# Security Details

STS took a lot of sweet talking before they approved our Github use, so we want to be on our best behavior (in addition to general data security concerns). As such, the following practices should be followed when using Github (expanded from the summary on the first page).

## Account Setup

Use your tn.gov email to create a personal account (can we use our existing account and add our tn.gov email?). Enable two-factor authentication for increased security. You will be able to contribute to your own teams repos using your own account. [section on hosting your own repos on your own account]

## Team Access

Admins for the TDH-CEDEP organization must maintain user access to the correct teams. This includes adding new members to the correct team, removing team members when they leave their positions, and performing regular user audits.

## Do you even need Github?

If you just want version control on a project, you can just use Git locally. Github is great for sharing projects and working with outside parties, but if the process is internal, there might not be a point to hosting it on Github at all.

## Public vs Private Repositories

The default choice for repositories should be to make them private. More public repos are just more ways for things to accidentally get published to the public. That being said, there are reasons why you might want a repo to be public. Public repos are great for sharing data and code with other parties, and even having them suggest improvements. You can even publish data packages that are useful for TDH employees who might not have their own Github accounts.

It is important to note, that a repository being private should not affect the sort of information published there. PHI, passwords, etc. are not allowed in any repo, public or private. [should there be a middle tier of data that can be in private repos, but not public ones , e.g. network paths?]

## Sensitive Data

The following data are not allowed to be published to any Github repository:

- PHI
- Non-public data (preliminary, data needing suppression, etc.)
- Usernames/Passwords/Credentials
- API tokens

In order to prevent this data from getting into Github, remove it from your codes and become used to excluding it from your work without consideration for if it will end up in Github. It is especially easy to leave notes or comments in code about test cases or outliers that might need to be deleted before saving a file.

You can also use gitignore functionality to hide certain files from Git, which can help when you need data for a code to run on your machine, but you don't want it in the version history that goes in Github.

[Consider a middle tier of info that can be in private but not public repos]

### Removing Sensitive Data from a Repo

If sensitive data gets pushed to Github, we need to take it down as fast as possible. Follow the following steps:

1. Notify an admin
2. Make the repository private (if it isn't already)
3. Edit the repo to remove the sensitive data
4. Edit the version history to remove the sensitive data [develop guidance because this can be complex]. Note information needed for follow-up investigation (time published and removed, author, file source, etc.)

5. Notify anyone else who needs to know (e.g., HIPAA protocol)
6. Determine steps to make the repo public again (if appropriate)