# Strategie de test

## Introduction

Dans le cadre de notre stratégie, nous visons à atteindre une fiabilité sans faille, assurant que l'application fonctionne de manière fluide et sans erreurs, pour que rien ne vienne perturber l'expérience utilisateur. Enfin, notre objectif est de garantir une performance exceptionnelle, permettant à l'application de gérer un grand nombre d'utilisateurs et de transactions sans ralentissement, évitant ainsi de se transformer en un dispositif lent et frustrant.

En développant des cas de test détaillés axés sur les fonctionnalités de l'application, nous allons mettre l'accent sur les scénarios d'utilisation réelle ainsi que sur les parcours moins fréquentés. De plus, l'intégration du Behavior-Driven Development (BDD) et du Test-Driven Development (TDD) dans notre processus de conception jouera un rôle clé. Cette approche nous permettra de mieux comprendre les exigences à travers le prisme des comportements utilisateurs et de garantir que chaque aspect de l'application est rigoureusement testé dès les premières étapes du développement, assurant ainsi une qualité et une fiabilité accrues.

Analyse Amdec

scénario de test :

- Création de Tournoi
    - Création Simple : Tester la création d'un tournoi avec les paramètres de base (nom, type, participants).
- Validation des Entrées : Essayer d'entrer des valeurs invalides pour voir si l'application gère bien les erreurs.
- Gestion des Participants

**TeamGenerator.test.js**

1. Test de l'instanciation de TeamGenerator

Description : Ce test vérifie que la classe TeamGenerator peut être instanciée correctement avec les paramètres players et playersPerTeam.

Méthode Testée : Constructeur de TeamGenerator.

Étapes du Test : Instancier un objet TeamGenerator avec des joueurs et le nombre de joueurs par équipe.
Vérifier que l'objet est une instance de TeamGenerator.

Assertions : L'objet créé est une instance de TeamGenerator.

2. Test de la génération des équipes

Description : Ce test vérifie que les équipes sont correctement générées avec le bon nombre de joueurs par équipe.

Méthode Testée : Méthode generateTeams() de TeamGenerator.

Étapes du Test :
Instancier un objet TeamGenerator avec une liste de joueurs et le nombre de joueurs par équipe.
Appeler la méthode generateTeams().

Assertions : Le nombre d'équipes générées est correct.
Chaque équipe a le bon nombre de joueurs.

3. Test de récupération des équipes

Description : Ce test vérifie que la méthode getTeams() renvoie les équipes générées correctement.

Méthode Testée : Méthode getTeams() de TeamGenerator.

Étapes du Test : Instancier un objet TeamGenerator avec une liste de joueurs et le nombre de joueurs par équipe.
Générer les équipes.
Appeler la méthode getTeams().

Assertions : Les équipes renvoyées sont identiques aux équipes générées.

4. Test des cas particuliers

Description : Ces tests vérifient la gestion des cas particuliers comme une liste de joueurs vide ou un nombre impair de joueurs.

Méthode Testée : Méthode generateTeams() de TeamGenerator.

Étapes du Test : Instancier un objet TeamGenerator avec des cas particuliers spécifiques. Appeler la méthode generateTeams().

Assertions : Les équipes sont générées correctement même avec une liste de joueurs vide ou un nombre impair de joueurs.

**TournamentGenerator.test.js**

1. Test de la génération des poules

Description : Ce test vérifie que les poules sont correctement générées avec le bon nombre d'équipes.

Méthode Testée : Méthode generatePoules() de TournamentGenerator.

Étapes du Test : Instancier un objet TournamentGenerator avec une liste d'équipes.
Appeler la méthode generatePoules().

Assertions : Le nombre de poules générées est correct.
Chaque poule contient le bon nombre d'équipes.

2. Test de la simulation des matchs de poules

Description : Ce test vérifie que les matchs de poules sont correctement simulés et que les équipes sont qualifiées pour les phases finales.

Méthode Testée : Méthode simulatePoulesMatches() de TournamentGenerator.

Étapes du Test : Instancier un objet TournamentGenerator avec une liste d'équipes.
Générer les poules.
Appeler la méthode simulatePoulesMatches().

Assertions : Les équipes qualifiées pour les phases finales sont correctes.

3. Test de la génération des phases finales

Description : Ce test vérifie que les phases finales sont correctement générées.

Méthode Testée : Méthode generateFinalStages() de TournamentGenerator.

Étapes du Test : Instancier un objet TournamentGenerator avec une liste d'équipes.

Générer les poules et simuler les matchs de poules.
Appeler la méthode generateFinalStages().

Assertions : Les phases finales sont générées correctement.

4. Test de la génération complète du tournoi (Test d'Intégration)

Description : Ce test vérifie que l'ensemble du processus de génération de tournoi, depuis la création des équipes jusqu'aux phases finales, fonctionne correctement.

Étapes du Test : Instancier un objet TournamentGenerator avec une liste d'équipes.
Appeler la méthode generateTournament().

Assertions : L'ensemble du tournoi est généré correctement.

AMDEC :

| Élément | Mode de Défaillance | Effets Potentiels | Gravité | Fréquence | Détection |
|---|---|---|---|---|---|
| Création de Tournoi | Création Simple : Échec de création du tournoi avec paramètres invalides ou manquants | Incapacité de créer un tournoi correctement, ce qui peut entraîner des erreurs dans les données du tournoi ou un dysfonctionnement de l'application | 3 | Moyenne | Moyenne |
| Création de Tournoi | Validation des Entrées: Gestion incorrecte des entrées invalides | Possibilité de saisir des données incorrectes ou manquantes lors de la création du tournoi, pouvant entraîner des erreurs ou un comportement inattendu de l'application | 4 | Élevée | Élevée |
| Création de Tournoi | Gestion des Participants: Échec de gestion des participants | Ajout incorrect des participants au tournoi, non-respect des restrictions de nombre de participants par équipe | 3 | Moyenne | Moyenne |
| Génération des Équipes | Équipes non générées correctement | Création incorrecte des équipes, entraînant un nombre incorrect de joueurs par équipe ou des équipes inégales | 3 | Moyenne | Moyenne |
| Génération des Poules | Poules non générées correctement | Création incorrecte des poules, avec un nombre incorrect d'équipes par poule | 3 | Moyenne | Moyenne |
| Simulation des Matchs de Poules | Échec de la simulation des matchs de poules | Incohérences dans les résultats des matchs de poules, qualification incorrecte des équipes pour les phases finales | 4 | Élevée | Moyenne |

| | | | | | |
|---|---|---|---|---|---|
| Génération des Phases Finales | Phases finales non générées correctement | Erreur dans la génération des phases finales, conduisant à un mauvais déroulement du tournoi ou à une sélection incorrecte des équipes gagnantes | 4 | Moyenne | Moyenne |
| Génération Complète du Tournoi (Test Fonctionnel) | Échec de la génération complète du tournoi | Erreur dans le processus de génération de tournoi, entraînant un dysfonctionnement de l'ensemble de l'application | 5 | Élevée | Faible |