Game Contest Server

Interactive, web based manager
for automated turn-based game contests

Introduction

- Previous Work
- User Documentation
- User Permissions
- Replays
- User Interface
- Test Driven Development
- 3+ Player Tournaments
- Future Work
- What We Learned

Previous Work

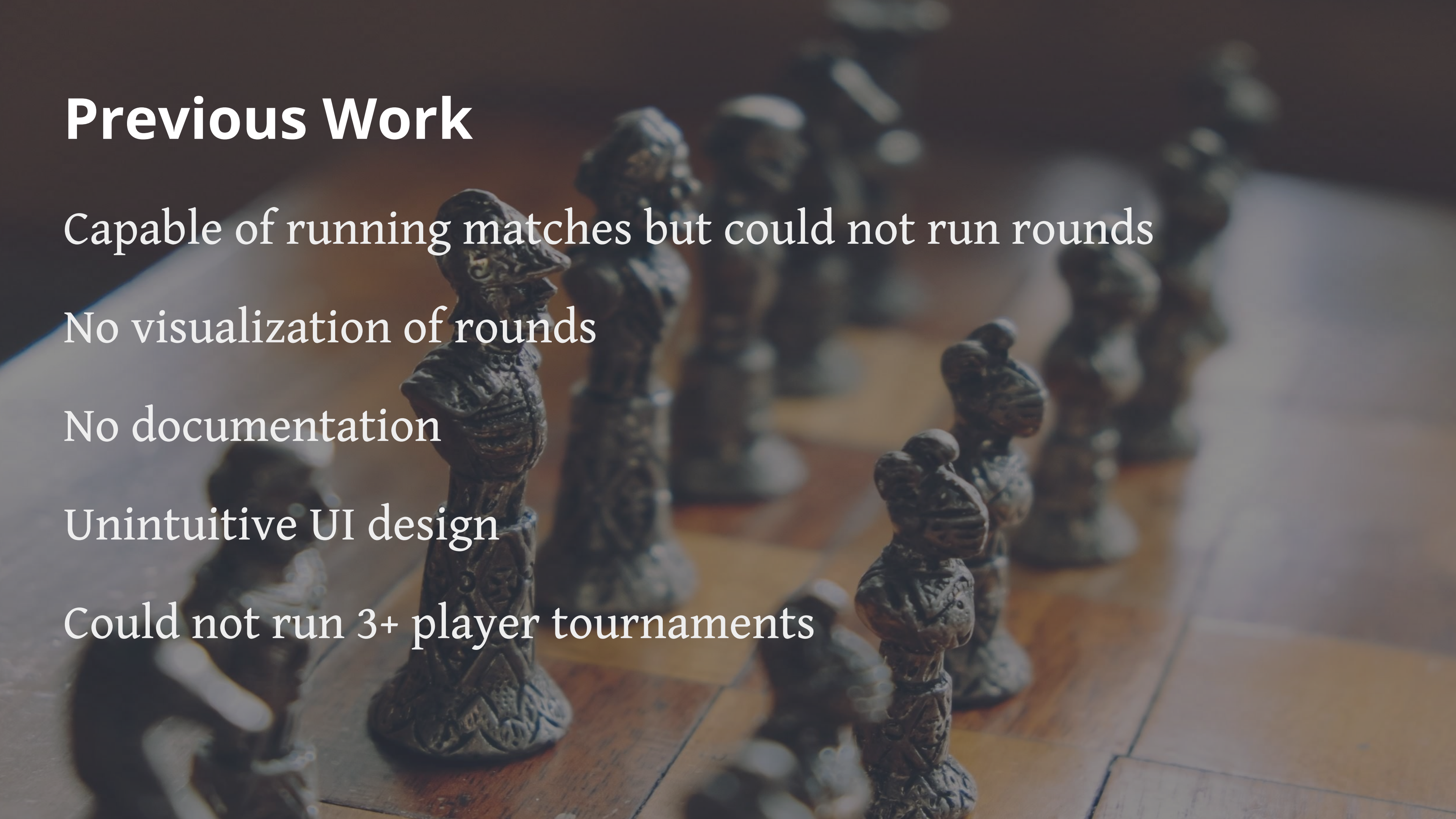
Expanded testing suite

Front-end improvements

Compressed file functionality (widens game scope)

Created the challenge feature

Fixed bugs



Previous Work

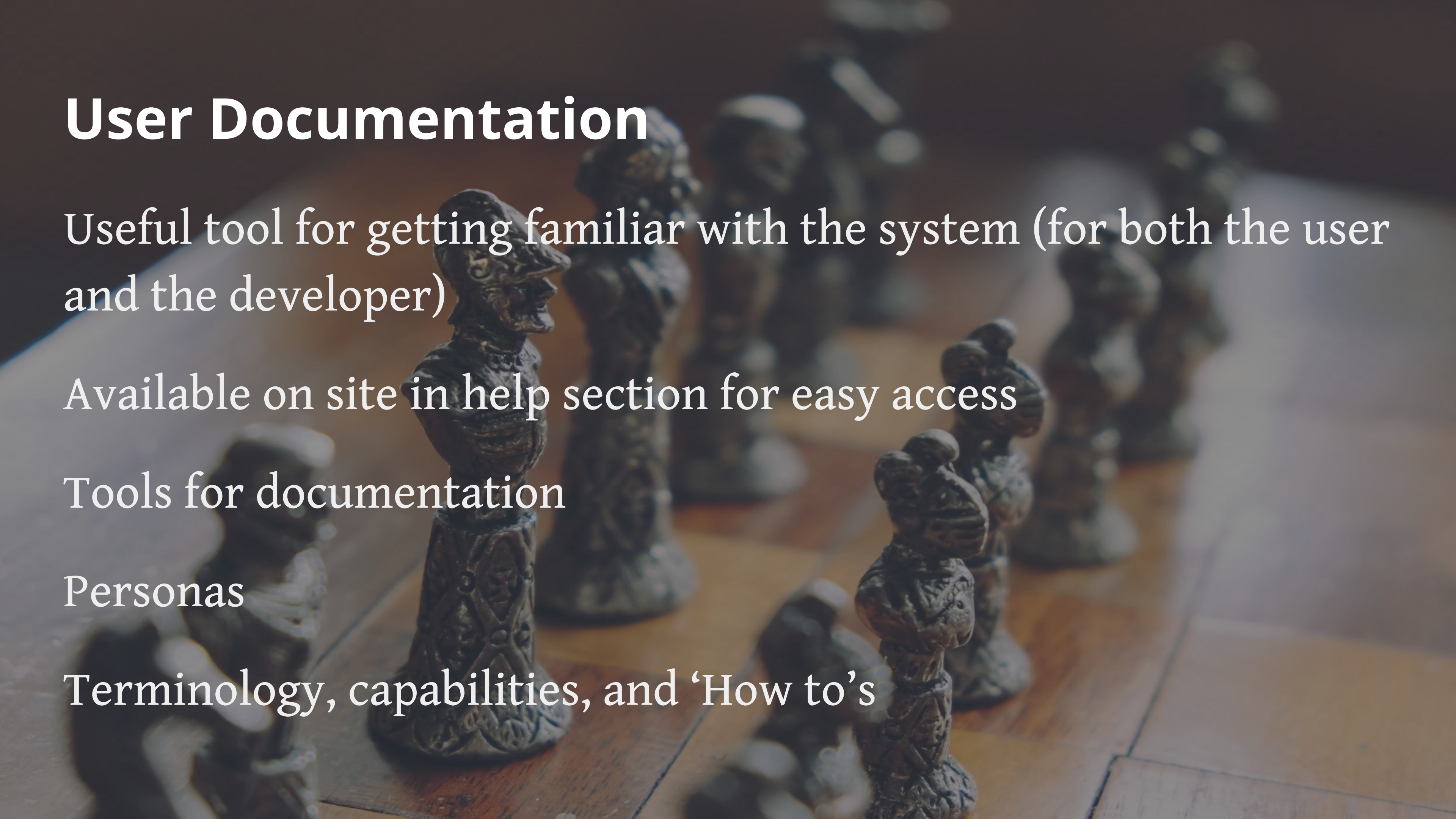
Capable of running matches but could not run rounds

No visualization of rounds

No documentation

Unintuitive UI design

Could not run 3+ player tournaments



User Documentation

Useful tool for getting familiar with the system (for both the user and the developer)

Available on site in help section for easy access

Tools for documentation

Personas

Terminology, capabilities, and ‘How to’s

User Permissions

No way for admins to edit user profiles or permissions

Required for developers to test features

Testing works!

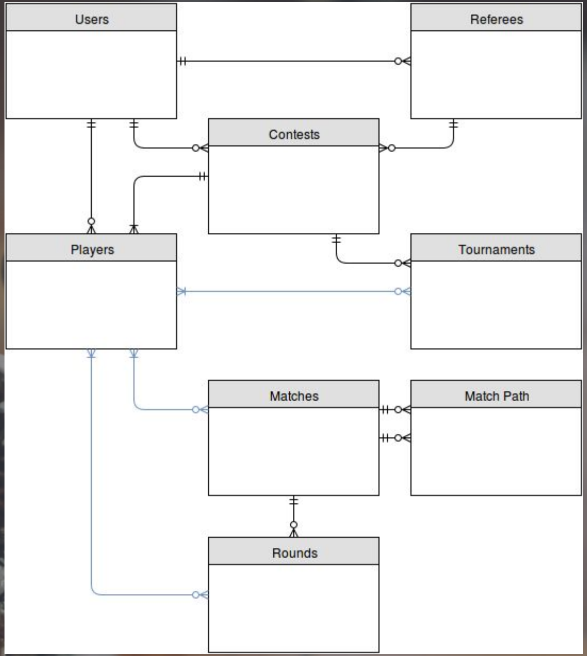
Replays

Enhancing the learning experience through visualization

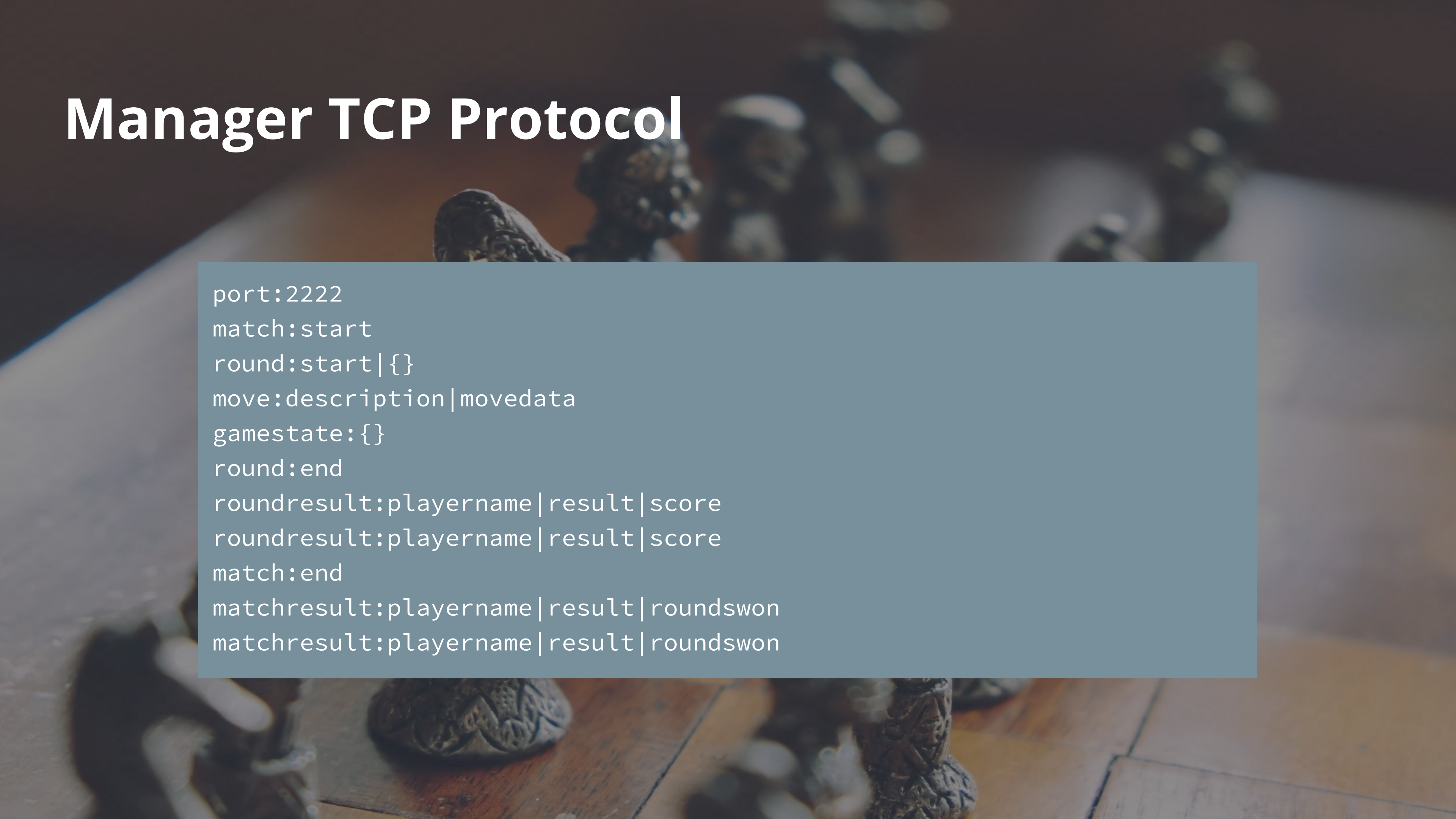
Normalizing Rounds

Problem: no concept of rounds in the database

Solution: updated database model



Manager TCP Protocol



```
port:2222
match:start
round:start|{}
move:description|movedata
gamestate:{}
round:end
roundresult:playername|result|score
roundresult:playername|result|score
match:end
matchresult:playername|result|roundswon
matchresult:playername|result|roundswon
```

Logging

Saving results of rounds for later viewing

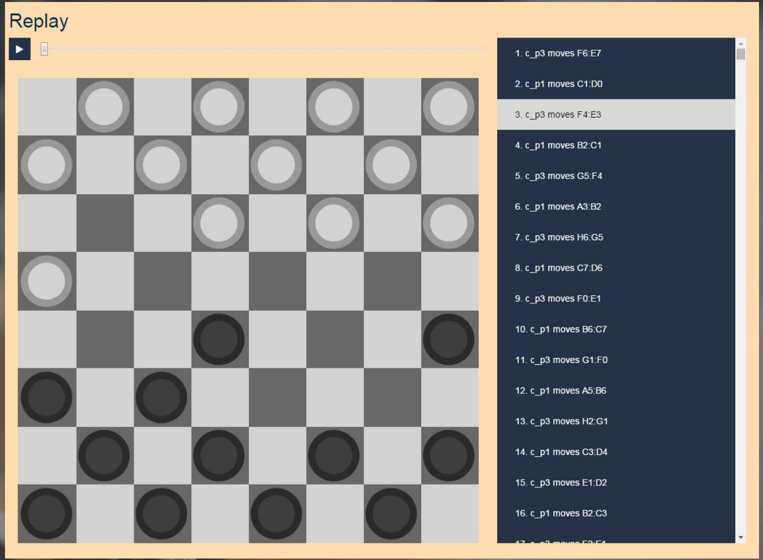
JSON API

Replay API

Rendering replays in the web browser

Pure JavaScript eases future maintenance

Extensible API allows different game types



Replay

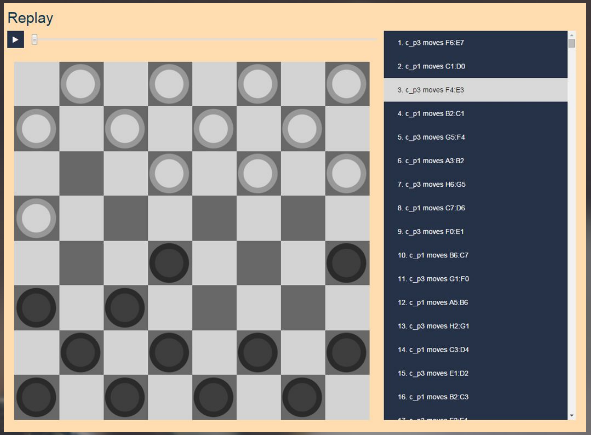
1. c_p3 moves F6:E7
2. c_p1 moves C1:D0
3. c_p3 moves F4:E3
4. c_p1 moves B2:C1
5. c_p3 moves G5:F4
6. c_p1 moves A3:B2
7. c_p3 moves H6:G5
8. c_p1 moves C7:D6
9. c_p3 moves F0:E1
10. c_p1 moves B6:C7
11. c_p3 moves G1:F0
12. c_p1 moves A5:B6
13. c_p3 moves H2:G1
14. c_p1 moves C3:D4
15. c_p3 moves E1:D2
16. c_p1 moves B2:C3

PIXI.js

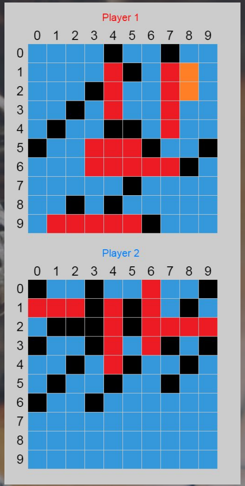
2D WebGL renderer with canvas fallback

Canvas supports non-WebGL platforms

Easy API


Replay

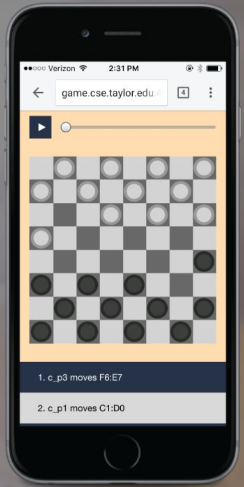
1. c_p3 moves F6:E7
2. c_p1 moves C1:D0
3. c_p3 moves F4:E3
4. c_p1 moves B2:C1
5. c_p3 moves G5:F4
6. c_p1 moves A3:B2
7. c_p3 moves H6:G5
8. c_p1 moves C7:D6
9. c_p3 moves F0:E1
10. c_p1 moves B6:C7
11. c_p3 moves G1:F0
12. c_p1 moves A5:B6
13. c_p3 moves H2:G1
14. c_p1 moves C3:D4
15. c_p3 moves E1:D2
16. c_p1 moves B2:C3


Player 1

	0	1	2	3	4	5	6	7	8	9
0	🟦	🟦	🟦	🟦	⬛	🟦	🟦	⬛	🟦	🟦
1	🟦	🟦	🟦	🟦	🟥	⬛	🟦	🟥	🟧	🟦
2	🟦	🟦	🟦	⬛	🟥	🟦	🟦	🟥	🟧	🟦
3	🟦	🟦	⬛	🟦	🟥	🟦	🟦	🟥	🟦	🟦
4	🟦	⬛	🟦	🟦	⬛	⬛	🟦	🟥	🟦	🟦
5	⬛	🟦	🟦	🟥	🟥	🟥	⬛	🟦	🟦	⬛
6	🟦	🟦	🟦	🟥	🟥	🟥	🟥	🟥	⬛	🟦
7	🟦	🟦	🟦	🟦	🟦	⬛	🟦	🟦	🟦	🟦
8	🟦	🟦	⬛	🟦	⬛	🟦	🟦	🟦	🟦	🟦
9	🟦	🟥	🟥	🟥	🟥	🟥	⬛	🟦	🟦	🟦

Player 2

	0	1	2	3	4	5	6	7	8	9
0	⬛	🟦	🟦	⬛	🟦	🟦	🟥	🟦	🟦	⬛
1	🟥	🟥	🟥	⬛	🟥	⬛	🟥	🟦	⬛	🟦
2	🟦	⬛	⬛	⬛	🟥	⬛	🟥	🟥	🟥	🟥
3	⬛	🟦	🟦	⬛	🟥	🟦	🟥	⬛	🟦	⬛
4	🟦	🟦	⬛	🟦	🟥	⬛	🟦	🟦	⬛	🟦
5	🟦	⬛	🟦	🟦	⬛	🟦	🟦	⬛	🟦	🟦
6	⬛	🟦	🟦	⬛	🟦	🟦	🟦	🟦	🟦	🟦
7	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
8	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
9	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦



1. c_p3 moves F6:E7
2. c_p1 moves C1:D0

Demo

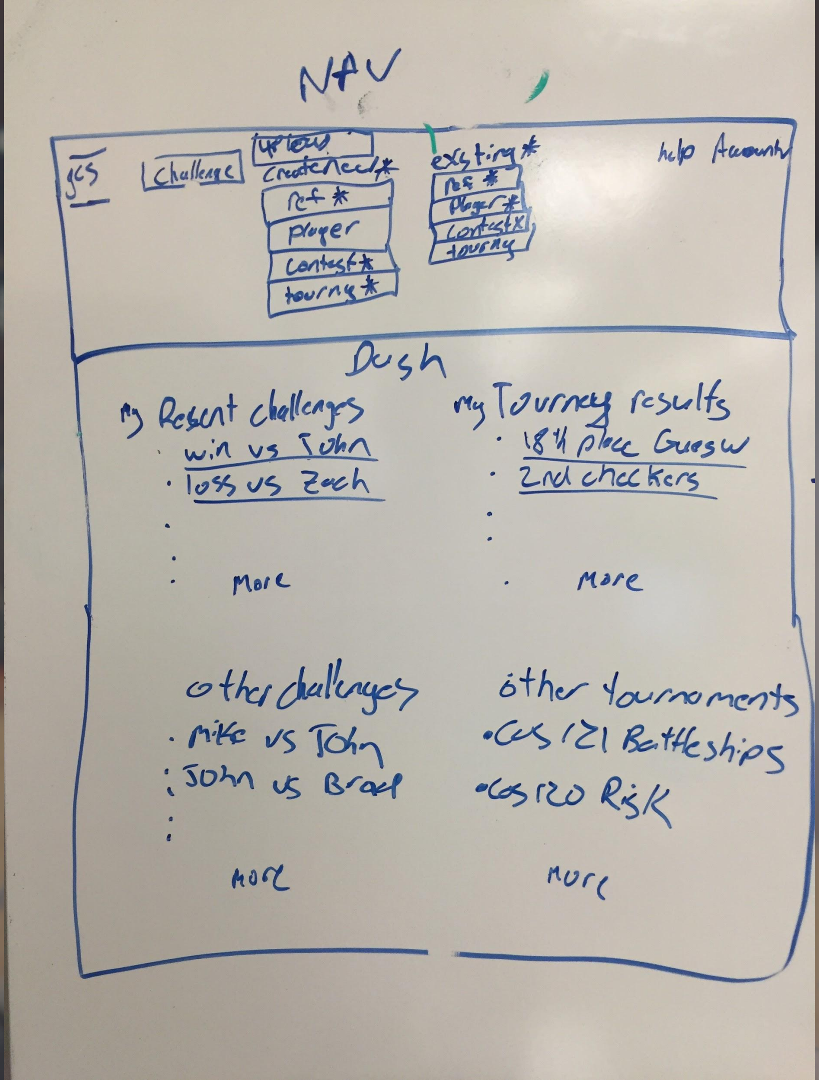
User Interface

- Redesign process
    - Wireframes
    - Testing
- Access all functionality from navbar

Dashboard & Navigation Menu

Area for all information

New Nav Bar

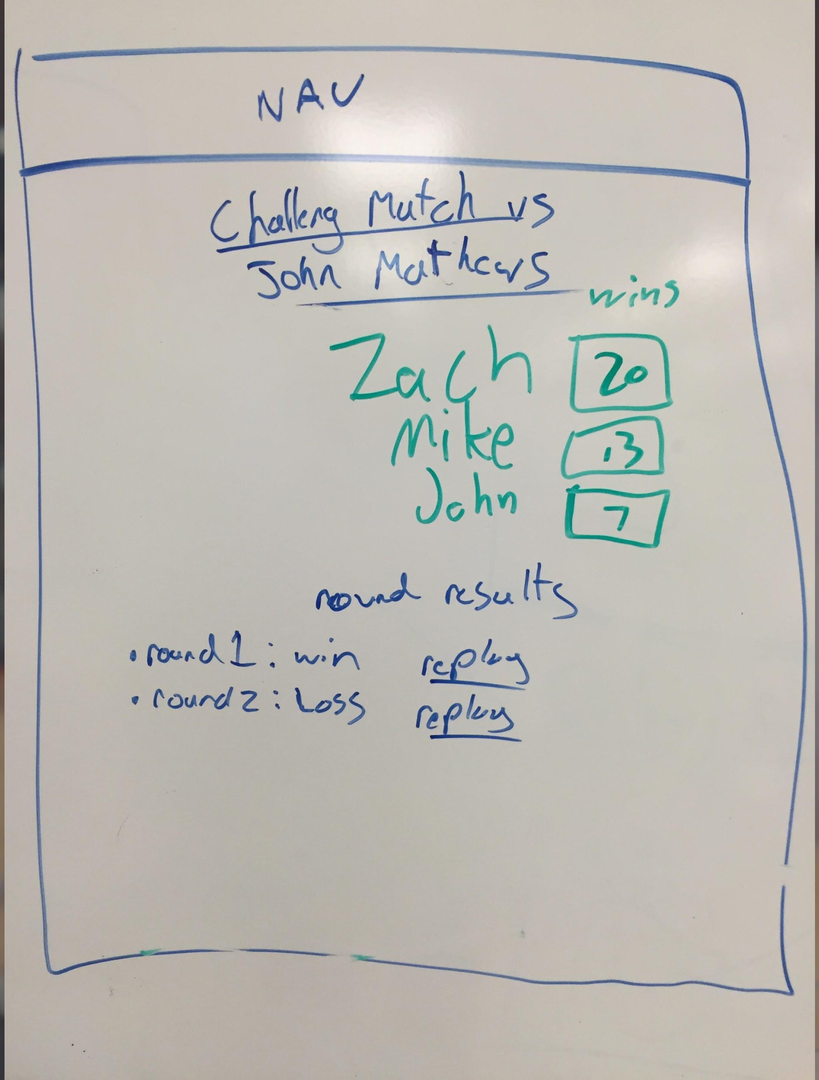


Match Results

Useful Information

Replays available

Easy to interpret

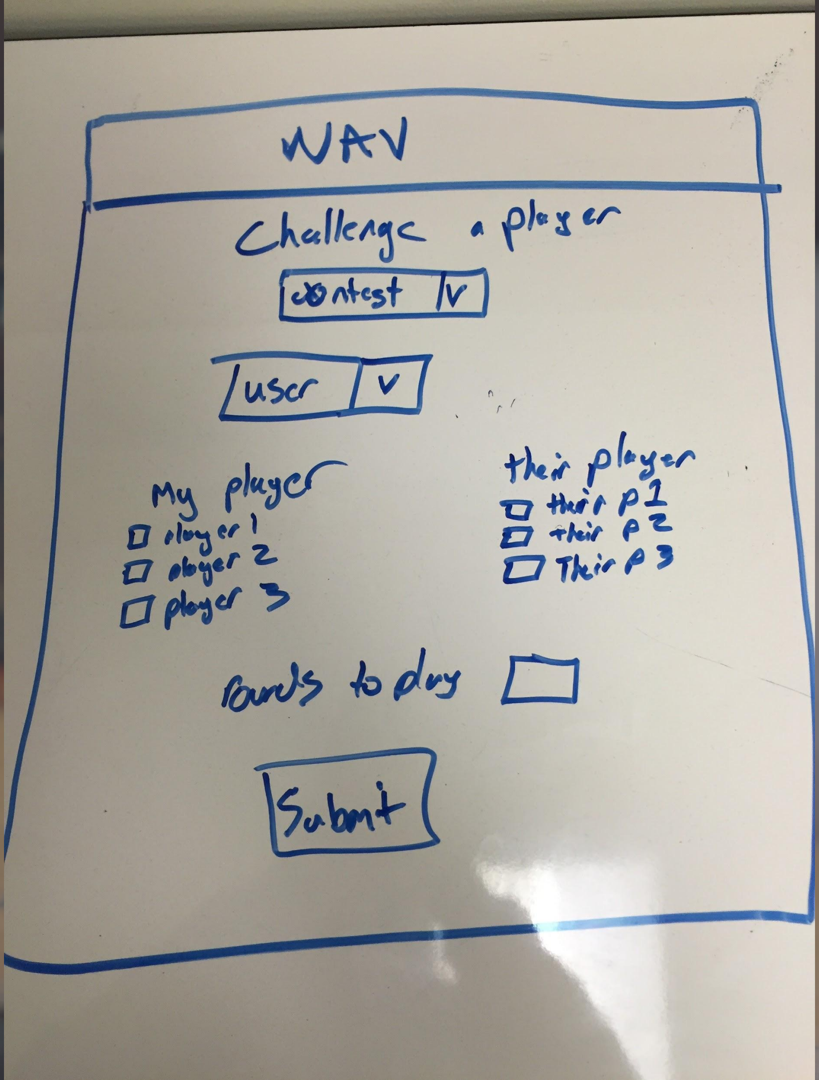


Challenge Page

From the nav

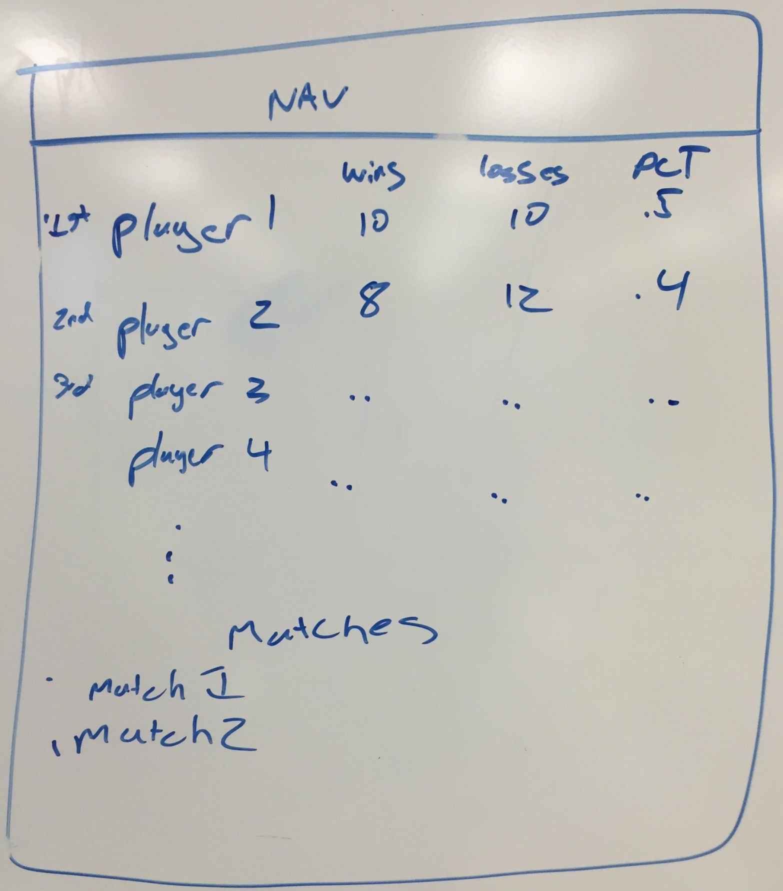
Any contest

Many clicks already



Round Robin

Clear & concise results



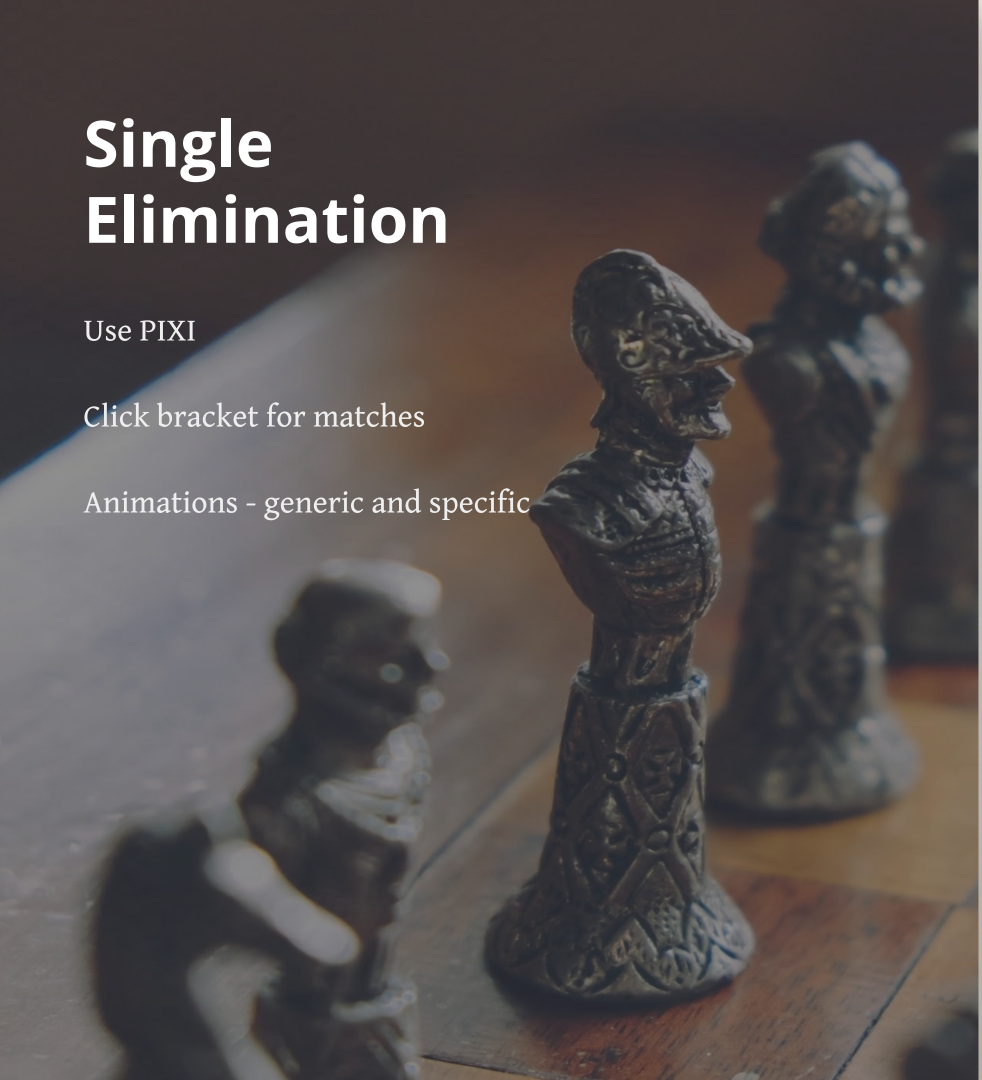
NAV

	wins	losses	PcT
1st player 1	10	10	.5
2nd player 2	8	12	.4
3rd player 3	..	..	..
player 4	..	..	..

⋮

Matches

- Match 1
- match2

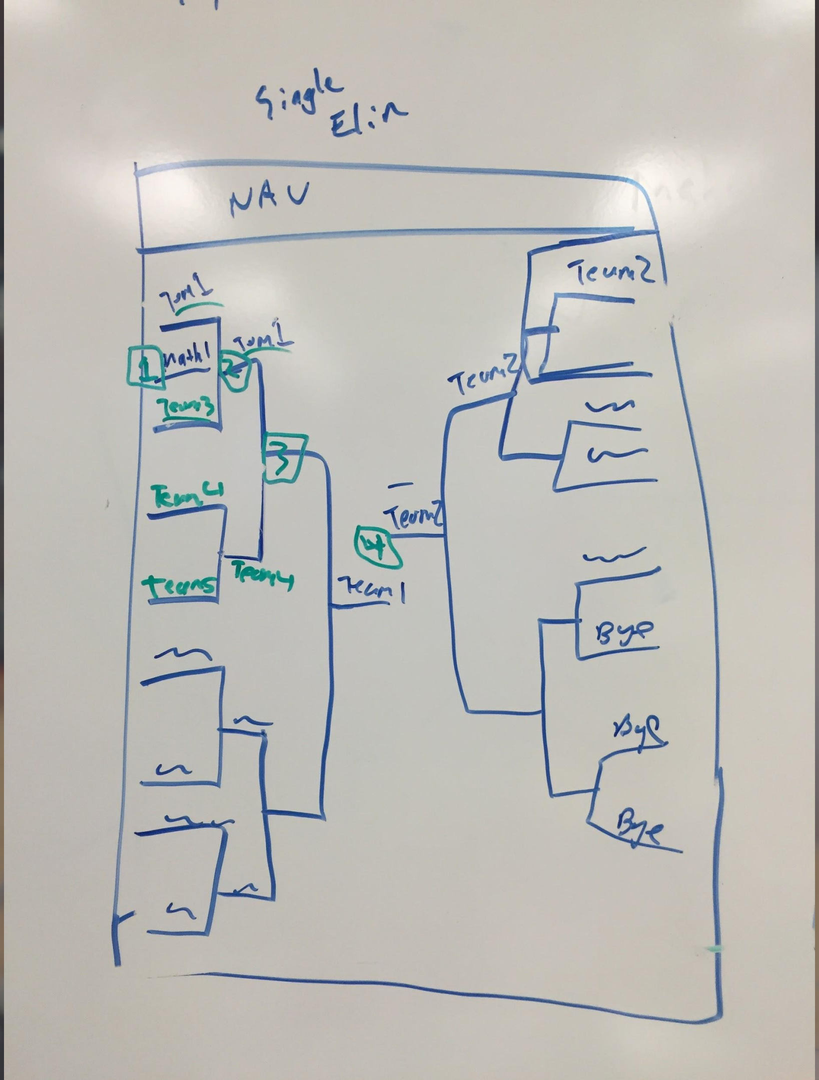


Single Elimination

Use PIXI

Click bracket for matches

Animations - generic and specific



Test Driven Development

- “Red, Green, Refactor”
- Tools
  - Rspec
  - Capybara
  - FactoryGirl
- Our test development strategy

Testing Is Important

- Last years group wished they started off testing
- It is the backbone of our project
- Helps focus development

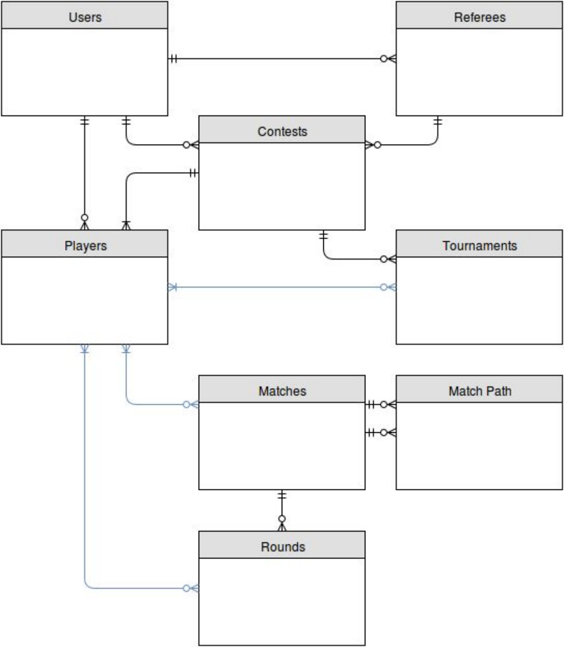
Admin Testing

- First major test additions
- Dealing with permissions
- Old tests proving useful

Challenges with testing

- Learning curve
- Ambiguous code of existing tests

Rounds Testing - Adding New Tests



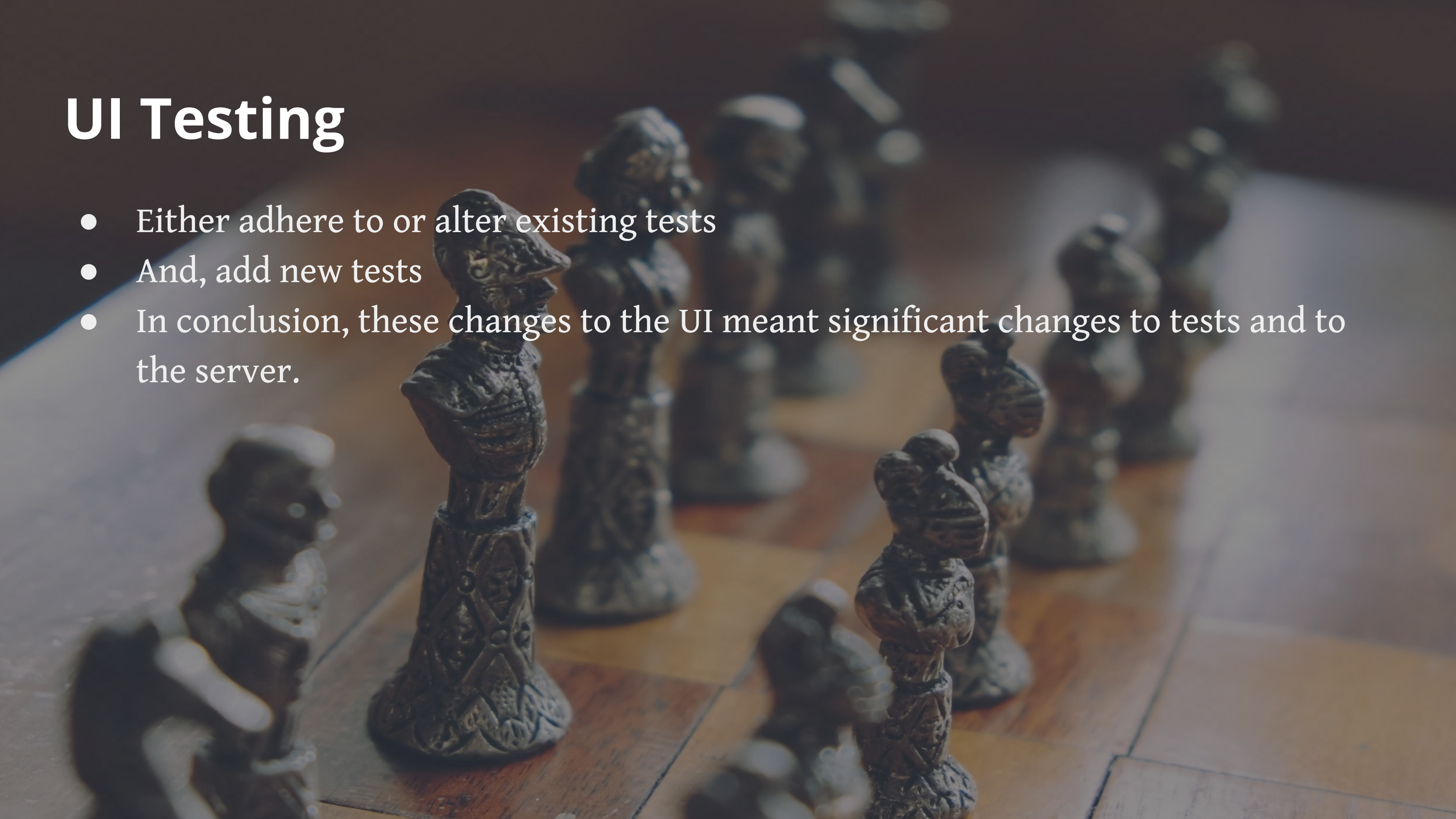
- The Rounds and PlayerRounds entities
- In what state is the round saved to the database?
- Viewing a round- which users should be able to view a round?
- How are rounds displayed?

Rounds Testing - Maintaining Existing Features

- As the design of the database changes, so the tests and the system must reflect these changes
  - Changing fields of existing tables
  - New relationships between existing tables and new tables

UI Testing

- Either adhere to or alter existing tests
- And, add new tests
- In conclusion, these changes to the UI meant significant changes to tests and to the server.



3+ player Tournaments

- Making sure players play an even amount of games with a good mix with a Round Robin style format.
- Conceptualizing a proper model that could accurately rank each player from best to worst.
- Condorcet Method - using a matrix of player wins/losses to one another

Future Work

- 3+ Player Tournament Integration
- Visualizing Single Elimination
- Re-play plugins for various game types
- More UI improvements
- Additional tournaments types

What We Learned

- Rails development process
- Collaboration in a large team
- Communication
- Time management

Acknowledgements

- Dr. Jonathan Geisler developed the requirements and provided guidance throughout the project
- Nate White and Nathan Lickey helped us manage our development VM
- Various CS professors for providing high-level guidance and support in the process
- Justice Juraschek for designing checkers assets.