<DNA Assembly Comparison Tool>

Summary of Our Project

Given a sequence to assemble, our tool analyzes the attributes of the sequence (e.g. G-C content, presence of restriction sites, melting temperature, repeat sequences) and assigns appropriate cost penalties based on the requirements of each assembly method. Currently, the tool is able to consider 3 different assembly methods (i.e. Gibson, BioBrick, GoldenGate). At the same time, the tool does a comparative search of the best company that is capable of synthesizing the sequence (considering its different attributes) at the lowest possible cost. The user receives as a final output the best company and the best assembly method for their desired sequence.

Front-end

The front end is written in ReactJs with different components. The login page is the first page the user will see and after they press the “Assemble my Sequence!” button, it will direct the user to the order page. Within the order page, the user can input their sequence and choose an option of having the sequence being either “dsDNA, ssDNA, RNA, or Amino Acids”. The front end also has an error checking system if the user inputs anything that does not resemble a sequence. After submitting the sequence, the page will then display a result box with a recommendation of the best company based on price and the best assembly method based on cost and the turnaround time (in days). If the user presses the “Company detail” button, it will lead to a page where there are cards with information showing each company’s capabilities, how they calculate their price, and the turnaround time to assemble the sequence. Furthermore, if they press the “Assembly details” button, cards displaying the details of each assembly method and the parts that the sequence was split into.

API

The API was made using Django (python) which connects to a database filled with the data of all different companies that was created from MongoDB Atlas. The API itself has 4 different functions of GET,POST,PUT, and DELETE which is used throughout the code to update,delete, or grab from the database. Furthermore, the API is used to grab the front end input of the user and then use such input to run the back-end algorithm and display the output back to the front end again.

Back-end

The back end of the code is written as a class file Sequence.py using Python. The back end of the code calculates melting temperature, GC content, and secondary structure. Furthermore, the back end of the code splits the desired DNA sequence into parts based on what is required of the 3 supported assembly methods: BioBrick, Golden Gate, and Gibson. For example, the Biobrick assembly method requires that some parts exist within the BioBrick standard 10 library, and any parts which must be created cannot violate any restriction site rules set by BioBrick standard 10. Furthermore, Golden Gate requires that the sequence melting temperature must be above a certain threshold, parts must be of a certain range of lengths to support Golden Gate assembly, and overhangs of 4-base pair length must not be the same for parts to ensure the sequence is assembled in the order. For Gibson, the back end possesses a random sample function which randomly splits the DNA sequence 100 times into parts (5 or less) of different eligible sizes for Gibson. Parts must not violate overhang rules and meet a certain melting temperature threshold. The set of parts with the least possibility for formation of secondary structure is chosen. The back end also penalizes DNA assembly cost and turnaround time (time for company to assemble the sequence and send it to the user) based on the length of time and expense to perform each assembly method.

Special instructions and dependencies

After having the correct dependencies and materials as described by the README.md file and requirements.txt, have 2 different terminals open. One terminal will be cd into the APISystem folder. Within said folder, run the command "*python manage.py runserve*r" which will start the Django server and connect to the MongoDB Atlas database. In a different terminal, cd into the dna-tool server and run the command "*npm start*" which will start the react.js server and run the react app in development mode.