Web Checkers Project Design Documentation :

Team Information:

- Jose Reyes Arias
- Moayad Alshangiti
- Harsha Vardhan Muppuri
- Murtaza Tamjeed
- Kale Arpita

Executive Summary

Web-checkers is a web application developed for users to play game of checkers. The game user interface supports drag-and-drop browser capabilities for making moves.

Purpose

Web-checkers allows users to play game of checkers with other players who are currently signed. Player should sign in with username and select an opponent,then play the game following american rules.

Glossary and Acronyms

Term	**Definition**
UI	user interface
Checkers	Board game for two players,which involve diagonal moves
Opponent	Challenged player
Move	Diagonal movement of pieces on board
Capture	Remove opponent’s pieces

Requirements

* Player must be able to sign-in with username before playing a game.
* Player must be able to play a game of checkers based upon the American rules.
* Player should be able select and challenge a player
* Player should be able to move the pieces on the board
* Player should be to capture the opponent piece
* Player should be able to forfeit the game
* Player should be able to sign out from the game

Definition of MVP

MVP for this project is that user should be able to sign-in,play against the opponent and should be able to resign the game

MVP Features

The minimal viable product includes below features:

1. Every player must sign-in before playing a game.
2. Two players must be able to play a game of checkers based upon the American rules.

3.Either player of a game may choose to resign, which ends the game.

Road map of Enhancements

Beyond the minimum features mentioned above, following are possible enhancement features to the system:
* Games can be stored for later review.
* Players can play asynchronously.
* A player may play more than one game at a time.
* Other players may view an on-going game that they are not playing.
* Players may play a game against an AI player.
* Players can play in checkers tournaments.

Application Domain

Basic checkers game is designed to play using American rules. Diagram 1 represents the application domain of the model. User can play against an opponent after signing In. With valid username, user is navigated to select a player page to challenge an opponent and start the game. Each player has a turn which is validated before every move and is switched between the players.

Architecture

Summary

Below diagram shows a high level view of application architecture :

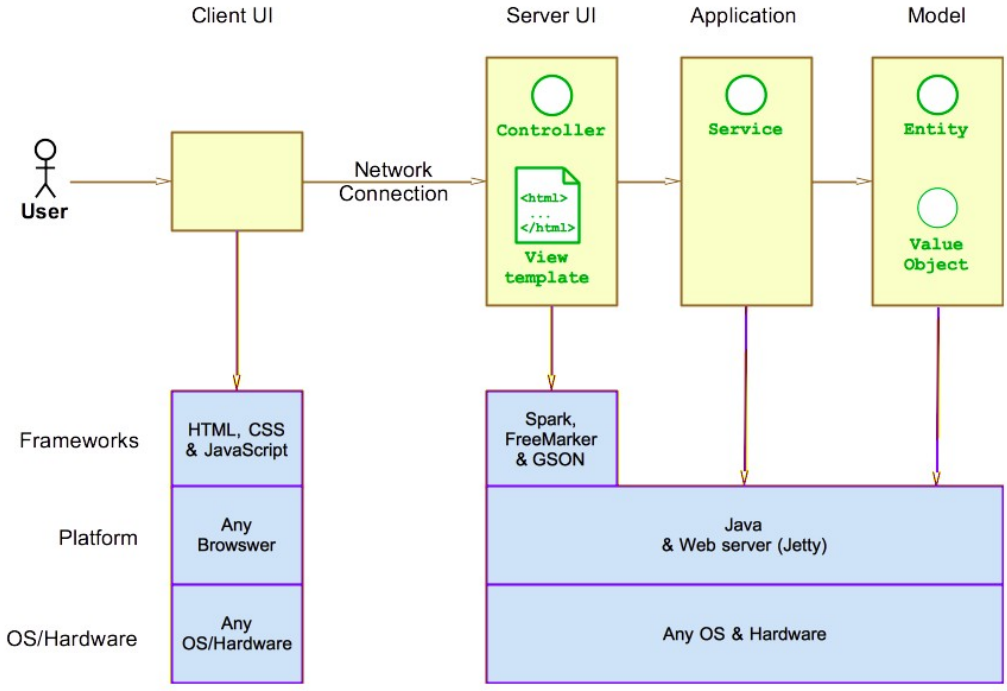

Diagram : Overview of application Interface

As a web application user interacts with the web appication. The User Interface is composed of HTML pages, CSS for styling of the page and javascript on the client side.The server side UI contains controllers and views. controllers are built using spark framework, and views are built using Freemarker framework.

Overview of User Interface

Game UI includes routes to different pages so that corresponding view templates are generated.
Current game implementation includes Webserver, GetSignInRoute, GetSignOutRoute, HomeController, PostSignInRoute and SignInController.

Webserver :

This route initializes all the HTTP routes and each route can be called with respective URLs.

GetSignInRoute :

This route provides sign In page for the user. It allows the to enter the user name and it is a mandatory step for the user to play the game.

GetSignOutRoute :

This re-routes the user to home page,when user clicks on sign out button. No page is generated instead user is navigated back to the homepage.

HomeController :

This route is basic controller of home page. Routes to sign in page if user is not signed in and to game page if already sign in.

PostSignInRoute :

This route accepts username input from user. This name used to create instance,verify username and allow user to play the game.

SignInController :

This route provide the sign in page for the user and user can enter the username to login.

Overview of Application Tier

This model tier has Game Center class that is capable of adding user to the game,verify if the user is already logged in and removing the user from the game when signed out. This information is stored and passed on to the UI tier classes.

Overview of Model Tier

This tier has the class Player that is needed to play the checkers game.

Player :

This class represents a player playing a game. It creates a new player with the user name selected by the player and also validates if the username according the constraints mentioned in the story.

Static models

Refer to diagram 2 for the UML class diagram of the web application

Dynamic models

Refer to diagram 3 for the state diagram of the web application

Diagrams :

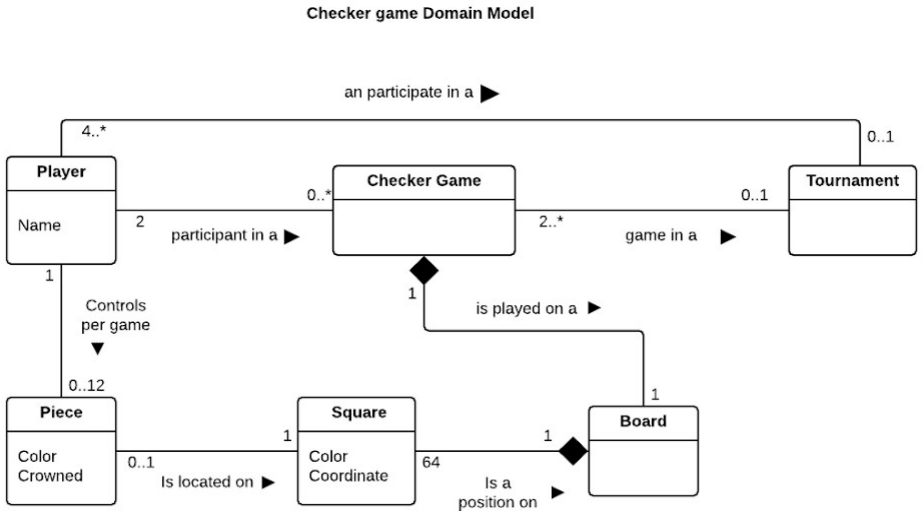

Diagram 1 : Domain model of web application

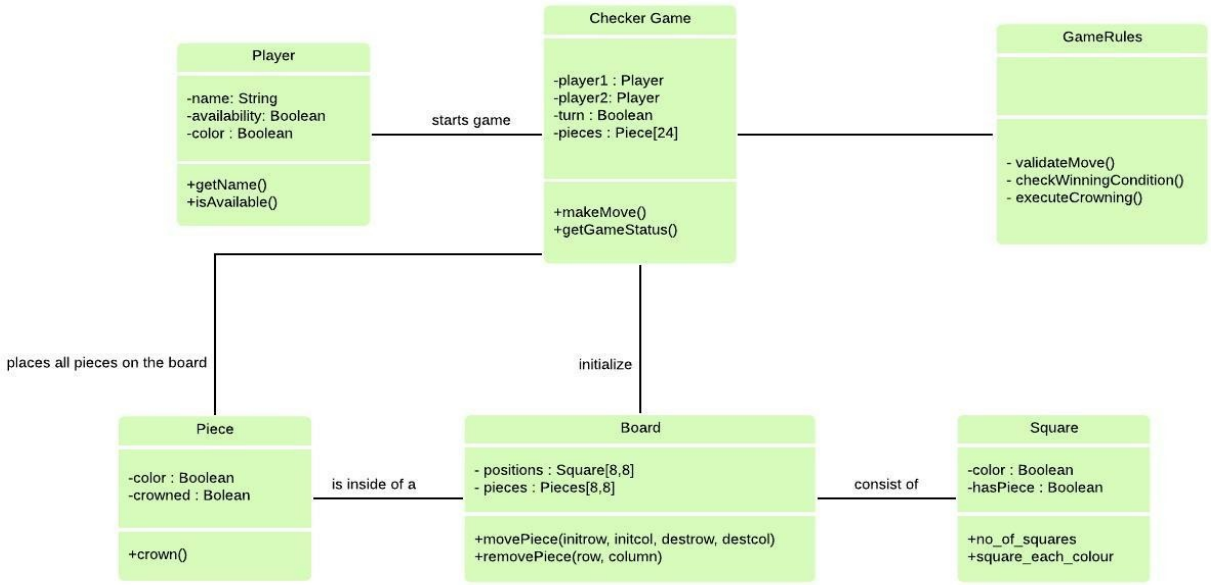

Diagram 2 : Class diagram for the game

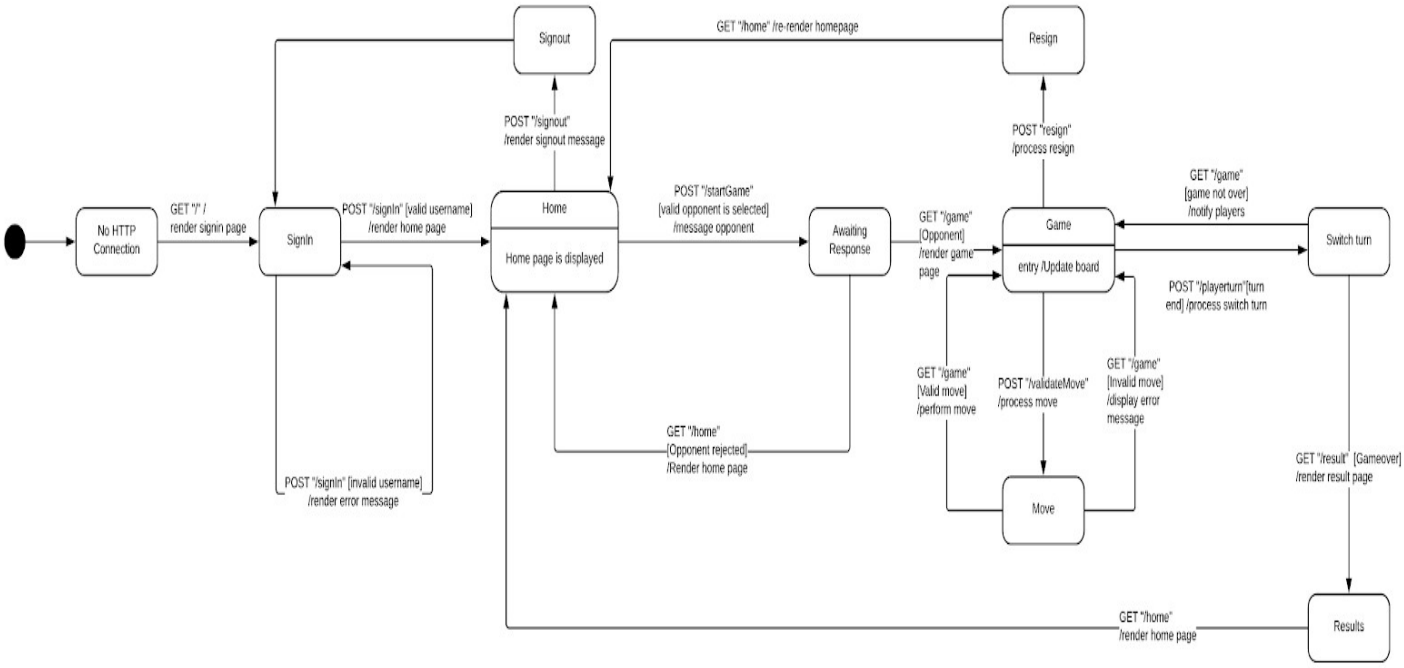

Diagram 3 : State diagram for the web application

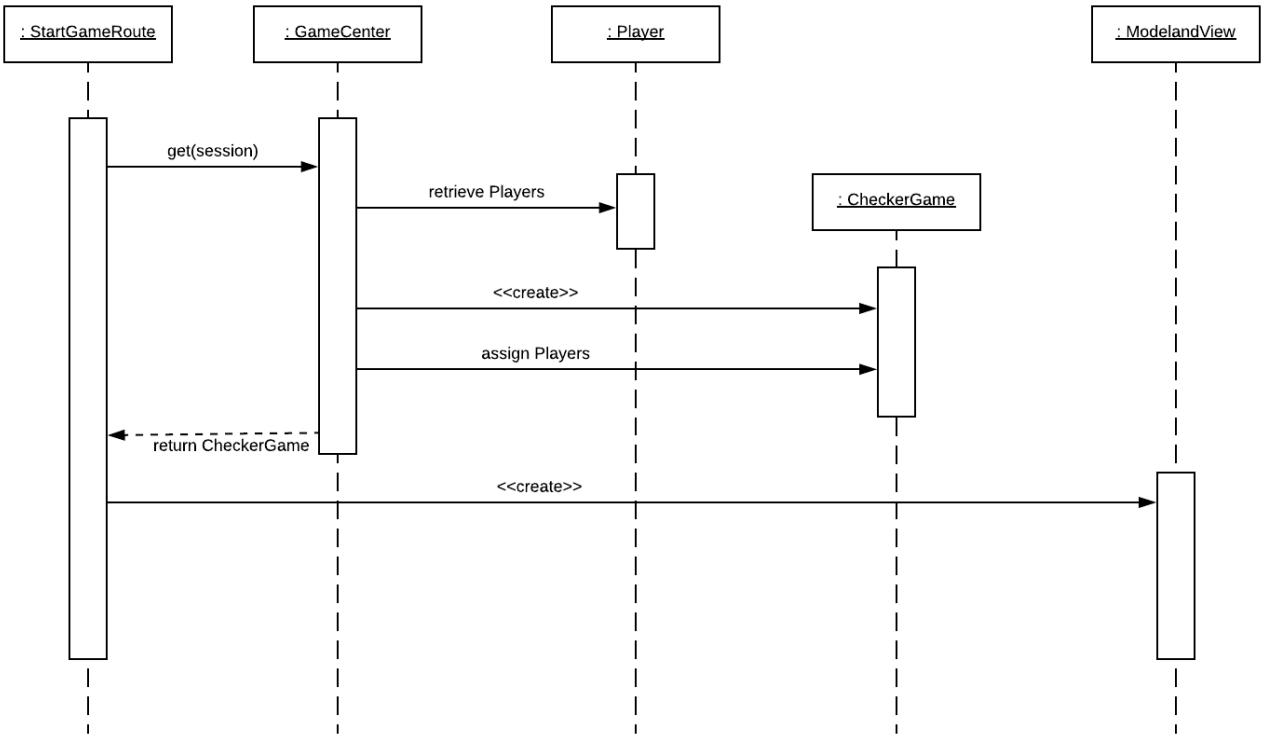

Diagram 4 : Sequence diagram for the game