Peyton Wolf

CST-250

3/29/2025

Assignment 2

Mark Smithers

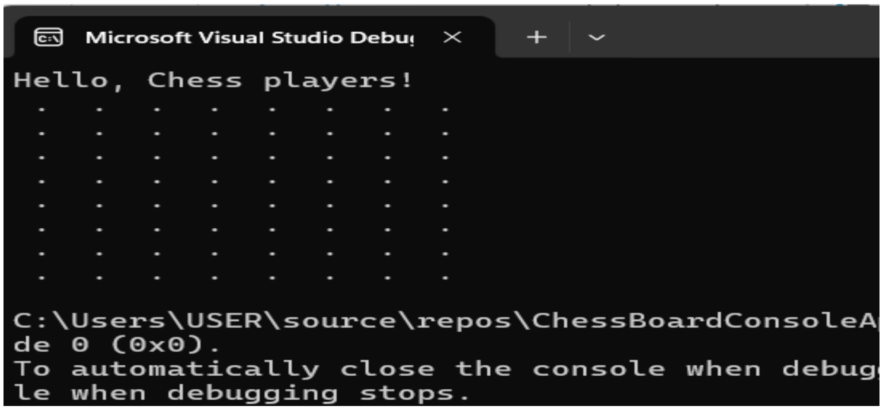
Loom Link

1. https://www.loom.com/share/c4e79434daa8411bb96b01bc40b6c2ac?sid=5f7379b1-57d9-4a9f-9613-d6c7b3bfca69
2. https://www.loom.com/share/3023ba28e98b4f118d16851a4ec75805?sid=0047dda8-ae63-4f9d-9c60-da0f09abb0e6

GitHub Link

https://github.com/KnoxHighStax/CST250/tree/main/Assignment2

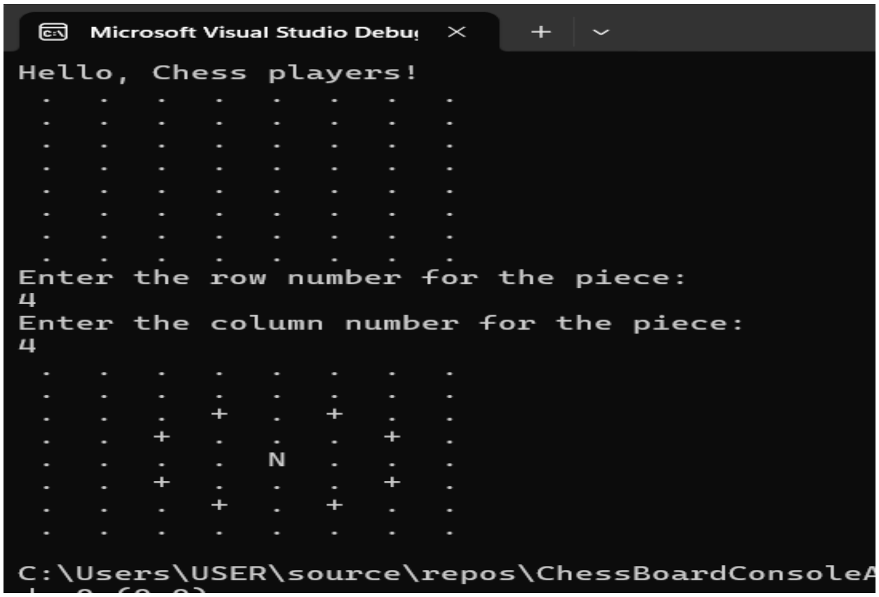
My First Chessboard



```
Hello, Chess players!
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .

C:\Users\USER\source\repos\ChessBoardConsoleA
de 0 (0x0).
To automatically close the console when debug
le when debugging stops.
```

In this screenshot you will see a demonstration of printing out my first chessboard through the console. We simply just have a greeting for the user and a chessboard being displayed.

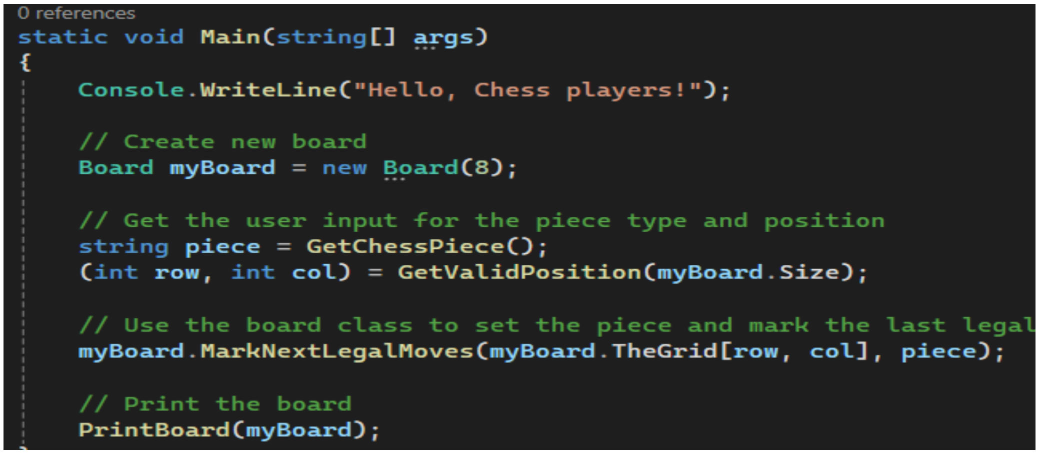
Placing Knight



```
Hello, Chess players!
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
Enter the row number for the piece:
4
Enter the column number for the piece:
4
  .   .   .   .   .   .   .   .
  .   .   .   .   .   .   .   .
  .   .   .   +   .   +   .   .
  .   .   +   .   .   .   +   .
  .   .   .   .   N   .   .   .
  .   .   +   .   .   .   +   .
  .   .   .   +   .   +   .   .
  .   .   .   .   .   .   .   .

C:\Users\USER\source\repos\ChessBoardConsoleA
```

In this screenshot we have added the ability for the console to receive user input. First, we are requesting the user to select the number for the row and column they would like to place the “Knight” game piece on, in TheGrid of cell being the chessboard. Once the row and column are selected the console will display a new chessboard with the piece on the cell the user selected so that the user can see all the possible moves of the “Knight”.

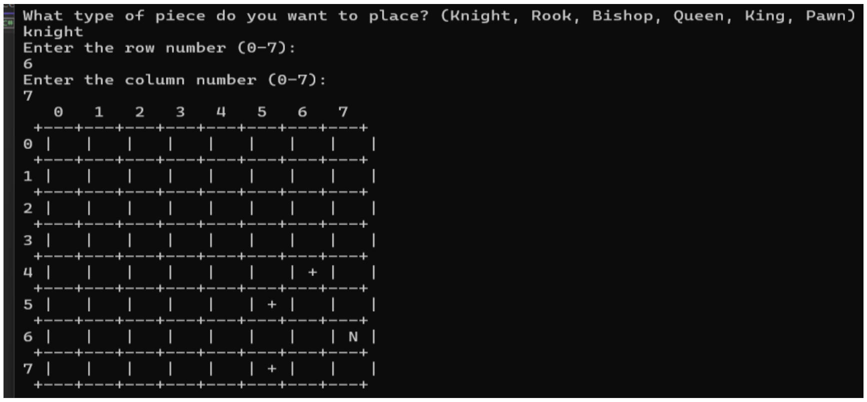
Refactoring Main Method



```
0 references
static void Main(string[] args)
{
    Console.WriteLine("Hello, Chess players!");

    // Create new board
    Board myBoard = new Board(8);

    // Get the user input for the piece type and position
    string piece = GetChessPiece();
    (int row, int col) = GetValidPosition(myBoard.Size);

    // Use the board class to set the piece and mark the last legal
    myBoard.MarkNextLegalMoves(myBoard.TheGrid[row, col], piece);

    // Print the board
    PrintBoard(myBoard);
```

In this screenshot I am showing how/what I have done to refactor the main method for this application. First, we start off the same by welcoming the player to the chess game and creating the board. Then this is where we made a little change; we start off by creating a string “piece” that getting the chess piece that is received by the user's input, which will also have some error handling to prevent you user trying to put in a name besides a legal chess game piece. Then continue the same by using the board class to set the selected piece and its legal moves.

Fixed Input Errors

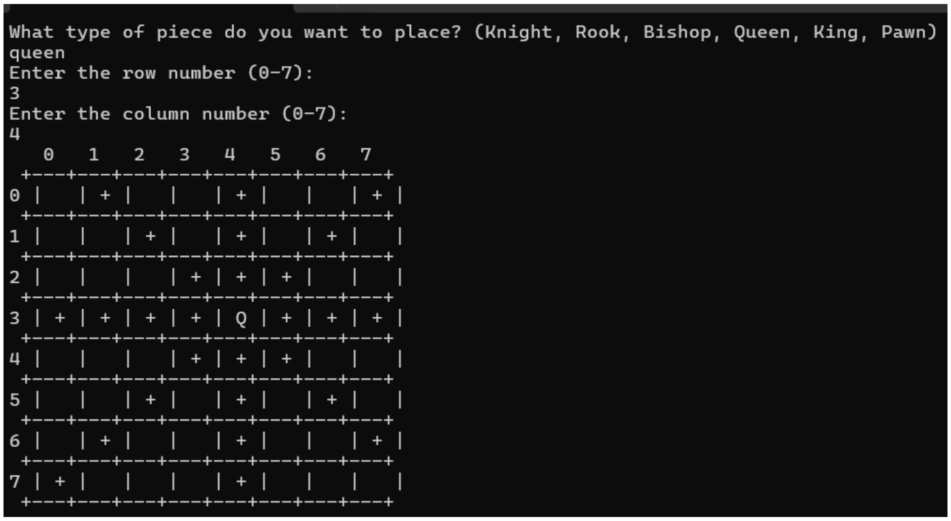


```
What type of piece do you want to place? (Knight, Rook, Bishop, Queen, King, Pawn)
knight
Enter the row number (0-7):
6
Enter the column number (0-7):
7
   0   1   2   3   4   5   6   7
 +---+---+---+---+---+---+---+---+
0 |   |   |   |   |   |   |   |   |
 +---+---+---+---+---+---+---+---+
1 |   |   |   |   |   |   |   |   |
 +---+---+---+---+---+---+---+---+
2 |   |   |   |   |   |   |   |   |
 +---+---+---+---+---+---+---+---+
3 |   |   |   |   |   |   |   |   |
 +---+---+---+---+---+---+---+---+
4 |   |   |   |   |   |   | + |   |
 +---+---+---+---+---+---+---+---+
5 |   |   |   |   |   | + |   |   |
 +---+---+---+---+---+---+---+---+
6 |   |   |   |   |   |   |   | N |
 +---+---+---+---+---+---+---+---+
7 |   |   |   |   |   | + |   |   |
 +---+---+---+---+---+---+---+---+
```

In this screenshot we are demonstrating how er have fixed the error for when the user is supposed to type a number between 0 and 7 for where to place the piece on the board.

Here we have fixed the error for legal moves trying to reach outside the range of the grid and disallowed other numbers and letters.

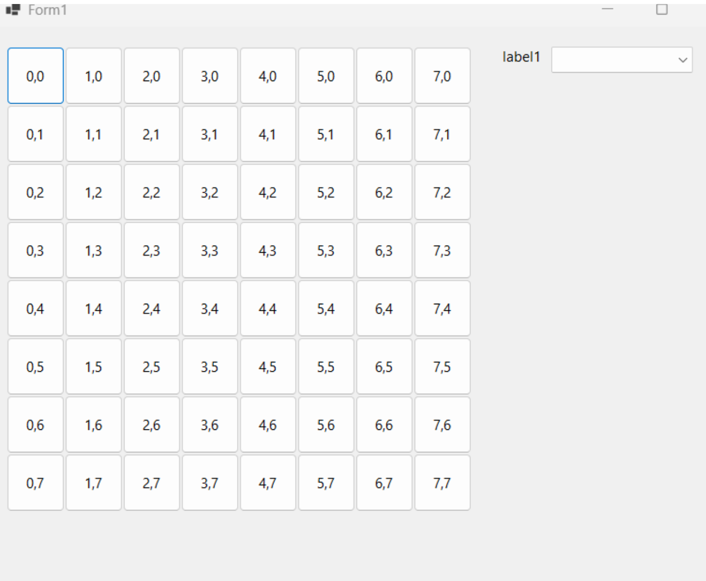
Multiple Pieces and Printing Outlines



```
What type of piece do you want to place? (Knight, Rook, Bishop, Queen, King, Pawn)
queen
Enter the row number (0-7):
3
Enter the column number (0-7):
4
    0   1   2   3   4   5   6   7
  +---+---+---+---+---+---+---+---+
0 |   | + |   |   | + |   |   | + |
  +---+---+---+---+---+---+---+---+
1 |   |   | + |   | + |   | + |   |
  +---+---+---+---+---+---+---+---+
2 |   |   |   | + | + | + |   |   |
  +---+---+---+---+---+---+---+---+
3 | + | + | + | + | Q | + | + | + |
  +---+---+---+---+---+---+---+---+
4 |   |   |   | + | + | + |   |   |
  +---+---+---+---+---+---+---+---+
5 |   |   | + |   | + |   | + |   |
  +---+---+---+---+---+---+---+---+
6 |   | + |   |   | + |   |   | + |
  +---+---+---+---+---+---+---+---+
7 | + |   |   |   | + |   |   |   |
  +---+---+---+---+---+---+---+---+
```

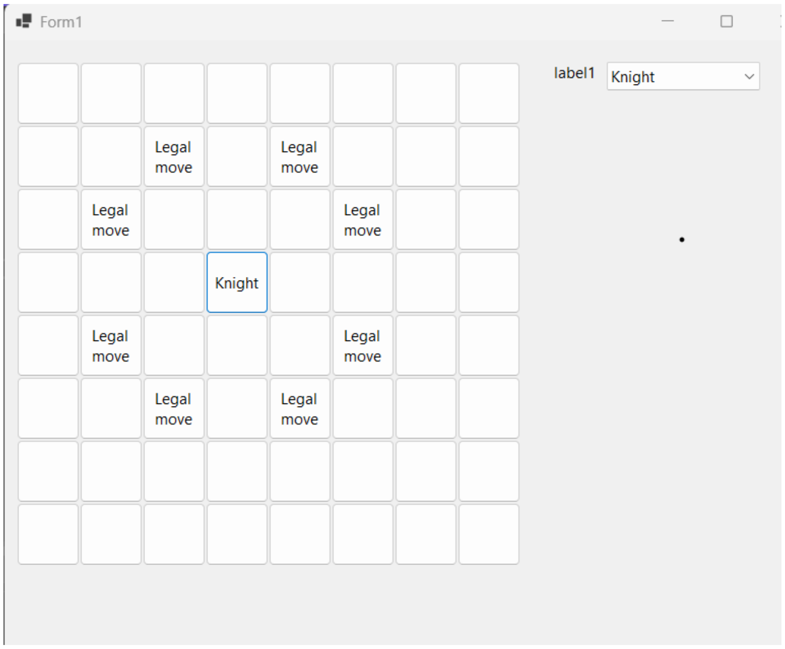
In this screenshot we are demonstrating how we have added in the back-end logic for the other game pieces in chess besides the Knight, as far as I can tell the logic for the movements is correct but will keep playing with the pieces positions to make sure. We have also added grid lines to our game board to make things look much more appealing and it is easier to see where the legal moves actually are for each piece.

GUI First Running



In this screenshot we are demonstrating the GUI that your user will be playing the Chess game on. Here we are first focusing on adding the grid/game board that will be made up of cells and that will be used to move the game pieces alongside of. We also have added a label and comboBox which we will add more functionality to at a later point.

GUI Chess Piece Legal Moves

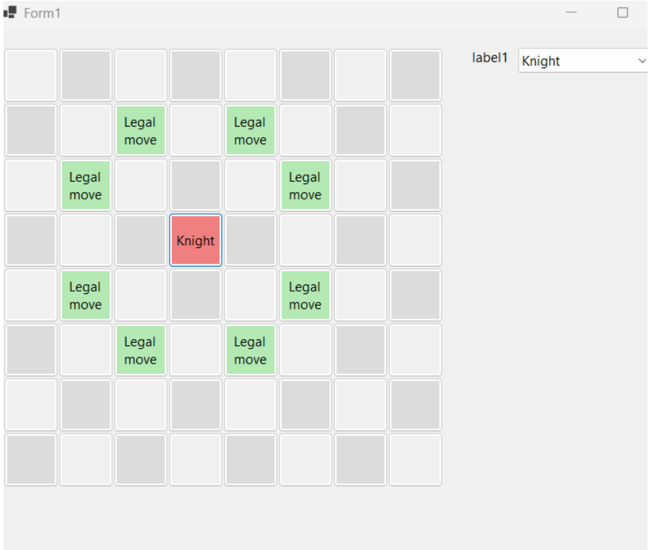


label1 Knight

		Legal move		Legal move			
	Legal move				Legal move		
			Knight				
	Legal move				Legal move		
		Legal move		Legal move			

This screenshot is a demonstration that we have added the ability for the user to select a chess game piece and then will be able to click any of the cells on the grid. Then the application updates and adds the game piece to the board as well as showing that specific game pieces all legal moves that it can make.

Adding Color

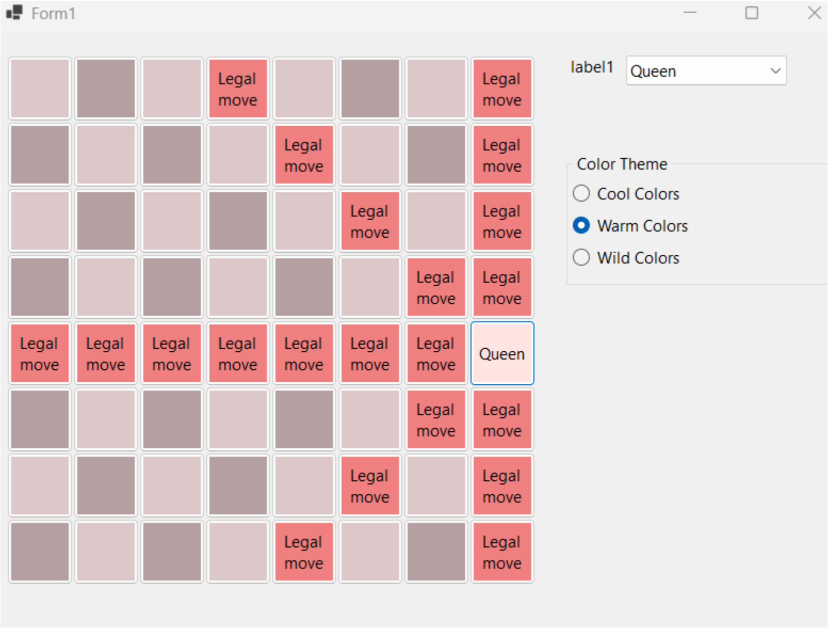


label1 Knight

		Legal move		Legal move			
	Legal move				Legal move		
			Knight				
	Legal move				Legal move		
		Legal move		Legal move			

In this screenshot we are just demonstration the ability that we have added of adding some color to the cell for what are legal moves, making them green and making the cell the game piece is taking up to be red.

Error Checking



			Legal move				Legal move
				Legal move			Legal move
					Legal move		Legal move
						Legal move	Legal move
Legal move	Legal move	Legal move	Legal move	Legal move	Legal move	Legal move	Queen
						Legal move	Legal move
					Legal move		Legal move
				Legal move			Legal move

label1 Queen

Color Theme

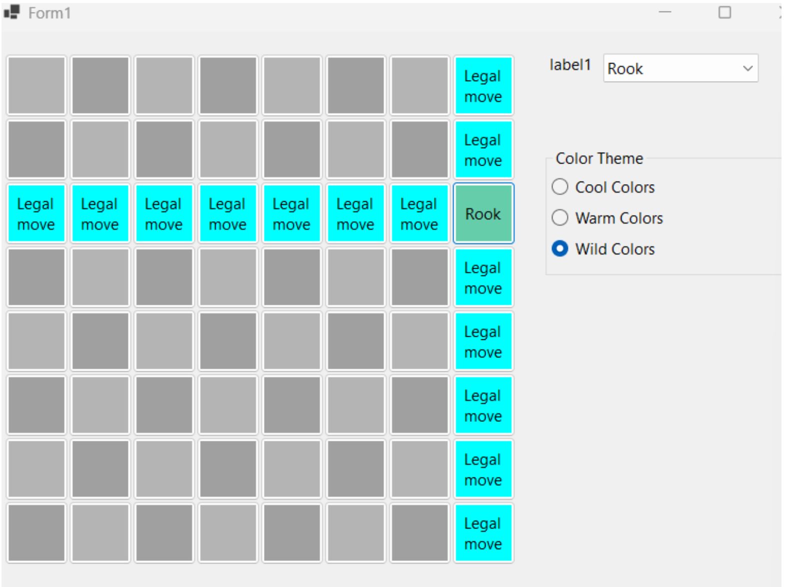
○ Cool Colors

● Warm Colors

○ Wild Colors

In this screenshot we are demonstrating how we have implanted code to fix all out of bounds errors that take place for each of the game pieces. The game board will only show the legal moves that are on the game board making it very easy for the user to follow along.

Multiple Pieces



							Legal move
							Legal move
Legal move	Legal move	Legal move	Legal move	Legal move	Legal move	Legal move	Rook
							Legal move
							Legal move
							Legal move
							Legal move
							Legal move

label1 Rook

Color Theme

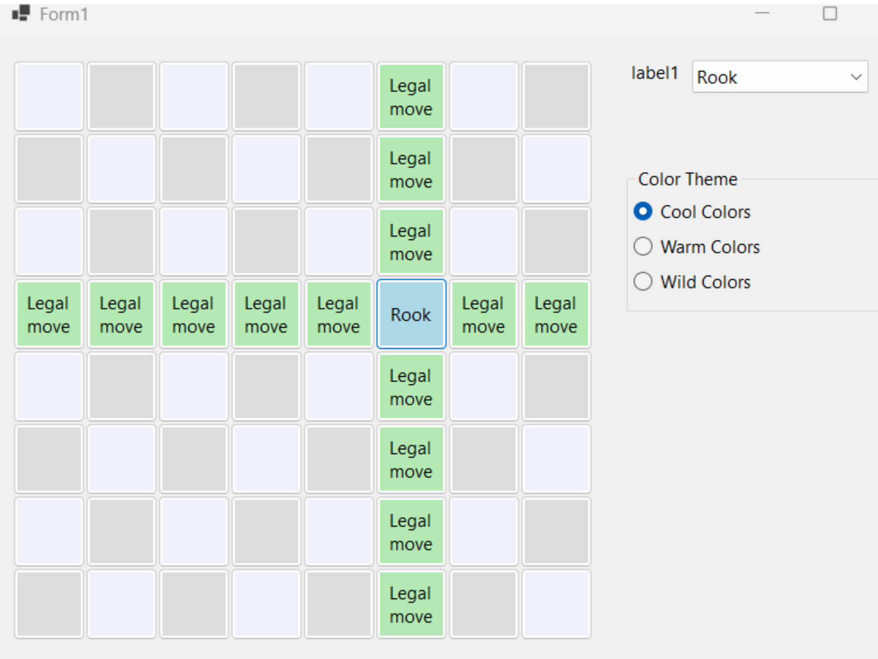
○ Cool Colors

○ Warm Colors

● Wild Colors

In this screenshot we are demonstration how we have added the back-end logic for each of the game pieces and the legal moves that they are allowed to make on the game board that are available based on the cell position that is selected.

Color Theme Picker



					Legal move		
					Legal move		
					Legal move		
Legal move	Legal move	Legal move	Legal move	Legal move	Rook	Legal move	Legal move
					Legal move		
					Legal move		
					Legal move		
					Legal move		

label1 Rook

Color Theme
- ● Cool Colors
- ○ Warm Colors
- ○ Wild Colors

In this screenshot we are demonstrating the different color theme that we have attached to this application. We have given the user the ability to change the color style based on their own preferences. The other two screenshots above are all done in the different color theme options that we have coded to show that each work perfectly.

1. What was challenging?
   I would have to say that probably the most challenging thing about this activity was trying to figure out the legal movements for each game piece. I had to start by deciding what order I was going to start them in cause after I bit of review, I realized that the King and the Rook are somewhat similar in their movements, so I started there and got those solved. Then the rest kind of just fell into place from there, the Bishop was just a diagonal variation of the Rook and the Queen was just a combination of the Rook and Bishop.
2. What did you learn?
   In this lesson I would say I have learned a lot about how to connect the backend logic to specific tools on the GUI side of thinks for a user to be able to interact with. I

gained way better knowledge on how to code the back-end movements based on what I would like the user to see (based on instructions).

3. How would you improve on the project?
   To improve on this project, I would probably like to play more with the GUI side of what the user would be able to see. I would really like to make some custom game pieces to upload or maybe give the user the option to select different pieces. Even could try to investigate how to allow for the user to unlock different game skin pieces.
4. How can you use what you learned on the job?
   I would say that the skills I have learned through this project are highly transferable to a real world software development job. In this project we have demonstrated a proficiency in object-oriented programming, problem-solving, and algorithm development and design. We created the legal move logic for each chess piece required, showing an ability to break down complex movement patterns into systematic rules, like how business requirements are translated into technical solutions. These skills would directly apply to developing enterprise applications, games, or potentially any system requiring complex rule based logic with clear user interfaces.