Threshold Correction: Maze Solver

Ben, Ethan, Kyle, Surya, Tommy

Tools

Discord: Primary Communication Tool: ★★★★★

Jira: Project Dashboard ★★★☆☆

Github: Code Management ★★★★☆

PostgreSQL: Database ★★★★☆

Heroku: Deployment environment ★★★★☆

Eclipse: IDE ★★★★☆

A*: Maze Solving algorithm ★★★★☆

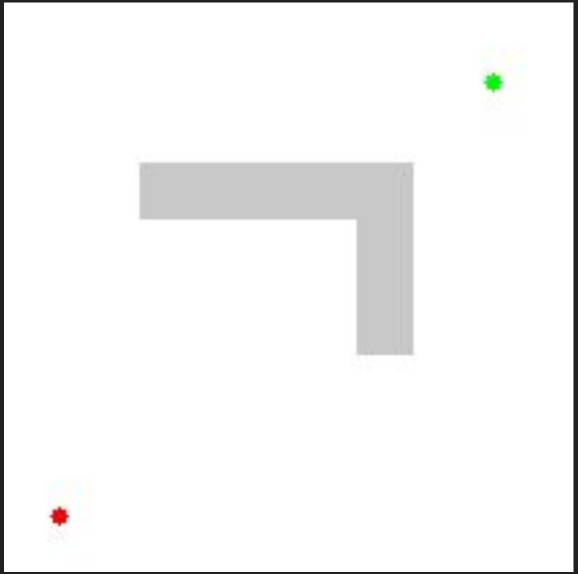
chart.js: Graphing Library ★★★★☆

A* Algorithm

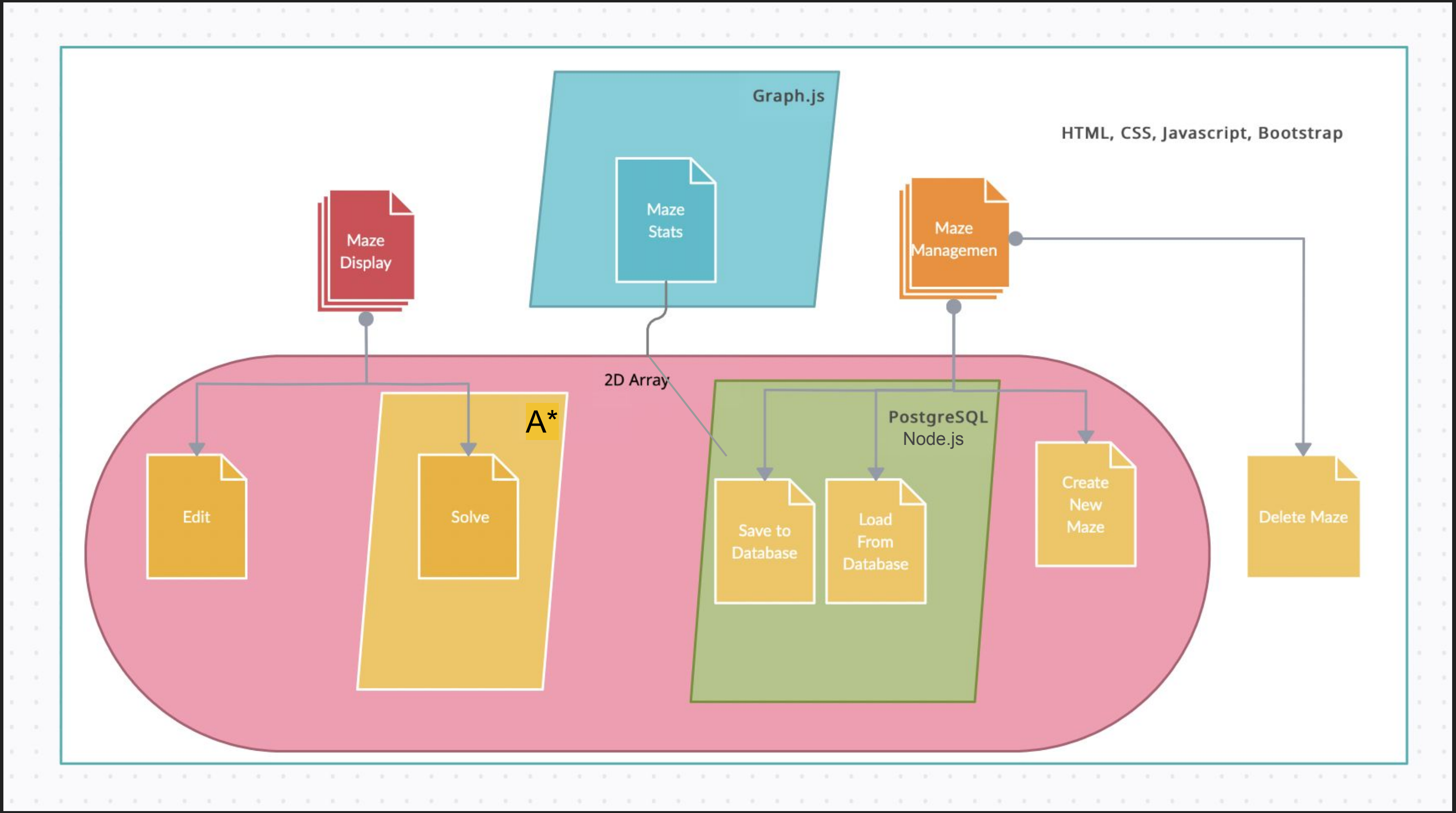
- Informed Search Algorithm
- Extension of Dijkstra's algorithm
- Formulated in terms of weighted graphs
- Starts from specific node and finds path to specified end node with smallest cost

- Where n is the next node, *g*(*n*) is the cost of the path from the start node to *n*, and *h*(*n*) is a heuristic function that estimates the cost of the cheapest path from *n* to the goal

$$f(n) = g(n) + h(n)$$



Architecture Diagram



Challenges

One challenge that we faced early on in the project was finding a viable way to perform the Machine learning on the maze solving. Eventually we ended up going with a Maze solving algorithm rather than using reinforcement learning to solve the maze as it would be much more efficient and easier to implement. It also meant that we could implement the algorithm in JS without needing to figure out how to connect our HTML to a python script.

A second challenge we faced was finding an effective way to save the maze information. Portions of the algorithm and front end had to be rewritten in order to save both the algorithm and solution across multiple html pages as well as being able to send that layout and solution data to the database. The solution was to use local session storage to save the maze layout and solution within the browser itself and enable the user to modify everything within the browser before saving to the database.

The last and biggest challenge that we faced was setting up node.js to tie the front and back end together effectively. While were able to get an effective front end built and working, and were able to create, test, and run a working database, our largest hurdle proved to be passing data back and forth between the front and back end via node.js.

Demo Time