# Documentation of the Game **"Art of Magic"**

# 1 Introduction

*Art of Magic* is a unique strategic card game where players are judged after each battle, evaluating their performance in the duel. The assessment is based on various aspects of their play, including remaining health after the battle, the number of cards of a certain class used, and other key moments of strategy and tactics. Judges analyze these factors to deliver a verdict that can influence future games and strategies of the participants.

# 2 Basic Rules

## 2.1 Beginning of the Game

Players choose their decks of 30 cards and start with 30 health points. The game's goal is to use the cards to reduce the opponent's health to zero.

## 2.2 Gameplay

On their turn, a player draws a card from the deck and can use mana to play cards. Players aim to attack the opponent and defend themselves, using creature cards, spells, and artifacts.

## 2.3 Victory

The player who first reduces the opponent's health to zero wins.

# 3 Game Modes

## 3.1 Single Player

Players can play against computer intelligence.

## 3.2 Multiplayer

Allows players to battle each other online.

# 4 Decks and Cards

## 4.1 Deck Creation

Players create decks by choosing cards from a common pool. Striving for deck balance is crucial.

## 4.2   Types of Cards

- Creatures: Can attack and defend.

- Spells: Provide various effects.

- Artifacts: Offer unique advantages or change the rules.
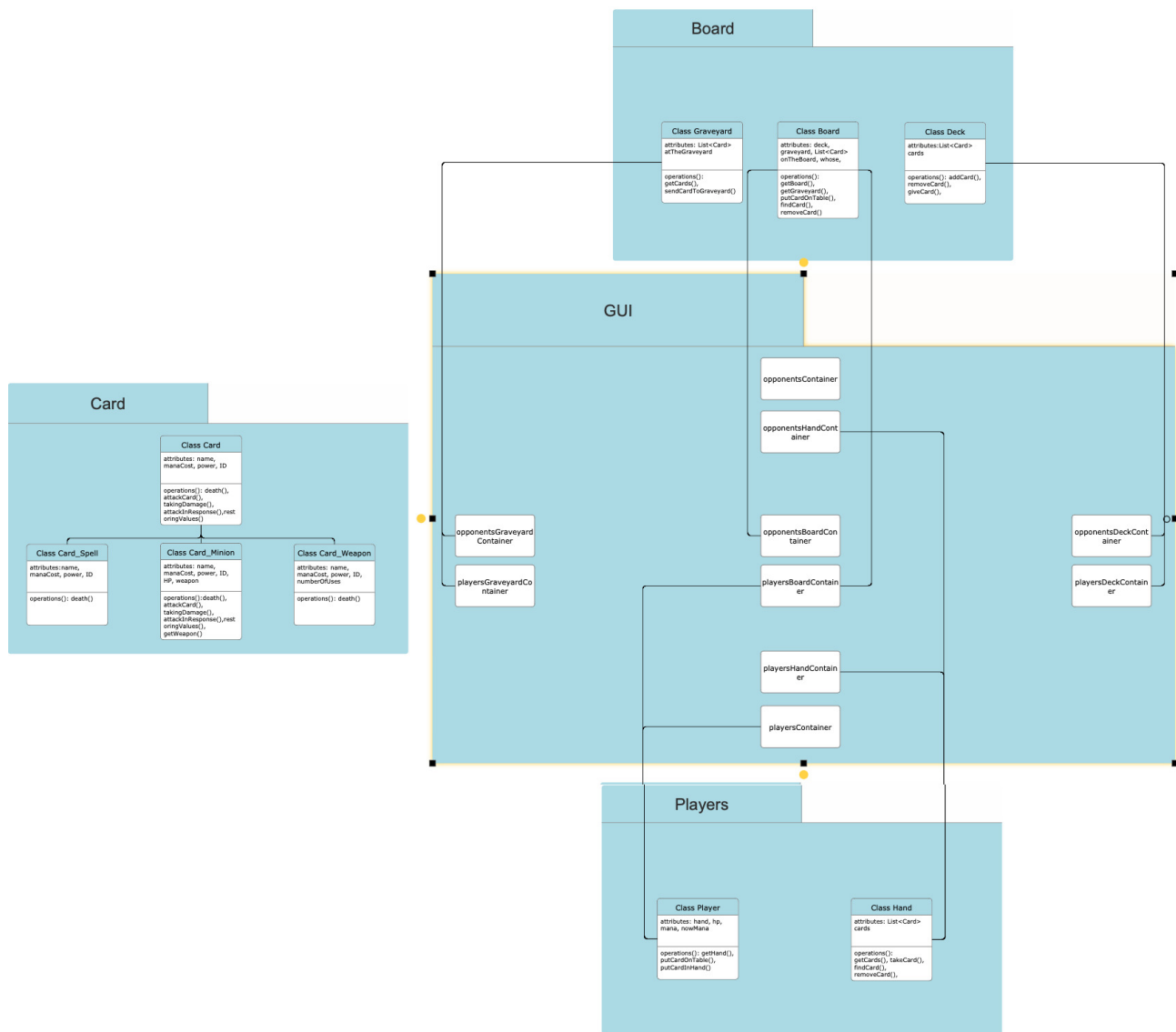
# 5   Game Interface

*Art of Magic* is designed with a user-friendly and intuitive interface, allowing easy management of cards and monitoring of the game's progress.
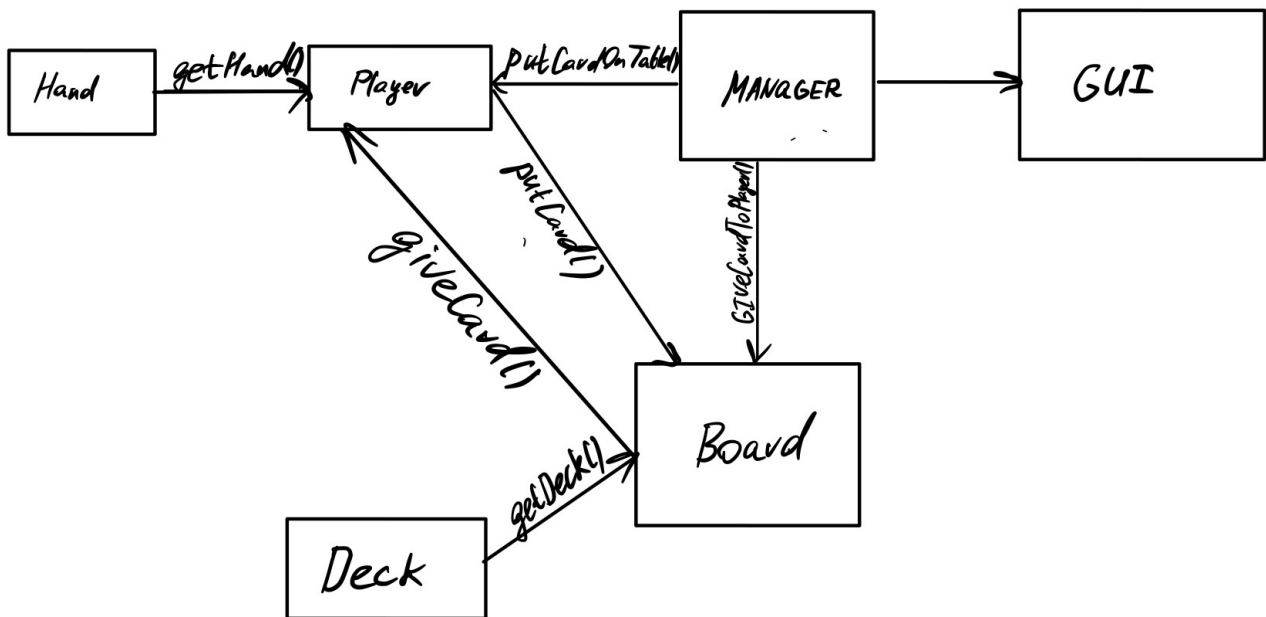
# 6   Conclusion

*Art of Magic* offers a deep and engaging gameplay experience, combining elements of strategy and tactics. Create your deck and participate in magical duels to claim the title of the greatest mage!
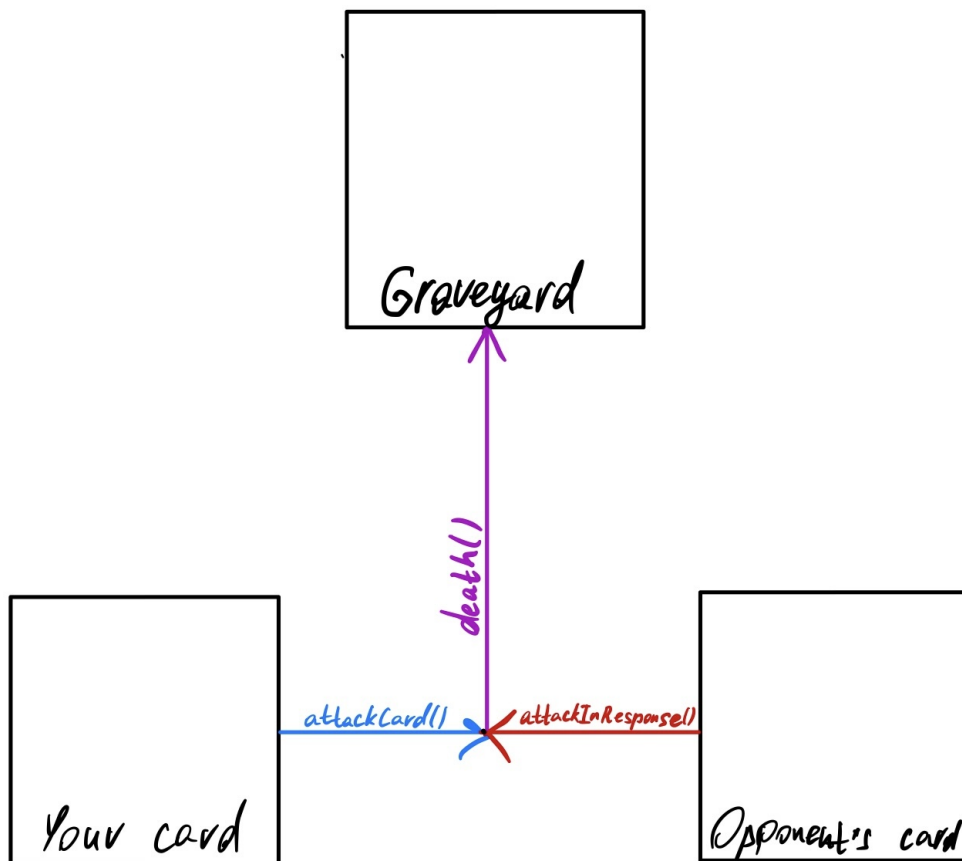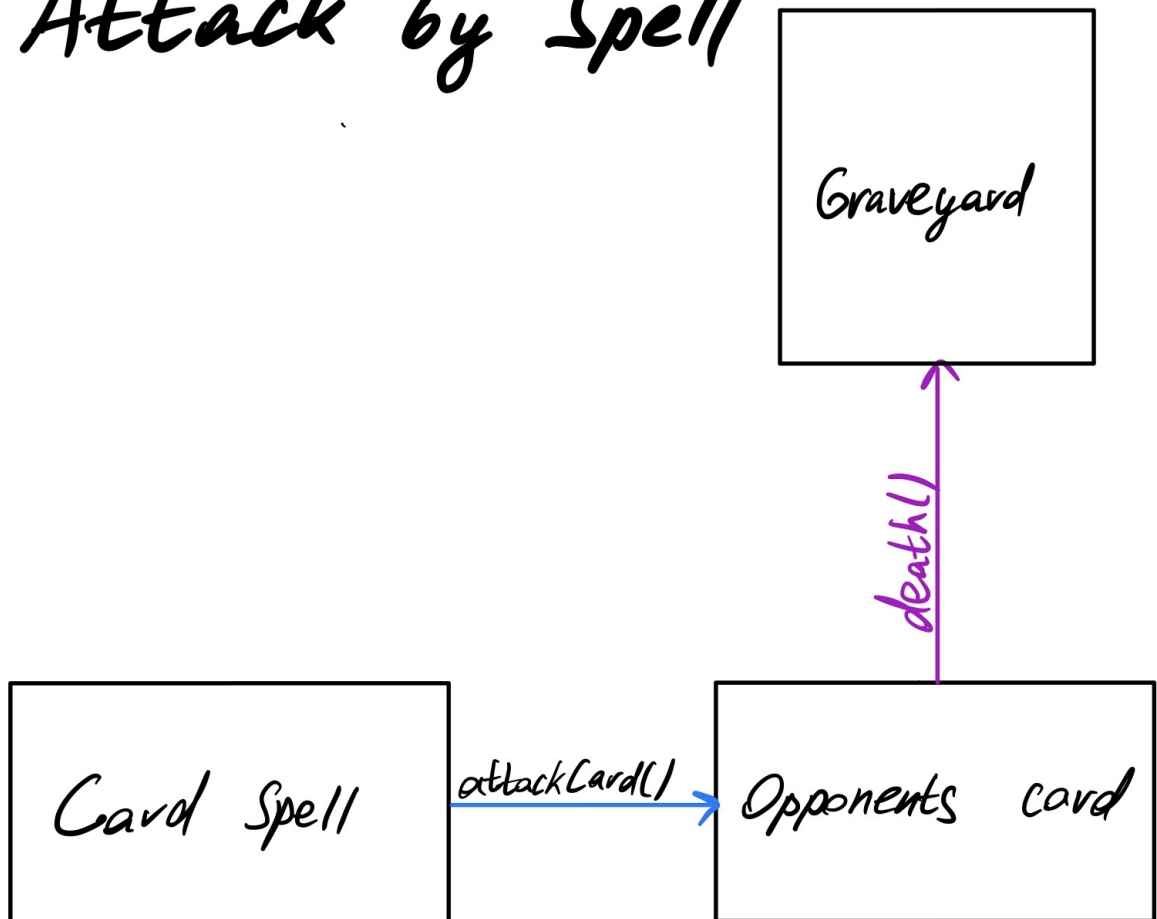
# 7 Class diagram

## Board

**Class Graveyard**

attributes: List<Card>
atTheGraveyard

operations():
getCards(),
sendCardToGraveyard()

**Class Board**

attributes: deck,
graveyard, List<Card>
onTheBoard, whose,

operations():
getBoard(),
getGraveyard(),
putCardOnTable(),
findCard(),
removeCard()

**Class Deck**

attributes:List<Card>
cards

operations(): addCard(),
removeCard(),
giveCard(),

## Card

**Class Card**

attributes: name,
manaCost, power, ID

operations(): death(),
attackCard(),
takingDamage(),
attackInResponse(),rest
oringValues()

**Class Card_Spell**

attributes:name,
manaCost, power, ID

operations(): death()

**Class Card_Minion**

attributes: name,
manaCost, power, ID,
HP, weapon

operations():death(),
attackCard(),
takingDamage(),
attackInResponse(),rest
oringValues(),
getWeapon()

**Class Card_Weapon**

attributes: name,
manaCost, power, ID,
numberOfUses

operations(): death()

## GUI

opponentsContainer

opponentsHandCont
ainer

opponentsGraveyard
Container

opponentsBoardCon
tainer

opponentsDeckCont
ainer

playersGraveyardCo
ntainer

playersBoardContain
er

playersDeckContain
er

playersHandContain
er

playersContainer

## Players

**Class Player**

attributes: hand, hp,
mana, nowMana

operations(): getHand(),
putCardOnTable(),
putCardInHand()

**Class Hand**

attributes: List<Card>
cards

operations():
getCards(), takeCard(),
findCard(),
removeCard(),

# 8 Logic

# Attack by Minion



Graveyard

Your card

Opponent's card

death()

attackCard()

attackInResponse()

blue color-sends a request
red color —responds to a request
purple color —will work under certain conditions

# Attack by Spell
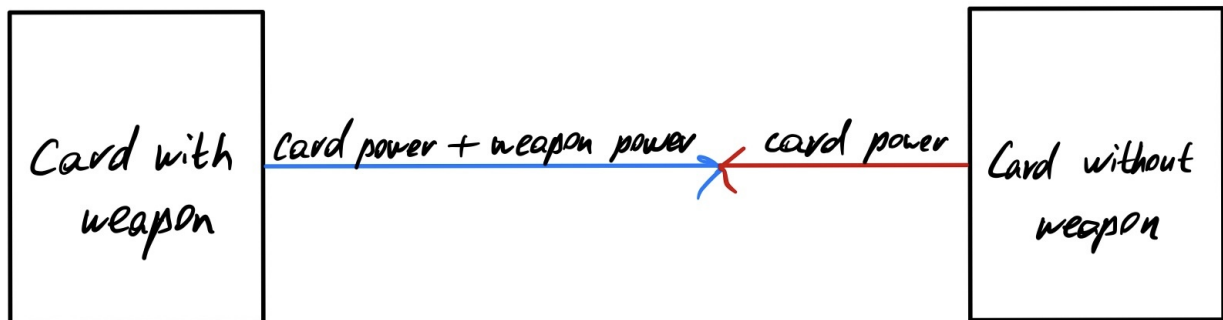
Graveyard

Card Spell —attackCard()→ Opponents card

death()

blue color—sends a request
red color —responds to a request
purple color —will work under certain conditions

# Attack with weapon



```
┌──────────────┐                                              ┌──────────────┐
│  Card with   │  Card power + weapon power    card power     │ Card without │
│   weapon     │ ─────────────────────────>< ──────────────  │   weapon     │
│              │                                              │              │
└──────────────┘                                              └──────────────┘
```

blue color — sends a request
red color — responds to a request