Chad Galloway

CST-250 Programming in C# II

Grand Canyon University

Nov. 2, 2025

Activity 2

Files

<https://github.com/CGalloway3/CST-250-Projects/tree/master/Activity%202>

Video

[Video Link]

FLOW CHART



Figure X: Flow chart of XXXXXXX

"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

UML Class Diagram



Figure X: UML Class Diagram

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

**Part 1**

Screen Shots

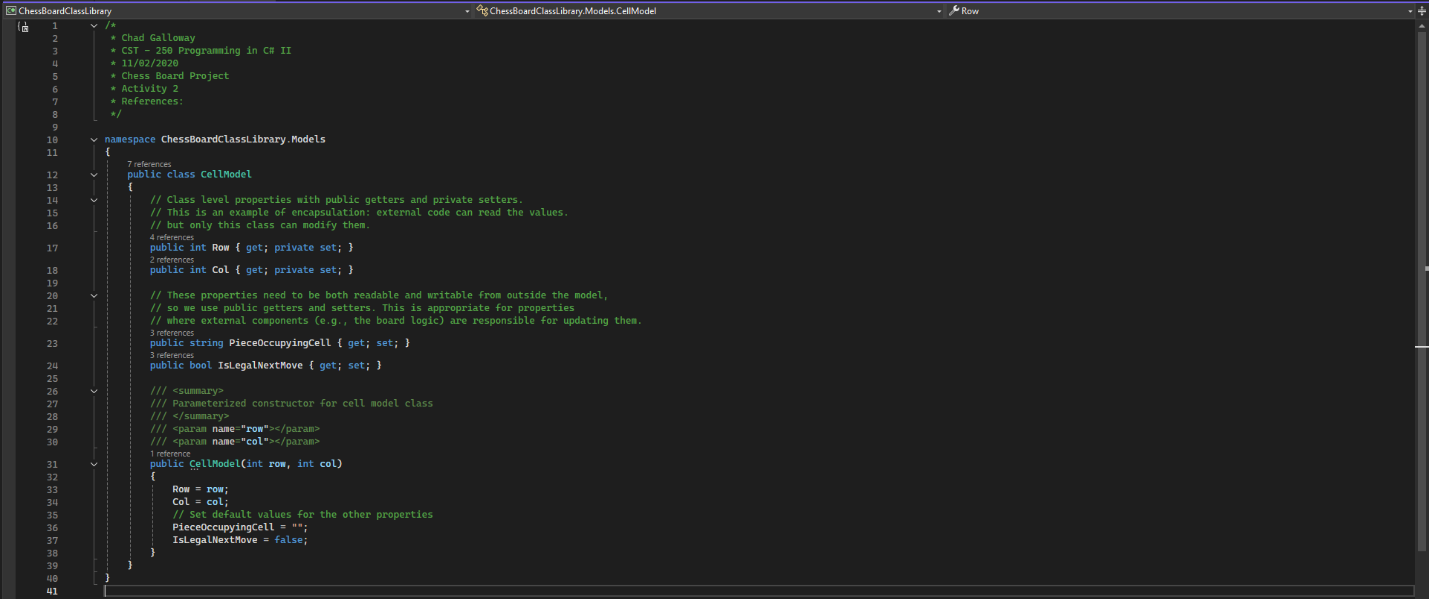


Figure 1: CellModel citations and constructor

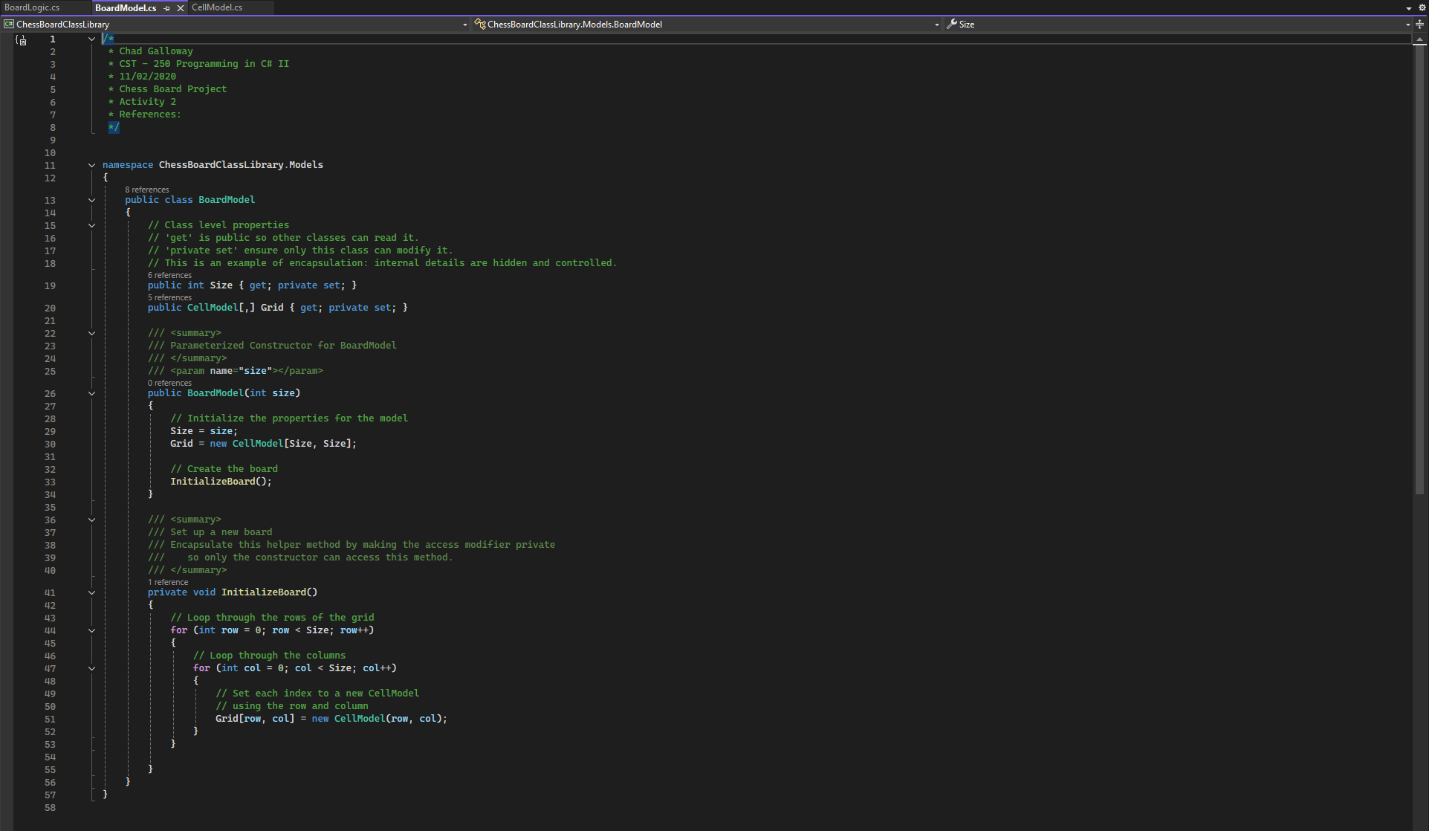


Figure 2: BoardModel citations and constructor

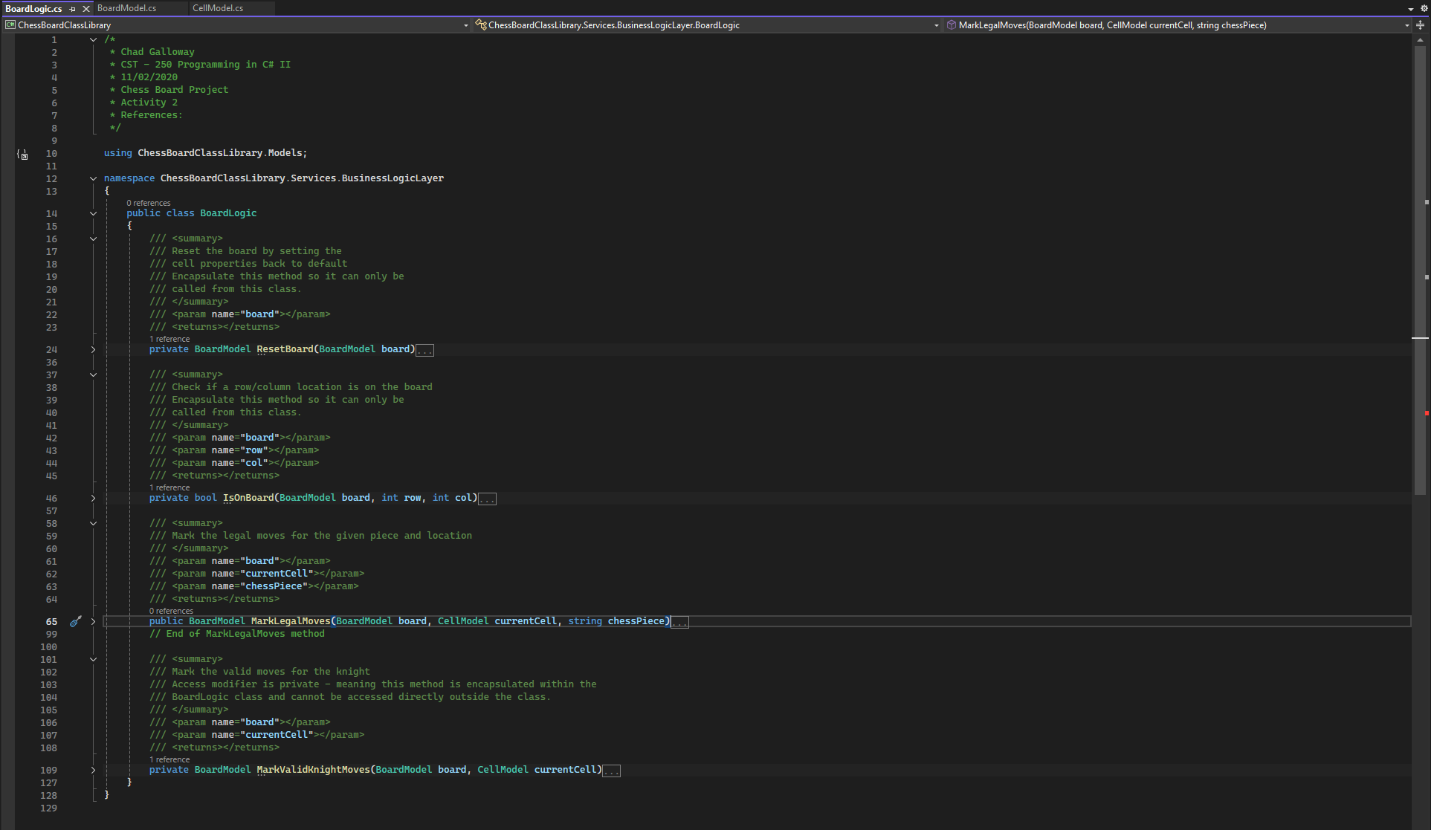


Figure 3: BoardLogic and collapsed methods with expanded summary

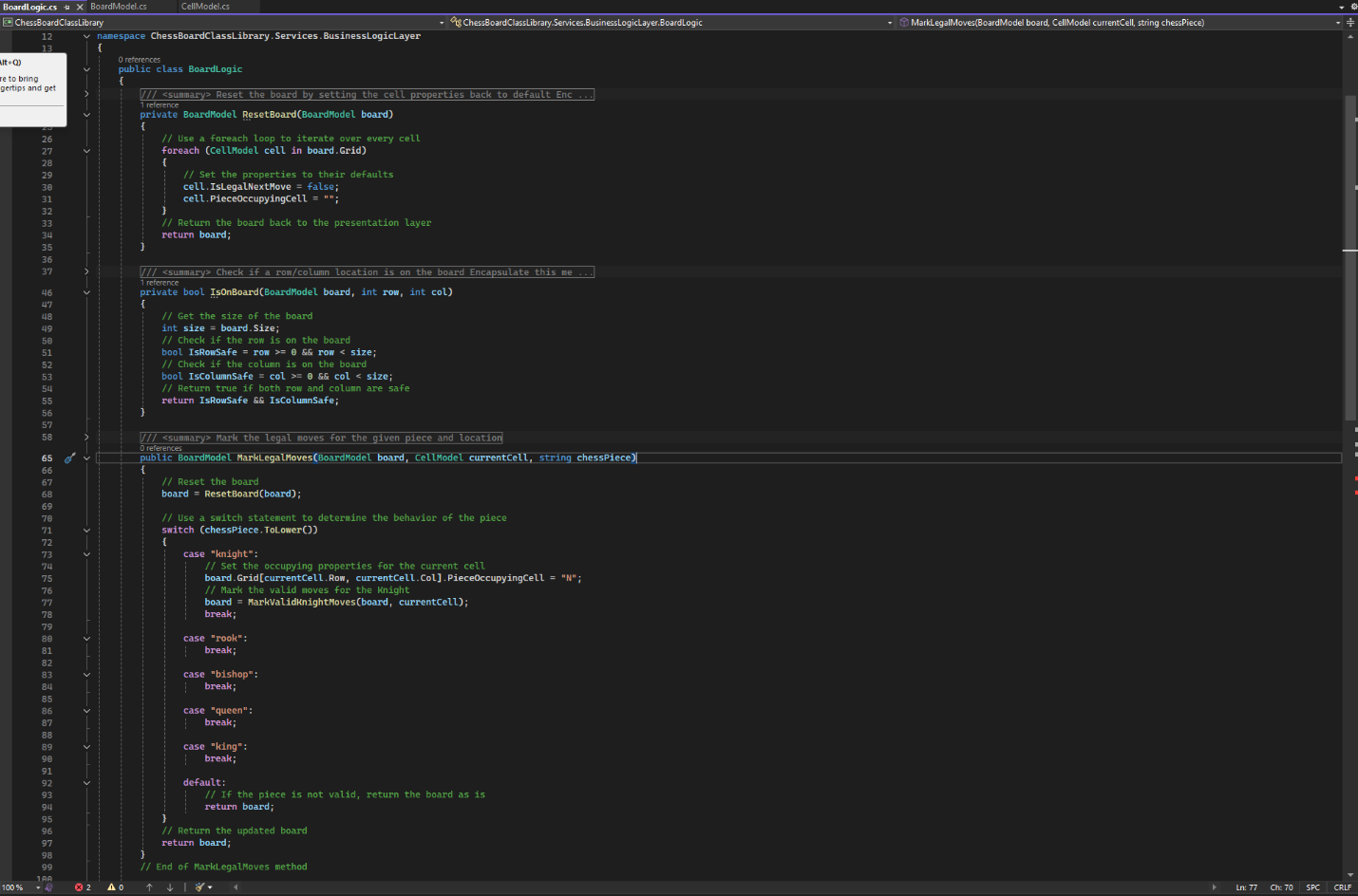


Figure 4: BoardLogic: ResetBoard(), IsOnBoard(), and MarkLegalMoves() with collapsed summaries

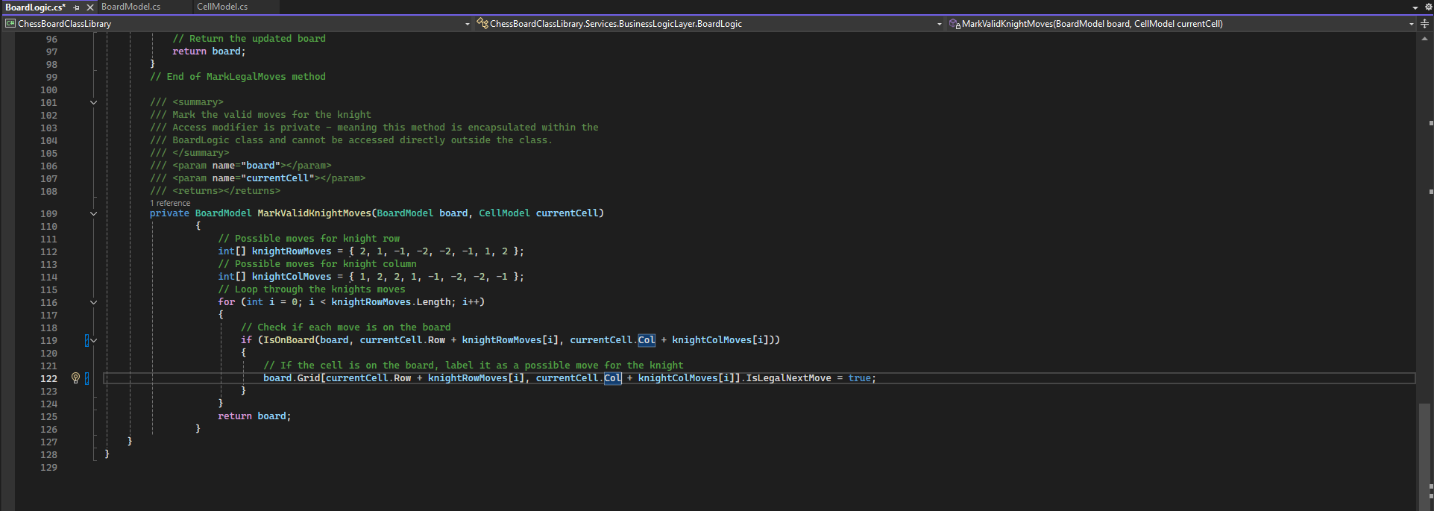


Figure 5: BoardLogic.MarkValidKnightMoves()

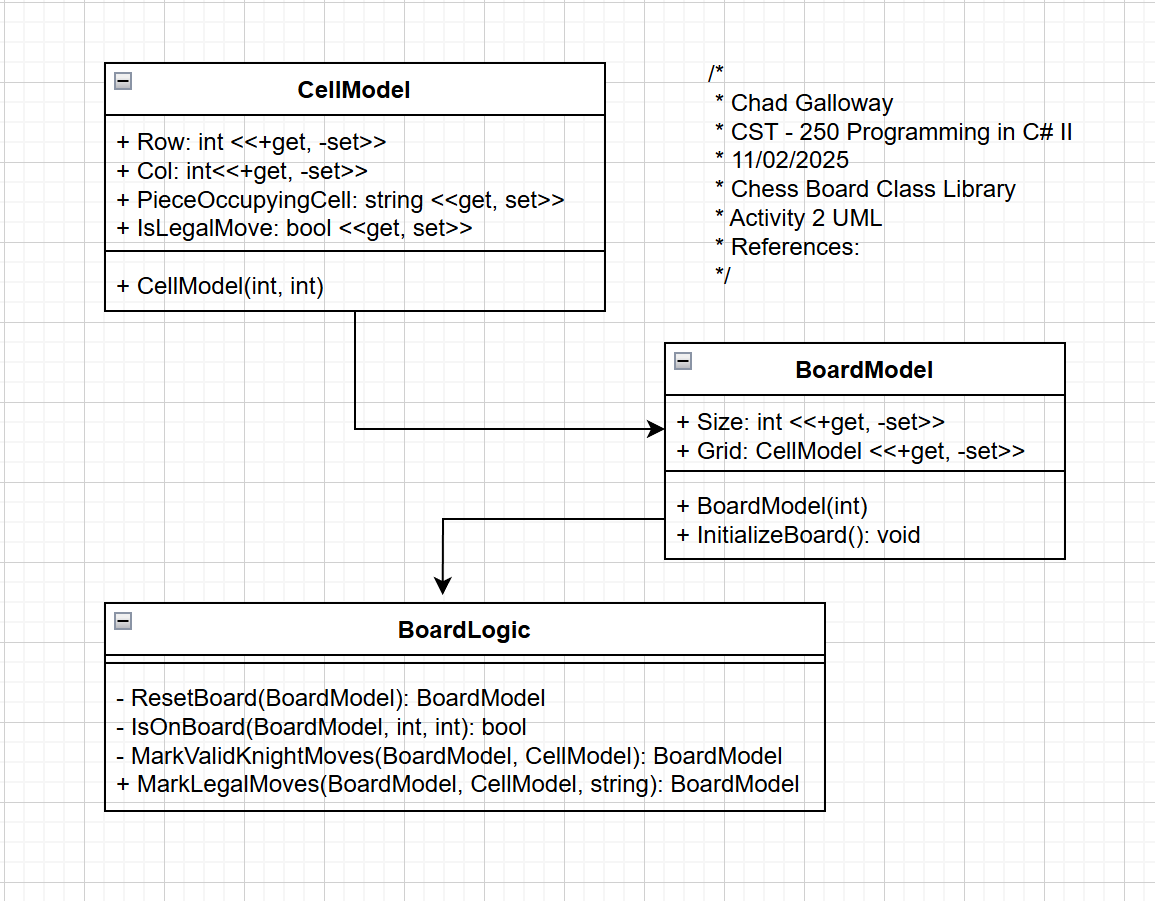


Figure 6: Screenshot of UML for CellModel, BoardModel, and BoardLogic

Challenges

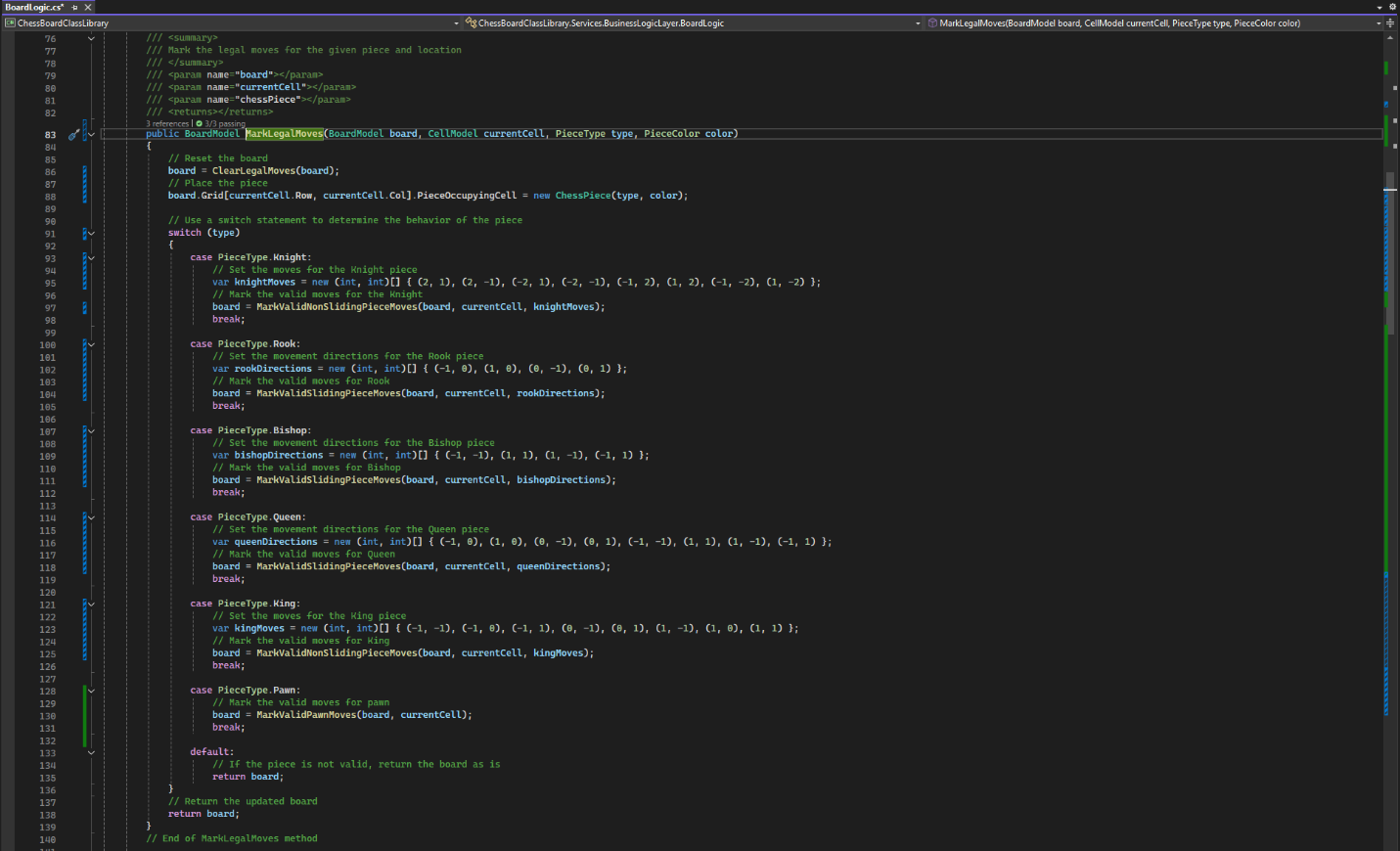


Figure 7: Updated and completed BoardLogic.MarkLegalMoves() method. Changed the string parameter for piece to type and color enums I created. Moved piece creation out of the cases and into the main body using a ChessPieice class I ceated.

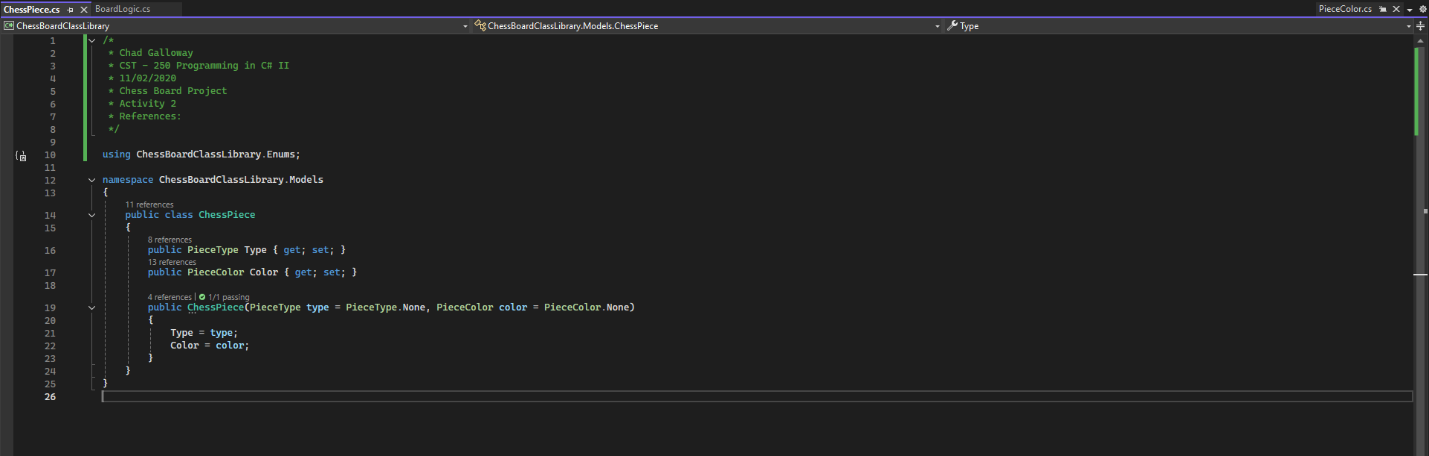


Figure 8: ChessPiece class. Contains piece type and piece color. Currently the piece doesn’t do anything (no methods). In the future it might have some movement methods added in we will see.

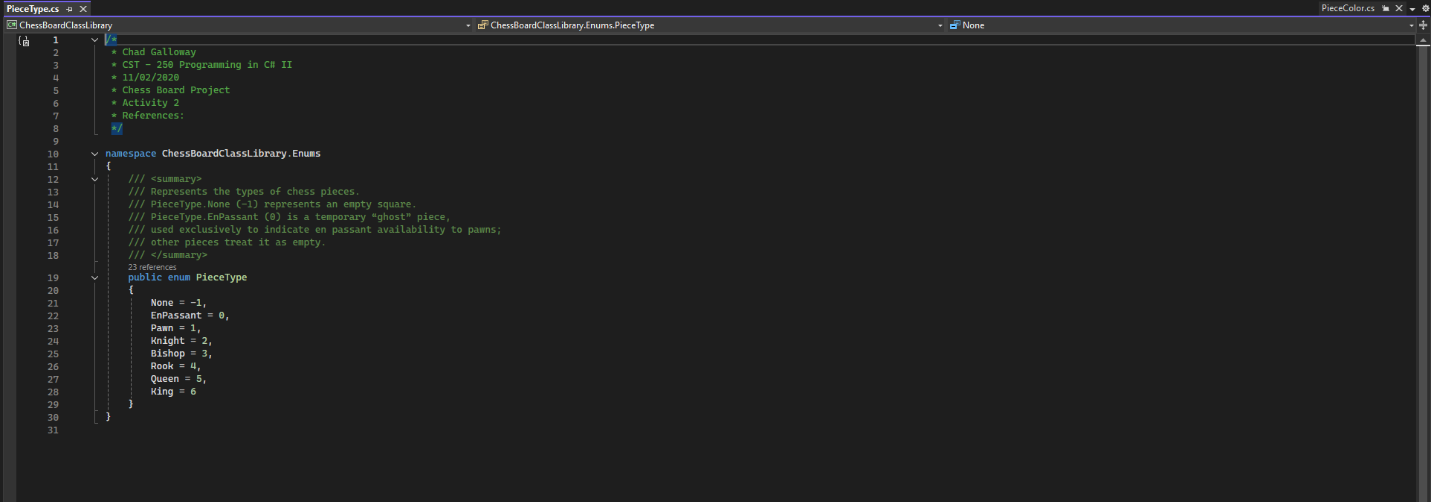


Figure 9: PieceType enum to hold all the different piece types including none and en passant a “ghost” piece type only visible in code to pawns for preforming the en passant move.

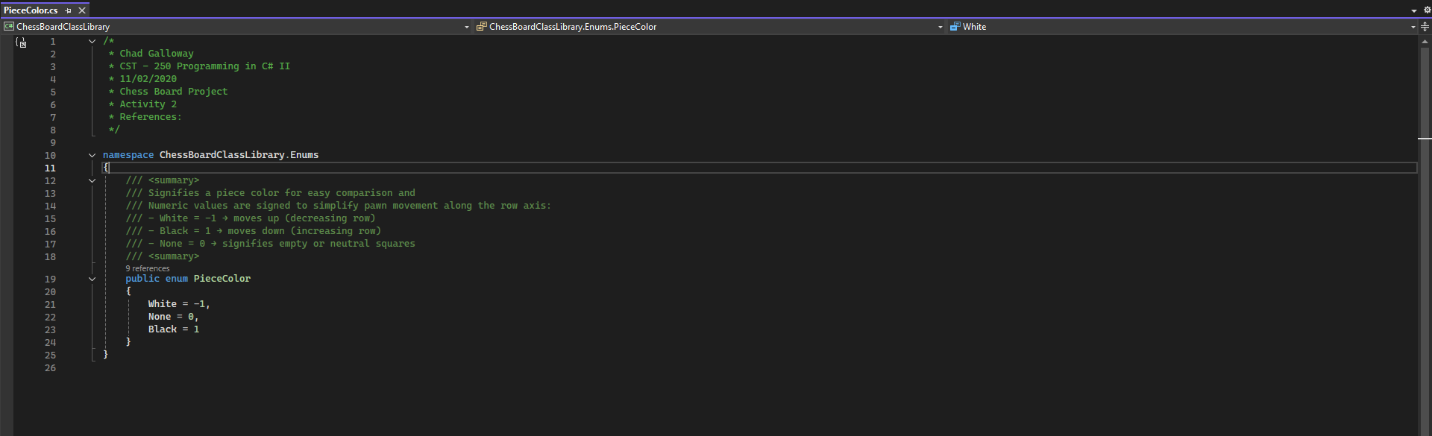


Figure 10: PieceColor emun to hold the colors of pieces. Used to determine if a legal move of taking an enemy piece is available and it is also a directional indicator of the movement directions of the two colored pieces for pawns. Black pawns move down the board +1, and white pawns move up the board -1.

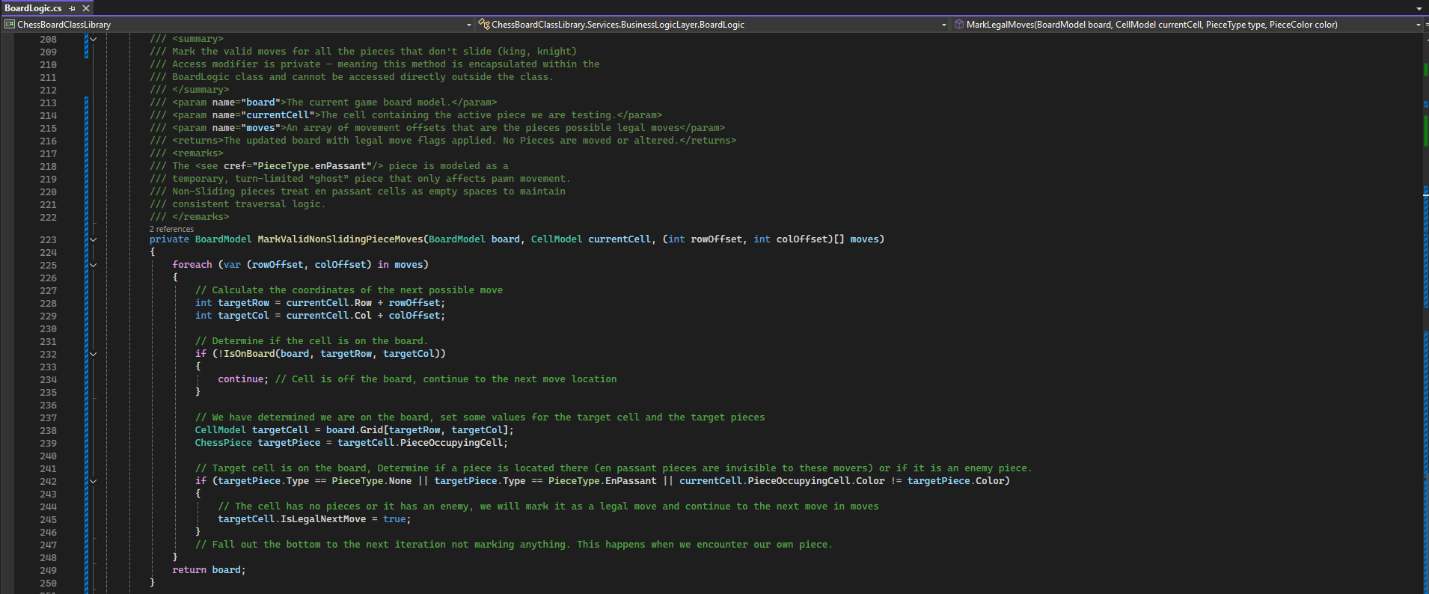


Figure 11: BoardLogic.MarkValidKnightMoves() was removed and combined with the king logic into a method for all non-sliding pieces MarkValidNonSlidingPieceMoves(). It handles knights and kings.

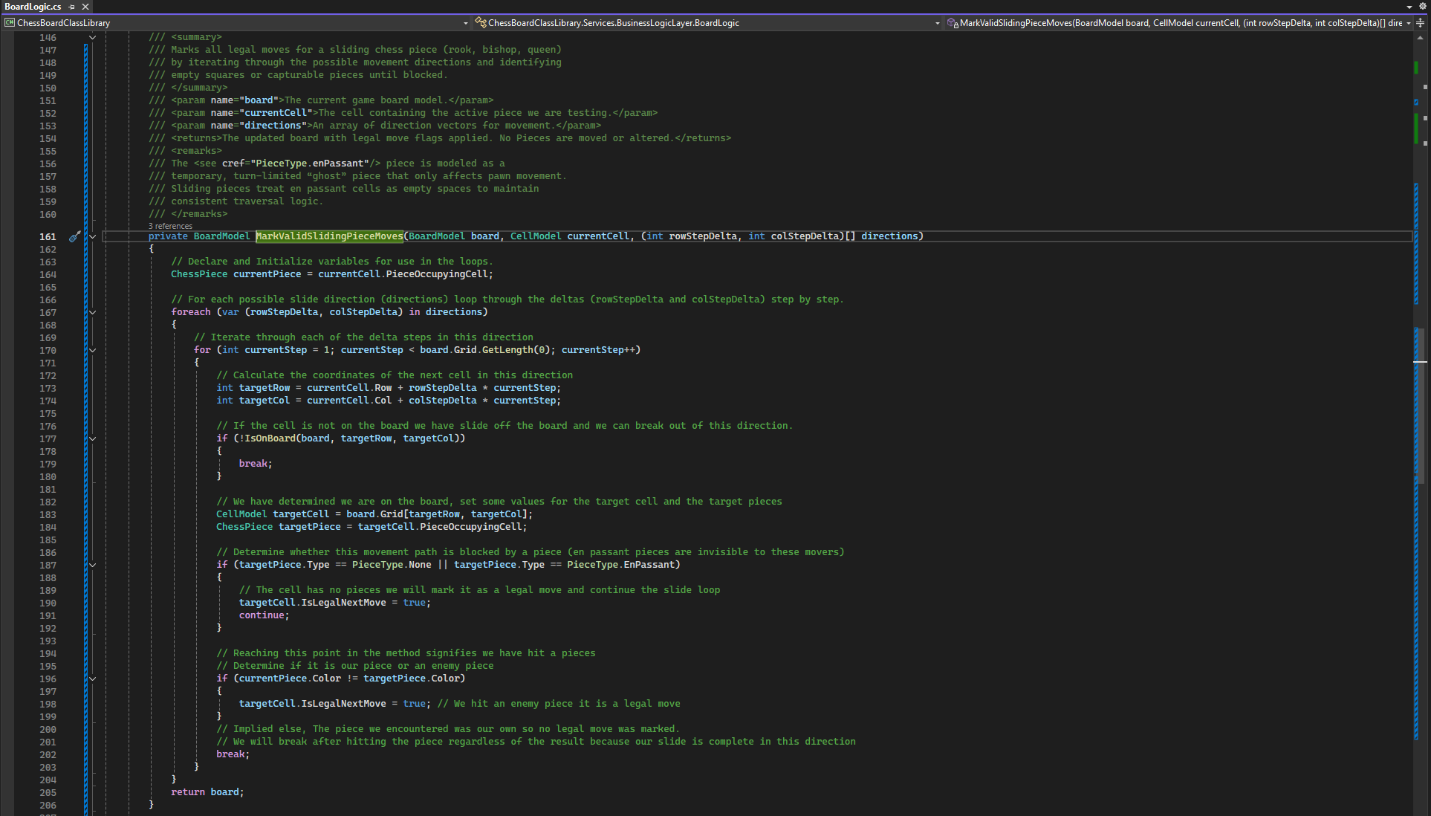


Figure 12: BoardLogic.MarkValidSlidingPieceMoves() was added to handle pieces that slide (i.e. rook, bishop, and queen)

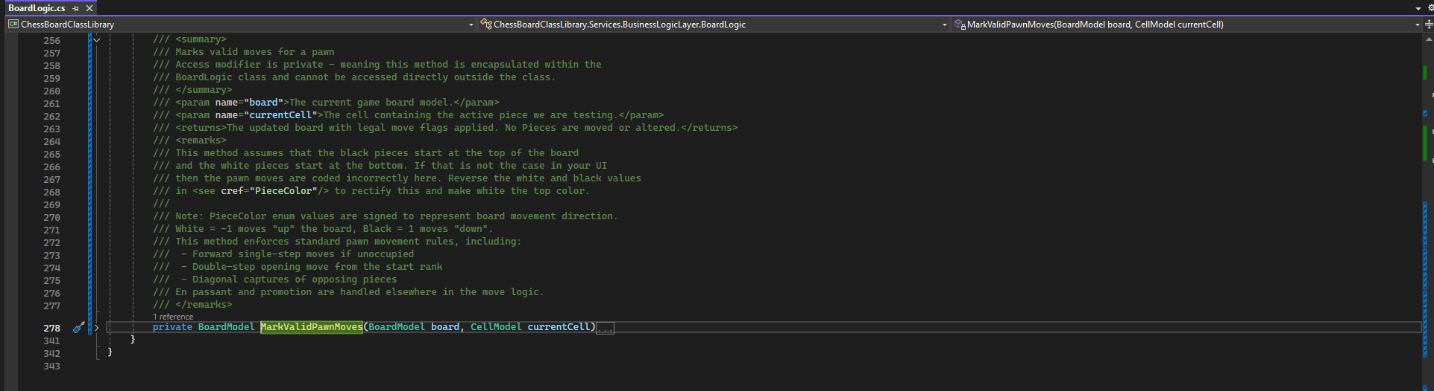


Figure 13: Summary for MarkValidPawnMoves()

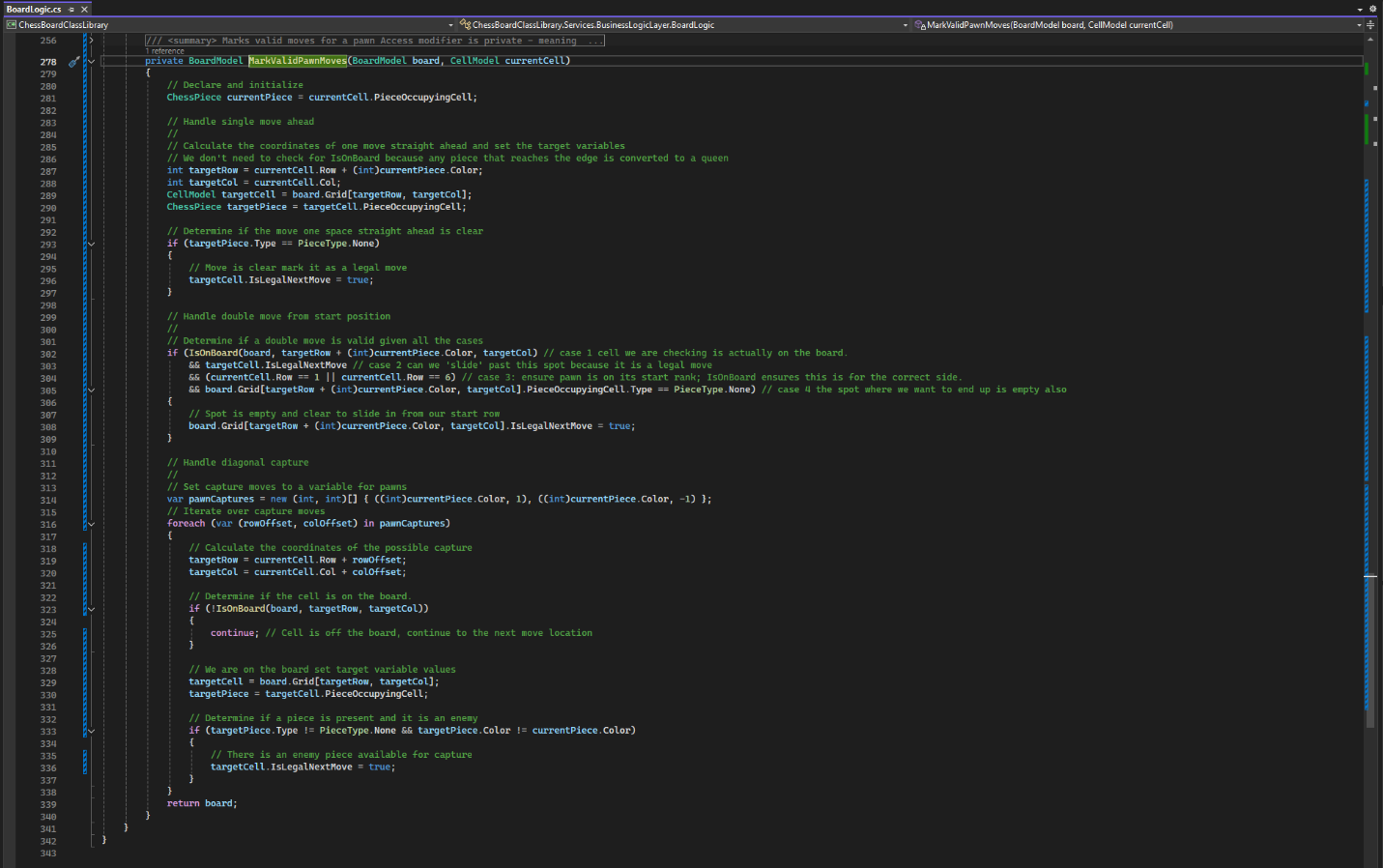


Figure 14: BoardLogic.MarkValidPawnMoves() marks all the movement cases for pawns. Promotions for reaching the last rank and en passant will be handled in the actual move logic. Here we are just highlighting what might be a legal move.

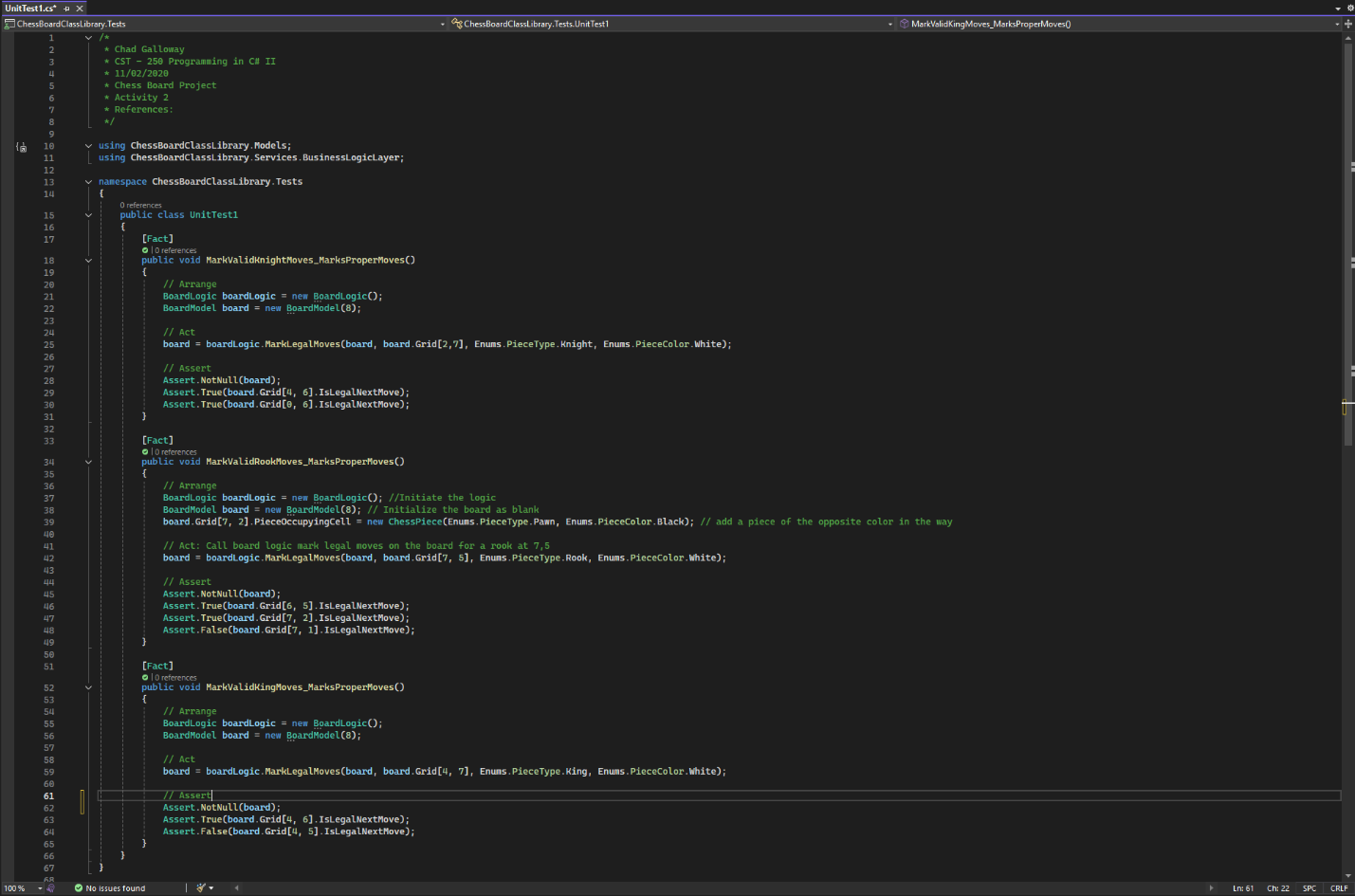


Figure 15: Several unit tests were created for the project to make coding the different MarkValid methods easier.

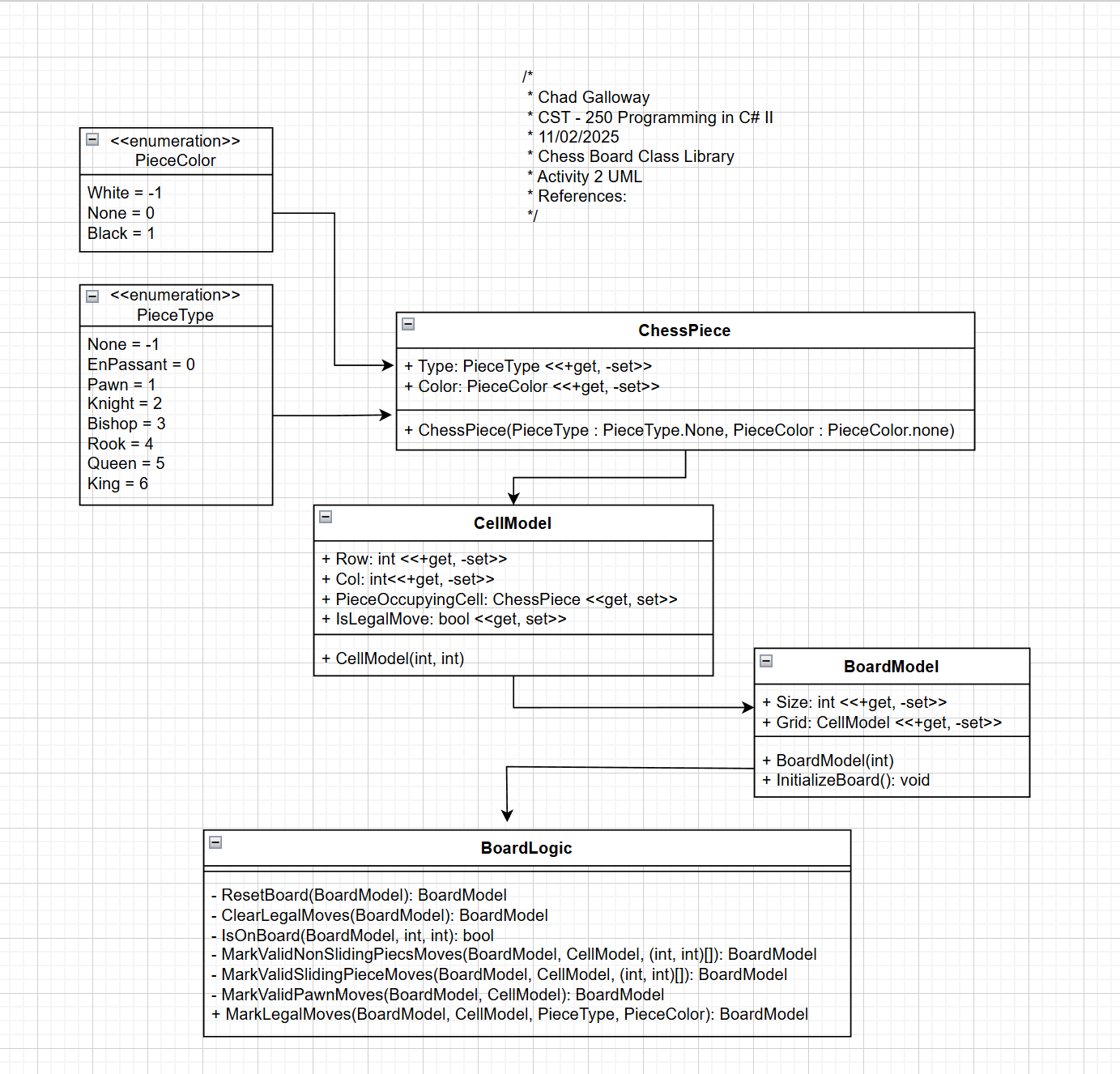


Figure 16: Projects UML so far.

Part 2

Screen Shots

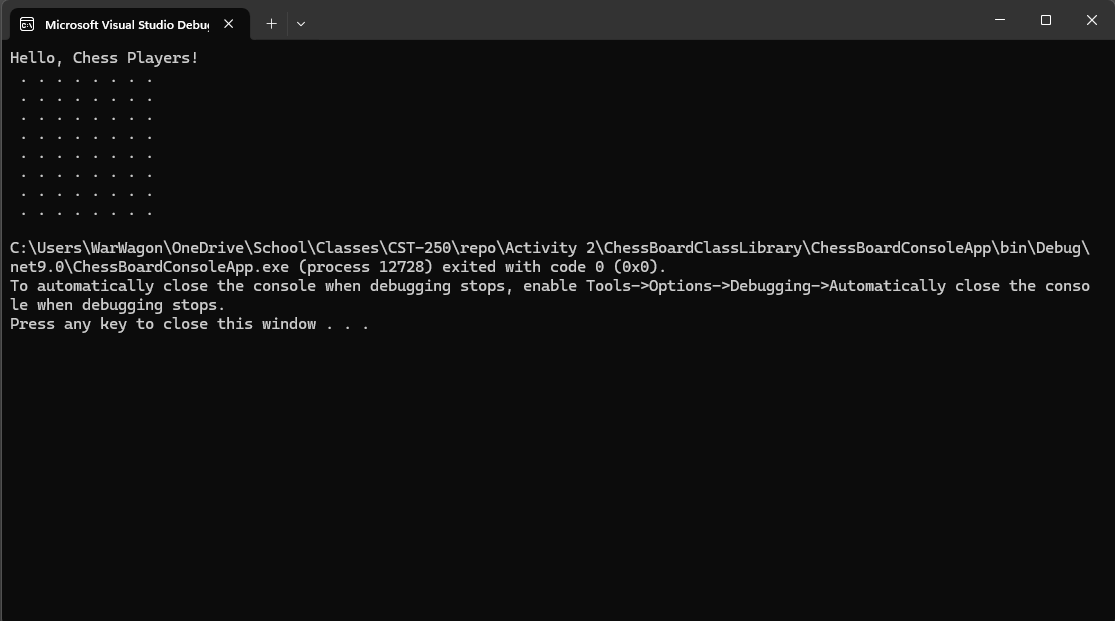


Figure 17: Part 2 App Running after writing PrintBoard() method

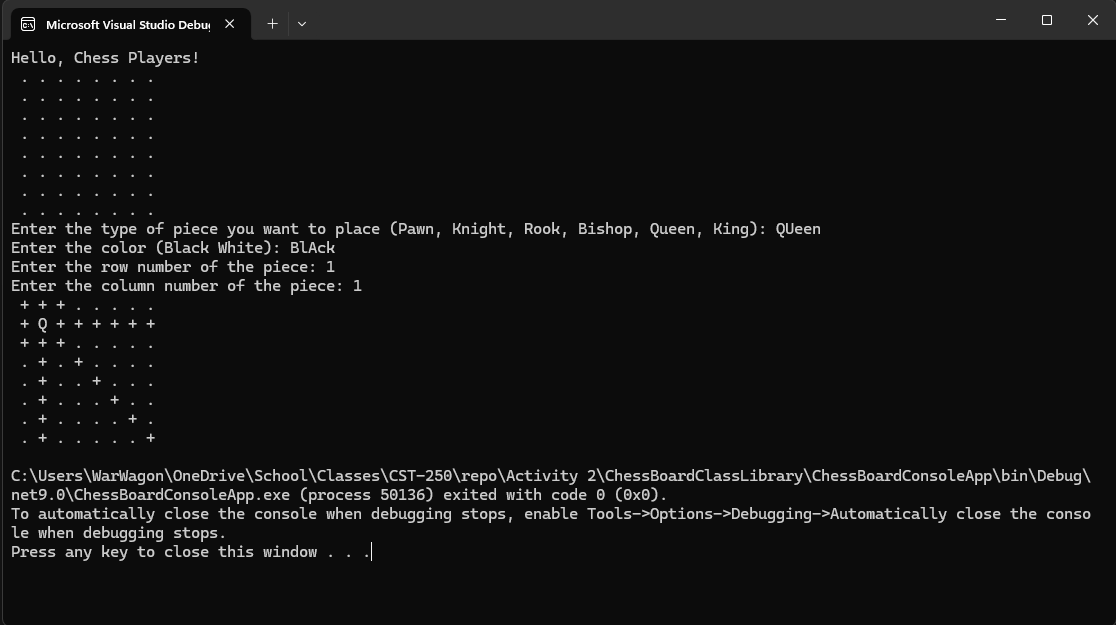


Figure 18: Screenshot of successful output with a Queen at 1, 1

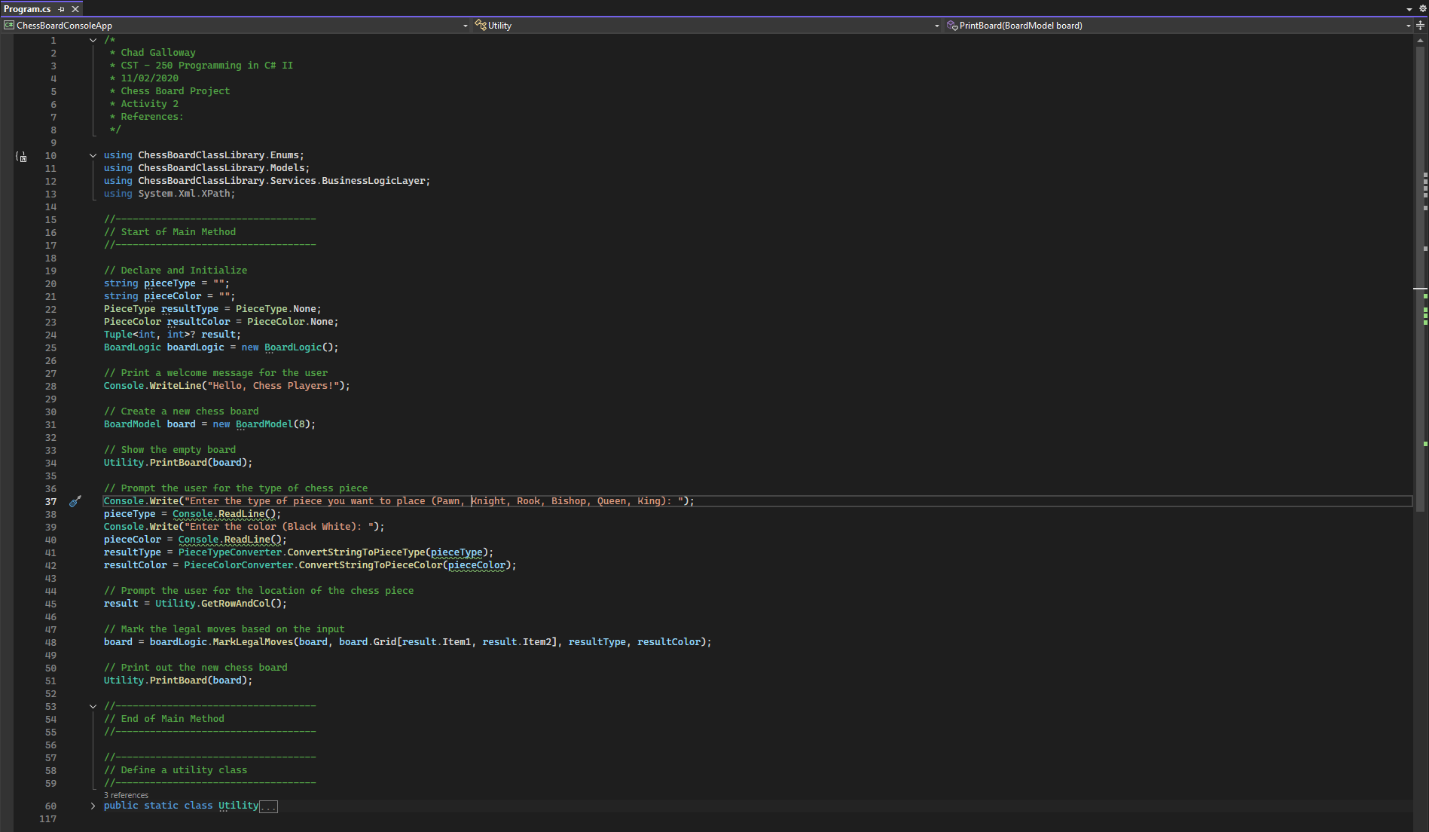


Figure 19: Screen shot of the Propgram.cs and its’ citations

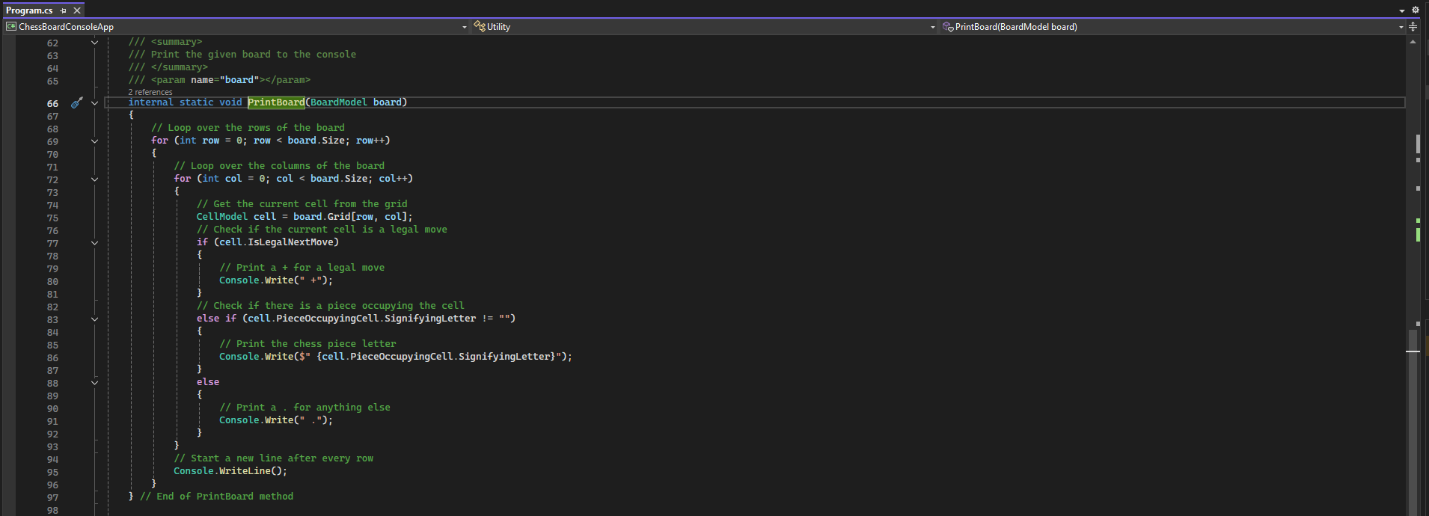


Figure 20: PrintBoard() method

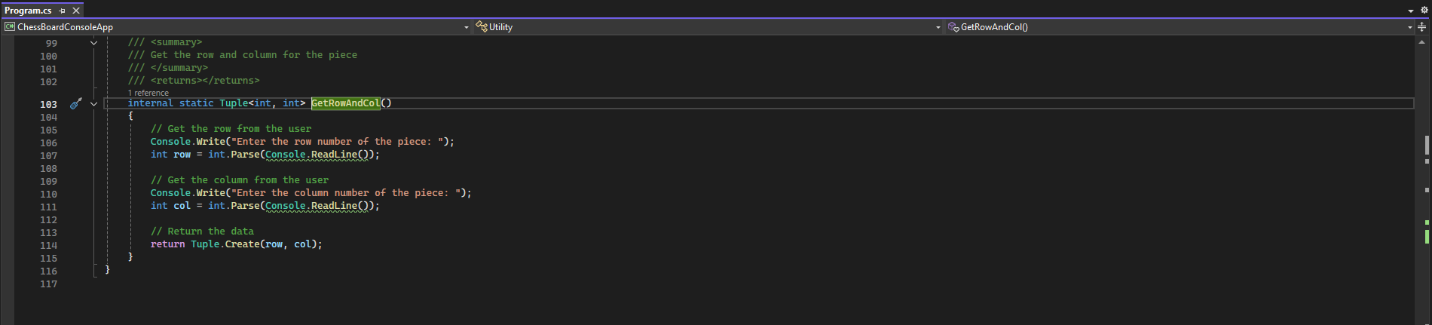


Figure 21: GetRowAndCol() method

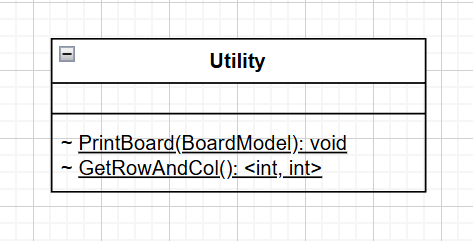


Figure 22: UML

Challenges



Figure 23: Console running after successful completion of the challenge tasks. The First board is defalult. The second board has some pieces added for demonstration and the third board shows off the capabilities of the MarkValid methods I created. White + is a possible move and a red + is a possible piece capture opportunity. With further moves behind pieces that are present being unavailable. Changing the Queen to black removes the red + and displays the piece type because the rook and the king displayed on board 2 are black. Further improvements would be changing the pieces display colors…. I just thought of a novel way to do it so see below.

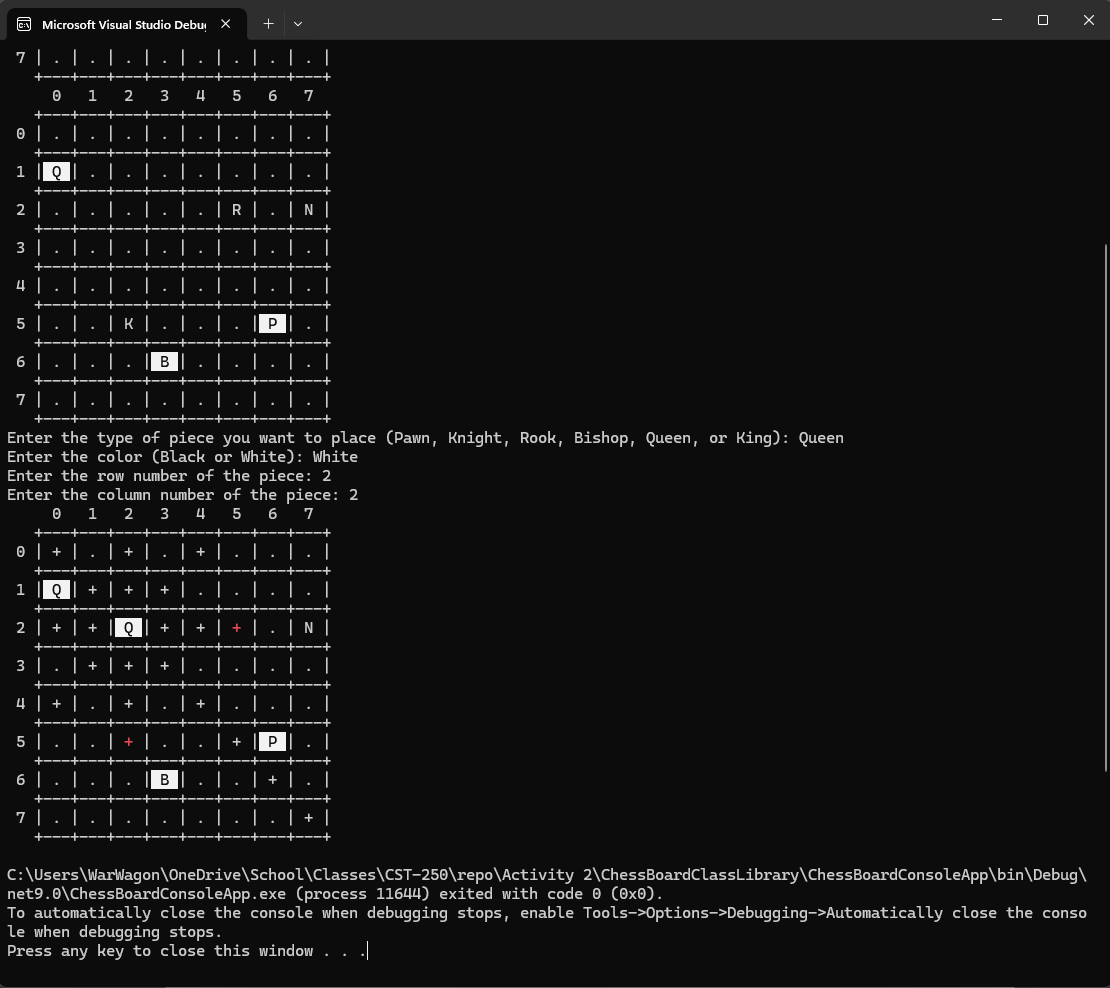


Figure 24: Screen shot of the application with colored pieces implemented. Some other color choices may be in order later like making valid moves yellow and attempting to properly color the board checkered black and white but those are lofty and a bit much for a simple console version when we are about to make a windows forms version with a much better GUI.

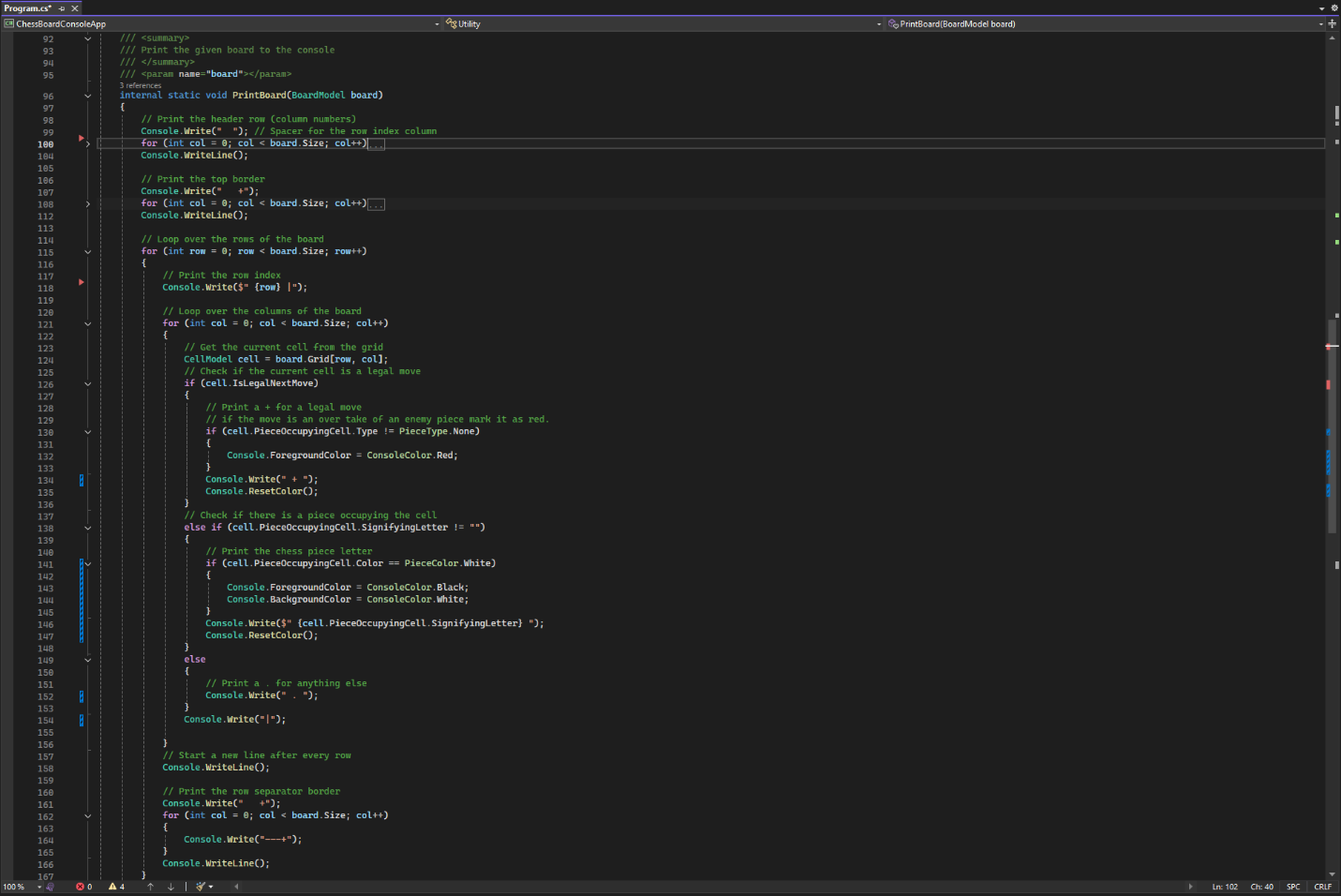


Figure 25: The print board method in the utility class

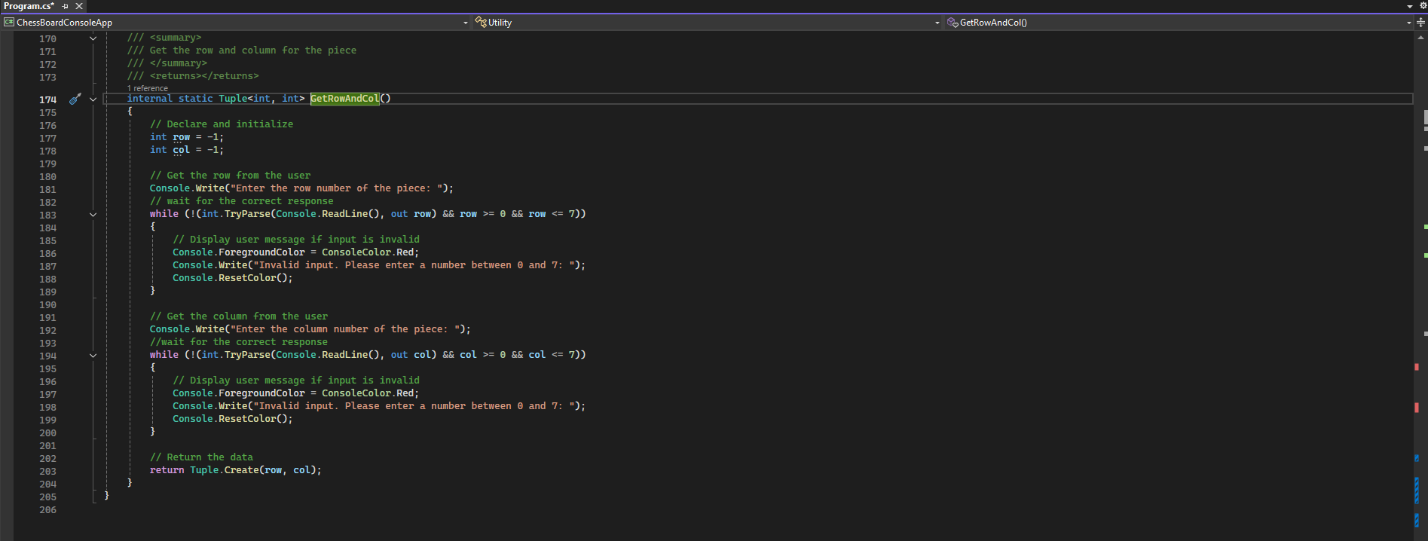


Figure 26: The get row and column method in the utility class

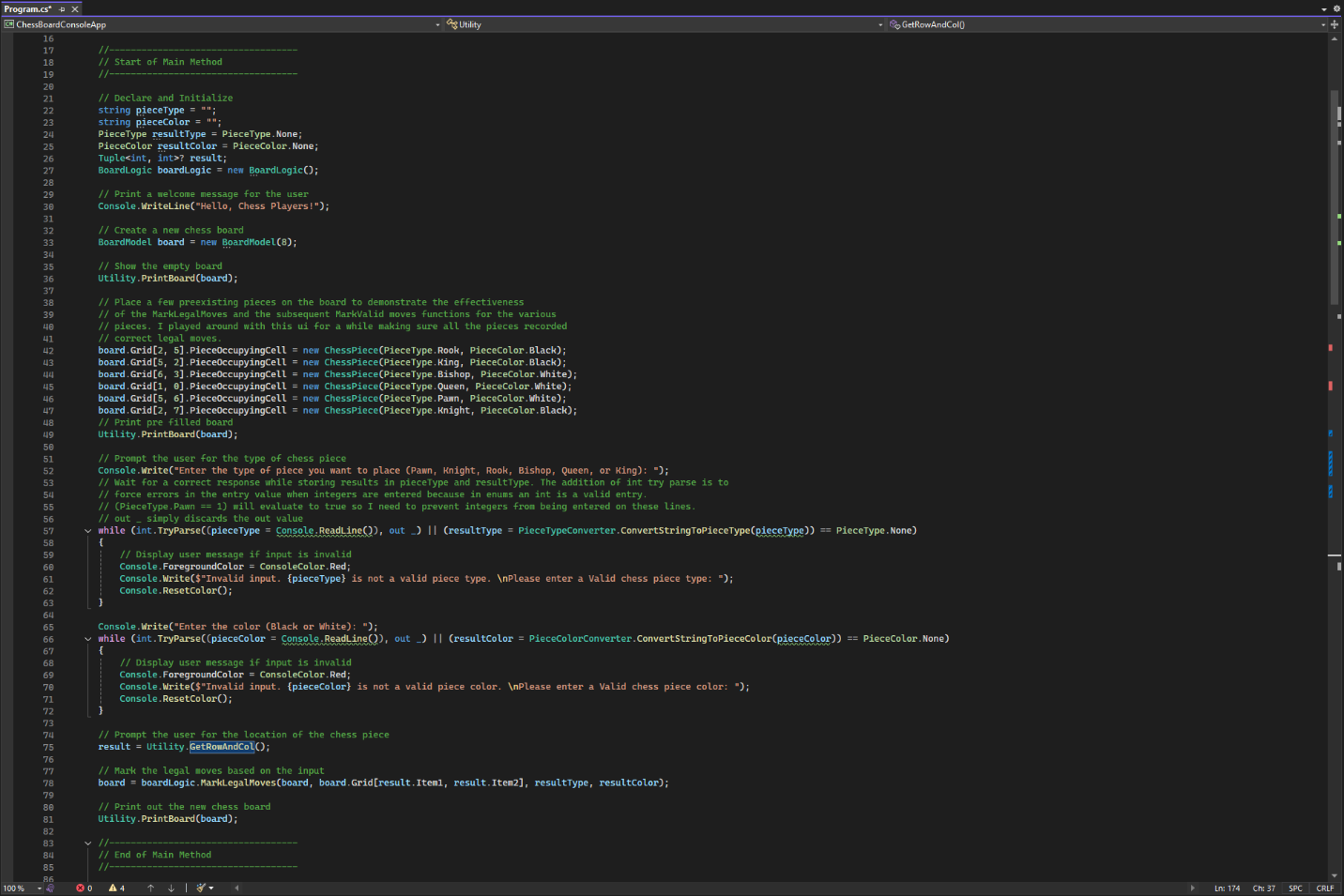


Figure 27: The main method for the console chess app

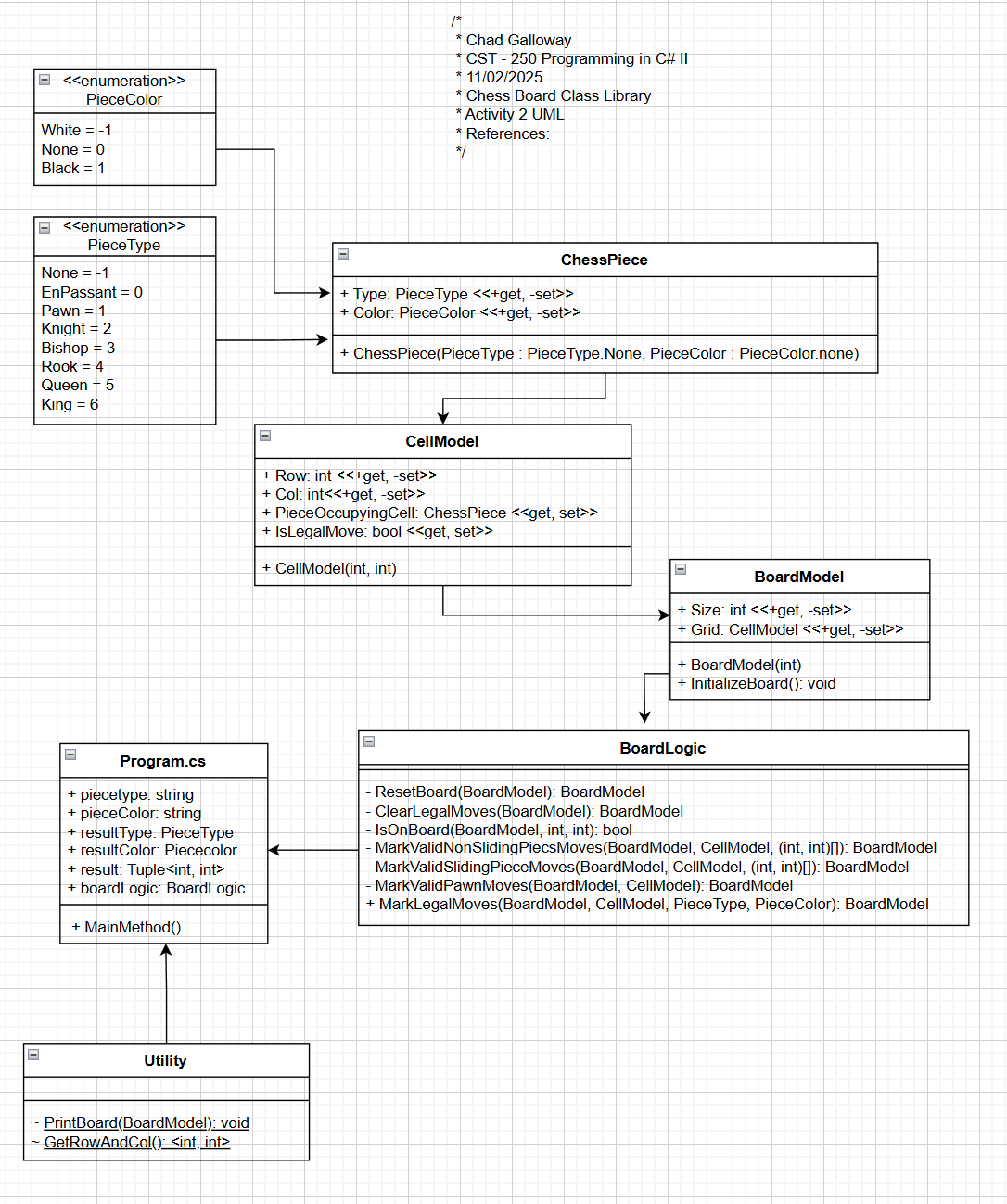


Figure 28: Current state of the application’s UML

ADD ON

**Follow up Questions**

1. What was challenging?
2. What did you learn?
3. How would you improve on the project?
4. How can you use what you learned on the job?