Advanced Programming Practices SOEN 6441- Winter 2024

RISK GAME

PROJECT BUILD 1

Submitted By :-

Sanjay Bhargav Pabbu - 40265895

Blesslin Jeba Shiny Augustin Moses - 40266442

Susmitha Mamula - 40277873

Poojitha Bhupalli - 40232528

Piyush Gupta - 40294464

Mahfuzzur Rahman - 40293992

Objective

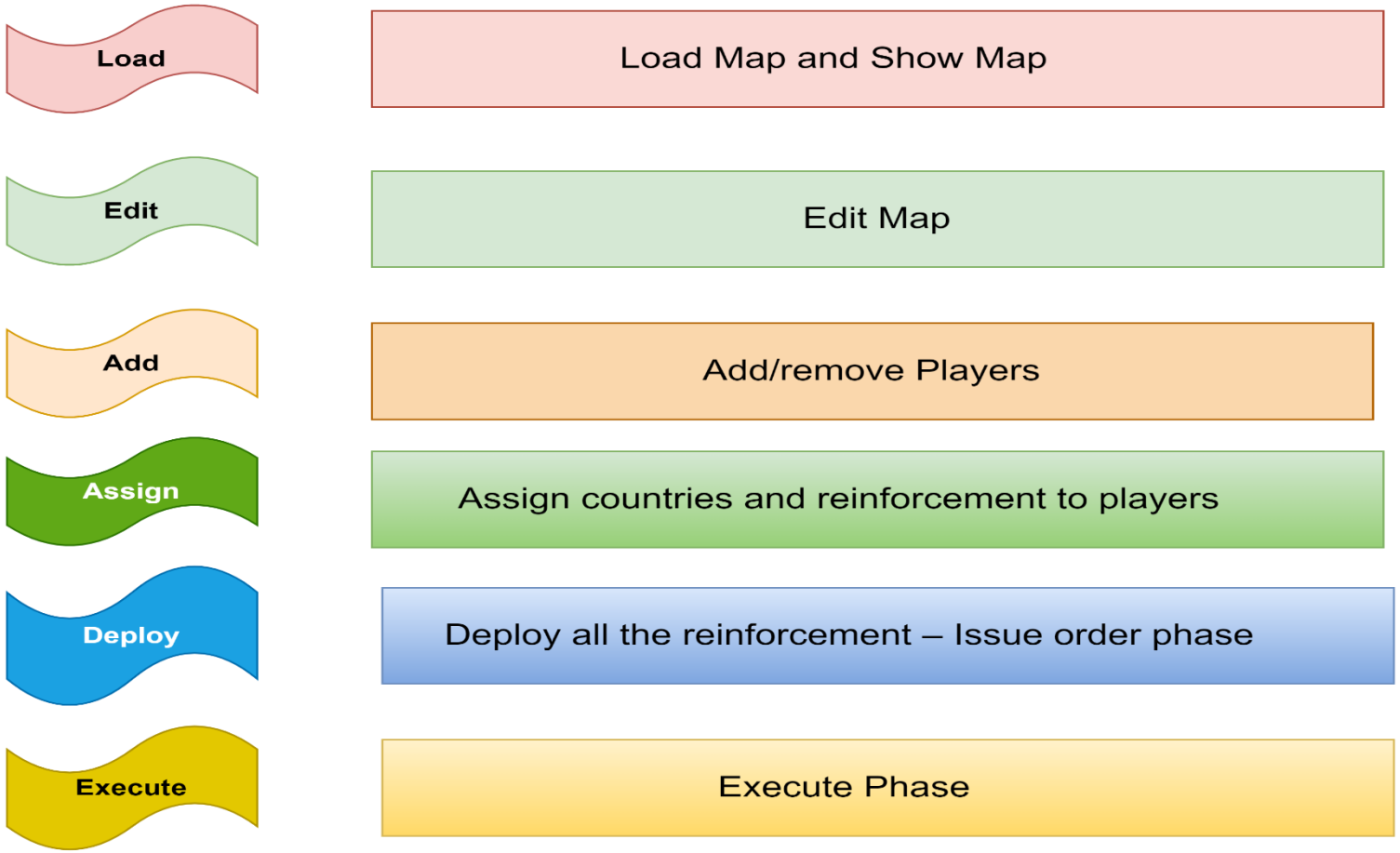
To create a Java application that complies with the "Warzone" version of the Risk game's rules, map files, and command-line play capabilities.

The game is divided into three phases: launch, main gameplay, and execute, which manages the whole process. Users of the game can import pre-existing maps, make new ones, and modify existing ones to add or remove continents, nations, nearby nations, and players.

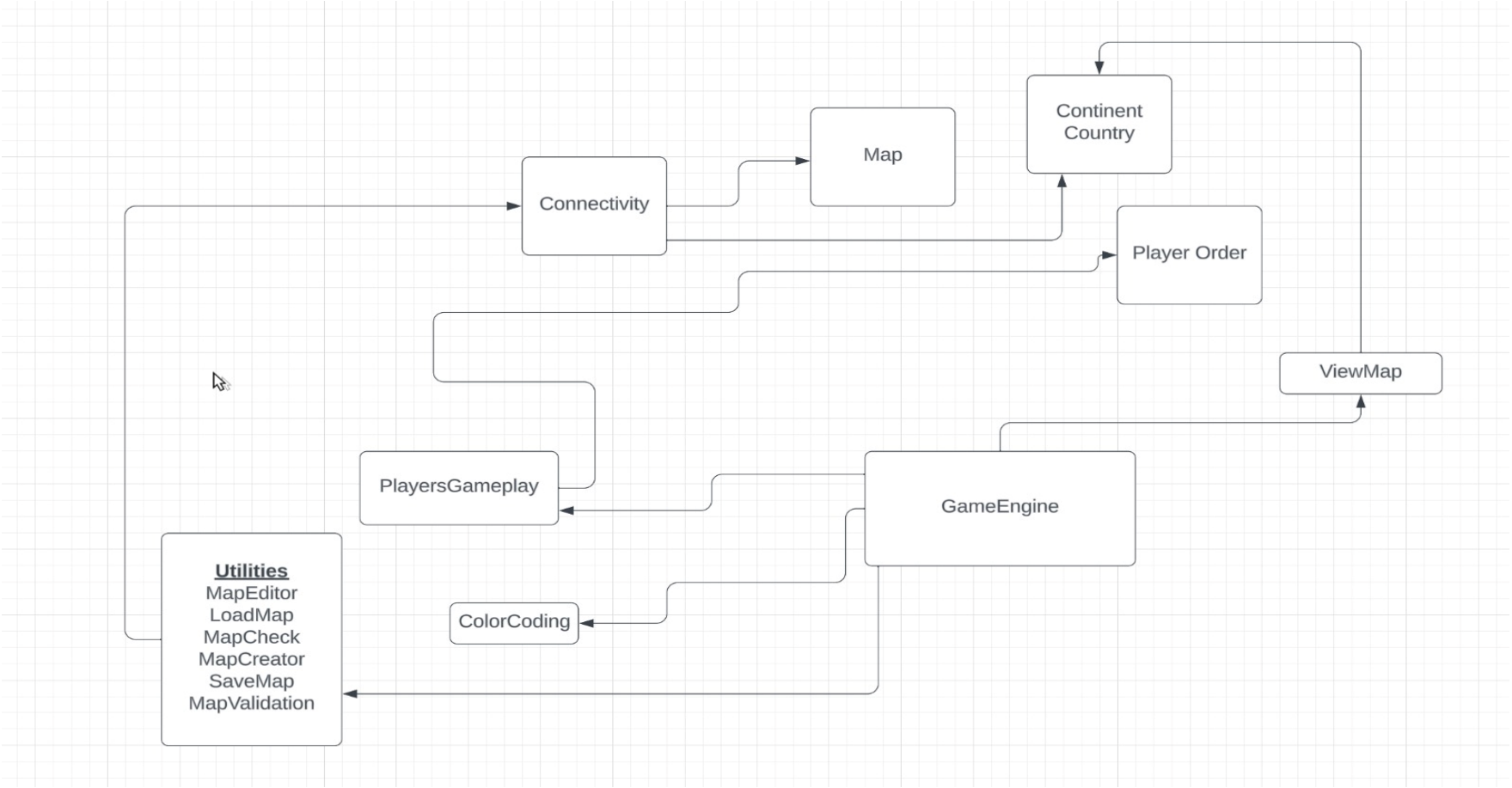
Players in the main game loop would be assigned countries to issue and carry out orders.

Until they place all of their armies, players issue deploy orders in a round robin fashion. In the execute phase, player orders to deploy armies to a country's CountryID are carried out.

Flow of Execution



Architecture Diagram



MVC Architecture

Our project follows MVC architecture to organize the structure of the game

Model : Includes data for Country, Continent, Player, Order and Map containing state and behavior that can be accessed across entire architecture of the project.

Utilities: Contains methods to perform startup, main game loop and execute phase.

View: Used to show the existing or updated map to the users in console. ShowMap displays Country, Continent, Neighboring countries and Total number of armies in tabular format.

Controller: Acts as an intermediary between Model and View which takes user inputs via the View, processes it by interacting with Model and then updates the view to reflect the result. GameEngine is the center of the game that controls the entire functionality of the game

Javadoc

Javadoc documentation on all the classes, data members, and member functions has been implemented to generate comprehensive and easily navigable documentation in HTML format.

Syntax:

```
/**
 * Javadoc comment
 *
 */
```

@param	provide information about method parameter or the input it takes
@see	generate a link to other element of the document
@version	provide version of the class, interface or Enum.
@return	provide the return value
@author	Describes an author
@throws	Throws exception if something goes wrong

Junit

Contains around 25 test cases covering the key game test scenarios listed below:

- Evaluate the "Assigncountries" feature's correctness.
- Verifies that modifying map commands such as adding or deleting countries, continents, and neighbors and checks if it works as intended.
- Verifies accurate distribution of armies among participants.

Thank You