Use case description: Initiating a Game

Primary actor:

- User

Stakeholders and interests:

- User
    - Start game by following given instructions
    - Choose level of difficulty
    - Choose number of players

Preconditions:

- User must use a system that meets the requirements to run the game
- User must properly follow the instructions on how to start the game

Postconditions:

- Select number of players and level of difficulty
- The board is displayed

Main success scenario:

1. User opens the game by following the instructions
2. The systems displays the options to “Start new game”, “Resume game” and “Exit game”
3. User selects Start new game
4. The system retrieves the list of how many players can play and displays it to the user
5. The system allows the user to select the number of players
6. User selects an option and the system records that
7. The system retrieves the list of difficulty settings and displays it to the user
8. User selects a level of difficulty and the system records that
9. The user is also given the option of color deficiency mode
10. If selected the system records it and displays the game board to the user

Alternate flows:

Resume game:

- The user selects Resume game
- The user retrieves last stored positions of the blocks on the board and displays it to the user

Exit game:

- The user selects Exit game
- The system terminates the game

Exceptions:

- If at any time the system is unable to retrieve, record or provide details while starting the game, the system informs the user of the problem and the use case ends

Special requirements:

- Colors of pieces must (or be able to) provide for the visually impaired (eg - color blindness)

Open issues:

- Make sure the system records and retrieves data for any selection made while starting the game

Use Case Description: Take a turn

Primary actor:

- Player

Stakeholders and interest:

- Player: Wants to take a turn by moving/rotating one of their pieces and placing it on the board
- Other players (AI): Wants to know what spaces are available to lay a piece and to lay that piece according to the board layout and difficulty

Preconditions:

- Player must select the level of difficulty, number of players and the board must be loaded

Postconditions:

- The piece was placed on the board or the board is full and the winner is determined

Main success scenario:

1. The system tells the player it’s their turn
2. The player selects a piece to be placed on the board
3. The system allows the player to rotate and flip the piece, then to select where the piece will be placed on the board
4. The player selects orientation and selects where the piece will be placed on the board

5. The system validates the placement of the piece
6. The system places the pieces on the board
7. The system informs the player their turn is over and the player whose turn it is next

Alternative flows:

Invalid placement of the piece

1. The system checks to see if there are any valid placements left for the piece in the players hand and the other pieces yet to be placed
2. The system informs the player that the placement is invalid, then allows the player to re-orientate and selects a new placement
3. Flow resumes at Main success scenario step 5.

No more place available for any pieces

1. If there are no players left that are able to place any more pieces, the winner is determined and the system ends the game
2. Use case ends

Exceptions:

- No more moves can be made by any player ending the game

Special requirements:

- Colors of pieces must (or be able to) provide for the visually impaired (eg - color blindness)

Open issues:

- Make sure that there is no way for turns to overlap
- Figure out a way to select which player goes first randomly