Dossier d'exercices : Mise en place d'une campagne de test

Basé sur la certification ISTQB Fondation

---

1. Pourquoi les tests sont-ils nécessaires ?

Exercice 1.1 : Analyse de cas réels

Objectif : Comprendre l'impact des défauts logiciels en analysant des exemples concrets.

Consigne détaillée :
Cherchez des exemples concrets d'accidents ou d'échecs causés par des défauts logiciels et précisez les éléments suivants :

- **La défaillance** : Que s'est-il passé ?
- **La cause de la défaillance** : Quel dysfonctionnement technique a provoqué la défaillance ?
- **La cause du défaut** : Quelle erreur humaine ou organisationnelle est à l'origine ?
- **Les conséquences** : Quels ont été les impacts (financiers, réputation, etc.) ?

Exercice 1.2 : Réflexion sur les risques

Objectif : Identifier les risques liés à l'absence de tests pour un logiciel critique.

Consigne détaillée :
Pour un **logiciel de gestion bancaire**, listez **5 risques majeurs** liés à l'absence de tests. Pour chaque risque, précisez :

- **Description** : En quoi consiste le risque ?
- **Impact sur les résultats financiers** : Pertes, amendes, etc.
- **Impact sur la réputation** : Perte de confiance des clients, médiatisation négative.
- **Impact sur la conformité réglementaire** : Non-respect des lois (ex : RGPD, LSF).

---

2. Que sont les tests ?

Exercice 2.1 : Étude de cas – Identifier les types de tests

Objectif : Appliquer les concepts de tests statiques, dynamiques, et d'acceptation à un scénario concret.

Consigne détaillée :
Vous travaillez sur un projet de développement d'une **application mobile de gestion de tâches**. Voici une liste d'activités liées aux tests. Pour chacune, précisez :

- **Type de test** : Statique, Dynamique, ou Acceptation.
- **Justification** : Pourquoi ce type de test est-il adapté à cette activité ?
- **Responsable** : Qui devrait réaliser ce test (développeur, testeur, utilisateur final) ?

Liste des activités :

1. **Revue des spécifications fonctionnelles** pour vérifier leur clarté et leur exhaustivité.
2. **Exécution de tests automatisés** pour valider le bon fonctionnement de la fonctionnalité "ajout d'une tâche".
3. **Validation par les utilisateurs finaux** que l'application répond à leurs besoins.
4. **Analyse du code source** pour détecter des erreurs de syntaxe ou des violations des bonnes pratiques.
5. **Test manuel** de l'interface utilisateur pour vérifier l'ergonomie et la facilité d'utilisation.

Exercice 2.2 : Distinguer erreurs, défauts et défaillances

Objectif : Appliquer les définitions théoriques d'erreurs, défauts et défaillances à des scénarios concrets.

Consigne détaillée :
Pour chaque scénario décrit ci-dessous, identifiez s'il s'agit d'une **erreur**, d'un **défaut** ou d'une **défaillance**. Justifiez votre choix en expliquant pourquoi ce scénario correspond à cette catégorie.

Scénario 1 : Oubli de gestion d'erreur

Contexte : Lors du développement d'une application de gestion de tâches, un développeur oublie d'inclure une gestion d'erreur pour le cas où un utilisateur essaie de supprimer une tâche qui n'existe pas.

- **Question** : S'agit-il d'une erreur, d'un défaut ou d'une défaillance ?

Scénario 2 : Plantage de l'application

Contexte : Lors de l'utilisation d'une application bancaire, un utilisateur saisit un caractère spécial dans le champ "Montant du virement". L'application plante immédiatement et affiche un message d'erreur système.

- **Question** : S'agit-il d'une erreur, d'un défaut ou d'une défaillance ?

Scénario 3 : Spécification incomplète

Contexte : Dans le cahier des charges d'une application de réservation de billets de train, il est mentionné que les utilisateurs doivent pouvoir rechercher des trajets, mais le format des données d'entrée (ex : date, heure) n'est pas précisé.

- **Question** : S'agit-il d'une erreur, d'un défaut ou d'une défaillance ?

Scénario 4 : Bug de calcul

Contexte : Dans une application de gestion de notes pour les étudiants, un bug est découvert : la moyenne des notes est calculée en ignorant les coefficients des matières.

- **Question** : S'agit-il d'une erreur, d'un défaut ou d'une défaillance ?

Scénario 5 : Problème d'interface utilisateur

Contexte : Lors de l'utilisation d'une application mobile, les utilisateurs remarquent que le bouton "Valider" ne répond pas quand ils cliquent dessus, empêchant la validation d'une commande.

- **Question** : S'agit-il d'une erreur, d'un défaut ou d'une défaillance ?

Scénario 6 : Mauvaise compréhension des exigences

Contexte : Lors de la conception d'une application de gestion de projets, l'équipe de développement interprète mal une exigence concernant la priorisation des tâches, ce qui conduit à une implémentation incorrecte.

- **Question** : S'agit-il d'une erreur, d'un défaut ou d'une défaillance ?

Scénario 7 : Fuite mémoire

Contexte : Une application de streaming vidéo commence à ralentir après plusieurs heures d'utilisation, car elle ne libère pas correctement la mémoire utilisée pour afficher les vidéos.

- **Question** : S'agit-il d'une erreur, d'un défaut ou d'une défaillance ?

Scénario 8 : Message d'erreur incompréhensible

Contexte : Dans une application de gestion de stocks, quand un utilisateur essaie d'ajouter un produit déjà existant, un message d'erreur technique et incompréhensible s'affiche ("Erreur 404XZ").

- **Question** : S'agit-il d'une erreur, d'un défaut ou d'une défaillance ?

---

3. Les 7 principes généraux des tests

Exercice 3.1 : Application des principes ISTQB

Objectif : Illustrer chaque principe ISTQB par un exemple concret.

Consigne détaillée :
Pour chaque principe, donnez un exemple **réel ou inventé** et expliquez en quoi il illustre le principe.

---

4. Processus de test et niveaux de test

Exercice 4.1 : Planification des niveaux de test

Objectif : Appliquer les niveaux de test à un projet fictif.

Consigne détaillée :
Pour un projet de développement d'une **application mobile de réservation de billets de train** :

1. **Définissez les objectifs** pour chaque niveau de test :
    - **Test de composants** : Vérifier que chaque fonction (ex : recherche de trajets) fonctionne isolément.
    - **Test d'intégration** : Vérifier l'interaction entre les modules (ex : paiement + réservation).

    - **Test système** : Tester l'application complète dans un environnement proche de la production.
    - **Test d'acceptation** : Valider que l'application répond aux besoins des utilisateurs finaux.
2. **Identifiez les bases de test** :
    - Exigences fonctionnelles, User Stories, maquettes, etc.
3. **Précisez les responsables** :
    - Développeurs (tests unitaires), testeurs (intégration/système), utilisateurs finaux (acceptation).

Exercice 4.2 : Conception de cas de test

Objectif : Rédiger des cas de test pour une fonctionnalité simple.

Consigne détaillée :
Pour la fonctionnalité **"Connexion utilisateur"** d'une application web :

1. **Listez 3 conditions de test** (ex : mot de passe oublié, identifiants incorrects).
2. Pour chaque condition, rédigez un **cas de test** avec :
    - **Pré-conditions** : État du système avant le test (ex : utilisateur enregistré).
    - **Données d'entrée** : Identifiants à saisir.
    - **Action** : Étapes à suivre (ex : cliquer sur "Connexion").
    - **Résultat attendu** : Comportement attendu (ex : accès au tableau de bord).

---

5. Types de tests

Exercice 5.1 : Classification des types de test

Objectif : Associer chaque type de test à sa caractéristique (fonctionnel, non-fonctionnel, structurel).

Consigne détaillée :
Classez les types de test suivants dans le tableau. Justifiez brièvement pour chaque ligne.

Exercice 5.2 : Étude de cas – Test d'acceptation

Objectif : Simuler un test d'acceptation utilisateur (UAT) pour une application de gestion de bibliothèque.

Consigne détaillée :

1. **Rédigez 3 critères d'acceptation** pour la fonctionnalité **"Emprunt de livre"** :

    - Exemple : "L'utilisateur doit pouvoir emprunter un livre disponible en 2 clics maximum."
2. **Décrivez une session de test d'acceptation** avec les utilisateurs finaux :
    - Qui participe ? (ex : bibliothécaires, étudiants)
    - Quel environnement ? (ex : bibliothèque réelle ou simulation)
    - Quels outils ? (ex : questionnaire de satisfaction, grille d'évaluation)

Exemple de réponse :

Critères d'acceptation :

1. L'emprunt d'un livre doit être possible en moins de 30 secondes.
2. L'utilisateur doit recevoir une confirmation par email.
3. Le système doit bloquer l'emprunt si le livre est déjà réservé.

Session de test :

- **Participants** : 5 bibliothécaires et 10 étudiants volontaires.
- **Environnement** : Version bêta de l'application déployée sur des tablettes en bibliothèque.
- **Outils** :
    - Grille d'évaluation avec des notes de 1 à 5 pour chaque critère.
    - Questionnaire de satisfaction post-test.

---

6. Analyse des causes racines

Exercice 6.1 : Méthode des 5 Pourquoi

Objectif : Appliquer la méthode des 5 Pourquoi pour identifier la cause racine d'un problème.

Consigne détaillée :
À partir du scénario suivant, utilisez la méthode des 5 Pourquoi pour identifier la cause racine.

Scénario : Les utilisateurs d'une application de paiement en ligne reçoivent des erreurs lors du paiement par carte bancaire.

Cause racine : **Documentation obsolète de l'API de la banque**.
Action corrective : Mettre à jour la documentation et former les développeurs.

---

7. Travail pratique : Campagne de test complète

Exercice 7.1 : Simulation de campagne de test

Objectif : Mettre en pratique l'ensemble des concepts pour un projet fictif.

Consigne détaillée :
Pour un **site e-commerce** (ex : vente de vêtements), réalisez les étapes suivantes :

1. **Planifiez les niveaux de test** :
    - Tests unitaires pour chaque fonction (ex : panier, paiement).
    - Tests d'intégration pour les interactions (ex : panier → paiement).
    - Tests système pour l'application complète.
    - Tests d'acceptation avec des clients réels.
2. **Rédigez 5 cas de test** pour la fonctionnalité **"Paiement par carte bancaire"** :
    - Exemple : Paiement réussi, carte expirée, solde insuffisant.
3. **Identifiez les risques** et proposez une **stratégie de test basée sur les risques** :
    - Exemple : Risque de fraude → tests de sécurité renforcés.
4. **Prévoyez une session de clôture** :
    - Rapport de synthèse avec le taux de couverture des tests.
    - Liste des bugs résiduels et leur criticité.