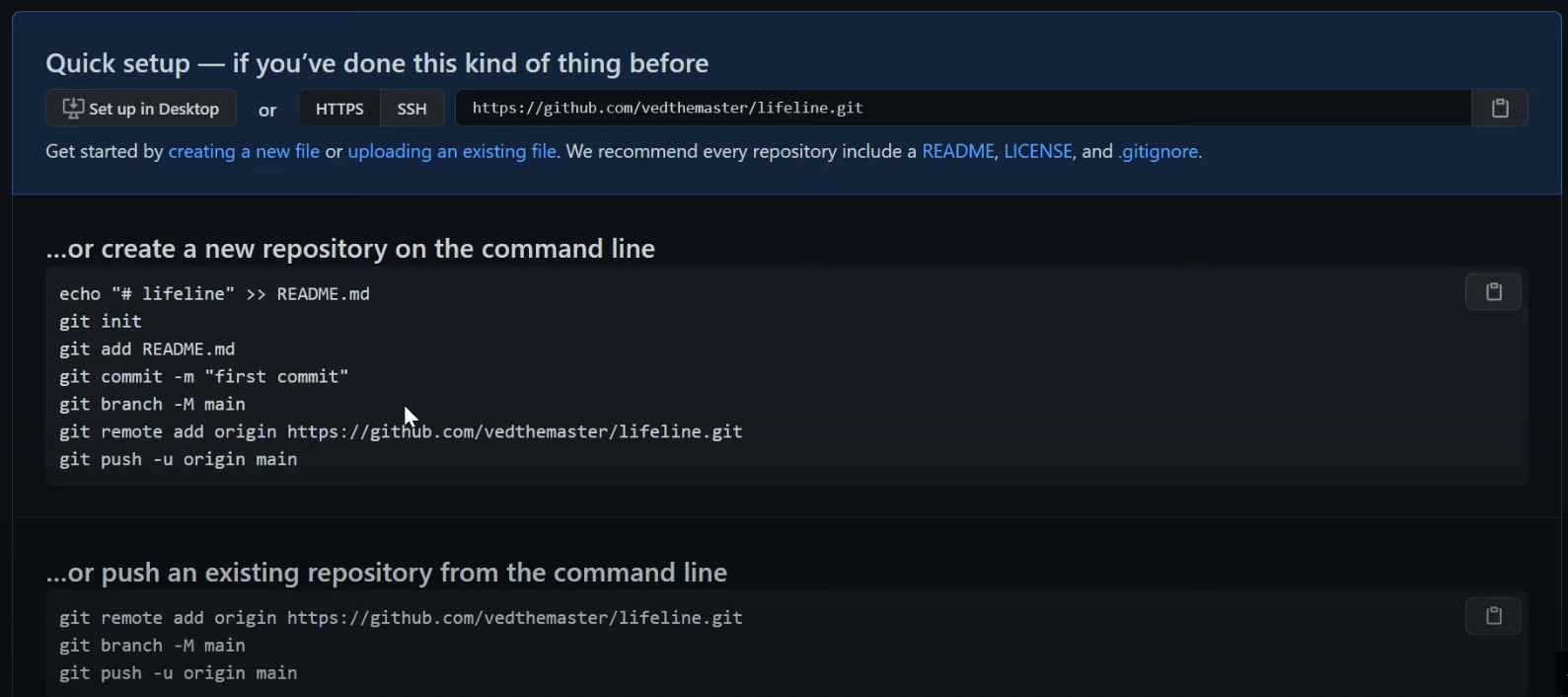
**GIT Commands and Errors**



**Strong recommendation before creating any repository:**

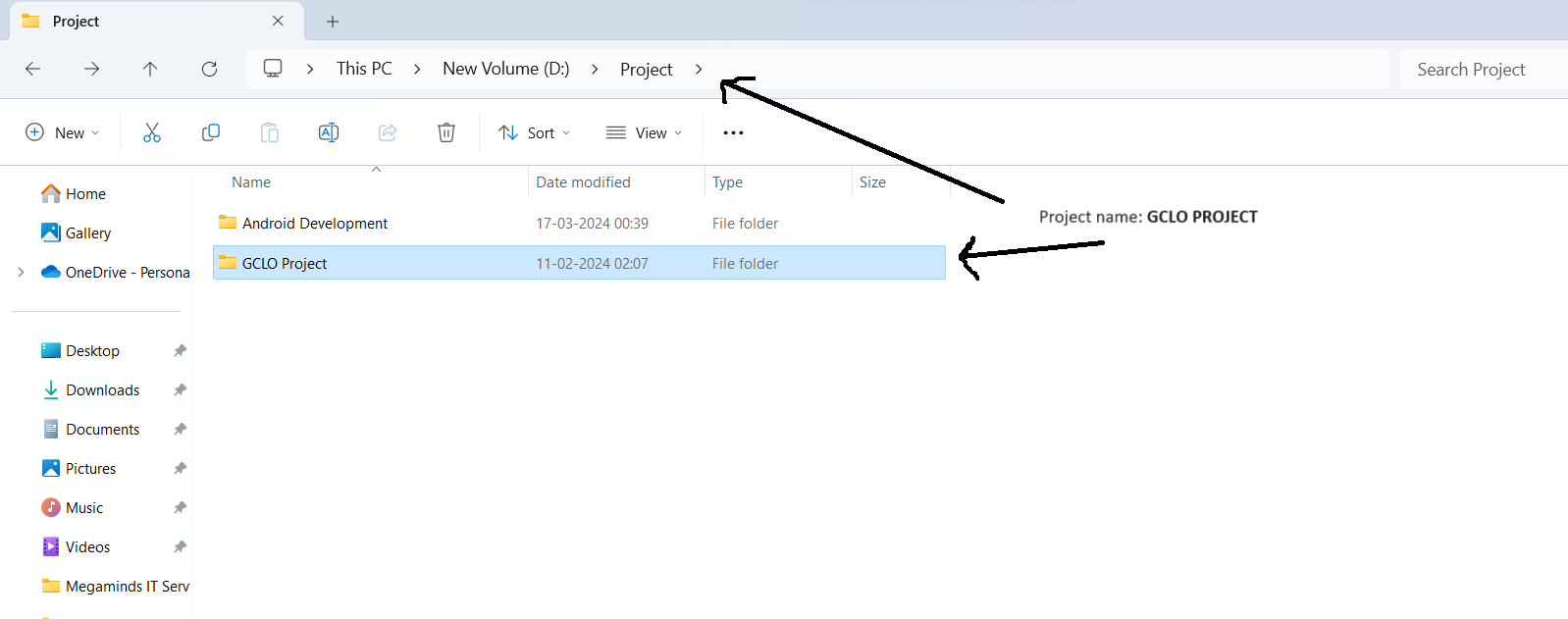
**Create a separate folder for each project**

**Example: Project name = Job Board**

**Create folder “Job Board”**

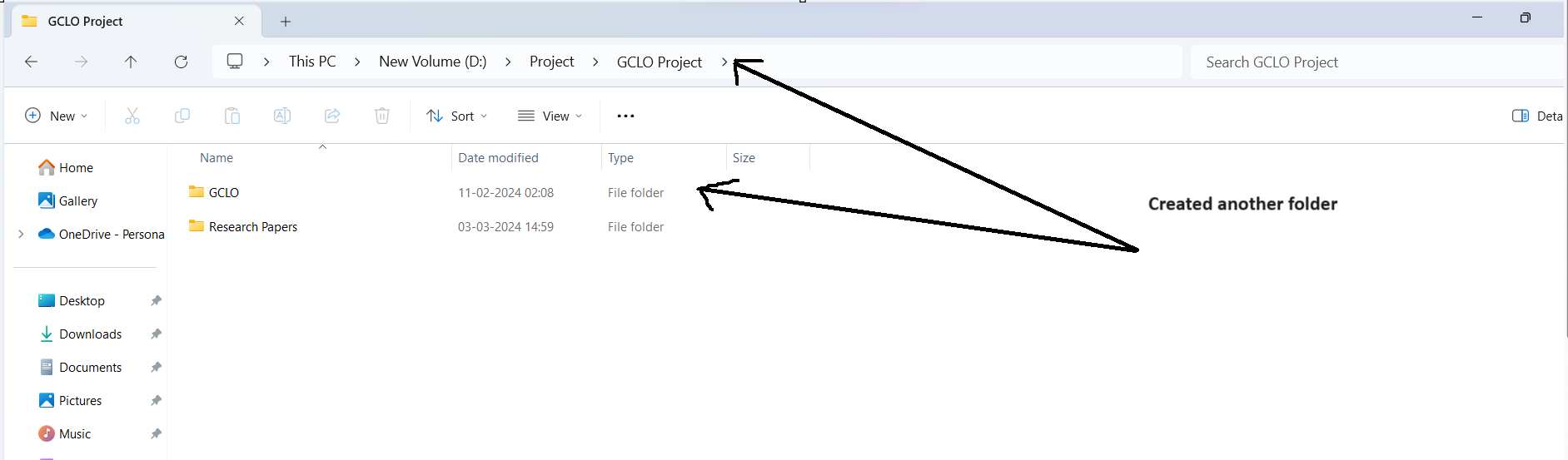
**Within this “Job Board” folder create another folder (with the same name or a different name)**

**How to create a folder for a project**

****

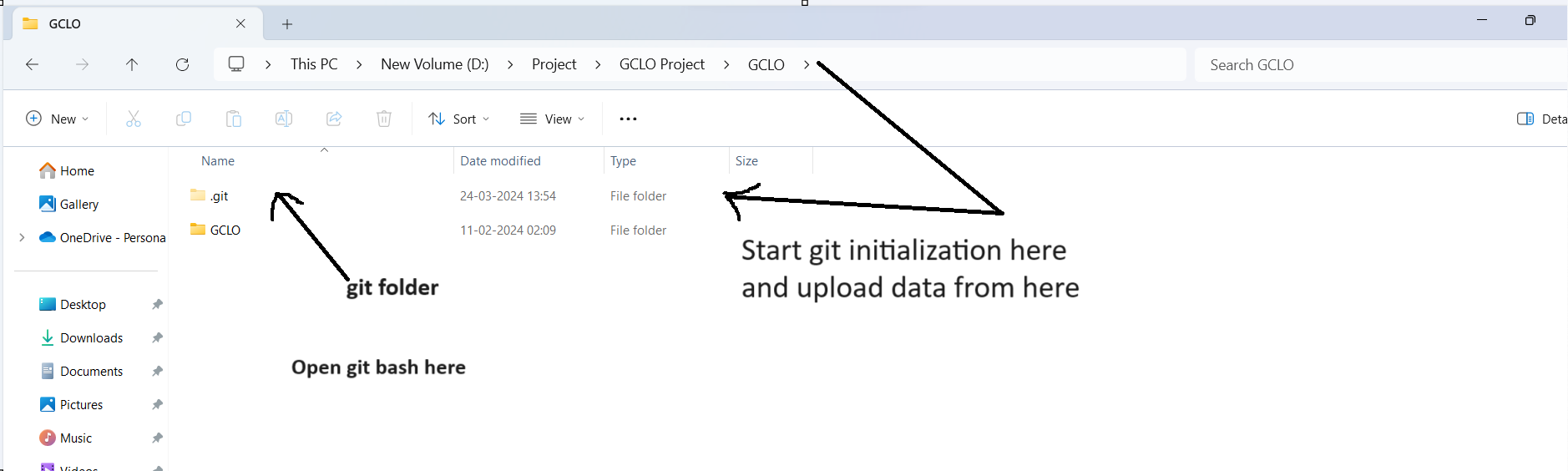
**Create another folder within “GCLO Project”**

**“GCLO Project” replace with your project’s folder name**

****

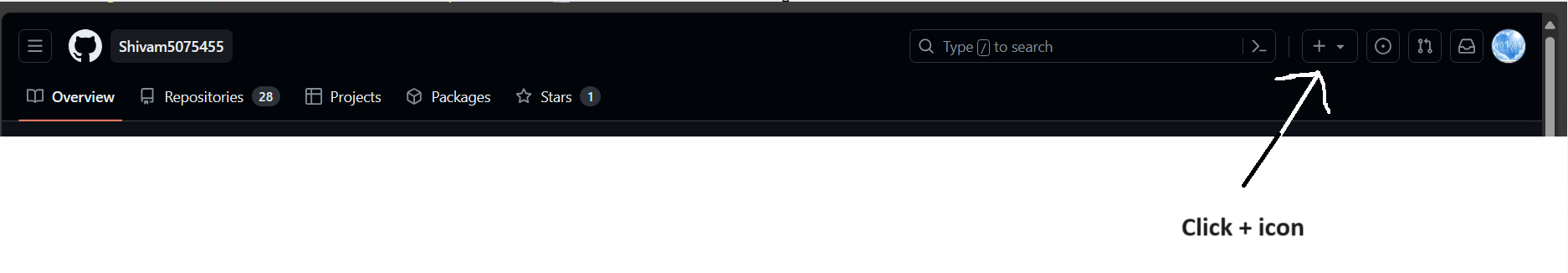
**Open git bash here and initialize git repository**

**Start code upload from here**

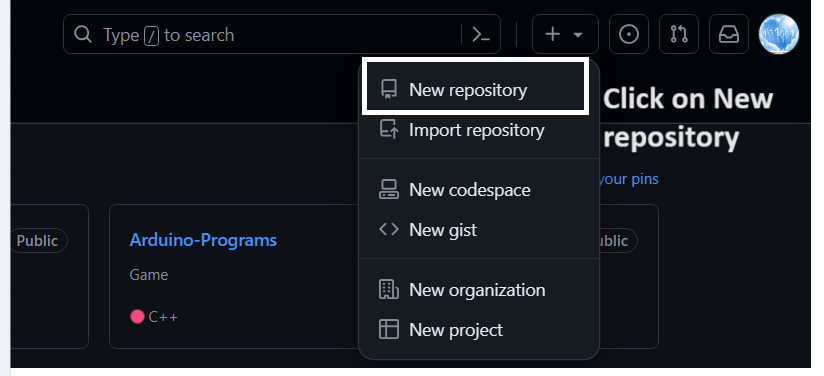
****

**How to create a new repository and upload code in new repository**

**Open github in any browser and click on + icon**

****

**Click on “New repository”**

****

Write the repository name and select access **private** or **public** and click on “**create repository”**

Step-1: git init

// replace **gitLink** with repository link

// Copy link of your repository

Step-2: git remote add origin gitLink

**// use any one step according to need step-3 or step3.1**

// dot(.) is used to add all file/folders at the same path

Step-3: git add .

// if you want to add particular file/folder, write file/folder name with it’s extension

Step-3.1 : git add file/folder name

// write message relevant to work changes or if first time commit write “First Commit”

Step-4: git commit -m "First commit"

// finally upload the code. here **master** is the branch name of repository

Step-5: git push -u origin master

**Update Existing File/Folder**

Step-1: go to same path where file/folder/project exists in local machine/laptop and open **git bash** at this location

// if you want to add particular file/folder write file/folder name

Step-2: git add file/folder name

// **OR**

// if you want to add all file/folder at the same path, use the dot(.)

Step-2: git add .

// write a message within double quotes relevant to changes in the file

Step-3: git commit -m “commit message”

Step-4: git push -u origin master

**// check status which file/folder is modified**

**// different colors have different meaning that shows status which file/folder modified. Modified file/folder shows in same color than unmodified file/folder**

$ git status

**//To add a particular file or folder**

git add --all folder name/file name

OR

git add <file/folder name>

//Your first option is Git’s suggestion. You can use the git restore --staged<file> command, as shown below.

$ git restore --staged Load.txt

//We can use the git reset command to unstage the Load.txt file, as shown below.

$ git reset Load.txt

**How to change branch**

git branch -M main (-M means modify)

**Error: due to change branch name**

**Solution:** --> **If you have changed branch after the push, then follow the below steps**

Step-1: git branch -m master main

Step-2: git fetch origin

Step-3: git branch -u origin/main main

Step-4: git remote set-head origin -a

Step-5: git commit -m “First commit”

Step-6: git push -u origin main

**“main”** is the current branch name after changing the branch name to “**main”**

**How to rename branch name and update with local existing repository**

**Rename using command:** git branch -m master main

**OR from browser**

Step-1: Open repository in github

Step-2: Go to “Settings” and rename branch under the section of default branch

If you have a clone before changing branch name -> Update with existing local repository

Open “git bash” at the location of the existing local repository

Step-3: git branch -m master main

Step-4: git fetch origin

Step-5: git branch -u origin/main main

Step-6: git remote set-head origin -a

**Create New Local Branch**

git branch branchName

**Move to New Local Branch**

git checkout branchName

**Merge Local Branch with Main Branch**

**Create New Local Branch**

#Before merge switch to main branch

git merge branchName

**Delete local branch**

git branch -d branchName

**Some common errors and their solutions**

**Error:** git push -u origin master

To https://github.com/Shivam5075455/Bharat-Intern.git

! [rejected] master -> master (fetch first)

error: failed to push some refs to 'https://github.com/Shivam5075455/Bharat-Intern.git'

hint: Updates were rejected because the remote contains work that you do

hint: not have locally. This is usually caused by another repository pushing

hint: to the same ref. You may want to first integrate the remote changes

hint: (e.g., 'git pull ...') before pushing again.

hint: See the 'Note about fast-forwards' in 'git push --help' for details.

**Solution:** Step-1: $ git fetch origin main/master:tmp

Step-2: $ git rebase tmp

Step-3: git push --set-upstream origin master

Step-4: $ git push -u origin master

**Error:** $ git push -u origin master

To https://github.com/Shivam5075455/Job\_Board\_News\_App.git

! [rejected] master -> master (non-fast-forward)

error: failed to push some refs to 'https://github.com/Shivam5075455/Job\_Board\_News\_App.git'

hint: Updates were rejected because the tip of your current branch is behind

hint: its remote counterpart. Integrate the remote changes (e.g.

hint: 'git pull ...') before pushing again.

hint: See the 'Note about fast-forwards' in 'git push --help' for details.

**Reason:** The error message you're seeing is because the local branch you are trying to push (master) has changes that you don't have in your remote repository on GitHub. This can happen if someone else has pushed changes to the same branch or if you have made changes in another location and pushed them to the remote repository.

**Solution:** git fetch origin

git merge origin/master

# Resolve any merge conflicts if there are any

git commit -m "Resolved merge conflicts"

git push -u origin master

**ERROR:**

$ git push -u origin main

To https://github.com/Shivam5075455/DynamicAppAPI.git

! [rejected] main -> main (non-fast-forward)

error: failed to push some refs to 'https://github.com/Shivam5075455/DynamicAppAPI.git'

hint: Updates were rejected because the tip of your current branch is behind

hint: its remote counterpart. Integrate the remote changes (e.g.

hint: 'git pull ...') before pushing again.

hint: See the 'Note about fast-forwards' in 'git push --help' for details.

**SOLUTION:** Pull Changes and Rebase:

# Fetch changes from the remote repository

$ git fetch origin

CHOOSE ONE MERGE OR REBASE

# If you want to merge the changes

$ git merge origin/main

# If you want to rebase your changes on top of the remote changes

$ git rebase origin/main

# If you are not sure which one to choose, you can do a pull with rebase

$ git pull --rebase origin main

$ git push origin main

**Alternatively, you can pull and rebase your changes on top of the remote changes.**

Run git pull --rebase origin main.

Resolve any conflicts if prompted.

Finally, push your changes.

run command

git pull --rebase origin main

# Resolve any rebase conflicts if prompted

git push origin main