# CHAPTER III

## General Theory

### **Git**



Picture ‎3.1 Git

Git is a mature, actively maintained open source project originally developed in 2005 by Linus Torvalds, the famous creator of the Linux operating system kernel. A staggering number of software projects rely on Git for version control, including commercial projects as well as open source. Developers who have worked with Git are well represented in the pool of available software development talent and it works well on a wide range of operating systems and IDEs (Integrated Development Environments).

* + 1. **GitHub**

****

Picture 3.2 GitHub

Though often used synonymously, Git and GitHub are two different things. Git is a particular implementation of version control originally designed by Linus Torvalds as a way of managing the Linux source code. Other systems of version control exist though they are used less frequently. Git can be used to refer both to a particular approach taken to version control and the software underlying it.

GitHub is a company which hosts Git repositories (more on this below) and provides software for using Git. This includes ‘GitHub Desktop’ which will be covered in this tutorial. GitHub is currently the most popular host of open source projects by number of projects and number of users.

Although GitHub’s focus is primarily on source code, other projects, such as the Programming Historian, are increasingly making use of version control systems like GitHub to manage the work-flows of journal publishing, open textbooks and other humanities projects. Becoming familiar with GitHub will be useful not only for version controlling your own documents but will also make it easier to contribute and draw upon other projects which use GitHub.

GitHub Desktop will allow us to easily start using version control. GitHub Desktop offers a Graphical User Interface (GUI) to use Git. A GUI allows users to interact with a program using a visual interface rather than relying on text commands. Though there are some potential advantages to using the command line version of Git in the long run, using a GUI can reduce the learning curve of using version control and Git. If you decide you are interested in using the command line you can find more resources at the end of the lesson.

### **Advantages and Disadvantages of Using Git**

* + 1. **Advantages**

1. **Performance**

The raw performance characteristics of Git are very strong when compared to many alternatives. Committing new changes, branching, merging and comparing past versions are all optimized for performance. The algorithms implemented inside Git take advantage of deep knowledge about common attributes of real source code file trees, how they are usually modified over time and what the access patterns are.

Unlike some version control software, Git is not fooled by the names of the files when determining what the storage and version history of the file tree should be, instead, Git focuses on the file content itself. After all, source code files are frequently renamed, split, and rearranged. The object format of Git's repository files uses a combination of delta encoding (storing content differences), compression and explicitly stores directory contents and version metadata objects.

1. **Security**

Git has been designed with the integrity of managed source code as a top priority. The content of the files as well as the true relationships between files and directories, versions, tags and commits, all of these objects in the Git repository are secured with a cryptographically secure hashing algorithm called SHA1. This protects the code and the change history against both accidental and malicious change and ensures that the history is fully traceable. With Git, you can be sure you have an authentic content history of your source code.

1. **Flexibility**

One of Git's key design objectives is flexibility. Git is flexible in several respects: in support for various kinds of nonlinear development workflows, in its efficiency in both small and large projects and in its compatibility with many existing systems and protocols.

* + 1. **Disadvantages**
       1. Git

## Working Procedure

* + 1. **Tool requirement for setup GitHub Desktop**

Before we start, we need to bring the items and software as follows:

1. Item
   * + 1. Laptop / PC



Picture 3.1 Laptop / PC

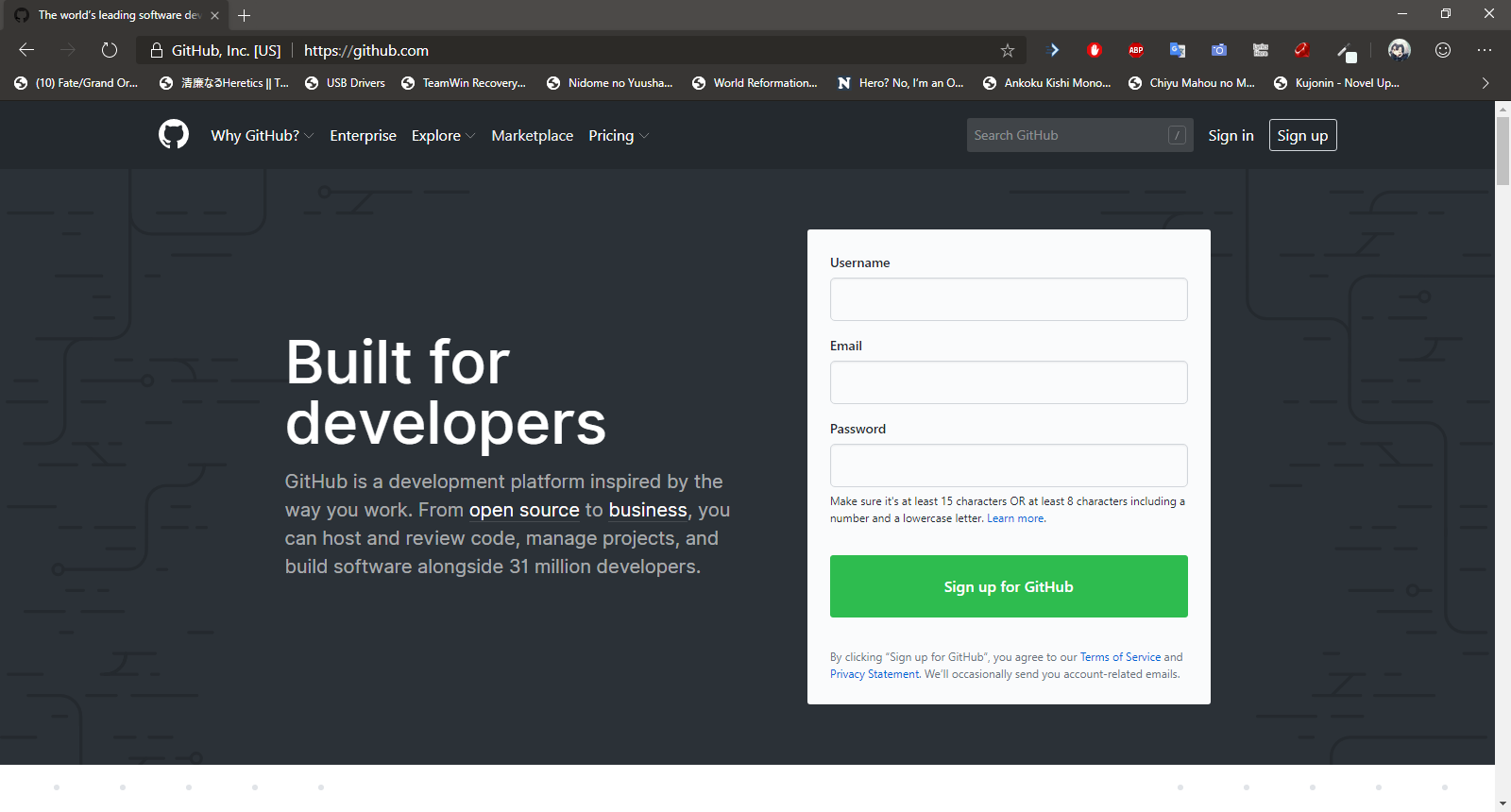
1. Software
   * + 1. GitHub Desktop Installer



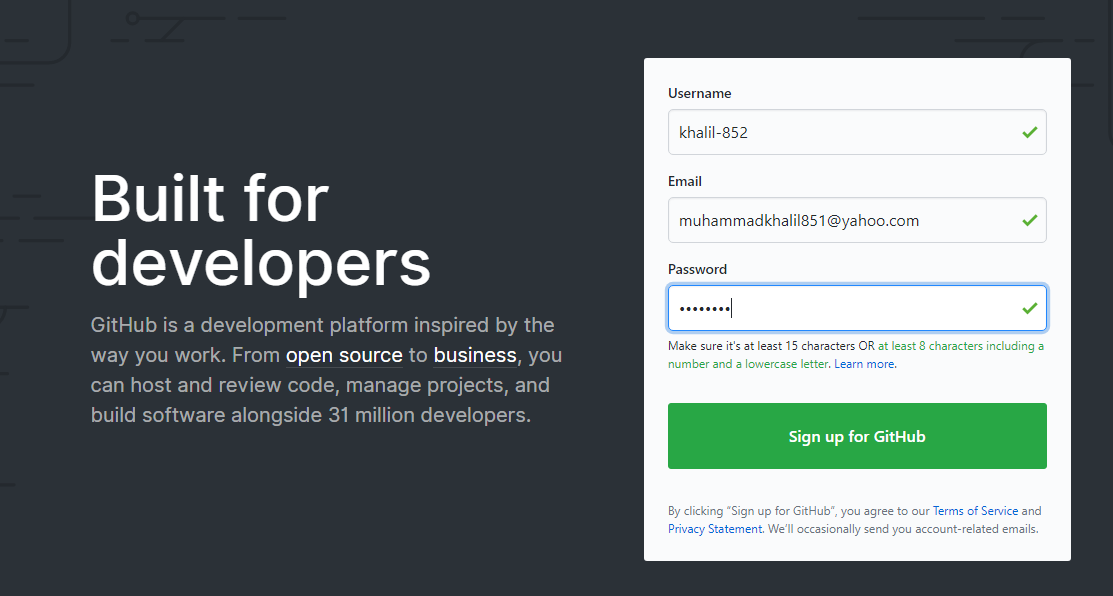
Picture 3.2 GitHub Desktop

* + 1. **Make GitHub Account**

To use GitHub and make full use of GitHub Desktop, we first need to create a GitHub account:

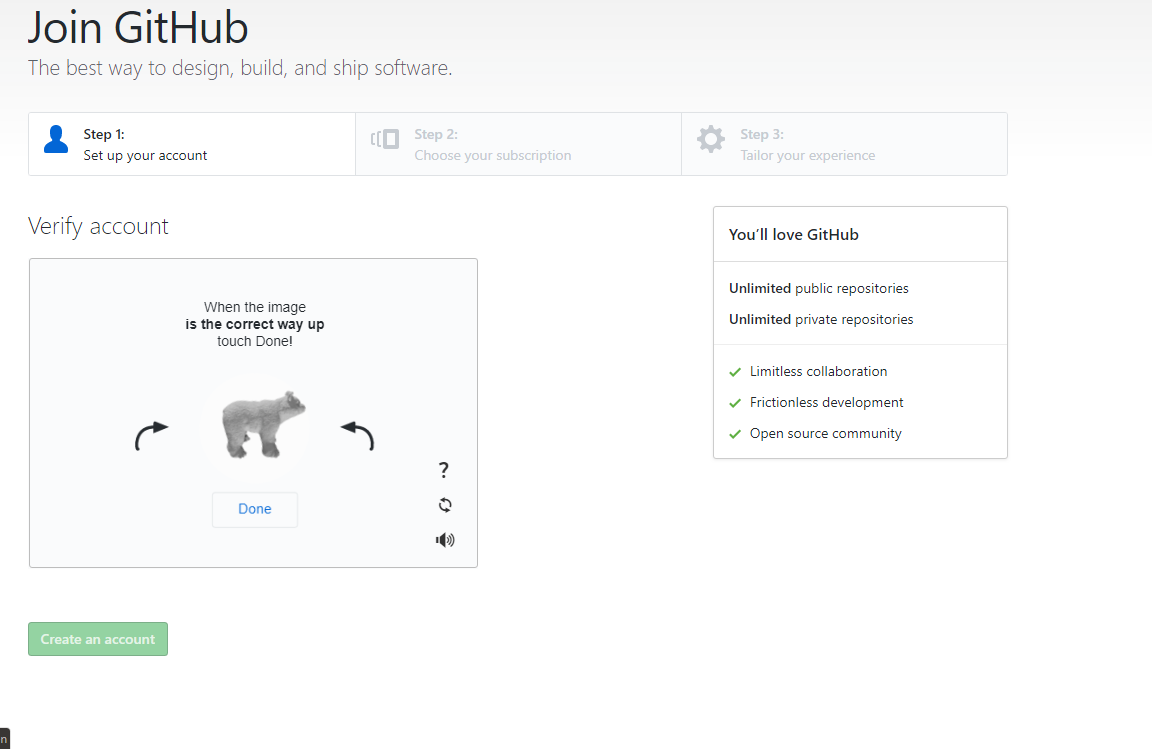
1. Open <https://github.com/> 

Picture 3.3 GitHub

1. Insert Username, email and password we want to use for account after that click sign up for GitHub 

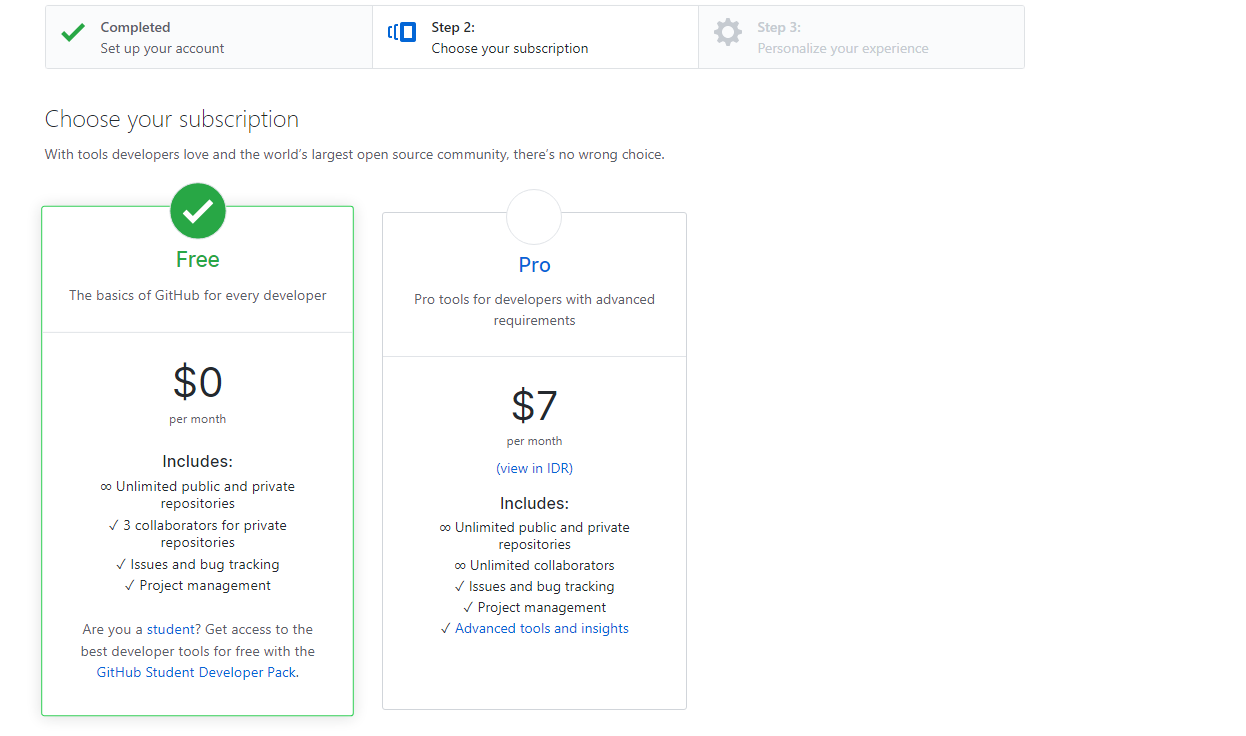
Picture 3.4 Sign Up

1. After that complete the step up for join GitHub. On step 1 we need verify account, they will give captcha for verify us is not a robot, solve the captcha then click done



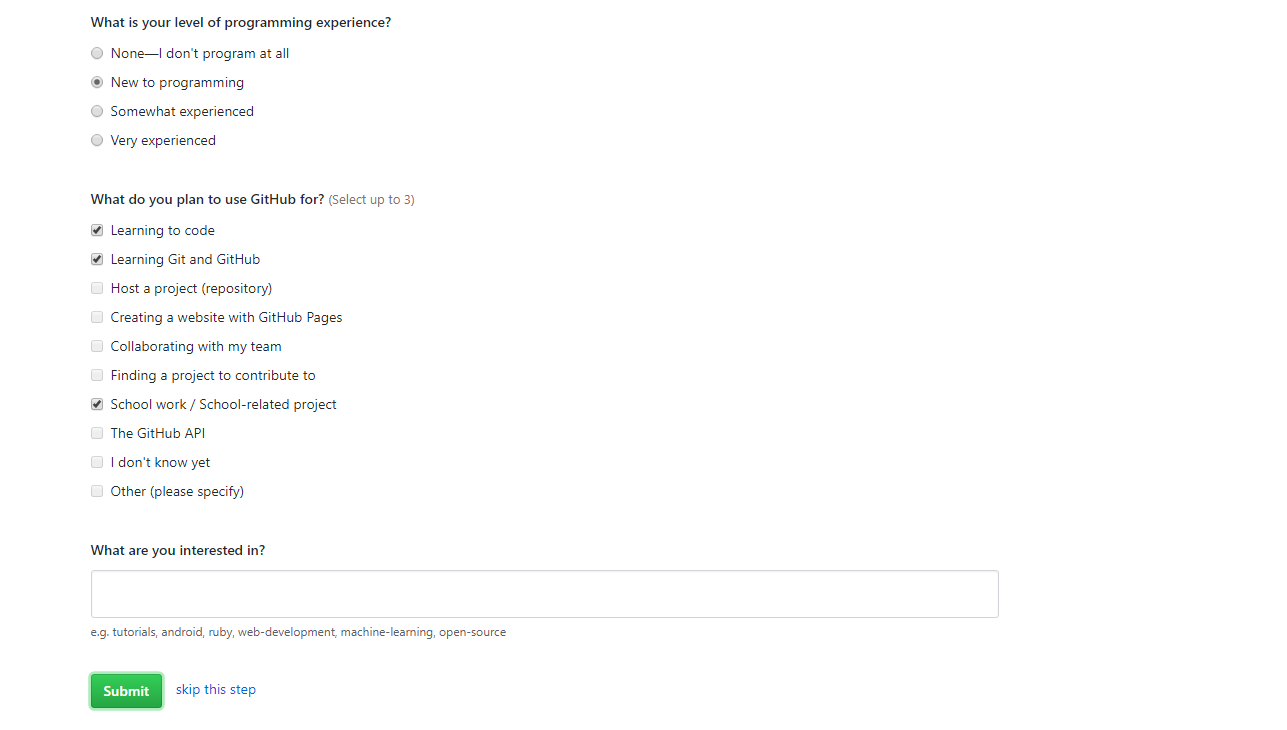
Picture 3.5 Step 1 Human validation

1. After that, choices subscription for account, on this report author use free subscription then click continue on the bottom page

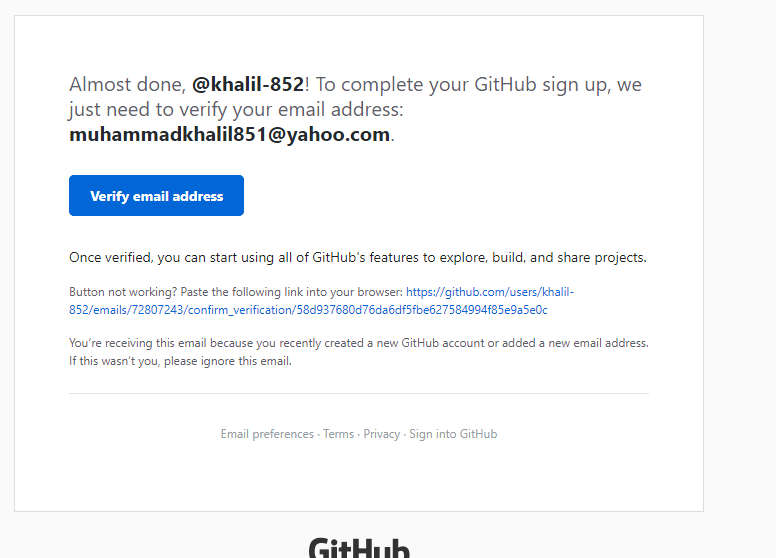


3.6 Choose Subscription

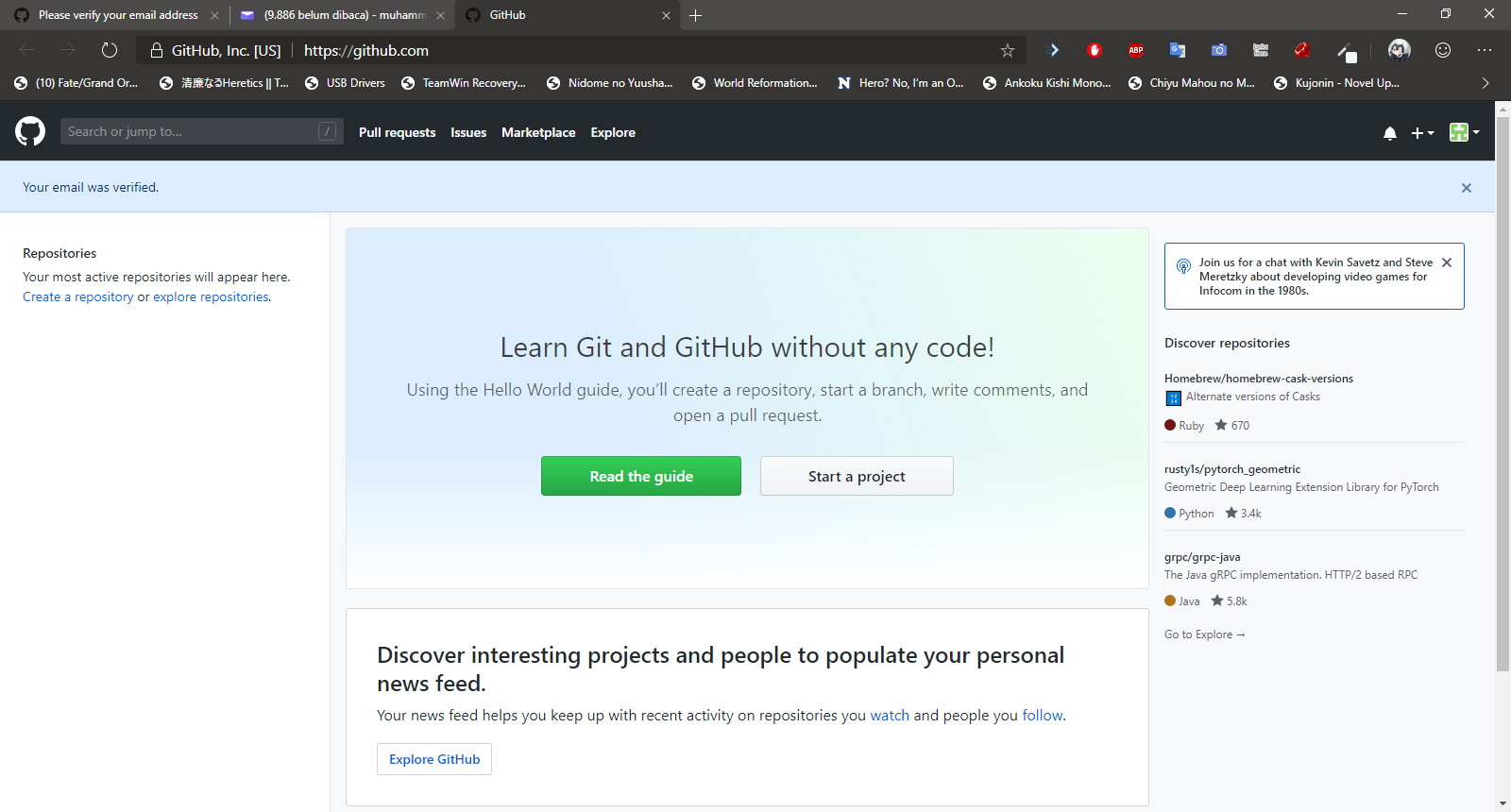
1. On step 3, insert our experience on programming and what do you want to do for use GitHub or just skip it with the button Skip this step, after insert it click submit to continue



Picture 3.7 User Experience

1. After that GitHub will send an email to our email address, click verify email address to completing making our account

Picture 3.8 Email verify

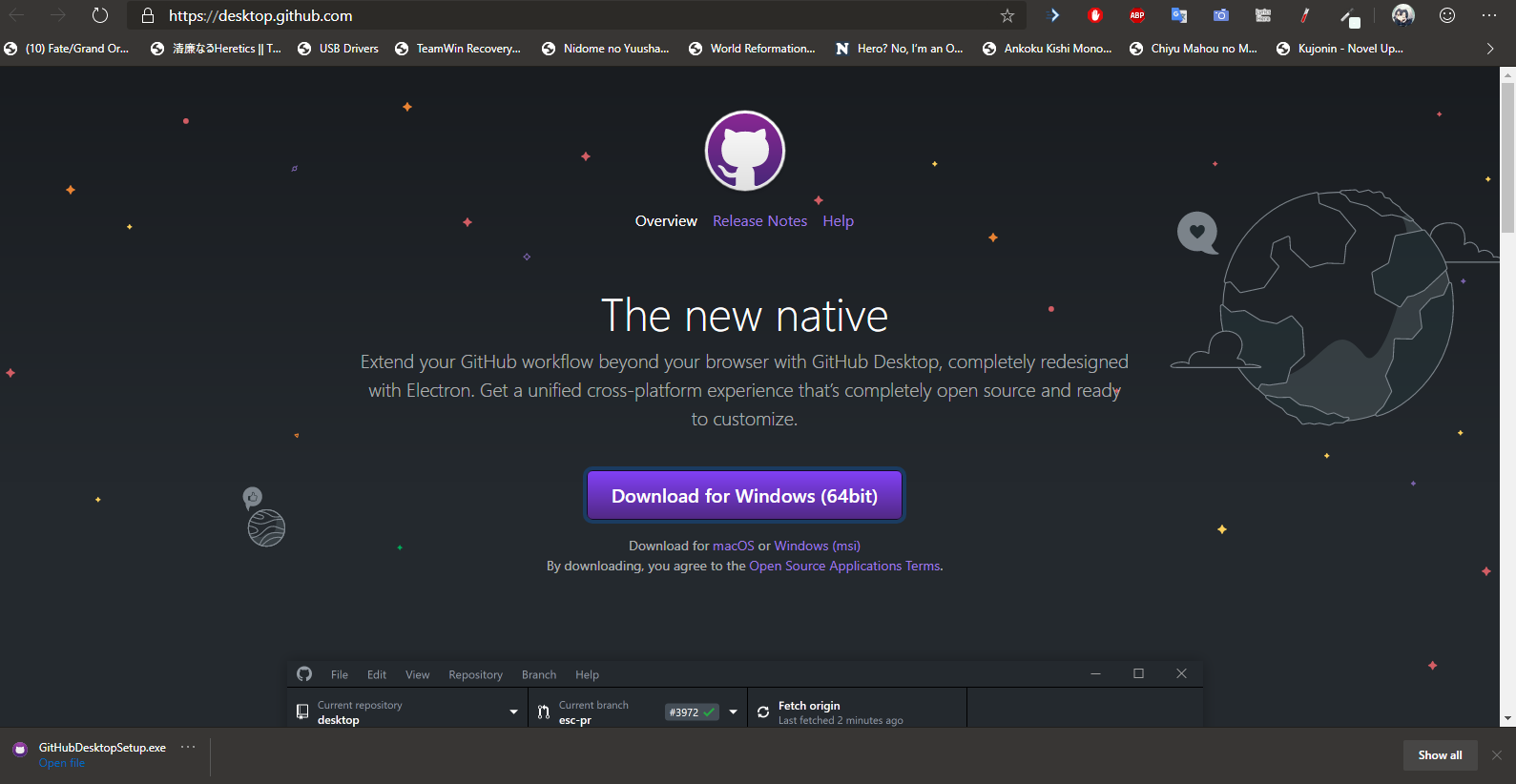


Picture 3.9 GitHub Home page

* + 1. **Install GitHub Desktop**

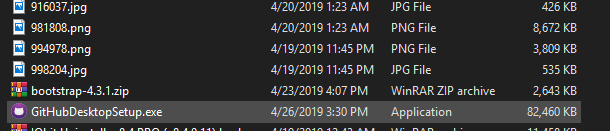
After create GitHub account, download and install GitHub Desktop:

1. Download GitHub Desktop from official Website: <https://desktop.github.com/>



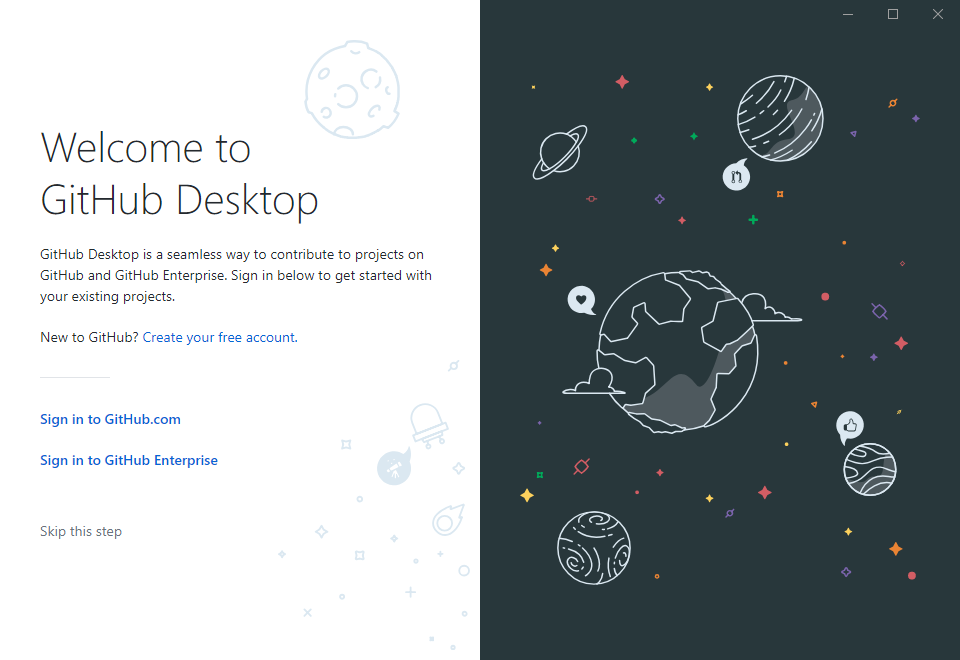
Picture 3.10 Download GitHub Desktop

1. Open the file to install GitHub Desktop, it will install automatically



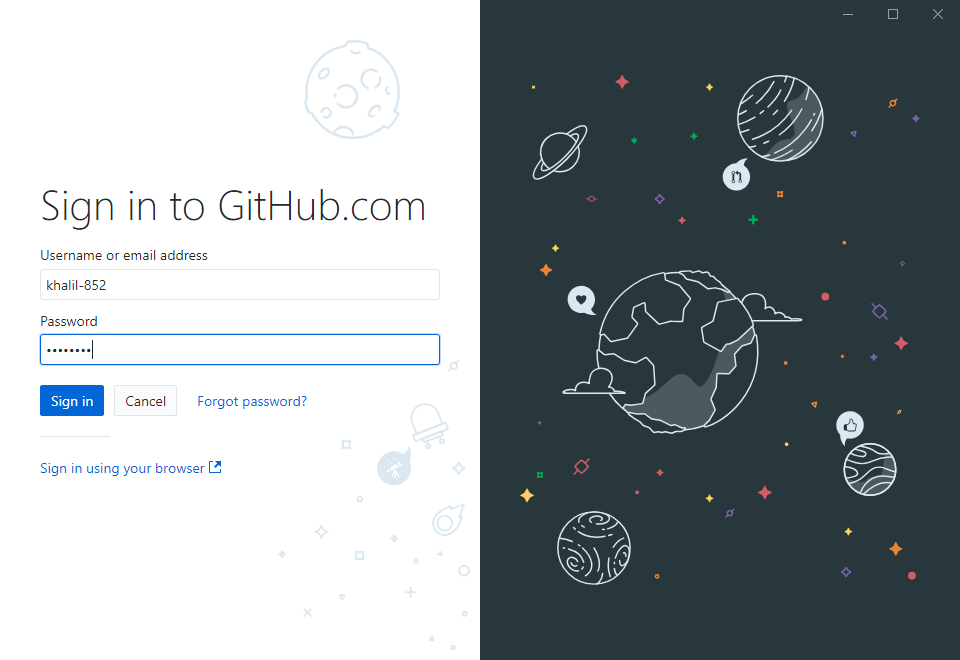
Picture 3.11 GitHub Desktop Installer

1. Click Sign in to GitHub.com to sign in with account we create

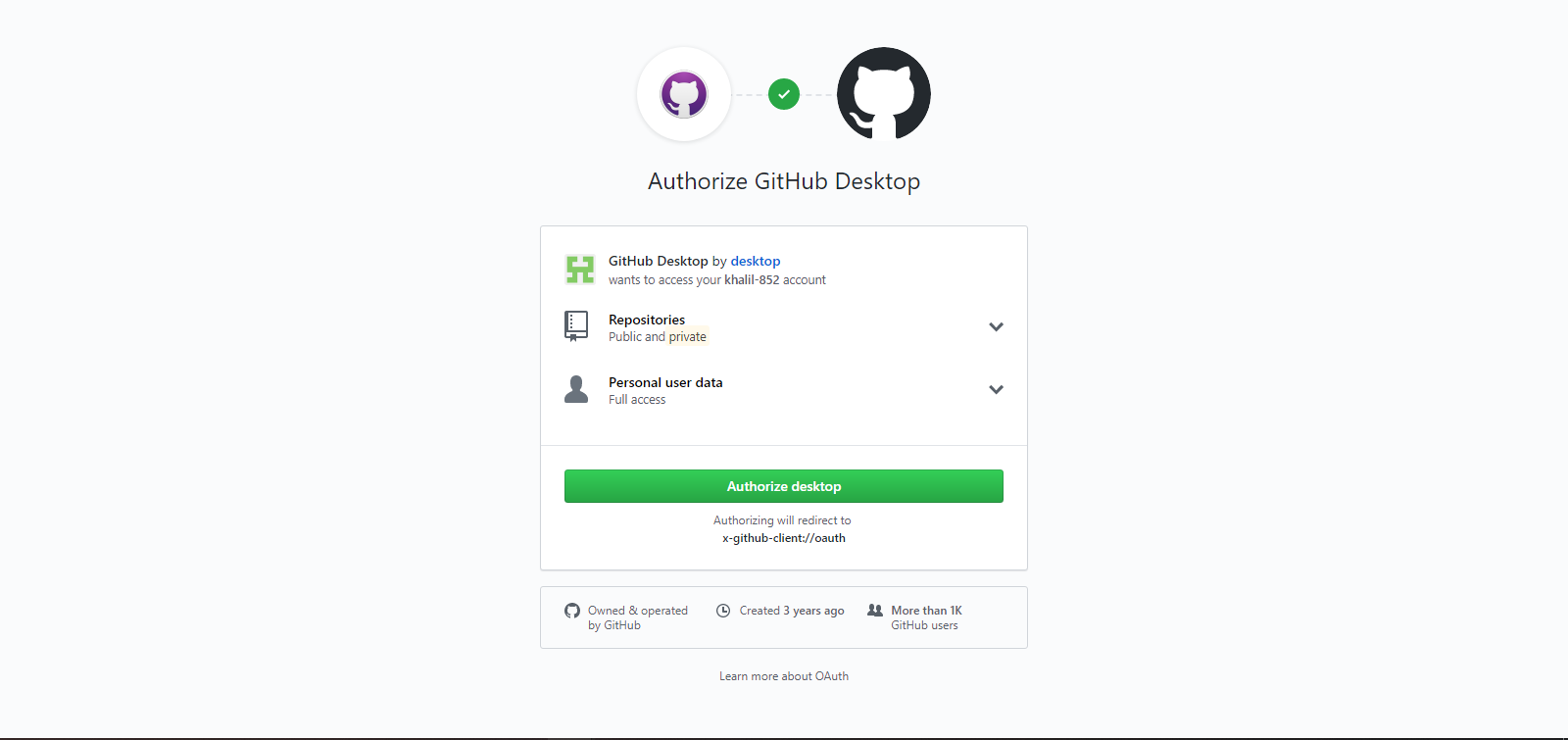


Picture 3.12 GitHub Desktop Sign In

1. Insert Username or email address and our password then click sign in, we can use our browser instead it will open our browser then click authorize desktop to sign in with our browser

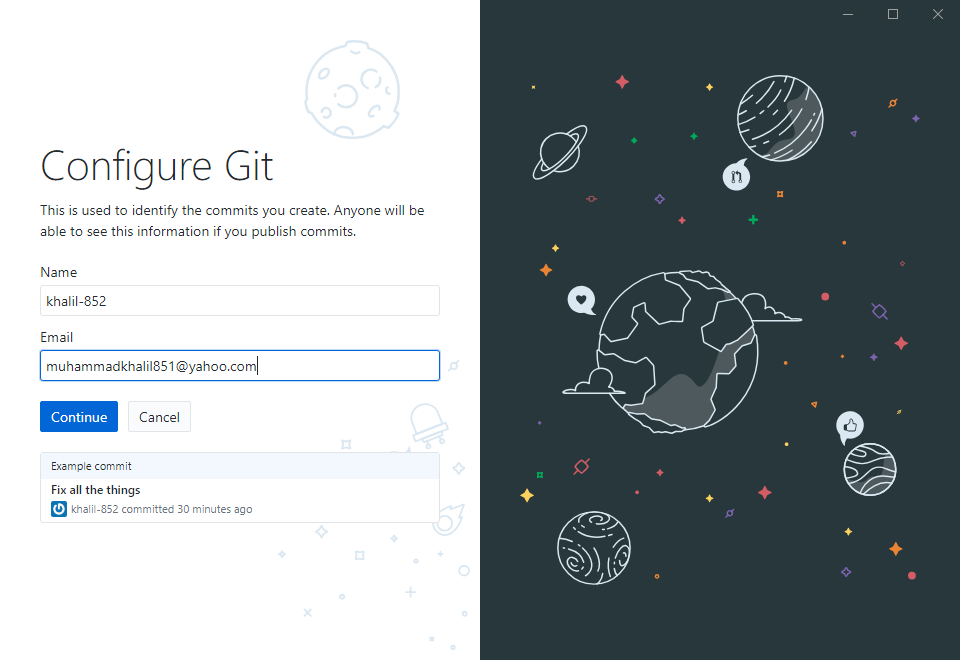


Picture 3.13 Sign in GitHub.com account



Picture 3.14 Authorize GitHub Desktop

1. Now configure Git just leave it as is if it automatically fills, if not, insert Name and email address for sync it with GitHub account (we can change it on setting), after that click continue to finishing installation GitHub Desktop.



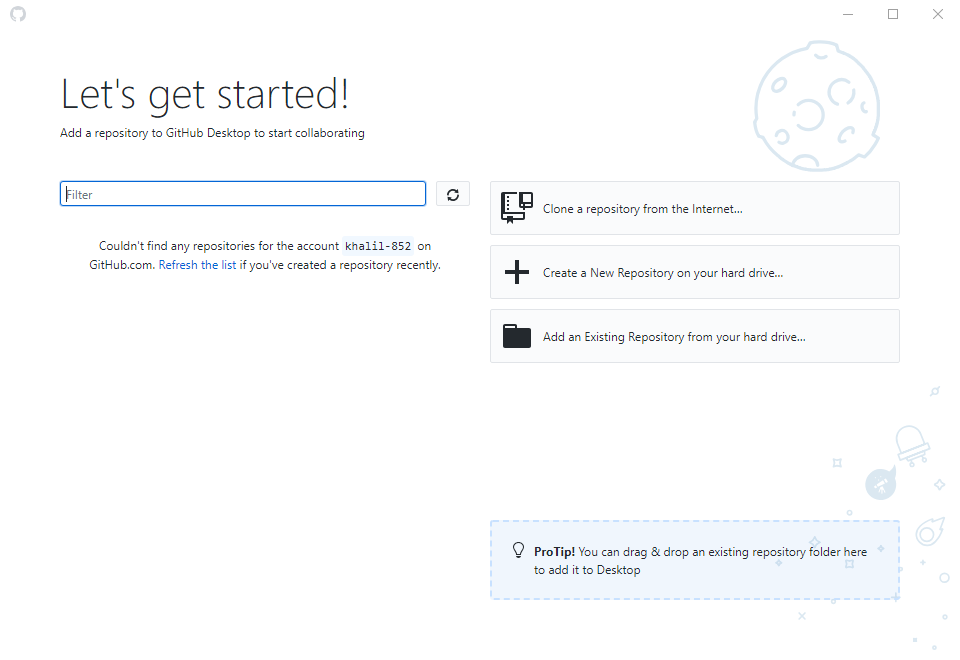
Picture 3.15 Configure Git for sync

* + 1. **Repository**

Git repository is a virtual storage of your project. It allows you to save versions of your code, which you can access when needed.

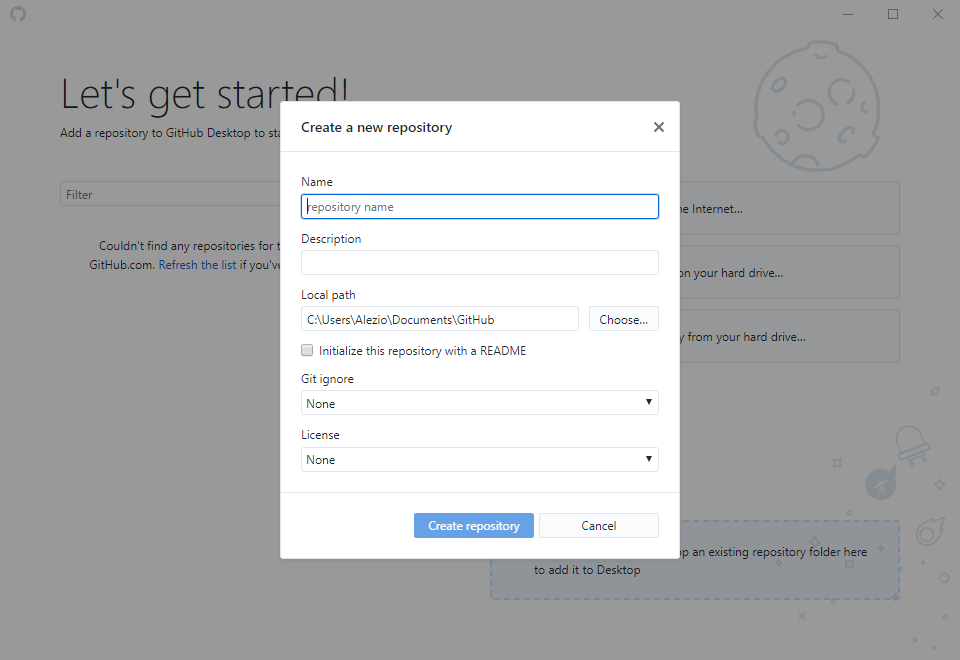
After install GitHub Desktop, make a new repository to GitHub desktop to start:

1. Click “Create a New Repository on your hard drive” to make a new repository



Picture 3.16 Set-up Repository

1. There a few columns that need to fill for making a repository:



Picture 3.17 Create local Repository

* Name: Name of repository that will be make
* Description
* Local path: where will the repository will be making on local hard drive
* Git ignore: when the file is change or remove, git will ignore it and will not record it on versions
* License: this is need it when we want to make a copyright for our project

On this report, author will make a website repository for developing website, after filling out the column, click “Create Repository” to create repository.

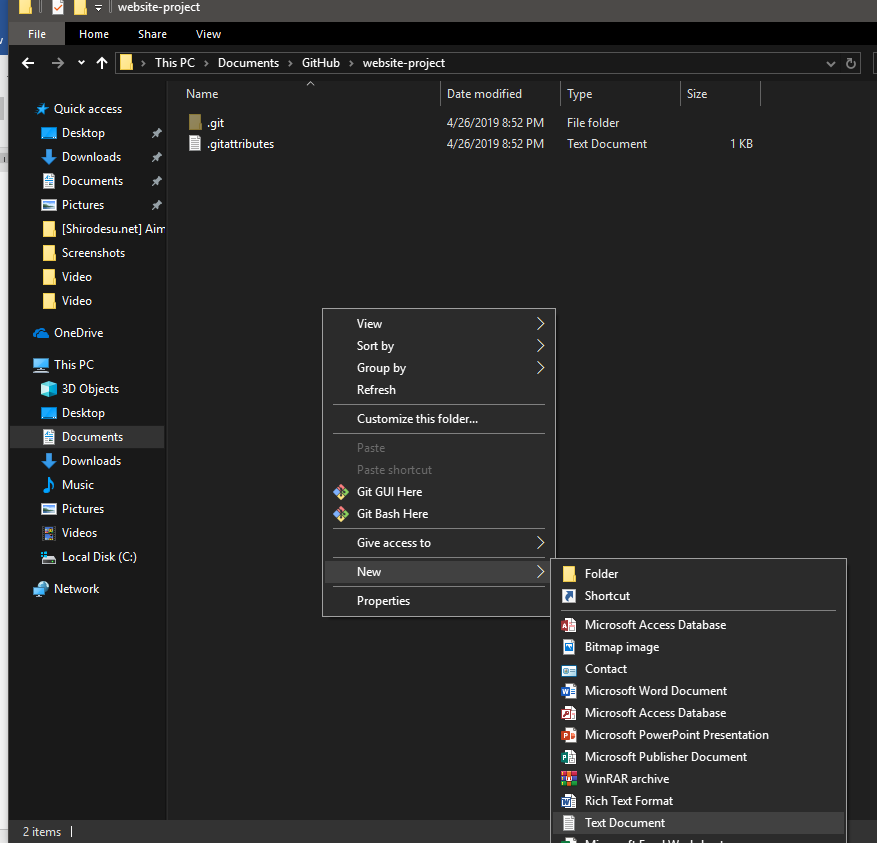
* + 1. **Changes and Commit**

GitHub Desktop have a slightly different ways of how command add, modified and staged work. In GitHub Desktop add, modified and staged is merged as one feature called Changes. Changes feature is that will automatically detect what files are added, changed and deleted in the local repository.

After create local repository we can add file to repository through directory we add-in to it.

Navigate to local repository we made, on this report author made local repository on “C:\Users\Alezio\Documents\GitHub\website-project”

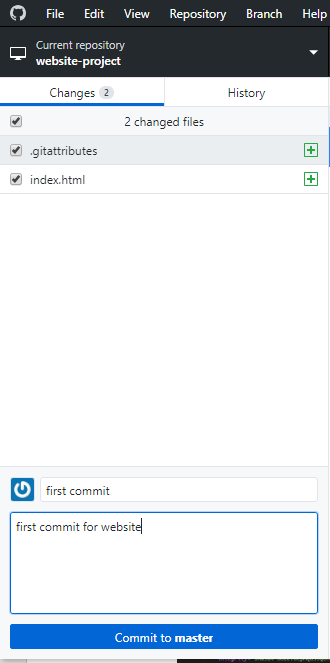
After that make a file name index.html with right click then Next > Text Document, then rename it to index.html



Picture 3.18 Create a new file

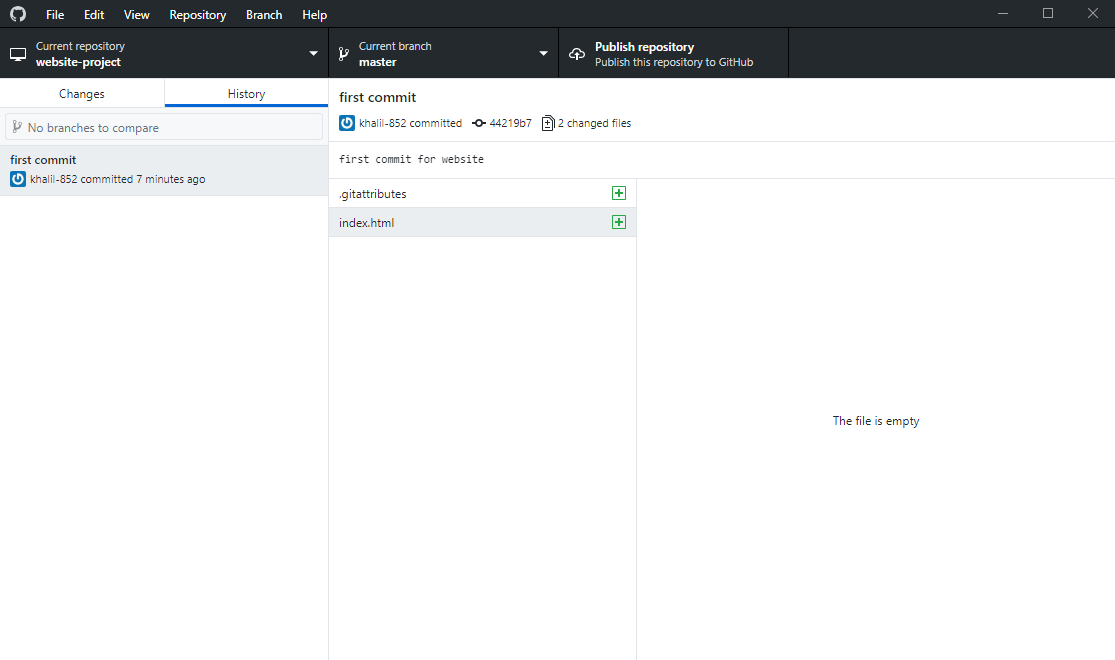
On GitHub Desktop, file that created or moving it to local repository will be added in Changes tab as 

To Commit what that change in repository, insert the title for commit that will be make in Summary that located in the bottom Changes tab, for the description it optional, then click commit to master to commit.



Picture 3.19 “Changes” feature

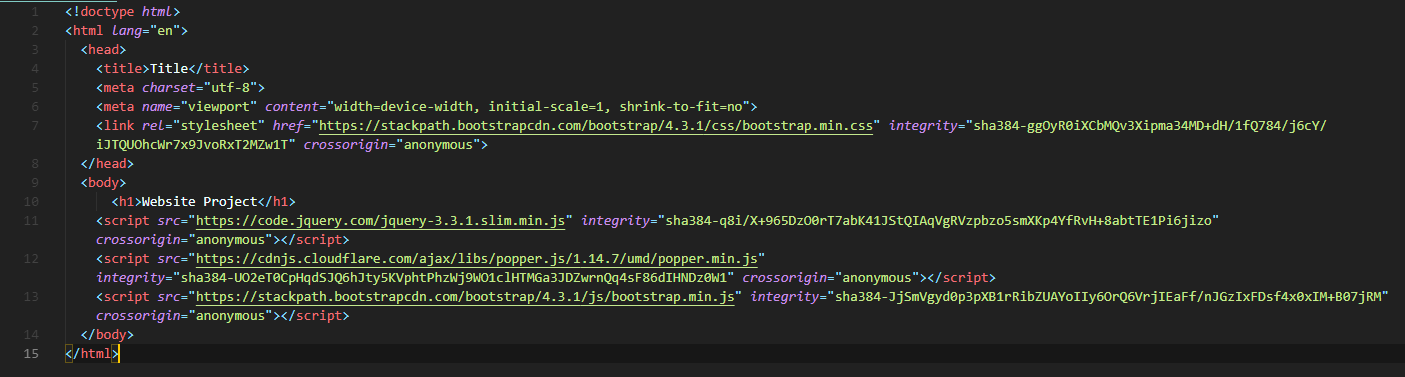
To see what has been committed, click on History tab beside Changes tab



Picture 3.20 “History” feature

Now open index.html with text editor (Notepad, Visual studio Code Atom, etc.). On this report author use Visual Studio code as text editor.

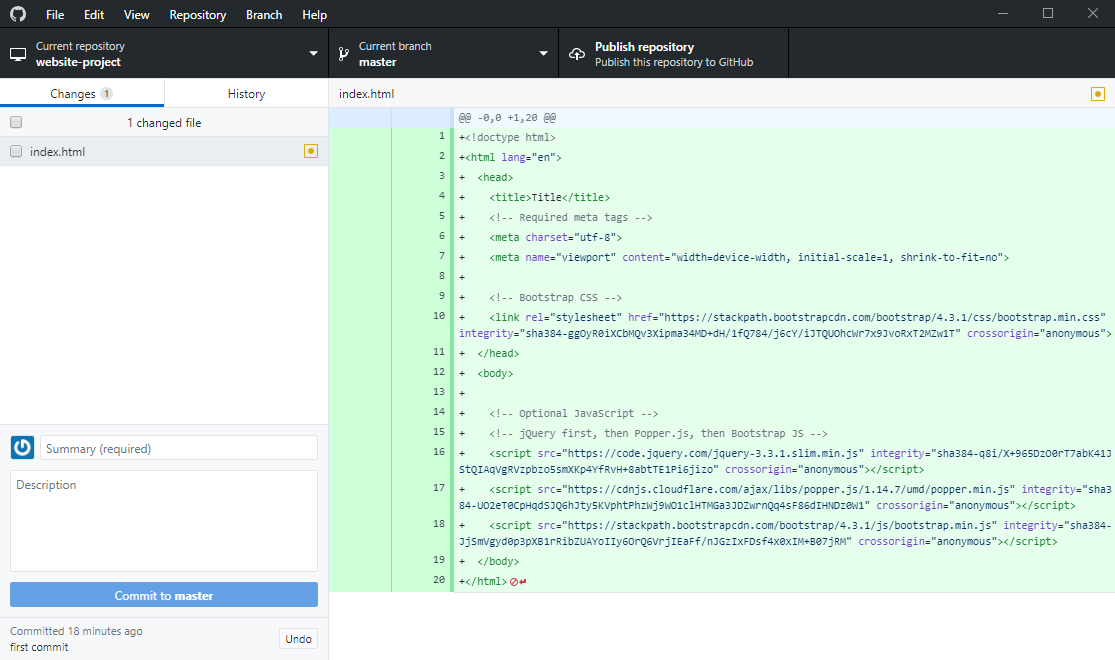
Create basic HTML template on the empty file, the author uses Bootstrap as template on this report After making it save the file (ctrl + S)



3.21 Coding line example

Now in the Changes tab there will be index.html file with  , this symbol is a sign that in the file there has been a change from the previous commit, on the right side you will see changes that have been occurred in the file:

* Text with green highlight is a text that added to file
* Text with red highlight is a text that have been remove from file



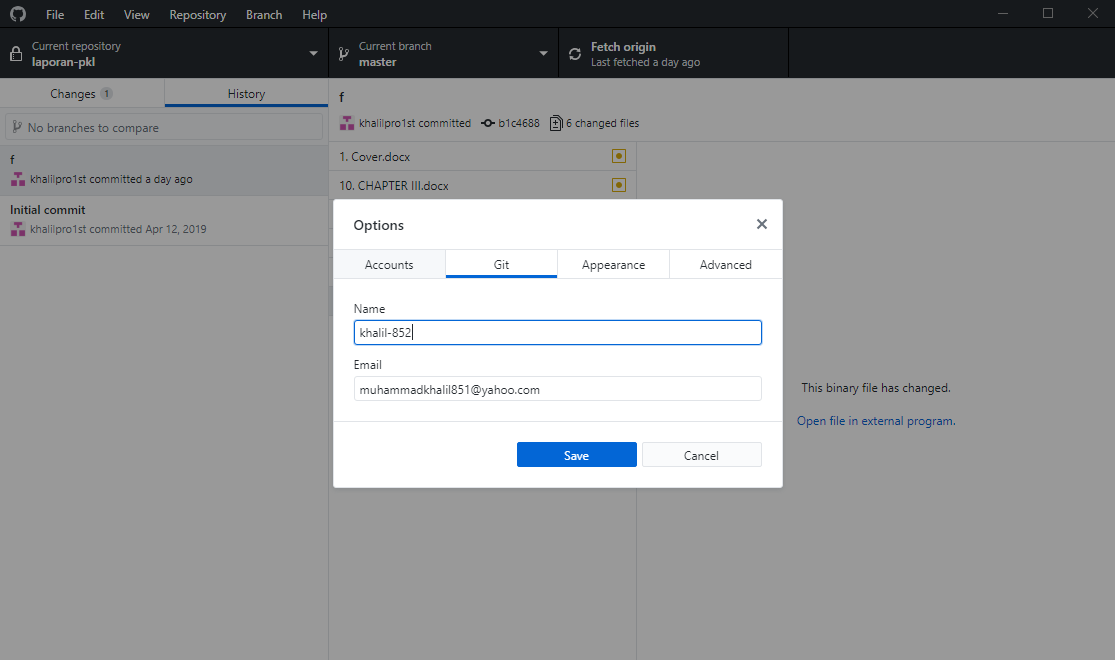
3.22 Line that have been changes

* + 1. **Remote Repository**

Remote repositories are versions of your project that are hosted on the Internet or network somewhere. You can have several of them, each of which generally is either read-only or read/write for you. On this report author use GitHub as remote repository with GitHub account.

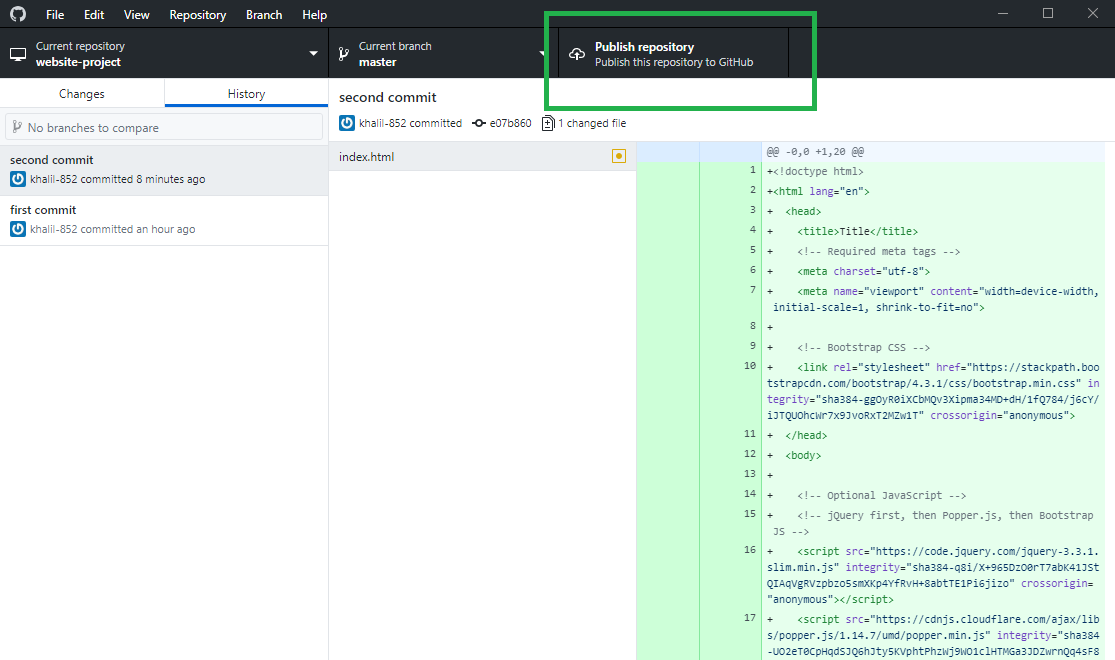
To add local repository to Remote repository on GitHub is with Push method. What the Push do is to push commits made on your local repository to a remote repository:

1. Open file > Options, Check if account on GitHub Desktop is same with that have been create earlier, then click Git to Check if Name and email is same with GitHub Account, after that click Save



Picture 3.22 GitHub Desktop Options

1. Now to push the Commit to GitHub, click Publish repository on top right GitHub Desktop or with Ctrl + P

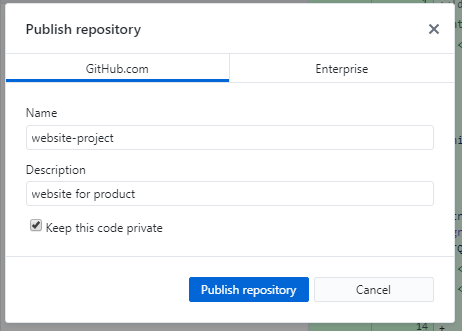


Picture 3.23 Publish repository Button

1. On Publish repository tab select GitHub.com, there will be column need to fill (it will automatically fill with information from local repository):

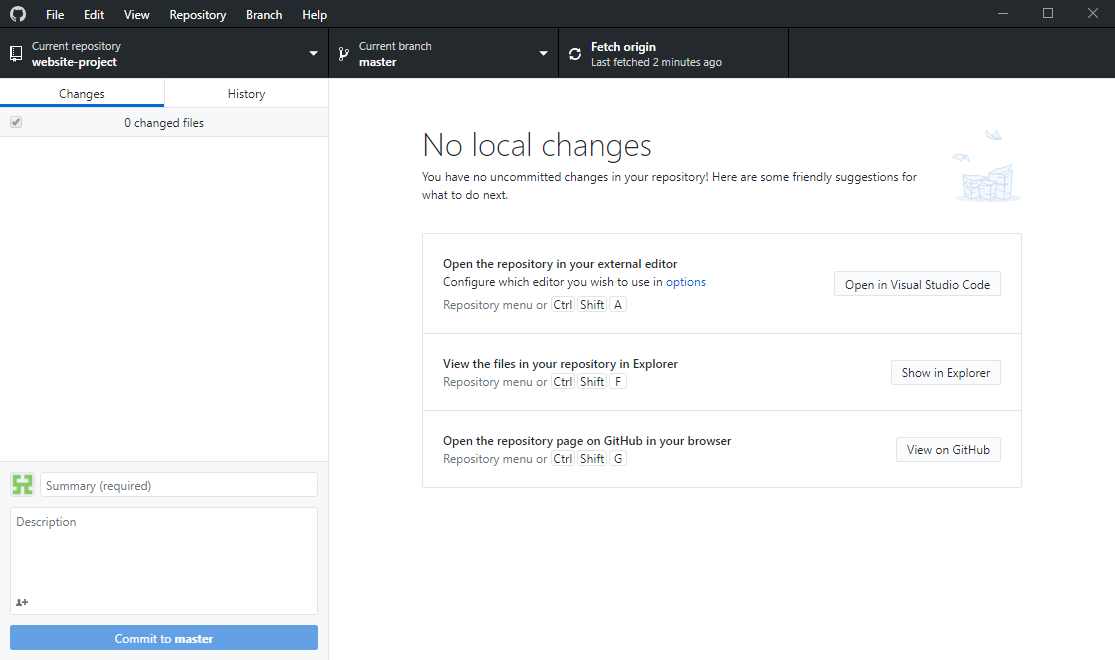
* Name: Name of remote repository on Github.com (try to make it same with local repository)
* Description: Description of repository (purpose, feature on it, etc.)
* Keep this code private: it will make repository private if it checked (only author or person with author permission can access it). On this report author unchecked it so anyone can access it

After that click Publish repository to push commit to GitHub.com



Picture 3.24 Create remote repository

1. If success Publish repository button will change to Fetch origin



Picture 3.25 Publish repository have changes to Fetch Origin