CP476 Project: Sudoku solver web visualization and representation

Members: Carla Castaneda, Keven iskander, Nicole Laslavic

Date: 2021-03-22, Revised: 2021-04-10

Introduction

We created a dynamic, web application that visualizes a sudoku puzzle solving algorithm. The unsolved sudoku puzzle is displayed on an html front-end with css styling, then processed through a POST and GET request to a python script that uses artificial intelligence to solve the puzzle and POST the returned puzzle to the user interface. The python algorithm was written by our group members for CP468 – Artificial Intelligence, however, it was re-written slightly to generate random puzzles and accept a list of integer values representing unsolved and solved puzzles. All the group members involved in this project were also authors of the Sudoku puzzle solving python script written for CP486. We used both AJAX and Flask framework to make HTTP requests on the server side. Users are able to also interact with the sudoku puzzle in the form of an HTML table, where each element represents a digit within the puzzle. Another feature we added was a timer to collect data on how fast our users are finishing the puzzles. We also store usernames and their sudoku puzzle completion times in our MongoDB database. On our web application the top five most recent entrees are displayed on the top right of the application.

Problem solving and algorithms

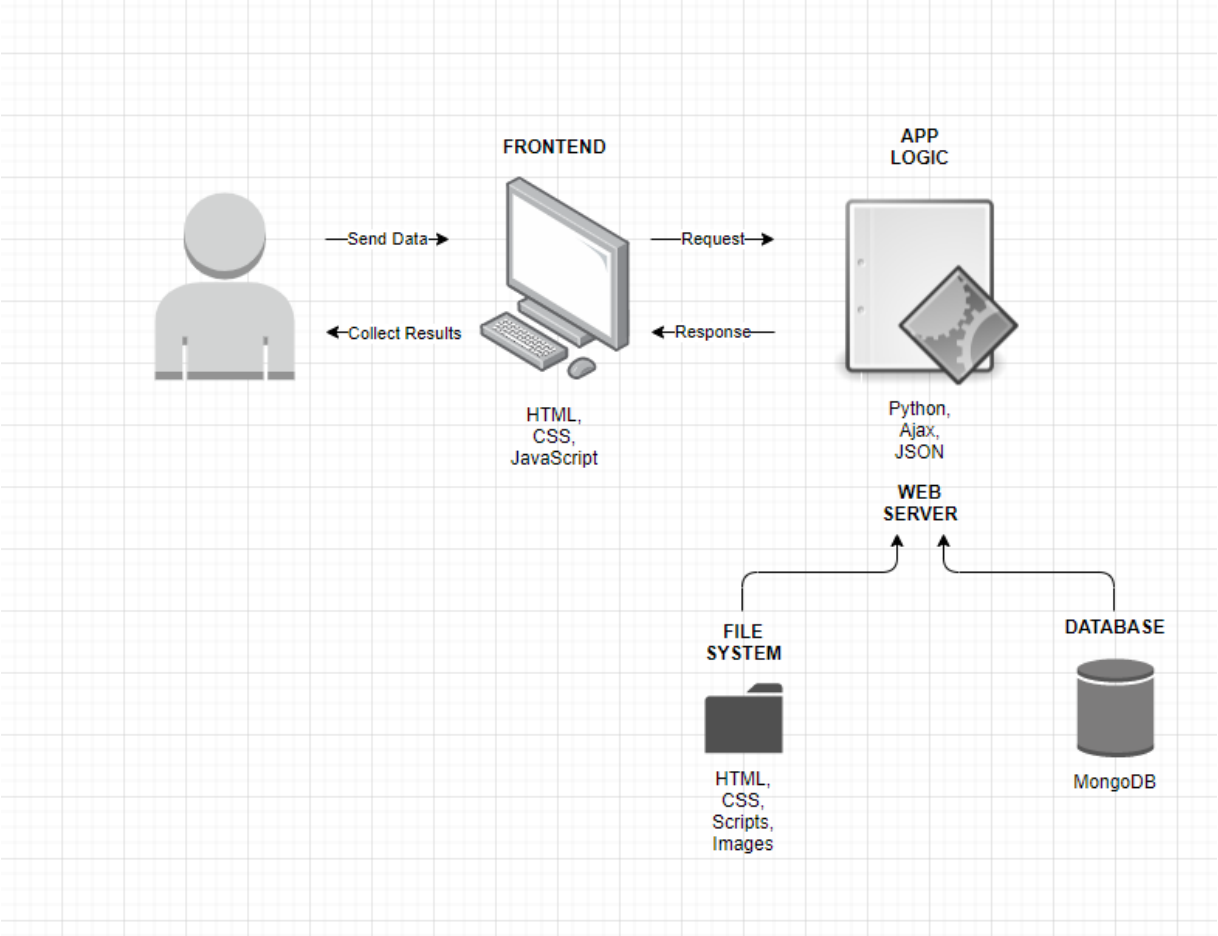
On the client side, we have an HTML table for users to input their values. Styling and formatting will also be implemented on the front-end user interface with CSS stylesheets. Javascript animations will be included to highlight squares red to communicate incorrect moves as well as a timer. From the server side, puzzles are stored and accessed in JSON format from the HTML form added to arrays within the python script. The most recent five players will have their names and completion times displayed top left of the application window.

The sudoku puzzle solving script was implemented using a combination of two well-documented artificial intelligence algorithms, the AC-3 algorithm, and the backtracking algorithm. The AC-3 process involves organizing constraints and evaluating the next best digit to insert without violating the rules of the puzzle. Unfortunately, AC-3 alone cannot always solve the puzzle to completion. That is where backtracking plays its role, by iterating backwards through the puzzle to try and re-insert values to solve the remainder of the puzzle. More information can be found in our repository LINK .

System Design

We used tools such as XAMPP and flask to locally view our website before deployment. For the database portion we used MongoDB to connect to a database that tracks the users and their times. The CGI programming tool we will be working with is AJAX. We also used an express API to make calls between the front end and the MongoDB database. To deploy we used Heroku which is a cloud application platform by linking our project repository.

Architectural Diagram



Milestones & schedule

Task ID	Description	Due date	Lead
1	Project research & team up	Day 5 of week 9	Everyone
2	Project proposal	Day 1 of week 10	Everyone
3	Start working on front end	Day 3 of week 10	Carla
4	Finish front end	Day 5 of week 10	Carla

5	Work on incorporating database components to front end	Day 1 of week 11	Keven
6	Start working on incorporating python script containing the AI sudoku code to front end	Day 3 of week 11	Nicole
7	Finish last meetings goal	Day 5 of week 11	Nicole
8	Clean up project and get ready for project demonstration	Day 1 of week 12	Keven
9	Project demonstration	Day 5 of week 12	Everyone
10	Project submission	Day 5 of week 13	Everyone

Future Goals

In the future we would want to incorporate two different databases, one for user times and one for computer times. Within the web application instead of the most recent five times we would like to display the top five times.

We would also like to implement the feature to restrict users from changing values after the sudoku script solved the puzzle to prevent users from cheating on their times and tricking the code.

References

Healey, Andrew. “Talking to Python from JavaScript (and Back Again!).” *Healeycodescom Blog RSS*, 11 Apr. 2019, healeycodes.com/javascript/python/beginners/webdev/2019/04/11/talking-between-languages.html.

Lee. “How to Code a Sudoku Solver in Python and Flask.” *Coding Projects Hub*, 23 Jan. 2021, codingprojectshub.com/how-to-code-a-sudoku-solver-in-python-and-flask/.

Laslavic, Iskander, Castaneda, Francis. 2020. CP468_A2. https://github.com/keveniskander/CP468_A2. (2021).