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: Banana
- Team members
  - Hersh Nagpal
  - Michael Kha
  - Luis Gutierrez
  - Matthew Bollinger
  - Christopher Daukshus

Executive Summary

WebCheckers is a web-based version of the game of checkers built using the Spark web framework and the FreeMarker template engine that is run on Java8.

Purpose

The goal of this project is to have a functional application that allows users to play a game of checkers with other users. Users select an opponent or wait to be selected to begin a game within the player lobby. Players can expect to play under the American rules for checkers until a player wins or resigns.

Glossary and Acronyms

Term	Definition
VO	Value Object
MVP	Minimum Viable Product
UI	User Interface

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

WebCheckers is an application in which players can challenge each other to checkers games over the internet. Players will be able to log in to a website and see a list of other players who are online. Clicking a player will challenge them to a game of checkers. If they accept, a game of checkers will be created. The game will follow the regulations of American Checkers. Players can resign at any time.

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.

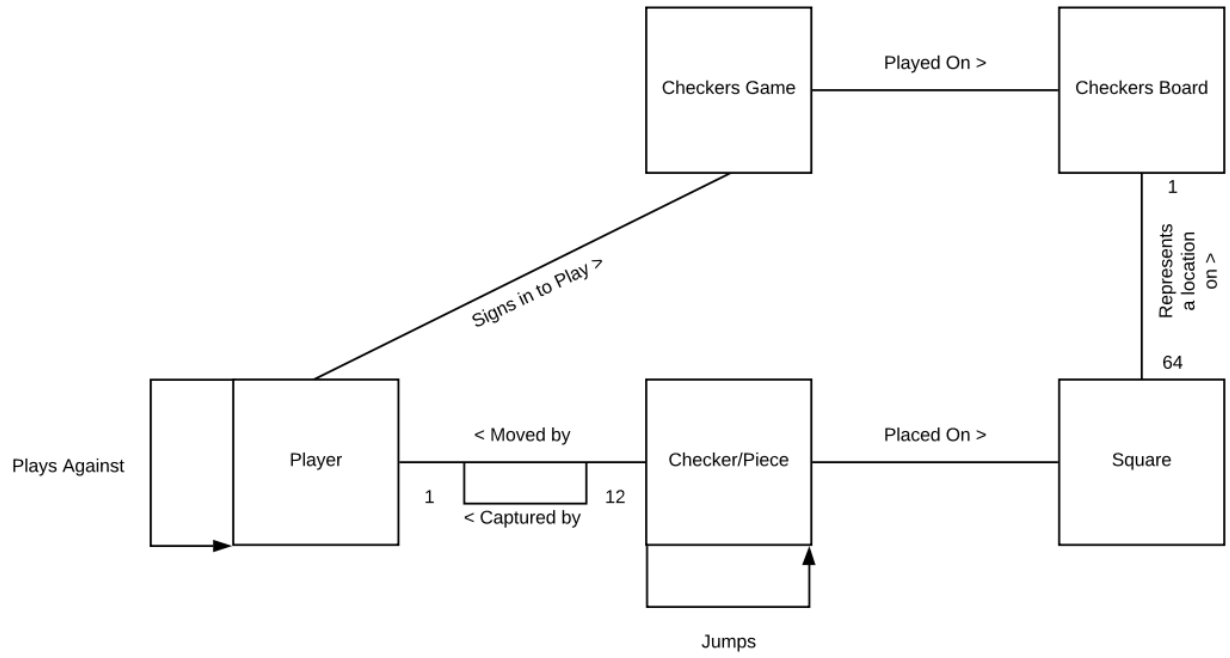

Figure 1: The WebCheckers Domain Model

Players sign in to play the game of checkers. The game is played on a standard 8x8 checkers board. The checker pieces are located on individual spaces. Players play against another player, taking turns moving pieces and capturing their opponents pieces.

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.

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.

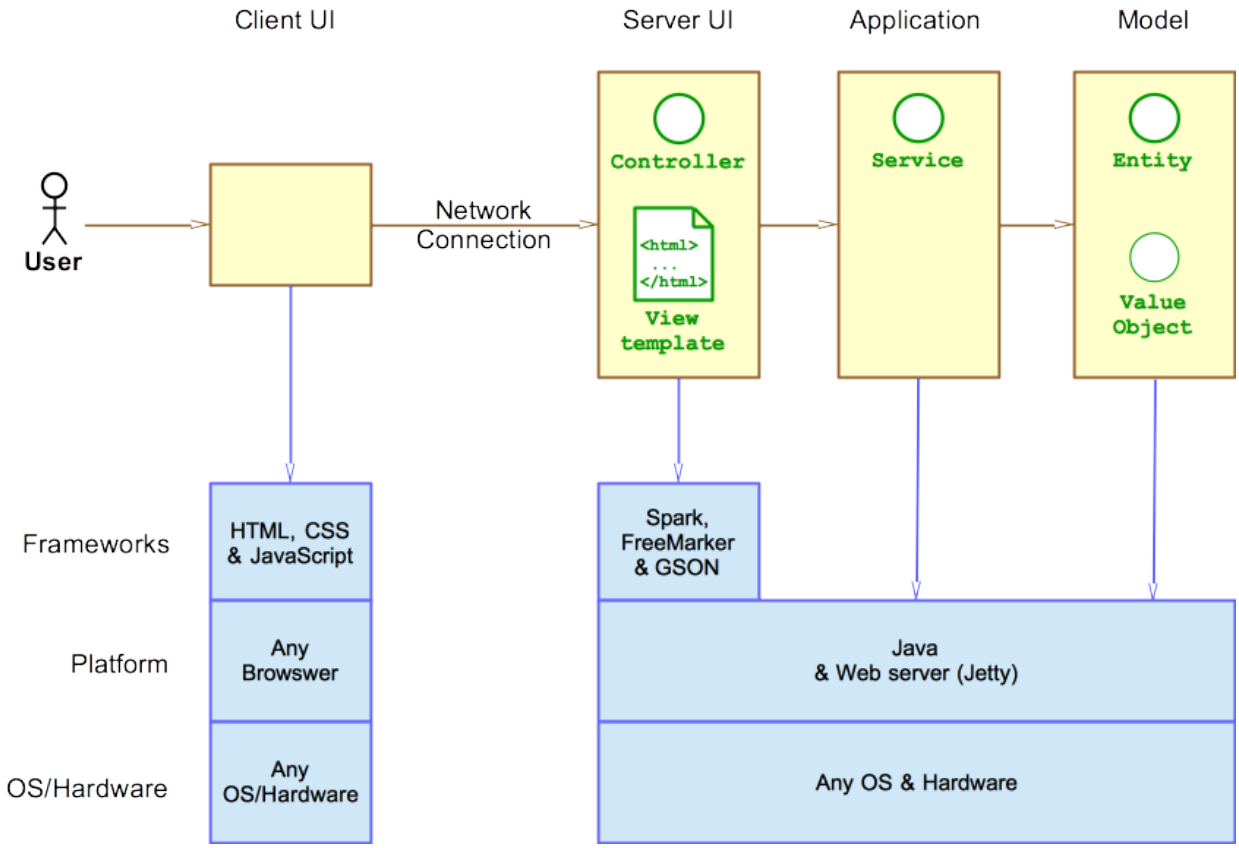

Figure 2: The Tiers & Layers of the Architecture

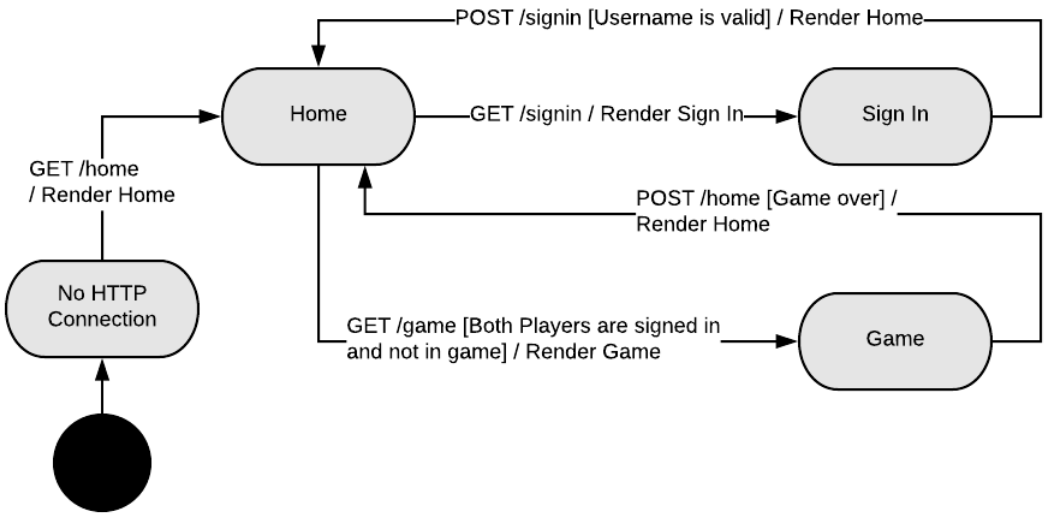

Figure 3: The WebCheckers Web Interface Statechart

Users can expect to be connected to the home page where they will have the option to sign in. Upon visiting the sign in page, users are prompted to enter a username. A username must follow specific criteria to be valid. Once a user has signed in, they will be able to see other players that are signed in on the home page. The user can start a game by selecting another player that is not yet in a game. The user that starts the game gets to go first and each player takes turns making moves until a winner has been decided. When a game of checkers is over, both users will return to the home page.

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.