PROJECT Design Documentation

> *The following template provides the headings for your Design Documentation. As you edit each section make sure you remove these commentary 'blockquotes'; the lines that start with a > character and appear in the generated PDF in italics.*

Team Information

- Team name: s1-d-labrats
- Team members
  - Mike White
  - David Allen
  - Will Peterson
  - Sokol Nguyen
  - Rafeed Choudhury

Executive Summary

This is a summary of the project.

Purpose

The purpose is to allow Checkers players to play a game of Checkers online.

Glossary and Acronyms

> *Provide a table of terms and acronyms.*

Term	Definition
VO	Value Object

Requirements

This section describes the features of the application.

> *In this section you do not need to be exhaustive and list every story. Focus on top-level features from the Vision document and maybe Epics and critical Stories.*

Definition of MVP

> *Provide a simple description of the Minimum Viable Product.*

MVP Features

> *Provide a list of top-level Epics and/or Stories of the MVP.*

Roadmap of Enhancements

> *Provide a list of top-level features in the order you plan to consider them.*

Application Domain

This section describes the application domain.

Our domain model represents a standard game of Checkers. Two players play a game of Checkers. The game is played on a board, which contains 64 squares. Each player controls 12 pieces which are each placed on a

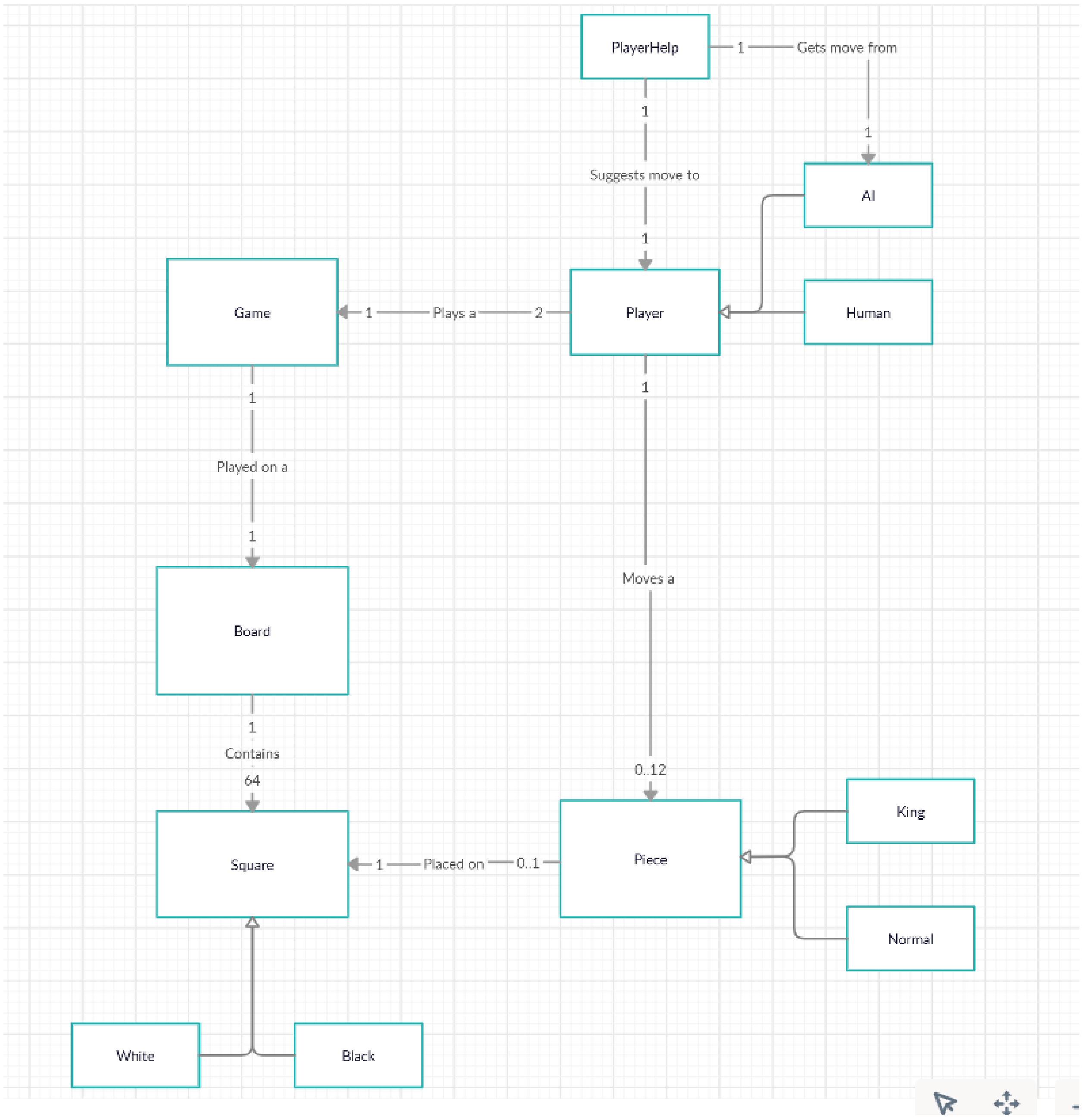

Figure 1: The WebCheckers Domain Model

square, which has to be black. The pieces can either be single pieces or king pieces. King pieces can move in all four directions and single pieces can only move forward diagonally. The player can choose to play against the AI. While playing against the AI, the player can also get help to pick the best moves.

Architecture and Design

This section describes the application architecture.

Summary

The following Tiers/Layers model shows a high-level view of the webapp's architecture.

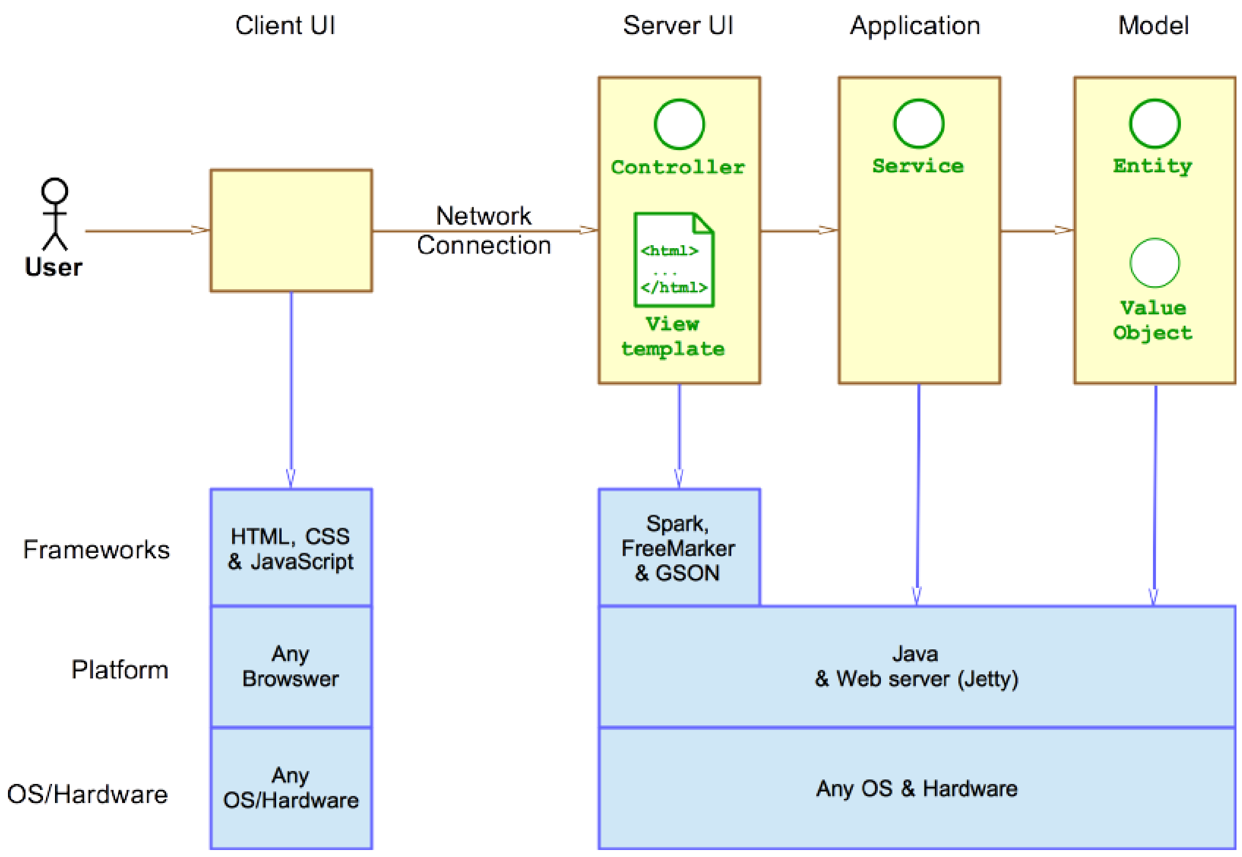

Figure 2: The Tiers & Layers of the Architecture

As a web application, the user interacts with the system using a browser. The client-side of the UI is composed of HTML pages with some minimal CSS for styling the page. There is also some JavaScript that has been provided to the team by the architect.

The server-side tiers include the UI Tier that is composed of UI Controllers and Views. Controllers are built using the Spark framework and View are built using the FreeMarker framework. The Application and Model tiers are built using plain-old Java objects (POJOs).

Details of the components within these tiers are supplied below.

Overview of User Interface

This section describes the web interface flow; this is how the user views and interacts with the WebCheckers application.

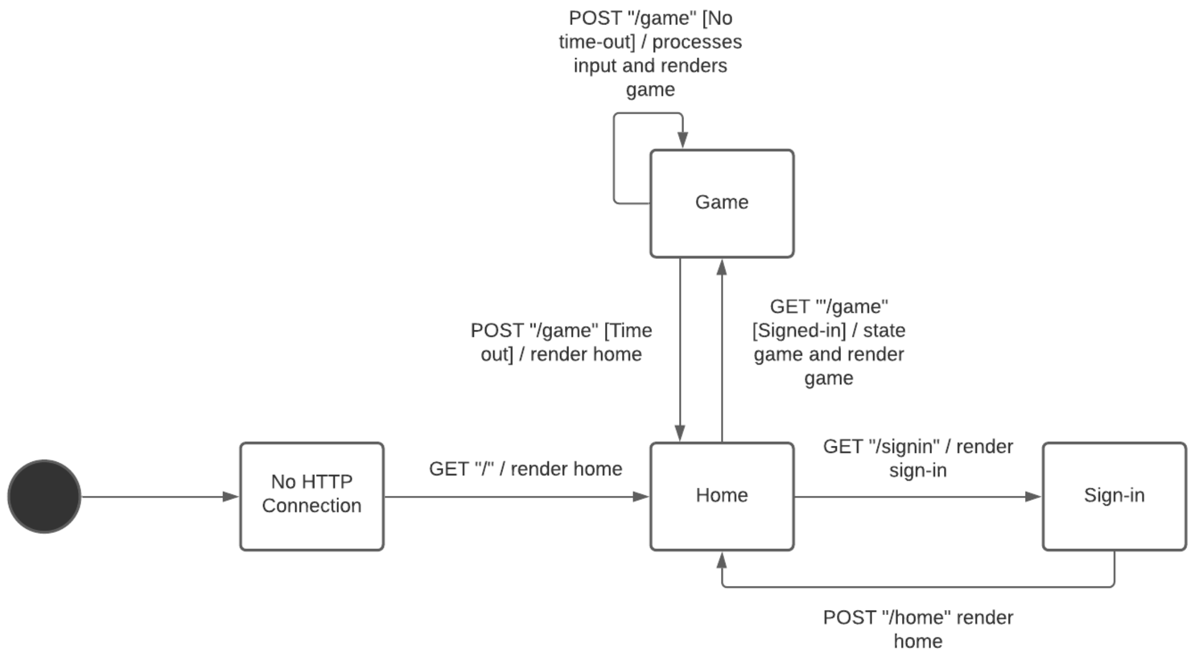
POST "/game" [No time-out] / processes input and renders game
Game
POST "/game" [Time out] / render home
GET ""/game" [Signed-in] / state game and render game
No HTTP Connection
GET "/" / render home
Home
GET "/signin" / render sign-in
Sign-in
POST "/home" render home

Figure 3: The WebCheckers Web Interface Statechart

When the player first goes to Web Checkers, there's a button to sign in. When the player clicks on that, they're sent to the Sign-In page. Once they've entered a valid username, they're taken back to the Homepage. There, they'll see the names of other players they can play against. Clicking one of those will take them to the game. When the game ends, they go back to the Homepage.

UI Tier

Provide a summary of the Server-side UI tier of your architecture. Describe the types of components in the tier and describe their responsibilities. This should be a narrative description, i.e. it has a flow or "story line" that the reader can follow.

At appropriate places as part of this narrative provide one or more static models (UML class structure or object diagrams) with some details such as critical attributes and methods.

You must also provide any dynamic models, such as statechart and sequence diagrams, as is relevant to a particular aspect of the design that you are describing. For example, in WebCheckers you might create a sequence diagram of the `POST /validateMove` HTTP request processing or you might show a statechart diagram if the Game component uses a state machine to manage the game.

If a dynamic model, such as a statechart describes a feature that is not mostly in this tier and cuts across multiple tiers, you can consider placing the narrative description of that feature in a separate section for describing significant features. Place this after you describe the design of the three tiers.

Application Tier

Provide a summary of the Application tier of your architecture. This section will follow the same instructions that are given for the UI Tier above.

Model Tier

Provide a summary of the Application tier of your architecture. This section will follow the same instructions that are given for the UI Tier above.

Design Improvements

Discuss design improvements that you would make if the project were to continue. These improvement should be based on your direct analysis of where there are problems in the code base which could be addressed with design changes, and describe those suggested design improvements. After completion of the Code metrics exercise, you will also discuss the resutling metric measurements. Indicate the hot spots the metrics identified in your code base, and your suggested design improvements to address those hot spots.

Testing

This section will provide information about the testing performed and the results of the testing.

Acceptance Testing

Report on the number of user stories that have passed all their acceptance criteria tests, the number that have some acceptance criteria tests failing, and the number of user stories that have not had any testing yet. Highlight the issues found during acceptance testing and if there are any concerns.

Unit Testing and Code Coverage

Discuss your unit testing strategy. Report on the code coverage achieved from unit testing of the code base. Discuss the team's coverage targets, why you selected those values, and how well your code coverage met your targets. If there are any anomalies, discuss those.