Plan de test
pour une application web

Ue Qualité Logiciel

Master 2 Informatique
parcours Ingénierie du développement logiciel

Niveau de confidentialité du document : privé

Date	Version	Description	Auteur	Destinataire	Date de diffusion	Date de validation	Validateur
21/12/2022	V1.0	Une application web dédiée aux réservations d'un restaurant.	Katia HALLAI	Magalie CONTENSIN	01/01/2023	01/01/2023	Magalie CONTENSIN

Sommaire

I.Introduction

A.Objectifs du document

- trouver un maximum d'anomalies,
- evaluer les risques,
- décrire les stratégies de tests et les tests à réaliser,
- identifier les livrable,
- valider l'application par rapport aux spécifications,
- identifier les ressources nécessaires.

B.Présentation du logiciel

Une application web dédiée aux réservations de tables dans un restaurant. Tout d'abord, la réservation se fait via un formulaire (sans authentification). Ensuite, un mail de confirmation sera envoyé au client. Pour confirmer, le client devra juste cliquer sur le lien contenu dans le mail. Au bout d'une heure, si aucune confirmation n'a été faite, la réservation sera annulée.
Le propriétaire du restaurant doit s'authentifier pour accéder à la liste des réservations et pour pouvoir l'exporter au format CSV.

C.Périmètre

Autour du formulaire de réservation : Les tests doivent détecter si des données sont manquantes ou erronées.
Autour du mail de confirmation : Les tests doivent détecter le bon envoi du mail, sa confirmation/ son annulation.
Autour de l'accès à la liste des réservation : Les tests doivent confirmer l'exclusivité d'accès pour le propriétaire et la possibilité d'exportation au formation CSV de la liste.

D.Contraintes

Les tests de performances doivent être effectués sur des ordinateurs dédiés.
Respect des deadlines.
Couvertures de l'ensemble des les lignes de code et des branches.
De préférence, le développeur ne soit pas lui-même le testeur ni le concepteur.

E.Stratégie de tests

Types de tests réalisés : tests unitaires

Objectif du test :
Vérifier que chaque unité du projet fonctionne comme prévu.
Technique :
Le testeur fournit un jeu de données en entrée, il vérifie que le résultat obtenu est celui attendu. Il conçoit les tests en s'appuyant sur les informations à sa disposition concernant le fonctionnement du programme.
La technique utilisée est celle des partitions d'équivalence qui peut être combinée avec les tests aux limites afin de tester les limites de chaque partition, pour de ne pas rater des erreurs dans les conditions.
Critère de complétion :
Tous les tests prévus ont été effectués et les résultats obtenus sont ceux attendus.
Commentaire :
Le taux de couverture du code doit être optimal.

Outils utilisés
Framework de tests : JUnit 5
Couverture des tests : JaCoCo (Java Code Coverage)

Sites de test et bases de test
Prévoir une ou plusieurs URL du site de test.
Prévoir un accès à la ou les bases de données de test.

Indicateurs utilisés
Taux de couverture des tests.

Comment les tests seront implémentés et exécutés, ordre d'exécution ?
Implanter des tests simples et des tests paramétrés. Utiliser des tests positifs et des tests négatifs.
L'ordre d'exécution des tests est secondaire.

Types de tests réalisés : tests de performance (montée de charge)

Objectif du test :
Vérifier les temps de réponse de l'application avec l'augmentation de la charge de travail.
Technique :
Augmenter progressivement le nombre d'utilisateurs.
Critère de complétion :
Tous les tests ont été effectués, les réponses ont été obtenues dans un délai acceptable.

Commentaire :
Les tests doivent être réalisés sur un ordinateur dédié

Outils utilisés
Logiciel de tests : WebLOAD
Ordinateur dédié.
Fichiers EXCEL pour le recueil des résultats.

Sites de test et bases de test
Prévoir une ou plusieurs URL du site de test.
Prévoir un accès à la ou les bases de données de test.

Indicateurs utilisés
Débit, temps de réponse et utilisations des ressources.

Comment les tests seront implémentés et exécutés, ordre d'exécution ?
Les tests sont implémentés avec une charge de plus en plus élevée pour un suivi efficace des performances. Augmenter progressivement le nombre d'utilisateurs, surveiller l'essai et recueillir les données résultantes afin que vos développeurs puissent les utiliser pour améliorer les performances du logiciel.

Types de tests réalisés : tests fonctionnels

Objectif du test :
Vérifier si l'application est conforme aux spécifications.
Technique :
Naviguer sur l'application, détecter et reporter les anomalies ou la non-conformité dans les fonctionnalités.
Critère de complétion :
Tous les tests prévus ont été exécutés, toutes les anomalies identifiées ont été enregistrées.
Tous les scénarios de tests ont été effectués, et les résultats sont reportés.
La solution développée correspond au besoin émis par le client dans les spécifications.
Commentaire :
Prévoir des cas de test (scenario).
Les tests doivent être réalisés sur l'application.

Outils utilisés
L'application en question.
Fichiers EXCEL pour le recueil des résultats.

Indicateurs utilisés
La conformité des fonctionnalités aux spécifications.
L'existence d'anomalies.

F.Hypothèses

Anomalies persistantes qui peuvent rallonger la date de livraison.
Coût supplémentaire pour des ressources humaines supplémentaires.

II.Ressources nécessaires

1.Tableau des Ressources Humaines nécessaires à l'exécution du plan de test

Fonction	N°	Responsabilité	Compétences
Responsable des tests	1	encadrement	
Concepteur de test	X	création du plan de tests + identifier, prioriser les cas de tests et leur ordre d'exécution.	outils de test, definition des caracteristiques de la qualité logiciels et le rôle des tests, maîtrise des différentes techniques de tests. définition et mise en oeuvre des scénarios de test, méthodes et normes qualité.
Développeur de test	1	écrire le code des tests	Java, JUnit
Testeur	X	exécuter les tests, stocker les résultats, documenter les anomalies, évaluer.	techniques, outils de reporting(Excel), pensée critique, résolution de problème, persuasion et négociation.
Valideur	1	évaluer la couverture du code et des cas de tests, analyser les anomalies, décider si les tests sont complétés et si succès.	JaCoCo Méthodologie de test approfondi.

2.Liste des ressources système nécessaires

Serveur de bases de données : nom du serveur, IP, nom de la base de données
Poste du testeur : exigences de configuration, logiciels, Excel
Poste du développeur de tests : exigences de configuration, Java, JUnit

III.Jalons et livrables

A.Jalons du projet de tests

Tâches	effort h/mois	Début	Fin
planification des tests	35	21/12/2022	27/12/2022
conception	21	28/22/2022	30/12/2022
implementation	35	02/01/2023	06/01/2023
exécution	14	06/01/2023	09/01/2023
validation	21	09/01/2023	11/01/2023

B.Livrables

Livrable	Date de livraison	Destinatinataire	Auteur
journaux de tests	30/12/2022	développeur de test	concepteur de test
rapports d'anomalies	09/01/2023	valideur	testeur
code source des tests	06/01/2023	testeur	développeur de test
rapport de couverture des tests	11/01/2023	responsable des tests	valideur
rapport de performance	11/01/2023	responsable des tests	valideur
rapport de synthèse	11/01/2023	responsable des tests	valideur

IV.Tests unitaires

Cas de test 1 : Réserver une table libre
- Identifiant unique : RESA-1
- Nom de l'unité à tester :
- Auteur : Testeur
- Description :
  Objectif du test : Vérifier qu'une table libre peut être réservée (test positif).
  Caractéristiques de l'environnement de test : JUnit 5 et des tests paramétrés.
- Contraintes liées à ce test :
  Système : sans installation, JUnit 5
  Humaine : Aucune intervention humaine.
  Document qui précise les contraintes : spécification et plan de test.
- Hypothèses et pré-conditions : La table doit être libre, aucune données manquantes ou erronées dans la saisie du formulaire.
- Données du test : client.name="Dupont", table =04
- Résultat attendu : La réservation de la table 04 par Dupont a été effectuée avec succès.
- Critère de validation : succès

Cas de test 2 : Réserver une table prise
- Identifiant unique : RESA-2
- Nom de l'unité à tester : ReservationTable
- Auteur : Testeur
- Description :
  Objectif du test : Vérifier qu'une table prise ne peut être réservée une autre fois(test négatif).
  Caractéristiques de l'environnement de test : JUnit 5.
- Contraintes liées à ce test :
  Système : sans installation, JUnit 5
  Humaine : Aucune intervention humaine.
  Document qui précise les contraintes : spécification et plan de test.
- Hypothèses et pré-conditions : La table doit être prise, aucune données manquantes ou erronées dans la saisie du formulaire.
- Données du test : client.name="Dupont", table =04
- Résultat attendu : Cette table est déjà réservée.
- Critère de validation : échec

V.Tests fonctionnels

Cas de test 1 : Réserver une table libre
- Identifiant unique : RESA-1
- Nom de l'unité à tester :
- Auteur : Testeur
- Description :
  Objectif du test : Vérifier qu'une table libre peut être réservée (test positif).
  Caractéristiques de l'environnement de test : JUnit 5 et des tests paramétrés.
- Contraintes liées à ce test :
  Système : sans installation, JUnit 5
  Humaine : Aucune intervention humaine.
  Document qui précise les contraintes : spécification et plan de test.
- Hypothèses et pré-conditions : La table doit être libre, aucune données manquantes ou erronées dans la saisie du formulaire.
- Données du test : client.name="Dupont", table =04
- Résultat attendu : La réservation de la table 04 par Dupont a été effectuée avec succès.
- Critère de validation : succès

Scénario test 1:
Le client arrive sur la plateforme, appuie sur réserver une table. Un tableau des tables lui est proposé avec les dates. En cliquant sur l'une des tables, un formulaire lui est proposé. Le client doit saisir correctement les informations demandées dans le formulaire puis envoyer la demande de réservation en cliquant sur valider. Un mail de confirmation lui sera envoyé. Le client doit appuyer sur le lien que comporte le mail pour confirmer la réservation. La table a été réservée avec succès.

Cas de test 2 : Réserver une table prise
- Identifiant unique : RESA-2
- Nom de l'unité à tester : ReservationTable
- Auteur : Testeur
- Description :
  Objectif du test : Vérifier qu'une table prise ne peut être réservée une autre fois(test négatif).
  Caractéristiques de l'environnement de test : JUnit 5.
- Contraintes liées à ce test :
  Système : sans installation, JUnit 5
  Humaine : Aucune intervention humaine.
  Document qui précise les contraintes : spécification et plan de test.
- Hypothèses et pré-conditions : La table doit être prise, aucune données manquantes ou erronées dans la saisie du formulaire.
- Données du test : client.name="Dupont", table =04
- Résultat attendu : Cette table est déjà réservée.
- Critère de validation : échec

Scénario test 2:
Le client arrive sur la plateforme, appuie sur réserver une table. Un tableau des tables lui est proposé avec les dates. En cliquant sur l'une des tables,un message lui est affiché disant que la table est déjà réservée.

Scénario test 3:
Le client arrive sur la plateforme et souhaite visualiser la liste des réservations. Il appuie sur "voir la liste des réservations". Un formulaire d'authentification lui est proposé.

Références

Cours de qualité logiciel sur les tests.
Document de spécification.