General functionality

The project is divided into two different programs.

User Interface

The functionality of the user interface can be explained with the help of 5 scenes:

1. Logo animation, 2. home scene, 3. sudoku scene, 4. scores scene, and 5. rules scene.

When starting the program, one first encounters the first scene: a short logo animation (1).

Thereafter, the home scene (2) is displayed with the date, our logo, and the option of pressing three buttons and thereby navigation to the remaining three scenes:

- play button(3): navigates to the daily sudoku under the condition it has not already been completed. We have implemented the sudoku such that it will either be displayed traditionally with numbers or with images.

- scores button(4): navigates either to a message that directs the user to play the sudoku today or under the condition it was already played, displays the top ten players with name and time (in seconds) and a graph with the minimum, maximum, average and player times (in seconds).

- rules button(5): here we have provided a short run down of the general sudoku rules and our concept.

If a user starts playing, the app downloads the sudoku for today from the server which is uniquely identified by today's date. Thereby, the sudoku can have different game modes (classic or diagonal) and different difficulties (easy, medium, hard). Game mode and difficulty are randomly chosen when the sudoku is uploaded to the server.

When the user solves the sudoku and enters his name, the needed time and the name are uploaded to the server and stored together with the current date.

Sudoku Uploader

The second program generates sudokus and uploads them to the server. Therefore, an admin passes a number n and a starting date to the program. The number n determines how many sudokus should be uploaded and the starting date defines from which date on the sudokus will be created. For instance, n=5 in combination with January 1st 2023 would create sudokus from January 1st to January 5th 2023.

Generating a sudoku itself works like the following. First, the most left column and the middle row of an empty sudoku grid are filled with random numbers according to the game mode rules. This method guarantees that the prefilled sudoku actually has a solution. After that, the prefilled sudoku is solved using a backtracking algorithm. Finally, an amount of numbers is deleted from the solution according to the difficulty (easy: 37, medium: 45, hard: 53). The amount of deleted numbers for hard sudokus was chosen by trial and error so that the program will not run into infinite loops in the

following steps. The procedure of deleting numbers works like the following. First, a random cell is chosen, and the corresponding number is deleted, Now the program checks if the resulting sudoku has more than one solution. This is done via a modification of the backtracking algorithm that does not stop after the first solution but rather if two solutions are found. Furthermore, the program checks if not more than one type of integer (from 1 to 9) has zero occurrences. Otherwise, this would also lead to an insolvability of the sudoku. If the resulting sudoku cannot be solved clearly, the selected number will not be deleted, and a different cell is chosen.

Server (general information)

The server that we use is a graph related database namely Neo4j. Neo4j is a graph-based database that efficiently represents complex relationships between different nodes, allowing for efficient work with the nodes. Neo4j uses the Cypher language to query data points from the database. It is used, for example, for social networks or to detect fraud.

Important note: If the free-tier database is not used for a few days, it will automatically switch to standby mode. In this case, you simply need to log in as follows and activate the DB with a click.

To use the user interface of Neo4j.com and view the data points of our Sudoku project, follow these steps:

1. Go to Neo4j.com and log in using the following credentials: sudokuprojektsd@gmail.com + SudokuProjekt1.
2. Once logged in, you will arrive at the dashboard. Go to “Instances” under “AuraDB”. This is Neo4j’s cloud database. For testing purposes, Neo4j provides a free-tier.
3. To access the database, click on “Open” on the “Instance01”.
4. To enter the database, you need to enter the following: password: lj6AD4ac4CFidRzRqtkeIzfKabcaYppX6lfIPzhZzkQ, User: neo4j
5. Now you can view all nodes, which in this case are “Playerresults” and the daily Sudokus.

Note: As the central idea of our program is to have one identical sudoku for all users every day, we have incorporated a dynamic date variable in our code. We have prepared four example days to be viewed. If the program is not run on the 19.06.2023, 20.06.2023, 21.06.2023, 22.06.2023 or it you wish to try out the different days, please change this.todayDate to either “20230619”, “20230620”, “20230621” or “20230622” right after this.todayDate = date.format(formatter); (after line 691).

Furthermore, we will have created Sudokus for the dates listed above. So, in theory there is no need to run the SudokuUploader if this.todayDate is set to a corresponding value. Of course, the program can be run anyway to test the functions. Besides, we created fake player results for these dates. Therefore, the score page will display several results to demonstrate the full functionality.