**SOEN 6441: RISKGAME BUILD 1 DOCUMENTATION:**

**GROUP MEMBERS:**

**40088288 Adeyinka Areje**  
**40087621 An Nguyen**  
**40104630 Naga Satish Dwarampudi**  
**40067061 Oluwatosin Olaleye**

**Table Of Contents**

**Scope.................................. 3**

**Objective............................. 3**

**System design...................... 4**

**Coding Conventions............. 5**

**Scope:**

The Scope of this project is determined by the build requirement uploaded on Module which is an operational model of the popular Risk game.

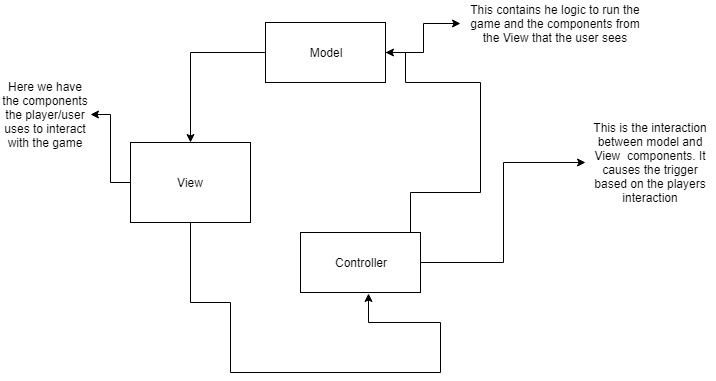
**Objective**

The Objective of this project is to simulate a real scenario of a project build from design to implementation. Here we will develop the popular risk board game. An initial simplified project requirement has been given which we as team have analyzed.

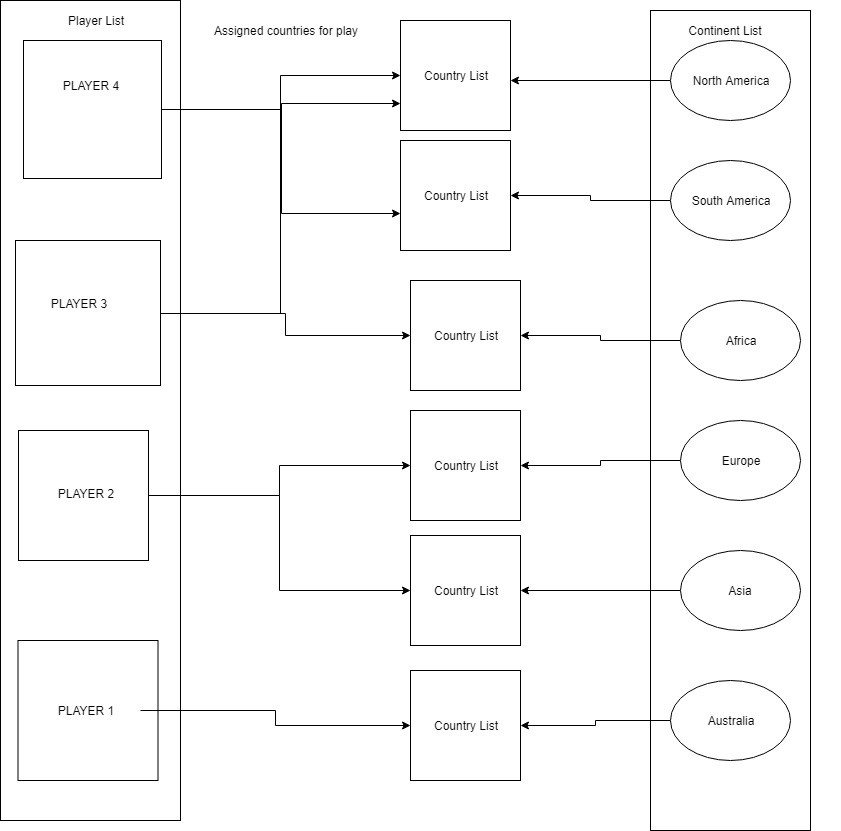
**Software Design and Architecture**

For this project the Adaptive/agile software development model was used simply due to the limited specifications of the project build, the intent to have a working model as soon as possible and also because of the collaborative efforts of the team members to ensure a positive feedback and a working build.

For the architecture, the MODEL VIEW CONTROLLER pattern was used for the gui and it is simplified in the diagram below:



Further information on the architectural design implementation is also showed in the below diagram:



The system architecture can be summarized as thus: At the load of the game, each player is assigned the number of countries, the UI allows each player to know the countries adjacent to him that can be attacked. They start of the game is also where continents with their country list are assigned armies.

**Coding Convention:**

This refers to the set of rules and guidelines to be followed when coding as well as other factors like comments, declarations etc. Not only is a good coding convention best practice it also ensures the readability and manageability of the code. As a team the follow coding conventions were agreed upon and used:

1. **Camel case Naming**: All classes and methods written in this project followed a specific Naming format samples include: RiskAddMapController, RiskAddMapController etc
2. **Descriptive commenting**: Throughout the project for every class/method a well-defined comment exist which is evident in the complete API document attached to this document.
3. **Descriptive/Self-explanatory Class/Method Naming**: A sample is the ***createContinent*** class – here this is the class that’s houses the lists of all the continents in the game. This naming pattern is one of many adopted in this project.