Github Lesson Lab

## 

## Creating a Repository

### Create and clone repository in GitHub

*Get into pairs and label yourselves as A and B,*

1. Enter GitHub web and enter “wad2-test” as the repository name and click “Create Repository”
2. Once successfully created, you should enter your new repository home page. Click “Settings” on the top > “Collaborators” and add your partner to your repository. Share your repository link (eg. <https://github.com/jermaineclx/wad2-test.git>) to your partner.
3. Open GitHub Desktop and click on “Clone a Repository from the Internet” > “URL” and paste your partner’s repository URL then “Clone”
4. You should see the “Current Repository” at the top left hand corner to be wad2-test once successfully cloned.
5. Open the repository in Visual Studio Code for easier editing of files.

## Working on the Repository

### Commit to local repo

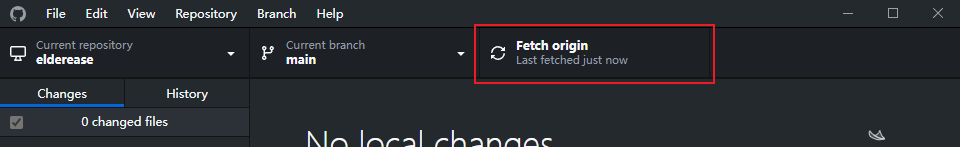
1. Both student A and student B download and place the index.html and styles.css files into the repository folder
2. Student A to complete **index.html**, by filling up bootstrap classes;
3. Student B to complete **styles.css**, by defining style for <header> and <div>
4. If in group of 3, Student C to create new file **password.env** inside the file
5. On Github Desktop, enter commit message and commit the changes to local repo

| Sample of completed index.html |
| --- |
| <!DOCTYPE html>  <html >  <head>  <title>Welcome to Our Page</title>  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">  <link rel="stylesheet" href="styles.css">  </head>  <ma>  <header>  <h1>Welcome to Our Page</h1>  </header>  <main>  <div class="container mt-4">  <div class="row">  <div class="col-md-6">  <p>This is the first paragraph. It is placed in the first column of the grid.</p>  </div>  <div class="col-md-6">  <p>This is the second paragraph. It is placed in the second column of the grid.</p>  </div>  </div>  </div>  </main>  </body>  </html> |
| Sample of completed styles.css |
| header {  background-color: green;  color: white;  text-align: center;  padding: 10px;  }  div{  border: 2px solid green;  } |

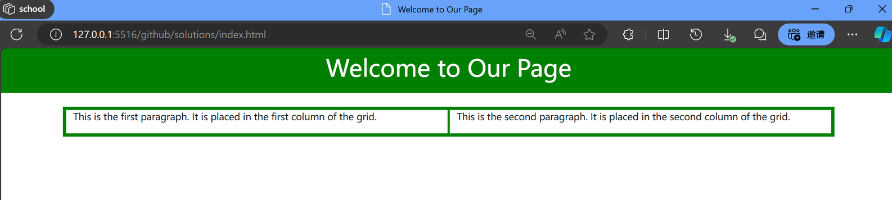
### 

### Commit to remote repo

1. Fetch origin



1. Push to main
   1. It should fetch origin again automatically
2. Open **index.html**, both students should see the updated page



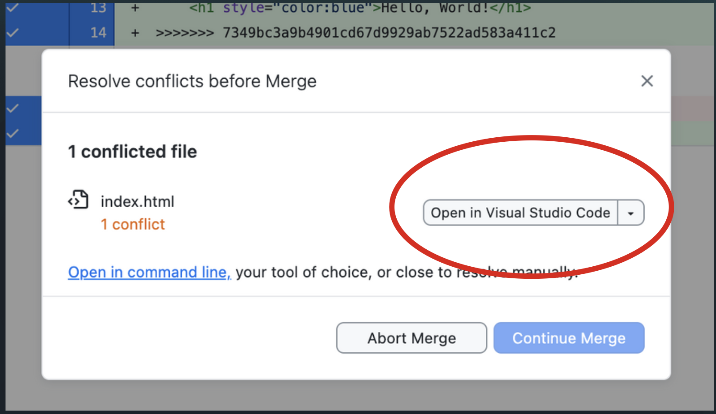
## Merging Conflicts

### Simulate a merge conflict

*Both students will need to play both Student A and B,*

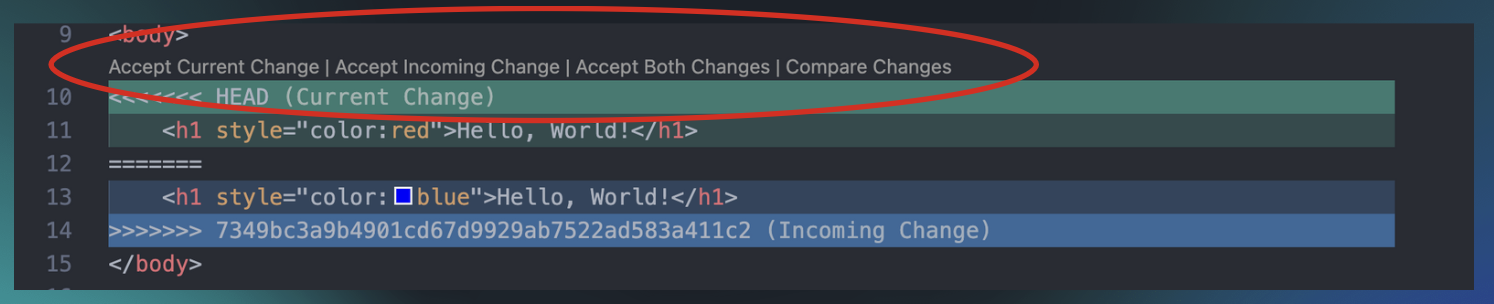
*If no friends, use Github web editor to simulate another person.*

1. Student A edits header to “background-color: red” in **styles.css** and commits the changes
2. Without pulling the latest changes, Student B edits header to “background-color: blue” in **styles.css** and commits the changes.
3. Student A tries pushing/pulling the changes and encounters a merge conflict
4. Open the code in visual studio code



1. Resolve the conflict by accepting the current or incoming changes OR

Type new code and remove the ‘<<<<<<<Head’, ‘==========’ and ‘>>>>>>>>>>>’ merge conflict indicators.



1. Complete the merge and push the changes in Github Desktop

## Important files

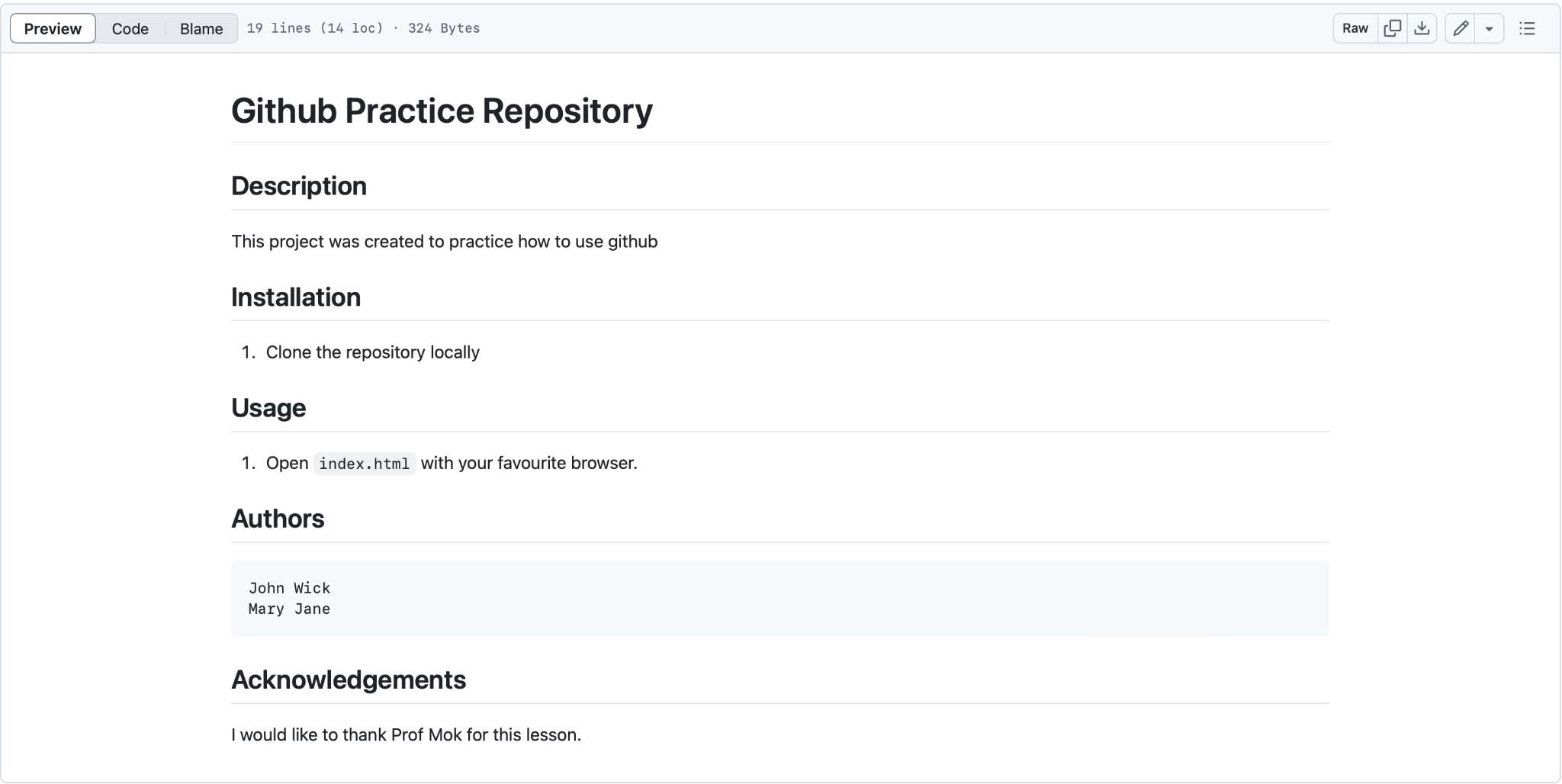
### Create readme file

1) Create a new file named ‘README.md’ in the root folder of repository

2) Add the following sections and add some content.

* Project Overview
* Installation Instructions
* Usage Examples
* Authors
* Acknowledgments

3) Style the content using Markdown language. Try to make the content look like the sample.



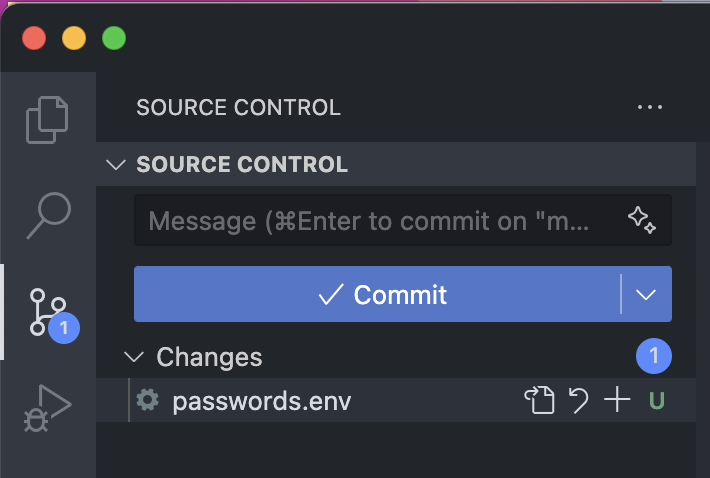
4) Stage, commit and push the changes as ‘create README file’

*Alternatively, use the tools on Github website to create the readme file*

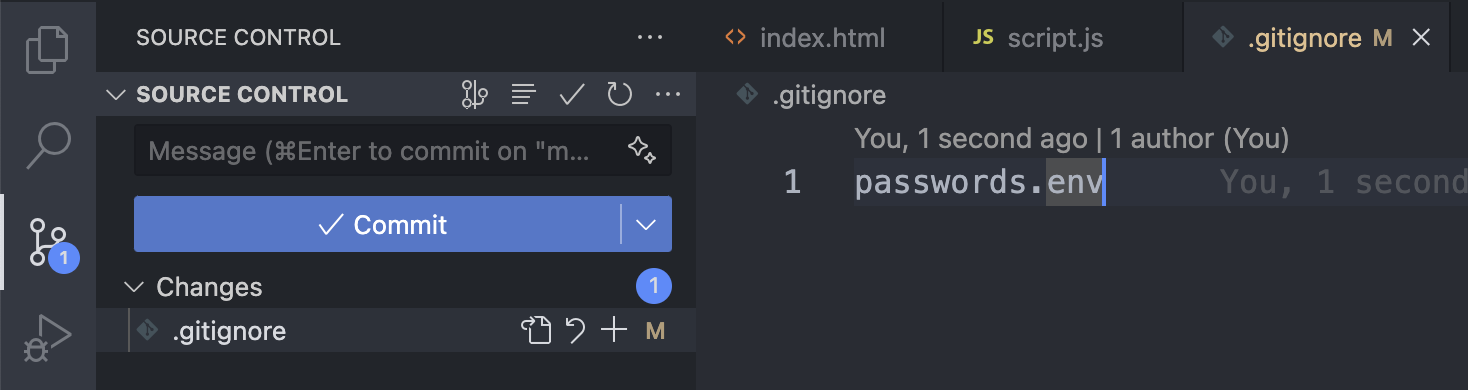
### 

### Create gitignore file

1. Create a new file named **.gitignore** in the root folder of the repository
2. Create a new file called **passwords.env** (Note that the file appears in source control)



1. Add **passwords.env** to .gitignore (Note that file should no longer appear in source control)

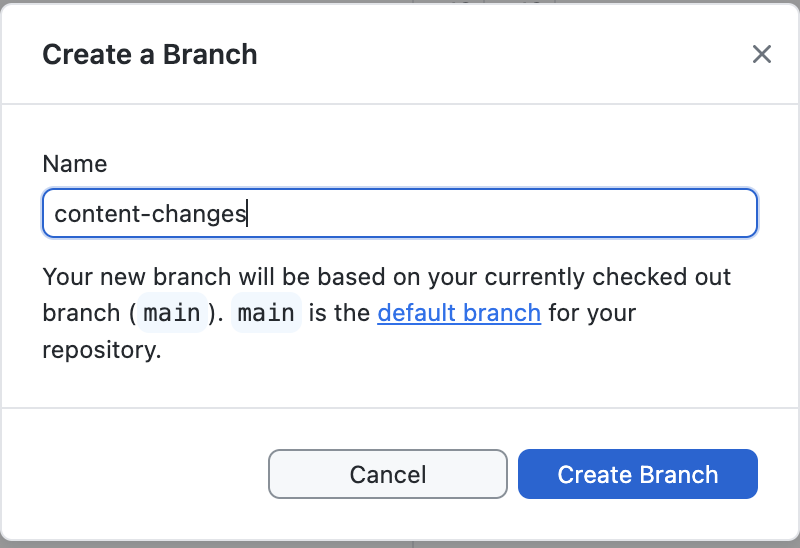


1. Stage, commit and push the changes as ‘create gitignore’

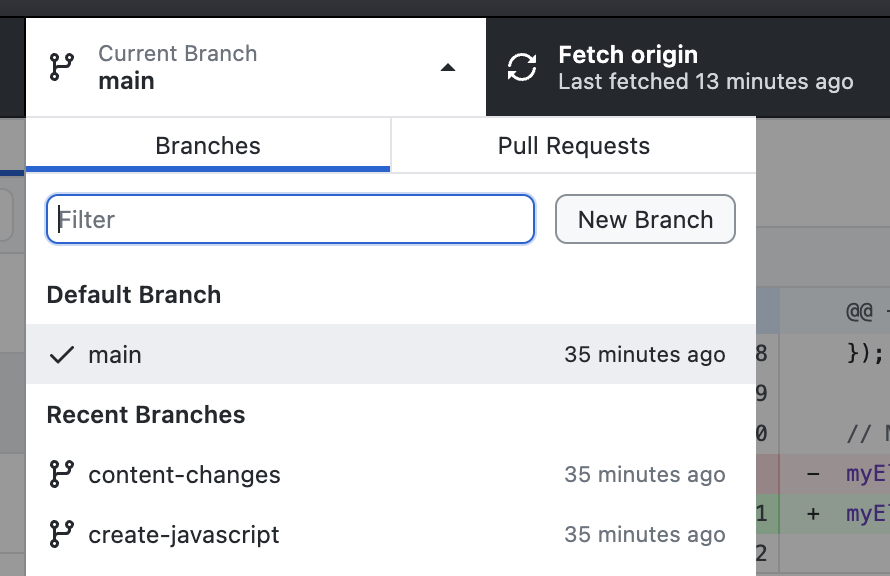
## Branching

### Create Branch

1. Create a new branch by selecting Branch > New Branch. Name the new branch “content-changes”



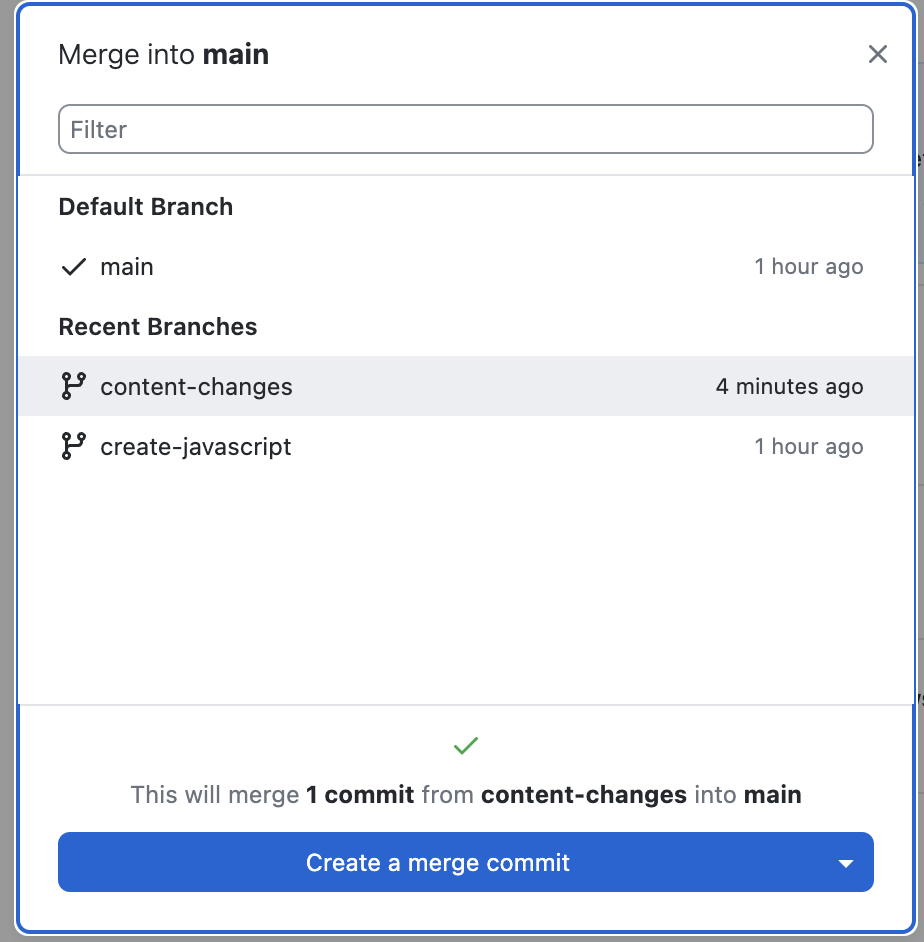
1. Publish the branch to Github
2. Switch over to “content-changes” branch



1. Edit the header to “background-color: purple” in **styles.css** and
2. Stage, commit and push the changes

### Merging Branch

1. Switch over to the main branch (Notice the latest commit is not in the commit history)
2. Merge the “content-changes” branch into the main branch by selecting Branch > Merge into Current Branch. Select “content-changes” branch (Fix conflicts if any)



1. Push the merge commit

## Find out more about

* Pull Request (PR)
* Branching techniques and conventions
* How to revert changes
* How to remove commits from version history
* Using command line
* Automated deployment
* Github analytics

## References

Github Official Documentation:

<https://docs.github.com/en>

Markdown Language:

<https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax>

2024 Ellipsis x .Hack Git Workshop (19 Aug)