CS3099 Deliverable 4 Report

Final Submission for the Fantastic Puzzles Fife Project Group 5

By 200022530, 200022418, 200006767, 200009744, 210003309

Word Count: 13766

Supervisor: Jason Jacques

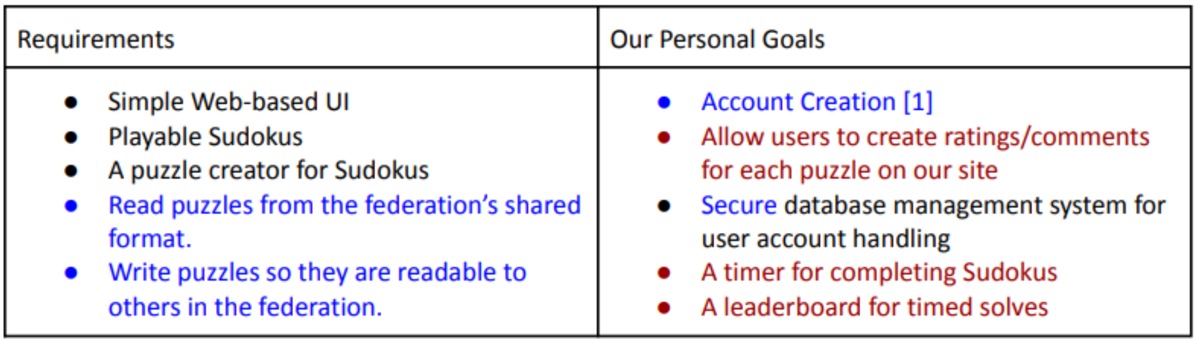
Table of Contents

Overview (200022530)

Our Junior Honours Project: Fantastic Puzzles Fife can be briefly defined as a federated puzzle website (primarily focused on Sudokus) allowing users to play, create, and share puzzles. Our website is compatible with other websites in our federation, The Aces Supergroup, allowing users from other sites in the federation to use our site and vice versa. A full description of our website's features is provided in our "Full Features List" below.

This report will provide a comprehensive list of all available features with relevant testing and a walkthrough of our design decisions. Additionally, we will discuss how we interacted with the supergroup and evaluate the techniques we used to keep our project on track and any limitations that come to mind with our implementation.

Our Previous Deliverables: Semester 1



Requirements	Our Personal Goals
● Simple Web-based UI ● Playable Sudokus ● A puzzle creator for Sudokus ● Read puzzles from the federation's shared format. ● Write puzzles so they are readable to others in the federation.	● Account Creation [1] ● Allow users to create ratings/comments for each puzzle on our site ● Secure database management system for user account handling ● A timer for completing Sudokus ● A leaderboard for timed solves

[Figure 1: Screenshot of features table from Deliverable 2's Report.]

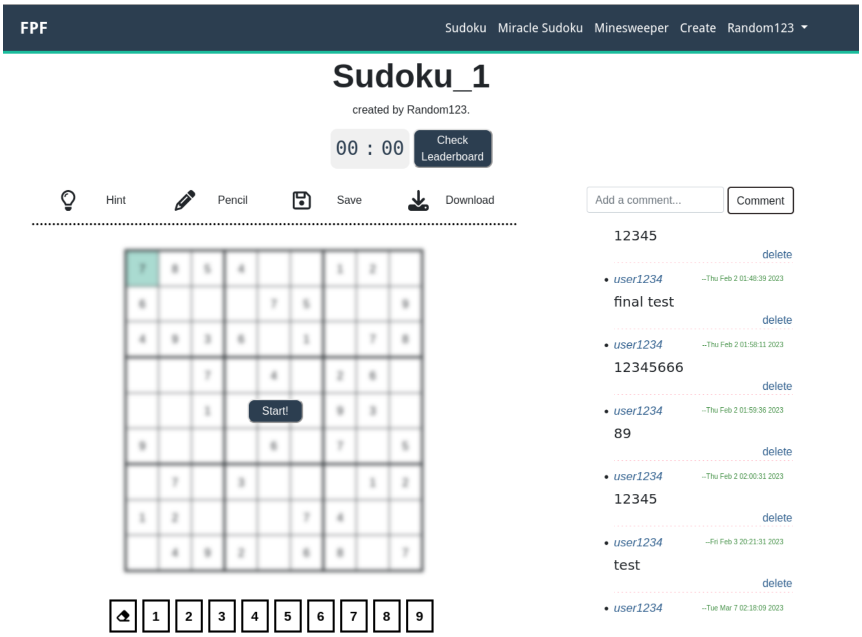
Key: Completed (Updated between Deliverable 1 and Deliverable 2), Incomplete

In our previous report, we highlighted features that we aimed to complete but had not fully implemented yet - noting that we would complete these first as priority in the following semester. However, upon reflection we decided to first go through our Supervisor's Semester 1 Feedback before adding these additional features.

To improve organisation of the project in the second semester, we started using Jira to track our product backlog and sprints - this will later be discussed in more detail later in this report.

Full Features List

Key: Additional Feature, Required by Specification

Location	Feature	(Further) Description
frontend/*	Simple, web-based user interface. Our website is well-displayed with easy navigation and conventional icons encouraging self-explanatory user interaction.  FPF Sudoku Miracle Sudoku Minesweeper Create Random123 Sudoku_1 created by Random123. 00 : 00 Check Leaderboard Hint Pencil Save Download Start! 1 2 3 4 5 6 7 8 9 Add a comment... Comment 12345 delete - *user1234* --Thu Feb 2 01:48:39 2023 final test delete - *user1234* --Thu Feb 2 01:58:11 2023 12345666 delete - *user1234* --Thu Feb 2 01:59:36 2023 89 delete - *user1234* --Thu Feb 2 02:00:31 2023 12345 delete - *user1234* --Fri Feb 3 20:21:31 2023 test delete - *user1234* --Tue Mar 7 02:18:09 2023 *[Figure 2: Screenshot of our playing Sudokus on our website]*	
NavBar.vue	Users can navigate to different pages using a navigation bar	A navigation bar sits on top of the website that provides users with quick and easy navigation around the site.
SudokuSelector.vue **SudokuPreview.vue**	Users can select which Sudoku they want to play	All our Sudokus can be accessed using a scroll wheel at the bottom of our displayed Sudokus. (These can also be scrolled through by holding shift and using the scroll wheel).
	Users get a blurred preview of a Sudoku before selecting it	Blurring the number of numbers in the grid provides some mystery of what kind of puzzle the user is navigating to and prevents them from solving them before starting the timer.
PlaySudoku.vue	Users can play Sudokus on our site	Users are able to play with both ***mouse and keyboard controls*** to improve the quality of our UI. *Note: For keyboard controls to work, a cell in the Sudoku must first be clicked.*
	Users can play miracle Sudoku	
	Pencil	Users can make pencil markings on Sudokus to aid their solve.
	Download	Users can download Sudokus to play on other sites in the federation.

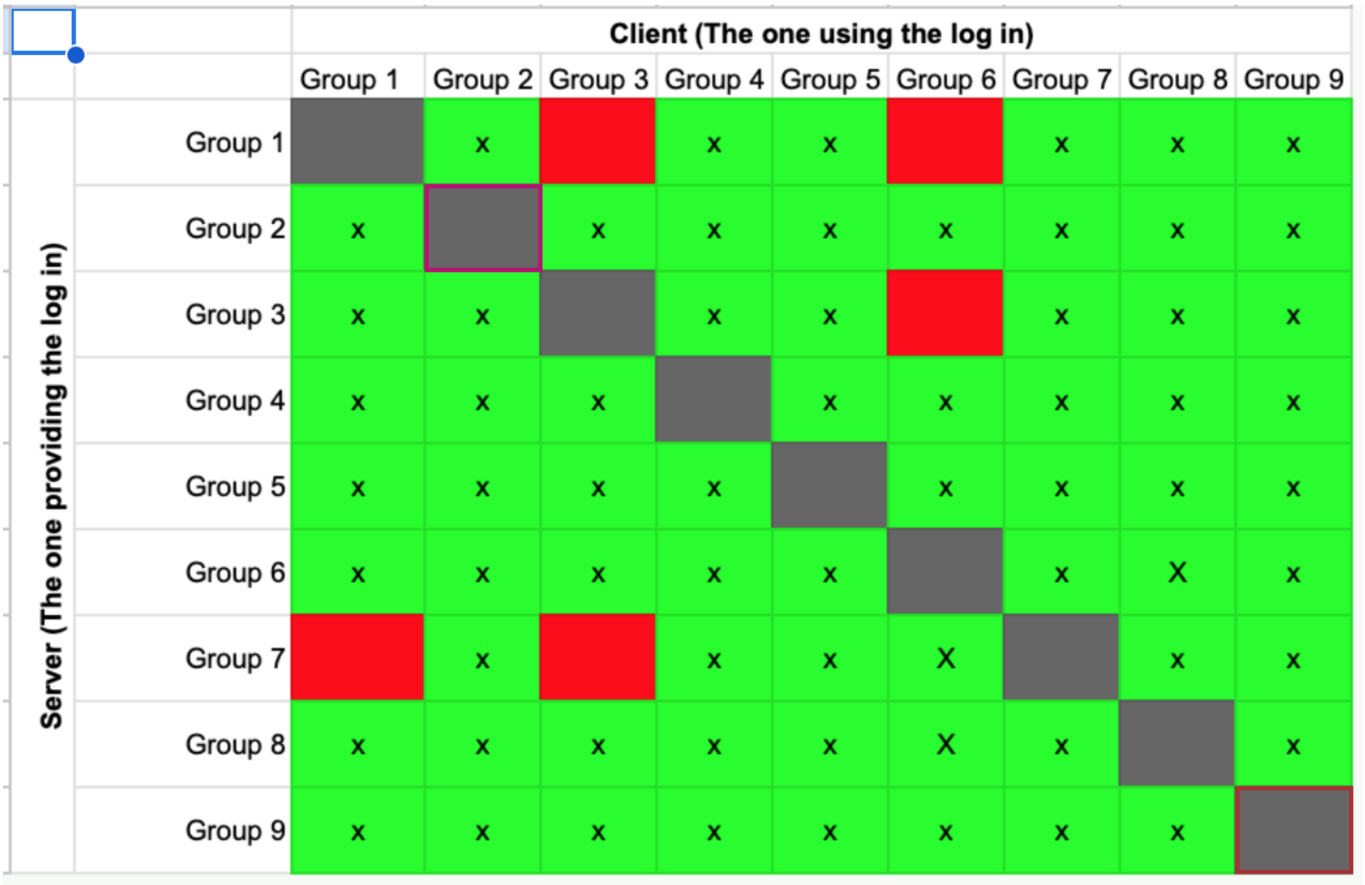
Location	Feature	(Further) Description
	Users can receive hints to aid solving a puzzle ()	Users can use only up to 3 hints, each use of a hint applies a 10, 15 or 20 second penalty to the timer. When choosing which hint cell to fill: ● It is the most likely cell that a human player would choose (the one with the fewest possible numbers, full explanation in the hint functionality section)
	Write comments	Users can comment on puzzles and have a discussion with other users about the specific puzzle
	Users can resume playing Sudokus from a saved state	This can be accessed by first going to their account page, playing from a saved state prevents users from timing solves and entering the leaderboard.
	Running Timer	When “Start” is pressed, the timer above the puzzle starts which is sent to the leaderboard upon submission.
Register.vue **Login.vue**	Users can register and login to the website.	New users can register to create an account on the website and log in to gain access to playing and creating Sudokus and other puzzles on the website.
	Users must meet specific password criteria	● At least one lowercase required ● At least one uppercase required ● At least one number required ● At least eight character required
ForgotPassword.vue	Forgot password	Allow users to reset their password using their email address. Whilst testing our website, we found it difficult to keep track of our passwords so we thought it would be convenient to add this feature.
FederationLogin.vue	Users from other websites in the federation can log in to gain access to our website	
	Users are given a unique username dependent on their group	
FederationRedirect.vue	Our users can login to other websites in the federation	
CommentList.vue	Users can view comments made by themselves and other users	Users can read other user’s comments and navigate to their account page.This can be found next to Sudokus whilst playing them.
AccountPage.vue	Users can view their and other users’ roles	Available roles include: ● Admin: ○ Delete all users’ comments. ○ Promote lower ranking users. ● Player: ○ Delete their own comments.

Location	Feature	(Further) Description
		● Creator: ○ Delete their own comments. ○ Cannot submit times to the leaderboard.
	Users can save puzzle states to their account and resume them	Puzzles can be saved above whilst playing them. Puzzles that are complete or haven't been started cannot be saved.
	Admins can promote other users to admin	This is only possible if the target user is not an admin. If it's possible a button will appear under the user's role and name.
	Users can read other user's bios	If the user hasn't saved a bio, displays "This user has not saved a bio."
	Users can write their own bios	Users can edit and save bios on their page.
PlayMinesweeper.vue ()	Users can play minesweeper with three difficulties and play their own customised minesweeper.	After selecting the difficulty level or creating a custom Minesweeper game, users will be redirected to the game page. This page features a Minesweeper board where users can begin playing the game. The page also displays the number of mines that are left to be discovered.
	Users can restart a game at any time.	Users will find a 'Restart Game' button which allows them to start a new game at any time.
MinesweeperSelector.vue ()	Users can select the difficulty of the minesweeper they want to play.	The Minesweeper selector page provides users with the option to choose the difficulty level of their game. Users can easily select from three different levels - Easy, Medium, and Hard - using dedicated buttons.
	User can customise their own minesweeper	For those who want more control over their game, there is also a 'Custom' button. Clicking on this button will reveal an input area where users can specify the number of rows, columns, and mines they want in their game.
CreateSudoku.vue	Users can create Sudoku puzzles on our site	
	Created puzzles are checked using a Sudoku solver to ensure they have one correct solution.	
	Only allows creators to create valid Sudokus	
Leaderboard.vue	Users can view a leaderboard showing the top 10 timed solves	By pressing the "Check Leaderboard" button on a Sudoku or Miracle Sudoku, users are able to view the current best 10 times for that puzzle (users can have multiple appearances in the leaderboard)

Location	Feature	(Further) Description
	Users can navigate to the accounts of the users who completed the top 10 solves of a puzzle	Clicking on the username for the respective row in the leaderboard takes the user to the account page of the selected user
	Creators of a puzzle cannot be on the leaderboard for it	Users (specifically creators) who created a puzzle are able to play it but are alerted when they solve the puzzle that it was not submitted

[Figure 3: Table summarising a list of all features for our website.]

Federation Compatibility



		Client (The one using the log in)								
		Group 1	Group 2	Group 3	Group 4	Group 5	Group 6	Group 7	Group 8	Group 9
Server (The one providing the log in)	Group 1	▨	x	🟥	x	x	🟥	x	x	x
	Group 2	x	▨	x	x	x	x	x	x	x
	Group 3	x	x	▨	x	x	🟥	x	x	x
	Group 4	x	x	x	▨	x	x	x	x	x
	Group 5	x	x	x	x	▨	x	x	x	x
	Group 6	x	x	x	x	x	▨	x	X	x
	Group 7	🟥	x	🟥	x	x	X	▨	x	x
	Group 8	x	x	x	x	x	X	x	▨	x
	Group 9	x	x	x	x	x	x	x	x	▨

[Figure 4: Aces Supergroup Compatibility Spreadsheet.]

Federation logins are not currently compatible with all groups, to summarise these compatibilities our federation created the table above (fig. 4). Our website is compatible with the following sites:

Logging In On Our Website	Logging In On Their Site
Group 1	Group 1
Group 2	Group 2
Group 3	Group 4
Group 4	Group 4
Group 6	Group 6
Group 7	Group 7
Group 8	Group 8
Group 9	Group 9

[Figure 5: Our Aces Supergroup Compatibility.]

Design

This section of our report will discuss our design decisions made both in the frontend and backend. Our frontend makes use of Vue to create and organise components and CSS to style our pages. We have a Flask Python backend that the frontend uses to communicate with the database as well as check the validity of created Sudokus. *(200022530)*

Frontend

Accounts

AccountPage.vue (200022530)

Player	Creator	Admin
● Viewing their account information	● Viewing their account information	● Viewing their account information
● Editing and saving their bio	● Editing and saving their bio	● Editing and saving their bio
● Resuming their saved sudoku	● Resuming their saved sudoku	● Resuming their saved sudoku
		● Promoting Player and Creators

[Figure 6: Table summarising account page functionality for different users.]

The account pages of our website provide the users access to the features shown in the above figure. Users do not automatically have a default bio after registering, this will be detected in the frontend and display an appropriate message. To ensure that users cannot edit each other's bios, the isCurrentUser() method uses the username of the current user that's stored in local storage to determine whether someone is viewing their own page.

Admins can promote only players and creators, the isPromotePossible() function uses the current user's role to determine whether they're an admin and then checks whether the user they're viewing is already an admin or not. All users that are not admins can be promoted to admin by other admins. As long as this remains the intended functionality, if more user roles were added, this method will not have to change. We decided to allow admins to promote other users so that as the site grows more more admins can be added to make sure the website keeps the desired environment by allowing more accounts to delete harmful comments.

Resuming saved sudokus makes use of the SudokuPreview component, passing it a prop with the saved position that is then ultimately passed to the PlaySudoku component. This prop will be used to populate the grid with the given saved state.

Register.vue (200022418)

This component displays a user registration form. The form includes fields for username, email, password, and confirm password, as well as a dropdown menu for selecting the desired user role. There are many validation checks to ensure that the user database only contains users with valid details such as a valid username, password, and email. If any of the validation checks fail, an error message is displayed to the user alerting them of what went wrong. If all of the validation checks pass, the user's registration information is sent to the backend where some more validation checks take place for duplicate usernames and emails. If those pass as well, the users information is stored in the database, this is confirmed to the frontend and the frontend redirects the user to the login page, where they can input their new details to login.

Login.vue (200022418)

This component displays a user login form. The login form uses Axios to send a POST request to the backend with the user details. The component methods include:

- userLogin
    - This handles basic validity checks for empty fields and form submission to send user details to the backend. Once the user details are checked in the backend, this function receives a message that will inform if the user is registered or not. If the user is registered, the user is redirected to the home page
- formInitialise
    - This resets the form
- doLogin
    - This handles the form submit event and calls the userLogin function to do the POST request and the formInititalise function to reset the form
- sendToGroup
    - This handles the partner site logins. This sends a POST request to retrieve the client ID for the selected group and redirects the user to the appropriate external OAuth service

The components uses the asset “down-arrow-svgrepo-com.svg” for the arrow symbol. This background image was taken from [13].

ForgotPassword.vue (200009744)

To address the issue of forgotten passwords, we have decided to add a "forgot password" feature to our website. Initially, during registration, users were only required to provide a username and password. However, this approach is less secure as anyone who knows a user's username could potentially reset their password. To improve security, we have made email a required field during registration. The email address provided will be kept secret and will serve as a memorable question. If a user forgets their password, they can reset it by providing their username and the email address they used during registration. Currently, we do not have the capability to send emails to the user's email address for authentication. As such, any email address that satisfies the email format can be used to register an account on our site.

Puzzles

Sudoku

PlaySudoku.vue (200022418)

This component contains all the logic that is concerned with playing a traditional or miracle sudoku. It contains the following features:

- Displays the sudoku
- Displays the name and the creator of the puzzle
- Displays a timer and provides a button to show the leaderboard for the current sudoku
- Provides buttons to for four different features: hint, pencil, save, and download
- Displays the comments section
- Distinguishes between selected cells, editable cells, and highlights incorrect entries
- When the player finishes the game, the component displays a custom alert box, which shows the solve time and allows players to rate the puzzle's difficulty

Additionally, this component manages a considerable number of Axios requests, which facilitate communication with the backend. The Axios requests serve various purposes such as:

- Loading the sudoku puzzle from the database in the backend
- Submitting a rating for the sudoku
- Checking if a sudoku has been correctly solved
- Checking for incorrect entries into the sudoku
- Submitting timer result to leaderboard
- Retrieving a hint from the backend
- Downloading the sudoku
- Saving the sudoku

CreateSudoku.vue (200022530)

At the top of the “Create a Sudoku” page, there is a button provided to allow users to switch between the type of Sudoku they are creating - providing a choice between Traditional and Miracle Sudoku. To communicate this clearly to the user a note is provided in the top-left of the page, since it is not clear from the icon alone what it is meant to do. Additionally, our note also explains what the purpose of our upload button is for: uploading compatible Sudoku file formats that match what has been outlined by the Aces Supergroup.[1]

When creating Sudokus, the page is initialised with a correctly sized 2D-Array (9 by 9) that is editable using both mouse and keyboard controls in the same way as was mentioned previously in PlaySudoku.vue. Some basic validity checks are fulfilled in the frontend of the website, before sending the grid to the backend’s Sudoku solver, and these are as follows:

- The Sudoku has been named
    - To prevent unnamed Sudokus, since our website uses the name of a Sudoku to retrieve it from the database (therefore they must be unique).
- The Sudoku name contains only letters, numbers, and underscores.
    - To prevent formatting issues or SQL injections

[1] Aces Supergroup. *aces-documentation*.

- The Sudoku name is between 5 and 40 characters long.
    - Ensures Sudokus are adequately named and that large Sudoku names don't cause future formatting problems.
- At least 17 cells are filled with a number as a solvable Sudoku with less than 17 filled cells does not exist.[2]

Pencil Annotations (200006767)

The PlaySudoku.vue component supports pencil annotations. A user can press the pencil button above the puzzle and each empty cell becomes a small 3x3 grid in which the user can make pencil annotations. This feature is particularly useful when solving Miracle Sudokus because it makes it easier for solvers to keep track of the possible numbers surrounding a completed cell. In particular, the people who invented the game always use pen and paper to keep track of possible values of empty cells, so our pencil annotation improves the user experience for Miracle Sudokus. The feature is available in the traditional version of the game too. The feature is implemented by splitting all empty cells into 3x3 grids and changing the colour of the pencil button to green to highlight the fact that the pencil is on (and the pencil button becomes blue again when the pencil is off). By pressing the erase button while in pencil mode, the user erases all the penciled numbers in the cell. If the user wants to remove a pencil annotation (exactly one number), they can just type the number using either the keyboard or the provided digit buttons. Pressing a digit button once in the pencil mode adds the annotation to the cell, and pressing it a second time would remove that particular digit annotation, leaving all the other digit annotations in the cell unchanged.

CommentList.vue (200009744)

The commentList.vue is a component of the play sudoku page, which displays all the comments related to the currently playing sudoku in an HTML list. Each comment includes the username of the sender, the time of creation, and the content of the comment. Users can click on the username to redirect to the sender's profile page to learn more about them.
Additionally, there is an input area where users can leave new comments. After submitting the comment, it will appear in the comment list without the need to refresh the page.
To retrieve all the comments, an Axios POST request is sent to our endpoint with the name of the sudoku puzzle as the parameter. The server will respond with all comments related to that puzzle in JSON format.

Addition of Miracle Sudoku (200006767)

The Miracle Sudoku reuses the same components as the Traditional Sudoku. The only distinction is that there is a query parameter in the url set to miracle=True. This parameter is used in all Axios post requests to the backend in the PlaySudoku component. In the CreateSudoku component, there is a button that can be pressed that toggles the variable showMiracleSudoku which determines the type of sudoku route used for backend posts. There is a component MiracleSudokuSelector that passes miracle: true as a prop to the SudokuSelector component. This component is used for displaying and retrieving the levels of the miracle sudoku from the database. Given the fact that the PlaySudoku, CreateSudoku, and the SudokuSelector are reusable components, the website could support multiple sudoku variants with very small changes to the frontend. For example, the current query parameter could be changed to type=" miracle/traditional/killer" and then just pass the type to all Axios post requests to the backend.

[2] *Minimum Sudoku.* Archived from Csse.uwa.edu.au by web.archive.org.

MineSweeper (200009744)

MineSweeperSelector.vue

Within the MinesweeperSelector.vue file, the page displays four buttons that represent the different difficulty levels of the game: Easy, Medium, Hard, and Custom. The Custom feature includes a straightforward checker that verifies the user's input for the row, column, and mine numbers. Additionally, to prevent any further bugs when generating the board, we limit the maximum number of cells, and the number of mines cannot exceed the total number of cells.

PlayMineSweeper.vue

Once the board is received from minesweeperGenerator.py, a minesweeper board is generated where each cell has various attributes:

- isMine: a boolean to indicate whether the cell is a mine.
- isShown: boolean to indicate whether the cell has been clicked.
- isFlagged: boolean to indicate whether the cell has been flagged.
- adjacentMines: number that indicates the number of mines surrounding it.
- ID (row-colmun): indicates the position on the board.
- diisplay: shows the number of mines arround it on the board. (The default value is an empty string (""), and the value will change to the number of adjacent mines (adjacentMines) once the cell has been revealed (isShown is true)).

To reveal a cell, the showCell(row, column) function is executed with the cell's ID as an argument. Once a cell is revealed, isShown becomes true, and it cannot be clicked again. If the clicked cell has no mines surrounding it, the showCell function will check all the adjacent cells. If an adjacent cell is not a mine, it will be revealed, and the showCell function will be executed again with the ID of the adjacent cell.
After a cell is revealed, the checkWin function is called to determine if the game has reached a winning state. Minesweeper has two winning stages: either all non-mine cells are revealed, or all mines are flagged (by right-clicking on the cell). The checkWin function goes through all the cells on the board and checks if either of these winning stages has been reached.
If the game is still ongoing, the player can continue to click on the cells, and the showCell and checkWin functions will be executed after each click. However, once the game has been won or lost, it will be over, and all cells will be disabled. At this point, all the mines on the board will be revealed.

Leaderboard (210003309) - Leaderboard.vue

Embedded within a regular Sudoku or Miracle Sudoku puzzle's page is a "Check Leaderboard" to the side of the timer which displays an overlay on the page showing as a table the top ten best times for that puzzle with the current user's best time being highlighted in green since the leaderboard allows for multiple submissions from the same user. This is because we wanted to follow our original goal for the site as a speedrunning competitive platform and so having users improve their scores and be able to see this on the leaderboards is something we felt was crucial to this.

Leaderboard.vue is the frontend component used to create the table within the overlay as well as populate it. It does this by getting the leaderboards' relevant data upon creation (*getLeaderboard()*), it sends the current puzzle's ID as a query parameter to the '/leaderboard' route which retrieves all of that puzzle leaderboard's player names and their associated times and returns it to the component as a dictionary 'scores'. The dictionary is split into entries which are then sorted in ascending order and finally the ranking of each row is set equal to the index in this sorted array (with 1 being added in the HTML as the array is zero-indexed) with the user's best time index also being stored (the value used to check which row in the table to highlight). Times are displayed after converting the stored values for time in seconds to MM:SS format, this is done by the *formatTime()* method. The usernames also have an onclick event attached so that they take the user to that user's profile and an onhover event to highlight the text.

Loadscreens (200022530)

We created a LoadScreen component which takes a message as a prop to allow us to easily create custom load screens for our pages. These can be seen when retrieving data from the database or when submitting a Sudoku after creating it.

Styling (200022530)

To style our pages we used CSS, all of the CSS we used can be found in frontend/assets/styles, most pages have their own independent CSS file, but we decided to use the same CSS file for pages and components that display Sudokus to make the aesthetics of our website consistent. Since some Sudoku grids on our website have different functions, for example: uneditable ones for previewing a Sudoku, completely editable ones for creating Sudokus, and grids for playing Sudokus, we created different classes for editable cells.

Backend

Endpoints

The majority of our endpoints are for providing an interface between the frontend and our database, allowing for the retrieval and submission of data. Some of the more complex aspects of our backend will be discussed later in this report where there is more processing required than the retrieval and submission of data. In this section are some important notes regarding endpoints, their security, and any interesting design decisions. (200022530)

accounts.py (200022530)

Responsible for all endpoints associated with management of pre-existing accounts, this includes getting user data, roles, saved Sudoku positions, and promoting users. This is primarily used by our Accounts page and other components to retrieve the role of a user. However, there are currently no security features implemented when promoting users and it is possible for anyone to send a promote POST to the backend of our website and promote anyone as long as they know their username.

> *The endpoint /registeradmin was created to allow us to populate the database with our first admin.*

comment.py and commentList.py (200009744)

Our comment endpoints are focused on handling requests related to comments. These endpoints include the ability to retrieve all comments, retrieve comments for a specific puzzle, and store new comments in the database.

When a client sends a request to retrieve all comments, the endpoint will retrieve all comments stored in the database and return them to the client in JSON format. If a client sends a request to retrieve comments for a specific puzzle with the argument puzzleName, the endpoint will query the database for comments associated with the given puzzleName and return them.

For the client sending a request to store a new comment in the database, the endpoint will accept the request body in a JSON format, parse it, and store the comment in the database. The comment will include the puzzle ID, username, and the comment itself. Once the comment is stored, the endpoint will return a success message to the client to confirm that the comment has been successfully added to the database.

puzzles.py (200022530)

When getting puzzles to display them, the GET requests in this file purposefully don’t return the solution. Even though the solution isn’t directly displayed in the frontend, with web developer tools the user may have been able to view the solution. Not returning the solution prevents users from attempting to cheat in this way.

We also have /backend/api/puzzles that we used during the development of the website to make testing whilst coding easier - allowing us to view all the data associated with puzzles stored on our website.

Login/Register (200009744)

To handle user registration, the API endpoint will accept the request body in a JSON format. The endpoint will first check if the username is already registered in the database. If the username is new, the endpoint will hash the password using the passlib.hash API and store the user details in the database. The user details will include the hashed password, username, email address, and selected user role. Once the user details are stored, the endpoint will return a success message to the client to confirm that the user has been successfully registered.

For user login, the API endpoint will accept the request body in a JSON format. The endpoint will hash the user input password using the passlib.hash API and check if the username and password match what is stored in the database. If the result matches, the endpoint will return a success message to the client to confirm that the user has been successfully logged in. Additionally, the username will be stored in local storage to help the browser remember the login status for future requests.

Leaderboard (210003309) - leaderboard.py

To populate the leaderboard table within a Sudoku/Miracle Sudoku level, a GET request is sent to the ‘/leaderboard’ route the required puzzle’s ID being sent as part of the query parameters. This ID is used to search the Leaderboard Collection within the database and all matching results are stored in a Cursor object ‘leaderboardData’; this object is iterated through and each dictionary variable within the object has its ‘username’ and ‘time’ keys stored in the new dictionary ‘scores’ which is finally returned and can be filtered in the frontend to populate the table with only the top 10 times.

When a user finishes a puzzle, a POST request is immediately sent to the ‘/leaderboard/submit’ route with the query parameters of: the puzzle’s ID, this is so that the relevant puzzle can have its leaderboard in the database added to; the time it took the player, this is stored and retrieved when populating the database as well as the user’s name, so their name can appear alongside their result.

The website allows puzzle creation through the Create tab in the navigation bar. The user will create a sudoku using the sudoku table and they will give it a name. If the sudoku has less than 17 clues, then an error from the frontend will tell them that they must have at least 17 clues. If it has more than 17 clues, it is submitted to the backend and the backend will test if the name is already in the database. All sudokus must have different names, so an error will appear in the front-end if the name is wrong. On the backend, the create_sudoku route will validate the created sudoku and if valid, it will add it to the database with the given name, and on the frontend, the user will be redirected to the level selector page.

In order to validate a created sudoku board, on the backend we have a python class called SudokuSolver (in the sudoku_solver.py file). This class is designed for solving sudokus, checking if solutions and partial solutions are correct, and generating hints for sudokus.

For solving a sudoku board, we are using an optimized backtracking algorithm that uses binary bit operators. Each row, column, and 3x3 box will be represented as an integer with 10 bits. Each digit from 1 to 9 is represented by the bitmask 1<<digit. We are doing this because checking if a digit is in a row/column/box can be checked by simply computing the value of (1<<digit) & (row/column/box 10-bit value). If the value is 0 it means the digit is not part of the row/column/box and this check was made extremely fast due to the bit operators used. Similarly, a number is added to a row by computing the bitwise OR between the bitmask of the number (1<<number) and the corresponding row, column, and box that the number is a part of.

The initializeBitmasks() method will take the initial board and create the bitmasks for the rows, columns, and 3x3 boxes using the logic described above. If there are two equal numbers on the same row/column/diagonal, then the method will exit and give the user an error such as "On column 1 there is more than one 9". No attempt to solve this board with the backtracking algorithm will be made if the board is invalid at this stage. If the board is valid, then we can apply the backtracking algorithm from the recursiveSolve() method. If this method detects 0 or more than 1 solution, the user will receive an error stating that the puzzle has no solution or multiple solutions. If the puzzle is valid then it is added to the database and the solution is also stored with the puzzle.

The recursiveSolve() method performs the following backtracking algorithm: Take an empty cell and check what numbers can be put in that cell by checking the corresponding row, column, and 3x3 box of the cell (these checks are extremely fast due to bitwise operators). For each possible value, add it to the grid, and move to the next cell. The next cell is the next cell to the right, or if the cell is the last one on a row, the next cell is defined as the first cell on the next row. The recursion continues until in one cell there is no possible number, or when we reach the end of the grid. If we reached the end of the 9x9 grid, that means that we visited all cells successfully and we have a solution. We add the solution and increase the counter for the number of solutions found. If we found 2 solutions, then the puzzle is invalid and the search will stop without exploring further possibilities. After each possible value is added in a cell, and performing the recursion with that value, we backtrack by removing that value from the cell and trying a new one next. Removing a value from a cell is easy because it involves using the XOR bitwise operators on the column, row, and 3x3 box of the cell and the bitmask of the value that will be removed. This algorithm was optimised as much as possible, but the runtime can still be very high (around 5-10 minutes) for some puzzles with 17-18 clues.

Some notes on the implementation: adding a value to a row is: row_value | (1<<value) and removing a value from the row is row_value ^ (1<<value). Checking if a number is part of a row is row_value&(1<<value) compared to 0. In the previous implementation, each such operation required

checking all the numbers on that particular row for checking if a number is already on a row. The bitwise operations mentioned above greatly increase the speed of the program and reduce the number of operations executed.

Miracle Sudoku Logic - 200006767

Our group decided to implement Miracle Sudoku as the second game on our website. Miracle Sudoku is a Sudoku variant that has 3 additional rules to the standard Sudoku ones: there can be no two adjacent cells (cell sharing on edge are called adjacent) that contain consecutive numbers, no two cells separated by a chess knight move can have the same value, and no 2 cells that are separated by a chess king move can have the same number in them.

Our website allows both creating and playing a Miracle Sudoku. The Sudoku and miracle Sudoku share the same functionality for getting a hint, allowing pencil annotations, level selection, detecting a mistake on the board, and detecting a fully completed correct board.

The main difference in the implementation of the variant compared to the traditional Sudoku implementation is the way we solve and create a miracle Sudoku. There are only 72 Miracle Sudoku valid complete 9x9 boards and we decided to exploit this feature.
In the miracle_Sudoku_solver.py file, we have a class called MiracleSudoku that extends the Sudoku solver class. We have 3 methods (check_adjacent_consecutive_move, check_king_move, and check_knight_move) that are given a board, the x and y coordinates of a cell, and a possible value for that cell and return true if the value can be placed there or false if the value breaks the rule (for example there is a knight move from this cell that has the values passed as input already) based on the completed numbers in the grid.

The method solve_miracle_Sudoku performs the same backtracking as the recursive solve function from the Sudoku solver, but it also checks for the 3 additional rules using the methods described above. The purpose of this method is to return a list self.miracle_Sudokus that contains all the completed miracle Sudokus. We will start with an empty board (full of zeroes) and call this method on it. After a number of minutes (less than 10) the method will return a list of 72 complete miracle Sudokus. Now that we can generate them, we only need to do this call once and then we can store them in the database. In the main.py file at the end of the file, there is a commented section that has a TODO comment mentioning that the piece of code commented must be run once when first deploying the app to the school server. The commented lines of code populate the miracleSudokuCollection with the solutions to all MiracleSudokus.

When creating a Miracle Sudoku, the user can use the create functionality on the front end by pressing the 2 arrows next to “Create a Sudoku”. When this button is pressed and a user inputs a miracle Sudoku and presses the save button, a request is sent to the backend to the create_Sudoku/miracle route. We first check that the name of the Sudoku is valid, and if it is we call the isSolvable() method from the MiracleSudokuSolver class. This method iterates over all 72 Sudokus in the database and checks if the grid that was sent matches any of the solutions (By matching we mean that all elements in the submitted grid are part of a board from the database). If there is no solution, or there are multiple solutions a descriptive error message is given to the user, otherwise, the Sudoku and its solution are stored in the database. Since there are 72 Sudoku boards and at most 81 checks for each, this method is extremely fast (less than 1s) and almost instant from the perspective of the user. This is a very important difference compared to the traditional Sudoku solver which can take up to 10 minutes to validate a board and produce a solution. Another important thing to notice is that the error messages are only “this board has no solution” or “this board has multiple solutions”, and they are not more descriptive because a Miracle

Sudoku would ideally only have 2-5 clues, and given the small space of solution a creator adding more clues would have to know the complete valid board beforehand anyways.

Hint Functionality - 20006767

A user is allowed to get up to three hints for a Sudoku puzzle. On the frontend, this is achieved by pressing the hint button above the Sudoku board. If the user uses 3 hints, they will not be able to ask for more and a message will be displayed. If by getting a hint the user wins then a winning message box appears immediately. This check is important because otherwise the user would be left without any possible moves left and the game will not end.

In order to receive a hint, a post request is sent to the /Sudoku/hint route (in the Sudoku.py file)with the Sudoku's name, a boolean that determines whether a puzzle and the partially completed 9x9 Sudoku grid as part of the request body. The function getHint() of the SudokuSolver class is used to return a board with one extra cell completed.

An important design decision was choosing which cell to complete given a partially completed board. We decided that we wanted to choose a cell that would be the most likely next number that a human player would choose by playing Sudoku with pen and paper. For example, if one of the cells of the Sudoku can be determined uniquely by looking at the already completed numbers (i.e. you look in a cell and notice that the only possible number is 1 for example), then that is the cell that will be revealed (if there are multiple cells, then only one will be revealed). Sometimes on hard Sudokus, this is however not always possible to determine just by looking at the numbers, so in that case, our algorithm will check the number of possible numbers in each cell (for each cell we check the numbers on the row, column, and 3x3 box and see what numbers could be there). We will pick one of the cells with the least number of possible digits, and reveal the correct number that is part of the solution from the database.

The implementation of the hint algorithm is the following: Initialise the bitmasks for all rows, columns, and 3x3 boxes (these were described in the SudokuSolver solver section) with the digits from the Sudoku grid received in the post request. Iterate over all the empty cells of the grid and see what numbers cannot be there by using the binary OR operator on the bitmasks (if a digit is on the corresponding row, column OR 3x3 box, then it cannot be placed in the cell). We count how many digits from 1 to 9 are not part of the bitmask (each digit is encoded as a bitmask as (1<<digit), so performing the AND binary operation between the digit bitmask and the number obtained by performing the OR described above, can be used to check whether the digit can be placed in that cell or not). In this way, we find the cell with the minimum number of possible digits in that cell and return the board with that cell completed according to the stored solution. If there is more than one such cell, we will pick the one that occurred last in our iteration.

The Miracle Sudoku will use the same hint function. Note that by using this hint function, there is a high chance that we will reveal a cell that is not the next one that a human solver would pick (because miracle Sudoku has multiple other rules). However, in this case, revealing another number of the board creates a new set of possible values for neighbouring cells that the human solver can explore starting from that number. Since when solving the Miracle Sudoku a user starts from one number and tries to guess different cells, and narrows down the values of neighbouring cells, having a new number on the board that is not a clear next pick makes the game more interesting (gives a different area of the grid to start exploring for example).

210003309 - *Aside*

An addition to the hint button was that of a 10/15/20 second penalty. From our playtesting of the site's puzzles, we felt there was multiple instances where using a hint drastically reduced the difficulty (more importantly the time taken) to find the value of a cell, select that cell and then type the correct value; this meant that it would be in a user's best interest to use 3 hints, especially in the midgame of a puzzle solve which would save them a lot of time. To balance this, we added this penalty system where for the first hint, 10 seconds was added to their final time, 15 for the second and 20 for the third. From playtesting of this system, we found out that users, as intended, were more strategic about their use of hints, only using them if they felt as though it would take them longer to work out the value than to use the hint. Since the leaderboard does not discriminate on the number of hints used by a player, the best strategy used to be using all three but with this modification, for some puzzles, it may be best not to.

Checking for mistakes and winning conditions (200006767)

Our website automatically checks each number that the user writes in a cell of the sudoku and miracle sudoku grids. If the number is wrong, the board will shake and the cell will become red, after which the number will be automatically deleted. All numbers equal to the value are also highlighted. The site detects if the number is correct by sending a post request to the backend.

In the sudoku.py file, the route /sudoku/check-solution uses the sudoku solver compareToSolution() method that checks each non-empty cell of the grid against the solution stored in the backend. This works for both traditional sudokus and miracle sudokus. Moreover, for checking if a complete board is correct we have a separate route "/sudoku/check-full-solution" that uses the same method with the additional parameter full=False. In this case, all elements of the board are compared against the solution (so if some cells were empty this would fail). If the check-full-solution check succeeds, then the user wins the game and the rating section appears in the frontend.

Minesweeper Logic (200009744)

In our site, all minesweeper puzzles are generated randomly using a Python script called minesweeperGenerator.py . To generate a new puzzle, the script requires the number of rows, columns, and mines to be specified.

The script begins by creating a 2D array with a specified number of rows and columns. Each element in the array represents a cell on the minesweeper board and is initialised to 0. This value will later be updated to represent the number of adjacent mines.

Next, the script randomly selects positions on the board to place the specified number of mines. This process is repeated until all mines have been placed on the board.

A list is created to store the positions of all mines on the board. The script then loops through each mine in the list and increments the value of the 8 adjacent cells. This step ensures that each cell on the board accurately reflects the number of adjacent mines.

Once all cells have been updated, the fully generated minesweeper board is returned to the endpoint /getminesweeper.

Databases (200009744)

In this project, we made the decision to use MongoDB as our database. One of the primary reasons for choosing MongoDB is its compatibility with the Python programming language, which is our language of choice for this project. MongoDB provides a straightforward API for easy manipulation of the database.

Additionally, our project requires the storage of data as key-value pairs, and MongoDB's JSON-like document structure is well-suited to this design. The ability to store data as documents make it easier to work with and modify the data as needed throughout the development process.
Another benefit of using MongoDB is its scalability, which allows us to expand our database as our project grows in size and complexity. This scalability also ensures that we can make any future changes or additions to our data storage needs without having to significantly alter our database structure.

Supergroup Login - 200006767

One of the core requirements of the project is allowing federated login within our supergroup. Last semester, our supergroup decided on implementing an OAuth authentication system that allows secure login functionality between the federation websites. The implementation is rather complex because it enforces multiple layers of security that are described in detail below.

When a user goes to the sign-up page of our website, they have a scroll-down button that allows them to log in with one of our partner websites. Once they select the website and press the login button, this button redirects them to the website that has their user information stored. The URL of this page will have the format: https://cs3099userXX.host.cs.st-andrews.ac.uk/oauth/authorize?client_id=aces5&redirect_url=https://cs3099user05.host.cs.st-andrews.ac.uk/oauth/redirect/{their website id}. Each website has an associated client_id, for example, aces-eight corresponds to group 8: https://cs3099user08.host.cs.st-andrews.ac.uk. In order to create the URL from above, on our login page, we append to the selected option website our client_id (aces5) and a redirect URL that is in the format https://cs3099user05.host.cs.st-andrews.ac.uk/oauth/redirect/{their website id}. (their website id could be for example aces-eight)

Once the client inputs their credentials (login details) into the other group's oauth/authorize web page described above, they are redirected back to our website to the redirect url from the url specified above: https://cs3099user05.host.cs.st-andrews.ac.uk/oauth/redirect/{their website id}. To this url they will append a query parameter code that represents the encrypted username and the client_id of the website that has the user information. On our website, this feature can be seen on the front end of the FederationLogin.vue page. We extract the redirect_url and client_id from the url and if the login is successful, then we encrypt the user and client_id on the backend. This encryption is performed using the /generate-code route on the backend. In order to ensure security encryption and easy decryption we used the python Fernet package that uses the DES encryption protocol. A random passphrase (secret key) was generated and stored on the backend. Once the encryption happens on the backend, we send the encrypted username to the frontend. The encrypted username code is appended to the redirect URL and the user is redirected to that URL, which is the website that they want to access.

On the redirect page, we extract the code from the URL and the client_id of the website that provided the code. Now the extracted information is sent to our backend using the oauth/redirect route. In the oauth/redirect backend route, the client_id is mapped to the backend of the corresponding website from the federation. We are sending a post request to the backend of the other website at the route oauth/token. This post request will send our client_id, the code received from the url, and a client_secret. The client secret for each website is securely stored on the backend of all the websites in the federation and it is used as an extra layer of security.

The oauth/token route receives a client_id, an access_code representing the code earlier encrypted by the backend (the one for which we used DES encryption) and the secret_code of the website sending the post request. Since the backend of the website stores both the client_id and the secret_code of the website that sent the post request, it will check that the client_id and secret_code match, if they don't then this is unauthorized access, and the response will be 404, terminating the login attempt. If they do match, then the post request must come from a website of the federation and the protocol will continue in the following way: the backend stores the key for the DES encryption used on the access_code, and it uses the key to decrypt the user information. For additional security, the decrypted access code is now encrypted again with a different secret key stored on the backend of the website and sent as a response to the post request (the newly encrypted data is labeled access_token). In the oauth/redirect route of our backend the protocol continues, and the access_token is now sent back to the other website using the client_url and the api/user/me route. This post request is an exchange from which we offer an encrypted access_token and we receive in return the unencrypted user information.

The api/user/me route receives the access_token, uses the secret_key stored on the backend to decrypt the access_token and if successful it will send the username as part of the response. In the response, we are also sending an origin that contains the url to our website, the name of the website, and the way we uniquely identify a user on our website (the username). When this information is sent, the response is received inside the oauth/redirect route and the username and url are taken out of the response object. Since the url of the website contains the group number, we can extract the group_number (a number from 1 to 9). We will send the username and the origin of the website to the frontend, FederationRedirect component in the format "username from group X". where X is a digit from 1 to 9. On the frontend, we store the username in local storage and send a post request to the backend route /register-supergroup-user (this can be found in the register.py file). If the user is already registered in the database, then we don't do anything, otherwise, we store the user in the database and give them creator permissions, then the user is logged into our website and their username can be clearly seen in on the right side of the website.

Some important decisions that we made, were that supergroup users will have the creator privilege and will be allowed to create puzzles on our website and an username on our website cannot contain the space character, therefore all supergroup usernames are different from the ones on our website because of their format "{username} from group X". Moreover, all groups in the federation agreed to use and enforce unique usernames on their websites, so there cannot be 2 people from the website with the same username. These decisions enforce the uniqueness of the usernames and accounts on our website and it also offers a nice visual distinction between normal and supergroup users in the comments section. (An example would be userTest from group 3 commented: Hello).

Sharing Puzzles (200009744)

Our site allows players to download sudoku puzzles. On the play sudoku page, there is a download button, when clicked, triggers our download puzzle endpoint, which sends a JSON file back. The file containing the sudoku grid, solution, variants, author, sudoku name, and sudoku ID to the user's device for download. Additionally, users can upload puzzles they have downloaded from other sites to our platform.

The downloaded file format is standardised across all groups in our supergroup, allowing players to download puzzles from our site and upload them to other sites, and vice versa.

Furthermore, we have an endpoint for retrieving all puzzles stored in our database. However, we currently do not have any specific usage for this endpoint.

To implement the endpoint for downloading puzzles, we have used the Flask API called send_file. This API allows us to send a file back to the client in response to a GET request. We have written a Flask route that receives the download request and retrieves the necessary puzzle data from our database. We then create a JSON file using this data and use the send_file method to send the file back to the client.

On the front end, we have implemented a function that handles the received file and initiates the download process. When the download button is clicked, the function sends a GET request to our endpoint and waits for the response. Once the response is received, the function process the file data, creates a new file and then creates a downloadable link for this new file for the user to save.

Testing

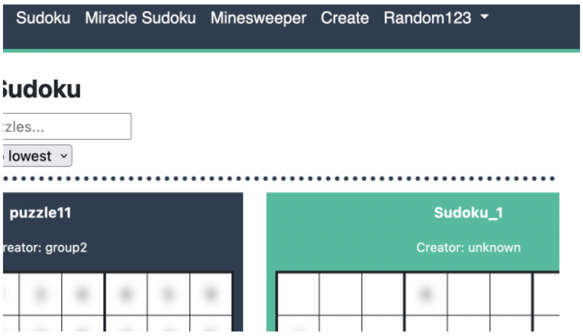
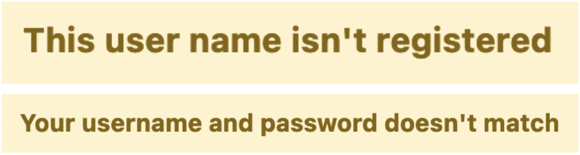
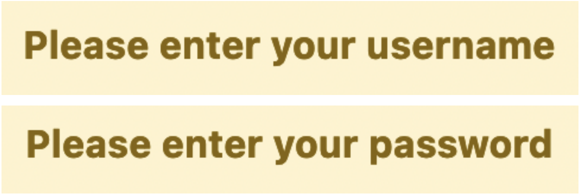
Key	
Functioning as intended.	Not functioning as intended.

Frontend

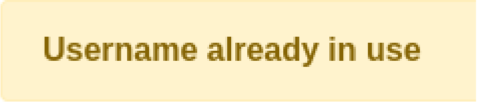
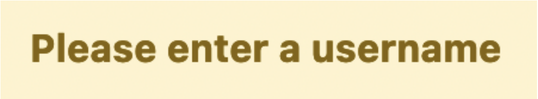
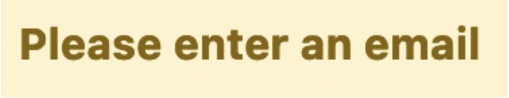
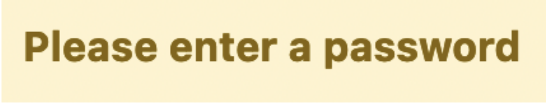
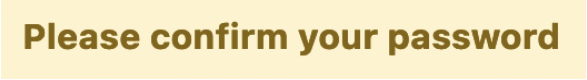
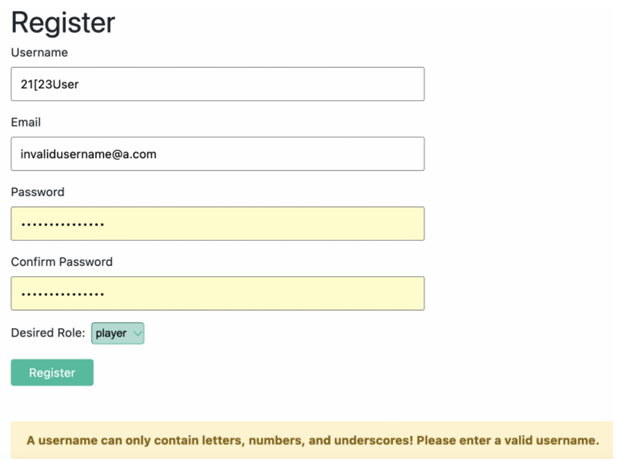
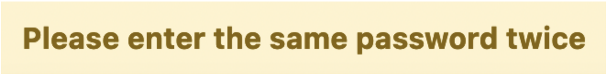
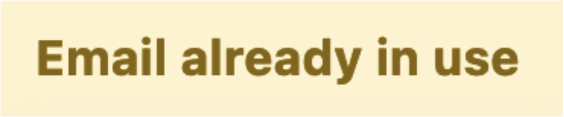
Accounts

Login (200022418)

To test login features, .This section is primarily concerned with code located in Login.vue

Description	Expected	Actual
Valid login details	Redirects user to the home page and displays their username in the navigation bar	Once the Random123 account is registered, it redirects to the home page and displays the username like so:  Sudoku Miracle Sudoku Minesweeper Create Random123 ıudoku zles... lowest puzzle11 reator: group2 Sudoku_1 Creator: unknown
Invalid login details (incorrect username or incorrect password)	An error message when invalid login credentials are entered.	 This user name isn't registered Your username and password doesn't match
Empty fields	An alert is displayed to the user asking them to fill the relevant fields	 Please enter your username Please enter your password

Register (200022418)

Description	Expected	Actual
Valid registration details	The user is redirected to the login page where they can login with their new account details	The user is redirected to the login page and the user can login to the website with the registration details they entered
Duplicate username registration	An account cannot be registered if the username is already in use and an alert message is displayed	 Username already in use
Empty fields	An alert is displayed to the user asking them to fill the relevant fields	 Please enter a username  Please enter an email  Please enter a password  Please confirm your password
Invalid username	A username can only consist of letters, numbers, and underscores. If any spaces or special characters are used, an alert will be displayed	 Register Username: 21[23User Email: invalidusername@a.com Password: ··············· Confirm Password: ··············· Desired Role: player A username can only contain letters, numbers, and underscores! Please enter a valid username.
Incorrect confirm password	Displays a message alerting the user that the confirm password is different to the password provided	 Please enter the same password twice
Duplicate email registration	An account cannot be registered if the email is already in use and an alert message is displayed	 Email already in use
Duplicate username registration with different capitalisation	An account with the same username but different capitalisation shouldn't be allowed to register on the website and an alert should be displayed to tell the user that the username is already in use	Random123 and random123 can be registered as two different accounts on the website

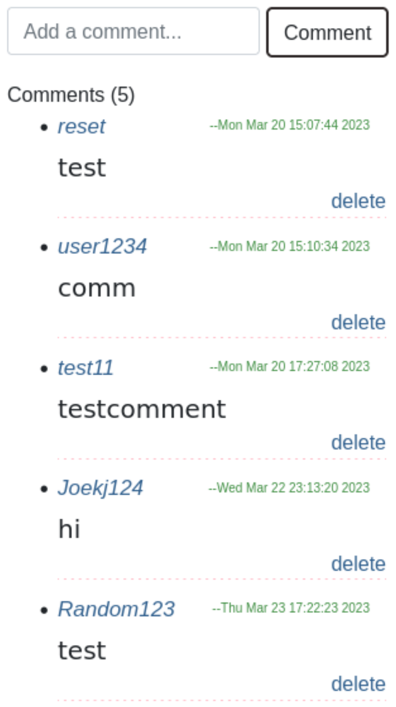
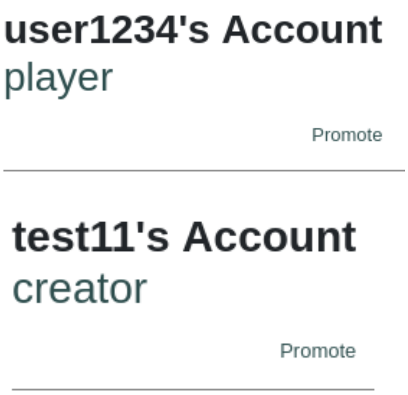
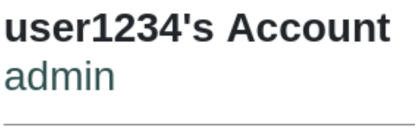
[Figure 7: Login Testing Table.]

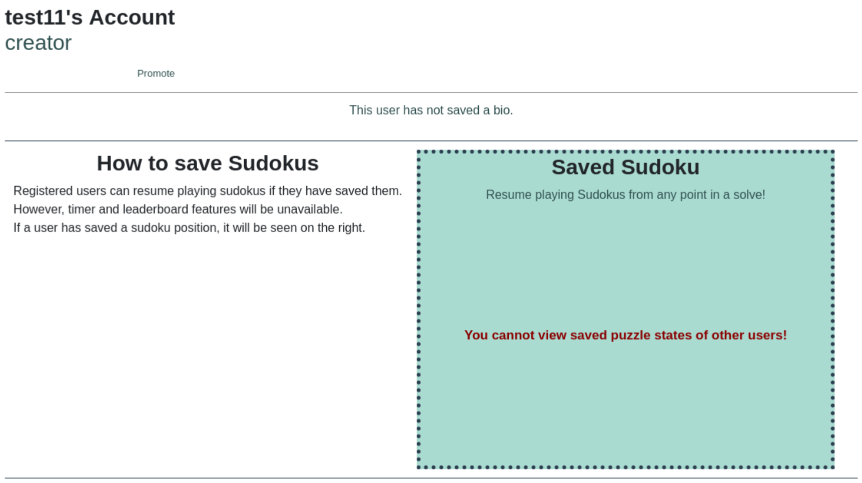
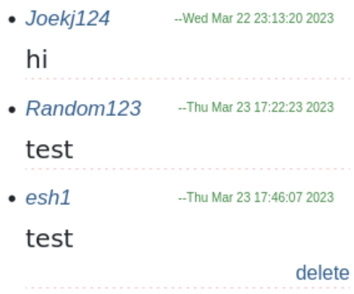
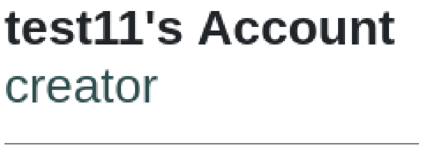
Permissions (200022530)

All the following tests were performed using the two following user accounts and have provided their login details below.

esh1 (Player)	Random123 (Admin)
Username: esh1 Password: Password1	Username: Random123 Password: Random123

[Figure 8: Account logins for Permissions tests.]

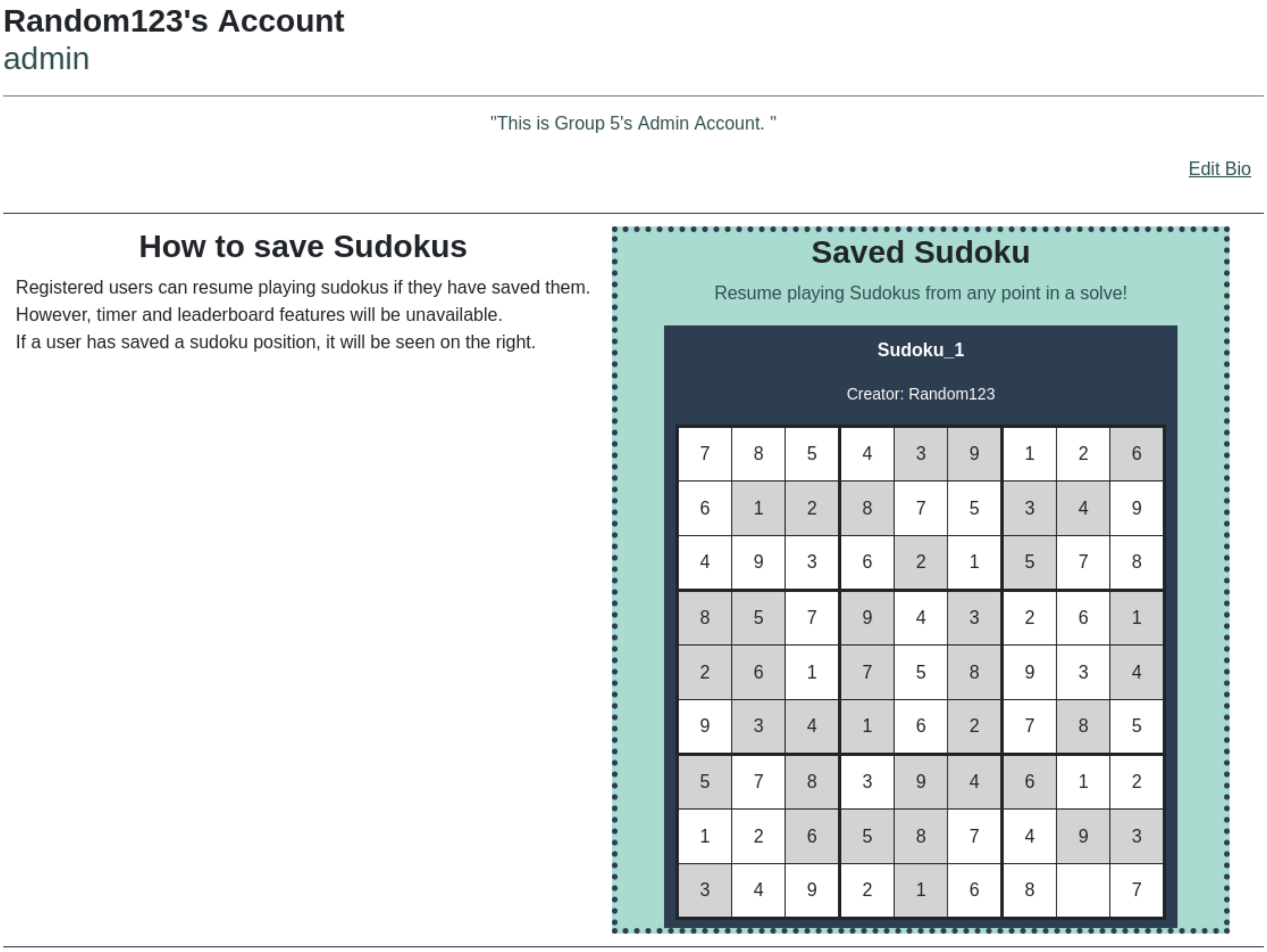
Description	Expected	Actual
The following tests were performed logged into the Random123 account.		
Admins can delete comments from all different user roles.		 Add a comment... Comment Comments (5) • *reset* --Mon Mar 20 15:07:44 2023 test delete • *user1234* --Mon Mar 20 15:10:34 2023 comm delete • *test11* --Mon Mar 20 17:27:08 2023 testcomment delete • *Joekj124* --Wed Mar 22 23:13:20 2023 hi delete • *Random123* --Thu Mar 23 17:22:23 2023 test delete As expected.
Admins can promote players and creators.	The promote button is displayed under player and creator accounts.	 **user1234's Account** player Promote **test11's Account** creator Promote
	Promoted creator or player users become admins after being promoted.	 **user1234's Account** admin

	The promote button is not displayed under admin accounts.	*(See above screenshot)*
Users cannot view Saved Sudokus of other users.	 **test11's Account** creator Promote This user has not saved a bio. **How to save Sudokus** Registered users can resume playing sudokus if they have saved them. However, timer and leaderboard features will be unavailable. If a user has saved a sudoku position, it will be seen on the right. **Saved Sudoku** Resume playing Sudokus from any point in a solve! **You cannot view saved puzzle states of other users!**	
Users are not provided the option to edit other user's bios.	*(See above screenshot)*	
The following tests were performed logged into the esh1 account.		
Users can delete their own comments.		 - *Joekj124* --Wed Mar 22 23:13:20 2023 hi - *Random123* --Thu Mar 23 17:22:23 2023 test - *esh1* --Thu Mar 23 17:46:07 2023 test delete As expected.
Player and Creator users cannot promote other users.		 **test11's Account** creator As expected.

[Figure 9: Permissions Testing Table.]

Account Pages (200022530)

In the below figure, we have provided evidence that our Account pages adequately displays account information from the database. All tests in this section were performed whilst logged in to the Random123 user account. The Saved Sudoku functionality will be tested later in the “Testing:Playing Sudokus” of this report.



Random123's Account
admin

"This is Group 5's Admin Account. "

Edit Bio

How to save Sudokus

Registered users can resume playing sudokus if they have saved them.
However, timer and leaderboard features will be unavailable.
If a user has saved a sudoku position, it will be seen on the right.

Saved Sudoku

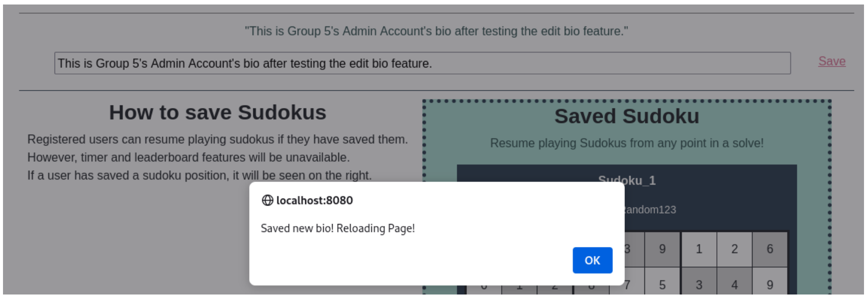
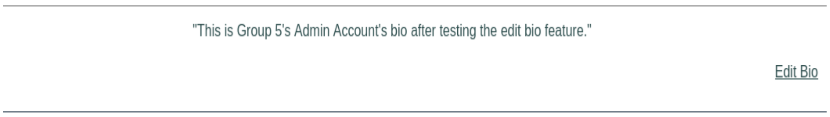
Resume playing Sudokus from any point in a solve!

Sudoku_1

Creator: Random123

7	8	5	4	3	9	1	2	6
6	1	2	8	7	5	3	4	9
4	9	3	6	2	1	5	7	8
8	5	7	9	4	3	2	6	1
2	6	1	7	5	8	9	3	4
9	3	4	1	6	2	7	8	5
5	7	8	3	9	4	6	1	2
1	2	6	5	8	7	4	9	3
3	4	9	2	1	6	8		7

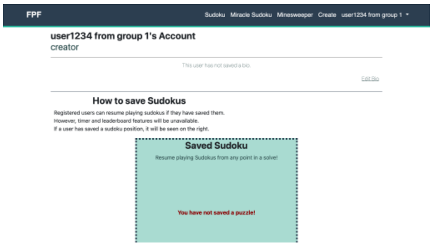
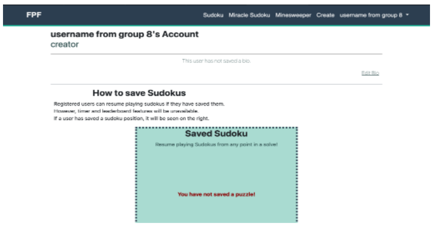
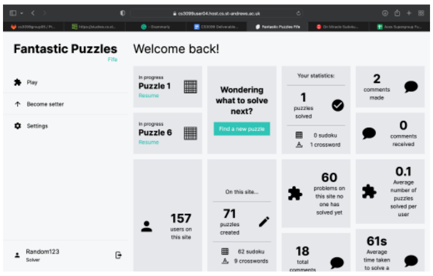
[Figure 10: Screenshot of Random123’s account page.]

Description	Expected	Actual
Users can edit and save their own bios.	 "This is Group 5's Admin Account's bio after testing the edit bio feature." This is Group 5's Admin Account's bio after testing the edit bio feature. Save How to save Sudokus Registered users can resume playing sudokus if they have saved them. However, timer and leaderboard features will be unavailable. If a user has saved a sudoku position, it will be seen on the right. Saved Sudoku Resume playing Sudokus from any point in a solve! Sudoku_1 localhost:8080 Saved new bio! Reloading Page! OK	
Users bios after editing are displayed on their account page.	 "This is Group 5's Admin Account's bio after testing the edit bio feature." Edit Bio	

[Figure 11: Account Pages Testing Table.]

Federation Login (200006767)

We tested Account logins with our supergroup by collaborating with the other teams in the supergroup and achieved full interoperability with all of them. This result is accurate as of Friday 24th March 2023. The results of our compatibility tests can be found in Federation Compatibility in our overview (this is a shared document of the supergroup).

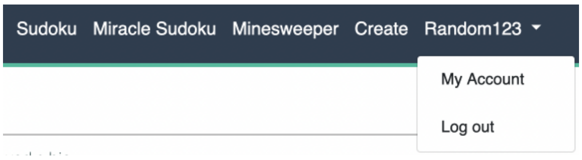
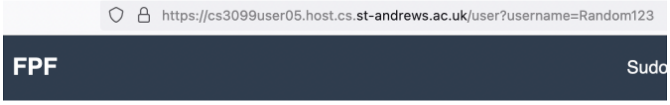
Description	Expected	Actual
Login in to our website, credentials on website cs3099user01	User logs in with creator permission, and has from group 1 appended to the username	Works as expected  FPF Sudoku Miracle Sudoku Minesweeper Create user1234 from group 1 Edit Bio user1234 from group 1's Account creator This user has not saved a bio. How to save Sudokus Registered users can resume playing sudokus if they have saved them. However, timer and leaderboard features will be unavailable. If a user has saved a sudoku position, it will be seen on the right. Saved Sudoku Resume playing Sudokus from any point in a solve! You have not saved a puzzle!
Login in to our website, credentials on website cs3099user08	User logs in with creator permission, and has from group 8 appended to the username	 FPF Sudoku Miracle Sudoku Minesweeper Create username from group 8 Edit Bio username from group 8's Account creator This user has not saved a bio. How to save Sudokus Registered users can resume playing sudokus if they have saved them. However, timer and leaderboard features will be unavailable. If a user has saved a sudoku position, it will be seen on the right. Saved Sudoku Resume playing Sudokus from any point in a solve! You have not saved a puzzle!
Login into group 4's website, credentials on our website, login in as user Random123 with password Random123	The other website can login to their website with a user stored on our website.	Works as expected  cs3099user04.host.cs.st-andrews.ac.uk Fantastic Puzzles File Play Become setter Settings Random123 Solver Resume Find a new puzzle Fantastic Puzzles Fife Welcome back! In progress Puzzle 1 In progress Puzzle 6 Wondering what to solve next? Your statistics: 1 puzzles solved 0 sudoku 1 crossword 2 comments made 0 comments received 157 users on this site On this site... 71 puzzles created 62 sudoku 9 crosswords 60 problems on this site no one has solved yet 0.1 Average number of puzzles solved per user 18 total comments 61s Average time taken to solve a

[Figure 12: Federation Logins Testing table.]

Navigation

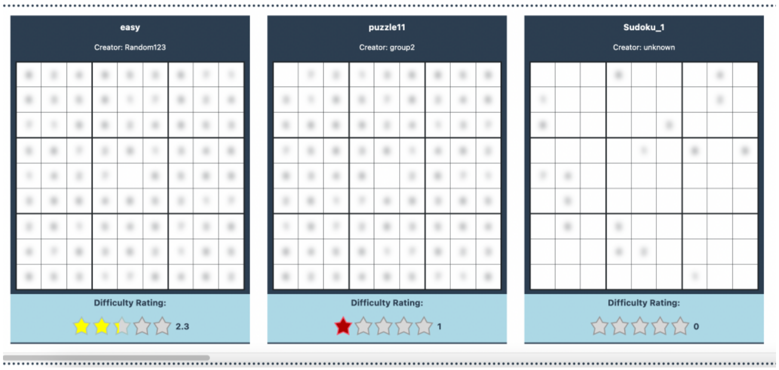
NavBar / Routes (200022418)

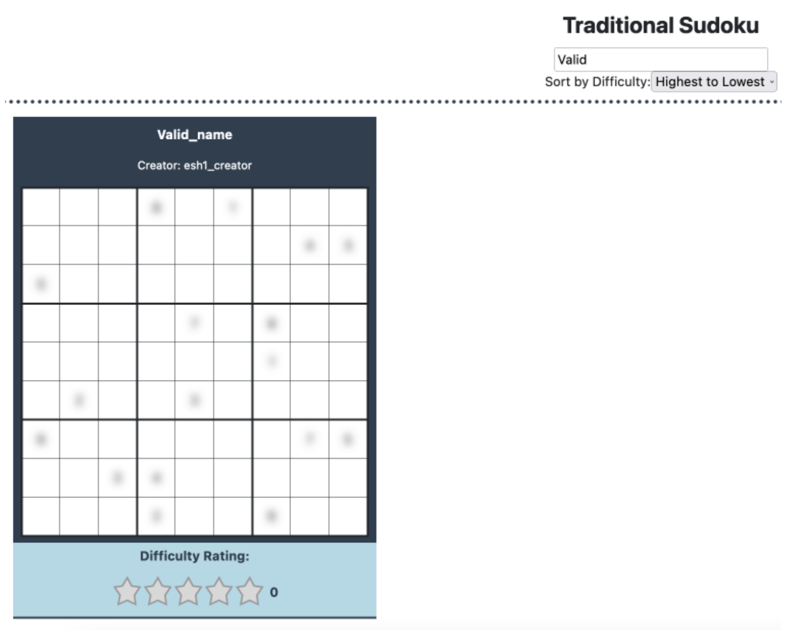
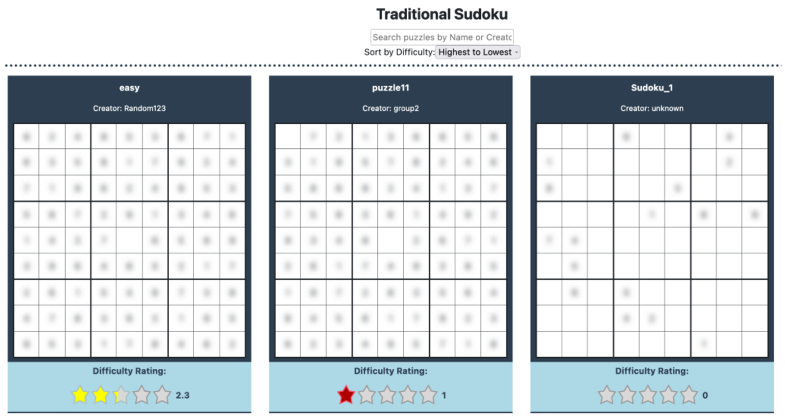
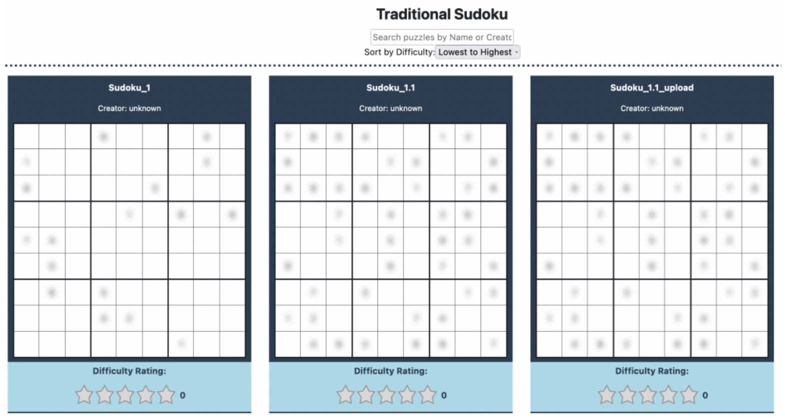
Description	Expected	Actual
FPF logo	Redirects the user to the home page when clicked	Works as expected
Sudokus	Redirects the user to the sudokus page when clicked - /sudokus	Works as expected
Miracle Sudokus	Redirects the user to the miracle sudokus page when	Works as expected

	clicked - /miracle-sudokus	
Minesweeper	Redirects the user to the minesweeper page when clicked - /minesweeper	Works as expected
Create	Redirects the user to the “Create Sudoku” page when clicked - /create	Works as expected
User dropdown menu	Provides the user with two options - view their account page or log out of their account	 Sudoku Miracle Sudoku Minesweeper Create Random123 My Account Log out
My Account	Redirects the user to their account page when clicked - /user?username=[username]	 https://cs3099user05.host.cs.st-andrews.ac.uk/user?username=Random123 FPF Sudo The url updates with the user’s username and the user is redirected to the account page
Log out	Logs the user out and redirects them to the login page - /login	Works as expected

[Figure 13: NavBar Testing Table.]

Home Page / Sudoku Selector (200022418)

Description	Expected	Actual
Traditional sudoku list	Users are able to see a list of traditional sudokus available on the website along with their name, creator, and difficulty rating	 easy Creator: Random123 Difficulty Rating: 2.3 puzzle11 Creator: group2 Difficulty Rating: 1 Sudoku_1 Creator: unknown Difficulty Rating: 0

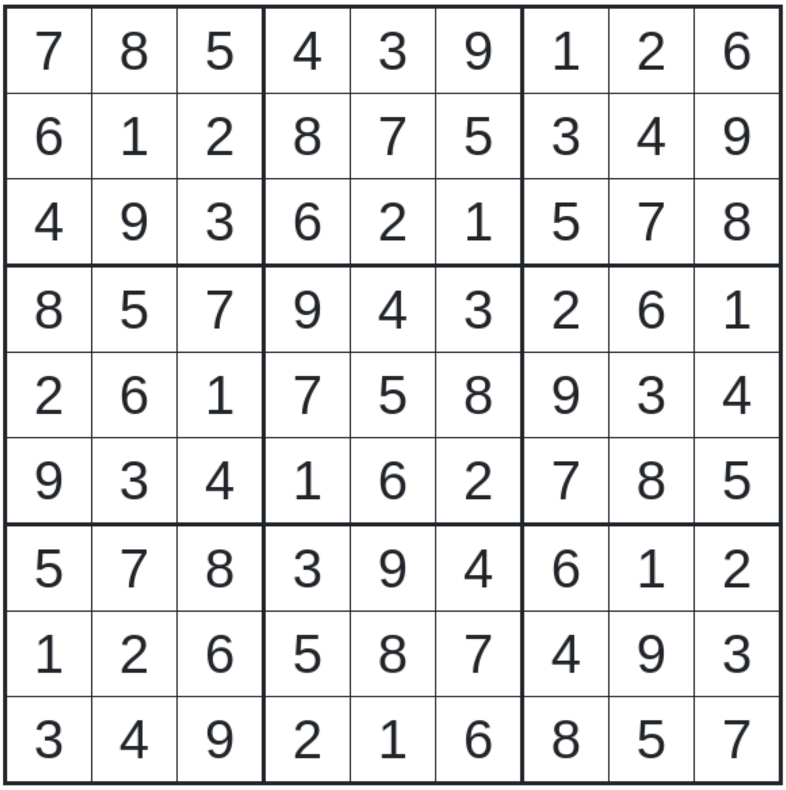
Search functionality	Users are able to search for sudokus by name or by the creator of the sudoku	 Traditional Sudoku Valid Sort by Difficulty: Highest to Lowest Valid_name Creator: esh1_creator Difficulty Rating: 0
Sort by rating functionality	Users are able to sort the list of sudokus on the website by difficulty rating	Highest to lowest:  Traditional Sudoku Search puzzles by Name or Creatc Sort by Difficulty: Highest to Lowest easy Creator: Random123 Difficulty Rating: 2.3 puzzle11 Creator: group2 Difficulty Rating: 1 Sudoku_1 Creator: unknown Difficulty Rating: 0 Lowest to highest:  Traditional Sudoku Search puzzles by Name or Creatc Sort by Difficulty: Lowest to Highest Sudoku_1 Creator: unknown Difficulty Rating: 0 Sudoku_1.1 Creator: unknown Difficulty Rating: 0 Sudoku_1.1_upload Creator: unknown Difficulty Rating: 0
Scrolling the selector shows all puzzles	Users are able to scroll horizontally through the selector to view all available sudokus	Works as expected

[Figure 14: Sudoku Selector Testing Table .]

Playing Sudokus

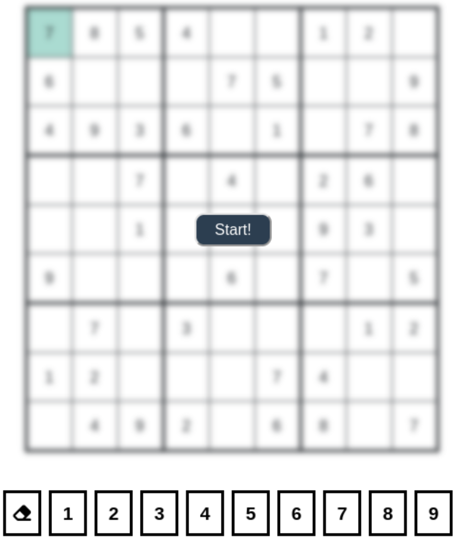
Gameplay (200022530)

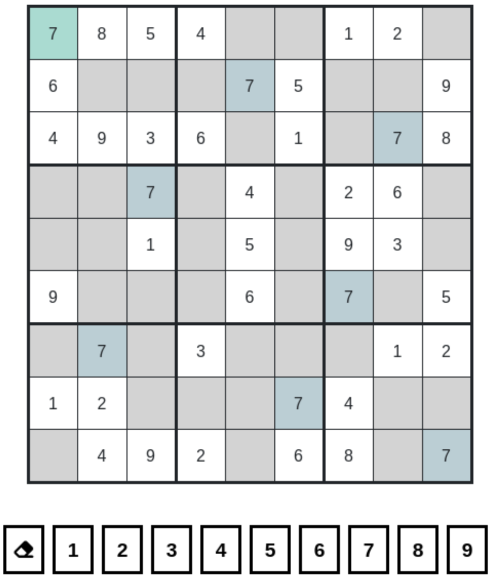
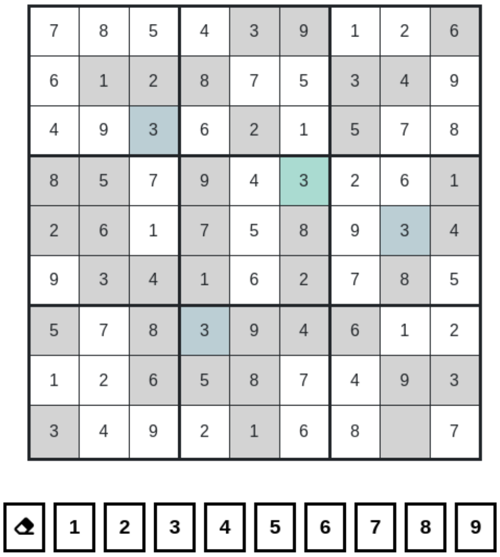
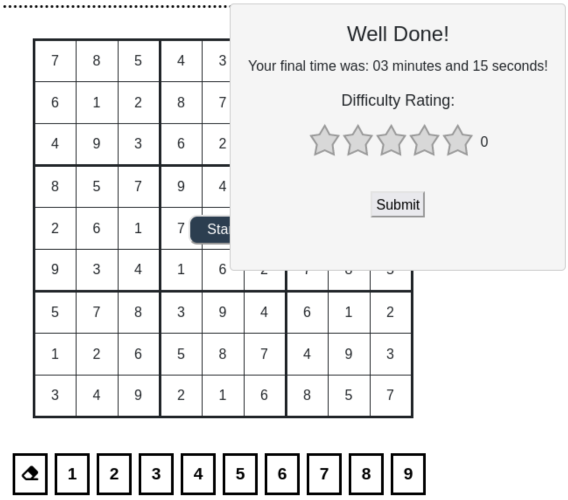
These tests are concerned with playing Sudokus in the frontend of our website, first testing without the implemented resuming functionality. All of these tests were performed on Sudoku_1 and Sudoku_2, but we have only provided screenshots of one for evidence. In the below figure, I have reproduced the solution of Sudoku_1 so that our testing can be followed easier.



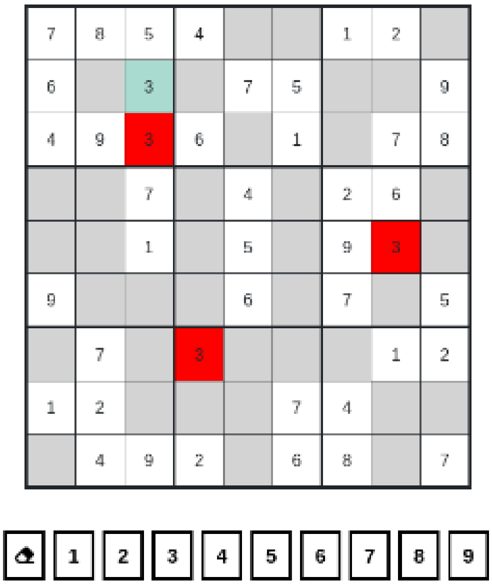
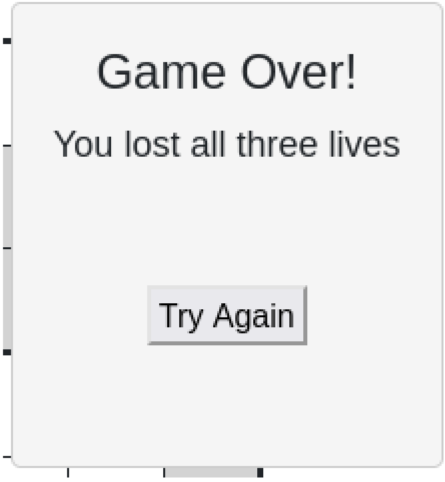
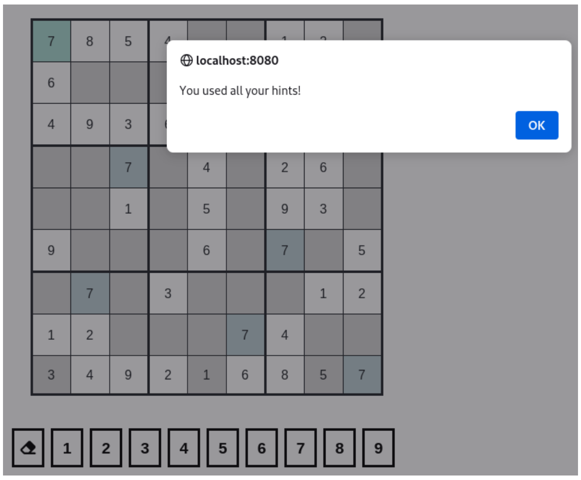
7	8	5	4	3	9	1	2	6
6	1	2	8	7	5	3	4	9
4	9	3	6	2	1	5	7	8
8	5	7	9	4	3	2	6	1
2	6	1	7	5	8	9	3	4
9	3	4	1	6	2	7	8	5
5	7	8	3	9	4	6	1	2
1	2	6	5	8	7	4	9	3
3	4	9	2	1	6	8	5	7

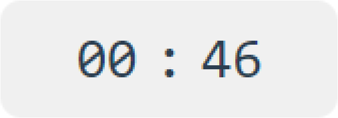
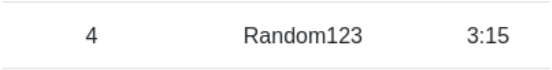
[Figure 15: Screenshot of Sudoku_1 solution.]

Description	Expected	Actual
Users cannot view Sudokus before starting the timer.	The Sudoku is blurred when first navigating to the puzzle's page.	

<table>
<tr><th>Description</th><th>Expected</th><th>Actual</th></tr>
<tr><td>Users can view Sudokus after pressing “Start!”.</td><td>After pressing start, the Sudoku becomes unblurred.</td><td>
<table>
<tr><td>7</td><td>8</td><td>5</td><td>4</td><td></td><td></td><td>1</td><td>2</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td>7</td><td>5</td><td></td><td></td><td>9</td></tr>
<tr><td>4</td><td>9</td><td>3</td><td>6</td><td></td><td>1</td><td></td><td>7</td><td>8</td></tr>
<tr><td></td><td></td><td>7</td><td></td><td>4</td><td></td><td>2</td><td>6</td><td></td></tr>
<tr><td></td><td></td><td>1</td><td></td><td>5</td><td></td><td>9</td><td>3</td><td></td></tr>
<tr><td>9</td><td></td><td></td><td></td><td>6</td><td></td><td>7</td><td></td><td>5</td></tr>
<tr><td></td><td>7</td><td></td><td>3</td><td></td><td></td><td></td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>2</td><td></td><td></td><td></td><td>7</td><td>4</td><td></td><td></td></tr>
<tr><td></td><td>4</td><td>9</td><td>2</td><td></td><td>6</td><td>8</td><td></td><td>7</td></tr>
</table>
</td></tr>
<tr><td>Users can enter numbers into the Sudoku with either the keyboard or mouse.</td><td>Entered numbers will remain in editable cells (the grey ones).</td><td>
<table>
<tr><td>7</td><td>8</td><td>5</td><td>4</td><td>3</td><td>9</td><td>1</td><td>2</td><td>6</td></tr>
<tr><td>6</td><td>1</td><td>2</td><td>8</td><td>7</td><td>5</td><td>3</td><td>4</td><td>9</td></tr>
<tr><td>4</td><td>9</td><td>3</td><td>6</td><td>2</td><td>1</td><td>5</td><td>7</td><td>8</td></tr>
<tr><td>8</td><td>5</td><td>7</td><td>9</td><td>4</td><td>3</td><td>2</td><td>6</td><td>1</td></tr>
<tr><td>2</td><td>6</td><td>1</td><td>7</td><td>5</td><td>8</td><td>9</td><td>3</td><td>4</td></tr>
<tr><td>9</td><td>3</td><td>4</td><td>1</td><td>6</td><td>2</td><td>7</td><td>8</td><td>5</td></tr>
<tr><td>5</td><td>7</td><td>8</td><td>3</td><td>9</td><td>4</td><td>6</td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>2</td><td>6</td><td>5</td><td>8</td><td>7</td><td>4</td><td>9</td><td>3</td></tr>
<tr><td>3</td><td>4</td><td>9</td><td>2</td><td>1</td><td>6</td><td>8</td><td></td><td>7</td></tr>
</table>
</td></tr>
<tr><td>Users are provided with a suitable message after completing a puzzle.</td><td>An alert pops up when the last cell is filled and the solution is correct. After submitting a rating, the user is redirected back to the Traditional Sudoku selector.</td><td>
Well Done!
Your final time was: 03 minutes and 15 seconds!
Difficulty Rating:
☆☆☆☆☆ 0
Submit

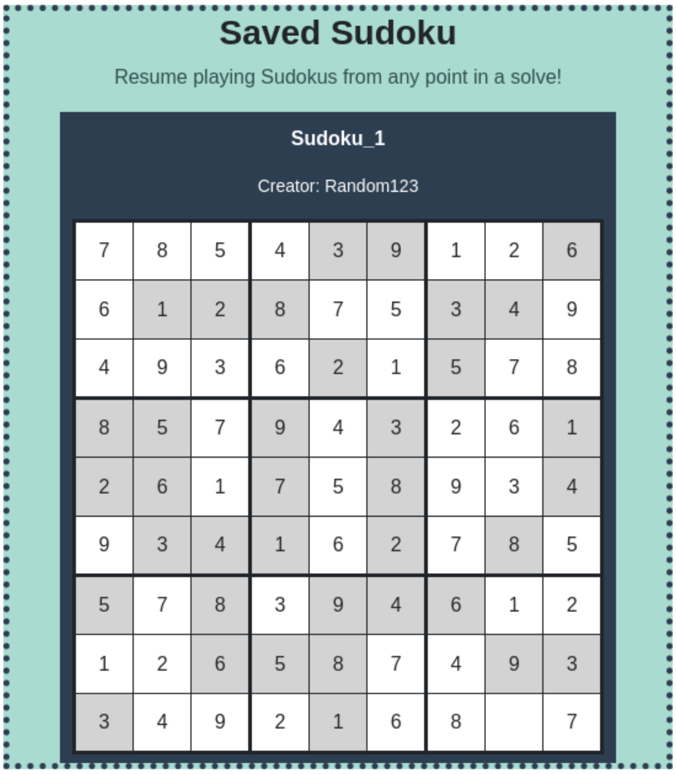
As expected.</td></tr>
</table>

Description	Expected	Actual
Users are adequately notified when they make a mistake.	The incorrectly placed number displays for a little time before it is removed. Other numbers that match the wrong number are highlighted.	 7 8 5 4 1 2 6 3 7 5 9 4 9 3 6 1 7 8 7 4 2 6 1 5 9 3 9 6 7 5 7 3 1 2 1 2 7 4 4 9 2 6 8 7 1 2 3 4 5 6 7 8 9 As expected.
Users fail puzzles after 3 mistakes.	A “Game Over!” alert is shown and allows the users to try again by refreshing the page. After losing, the user cannot edit the grid anymore and the timer stops.	 Game Over! You lost all three lives
Users can use up to 3 hints.	Users can use no more than 3 hints and are notified when they have reached that limit.	 localhost:8080 You used all your hints!

Description	Expected	Actual
Using a hint adds a penalty	Hint penalties correspond to the number of hints used so far.	 00 : 46 *Three successive uses of hint straight after game start (10 + 15 + 20)*
Users can submit times to the leaderboard.		 4 Random123 3:15

[Figure 16: Table for testing Traditional Sudoku gameplay.]

The following tests will ensure that resuming a user's saved Sudoku functions as intended where it would differ from a non-saved Sudoku. All tests were performed using the saved Sudoku state shown in the figure below.



Saved Sudoku

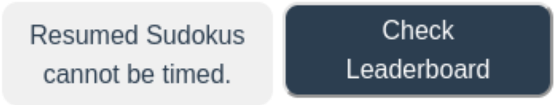
Resume playing Sudokus from any point in a solve!

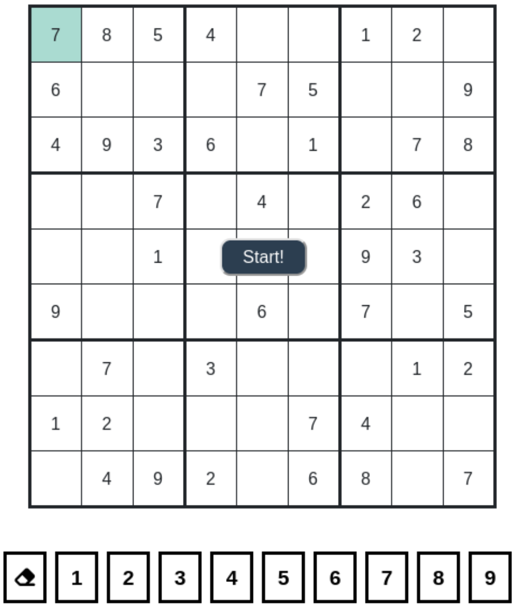
Sudoku_1

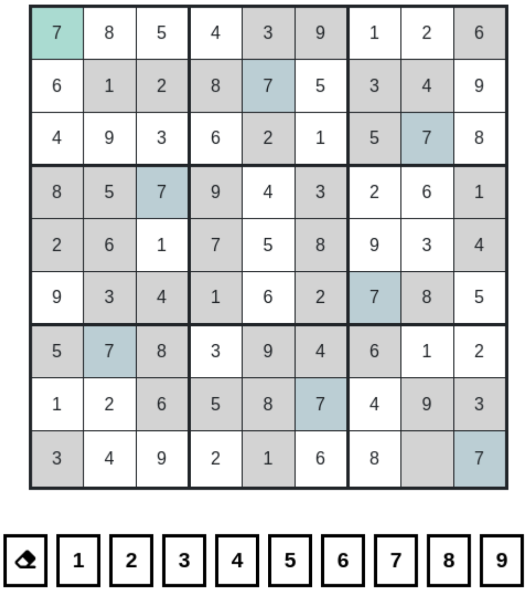
Creator: Random123

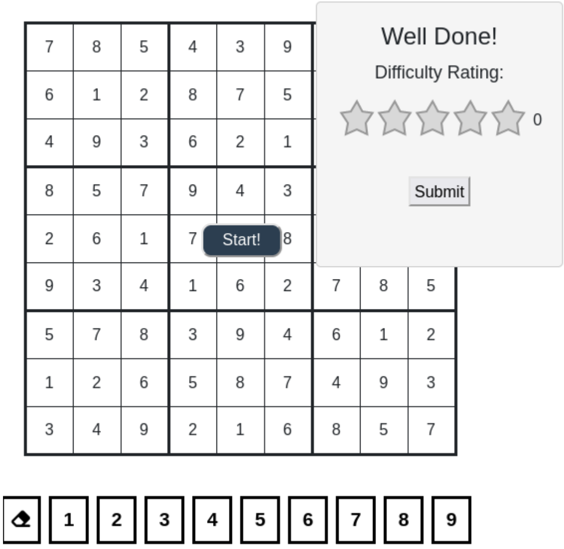
7	8	5	4	3	9	1	2	6
6	1	2	8	7	5	3	4	9
4	9	3	6	2	1	5	7	8
8	5	7	9	4	3	2	6	1
2	6	1	7	5	8	9	3	4
9	3	4	1	6	2	7	8	5
5	7	8	3	9	4	6	1	2
1	2	6	5	8	7	4	9	3
3	4	9	2	1	6	8		7

[Figure 17: Screenshot of Saved Sudoku for following tests.]

Description	Expected	Actual
Users cannot time themselves playing resumed Sudokus.	Timer is replaced with text notifying the user of this feature.	 Resumed Sudokus cannot be timed.

<table>
<tr>
<td>Users can see saved Sudoku’s original state before starting</td>
<td>Users can see the saved Sudoku puzzle, but it won’t be populated with their saved state until they click “Start!”.</td>
<td>

<table>
<tr><td>7</td><td>8</td><td>5</td><td>4</td><td></td><td></td><td>1</td><td>2</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td>7</td><td>5</td><td></td><td></td><td>9</td></tr>
<tr><td>4</td><td>9</td><td>3</td><td>6</td><td></td><td>1</td><td></td><td>7</td><td>8</td></tr>
<tr><td></td><td></td><td>7</td><td></td><td>4</td><td></td><td>2</td><td>6</td><td></td></tr>
<tr><td></td><td></td><td>1</td><td colspan="3">Start!</td><td>9</td><td>3</td><td></td></tr>
<tr><td>9</td><td></td><td></td><td></td><td>6</td><td></td><td>7</td><td></td><td>5</td></tr>
<tr><td></td><td>7</td><td></td><td>3</td><td></td><td></td><td></td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>2</td><td></td><td></td><td></td><td>7</td><td>4</td><td></td><td></td></tr>
<tr><td></td><td>4</td><td>9</td><td>2</td><td></td><td>6</td><td>8</td><td></td><td>7</td></tr>
</table>

</td>
</tr>
<tr>
<td>Users can play from resumed state.</td>
<td>After pressing the start button users can resume from their saved position.</td>
<td>

<table>
<tr><td>7</td><td>8</td><td>5</td><td>4</td><td>3</td><td>9</td><td>1</td><td>2</td><td>6</td></tr>
<tr><td>6</td><td>1</td><td>2</td><td>8</td><td>7</td><td>5</td><td>3</td><td>4</td><td>9</td></tr>
<tr><td>4</td><td>9</td><td>3</td><td>6</td><td>2</td><td>1</td><td>5</td><td>7</td><td>8</td></tr>
<tr><td>8</td><td>5</td><td>7</td><td>9</td><td>4</td><td>3</td><td>2</td><td>6</td><td>1</td></tr>
<tr><td>2</td><td>6</td><td>1</td><td>7</td><td>5</td><td>8</td><td>9</td><td>3</td><td>4</td></tr>
<tr><td>9</td><td>3</td><td>4</td><td>1</td><td>6</td><td>2</td><td>7</td><td>8</td><td>5</td></tr>
<tr><td>5</td><td>7</td><td>8</td><td>3</td><td>9</td><td>4</td><td>6</td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>2</td><td>6</td><td>5</td><td>8</td><td>7</td><td>4</td><td>9</td><td>3</td></tr>
<tr><td>3</td><td>4</td><td>9</td><td>2</td><td>1</td><td>6</td><td>8</td><td></td><td>7</td></tr>
</table>

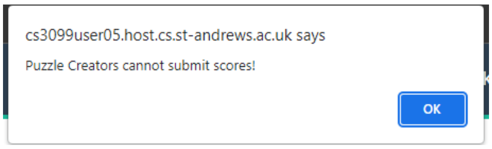
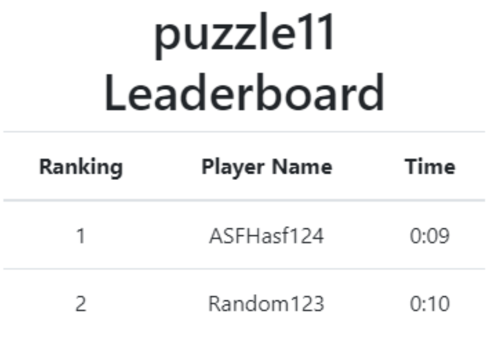
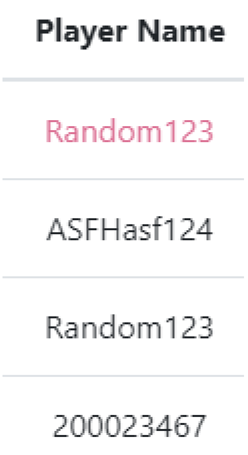
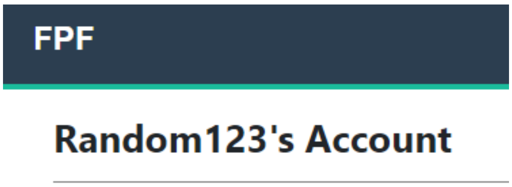
</td>
</tr>
<tr>
<td>Users cannot see their timed solve on completion of the Sudoku.</td>
<td></td>
<td>

<table>
<tr><td>7</td><td>8</td><td>5</td><td>4</td><td>3</td><td>9</td><td></td><td></td><td></td></tr>
<tr><td>6</td><td>1</td><td>2</td><td>8</td><td>7</td><td>5</td><td></td><td></td><td></td></tr>
<tr><td>4</td><td>9</td><td>3</td><td>6</td><td>2</td><td>1</td><td></td><td></td><td></td></tr>
<tr><td>8</td><td>5</td><td>7</td><td>9</td><td>4</td><td>3</td><td></td><td></td><td></td></tr>
<tr><td>2</td><td>6</td><td>1</td><td>7</td><td>Start!</td><td>8</td><td></td><td></td><td></td></tr>
<tr><td>9</td><td>3</td><td>4</td><td>1</td><td>6</td><td>2</td><td>7</td><td>8</td><td>5</td></tr>
<tr><td>5</td><td>7</td><td>8</td><td>3</td><td>9</td><td>4</td><td>6</td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>2</td><td>6</td><td>5</td><td>8</td><td>7</td><td>4</td><td>9</td><td>3</td></tr>
<tr><td>3</td><td>4</td><td>9</td><td>2</td><td>1</td><td>6</td><td>8</td><td>5</td><td>7</td></tr>
</table>
Well Done!
Difficulty Rating:
☆☆☆☆☆ 0
Submit

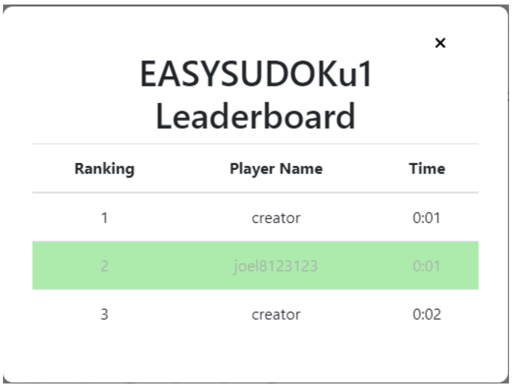
</td>
</tr>
</table>

Users cannot submit times to the leaderboard when resuming a Sudoku.	As expected.

[Figure 18: Resuming Sudokus Testing Table.]

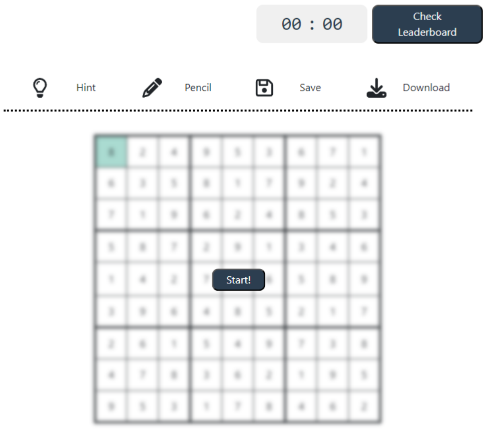
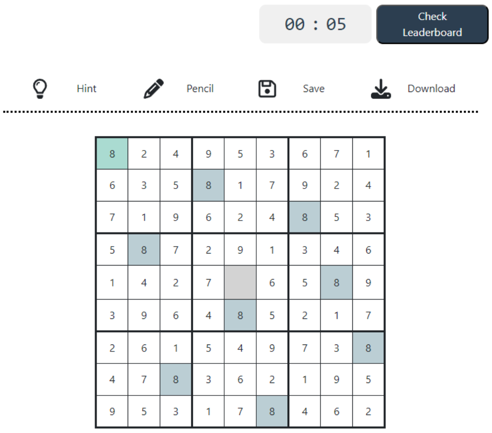
Leaderboard (210003309)

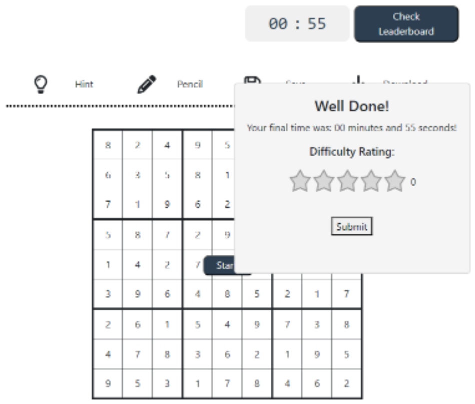
Description	Expected	Actual
Puzzle creators cannot submit times to puzzle's leaderboard	If the creator of the puzzle solves it, their score cannot be added to the leaderboard	 cs3099user05.host.cs.st-andrews.ac.uk says Puzzle Creators cannot submit scores! OK
Players can submit times to puzzle leaderboard	If any user other than the puzzle's creator solves a puzzle, the time it took them it submitted to the leaderboard	 puzzle11 Leaderboard Ranking · Player Name · Time 1 · ASFHasf124 · 0:09 2 · Random123 · 0:10 "Puzzle11" Creator is group2
Clicking player names on leaderboard takes user to their account page	When any user checks the leaderboard and hovers over any player's name, they can see that it is highlighted and links to that user's account page	 Player Name Random123 ASFHasf124 Random123 200023467  FPF Random123's Account

Best Time is highlighted	If the current user has previously solved this puzzle and they are on the leaderboard, their best time on the leaderboard is highlighted in green	 EASYSUDOKu1 Leaderboard Ranking | Player Name | Time 1 | creator | 0:01 2 | joel8123123 | 0:01 3 | creator | 0:02

[Figure 19: Leaderboard Testing Table .]

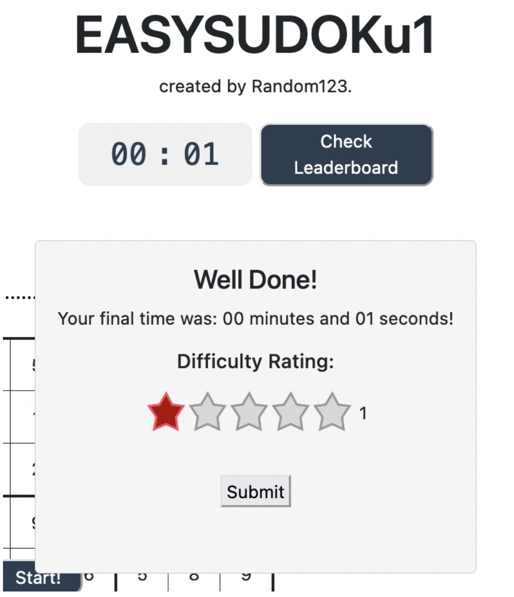
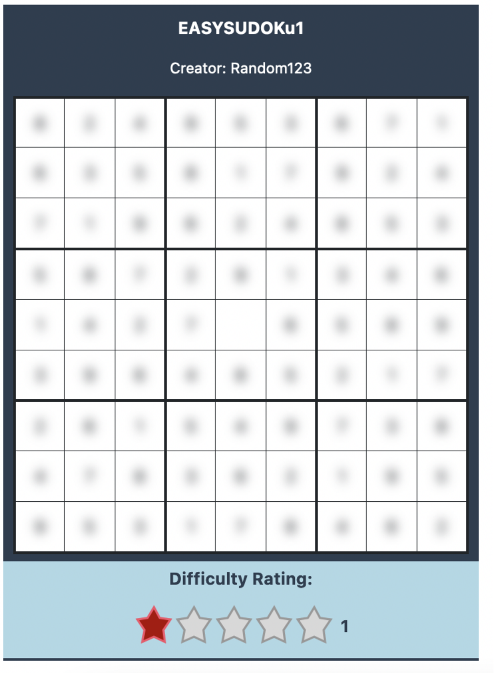
Timer(210003309)

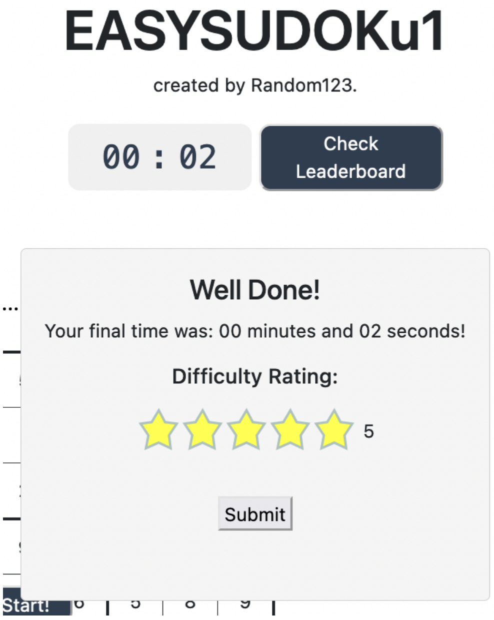
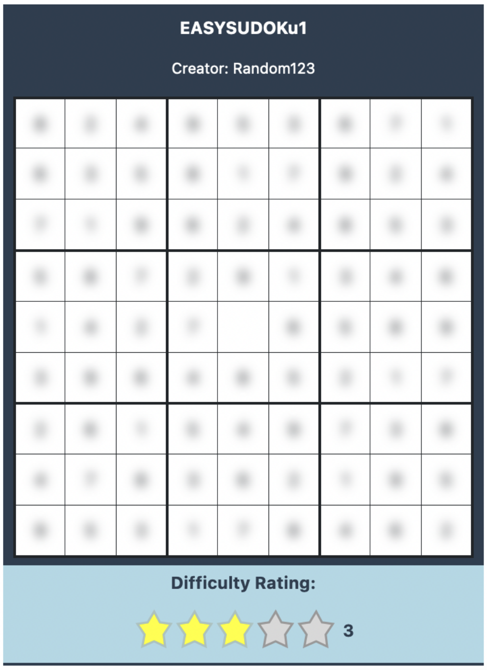
Description	Expected	Actual
Timer Starts at 00:00	On puzzle selection, before starting a puzzle, the timer should start at 00:00	00 : 00 Check Leaderboard Hint Pencil Save Download Start!
Timer updates	On pressing start, the timer should change its value in real-time	00 : 05 8 2 4 9 5 3 6 7 1 6 3 5 8 1 7 9 2 4 7 1 9 6 2 4 8 5 3 5 8 7 2 9 1 3 4 6 1 4 2 7 6 5 8 9 3 9 6 4 8 5 2 1 7 2 6 1 5 4 9 7 3 8 4 7 8 3 6 2 1 9 5 9 5 3 1 7 8 4 6 2

Timer ends	On submission of a solved puzzle, the timer should stop changing	 00 : 55 Check Leaderboard Hint Pencil Star 0 Submit Well Done! Your final time was: 00 minutes and 55 seconds! Difficulty Rating:

[Figure 20: Timer Testing Table .]

Rating (200022418)

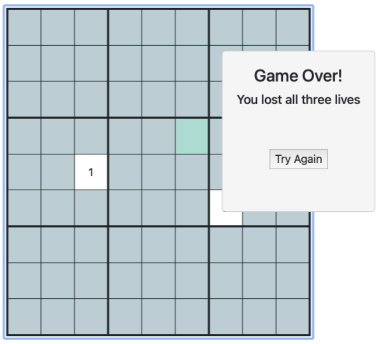
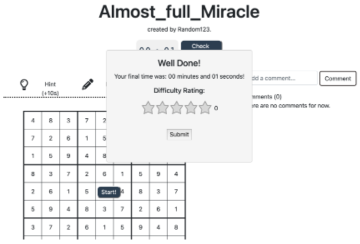
Description	Expected	Actual
Users can rate sudokus based on difficulty	After finishing a sudoku, users are prompted with an alert box that congratulates them, provides them with the time of their solve, and asks them to rate the sudoku out of 5 stars	 **EASYSUDOKu1** created by Random123. 00 : 01 **Well Done!** Your final time was: 00 minutes and 01 seconds! Difficulty Rating: ★☆☆☆☆ 1
The sudoku's rating is updated on the home page	Once a sudoku is rated, the user is redirected to the home page and the difficulty rating under the sudoku is updated	 **EASYSUDOKu1** Creator: Random123 Difficulty Rating: ★☆☆☆☆ 1

The sudoku's rating is updated as the average rating	The sudoku's rating is displayed as the average of all the ratings given by the players	Second rating given to EASYSUDOKu1 (first rating given above):  EASYSUDOKu1 created by Random123. 00 : 02 Well Done! Your final time was: 00 minutes and 02 seconds! Difficulty Rating: 5 Updated rating on the home page (two raters, one of which rated the puzzle a 1 star and the other of which rated the puzzle 5 stars):  EASYSUDOKu1 Creator: Random123 Difficulty Rating: 3

[Figure 21: Rating Testing Table .]

Miracle Sudoku Gameplay (200006767)

The Miracle Sudoku and the normal sudoku use the same Vue component, PlaySudoku.vue, so we are only testing the functionality that is different, which in this case is the one in the table from below:

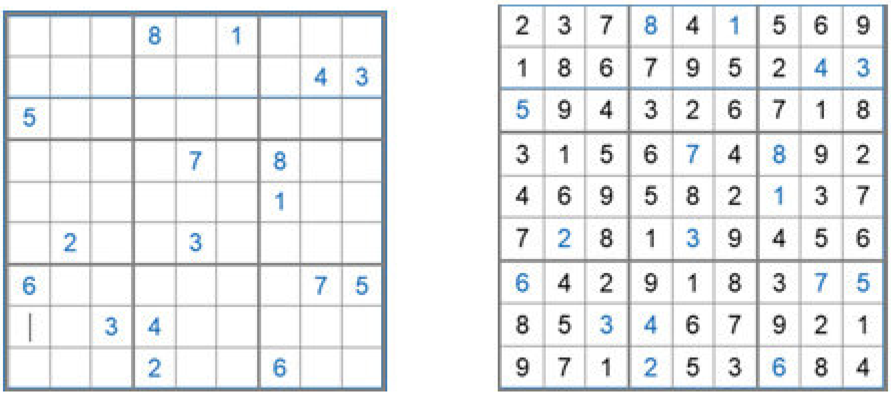
Description	Expected	Actual
Test making 3 mistakes on a miracle sudoku. Each mistake is erased and after 3 mistakes you lose the game	The mistake gets erased and you lose after 3 mistakes	Works as expected  Game Over! You lost all three lives Try Again 1
Test winning a miracle sudoku works	Winning a miracle sudoku displays the well done winning message	Works as expected  Almost_full_Miracle created by Random123. Check Well Done! Your final time was: 00 minutes and 01 seconds! Difficulty Rating: ☆☆☆☆☆ 0 Submit Hint (+10s) dd a comment... Comment mments (0) re are no comments for now. Start!

[Figure 22: Miracle Sudoku Gameplay Testing Table .]

Creating Sudokus (200022530)

esh1_creator (Creator)
Username: esh1_creator Password: Password1

[Figure 23: esh1_creator login details]

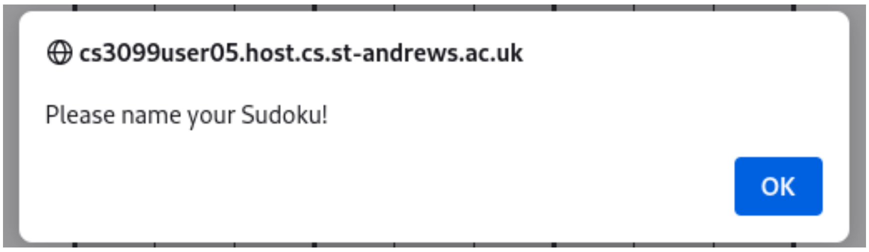
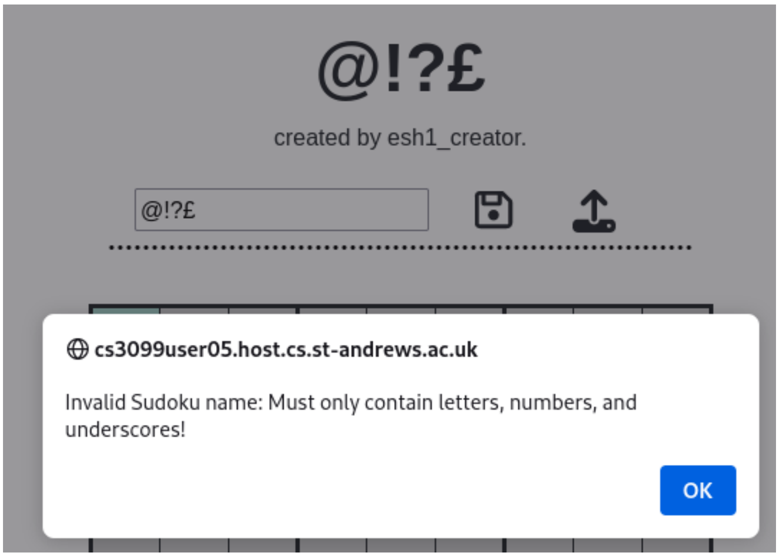


			8		1			
							4	3
5								
				7		8		
						1		
	2			3				
6							7	5
|		3	4					
			2			6		

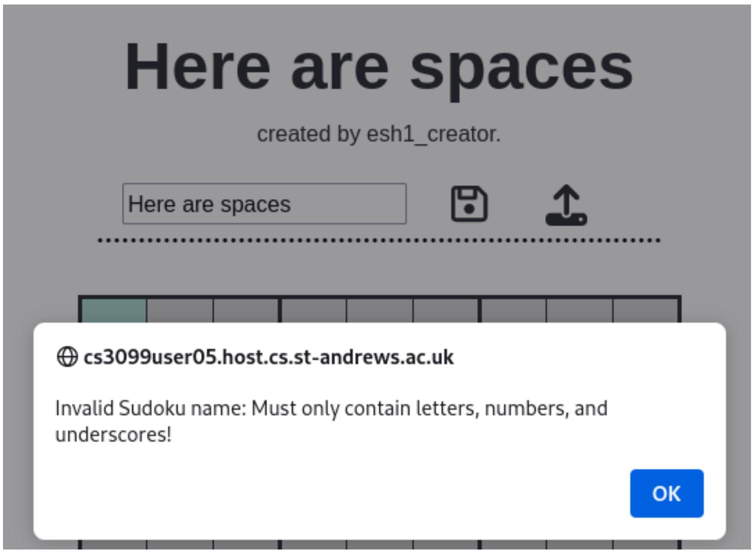
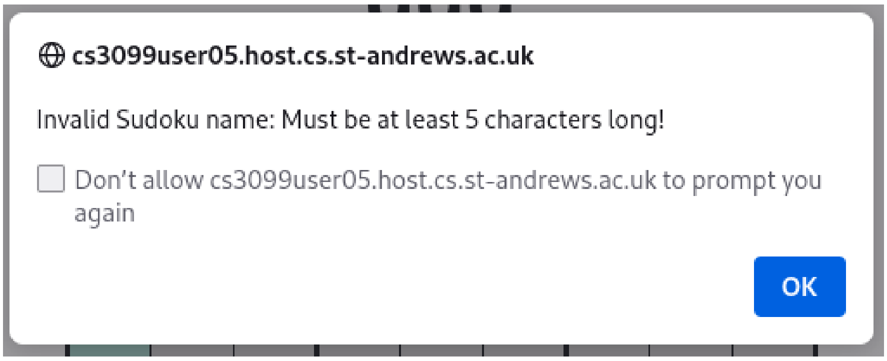
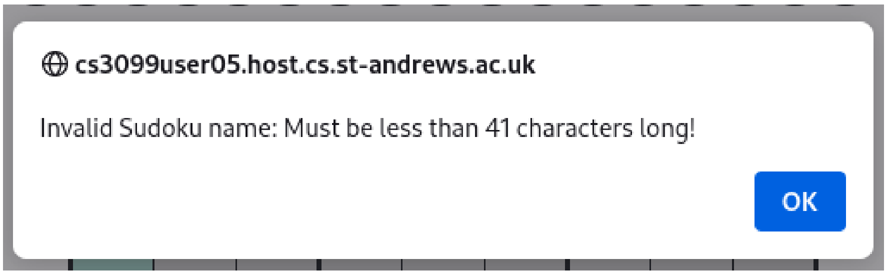
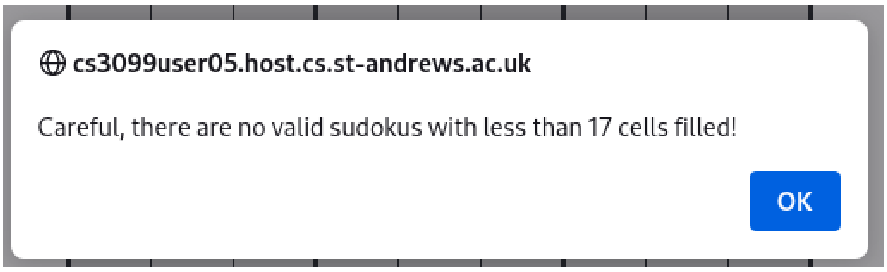
2	3	7	8	4	1	5	6	9
1	8	6	7	9	5	2	4	3
5	9	4	3	2	6	7	1	8
3	1	5	6	7	4	8	9	2
4	6	9	5	8	2	1	3	7
7	2	8	1	3	9	4	5	6
6	4	2	9	1	8	3	7	5
8	5	3	4	6	7	9	2	1
9	7	1	2	5	3	6	8	4

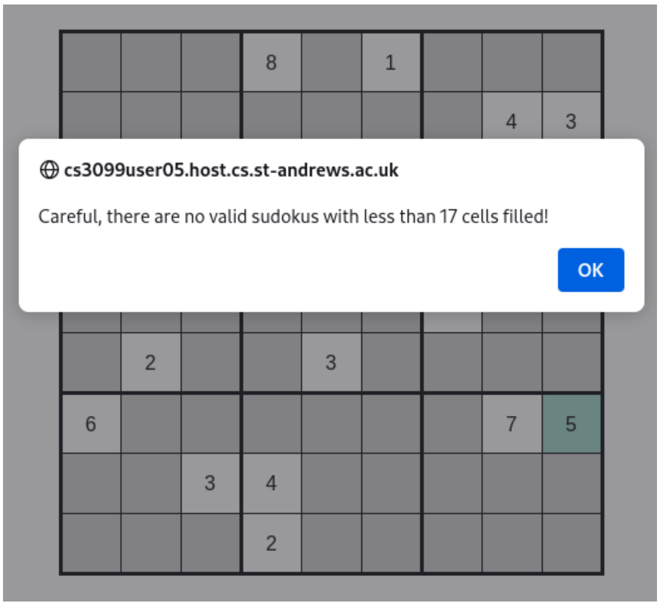
[Figure 24: 17 Cell Sudoku example.[3]]

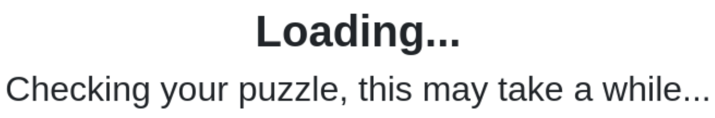
The above login details and 17 cell Sudoku were used for testing our Create Sudoku feature. We have tested edge cases for naming Sudokus, invalid Sudokus, and one difficult to solve Sudoku.

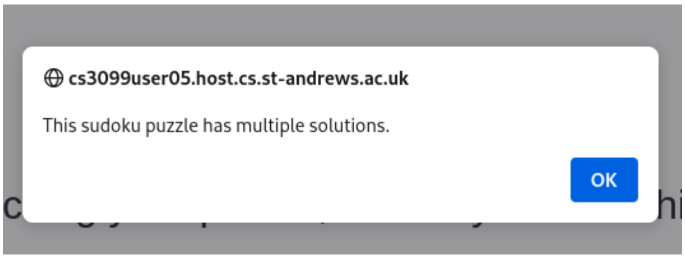
Description	Expected	Actual
Cannot create a Sudoku with no name	 cs3099user05.host.cs.st-andrews.ac.uk Please name your Sudoku! 	
Cannot create a Sudoku with special characters	 @!?£ created by esh1_creator. @!?£ cs3099user05.host.cs.st-andrews.ac.uk Invalid Sudoku name: Must only contain letters, numbers, and underscores! 	

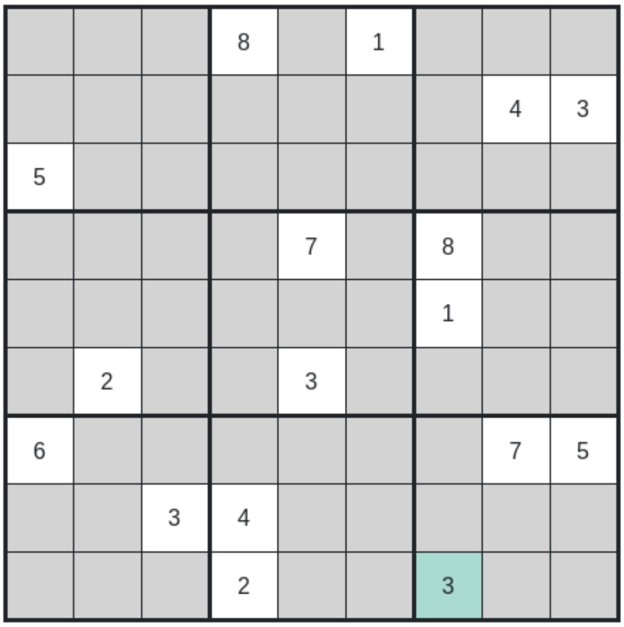
[3] Haradhan Chel. *A novel multistage genetic algorithm approach for solving Sudoku puzzle - Scientific Figure on ResearchGate*. Fig. 1: A sudoku with 17 clues and its unique solution.

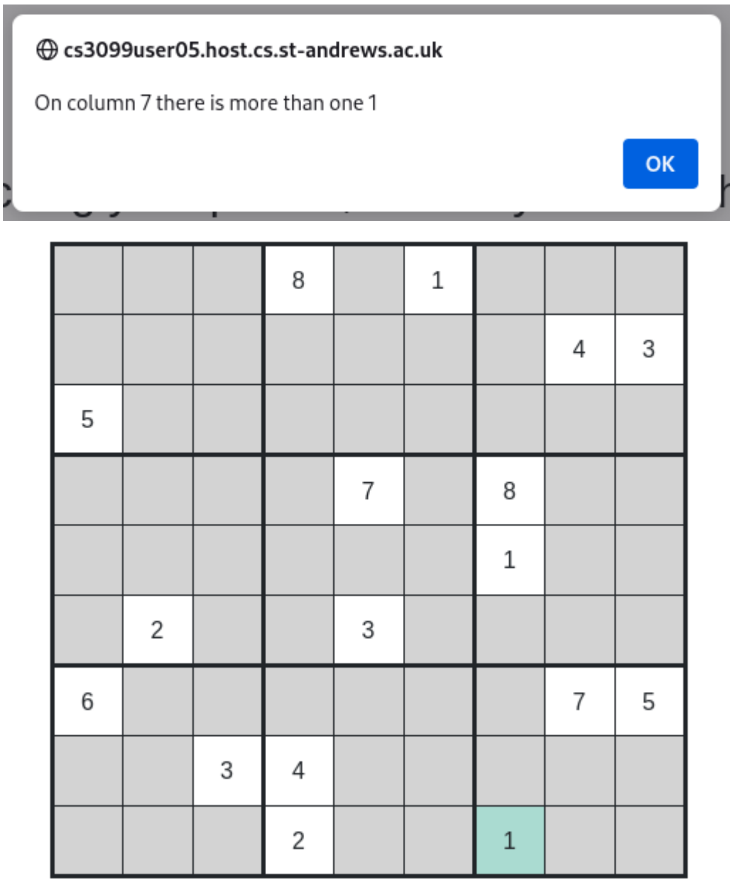
Cannot create a Sudoku with spaces	 # Here are spaces created by esh1_creator. Here are spaces cs3099user05.host.cs.st-andrews.ac.uk Invalid Sudoku name: Must only contain letters, numbers, and underscores!
Cannot create a Sudoku with a name that's less than 5 characters long.	 cs3099user05.host.cs.st-andrews.ac.uk Invalid Sudoku name: Must be at least 5 characters long! ☐ Don't allow cs3099user05.host.cs.st-andrews.ac.uk to prompt you again
Cannot create a Sudoku that's longer than 40 characters.	 cs3099user05.host.cs.st-andrews.ac.uk Invalid Sudoku name: Must be less than 41 characters long!
Rejects empty Sudoku	 cs3099user05.host.cs.st-andrews.ac.uk Careful, there are no valid sudokus with less than 17 cells filled!

<table>
<tr><td>Rejects Sudoku with only 16 cells</td><td>

cs3099user05.host.cs.st-andrews.ac.uk
Careful, there are no valid sudokus with less than 17 cells filled!

</td></tr>
<tr><td>Accepts 17_Cell_Sudoku</td><td>

<b>Loading...</b>
Checking your puzzle, this may take a while...

Submitting a 17 cell Sudoku takes a long time for the solver to check, this is because it is a particularly difficult case. Once the Sudoku has been created the user is redirected to the main page and can find the puzzle they created.
</td></tr>
<tr><td>Rejects Sudoku with multiple solutions</td><td>

cs3099user05.host.cs.st-andrews.ac.uk
This sudoku puzzle has multiple solutions.


<table>
<tr><td></td><td></td><td></td><td>8</td><td></td><td>1</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>4</td><td>3</td></tr>
<tr><td>5</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td>7</td><td></td><td>8</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td></tr>
<tr><td></td><td>2</td><td></td><td></td><td>3</td><td></td><td></td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td>7</td><td>5</td></tr>
<tr><td></td><td></td><td>3</td><td>4</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>2</td><td></td><td></td><td>3</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

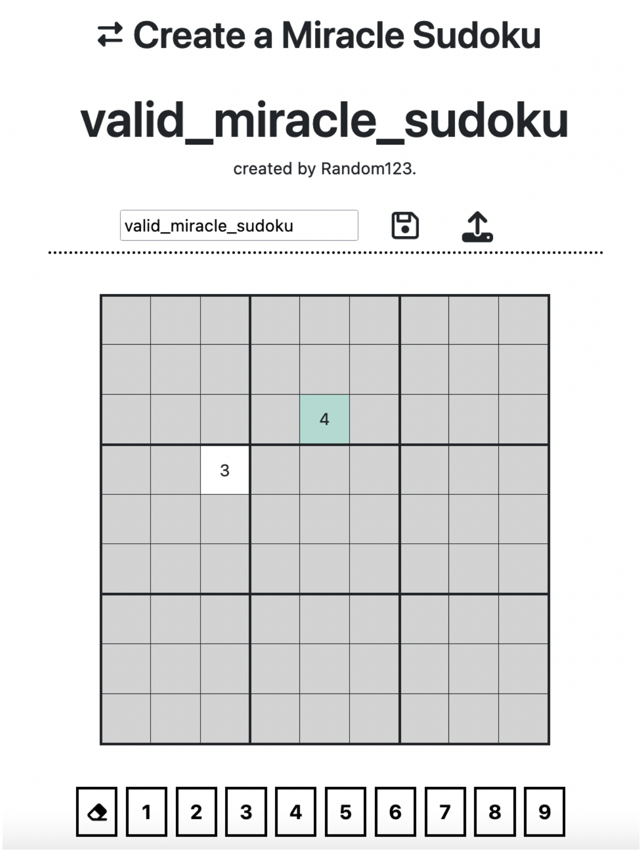
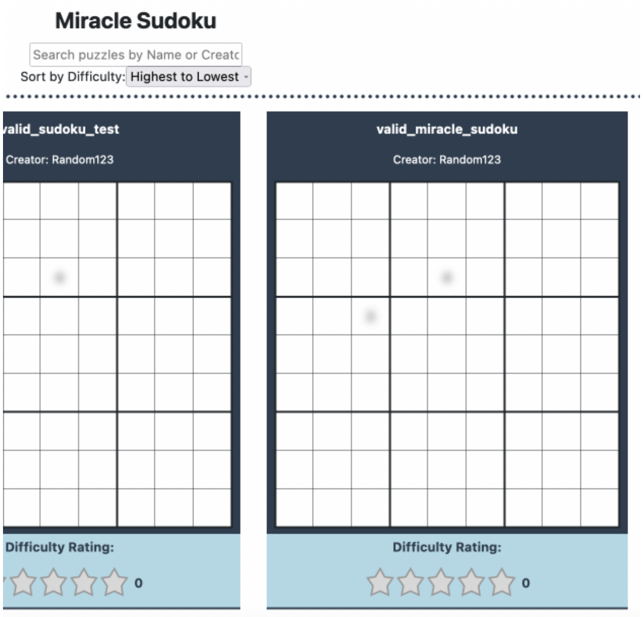
<table>
<tr>
<td>Rejects invalid Sudoku</td>
<td>


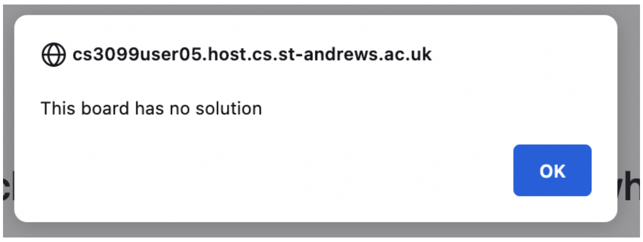
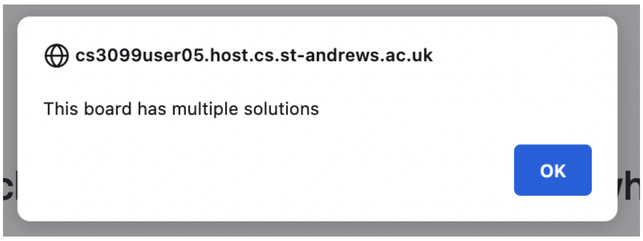
cs3099user05.host.cs.st-andrews.ac.uk

On column 7 there is more than one 1

<table>
<tr><td></td><td></td><td></td><td>8</td><td></td><td>1</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>4</td><td>3</td></tr>
<tr><td>5</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td>7</td><td></td><td>8</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td></tr>
<tr><td></td><td>2</td><td></td><td></td><td>3</td><td></td><td></td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td><td></td><td></td><td>7</td><td>5</td></tr>
<tr><td></td><td></td><td>3</td><td>4</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>2</td><td></td><td></td><td>1</td><td></td><td></td></tr>
</table>
</td>
</tr>
</table>

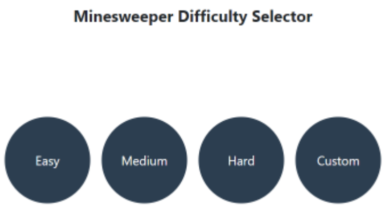
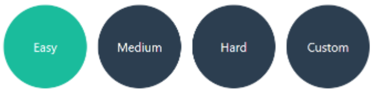
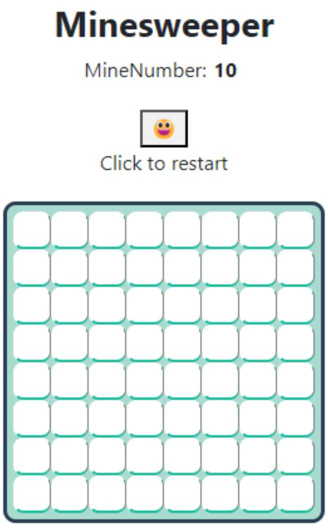
[Figure 25: Table for testing creating Sudokus.]

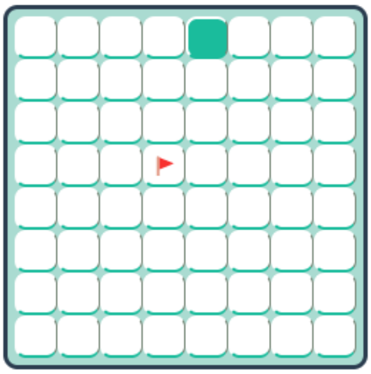
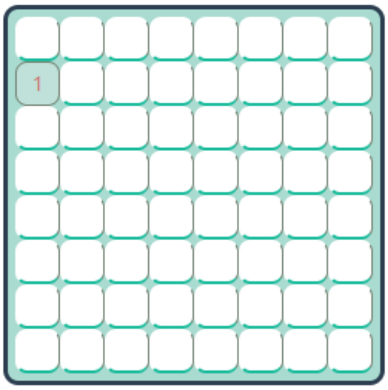
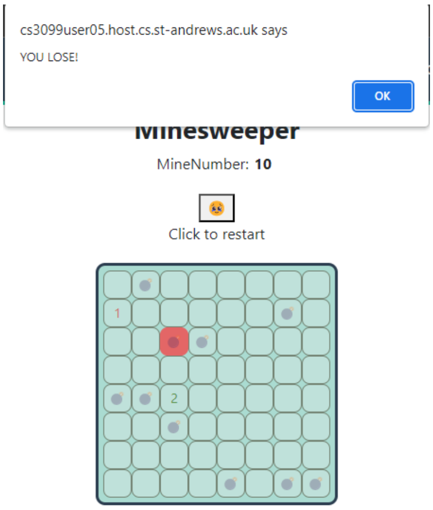
Description	Expected	Actual
Valid Miracle Sudoku	If a user submits a valid miracle sudoku, the user should be redirected to the home page and the miracle sudoku should be added to the miracle sudoku page	Creating a valid miracle sudoku:  ⇄ Create a Miracle Sudoku valid_miracle_sudoku created by Random123. valid_miracle_sudoku Miracle sudoku page display:  Miracle Sudoku Search puzzles by Name or Creatc Sort by Difficulty: Highest to Lowest alid_sudoku_test Creator: Random123 Difficulty Rating: 0 valid_miracle_sudoku Creator: Random123 Difficulty Rating: 0

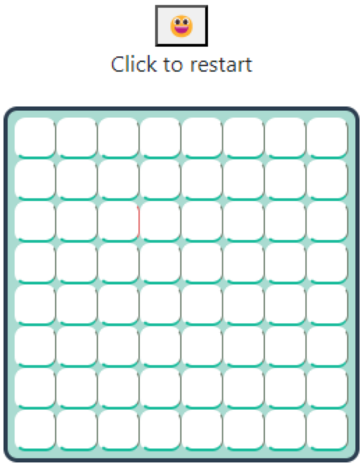
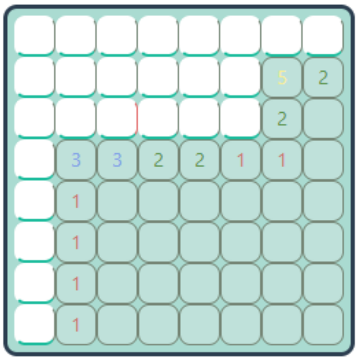
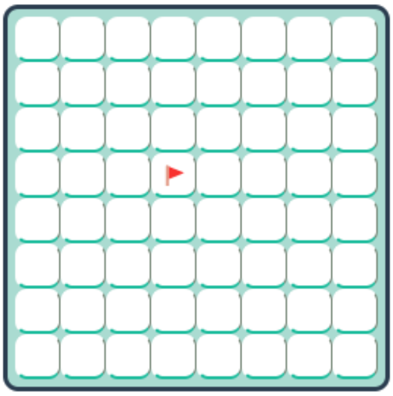
Invalid Miracle Sudoku: no solutions	If a user submits a miracle sudoku that has no solutions, an alert should be displayed and the sudoku should not be submitted	 cs3099user05.host.cs.st-andrews.ac.uk This board has no solution
Invalid Miracle Sudoku: multiple solutions	If a user submits a miracle sudoku that has multiple solutions, an alert should be displayed and the sudoku should not be submitted	 cs3099user05.host.cs.st-andrews.ac.uk This board has multiple solutions

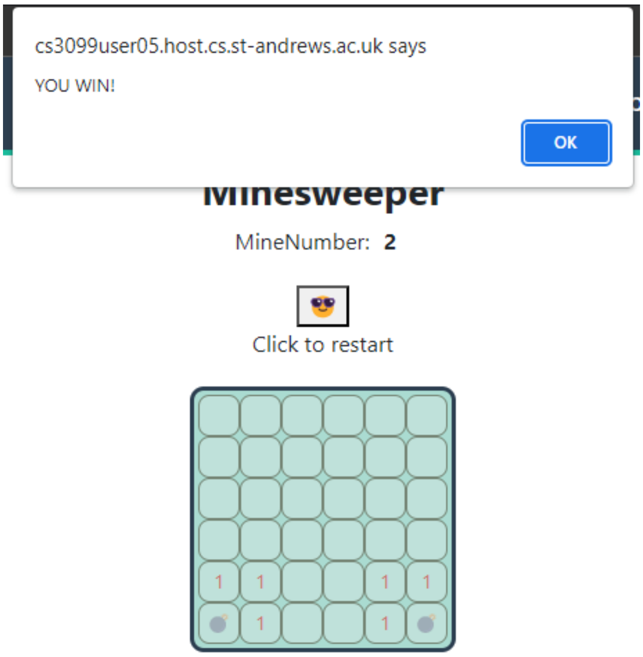
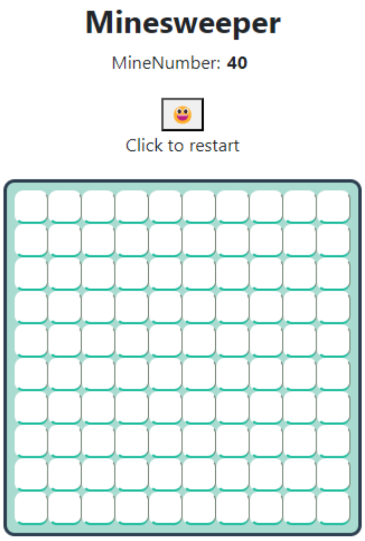
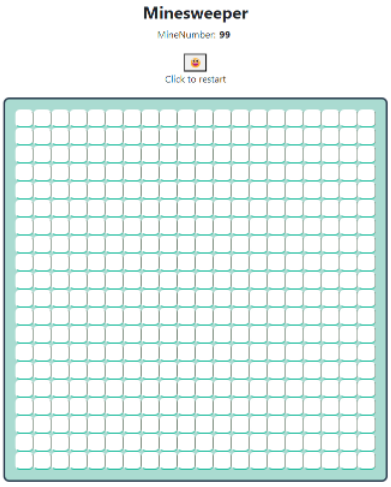
[Figure 26: Creating Miracle Sudoku Testing Table.]

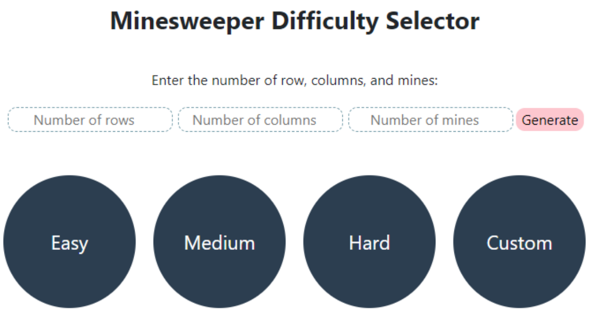
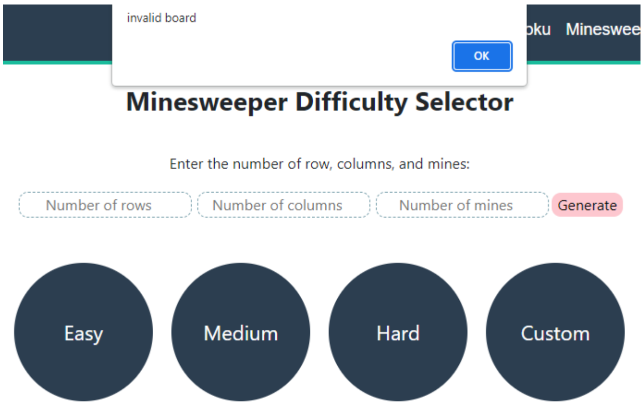
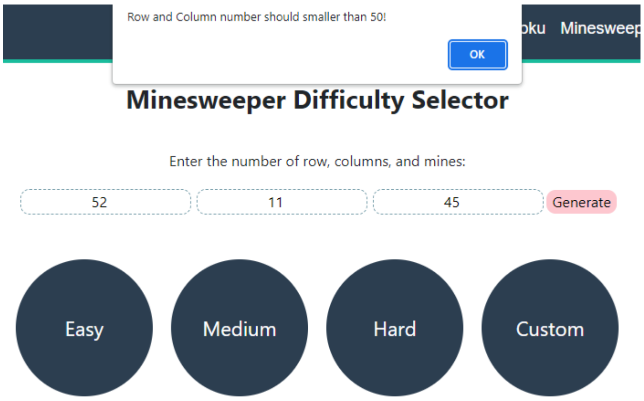
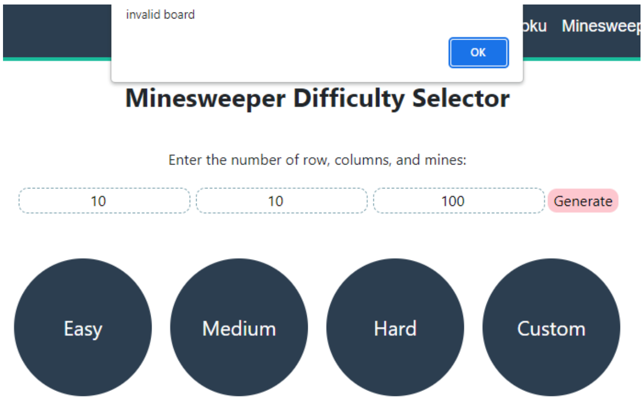
MineSweeper Gameplay (210003309)

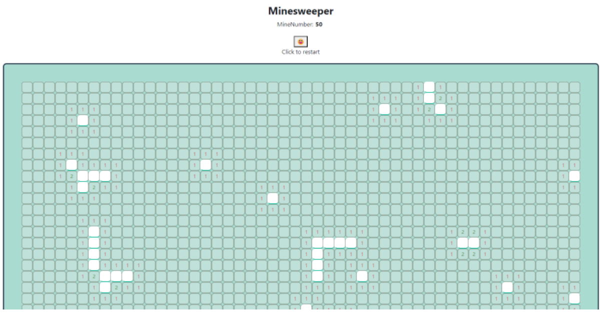
Description	Expected	Actual
MineSweeper main page	On pressing Minesweeper on the navigation bar, the difficulty selector page should show	 Minesweeper Difficulty Selector
Hover highlighting	On the difficulty selector, hovering over an option should change the colour of the option	
Board display	On pressing any difficulty option, a board should appear fitting the appropriate size grid, the number of mines should be displayed as well as restart icon	 Minesweeper MineNumber: **10** Click to restart

Hovering	When hovering over a square, that square should change colour	
Revealing numbers	When pressing non-mine squares, the number of mines surrounding that square should be displayed	
Mine Selection	When a mine is pressed (left-click), the display should change to reflect this and an alert should be shown to the user	 cs3099user05.host.cs.st-andrews.ac.uk says YOU LOSE! Minesweeper MineNumber: **10** Click to restart

Restart	Restarting a grid should create a new minesweeper level	
Empty Square	For squares where there are no surrounding mines, all connected squares that aren't mines should be displayed	
Flagging squares	When right-click is pressed on a square, a flag should appear on that square	

Game Win	When all non-mine squares are revealed, the game should end and this should be made clear to the user	 cs3099user05.host.cs.st-andrews.ac.uk says YOU WIN! OK Minesweeper MineNumber: **2** Click to restart 1 1 1 1 1 1
Medium Difficulty	The medium difficulty mode should have more mines and a bigger grid compared to easy mode	
Hard Difficulty	The hard difficulty mode should have more mines and a bigger grid compared to medium mode	

Custom Difficulty	When “Custom” is pressed, a generator should appear showing three values the user can change	 Minesweeper Difficulty Selector Enter the number of row, columns, and mines: Number of rows Number of columns Number of mines Generate Easy Medium Hard Custom
Invalid Generation (<= 0)	When invalid values are typed into the generator, or none at all, an alert should appear notifying the user. These include no value given for a field (value 0) and 0 or less than it (negative numbers) in a field.	 invalid board Minesweeper Difficulty Selector Enter the number of row, columns, and mines: OK oku Minesweep Number of rows Number of columns Number of mines Generate Easy Medium Hard Custom
Too large column/row values for generator	When values larger than 50 are inputted into row or column size, an alert should appear to the user informing them of the invalidity	 Row and Column number should smaller than 50! Minesweeper Difficulty Selector Enter the number of row, columns, and mines: 52 11 45 OK oku Minesweep Generate Easy Medium Hard Custom
Too many mines	If the number of mines inputted are greater than or equal to the number of squares in the grid, an alert should appear telling the user of its invalidity	 invalid board Minesweeper Difficulty Selector Enter the number of row, columns, and mines: 10 10 100 OK oku Minesweep Generate Easy Medium Hard Custom

Custom Level	If generated correctly, the correct sized board should appear and should function the same as any other level	 This from a max size grid (50 x 50) with 50 mines

[Figure 27: Minesweeper Gameplay Testing Table.]

Backend

Sudoku and Miracle Sudoku Logic (200006767)

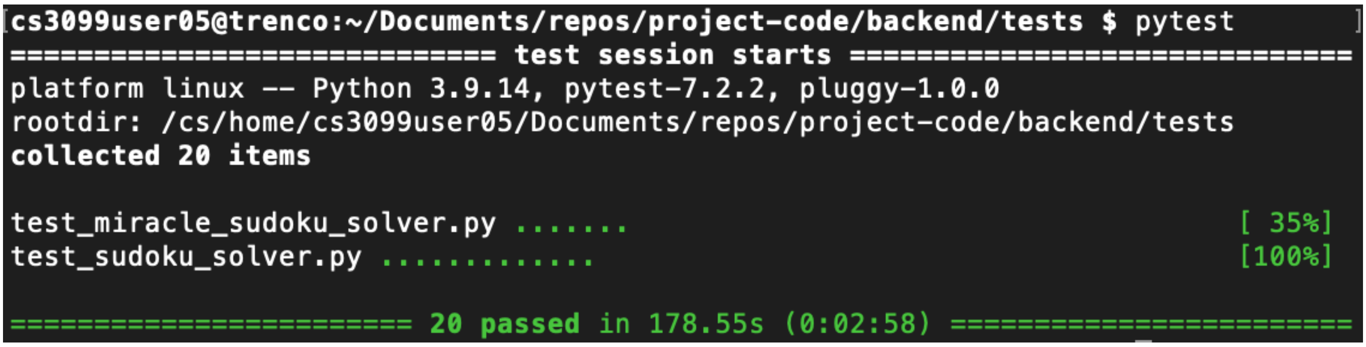
On the backend side we focused on testing the main functionality of the sudoku and miracle sudoku logic. In order to achieve a satisfactory level of testing in addition to the tests made while testing the frontend that also test backend features, we also decided to write a number of pytest Unit Tests for the SudokuSolver and MiracleSudokuSolver classes.

The tests can be found in the “tests” folder of the backend folder.

INSTRUCTIONS FOR RUNNING THE TESTS:
cd backend/tests
pip install pytest

Command for running the tests: pytest

The result of running this command can be seen in the following screenshot:



```
[cs3099user05@trenco:~/Documents/repos/project-code/backend/tests $ pytest                ]
============================= test session starts ==============================
platform linux -- Python 3.9.14, pytest-7.2.2, pluggy-1.0.0
rootdir: /cs/home/cs3099user05/Documents/repos/project-code/backend/tests
collected 20 items

test_miracle_sudoku_solver.py .......                                    [ 35%]
test_sudoku_solver.py .............                                      [100%]

======================== 20 passed in 178.55s (0:02:58) ========================
```

Running the tests can take up to 3-5 minutes because some boards take a long time to validate.

In the test_sudoku_solver.py file, we are testing that boards with non-integer values are invalid, boards with 2 equal numbers on the same row/column/3x3 box are invalid, we are also testing a board with no solution, a board with exactly 2 solutions, testing boards with 17 clues, and also a valid sudoku board. The tests have descriptive names that are self-explanatory. We are also testing the function that checks a partial solution to the sudoku and the full solution to the sudoku (we are testing both valid and invalid partial solutions). The exact position of one mismatch is also used for one of the full solution tests. Test data taken from sources [10], [11], [12] (also cited in the code).

In the test_miracle_sudoku.py file, we are testing that valid miracle sudoku boards can be solved (3 boards tested), we are also testing 2 boards with multiple solutions and 1 board with no solutions. Finally, we also have a complex test that tests the solve_miracle_sudoku() function that generates all valid miracle sudokus. We are testing that 4 known miracle sudokus are generated, and we are also verifying that the number of solutions generated is 72 (the total number of miracle sudokus according to [5]). This code was used when adding miracle sudoku data to the database for the first time. The boards for testing miracle sudoku functionality are from sources [6], [7], [8], [9] (also cited in the code).

Evaluation

Teamwork

SCRUM & Jira (200022530)

Our sprints were two weeks long and every week before our supervisor meetings we would have a SCRUM meeting. During this meeting, we would move issues from our product backlog and incomplete items from the previous sprint into the new sprint. After deciding what tasks we wanted to complete in a sprint, we would assign them to different members of the group. This was all organised using Jira as it provided us with an easy to use interface for communication.

All members of our group had access to Jira, allowing them to assign themselves to tasks from the sprint or product backlog if they were unable to attend a meeting or add any issues they found to the product backlog. Compared to the previous semester where we did all of this over discord and had a lack of mechanisms in place for members of the group who were unable to attend, we saw great improvements in our team dynamics and organisational power - therefore reducing disagreements and making it easier to work as a team.

Sprint	**Weeks**	**Overview**
**Bug fixes for parts were performed throughout for many issues and have been removed from the summaries provided here for improved clarity.*		
1	1&2	● Sudokus are checked during solves, notifying users when they make a mistake. ● Users can ask for a limited number of hints. ● Graphical Sudoku selection screen. ● Semester 1 Feedback response: ○ Improve code quality with comments and documentation ● Commenting functionalities on puzzles ● Timing functionalities on puzzles ● Rating functionalities on puzzles
2	3&4	● When playing Sudokus all numbers that match the one selected are highlighted ● Miracle Sudokus ● Users can view their role and name on their account’s page ● Commenting functionalities on puzzles ● Timing functionalities on puzzles ● Rating functionalities on puzzles
3	5&6	● Timing functionalities on puzzles ● Rating functionalities on puzzles ● Sudoku grids are blurred before starting them ● Miracle Sudokus can be previewed and selected ● Retesting federation logins ● Improving Sudoku CSS (Hints, Pencil, Save, Download) ● Import and export puzzles ● Highlight numbers that match the currently selected cell’s number ● Minesweeper

4	7&8	● Sudoku grids are blurred before starting them ● Miracle Sudokus can be previewed and selected ● More Account page functionality ● Loadscreens ● User permissions ● Leaderboard functionalities on puzzles ● Create report ● Resuming Sudokus ● Bug fixes, styling improvements, and polish
5	9	● User permissions, e.g., deleting comments ● Leaderboard functionalities on puzzles ● Bug fixes, styling improvements, and polish ● Store and display who created a puzzle ● Forgot password functionality ● Report writing ● Username suffixes produced when users are from federation websites ● Sort Sudokus in Sudoku selector ● Search Sudokus in Sudoku selector
Unresolved Backlog Items		● Users can delete their Sudokus and see Sudokus they created on their account page ● A website tutorial ● Users can add friends and see their friend's list

[Figure 28: Table summarising contents of each sprint during Semester 2.]

One of the benefits of using Agile development is that, even though we did not complete our product backlog, we still have a functioning website with the majority of the features we wanted to implement. Agile development has allowed us to get an MVP product produced in Semester 1 with all of our basic functionalities and the highest quality website we could provide in the project time we have been given in this deliverable. As a team, the improvements in how we used Agile development demonstrate the benefits of providing more structure to teamwork. Following our reflections, we have all made improvements.

In addition to using Jira, we also had a personal log that every member would update every two days. The purpose of this log was to replace a traditional scrum stand-up meeting. The benefits of these logs are that we were able to keep in touch with each other and know what progress has been made. Knowing our progress on a regular bases allowed us to understand our pace of completing tasks and allowed us to create realistic sprint cycles in terms of workload.

Supergroup Interactions (200022530)

We delegated one group member to attend the supergroup meetings every week, except for some instances where they were unable to make it and another member took their place. Our group representative took part in each meeting by contributing to the conversation with suggested solutions from our team, for example when deciding puzzle formats, and relaying information back to the group. In the last remaining weeks, supergroup meetings were replaced with meetups for testing federation compatibility which were attended by the relevant group member and has been discussed at greater length in the testing section of this report. As mentioned in our previous reports, the supergroup ensured adequate communication by the organisation of meeting minutes and the GitHub Wiki.

Limitations and Potential Improvements (200022418)

- Limitation: The current list of sudokus in the sudokus and miracle sudokus pages loads all puzzles from the database at once. This could result in the website slowing down or lagging for a large database
    - Improvement: implement pagination or lazy loading to load only a certain number of Sudokus at a time, improving the performance of the selector.
- Limitation: Difficulty rating is subjective - users of different skill levels will assess the difficulty of a puzzle differently
    - Improvement: implement an algorithm in the backend that immediately assesses the difficulty of a sudoku by solving the puzzle in a human-like manner. This could involve creating a set of rules that mimic the way humans solve puzzles, and then using these rules to generate a difficulty rating for each puzzle.
- Limitation: The website doesn't respond well when the window is resized, which could affect the user experience
    - Improvement: implement all major CSS features using viewport width to allow all features to scale with the size of the window.
- Limitation: Lack of support for Miracle Sudokus
    - Improvement: implement a rating system for miracle sudokus.
- Limitation: Lack of support for Minesweeper
    - Improvement: add similar support for traditional sudokus for minesweeper. For example, a rating system, timers, and leaderboards.
- Limitation: Duplicate usernames with different capitalisations act as different users completely
    - Improvement: modify the backend code to store both the original input and the lowercase version of the username in the user database. When a new user is registering a username, convert the username to lowercase and compare it against the lowercase versions of all usernames in the database.
- Limitation: If a user does not rate a puzzle, the rating stored is 0
    - Improvement: do not send a rating to the backend if a user has neglected to submit a rating.
- Limitation: Using local storage to store user session data. This is bad practice for security because local storage is accessible by JavaScript, which means that attackers can access and manipulate the data by running scripts. This could lead to session hijacking and other security issues.
    - Improvement: using browser cookies for user sessions. This is better for security reasons because cookies expire after a certain amount of time. Further, cookies can be made to be accessible over only HTTP(S), preventing any JavaScript attacks.
- (200022530) Improvement: Add more features to user’s account pages, for example a friends list, viewing their ranked solves, a list of the Sudokus they created.
- (200022530) Improvement: We could’ve made better use of Vue components to separate different functionalities up better, for example many features inside PlaySudoku could have their own components, such as when playing Miracle Sudoku and the Timer. It would’ve then been easier to use these components again elsewhere, similarly to LoadScreen.vue. Additionally, the rating system was implemented for traditional sudokus, however, since miracle sudokus share the same component, it looks like the rating feature is supported for miracle sudokus as well.
- (200022530) Improvement: Fixed any potential security issues, for example when using post requests to promote users to admin.

Conclusion (200022530)

In conclusion, our website meets all required functionalities with additional functionalities, including puzzle types aside from Sudoku (such as, Miracle Sudoku and Minesweeper) and extra social features for users to get further use out of our website. In combination our features will allow for a more competitive environment for solving Sudokus with leaderboards, timers, and saving methods (for practice) as was outlined as a goal in our first deliverable. The different control options (keyboard and mouse support) will make our website easier to interact with - making timed solves feel more natural.

There are some things that given more time we would like to have added, changed, or completed - as evidenced in the evaluation section of this report. Most notably, it would have been nice for us to have tested our website's security at a greater depth and implement any required fixes. However, overall we are pleased with the functionality of our website.

References

[1] Aces Supergroup. *Aces Supergroup Functionality Testing*. https://docs.google.com/spreadsheets/d/1IC9cChPLV60ARqWsPBPF_Q4Ngs6LOKX4ArvTUH0HMTA/edit#gid=0. Date Accessed: 24/3/2023.

[2] Haradhan Chel. *A novel multistage genetic algorithm approach for solving Sudoku puzzle - Scientific Figure on ResearchGate*. Fig. 1: A sudoku with 17 clues and its unique solution. https://www.researchgate.net/figure/A-Sudoku-with-17-clues-and-its-unique-solution_fig1_311250094. Date Accessed: 24/3/2023.

[3] Aces Supergroup. *aces-documentation*. https://github.com/DMJamboe/aces-documentation/wiki. Date Accessed: 24/3/2023.

[4] Archived from Csse.uwa.edu.au by web.archive.org. *Minimum Sudoku.* https://web.archive.org/web/20061126162713/http://www.csse.uwa.edu.au/~gordon/sudokumin.php. Date Accessed: 24/3/2023.

[5] https://ethmcc.github.io/miracle-sudoku/
Date Accessed: 24/3/2023

[6] https://www.popularmechanics.com/science/a32605317/miracle-sudoku-hardest-puzzle-ever/
Date Accessed: 24/3/2023

[7] https://www.youtube.com/watch?v=Tv-48b-KuxI/
Date Accessed: 24/3/2023

[8] https://logic-masters.de/Raetselportal/Raetsel/zeigen.php?chlang=en&id=0004E0/
Date Accessed: 24/3/2023

[9] https://www.reddit.com/r/sudoku/comments/luqsev/miracle_sudoku_solved/?sort=top/
Date Accessed 24/3/2023

[10] https://leetcode.com/problems/sudoku-solver/description/, (example testcases)
Date Accessed 24/3/2023

[11]https://www.reddit.com/r/sudoku/comments/7q76ay/friend_tells_me_that_this_is_unsolvable_sudoku/
Date Accessed 24/3/2023

[12] https://www.quora.com/Does-a-sudoku-have-multiple-solutions/
Date Accessed 24/3/2023

[13] https://www.svgrepo.com/svg/80156/down-arrow?edit=true