Design Document for: Quoridor

Group TZ\_4

#### Member 1: Carter Awbrey

#### Member 2: Mazin Bashier

#### Member 3: Seth Leon

#### Member 4: Ellery Sabado

## Block Diagram

## Design Explanation

**Frontend:**

In the Login Page, the user will be able to enter a username and password. The user has the choice to either create a new account or login to an existing account using the given information. When clicking create an account, the userID, password are sent as a request to the server as a POST.

In their Game History sers will be able to see their previously played games, moves made in previously played games, and previous opponents. Requests will be sent as POST.

In the Leaderboard, users will be able to see the moves made in a game, previous opponents played. Additionally, users will be able to see everything in their Game History, and the ranking between the players. Requests will be sent as POST.

We separated the UI and the logic of the game for the project. Every UI layout with its appearance has a UI that corresponds to it. We have 2 customized layouts that are for the Board and Chat features.

We also have a different layer for data processing and control. We have a Util layer that is responsible for handling the Json requests. These 2 work together to get everything we need without messing anything else up.

**Backend:**

All of the server’s work begins whenever it receives a packet of information via Spring Boot’s controller mapping. These packets of information convey what specific action the player wants to perform, the data required to complete that action, and extra data that almost all of the packets require. The packets, sent as JSONs, are converted into a super-packet that then reads the action and data, converting it into and executing a sub-packet.

Sub-packets are how client requests are converted into changes server-side. In addition, an interaction layer between the networking and packets and the main game services helps simplify the creation and understanding of the sub-packets. With the interaction layer, the sub-packets are able to act upon and return the inner workings of the server and games without being overly long or complicated.

Finally, as these sub-packets are being executed, any changes to the database are saved via the repositories, obtained by passing them through the sub-packets. Similarly, important game data that needs to be updated client-side are sent to players via WebSockets, created when they join a game.

The server allows clients to perform actions through these elements, ranging from logging on to playing the game.

## Table Relationships

