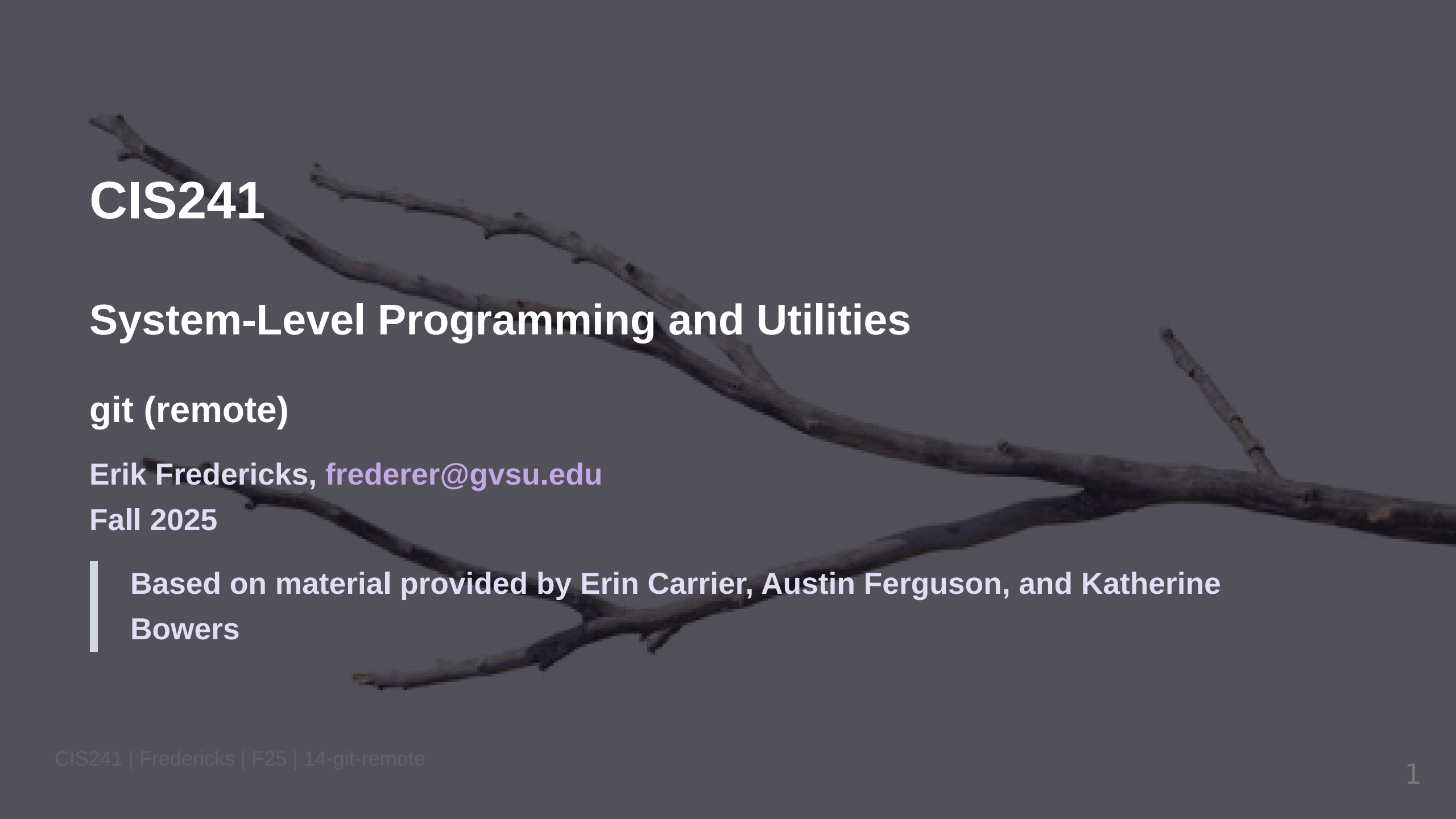




CIS241

System-Level Programming and Utilities

git (remote)

Erik Fredericks, frederer@gvsu.edu

Fall 2025

Based on material provided by Erin Carrier, Austin Ferguson, and Katherine Bowers

Everything we've done so far has been local

But what about remote? Off your
computer? Somewhere else?

- Somewhere ... out there?!?!



git remotes!

Meaning, somebody is running a git server somewhere

- E.g., GitHub, GitLab, bitbucket, etc.
 - Or, you can run your own git server if you want

Let's say you have a local repo and want to tie it to a remote server:

1. Go to your server (i.e., GitHub) and create a new **empty** repo
 - No README, license, nothing. That will potentially induce a conflict! Add that later
2. `git remote add origin remote_url`
3. `git push --set-upstream origin main`

No local or remote (i.e., the easiest way)

1. Go to your server (i.e., GitHub) and create a new repo with your favorite license and a README file
2. `git clone remote_url`
3. Then, your usual workflow (edit, add, commit, `push`)

push?

Update the remote with your commit!

- `git push`

However, since we're remote, check if anybody else made changes!!

- `git fetch` - see what changes are made on remote
- `git merge` - merge those changes with mine
- `git push` - push it up cleanly



A new workflow!

When working with remotes, the easiest way to avoid merge conflicts are to always `pull` before you make a change

1. `git pull`
2. (Make your changes and stage)
3. Do the `fetch` and `merge` if necessary
4. Then do your pushing

What if we end up with a merge conflict?

That's the topic of the next slide deck, however one easy way to deal:

1. Clone the repo to a new folder
2. Merge your changes into that new folder
3. Push those changes in that new folder
4. Either that new folder is your new local repo or you `pull` down the changes to your existing local repo

Things to not do

- `rebase`
- Commit from a detached head (checkout a previous commit then make a commit)
- Reset to a commit
- Amend a commit once it's been pushed
- **DO NOT REWRITE HISTORY IN ANY WAY**
- Also, push a secrets/password/API key file to GitHub without anonymizing it
 - GitHub deletions **never** go away
<https://www.techtarget.com/searchsecurity/news/366599096/Researcher-says-deleted-GitHub-data-can-be-accessed-forever>

Right then, how do we use GitHub for a remote

Changes in past years means that you cannot use a username/password to update your remote on the terminal!

- You need to use SSH keys
- Meaning when you clone, you need to use the SSH link, **not** the HTTPs link!

<https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account>