GITHUB - WHAT IS IT AND HOW TO STAND OUT:

First things first: What is GitHub?

Simply put, it is an open source code sharing and publishing service. GitHub allows users to collaborate with others, manage tasks and create projects. The open source element allows it where contributors can collaborate with each other all over the world! One of the most essential functionalities of GitHub is "forking" – copying a repository from one user's account to another. This allows you to take a project that you don't have access to and modify it on your own account. If you make changes you'd like to share, you can send a notification called a "pull request" to the original owner. That user can then merge the changes. In addition to these tools, you want to create issues - so people can see you are communicating with others, and helping the community.

Why is it important?

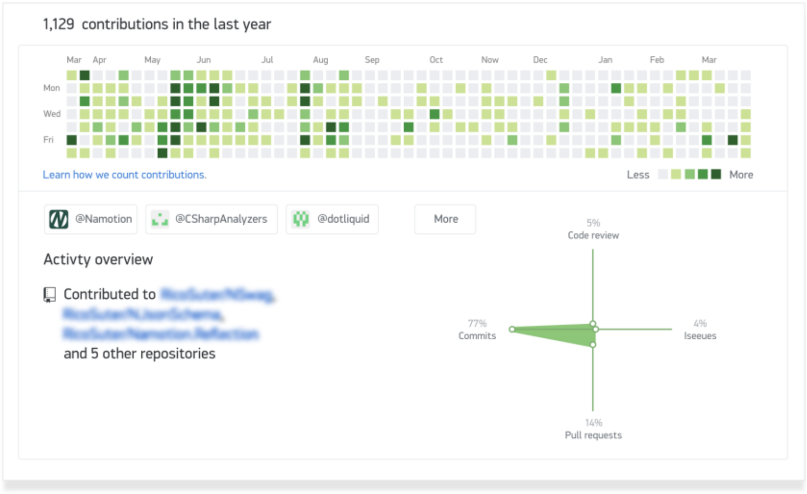
GitHub has become an industry standard. Many companies use GitHub to manage projects that are completed in teams. Employers will also often look for a GitHub profile when you are applying for roles and may ask about your GitHub experience in an interview. Additionally, showcasing your GitHub projects are a great way to show your coding skills and can show personal projects you have worked on and are passionate about.

What makes a 'good GitHub'?

1. **Picture** (of you!)
2. **Bio -** This would contain past and present employers, languages and frameworks you enjoy using, or are currently learning. Include the type of company or kind of projects you would be interested in the future. Add any other relevant links (portfolio, blog, etc.). Here is a sample bio.
3. **Projects and repositories! -** This may seem straightforward but the entire purpose of a GitHub profile is to showcase what you are working on! You can intentionally pin things you want employers to focus on. Here are some project suggestions if you do not have anything on your profile yet:
   a. Websites
   b. Programming exercises ***pro tip - code the algorithms you learn in algorithms classes!***
   c. Games
   d. Apps
   e. Plug-ins
   f. Homeworks from classes you've completed
   g. Internship projects you worked on*
4. **Community debugging and reporting -** GitHub is all about community. You want to show you work well with others and showing how you code and solve problems in teams is essential. If you are someone who reports bugs with code and is able to make suggestions to fix it, this shows your problem solving and team work ability!
5. **Versatility -** Show your ability to use different tech stacks and tools - projects you work on should be different to show just how versatile you are. Make sure to have at least 1 project for each language you code in!
6. **Completeness -** make sure you finish projects you start

    a. When pinning projects only pin what you want to share (quality over quantity!). Also, don't pin projects that you haven't worked in in a while (a project you completed 2 years ago is probably too old)

7. **Contributions -** employers look for contributions. The more you can contribute the better - but also make sure you find the balance between quality and quantity! Contributions show you collaborate, and are working towards a better code for the sake of the project.



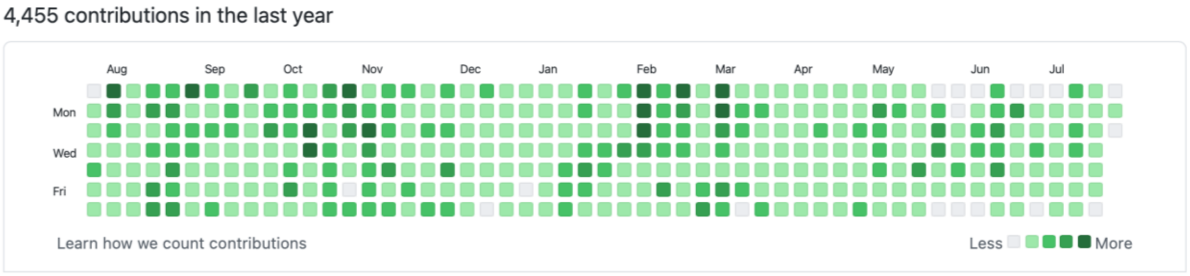
1,129 contributions in the last year

Learn how we count contributions.

Activty overview

Contributed to [redacted]
and 5 other repositories

An even greater contribution graph!



4,455 contributions in the last year

Learn how we count contributions

8. **README files** - Always add a README.md file. Here is an example of a great README file. It should contain information that can be read by anyone viewing the reposition (that means someone with no technical background; if you're not sure your README file is easily understood, ask a friend who does not have a technical background). Most times folks don't have time to review your code, so having a complete README is essential. Your README file should contain:
    a. Description of what the person sees
    b. Technologies, libraries and frameworks
    c. How to install and use
    d. Current state
    e. Link to the live site
    f. Any other important information
9. **CONTRIBUTING file -** You want to explain all the details that explain necessary information that someone would need to contribute to your project this can include:
    a. Contributions you accept
    b. How to submit contributions

10. **Profile setting tips**
    a. Unclick “Keep my email address private” in the email settings of your account so people can easily contact you.
    b. Click “Available for hire” if you’re looking for jobs.
    c. Click “Available for hire” if you’re looking for jobs.

Practical Tips:

- Make sure your repos and descriptions are professional. Something like “Final push!” that may have made sense in class or when working with friends, will likely not be helpful for your job application
- Depending on the roles you are interested in, you may want to highlight more complex projects (a front-end engineer may have different projects than a full-stack engineer). You can find tips for project complexity by reviewing our Technical Project Guide
- Be sure to use the pin feature it essentially brings the projects you are most proud of to the forefront

Sample Profiles:

We’ve provided some sample GitHub profiles in case that provides help (notice that each GitHub URL is short and easy to link [no extra numbers/letters]):

- Alex Paul https://github.com/alexpaul (many contributions, detailed README files, a clear and creative bio!)
- Ashli Rankin https://github.com/Ashlirankin18 (clear README files, good number of repositories, and a nice number of contributions)

How to use Github in development/work?

- When dealing with a new issue in a project, create a new branch, fix the issues, ask colleagues for review then merge your own branch into the main branch
- Testing after push on github https://coderefinery.github.io/testing/continuous-integration/
- Make sure to name your releases in an organized way here is a release Naming Convention

GitHub Help

If you would like help with something you are working on, the GitHub forums are a great place to ask questions and get responses from developers around the world (Stack Overflow is another great resource for getting programming help - you should become familiar with using this website to search for answers to coding problems as well).

After you have signed up for a GitHub account, you will have access to the Learning Lab where you can complete lessons that teach you how to use the different features of GitHub.
Learning paths and modules for GitHub by Microsoft.

If you are looking to contribute to open source projects check #good-first-issue.

Notes:

- Internship Projects: Before making your GitHub project that you worked on at your internship public, be sure to check with your manager(s) about this to ensure you aren't violating any confidentiality agreements
- You may want to set up a GitHub Action for automatically syncing LeetCode submissions to a GitHub repository https://github.com/joshcai/leetcode-sync

This guide was built using the following resources:

https://blog.pitchme.co/2020/07/09/7-tips-improve-your-github/
https://inspirezone.tech/checklist-for-every-open-source-repository-maintainer/
https://techcrunch.com/2012/07/14/what-exactly-is-github-anyway/
https://techbeacon.com/app-dev-testing/what-do-job-seeking-developers-need-their-github
https://handbook-for-cuny-hunter-cs-students.webnode.com