

OSCAR STIP Project Summer 2019

TATA Webtool Overview

Documentation by Quang Vo

Preface

The purpose of this document is to provide an overview of how TATA webtool code (both front end & back end) is organized. It has 2 main parts: **Front End (React.js)** and **Back End (Django)**, as well as a small session for **Apache server**. Before working on this project, you should familiarize yourself with **HTML, CSS, JavaScript, and Python**. You also should read the documentation of **React.js, Django REST framework, Plotly.js**, and some basic **UNIX/Linux commands**, otherwise you will be **LOST!!!**

It will be intimidating when you first look at the code, so I tried my best to make this document easy to understand (I hope ☺). There will be some details that I overlooked in this document, and I am sorry for that. However, I am still available on campus until December 2019, therefore I am willing to sit and walk you through everything if you have difficulty understanding the code. Just email me at gvo8@masonlive.gmu.edu or ask Luis and Tyrus, I will be there for you ;)

Front End (React.js)

Install yarn (or npm) and some yarn commands:

yarn install: install all the dependencies defined in a package.json file. (should run only **once** when you first start this project)

```
vos-MacBook-Pro:webtoolFE ryanvo1$ yarn install
yarn install v1.12.1
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
warning "react-scripts > @typescript-eslint/eslint-plugin@1.6.0" has unmet peer dependency "typescript@*".
warning "react-scripts > @typescript-eslint/parser@1.6.0" has unmet peer dependency "typescript@*".
warning "react-scripts > ts-pnp@1.1.2" has unmet peer dependency "typescript@*".
warning "react-scripts > @typescript-eslint/eslint-plugin > @typescript-eslint/typescript-estree@1.6.0" has unmet peer dependency "typescript@*".
warning "react-scripts > @typescript-eslint/eslint-plugin > tsutils@3.10.0" has unmet peer dependency "typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev || >= 3.4.0-dev".
[4/4] Building fresh packages...
success Saved lockfile.
warning Your current version of Yarn is out of date. The latest version is "1.17.3", while you're on "1.12.1".
info To upgrade, run the following command:
$ curl --compressed -o- -L https://yarnpkg.com/install.sh | bash
* Done in 24.65s.
```

yarn start: run the program on <http://localhost:3000>, for debugging and testing functions in JavaScript, HTML, CSS, but is **not** something we need for the server.

```
vos-MacBook-Pro:webtoolFE ryanvo1$ yarn start
yarn run v1.12.1
$ react-scripts start
```

yarn add ... : to add a certain package to be used in the code
Ex: yarn add react-select

yarn build: build the React production “build” folder for deployment with Django, and we **need** this for the server.

```
vos-MacBook-Pro:webtoolFE ryanvo1$ yarn build
yarn run v1.12.1
$ react-scripts build
Creating an optimized production build...
```

When the **yarn build is done**, you will see something like this:

```
The build folder is ready to be deployed.
You may serve it with a static server:

yarn global add serve
serve -s build

Find out more about deployment here:

https://bit.ly/CRA-deploy

✨ Done in 497.02s.
```

Before upload to the server: change the axios paths when working on either your local computer or on mason server, in the following files in **webtool/webtoolFE/src/components**: algorithmPage.js, csvReader.js, resultPage.js, homepage.js, gtexModal.js, groupingPage.js, finalPlots.js, and finalTables.js.

Ex:

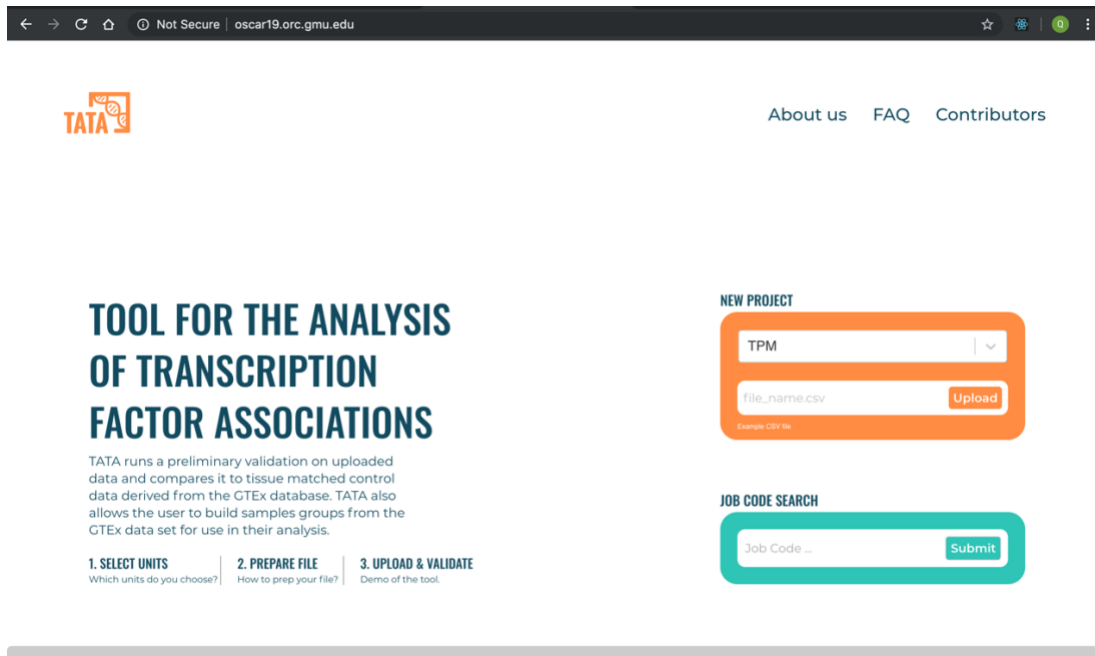
```
//axios.post('http://127.0.0.1:8000/backend/list3', //for localhost
axios.post('http://oscar19.orc.gmu.edu/backend/list3', //for mason server
```

Data flow in front end is in this order:

homepage.js & Contributions.js & csvReader.js -> validationPage.js -> taskPage.js ->
groupingPage.js & gtexModal.js & loadingModal.js -> batchPage.js -> algorithmPage.js ->
loadingPage.js -> resultPage.js & finalPlots.js & finalTables.js.

Let's walk through each page!

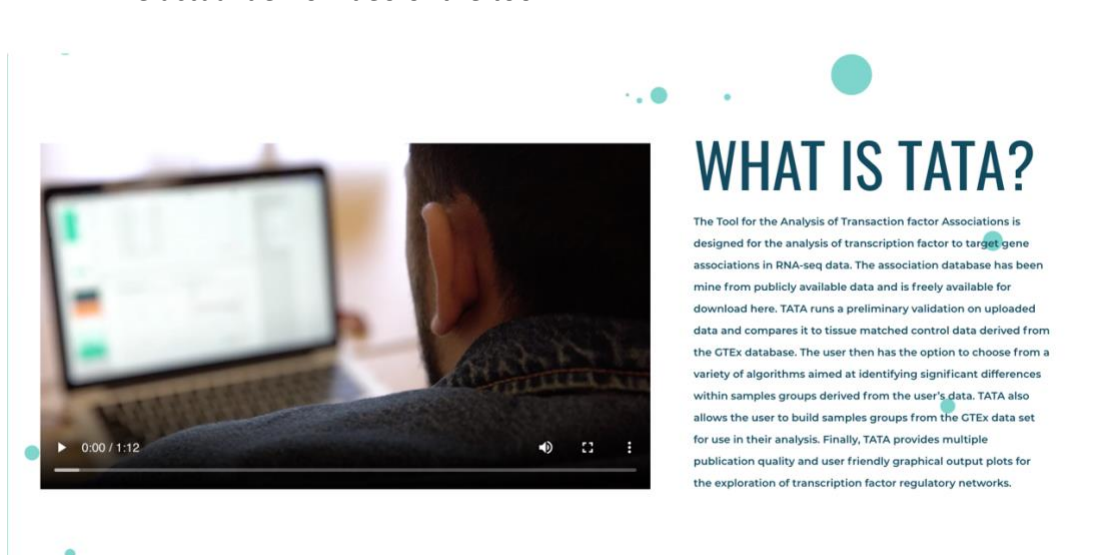
For homepage (homepage.js & homepage.module.css)
and 'yellow box' csv reader (csvReader.js & csvReader.css):



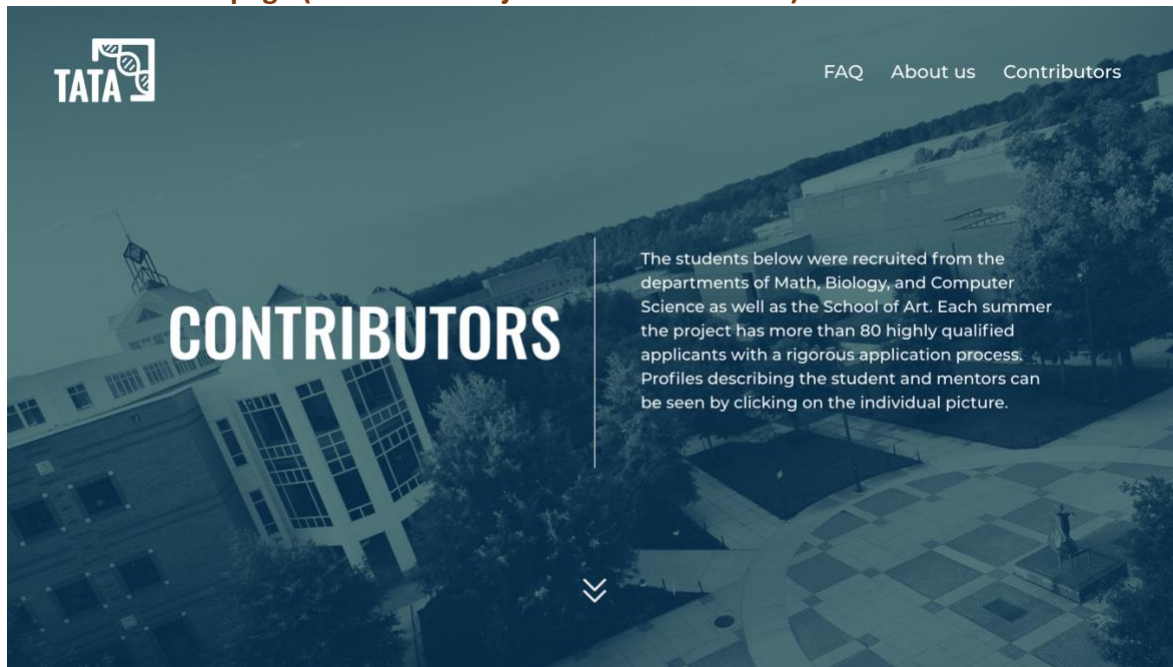
Most of the functionalities for the homepage are done!

What are we missing? The content for:

- The FAQ page
- Which units do you choose?
- How to prep your file?
- The actual demo video of the tool

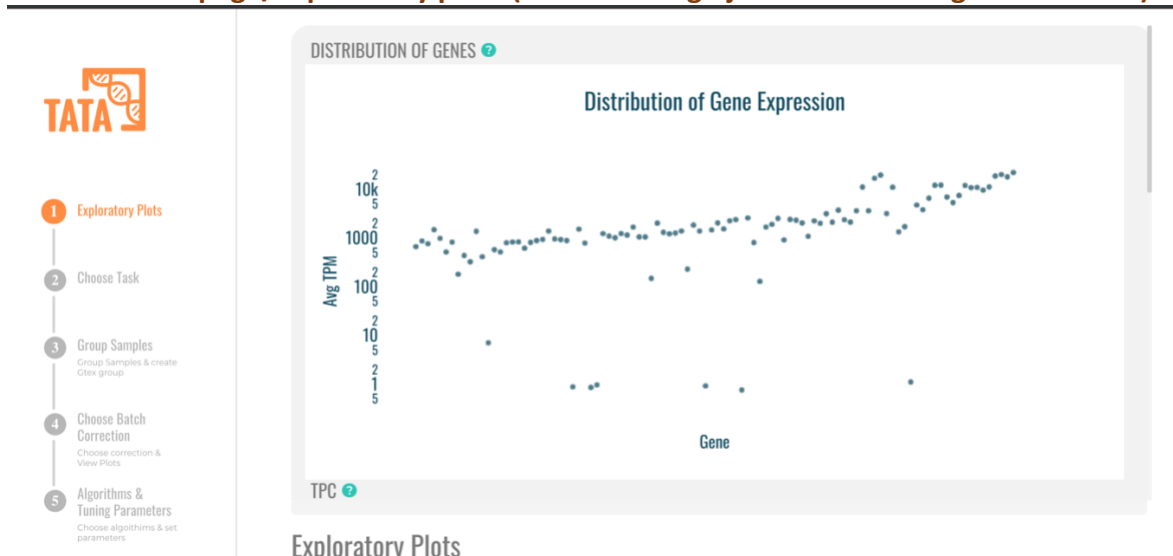


For contributors page (Contributions.js & Contributions.css):



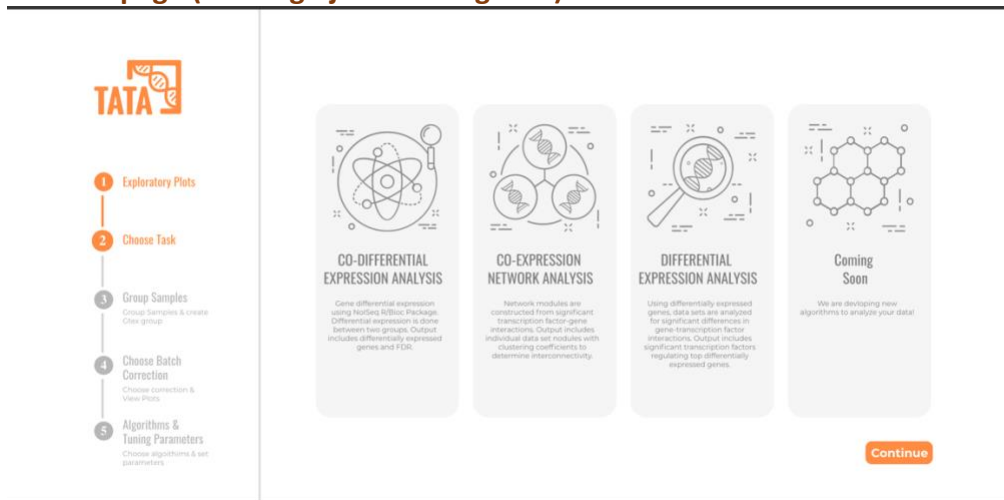
The contributors page is done!

For validation page/exploratory plots (validationPage.js & validationPage.module.css):



The validation page is done!

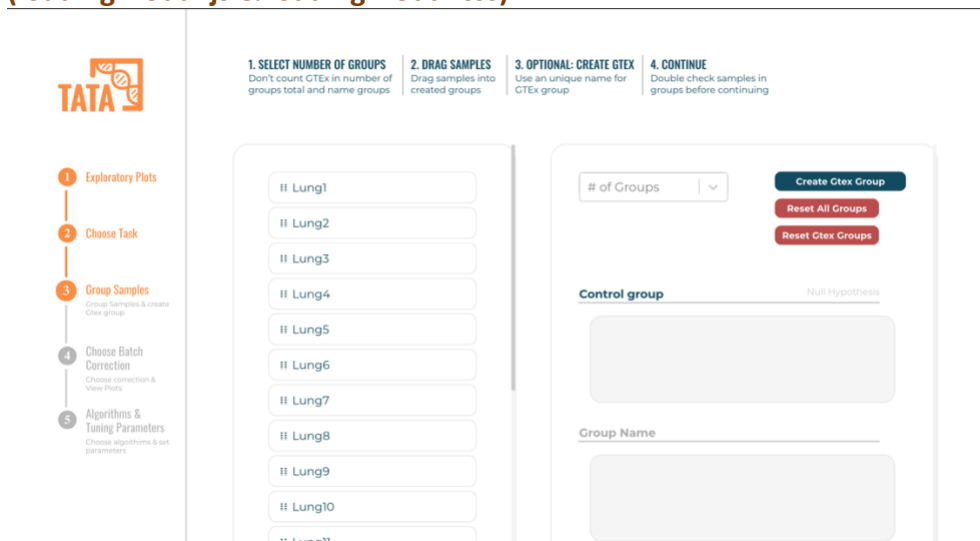
For task page (taskPage.js & taskPage.css):



The task page is done for now!

Note: The task chosen from this page will show on the algorithm page (later). Right now the last 'Coming Soon' button is disabled. When we have a new task available, just change the name and enable this button (instruction in the actual code)

For grouping page (groupingPage.js & groupingPage.css) and loading modal popup (loadingModal.js & loadingModal.css)

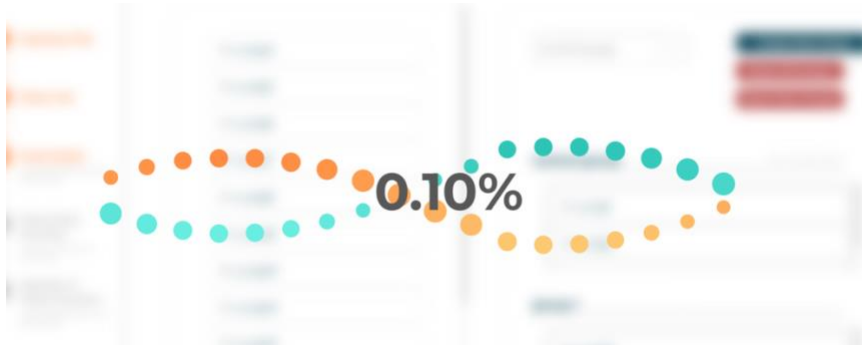


The grouping page is 75% done!

Note:

- Right now we can have up to **7** normal groups and **4** GTEx groups (**11 total**), if more than that there will be an alert saying we are exceeding the allowed number of groups.
- Also, we **cannot leave any group empty**, otherwise the code will break, and the DNA loading percentage (shown below) will go on forever ☹️.
- This DNA percentage is increasing 0.01%/sec and the limit is 99%, so if we want to change the speed, just change it groupingPage.js, in startProgress() function.

- **Most importantly, the drag and drop feature doesn't work on the tablet**, because the way it was implemented is "onClick", which requires a mouse. Therefore, we (may) need to implement extra "onTouch" in order for this feature to work on the tablet's touchscreen.



For GTEx modal (gtexModal.js & groupingPage.css):

Select GTEx

Characteristics

GTEx Group Name:

Sample Count:

0

↻

Age Range:
 Sex:
 Death:

Select Tissue type(s):

<input type="text" value="Adipose Tissue"/>	<input type="text" value="Ovary"/>
<input type="text" value="Adrenal Gland"/>	<input type="text" value="Pancreas"/>
<input type="text" value="Bladder"/>	<input type="text" value="Pituitary"/>
<input type="text" value="Brain"/>	<input type="text" value="Prostate"/>
<input type="text" value="Breast"/>	<input type="text" value="Salivary Gland"/>
<input type="text" value="Cervix Uteri"/>	<input type="text" value="Skin"/>
<input type="text" value="Colon"/>	<input type="text" value="Small Intestine"/>
<input type="text" value="Esophagus"/>	<input type="text" value="Spleen"/>
<input type="text" value="Fallopian Tube"/>	<input type="text" value="Stomach"/>
<input type="text" value="Heart"/>	<input type="text" value="Testis"/>
<input type="text" value="Kidney"/>	<input type="text" value="Thyroid"/>
<input type="text" value="Liver"/>	<input type="text" value="Uterus"/>
<input type="text" value="Lung"/>	<input type="text" value="Vagina"/>

The GTEx modal popup is done!

Note: when we create a new GTEx group, we need to make sure the GTEx group name is **unique/different** from the ones we have had, otherwise **the new one won't be created**.

For batch correction page (batchPage.js & batchPage.module.css):



The batch correction page is done!

Note: when we say yes/no for the question in the picture, it will show Correction Applied/No correction on the algorithm page (later). The default option is Yes.

For algorithm/tuning parameters page (algorithmPage.js & algorithmPage.css):

TATA

1 Exploratory Plots

2 Choose Task

3 Group Samples

4 Choose Batch Correction

5 Algorithms & Tuning Parameters

BATCH CORRECTION:

Correction Applied

ALGORITHM:

Differential Ex...

SAMPLE VARIANCE:

Unequal

False Discovery Rate Tuning Parameters

0.376

Bonferroni Alpha Tuning Parameters

0.678

Run Task

The algorithm/tuning parameters page is done!

Note: The BATCH CORRECTION box reflects the option chosen from batch correction page. The ALGORITHM box reflects the task chosen from the task page (this task **can** still be changed here). The SAMPLE VARIANCE has the default 'Unequal' value. The FDR has the min/max of 0.000/1.000, and the BA has the min/max of **0.001**/1.000 (BA can't be 0). They both have default value of 0.050.

For loading page (loadingPage.js & loadingPage.css):
Upper part:



[Home](#) [About us](#) [FAQ](#)

JOB CODE:20190814-RXYT5EE6



Lower part:



Make sure to remember your job code or input your email for a reminder when the job is done.

Submit

Start New Task

Cancel This Task

This loading page is working, but **only 50%** done!

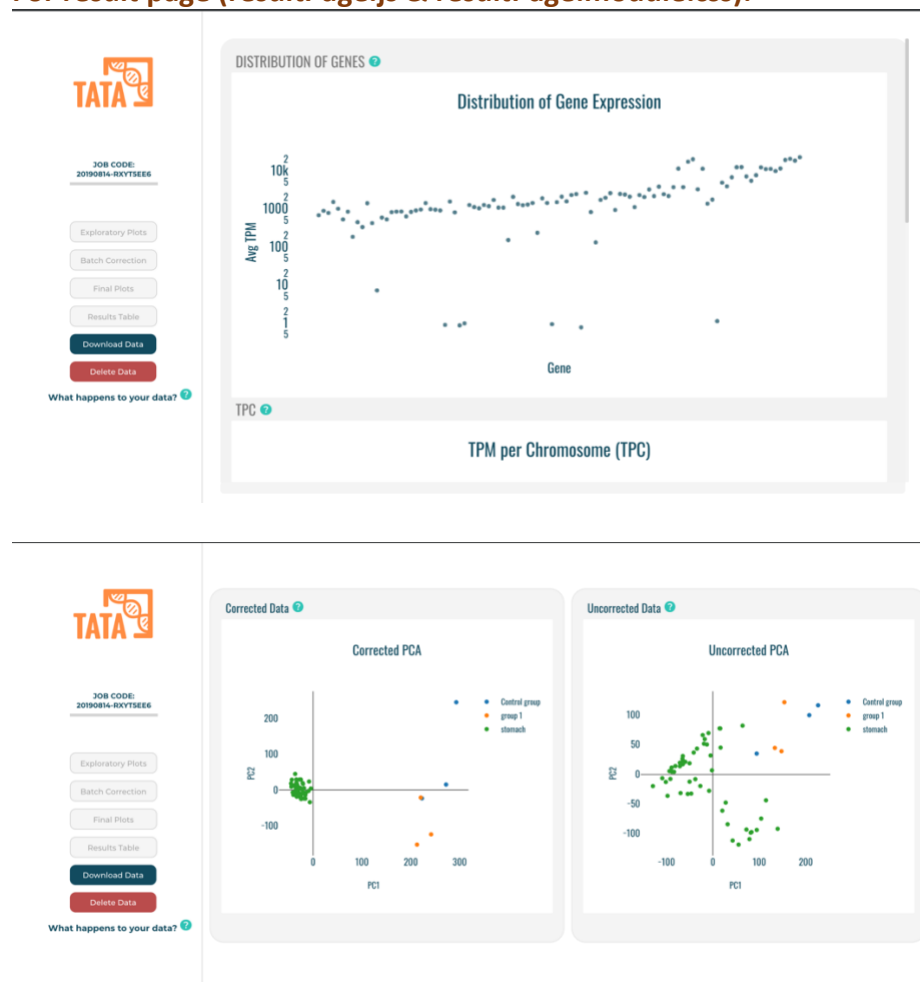
Note: The job code created is displayed correctly. **The percentage depends on where we enter the job code.**

- If we hit 'Run Task' on the algorithm page, the % will depend on when the back end is done processing and returns data to front end, and this DNA percentage is increasing 0.01%/sec and the limit is 99%, so if we want to change the speed, just change it in the actual code.

What are we missing?

- If the user enter the job code **on the home page**, it will look for the folder associated with that job code in **csvDatabase**, read the percentage in the **progress.csv**. If it is **less than $< 100\%$** , it will go to loading page and display that % (what will happen next? Ask Luis or Tyrus!). If it is exactly 100%, which means the task was complete, it will skip the loading page and go directly to the result page.
- The **email session** on the left is not implemented yet, in the future it should accept a valid email address, then send the job code and processed data to the user.
- The **“Start a new task”** button is not implemented yet, should go to the home page on a new tab.
- The **“Cancel this task”** button is not implemented yet, ask Luis or Tyrus what will happen when we hit this button.

For result page (resultPage.js & resultPage.module.css):



In the result page, the ‘Exploratory Plots’ and ‘Batch Correction’ buttons implementations are both done!

For final plots and final tables (finalPlots.js, finalTables.js & finalPlots.module.css):



JOB CODE:
20190814-RVY7SEEE

Exploratory Plots

Batch Correction

Final Plots

Results Table

Download Data

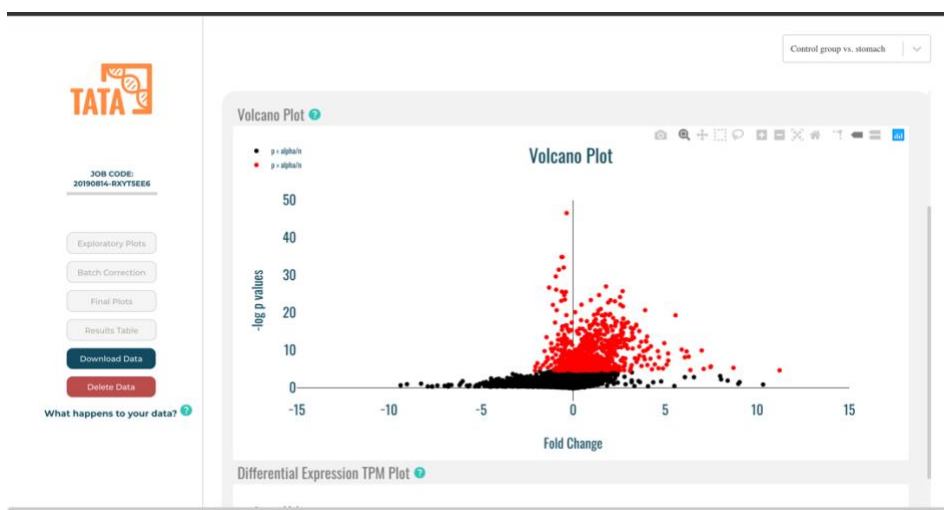
Delete Data

What happens to your data?

Control group vs. group 1

	Control group vs. group 1	Fold Change	T values	P values	Negative log 10 P values	Control group raw mean	group 1 raw mean	Mean Expression
0	ENSG00000153561	2.353e-5	-7.655e-5	9.999e-1	2.493e-5	1.490e+1	1.463e+1	1.476e+1
1	ENSG00000170871	2.343e-5	-8.094e-5	9.999e-1	2.579e-5	1.532e+1	1.566e+1	1.549e+1
2	ENSG00000178567	-1.685e-5	9.395e-5	9.999e-1	3.035e-5	2.596e+1	2.479e+1	2.533e+1
3	ENSG00000123429	2.068e-5	-9.486e-5	9.999e-1	3.089e-5	8.823e+1	8.392e+1	8.607e+1
4	ENSG00000153993	-5.075e-5	1.066e-4	9.999e-1	3.390e-5	5.279e+0	4.408e+0	4.809e+0
5	ENSG00000196491	2.451e-5	-1.602e-4	9.999e-1	5.289e-5	7.233e+1	6.983e+1	7.099e+1
6	ENSG00000142267	1.479e-4	-1.707e-4	9.999e-1	5.405e-5	1.027e+1	9.099e+0	9.683e+0
7	ENSG00000123960	4.784e-5	-1.680e-4	9.999e-1	5.448e-5	7.797e+0	7.76e+0	7.407e+0
8	ENSG00000123975	2.309e-4	-2.182e-4	9.998e-1	7.107e-5	6.773e+1	8.303e+1	7.238e+1
9	ENSG00000125354	1.321e-4	-4.508e-4	9.997e-1	1.263e-4	1.997e+1	1.895e+1	1.946e+1
10	ENSG00000126983	-2.150e-5	4.454e-4	9.997e-1	1.389e-4	7.603e+2	7.587e+2	7.594e+2
11	ENSG000001204381	-1.800e-4	6.870e-4	9.995e-1	2.123e-4	1.580e+1	1.633e+1	1.607e+1
12	ENSG00000120710	-1.864e-4	7.207e-4	9.995e-1	2.226e-4	6.625e+1	7.027e+1	6.824e+1
13	ENSG00000185065	1.013e-4	-8.802e-4	9.994e-1	2.792e-4	4.040e+1	3.992e+1	4.021e+1
14	ENSG00000127720	3.736e-4	-9.176e-4	9.993e-1	2.977e-4	4.372e+0	4.155e+0	4.264e+0
15	ENSG000001230812	1.043e-4	-9.326e-4	9.993e-1	2.999e-4	1.064e+0	1.003e+0	1.033e+0
16	ENSG000001233090	-1.918e-4	9.789e-4	9.993e-1	3.027e-4	6.846e+1	6.553e+1	6.700e+1
17	ENSG00000189233	-2.628e-4	1.009e-3	9.993e-1	3.133e-4	1.054e+0	1.029e+0	1.041e+0

The final tables page is done!



The final plots page is **only 75%** done!

Note: There are 3 plots in final plots page, and right now we can only display 2 of them (Volcano & Differential only).

What are we missing?

- Due to some limitations of Apache server, React.js, and Django, we cannot display the **heatmaps** (with format .png and are stored in **csvDatabase/jobCode/results** folder), therefore we **have to** find way to display them in the future.
- Also, when other algorithms are done (such as Co-Differential Expression & Co-Expression Network), more images .png will be created (such as **network graphs**), so it will be really bad if we can't display them ☹

For Download Data & Delete Data buttons:

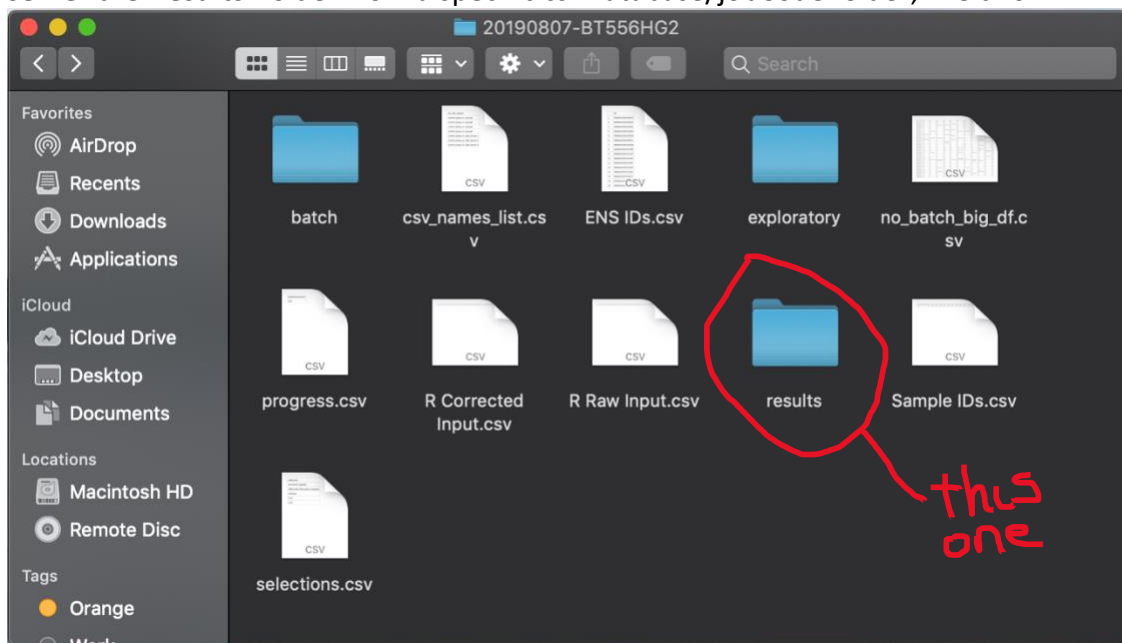
Download Data

Delete Data

What happens to your data? ?

What are we missing?

- Due to some limitations of Apache server, React.js, and Django, we cannot download or delete data (in resultPage.js).
- In the future, we **need to** find a way so that the user can download or delete **from the server** the 'results' folder from a specific csvDatabase/jobCode folder, like this:



For error page (errorPage.js & errorPage.css):



What are we missing?

- This error page **needs to be** implemented in the future. Right now if something on the **back end breaks**, the percentage in loading modal/page will keep increasing forever, and unfortunately the user **won't** know if an error has occurred and will wait forever ☹️
- **Suggested solution:** maybe do “try-except” for each function in **inputPage/views.py**, then send back to front end a signal to display the error page (?)

For webtoolFE/src/App.js & App.css:

```
Js App.js x
54
55 render() {
56   return (
57     // Render and route specific js file depend on given exact path
58     <div className="app-container">
59       <Route exact path="/" //need to be changed to Webtool later
60       render={() => (
61         <HomePage
62         />
63       )
64       }
65     />
66     <Route exact path="/validation"
67     render={() => (
68       <ValidationPage /> //added by Quang for testing
69     )
70     }
71   />
72   <Route exact path="/taskpage"
73   render={() => (
74     <TaskPage
75     />
76   )
77   }
78 />
79 <Route exact path="/groupingpage"
80 render={() => (
81   <GroupingPage
82   />
83 )
84 }
85 />
86 <Route exact path="/batchpage"
87 render={() => (
88   <BatchPage
```

This file contains the paths of the pages we want to have in our TATA webtool, **please update it when you want to add a new page.**

-----Front End ends-----

-----Back End (Django)-----

Before upload to the server: change from local paths to server paths in the following files:

In **webtool/inputPage** folder: **ds_class.py**, **GTEXDataAnalyzer.py**, **query.py**, **views.py**

⇒ Look for the comments for the paths in the actual code!

In **webtool/webtoolBE** folder: change **settings.py** as below:

```
#ALLOWED_HOSTS = ['oscar19.orc.gmu.edu'] # Use this one when deploy onto Apache server
ALLOWED_HOSTS = [] # For local server testing only
```

What files to deal with?

For back end (Django part), we are dealing with these files:

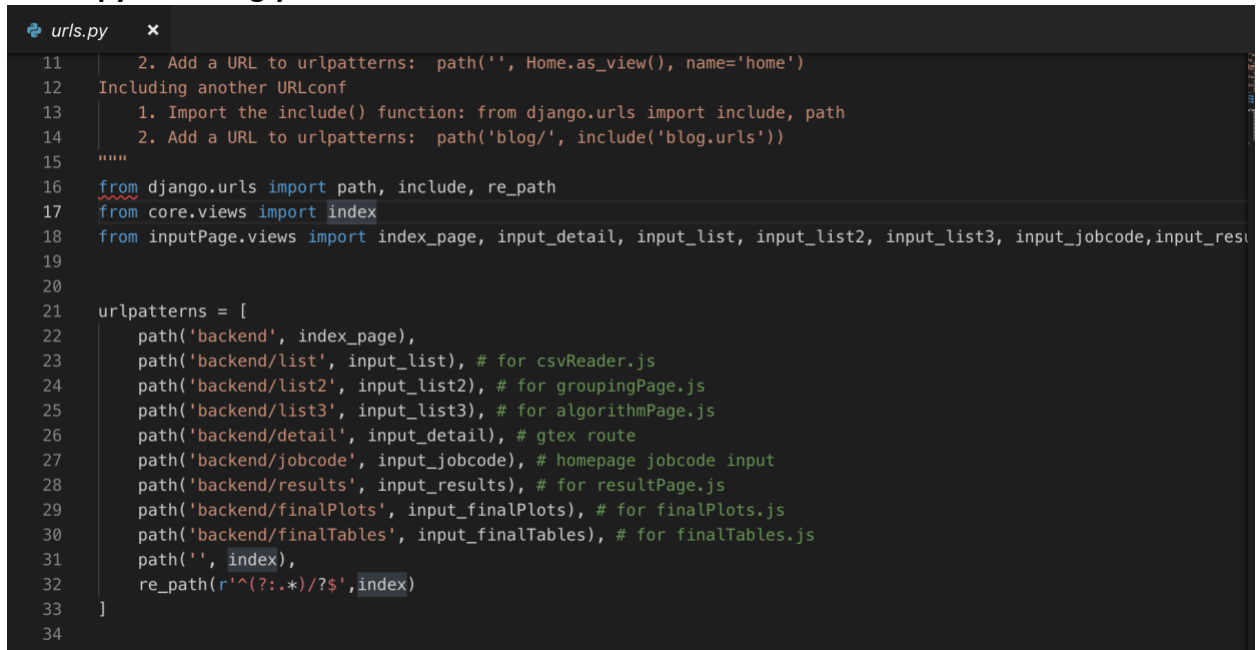
- **webtoolBE/settings.py**: change the part shown above.
- **inputPage/ds_class.py**: this Python file was written by Math team, and it's done! Just make sure you **change the path** in the file when working on the server.
- **inputPage/query.py**: this Python file was also written by Math team, and it's done! Just make sure you **change the path** in the file when working on the server.
- **inputPage/GTEXDataAnalyzer.py**: also written by Math team. This file holds most of the algorithms for the TATA tool as well as creating the job code folder. Please contact Luis or Tyrus for the documentation for this file (written by Jae-Moon Hwang).
Note: In the future we need to replace the current used **GTEXDataAnalyzer.py** with another file called **GTEXDataAnalyzer_(to be used in 2020).py** (in the same **inputPage/** folder), because the new one has the algorithm for Co-Differential Expression (which I didn't have enough time to merge it into my working code).

- **inputPage/views.py**: this file is the **communication** between React.js (front end) and Django (back end), and **is the one we have to deal with most of the time**. It has the following functions that receive request from front end, process data on the back end, and send back processed data to front end:

- **input_list(request)**: for axios.post in csvReader.js
- **input_list2(request)**: for axios.post in groupingPage.js => batchPage.js
- **input_list3(request)**: for axios.post in algorithmPage.js => resultPage.js
- **input_detail(request)**: for axios.post in gtexModal.js
- **input_results(request)**: for axios.post in resultPage.js
- **input_finalPlots(request)**: for axios.post in finalPlots.js
- **input_finalTables(request)**: for axios.post in finalTables.js
- **input_jobcode(request)**: for axios.post in homePage.js => loadingPage.js

Note: to understand what each function does, **please look at the comments in the actual code**.

- **webtoolBE/urls.py:** for every new function we create in views.py, **we have to update it in urls.py accordingly.**



```

11      2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12      Including another URLconf
13      1. Import the include() function: from django.urls import include, path
14      2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15      """
16      from django.urls import path, include, re_path
17      from core.views import index
18      from inputPage.views import index_page, input_detail, input_list, input_list2, input_list3, input_jobcode, input_resu
19
20
21      urlpatterns = [
22          path('backend', index_page),
23          path('backend/list', input_list), # for csvReader.js
24          path('backend/list2', input_list2), # for groupingPage.js
25          path('backend/list3', input_list3), # for algorithmPage.js
26          path('backend/detail', input_detail), # gtex route
27          path('backend/jobcode', input_jobcode), # homepage jobcode input
28          path('backend/results', input_results), # for resultPage.js
29          path('backend/finalPlots', input_finalPlots), # for finalPlots.js
30          path('backend/finalTables', input_finalTables), # for finalTables.js
31          path('', index),
32          re_path(r'^(?!.*)/?$', index)
33      ]
34

```

Some important paths on the server:

- Link to the csvDatabase:
/var/www/html/webtool/csvDatabase
- Link to the GTExDatabase:
/var/www/html/webtool/inputPage/GTExDatabase
- Link to the R scripts:
/var/www/html/webtool/inputPage/R_batch_correction_scripts
Note: for more information about the installed R packages on the server, please contact Luis and Tyrus.

Some important Django commands:

virtualenv webtoolBEenv: create a Python virtual environment so that our Django project will be separate from the system's tools and any other Python projects we may be working on. **(this one was already done, unless you want to deploy the web app all over again)**

source webtoolBEenv/bin/activate: activate the virtual environment

To install necessary packages: (should be done **once** on the first time working on this project):

(webtoolBEenv) bash-3.2\$ pip install django

(webtoolBEenv) bash-3.2\$ pip install djangorestframework

(webtoolBEenv) bash-3.2\$ pip install django-cors-headers

```
(webtoolBEenv) bash-3.2$ pip install pandas
(webtoolBEenv) bash-3.2$ pip install sklearn
(webtoolBEenv) bash-3.2$ pip install pygments
(webtoolBEenv) bash-3.2$ pip install seaborn
(webtoolBEenv) bash-3.2$ pip install statsmodels
(webtoolBEenv) bash-3.2$ pip install matplotlib
(webtoolBEenv) bash-3.2$ pip install numpy
(webtoolBEenv) bash-3.2$ pip install plotly (for 2020)
(webtoolBEenv) bash-3.2$ pip install network (for 2020)
```

deactivate: back out of our virtual environment

django manage.py runserver: to run server on port <http://127.0.0.1:8000>

```
^C(TATA-Web-Server) bash-3.2$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
August 15, 2019 - 05:32:42
Django version 2.2.2, using settings 'webtoolBE.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Apache server

Link for deploying Django on Apache server: (please read & follow the instruction in this link to deploy the web app on the server)

https://www.digitalocean.com/community/tutorials/how-to-serve-django-applications-with-apache-and-mod_wsgi-on-ubuntu-16-04

Note: Whenever we change something in any files, we need to restart the server by running these 2 commands:

sudo service apache2 restart

sudo systemctl restart apache2

To see what error the server is having:

cd /var/log

sudo cat apache2/error.log

Looking at the error.log is really important to see whether the server is working correctly or not, and what kind of error it's having.

-----**Back End (Django) ends**-----

That's it! Good luck with the continuation of this project. Don't hesitate to reach out to me at gvo8@masonlive.gmu.edu if you have any questions ;)