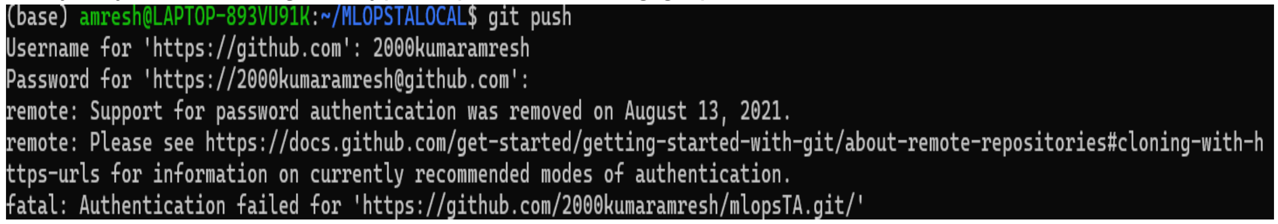
# Chapter 1: - Some more git commands and instructions

Many of you are facing this type of problem during git push

Many of you are facing this type of problem during git push



```
(base) amresh@LAPTOP-893VU91K:~/MLOPSTALOCAL$ git push
Username for 'https://github.com': 2000kumaramresh
Password for 'https://2000kumaramresh@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/get-started/getting-started-with-git/about-remote-repositories#cloning-with-h
ttps-urls for information on currently recommended modes of authentication.
fatal: Authentication failed for 'https://github.com/2000kumaramresh/mlopsTA.git/'
```

The solution is : You need to use a **Personal Access Token (PAT)** or **SSH key** for authentication. (SSH key authentication material has already been provided)

1. **Generate a Personal Access Token (PAT)**
   - Go to GitHub's Personal Access Tokens page.
   - Click on "Generate new token".
   - Give your token a name and select the appropriate scopes (e.g., repo for full control of private repositories).
   - Click "Generate token".
   - Copy the token—you won't be able to see it again. (For safety, just copy it and save it somewhere else; otherwise, in the worst case, you will need to generate it again.)
2. **Use the PAT Instead of a Password**
   - When you run git push, and you're prompted for a username, enter your GitHub username.
   - When prompted for a password, paste the PAT instead of your GitHub password.
3. **Optionally: Store Credentials (Optional)**

   You can store your credentials to avoid entering them every time by using Git's credential helper:

   ```
   git config --global credential.helper store
   ```

After running this command, the next time you enter your username and PAT, Git will store them in plain text in your local machine's Git config, so you ***won't have to re-enter them.***

**NOTE: Please create a separate directory for this course (name it according to your convention). Ensure that you manage your Git files, branches, and pull requests carefully, as improper handling may lead to issues as we progress in this course and could affect your grading.**

## GITHUB BRANCHES

You can read in detail at: [About branches - GitHub Docs]

- We use a branch to isolate development work without affecting other branches in the repository.
- Each repository has one default branch(named as main), and can have multiple other branches.
- You can merge a branch into another branch using a pull request.
- Unless you specify a different branch, the default branch in a repository is the base branch for new pull requests and code commits.
- You can change the default branch for an existing repository.

**To create and delete branches from GitHub(REPO), follow this link**:

Creating and deleting branches within your repository - GitHub Docs

**Commands that you can use from the terminal (To create branches and push files)**

1. Create a New Branch:

```
git branch ML_assignment2
```

2. Switch to the New Branch:

```
git checkout ML_assignment2
```

3. Create and Switch to the New Branch in One Command:

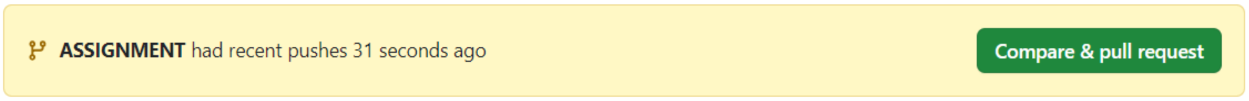
```
git checkout -b ML_assignment2
```

4. Push the New Branch to a Remote Repository(GitHub):

`git push -u origin ML_assignment2` **or** `git push --set-upstream origin ML_assignment2`

5. Go to GitHub, and in the repository that you pushed,
   a. Click on "Compare & pull request" on GitHub and then create a pull request.

GitHub compares your branch (ML_assignment2) with the main branch. You can add a title and description for the pull request, explaining what changes were made and why.


ASSIGNMENT had recent pushes 31 seconds ago

- **b.** Review: Collaborators can review the pull request, add comments, and request changes if needed
- **c.** Merge the Pull Request:
    - i. Once the pull request is approved, it can be merged into the main branch.
    - ii. The changes in the new branch are then incorporated into the main branch.
- **d.** Then confirm merge. (now you will see all the files in the main branch.)

**Commands To delete files and branch**

**A. Delete a file from the branch**

- **a.** Switch to the Branch (if you're not already on it): git checkout branch-name
- **b.** Delete the Files: rm path/to/file

  Example: **rm example.txt**

- **c.** After deleting the files, you need to stage the changes: git add -u (or you can explicitly add the deleted files: git add path/to/file)

Commit the changes to the branch: git commit -m "Delete unnecessary files"

Push the Changes to the Remote Repository: git push origin branch-name

**If you want the file to be deleted in the main branch as well, you have a few options:**

**Option 1:** Delete the File Directly in **main** (the same way as above.)

**Option 2:** Merge the Changes from the Other Branch.

- git checkout main
- git merge branch name
- git push origin main

(you may need to resolve some conflicts due to an unmerged file.)

**B. Delete a branch**

1. Ensure you are not currently on the new-branch(let's say ASSIGNMENT) branch:

git checkout main

2. Delete the Local Branch:

git branch -d ASSIGNMENT

- If the branch has unmerged changes and you want to forcefully delete it, use:

```
git branch -D ASSIGNMENT
```

3. Deleting a Remote Branch(i.e. github)

```
git push origin --delete ASSIGNMENT
```

# Chapter 2: Docker

Reading material on: - [Docker Curriculum](#)
**Docker installation guide:** [Install Docker Engine on Ubuntu](#)
Follow Install using the apt repository method (easy to install from this).
**Or** you can install **Podman** if you face any difficulties while installing **Docker (**but try to install **Docker):** `sudo apt-get -y install podman`

**Step-by-Step Guide to Install and Verify Docker on Ubuntu**

**Step 1**: Update the Package Index**:** `sudo apt-get update`
Description: This command updates the local package index with the latest information from the repositories. It ensures that you install the latest versions of packages.

**Step 2:** Install Required Packages

```
sudo apt-get install \
ca-certificates \
curl \
gnupg \
lsb-release
```

Description: This command installs the necessary packages for adding a new repository:

- **ca-certificates:** Ensures that your system can verify the authenticity of SSL certificates.
- **curl:** A tool to transfer data from or to a server, used here to fetch Docker's GPG key.
- **gnupg:** Provides tools for secure communication and data storage, essential for managing GPG keys.
- **lsb-release:** Provides Linux Standard Base (LSB) information about the distribution.

**Step 3:** Add Docker's Official GPG Key
`curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor` (space used)`-o /usr/share/keyrings/docker-archive-keyring.gpg`
Description: This command downloads Docker's GPG key and stores it in a keyring file. The -fsSL options make curl fail silently on server errors, follow redirects, and use SSL.

**Step 4:** Set Up the Docker Repository

```
sudo bash -c 'echo "deb [arch=$(dpkg --print-architecture)
signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" >
/etc/apt/sources.list.d/docker.list'
```

Description: This command adds Docker’s official repository to your system’s package Sources.

It specifies:

- **arch=$(dpkg --print-architecture):** Gets the architecture of your system (e.g., amd64).
- **$(lsb_release -cs):** Gets the codename of your Ubuntu release (e.g., focal for 20.04).

The command writes this information to a new file named docker.list in the **/etc/apt/sources.list.d/ directory.**

**Step 5**: Verify the Docker Repository

```
cat /etc/apt/sources.list.d/docker.list
```

Description: This command displays the contents of the docker.list file to verify that the Docker repository was added correctly.

**Step 6**: Update the Package Index Again

```
sudo apt-get update
```

Description: This command updates the package index again, now including the Docker repository.

**Step 7**: Install Docker Engine

```
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

Description: This command installs Docker Engine and related packages:

- **docker-ce:** The Community Edition of Docker.
- **docker-ce-cli:** The command-line interface for Docker.
- **containerd.io:** The container runtime used by Docker.

**Step 8**: Start Docker

```
sudo systemctl start docker
```

Description: This command starts the Docker service on your system.

**Step 9**: Enable Docker to Start on Boot

```
sudo systemctl enable docker
```

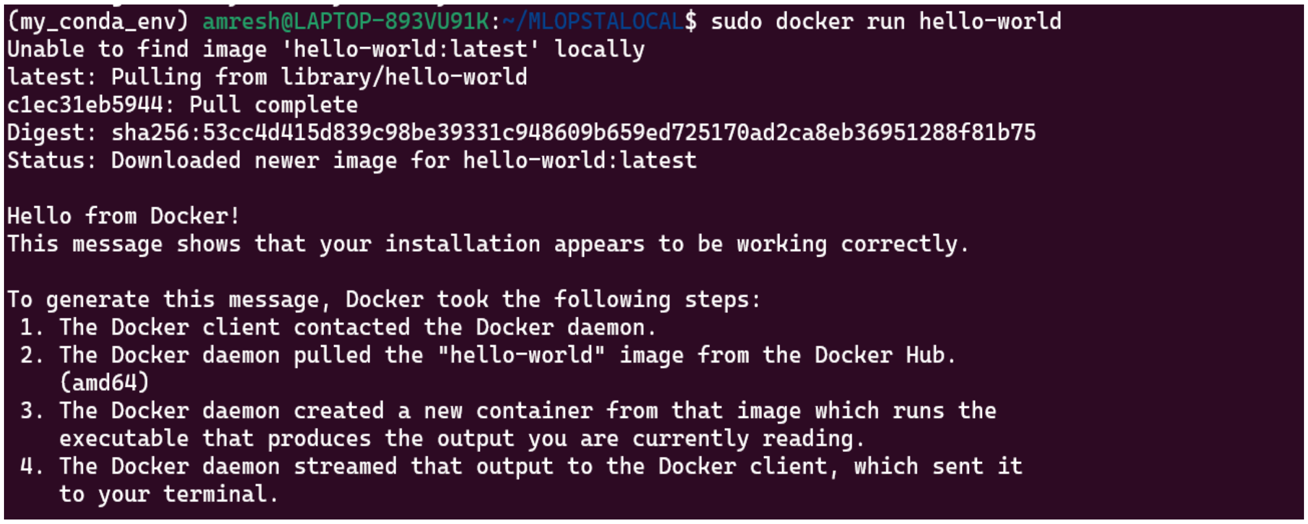
Description: This command ensures that the Docker service starts automatically when the system boots up.

**Step 10**: Verify Docker Installation

sudo docker run hello-world

Description: This command runs a test Docker container that prints a message indicating that the Docker is installed and functioning correctly. If Docker is installed properly, you will see a message from Docker confirming that it's working.

**The output will be like this:**



```
(my_conda_env) amresh@LAPTOP-893VU91K:~/MLOPSTALOCAL$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:53cc4d415d839c98be39331c948609b659ed725170ad2ca8eb36951288f81b75
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.
```

**If you do not get output like** Hello from Docker!
Then go to [hello-world - Official Image | Docker Hub]()
**Copy the docker pull code(docker pull hello-world) and paste it on the terminal**
**Then again, run** docker run hello-world