Plan de test

1. Introduction

L'Objectif de ce plan de test sera de valider la partie fonctionnalité.
En termes de périmètre, la couverture s'étendra à la création, la gestion, l'exécution et le suivi des résultats des tournois.

Notre scope comprend des fonctionnalités essentielles à les générations de tournois :

- La création des équipes à partir de joueurs
- Les créations de tournois

Quant à notre out of scope il comprend la fonctionnalité qui permet de charger des fichiers afin de générer des équipes ou de charger des équipes prêtes à concourir.

Dans le but de nous assurer que notre programme répond aux fonctionnalités attendues, nous allons mettre en place différents types de test :

- Test Unitaires : Dans le but de tester les parties importantes de lié à la logique de notre code
- Test d'intégration : Ici notre souhait est de nous assurer que l'ajout de nouvelles fonctionnalités ne viennent pas casser le code existant
- Tests Fonctionnels : Pour vérifier chaque fonctionnalité contre ses spécifications.
- etc.

2. Environnement de Test

Le projet étant dans une phase de prototype. Nous exécuterons les tests sur nos propre machine. Il n'est pas nécessaire d'automatiser les tests aujourd'hui ou de prévoir des environnements de déploiement

3. Analyse de risque

En nous basant sur le scope défini, une première analyse de risque peut être réalisée pour identifier les zones critiques du système où des problèmes sont plus susceptibles de survenir. Voici quelques risques potentiels à prendre en compte :

Risque	Probabilité	Impact	Criticité
Complexité des règles de génération de tournois	Moyenne	Les règles complexes peuvent entraîner des erreurs dans la génération de tournois et des résultats imprévisibles.	Élevée
Gestion incorrecte des équipes	Moyenne	Une mauvaise gestion des équipes peut conduire à des tournois mal équilibrés ou à des joueurs mal répartis, affectant l'équité de la compétition.	Moyenne
Dépendance aux entrées utilisateur	Élevée	Une forte dépendance aux entrées utilisateur peut entraîner des erreurs de saisie ou des données incorrectes, affectant la qualité des tournois générés.	Élevée
Évolutivité	Moyenne	Une faible évolutivité peut rendre difficile l'ajout de nouvelles fonctionnalités ou la correction de bogues à mesure que les besoins changent, limitant ainsi la durée de vie du système.	Élevée
Stabilité du système	Moyenne	Des problèmes de stabilité peuvent entraîner des plantages ou des dysfonctionnements du système, entraînant des interruptions de service et une perte de confiance des utilisateurs.	Élevée
Performance	Moyenne	Des performances insatisfaisantes peuvent entraîner des retards dans la génération des tournois ou dans l'affichage des résultats, affectant négativement l'expérience utilisateur.	Moyenne

Scénario de test 1 : Génération du bon nombre de phases finales

Description : Ce test vérifie si la méthode **generateTournament** de la classe **TournamentGenerator** génère le bon nombre de phases finales pour un tournoi avec un ensemble donné d'équipes.

Préconditions : Des équipes doivent être fournies à la classe **TournamentGenerator**.
Étapes du test :

1. Créer un ensemble d'équipes avec un nombre prédéfini de joueurs.
2. Appeler la méthode **generateTournament** de la classe **TournamentGenerator** avec cet ensemble d'équipes.
3. Vérifier que la longueur du tableau **finalStages** généré est supérieure à zéro.

Résultat attendu : La méthode **generateTournament** doit générer un nombre de phases finales supérieur à zéro.

Scénario de test 2 : Qualification correcte des équipes pour les phases finales

Description : Ce test vérifie si la méthode **simulatePoulesMatches** de la classe **TournamentGenerator** qualifie correctement les équipes pour les phases finales après avoir généré les poules.

Préconditions : Les poules doivent être générées à l'aide de la méthode **generatePoules**.

Étapes du test :

1. Créer un ensemble d'équipes avec un nombre prédéfini de joueurs.
2. Générer les poules en appelant la méthode **generatePoules** de la classe **TournamentGenerator**.
3. Appeler la méthode **simulatePoulesMatches** pour qualifier les équipes pour les phases finales.
4. Vérifier que la longueur du tableau **finalStages** généré est supérieure à zéro.

Résultat attendu : La méthode **simulatePoulesMatches** doit qualifier un certain nombre d'équipes pour les phases finales.

Scénario de test 3 : Génération du bon nombre de phases finales après simulation

Description : Ce test vérifie si la méthode **generateFinalStages** de la classe **TournamentGenerator** génère le bon nombre de phases finales après simulation des matches de poules.

Préconditions : Les équipes doivent être qualifiées pour les phases finales à l'aide de la méthode **simulatePoulesMatches**.

Étapes du test :

1. Créer un ensemble d'équipes avec un nombre prédéfini de joueurs.
2. Générer les poules en appelant la méthode **generatePoules** de la classe **TournamentGenerator**.
3. Qualifier les équipes pour les phases finales en appelant la méthode **simulatePoulesMatches**.
4. Générer les phases finales en appelant la méthode **generateFinalStages**.
5. Vérifier que la longueur du tableau **finalStages** généré est supérieure à zéro.

Résultat attendu : La méthode **generateFinalStages** doit générer un nombre de phases finales supérieur à zéro après la simulation des matches de poules.