Clean up documentation:

- ☑ ~~No annotations in the Server Class Diagram (or any diagram) - max~~
- ☑ ~~Remove unnecessary classes and methods from all class diagrams~~
    - ☑ ~~Server Class Diagram - Daniel~~
    - ☑ ~~Domain Class Diagram - Daniel~~
    - ☑ ~~(I think there is still many unnecessary methods and variables that could be removed to make space)~~
    - ☑ ~~UI Class Diagram - max~~
- ☑ ~~Make text bigger, reduce whitespace -max~~
    - ☑ ~~Server Class Diagram~~
    - ☑ ~~Domain Class Diagram~~
    - ☑ ~~UI Class Diagram~~
- ☑ ~~Create user documentation~~
- ☑ ~~GET /board should not return 404, return empty array in YAML - Daniel Ma~~
- ☑ ~~GET /board/{id} should return a 404 in YAML - Daniel Ma~~

Server:

- ☑ ~~Create the inaccessible list and nextPuzzle when a new user is created in the postUser method of UserController -Daniel Ma~~

Client:

- ☑ ~~Ensure that userID.txt is deleted if blank in databaseLoader - Daniel~~
- ☑ ~~Implement connection to the server~~
    - ☑ ~~Ask Dr. Wolff how we should communicate with the server from the client~~
    - ☑ ~~Ask Dr. Wolff how we should store the user id~~
- ☑ ~~Finish the board processing -Max~~
    - ☑ ~~Update invalidStars and validStars properly -Max~~
    - ☑ ~~Update isSolved method to use the solution list~~
- ☑ ~~Complete the LevelMenuScene (to work with User and load the boards)~~
    - ☑ ~~Buttons should load based on the inaccessible list, completed list, and nextPuzzle variable - Randy~~
    - ☑ ~~Clicking on a button should load the Board and the PuzzleScene~~
- ☑ ~~When a board is completed in PuzzleUI, update User and the User in the database when a puzzle is completed, and generate a new LevelMenuScene~~
    - ☑ ~~Board's isComplete returns true, so PuzzleUI's instance of User does a updateNextPuzzle -Kade~~
    - ☑ ~~PuzzleUI uses DatabaseLoader to do an updateUser on the database -Kade~~
    - ☑ ~~PuzzleUI calls event when board isComplete~~
    - ☑ ~~LevelMenuScene is updated via an eventListener and PuzzleSceneSwitcher~~
    - ☑ ~~Some kind of event listener to update the DatabaseLoader and LevelMenuScene? - Randy~~
- ☑ ~~Remove JSONStatus~~
- ☑ ~~Remove JSONReader~~

- ☑ ~~Remove JSONPuzzleGenerator - Daniel Ma~~
- ☑ ~~Implement DatabaseLoader class (domain) to create the User and various boards -Kade~~
    - ☑ ~~Store user ID in the client using a text file -Kade~~
    - ☑ ~~updateUser method -Kade~~
    - ☑ ~~Update the Domain Class Diagram -Kade~~
    - ☑ ~~Implement the getBoard method -Kade~~
- ☑ ~~Get all REST APIs wired up~~
    - ☑ ~~Send to BoardController to download boards~~
    - ☑ ~~Send to UserController to interact with user data~~
    - ☑ ~~Additional testing?~~
- ☐ [Bonus] Add Additional “fun” features.
    - ☑ ~~Add an error message when the client runs when the server is not running.~~
    - ☑ ~~Store location of stars for the current puzzle loaded on the client~~
    - ☑ ~~Add new puzzles to the database.~~
    - ☐ Implement numStars in Board (nope)
    - ☐ Make sure Board works with different sizes (nope)
    - ☐ Add Premium Subscriptions - Randy
        - ☑ ~~Added Button~~
        - ☐ 4 Hints per month
        - ☐ New Themes and stylish profile pictures
        - ☐ No advertisements
    - ☐ Mapping specific buttons(star, dot, remove) to certain keys on keyboard
        - ☑ ~~Right-click to remove a star or dot from a tile.~~

File Structure:

- ☐ Remove iteration 1, 2, and 3 folders for documentation (idea)
- ☑ ~~Rename json folder to user~~
- ☑ ~~Remove all unused files/packages~~
- ☑ ~~Replace instances of starb with starbattle (StarbClient, StarbServer, starb package in both main and test) (idea)~~

Presentation:

- ☑ ~~Answer all questions:~~
    - ☑ ~~If the project continues, what things might change, and how would the design be implemented to support future change?~~
    - ☑ ~~What was the team’s process? What went well? What didn’t? What would you change?~~
    - ☑ ~~What would you change about the overall design if you could?~~
- ☑ ~~5-minute software demo~~
- ☑ ~~Make sure the presentation only takes up to 25 minutes~~

Deliverables (due December 2nd)

- ☑ ~~Your RESTful API documentation~~

☑ ~~An updated class diagram~~
☑ ~~Updated package diagram to include server and client code~~
☑ ~~Complete implementation of your client and server application~~
□ A presentation to the class during the last week of the semester
☑ ~~User documentation~~