description** : Vérifie que les utilisateurs et les administrateurs sont correctement redirigés après une connexion réussie.

**Résultat attendu** : Codes de statut 302 et redirection vers la page correspondante.

**Utilisateur Inexistant** :

**Description** : Vérifie la réponse lorsque l'on tente de se connecter avec un email non enregistré.

**Résultat attendu** : Code de statut 200 et message d'erreur spécifique visible dans la réponse.

**Utilisateur Non Confirmé** :

**Description** : Évalue la réponse lorsqu’un utilisateur non confirmé tente de se connecter.

**Résultat attendu** : Code de statut 200 et message indiquant qu'il doit confirmer son compte.

**Création de Session à la Connexion** :

**Description** : Assure qu'une session utilisateur est correctement créée à la connexion réussie.

**Résultat attendu** : La session de Flask doit contenir l'id utilisateur.

**Connexion avec Données Vides** :

**Description** : Teste la réponse du système lors de la tentative de connexion sans fournir d'email ni de mot de passe.

**Résultat attendu** : Code de statut 200 et message d'erreur pertinent.

***Tests de Charge***

**Visualisation des Matchs :**

**Description** : Vérifie le comportement de l'application lorsque de multiples utilisateurs accèdent simultanément à la page de visualisation des matchs. Cela teste la capacité du serveur à gérer une charge élevée de trafic sur cette route spécifique.

**Résultat attendu** : Les serveurs doivent répondre avec succès (codes de statut HTTP 200) à toutes les demandes sans erreurs significatives. Les temps de réponse doivent rester en dessous d'un seuil prédéfini même sous charge maximale, indiquant que l'expérience utilisateur n'est pas compromise. Le serveur doit être capable de gérer un nombre spécifié de requêtes par seconde (RPS) sans dégradation des performances ou erreurs accrues.

1. **Exécution et Suivi**

Exécuter les tests dans un environnement contrôlé, en s'assurant que toutes les dépendances sont correctement configurées. Surveiller les résultats et recueillir les logs pour analyser les éventuels échecs ou comportements inattendus.

***Test Unittest* :**

Pour exécuter un test unitaire en utilisant le framework Unittest de Python :

* Tests dans un fichier Python.
* Ouvrez terminal.
* Exécutez les tests en utilisant la commande suivante : python -m unittest test\_login.py

**Test Pytest :**

Pour exécuter des tests en utilisant Pytest :

* Tests dans un fichier Python.
* Ouvrez terminal.
* Exécutez les tests en utilisant la commande suivante : pytest
* Pytest découvrira et exécutera automatiquement tous les fichiers et fonctions de test.

**Test Locust :**

Pour exécuter un test de charge avec Locust :

* Tests dans un fichier Python.
* Ouvrez terminal.
* Exécutez Locust en utilisant la commande suivante : locust -f chargeTest.py
* Cela démarrera l'interface web de Locust. Ouvrez un navigateur et allez à http://localhost:8089 pour configurer et démarrer votre test de charge.

1. **Rapport de Résultats**

***Rapport de Tests pour le Processus d'Authentification***

Date de réalisation des tests : 17/04/2024

**1. Introduction**

Ce rapport présente les résultats des tests d'intégration avec Selenium et des tests unitaires avec Flask pour le processus d'authentification de l'application web développée avec Flask.

**2. Environnement de Test**

Navigateur : Chrome

Serveur web : http://127.0.0.1:5000/

Base de données : bdsuperbowlTest

Système d'exploitation : macOs 14.4 (23E214)

**3. Résumé des Tests**

Tous les cas de test ont été exécutés avec succès. Les fonctionnalités de connexion pour les utilisateurs et les administrateurs ont été validées, y compris la gestion des erreurs pour les identifiants incorrects et les comptes non confirmés.

**4. Détails des Tests**

**Tests d'Intégration avec Selenium :**

Test de Connexion Réussie (Utilisateur) : Succès

Test de Connexion Réussie (Administrateur) : Succès

Test d'Échec de Connexion (Utilisateur) : Succès

**Tests Unitaires avec Unittest**

Connexion Réussie (Utilisateur et Administrateur) : Succès

Utilisateur Inexistant : Succès

Utilisateur Non Confirmé : Succès

Création de Session à la Connexion : Succès

Connexion avec Données Vides : Succès

**Tests de Charge**

Visualisation des Matchs : Succès

1. **Conclusions**

Les résultats indiquent que le processus d'authentification est robuste et fonctionne comme attendu. La gestion des erreurs est effective et les sessions sont correctement établies. Il n'y a pas eu de défaillances durant le processus de test.

1. **Captures d’écran**

*Test integration selenium :*

Test de Connexion Réussie (Utilisateur)

Une image contenant texte, capture d’écran, conception

Description générée automatiquement

Résultat :

Une image contenant texte, capture d’écran, Police, noir

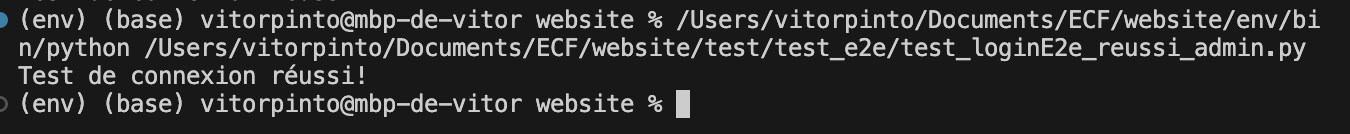
Description générée automatiquement

Test de Connexion Réussie (Administrateur) : Succès

Une image contenant texte, capture d’écran, logiciel, conception

Description générée automatiquement

Résultat :

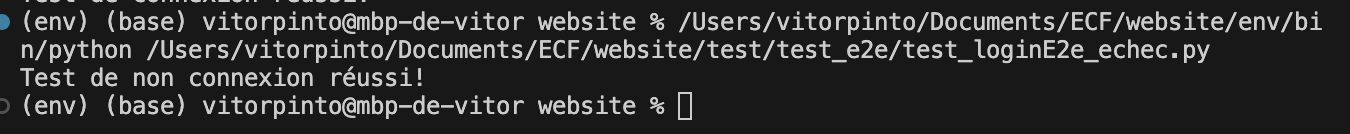


Test d'Échec de Connexion (Utilisateur) :

Une image contenant texte, capture d’écran

Description générée automatiquement

Résultat :



*Tests Unitaires Unittest*

Une image contenant texte, capture d’écran, menu

Description générée automatiquement

Résultat :

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

*Tests de Charge*

*Une image contenant texte, carte de visite, capture d’écran

Description générée automatiquement*

Résultat :

*Une image contenant texte, ligne, Police, nombre

Description générée automatiquement*