

GitHub As Source Control

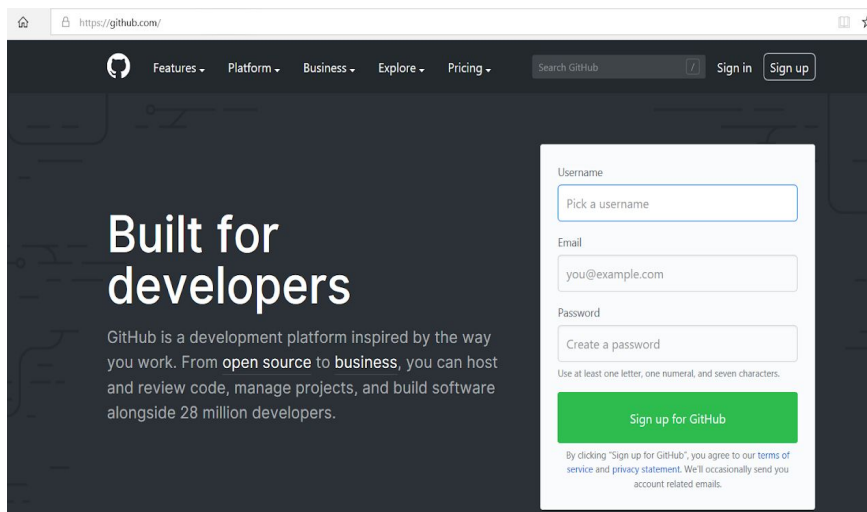
Document with how-to steps: account set up, linking IDE to source control, creating new separate branches, syncing to master branch, merging separate branches to the master branch, restoring old versions, and viewing contribution history of each user.

Great external guide to understand this process

<https://www.atlassian.com/git/tutorials/comparing-workflows/feature-branch-workflow>

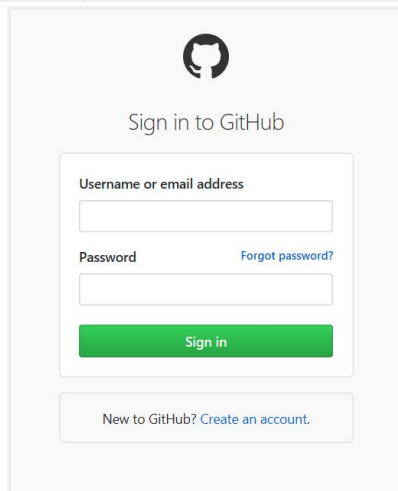
GitHub SetUp

1. Go to github.com and sign up using your gmail account.



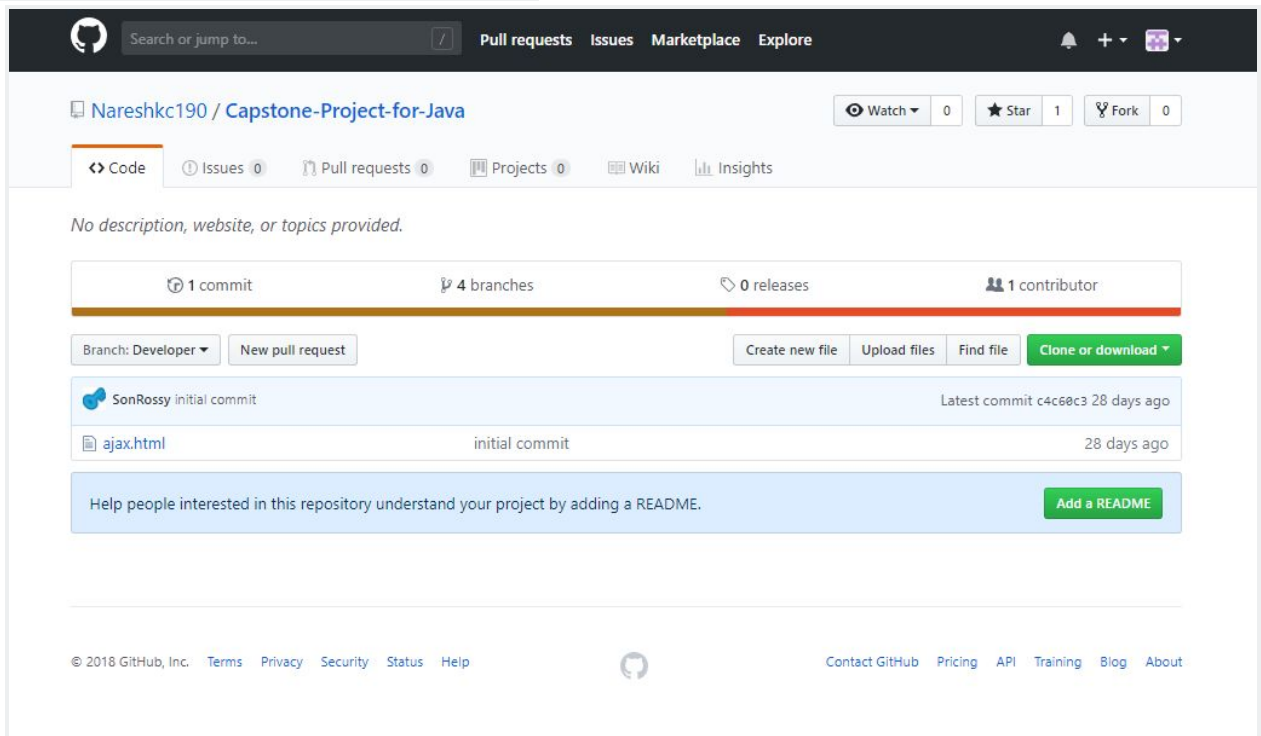
The screenshot shows the GitHub homepage with a dark background. On the left, the text "Built for developers" is prominently displayed, followed by a description of GitHub as a development platform. On the right, there is a white sign-up form with fields for "Username", "Email", and "Password". Below the password field, there is a note: "Use at least one letter, one numeral, and seven characters." A green "Sign up for GitHub" button is at the bottom of the form. Above the button, there is a link to the terms of service and privacy statement.

2. Once you create an account, login in using your email address.

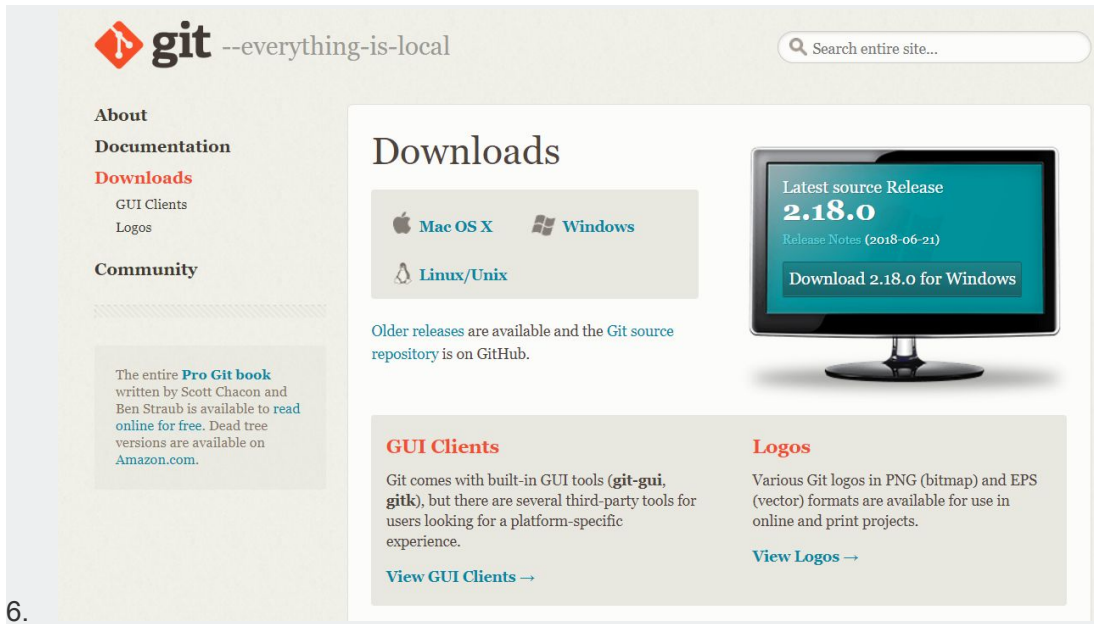


The screenshot shows the GitHub login page. At the top, there is the GitHub logo and the text "Sign in to GitHub". Below this, there is a white login form with two input fields: "Username or email address" and "Password". A link "Forgot password?" is next to the password field. A green "Sign in" button is at the bottom of the form. Below the login form, there is a link "New to GitHub? Create an account."

3. Go to <https://github.com/Nareshkc190/Capstone-Project-for-Java>
4. Make sure the default branch is Developer.



5. Now download git and install at <https://git-scm.com/>. Install with recommended settings. Will not interact with PuTTY. Use command prompt as default.



- 6.
7. Open the Git Bash.
8. Create an empty folder anywhere. You can name it "Git".
9. Go back to Git terminal locate folder type "cd" and the file path to your Git folder (you can copy it with ctrl+c and right-click paste). Also make sure to change any "\" to "/" Example "cd /c/Git".

```
Clarissa@DESKTOP-479G6M8 MINGW64 ~
$ cd/c
bash: cd/c: No such file or directory

Clarissa@DESKTOP-479G6M8 MINGW64 ~
$ cd /c

Clarissa@DESKTOP-479G6M8 MINGW64 /c
$ ls
'$Recycle.Bin'/      eula.1040.txt      hp/                 'Program Files'/      SYSTEM.SAV/
BOOTNXT              eula.1041.txt      inetpub/            'Program Files (x86)'/  Users/
'Documents and Settings'@ eula.1042.txt      Intel/              'ProgramData'/         VC_RED.cab
eula.1028.txt         eula.2052.txt      Logs/               Recovery/              vcredist.bmp
eula.1031.txt         eula.3082.txt      msdia80.dll*        swapfile.sys           Windows/
eula.1033.txt         Git/               pagefile.sys        SWSetup/
eula.1036.txt         hiberfil.sys       PerfLogs/           'System Volume Information'/

Clarissa@DESKTOP-479G6M8 MINGW64 /c
$ cd git/
```

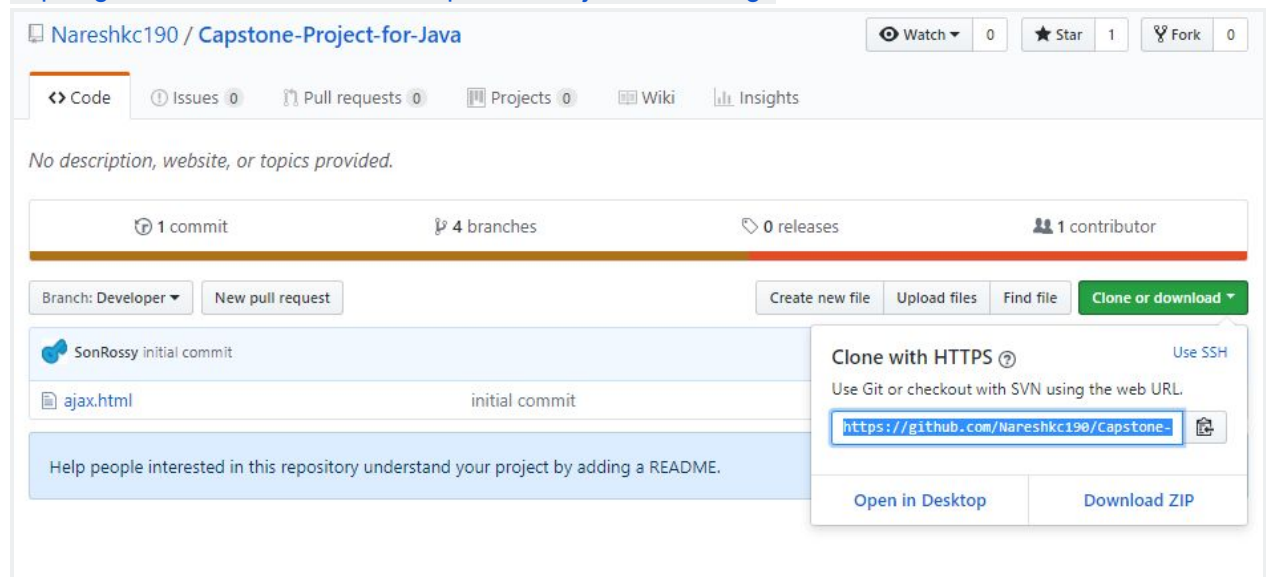
10. Setup github account in terminal using what you used to set up your actual account earlier
"git config --global user.name "Profile name you chose""
11. Now do the next line git config --global user.email "email@example.com"

```
Clarissa@DESKTOP-479G6M8 MINGW64 /c/git
$ git config --global user.name "clarissam5"

Clarissa@DESKTOP-479G6M8 MINGW64 /c/git
$ git config --global user.email clarissam037@gmail.com
```

12. Go back to the main repository setup for the capstone. Copy the HTTPS using the copy button as shown below. Git clone

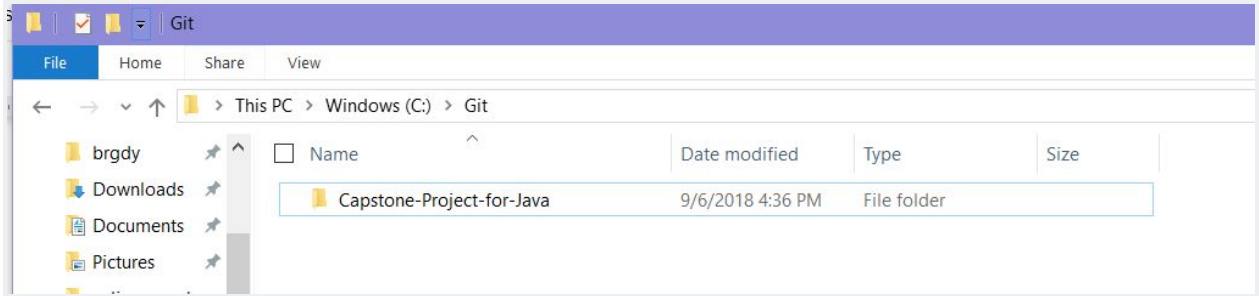
<https://github.com/Nareshkc190/Capstone-Project-for-Java.git>



13. Next write the “git clone” command and paste what you have copied. Then press Enter.

```
Clarissa@DESKTOP-479G6M8 MINGW64 /c/git
$ git clone https://github.com/Nareshkc190/Capstone-Project-for-Java.git
Cloning into 'Capstone-Project-for-Java'...
remote: Counting objects: 35, done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 35 (delta 3), reused 28 (delta 1), pack-reused 0
Unpacking objects: 100% (35/35), done.
```

14. You can physically check the folder you made to see if the repository is in there.



Creating a New Feature Branch

1. Make sure your file path is inside the respiratory folder. Go inside the repository and check the current git branch by typing “git branch.” Check to make sure it says Developer.

```
Clarissa@DESKTOP-479G6M8 MINGW64 /c/git
$ cd Capstone-Project-for-Java/

Clarissa@DESKTOP-479G6M8 MINGW64 /c/git/Capstone-Project-for-Java (Developer)
$ git branch
* Developer
```

2. If you are not on developer branch use git checkout developer

```
Clarissa@DESKTOP-479G6M8 MINGW64 /c/Git/Capstone-Project-for-Java (Developer)
$ git checkout developer
Switched to branch 'developer'
```

3. Create a new branch -- “git branch *branchname*” *branchname* should be the feature you’re working on aka. Card. Make sure the name of the new branch is one word.
4. Type “git checkout *Your branch name*.” This will switch you to the branch you just created.

```
Clarissa@DESKTOP-479G6M8 MINGW64 /c/Git/Capstone-Project-for-Java (Developer)
$ git branch RegistrationCode

Clarissa@DESKTOP-479G6M8 MINGW64 /c/Git/Capstone-Project-for-Java (Developer)
$ git checkout RegistrationCode
Switched to branch 'RegistrationCode'
```

15. To implement the new branch on the repository type “git push origin *branchname*” It will prompt you to sign into your github account in a small window.
16. Go to capstone project in github and verify that your branch exists.

17. Check if the folder is a git repository by typing "git status *branchname*"

```
Clarissa@DESKTOP-479G6M8 MINGW64 /c/Git/Capstone-Project-for-Java (developer)
$ git status Customer Registration form implementation
On branch developer
nothing to commit, working tree clean
```

18. If you made a file and want to add changes from all tracked and untracked files type "git add -A"

```
Clarissa@DESKTOP-479G6M8 MINGW64 /c/Git/Capstone-Project-for-Java (Clarissa)
$ git add -A
```

19. Commit file "git commit -a -m "your comment"

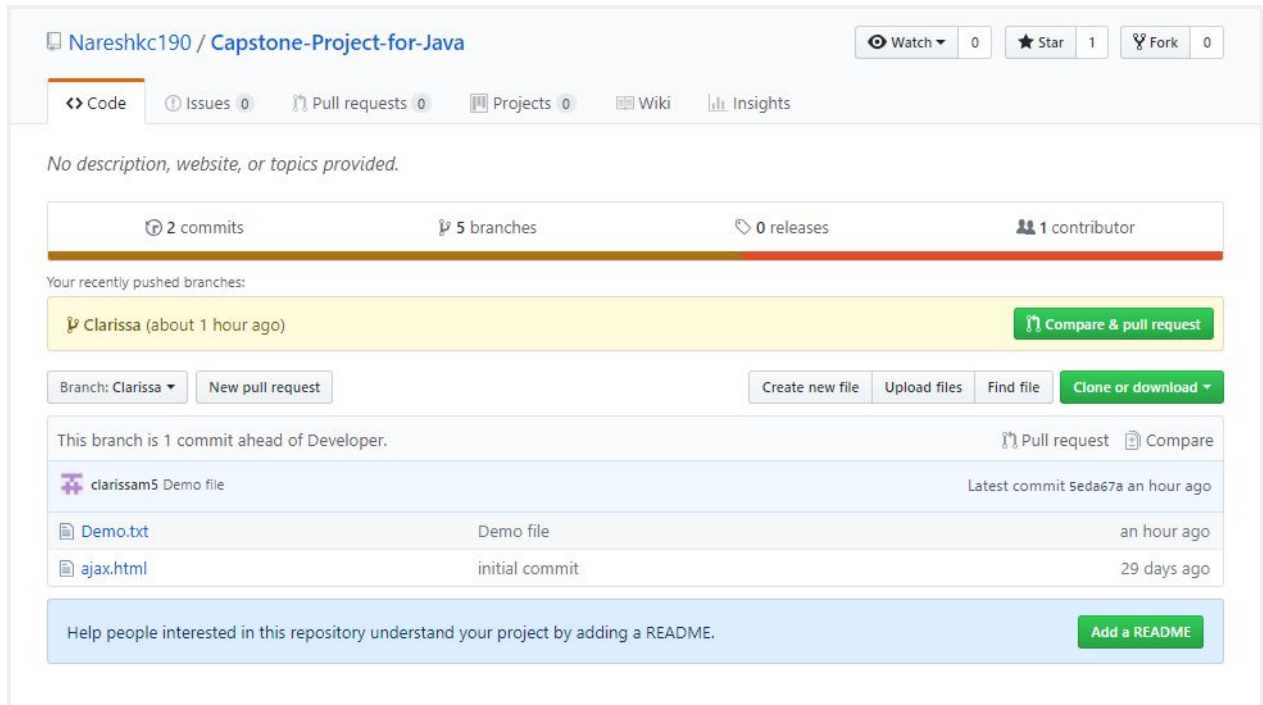
```
Clarissa@DESKTOP-479G6M8 MINGW64 /c/Git/Capstone-Project-for-Java (Clarissa)
$ git commit -a -m "Demo file"
[Clarissa 5eda67a] Demo file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Demo.txt
```

20. Push to your branch "git push origin *branchname*"

```
Clarissa@DESKTOP-479G6M8 MINGW64 /c/Git/Capstone-Project-for-Java (Clarissa)
$ git push origin Clarissa
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 272 bytes | 272.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/Nareshkc190/Capstone-Project-for-Java.git
* [new branch] Clarissa -> Clarissa
```

- When you are trying to push it will ask for your github credential login using username and password
- Once your credentials are verified it will push into github.

21. Go to capstone webpage and verify your branch exists with the contents you pushed .



Merging separate branches to the Developer branch

1. Confirm the receiving branch type "git status" and if needed execute git checkout <receiving> (Developer in our case) to switch to the receiving branch.
2. Make sure the receiving and the merging branch are up to date. Execute git fetch to pull the latest remote commits to. When fetch is done, execute git pull to ensure the latest updates
 - a. Pull requests lets others review your code when you are ready
3. Now execute merge by typing git merge <branch name>. This will merge <branch name> into receiving branch (Which is the branch we switched to using the checkout & should be the Developer branch).

Restoring old versions

A. Head Branch

1. This will rewind your HEAD branch to the specified version. All commits that came after this version are effectively undone; your project is exactly as it was at that point in time.

The reset command comes with a couple of options, one of the more interesting ones being the "--soft" flag. If you use it instead of --hard, Git will keep all the changes in those "undone" commits as local modifications:

```
$ git reset --soft 0ad5a7a6
```

```
$ git reset --hard 0ad5a7a6
```

B. Local Branch

1. As said, using the reset command on your HEAD branch is a quite drastic action: it will remove any commits (on this branch) that came after the

specified revision. If you're sure that this is what you want, everything is fine.

However, there is also a "safer" way in case you'd prefer leaving your current HEAD branch untouched. Since "branches" are so cheap and easy in Git, we can easily create a new branch which starts at that old revision:

Normally, the checkout command is used to just switch branches. However, providing the -b parameter, you can also let it create a new branch (named "old-project-state" in this example). If you don't want it to start at the current HEAD revision, you also need to provide a commit hash - the old project revision we want to restore.

Voilà: you now have a new branch named "old-project-state" reflecting the old version of your project - without touching or even removing any other commits or branches.

\$ git checkout -b old-project-state 0ad5a7a6

Viewing contribution history of each user

Navigate to the Capstone-Project-for-Java on the Github website, click code, and click commits to see contribution history of each user.

The screenshot displays the GitHub repository page for **Nareshkc190 / Capstone-Project-for-Java**. The repository has 0 Watchers, 1 Star, and 2 Forks. The **Code** tab is selected, showing the commit history for the **master** branch.

The commit history is organized by date, with the following entries:

- Commits on Aug 29, 2018**
 - Set theme jekyll-theme-hacker** by Nareshkc190 committed 16 days ago. Commit hash: 3182bb5.
- Commits on Aug 24, 2018**
 - Create README.md** by SonRossy committed 21 days ago. Commit hash: 70b6e7e. This commit is marked as **Verified**.
- Commits on Aug 23, 2018**
 - Merge branch 'master' of https://github.com/Nareshkc190/Capstone-Proj...** by SonRossy committed 22 days ago. Commit hash: 5eb4b67.
 - adding jsp features** by SonRossy committed 22 days ago. Commit hash: e324402.
- Commits on Aug 9, 2018**
 - initial commit** by SonRossy committed on Aug 9. Commit hash: c4c60c3.

