# Operating Systems Lab: More on Git

By this lesson, you probably know the following:

1. Installing and setting up Git.
2. Create your own GitHub account.
3. Create a public repository on GitHub
4. Create your own repository on GitHub and use it as your local repository.
5. Create your local repository and push it on your global GitHub repository.

In today’s lesson, we will cover:

1. Ignoring Files
2. Branching
3. Merging
4. Global Repository
5. Forking

Let’s get started

## Ignoring Files

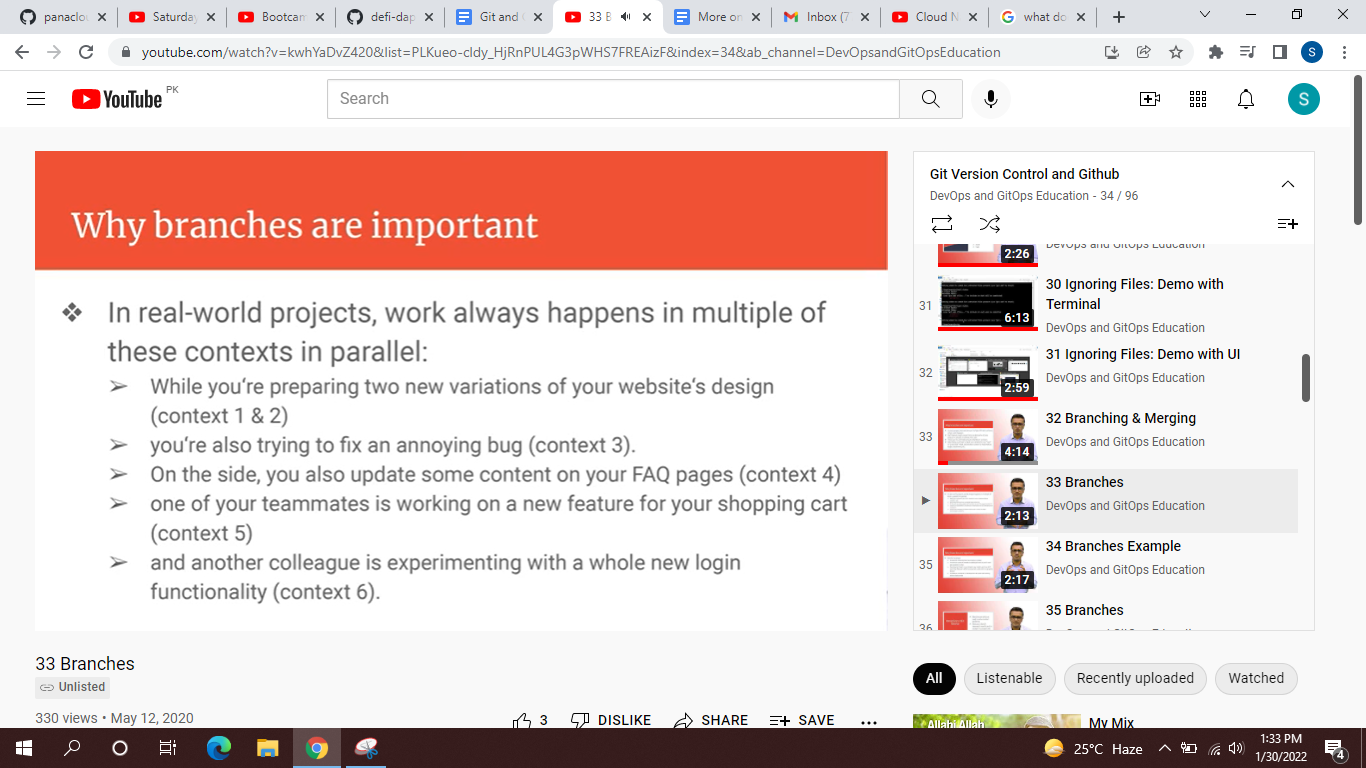
Sometimes you might not want to include files in your commit. Those files are usually OS dependent and each person installs them according to their OS. The files could be DS files, Node modules, builds and logs.

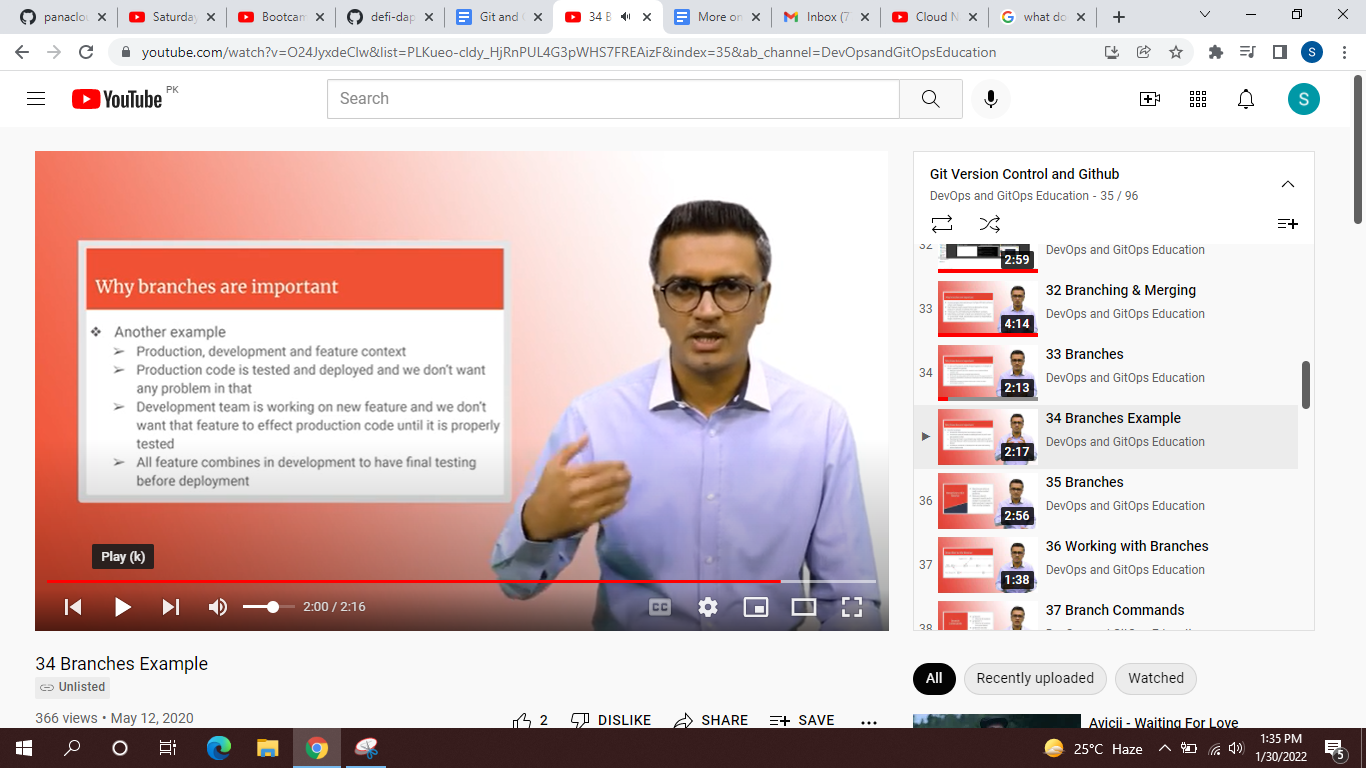
To ignore such files in Git, perform the following steps:

1. Create a file called “.gitignore” in your project directory.
2. Open the files using Notepad and type the file or folder names that you want to ignore.
   1. File name: sampleFile.txt
   2. Files with similar extensions: \*.xlsx
   3. Folders: sampleFolder

## Branching and Merging

Branching is useful in maintaining different features and different cycles of software development. The master branch will always consist of stable code.





To work on a branch, check the branches first. By default, your local respiratory system is on the “master” branch.

1. To check the branches, type: git branch
2. To create new branch in your repo: git branch branch\_name
3. To switch to the newly created branch: git checkout branch\_name
4. To check difference between two branches: git log master..dev
5. To go back to master branch: git checkout master
6. To merge your branch into master branch: git merge dev

Make sure to switch to master branch before merging if you want to merge new\_branch to master branch.

## Global Repository

You can use GitHub, BitBucket and GitLab. It is your choice. To fetch the repository, copy/paste the command from the GitHub website. The commands used in this step are:

1. Push
2. Clone
3. Fetch
4. Merge
5. Pull (perform fetch and merge together)
6. Remote

There are two ways to access remote repo: your own repo or any other public repo. Let’s look at each step in detail:

### Pushing on your own global repo and using master branch

1. Create your own GitHub repo.
2. Go to the required folder and type: git clone URLofremoterepo
3. Navigate to the cloned repo using cd command.
4. Check the status
5. Perform some new operations in that repo.
6. Commit the file (status, add, status, commit, status)
7. After final status, you will see the message that your branch is ahead of the global master branch. For more details, check the log using git log.
8. Now push it using git push. It will ask for username and password.
9. Now check the status again.

### Fetching updates from your own global repo

In this step, you will learn how to keep your local repo updated.

1. Make some changes in global repo from your GitHub account.
2. Go to cmd and check status. You will see that there is no difference on local repo. That is because the local repo is not aware of changes on the server.
3. To fetch those changes use the git fetch command. Check the status again. You will see that there is no change. That is because the fetch command only fetches the changes. It does not apply the changes. For this, you will use the git merge command.
4. Instead of using these two commands, you can simply use git pull.

### Another method to public to global repo

You can either clone your remote repo, or you can connect with it while pushing the changes. For this step, you will only create the global repo, but you won't clone it. Instead you will perform the following steps:

1. Create a local repo and do some work in it.
2. Check if there is any connection to global repo: git remote -v
3. Connect to remote repo using: git remote add yourRepoName URLofRemoteRepo
4. Now check the remote connection again. You will see the URL.
5. Next, push it using: git push -u youRepoName branchName
   1. Example: git push -u hello master
   2. Flag -u establishes the connection between local and remote repo.

### Pushing on your own global repo and using another branch

1. Let's assume you are connected to the remote repo.
2. Create another branch (branch command), switch to that branch (checkout command), and perform some operations.
3. Check the status of your local branch (status), switch to local master, merge changes, and push it to global repo using: git push -u origin master
4. Alternatively, you can push that local branch too instead of merging it with the local master: git push -u origin localBranch

## Your Task

Attempt the quiz!

Practice what we have covered in today’s class.

Check the project sheet for your tasks.

Try cloning my repository and pushing some changes as a separate branch.

## Industry practices you may want to explore

stash, short-lived and running branches, git workflow, forking and markdown.