Metrics Report #2

Metric: Average Lines of Code Per Method

‘

Prepared By Victoria Law

Submitted March 11th, 2019

CSC 405 Capstone, Team Flash

Winter 2019 Dr. Da Silva

Frontend LOC

For our frontend, we are using ReactJS.

Frontend			
File Name	**Total LOC**	**Number of Methods**	**Average LOC**
index.html	40	1	40
App.js	22	2	11
App.test.js	7	1	7
index.js	18	1	18
serviceWorker.js	131	4	32.75
index.js	142	9	15.77777778
rootReducer.js	20	3	6.666666667
actions.js	1	1	1
reducer.js	19	2	9.5
index.js	192	10	19.2
actions.js	2	2	1
index.js	1	1	1
popup_container.js	90	8	11.25
reducer.js	23	2	11.5
index.js	109	10	10.9
index.js	140	11	12.72727273
index.js	114	8	14.25
index.js	8	1	8
actions.js	1	1	1
index.js	1	1	1
reducer.js	22	2	11
files.js	125	5	25
index.js	187	12	15.58333333
upload_button.js	28	2	14
welcome.js	80	5	16
index.js	57	7	8.142857143
column.js	63	7	9
editable_item.js	40	3	13.33333333

explorer.js	138	6	23
index.js	169	14	12.07142857
item.js	55	4	13.75
preview.js	48	5	9.6
search.js	59	4	14.75
spacer.js	7	1	7
truncate.js	17	1	17
index.js	108	7	15.42857143
index.js	51	2	25.5
index.js	38	2	19
animation.js	53	1	53
color.js	95	1	95
fontSize.js	11	1	11
index.js	29	1	29
lineHeight.js	4	1	4
spacing.js	10	1	10
transition.js	3	1	3
Total	**2578**	**175**	**14.73142857**

Backend LOC

For our backend, we are using Python.

Backend			
File Name	**Total LOC**	**Number of Methods**	**Average LOC**
csvToJson.py	75	7	10.71428571
server.py	62	3	20.66666667
test_endpoints.py	8	2	4
CsvInterpreter.py	59	7	8.428571429
DataClassifier.py	19	3	6.333333333
Tester.py	10	2	5
TrainingDataMaker.py	36	3	12
Total	**269**	**27**	**9.962962963**

Analysis/Observations

The backend functions were written in Python and used to make the API, the ML, the data categorizing, the tags, and converting from CSV to JSON files. The frontend files counted were only the react components and didn't include the constructor() or render() functions. The frontend functions were written mainly in Javascript and used to make the frontend endpoints, and the looks of the application. The goal of this metrics is to see the average amount of lines of code per function. Generally, we want to have fewer lines of code per function so that it is decoupled. Having fewer lines of code per function makes debugging easier because there isn't a big chunk of code to look through.

Comparing the first metric to the current one, we are doing better in terms of having less lines of code per function. For the frontend, we were able to reduce it by 1 line. For the frontend, the average lines of code before was 15, and now it's 14. The largest file came from index.js, and that's understandable because that's where we have all the code for how the application looks like. For the backend, we reduced it by 3 lines of code per function. Before, the average lines of code for the backend was 13, and now, it's 10. For the backend, the file with the most lines of code is the CSV to JSON file. This is also fine, because we have the most functions in that file, we need the different functions to parse through the different headers, and to take in the file, and send it back to the frontend.