Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
****
Université de Sousse
****
École Supérieure des Sciences et de la Technologie de Hammam Sousse

Plan de Validation des Tests (PVT)

PRO-PFE : Préparation et Réalisation pour l'Orientation vers le PFE

Licence Informatique - LI2

N° de groupe : 20

Titre du projet : Plateforme de location et partage de véhicules

Réalisé par :

– Ahmed Miled
– Rayen Monceur
– Wajdi Ben Aicha

Année universitaire 2024–2025

Table des matières

Résumé Exécutif

Ce document présente le Plan de Validation des Tests (PVT) du **projet PRO-PFE n° 09** intitulé *" Plateforme de location et partage de véhicules"*.

Description du projet :
Ce projet consiste en la réalisation d'un site web de location de voitures destiné à mettre en relation des clients avec des agences de location.
L'objectif principal est de permettre aux utilisateurs de consulter les véhicules disponibles, d'effectuer des réservations en ligne, et pour les propriétaires d'agences, de gérer leurs offres via une interface dédiée. Le site intègre un système d'authentification différencié pour les clients et les propriétaires d'agences, une page d'accueil présentant les véhicules disponibles sous forme de cartes interactives avec des modales pour les détails, ainsi qu'un tableau de bord spécifique à chaque type d'utilisateur après connexion.
Une interface d'administration permet également la gestion des comptes des propriétaires d'agences.
L'intérêt pédagogique de ce projet réside dans la mise en œuvre complète d'une application web dynamique, impliquant la gestion des rôles utilisateurs, l'interaction entre le frontend et le backend, et la conception d'une base de données adaptée. Il permet d'aborder diverses technologies web telles que HTML, CSS, JavaScript, PHP et MySQL.

Ce PVT s'appuie sur les bonnes pratiques définies par l'ISTQB afin de garantir la qualité, la fiabilité et la sécurité du produit développé. Il définit les objectifs, la portée, la stratégie de test ainsi que les types de tests à réaliser.

Chaque type de test est accompagné de jeux d'essai concrets et de scénarios détaillés, avec un système de priorisation des défaillances pour faciliter la détection, la gestion et la correction des erreurs durant le cycle de développement.

1 Objectifs du Test

— Vérifier la conformité de l'application avec les exigences fonctionnelles et non-fonctionnelles.
— Détecter les défauts majeurs avant la mise en production.
— Assurer la robustesse, la performance et la sécurité de l'application.

2 Portée des Tests

2.1 Fonctionnalités Incluses

— Authentification (connexion/inscription)
— Protection contre les attaques SQL
— Performance sous forte sollicitation

2.2 Fonctionnalités Exclues

— Intégration avec API externes (non prioritaire)
— Module de statistiques avancées

3 Planning de Validation

Phase	**Activités**	**Période prévue**
Planification	Définition de la stratégie, choix des outils	14/04/2025
Conception des tests	Rédaction des cas de test, jeux d'essai	15/04/2025-21/04/2025
Exécution des tests	Lancement des scénarios de test	22/04/2025-24/04/2025
Correction	Analyse des résultats, correction des anomalies	25/04/2025-26/04/2025
Validation finale	Rapport de test et preuves de conformité	27/04/2025

4 Équipe QA

— Chef de projet QA : Wajdi BEN Aicha
— Testeurs : Ahmed Miled, Rayen Monceur , Wajdi BEN Aicha
— Rôle encadrant : validation du protocole de test

5 Outils de Test

— **Manuel** : Tests de sécurité (injection SQL)
— **PHPUnit** : Tests de performance et surcharge , Tests d'authentification

6 Stratégie et Types de Tests

6.1 Tests d'authentification

Objectif : Vérifier que chaque fonctionnalité du système se comporte conformément aux spécifications fonctionnelles définies.

Scénario 1 – Connexion client avec identifiants valides

Précondition : Un compte client existe déja dans la base de données avec un email et mots de passe predefinies.
Étapes :

1. Accéder à la page de connexion.
2. Saisir testExiste@gmail.com et 123.
3. Cliquer sur "Se connecter".

Résultat attendu : Le client est authentifié avec succès et redirigé vers son tableau de bord.
Priorité : **1 – Critique**

Scénario 2 – Connexion client avec identifiants invalides

Précondition : Aucun compte existant avec l'email saisie.
Étapes :

1. Accéder à la page de connexion.
2. Saisir des informations quelconque.
3. Cliquer sur "Se connecter".

Résultat attendu : Le système affiche un message d'erreur "Email ou mot de passe incorrect".
Priorité : **2 – Critique**

6.2 Tests de Sécurité

Objectif : Vérifier la résistance de l'application aux attaques connues comme les injections SQL.

Scénario 3 – Injection SQL sur le champ "email"

Précondition : Absence de protection par requêtes préparées.
Étapes :

1. Saisir `' OR '1'='1` dans le champ email.
2. Entrer un mot de passe quelconque.
3. Cliquer sur "Se connecter".

Résultat attendu : La tentative est bloquée, email incorrect. Veuillez essayer une autre fois.
Priorité : **1 – Fatal**

Scénario 4 – Injection SQL sur le champ "mot de passe"

Précondition : Absence de protection par requêtes préparées.
Étapes :

1. Saisir `' OR '1'='1` dans le champ mot de passe.
2. Entrer un email valide.
3. Cliquer sur "Se connecter".

Résultat attendu : La tentative est bloquée, Mots de passe. Veuillez essayer une autre fois.
Priorité : **1 – Fatal**

6.3 Tests de Performance / Surcharge

Objectif : Vérifier le comportement de l'application sous forte charge et évaluer ses limites en termes de performance.

Scénario 5 – 500 réservations en même temps

Précondition : Serveur déployé avec 1 Go de RAM.
Étapes :

1. Utiliser JMeter pour simuler 500 utilisateurs faisaient une requête en même temps.
2. Observer la latence et le taux d'erreur.

Résultat attendu : Aucun crash, temps de réponse inférieur à 1 secondes.
Priorité : **2 – Élevée**

7 Jeux d'Essai

Jeu 1 : Connexion Utilisateur

— **Email** : test1@example.com
— **Mot de passe** : 123
— **Résultat attendu** : Connexion réussie

Jeu 2 : SQL Injection

— **Email** : `test1@gmail.com`
— **Mot de passe** : `' OR '1'='1`
— **Résultat attendu** : Rejet de la requête

Jeu 3 : Test de surcharge

— **Requête** : Exécuter 500 fois une requête SQL (SELECT * FROM reservations) en boucle rapide dans un même test (simulation d'accès massif).
— **Résultat attendu** : Temps total d'exécution du test inférieur à 1 seconde.

8 Critères de Validation et Attentes des Tests

REMARQUE : À quoi sert cette section ?

Cette section a pour but de définir clairement les conditions dans lesquelles un test peut être considéré comme réussi ou échoué.

8.1 Quand valider un test ?

- **Conditions préalables** :
  - La base de données contient des jeux d'essai réalistes :
    - Véhicules avec différentes catégories (économique, SUV, luxe).
    - Réservations actives/annulées pour tester les conflits de dates.
  - Les comptes utilisateurs nécessaires sont créés.
  - Serveur configuré avec PHP 8.x, MySQL, et certificat SSL actif.
- **Moment du test** :
  - Après chaque ajout au module de réservation.
  - Avant la démonstration finale.
  - Après chaque correction de bug détecté ou signalé.
- **État du code** : Le code de la fonctionnalité à tester est terminé, intégré, et ne génère pas d'erreurs à l'exécution.

8.2 Que doit-on attendre de ce test ?

Tests Fonctionnels

- *Résultat attendu* : Le test doit permettre de confirmer que le comportement observé est conforme aux spécifications.
- *Détection d'anomalies* : Tout écart avec le comportement attendu est noté comme une anomalie et doit être corrigé.
- *Performance et stabilité* : Le système doit répondre dans un temps raisonnable, même sous charge.

Tests de Sécurité

- *Résultat attendu* : Les tentatives d'attaques connues (comme les injections SQL) doivent être bloquées efficacement.
- *Détection d'anomalies* : Toute faille ou contournement des sécurités attendues est considéré comme une anomalie critique.
- *Performance et stabilité* : L'application doit rester stable et fonctionnelle même face à des entrées malveillantes.

Tests de Performance / Robustesse

— *Résultat attendu* : L'application doit rester réactive et stable, même sous forte charge.

— *Détection d'anomalies* : Des ralentissements, plantages ou erreurs anormaux indiquent des faiblesses de performance.

— *Performance et stabilité* : Le temps de réponse doit rester acceptable (inférieur à 1 seconde), sans crash.