Use case: Click Square

Iteration: 1

Primary Actor: The Players

Goal in context: To receive input from player

Preconditions: The chess game has started

Trigger: Player clicks square on screen with mouse

Scenario:

1. A player wants to select one of their pieces
2. A player wants to deselect their currently selected piece
3. A player wants to move their currently selected piece

Post conditions: The GUI checks who the current player is and decides whether to accept the click based on whether it was made on the active player's screen or not

Exceptions:

1. If it isn't the turn of the player that clicked the square, the click will be ignored.

Priority: High. It is necessary for playing the game.

When available: 1st iteration

Frequency of use: High (ie. every time a player decides to select, deselect, or move a piece in a chess game)

Channel to actor: Window on personal device screen that corresponds to the player's assigned colour (i.e. white or black)

Secondary actors: n/a

Channel to secondary actors: n/a

Open issues: Should the inactive player's screen be designed to not listen to clicks or should the clicks be heard but lead to no action?

Use case: Process Click

Iteration: 1

Primary Actor: The active player

Goal in context: To perform the action the active player instructed the game to do when they clicked a square.

Preconditions: The active player clicked a square

Trigger: The active player clicked a square.

Scenario:

1. The active player wants to select one of their pieces
2. The active player wants to deselect their currently selected piece
3. The active player wants to move their currently selected piece

Post conditions: Based on whether a square was already selected and whether the click was performed on a valid move destination, the necessary game state and screen presentation updates will be made.

Exceptions: Hardware issues.

Priority: High. Necessary for playing the game.

When available: 1st iteration.

Frequency of use: High (i.e. every time the active player decides to select, deselect, or move a piece in a chess game)

Channel to actor: Window on personal device screen that corresponds to the player's assigned colour (i.e. white or black)

Secondary actors: n/a

Channel to secondary actors: n/a

Open issues: Should the GUI call different methods from the Game Logic layer based on whether the click was performed on a valid move option and then update the screen in a targeted manner that is enabled by that knowledge or should there be a generic processClick() function that is executed followed by a generic process to update the screen?

Use case: Select Square

Iteration: 1

Primary Actor: Active player

Goal in context: To select a square that contains one of the active player's pieces and present the associated move options, as well define the piece to be moved as a set up for making a move.

Preconditions: It was determined that the square that the active player clicked on contains one of their pieces.

Trigger: The active player clicks on a square that contains one of their pieces

Scenario:

1. The active player wants to select one of their pieces in order to move it.
2. The active player wants to select one of their pieces in order be presented with the possible moves that can be made in order to be aided in deciding what to do.

Post conditions: The square that was clicked on will be selected and the corresponding move options will be stored in the game logic as well as be visible on the screen. If the active player clicks on a square that represents a valid move option the currently selected piece will be moved.

Exceptions: Hardware issues.

Priority: High. Necessary for playing the game.

When available: 1st iteration.

Frequency of use: High (i.e. every time the active player in a chess game clicks on a square that contains one of their pieces)

Channel to actor: Window on personal device screen that corresponds to the player's assigned colour (i.e. white or black)

Secondary actors: n/a

Channel to secondary actors: n/a

Open issues: Since the GUI will present the move options, it could theoretically determine whether a clicked square is a valid move option based on whether it is marked as a move option, so should the GUI take advantage of this possibility or instead rely on the Game Logic to determine if a clicked square is a valid move option?

Use case: Deselect Previously Selected Square

Iteration: 1

Primary Actor: The active player

Goal in context: To deselect the currently selected square so as to remove the distraction of being presented with the possible moves corresponding to a particular piece on the board and/or avoid the risk of accidentally making a move by clicking on a valid move option.

Preconditions: It was determined that the square that the active player clicked on is neither a valid move option nor a square that contains one of the player's pieces.

Trigger: The active player clicks on a square that both isn't a valid move option and doesn't contain one of their pieces.

Scenario:

1. The active player wants deselect the currently selected square without selecting another square in order to remove the distraction of seeing the move options for a specific piece.
2. The active player wants to not have any piece selected in order to avoid accidentally making a move by clicking on a valid move option.

Post conditions: There will no longer be a selected piece and no move options. The only action the player can take to continue the game is to select a piece.

Exceptions: Hardware issues.

Priority: Low. Not necessary for playing the game, but slightly improves the experience of the players.

When available: 1st iteration.

Frequency of use: Medium to High (i.e. every time the active player decides to remove all selections from their screen during a chess game)

Channel to actor: Window on personal device screen that corresponds to the player's assigned colour (i.e. white or black)

Secondary actors: n/a

Channel to secondary actors: n/a

Open issues: Should it instead be the case that when a player clicks on a square that is neither a valid move options nor a square that contains one of their pieces, that this action causes the square to be selected instead of just deselecting the currently selected square.

Use case: Make Move

Iteration: 1

Primary Actor: The active player

Goal in context: To move the currently selected piece to the square that is clicked on

Preconditions: A piece that is capable of moving is selected on the active player's screen and it is determined that the square that the active player clicked on is a valid move option.

Trigger: The active player clicks on a square that constitutes a valid move option for the currently selected piece.

Scenario:

1. The active player has decided what move they want to make and wants to instruct the game to make that move.

Post conditions: The move will have been made and the new game state will be stored in the Game Logic and presented on both of the player's screens. The other player will be the active player.

Exceptions: Hardware issues.

Priority: High. Necessary for playing the game.

When available: 1st iteration.

Frequency of use: High (i.e. every time a player makes a move in a chess game)

Channel to actor: Window on personal device screen that corresponds to the player's assigned colour (i.e. white or black)

Secondary actors: n/a

Channel to secondary actors: n/a

Open issues: Since the GUI will present the move options, it could theoretically determine whether a clicked square is a valid move option based on whether it is marked as a move option, so should the GUI take advantage of this possibility or instead rely on the Game Logic to determine if a clicked square is a valid move option?

Use case: Update Active Player's Screen

Iteration: 1

Primary Actor: The active player

Goal in context: To update the active player's screen to present the new state of selection and corresponding move options.

Preconditions: The active player clicks on a square that either results in a new square selection to be made or an absence of a square selection to be established.

Trigger: The active player clicks on a square that either results in a new square selection to be made or an absence of a square selection to be established.

Scenario:

1. The active player has changed or undone the current square selection and so the new state of selection and move options must be presented to the active player.

Post conditions: The active player will be able to see the new state of selection and move options.

Exceptions: Hardware Issues

Priority: High. Very important to show the active player which of their pieces is currently selected

When available: 1st iteration

Frequency of use: High (i.e. every time a player changes the state of their selection during a chess game)

Channel to actor: Window on personal device screen that corresponds to the player's assigned colour (i.e. white or black)

Secondary actors: n/a

Channel to secondary actors: n/a

Open issues: What specific method will we implement to mark squares as valid move options on the screen?

Use case: Update Both Player Screens

Iteration: 1

Primary Actor: the active player

Goal in context: To update both player's screens to show the new state of the game after a move is made.

Preconditions: The active player has made a move and the screens have not yet been updated to the new state of the game.

Trigger: The active player makes a move.

Scenario:

1. A player has made a move and so both player's need to be presented with the new state of the game.

Post conditions: The new active player is able to see the current state of the game and make selections and perform the next move.

Exceptions: Hardware issues.

Priority: High. It is necessary for the player's to know the state of the game.

When available: 1st iteration

Frequency of use: High (i.e. every time a move is made by a player during a chess game)

Channel to actor: Window on personal device screen that corresponds to the player's assigned colour (i.e. white or black)

Secondary actors: n/a

Channel to secondary actors: n/a

Open issues: Should we follow through with the current plan of having two separate windows for each player?

Use case: Switch Active Player

Iteration: 1

Primary Actor: The active player

Goal in context: To switch which player is considered the active player after a move is made.

Preconditions: A player has made a move and the resulting state of that player's captured pieces has been recorded in the Game Logic

Trigger: A move is made by the active player.

Scenario:

1. A player makes a move which results in it becoming the other player's turn

Post conditions: The player that was previously considered the inactive player will now be considered the active player and be able to select pieces on their screen and make a move, while the player that was previously considered the active player will be considered the inactive player, and will no longer be able to interact with the chess board on their screen until the other player makes a move.

Exceptions: Hardware issues.

Priority: High. Necessary for playing the game.

When available: 1st iteration.

Frequency of use: High. (i.e. every time a player makes a move in a chess game)

Channel to actor: Window on personal device screen that corresponds to the player's assigned colour (i.e. white or black)

Secondary actors: n/a

Channel to secondary actors: n/a

Open issues: What is the best way of implementing a turn based game in a GUI and, as asked above, will we follow through with the current plan of having two separate windows for each player?