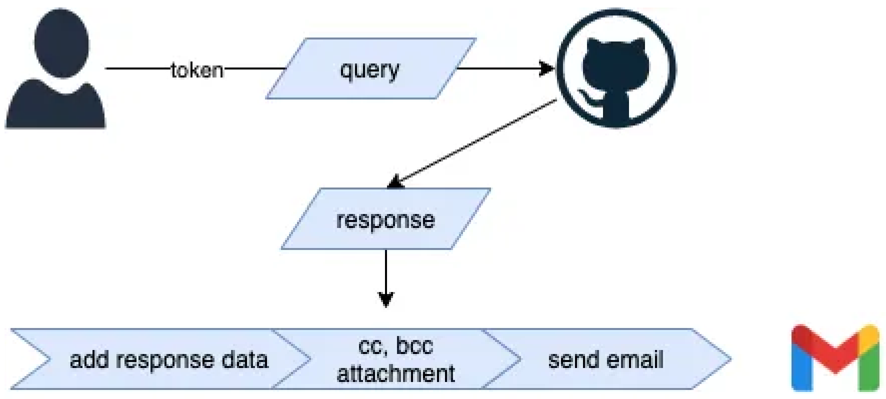
Google Apps Script to GitHub GraphQL

> *How to Connect to GitHub GraphQL via Google Apps Script.*

In this script, I will be going over how to make API query calls to GitHub GraphQL from Google Apps Script. The response data derived from the query will then be emailed.



Project Requirements:

- GitHub Token
- Google Apps Script

Setting Up:

- First, a query must be established. I use GitHub's GraphQL Explorer to configure and tailor the query to my particular needs. The data provided by this tool is live and real-time.
- Second, a personal access token from GitHub with the correct permissions for the query must be created. For example, read-only for repositories, write access to GPG keys, or even full control. For security purposes, ensure that the token has only the required permissions. For reference.
- Third, in Apps Script add the services required. For this project, I added Gmail API and Drive API.

<Script>

```
function gitquery() {

var token = 'X';

const query = ` \
  query { \
  organization(login: "X") { \
    membersWithRole(first:100){ \
      edges{ \
        node{ \
          login \
        } \
      } \
    } \
  } \
} \
  `;

var ql = 'https://api.github.com/graphql';
var response = UrlFetchApp.fetch(ql, {
    method: "POST",
    contentType: 'application/json',
    headers: { Authorization: 'Bearer ' + token},
    payload: JSON.stringify({query: query})
    });

var data = (response.getContentText());
 Logger.log(data);

var body = data.toString();
var recipient = 'test@gmail.com';
var subject = 'Test Email';

var file = DriveApp.getFileById("X");

MailApp.sendEmail(recipient, subject, body, {
     cc: 'cc@gmail.com',
     bcc: bcc@gmail.com,
     attachments: [file.getAs(MimeType.PNG)]
     });

}
```

> *<Script> Explained*

- Enter the token data as a string.

- I created a constructor called query. And pasted my contents from Explorer. Every time in which a new line is created, a backslash must be entered. In order to make more than one query, create another constructor and response variable. In the payload adjust the variable to match the query constructor name. ie: if I named my second query construction query2, then in the payload, I would rename it to read as {query: **query2**}

- The API is called upon using the token to query into GraphQL. The data is entered into the response variable. To get text from response, the data variable is used and logged.
- To send the email, I defined the recipient, subject and the body is stringified.
- To attach a file from drive. I created the file variable in which the DriveAPI will get a file by the ID, this PNG file gets attached into the email.
  For reference