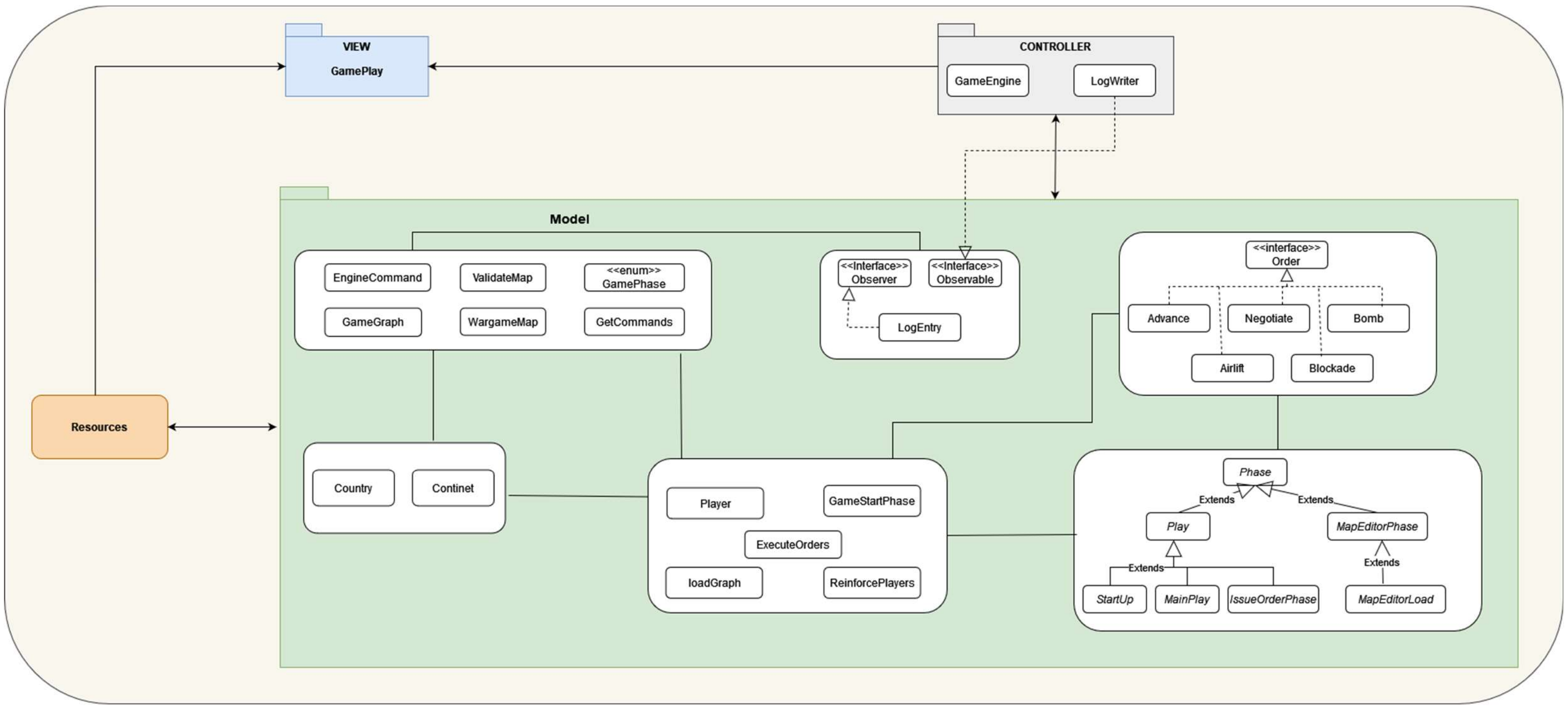
ARCHITECTURE – BUILD 2



The UML package diagram above is the architecture diagram of the wargame command line project created for Build 2. In this build we added additional functionality on top of build1 and as specified in the build 2 requirements document, we refactored the code to include 3 design patterns as given below:

- State Pattern
- Observer Pattern
- Command Pattern

Description of the architecture diagram is given below:

BEGINGAME Phase: The `GamePlay.java` file serves as the main entry point for the Risk game, initializing it with two valid commands: `editmap` or `loadmap`. These commands trigger the `GameEngine.java` class, which parses user commands and calls relevant functions to execute the command.

The game then follows **State Design Pattern** where depending on users command respective State of the Game is set.

MapEditor State:

1. **editmap:** Opens existing map files or creates a new map. Utilizes `RunCommand.java` to parse and implement commands related to editing continents, countries, and neighbors. Save and load map commands are available to persist changes.
2. **loadmap:** Calls the `loadMap()` function in `RunCommand.java` to load a validated map, transitioning the game to the STARTUP phase.

Play State:

STARTUP Phase**:** Players are added or removed as per user commands, and countries are allocated. Commands include `showmap`, `gameplayer -add playerName -remove playerName`, and `assigncountries`.

MainPlay State:

AssignReinforcements**:** After players are added and countries assigned, `AssignReinforcements` calculates armies for each player based on owned countries. The round-robin turn sequence begins for players to deploy armies on their countries.

TAKETURN **Phase:** Implements round-robin turns for players to issue orders one at a time. After each player's ISSUE_ORDERS phase, `TURNEND` is triggered to move to the next player's turn.

EXECUTEORDERS **Phase:** Once all orders are collected, the Execution Phase sequentially executes the first order from each player in a round-robin fashion. It checks for a winner if all territories are conquered and removes players if they lose all territories.

ISSUEORDER State:

ISSUE_ORDERS Phase: Players issue orders to the game, with orders collected only if valid. Corresponding army units are reduced upon collection. Players can provide orders and halt to proceed to the Execution phase.

The **Command Design Pattern** is implemented as specified, where in we have defined Order Interface that has execute() method and that is implemented by respective type of orders such as Advance, Negotiate/Diplomacy, Bomb, AirLift and Blockade. These commands are called by GameEngine and Player is the invoker class

The **Observer Design Pattern** is implemented by defining interface Observable and Observer that are implemented by LogWriter and LogEntry class respectively. LogEntry is invoked in GameEngine class.