**SOEN 6441: RISKGAME BUILD 2 DOCUMENTATION:**

**Refactoring**

**GROUP MEMBERS:**  
**40087621 An Nguyen**  
**40104630 Naga Satish**

**List of potential refactoring:**

1. Instead of using “**== null**” or “**!= null**” changed to **Objects.isNull** && **Objects.nonNull.** Here is one example that we have implemented:

Une image contenant capture d’écran

Description générée automatiquement

1. Instead of using “+” for string concatenation we use StringBuilder. Reason: Strings are immutable objects, so concatenation doesn't simply add the new String to the end of the existing string. Instead, in each loop iteration, the first String is converted to an intermediate object type, the second string is appended, and then the intermediate object is converted back to a String. Further, performance of these intermediate operations degrades as the String gets longer. Therefore, the use of StringBuilder is prefered. Here is the function we decided to implement it:

Une image contenant capture d’écran

Description générée automatiquement

1. Instead of **keySet** using **entrySet** for map looping. Reason: When only the keys from a map are needed in a loop, iterating the keySet makes sense. But when both the key and the value are needed, it's more efficient to iterate the entrySet, which will give access to both the key and value, instead. Here is one example:

Une image contenant capture d’écran

Description générée automatiquement

1. Instead of using hard coding strings use constants, only if the constant is used multiple times. Here is the class containing all of the constants:

Une image contenant capture d’écran

Description générée automatiquement

1. Removed the unused parameters.
2. Une image contenant capture d’écran

   Description générée automatiquementIf we have more than 2 conditions to check move the conditions to a separate method. Here is one method we follow this approach:
3. Move the Junit test files to a separate folder. Build 1 we made a mistake by putting the test files in the same folder with the files that need testing. Now we have moved it to separate folders.

Une image contenant texte

Description générée automatiquement

1. In build 1, we implemented the reinforcement, attack and fortification phase by having the RiskBoardModel contains those methods. Now in according to build 2 requirements, we have changed to implement it as methods of Player Class. Here is the UML diagram for Player class:

Une image contenant capture d’écran

Description générée automatiquement