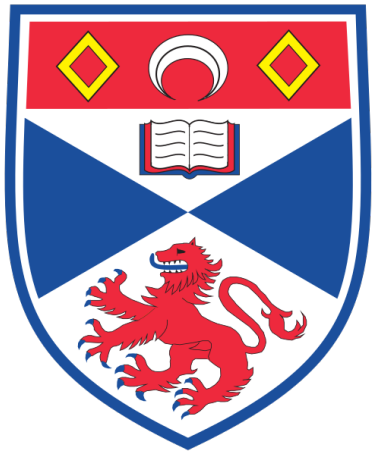


CS3099 Final Report

Group Matriculation Numbers

200002872
200002880
200011490
200003650
200017941

March 24, 2023

1. Abstract (200011490)

This report details the development process of a federated website for the publishing and solving of puzzles such as Sudoku. This website forms part of a group of sites, each of which host puzzles in their own right but are also able to communicate within the group to allow a user access to all sites with a single account. A notable element of this development process is the fact that throughout, the team has employed the agile and scrum methodologies in order to increase the rate at which features were deployed.

2. Declaration

"We declare that the material submitted for assessment is our own work except where credit is explicitly given to others by citation or acknowledgement. This work was performed during the current academic year except where otherwise stated. The main text of this project report is **9,574** words long, including project specification and plan. In submitting this project report to the University of St Andrews, we give permission for it to be made available for use in accordance with the regulations of the University Library. We also give permission for the report to be made available on the Web, for this work to be used in research within the University of St Andrews, and for any software to be released on an open source basis. We retain the copyright in this work, and ownership of any resulting intellectual property."

Contents

3. Introduction (200011490)

The aim of this project was to create a federated website for publishing and solving puzzles such as Sudoku. The user should be able to create, solve, rate and discuss a variety of puzzle types as well as log in to a number of different sites in the super-group with a single login. The project specification was deliberately vague and gave us the opportunity to be creative in the additional features that we implemented. Often in this module there is a twist thrown in for the second semester, however, this was not the case this year, allowing us to continue on with the original plan that we had set out.

The "federated" aspect of this project meant close collaboration with a super-group. We were a part of super-group 8 alongside eight other teams. The supergroup met typically once a week with one member of each group nominated to attend the meetings. Here they discussed protocols and standards for the parts of the project that would require inter-group communication.

As User Interfaces are a major part of this project, we decided it was preferable to use a library to simplify the design of those UIs. We chose React [1] as it is easy to use, popular and a number of the group members were already familiar with it. For the back end of our website we chose to use NextJS [2] as it is a large full-stack framework that has been designed with React in mind, making integration in our project easier. Finally, for our database, we have chosen to use MongoDB [3].

On the whole, our project was successful. We achieved all of the required functionalities that were set out at the beginning of the year as well as implementing a number of additional features chosen by ourselves. As a group we have all developed new skills, not only building on our previous experiences of the stack, but also learning more about technologies and libraries that were new to us. Moreover, we have all gained a better understanding of the agile and scrum methodologies and we have successfully put them into practice. Together we discovered which concepts and practices worked well and which didn't and we have adapted our way of working to be as productive as possible.

4. Use of Agile Methodology (200011490)

4.1 Agile terminology

Part of the specification for this project was to work under the agile methodology. The agile methodology is a way of managing a project that splits it up into small sections ensuring working implementations are delivered quickly. The methodology follows a manifesto which can be found in the Agile Manifesto [4]. The methodology and it's fundamental aspects through a series of lectures. This meant learning a number of new concepts and terms. These included:

- **Scrum** - a framework which allows teams to structure and manage their productivity through breaking work into small goals which can be completed in short iterations.
- **Sprint** - a fixed-length block of time in which set tasks are completed by a development team.
- **Backlog** - a prioritised list of tasks that need completed through out the development process.
- **User Stories** - functional increments of work that have been divided up and framed from the point of view of a user.
- **Kanban board** - a tool to visualise workflow consisting of columns, each of which represents a stage in the development process. Tasks are moved through the columns to allow all of the team to keep track of the status of each one.
- **Planning Poker** - A approach used by teams to estimate the workload involved in a specific user story. Each member silently and simultaneously suggests a number (according to the Fibonacci sequence) and any dubiety is discussed and a consensus reached.

4.2 Scrum

We arranged our word using the scrum framework. This involved setting up a backlog of user stories based on our initial project plan, seen in figure A.1, which we created as a team based on the requirements set out in the project specification. These user stories were then all given a priority and a number of story points, decided on using planning poker. At this point we were ready to select some user stories to add to our first sprint. Each sprint that

we ran, lasted around two weeks and contained a backlog of it's own with each user story assigned to a team member. Additionally, for each sprint we assigned two roles to members of the team. The first role is Product Manager who is the first point of call for queries or problems with the requirements. The second role is scrum master. The scrum master is the team facilitator, they monitor progress and ensure everyone in the team has what they need to succeed.

Between each sprint we held a scrum meeting, a selection of minutes taken from these meetings can be seen in Appendix B. During these meetings we would reflect on what when well in the sprint as well as what we achieved. In this meting we also planned the next sprint, adding new user stories to the sprint backlog and reassigning the roles. Then throughout each sprint, we had to keep a record of what was being achieved by each member each day. This is often done as a short daily meeting but for us, we found it worked best for our updates to be given through adding posts to a Discord [5] channel where group members can post updates whenever is most convenient to them. Discord is where the majority of our communication took place and so we created a number of dedicated channels as can be seen in figure C.1

4.3 Use of Jira

We decided to use a software called Jira [6] in order to organise our scrum. Jira is a product management software which allows teams to plan, track and manage their products. In terms of our project, Jira has allowed us to manage our our scrum and keep everyone on the same page. Firstly, we were able to visualise our product backlog, set story points and add user stories to sprints. Then, throughout the course of each sprint, we were able to use the Kan-ban board generated for us to keep track of our progress in each task. A section of our kanban board from the final sprint can be seen in figure D.1.

Finally, after each sprint, Jira generated a number of reports which allowed us to visualise our progress through figures and tables. A number of these figures were also utilised to back up our writing in many of the previous project reports.

4.4 Sprint cycles

In total we completed six sprints. Our first sprint ran from 11th October to the 25th October. For this sprint we chose the user stories which had the highest priority or were needed to implement further stories. We then split these stories into smaller sub-tasks which were more manageable for individuals to complete. Throughout the sprint, we worked on our own tasks but made sure to communicate frequently to keep everyone on the same page.We used Discord and in particular the stand-up channel to keep up do date with each other's progress. Any questions about the project requirements were asked of the product owner and any other queries were directed to the scrum master. We also introduced dedicated front-end and back-end Discord channels for more specific discussion. In hindsight, we took on too much work for the first sprint and this showed as we completed none of the tasks. Some were in testing but waiting for other elements to be developed before we could verify

their correctness. Luckily, as we had taken on so much for sprint 1, we were able to carry all the unfinished tasks into sprint 2 and add the few remaining necessary features as new stories.

Our second sprint ran between the 25th October and 11th November. This is slightly longer than our usual sprint length, but we decided to extend it up to the first deadline of the semester as it only added a few days. The backlog for our second sprint mainly consisted of user stories which had been rolled over from the first sprint. This was mainly because different user stories were being completed at different rates which in turn delayed the integration and testing of some stories. In this sprint we focused on integrating each of the individual elements that we had separately been working on so far and squashing any bugs that had arisen. We also decided to change the way we were styling our pages, introducing Bootstrap [7] rather than raw CSS. Finally for this sprint, we tested each element of our website. The integration element of this sprint meant that we were communicating more as a group. We made good use of the front-end and back-end Discord channel to question/discuss the relevant areas of the stack. We also ended up meeting in-person more often to work collaboratively. We did, however, still struggle to align our schedules well enough to have physical stand-up meetings and so we continued to make good use of our stand-up Discord channel. Another benefit of this channel is that we could look back through the messages to see when certain progress had been made and by whom.

The third sprint started after the inter-semester break and ran from the 19th Jan to 2nd Feb. This sprint mainly consisted of creating unit tests for the existing features and setting up some pages that would be required for upcoming features. Due to the nature of these tasks, this sprint was another more individual one and so it took until we reconvened at the end of the sprint to realise that not much progress had been made on the tests as everyone had other commitments. This meant that few tasks were fully completed in this sprint and a decision was taken to put testing on the back-burner for now and get some more features implemented instead. One feature which did come from this sprint was the leader-board which had not previously been in our plan but we agreed that it was a relatively straightforward and useful feature to include on our platform.

Due to the lack of progress in our 3rd sprint we set out to make a bigger impact in sprint 4. This sprint ran from 2nd Feb to 16th Feb and consisted of smaller features which we could achieve in one sprint. This included the user being able to view a profile with their details, identify mistakes in the board and use pencil markings to make notes while working through a puzzle. This sprint included more collaboration again as we were having to integrate front-end with back-end in order to fully complete tasks. In terms of the Kan-ban board, many tasks looked unfinished by the end of the sprint as they were not quite fully integrated but a great deal was achieved in these two weeks.

Our penultimate sprint ran over a week-long holiday and so it lasted three weeks rather than two. It ran from 16th Feb to 9th Mar. The tasks added to this sprint were mainly focused on adding the final piece of require functionality - an additional puzzle type. For this we chose to implement Wordoku and so the majority of the group were working on implementing that. The other few group members tried to wrap up the integration and styling of the remaining features from the sprint before. This sprint was for the most part successful and left only a few bugs to be squashed and interactions to be completed in the final sprint.

The final sprint ran from the 9th Mar to the 23rd Mar and consists mainly of tasks which aim to clean up and refine the features that we have set up. This sprint didn't bring any new user stories from the back end as we decided it was best not to introduce new features so close to the deadline. This left us with a number of unattended user stories left in the project backlog but we knew that we had aimed high with the amount of work we had proposed and we have a number of features, such as the leader-board, which have been implemented but were not part of our original plans.

Below, in figure 4.1, is a report generated by Jira which shows the number of tasks completed in each sprint. It shows that, in general, we were consistent in the amount of work we achieved throughout the six sprint cycles. The navy section represents the tasks which were added to the backlog but not implemented. It is a relatively small number and would probably have been completed with an extra sprint or two.

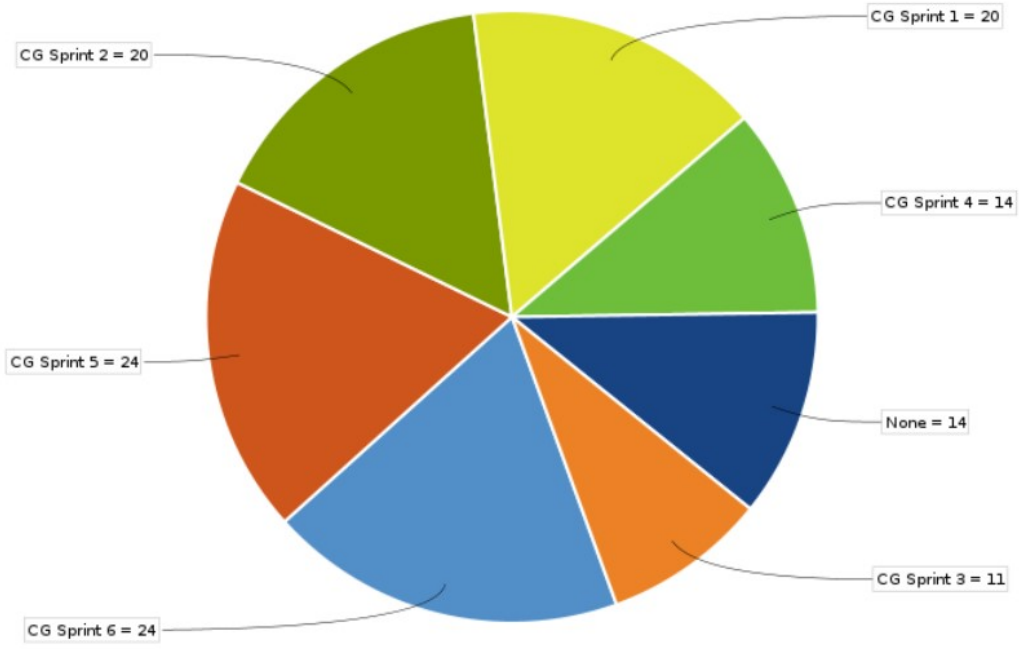


Figure 4.1: Report created by Jira, showing the number of tasks completed in each sprint

5. Supergroup Interaction (200002872)

5.1 Meetings

The supergroup organised a weekly meeting time slot, which we maintained through both the first and second semester, during which at least one member of each constituent group was expected to show up and discuss the coordination of supergroup interaction. Our group sent a member every week for which a meeting was held. Meeting minutes for each of these meetings were recorded, and are available in a zip file accompanying this submission.

Weekly meetings ended at Week 6 of Semester 2, after which point the supergroup decided to finalise all agreed upon interoperability requirements to ensure that all groups had sufficient time to implement them. This decision was made in order to avoid a repeat of the first semester, during which last minute changes were made to the interoperability requirements causing a large amount of stress and overwork. After this, the supergroup instead started organising times for supergroup members to meet up and debug code together. This was particularly useful when it came to the supergroup interoperability requirements as they typically required two groups to properly test.

Meetings were allocated an hour, though few typically filled the full time limit. During a meeting, we would suggest ideas for how to implement functionality or give feedback on previous suggestions. Often a discussion would be had in one meeting and then groups would prepare suggestions for implementation to be ready for the next meeting. This way, supergroup representatives would be able to discuss with their group on any decisions before the supergroup would vote. It also allowed members to perform research in order to make more informed decisions and designs.

Our group opted to have a single, constant representative. Some groups rotated their representative between team members. It turned out that our approach had fewer issues overall, as groups with rotating representatives were often not up to date on the current items of discussion for meetings. This led to a handful of incidents where a group's representative would agree to a certain proposal, only to have another member from that group reject it later requiring the supergroup to backtrack on its plans. In contrast, our group's representative was able to be singly responsible for ensuring that the group's issues were brought to the supergroup and were able to stay up to date by going to all supergroup meetings. This meant there was no additional time required to be caught up to speed on current plans.

5.2 Online communication

For online communication, the supergroup used a Discord [5] server to communicate meeting times and facilitate discussion. The server is set up to allow people to share useful links, discuss issues they may have, or debug interoperability aspects. The server allows users to tag themselves by their group, allowing all supergroup members to easily identify which group someone is a member of. This also allows anyone to 'ping' that group, directly notifying them of a question or issue pointed at that group in particular.

One use of the server was for longer form discussion. When changes were proposed to the interoperability requirements, there was a week long discussion on the server about the specifics which would've been hard to have in person due to how long they lasted.

The server was also useful as a document resource. It is where all meeting minutes were posted, and links to useful resources were shared between groups.

5.3 Group Testing

At specific points in time, the supergroup organised meetings where all members would test the interoperability of their websites to make sure they conformed to the interoperability requirements. These typically happened towards the end of a semester, since it would mean that all groups had ideally had time to properly implement the required features.

Turnout at these meetings was generally high, though a handful of groups did not show up at times causing frustration as groups could not confirm interoperability with them. Nevertheless, groups were able to check that their website was interoperable with a majority of other websites. Interoperability with a majority of websites was generally an indication that a website fully conformed to the specification. Our group attended all important meetings and were able to confirm interoperability with a majority of websites as seen in Figure E.1. On a whole, the meetings were useful to ensure that testing was done properly.

5.4 Interoperability Requirements

The group set out interoperability requirements, which are detailed in Appendix E. These requirements were centrally hosted on a GitHub repository, which was private to ensure that no-one outside of the supergroup could get access. This was done out of consideration for Good Academic Practice. Access to the GitHub repository was vetted within the Discord server, which all supergroup members were expected to be a part of.

The general mindset in terms of interoperability was to have it be as minimal as possible. This was the general consensus of the supergroup, as all groups were already burdened by a significant amount of work and wanted to limit any additional work as much as possible.

Some groups found themselves unable to implement the requirements properly, and in an attempt to still achieve interoperability requested special consideration be made. As such, some special cases have been made for specific groups regarding the location of their endpoints.

6. Implemented Functionality

6.1 Login/Registration (200017941)

Upon entering the website the first page that appears prompts the user to choose between either registering an account or logging in with a pre-existing account. The registration page requires the user to enter their username, email and password. If these values are valid, then the user is automatically directed to the login page where the user is required to re-enter their username and password.

6.2 Cross-site login (200002872)

As per supergroup requirements, our website implemented cross-site login capabilities. This allows anyone who has an account in the Aces Supergroup Federation to log in to our website using their account, and also for anyone from our website to use their account to log in to any website in the Aces Supergroup Federation. It stores foreign user data locally so that it can allow users who are logged in from using another website to comment, vote, and perform all activities in the same manner as a local user. It follows the protocol and design set out by the supergroup, which is based on the an OAuth2 authorization flow [8]. This requires exchanging a secret key for each website beforehand, which was done within the supergroup. An example of OAuth2 authorization flow is visible at Figure E.2.

6.3 Puzzle List (200003650)

The puzzle listing pages give users a view of all the available puzzles on the website as stored in the puzzles database. Any puzzles the users have created are also included. The listing of Sudoku and Wordoku puzzles are on separate pages by querying the database on the variant of puzzles. Each individual puzzle card displays the name of a puzzle, its corresponding difficulty and the rating voted by users across the site. Users can click on puzzle cards and be taken to a page that initializes the corresponding puzzle.
We got most of our puzzles from Krazydad Sudoku [9].

6.4 Solve Sudoku Puzzles (200002880, 200003650, 200017941)

Users can access various Sudoku to interact and fill in the Sudoku with their own answers. This page, sudoku-page.js, accesses the Sudoku component defined in sudoku.js to display the Sudoku puzzle. The sudoku-page queries the database for a puzzle via its id and passes the result to the Sudoku component which it uses to initialise the puzzle. Users can interact with the board by clicking on the desired box which highlights it. Users have the option to input their solution with their keyboards or by clicking on the 3x3 number grid next to the Sudoku. The number grid also provides an option for users to input temporary values, pencil markings, into the Sudoku. By clicking on the pencil mode button, users enter temporary input mode and any values input this way will be gray and not recorded by the Sudoku board.

The Sudoku component has functions to find any real-time mistakes on the board and highlights mistakes in red, as well as to detect whether the board is solved. When the puzzle is solved, users will be notified by a popup and have the option to explore other Sudoku. The puzzle is stored in json format and can be downloaded from the sudoku-page to be transferred between federation sites. Puzzles meeting the format can also be uploaded on the create-sudoku page.

In addition, the sudoku page also has a comment section attached to it which allows all users solving any particular puzzle to add or edit their comments under the puzzle board. All the comments are permanently saved so upon entering the puzzle page, any comments that have previously been entered will be displayed chronologically.

6.5 Create Sudoku Puzzles (200002880, 200003650)

Users can interact with an empty Sudoku board to create their own puzzles. This Sudoku board is displayed by accessing the Sudoku component via the create-sudoku.js page. Users have access to the 3x3 number grid functionality to interact with the board but also have the ability to use their own keyboard. The pencil mode, temporary value entry, is also possible to aid the user. There is also a function for the board to automatically find the solution (the solution is not shown to the user), used to validate the puzzle when it's created to ensure that the puzzles store their own solutions, to be compliant with the supergroup defined puzzle format. Puzzles must have only one solution to be validated properly.

6.6 Wordoku (200002880)

The Wordoku is built upon the sudoku, since it is a largely identical game, with much of wordoku.js being copied from sudoku.js in the interest of time, though it would have been better for the wordoku to inherit from the sudoku.

To play Wordoku, the user can access a variant of the sudoku page which displays a list of wordoku puzzles. The same sudoku-page is used, calling the relevant Wordoku component if it sees that the puzzle it's gotten is a wordoku.

The wordoku is largely identical to the sudoku, only instead of using the numbers 1-9 it uses

the letters (or just characters) in a 9 letter word. This word is displayed for the user.
To create a wordoku, like the sudoku page, the create sudoku page can also be used to create a wordoku puzzle. This works largely identically to the sudoku but also requires that the user enter a word to base the wordoku off of.

6.7 Puzzle Upload/Download (200002880)

The user can upload or download puzzles in a format that is compatible between federated sites. This format was agreed upon by the supergroup.
The create page allows for one or more files to be uploaded directly. Simply use the relevant button to browse and select files then upload them.
The sudoku page allows for downloading the current puzzle. This will download the puzzle as stored in the database, no changes made on the page will affect it.
This functions identically between wordoku and sudoku.

6.8 Puzzles API (200002872)

The website offers an API through which people can access the puzzles offered. This API follows the supergroup interoperability requirements as detailed in Appendix E. This allows for a resource through which other programmers can easily access the puzzles that we have, which is much easier than having them manually download each puzzle.

In Figure E.3, it can be seen that on Group 2's websites our puzzles are listed and solvable. They are making use of this API to access these puzzles.

6.9 Profile Page (200011490)

The profile page contains a profile about the user with a set user avatar, their username, their email address and their current rank. Additionally, this page includes an ordered list of all of the puzzles that have been completed on the platform by the user since their account was registered. The page follows the same styling as the leader board page and the home page, all of which are slightly different to but consistent with the puzzle-related pages.

6.10 Leader board (200011490)

The leader board was an additional feature that we had not originally planned to add. It contains all of the registered players, ordered by their rank. This adds a competitive aspect to the platform as users are encouraged to complete more puzzles in order to improve their rank and work their way up the leader board. The page follows the same styling as the profile page and the home page.

6.11 Voting (200002872)

Each puzzle starts at a value of zero, and each user can individually either give a puzzle an upvote, downvote, or no vote at all. This effects the puzzle's rating, with an upvote contributing +1 point and a downvote contributing -1 point. Combined, this score shows on the puzzle listing page to give an indication of how popular a given puzzle is. Ratings are shown in orange if positive, and blue if negative for an at a glance indication of whether a puzzle is seen as good or bad on the whole. This system is inspired by the rating system used by Reddit [10].

7. Changes in Plan (200003650)

As mentioned in our initial plan (Appendix A), we have split the functionalities we will implement into two parts. The basic features are required and should be implemented, while the additional features can be chosen to be implemented depending on the time the group has afterwards.

7.1 Basic Features

The basic features from our initial plan are all implemented and fully functional. Users have the ability to register with their desired username and password and be able to log in with their respective combination. A fully functional Sudoku board is implemented, allowing interaction and checks if users have provided the correct solution to the Sudoku. The create Sudoku page provides users with the ability to add further Sudoku puzzles to the set of puzzles provided by the site. Users can use their username password combination to log into sites created by other groups within the Supergroup Aces. Users can also transfer puzzles between the sites in the Supergroup by downloading the puzzle and uploading the puzzle file on the other site. Likewise, the same action can be done to upload puzzle files on our site. The site is easy to navigate and interact with well-labeled buttons. Users have options to interact with the Sudoku board using the number grid and their keyboard.

7.2 Additional Features

The group has attempted to implement as much of the additional features to the best of our abilities. As anticipated, not all of these features are implemented due to a mix of time limitation and challenges faced in development. Another puzzle type, the Wordoku is implemented and can be interacted and solved like the Sudoku. The site offers validation and Wordoku creation along with other Sudoku functionalities on the Wordoku. However, the pencil mode and 3x3 grid for input is not provided for Wordoku interaction due to time limitation. Rating with an up-vote or down-vote option is implemented with two buttons available for interaction and accumulating vote value for each puzzle, displayed on the puzzle listing page and the puzzle page itself. Puzzle difficulty has been included as a supergroup puzzle format and thus was implemented integrated in the Sudoku board and API. The difficulty of each puzzle is displayed in the puzzle listing page and on the puzzle page itself. A comment section at the bottom of each puzzle page is implemented with the ability for each user to write their own comments and post it in the comment section. Each

comment can be edited with an edit box displayed beneath it. Profile pages are implemented providing information about each user and a list of puzzles they have solved on the site. In terms of a rank system, the leader board is implemented to display the ranks of all the users on the website. The ranks are based on the amount of puzzles a user has solved and also the difficulty of each puzzle solved. Pencil markings, temporary value inputs on the Sudoku board, is implemented to help the user with Sudoku solving. Music playing in the background was attempted but was unable to be implemented due to time limitations and challenges faced in development. Other additional features were agreed by the team to not be attempted as little time was present, and any time left was only sufficient for polishing the existing features and managing any potential errors in testing.

8. Summary of Testing

8.1 Front-end (200003650, 200002880)

Testing of the front-end is done during development and at the end of development with the use of web browsers (e.g. Firefox and Google Chrome). This method of testing is chosen over unit testing because the latter is too time consuming and challenging to be done with React. Since front-end testing is mostly about the display and operation of the site, there is not an immediate need and importance of unit testing over the use of web browsers.

The use of web browser for testing is used during development to ensure any flaws and errors are detected towards the early stage of development. Moreover, this method can also ensure that the site is displayed and styled as desired.

At the end of development, a comprehensive table (see Appendix F.1) is drawn up containing tests with desired output and performance from the site. The table is went through manually and tested by group members on different web browsers with various screen proportions and operating system. The output of the tests are recorded in Appendix F.

8.2 Back-end (200017941)

For the most part unit tests were not used for testing the back-end's performance. This was mainly due to the initial learning curve being quite steep and later on having the manage all the user stories and having them implemented. All of this was proven to be quite challenging, but that being said, the testing strategy involved manually testing the code in two main ways. The first way in which the code was tested is through black-box testing. This is where the code itself is ignored and the only thing of concern is the output based on the initial input. So for example, when it came to testing the functionality of an API that allows users to login, the login details of the user that has been entered is recorded followed by the output produced by the API. In this case it would be whether the user login details were valid or invalid. The second way in which testing was conducted was through white-box testing. This involves a dry run where each and every line of code is checked to see whether it processes the data correctly. This involved placing several print functions at appropriate places to ensure that the program flow is being traced and at each point whether the actual output matches the expected output. Therefore a note is made at each point in the program beforehand as to what the correct value is supposed to be.

The testing table is visible at G.1.

8.3 Database (200002872)

Database access testing is done through unit tests written using Jest, a testing suite for JavaScript with support for React and a focus on simplicity [11]. Unit tests are useful for this purpose, as database methods are typically atomic and are easier to predict than an event driven interface. The tests are written at `__tests__/database.test.js`. The difficulty with running tests on the database is that they perform stateful operations on the data itself, which would cause the database to be negatively effected by the operation of tests. As such, for typical testing purposes, these tests can only be run on a local test database or a dockerized database instance. For basic testing the former is used, while for the continuous integration pipeline the latter is used.

One issue encountered with the testing pipeline was that at some point the MongoDB docker image updated, causing issues as it had changed some formatting. As such, the image had to be kept at an older version.

To ensure continuous integration, a testing pipeline was set up within the GitLab repository. This has been useful as it ensures that any new database methods function as expected thus limiting the bugs that may appear from their use. Using Jest also allows the tests to display test coverage, visible in Figure H.1, which is helpful in identifying how much of our code is currently being properly tested. An example showing all tests passing is visible at Figure H.2, and the outputted list of all tests is shown at Figure H.3.

9. Evaluation

9.1 Comparison with specification (200011490)

Our platform meets all of the requirements laid out in the original project specification. The specification set out the following necessary elements:

- Your server hosting a puzzle website, all sites hosting Sudoku and each site providing other types of puzzle.
- A federation of sites in your supergroup.
- Ability of a user to access all puzzle sites of the federation with a single account, but have different roles in each site (e.g. solver, setter, administrator, etc.)
- Ability to migrate puzzles between sites where both sites understand that type of puzzle.
- Other features (possibly many!) to make your site and/or federation stand out!

We have a functioning platform which allows users to register and log-in and then create and complete Sudoku puzzles. Our platform also hosts an additional puzzle type in Wordokus. Our site forms part of a federation. As seen as in the testing above, all users who are registered with a single website in the federation should be able to use their login details to log into any of the other sites in our federation. Additionally, within this federation, we are able to transfer puzzles with other groups through our upload and download features. Furthermore, we have implemented a number of additional pages such as a leader board based on the user's experience and a profile page to display the user's details. There are also a number of additional features added to the puzzle page itself including comments, up-votes and down-votes, mistake highlighting and pencil markings.

9.2 Comparison with Sudoku.com (200002872)

Sudoku.com [12] is a website with a similar purpose to ours, that is an online collection of Sudoku puzzles.

The first notable thing about Sudoku.com is that rather than showing you a list of puzzles, it only shows one puzzle at a time allowing you to request a new puzzle whenever you want. This one-page design is very different to our approach. It is clean, reduces navigation work and therefore makes things easier on the user, and makes it easy to find a random puzzle

to solve. Out site's approach, however, makes specific puzzles more important. It allows for the sharing of a specific puzzle, which is fun and promotes website growth by allowing for natural dissemination of the website between friends.

Sudoku.com also lacks a comment section for puzzles or puzzle rating, both of which promote user engagement on our website and are an attempt to create a proper sense of community. This comes from the fact that users do not need to log in in order to access puzzles on Sudoku.com. This again makes the website more easily accessible. By enforcing a user account, however, our website can keep track of user progress and award experience. This allows our website to display all leaders on a leaderboard, increasing the competitive nature of the site and hopefully increasing the sense of a community to retain players.

Sudoku.com offers users a limited number of hints, giving them clues as to what to look at for solving the sudoku. This is an interesting and useful feature, as it helps the user to learn sudoku solving techniques while they play and means players are less likely to get angry and quit out of a difficult puzzle. This would've been good to include, however it is very technically advanced and not something we could've easily included given our technical skill and time limitations.

Sudoku.com implements a timer, so users have the incentive to complete sudokus as quickly as they can. This adds a fun competitive aspect to the website, and it something that we were hoping to implement but did not have sufficient time to add.

Overall, despite both being puzzle solving websites our website and Sudoku.com have very different approaches. While Sudoku.com has a much nicer puzzle solving experience, our website is fleshed out with a more selective puzzle catalogue.

9.3 Comparison with Puzzler (20017941)

Another website we chose to evaluate our website against is Puzzler [13]. After comparing the two, there are a few noticeable similarities, however this is overshadowed by the amount of differences there are. This is mainly due to how much more content Puzzler incorporates.

To start off with the similarities, both websites display the puzzle icons as a grid which is a neat and compact way of designing the User Interface. Both also have a pop-down menu which allows the user to navigate to different parts of the website much more conveniently. The last noticeable similarity is that both have a user profile page which displays information about the user.

On the other hand, when it comes to the differences, the first thing that's obvious is that Puzzler offers a much wider range of puzzles, making their website much more appealing and engaging. They also have a store allowing users to purchase items from a shop and even magazine subscriptions. Additionally, they also have various puzzle apps, though comparisons should not be made to these features as the website was created by a company and therefore has monetary value, whereas our website is merely an assignment. Moving onto more quality-of-life features, Puzzler incorporates features such as a search bar, a back button, an indicator as which path in the website the user is currently on and lastly a description of how each game is played. All these features put Puzzler ahead in terms of convenience. Though on the flip side an argument could be made about how much more straightforward our User Interface is due to its minimalist aesthetic, but this boils down to our website not being as

packed with content. Additional features present on their website include a timer and the ability to logout, which would have been a useful functionality to have on our website as it is only logical to give users the ability to logout when they can already login. There are also stylistic features, for example the board in their website can be changed between light and dark mode which is always good as it allows the user to choose what they prefer.

However, there are some features we have implemented which is absent in their website. These include a point system which allows users to earn points through solving puzzles. This naturally leads to being able to have user statistics which are displayed in the profile page of our website. This makes it a lot more meaningful to have a user profile compared to their website which only displays the first and last name of the user. Another feature which is brought about as a result of having a points system is a leader board where all the users on the website are given a ranking. This is a neat addition as it makes the experience of solving puzzles more interesting and fun, introducing a competitive aspect to it. Thought It has to be mentioned that their website has several different competitions which users can participate in. Lastly the main way in which our website differs compared to their website is that ours is entirely run by the community. We give users the freedom to create and publish puzzles. We also allow users to rate different puzzles based on whether they like it or not and each puzzle board has a comment section attached to it. All these features strength the sense of community which our website provides. This is noticeably absent from their website as the company runs it and therefore controls the content.

In conclusion, our website does not have as much to offer compared to theirs in terms of the various puzzle variants and other additional features. But one thing our website does do better in is encourage users to shape the website and for it to be built by the community. And this sense of community is something that is very valuable.

9.4 Comparison with Live Sudoku (200003650)

Live Sudoku [14] is another website that provides comprehensive Sudoku solving functionalities, with a collection of Sudoku puzzles and functionalities similar to what our group aimed to achieve.

The website has a minimalist user interface design that differed from our site. While our site is styled to be more visually appealing with the use of color and layout, Live Sudoku is simple and minimal which has its own advantages. For instance, users can easily access all the functionalities of the site on its home page. Nevertheless, the design can also come off as boring and bland to users looking for a more leisure or interactive experience with Sudoku solving.

Live Sudoku contains similar functionalities on its Sudoku page as our site with number pads, pencil markings, cell highlighting and checking solutions, etc. In addition, it supports additional features that our group wanted to implement such as a timer, saving progress, etc. The website also provides comprehensive tips and textual explanations on their buttons and functionalities, which would be a nice addition to our site. Live Sudoku also displays tips and techniques on solving different Sudoku difficulties which we did not consider about as one should not assume that all users know how to play Sudokus on different difficulties. On the other hand, the comments and voting functionalities present on our site is not provided

in Live Sudoku. The presence of these two functionalities give our site a more playful environment compared to Live Sudoku, which seems to have a more serious play style. Live Sudoku also displays a list of similar levelled Sudoku and a leader board specific to each Sudoku displaying users that solved that Sudoku and their respective time. This could be a good feature to our site, but at the same time could change the tone of our site into a much more competitive community.

Similarly to our site, Live Sudoku provides options for users to sign in and register in order to play multiplayer and earn points from solving Sudoku. Unlike our site, however, users are able to solve Sudoku without signing in, at the cost of losing progress on their Sudoku. Live Sudoku allows users to log in with their Facebook which can be more convenient and allows a sharing feature at the same time. This could be a good addition to our site which can also allow more publicity on the site.

Moreover, Live Sudoku implements a live chat feature on their homepage that allows all users, including non-registered ones, to type a live message that is broadcasted to others viewing the live chat. Nevertheless, the purpose of this chat considering its location on the homepage and the risk of inappropriate messages from non-registered users is too insignificant. A similar chat re-allocated to a specific Sudoku page or multiplayer situation could be a good addition to our site to increase PvP, user vs user, interactions.

Live Sudoku, on the other hand, does not provide other puzzle types such as Wordoku on our site. They also do not provide users with the ability to create their own puzzles. Which as mentioned in previous sections is an important function to our aim of a community-driven site.

9.5 UX Evaluation (200011490)

The user experience of our platform has been analysed using Google Lighthouse [15]. Google Lighthouse is an automated tool for auditing for performance, accessibility, SEO, etc. of a web pages. For our platform, we have used the login page which is similar in styling to all other pages and I have audited accessibility and best practice.

The accessibility tests include a number of checks including image elements having alternative text, the contrast ratio of background and foreground colours, and heading elements appearing in sequentially-descending order. Of the 27 accessibility tests that Lighthouse deemed relevant for the given page, all except one have passed. Evidence of this can be seen Appendix H. The one test that has failed is *"HTML element does not have a [lang] attribute"*. Lighthouse states that this is could cause a problem as "If a page doesn't specify a lang attribute, a screen reader assumes that the page is in the default language that the user chose when setting up the screen reader. If the page isn't actually in the default language, then the screen reader might not announce the page's text correctly". This would have been an easy fix had we had more time.

In terms of the best practice tests, these included checks such as using HTTPS, avoiding deprecated APIs and serving images with appropriate resolution. Our page passes 92 percent of the tests which were run from this category. Those which failed included a browser error being logged on the console, which does not affect the user experience, and a missing source map for large-first party JavaScript. Source maps translate minified code to the original

source code are used to help developers debug during production and therefore has no affect on the UX either.

Overall, the results are very positive and show is that we have a good user interface which provides the user with a satisfactory experience.
Additionally, we have completed a heuristic evaluation of our side, comparing it's usability to Nielsen's 10 usability heuristics as described by the Nielsen Norman Group [16].

- The first heuristic states surrounds "Visibility of system status", stating that the user should always be kept informed about what is going on. Our platform does this well through informative alerts on both the login page and the puzzle solving page.

- The second heuristic looks for a "Match between System and the real world". This means that the design should use language and concepts already familiar to the user. In our platform this can be seen through the layout of our pages matching those of similar websites that already exist. Additionally, our ability to add pencil markings imitates the way in which many people will take notes with a pencil if completing a puzzle on paper.

- The next heuristic enforces "user control and freedom". This is implemented in our platform through the "Solvesudo" logo in the top left of the each page which gives the user an easy way to return to the home page from anywhere in the website and also through the menu drop-down which allows a user to navigate between selected pages.

- Another heuristic surrounds "Consistency and standards". Our platform shows consistency through the styling as, while not all pages are the same layout they use the same colours, fonts and button labels.

- The fifth heuristic is regarding error prevention. It deals with designing the interface to best prevent problems before they happen rather than trying to fix them once they do. There are a number of examples of this in our platform. To begin with, the page for creating puzzles contains a brief set of instructions as we found that the different creation methods could get confusing. Furthermore, there is input validation built in to the login and registration forms which will prevent users from entering data which is going to return an error.

- The sixth heuristic is in regards to "recognition rather than recall". The idea is that the user should not have to remember information from one part of the interface to another. This can be seen in our platform through the fact that so many of the extra features are included on the puzzle page itself. This way, the user does not have to remember anything about a puzzle in order to vote it up or down, download it or leave a comment. These features are all available on the puzzle page for each puzzle so the user knows exactly which puzzle they are referring to or interacting with.

- The next heuristic is to do with "flexibility and ease of use". The aim of this heuristic is to allow users to use the features in different ways depending on their goal. This is apparent in our platform through the choice of features that the user can choose from. For example, the user may want to speed through the puzzles to build up

their experience, in doing so they may avoid the pencil markings feature all together. Alternatively, the user may wish to spend more time on the puzzle and make use of all of the additional features including the chat and the pencil markings.

- Another heuristic focuses on "aesthetic and minimalist design". An important element of this is prioritising displaying the content and features which best support primary goals of the interface. Our platform design does this through ensuring that the most visible and prevalent piece of content on the relevant pages is the grid that is needed for the main focus and goal of the platform, solving puzzles. All other features are off to the side and do not grab the user's attention away from the puzzle itself

- The penultimate heuristic is "help users recognize, diagnose, and recover from errors". This one is clear to see in our platform as we have introduced the highlighting of cells which have an incorrect input in them. This allows the user to immediately see that there is an issue with a particular cell of their solution.

- The final heuristic surrounds "help and documentation". This one talks about how it may be necessary to provide documentation to help users understand how to complete their tasks. We have done this with the block of instructions at the top of the puzzle creation page. This page needs documentation as it is likely the most confusing page we have since there are different sets of steps for different types of puzzle creation

9.6 Security (200002872)

In terms of security, the only real consideration we need to take into account is user information. The only sensitive information we gather is passwords.

All passwords in our database are hashed using the bcrypt library, and are stored seperately from other user information to minimise the chance of programmer error making them accessible. No functions exist within the code to directly access these hashed passwords, only to compare data to them. When new user information is registered, the passwords are transmitted in plaintext to the back end where they are hashed. This is secure because all communication is running over HTTPS, thus encrypting the data and ensuring it isn't intercepted mid transit.

On the database end, access to the database itself is restricted and it is necessary to log in with an admin account to access any information. This secures the database itself from direct attacks. Authentication information for the database is stored in a configuration file that is not accessible within the repository. As such, user passwords are stored very safely within our database and attempts have been made to protect their release from malicious attackers and programmer incompetence.

One other important piece of secret information is the website's `client_secret` which it uses as authentication when performing a cross-site log in. This is secured by only being transferred from the website's back end over HTTPS, meaning neither the user nor any attackers can access this. If this information were to be leaked, then a fake website could pretend to be ours and log in to other websites within the supergroup.

10. Conclusion (200011490)

10.1 Key achievements/downfalls

Looking back on this project we are able to identify a number of achievements that we have made as a team. Firstly, we are proud of the fact that we have met all functional requirements. This applies both to now, in our final deliverable, and also to the minimum viable product which we had to submit at the end of semester one. Another accomplishment is the user interface of our platform. Since incorporating bootstrap, it is bright and fun which fits the nature of the platform and its use, but is also practical and fit for purpose. Moreover, as seen in the UX evaluation above, it is accessible for a broad range of users. On the other hand, there are also some inevitable downfalls in the platform we have produced. These include the non-puzzle related features available to the user. While we regard the user profile page as a good addition, it lacks a few extra features which could have improved it. Examples of these include the ability to input their own profile picture or their pronouns and the ability to update their details or even delete their account. Additionally, we are aware that some of the pages have uncoordinated interface designs. This is due to a number of people working on the front-end of the same page and could have been resolved with a bit more downtime to polish things up.

10.2 What we learned

Throughout the course of the two semesters in which this project ran, we have ll learned a lot. To begin with, many of us started with very little or no knowledge of the Agile Methodology [4]. This project has taught us not only the theory behind the concepts of scrum and agile but also gave us first-hand experience of implementing the framework into the way we work. Additionally, the implementation of the platform itself has taught us all lessons. Some of us had never used React [1] or NextJS [2] before and so learning to use these to achieve the tasks we wanted to achieve was a learning curve. However, even those who had experience with these frameworks have learned something new. The nature of working in a group of this size meant that everyone had to drop into each others areas of expertise to help out. Between front-end, back-end and databases, everyone gained more experience in the areas that they were less confident in. Finally, we all had to learn how to manage our time between this module and our others. The fact that the project was constantly active through all of our other deadlines means that we had to ensure we were working sprints around the other modules that people in the group were taking as well as

any other non-work-related commitments. While we didn't always nail this concept, when we did it allowed us to continue to be as productive as possible through each sprint and not end up with outstanding tasks building up and rolling over into the next sprint

10.3 What we would have done differently

As to be expected with a project of this size, there are certain things that we would have done differently. Firstly, we would have taken more care to keep the code efficient and readable. This was often difficult with everyone working together and with different levels of expertise. However, it would have been of benefit to us as it would have made the code quicker and easier to read. This would have benefited the less experienced users as they tried to understand existing code in order to integrate their own work but would have made debugging a much nicer process. Secondly, we feel we should have been conducting more unit tests while we were developing new features. This was our plan to begin with and would have meant we wouldn't have been left with so much testing, and in turn debugging, to be doing just before deadlines. Instead, in most cases we made it more stressful for ourselves by putting testing off until the last minute. Finally, in terms of communication, we could have been more intentional in organising stand-up meetings. While the discord channel described before was a better option for us, it was was not used as often as it should have been. As a team, we did communicate a lot but not through daily stand-ups meaning it was possible to go for long periods of time without hearing from someone if they weren't being vocal in the general chat. Enforcing stand-ups better would have been a good way to ensure we heard progress from every group member every day.

References

[1] [Online]. Available: https://reactjs.org/

[2] [Online]. Available: https://nextjs.org

[3] [Online]. Available: https://www.mongodb.com/docs/

[4] [Online]. Available: http://agilemanifesto.org/

[5] [Online]. Available: https://discord.com

[6] [Online]. Available: https://www.atlassian.com/software/jira

[7] J. Thornton and M. Otto, "Get started with bootstrap." [Online]. Available: https://getbootstrap.com/docs/5.3/getting-started/introduction/

[8] "Oauth2 authentication for api access: Api documentation." [Online]. Available: https://www.aha.io/api/oauth2

[9] "Krazydad Sudoku," Mar. 2023, [Online; accessed 24. Mar. 2023]. [Online]. Available: https://krazydad.com/sudoku

[10] "Reddit." [Online]. Available: https://www.reddit.com/

[11] [Online]. Available: https://jestjs.io/

[12] "Sudoku.com." [Online]. Available: https://sudoku.com/

[13] "Online puzzles." [Online]. Available: https://www.puzzler.com/online-puzzles

[14] [Online]. Available: https://www.livesudoku.com

[15] [Online]. Available: https://developer.chrome.com/docs/lighthouse/overview/

[16] [Online]. Available: https://www.nngroup.com/articles/ten-usability-heuristics/

[17] Auth0, "Code flow." [Online]. Available: https://auth0.com/docs/get-started/authentication-and-authorization-flow/authorization-code-flow

A. Initial project plan

The functionalities of our project will be split into two, where for the first semester, we will be focusing on implementing a few basic features which are required, and for the second semester additional features which we decided on will be implemented.

Basic features:

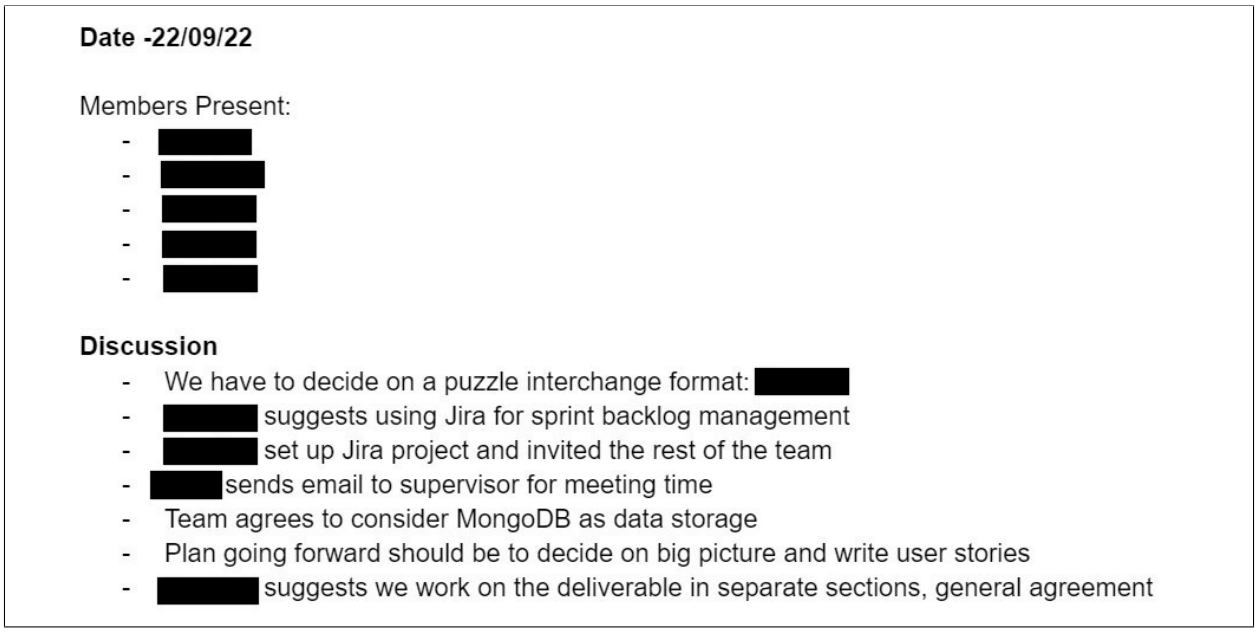
- The ability for users to register their own account
- An initial set of standard Sudoku puzzles which the user can solve
- Allowing further Sudoku puzzles to be added to this initial set by users
- The ability for users to login to all puzzle sites in the federation
- The ability to transfer puzzles between the sites in the federation
- A simple and usable user interface

Additional features:

- Other puzzles such as Wordoku and crossword
- Different game modes:
    - Timed puzzles: user race against the time to solve the puzzle as quickly as possible
    - Versus: different users battle against each other to see who solves a puzzle first
- Being able to record puzzles when solving them
- Ratings for each puzzle which done by the user either upvoting or downvoting a puzzle
- Each puzzle to have its own level: easy, medium, and hard
- A comment section with each puzzle where users can communicate with each other
- A live chat allowing players who are playing against each other to interact
- User profiles which display certain information about the user such as the number of puzzles solved
- A rank system where users can earn points from solving puzzles based on the difficulty, and for this to add to their level
- Hints provided whenever the user requires assistance
- Pencil markings the user can use to help them solve the puzzle
- The ability for user to enhance the aesthetic of their puzzles by decorating it
- The ability for users to earn coins to unlock cosmetics such as puzzle skins
- Music playing in the background
- Auto generated Sudoku puzzles to be added for users to solve

Figure A.1: Initial plan of features to be implemented, taken from our first deliverable

B. Selection of meeting minutes



Date -22/09/22

Members Present:

- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]

Discussion

- We have to decide on a puzzle interchange format: [redacted]
- [redacted] suggests using Jira for sprint backlog management
- [redacted] set up Jira project and invited the rest of the team
- [redacted] sends email to supervisor for meeting time
- Team agrees to consider MongoDB as data storage
- Plan going forward should be to decide on big picture and write user stories
- [redacted] suggests we work on the deliverable in separate sections, general agreement

Figure B.1: Meeting minutes from first group meeting

Date - 11/10/22

Roles decided for sprint 1

- Scrum Master - [redacted]
- Product Manager - [redacted]

- Split user stories into tasks
- Assigned them to team members
- Started sprint
- Made sure everyone was happy with GitLab
- Set up Next.js project
- Set up front-end and back-end channel in Discord

Figure B.2: Meeting minutes from sprint 1 panning meeting

Date - 24/10/22

What have people been doing?

- [redacted] working on databases
- [redacted] two pages, admin and registration page and main css file
  - Page for listing puzzles also
- [redacted], working on login endpoint
- [redacted] worked on sudoku's internal representation
- [redacted] ill so not much work but planning on finishing

Nnamdi recommends helping ill members by taking on some work
Talking about continuous integration, Nnamdi recommends getting it sorted.

Meeting swapped from 3-4pm on Tuesday.

Figure B.3: Meeting minutes from a Supervisor meeting

C. Discord set-up

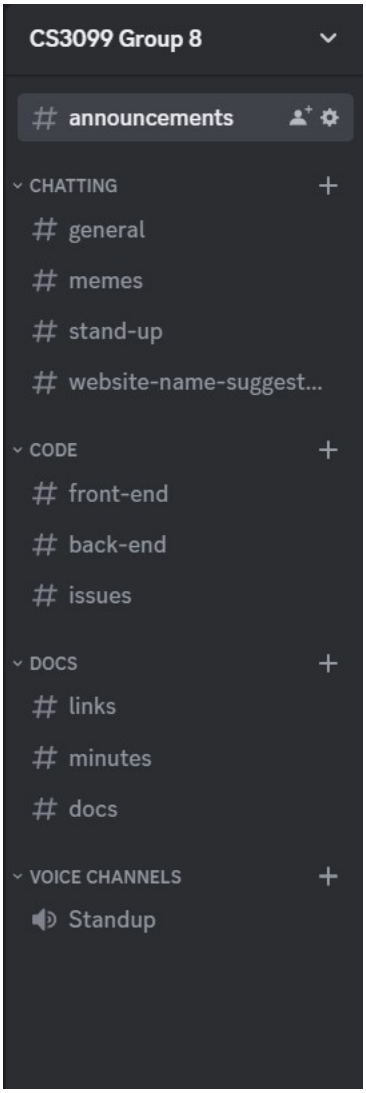


Figure C.1: Screenshot of the Discord channels we created for group communication

D. Jira generated Kan-ban board and report

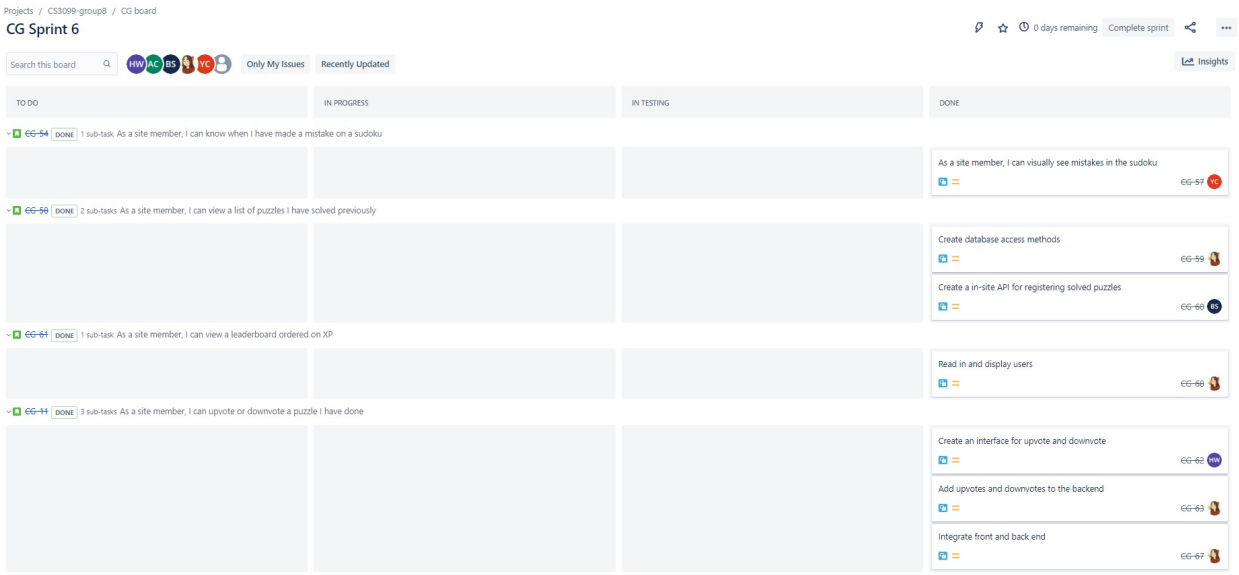


Figure D.1: Screenshot of a section of our Kan-ban board on Jira, as we finished up the last sprint

E. Supergroup Interoperability

Site logging into → Account provider ↓	1	2	3	4	5	6	7	8	9
1	-	✓							✓
2	✓	-	✓	✓	✓	✓	✓	✓	✓
3	✓	✓	-	✓	✓	✓	✓	✓	✓
4	✓	✓	✓	-	✓*	✓	✓	✓	✓
5	✓	✓	✓	✓	-	✓	✓	✓	✓
6	✓	✓		✓	✓	-	✓	✓	✓
7		✓		✓	✓	✓	-	✓	✓
8	✓	✓	✓	✓	✓	✓	✓	-	✓
9	✓	✓	✓	✓	✓	✓	✓	✓	-

Figure E.1: Table of Supergroup Interoperability [17]

E.1 API Requirements

- /oauth/authorize
- /oauth/token
- /api/user/me
- /api/puzzles

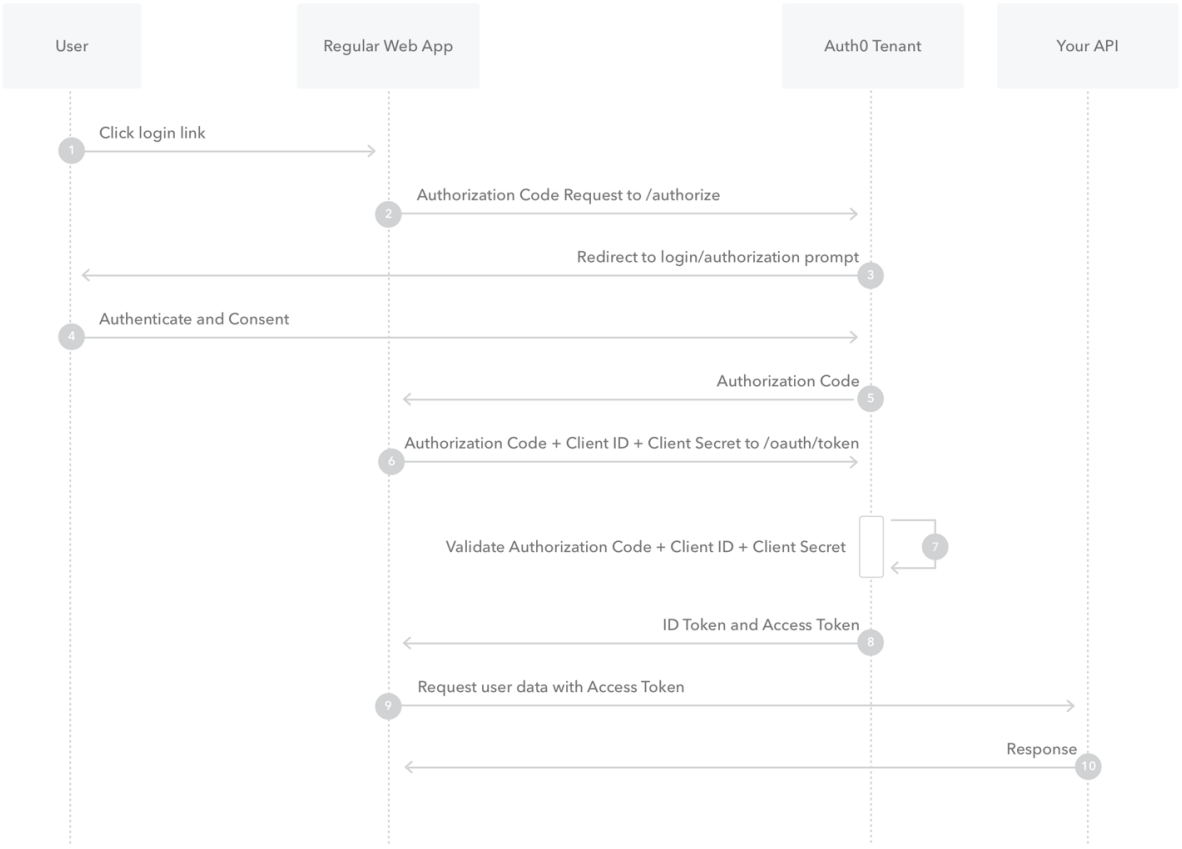


Figure E.2: OAuth2 Diagram of Flow

Figure E.3: Group 2 using our puzzle API to display our puzzles

F. Front-end testing

No.	Test	Expected Result	Actual Result
	Sudoku Creation		
1	Attempt to create invalid sudoku with too many solutions	Can't validate sudoku	✓F.1
2	Attempt to create invalid sudoku with no solutions	Can't validate sudoku	✓
3	Create sudoku	Sudoku created and available from list	✓
4	Validate sudoku then change it then attempt to save	Should require revalidation	✓F.2
5	Upload sudoku file	Sudoku is added to list	✓
6	Upload multiple sudoku files	Sudokus are added to list	✓
	Sudoku Interaction		
7	Download sudoku file	Current sudoku is downloaded to local machine	✓F.3
8	Play and solve created sudoku	Upon solving website alerts user	✓F.4
9	Select editable Sudoku box	Box is highlighted with purple	✓F.5
10	Select Sudoku box with existing value	Unable to select	✓F.6
11	Select Sudoku box and input with keyboard	Value is displayed in box	✓F.7
12	Select Sudoku box and input with number pad	Value is displayed in box	✓F.8
13	Turning on pencil mode and input with number pad	Grey value is displayed in box, Sudoku is not solved	✓
14	Input wrong value in Sudoku	Value and same values in the same row/column are highlighted with red	✓
15	Solve Sudoku	Alert window pops up with redirect to puzzle list page	✓
	Wordoku Creation		
16	Attempt to create invalid wordoku with too many solutions	Can't validate wordoku	✓
17	Attempt to create invalid wordoku with no solutions	Can't validate wordoku	✓
18	Create wordoku	Wordoku created and available from list	✓
19	Validate wordoku then change it then attempt to save	Should require revalidation	✓

20	Upload wordoku file	Wordoku is added to list	✓
21	Upload multiple wordoku files	Wordoku are added to list	✓
	Wordoku Interaction		
22	Download wordoku file	Current wordoku is downloaded to local machine	✓
23	Play and solve created wordoku	Upon solving website alerts user	✓
24	Select Wordoku box with existing value	Unable to select	✓
25	Select Wordoku box and input with keyboard	Value is displayed in box	✓
26	Input wrong value in Wordoku	Value and same values in the same row/column are highlighted with red	✓
27	Solve Wordoku	Alert window pops up with redirect to puzzle list page	✓
	Comments		
28	Create a comment	Comment shows up in comment field with username	✓
29	Edit existing comment	Edited comment is displayed in place of previous comment	✓
30	Re-entering page	Existing comments are displayed	✓
	Voting		
31	Click on upvote button	Vote value is increased on page and on listing page	✓
32	Click on downvote button	Vote value is decreased on page and on listing page	✓
	Navigation		
33	Click on site logo on navigation bar	Navigate to home page	✓
34	Click on menu on navigation bar	Menu list drop down	✓
35	Click on Sudoku puzzles on puzzle list page	Direct to respective Sudoku page	✓
36	Click on Wordoku puzzles on puzzle list page	Direct to respective Wordoku page	✓
	Supergroup		
37	Exchanging puzzles with other sites	Puzzles from other sites work when uploaded, can upload to others	Works with most

Table F.1: Front end testing table

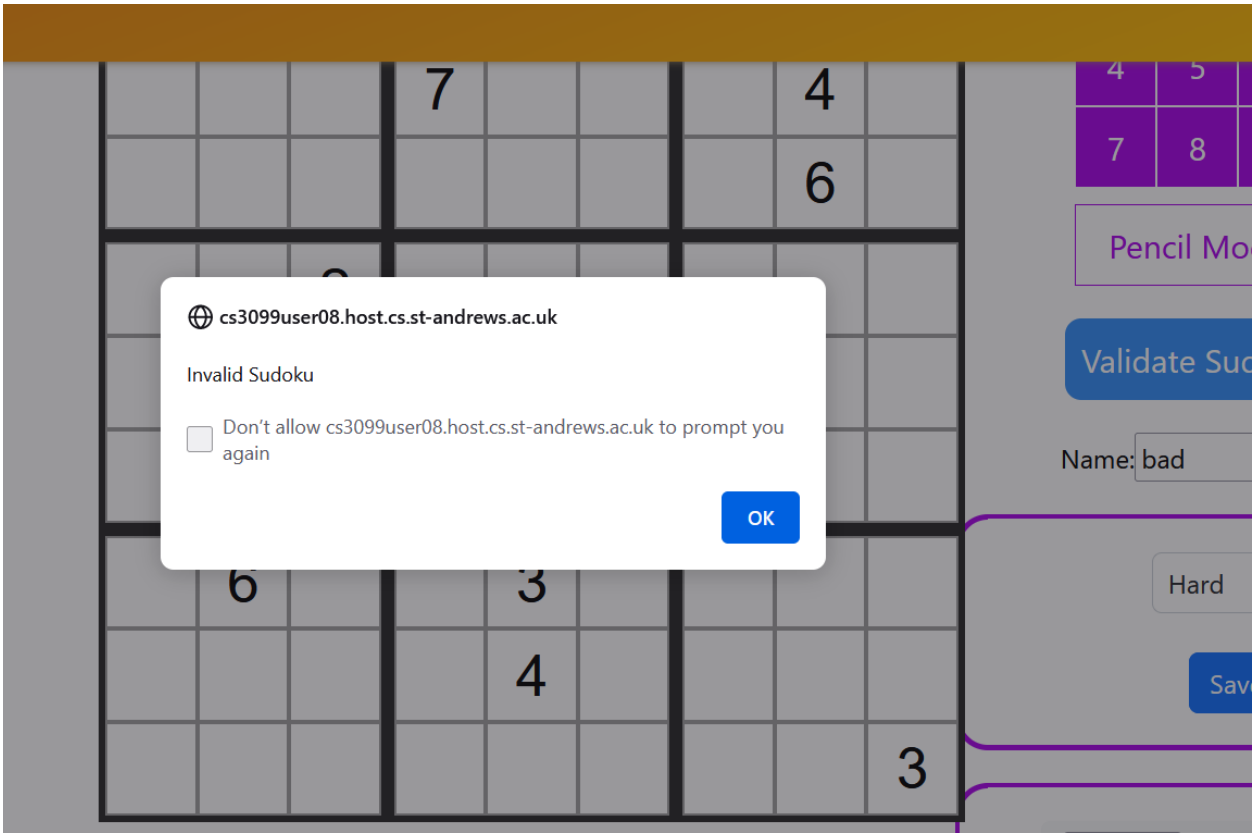


Figure F.1: Testing sudoku

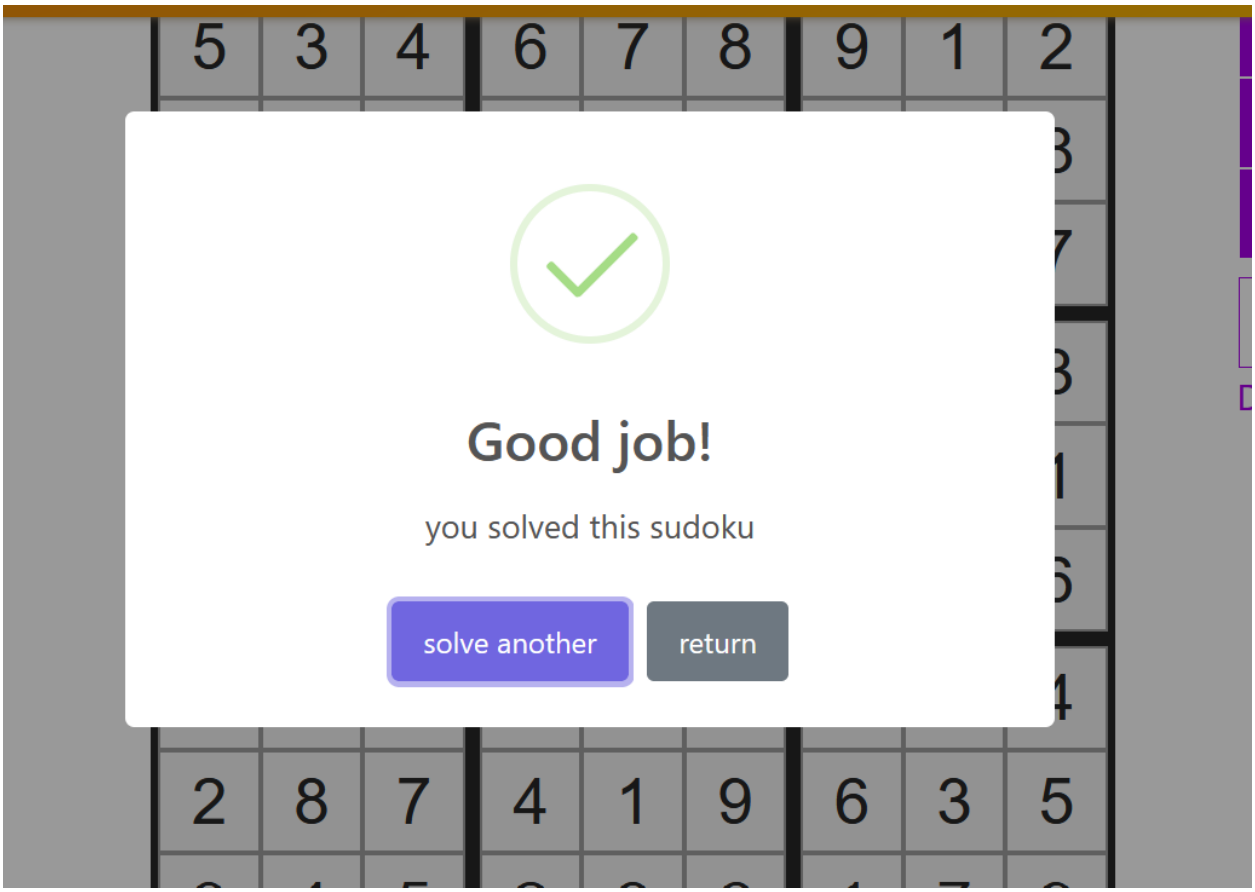


Figure F.2: Testing sudoku

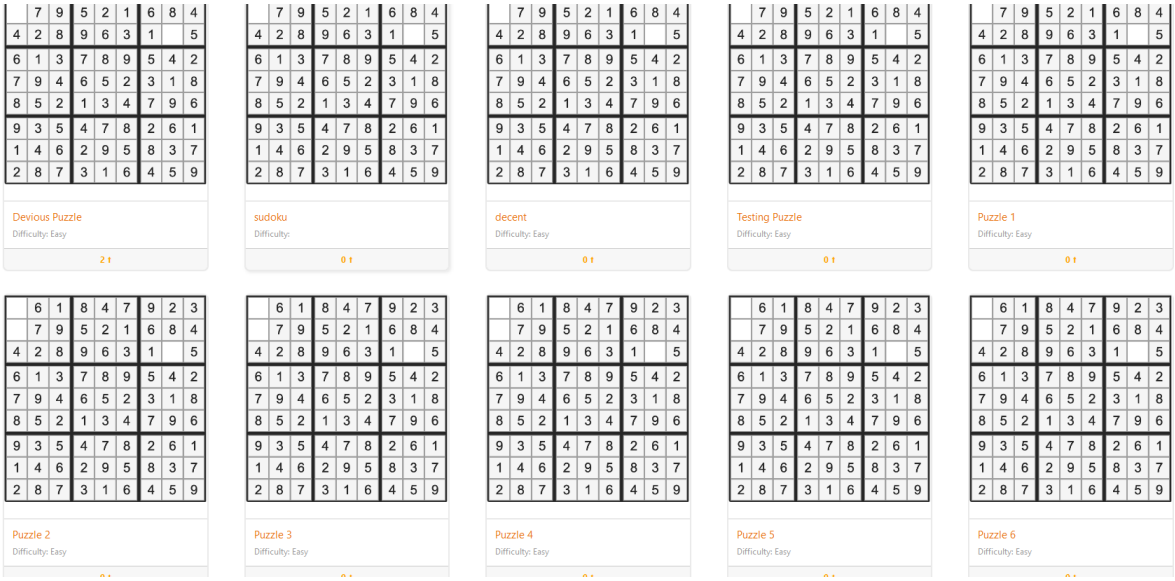


Figure F.3: Testing sudoku

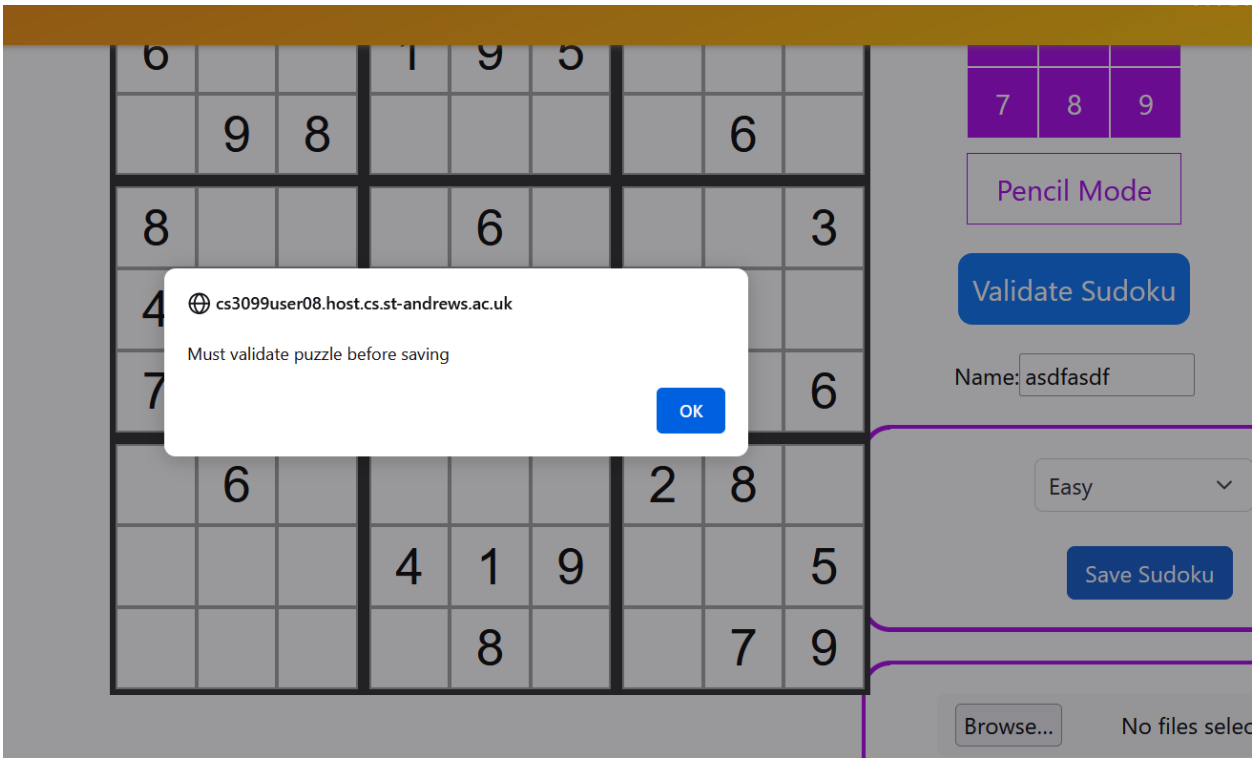


Figure F.4: Testing sudoku

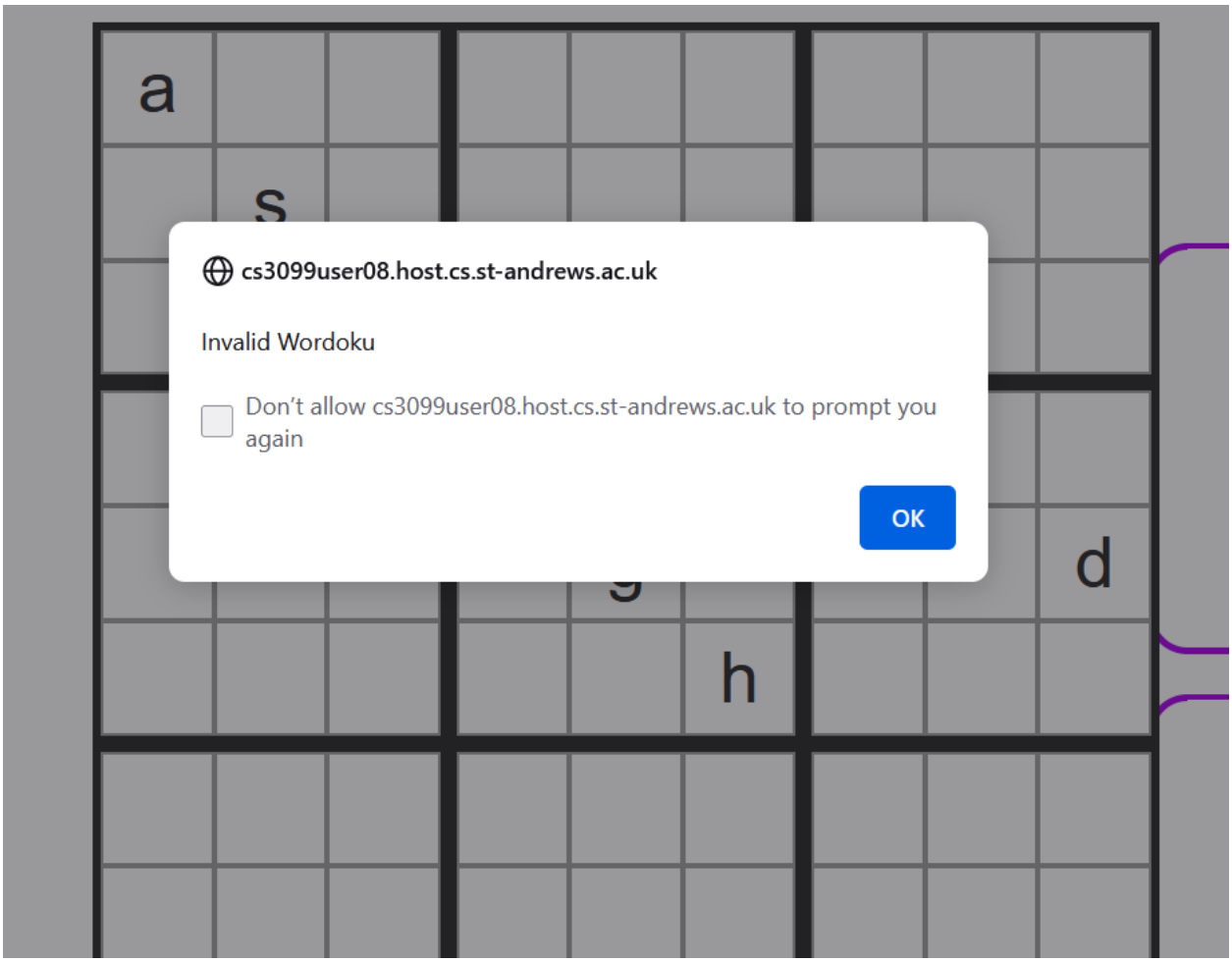


Figure F.5: Testing wordoku

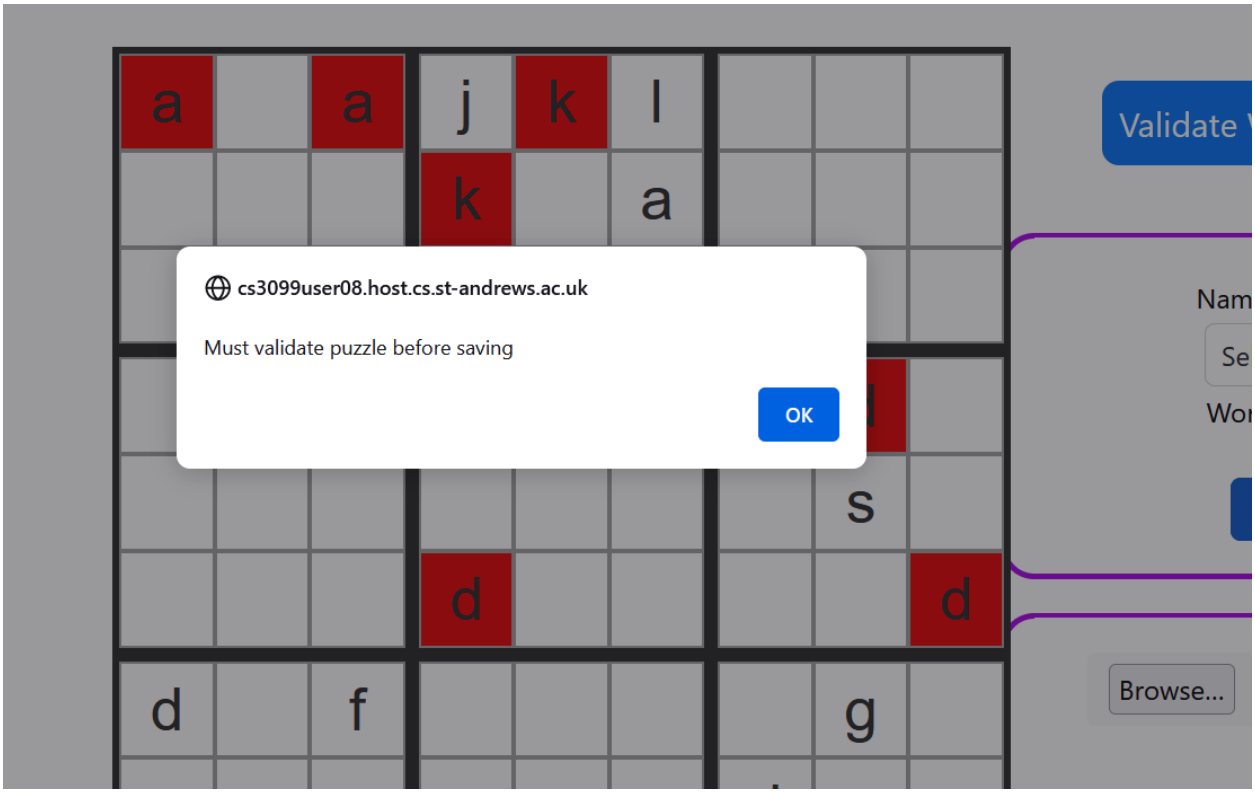


Figure F.6: Testing wordoku

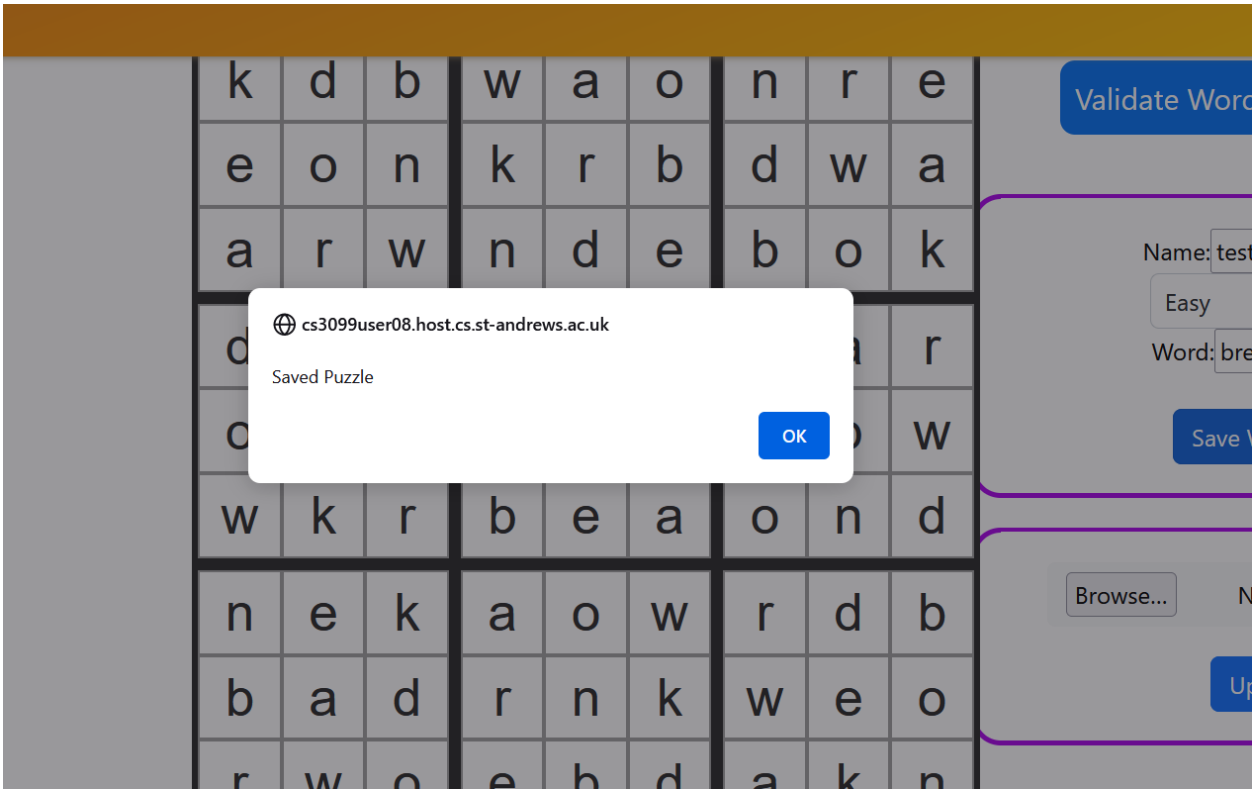


Figure F.7: Testing wordoku

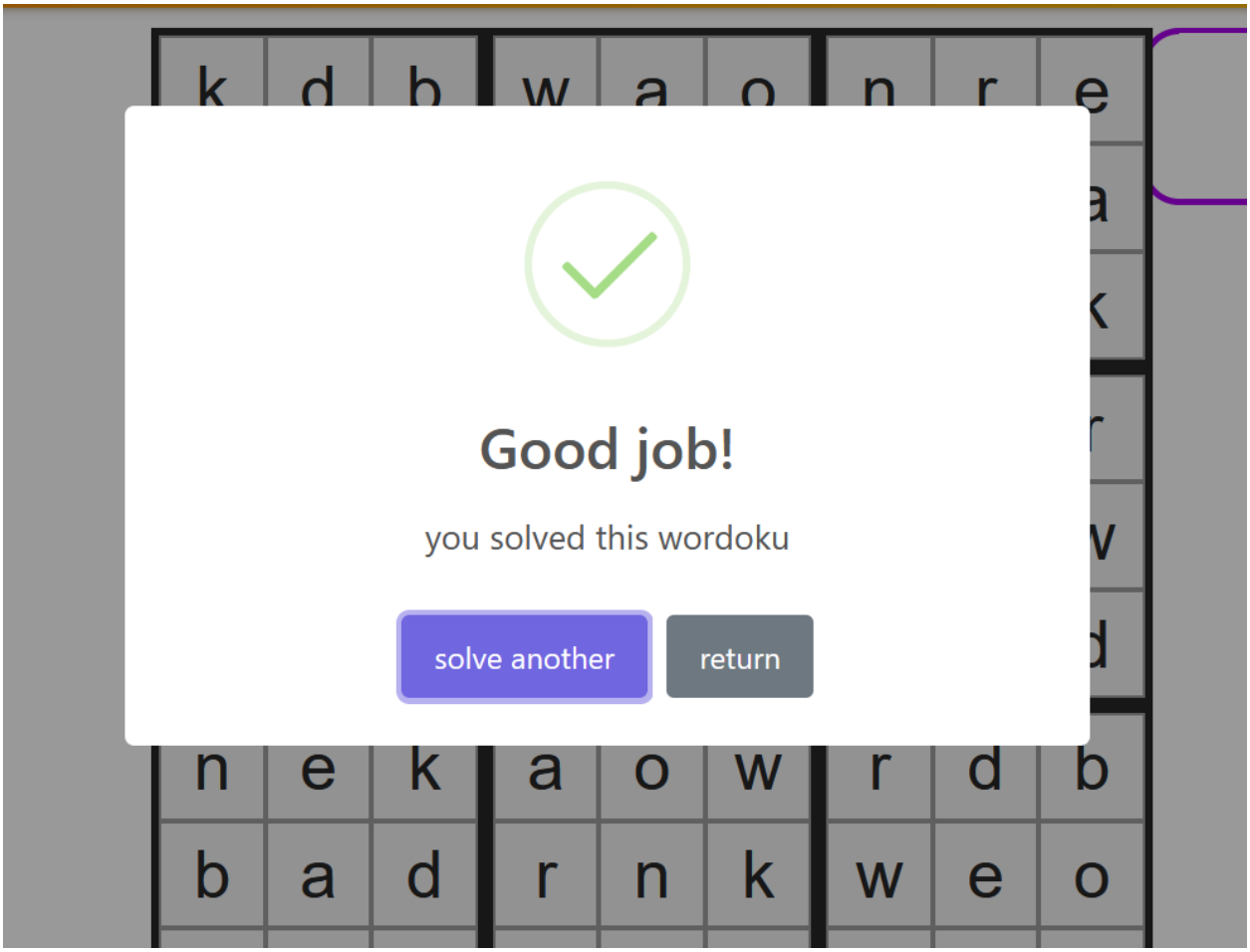


Figure F.8: Testing wordoku

G. Back-end testing

Functionality	**Test**	**Result**	**Passed**
User registration	Entering the following registration details: username: “usernametest2“, email: “emailtest2“ and password: “passwordtest2“.	The registration details being received correctly. Shown at figure G.1.	✓
Rejecting already registered users	Entering the same information in the registration page again.	A message saying ”User is registered”.	✓
User login	Re-entering the username and password from the registration page.	The correct login details being received in the API and a JWT token being created which using this data which. Shown at figure G.2.	✓
Rejecting invalid login details	Entering the same information in the registration page again.	a message saying ”User is registered”.	✓
Displaying previous comments	Entering incorrect login details.	A message saying ”Invalid credentials”.	✓
Adding a new comment	Typing in a new comment and pressing the add button.	The comment being displayed at the bottom of all the existing comments with the correct username and text. Shown at figure G.3.	✓
Editing a previous comment	Typing a new comment under the text-box of an existing one and clicking the edit button.	The text of the chosen comment being changed. Shown at figure G.5.	✓

Table G.1: Results of back-end function testing

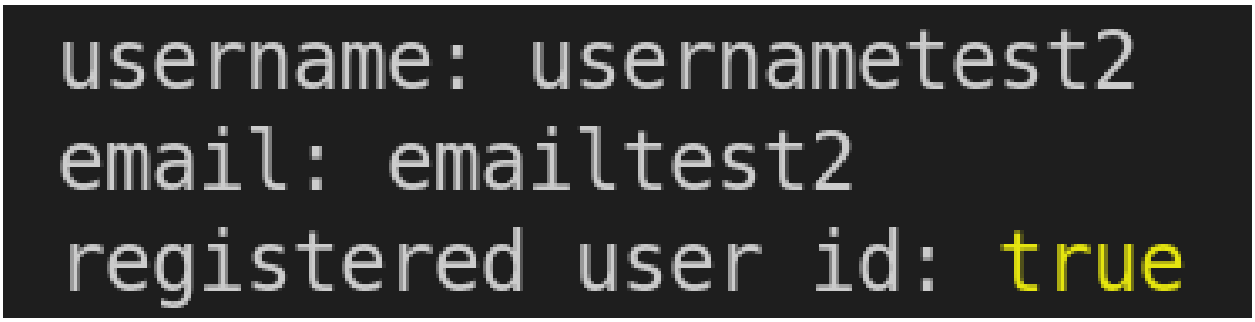


Figure G.1: Testing the registration page

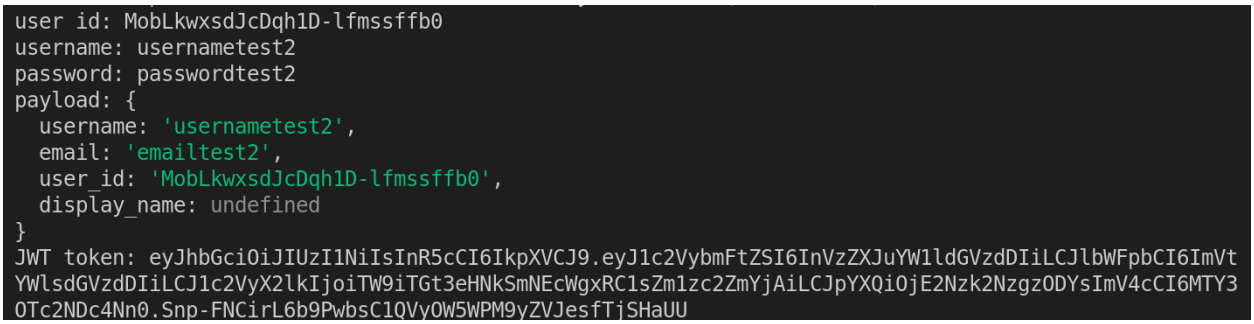


Figure G.2: Testing the registration page

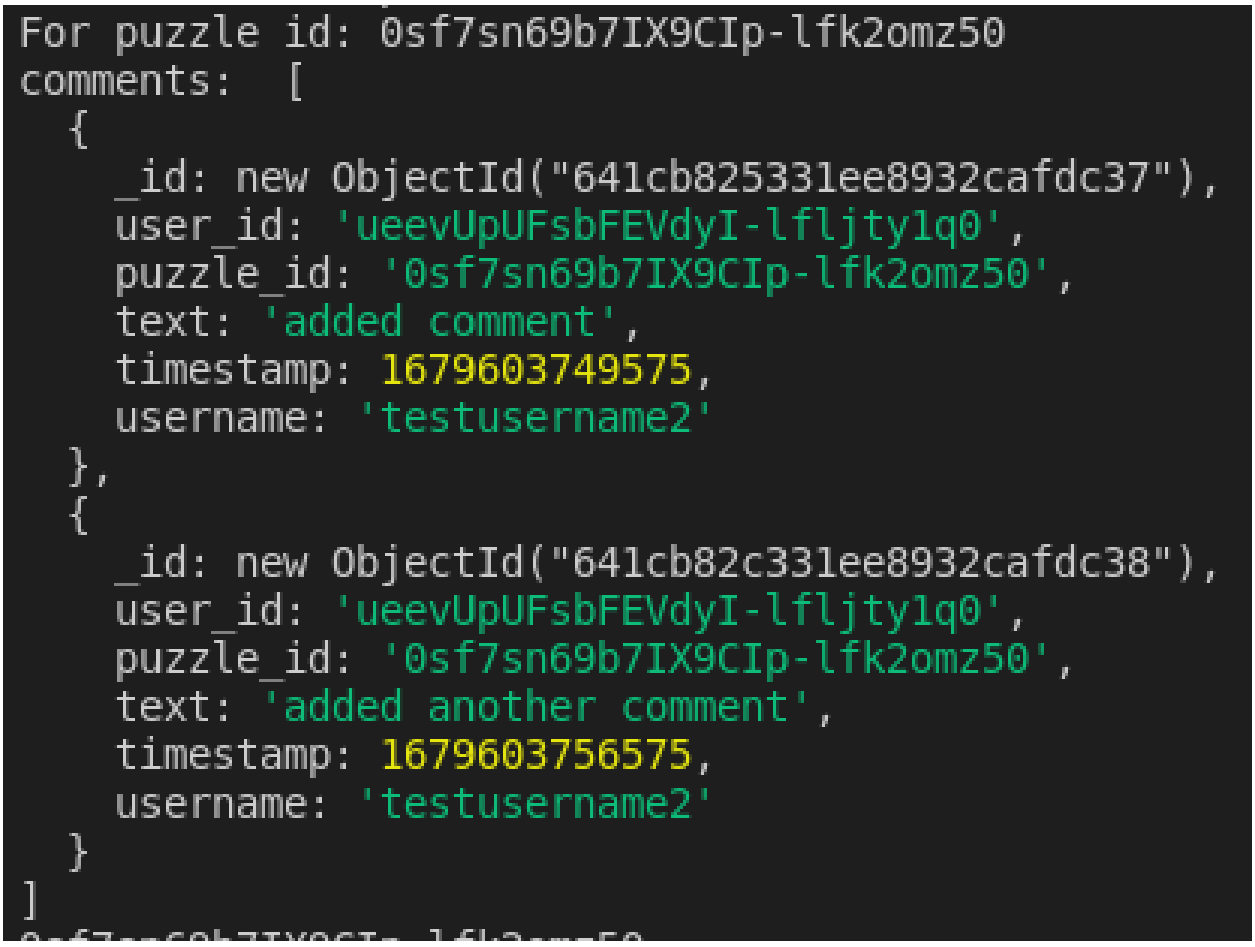


Figure G.3: Test for getting all the previous comments

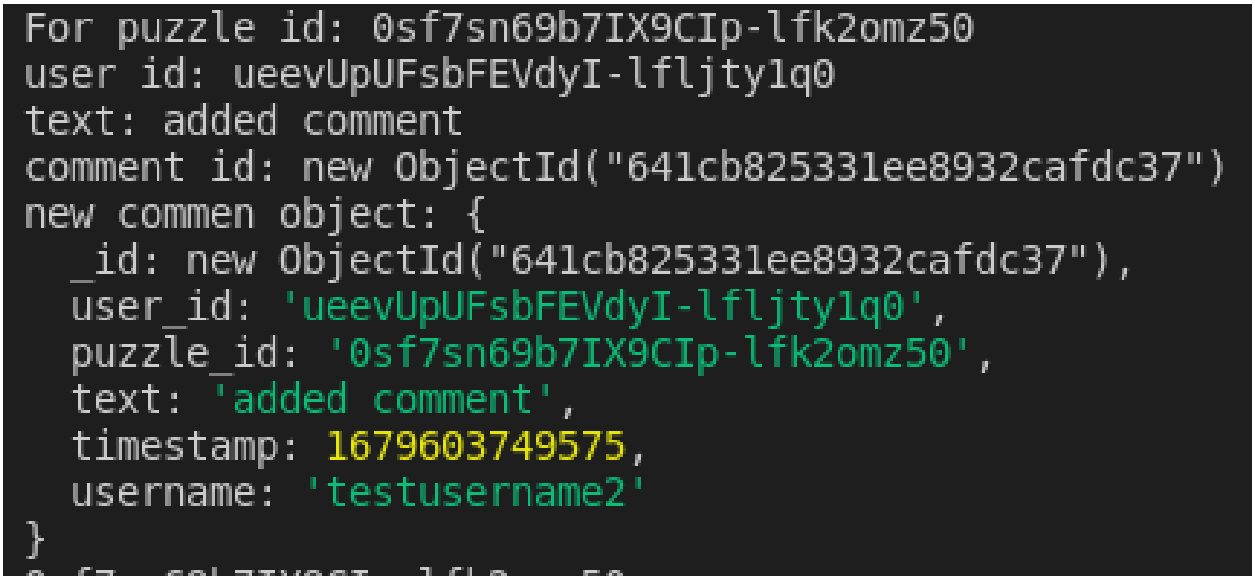


Figure G.4: Test for adding comments

```
For puzzle id: 0sf7sn69b7IX9CIp-lfk2omz50
comment id: 641cb82c331ee8932cafdc38
new text: changed again
edited comment object: {
  _id: new ObjectId("641cb82c331ee8932cafdc38"),
  user_id: 'ueevUpUFsbFEVdyI-lfljty1q0',
  puzzle_id: '0sf7sn69b7IX9CIp-lfk2omz50',
  text: 'changed',
  timestamp: 1679603756575
}
```

Figure G.5: Test for editing comments

H. Database Testing

```
 ============================== Coverage summary ==========================
=====
 Statements   : 6.34% ( 60/945 )
 Branches     : 1.86% ( 8/428 )
 Functions    : 8.26% ( 19/230 )
 Lines        : 6.71% ( 58/864 )
 ================================================================================
=====
```

Figure H.1: Output of Test Coverage

Summary

9 tests | 0 failures | 0 errors | 100% success rate | 915.00ms

Jobs

Job	Duration	Failed	Errors	Skipped	Passed	Total
test	915.00ms	0	0	0	9	9

Figure H.2: Output of Test Summary on GitLab

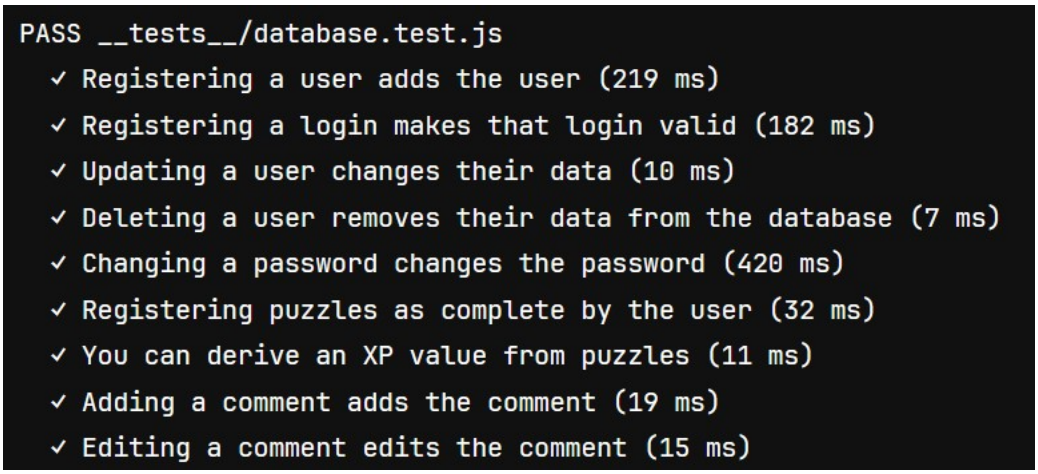


Figure H.3: List of unit tests run

I. Google lighthouse Results

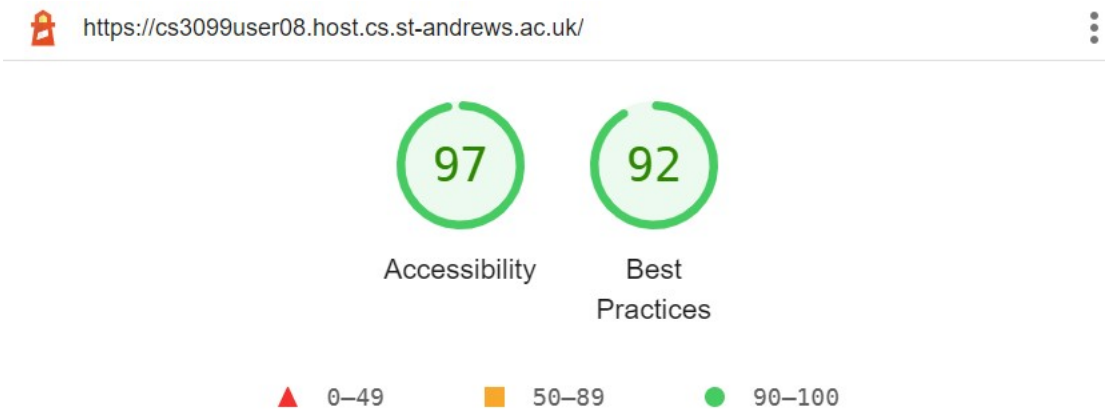


Figure I.1: Results of the UX audit using Google Lighthouse

J. Sitemap

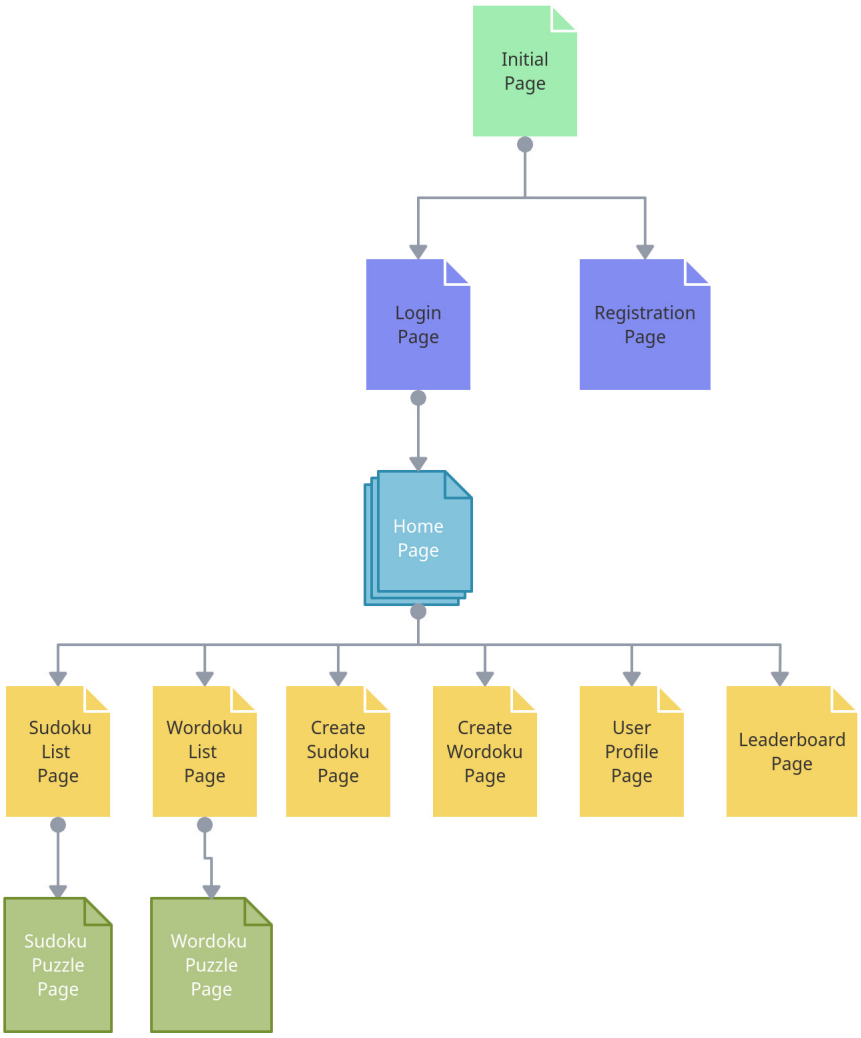


Figure J.1: Shows the structure of our website