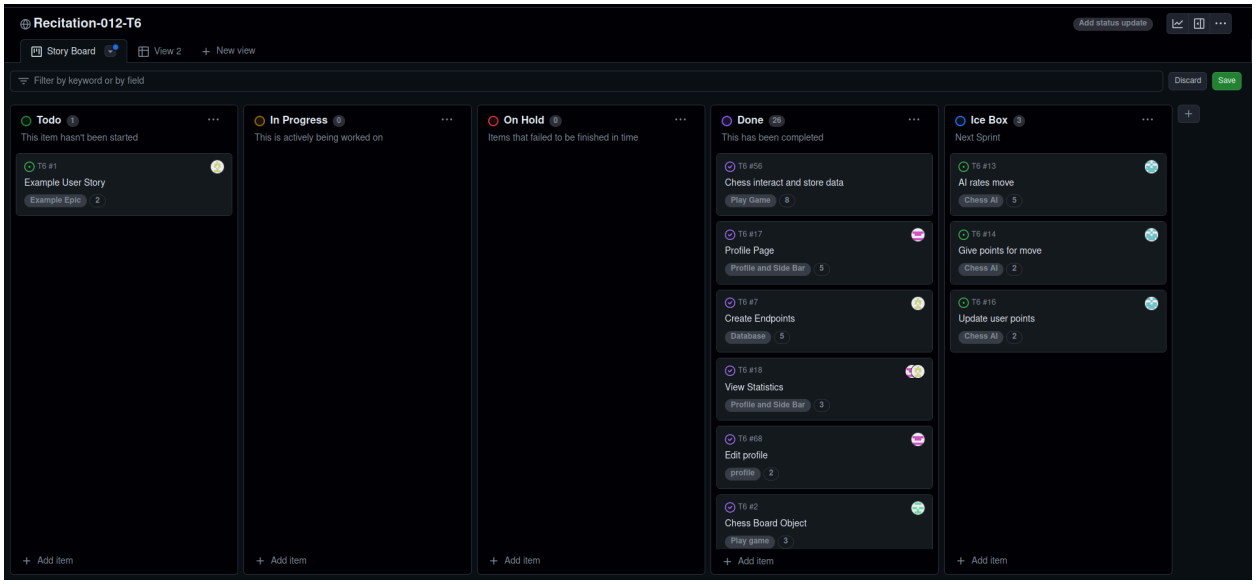
**Title:**
Deco Chess

**Group Members:**
Cole Downs
Emma Kalikstein
Isabella Bates
Jackson Stepka
Mathias Labuszewski
Paul Tracy

**Project Description:**
For our project we wanted to create a chess-playing website with multiple parts: a daily chess puzzle, playing against chess AIs, playing against another user. On this website, users can create an account and start out with zero chess points. They can gain chess points from solving the daily puzzle, solving other chess puzzles, and by making 'smart' moves against the AI or against other players. We will decide how good a move is and how many points it is worth by ranking the move with our AI. When a move is played, it will be recorded and then calculated how many points it will earn. However, the amount of points gained will not be shown until the end of the game, so as to not give an unfair advantage to any one player. As you earn points, you can unlock skins for your chess pieces. If you win a game, you gain a loot box where you are given a random skin. We will also have a leaderboard that lists the users with the highest points in a table. Along with a customizable profile page, a user's profile picture and bio, that houses a user's current statistics. Such as number of games won/lost, a ratio of the two, and their current and lifetime total of points. The user will also be able to view their game history in the profile page.

**Project Tracker:** https://github.com/users/Pandabear1125/projects/3/views/1?filterQuery=




**Video:**
https://drive.google.com/file/d/1TxjCIHSlz6eiNs2dNlknxwFOvn3jpmyV/view?usp=sharing

**VCS:**
https://github.com/Pandabear1125/T6

- 'README.md' states what the project is and how to run it
- Source code is in ProjectSourceCode file

**Contributions:**

Jackson Stepka

- I worked on the database and helped with the endpoints. The database stores user data like username/password and chess data like wins, losses, and games.

Cole Downs

- I worked on the API for the Register, Login, and Home pages. Along with the handlebars, HTML design, and style for those pages.

Emma Kalikstein

- I worked on the game type selection feature that opted the user for player type and whether they wanted to play standard, blitz, or bullet. Choosing between these games updates the timer to start at 10 minutes, 5 minutes, or 3 minutes. I also created the timers countdown function so that it counts down during each player's turn and also pauses the timer for the opposite player.

Isabella Bates

- I wrote the code for the "Player vs Computer" mode and the chess AI. I used a chess API to have the AI think of the best move given the player's previous move. I wrote the API endpoints for the chess API and tested the responses in Postman. I wrote the functions

that played the AI move and handled the interactions between the player and the AI. I worked on the chai tests for the API endpoints for the register and login pages, and wrote the testing plan. I set up our meetings with the TA.

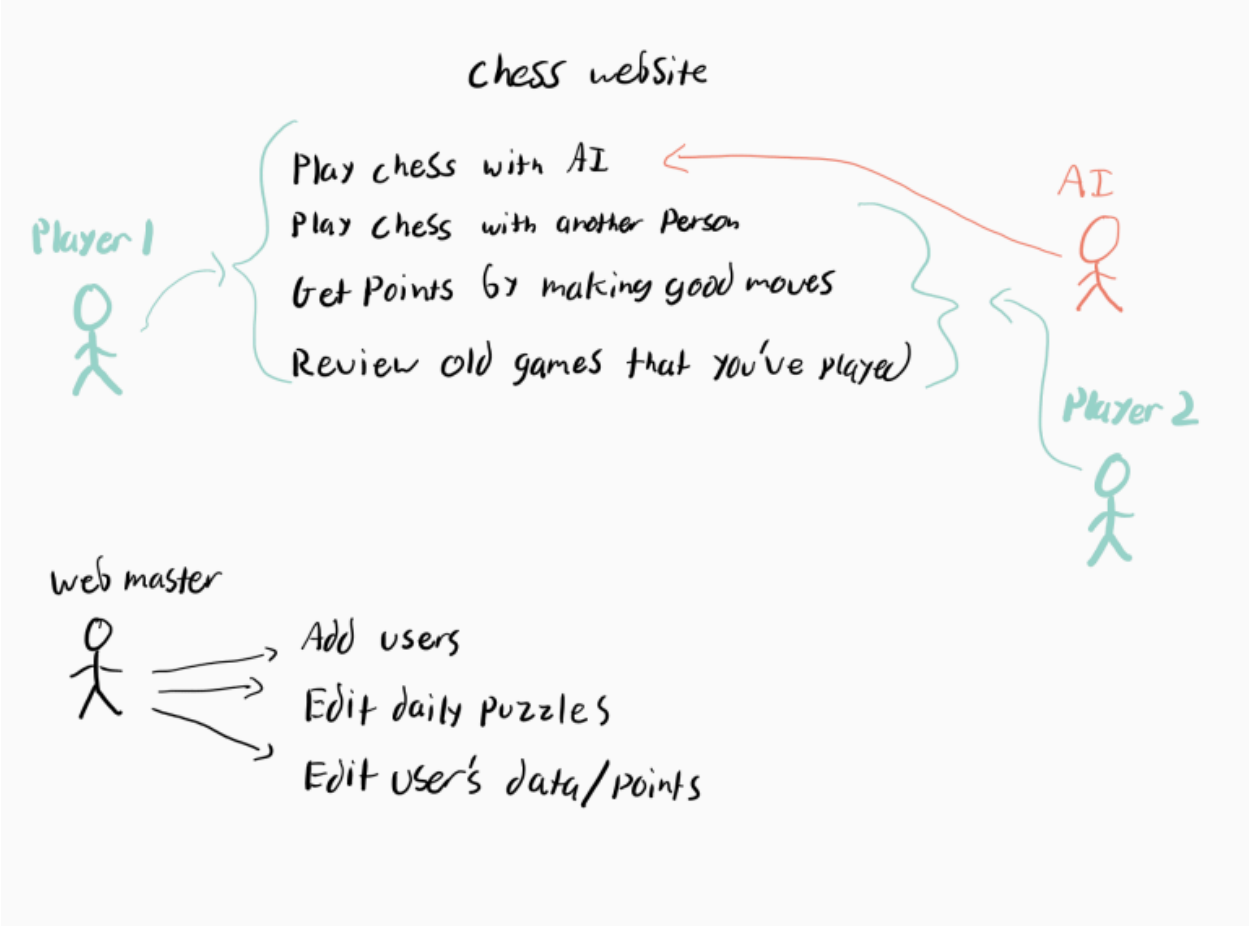
Mathias Labuszewski

- Created the 2d canvas methods and classes for the chess game. I created dynamic methods for loading images and gathering data to render for the chessboard. Cross linked the js to interact with the AI hander and the Timer. Created callbacks for each event to aid in interaction and to help interweave parts of the app. Worked with Isabella to help build and deploy the chessAI, and find public API endpoints to use.

Paul Tracy

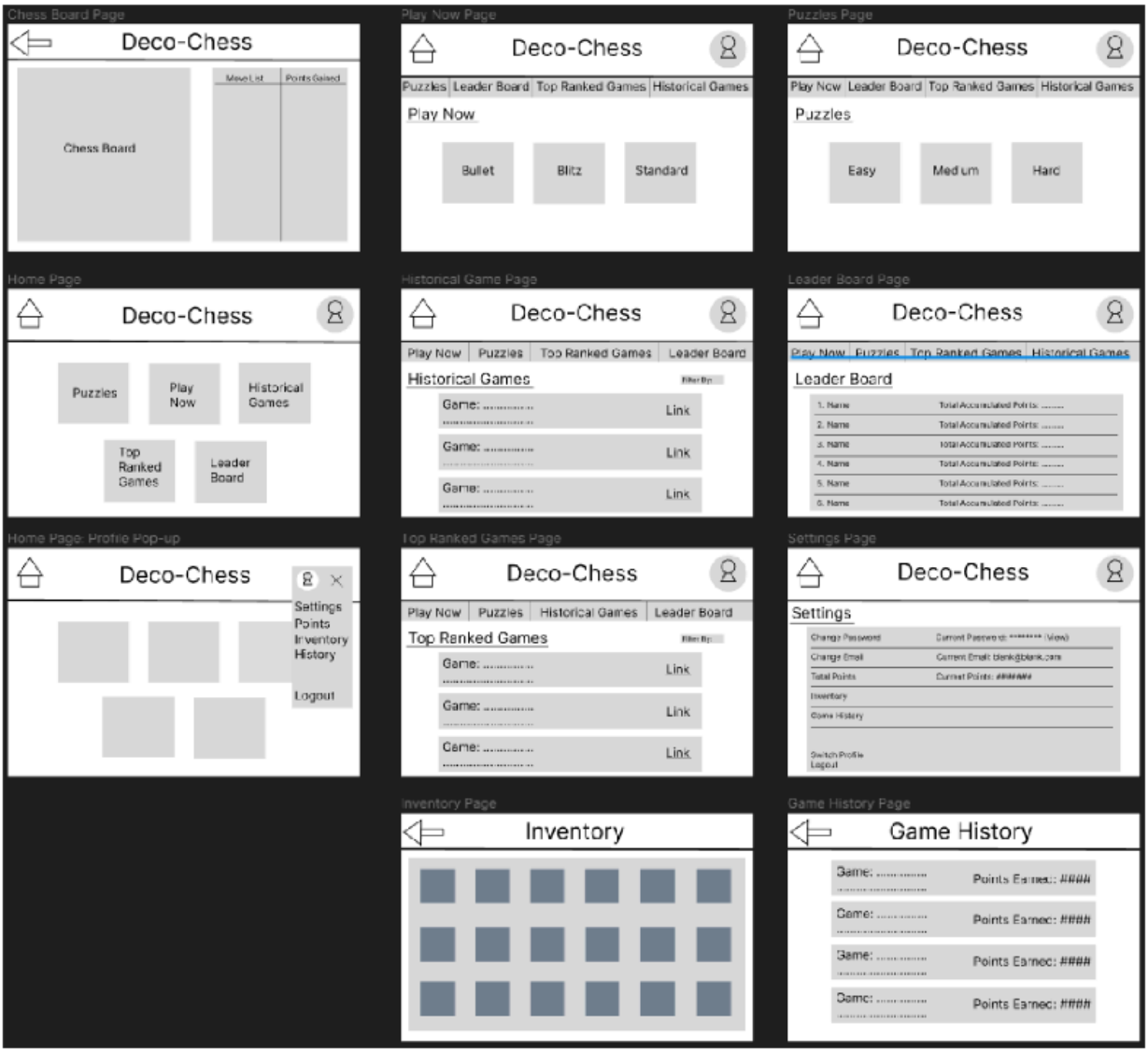
- The main thing that I worked on was the profile page. This includes the HTML, minor css, and the endpoints. The endpoints queried the database to grab unique information about each user including their profile picture, user biography, and some user statistics. I couldn't figure out how to display the games so the games on the profile are just a placeholder.

**Use Case Diagram:**



**Wireframes:**

https://www.figma.com/file/LhKSdXfUocCZfQVAgygkX7/Wireframe-for-SoftDev-Project?type=design&node-id=0%3A1&mode=design&t=oYFKJ6rUmPmWBZ3Z-1



**Test Results:**

The four user tests that we tested are:

1. Users should be able to register and login.
2. Users should be able to, after being logged in, select to play a game and navigate to the home page.
3. Users should be able to edit their bio, and view the edited data afterwards.
4. Users should be able to change their profile picture, and view the edited data afterwards.

As to the success of these tasks from our users, all of the tests were passed successfully with only minor stalls, except for task 4 which took a lot longer before it was passed. This was due to the nature of the description above the input for when a user wanted to change their picture. Above the input field instructions were left to help the user, specifically "Enter a link to your profile picture." We found that the users were following those instructions, but that they were flawed. Our feature of changing the profile picture works seamlessly when the link is the picture's image address. So we changed the description above the input field to be, "Enter the image address of your profile picture." Also, when going through testing we found that the users struggled slightly when it came to registering their account. We had parameters which would not allow a username to have more than 20 characters and/or contain a special character. However, when a username did not follow those directions an error would pop up and say, "Invalid input." This did not convey what was wrong with the username, and many of our user's that ran into this issue were confused as to what was wrong. Granted they were able to put in a correct username quickly, we thought that changing the error message would be very beneficial. So now it reads, "Username can not have special characters or be more than 20 characters long." For the other tasks the users got through them very quickly and thought the methods to achieve them were very straightforward.

**Deployment:**

In order to run our app, you must clone the repository. Then, navigate to the ProjectSourceCode folder. Within this folder, open a terminal. Run the command "docker-compose up -d". Once this command runs, open in your web browser the url: "http://localhost:3000/". This will navigate to our login page and then you can start enjoying our wonderful app.