**Lets get Trendy**

**Your ultimate goal:** A simple and easy to use Google Trends comparison tool.

Stage 1: Users show up and are presented with 5 side by side search bars where they can enter different terms to compare. The user will then push a button “Best trend” and will be presented with 5 side by side charts each representing the google trends for each term over the past 90 days. Those charts which are trending will have a green box around them. The best one will have an arrow bouncing on it (or some other way of indicating that it is the best). The user will be able to click on any of the charts in order to expand it for better viewing and additional info (don’t worry for now about what the additional info will be. Its just an option I want to leave open for future development)

Stage 2: below each trend graph will be a button “hypertrend” which will take the search term from the graph and generate 5 new expansions on the term (“Frozen” would become “Frozen T-shirt”, “Frozen backpack”, “Frozen mug“ etc. I’ll write the function that actually generates the suggestions so just worry about the frontend). Just like stage 1, each term will receive a graph with a green box if it is trending and be expandable as well as have some sort of arrow indicating the best one.

Do all this and you get 1% of the company and then can choose after the summer if you want to stay involved (and work something out for a much bigger share in exchange for continued development (cause if this actually gets off the ground, we need someone to take charge while I’m in the army))

The overall idea is that we can very easily sell subscriptions to the website since its directed at the exact process e-commerce researchers currently use but is much faster.

------------------------------------------------------------------------------------------------------------

What you have to work with:

The backend for stage 1: is done. There is a function getGoogleTrends which takes a search term and returns the trends as well as whether or not the term is “trending” (which we are defining as having over 50% of the days being above 50% on googles trending index).

The frontend currently includes a library that displays the googleTrends as well as has a search bar (which will need to be duplicated 5 times). Feel free to make any changes you want.

**Libraries and technologies.**

Backend: Sails.js – a server framework. You shouldn’t need to touch this at all except to run the server each time which will get and return the trends for you.

Frontend:

Vue - a popular frontend JS framework. Much simpler to use than react or angular and gaining popularity fast. Components (front end pieces) in Vue are simple and get inserted into the overall html just like html components.

Webpack – a powerful front end server frame work. This will run the project on your computer and then all you need to do is open a browser to see the site. There are lots of different plugins for it. My code requires the “vue-cli” plugin (discussed below)

------------------------------------------------------------------------------------------------------------

**Getting started**

I recommend VS code as a good project editor for NodeJS

You will need to install Node.js and NPM (the library installer for Node.JS).

Once you get those, packages can be installed in by running “npm install <name of package>”. Google “npm install <package name>” if it ever doesn’t work. Its likely that you spelled it wrong.

Also when using a project someone else as created you just need to navigate to the project directory and run “npm install” which will look through a file called package.js and install all of the necessary dependencies.

Then go through the project described here <https://medium.com/@ahmedaminehachimi/paypal-checkout-workflow-with-vue-js-and-sails-js-4f151186a601>

to get a feel for vue, webpack and sails

To actually run the project you will need to navigate your terminal to the project folder. You will need two terminal tabs. One for the server and one for the frontend. The frontend is run using cd frontend && npm start

The backend is run using cd backend && sails lift

Since you will need to do this a lot you should download something for your terminal called ttab which will allow you to open and run things in a second tab from the command line. Then all you need to start the whole project is

ttab -d /$HOME/desktop/z/backend sails lift && cd $HOME/desktop/z/frontend &&npm start

Note: this will only work if the main folder (‘z’) is in your desktop. (I named it z so it goes to the very end of the list)

**About the project layout**

The folders are divided into “frontend”, “backend”, and “sharedFrontAndBack” (which currently only has holdovers related to parts of the project I edited out for simplicity). You will probably only need to edit “frontend.” In fact everything you need should be in “frontend/**src**”

src/**App.vue -** This is the effective home base for the project. It combines all components as well as contains the functions that call the server to get the google trends. You will need to place components here in order to display the google trends displayer components (which you will need to build)

**components/googleChartVue.vue** – the components that displays a chart (for showing the googleTrends) It uses googles actual library so the chart looks exactly like Google’s.

**router/index.js** – Think of this file like a train station linking the components folders and app.js. When you want to add a new component the way to do it is first add it to the components folder and then import it at the top of router/index.js and add it to the components object in router/index.js (copy googleChartVue)

src/main.js – you proabably don’t need to touch this

------------------------------------------------------------------------------------------------------------

**The real work**

Now that you are familiar with the project and with vue (I’m sure you are super confused as I was when I started but don’t worry you learn by doing) you can start the real work.

As you can see in App.js I have already created the 5 search bars and am displaying two of the google trends charts (the other 3 can be displayed by copying the relevant code in app.js and changing the numbers to 3,4 and 5)

**1.** the first step is to figure out how to organize the googleCharts neatly such that they can all be show together.

**2.** Step 2: Green highlight: Highlight the charts where the “fiftyOverFifty” variable (returned as part of the trendsObj which is returned from the call to the server) is true

**3.** Step 3: Click and expand: make it so each of the charts can be clicked and expanded so that the chart can be seen more clearly. Be sure to leave some room off to the side so we can add stats in later

**4.** Winner indicator – come up with a way to indicate which product is best (I will add a function that does the actual determining for now you can just pretend the first one is always best (but be sure to make sure that the code that shows which one is best works on all of the charts))

5. Part 2 should be substantially easier. We will cross that bridge when we get there.

------------------------------------------------------------------------------------------------------------

**Have Fun**

Remember, there is no pressure on any of this. If you don’t get it done then no harm done and at least you will get some good experience with interactive web dev (but you will only get equity if you finish both stage 1 and stage 2). Let me know when you inevitably get stuck, I’m here to help.