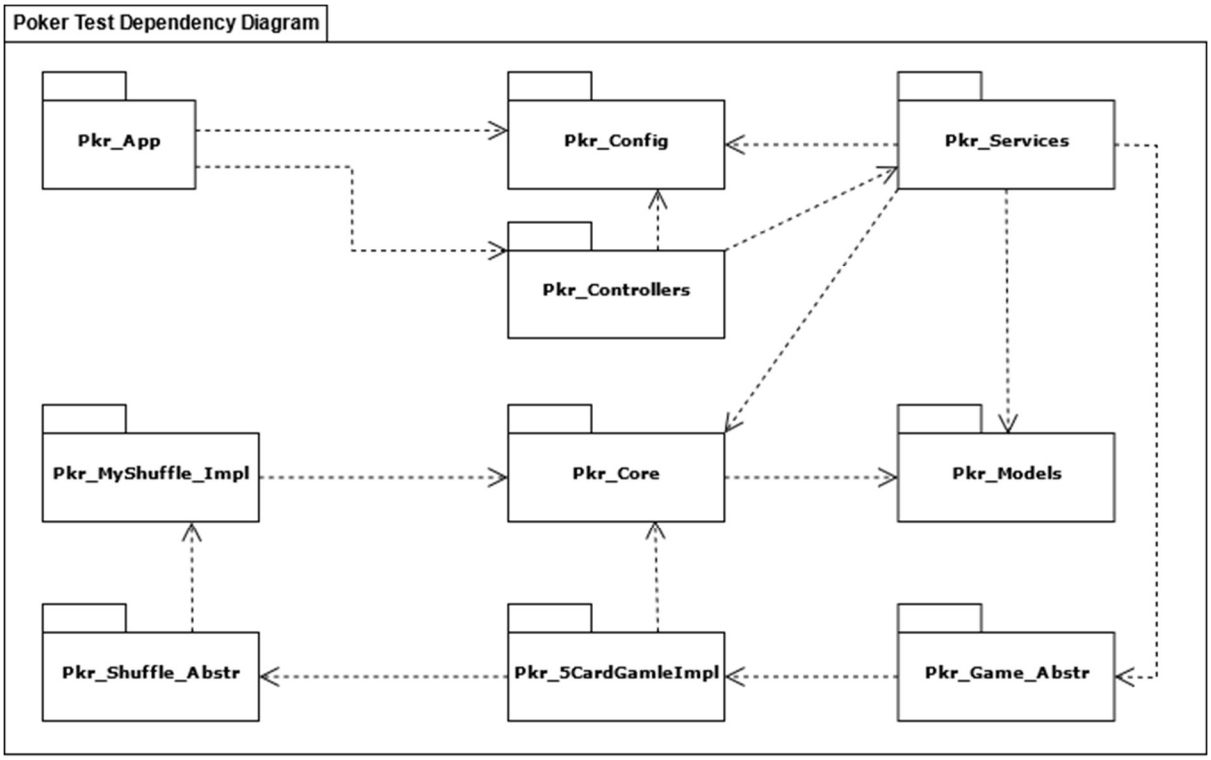
Poker Test – Technical Assessment

Write Up on my Design

Technologies use:

- Java 11
- Spring Framework version 5.3.4
    - Spring Dependency Injection
- Maven
    - Running and building application
- Bash Scripts
    - To automate running and/ or running the application on nix OS environments



Note On My Design:
I had chosen to build this solution using a layered approach and keeping it modular. I grouped the components into module splitting my solution into a few of layers, Application, Controller, Services and Data layers.

I have kept this solution, stateless, which would be important if we chose to convert it into a REST API or a Web Application that could server up multiple users i.e., game instance simultaneously.

I implemented a static map to store different game states against a uniquely generated gameId.

Thus, we will be able to store in-mem / cache different in future iterations. This static map (GameStore) to some degree mimics a caching /in-mem database and can easily be swapped out for one.

I have attempted to make this solution extendable and decoupled, by applying Open-Close, Single Responsibility and Separation of Concern design principles. I did this by decoupling the specific game 5Card draw and the shuffling modules from the rest of the solution by using “abstract modules” and clearly defines interfaces.

This allows me to easily swap out my game implementation for a different game (poker variant single player) or add a new one. Similarly, I can swap out my specific shuffling module for another implementation, (maybe a an improved one).

Therefore, as long as the new implementations adhere to my interfaces it would ensure consistency with my general design and would allows for a decoupled and easily extendable solution and that may be converted to a REST API or a Web Applications able to serve multiple games simultaneously, in the future.

Unit Tests:
I have included sets of unit tests on the Controller & Services layers / modules to test all the functionality required for running and evaluating games, this includes, dealing hands, shuffling logic, and generate gameIds.

I have also created a set of unit tests in my game implementation module to test each of the ranking rules for that specific game implementation (5Card Draw).