

# Software Engineering Tools Lab

## Assignment No-4

**PRN: 2020BTECS00033**

**Name: Prathamesh Raje**

**Batch: T6**

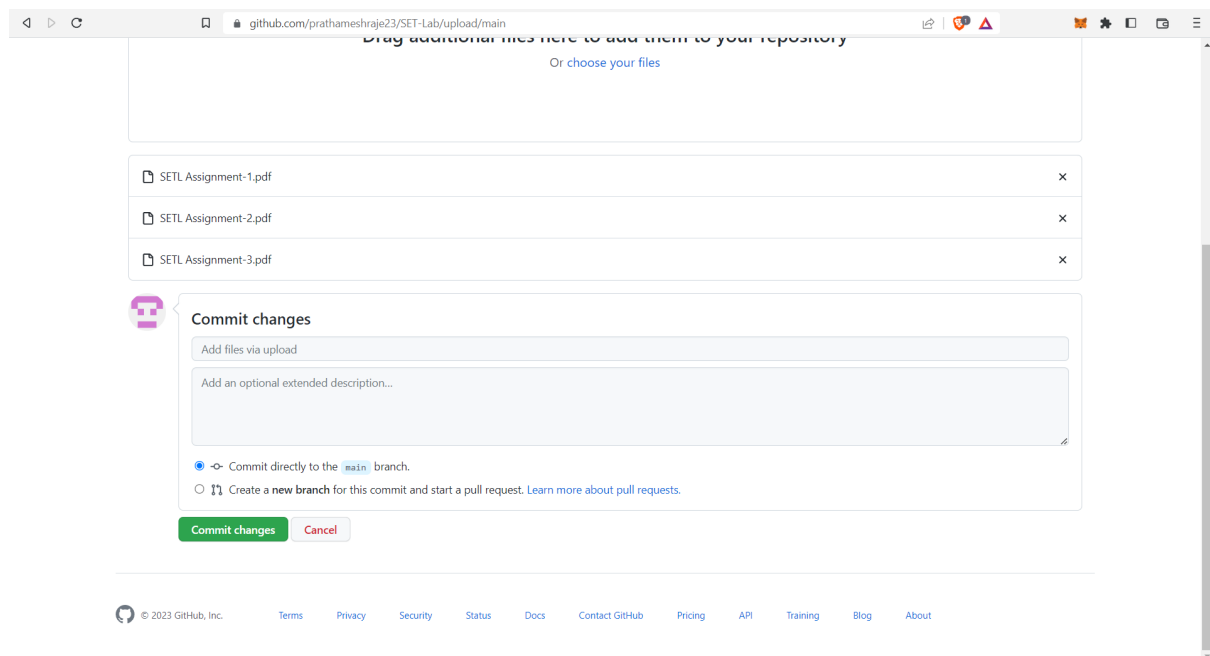
### Title - GitHub

Q 1. Create a repository on GitHub named SET Lab and add files into it (you can add implementation files of previous assignment) perform below operations on it.

(Add screenshot as an answer to every question)

Ans.


#### 1. Perform commit on added files



## 2. Perform update to the existing files (show history)

30  
31  
32 This is update in readme file.

Attach files by dragging & dropping, selecting or pasting them.



### Commit changes

Update README.md

Add an optional extended description...

☒ Commit directly to the `main` branch.  
☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes

Cancel

## 3. Create another branch

main 1 branch 0 tags

Go to file Add file <> Code

Switch branches/tags

new

Branches Tags

Create branch: new from 'main'

View all branches

0b956c1 1 minute ago 3 commits

Update README.md 1 minute ago

Add files via upload 3 minutes ago

Add files via upload 3 minutes ago

Add files via upload 3 minutes ago

## 4. Create pull request

prathameshraje23 / SET-Lab Public

Pin Unwatch 1 Fork 0 Star 0

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

### Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: main compare: new ✓ Able to merge. These branches can be automatically merged.

Added CPP file

Write Preview

H B I <> @ ↺ ↻

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

Reviewers

No reviews

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Use [Closing keywords](#) in the description to automatically close issues.

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

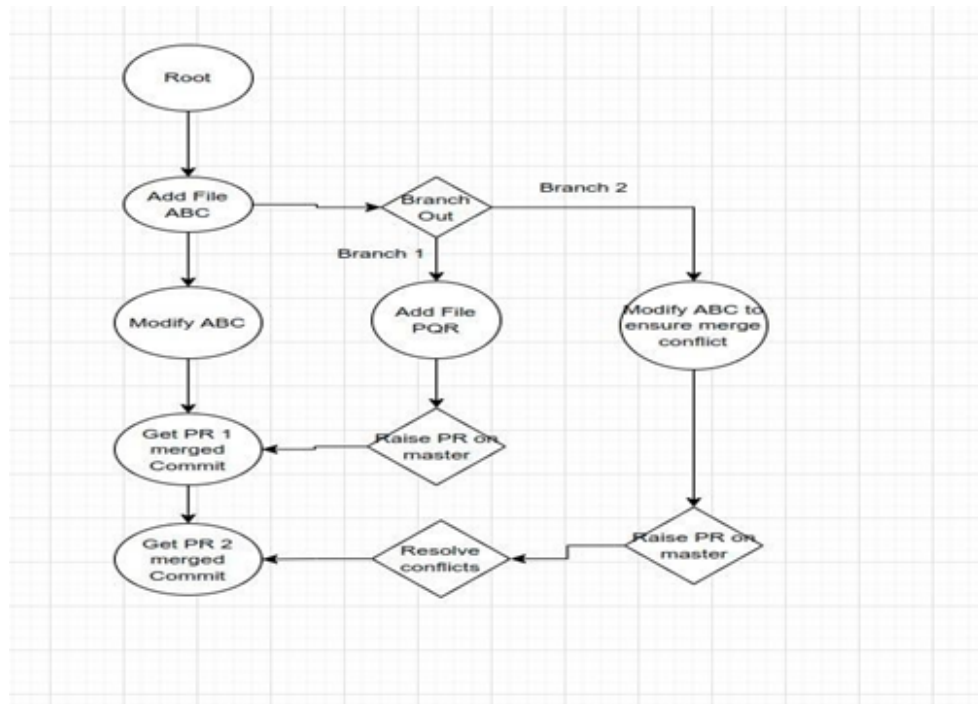
## 5. Perform merging of both branches

The screenshot shows a GitHub pull request titled "Added CPP file #1". The pull request is from the "new" branch to the "main" branch, created by user "prathameshraj23". The interface includes a header with "Open" and "Code" buttons, a progress bar showing "+92 -0" changes, and a list of metrics: Conversation (0), Commits (1), Checks (0), and Files changed (1). Below the header, there is a comment section with a comment from "prathameshraj23" stating "No description provided." and a button to "Add files via upload". A green box highlights the "Merge pull request" button, with a note that the branch has no conflicts with the base branch. To the right, there are settings for reviewers, assignees, labels, projects, milestones, and notifications. The bottom section shows a "Write" and "Preview" tab for adding a comment.

## 6. Perform Fork operation

The screenshot shows the "Create a new fork" form on GitHub. The form is for the repository "prathameshraj23 / SET-Lab". The "Owner" is "Walchand-Linux-Users-Group" and the "Repository name" is "SET-Lab". A green checkmark indicates that the repository name is valid. Below the form, there is a description of a fork and a checkbox labeled "Copy the main branch only", which is checked. A note at the bottom states: "You are creating a fork in the Walchand-Linux-Users-Group organization (Walchand College of Engineering, Sangli)." A green "Create fork" button is at the bottom.

Q 2. For the diagram given below create a GitHub repository and perform operations given in the diagram. (Perform commit operations as given)(Add screenshots as an answer to this question)



## 1. ABC.txt in Main Branch:

main 1 branch 0 tags [Go to file](#) [Add file](#) [Code](#)

prathameshraje23 Initial commit a07daa2 now 1 commit

ABC.txt	Create ABC.txt	now
README.md	Initial commit	now

## 2. Creating Branch:

main 1 branch 0 tags [main](#) 2 branches 0 tags

Switch branches/tags

B1

Branches Tags

Create branch: B1 from 'main'

[View all branches](#)

Switch branches/tags

B2

Branches Tags

Create branch: B2 from 'main'

[View all branches](#)

## 3. Adding PQR.txt in branch B1

## 4. Modifying ABC.txt in Main branch

SET-Lab-Q.2 / ABC.txt in main

<> Edit file

Preview changes

```
1 Hello, I am ABC
2 Modifying in Main branch
```

## 5. Modifying ABC.txt in B2 branch to ensure merge conflict

SET-Lab-Q.2 / ABC.txt in B2

<> Edit file

Preview changes

```
1 Hello, I am ABC
2 Modifying ABC.txt in B2 branch.
```

## 6. Raising and merging PR1 from B1

### Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: main

compare: B1

✓ Able to merge. These branches can be automatically merged.

Added PQR.txt

Write

Preview

H B I ≡ <> 🔗 ≡ ≡ @ ↗ ↶

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

## Added PQR.txt #1

Open prathameshraje23 wants to merge 1 commit into `main` from `B1`

Conversation **0** Commits **1** Checks **0** Files changed **1**



prathameshraje23 commented now

Owner Tip ...

No description provided.

Added PQR.txt

Verified

7dbf6f7

Add more commits by pushing to the `B1` branch on prathameshraje23/SET-Lab-Q2.



Require approval from specific reviewers before merging

Branch protection rules ensure specific people approve pull requests before they're merged.

Add rule

×

Continuous integration has not been set up

GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

This branch has no conflicts with the base branch

Merging can be performed automatically.

Merge pull request

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

## 7. Raising PR2 from B2 and resolving merge conflict

### Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: main compare: B2 Can't automatically merge. Don't worry, you can still create the pull request.



Update ABC.txt

Write Preview

H B I @

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

## 8. Conflict

Add more commits by pushing to the `B2` branch on prathameshraje23/SET-Lab-Q2.



This branch has conflicts that must be resolved

Resolve conflicts

Use the [web editor](#) or the [command line](#) to resolve conflicts.

Conflicting files

ABC.txt


Merge pull request

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

## 9. Resolve Conflict


## Update ABC.txt #2


Resolving conflicts between `B2` and `main` and committing changes → `B2`


1 conflicting file	ABC.txt
 ABC.txt ABC.txt	<pre>1 Hello, I am ABC 2 &lt;&lt;&lt;&lt;&lt;&lt; B2 3 Modifying ABC.txt in B2 branch. 4 ===== 5 Modifying in Main branch 6 &gt;&gt;&gt;&gt;&gt;&gt; main 7</pre>


### 10. Final merge after resolving

Add more commits by pushing to the `B2` branch on `prathameshraje23/SET-Lab-Q2`.



 Require approval from specific reviewers before merging  
[Branch protection rules](#) ensure specific people approve pull requests before they're merged. Add rule ×

 Continuous integration has not been set up  
[GitHub Actions](#) and [several other apps](#) can be used to automatically catch bugs and enforce style.

 This branch has no conflicts with the base branch  
Merging can be performed automatically.

Merge pull request ▼ You can also open this in [GitHub Desktop](#) or view [command line instructions](#).

Q 3. What is GitHub desktop? How to install GitHub on a local machine? Install GitHub on your local machine and access the repository created in question no 1 (add screenshots).

Ans.

GitHub Desktop is an open-source application that lets you interact with GitHub via a graphic user interface (GUI) instead of relying on a command line or web browser. GitHub Desktop incentivizes you and your team to work together while employing best practices with Git and GitHub.

GitHub Desktop enables developers to activate commands such as repository creation, pull requests, and commits with just a simple click. This extra convenience adds an extra element of flexibility to working with Git and collaborating with other developers.

So, to sum it up, Git is a version control system that helps you manage your code and keep track of it, and GitHub is a cloud-based hosting platform that enables developers to manage their Git repositories. GitHub Desktop is an application that lets users interact better with GitHub through a GUI.

The installation of GitHub Desktop is as simple as any other Windows application installation. All you need to do is:

Open a browser.

Visit [desktop.github.com](https://desktop.github.com).

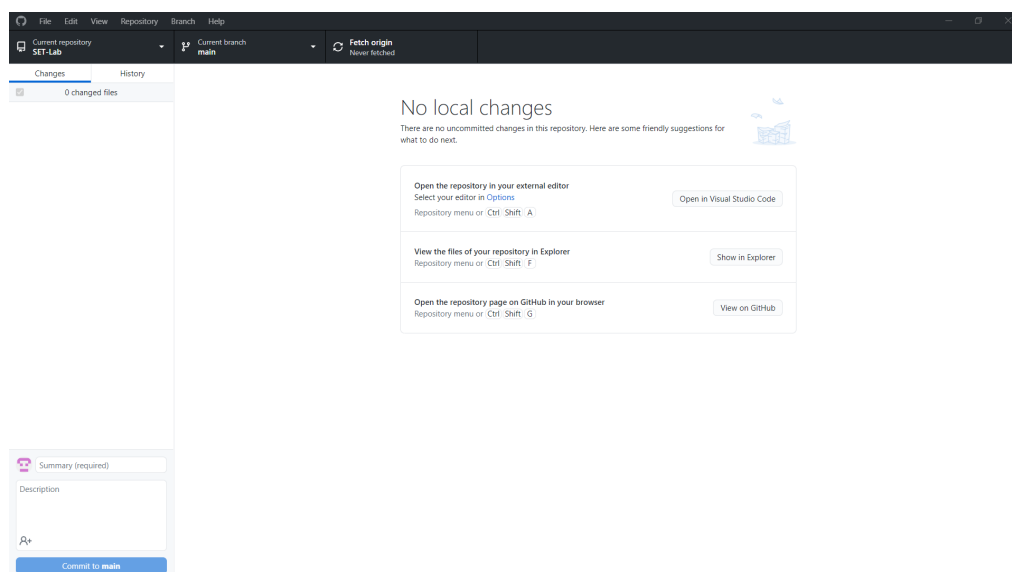
Click Download for Windows (64bit).



When prompted, click Run.

Allow the installation to download and install.

Once the installation completes, GitHub Desktop will launch.





Q 4. Differentiate between GitHub, Git and GitLab.

Ans.

Git	GitHub
1. It is a software	1. It is a service
2. It is installed locally on the system	2. It is hosted on Web
3. It is a command line tool	3. It provides a graphical interface
4. It is a tool to manage different versions of edits, made to files in a git repository	4. It is a space to upload a copy of the Git repository
5. It provides functionalities like Version Control System Source Code Management	5. It provides functionalities of Git like VCS, Source Code Management as well as adding few of its own features

GitHUb VERSUS GiTLab	
GitHub	GitLab
A web based hosting service for version control using Git	A web based Devops lifecycle tool that provides a Git repository manager
Written in Ruby	Written in Ruby, Go and Vue.js
Launched in year 2008	Launched in year 2011
Provides an easy to use intuitive UI	Provides more convenient UI than GitHub
More popular than GitLab	Less popular than GitHub
Provides various third party integrations for continuous integration and continuous delivery work	Offers its own pre-built continuous integration and continuous delivery support
	Visit <a href="http://www.PEDIAA.com">www.PEDIAA.com</a>

Q 5. What is version control? Explain with examples.

Ans.

1. Version control (also known as revision control or source control) is a category of processes and tools designed to keep track of multiple different versions of software, content, documents, websites and other information in development.
2. Any system that provides change tracking and control over programming source code and documentation can be considered version control software.

3. The practice has been a part of creative processes almost as long as writing has existed.
4. The purpose of version control is ensuring that content changes under development go as planned.
5. While version control is often carried out by a separate application, it can also be embedded into programs such as integrated development environments (IDEs), word processors, spreadsheets and, especially, collaborative web documents and pages.
6. Version control allows servers in multiple locations to run different versions on different sites, even while those versions are being updated simultaneously.
7. The most powerful and complex version control systems are used in software development.
8. Version control often operates by locking files and using a check-out / check-in system for changed versions. Versions may be identified by labels or tags; approved versions or those that are especially significant may be designated baselines.
9. Checked-out versions may be worked on by different groups or individuals as branching code from the main trunk.
10. When versions are checked out and checked in, the first to check in is sure to succeed. If other versions are checked out, some systems may provide for version merging to allow further changes to be added into the central repository.
11. Another method used in version control is branching, in which programs in development are copied for development in parallel versions, retaining the original and working on the branch or making different changes to each.
12. Each copy is considered a branch; the original program from which the branch is taken is referred to as the trunk, the baseline, the mainline or the master.
13. Version control is generally based on a client-server model. Another method is distributed version control, in which all copies are in a codebase repository and changes are synchronised through patches or changes shared from peer to peer.