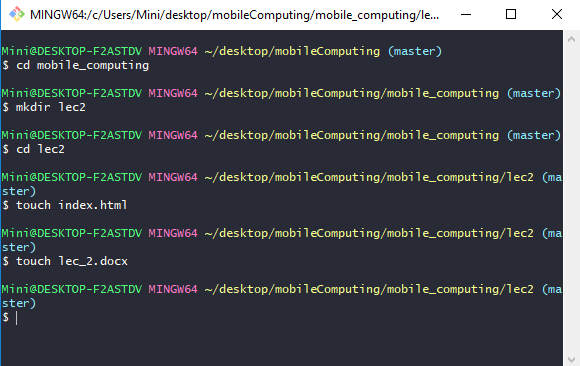
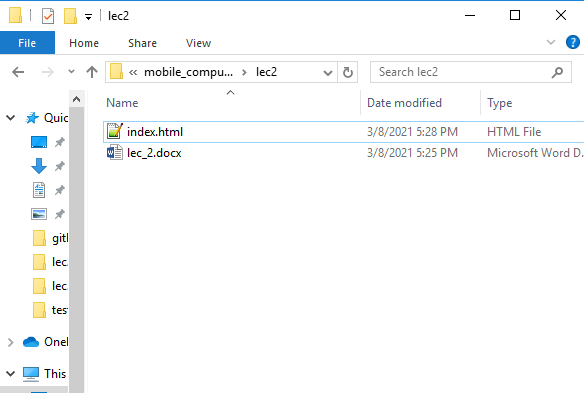
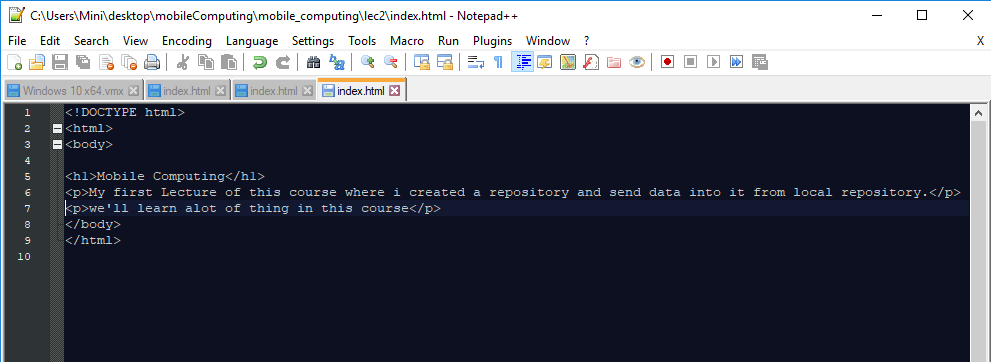
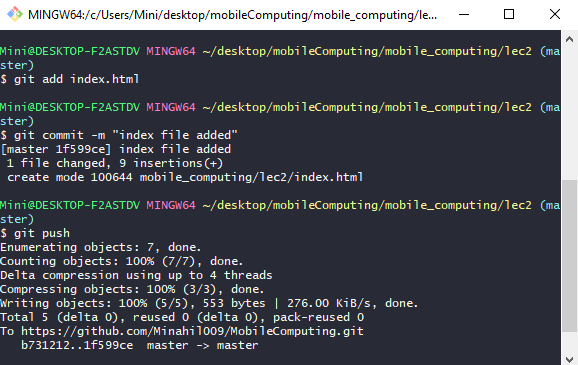
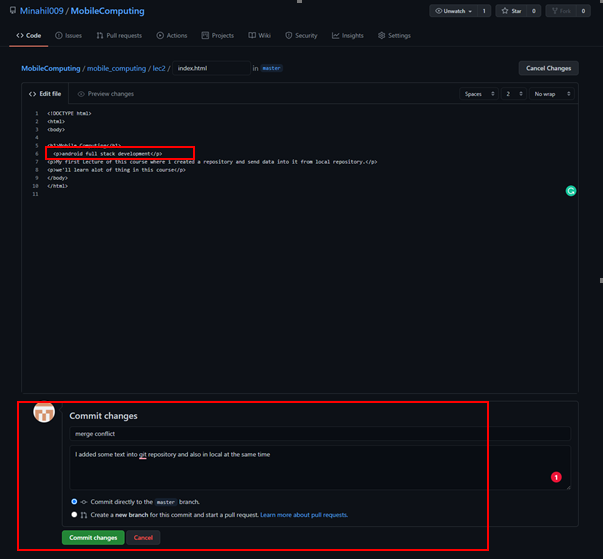
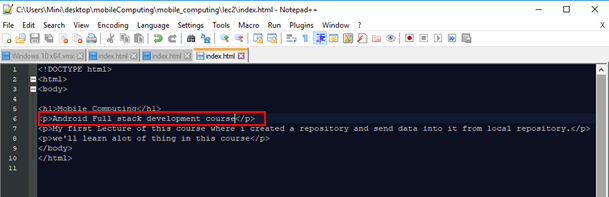
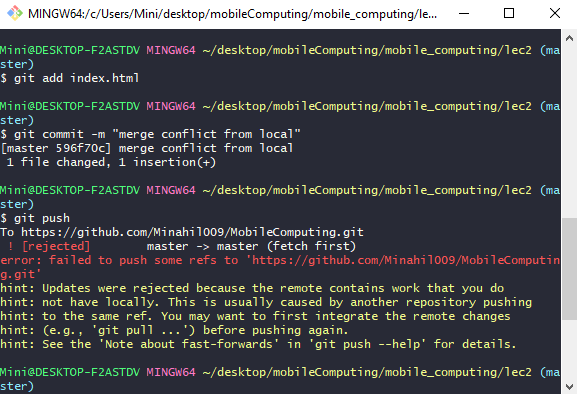
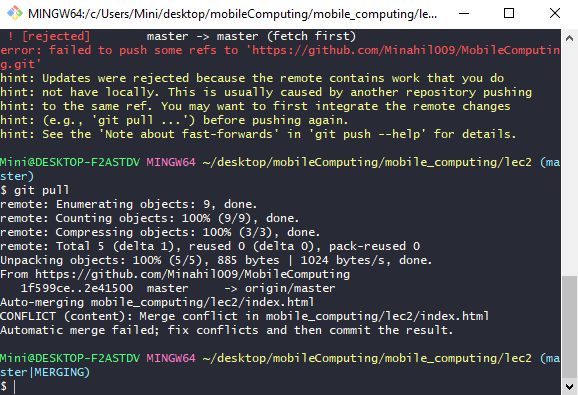
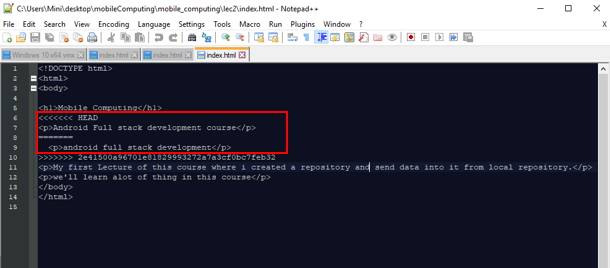
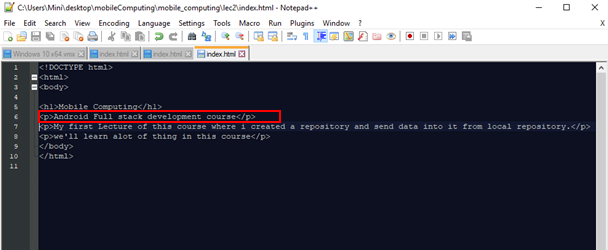
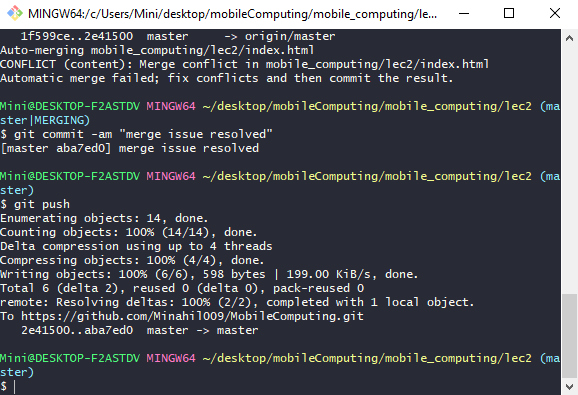
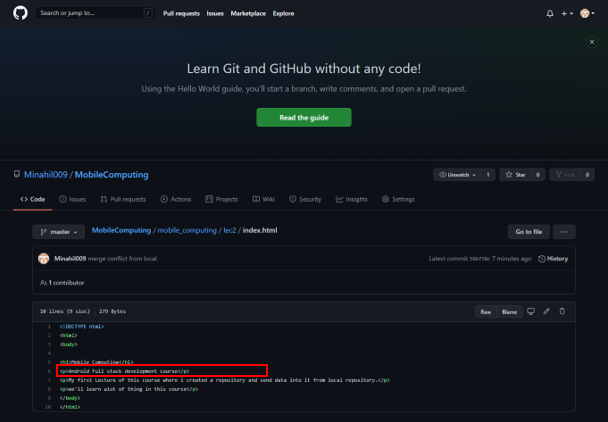
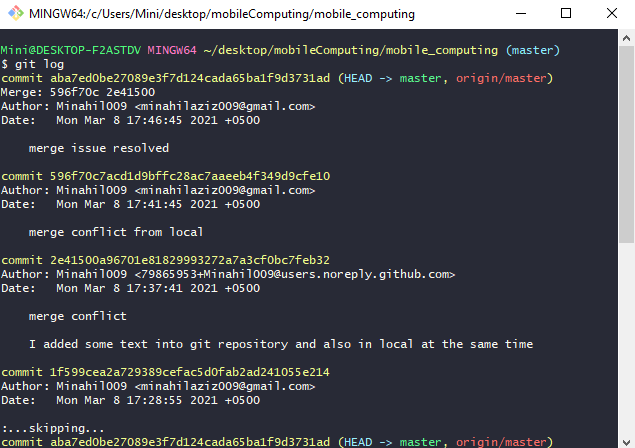
**Mobile Computing**

**Lecture 02:**

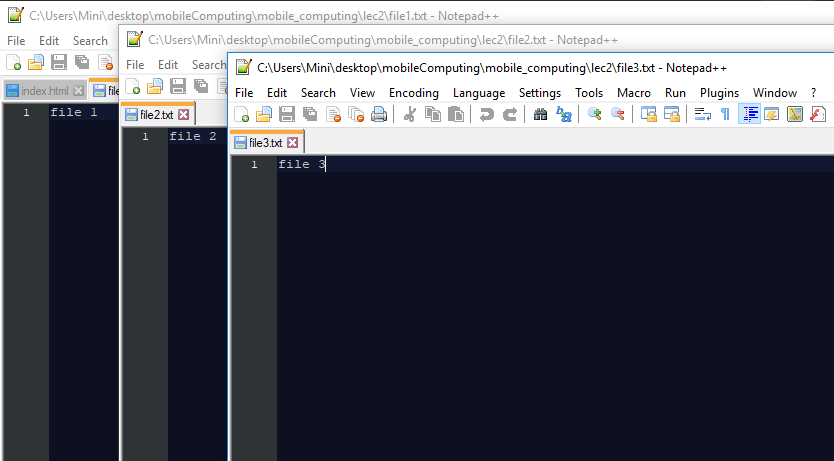
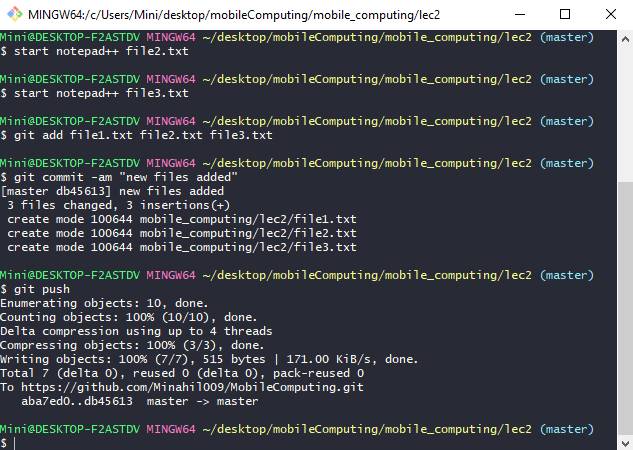
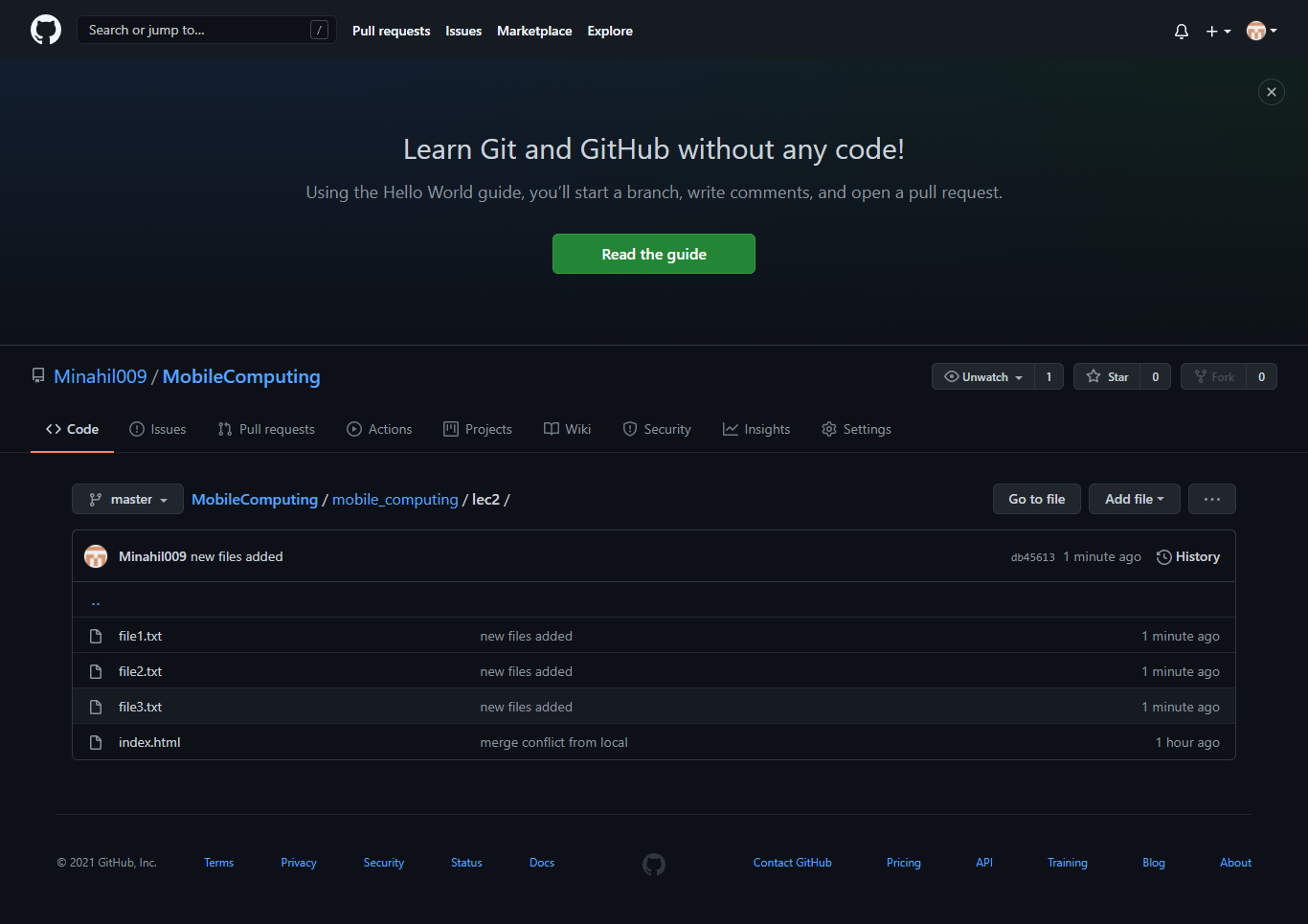
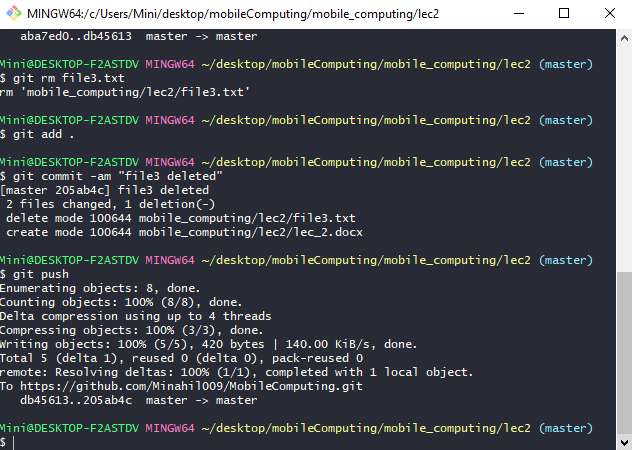
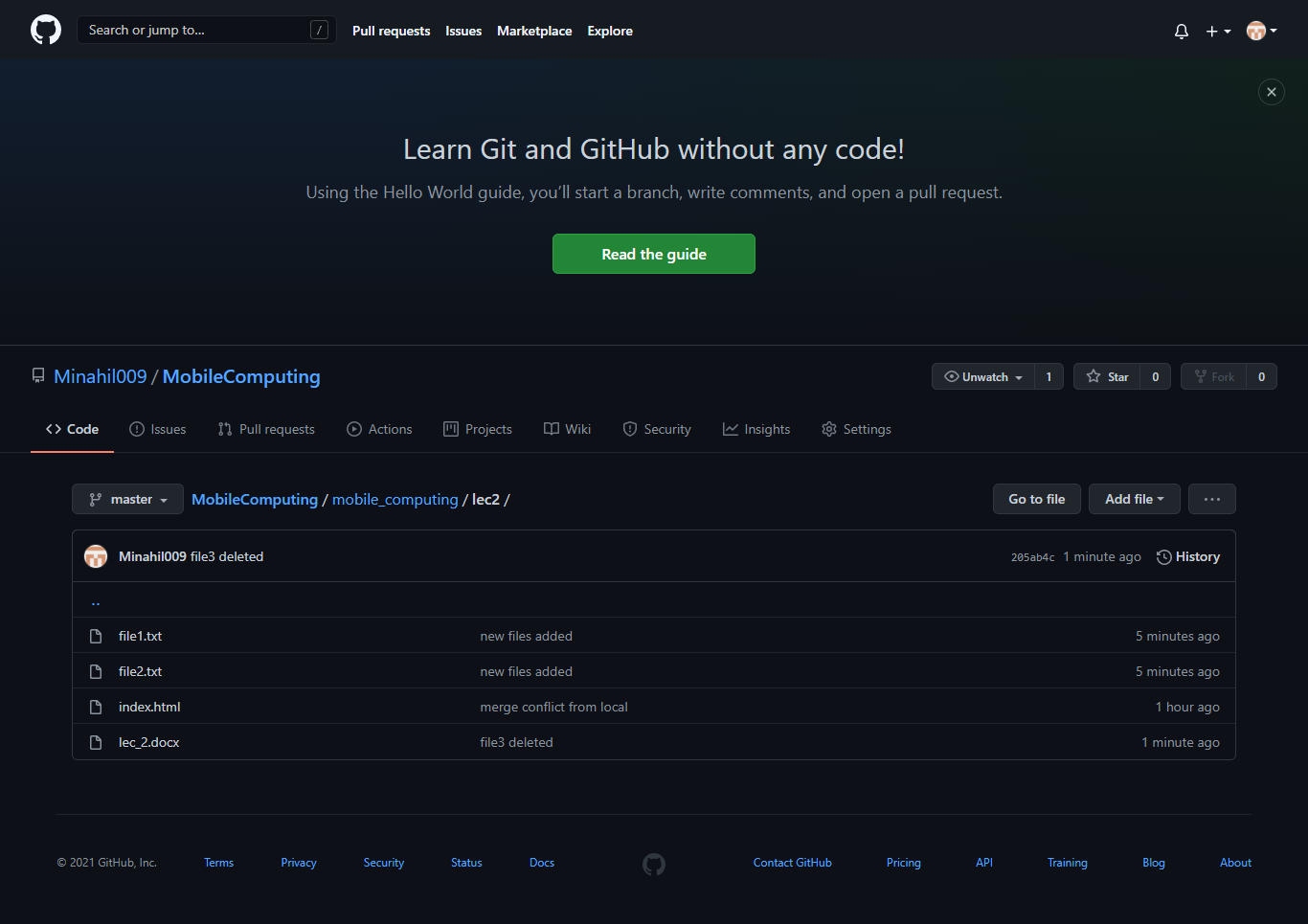
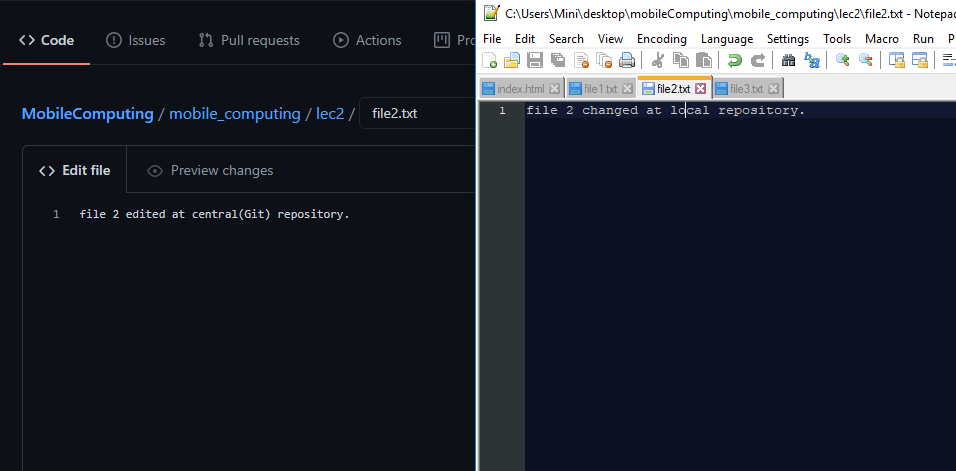
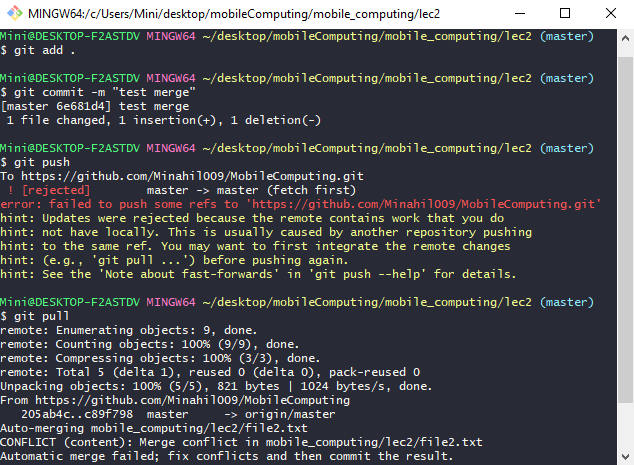
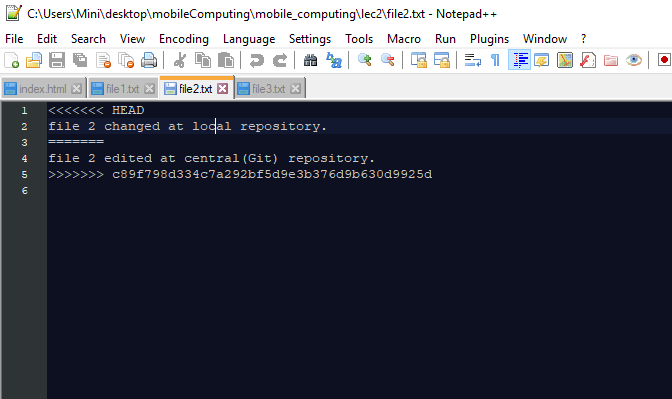
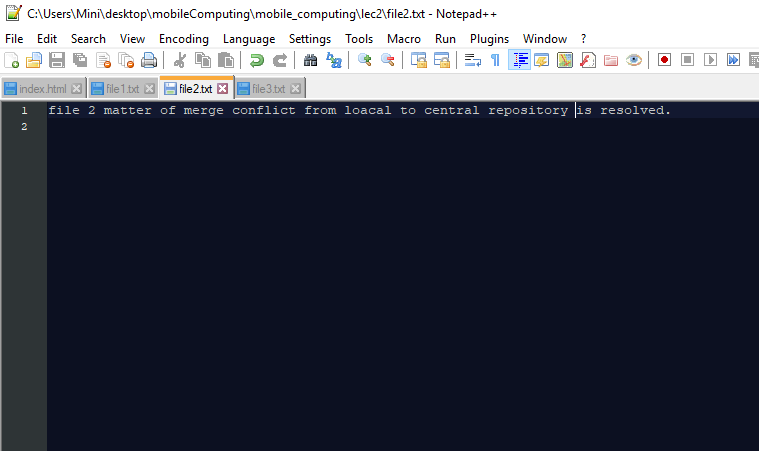
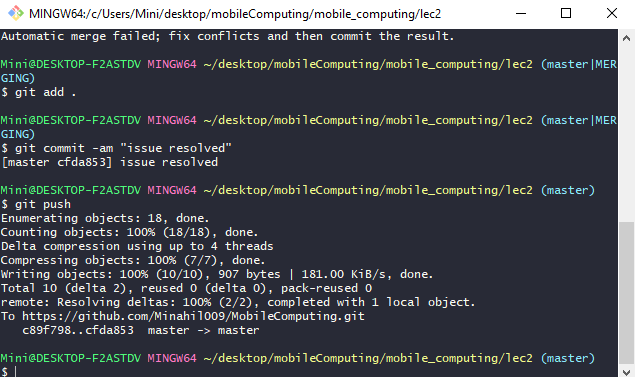
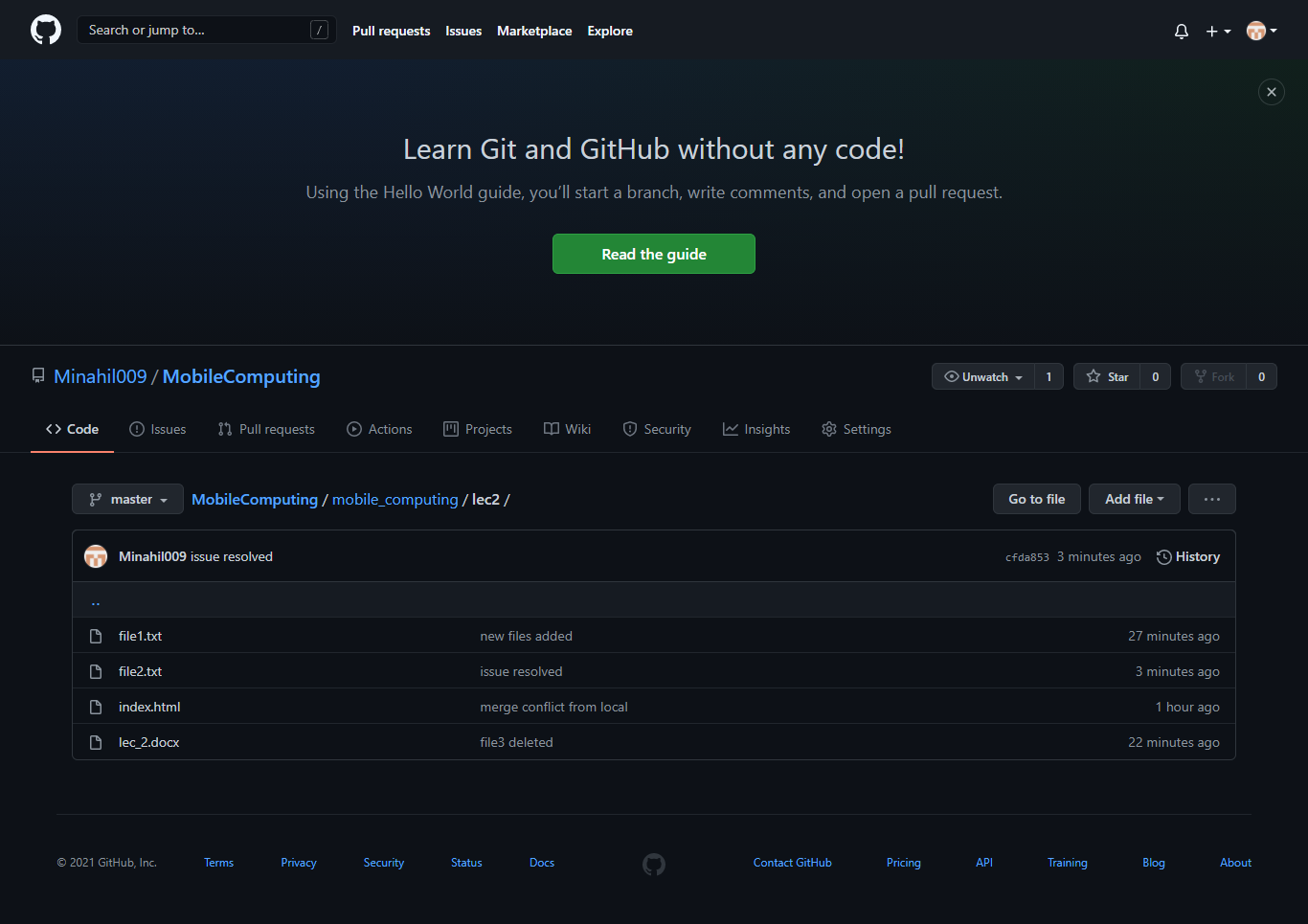
**Teacher Name: Sir Haq Nawaz Student Name: Minahil Aziz**

In lecture one we studied:

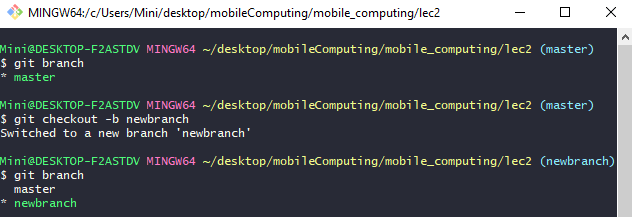
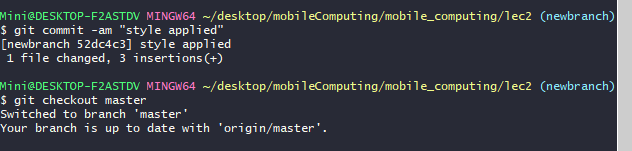
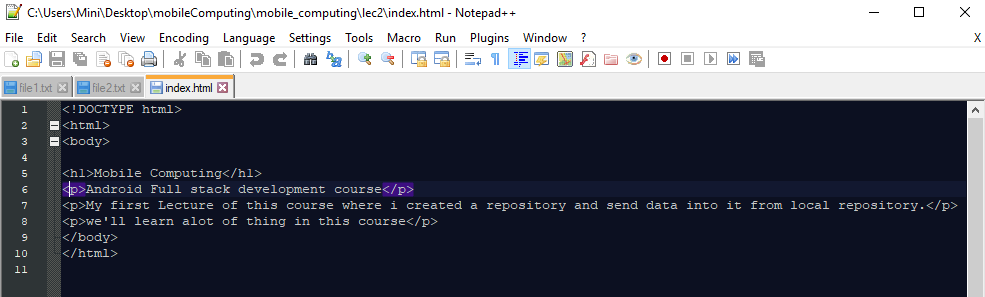
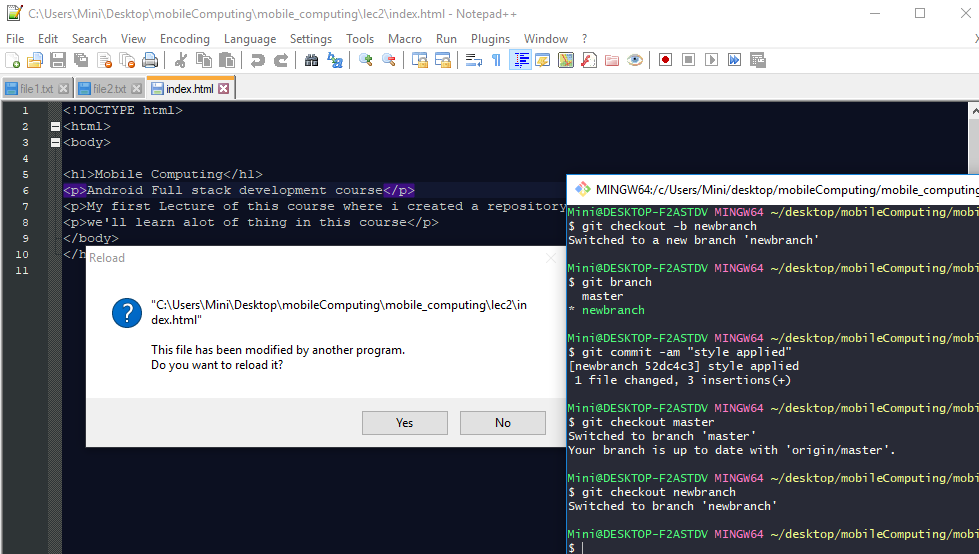
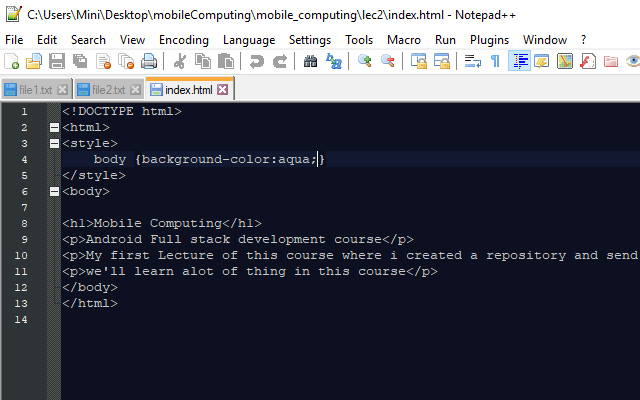
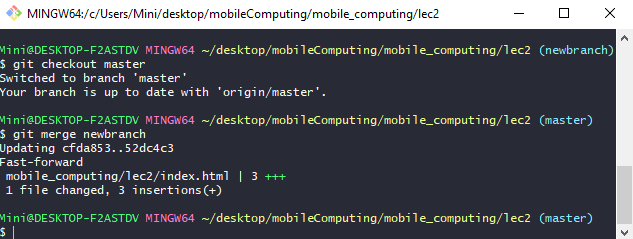
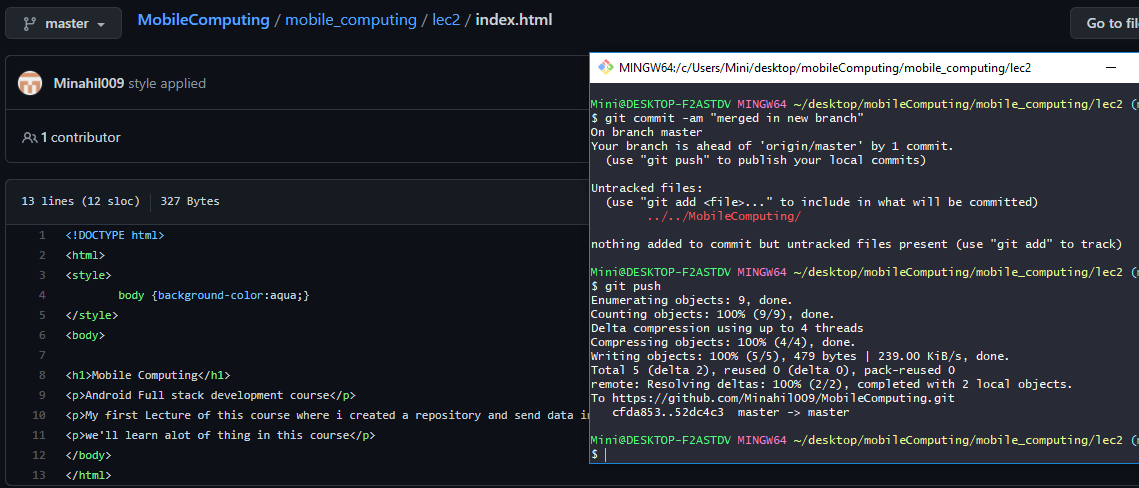
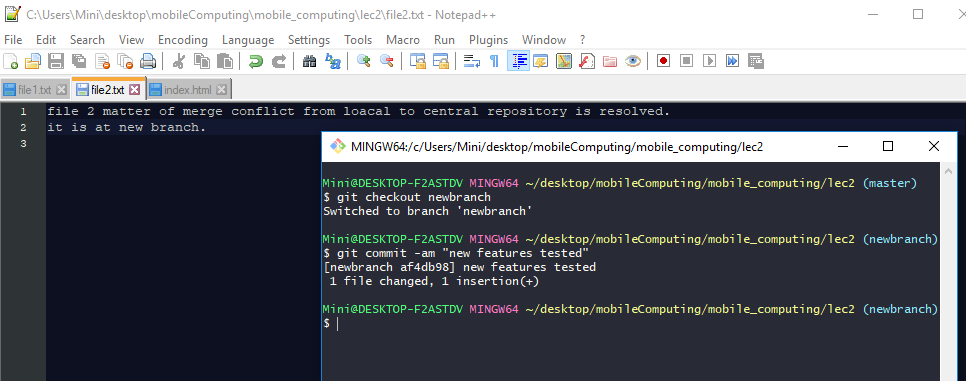
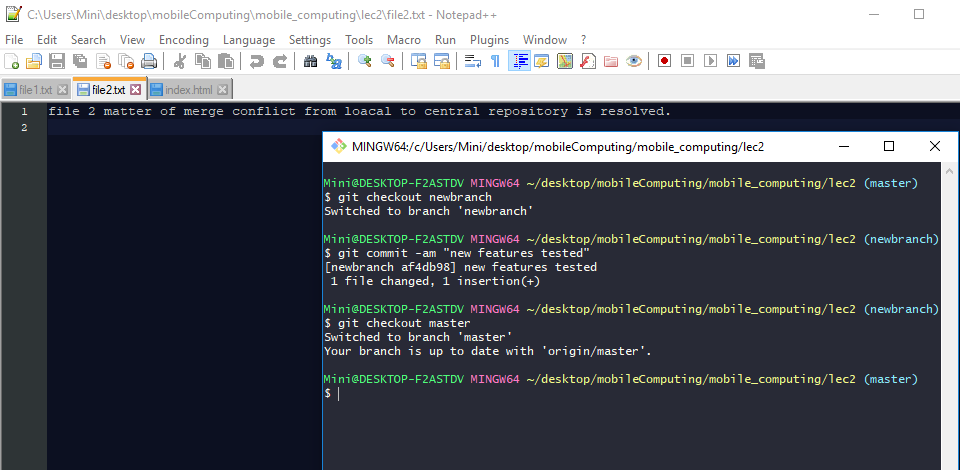
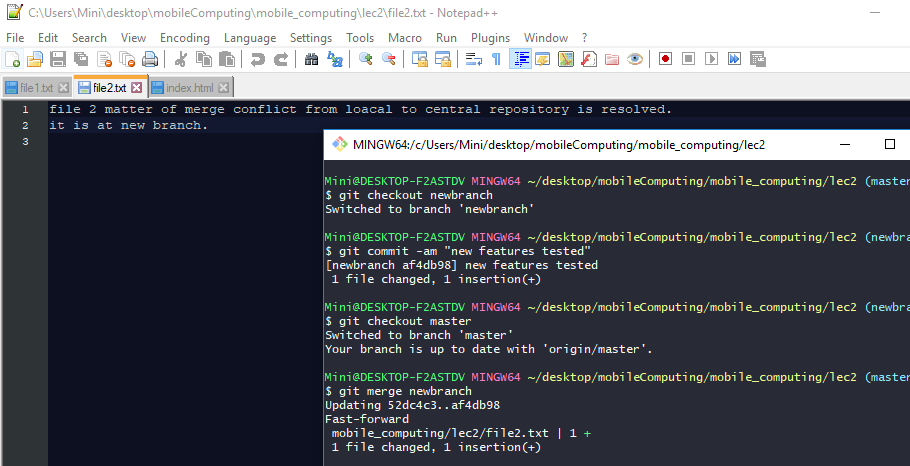
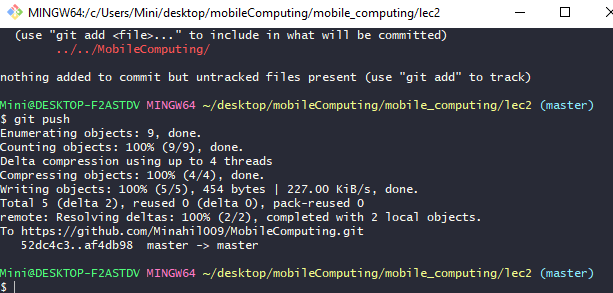
* Merge conflict (local to central repository)
* Create branches
* Merge data of branches
* Delete files
* Delete branches

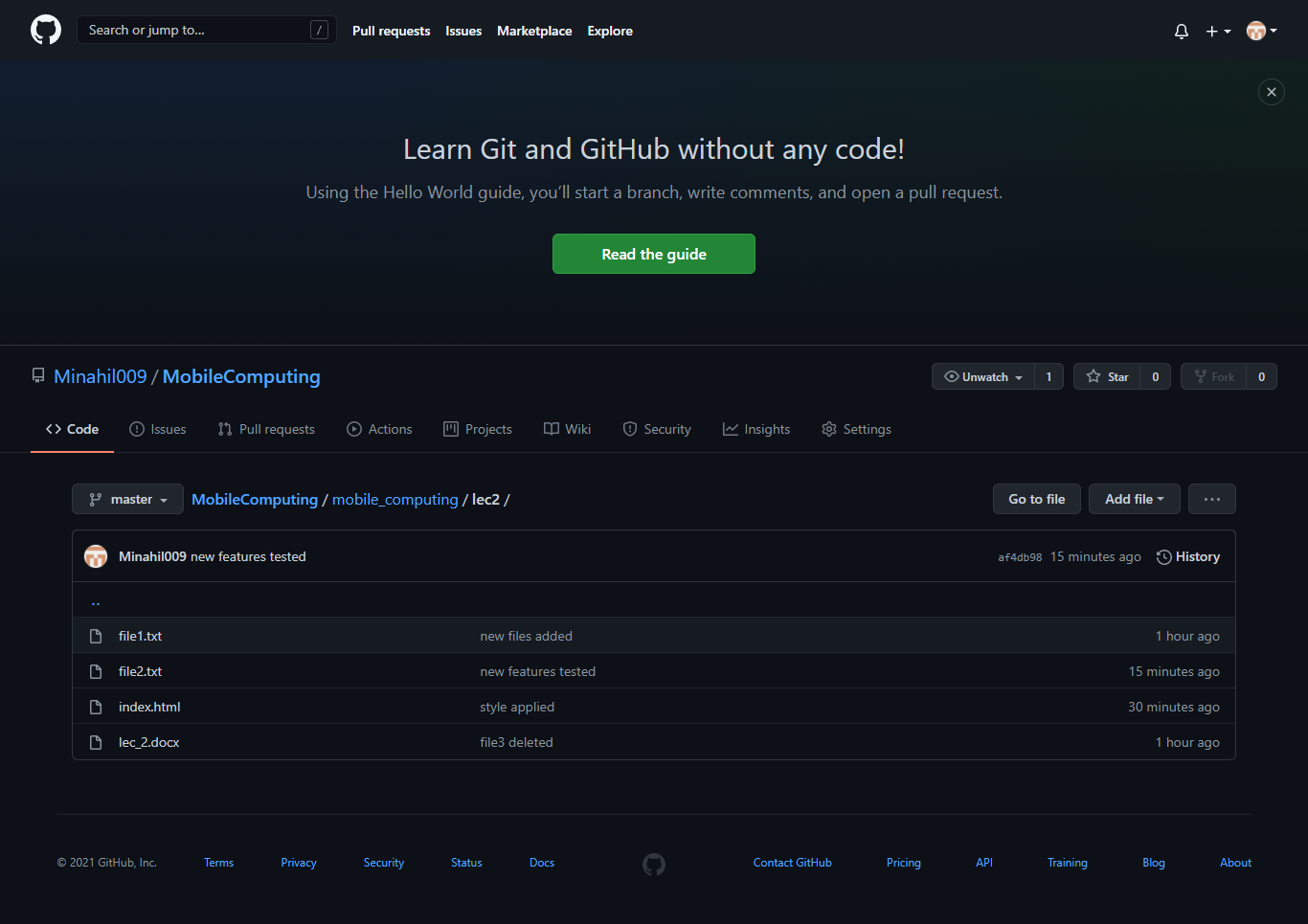
1. Create directory and files. 
2. so as you see, the folder and files are created. 
3. Then open **index.html** file, and **add** some data in it. 
4. After that, **add** this file to git, then run **commit** and then **push** it into central repository. 
5. Now, to **merge** central and local repository, first, **edit** code into central repository and then **commit changes**. 
6. Now, **without pulling** central data, just **go into local** repository and **add some text** to same line at the same time. 
7. Then **add** that file into git and run **commit** command, now, when you **push** it to git, it gives an error to firstly **pull** central repository data into it. 
8. So, run git **pull** command and it gives an error (which tells you that there is a conflict in merging so resolve it first then commit again). 
9. So, **open your file** from local repository, you can see there are **two statements** showing (one of local and 1 of central repository). 
10. Select the data you want to add and del other one, then save. 
11. Now, **commit** again according that error and then git **push**. 
12. so, now, the new changes will be added in the both of the repositories, as you see in below pic. 
13. you can also see the details of the commands and actions you perform using **git log** command. 

**Another example,**

1. Now, **create** 3 files.
2. Then, **open** in notepad++ (or any other software) and data in it. 
3. After that, **add** these files to git, then run **commit** command and **push** into repository. 
4. So, as you see below, all files are added to central repository. 
5. If you want to delete file, use **rm** command, then run **commit** command and **push** into repository. 
6. So, file3 is deleted form git repository. 
7. Now, suppose you want to merge file 2 data, then add changes in both repositories at the same time and save it. 
8. After that, **add** that file into git repository and run **commit** command, now, when you **push** it to git, it gives an error to firstly **pull** central repository data into it. So, run git **pull** command and it gives an error (which tells you that there is a conflict in merging so resolve it first then commit again). 
9. So, **open your file** from local repository, you can see there are **two statements** showing (one of local and 1 of central repository). 
10. Select the data you want to add and del other one, then save. 
11. Now, **commit** again according that error and then git **push**. 
12. so, now, the new changes will be added in the both of the repositories, as you see in below pic. 

**CREATING BRANCHES:**

1. first of all, check your **branch** status, which is master (by default), then **create a new branch** **and switch** terminal to that newbranch using **git checkout –b newbranch** command. Then again check your branch **status,** so, it will switch to newbranch.
2. Run **commit** command, then switch to master branch using **git checkout master** command. 
3. Then open your file, you can see that the code will be changed to that one, which you done in master branch. 
4. Again switch to new branch. 
5. You will again get that code, which you did after switching in new branch. 
6. Now, to merge both branches, **switch** the branch first (in which you want to merge the data), and then merge data using **git merge [branch\_name]** command. 
7. After that, **push** it into repository and check the git repository (it is also updated). 
8. Another example, do the same with this file.(switch to new branch then add changes into file) 
9. Again **switch** to master branch, data will be changed to that one, which you did in master file. 
10. Then **switch** to master branch and **merge** data into it. 
11. Now, there is no more need of new branch. So, delete it (if you want) using **git branch –d newbranch** command. 
12. Then **git push.** 
13. You can see, everything is updated on git repository also.



**So, that’s all from lecture 2.**

**Stay happy ;)**