**Initiating a game**

Iteration: 1
Primary Actor: Player
Goal in Context: Starting a game with another player
Preconditions: The player is logged in
Trigger: The player selects a game and clicks play

Scenario:

1. The game matches another player of similar skill level
2. An object of the game selected is created
3. The UI displays the game board to both players

Postcondition: the selected game is ready to play for both players
Exceptions:

1. The player is matched to themself

Priority: high
When available: initial release
Frequency of Use: high, once every game
Channel to actor: network UI update
Secondary Actors: game host server
Channel to secondary Actors: network Open
Issues:

**Game Data fetch by Client-side computer**

Use Case:
Iteration: 1
Primary Actor: Client-side UI
Goal in Context: allow the client to retrieve necessary data for gameplay
Preconditions: the player is logged in and in a game
Trigger: the player initiates a game, the game is refreshed

Scenario:

1. The computer fetches the game data
2. The UI reflects any changes

Postcondition: the user can see the updated game board
Exceptions:

1. The client is attempting to fetch unauthorized data

Priority: high
When available: 1
Frequency of Use: high, every time a game is refreshed
Channel to actor: network
Secondary Actors: end user, game logic server
Channel to secondary Actors: UI display, network
Open Issues: What should the refresh rate be?

**Dropping a Connect Four Checker**

Iteration: 1
Primary Actor: Active player
Goal in Context: The active player dropping their checker, completing their turn
Preconditions: A valid game is active
Triggers: The beginning of the game and the end of a turn that doesn't result in the end of the game.

Scenario:

1. A game is initiated and validated
2. Player 1 makes the first move

Postcondition: the move reflected in the UI and displayed to both players
Exceptions:

1. The game is not valid, for example, a player is matched with themself.
2. A player quits or disconnects, ending the game.

Priority: high, this is the core of the Connect Four gameplay.
When available: first release.
Frequency of Use: high, used multiple times every game.
Channel to actor: network UI update
Secondary Actors: Passive player, game host server
Channel to secondary Actors: network Open
Issues:

**Moving a Checker Checkers**

Iteration: 1

Primary Actor: Active Player

Goal in Context: The active player mover their piece to a different square in accordance with the rules of the game

Preconditions: A valid game is active

Trigger: The beginning of the game and the end of a turn that doesn't result in the end of the game.

Scenario:

1. A game is initiated and validated
2. Player 1 makes the first move

Postcondition: the move reflected in the UI and displayed to both players

Exceptions:

1. The game is not valid, for example, a player is matched with themself.
2. A player quits or disconnects, ending the game.
3. Then checker has no legal moves

Priority: high, this is the core of the Checkers gameplay.

When available: first release.

Frequency of Use: high, used multiple times every game.

Channel to actor: network UI update

Secondary Actors: Passive player, game host server

Channel to secondary Actors: network  Open

Issues:

**Promoting to a King Checkers**

Iteration: 1
Primary Actor: Active player
Goal in Context: The active player promotes one of their men to the pieceType King
Preconditions: A valid game is active
Trigger: A man reaches the opposite end of the board

Scenario:

1. Active player moves a man to the opposite end row
2. Active player's piece is changed to a King

Postcondition: the move and promotion reflected in the UI and displayed to both players Exceptions:

1. The game is not valid, for example, a player is matched with themself.
2. A player quits or disconnects, ending the game.
3. The player has a forced capture

Priority: high, this is a core mechanic Checkers gameplay.
When available: first release.
Frequency of Use: high, used a few times a game. Channel to actor: network UI update
Secondary Actors: Passive player, game host server
Channel to secondary Actors: network Open
Issues:

**Ending a game**

Iteration: 1
Primary Actor: active player
Goal in Context: displaying to all players that the game is over, and if they lost, won or drew.
Preconditions: A game is active
Trigger: A wining, losing, or drawing move, or a player quits

Scenario:

1. The server detects that the game is over
2. The players' stats are updated
3. The UI displays to all players whether they won, drew or lost
4. The UI displays the option to leave the game

Postcondition: The game is over
Exceptions: The game will always end

Priority: high
When available: first release
Frequency of Use: High, used every game
Channel to actor: network
Secondary Actors: passive players, game logic server Channel
to secondary Actors: network Open Issues: