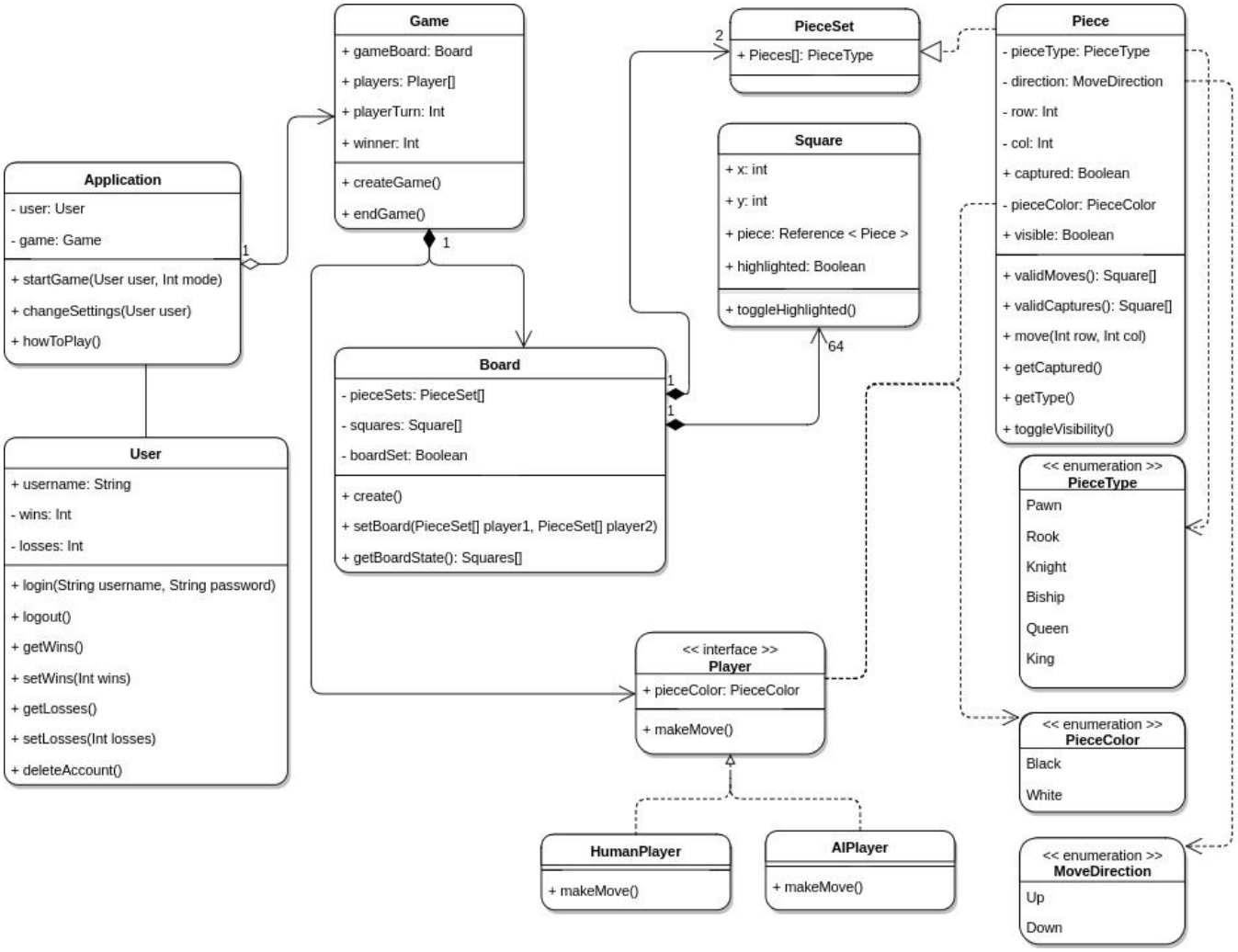
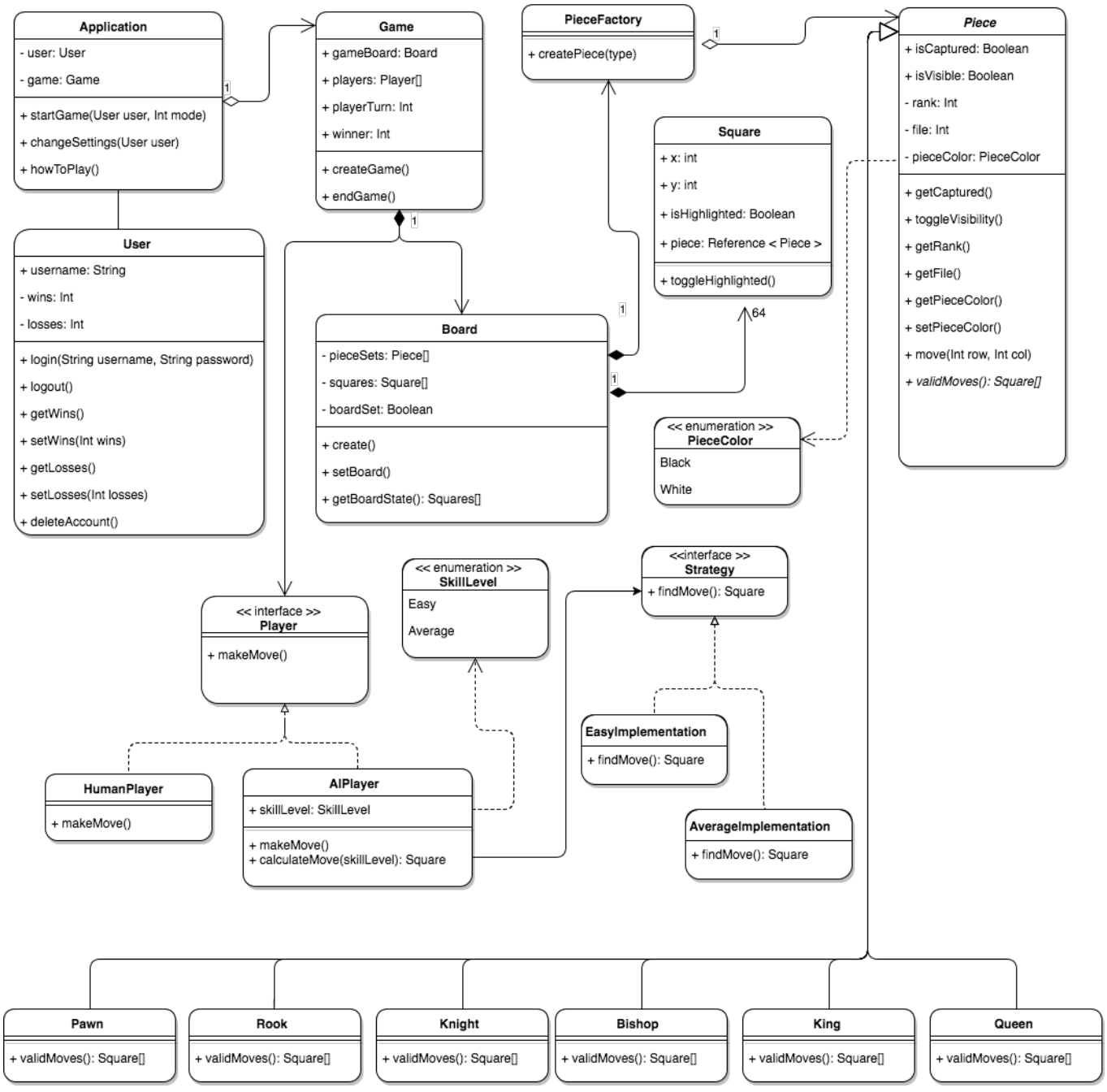
Team: Hunter Leise
Vincent Mahathirash
Raymond Duncan

Title: Chesspionage

Part 2 Class Diagram:

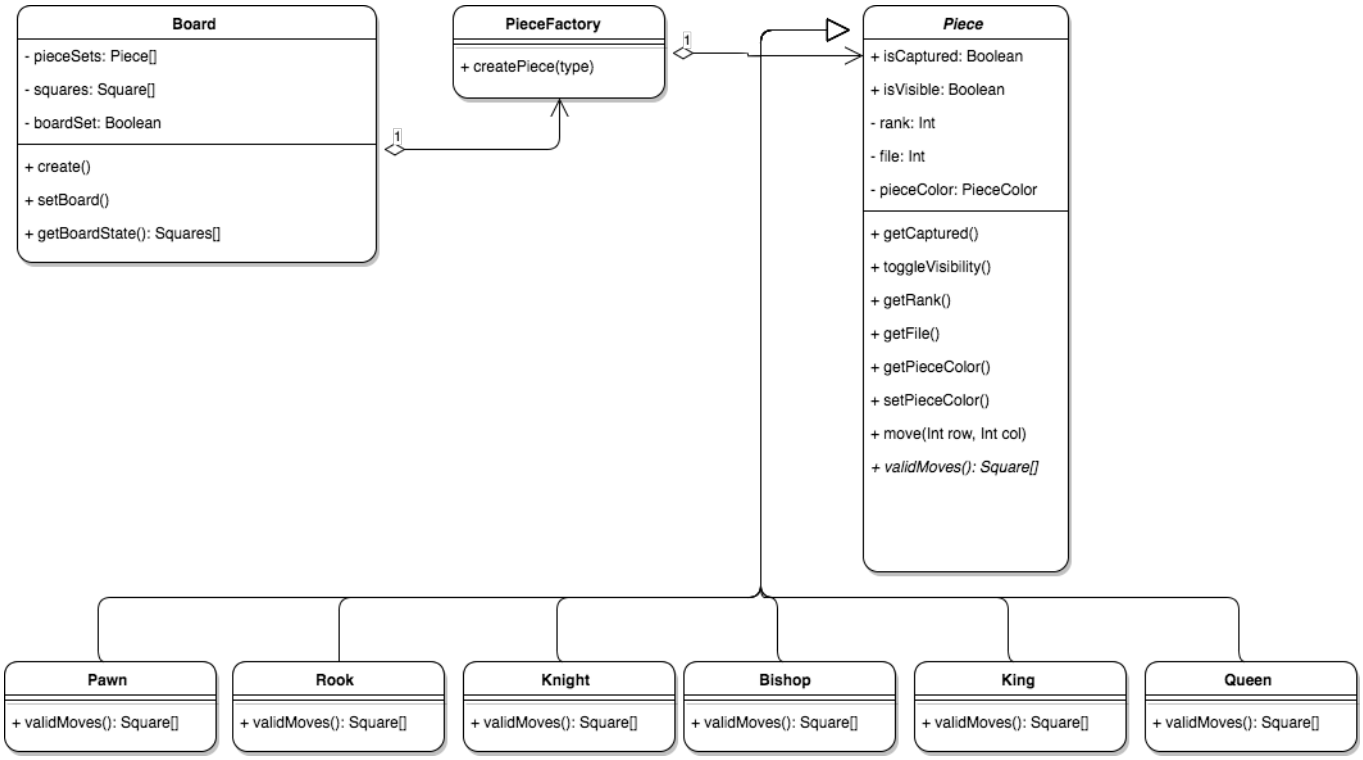


New Class Diagram:

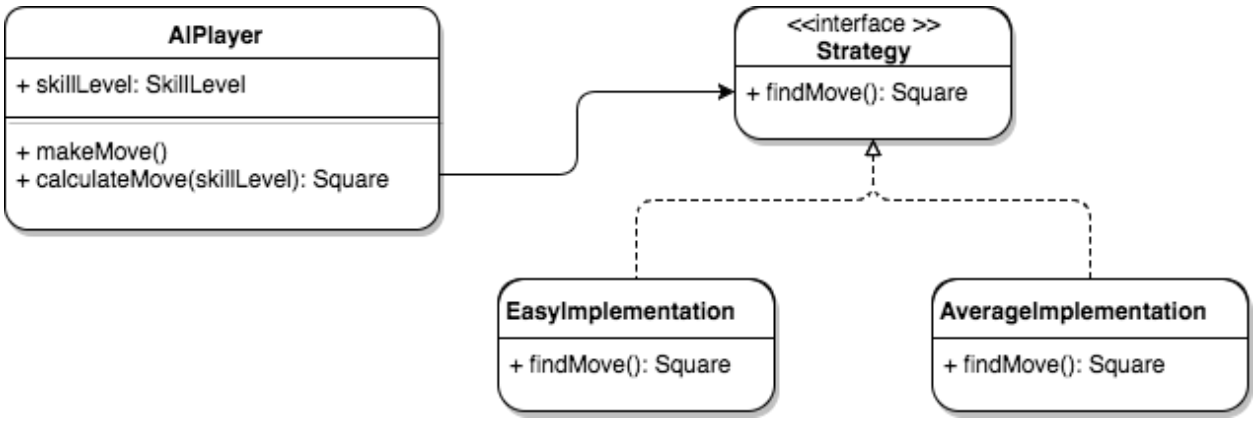


Explanation:

We chose to implement a **factory method** design pattern for the creation of pieces because our original plan of using enumeration didn't easily allow us to customize the validMoves method for each piece type. By using enumeration, we would have needed to use a switch case statement inside the piece class's validMoves method to control how each type of piece moved. By factoring this out into specific piece subclasses, we not only make the code cleaner to read, but we also make it more extensible in the future.



We chose to implement a **strategy** design pattern for the AI player decision making because it allowed us to easily control the AI's difficulty by changing which strategy implementation we use at runtime. In doing so, we can easily extend the AI strategy to include more difficulty levels in the future.



Other than those two design patterns, we chose to get rid of the PieceSet class because it could be easily represented as an array of Pieces in the Board class instead, we added a few getters and setters for private variables that we forgot in our original class diagram, and we removed some unnecessary methods such as ValidCaptures in the Piece class. In addition to those changes, we fixed small issues mentioned in our project 2 feedback such as a few arrow notation errors and interfaces containing attributes.