Use Case Descriptions

1. **Set up a game**

- Primary Actor(s)

  Player(s) : User of the system who will open the system.

- Stakeholders and Interests

  Player(s) : Users will open the game and select from the menu for desired playing settings.
  AI : System set up non-human players to play with humans.
  Members : Will validate group members" code to open the game.
  Marker : Will satisfy marker's requirement for this process.

- Preconditions

  The system is a runnable application where user will click to open it and select from the menu to set up a desire game board of selected players.

- Postconditions

  A desired gameplay board will be set up for users to play.

- Main Success Scenario
  1) The user clicks on the application to open the game system.

2) The system provides the user opportunity to select a new game, previous game or exit the use case.
3) The user selects a new game.
4) The system provides the opportunity to select from two player or four player settings.
5) The user selects two players setting.
6) The system provides the opportunity to select player as human or AI.
7) The user selects players as human or human and AI.
8) The system set up a game board window with two players.

- Alternative Flows

Alt 1: *User selects previous game.*

1) After first two steps above, the user selects previous game.
2) The system loads the previous game window with saved action events.

Alt 2: *User selects four player settings.*

1) After first three steps, the user selects four player settings.
2) The system provides the opportunity to select players as human or AI.
3) The user selects players as all humans or humans and Ais.
4) The system set up a game board window with four players.

Alt 3: *User exits use case.*

1) At any of user steps above the user can elect to exit the user case.

- Exceptions

    - The system does not open after the user clicks on the application.
    - The system can not load the previous saved game.
    - The system can not set up the board as per instruction.

- Special Requirements

  The application must show the main window providing all the options as new game, previous game and exit game in an interactive way. The use case will set up a board as soon as all the options are selected.

- Open Issues

  If the user tries to open a previous game which was not recorded to the system then that function will not work show an error. The user can not select all players as AI; One human player must be selected.

2. **Take A Turn**

- Primary Actor(s)

  Player(s) : User (human or non-human) moves their pawn to a particular direction.

- Stakeholders and Interests

  Player(s) : User will be able to move their pawns to play the game.
  AI : Non-human user will be able to move its pawns.
  Members : Will validate group members" code to pawn movement.
  Marker : Will satisfy marker's requirement for this process.

- Preconditions

  A game board is set up with required number of players. The user must select the use case "take a turn" to move their pawn over the use case "set up a wall". The user must wait for their turn to perform this use case.

- Postconditions

  Player's pawn will move to the desired direction according to the game rules.

- Main Success Scenario

1) The system identifies the next players turn and provide the opportunity to select from the use cases “take a turn” or “set up a wall”.
2) The user selects “take a turn”
3) The system provide the opportunity to move the pawn to the right, left, forward, backward one block if the block is empty or select use case “jump over the player” if opponent is in the forward space or select use case “move in diagonal direction” if the opponent is in the forward space and there is a wall behind the opponent.
4) The user selects move forward.
5) The system moves the pawn forward one block.

- Alternative Flows

Alt 1: *User selects jump over the player*

1) After first three steps, the user selects the use case “jump over the player”
2) The system moves the pawn to second block from it’s initial block jumping over opponent’s pawn.

Alt 2: *User selects move in diagonal direction*

1) After first three steps, the user selects the use case “move in diagonal direction”
2) The system then provides the opportunity to select in which side of the opponent pawn the user wants to move given that the block is empty or reachable.
3) The user selects a side.

4) The system moves the pawn diagonally.

Alt 3: *User exits use case.*

2) At any of user steps above the user can elect to exit the user case.

- Exceptions

  - The pawn moves in any direction even the block is unreachable because of the walls.
  - The pawn can not jump over opponent's pawn.
  - The system prompts for the use case take a turn even if a players awn reach to its winning block.

- Special Requirements

The system will show the moving direction options if the block adjacent to the pawn is empty and there is no wall in between. The system will prompt a message showing the pawn can not move to that block if the player still tries to move there. Each player should wait for their turn to come.

- Open Issues

This use case's functionality depends on other use cases too. The proper design of that has not yet been constructed. Also the full functionality of this use case depends on the limitations of the source code. s