

### Lab Exercise – GitHub

IT3040 – IT Project Management

Semester 1

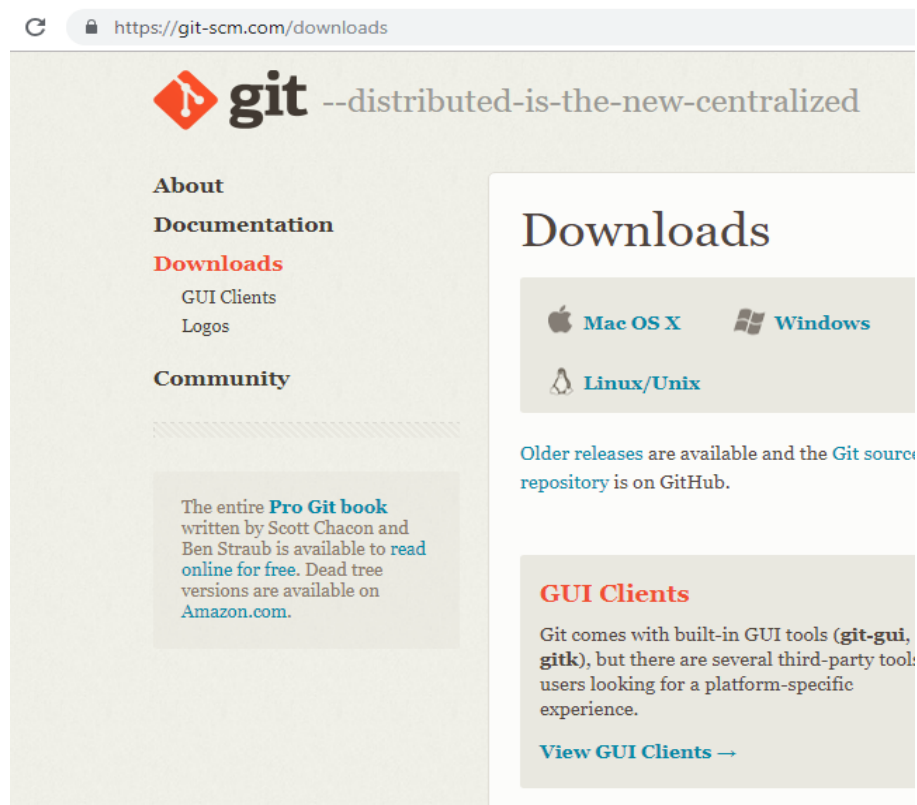
GitHub is a code hosting platform for version control and collaboration. It provides an environment for many users to work together as a group.

#### Objective:

To get familiarize with Git repositories, branches, commits, push, and pull requests.

#### 1. Git Bash – Desktop Repository

##### 1.1. Download Git for windows



##### 1.2. Check the version to make sure it installed properly

```
dilani.l@c-08-AC-1452 MINGW64 ~ (master)
$ git version
git version 2.20.1.windows.1

dilani.l@c-08-AC-1452 MINGW64 ~ (master)
$ |
```

### Lab Exercise – GitHub

IT3040 – IT Project Management

Semester 1

#### 2. Creating a local repository using Git Bash

- 2.1. Create a folder named MyApp-1
- 2.2. Right click and select “Open Git Bash Here”
- 2.3. Add index.html
- 2.4. Open index.html in an editor and add some basic html tags
- 2.5. Initialize this folder as the Git repository. It will create a hidden .git folder
- 2.6. Add your name and email to Git
- 2.7. Add index.html file to Git
- 2.8. Commit the changes to Git

*To add any number of html files* `git add *.html`

2.3	<pre>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (master) \$ touch index.html</pre>
2.5	<pre>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (master) \$ git init Initialized empty Git repository in C:/Users/dilani.l/Desktop/MyApp-1/.git/</pre>
2.6	<pre>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (master) \$ git config --global user.name 'dilani Lunugalage'  dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (master) \$ git config --global user.email 'dilani.l@slit.lk'</pre>
2.7	<pre>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (master) \$ git add index.html</pre>
2.8	<pre>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (master) \$ git commit -m "edited index file" [master f7f0058] edited index file 1 file changed, 1 insertion(+)</pre>

### Lab Exercise – GitHub

**IT3040 – IT Project Management**

**Semester 1**

#### 3. Creating a Branch

- 3.1. Create a branch name “log-in”
- 3.2. Swap from “master” to “log-in” branch
- 3.3. Add login.html file to “log-in” branch
- 3.4. Commit to the repository
- 3.5. Swap to “master”. Note that login.html is not available in master branch.
- 3.6. Merge “log-in” with “master” using the following command: `$ git merge log-in`

Now the login.html file is in master branch as well.

3.1	<code>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (master)</code> <code>\$ git branch log-in</code>
3.2	<code>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (master)</code> <code>\$ git checkout log-in</code> Switched to branch 'log-in'
3.3	<code>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (log-in)</code> <code>\$ touch login.html</code> <code>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (log-in)</code> <code>\$ git add .</code>
3.4	<code>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (log-in)</code> <code>\$ git commit -m "change1"</code> [log-in d6c8b30] change1 1 file changed, 0 insertions(+), 0 deletions(-) create mode 100644 login.html
3.5	<code>dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyApp-1 (log-in)</code> <code>\$ git checkout master</code> Switched to branch 'master'

### Lab Exercise – GitHub

IT3040 – IT Project Management

Semester 1

#### 4. Remote Repository

- 4.1. Click on avatar or identicon (top right corner) and select new repository
- 4.2. Name your repository as “MyFirstApp”
- 4.3. Write a short description


## Create a new repository

A repository contains all the files for your project, including the revision history.

---

Owner

Repository name

 Dilani-Lunugalage ▾


 / 

MyFirstApp ✓


Great repository names are short and memorable. Need inspiration? How about [bookish-fiesta](#).

Description (optional)

---

☒  Public

Anyone can see this repository. You choose who can commit.

☐  Private


You choose who can see and commit to this repository.

---

☐ Initialize this repository with a README

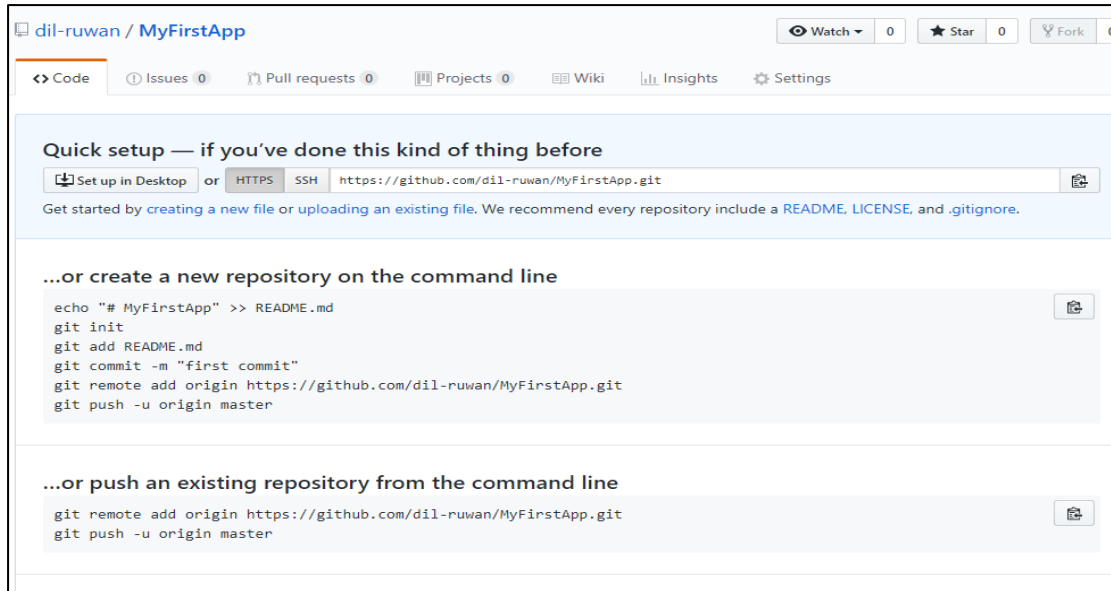
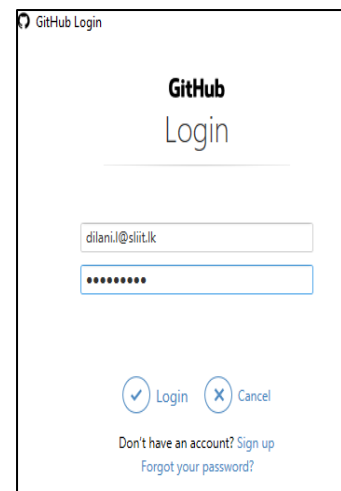
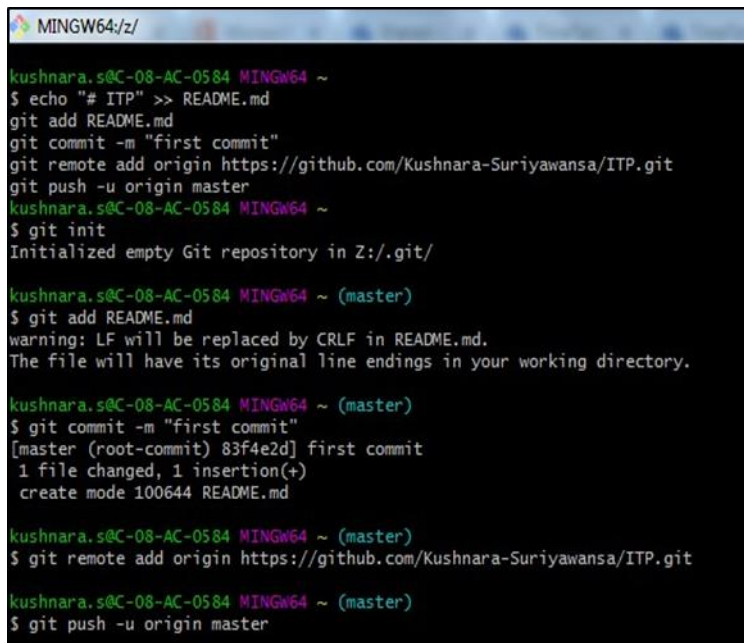
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None ▾

Add a license: None ▾ 

---

Create repository

**Lab Exercise – GitHub****IT3040 – IT Project Management****Semester 1****4.4. Copy the commands using the highlighted button****4.5. Open Git Bash.****4.6. Execute the commands in Git Bash****4.7. Provide username and password used in GitHub account.****4.8. Refresh the GitHub in browser.**

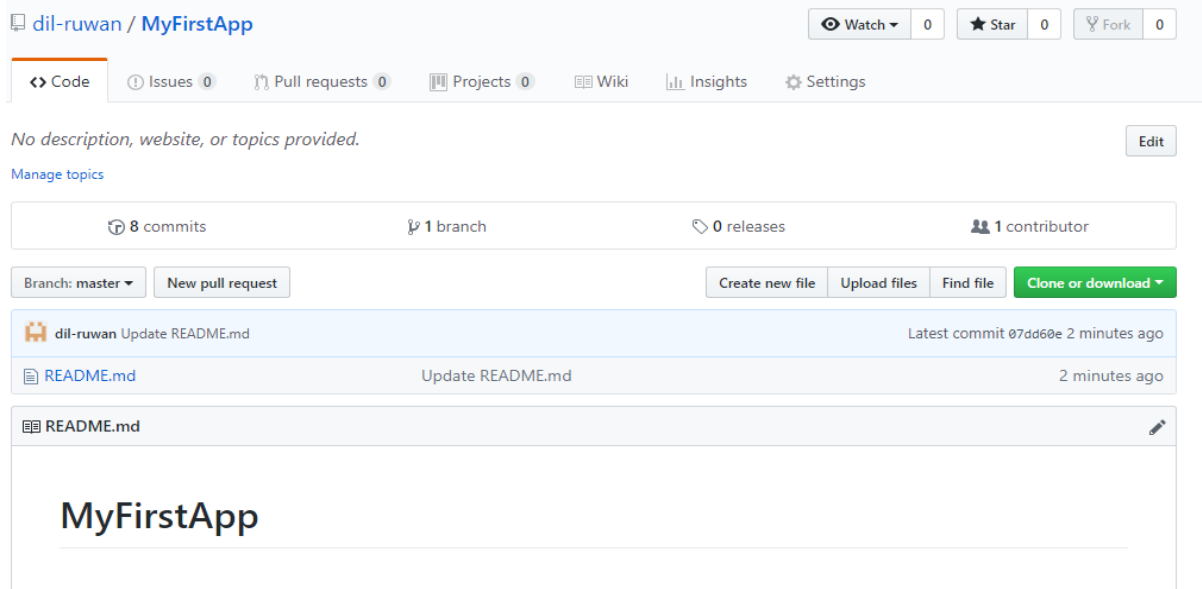
***Note that the committed README.md file is in the repository.***

### Lab Exercise – GitHub

IT3040 – IT Project Management

Semester 1

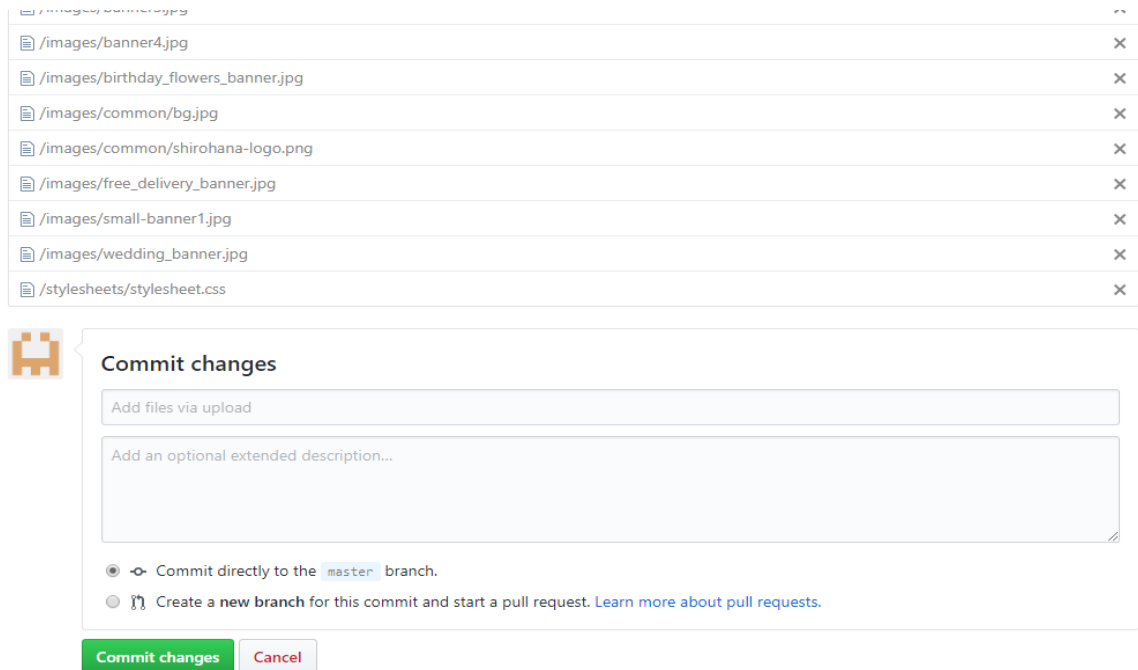
#### 5. Upload projects/files to the repository



The screenshot shows the GitHub repository page for 'MyFirstApp' by user 'dil-ruwan'. At the top, there are buttons for Watch (0), Star (0), and Fork (0). Below this is a navigation bar with links for Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings. A message states 'No description, website, or topics provided.' with an 'Edit' button. Below this, it shows '8 commits', '1 branch', '0 releases', and '1 contributor'. There are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. A commit history table shows a recent commit 'Update README.md' by 'dil-ruwan' 2 minutes ago. Below the table, the content of the README.md file is displayed, showing the text 'MyFirstApp'.

5.1. Select upload files option and add files/folders of the given sample project.

5.2. Commit changes to the repository.



The screenshot shows the 'Commit changes' dialog box in GitHub. It lists several files to be committed: /images/banner4.jpg, /images/birthday\_flowers\_banner.jpg, /images/common/bg.jpg, /images/common/shirohana-logo.png, /images/free\_delivery\_banner.jpg, /images/small-banner1.jpg, /images/wedding\_banner.jpg, and /stylesheets/styleSheet.css. Below the file list, there is a section for 'Commit changes' with a text input for 'Add files via upload' and a larger text area for 'Add an optional extended description...'. At the bottom, there are two radio button options: 'Commit directly to the master branch.' (selected) and 'Create a new branch for this commit and start a pull request. Learn more about pull requests.' Below these options are 'Commit changes' and 'Cancel' buttons.

**Note that the committed files/folders will now be in your repository.**

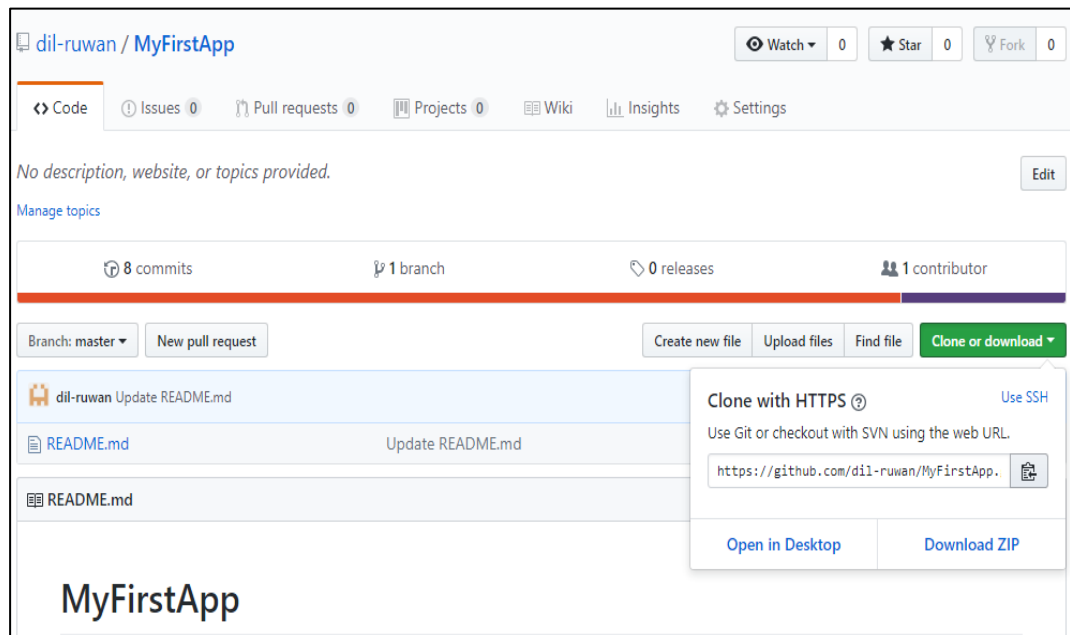
### Lab Exercise – GitHub

#### IT3040 – IT Project Management

#### Semester 1

#### 6. Clone the uploaded project to your machine

##### 6.1. Select **clone** or **download** option and copy the URL



##### 6.2. Open GitBash from desktop (right click and execute 'GitBash here')

##### 6.3. Execute **git clone <URL>** command in Git Bash

Example: **git clone** <https://github.com/dil-ruwan/MyFirstApp>

```
dilani.1@C-08-AC-1452 MINGW64 ~/Desktop (master)
$ git clone https://github.com/dil-ruwan/MyFirstApp.git
Cloning into 'MyFirstApp'...
remote: Enumerating objects: 50, done.
remote: Counting objects: 100% (50/50), done.
remote: Compressing objects: 100% (34/34), done.
remote: Total 50 (delta 3), reused 21 (delta 1), pack-reused 0
Unpacking objects: 100% (50/50), done.

dilani.1@C-08-AC-1452 MINGW64 ~/Desktop (master)
$ |
```

Now there will be a folder from the name of the repository on your desktop. This folder will have all files of the project which are uploaded to the GitHub repository. By default, this project is initialized as a Git project and you will be able to see the hidden .git folder. Thus, now you are able to execute Git commands using the Git bash for this cloned project.

### Lab Exercise – GitHub

IT3040 – IT Project Management

Semester 1

#### 7. Executing Git commands.

*Note: To execute the Git commands to the project, open Git Bash inside the project folder. (Go inside the folder-> right click -> Git Bash here)*

- 7.1. Gives the changes (not yet saved/added) that are made. **There is no change.**
- 7.2. Edit and save index.html file in the root project folder. Run the same command and see the output.
- 7.3. Add/save changes that were made to the given file.
- 7.4. Record the changes as a commit to local repository. Commit can be thought of as a version of the project.
- 7.5. Send changes to the master repository.

7.1

```
dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
$ git diff

dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
$
```

7.2

```
dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
$ git diff

dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
$ git diff
diff --git a/index.html b/index.html
index 8e02e02..dabe924 100644
--- a/index.html
+++ b/index.html
@@ -2,7 +2,7 @@
 <html>

 <head>
-  <title>Shirohana Flowers</title>
+  <title>Welcome to Shirohana Flowers</title>AM
 </head>

 <body>

dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
$
```

7.3

```
dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
$ git add index.html
```

7.4

```
dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
$ git commit -m "index file changed"
[master 760f7c3] index file changed
1 file changed, 1 insertion(+), 1 deletion(-)

dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
```



### Lab Exercise – GitHub

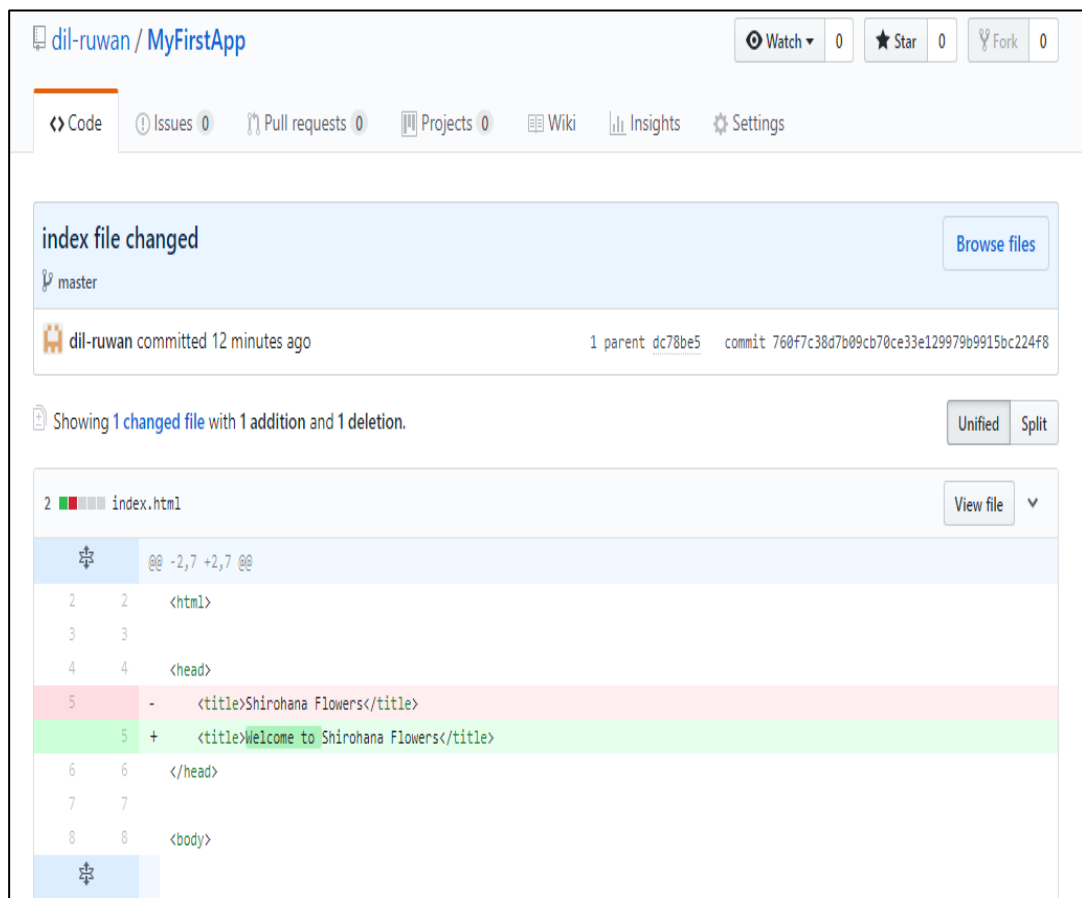
IT3040 – IT Project Management

Semester 1

7.5

```
dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 305 bytes | 305.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/dil-ruwan/MyFirstApp.git
   dc78be5..760f7c3  master -> master
```

**Note the commit request in the master project repository of GitHub.**



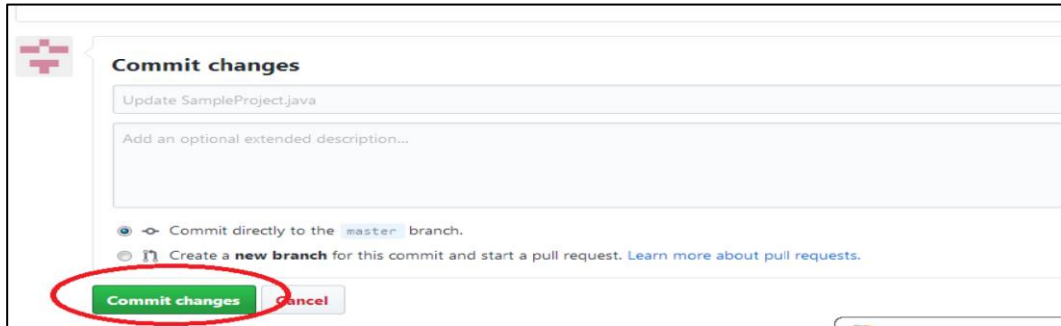
**Note that the index.html file of the master project will now have the change you have made in local project.**

### Lab Exercise – GitHub

IT3040 – IT Project Management

Semester 1

7.6. Edit the SampleProject.java file in GitHub and commit the change.



7.7. Execute **Git pull** command in Git Bash

```
dilani.l@C-08-AC-1452 MINGW64 ~/Desktop/MyFirstApp (master)
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/dil-ruwan/MyFirstApp
 760f7c3..549c1c4  master    -> origin/master
Auto-merging index.html
hint: Waiting for your editor to close the file... 1 [sig] bash 64! sigpac
ket::process: Suppressing signal 18 to win32 process (pid 7368)
Merge made by the 'recursive' strategy.
 index.html | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

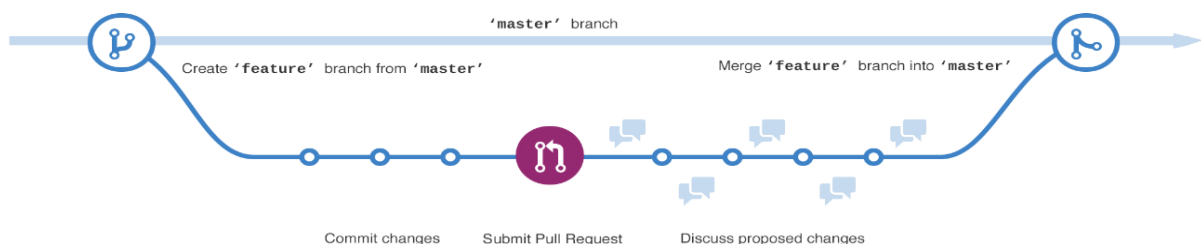
*Note that the changes made in master repository will be available in the local project of your computer.*

## 8. Create a branch in GitHub

**Branching** allows to work on different versions of a repository at one time.

Repository has one branch named master by default which is considered to be the definitive branch. Branches are used to experiment and make edits before committing them to master.

When a branch is created off the master branch, a copy or snapshot of the master at that time will be created. If one user made changes to the master branch another user can pull those updates while working on his/her own branch.

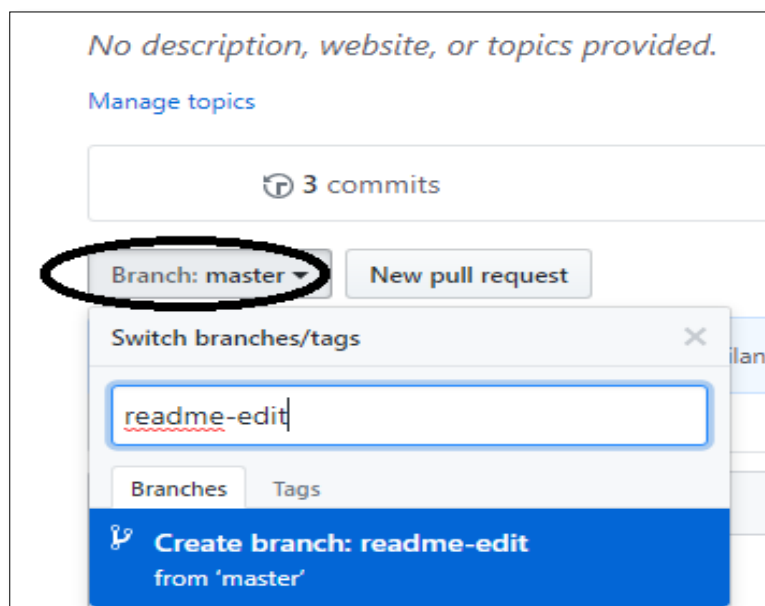


### Lab Exercise – GitHub

#### IT3040 – IT Project Management

#### Semester 1

- 8.1. Go to your new repository MyFirstApp.
- 8.2. Click the drop down at the top of the file list that says **branch: master**.
- 8.3. Type a branch name and readme-edits into the new branch text box.
- 8.4. Select **Create branch** box or hit "Enter" on your keyboard.



Now there are two identical branches, master and readme-edit.

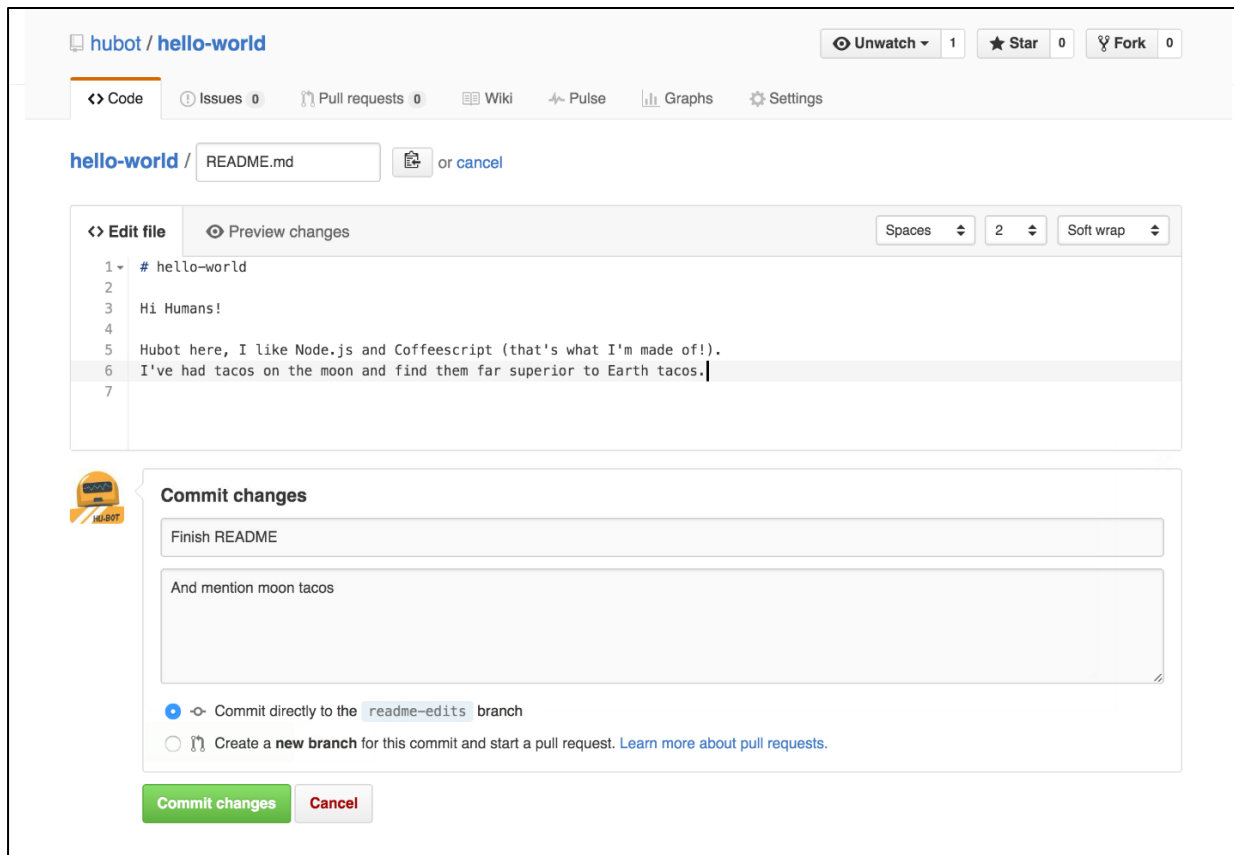
#### 9. Make and commit changes

- 9.1. Click the README.md file
- 9.2. Click the pencil icon in the upper right corner of the file view to edit
- 9.3. Write your details in the editor
- 9.4. Write a commit message that describes the changes
- 9.5. Click **commit changes** button

### Lab Exercise – GitHub

IT3040 – IT Project Management

Semester 1



#### 10. Open a pull request

- 10.1. Click the pull **request** tab and select **new pull request**.
- 10.2. In the **Example Comparisons box**, select the branch, readme-edits to compare with master (the original).
- 10.3. Note the changes in diffs on the compare page.
- 10.4. Once the changes are finalized, click the create **pull request** button.
- 10.5. Give a title and write a brief description of changes for the pull request.

#### 11. Merge your pull request

- 11.1. Click **merge pull request** button to merge changes into the master.
- 11.2. Click **confirm merge**.
- 11.3. Delete the branch using **delete branch** button, since its changes have been incorporated.

### Lab Exercise – GitHub

#### IT3040 – IT Project Management

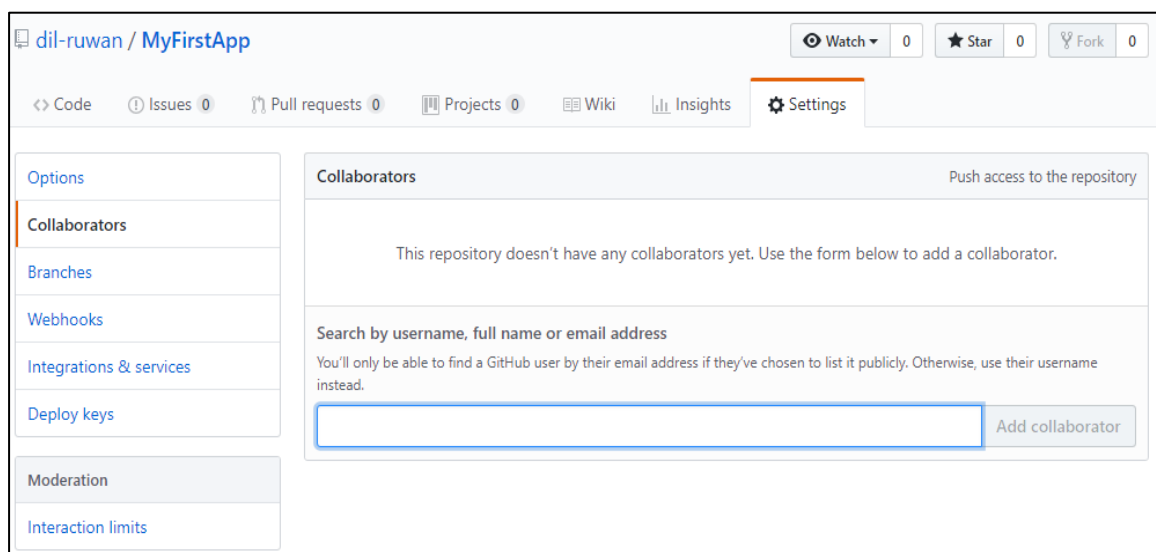
#### Semester 1

##### 12. Multiple users and single repository.

There are two main workflows. Both assume each developer has set up a GitHub account which is free and sensible:

12.1. Add other developers as collaborators to your main repository so they can push their changes to it.

Collaborators are preferable if there are group members making or maintaining a software product. There is one repository for the project on GitHub and each developer will have a local repository for their development work.



12.2. Let other developers fork your repository and issue pull requests to integrate changes.

The forking is useful if there are other teams or developers using their own version of your product. It gives them a way (pull requests) to offer their changes back to enhance your product. Each fork is generally considered as a distinct viable variant of your project. Forking is also useful in an open source environment while you are 'coming to trust' a potential collaborator. If you find yourself routinely accepting their pull requests, you may 'promote' them to be a collaborator.