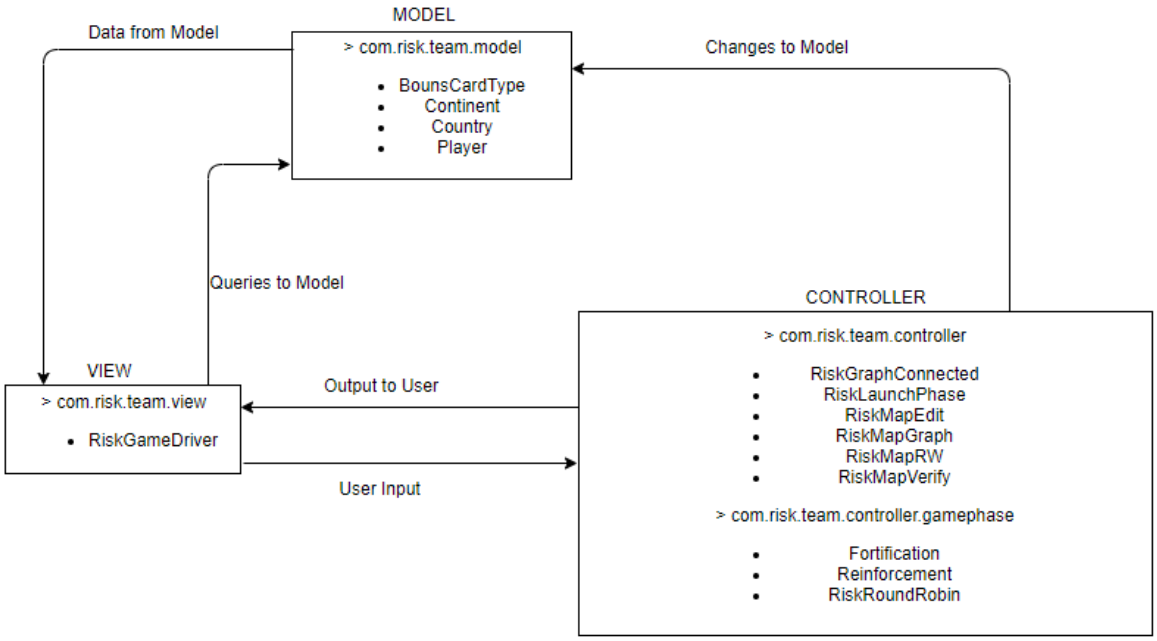
Team 17: SOEN 6441 WW Advanced Programming Practices
Architecture Document

Architectural Design:

Our RISK game is designed based on Model View Controller (M.V.C) architecture. We have divided and distributes our modules in the parts **MODEL, VIEW and CONTROLLER**. Below is the block diagram of our implementation of MVC architecture.



Modules in Model:

We have 4 models described below:

1. Player: Player class is used to model the actual player entities of the game.
2. Country: Country class models country entity, where each player owns few countries in the beginning and their aim is to get ownership of all the countries.
3. Continent: Continent class models Continent entity, which can be considered as a superset of countries, where each country belongs to one particular continent.
4. BounsCardType (Interface): It is used to model cards which are allocated after the attack phase if any player is eligible for the card.

Modules in Controller:

It further has a folder for representing the **gamephase** phases of reinforcement and fortification along with a class to provide player chances in a round robin fashion.

1. RiskMapEditor: This class provides functionality to edit or create a map from scratch.

2. RiskMapGraph: RiskMapGraph creates a connected graph from valid map data and provides methods to modify the graph.
3. RiskGraphConnected: Checks whether a graph is connected or not. In case of fortification , it checks whether there is adjacent path through connection or not.
4. RiskMapRW: This class reads the data from a **.map** file and provides it to RiskMapEditor's object. It also gets data from RiskMapEditor's object and writes it to an empty **.map** file.
5. RiskMapVerify: It is responsible for verifying the correctness of a map which is to be loaded for game play.
6. RiskLaunchPhase: It takes data from RiskMapRW, and initializes data for Players, countries and armies. Here countries are randomly assigned to each player.
7. Gamephase: It has the following classes.

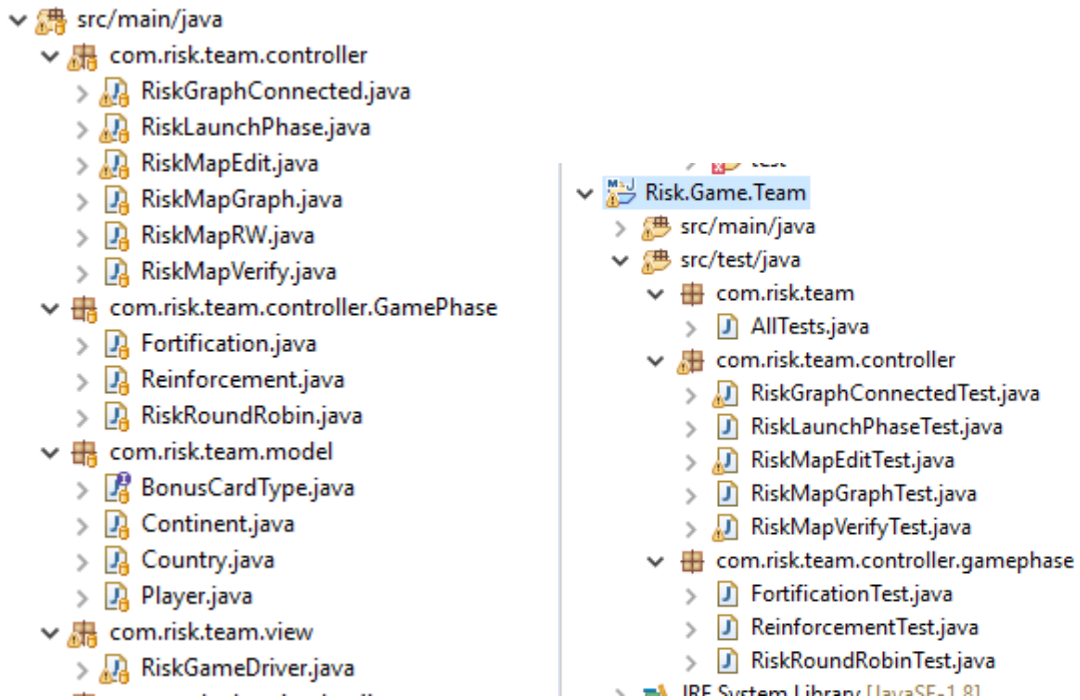
   Reinforcement: It has methods to get the number of armies calculated for assignment to each player.

   Fortification: It provides method to pass the armies form one country to another.

   RiskRoundRobin: It provides functionality for round robin turn traversal among the players.

Modules in View:

1. RiskGameDriver: Provides an interface for the user to interact with the game. It launches the main window of the game to load, create and edit the map file.



Same hierarchy is followed for **Test Folder** and **TestClasses**.There is one to one Mapping of Each Java Tested Class to Test Class.