### 3.1.4 Game Design

R1.4.1 Each game will have a game id number and user id numbers for the players. **Priority 2**

R1.4.2 There will be a ChessBoard class which has the following properties:

R1.4.2.1 Board:ChessPiece[8][8] - An 8x8 array of ChessPieces. **Priority 1**

R1.4.2.2 state:ENUM - An enumerated state (white to move, black to move, white win, black win, draw). **Priority 1**

R1.4.2.3 validateMove(ChessMove):boolean - Checks that the user's move is legal. If the user enters an illegal move the game should display a message to the user, but otherwise do nothing. **Priority 1**

R1.4.2.4 update(ChessMove):void - Send the move to the server. **Priority 1**

R1.4.2.5 history:List<ChessMove> - The game history. **Priority 3**

R.1.4.3 There will be a ChessPiece abstract class which has the following properties:

R1.4.3.1 x:int, y:int - The piece's location on the board. **Priority 1**

R1.4.3.2 create():void - Instantiates a new piece on the board. **Priority 1**

R1.4.3.3 destroy():void - Removes a chess piece from the board. This occurs when an enemy piece moves to occupy the square this piece occupies. **Priority 1**

R1.4.3.4 generateMoves(ChessBoard):List<ChessMove> - For move validation. Each piece will be able to return a list of legal moves. **Priority 1**

R1.4.3.5 There will be individual chess piece classes (Rook, Bishop, Pawn, etc.) that extend the ChessPiece class. These classes will override the above functions where necessary (move generation) and provide additional functionality if required (promotion). **Priority 1**

R1.4.4 There will be a ChessMove class which has the following properties:

R1.4.4.1 id:int - The move number and id (ply). **Priority 1**

R1.4.4.2 fromX:int, fromY:int - The starting location on the board. **Priority 1**

R1.4.4.3 toX:int, toY:int - The ending location on the board. **Priority 1**