Le but de ce moteur de jeux sera de gérer:

\*la logique du jeu

\*les règles

\*les mécanismes de jeu,

\*les interactions entre les cartes,

\*les phases de jeu

\*maintient l'état actuel du jeu ( points de vie, les effets en cours,) etc.

\*gérer les actions des joueurs,

\*les effets de cartes,

\*les attaques

\*etc

le moteur => catalogue( pour créer les decks de chaque joueur en fonction des cartes sélectionnées.)

Interfaces :

**CardInterface** : represente les propriétés et méthodes communes à toutes les cartes du jeu, telles que le coût, les dégâts, etc.

getCost(): int : Renvoie le coût de la carte.

getName(): string : Renvoie le nom de la carte.

getDamage(): int : Renvoie les dégâts infligés par la carte.

**ActionInterface** : pour représenter une action effectuée par un joueur. Les actions peuvent être jouer une carte, attaquer, défendre (jsp si ca existe)

playCard(CardInterface card, PlayerInterface targetPlayer): Permet à un joueur de jouer une carte sur un joueur cible.

attack(PlayerInterface attacker, PlayerInterface target): Permet à un joueur d'attaquer un autre joueur.

**PlayerInterface :**

getHealth(): int : Renvoie la santé actuelle du joueur.

getMana(): int : Renvoie la quantité de mana disponible pour le joueur.

drawCard(): CardInterface : Permet au joueur de piocher une carte.

takeDamage(int amount): Fait subir des dégâts au joueur.

playAction(ActionInterface action): Permet au joueur d'effectuer une action spécifique.

**DeckInterface :**

getCards(): List<CardInterface> : Renvoie la liste des cartes présentes dans le deck.

addCard(CardInterface card): Ajoute une carte au deck.

shuffle(): Mélange les cartes dans le deck.

**GameInterface :**

startGame(): Démarre une nouvelle partie.

endGame(PlayerInterface winner): Termine la partie et déclare un gagnant.