Use Case Description - CHECKER

Use Case: Start a New Game

Iteration: 1

Primary Actor: Player

Goal in Context: The player initiates a new game session and invites another player or selects an AI opponent.

Preconditions: The player must be logged in and have access to the game selection interface.

Trigger: The player selects "Start New Game" from the main menu.

Scenario:

1. The player selects a board game from the available list.
2. The system displays matchmaking options (invite a friend, random matchmaking, or AI opponent).
3. The player selects an opponent.
4. The system initializes the game board and displays it to both players.
5. The game begins with the first player's turn.

Postconditions: The game session is successfully created, and players are connected.

Exceptions:

1. If matchmaking fails, the system notifies the player and offers retry options.
   Priority: High

When Available: Immediately after login

Frequency of Use: High

Channel to Actor: Game interface (GUI)

Secondary Actors: Opponent player or AI

Channel to Secondary Actors: Multiplayer server

Open Issues: None

Use case: Single Player Mode

Iteration: 1

Primary Actor: Player

Goal in Context: The player plays a game against an AI opponent instead of a human player.

Preconditions: The player must be logged in and have access to the game selection interface.

Trigger: The player selects the “Single Player” option from the game menu.

Scenario:

1. The player selects “Single Player” mode from the menu.
2. The system prompts the player to select a board game.
3. The system assigns an AI opponent with a configurable difficulty level.
4. The game board is initialized, and the player takes the first turn.
5. The AI opponent responds to moves based on the selected difficulty.
6. The game continues until a win/loss/draw condition is met.
7. The system updates the player's game history and records the match outcome.

Postconditions: The single-player session is successfully completed, and results are recorded.

Exceptions:

1. If the AI logic fails, the system displays an error message and resets the game.
2. If the player exits mid-game, the session is terminated.

   Priority: High

When Available: After login

Frequency of Use: High

Channel to Actor: Game interface

Secondary Actors: AI opponent

Channel to Secondary Actors: None

Open Issues: None

Use case: Multiplayer Mode

Iteration: 1

Primary Actor: Player

Goal in Context: The player joins or hosts an online multiplayer game.

Preconditions: The player must be logged in and connected to the multiplayer server.

Trigger: The player selects the multiplayer mode option.

Scenario:

1. The player selects “Multiplayer Mode.”
2. The system displays available matchmaking options.
3. The player selects a matchmaking option (quick match, invite friend, ranked game, etc.).
4. The system connects the player to an opponent.
5. The game begins.

Postconditions: The multiplayer session starts successfully.

Exceptions:

1. If no opponent is found, the system retries matchmaking or offers AI mode.
2. If the connection is lost, the system notifies the player.

Priority: High

When Available: After login

Frequency of Use: High

Channel to Actor: Game interface

Secondary Actors: Opponent player

Channel to Secondary Actors: Multiplayer server

Open Issues: None

Use case: Validate Player Turn

Iteration: 1

Primary Actor: System

Goal in context: Ensure that only the player whose turn it is can make a move.

Preconditions: A game must be in progress.

Trigger: A player attempts to make a move.

Scenario:

1. The player selects a piece to move.
2. The system checks whose turn it is.
3. If it is the player's turn, the move is validated.
4. If it is not the player's turn, the system prevents the move and notifies the player.

Post conditions: Moves are only made by the correct player.

Exceptions:

1. If turn tracking fails, the system logs an error.

   Priority: High

When available: During gameplay

Frequency of use: High

Channel to actor: Game interface

Secondary actors: Opponent player

Channel to secondary actors: Multiplayer server (if applicable)

Open issues: None

Use case: Detect Invalid Piece Selection

Iteration: 1

Primary Actor: System

Goal in context: Ensure players select only valid game pieces.

Preconditions: A game must be in progress.

Trigger: The player attempts to select a piece.

Scenario:

1. The player clicks on a piece.
2. The system checks if the piece belongs to the player and is eligible for movement.
3. If valid, the piece is selected.
4. If invalid, the system prevents selection and notifies the player.

Post conditions: Only valid pieces can be selected.

Exceptions:

1. If the system fails to validate, an error message is logged.

   Priority: High

When available: During gameplay

Frequency of use: High

Channel to actor: Game interface

Secondary actors: None

Channel to secondary actors: None

Open issues: None

Use Case: Check for Legal Moves

Iteration: 1

Primary Actor: System

Goal in Context: The system checks if the player has legal moves available.

Preconditions: The game is in progress, and it is the player's turn.

Trigger: The player selects a piece.

Scenario:

1. The player selects a piece.
2. The system calculates and highlights legal moves.
3. If no legal moves exist, the system notifies the player.

Postconditions: The system displays valid moves or notifies the player of no options.

Exceptions:

1. If the game logic fails, the system logs an error.

   Priority: High

   When Available: During gameplay

   Frequency of Use: High

   Channel to Actor: Game interface

   Secondary Actors: None

   Channel to Secondary Actors: None

   Open Issues: None

Use case: Enforce Time Limit for Moves (Timed Mode)

Iteration: 1

Primary Actor: System

Goal in context: Ensure players make moves within a specified time limit.

Preconditions: Timed mode must be enabled.

Trigger: A player’s turn begins.

Scenario:

1. The system starts a countdown timer when a player's turn begins.
2. If the player makes a move within the time limit, the game continues normally.
3. If time expires, the system end the player's turn

Post conditions: The game progresses within the time constraints.

Exceptions:

1. If the timer malfunctions, the system logs an error.

   Priority: Medium

When available: During gameplay in timed mode

Frequency of use: Medium

Channel to actor: Game interface

Secondary actors: Opponent player

Channel to secondary actors: Multiplayer server (if applicable)

Open issues: None

Use case: Enforce Jump Rule (Forced Capture)

Iteration: 1

Primary Actor: System

Goal in context: Ensure that mandatory captures are enforced in games

Preconditions: A game must be in progress.

Trigger: The player attempts to move when a capture is available.

Scenario:

1. The player selects a move.
2. The system checks if a capture is available.
3. If a capture is required, the system forces the player to execute it by disabling all non-capture moves.
4. If no captures exist, the player is allowed to move normally.

Post conditions: Mandatory captures are enforced.

Exceptions:

1. If capture detection fails, the system logs an error.
   Priority: High

When available: During gameplay

Frequency of use: Medium

Channel to actor: Game interface

Secondary actors: Opponent player

Channel to secondary actors: Multiplayer server (if applicable)

Open issues: None

Use Case: Make a Move

Iteration: 1

Primary Actor: Player

Goal in Context: The player performs a move during their turn in the game.

Preconditions: The game must be active, and it must be the player's turn.

Trigger: The player selects a piece and moves it to a valid location.

Scenario:

1. The system highlights possible moves for the selected piece.
2. The player chooses a valid move.
3. The system validates the move and updates the game board.
4. The system records the move in the game history.

Postconditions: The move is registered, and the turn shifts to the opponent.

Exceptions:

1. If the move is illegal, the system notifies the player and prevents it.

   Priority: High

   When Available: During a game session

   Frequency of Use: High

   Channel to Actor: Game interface

   Secondary Actors: Opponent player or AI

   Channel to Secondary Actors: Multiplayer server

   Open Issues: None

Use case: Undo Move

Iteration: 1

Primary Actor: Player

Goal in context: The player undoes their previous move.

Preconditions: The game must allow undo functionality, and it must be the player's turn.

Trigger: The player selects the “Undo Move” option.

Scenario:

1. The player requests to undo their last move.
2. The system verifies if undoing is permitted.
3. If allowed, the system reverts the game state to before the move.
4. The player’s turn is restored.

Post conditions: The previous move is undone, and the player can make a new move.

Exceptions:

1. If undo is not allowed (e.g., multiplayer restrictions), the system notifies the player.
   Priority: Medium

When available: During gameplay

Frequency of use: Low to medium

Channel to actor: Game interface

Secondary actors: Opponent player (if applicable)

Channel to secondary actors: Multiplayer server (if applicable)

Open issues: None

Use Case: Capture an Opponent's Piece

Iteration: 1

Primary Actor: Player

Goal in Context: The player captures an opponent's piece according to the game rules.

Preconditions: The player must make a legal move that results in capturing an opponent's piece.

Trigger: The player selects a piece and moves it to a position occupied by an opponent's piece.

Scenario:

1. The player selects a piece and moves it onto an opponent's piece.
2. The system verifies if the move is valid.
3. If valid, the opponent's piece is removed from the board.
4. The system updates the game board and records the action.

Postconditions: The opponent's piece is removed, and the game continues.

Exceptions:

1. If the move is illegal, the system prevents the action.

   Priority: High

   When Available: During a game session

   Frequency of Use: Medium

   Channel to Actor: Game interface

   Secondary Actors: Opponent player or AI

   Channel to Secondary Actors: Multiplayer server

   Open Issues: None

Use Case: King a Piece

Iteration: 1

Primary Actor: Player

Goal in Context: The player moves a piece to the opponent’s back row, promoting it to a "King".

Preconditions: The game must support piece promotion.

Trigger: The player moves a piece to the last row on the opponent’s side.

Scenario:

1. The player moves a piece to the last row.
2. The system checks if the game supports promotions.
3. If valid, the piece is upgraded to a "King" with new movement capabilities.
4. The system records the promotion event.

Postconditions: The piece is promoted, and gameplay continues.

Exceptions:

1. If the move is invalid, the system prevents it.

   Priority: Medium

   When Available: During gameplay

   Frequency of Use: Low

   Channel to Actor: Game interface

   Secondary Actors: Opponent player or AI

   Channel to Secondary Actors: Multiplayer server

   Open Issues: None

Use Case: Detect End of Game (Win or Draw)

Iteration: 1

Primary Actor: System

Goal in Context: The system determines when a game is won, lost, or ends in a draw.

Preconditions: The game is in progress.

Trigger: A player meets the winning or draw condition.

Scenario:

1. The system continuously checks for win/loss conditions.
2. If a player meets the win condition, the game ends.
3. If a draw condition is met, the system declares a draw.
4. The system updates the leaderboard and player stats.

Postconditions: The game session ends, and results are recorded.

Exceptions: None

Priority: High

When Available: End of gameplay

Frequency of Use: Once per game

Channel to Actor: Game interface

Secondary Actors: Opponent player or AI

Channel to Secondary Actors: Multiplayer server

Open Issues: None