

# Task 1

## GIT Commands

### Initializing a Github in Local Machine.

1. Making directory for github projects.

- **mkdir git\_projects**

MINGW64:/c/Users/Akash/Documents/industry\_ready\_projects/1 git/git\_projects

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git
$ mkdir git_projects

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git
$ cd git_projects

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects
$
```

2. Creating a local repository for Git .

MINGW64:/c/Users/Akash/Documents/industry\_ready\_projects/1 git/git\_projects

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects
$ mkdir my_first_repo

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects
$ git init
Initialized empty Git repository in C:/Users/Akash/Documents/industry_ready_projects/1 git/git_projects/.git/

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects (master)
$ ls -la
./  ../  .git/  my_first_repo/

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects (master)
$ |
```

- We have initialized an empty repository using "**git init**"
- Notice there is a **".git/"** folder created inside a repository which means our local repository is ready.
- Notice it is showing **"Master"** branch which means whatever work we do it will go into Master branch.

-

3. Getting status of Git

- Git status tell the current state of the repository. For ex: whether new file is created, if there are any changes, whether the code is committed

MINGW64:/c/Users/Akash/Documents/industry\_ready\_projects/1 git/git\_projects

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects (master)
$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects (master)
$
```

- There are no commits in the above
- Lets create a file and then check the status again

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects (master)
$ touch file.txt

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file.txt

nothing added to commit but untracked files present (use "git add" to track)

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects (master)
$ |
```

- Once the file is created we check the status
- It is showing "**Untracked files**" it means in the working area some changes has been done or a file is added in the local Repository.

## Cloning a Remote Repository.

4. Lets clone [https://github.com/sourangshupal/my\\_calculator](https://github.com/sourangshupal/my_calculator)
- **git clone** [https://github.com/sourangshupal/my\\_calculator](https://github.com/sourangshupal/my_calculator)

MINGW64:/c/Users/Akash/Documents/industry\_ready\_projects/1 git

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git
$ git clone https://github.com/sourangshupal/my_calculator
Cloning into 'my_calculator'...
remote: Enumerating objects: 45, done.
remote: Counting objects: 100% (45/45), done.
remote: Compressing objects: 100% (27/27), done.
remote: Total 45 (delta 15), reused 42 (delta 14), pack-reused 0
Receiving objects: 100% (45/45), 6.12 KiB | 569.00 KiB/s, done.
Resolving deltas: 100% (15/15), done.

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git
$ ls -a
./  ../  '3-Git .pdf'  GIT.docx  git_projects/  my_calculator/  '~$GIT.docx'

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git
$
```

- This command will clone entire github repository

## Checking git logs.

### 5. Checking logs

- **git log**

MINGW64:/c:/Users/Akash/Documents/industry\_ready\_projects/1 git/my\_calculator

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master)
$ git log
commit 54683c7ded71473628f1cd7eb836053d667060c1 (HEAD -> master, origin/master, origin/HEAD)
Merge: 9c85077 90ab3fc
Author: Sourangshu Paul <sourangshu@ineuron.ai>
Date: Tue Sep 7 14:48:37 2021 +0530

    Merge remote-tracking branch 'origin/master'

commit 9c85077612b3d2808dcae17ac78852c116c7155f
Author: Sourangshu Paul <sourangshu@ineuron.ai>
Date: Tue Sep 7 14:47:56 2021 +0530

    Tag Added

commit 90ab3fc2a4c93acede7def388cb4da8d968b10e1
Author: Sourangshu Paul <14earcs0158@aryacollege.in>
Date: Tue Sep 7 11:04:53 2021 +0530

    Create Readme.md

    Added small Description

commit f380a22c1b32e08fcc76f21374ac90415e0bc197 (tag: v1.0)
Author: paul <paulbindass@gmail.com>
Date: Tue Aug 31 14:35:13 2021 +0530

    Debug Parameter updated

commit e62d145144853a0e9e4f6bf7867c08f540dc313e
Author: paul <paulbindass@gmail.com>
Date: Tue Aug 31 14:34:04 2021 +0530

    template updated

commit 2a2984df3e14d7418437828e8537d4dfa228e5fa
Author: paul <paulbindass@gmail.com>
Date: Tue Aug 31 14:32:05 2021 +0530

    template updated

commit 9e5cbfbd92a248091fa5ea18c73066b1c4f138e5
Author: paul <paulbindass@gmail.com>
Date: Tue Aug 31 14:30:09 2021 +0530

    Route name updated

commit 10455d5ca013db478e913866119054a30a2e93fc
Author: paul <paulbindass@gmail.com>
```

### Shorter git one liner log

- **git log --oneline**

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master)
$ git log --oneline
54683c7 (HEAD -> master, origin/master, origin/HEAD) Merge remote-tracking branch 'origin/master'
9c85077 Tag Added
90ab3fc Create Readme.md
f380a22 (tag: v1.0) Debug Parameter updated
e62d145 template updated
2a2984d template updated
9e5cbfb Route name updated
10455d5 Host and App updated
f2769c5 Flask App File added
a3a0d5d Requirements File added
b34f1ca Calculator Args File added
a1bb916 Calculator File added
```

check who has done changes at what time

- **git log --stat**

```
MINGW64~/c/Users/Akash/Documents/industry_ready_projects/1 git/my_calculator

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master)
$ git log --stat
commit 54683c7ded71473628f1cd7eb836053d667060c1 (HEAD -> master, origin/master, origin/HEAD)
Merge: 9c85077 90ab3fc
Author: Sourangshu Paul <sourangshu@ineuron.ai>
Date: Tue Sep 7 14:48:37 2021 +0530

    Merge remote-tracking branch 'origin/master'

commit 9c85077612b3d2808dcae17ac78852c116c7155f
Author: Sourangshu Paul <sourangshu@ineuron.ai>
Date: Tue Sep 7 14:47:56 2021 +0530

    Tag Added

    .gitignore | 4 +++-
    1 file changed, 3 insertions(+), 1 deletion(-)

commit 90ab3fc2a4c93acade7def388cb4da8d968b10e1
Author: Sourangshu Pal <l4earcs0158@aryacollege.in>
Date: Tue Sep 7 11:04:53 2021 +0530

    Create Readme.md

    Added small Description

    README.md | 5 +++++
    1 file changed, 5 insertions(+)

commit f380a22c1b32e08fcc76f21374ac90415e0bc197 (tag: v1.0)
Author: paul <paulbindass@gmail.com>
Date: Tue Aug 31 14:35:13 2021 +0530

    Debug Parameter updated

    app.py | 2 +-
    1 file changed, 1 insertion(+), 1 deletion(-)

commit e62d145144853a0e9e4f6bf7867c08f540dc313e
Author: paul <paulbindass@gmail.com>
Date: Tue Aug 31 14:34:04 2021 +0530

    template updated

    app.py | 2 +-
    1 file changed, 1 insertion(+), 1 deletion(-)
```

+/- shows how many lines have been added or removed.

check changes in details

- **git log --patch**

check details with respect to the "SHA" or "Commit ID" of commits

- **git log --oneline**

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master)
$ git log --oneline
54683c7 (HEAD -> master, origin/master, origin/HEAD) Merge remote-tracking branch 'origin/master'
9c85077 Tag Added
90ab3fc Create README.md
f380a22 (tag: v1.0) Debug Parameter updated
e62d145 template updated
2a2984d template updated
9e5cbfb Route name updated
10455d5 Host and App updated
f2769c5 Flask App File added
a3a0d5d Requirements File added
b34f1ca Calculator Args File added
a1bb916 Calculator File added
```

- **git show e62d145**

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master)
$ git show e62d145
commit e62d145144853a0e9e4f6bf7867c08f540dc313e
Author: paul <paulbindass@gmail.com>
Date: Tue Aug 31 14:34:04 2021 +0530

    template updated

diff --git a/app.py b/app.py
index b9a033a..503700b 100644
--- a/app.py
+++ b/app.py
@@ -49,4 +49,4 @@ def math_operation_via_postman():

    if __name__ == '__main__':
-       app.run(host="127.0.0.1", port = 8080)
+       app.run(host="127.0.0.1", port = 8080, debug=True)
```

## Commits

### 6. Adding files creating a file

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ vi new_file.txt
```

### Check the status

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    file.txt
    new_file.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Notice 2 files are untracked

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file.txt
        new_file.txt

nothing added to commit but untracked files present (use "git add" to track)
```

so lets add these all file into the staging area of repository using

- **git add .**

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git add .
warning: in the working copy of 'new_file.txt', LF will be replaced by CRLF the next time Git touches it

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file.txt
        new file:   new_file.txt
```

Now we can observe once we use "git add ." notice files are now not showing untracked, it means that files have been moved to staging area and are ready to be committed.

## 7. Committing

- **git commit -m <appropriate message>**

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git commit -m "two new files added"
[master (root-commit) be5a877] two new files added
2 files changed, 6 insertions(+)
create mode 100644 file.txt
create mode 100644 new_file.txt
```

Using this we commit all the changes form our staging area to our Github repository

## 8. Difference

### - **git diff**

using the above command we can find which lines have been added or removed by the user.

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git diff
warning: in the working copy of 'new_file.txt', LF will be replaced by CRLF the next time Git touches it
diff --git a/new_file.txt b/new_file.txt
index 73ef92e..6289475 100644
--- a/new_file.txt
+++ b/new_file.txt
@@ -3,4 +3,5 @@ banana
 mango
 guavava
 pomogranate
-
+strawberry
+grapes
```

## 9. Git Restore

`git restore --staged <file>`

"git restore" command helps to unstage or even discard uncommitted local changes

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git stage
Nothing specified, nothing added.
hint: Maybe you wanted to say 'git add .'
hint: Turn this message off by running
hint: "git config advice.addEmptyPaths false"

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    plants.txt

nothing added to commit but untracked files present (use "git add" to track)

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
```

File added is brought to staging area

## 10. Tagging

It use to tag the latest commit.

For ex: we have done many commits and we say work done up to current commit point is version V1.0.1, so we tag it

```
MINGW64:/c:/Users/Akash/Documents/industry_ready_projects/1 git/git_projects/my_first_repo

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git log --oneline
6ee7741 (HEAD -> master) fruits updated
db42cfd cars file updated
fe4d52a cars file added
3c46a06 plant file added
42c265c fruits added
be5a877 two new files added
```

## Annotated Tags

Includes information regarding who created tag, date & time of creation and message

- `git tag -a v1.0.2`

- Let's add annotated tag

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git tag -a v1.0.1
```

- Check tags in current repository

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git tag
v1.0.1
```

- Tag is by default applied to latest commit

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git log --oneline
6ee7741 (HEAD -> master, tag: v1.0.1) fruits updated
db42cfd cars file updated
fe4d52a cars file added
3c46a06 plant file added
42c265c fruits added
be5a877 two new files added
```

- If we want to tag some older version than we use its "sha" or "commit id"

- `git tag v0.0.1`

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git tag -a v0.0.1 3c46a06
```

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git log --oneline
6ee7741 (HEAD -> master, tag: v1.0.1) fruits updated
db42cfd cars file updated
fe4d52a cars file added
3c46a06 (tag: v0.0.1) plant file added
42c265c fruits added
be5a877 two new files added

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git tag
v0.0.1
v1.0.1
```



Lightweight tag: it does not contain additional information

- **git tag v1.0.2**

```
MINGW64:/c/Users/Akash/Documents/industry_ready_projects/1 git/git_projects/my_first_repo

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git log --oneline
7d570c4 (HEAD -> master) fast food file added
6ee7741 (tag: v1.0.1) fruits updated
db42cfd cars file updated
fe4d52a cars file added
3c46a06 (tag: v0.0.1) plant file added
42c265c fruits added
be5a877 two new files added

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git tag v1.0.2

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git log --oneline
7d570c4 (HEAD -> master, tag: v1.0.2) fast food file added
6ee7741 (tag: v1.0.1) fruits updated
db42cfd cars file updated
fe4d52a cars file added
3c46a06 (tag: v0.0.1) plant file added
42c265c fruits added
be5a877 two new files added

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git tag
v0.0.1
v1.0.1
v1.0.2
```

Deleting a tag

- **git tag -d v0.0.1**

```
MINGW64:/c/Users/Akash/Documents/industry_ready_projects/1 git/git_projects/my_first_repo

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git log --oneline
7d570c4 (HEAD -> master, tag: v1.0.2) fast food file added
6ee7741 (tag: v1.0.1) fruits updated
db42cfd cars file updated
fe4d52a cars file added
3c46a06 (tag: v0.0.1) plant file added
42c265c fruits added
be5a877 two new files added

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git tag -d v0.0.1
Deleted tag 'v0.0.1' (was 1a33b7b)

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git log --oneline
7d570c4 (HEAD -> master, tag: v1.0.2) fast food file added
6ee7741 (tag: v1.0.1) fruits updated
db42cfd cars file updated
fe4d52a cars file added
3c46a06 plant file added
42c265c fruits added
be5a877 two new files added
```

## 11. Branching

Branches allows you to work on different parts of a project without impacting the main branch.

When the work is complete, a branch can be merged with the main project.

You can switch between branches and work on different projects without them interfering with each other.

### Creating Branches

```
MINGW64:/c/Users/Akash/Documents/industry_ready_projects/1 git/git_projects/my_first_repo
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git branch
* master

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git branch bugfixed#5403

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git branch
  bugfixed#5403
* master

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$
```

We have simply written

```
git branch bugfixed#5403
or
git switch bugfixed#5403
```

which created new branch.

Here we can notice we are currently in master branch

### - Switching branches

```
git checkout bugfixed#5403
```

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git checkout bugfixed#5403
Switched to branch 'bugfixed#5403'
```

## - Creating and switching in one go

`git checkout -b levelface`

or


`git switch -b levelface`

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (bugfixed#5403)
$ git checkout -b levelface
Switched to a new branch 'levelface'

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (levelface)
$ git branch
  bugfixed#5403
* levelface
  master
```

## Example

### 1 check branches

 MINGW64:/c/Users/Akash/Documents/industry\_ready\_projects/1 git/my\_calculator

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master)
$ git branch
* master
```

### 2 Create and switch to new branch

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master)
$ git checkout -b dev
Switched to a new branch 'dev'
```

### 3 Adding comments to new branch

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (dev)
$ start .

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (dev)
$ ls
README.md  app.py  calculator.py  calculator_args.py  requirements.txt  static/  templates/

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (dev)
$ vi app.py

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (dev)
$ git add .

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (dev)
$ git status
On branch dev
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   app.py

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (dev)
$ git commit -m "comments added for Imports and Routes"
[dev ddb88ea] comments added for Imports and Routes
1 file changed, 3 insertions(+), 1 deletion(-)
```

```
app.py > math_operation
1  #Import Section
2  from flask import Flask, render_template, request, jsonify
3
4  app = Flask(__name__)
5
6  #Routes for Homepage
7  @app.route('/', methods=['GET', 'POST']) # To render Homepage
8  def home_page():
9      return render_template('index.html')
10
11  #Route for calculator app
12  @app.route('/calculator', methods=['POST']) # This will be called from l
13  def math_operation():
14      if (request.method=='POST'):
15          operation=request.form['operation']
16          num1=int(request.form['num1'])
```

These changes are only done in "dev" branch once the branch is switched then we may not see changes.

Lets again switch to Master branch

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (dev)
$ git switch master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master)
$
```

Now check VS code project again

```
app.py x
app.py > math_operation
1  from flask import Flask, render_template, request, jsonify
2
3  app = Flask(__name__)
4
5
6  @app.route('/', methods=['GET', 'POST']) # To render Homepage
7  def home_page():
8      return render_template('index.html')
9
10  @app.route('/calculator', methods=['POST']) # This will be called from l
11  def math_operation():
12      if (request.method=='POST'):
13          operation=request.form['operation']
14          num1=int(request.form['num1'])
15          num2 = int(request.form['num2'])
```

Notice in Master branch there are no changes.

## 12. Merging

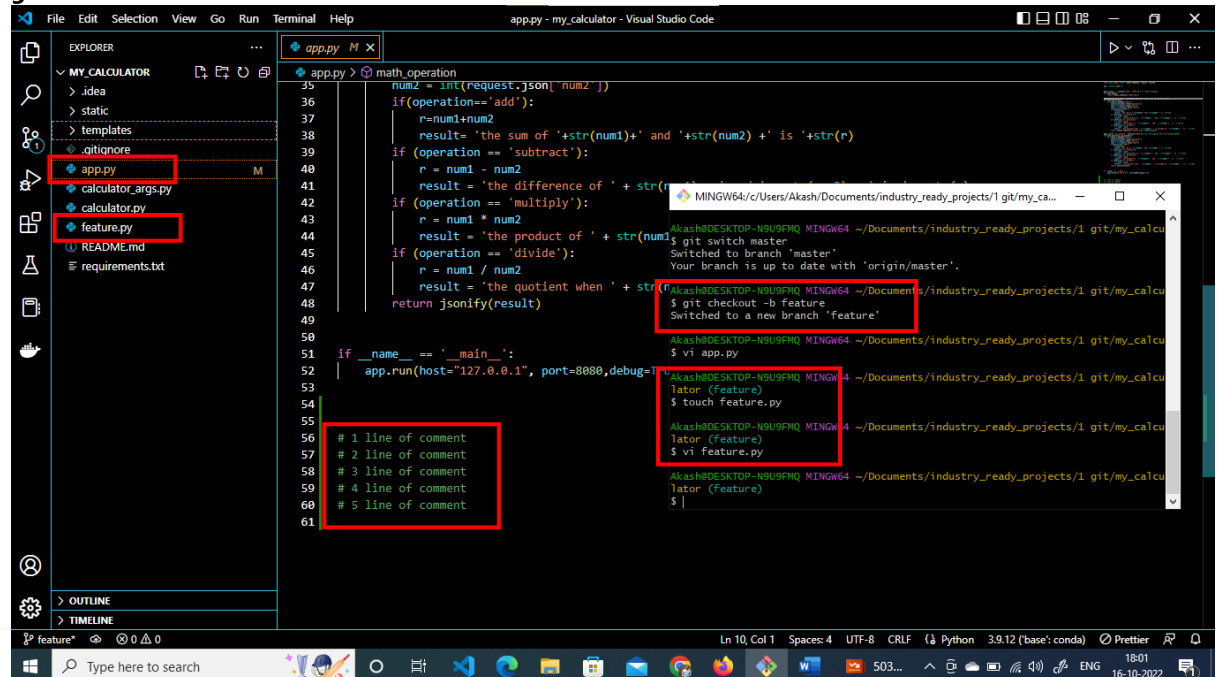
Merging takes the contents of source branch and integrates them with a target branch.

In merging, only the target branch is changed. The source branch history remains the same

### - Fast Forward Merge

Lets create a new branch "feature" and do some changes and also a new file feature.py

**git checkout -b feature**



The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left. The Explorer shows a project named 'MY\_CALCULATOR' with files like '.idea', 'static', 'templates', '.gitignore', 'app.py', 'calculator\_args.py', 'calculator.py', 'feature.py', 'README.md', and 'requirements.txt'. The 'feature.py' file is highlighted. The main editor shows the code for 'app.py', which is a Flask application. The code includes a 'math\_operation' function that handles addition, subtraction, multiplication, and division. The code is as follows:

```
35 num2 = int(request.json['num2'])
36
37 if(operation=='add'):
38     r=num1+num2
39     result= 'the sum of '+str(num1)+' and '+str(num2)+' is '+str(r)
40
41 if (operation == 'subtract'):
42     r = num1 - num2
43     result = 'the difference of ' + str(num1) + ' and ' + str(num2) + ' is ' + str(r)
44
45 if (operation == 'multiply'):
46     result = 'the product of ' + str(num1) + ' and ' + str(num2) + ' is ' + str(r)
47
48 if (operation == 'divide'):
49     r = num1 / num2
50     result = 'the quotient when ' + str(num1) + ' is divided by ' + str(num2) + ' is ' + str(r)
51
52 return jsonify(result)
53
54
55 if __name__ == '__main__':
56     app.run(host="127.0.0.1", port=8080, debug=True)
```

The terminal window at the bottom shows the following commands and output:

```
$ git switch master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
$ git checkout -b feature
Switched to a new branch 'feature'
$ vi app.py
$ touch feature.py
$ vi feature.py
```

These changes are only available in branch "Feature" and not in other branches.

Let's add and commit these changes.

```

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calcu
lator (feature)
$ git add .
warning: in the working copy of 'feature.py', LF will be replaced by CRLF the ne
xt time Git touches it

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calcu
lator (feature)
$ git commit -m "feature branch work started"
[feature 377bdcf] feature branch work started
 2 files changed, 9 insertions(+)
 create mode 100644 feature.py

```

Let's see commit logs of all the branches

`git log --oneline --graph --all`

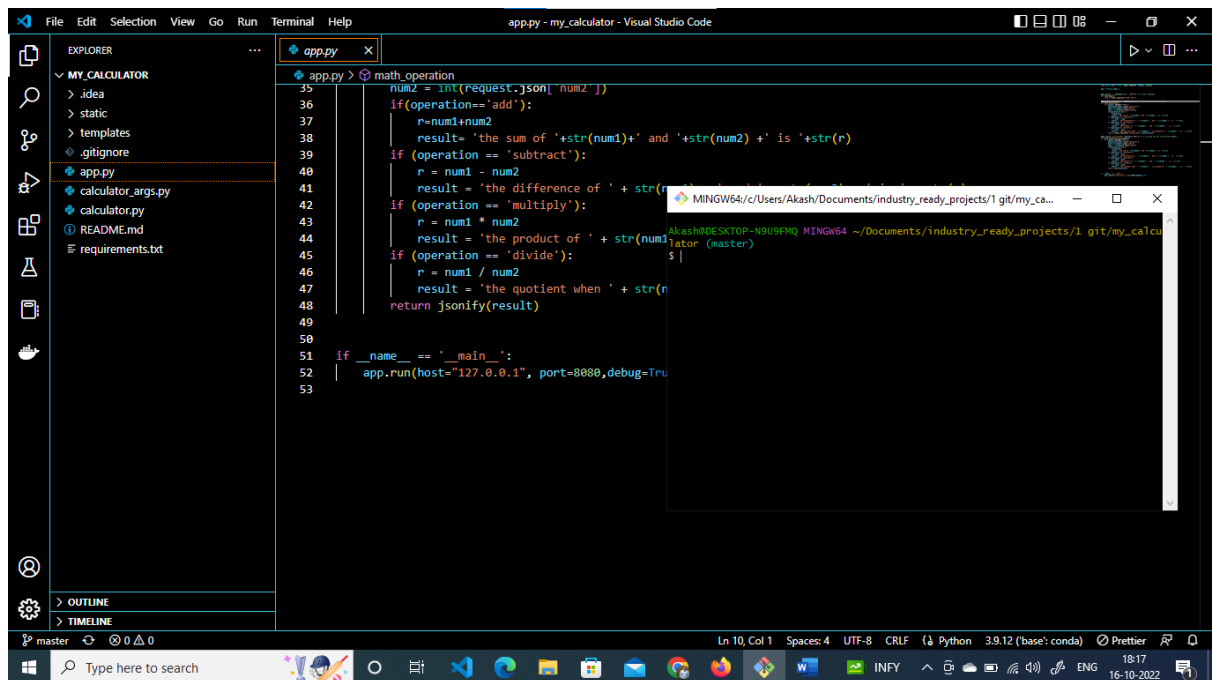
```

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (Feature)
$ git log --oneline --graph --all
* 377bdcf (HEAD -> feature) feature branch work started
| * ddb88ea (dev) comments added for Imports and Routes
|/
| * 54683c7 (origin/master, origin/HEAD, master) Merge remote-tracking branch 'origin/master'
|/
| * 90ab3fc Create Readme.md
* | 9c85077 Tag Added
|/
* f380a22 (tag: v1.0) Debug Parameter updated
* e62d145 template updated
* 2a2984d template updated
* 9e5cbfb Route name updated
* 10455d5 Host and App updated
* f2769c5 Flask App File added
* a3a0d5d Requirements File added
* b34f1ca Calculator Args File added
* a1bb916 Calculator File added

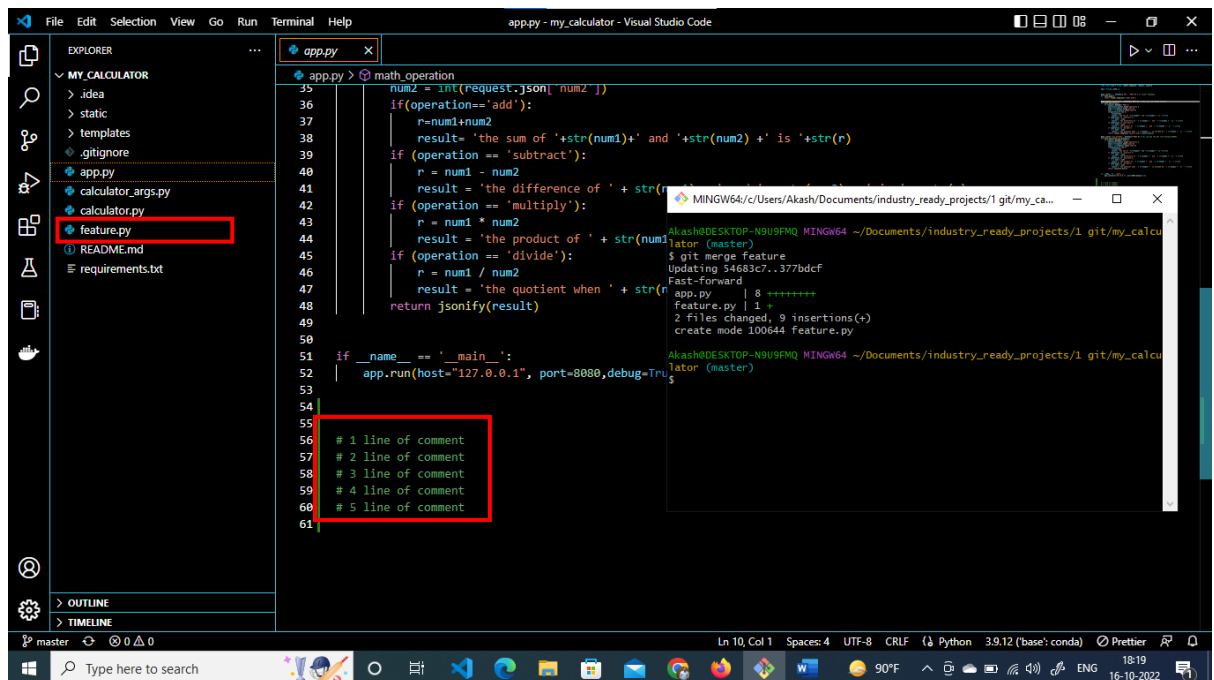
```

Now we will merge Feature branch into Master branch

## Before merging



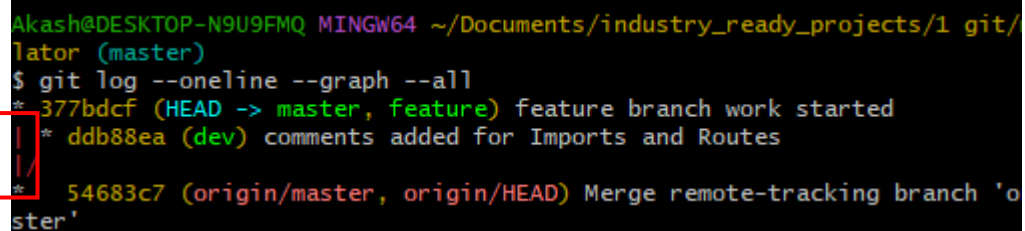
## After merging



All the changes which were done in Feature branch have now reflected into Master branch.

This type of simple merge is called Fast Forward Merge

Notice there is an direct line from Master branch to feature branch



```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/
lator (master)
$ git log --oneline --graph --all
* 377bdcf (HEAD -> master, feature) feature branch work started
| * ddb88ea (dev) comments added for Imports and Routes
|/
* 54683c7 (origin/master, origin/HEAD) Merge remote-tracking branch 'origin/master'
```

This means that basic content of Master branch and the Feature branch are exactly the same.

Now after some number of commits in Master branch we are simply copying content of Master branch into the Feature branch and now in Feature branch we are making few changes, so Feature branch can be thought off as it is few commits ahead of Master branch.

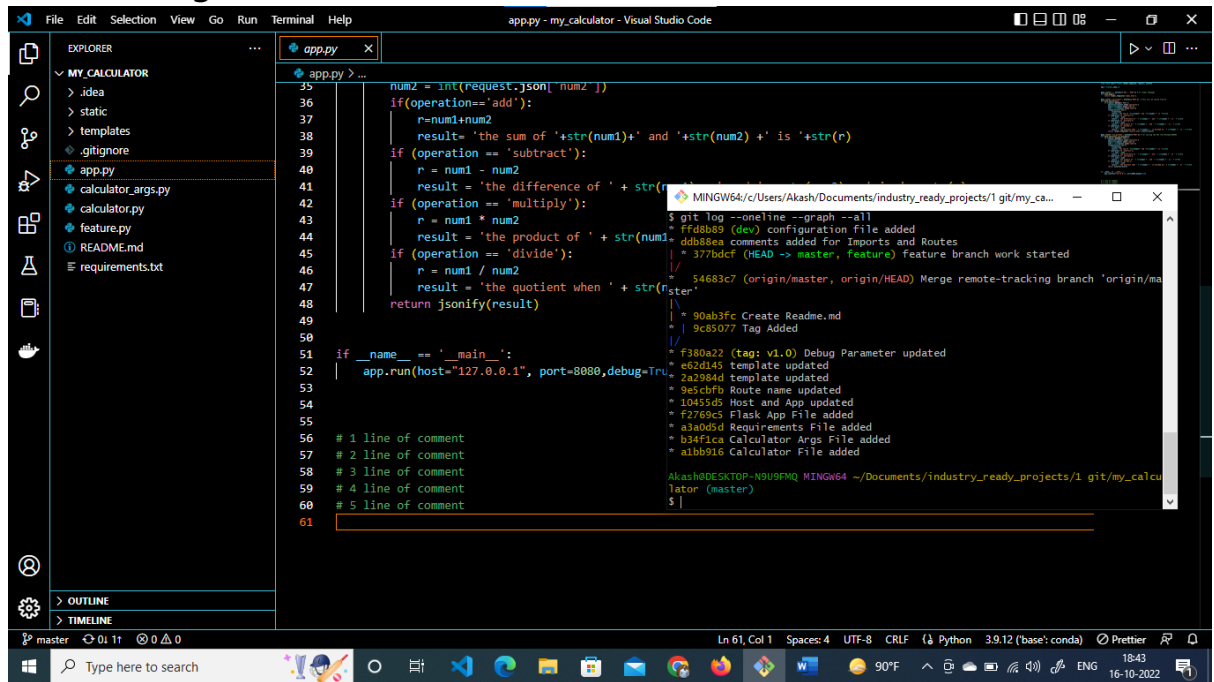
Now we can finally merge Feature branch using merge command so changes of both the branch will synchronize in Master branch

- **Regular Merge**

This is same as Fast Forward Merge but it is used when both branches are going into different directions .

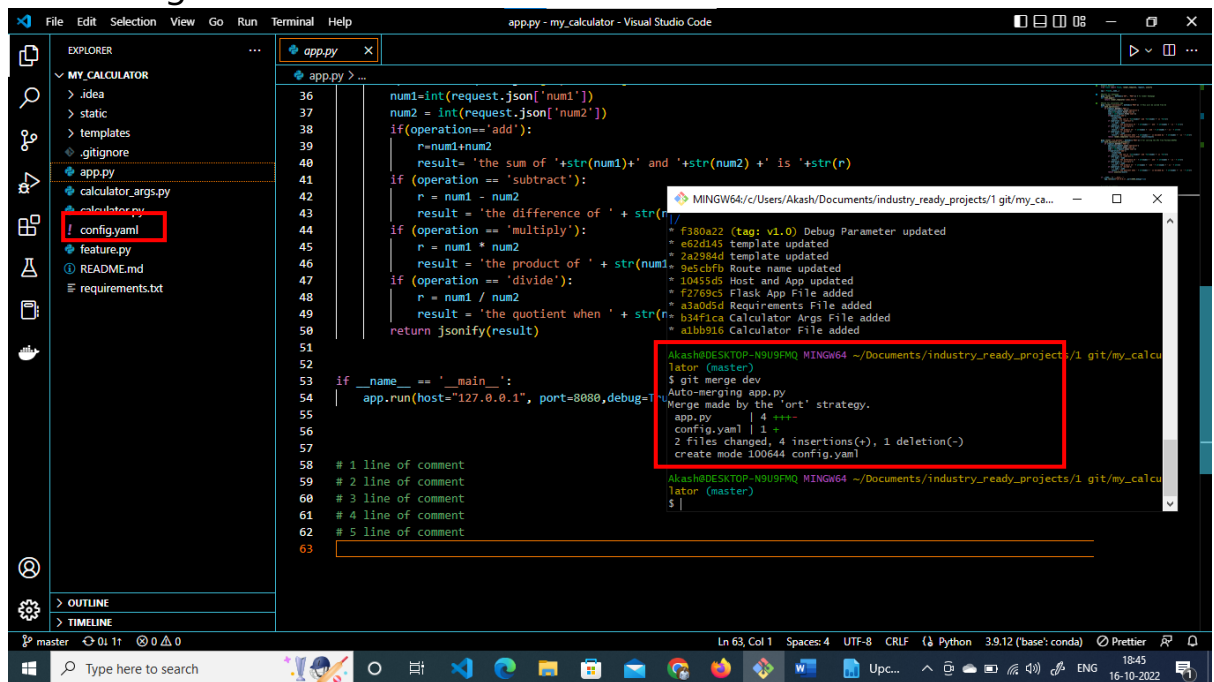


## Before Merge



```
35 num2 = int(request.json['num2'])
36
37 if operation == 'add':
38     r = num1 + num2
39     result = 'the sum of ' + str(num1) + ' and ' + str(num2) + ' is ' + str(r)
40
41 if operation == 'subtract':
42     r = num1 - num2
43     result = 'the difference of ' + str(num1) + ' and ' + str(num2) + ' is ' + str(r)
44
45 if operation == 'multiply':
46     r = num1 * num2
47     result = 'the product of ' + str(num1) + ' and ' + str(num2) + ' is ' + str(r)
48
49 if operation == 'divide':
50     r = num1 / num2
51     result = 'the quotient when ' + str(num1) + ' is divided by ' + str(num2) + ' is ' + str(r)
52
53 return jsonify(result)
54
55 if __name__ == '__main__':
56     app.run(host='127.0.0.1', port=8080, debug=True)
```

## After Merge



```
36 num1 = int(request.json['num1'])
37 num2 = int(request.json['num2'])
38
39 if operation == 'add':
40     r = num1 + num2
41     result = 'the sum of ' + str(num1) + ' and ' + str(num2) + ' is ' + str(r)
42
43 if operation == 'subtract':
44     r = num1 - num2
45     result = 'the difference of ' + str(num1) + ' and ' + str(num2) + ' is ' + str(r)
46
47 if operation == 'multiply':
48     r = num1 * num2
49     result = 'the product of ' + str(num1) + ' and ' + str(num2) + ' is ' + str(r)
50
51 if operation == 'divide':
52     r = num1 / num2
53     result = 'the quotient when ' + str(num1) + ' is divided by ' + str(num2) + ' is ' + str(r)
54
55 return jsonify(result)
56
57 if __name__ == '__main__':
58     app.run(host='127.0.0.1', port=8080, debug=True)
```

```

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calcu
lator (master)
$ git log --oneline --graph --all
* 5d8b23f (HEAD -> master) Merge branch 'dev'
|
| * ffd8b89 (dev) configuration file added
| * cdb88ea comments added for Imports and Routes
| * 377bdcf (feature) feature branch work started
|/
* 54683c7 (origin/master, origin/HEAD) Merge remote-tracking branch 'origin/ma
ster'
|
| * 90ab3fc Create Readme.md
| * 9c85077 Tag Added
|/
* f380a22 (tag: v1.0) Debug Parameter updated
* e62d145 template updated
* 2a2984d template updated
* 9e5cbfb Route name updated
* 10455d5 Host and App updated
* f2769c5 Flask App File added
* a3a0d5d Requirements File added
* b34f1ca Calculator Args File added
* a1bb916 Calculator File added

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calcu
lator (master)
$

```

## Merging Conflicts

“merge conflicts” can occur in the process of integrating commits from a different source branch.

When a merge cannot be full performed automatically then a merge conflict takes place.

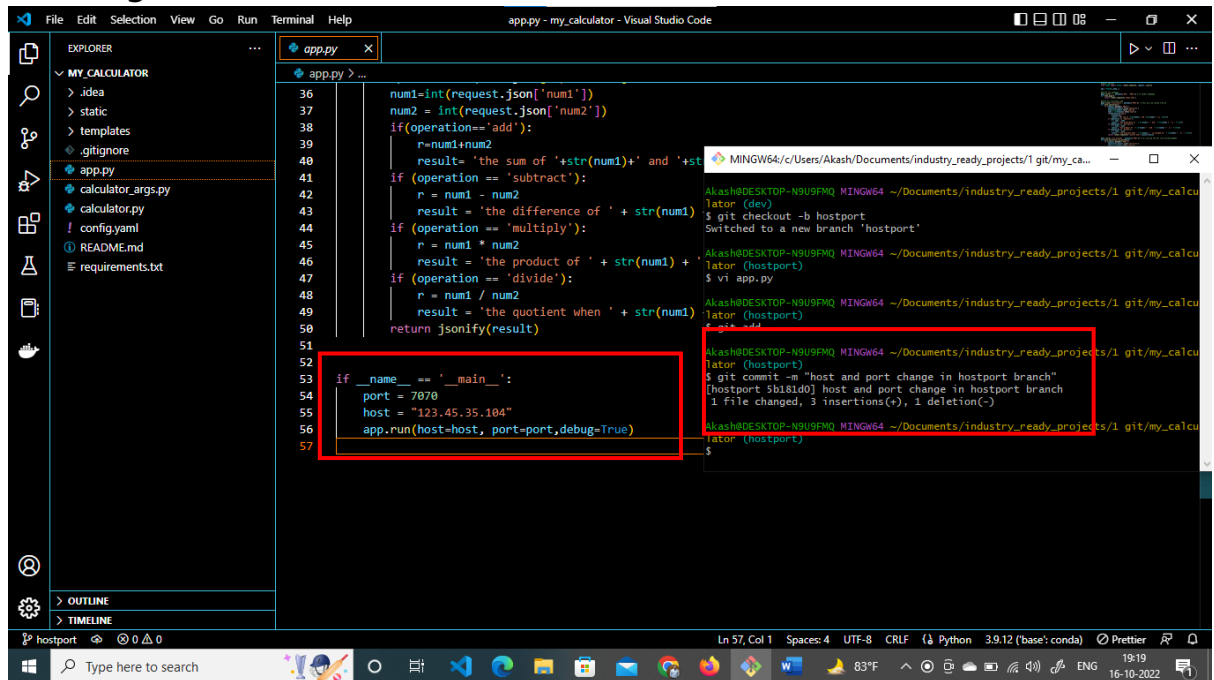
Git will try to combine as much as it can, but then it will leave special markers (e.g. >>> and <<<)

Example

Lets change host and port number in Master branch



## Creating a new branch from Dev branch



The screenshot shows the Visual Studio Code interface with a Python file named `app.py` open. The code implements a simple calculator with operations: add, subtract, multiply, and divide. A red box highlights the main execution block in `app.py` (lines 53-56):`if __name__ == '__main__':  
 port = 7070  
 host = "123.45.35.104"  
 app.run(host=host, port=port, debug=True)`

To the right, a terminal window shows the following commands and output:

```
MINGW64~/Users/Akash/Documents/industry_ready_projects/1 git/my_calculator (dev)  
$ git checkout -b hostport  
Switched to a new branch 'hostport'  
$ vi app.py  
$ git add  
$ git commit -m "host and port change in hostport branch"  
[hostport 5b181d0] host and port change in hostport branch  
1 file changed, 3 insertions(+), 1 deletion(-)  
$
```

## Let's switch back to the master and observe git log

```
MINGW64~/Users/Akash/Documents/industry_ready_projects/1 git/my_calculator  
* 5b181d0 (hostport) host and port change in hostport branch  
* 837b396 (HEAD -> master) port and host updated  
* 6d1031e port and host updated  
* bebdca3 port and IP updated  
* 5d8b23f Merge branch 'dev'  
~  
* | ffd8b89 (dev) configuration file added  
* | ddb88ea comments added for Imports and Routes  
* | 377bdcf (feature) feature branch work started  
~  
* 54683c7 (origin/master, origin/HEAD) Merge remote-tracking branch 'origin/master'  
~  
* 90ab3fc Create Readme.md  
* | 9c85077 Tag Added  
~  
* f380a22 (tag: v1.0) Debug Parameter updated  
* e62d145 template updated  
* 2a2984d template updated  
* 9e5cbfb Route name updated  
* 10455d5 Host and App updated  
* f2769c5 Flask App File added  
* a3a0d5d Requirements File added  
* b34f1ca Calculator Args File added  
* a1bb916 Calculator File added  
~
```

Now let's try to merge hostport branch with master branch

```

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master)
$ git merge hostport
Auto-merging app.py
CONFLICT (content): Merge conflict in app.py
Automatic merge failed; fix conflicts and then commit the result.

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master|MERGING)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 7 commits.
  (use "git push" to publish your local commits)

You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   app.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        "\357\200\233\357\200\272q"

no changes added to commit (use "git add" and/or "git commit -a")

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master|MERGING)
$ |

```

We observe that there is a conflict, Let's check app.py to understand the conflict

```

46         result = 'the product of ' + str(num1) + ' and ' + str(num2) + ' is ' + str(r)
47     if (operation == 'divide'):
48         r = num1 / num2
49         result = 'the quotient when ' + str(num1) + ' is divided by ' + str(num2) + ' is ' + str(r)
50     return jsonify(result)
51
52
53 if __name__ == '__main__':
54     <<<<<<< HEAD (Current Change)
55     port = 5555
56     host = "66.77.91.4"
57     app.run(host=host, port=port, debug=True)
58
59
60
61 # 1 line of comment
62 # 2 line of comment
63 # 3 line of comment
64 # 4 line of comment
65 # 5 line of comment
66
67     port = 7070
68     host = "123.45.35.104"
69     app.run(host=host, port=port, debug=True)
70 >>>>>> hostport (Incoming Change)
71

```

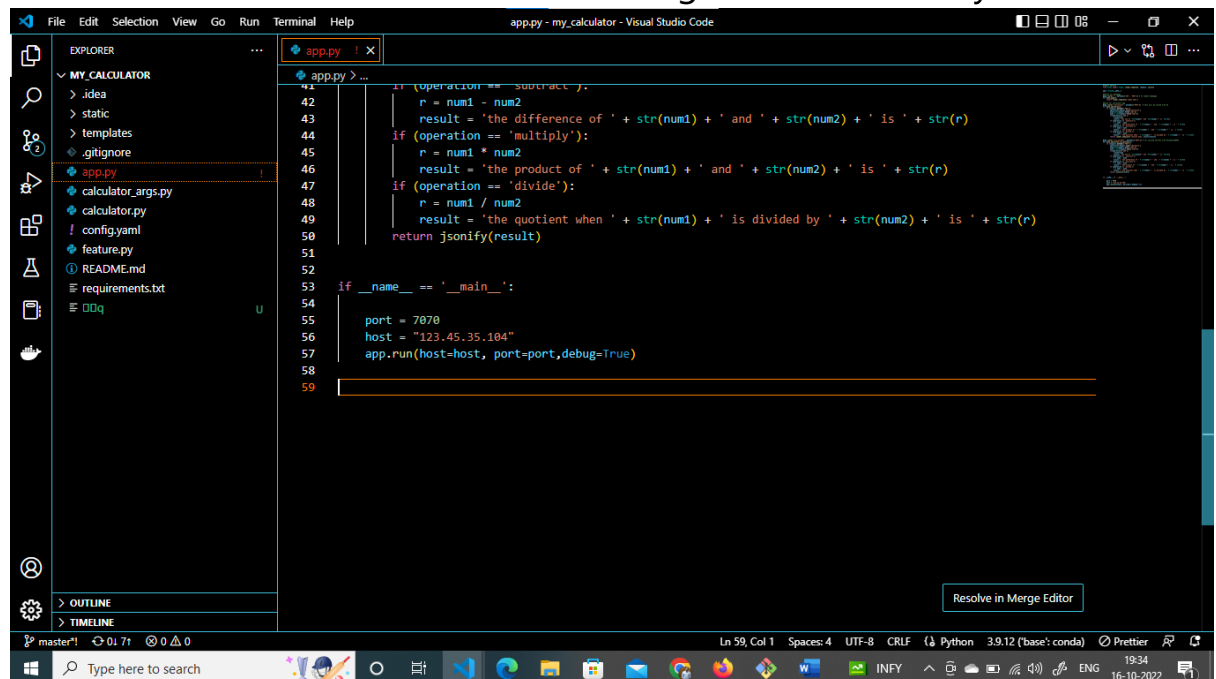
<<< : conflicting parts of current branch on which changes has to be done

||| : merged ancestor

=== : end of ancestor branch/dev branch

>>> : incoming conflicting changes which has to be done on current branch

In such cases we have to edit these configurations manually.



Now we simply add the changes and commit it

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master|MERGING)
$ git add .
warning: in the working copy of '00q', LF will be replaced by CRLF the next time Git touches it

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master|MERGING)
$ git commit -m "Final host port change"
[master 89ab726] Final host port change

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/my_calculator (master)
$
```

Now commit operation is completed by taking care of conflict manually.

## 13. Cloning Repositories

Git clone is a git command line utility which is used to target an existing repository and create a clone, or copy of the target repository.

Git supports a few network protocols to connect to remote repos like connection strings, ssh links etc.

**git clone <Link to Remote repository>**

MINGW64:/c/Users/Akash/Documents/industry\_ready\_projects/1 git

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git
$ git clone https://github.com/TheAlgorithms/Python.git
Cloning into 'Python'...
remote: Enumerating objects: 14563, done.
remote: Total 14563 (delta 0), reused 0 (delta 0), pack-reused 14563
Receiving objects: 100% (14563/14563), 12.54 MiB | 10.04 MiB/s, done.
Resolving deltas: 100% (8446/8446), done.
Updating files: 100% (1140/1140), done.

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git
$
```

## 14. Repositories remote operations

Remote operations are needed to communicate between local and remote repositories

List all Remotes

**git remote -v**

Adding Remotes

**git remote add origin\_name https\_link/sshlink**

MINGW64:/c/Users/Akash/Documents/industry\_ready\_projects/1 git/git\_projects/my\_first\_repo

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo ((v1.0.2))
$ git remote -v
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo ((v1.0.2))
$ git remote add test_origin https://github.com/akash-soni/test_repo.git
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo ((v1.0.2))
$ git remote -v
test_origin      https://github.com/akash-soni/test_repo.git (fetch)
test_origin      https://github.com/akash-soni/test_repo.git (push)
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo ((v1.0.2))
$
```

(Push) -> to push code from local to remote

(Fetch) -> to push code from remote to local

## Adding Remotes

Git remote remove origin name

```
MINGW64:/c/Users/Akash/Documents/industry_ready_projects/1 git/git_projects/my_first_repo

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo ((v1.0.2))
$ git remote remove test_origin

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo ((v1.0.2))
$ git remote -v

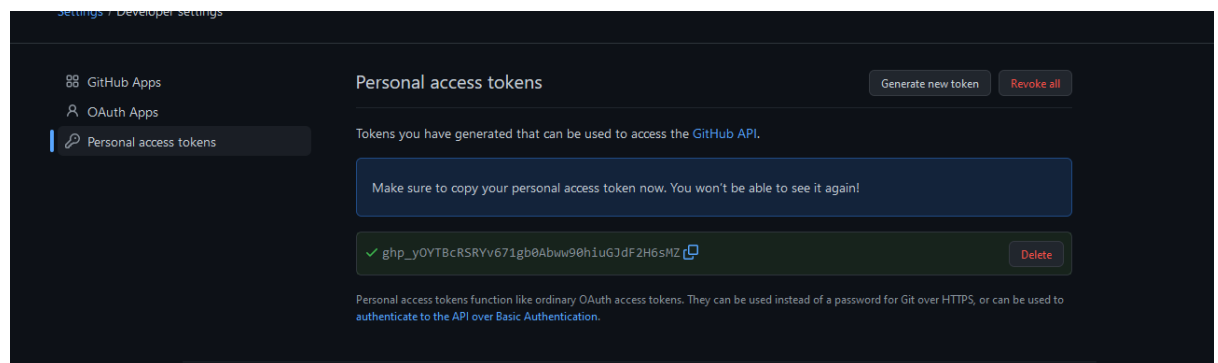
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo ((v1.0.2))
$ |
```

## 15. Personal Access Tokens

Before pushing any code from local to remote repository we need to setup the Personal Access Token in Github.

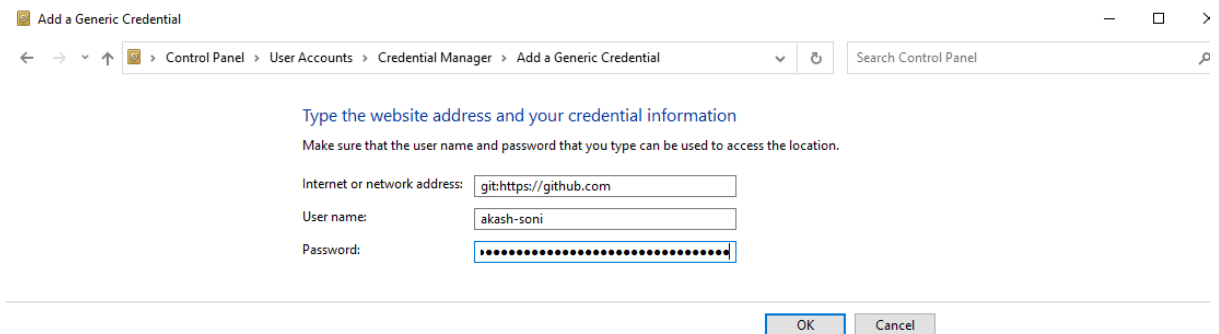
Personal Access Tokens(PAT's) are an alternative to using passwords for authentication to GitHub when using the GitHub API or the command line.

Github.com → Settings → Developer Settings → Personal Access Tokens → Generate New Tokens → Generate





Copy token → go to credential manager → Windows Credentials  
→ Add Generic Credential



```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo ((v1.0.2))
$ git push test_origin master
Enumerating objects: 22, done.
Counting objects: 100% (22/22), done.
Delta compression using up to 4 threads
Compressing objects: 100% (17/17), done.
Writing objects: 100% (22/22), 1.89 KiB | 193.00 KiB/s, done.
Total 22 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), done.
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/akash-soni/test_repo/pull/new/master
remote:
To https://github.com/akash-soni/test_repo.git
 * [new branch]      master -> master
```

## 16. Pull Requests

Generally when working on large projects there are many collaborators who will update the code by merging into the main, but we cannot allow any collaborator to merge directly without verification.

So a collaborator will generate a pull request that he have some code to push. This code will get verified then only it will be merged with the main code.

Pull requests let you tell others about changes you've pushed to a branch in a repository on GitHub. Once a pull request is opened, you can discuss and review the potential changes with collaborators and add follow-up commits before your changes are merged into the base branch.

Example:

Lets create a new branch and do some task and commit it

```
MINGW64:/c/Users/Akash/Documents/industry_ready_projects/1 git/git_projects/my_first_repo

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo ((v1.0.2))
$ git checkout -b akash
Switched to a new branch 'akash'

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (akash)
$ ls
cars.txt  fast_food.txt  file.txt  new_file.txt  plants.txt

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (akash)
$ vi mywork.txt

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (akash)
$ git add .
warning: in the working copy of 'mywork.txt', LF will be replaced by CRLF the next time Git touches it

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (akash)
$ git commit -m "Raise pull Request"
[akash 995da87] Raise pull Request
1 file changed, 2 insertions(+)
create mode 100644 mywork.txt
```

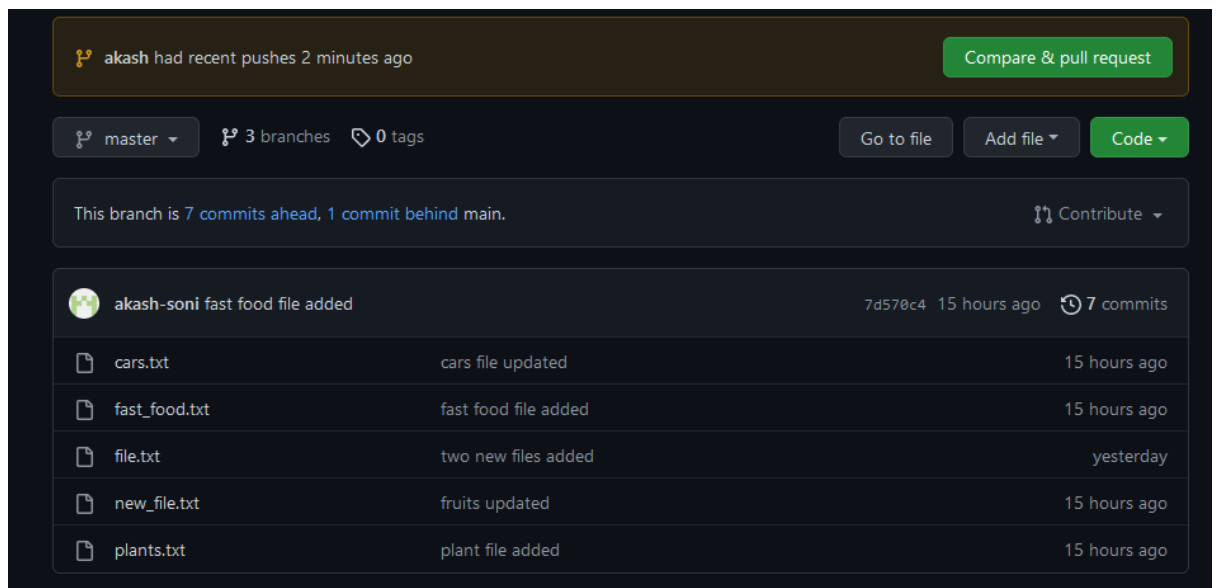
Now we Push the changes and observe what happens at git remote repository

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (akash)
$ git remote
test_origin

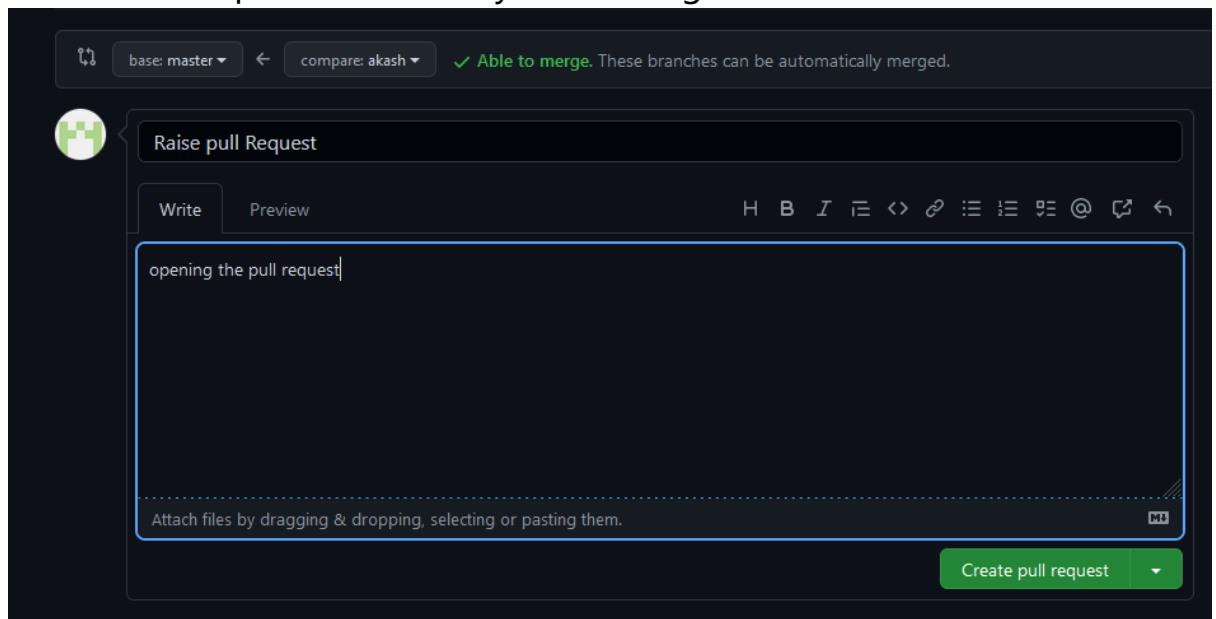
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (akash)
$ git push test_origin akash
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 328 bytes | 82.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'akash' on GitHub by visiting:
remote:   https://github.com/akash-soni/test_repo/pull/new/akash
remote:
To https://github.com/akash-soni/test_repo.git
 * [new branch]      akash -> akash

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (akash)
$
```

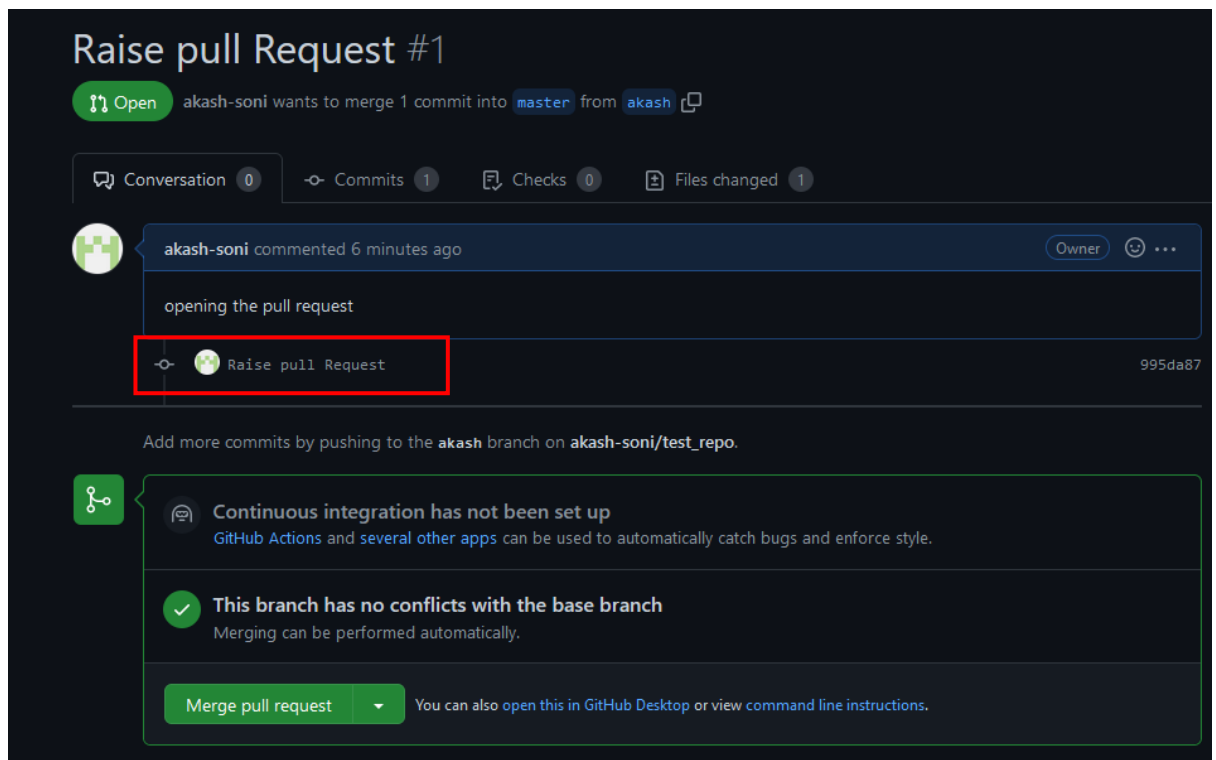
Observe a "Compare & pull request" option popped up



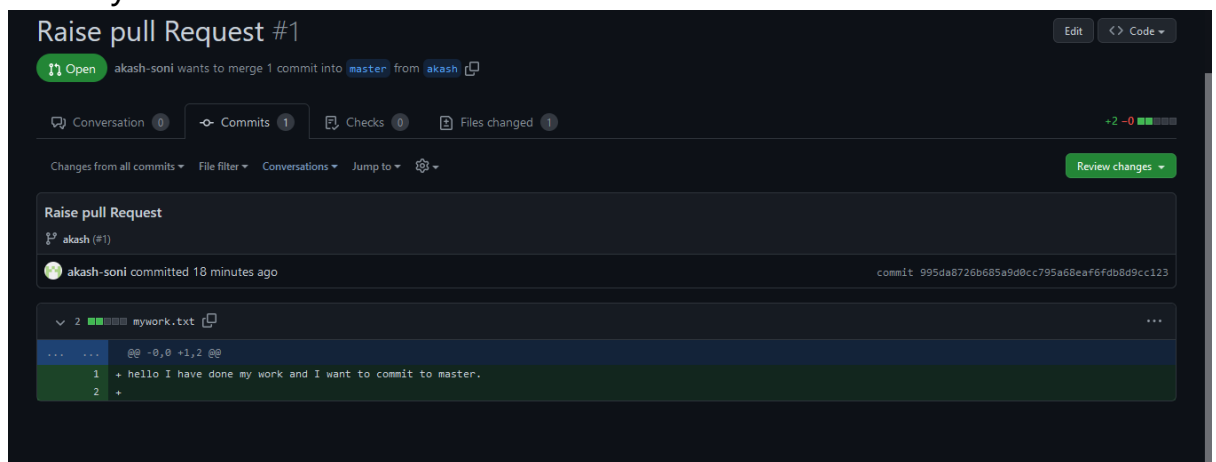
Click on the option and write your message



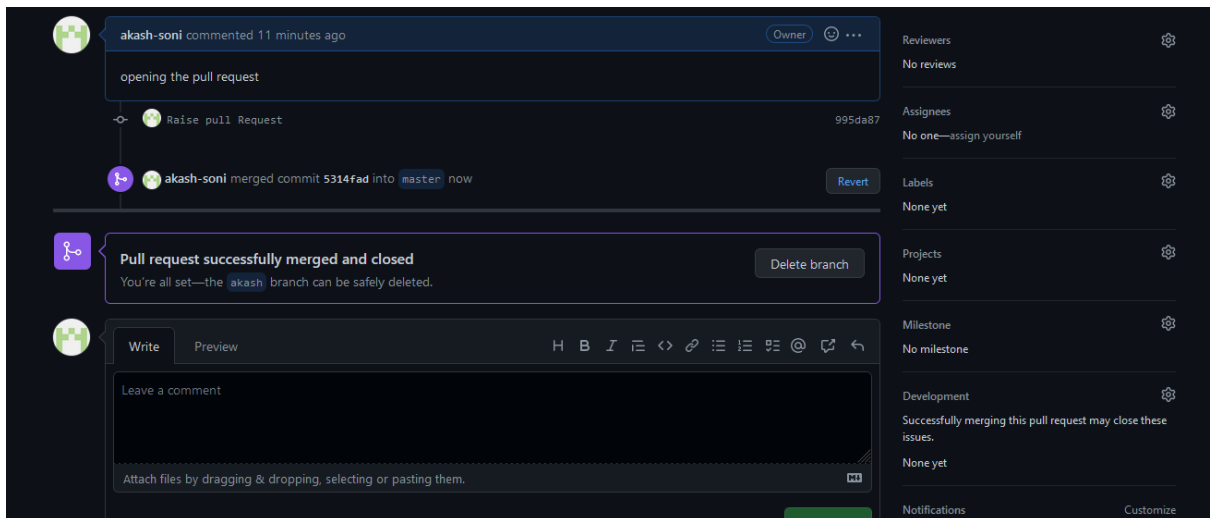
After this a "Merge pull request" option will get activated



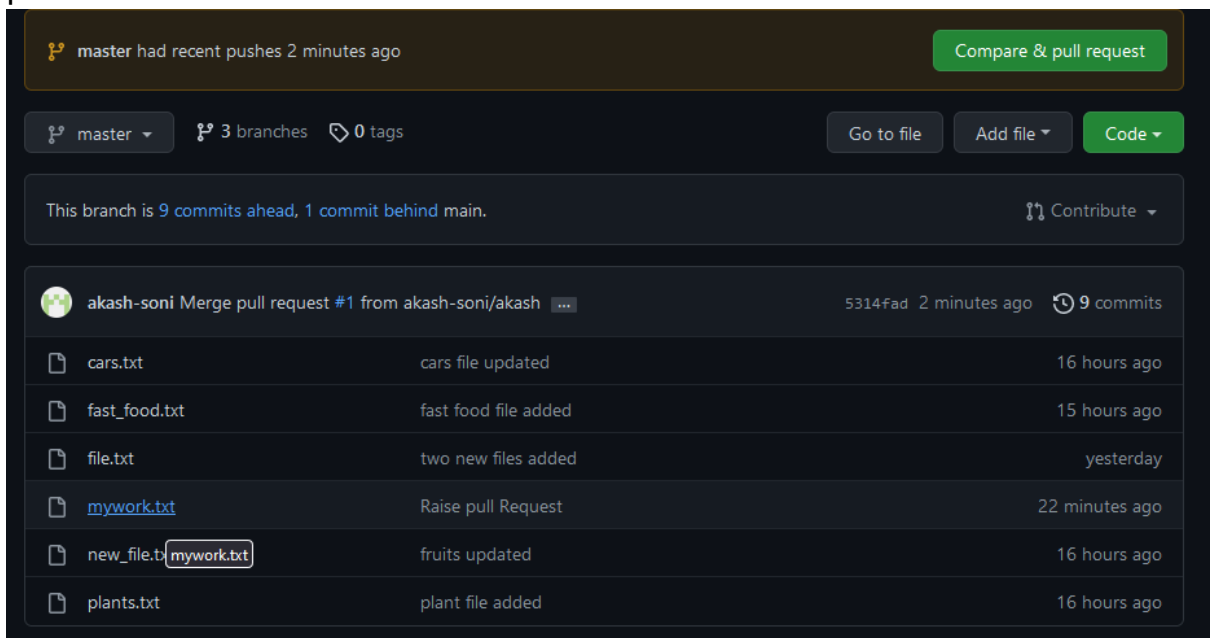
We can now review the Request by clicking on the comment directly



Once review is done, we can Merge with the Master branch



Now the branch is merged and if we want, we can delete the previous branch.



## 17. Fetching and Pulling

**Git fetch** is a command that allows you to download objects from another repository/remote repository to local repository

**Idea is updating the local from remote.**

Example

```
Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git fetch test_origin master
From https://github.com/akash-soni/test_repo
* branch          master       -> FETCH_HEAD

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git merge test_origin/master
Updating 7d570c4..5314fad
Fast-forward
 mywork.txt | 2 ++
 1 file changed, 2 insertions(+)
 create mode 100644 mywork.txt

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ |
```

We can observe that we want to fetch contents of master repo into local repo.

After that we have simply merged contents of our master branch with local repository.

Notice mywork.txt was not part of master branch in local but now it is also present in local Repository as well.

**Git pull** is a command that allow you to fetch from and integrate with another repository/remote repo or local branch.

(Fetching requires **1<sup>st</sup> git fetch** and then **git merge** while Pull includes both **git pull**(Fetching + Merging))

A git pull is actually a git fetch followed by an additional action(s) – typically git merge.

Example :-

Lets create a Readme.md in master branch.

```
MINGW64:/c/Users/Akash/Documents/industry_ready_projects/1 git/git_projects/my_first_repo

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ ls
cars.txt  fast_food.txt  file.txt  mywork.txt  new_file.txt  plants.txt

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ |
```

In the above observe that there is no Readme.md in local repo  
Now we will use git pull as follows:

```
MINGW64:/c/Users/Akash/Documents/industry_ready_projects/1 git/git_projects/my_first_repo

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ ls
cars.txt  fast_food.txt  file.txt  mywork.txt  new_file.txt  plants.txt

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ git pull test_origin master
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 684 bytes | 31.00 KiB/s, done.
From https://github.com/akash-soni/test_repo
* branch          master       -> FETCH_HEAD
   5314fad..b2c6b15 master       -> test_origin/master
Updating 5314fad..b2c6b15
Fast-forward
 README.md | 2 ++
1 file changed, 2 insertions(+)
create mode 100644 README.md

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$ ls
README.md  cars.txt  fast_food.txt  file.txt  mywork.txt  new_file.txt  plants.txt

Akash@DESKTOP-N9U9FMQ MINGW64 ~/Documents/industry_ready_projects/1 git/git_projects/my_first_repo (master)
$
```

Observe that Readme.md is now also part of local repo

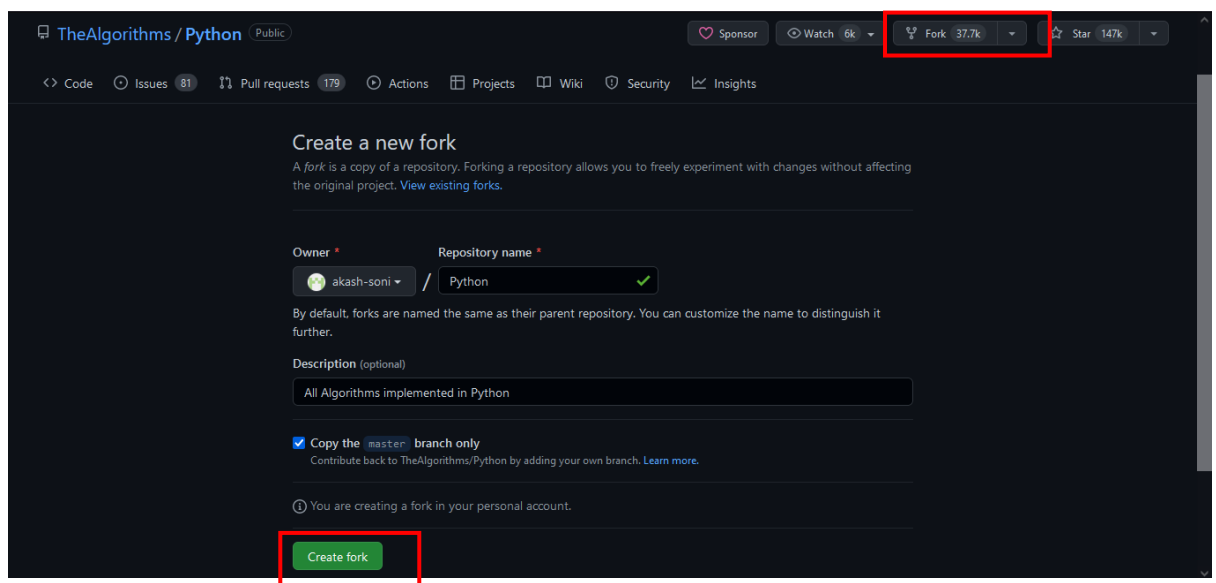
## 18. Forking

A fork is a copy of repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Reasons for forking the repository:

- Propose changes to someone else's project.
- Use an existing project as a starting point

Basically, copying entire repo from original account to your repo



When we create a fork it will create an exact same repo into the branch.