Document

Limitations

- The unit tests ended up needing four different docker containers to run without interfering with each other, and needed to be stopped, removed and spun up in between each test run. This made making small and frequent small changes laborious which was particularly inconvenient for TDD. As the entire test suite also got run as part of a pre-commit hook, this also has the same effect on ease of committing incremental changes.

- When a user creates a new question they cannot choose the position of the correct answer when displayed as part of a question, there is a get around, however it is not a good user experience.

- The use of big lists on every page did not take into account accessibility which is something that I wanted to do.

- I originally wanted the add answer and add question buttons to open up a modal, I however did not have time to implement this, and should have taken time into account more when doing the design.

- The view quiz page holds logic for both Restricted and View users, the page relies on a data structure that holds all quiz data including correct answers, which is also contained in a script on the page. This means that a restricted user would be able to find correct answers if they looked in the page source/ ‘hacked’ the front end in the inspect console.

- The above mentioned data structure is created by quite a nasty function. Creating it involved quite a lot of looping and mapping, and I am sure that there would be a nicer way to do it but I didn't have time to spend on de-complexifying it.

- A new quiz can be created without questions. Whilst I stand by my design for this, it means that the view quiz page is empty before questions are added. ‘No questions added yet’ or something similar should have been shown on that page.

- The order of checks made in the endpoints, e.g. ‘are all fields filled in’, ‘are there already 5 answers’ are not covered by the manual tests. It would be frustrating to a user to hit an error such as ‘all fields need to be filled in’ to later be told they can’t perform the action anyway.

- I didn't take answer and question constraints into account in the design, in the end I decided to display an error message at the top of the page. At one point I had written javascript to hide buttons etc which should not have been pressed. The endpoints

still needed to be protected, and decided to write in error messages and get rid of the javascript as it was getting messy and buggy, and no feedback was being provided to the user explaining why the buttons were not available.

- The way the sql tables have been designed does not allow for defence against question/answer limits being introduced into the database. I tried to write the endpoints as defensively as possible but I would have liked to find a way to enforce this in the database and think it would be worth spending time considering.

- When adding a question a user only has the option to add three answers, and will have to add more answers after creating the question if they want more.

Future improvements

- Add integration tests using a tool like supertest.

- Add automated E2E tests using a tool like cypress.

- Create and publish a node docker image with required packages pre installed, use this in the docker compose file to consume the tests as volumes and run them in a container to avoid all of the spinning up and down. Profiles could also be added to run each suite separately and be added as separate npm scripts in the package.json file.

- Create a circle ci pipeline to run the tests before deployment to heroku on the condition that they pass.

- Properly test the website using a screen reader to identify accessibility issues in order to fix them.

- Investigate protecting ‘404’ urls with middleware, so that users that are not logged in are taken to the login page rather than the 404 page when they go to a url that does not exist.

- When adding a new question make the correct answer selectable my drop down once the fields have been filled in to allow for a more random position when displayed as part of a question.

- When adding a new question, give the user the option to add up to 5 answers.

- Create a separate page for restricted users to see the quiz where the quiz data structure is not exposed in the javascript.

- Debug the ‘getQuiz’ query to try and decomplexify the mapping.

- Add a ‘No questions have been added in this quiz yet’ message to the view quiz page if a quiz has no questions. Consider not including quizzes that have no questions in the quizzes that are shown to Restricted and View users.

- Remove add and delete options from questions and answers that have met their constraint. Consider offering feedback to users on why there is no add/delete option in the form of text in place of where the button should be ‘e.g. questions with less than 3 answers cannot have an answer deleted’.