CS 240: GitHub Overview Transcript

[00:00:00] The next thing we need to learn about is GitHub.

[00:00:03] GitHub is a website that hosts get repositories.

[00:00:09] So what that means is you can create an account on GitHub and, and we can just open up GitHub in a, in a browser. So just go to github.com.

Start visual description. The professor demonstrates how to create an account on GitHub by navigating to github.com and logging in. The screen shows the GitHub homepage with options to sign in or sign up. End visual description.

[00:00:19] I'm already logged in. I already have an account. I'm already logged in.

[00:00:22] And so it just took me to my account. But the first time you go to get hub, um of course, you won't have an account.

[00:00:29] And so the first step would be to create an account. So just go through the, the account creation process and once you're logged in, um what it shows you here on the left is it shows you a list of all your, your get projects, your repositories that you've created.

[00:00:44] And so in my case, um I've created a bunch of repositories.

[00:00:49] So if I scroll down through here, you know, a lot of these repositories are things that I created.

[00:00:54] The other thing you can do is you can share your repositories um with other people or give them access to your repositories.

[00:01:00] And so there's a lot of repositories in my list that other people have given me access to their repositories.

[00:01:06] So anything that doesn’t start with my, my username I didn’t create, but somebody gave me access to.

[00:01:13] And so um the basic operation you can do in, in GitHub is just create a new repository and which we’ll, we’ll talk about soon, but you can create a new repository.

[00:01:28] And now your repository is up in the cloud.

Start visual description. The professor demonstrates creating a new repository on GitHub. The screen shows the process of naming the repository, adding a description, and setting the repository to public or private. End visual description.

[00:01:32] And the advantage of that is that it gives other people access to the repository. So, if you’re working in a group or a team of people, um it’s a lot more convenient when your GIT repository is, is hosted in the cloud so that everybody can connect to it from their, their personal development computer.

[00:01:53] So one advantage of GitHub is that collaboration with others, another advantage is backup.

[00:02:00] Now, earlier, we said that um a, a reason to commit your, your changes in your repository frequently was um that you could, you know, basically back up your, your changes, back up your files by doing that.

[00:02:13] But if your repository is just local to your computer in your doc git folder, well, you know, it’s a backup of a sort, but if your, your laptop dies or your hard drive or SSD dies, uh you’re still going to lose your code.

[00:02:26] So the nice thing about using GitHub is that your files are actually backed up on a remote system so that if your, your computer goes down, those files are still safely housed on the, on the GitHub site.

[00:02:40] So backup um is another advantage of GitHub. A third advantage is GitHub provides a platform where you can share your work with others.

[00:02:49] So if you keep your, your repositories and projects up in GitHub, then it's easy to share those with other people as far as promoting uh the work that you've done.

[00:02:58] For example, if you're looking for a job and you want to show employers some of the cool projects that you've done.

[00:03:05] If you put your projects in GitHub, then it's easy to give them a link to the, to your project, so they can look through your code and be impressed and hopefully hire you.

[00:03:13] And so those are some of the advantages of, of using GitHub.

[00:03:18] So everything we've done previous to this was just working on a local repository that was just on my, my personal development computer.

[00:03:27] So now we're going to elevate our projects up to the cloud.

[00:03:32] Now, once you've created a project in the cloud in the cloud, let's assume that you want to work on your project with a bunch of other people.

[00:03:41] And so let's look at how the workflow if you're using what the workflow looks like as you collaborate with others.

[00:03:49] So in this case, let's assume that we've, we've got a repository and get a and each person that's using that repository is going to have a local copy of the repository. So, the first thing you do, if you're going to use this repository is you run a command called Get clone.

[00:04:11] And when you clone a repository, what that does is it creates a local copy of that repository on your computer.

Start visual description. The professor demonstrates cloning a repository using the Git clone command. The screen shows the terminal where the command is typed, and the local copy of the repository being created on the computer. End visual description.

[00:04:19] So that's kind of the workflow is you would create the repository through the GitHub web interface.

[00:04:23] And then on your own personal computer, you would, you would run the GIT clone command, and you would type in the URL of the repository and that would create a copy of that repository on your local machine.

[00:04:35] And at that point, you can start working with the code making changes to it.

[00:04:40] So in this slide, Paul is, is working on a GitHub project. So, he, he's already cloned the project. So, he has a local copy of it of the repository. And then let's just assume that he made a change to a file named a dot Js.

[00:04:56] And what he's going to do is having made a change to that file, he's going to commit has changed to the repository.

[00:05:04] Now, when Paul commits that change to that file, it only commits it to the local copy of the repository on Paul's machine, it doesn't actually push that change all the way up into GitHub.

[00:05:16] So as a developer, I can be on my local machine, I can make changes to the repository.

[00:05:20] I can commit those changes, but just know that those changes are only local um until um you explicitly share those changes with your teammates.

[00:05:31] And so there's another commanding G called push.

[00:05:34] So eventually Paul is going to run and get push command.

[00:05:38] And when he does that, it’s going to take all the changes he’s made locally to the repository. It’s going to push those changes up into github’s copy of the repository.

Start visual description. The professor demonstrates pushing changes to GitHub using the Git push command. The screen shows the terminal where the command is typed, and the changes being uploaded to the GitHub repository. End visual description.

[00:05:46] And once he pushes his changes up to GitHub that makes his changes visible to all the other people he’s working with.

[00:05:54] And so having pushed the changes, then even if, if Paul deletes his whole folder, he just, he just deletes the whole project, even the dot Get folders deleted, it’s just all gone.

[00:06:06] He can still get his code back by running a get clone command again.

[00:06:09] And so if he did that, it would bring all the code down from GitHub and recreate the repository on his machine. And so, um that just kind of emphasizes the backup nature of what you get with GitHub.

[00:06:25] Now, let’s say Paul’s working with uh somebody named Ringo, he can clone the repository on his machine.

[00:06:33] And then what he can do occasionally is he can run and get pull command.

[00:06:39] So if you’re working with other people, occasionally, you need to run this command, get pull.

[00:06:44] And what that means is go to the GitHub repository and bring down all the changes that other people have made to the repository.

[00:06:51] Since the last time I pulled So, even though he's cloned, the repository, he won't see all the changes that other people are making to the repository until he does a pull.

[00:07:02] And so um in the previous slide, if you'll recall, Paul did a get push.

[00:07:06] So he pushed his changes up into, get the opposite of that command is the pole command which brings down changes other people made to my machine.

[00:07:15] And so get push, get pull or are things that you would be doing frequently.

[00:07:22] Now, in this case, let's say Paul makes Paul and Ringo are working together and Paul makes a change to this file.

[00:07:30] He pushes that change up to get hub and then at some point, Ringo is going to pull, pull those changes and then he can make a change to that file locally and then he can go ahead and push his changes and then Paul would pull those changes and so you're constantly pushing and pulling uh changes just so you can stay in sync with, with your teammates.

[00:07:53] And so this this slide just kind of shows that process that one of the team members is going to push changes and the others are going to pull.

[00:08:01] Now, how often you push, and pull is up to you. But um you would want to push your changes to GitHub just as soon as it makes sense for other people to have access to those changes.

[00:08:10] You don't want to wait a long time.

[00:08:12] Um Now you might be making commits locally to the repository.

[00:08:16] But, but you do want to push those up so other people can see the work that you're doing and have access to it.

[00:08:21] And then you also want to do a poll frequently.

[00:08:24] So at least every day, every time you work on the project, you probably want to do a poll.

[00:08:28] Now, there may be times when you don’t do that for a while for some reason, but you want to do push and pull frequently.

[00:08:34] Now, one thing to be aware of when you’re working with other people through GitHub is that sometimes you have what’s called a conflict.

[00:08:41] So in in this case, let’s say Paul and Ringo are, are still working together.

[00:08:46] And in this case, Paul makes a change to the A dot js file and then he pushes that change up to GitHub.

[00:08:55] Now he did that after the last time, Ringo pulled changes.

[00:08:59] So at this point, Ringo is unaware of these, this changes that Paul made.

[00:09:04] But let’s suppose locally in his copy of the repository Gringo also changes the file A dot Js and then he tries to push that change up to GitHub.

[00:09:15] Well, at that point, get hub or G is going to notice that Ringo doesn’t have the latest copy of the code.

[00:09:24] So G is smart enough to realize that that Ringo is trying to push code, but he doesn’t have the latest changes that other people have made.

[00:09:33] And so it’s going to prevent Ringo from, from pushing his change.

[00:09:37] And so he’s going to basically require that uh Ringo do a poll.

[00:09:44] Now, when Ringo does a pull then gets going to notice that there’s a conflict, Paul made a change to the same file that Ringo changed.

[00:09:52] And so gets going to notice that, hey, both of you guys changed the same file.

[00:09:57] And so that's where the conflict comes in.

[00:09:59] And so this is called a merge conflict.

[00:10:02] And so when we have that conflict, essentially, Ringo is going to have to edit this file a dot Js manually and figure out OK, multiple people are changing this file, what should it be? And so, Ringo is going to have to, to fix this file and make sure that it includes his changes and Paul's changes and then once he's fixed it, then he can go ahead and push the file up into GitHub.

[00:10:24] Now this, this isn't something that's going to be very relevant in this class because you're, you're not working in groups, but just realize that when you do work with other people, uh sometimes the changes others make will conflict with the changes that you're making and get will, will detect that and it will, will ask you to resolve the situation by manually fixing the file to be what it should be.

[00:10:45] And most uh I Ds, all I Ds anymore, pretty much have a nice interface, a graphical interface that you can use to compare the two versions of the file and figure out which pieces of which file you want to keep and you can kind of cobble together the, the final file that you actually want to keep and then you can commit and push um that file up to.

[00:11:10] So merge conflicts are a big deal when working with teams.

[00:11:14] But you'll, you'll experience that more later when you actually do work with teams either in other classes or at work.