BoilerChess

Team 18 - Product Backlog

Kyle Copenhaver, Eric Lee, Nick Zetzl, Sam Kwarteng

Problem Statement

With this project, we aim to create a web-based version of Chess that will challenge people to develop and harness critical thinking and strategic planning. By matching players against those of similar skill, users can find competitive play easily, or we provide the option for players to face off against their friends. This project will also provide unique programming challenges as we explore the innumerable possible game states and interactions. Not only this, but a logic based game provides many different aspects to practice our software testing skills.

Background Information

Audience

There is always a need for people to expand their critical thinking and improve their problem solving abilities, and chess is an excellent method of doing so.

Similar Applications

There does exist many similar versions of this, but our goal is not only to help fulfill this societal need, but also to improve our own skills in testing software. A game of chess provides many clear test cases and since it is a logic game with nearly $7.7 * 10^{45}$ states, there are more than a few interactions which would need to be tested.

Limitations

As for limitations, the most obvious concern is players of higher skill not being challenged enough, and less skilled players being against opponents they can’t possibly beat. To solve this, we will implement a skill based Matchmaking Rating (MMR), to track how many wins/losses players get against each other.

Environment

Our project will primarily be implemented in Javascript, using NodeJS for the back end and a different front-end framework to handle the user’s client. As for testing, there are a few automatic test frameworks such as Karma or Protractor which we’ll use to automatically test our software. We also plan to use a sqlite database to store information about the users, games, etc.

Functional Requirements:

Backlog ID	Functional Requirement	Hours	Status
1	As a user, I would like to be able to create a profile which contains my email, username, password.	40	Complete
2	As a user, I would like to see my profile which will show the email, username.	10	Complete
3	As a user I would like to be able to play on my own without an opponent.	20	Complete
4	As a user, I would like a display showing all the past moves of the current game.	6	In Progress
5	As a user I would like to log in with my username and password.	10	Complete
6	As a user, I would like squares to be highlighted that are valid moves for the currently selected piece when it is my turn	6	Complete
7	As a user I would like my username and password to be saved for the next time I attempt to log in	10	Complete
8	As a user I would like to know if my password or username is incorrect	10	Complete
9	As a user I would like to know if the game has ended due to checkmate	20	Complete
10	As a user I would like to make sure all moves are final with no redo’s	15	Complete

Non-Functional Requirements:

1. As a user, I would like a simplistic user interface that still gives a good amount of functionality
2. As an administrator, I would like for the website to be easy to maintain
3. As a user, I would like the game to run without lag or framerate issues
4. As a user, I would like the game to not use any of my computer's personal storage
5. As a user, I would like for any personal information to remain secure

Use Cases:

- Create a profile (1)

1. Site prompts user to create profile	
2. User puts in email, username, password, etc	3. Server check for valid info
4. If necessary, user puts in new information	5. Once valid info is sent by user, server places user in database
	6. User is redirected to login

- Log In (1)

1. User navigates to log in page	
2. User puts their information in	3. Server runs check against database for correct user/password
	4. Server prompts user for correct password if necessary
5. User re-enters information	6. Once valid information has been entered, user is navigated to home

- Player Profile Page (2)

1. User navigates to login page and logs in	2. Server validates information and redirects them if it is correct
3. User navigates to their profile page	4. Server pulls up profile information from their entry in database, including match history, MMR, and friends

- Checking match move history in current game (3)

1. While in game, user selects "View Move History"	2. Server brings up list of moves in the current match by each player on a sidebar

- Displaying legal moves for selected piece (4)

1. While in game and during the user’s turn, the user selects a piece	2. Server checks the piece the user has selected and calculates the possible moves for the piece
	3. Server updates the screen to show legal moves