User Story 1 : En tant qu'entraîneur, je souhaite pouvoir récupérer le nombre de stratégies disponibles d'un footballeur afin de planifier les tactiques de l'équipe.

Critères d'acceptation :
- L'entraîneur peut accéder au nombre de stratégies disponibles d'un footballeur.
- Lorsqu'un footballeur est créé, il a un nombre de stratégies par défaut.
- L'entraîneur peut obtenir le nombre de stratégies disponibles à partir d'un objet footballeur.

Test Unitaire : testGetgetnombrestrategies()

User Story 2 : En tant qu'entraîneur, je souhaite connaître le nombre de stratégies utilisées par un footballeur afin de suivre leur utilisation sur le terrain.

Critères d'acceptation :
- L'entraîneur peut accéder au nombre de stratégies utilisées par un footballeur.
- Lorsqu'un footballeur est créé, il a un nombre de stratégies utilisées par défaut.
- L'entraîneur peut obtenir le nombre de stratégies utilisées à partir d'un objet footballeur.

Test Unitaire : testGetgetstrategiesutilises()

User Story 3 : En tant qu'entraîneur, je souhaite calculer la différence entre le nombre de stratégies disponibles et le nombre de stratégies utilisées par un footballeur afin de mesurer leur efficacité sur le terrain.

Critères d'acceptation :
- L'entraîneur peut obtenir la différence entre le nombre de stratégies disponibles et le nombre de stratégies utilisées par un footballeur.
- La différence est calculée correctement en soustrayant le nombre de stratégies utilisées du nombre de stratégies disponibles.

Test Unitaire : testDifference()

User Story 4 : En tant qu'entraîneur, je souhaite pouvoir vérifier la différence totale entre le nombre de stratégies disponibles et le nombre de stratégies utilisées de tous les footballeurs de mon équipe afin d'évaluer la performance globale de l'équipe.

Critères d'acceptation :
- L'entraîneur peut appeler la méthode verifyDifference() sur un objet Equipe pour obtenir la différence totale entre le nombre de stratégies disponibles et le nombre de stratégies utilisées de tous les footballeurs de l'équipe.
- La méthode verifyDifference() calcule correctement la différence totale en additionnant les différences de tous les footballeurs de l'équipe.

Test Unitaire : testverifyDifference()

User Story 5 : En tant qu'entraîneur, je souhaite pouvoir ajouter un footballeur à mon équipe afin de former une composition d'équipe complète.

Critères d'acceptation :
- L'entraîneur peut ajouter un footballeur à l'équipe en utilisant la méthode addFootballeur().
- Le footballeur est correctement ajouté à la liste des footballeurs de l'équipe.
- L'équipe est associée au footballeur ajouté en utilisant la méthode addEquipe() de la classe Footballeur.

User Story 6 : En tant qu'entraîneur, je souhaite connaître le nombre de footballeurs présents dans mon équipe afin de gérer la taille de l'équipe.

Critères d'acceptation :
- L'entraîneur peut obtenir le nombre de footballeurs dans l'équipe en appelant la méthode getSize() sur un objet Equipe.
- La méthode getSize() renvoie le nombre correct de footballeurs présents dans l'équipe.

User Story 7 : En tant qu'entraîneur, je souhaite connaître le nom d'un Footballeur.

Critères d'acceptation :
- L'entraîneur peut obtenir le nom du Footballeur en appelant la méthode getNom() sur un objet Footballeur.
- La méthode getNom() renvoie le nom du Footballeur.

User Story 8 : En tant qu'utilisateur, je souhaite pouvoir obtenir le titre d'un film afin de connaître son nom.

Critères d'acceptation :
- L'utilisateur peut appeler la méthode getTitre() sur un objet Film pour obtenir le titre du film.
- La méthode getTitre() renvoie correctement le titre du film.

User Story 9 : En tant qu'utilisateur, je souhaite pouvoir obtenir le réalisateur d'un film afin de connaître la personne qui l'a réalisé.

Critères d'acceptation :
- L'utilisateur peut appeler la méthode getRealisateur() sur un objet Film pour obtenir le réalisateur du film.
- La méthode getRealisateur() renvoie correctement le nom du réalisateur du film.

User Story 10 : En tant qu'utilisateur, je souhaite pouvoir modifier le titre d'un film afin de mettre à jour son nom.

Critères d'acceptation :
- L'utilisateur peut appeler la méthode setTitre() sur un objet Film pour définir un nouveau titre pour le film.
- Le titre du film est correctement mis à jour avec la nouvelle valeur spécifiée.

User Story 11 : En tant qu'utilisateur, je souhaite pouvoir modifier le réalisateur d'un film afin de mettre à jour les informations sur le réalisateur.

Critères d'acceptation :
- L'utilisateur peut appeler la méthode setRealisateur() sur un objet Film pour définir un nouveau réalisateur pour le film.
- Le réalisateur du film est correctement mis à jour avec la nouvelle valeur spécifiée.

User Story 12 : En tant qu'utilisateur, je souhaite pouvoir obtenir la liste des cinémas où le film est diffusé afin de connaître les lieux de projection.

Critères d'acceptation :
- L'utilisateur peut appeler la méthode getListeCinemas() sur un objet Film pour obtenir la liste des cinémas où le film est diffusé.
- La méthode getListeCinemas() renvoie correctement la liste des cinémas où le film est diffusé.

User Story 13 : En tant qu'utilisateur, je souhaite pouvoir obtenir le nom d'un cinéma afin de connaître son nom.

Critères d'acceptation :
- L'utilisateur peut appeler la méthode getNom() sur un objet Cinema pour obtenir le nom du cinéma.
- La méthode getNom() renvoie correctement le nom du cinéma.

User Story 14 : En tant qu'utilisateur, je souhaite pouvoir ajouter un film à la liste des films diffusés par un cinéma.

Critères d'acceptation :
- L'utilisateur peut appeler la méthode addFilm() sur un objet Cinema en passant un objet Film en tant que paramètre pour ajouter un nouveau film à la liste des films diffusés par le cinéma.
- Le film est correctement ajouté à la liste des films du cinéma.
- Le cinéma est associé au film ajouté en appelant la méthode addCinema() de la classe Film.

User Story 15 : En tant qu'utilisateur, je souhaite pouvoir connaître le nombre de films diffusés par un cinéma.

Critères d'acceptation :
- L'utilisateur peut appeler la méthode nbFilms() sur un objet Cinema pour obtenir le nombre de films diffusés par le cinéma.
- La méthode nbFilms() renvoie correctement le nombre de films diffusés par le cinéma.

User Story 16 : En tant qu'utilisateur, je souhaite pouvoir obtenir la liste des films diffusés par un cinéma.

Critères d'acceptation :
- L'utilisateur peut appeler la méthode getFilms() sur un objet Cinema pour obtenir la liste des films diffusés par le cinéma.
- La méthode getFilms() renvoie correctement la liste des films diffusés par le cinéma.

User Story 17 : En tant qu'utilisateur, je souhaite pouvoir créer un objet FilmFoot en associant un film et un Footballeur principale qui y joue.

Critères d'acceptance :
- L'utilisateur peut créer une instance de la classe FilmFoot en fournissant un objet Film et un objet Footballeur.
- La classe FilmFoot stocke correctement le film et le Footballeur principal.

User Story 18 : En tant qu'utilisateur, je souhaite pouvoir récupérer le film associé à la classe FilmFoot.

Critères d'acceptance :
- L'utilisateur peut appeler la méthode getFilm() pour obtenir l'objet Film associé.
- La méthode getFilm() renvoie le film associé.

User Story 19 : En tant qu'utilisateur, je souhaite pouvoir définir un nouveau film pour la classe FilmFoot.

Critères d'acceptance :
- L'utilisateur peut appeler la méthode setFilm() pour définir un nouveau film.
- La méthode setFilm() met à jour correctement la référence du nouveau film.

User Story 20 : En tant qu'utilisateur, je souhaite pouvoir avoir la liste des cinémas où le film associé à la classe FilmFoot est diffusé.

Critères d'acceptance :

- *L'utilisateur* peut appeler la méthode jouerFilm() pour afficher le titre du Film, le nom du Footballeur principale et la liste des cinémas où ce dernier est diffusé.
- La méthode jouerFilm() affiche correctement le titre du film, le nom du Footballeur principal et la liste des cinémas où ce dernier est diffusé.

Couverture des Tests : 80%