

## FRONT END DEVELOPMENT FRAMEWORKS

### WEEK-2 Experiment

#### Aim:

To create multiple feature branches, add specific HTML content in each, and merge them into the main branch using Git.

#### Description:

To begin, create and switch to a new branch named about-feature using the command `git checkout -b about-feature`. In this branch, edit or create an `index.html` file to include an "About" section with appropriate HTML such as `<h2>About Us</h2>` and a paragraph tag `<p>` describing your project or team. Save the file and commit the changes with a meaningful message like `git commit -am "Add About section to index.html"`. Next, switch back to the main branch using `git switch main` and create a new branch `contact-feature` from it using `git checkout -b contact-feature`. In this branch, modify `index.html` to add a contact form using HTML elements like `<form>`, with input fields for Name, Email, and a textarea for Message, then commit these changes. After committing, switch to the main branch and use `git merge about-feature` and `git merge contact-feature` to bring the changes into the main branch one by one. If any merge conflicts occur (such as overlapping changes in `index.html`), resolve them manually by editing the file, removing conflict markers (`<<<<<<<<, =====, >>>>>>>>`), and preserving the correct content. Finally, save and commit the resolved file, then verify that the final version of `index.html` contains both the About section and the Contact form.

#### Branching:

First, we are going to talk about branching.

Branching is a powerful feature in Git and GitHub that allows you to create separate lines of development within a repository. It enables you to work on different features, bug fixes, or experiments without affecting the main codebase. Here's a brief overview of some common commands and their explanations for branching in GitHub:

#### Create a new branch:

- **Command:** `git branch <branch_name>`
- **Explanation:** This command creates a new branch with the specified name based on the current branch.
- **Example:** `git branch dev`
- **Output:**

```
~/branching$ git branch dev  
~/branching$
```

## List all branches:

- **Command:** `git branch`
- **Explanation:** This command lists all the branches in the repository, highlighting the current branch.
- **Example:** `git branch`
- **Output:** Based on the provided screenshot, it is apparent that the current branch is "master" as highlighted.

```
decepticon:~/branching$ git branch
de1
dev
* master
```

## Switch to a branch:

- **Command:** `git checkout <branch_name>`
- **Explanation:** This command switches to the specified branch, allowing you to start working on it.
- **Example:** `git checkout dev`
- **Output:** From the given screenshot, you can see that now I am currently in the "dev" branch.

```
decepticon:~/branching$ git checkout dev
Switched to branch 'dev'
decepticon:~/branching$ git branch
de1
* dev
master
decepticon:~/branching$
```

- Please note that an alternative command for switching branches is `git switch <branch_name>`.

```
decepticon:~/branching$ git switch de1
Switched to branch 'de1'
decepticon:~/branching$ git branch
* de1
  dev
  master
decepticon:~/branching$
```

## Create a new branch and switch to it:

- **Command:** `git checkout -b <branch_name>`
- **Explanation:** This command creates a new branch with the specified name and immediately switches to it.
- **Example:** `git checkout -b dev1`
- **Output:** As shown, a new branch named "dev1" has been created, and we have switched to this branch successfully.

```
decepticon:~/branching$ git checkout -b dev1
Switched to a new branch 'dev1'
decepticon:~/branching$ git branch
  de1
  dev
* dev1
  master
```

## Delete a branch:

- **Command:** `git branch -d <branch_name>`
- **Explanation:** This command deletes the specified branch. Note that you cannot delete the branch you are currently on.
- **Example:** `git branch -d de1`
- **Output:** The "dev" branch has been deleted.

```
decepticon:~/branching$ git branch -d de1
Deleted branch de1 (was 995a230).
decepticon:~/branching$ git branch
  dev
* dev1
  master
```

But the "dev1" branch is not deleted since we are currently on the "dev1" branch.

```
decepticon:~/branching$ git branch -d dev1
error: Cannot delete branch 'dev1' checked out at '/home/decepticon/branching'
```

## Pull Request:

Pull requests (PRs) are a way to propose changes and collaborate on code within a Git repository. They allow team members to review, discuss, and provide feedback on the proposed changes before merging them into the main codebase.

To create a pull request, follow these steps:

1. Make sure you are on the branch where your changes are located (e.g., "dev").

```
decepticon:~/branching$ git branch
* dev
  dev1
  master
decepticon:~/branching$
```

## Push the branch to the remote repository:

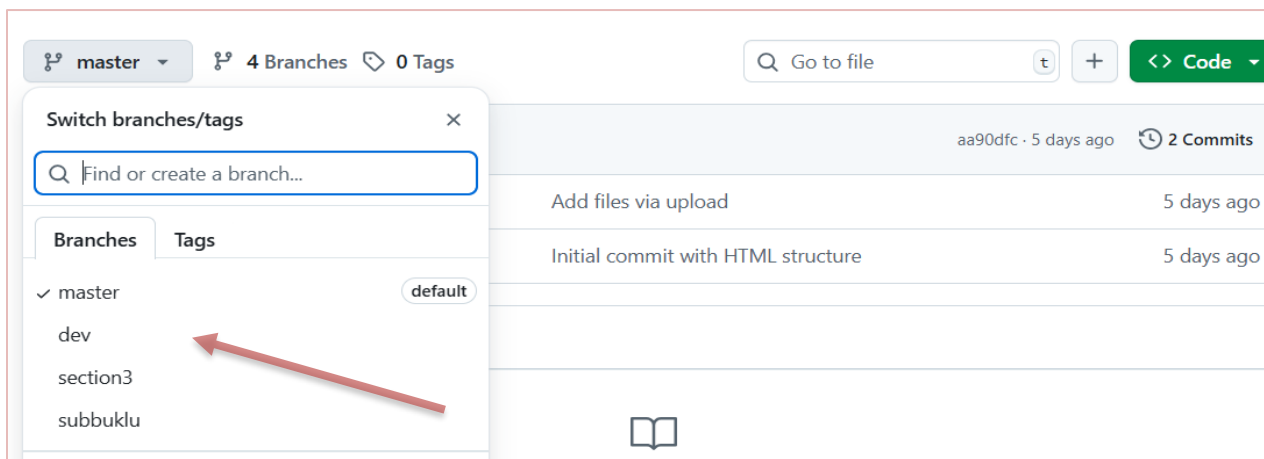
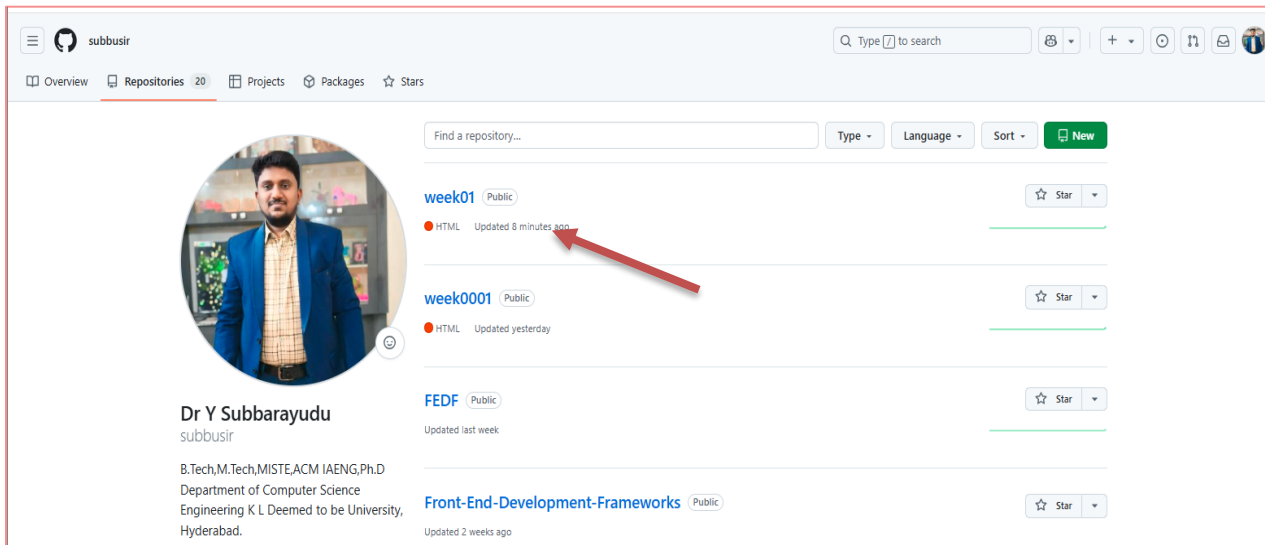
- **Command:** `git push origin <branch_name>`
- **Explanation:** This pushes the branch to the remote repository on GitHub.
- **Example:** `git push origin dev`
- **Output:**

```

decepticon:~/branching$ git push origin dev
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 293 bytes | 293.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'dev' on GitHub by visiting:
remote:   https://github.com/sajdakabir/branching/pull/new/dev
remote:
To github.com:sajdakabir/branching.git
 * [new branch]      dev -> dev

```

2. On the GitHub repository page, locate the branch you pushed and click on the



## Merging:

Merging combines the changes from one branch into another. After a pull request is reviewed and approved, you can merge it into the target branch (e.g., main) to incorporate the changes.

### Here's how to merge a pull request:

1. On the GitHub repository page, navigate to the pull requests.

Here you can see all the Pull requests.

2. Verify that the changes are correct and ready to be merged.
3. Click the "Merge pull request" button.

Branch	Updated	Check status	Behind / Ahead	Pull request
dev	18 minutes ago		1   0	
section3	21 minutes ago		1   0	
subbuklu	33 minutes ago		0   0	

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff comparisons](#).

base: master ← compare: dev

There isn't anything to compare.

master is up to date with all commits from dev. Try [switching the base](#) for your comparison.

subbusir / week01

Type to search

[Code](#) [Issues](#) [Pull requests 1](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

## Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff comparisons](#).

base: dev

compare: master

Add files via upload #1

hai this is subbarayudu

View pull request

1 commit

1 file changed

1 contributor

## Add files via upload #1

Open subbusir wants to merge 1 commit into dev from master

Conversation 0

Commits 1

Checks 0

Files changed 1

subbusir commented 2 minutes ago

hai this is subbarayudu

Owner

Add files via upload

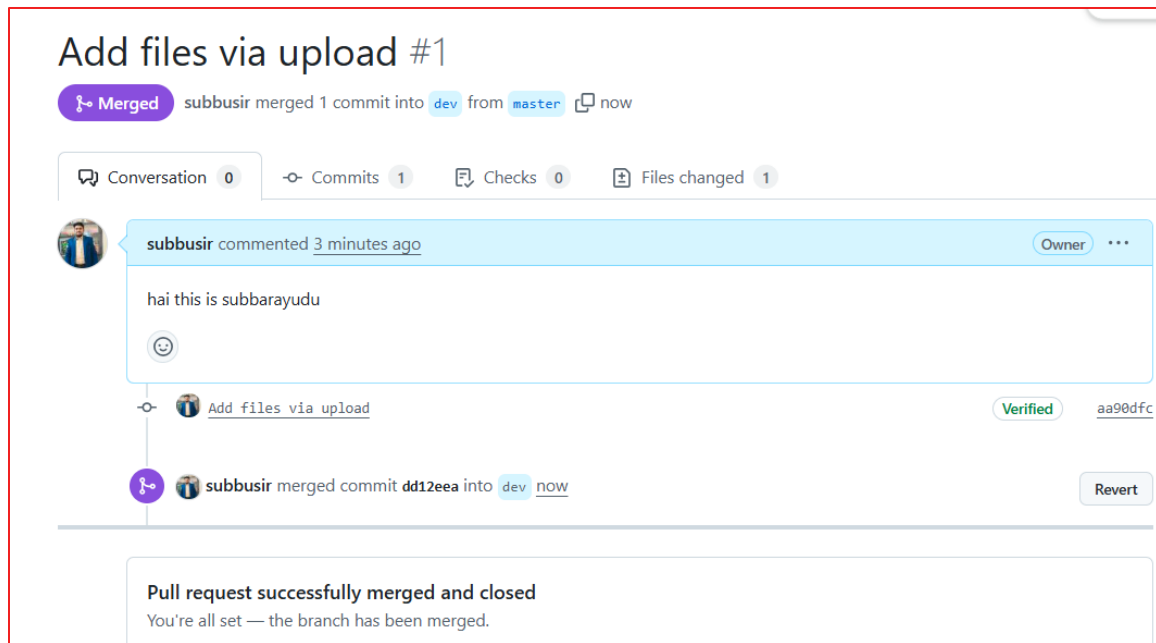
Verified aa90dfc

No conflicts with base branch

Merging can be performed automatically.

Merge pull request

You can also merge this with the command line. [View command line instructions.](#)



### Tasks:

1. Create and switch to a new branch named about-feature.
2. In the about-feature branch, add an About section using appropriate HTML headings and paragraph text.
3. Commit your changes with a meaningful message.
4. Create another branch contact-feature from main.
5. In the contact-feature branch, add a Contact form with fields for Name, Email, and Message.
6. Commit your changes and merge both branches into the main branch one by one.
7. Resolve any merge conflicts and verify the final version of index.html.

### Solutions:

#### 1 Create and switch to about-feature branch

*A new branch can be created using the branch command.*

**Syntax:**

```
$ git branch branchname
```


**Example:**

```
$ git branch about-feature
```

This command creates a new branch named **about-feature**, but **does not switch** to it automatically.



*To switch to the newly created branch, use the checkout command:*




```
Admin@subbusir MINGW64 ~/week01 (dev)
$ git checkout about-feature
Switched to branch 'about-feature'
Admin@subbusir MINGW64 ~/week01 (about-feature)
```

**Example:**

\$ git checkout about-feature

**Sample Output:**



```
Admin@subbusir MINGW64 ~/week01 (about-feature)
$ git branch contact-feature
```

**Tip:** You can combine both steps using:

\$ git checkout about-feature

```
Admin@subbusir MINGW64 ~/week01 (about-feature)
$ git checkout contact-feature
Switched to branch 'contact-feature'
```

This command creates and switches to the new branch in one step.

**2 . In the about-feature branch, add an About section using appropriate HTML headings and paragraph text.**

*Add About section in HTML (index.html)*

```
<!DOCTYPE html>
<html lang="en">
<body>
  <h2>About Us</h2>
  <p>Welcome to our website. We are committed to delivering high-quality web
  experiences using modern frameworks and tools.</p>
</body>
</html>
```

### 3. Commit your changes with a meaningful message.

#### *Commit changes*

```
$ git add index.html
```

```
$ git commit -m "Added About section"
```

### 4 Create another branch contact-feature from main.

```
$ git branch contact-feature
```

```
$ git checkout contact-feature
```

```
Admin@subbusir MINGW64 ~/week01 (about-feature)
$ git checkout contact-feature
Switched to branch 'contact-feature'

Admin@subbusir MINGW64 ~/week01 (contact-feature)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
```

### 5. In the contact-feature branch, add a Contact form with fields for Name, Email, and Message.

#### *Add Contact form in HTML (index.html)*

```
<h2>Contact Us</h2>
```

```
<form>
```

```
  <label for="name">Name:</label><br />
```

```
  <input type="text" id="name" name="name"><br />
```

```
  <label for="email">Email:</label><br />
```

```
  <input type="email" id="email" name="email"><br />
```

```
  <label for="message">Message:</label><br />
```

```
  <textarea id="message" name="message"></textarea><br />
```

```
  <input type="submit" value="Submit">
```

```
</form>
```

### 6. Commit your changes and merge both branches into the main branch one by one.

#### *Commit Contact form*

```
git add index.html
```

```
git commit -m "Added Contact form"
```

### Merge both branches into master

git checkout master

```
Admin@subbusir MINGW64 ~/week01 (contact-feature)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

Admin@subbusir MINGW64 ~/week01 (master)
$ |
```

git merge about-feature

default branch

```
Admin@subbusir MINGW64 ~/week01 (master)
$ git merge about-feature
Already up to date.

Admin@subbusir MINGW64 ~/week01 (master)
$ |
```

git merge contact-feature

```
Admin@subbusir MINGW64 ~/week01 (master)
$ git merge contact-feature
Already up to date.

Admin@subbusir MINGW64 ~/week01 (master)
$ |
```

### 7. Resolve any merge conflicts and verify the final version of index.html.

git add index.html

git commit -m "Merged about-feature and contact-feature with resolved conflicts"

#### Example of Resolved index.html:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <title>My Website</title>
```

```
</head>
```

```
<body>
```

```
  <h2>About Us</h2>
```

```
  <p>Welcome to our website. We are committed to delivering high-quality web
  experiences using modern frameworks and tools.</p>
```

```
  <h2>Contact Us</h2>
```

```
  <form>
```

```
    <label for="name">Name:</label><br />
```

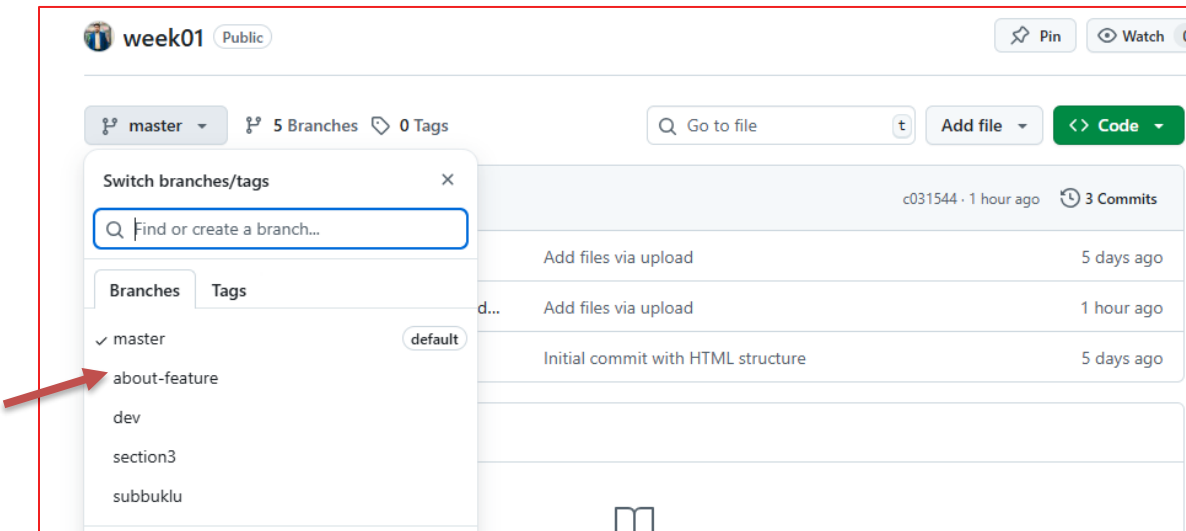
```
<input type="text" id="name" name="name"><br />

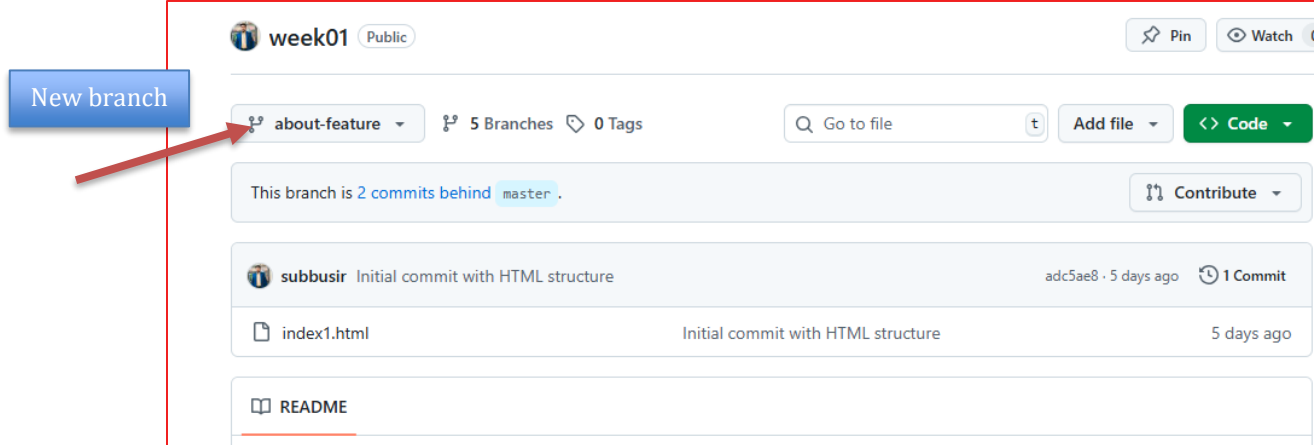
<label for="email">Email:</label><br />
<input type="email" id="email" name="email"><br />

<label for="message">Message:</label><br />
<textarea id="message" name="message"></textarea><br />

<input type="submit" value="Submit">
</form>
</body>
</html>
```

```
Admin@subbusir MINGW64 ~/week01 (master)
$ git push origin about-feature
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 391 bytes | 130.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'about-feature' on GitHub by visiting:
remote:   https://github.com/subbusir/week01/pull/new/about-feature
remote:
To https://github.com/subbusir/week01.git
 * [new branch]      about-feature -> about-feature
```





### Result:

As a result of following the branching and merging process, the final index.html file in the main branch successfully includes both the "About" section and the "Contact" form. The About section provides an overview using proper headings and paragraph text, while the Contact form includes input fields for Name, Email, and Message, allowing users to submit their details. All changes from the about-feature and contact-feature branches were committed with meaningful messages and merged into the main branch. Any merge conflicts encountered during the process were resolved correctly, resulting in a clean and fully functional HTML page that reflects contributions from both feature branches.