Use Case: Start New Game

Iteration: 1
Primary Actor: Player
Goal in context: Begin a new Connect 4 game session.
Preconditions: The system is powered on and ready to start a game.
Trigger: The player selects the "Start Game" option.

Scenario:

1. Player clicks the "Start Game" button.
2. The system initializes a new Connect 4 board (6x7 grid).
3. The system displays the game board on the screen.
4. The game sets Player 1 as the active player.

Post conditions: The game board is displayed, and Player 1 is prompted to make the first move.

Exceptions:

- The "Start Game" button is unresponsive.
- The system encounters an error during initialization.

Priority: High – Starting the game is essential for gameplay and user experience.
When available: Within 1 sprint (first iteration).
Frequency of use: Once per game session.
Channel to actor: Interaction through a physical button or a mouse click.
Secondary actors: N/A
Channel to secondary actors: N/A
Open issues: N/A

Use Case: Player Makes a Move

Iteration: 1
Primary Actor: Player
Goal in context: Allow the player to make a valid move by placing a piece on the board.
Preconditions: The game is initialized, and it is the player's turn.
Trigger: The active player decides to make a move.

Scenario:

1. The system prompts the player to select a column.
2. The player chooses a column.
3. The system validates if the column is not full.
4. The system places the player's piece at the lowest available row in the selected column.
5. The system updates and displays the new state of the board.

Post conditions: The board is updated, and the game passes the turn to the next player.

Exceptions:

- The selected column is full.
- The player inputs an invalid column number.

Priority: High – Central to the game's core functionality.
When available: Within 1 sprint (first iteration).
Frequency of use: Multiple times per game session (each turn).
Channel to actor: Player interaction through a mouse or touchscreen.
Secondary actors: N/A
Channel to secondary actors: N/A
Open issues: N/A

Use Case: Check for Win Condition

Iteration: 1
Primary Actor: System
Goal in context: Verify if a move results in a winning condition for the current player.
Preconditions: A player has made a move, and the game board is updated.
Trigger: After the player’s piece is placed on the board.

Scenario:

1. The system checks for horizontal, vertical, and diagonal (both forward and backslash) win conditions.
2. If a win condition is detected, the system announces the winner.
3. If no win is detected, the game continues to the next player's turn.

Post conditions: The game ends if a win is detected otherwise gameplay continues.

Exceptions:

- An error in the win-checking algorithm causes false detection.

Priority: High – Essential for determining the game outcome.
When available: Within 1 sprint (first iteration).
Frequency of use: After every move made by a player.
Channel to actor: N/A
Secondary actors: N/A
Channel to secondary actors: N/A
Open issues: Optimization of win-checking logic for performance.

Use Case: End Game (Win or Draw)

Iteration: 1
Primary Actor: System
Goal in context: Conclude the game session when a win or draw condition is met.
Preconditions: The system has detected a win or the board is full with no available moves.
Trigger: A win or draw is detected after a move.

Scenario:

1. The system checks the game board for winning or draw conditions.
2. If a win is detected, the system announces the winner.
3. If the board is full, the system announces a draw.
4. The system provides an option to start a new game or exit.

Post conditions: The game session is concluded, and a new game can be started or the system exits.

Exceptions:

- Display issues during the result announcement.

Priority: High – Important for finalizing the game session.
When available: Within 1 sprint (first iteration).
Frequency of use: Once per game session.
Channel to actor: Displayed on a screen or monitor.
Secondary actors: N/A
Channel to secondary actors: N/A
Open issues: Handling unexpected board state during result processing.

Use Case: Validate Move

Iteration: 1
Primary Actor: System
Goal in context: Ensure that a player's selected move is valid before placing the piece.
Preconditions: The player has chosen a column.
Trigger: The player attempts to place a piece in a selected column.

Scenario:

1. The system receives the player's selected column.
2. The system checks if the column is not full.
3. If valid, the system allows the move; if not, it prompts the player to select a different column.

Post conditions: The move is either allowed or rejected based on validation.

Exceptions:

- Column is full, preventing piece placement.

Priority: High – Essential for maintaining game integrity.
When available: Within 1 sprint (first iteration).
Frequency of use: After each player input.
Channel to actor: Interaction through a mouse or touchscreen.
Secondary actors: N/A
Channel to secondary actors: N/A
Open issues: N/A

Use Case: Switch Player Turns

Iteration: 1
Primary Actor: System
Goal in context: Manage and switch turns between players after each move.
Preconditions: The game is in progress with at least one move made.
Trigger: A player successfully places a piece.

Scenario:

1. The system checks which player made the last move.
2. The system switches the current player to the next player.
3. The system prompts the new current player to make their move.

Post conditions: The current player is updated, and the new player is informed it is their turn.

Exceptions:

- There is an error in determining the current player.

Priority: High – Necessary for proper game flow and fairness.
When available: Within 1 sprint (first iteration).
Frequency of use: After every move made by a player.
Channel to actor: N/A
Secondary actors: N/A
Channel to secondary actors: N/A
Open issues: N/A

Use Case: Display Game Board

Iteration: 1
Primary Actor: System
Goal in context: Present the current state of the Connect 4 board visually to the players.
Preconditions: The game has started, and there have been moves made.
Trigger: The game starts and after each player's turn.

Scenario:

1. The system generates a visual representation of the current game board.
2. The system prints the board layout with column numbers and piece placements.
3. The system updates the display after each move.

Post conditions: The players can see the current state of the game board at all times.

Exceptions:

- Display errors prevent the board from being shown.

Priority: High – Important for player engagement and understanding of the game state.
When available: Within 1 sprint (first iteration).
Frequency of use: After every move and at game start.
Channel to actor: Visual output on the screen.
Secondary actors: N/A
Channel to secondary actors: N/A
Open issues: N/A

Use Case: Display Game Result

Iteration: 1
Primary Actor: System
Goal in context: Announce the result of the game (win/draw) to the players.
Preconditions: The game has concluded due to a win or draw condition.
Trigger: A player has won or the board is full.

Scenario:

1. The system determines the result of the game.
2. The system displays the result (e.g., "Player 1 wins!" or "It's a draw!").
3. The system offers options to restart the game or exit.

Post conditions: The players are informed of the game result, and options for the next steps are presented.

Exceptions:

- An error occurs while displaying the result.

Priority: High – Important for player satisfaction and next steps.
When available: Within 1 sprint (first iteration).
Frequency of use: Once per game session, at the end of the game.
Channel to actor: Visual output on the screen.
Secondary actors: N/A
Channel to secondary actors: N/A
Open issues: N/A