Report - Get GitHub Repository Statistics

💡 **Project Information**

Language: JavaScript

Environment: Node 20.11.0

Package Management: Npm

Module type: ESM

IDE: VsCode

SubTask 1: Statistics of various attributes

1.1 Process

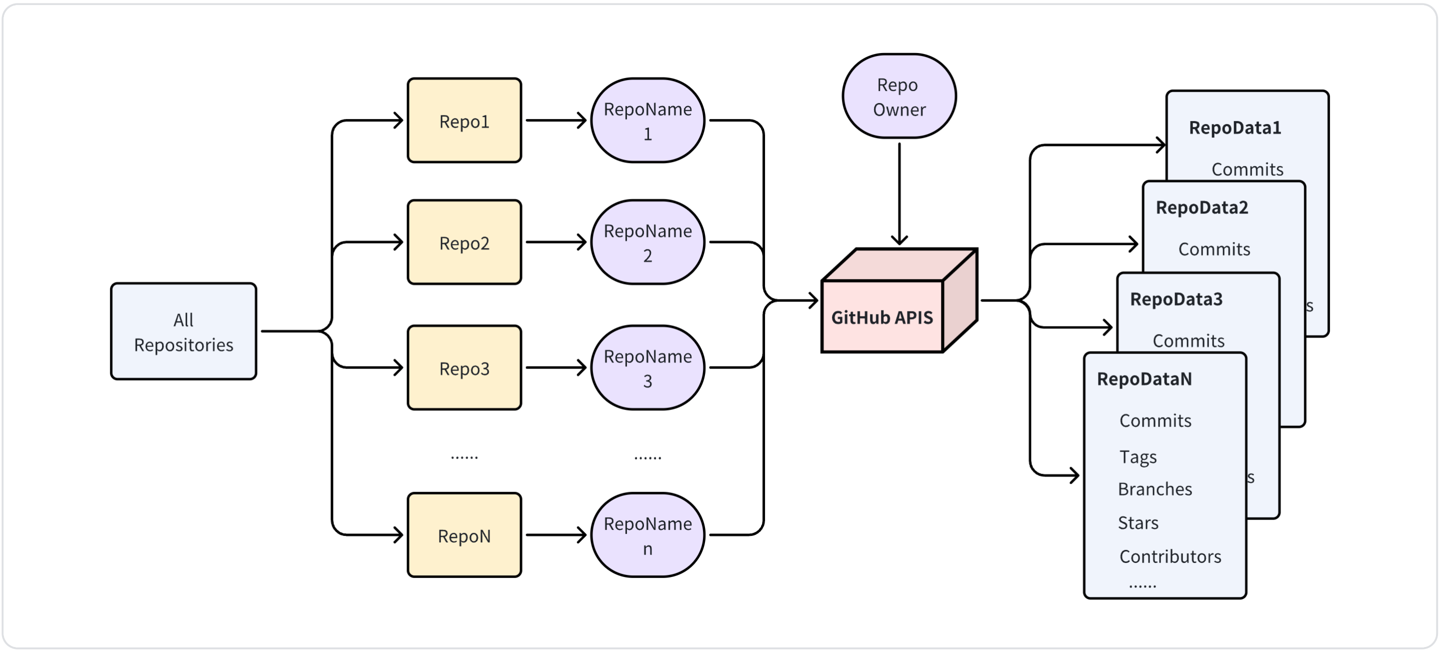
1.1.1 Main Steps

📌 Step 1: Get all repositories belonging to **RepoOwner** (Kaggle)

📌 Step 2: Combine each **RepoName and RepoOwner** as parameters and visit Github API, and we can get the **RepoData** object for each Repository.

```
    // the structure of RepoData
    interface RepoData {
        commits: number,
        stars: number,
        contributors: number,
        branches: number,
        tags: number,
        forks: number,
        releases: number,
        closedIssues: number,
        environments: number,
    }
```

Step 1 and 2 are shown in the diagram below.



📌 Step 3: Summarize all **RepoData** and calculate each category (commits/tags/...)'s total and median number.

1.1.2 Github API

(1) Authorization

When using GitHub APIs, we need to put an **access token** in the request header. The access token can be generated on GitHub=>Settings.

```
export const ACCESS_TOKEN =

"github_pat_11AQH247A026xR2YQepoKy_QwXbLlj7r803VX5BPsl0gS7KFPK4wFs5ZD3c3fCKb9OR
B4S2WH62pXKi3uu";

const octokit = new Octokit({
  auth: ACCESS_TOKEN,
});
```

(2) APIs used in this task

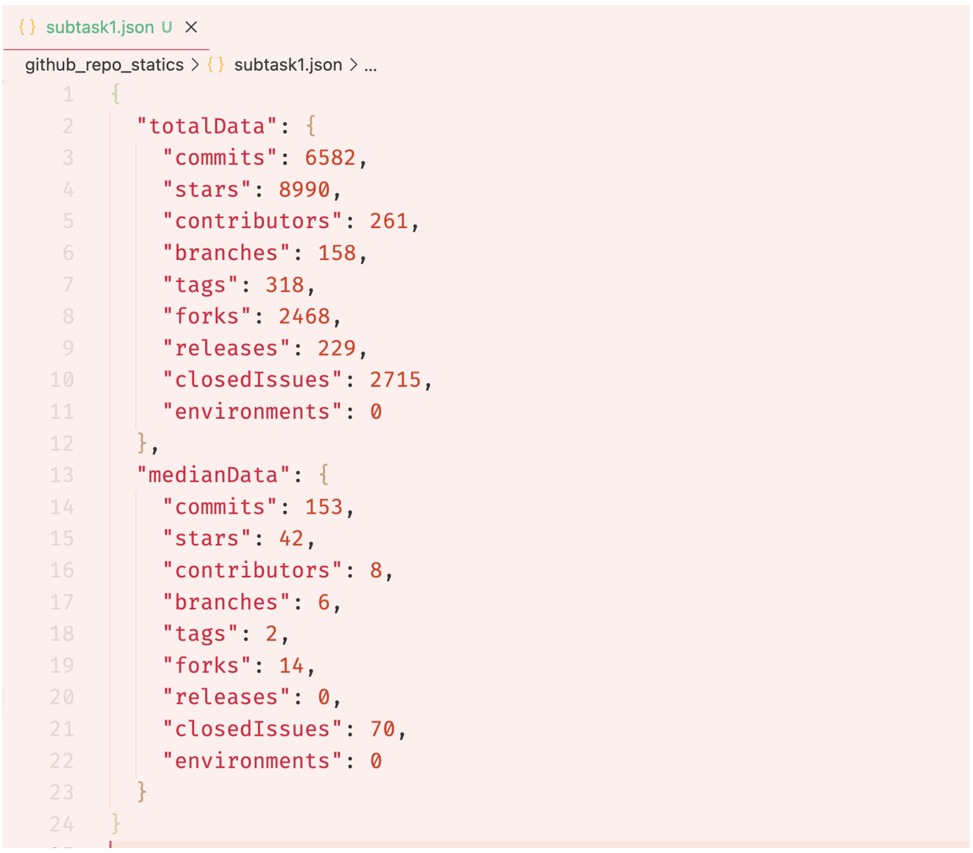
```
import { OWNER } from "./constant.js";
export const keyAPIS = (repo) => ({
  commits: `/repos/${OWNER}/${repo}/commits`,
  stars: `/repos/${OWNER}/${repo}`,
  contributors: `/repos/${OWNER}/${repo}/contributors`,
  branches: `/repos/${OWNER}/${repo}/branches`,
  tags: `/repos/${OWNER}/${repo}/tags`,
  forks: `/repos/${OWNER}/${repo}/forks`,
  releases: `/repos/${OWNER}/${repo}/releases`,
  closedIssues: `/repos/${OWNER}/${repo}/issues?state=closed`,
  environments: `/repos/${OWNER}/${repo}/environments`,
});

export const commonAPIs = {
  ALL_REPOS: `/users/${OWNER}/repos`,
};
```

1.2 JSON Result

> A total of 11 repositories under Kaggle were analyzed. The data for each repository can be found in the "log" folder.

The statistical results are as follows table.



```
{
  "totalData": {
    "commits": 6582,
    "stars": 8990,
    "contributors": 261,
    "branches": 158,
    "tags": 318,
    "forks": 2468,
    "releases": 229,
    "closedIssues": 2715,
    "environments": 0
  },
  "medianData": {
    "commits": 153,
    "stars": 42,
    "contributors": 8,
    "branches": 6,
    "tags": 2,
    "forks": 14,
    "releases": 0,
    "closedIssues": 70,
    "environments": 0
  }
}
```

The results are also shown in report.html in the project which will be automatically opened after starting the project.

> 😢 Undone work: the number of `environments` is always 0. Maybe it is coding or API problem. I doubt it is the database visited by API because the request works normally and returns code 200 and proper data. But I am not sure and I don't have time to fix it because of the time limit. Sorry about that.

1.3 Optimization

1.3.1 Encapsulation of public components.

- Encapsulate Request

```
const octokit = new Octokit({
  auth: ACCESS_TOKEN,
});

const commonHeader = {
  owner: process.env.Owner,
  per_page: 100,
  headers: {
    "X-GitHub-Api-Version": "2022-11-28",
  },
};

export const octoRequest = async (url, params = {}, isPagination = true) => {
    // ...
    const response = await octokit.request(`GET ${url}`, {
        ...commonHeader,
        ...params,
      });
    // ...
 }
```

- Encapsulate util Methods

The common utils methods can be reused in different situations globally.

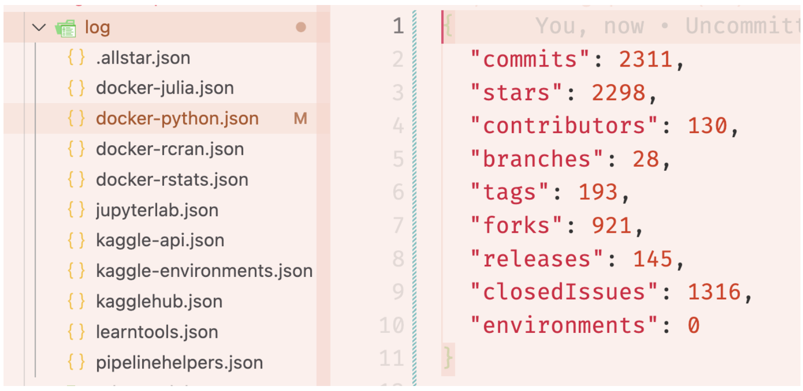
```
export const getAllRepos = async () => {
  const repos = await octoRequest(commonAPIs.ALL_REPOS);
  return repos;
};

export const getTotalNumber = (allData) => {
  xxx
};

export const getMedianNumber = (allData) => {
 xxx
 };
```

1.3.2 Keep intermediate results

I print the RepoData corresponding to each repository to the "log", to keep intermediate results for process tracing.



```
{
  "commits": 2311,
  "stars": 2298,
  "contributors": 130,
  "branches": 28,
  "tags": 193,
  "forks": 921,
  "releases": 145,
  "closedIssues": 1316,
  "environments": 0
}
```

1.3.3 Customize Repository Account

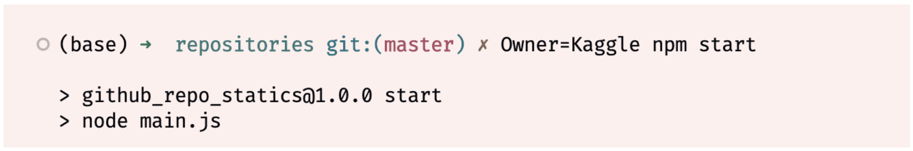
Owner(Kaggle in this task) is just a parameter passed to the GitHub API and can be any custom value.

In this project, the solution I implemented is to inject it into the global variable during project startup (npm start), making it accessible globally. In this case, we don't even need to write down the word "Kaggle" in the code. All you need to do is to type in the **owner** in the startup shell.

```
// process.env.Owner can be accessed globally
const commonHeader = {
  owner: process.env.Owner,
  per_page: 100,
  headers: {
    "X-GitHub-Api-Version": "2022-11-28",
  },
};
```

This is the startup Shell

```
Owner=Kaggle npm start
```



```
○ (base) ➜  repositories git:(master) ✗ Owner=Kaggle npm start

  > github_repo_statics@1.0.0 start
  > node main.js
```

SubTask 2: Statistics of source code lines per programming languages

2.1 Method Choice

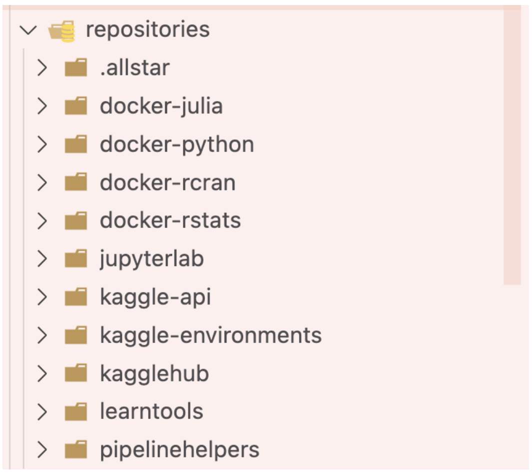
1. AST

For different languages, the libraries for AST parsing are different. Therefore, it is challenging to design a universal method.

2. Code line counting tool： **cloc**

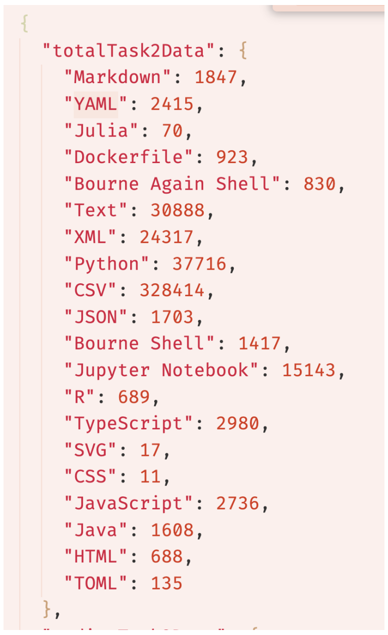
```
    // ---- SubTask2: Statistics of source code lines per programming languages -
  --//
    await cloneRepos(allRepos);
    await scanCode();
```

All repositories are kept in this folder. But it is unnecessary to commit it to git.

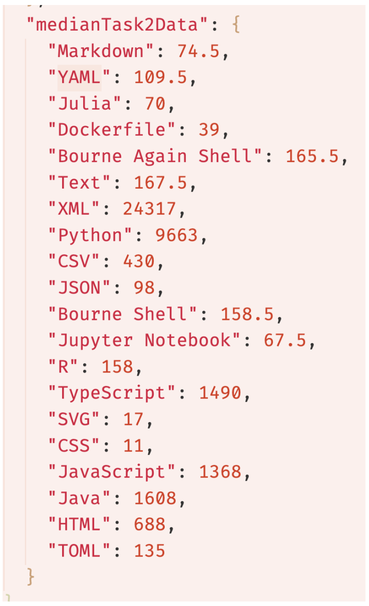


```
repositories
  .allstar
  docker-julia
  docker-python
  docker-rcran
  docker-rstats
  jupyterlab
  kaggle-api
  kaggle-environments
  kagglehub
  learntools
  pipelinehelpers
```

2.2 JSON Result



```
{
  "totalTask2Data": {
    "Markdown": 1847,
    "YAML": 2415,
    "Julia": 70,
    "Dockerfile": 923,
    "Bourne Again Shell": 830,
    "Text": 30888,
    "XML": 24317,
    "Python": 37716,
    "CSV": 328414,
    "JSON": 1703,
    "Bourne Shell": 1417,
    "Jupyter Notebook": 15143,
    "R": 689,
    "TypeScript": 2980,
    "SVG": 17,
    "CSS": 11,
    "JavaScript": 2736,
    "Java": 1608,
    "HTML": 688,
    "TOML": 135
  },
```



```
  "medianTask2Data": {
    "Markdown": 74.5,
    "YAML": 109.5,
    "Julia": 70,
    "Dockerfile": 39,
    "Bourne Again Shell": 165.5,
    "Text": 167.5,
    "XML": 24317,
    "Python": 9663,
    "CSV": 430,
    "JSON": 98,
    "Bourne Shell": 158.5,
    "Jupyter Notebook": 67.5,
    "R": 158,
    "TypeScript": 1490,
    "SVG": 17,
    "CSS": 11,
    "JavaScript": 1368,
    "Java": 1608,
    "HTML": 688,
    "TOML": 135
  }
```

The results are also shown in report.html in the project.

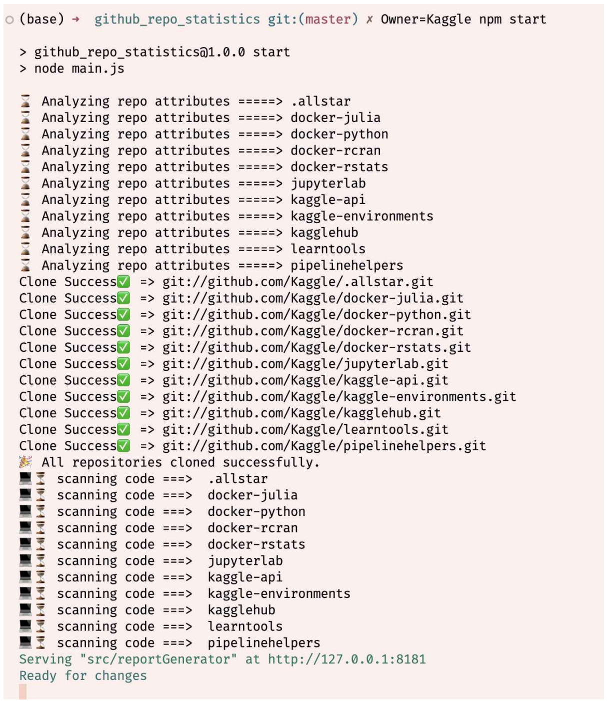
2.3 Optimization

2.3.1 Use promisify to change exec to a Promise.

```
const execPromise = promisify(exec);
```

Final Report

Process notice in terminal

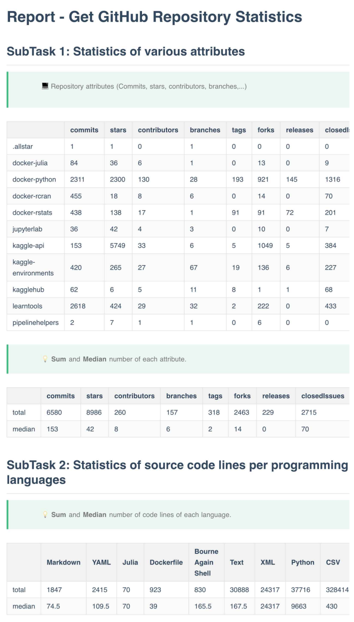


```
○ (base) ➜  github_repo_statistics git:(master) ✗ Owner=Kaggle npm start

  > github_repo_statistics@1.0.0 start
  > node main.js

  ⌛ Analyzing repo attributes =====> .allstar
  ⌛ Analyzing repo attributes =====> docker-julia
  ⌛ Analyzing repo attributes =====> docker-python
  ⌛ Analyzing repo attributes =====> docker-rcran
  ⌛ Analyzing repo attributes =====> docker-rstats
  ⌛ Analyzing repo attributes =====> jupyterlab
  ⌛ Analyzing repo attributes =====> kaggle-api
  ⌛ Analyzing repo attributes =====> kaggle-environments
  ⌛ Analyzing repo attributes =====> kagglehub
  ⌛ Analyzing repo attributes =====> learntools
  ⌛ Analyzing repo attributes =====> pipelinehelpers
  Clone Success✅ => git://github.com/Kaggle/.allstar.git
  Clone Success✅ => git://github.com/Kaggle/docker-julia.git
  Clone Success✅ => git://github.com/Kaggle/docker-python.git
  Clone Success✅ => git://github.com/Kaggle/docker-rcran.git
  Clone Success✅ => git://github.com/Kaggle/docker-rstats.git
  Clone Success✅ => git://github.com/Kaggle/jupyterlab.git
  Clone Success✅ => git://github.com/Kaggle/kaggle-api.git
  Clone Success✅ => git://github.com/Kaggle/kaggle-environments.git
  Clone Success✅ => git://github.com/Kaggle/kagglehub.git
  Clone Success✅ => git://github.com/Kaggle/learntools.git
  Clone Success✅ => git://github.com/Kaggle/pipelinehelpers.git
  🎉 All repositories cloned successfully.
  💻⏳ scanning code ===>  .allstar
  💻⏳ scanning code ===>  docker-julia
  💻⏳ scanning code ===>  docker-python
  💻⏳ scanning code ===>  docker-rcran
  💻⏳ scanning code ===>  docker-rstats
  💻⏳ scanning code ===>  jupyterlab
  💻⏳ scanning code ===>  kaggle-api
  💻⏳ scanning code ===>  kaggle-environments
  💻⏳ scanning code ===>  kagglehub
  💻⏳ scanning code ===>  learntools
  💻⏳ scanning code ===>  pipelinehelpers
  Serving "src/reportGenerator" at http://127.0.0.1:8181
  Ready for changes
```

Generated Report will be automatically opened



Report - Get GitHub Repository Statistics

SubTask 1: Statistics of various attributes

💻 Repository attributes (Commits, stars, contributors, branches,...)

	commits	stars	contributors	branches	tags	forks	releases	closedI
.allstar	1	1	0	1	0	0	0	0
docker-julia	84	36	6	1	0	13	0	9
docker-python	2311	2300	130	28	193	921	145	1316
docker-rcran	455	18	8	6	0	14	0	70
docker-rstats	438	138	17	1	91	91	72	201
jupyterlab	36	42	4	3	0	10	0	7
kaggle-api	153	5749	33	6	5	1049	5	384
kaggle-environments	420	265	27	67	19	136	6	227
kagglehub	62	6	5	11	8	1	1	68
learntools	2618	424	29	32	2	222	0	433
pipelinehelpers	2	7	1	1	0	6	0	0

💡 **Sum** and **Median** number of each attribute.

	commits	stars	contributors	branches	tags	forks	releases	closedIssues
total	6580	8986	260	157	318	2463	229	2715
median	153	42	8	6	2	14	0	70

SubTask 2: Statistics of source code lines per programming languages

💡 **Sum** and **Median** number of code lines of each language.

	Markdown	YAML	Julia	Dockerfile	Bourne Again Shell	Text	XML	Python	CSV
total	1847	2415	70	923	830	30888	24317	37716	328414
median	74.5	109.5	70	39	165.5	167.5	24317	9663	430