Project Summary

- **What is the Project called?**
  - Exploding Kittens Game
- **Who is on your team (include all names)?**
  - Arpita Ambavane
  - Abishek Goutham
- **What is the high-level overview of your semester project? What are you trying to accomplish? What will your system do when you are done?**
  - An Exploding Kitten game is a strategy and luck powered two player game. In this game, players take turns to play their cards. Players draw cards until somebody draws an Exploding Kitten, at which point they explode and are out of the game. To avoid exploding, they can defuse the kitten OR use powerful action cards to move or avoid the Exploding Kitten. The players have the option to play as many cards as they want during their turn, but they must conclude their turn by drawing a card from the deck of unplayed cards. The game's objective is to avoid drawing the Exploding Kitten card. The last player left alive wins.

Project Requirements

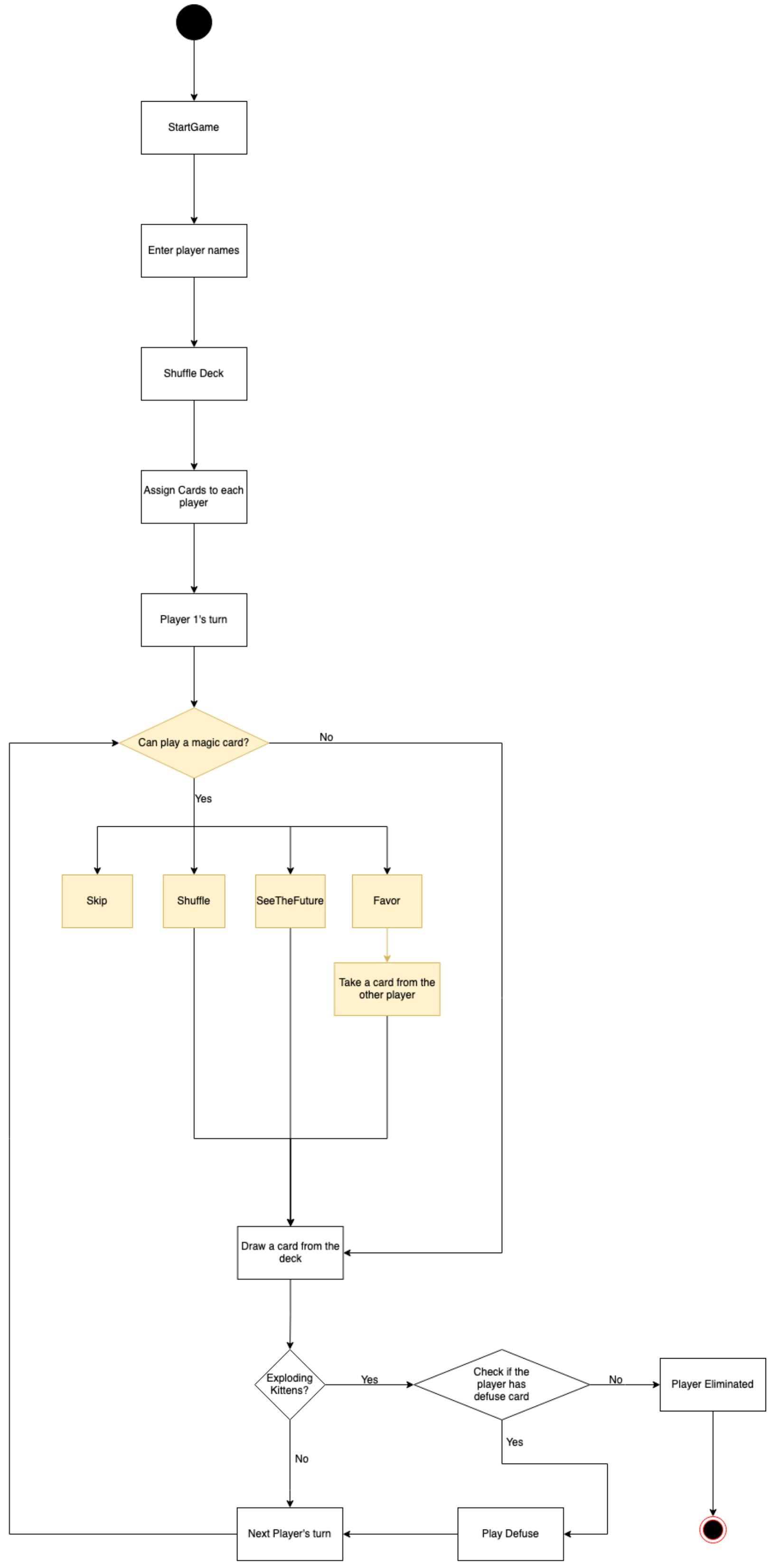
- **functional capabilities (what it can do)**
  1. Taking the Names of the players (input from user).
  2. Shuffle the deck of cards at the beginning of the game.
  3. Deal seven cards to each player at the start of the game.
  4. Allow players to take turns playing a card from their hand.
  5. Display the Number of cards left after playing.
  6. Display the cards the player has
  7. Implement the various card actions, such as exploding kittens, defusing, skipping a turn, and so on.
  8. Display the current card on play or played by another player
  9. Notifying the turn of the player - For Example: ‘Abishek’s Turn’
  10. Allows the player to draw a card from the deck.
  11. Alert the user if the option picked is invalid.
  12. To place the Exploding Kitten in the deck after playing defuse card - Give options to the player to choose where to place the Exploding Kitten card in the pile.
  13. Update the number of cards from players when a favor card is played
  14. End the game when only one player is left or when all exploding kittens have been defused.
- **Constraints (such as platforms, number of users, etc.)**
  1. Number of users : 2 players
  2. The game should be developed using Java programming language.
  3. The game should be developed to run on a web browser.
  4. The user should require Game knowledge
- **non-functional characteristics (performance goals, security, usability, etc.).**
  1. Goals: Have a fully working UI using REST-API. Eventually, configure web sockets for real time communication.

ii. Usability: The cards on the screen would be a click event. If time permits, the drag-drop feature for the cards can be configured.

Users and Tasks: Use Cases (Text or UML Diagrams) - 10 points

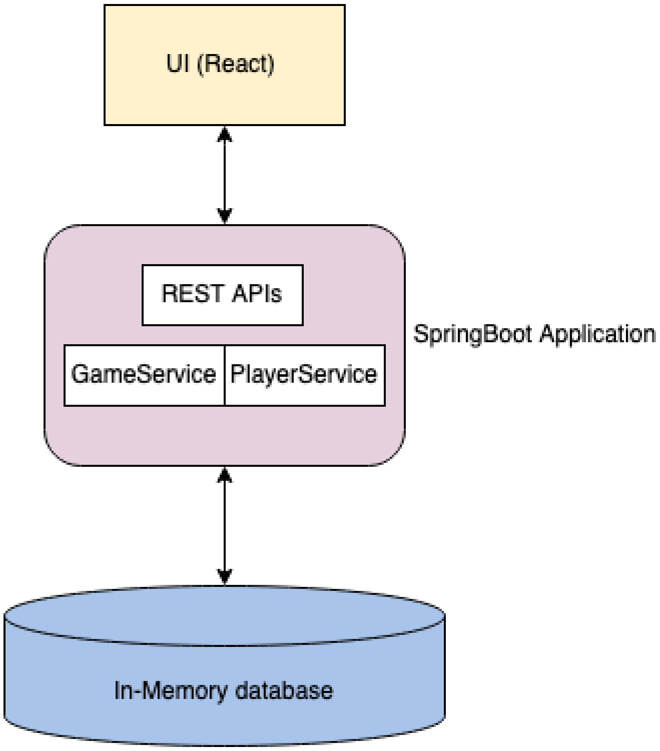
- **How many different types of users will your system have? What tasks do they need to accomplish with your system?**
    - There will be two users of type Player.
    - Create a new game with another player
    - Play a card from their hand
    - Draw a card from the deck
    - Defuse an exploding kitten
    - Skip their turn
    - Shuffle the deck
    - Favor
    - See the Top 3 cards from the deck by playing ‘See The Future’ Card
    - Try to avoid exploding kitten card from deck
- **Document how the system will support all major user tasks by providing one or more use cases (using a text template or UML).**
    - Use case: Play a Card
    - Primary Actor: Player
    - Goal in Context: A player wants to play a card from their hand.
    - Preconditions: The player has started the game and is currently in the game.
    - Trigger: The player selects a card to play from their hand.
    - Scenario:
        - **i.** The system verifies that the selected card is valid for the player to play based on the game rules.
        - **ii.** The system performs the action of the selected card.
        - **iii.** The card is removed from the player's hand and placed in the discard deck.
        - **iv.** The player's turn ends, and it becomes the opponent's turn.
        - **v.** The system updates the game state.
- **Try to think of the various problems or variations that can occur while trying to accomplish these tasks, and document those alternative situations in the use cases.**
    - If the selected card is not valid for the player to play, the system displays an error message and prompts the player to select a valid card.
    - If the player has no valid cards to play, the system automatically draws a card for them from the deck and ends the player’s turn.

UML Activity Diagram



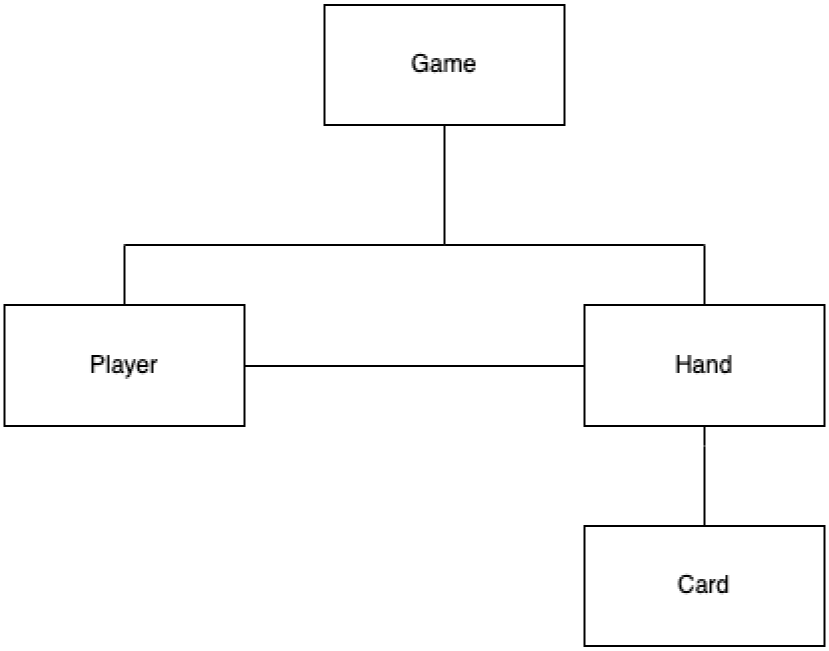
Architecture Diagram

- We are building a web page as a UI for the user to interact with and using REST APIs we make API calls to the backend which has an in-memory database.
  Frontend - React
  Backend - Spring Boot



Data Storage

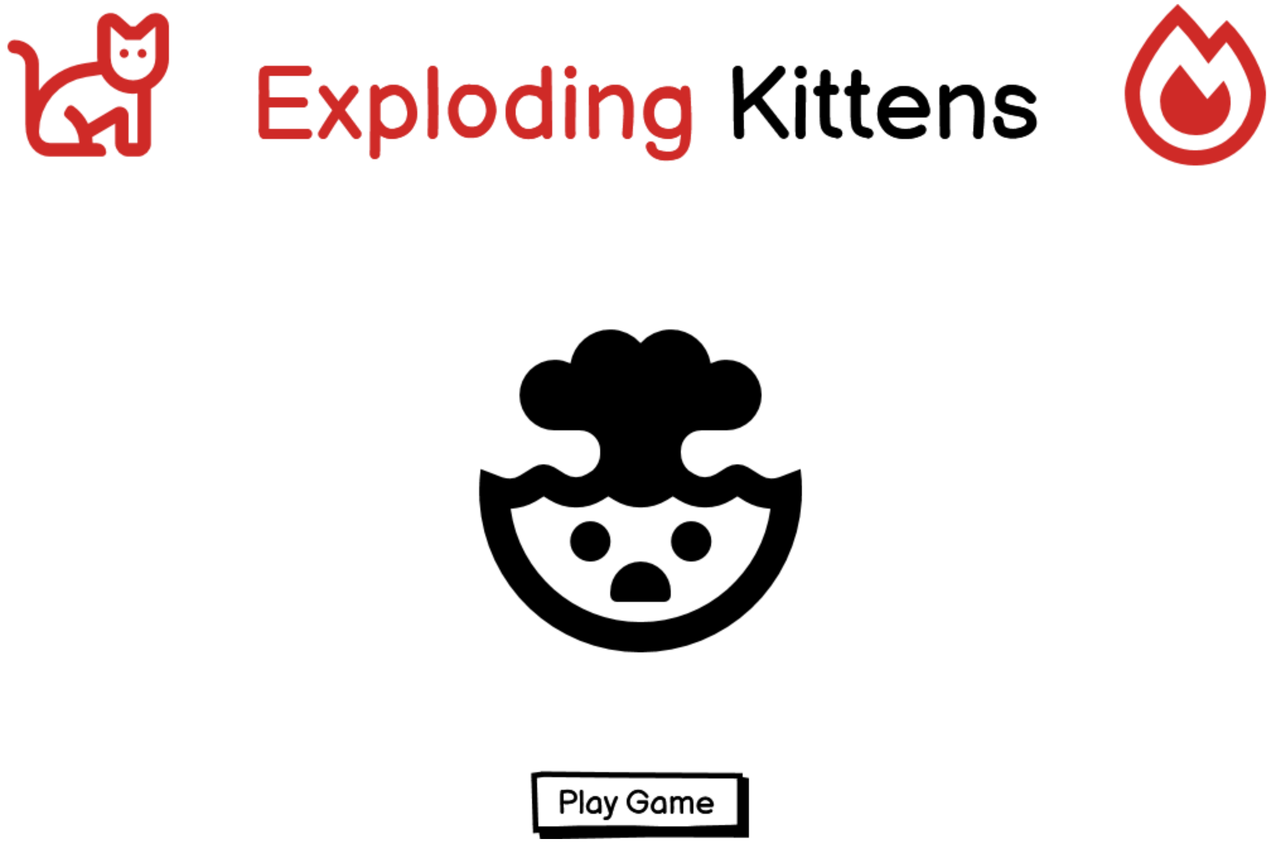
- The following data will be stored in memory :
  - Players Hand
  - Number of cards in hand for each player
  - Names of Players
  - Current card which is being played
  - Card drawn from the Deck
  - Active player
  - Count of all Magic Cards
- Classes used to access data:
  - Game
  - Player
  - Card
  - Hand



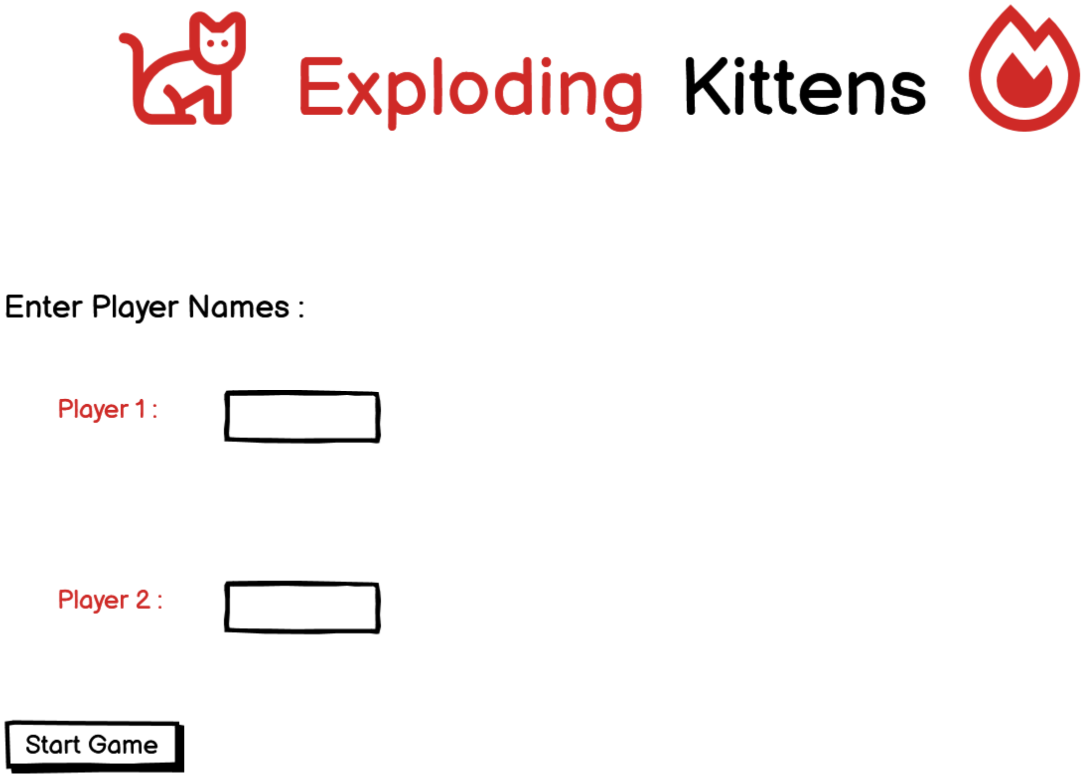
UI Mockups/Sketches

1. Screen 1 displays the home page of the Game. When the user hits the 'Play Game’ button, he is taken to Screen 2.
2. IN Screen 2, the user is asked to enter the names of players. After the user hits the Star Game’ button, the Game is started in Screen 3.
3. Screen 3 displays the two players with their names, number of cards in hand, Main Deck (to draw new cards), Discard Deck (to display played cards by other players), and a Play Card button after selecting a specific card from the Hand.

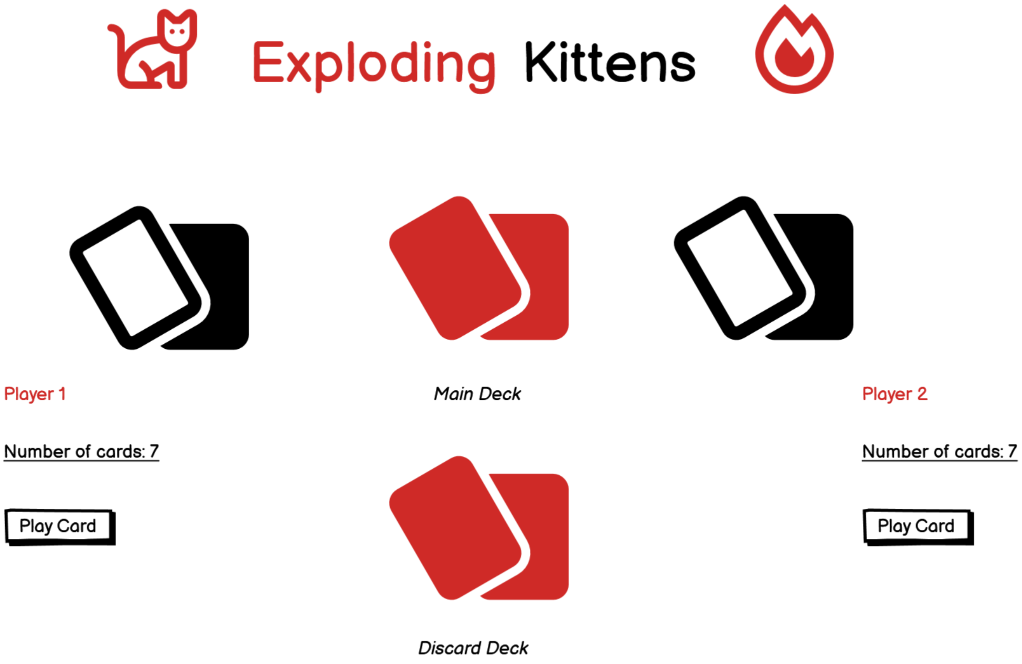
Screen 1 :



Screen 2:

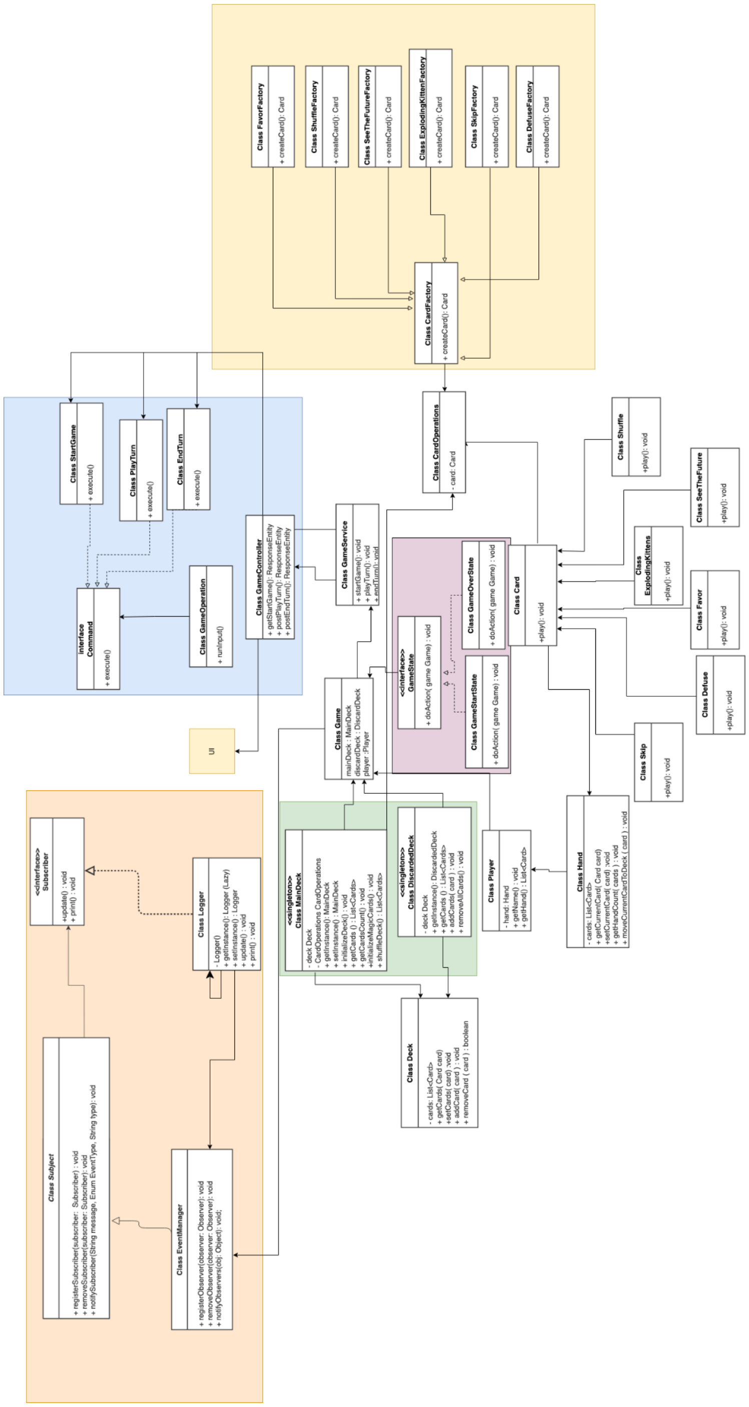


Screen 3:



UML Class Diagram & Pattern Use

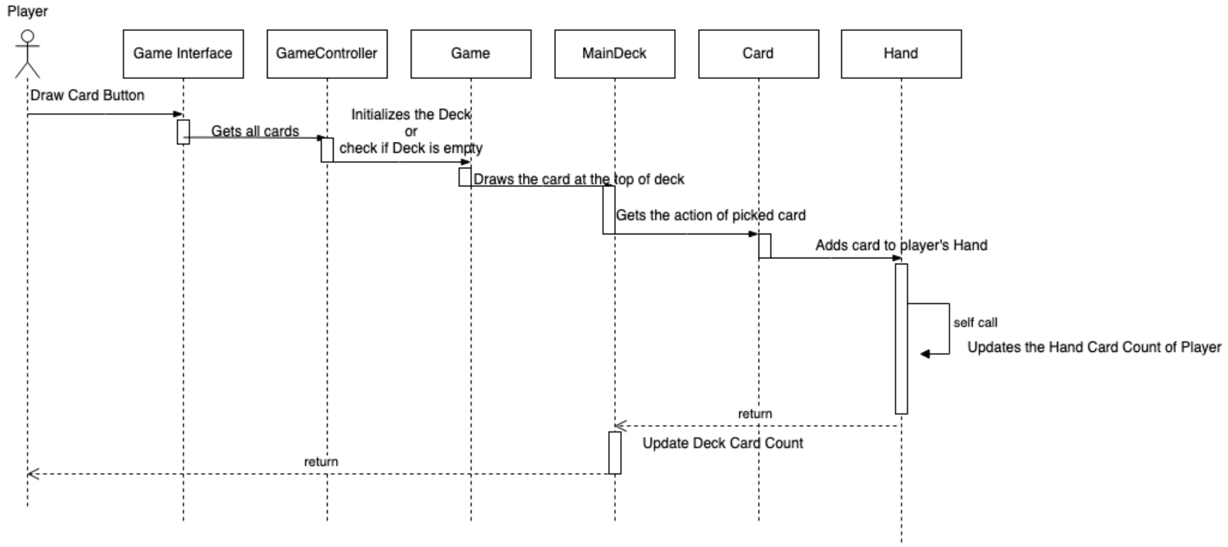
- Design Patterns used
  a. Command Pattern (Blue)
  b. Factory Pattern (Yellow)
  c. State Pattern (Purple)
  d. Observer Pattern (Orange)
  e. Singleton Pattern (Green)



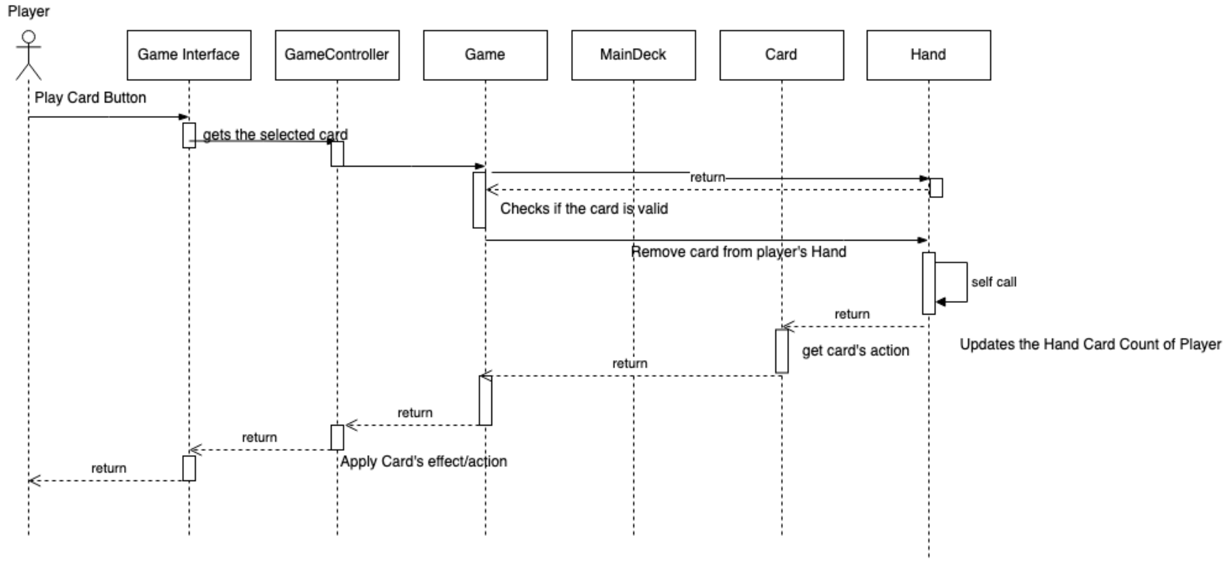
User Interactions/UML Sequence Diagrams

Three primary interactions that a user may have with an Exploding Kittens game for two players:

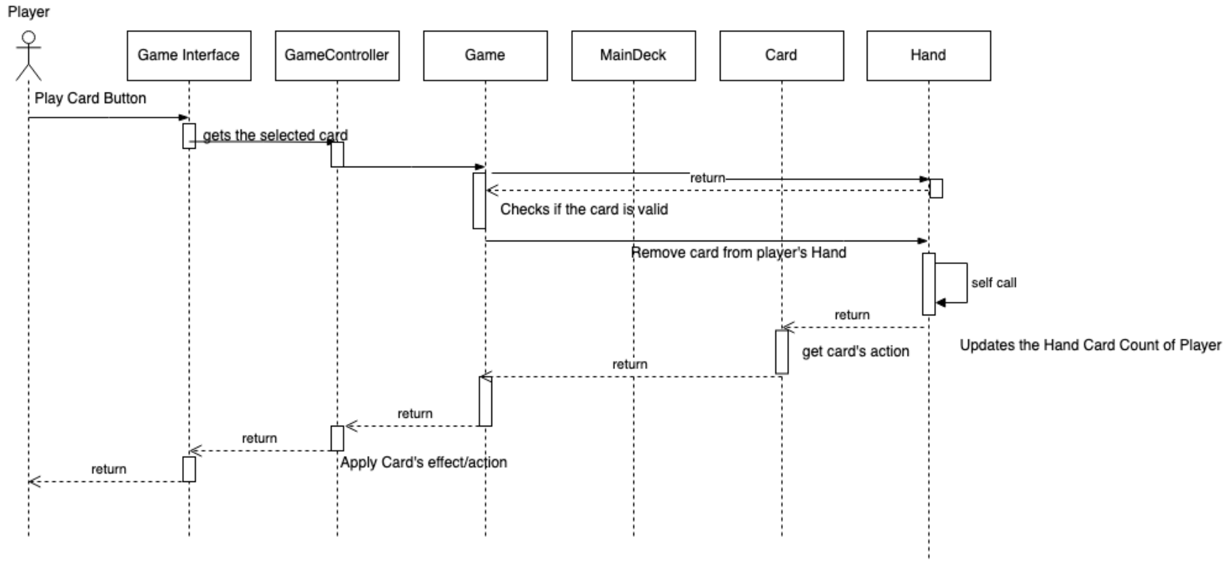
<u>Draw a Card:</u> The user selects the 'Draw' option from the game interface, and the game controller responds by allowing the user to draw a card from the draw pile. The player's hand will be updated with this new picked card from the deck. This interaction involves the player, game controller, main deck, card and the hand object.



<u>Play a Card:</u> The user selects a card from their hand and chooses the 'Play Card' option from the game interface. The game controller then checks if the card is valid to play and updates the game state accordingly. This interaction involves the player, game controller, main deck, card and the player's hand object.



Defuse an Exploding Kitten: When the user draws an exploding kitten card, they have the option to defuse it if they have a defuse card in their hand. The user selects the 'Defuse' card from the game interface, and the game controller checks if the player has a defuse card. If the player has a defuse card, it is played, and the exploding kitten card is then placed back in the main deck. This interaction involves the player, game controller, game, main deck, player's hand object, and discard deck object.



Player
Game Interface
GameController
Game
MainDeck
Card
Hand
Play Card Button
gets the selected card
return
Checks if the card is valid
Remove card from player's Hand
self call
return
Updates the Hand Card Count of Player
get card's action
return
return
return
Apply Card's effect/action
return