IT'S LMN🌳ree, MY DEAR WATSON

Sarah Yoon, Max Bertfield, Brian Lu, Zicheng Zhen

Overview

Our site takes a spreadsheet of classes and their prerequisites and provides a dynamic, interactive visualization of the data as a directed graph. The user is presented with a graph representing all the possible courses and their prerequisites. At this point, the user can select classes they want to take at some point. With these preferences, their prerequisites, and the school's graduation requirements taken into account, a suggested path is generated. These selected courses can then be visualized on another page as a tree, with each level representing a semester/trimester.

Basic Features

- Take hardcoded data about classes at Stuyvesant and certain universities and represent them intuitively and dynamically in a directed graph
- Allow user to input data for available courses and prerequisites
- Create paths in directed graph through the department's courses that cover both desired classes and graduation requirements
- Allows user to select their own path in a tree of the department's courses, revealing each possible next course along the way

Tentative Features

- Ability to generate a graph based on course/prereq data that the user inputs
- Ability for user to input graduation requirements

Database Structure

We will use SQLite3 to store data. Our database will consist of ____ tables. One table will store a list of the generated trees and graphs and assign each an ID. Another table will store connections between different classes, grouping them under the ID assigned in the other table.

- TABLE **graphlist**
    - INT **id** - ID of the created graph
    - TEXT **name** - Name of the created graph
    - INT **classcount** - Number of courses in the created graph

- TABLE **connections**
    - INT **id** - ID of the graph a connection belongs to
    - INT **courseid** - ID of the course in the range [0, classcount)
    - TEXT **coursename** - Name of the course
    - TEXT **connections** - String representing a list of connections, stored in a String to be parsed (e.g. “1,3,4” would represent a connection from the current course to courses with IDs 1, 3 and 4)

Python and JS Modules

TreeGenerator.py
- Genereates a tree from a given csv representing courses and their prereqs

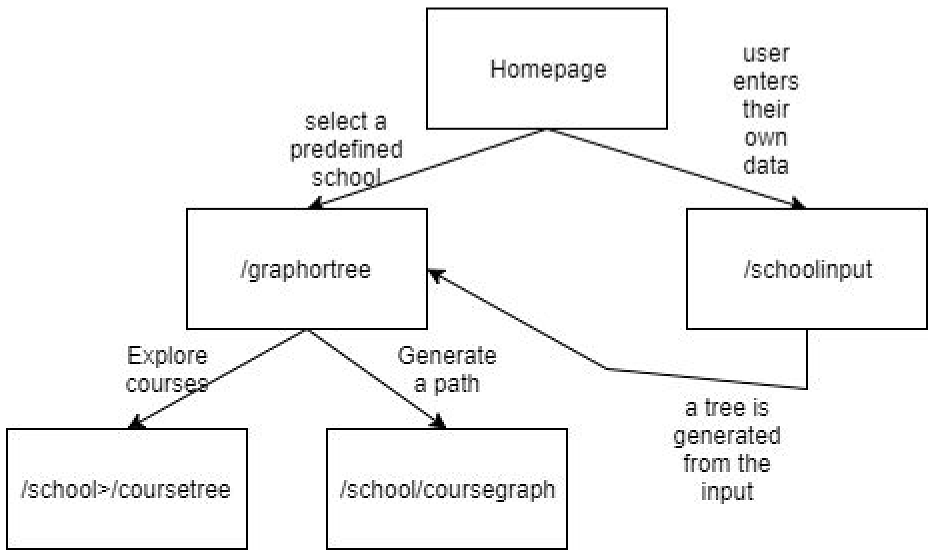
tree .js
- The d3 code to display a tree, and grow the tree with clicks

GraphTraversal.py
- Finds a path through the course graph given certian must have classes, prereqs and graduation requirements

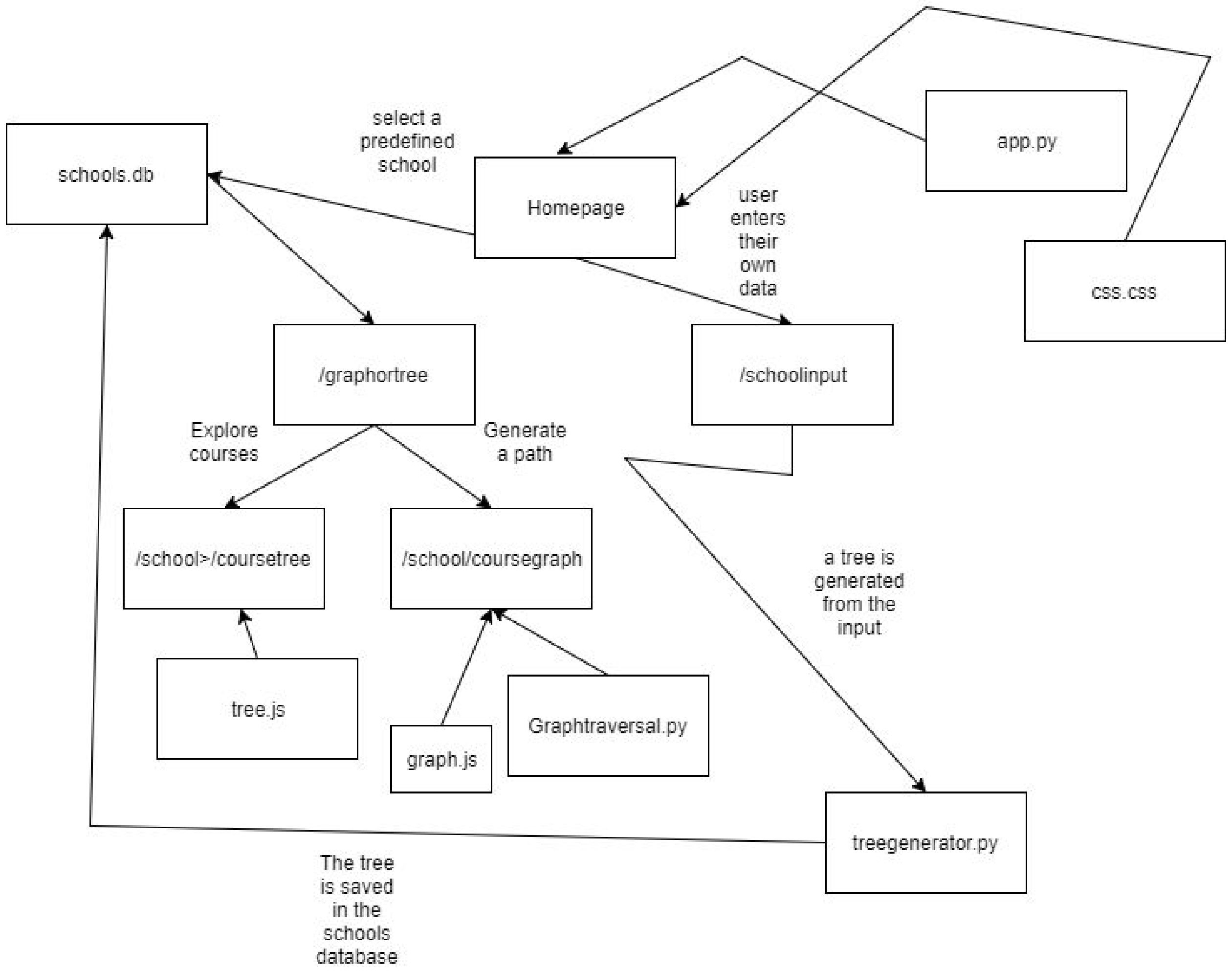
Graph.js
- d3 code for displaying the graph and genereated path

App.py
- The flask application backend

Sitemap



Component Map



Task Breakup

- Sarah/Zicheng: Course/prereq data to directed graph, suggested path generation algo
- Max: d3 representations of directed graph and tree
- Brian: MongoDB, CSS and Flask

Timeline

5/15: Structure out hardcoded data (based off Stuy curriculum)
5/16: Implement hardcoded data into CSV file
5/18: Put CSV files into database
5/19: Start basic design of front end page
5/22: Start flask and input of backend files
5/25: Start database functions
5/26: Start functions to generate the basic path
5/27: Start JS functions to display data
6/1: Finish database functions
6/3: Finish functions to generate basic path
6/5: Finish JS functions to provide dynamically displayed data
6/7: Polish front end
6/9: Start algorithm for career/graduation planning
6/16: Finish algorithm
6/19: Implement algorithm
6/20: Polish

Style Guide

- snake_case for variables
- Each function has a function header with input, output and a brief description
- Inline comments explaining the purpose of every for and while loop
- Inline comments explaining the purpose of every list comprehension