DuplicateStringChecker

Xiao He
June 10, 2020
xiaohe@andrew.cmu.edu

This web-based duplicate string checker takes string input from user and outputs the duplicated character(s) in this string. It uses `React` for frontend and `NodeJS Express` for backend.

Setup

- To run the frontend, navigate to the `client` folder from terminal
- From the `client` root folder, run `npm install` and then `npm start`.
- Navigate to `http://localhost:3000/`, you should see the interface.
- Now we need to run the backend. In a separate `terminal` tab, navigate to `api` folder
- Make sure you have `NodeJS` installed. if not, go to this link `https://nodejs.org/en/download/` to download Node.js.
- From `api` root folder, run `npm install` and then `npm start`
- To ensure API is working properly, navigate to `http://localhost:9000/stringCheckAPI/abb`, you should see `{"b":2}` on the screen
- Now that you have both backend and frontend running, typing or paste string to the text area and check the results. There are some sample texts on the webpage and in the project folder called `sample-data-xxx.txt`.
- Please don't hesitate to reach out to me if you have any suggestions!

Entry Points of Code

Frontend

`App.js` contains a static information sidebar `InfoSidebar.js` and the `StringCheckerPad.js`(client/src/components/StringCheckerPad) interface.

Backend

API is setup through `App.js` and `stringCheckAPI.js`(api/routes/stringCheckAPI.js)

Project Requirements

The component should display the duplicates contained in the input string and their count (i.e. how many times each duplicate appears in the input string).

1. Count duplications of only one character. Examples:
   abbab —> (a)=2, (b)=3
   bbbaabbbaa —> (a)=4, (b)=6
   The application logic to count the number of duplicates should be performed in the frontend.
2. Count all duplications, regardless of the number of characters. Examples:
   abbab —> (ab)=2, (a)=2, (b)=3
   bbbaacbbbaa —> (bbbaa)=2, (baa)=2, (bba)=2, (bbb)=2, (aa)=2, (bb)=2, (a)=4, (b)=6

The application logic to count the number of duplicates should be performed in the backend, and the frontend should access this functionality through an API call.

Other

I incorporated Javascript canvas to render the results when the datasets get relatively large and rendering objects using React gets slow. For backend, I was deciding between Python Django backend and Node Express and decide to use the later for it is convenient to setup for a single api call. Since we aren't storing anything to the database I think Express would do the work.

Future works

1. I think adding unit tests is definitely helpful.
2. There is height limit of canvas so I would investigate pagination more. Also if time permits I would investigate ways to render text and line clearer with canvas.
3. Just a thought, I think an alternative to canvas would be loading 1 page of results to user first, and then use `pagination` or `See More` button to render remaining chunks of datasets.