VCS (Version Control System)

TUGAS PEMOGRAMAN 2

NAMA : ADI TIO ILHASA

NIK : 311410014

KELAS : TI.14.E.1

DOSEN : Bpk. AGUNG NUGROHO S.KOM

Mengenal Version Control

Version Control merupakan sistem yang mencatat perubahan file atau kumpulan file (baik kode program maupun jenis file yang lain) dari waktu ke waktu sehingga Anda dapat mengingat versi tertentu pada saat mendatang.

Jika Anda adalah seorang programmer atau desainer grafis atau web dan ingin menyimpan setiap versi atau perubahan dari kode atau layout Anda, maka akan sangat bijaksana jika Anda menggunakan VCS (Version Control System). CVS memungkinkan Anda untuk mengembalikan file pada keadaan sebelumnya, mengembalikan seluruh proyek pada keadaan awal, mereview perubahan yang telah dibuat dari waktu ke waktu, melihat siapa yang terkahir melakukan perubahan file sehingga mungkin menyebabkan masalah.

Selama perkembangannya terdapat 3 jenis VCS:

Local Version Control System

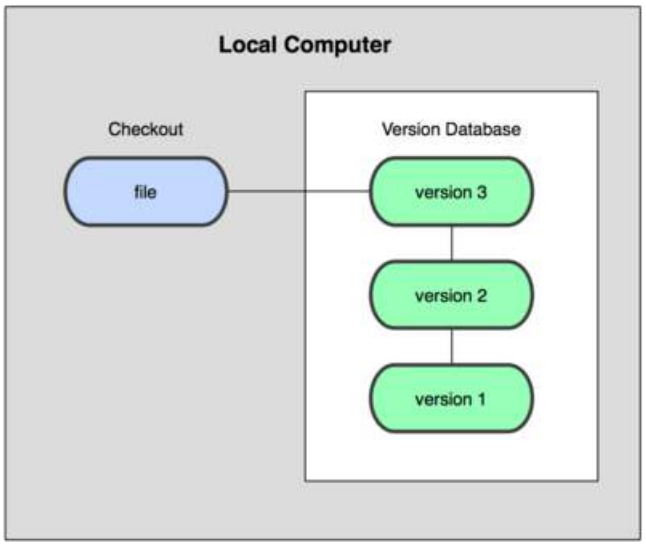
Centralized Version Control System

Distributed Version Control System

Local Version Control System

Sebagian besar orang melakukan pengontrolan versi dengan cara menduplikasi file-file ke direktori yang lain mungkin dengan memberikan penanggalan. Cara seperti ini sangat umum karena sangat sederhana, namun cenderung rawan kesalahan. Bisa jadi Anda lupa letak direktori Anda atau terjadi kesalahan penulisan file atau menyalin file yang salah.

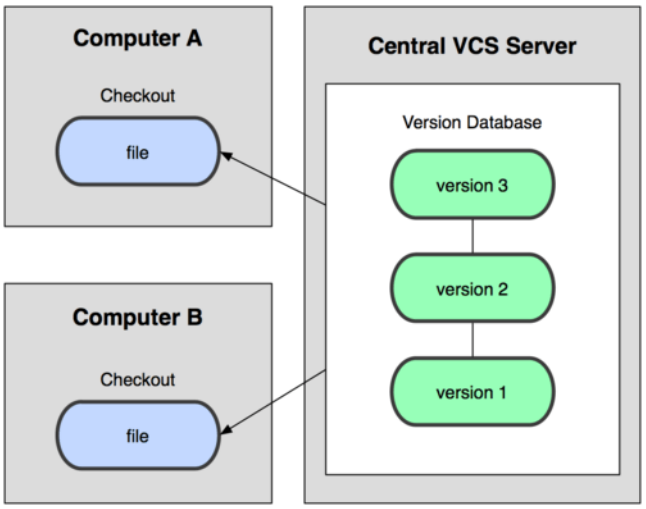
Untuk mengatasi permasalahan ini, para programmer mengembangkan VCS lokal yang memiliki basis data sederhana untuk menyimpan semua perubahan pada file yang terhubung dengan VCS. Salah satunya adalah RCS. RCS bekerja dengan cara menyimpan kumpulan patch dari satu perubahan ke perubahan lainnya dalam format khusus pada drive. Patch ini yang kemudian dapat digunkan untuk mengembalikan keadaan suatu file pada keadaan sebelumnya.



Centralized Version Control System

Permasalahan yang terjadi ketika menggunakan Local VCS adalah dukungannya terhadap kolaborasi. Maksudnya adalah ketika suatu file atau project ingin digunakan oleh lebih dari satu orang. Maka dari itu dibangunlah Centralized Version Control Systems (CVCSs). Sistem ini diantaranya CVS, Subversion dan Perforce yang memiliki sebuah server untuk menyimpan setiap versi file dan dapat terdiri dari beberapa user yang nantinya dapat menyalin (checkout) file dari server pusat.

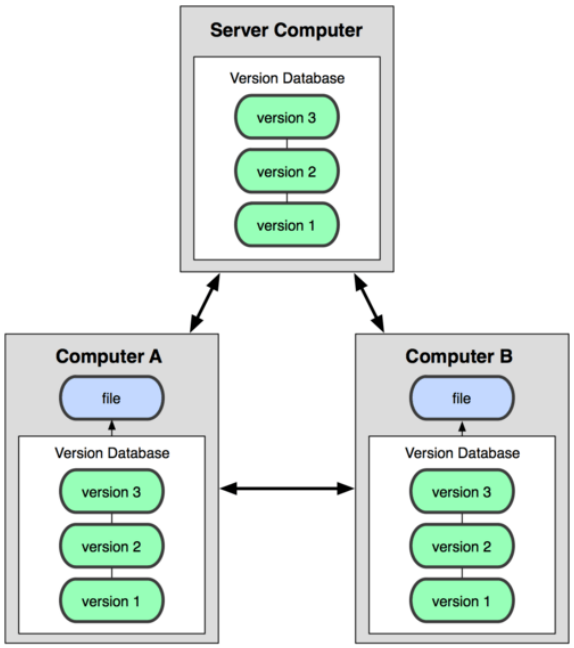
Sistem ini memiliki beberapa kelebihan, terutama jika dibandingkan dengan Local VCS. Misalnya, setiap orang pada tingkat tertentu dapat mengetahui apa yang orang lain lakukan pada file atau proyek. Administrator memiliki kendali yang tinggi atas siapa dan apa yang dapat dilakukan oleh user/klien. Sehingga ini menjadi lebih mudah dibandingkan menangani database lokal pada setiap user/klien.



Meski demikian, sistem dengan tatanan seperi ini memiliki kelemahan serius. Kelemahan nyata yang direpresentasikan oleh sistem dengan server terpusat. Jika server mati untuk beberapa jam, maka tidak ada seorangpun yang bisa berkolaborasi atau menyimpan perubahan terhadap apa yang mereka relah kerjakan. Jika hardisk yang menyimpan basis data mengalami kerusakan, dan salinan yang benar/fix belum tersimpan, Anda akan kehilangan setiap perubahan dari proyek kecuali snapshot-snapshot yang dimiliki oleh setiap user/klien pada komputernya masing-masing. Lokal VCS juga mengalami nasib yang sama jika Anda menyimpan seluruh histori perubahan proyek pada satu tempat, Anda mempunyai resiko kehilangan semuanya.

Distributed Version Control System

Inilah saatnya bagi Distributed Version Control System untuk mengambil bagian. Dalam sebuah DVCS (seperti Git, Mercurial, Bazaar atau Darcs), klien tidak hanya melakukan checkout untuk snapshot terakhir setiap file, namun mereka memiliki salinan penuh dari repositori tersebut. Jadi jika server mati, dan sistem berkolaborasi melalui server tersebut, maka klien manapun dapat mengirimkan salinan repositori tersebut kembali ke server. Setiap checkout/commit pada DVCS merupakan sebuah backup dari keseluruhan data.



Lebih jauh lagi, kebanyakan sistem seperti ini mampu menangani sejumlah remote repository dengan baik, jadi Anda dapat melakukan kolaborasi dengan berbagai kelompok kolaborator dalam berbagai cara secara bersama-sama pada suatu proyek. Hal ini memungkinkan Anda untuk menyusun beberapa jenis alur kerja yang tidak mungkin dilakukan pada CVCSs.

Ada beberapa VCS yang umum digunakan diantaranya adalah

1. Bazaar

Bazaar (sumber: http://bazaar.canonical.com/en/)

Bazaar merupakan bagian dari GNU Project, Bazaar adalah free softwareyang disponsori oleh Canonical. Salah satu layanan yang menggunakan Bazaar adalah Launchpad, sebuah tempat dimana aplikasi Ubuntu dikembangkan dan dipantau oleh komunitas. Bazaar dapat digunakan di Windows, Ubuntu, Debian, Red Hat, SUSE, OS X, FreeBSD, Solaris, Gentoo, dan lainnya.Bazaar memiliki versi GUI yang dapat memudahkan pengguna, Anda dapat bekerja secara offline dengan menggunakan Bazaar, mempunyai sistem any workflow, gatekeeper workflow, dan centralized workflow, mempunyai fiturrename tracking dan smart merging, dan kecepatan dan efisiensi penyimpanan yang sangat tinggi. Saat ini Bazaar berada di versi 2.6.0

contoh penggunaan Bazaar (dikutip dari Dokumentasi Bazaar):

```
$ bzr init-repo sample
Shared repository with trees (format: 2a)
Location:
  shared repository: sample
$ bzr init sample/trunk
$ cd sample/trunk
Created a repository tree (format: 2a)
Using shared repository: /home/john/sample/
```

kelebihan bazaar

1. Bazaar dapat digunakan di Windows, Ubuntu, Debian, Red Hat, SUSE, OS X, FreeBSD, Solaris, Gentoo, dan lainnya.
2. Bazaar memiliki versi GUI yang dapat memudahkan pengguna, Anda dapat bekerja secara offline dengan menggunakan Bazaar, mempunyai sistem any workflow, gatekeeper workflow, dan centralized workflow, mempunyai fiturrename tracking dan smart merging, dan kecepatan dan efisiensi penyimpanan yang sangat tinggi.

https://www.codepolitan.com/10-version-control-system-yang-harus-kamu-kenal/

Kekurangan bazaar

1. Bazaar bukan sistem benar-benar didistribusikan seperti Git dan Mercurial. Hal ini membatasi kegunaan Bazaar di dunia Open Source hidup cepat di mana forks terjadi dan batas-batas organisasi http://zakalwe.fi/~shd/articles/why_not_bazaar.html

2. Subversion (SVN)

Subversion (sumber: https://subversion.apache.org/)

SVN adalah free VCS yang didesain mirip dengan CVS dan lebih sederhana. SVN mendukung atomic commits dari sebuah file dan melakukan versioningterhadap direktori, symbolic links, dan meta-data. Selain itu mendukungversioning terhadap penamaan, penyalinan, dan penghapusan suatu file atau direktori.SVN merupakan bagian dari Apache Software Foundation. SVN bersifat open source yang didirikan pada tahun 2000 oleh CollabNet Inc. SVN dinikmati luas oleh kalangan komunitas ataupun enterprise. SVN tersedia untuk Linux, OS X, FreeBSD, dan Windows. Saat ini SVN berada di versi 1.8.9

contoh penggunaan SVN (dikutip dari SVN Book):

```
$$ svn checkout http://svn.example.com/svn/repo/trunk
A    trunk/README
A    trunk/INSTALL
A    trunk/src/main.c
A    trunk/src/header.h
…
Checked out revision 8810.
$
```

Kelebihan

1. Mencatat perubahan code dan pembuat perubahan 2. Menyediakan fungsi undo untuk mengembalikan keadaan code ke titik tertentu 3. Melihat riwayat perubahan code, dari pertama dibuat hingga keadaan yang sekarang 4. Memungkinkan penulisan code secara paralel tanpa ada kejadian anggota tim menimpa pekerjaan anggota tim yang lain.

Kekurangan

1. Tidak mendukung atomic commit 2. Tidak mendukung penyimpanan file binary 3. Tidak mendukung rename file atau folder 4. Tidak dapat menyimpan perubahan pada file yang sudah didelete 5. Ijin akses tidak dapat diatir per folder

3. Mercurial

Mercurial (sumber: http://mercurial.selenic.com/)

Mercurial atau yang biasa disingkat menjadi Hg. Mercurial adalah VCS yangfree. Mercurial memiliki konsep distributed source control management tool. Mercurial efisien untuk menangani proyek yang memiliki bermacam ukuran dan jenis. Mercurial mendukung berbagai workflow. Dan setiap Anda melakukan kloning sebuah proyek dengan Mercurial, seluruh riwayat proyek akan disalin.Mercurial dapat digunakan di Windows, OS X, Linux, dan varian Unix lainnya. Mercurial memiliki guide yang dapat mempermudah pengguna baru untuk langsung produktif dengan menggunakan Mercurial. Saat ini Mercurial berada di versi 3.1-rc

contoh penggunaan Mercurial (dikutip dari Homepage Mercurial):

```
$ hg clone http://selenic.com/repo/hello
$ cd hello
$ (edit files)
$ hg add (new files)
$ hg commit -m 'My changes'
$ hg push
```

4. CVS

cvs - Concurrent Versions System

CVS (sumber: http://www.nongnu.org/cvs/)

CVS adalah free VCS yang digunakan oleh mayoritas proyek free software. Saat ini CVS perlahan mulai tergantikan oleh sistem VCS lain yang lebih baru. CVS mendukung pengembangan konkuren oleh banyak pengembang baik secara lokal atau diatas jaringan. CVS sangat kurang mendukung untukatomic commits dan pemindahan / penamaan ulang file.Dengan menggunakan VS Anda dapat merekam riwayat dari file dandokumen. CVS mendukung unreserved checkouts yaitu lebih dari satu pengembang dapat mengerjakan file yang sama secara bersamaan. CVS Server dapat berjalan di berbagai varian Unix dan CVS Client dapat berjalan di Windows dan varian Unix. Terkadang CVS dapat melakukan server modedi sisi client. Saat ini CVS berada di versi 1.11.23

contoh penggunaan CVS (dikutip dari Dokumentasi CVS):

```
$ mkdir -p foo/bar
$ cp ~/myfile foo/bar/myfile
$ cvs add foo foo/bar
$ cvs add foo/bar/myfile
```

5. RCS

GNU RCS

GNU RCS (sumber: http://www.gnu.org/software/rcs/rcs.html)

Revision Control System atau RCS mampu mengelola revisi dari banyak file. RCS mengotomasi penyimpanan, pengambilan, pencatatan, pemeriksaan, dan penggabungan dari sebuah revisi. RCS berguna untuk teks yang sering direvisi seperti source code, program, dokumentasi, grafik, jurnal, dan surat.RCS pertama kali dikembangkan oleh Walter F. Tichy di Purdue University pada awal 1980. RCS didesain dengan lebih unggul dibandingkan pendahulunya yaitu Source Code Control System (SCCS). Peningkatan RCS dibandingkan SCCS diantaranya adalah antar muka yang lebih mudah dan pengambilan yang lebih cepat dari penyimpanan. RCS menggunakan GNU Diffutils untuk melihat perbedaan diantara dua versi yang berbeda. Saat ini RCS berada di versi 5.9.2

contoh penggunaan RCS (dikutip dari Dokumentasi RCS):

```
$ whoami
ttn

$ co -l -f z
RCS/z,v  -->  z
revision 1.1 (locked)
done

$ co -S -l -f z
RCS/z,v  -->  z
co: RCS/z,v: Revision 1.1 is already locked by ttn.
```

6. Perforce

Perforce (sumber: http://www.perforce.com/)

VCS yang satu ini bersifat proprietary dan berbayar jika Anda ingin menggunakannya untuk lebih dari 20 pengembang. Jika menggunakannya untuk 20 pengembang Anda akan mendapatkan Perforce secara free.Perforce memiliki keunggulan dengan sebuah sistem yang dinamakanContinous Delivery. Dengan menggunakan disiplin tersebut software dapat dirilis menjadi production kapanpun. Perforce ini dipercaya oleh vendor - vendor besar seperti Netflix, Samsung, Salesforce, dan New York Stock Exchange.Perforce memiliki keunggulan seperti massive scalability, hybrid version control, social coding, large binaries, dan unified security. Perforce dapat digunakan di Linux, varian Unix, OS X, dan Windows. Saat ini Perforce berada di versi 2014.1 untuk versi Free.

contoh penggunaan Perforce (dikutip dari Dokumentasi Perforce):

```
p4 sync //depot/Misc/manuals/...#head
        //depot/Misc/manuals/recommended_configuration.doc                                    added
c:\p4clients\bruno-1492\Misc\manuals\recommended_configuration.doc
        //depot/Misc/manuals/triggers.doc    added    c:\p4clients\bruno-1492\Misc\manuals\triggers.doc
        //depot/Misc/manuals/vendor_branches.doc                                              added
c:\p4clients\bruno-1492\Misc\manuals\vendor_branches.doc
        Sync copied 3 files to workspace (including 3 that were added)
        Sync removed no files from workspace
Sync completed with no warnings or errors.


p4 edit //bruno-1492/Misc/manuals/recommended_configuration.doc
        //depot/Misc/manuals/recommended_configuration.doc#1 - opened for edit
Opened 1 files for edit


p4 revert //depot/Misc/manuals/recommended_configuration.doc
revert complete. 1 file affected.
```

7. ClearCase

IBM Clear Case (sumber: http://www-03.ibm.com/software/products/en/clearcase)

VCS yang ini juga merupakan VCS proprietary dan berbayar. Harga lisensi untuk penggunanya dalah $ 5.500. Mantap kan harganya?. IBM Rational ClearCase ini mendukung untuk version control, workspace management,parallel development support, dan auditing.Tentunya ClearCase ini dapat diintegrasikan dengan produk IBM lainnya, misal WebSphere. Selain itu ClearCase memiliki effective IP security yang menjamin source code yang ditulis akan aman dari kontrol yang tidak diizinkan.

contoh penggunaan ClearCase (dikutip dariDokumentasi ClearCase):

```
cleartool checkout –nc file1.txt
cleartool: Error: Unable to perform operation "checkout" in replica
"lexington" of VOB "/vobs/dev".
cleartool: Error: Master replica of branch "/main" is "london".
cleartool: Error: Unable to check out "file1.txt".

cmd-context: deliver –complete
Resume deliver
FROM: stream "chris_webo_dev"
TO: stream "integration"
Using integration view: "webo_integ".
Do you wish to continue with this deliver operation? [no] yes
Are you sure you want to complete this deliver operation? [no] yes
Deliver has completed
FROM: stream "chris_webo_dev"
TO: stream "integration"
Using integration view: "webo_integ".
```

8. GNU Arch

GNU Arch (sumber: http://www.gnu.org/software/gnu-arch/)

GNU Arch adalah salah satu decentralized version control system yang paling awal. Hanya saja saat ini Arch mulai kurang aktif dikembangkan. GNU Arch murah dan mudah untuk dikelola, dan tidak perlu memberikan permisi untuk setiap peserta proyek. Mempunyai fitur branching dan merging yang siap memudahkan tim yang mengembangkan free software.Menurut halaman resminya GNU Arch ini digunakan untuk membantu pengembangan free software. Disarankan bagi yang menggunakan CVS atau sistem lainnya untuk bermigrasi ke GNU Arch. GNU Arch dibuat dan dikelola oleh Tom Lord. Saat ini GNU Arch berada di versi 1.3.5

contoh penggunaan GNU Arch (dikutip dari Halaman GNU Arch):

```
cd your_work_directory
tla register-archive http://www.atai.org/archarchives/lord@emf.net--2005-MIRROR/
tla register-archive http://www.atai.org/archarchives/atai@atai.org--public/
tla get tla--atai-dists--1.3.4 tlasrc
cd tlasrc
tla build-config config
```

9. GNU CSSC

GNU CSSC (sumber: http://www.gnu.org/software/cssc/)

CSSC merupakan proyek dari GNU Project yang menggantikan SCCS. SCCS adalah VCS yang bersifat proprietary yang tersedia untuk versi komersial dari Unix. GNU CSSC ini dikembangkan sebagai versi free untuk Unix. Dengan menggunakan CSSC, sebuah proyek dapat dikontrol menggunakan sistem lain seperti Git atau SVN. Saat ini GNU CSSC berada di versi 1.3.0

contoh penggunaan GNU CSSC (dikutip dari Dokumentasi GNU CSSC):

```
prs s.myfile.c
prs SCCS
prs -e -d:P: s.main.c | sort -u
prs -l -c`date +%y%m%d --date "last week"` SCCS
```

10. Git

Git (sumber: http://git-scm.com/)

Git merupakan VCS yang dikembangkan oleh Linux Torvald ketika mengembangkan Linux (kernel). Git merupakan decentralized version control system. Git mempunyai keunggulan seperti repository syncing, bekerja secara offline, cheap local branching, staging area yang nyaman, mampu menangani proyek besar seperti Kernel Linux secara efektif dalam hal kecepatan dan ukuran data, mendukung non-linear development, danmultiple workflow. Selain itu Git digunakan di berbagai layanan VCS seperti Github, Bitbucket, Assembla, dan Gitorious

contoh penggunaan Git (dikutip dari Dokumentasi Git):

```
$ git init
$ git add *.c
$ git add README
$ git commit -m 'initial project version'
$ git push origin masterer
```

GitHub adalah tempat upload project - project opensource dimana project tersebut bisa dikembangkan oleh programmer lain, hal ini baik sekali bila kita memilki project dengan sebuah team bisa dikatakan juga sosial medianya programmer hehehehe :D . Kelebihan github adalah ketika kita membuat sebuah repository project kemudian menguploadnya lalu kita membuat perubahan ata penambahan pada project itu dan di upload ulang ke repository itu maka project yang lama masih di simpan dan tidak akan hilang, jadi kita bisa membuat versi - versi dari project yang kita buat dan hal ini sangat memudahkan kita bila kita ingin ke versi sebelumnya.

Oke langsung ke permasalahan yaitu bagaimana cara upload file ke repository GitHub yang kita miliki di windows (saya pakai windows7 akan lebih mudah lagi kalau memakai linux).

Perangkat yang di gunakan

1. Akun GitHub, Untuk upload project ke GitHub tentunya kita harus memilki akaun GitHub, caranya langsung buat akun di websitenya https://github.com kemudian daftar.
2. Sofware Git, Untuk sofware ini anda bisa download secara gratis di http://git-scm.com/ , kalo sudah di download silahkan anda install.

Setelah kita mempunya akun GitHub dan mengistall software git maka kita bisa langsung meng upload project kita.

Buatlah repository di GitHub dengan mengklik icon repo "Create new repository".

Kemudian beri nama repository nya lalu beri deskripsi untuk repository yang kita buat jika perlu, kemudian setting public/private, kalau public berarti bisa di akses oleh semua orang, kemudian centang "initialize this repository with a README" dan tambahkan kategori repository jila perlu.

kemudian klik "create repository".

Jika repository berhasil dibuat, anda akan di berikan kunci akases berupa HTTP/SSH, ini yang akan kita gunakan untuk remote repository dari software GIT. Misal saya punya kunci HTTP http://github.com/acchoblues/APeK.git

Setelah anda berhasil membiat repository sekarang klik kanan pada folder project yang akan di upload.

Klik kanan pada project klik "Git Bash".

Kemudian akan muncul command prompt / CMD

Jika anda baru pertama kali meggunakan software GIT, sebaiknya konfigurasi username dan email dulu.

Ketik

```
Git config --global user.name "username anda"
Git config --global user.email isi_dengan_email_anda@ymail.com
```

Setelah melakukan konfigurasi username dan email, sekarang kita lakukan inisiasi, ketikan

```
Git init
```

Kemudian kita tambahkan semua file yang ada dalam folder project kita, ketikan

```
Git add *
```

Kemudian kita buat commit project nya, misal disini saya kasih commit “versi 1.0.0” , ketikan

```
Git commit –m "versi 1.0.0"
```

Setelah kita buat commit untuk project nya, sekarang kita remote repository yang kita buat tadi, tentunya kita menggunakan kunci HTTP yang ada pada repository tadi, kalo ane kan tadi contoh nya http://github.com/acchoblues/APeK.git , ketikan

```
Git remote add origin http://github.com/acchoblues/APeK.git
```

Setelah me-remote repository kita tadi, sekarang kita pull project nya, ketikan

```
Git pull origin master
```

Terakhir kita kirim project kita ke repository kita, ketikan

```
Git push origin master
```

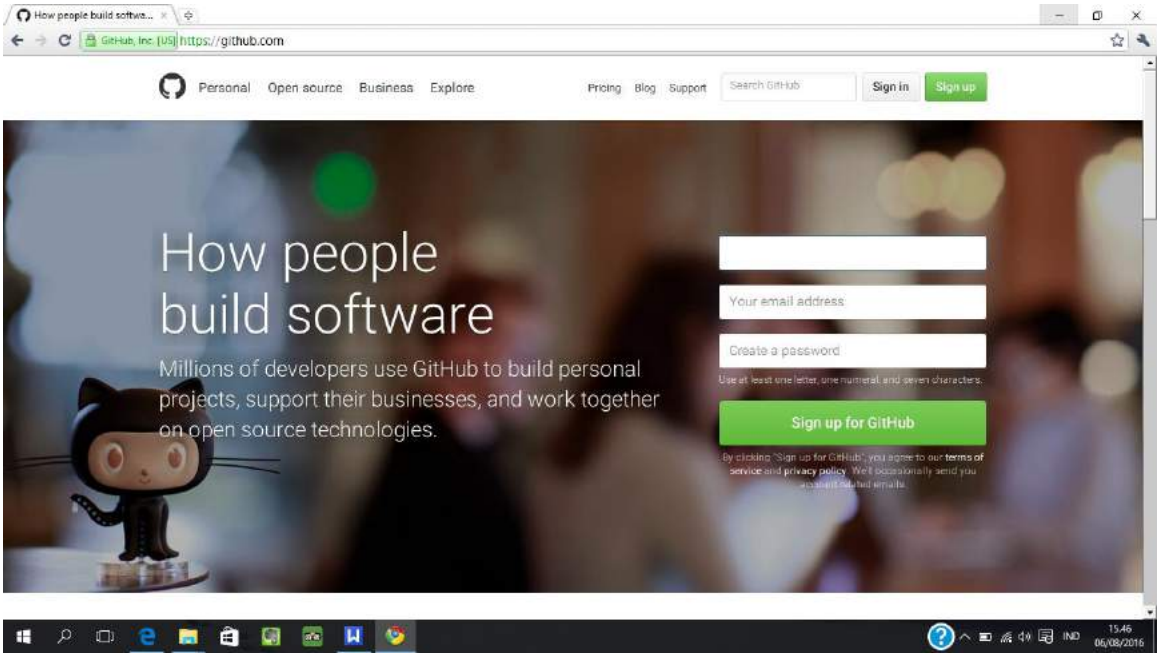
Biasanya ketika kita ketikan perintah push ini, kita akan diminta username dan password kita dan perlu di perhatikan untuk password nya biasa nya ketika kita mengetikan password maka pada command prompt nya tidak ditampilkan karakter apapun.

So jangan khawatir tunggu sampai selesai di upload projectnya.

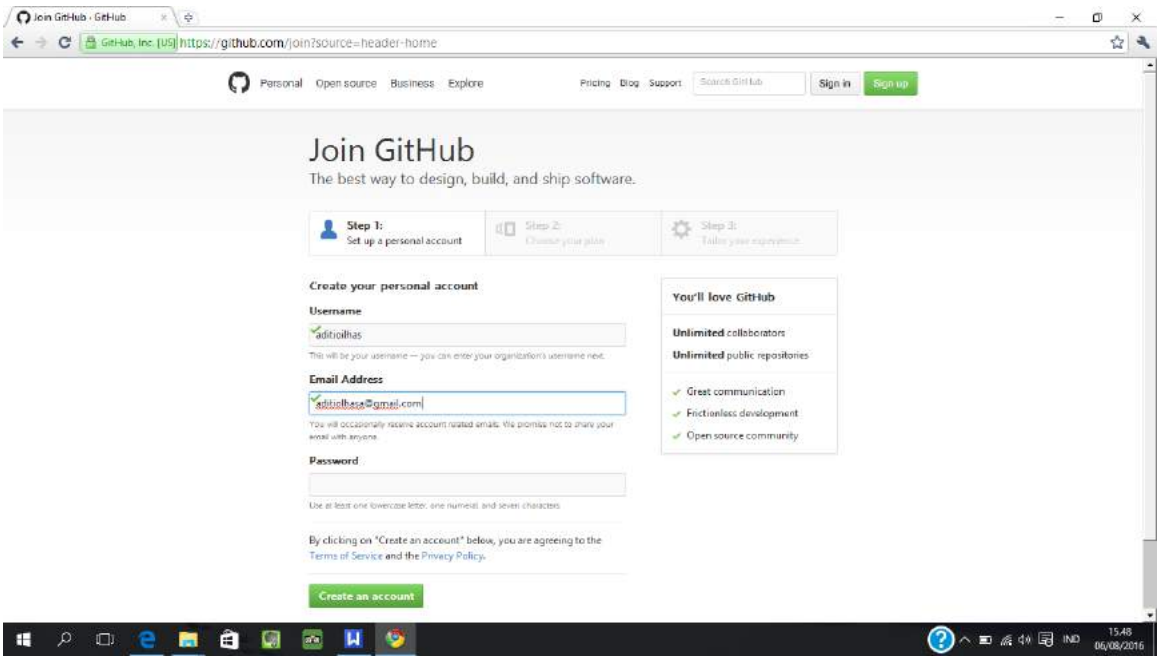
kalau sudah selesai proses uploadnya silahkan refresh repository anda, insya allah jika sesuai dengan instruksi di atas maka project anda sudah terupload disana.

A. Daftar GitHub

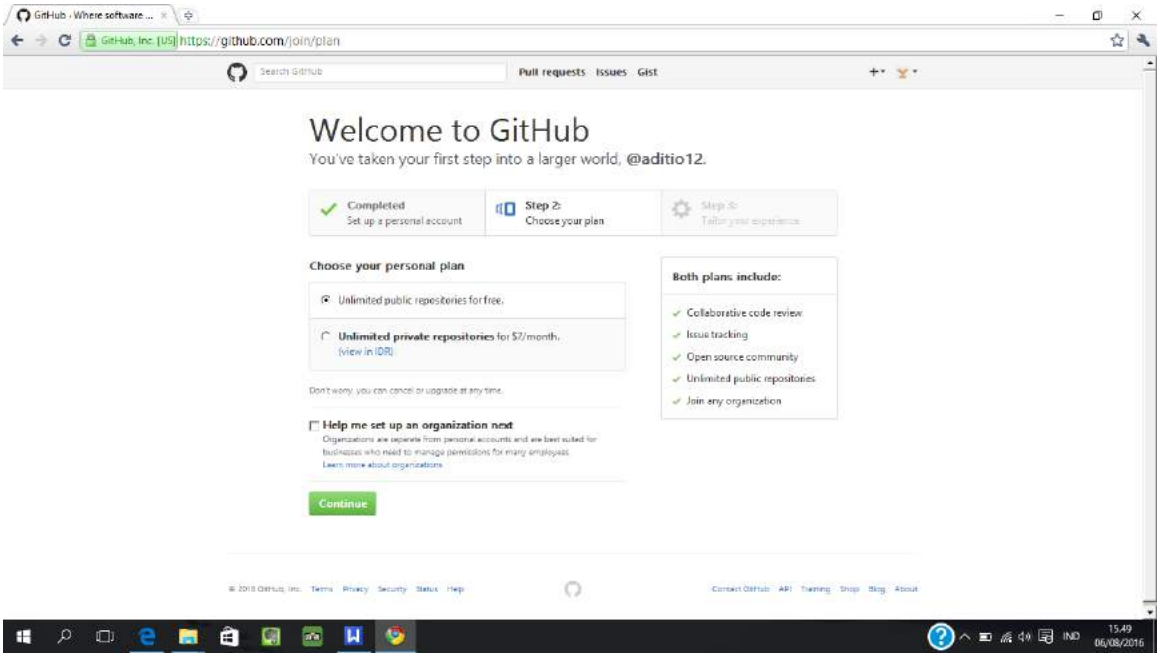
1. Cara membuat / daftar GitHub tekan Sign Up



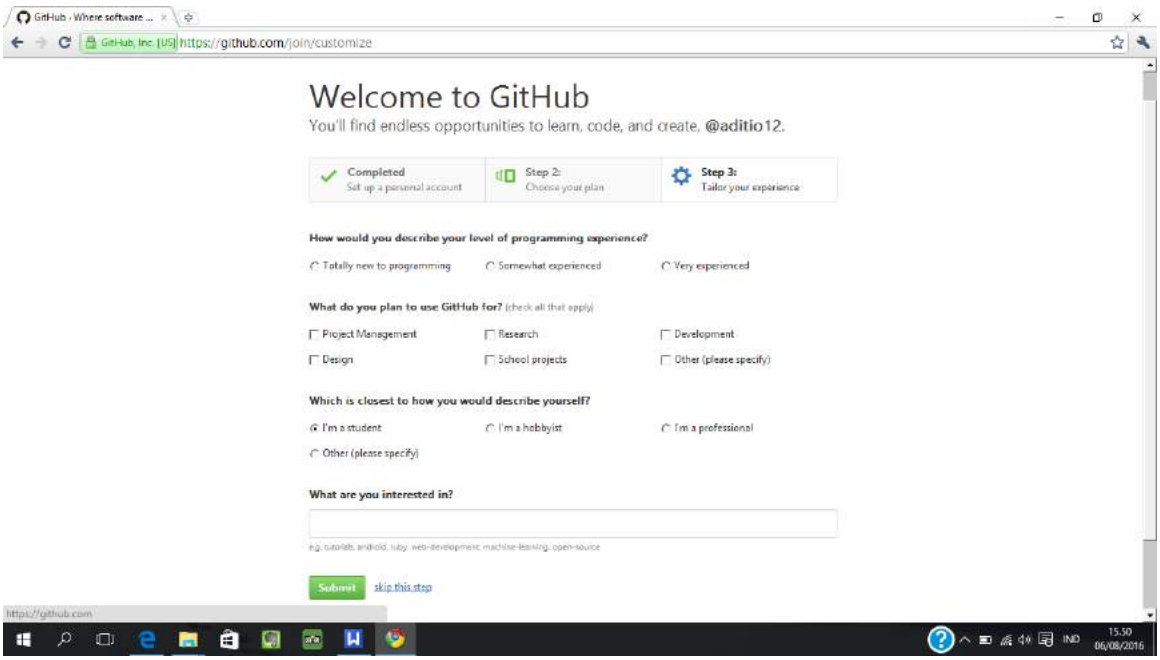
2. Lalu isi formulir step 1



3. Lalu isi formulir step 2



4. Lalu isi formulir step 3 lalu tekan skip this step

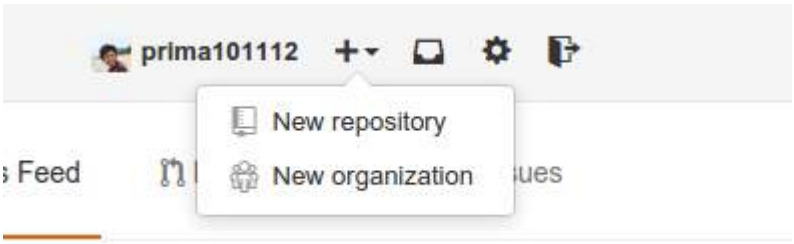


B, Membuat Repository di github.

untuk membuat repo atau repository di github sangat mudah. berikut cara nya.

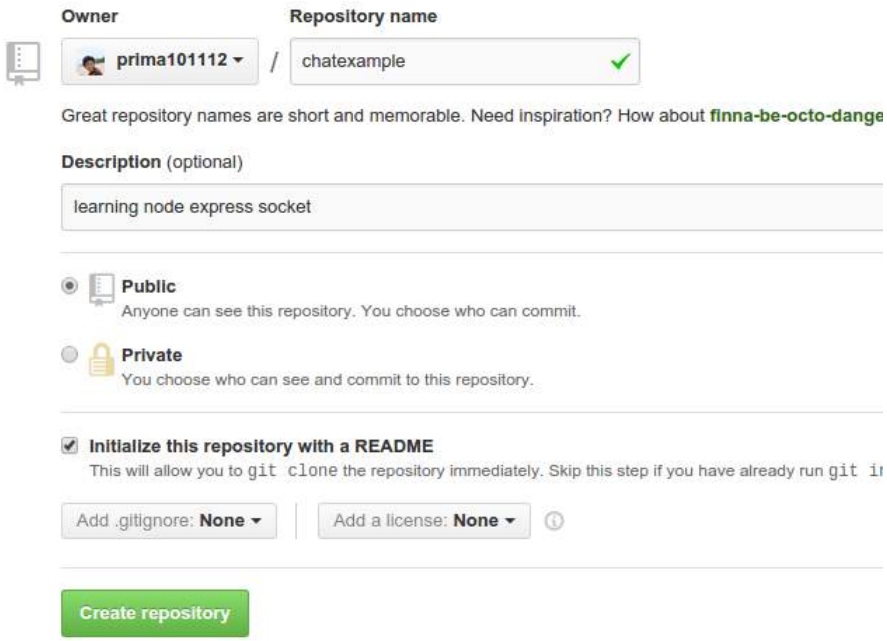
Kamu harus punya akun github atau kalau belum anda bisa mendaftar disini asal project yang kamu buat opensource github gak akan narik biaya.

langsung kamu masuk ke dasboard dan klik new repository.



membuat repository di github

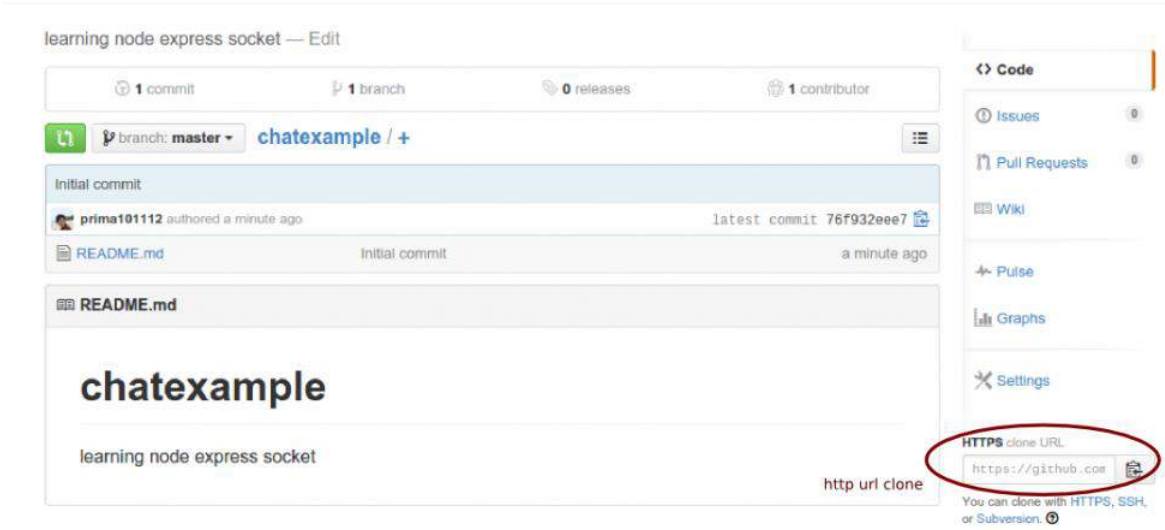
setelah itu masukan detail repo kamu.



membuat repo di github

disana saya centang initialize this reo with readmy biar ntar waktu clone kita gak usah bikin file readme sendiri kita tinggal edit.
mempercepat pekerjaan dan dokumentasi.

Jika sudah seperti ini berarti anda sudah berhasil membuat repo di github tinggal kita clone repo kita kedalam folder local di komputer kita.



membuat repository di github

yang saya lingkari merahj adalah url https untuk clone repo

Untuk Clone Repo

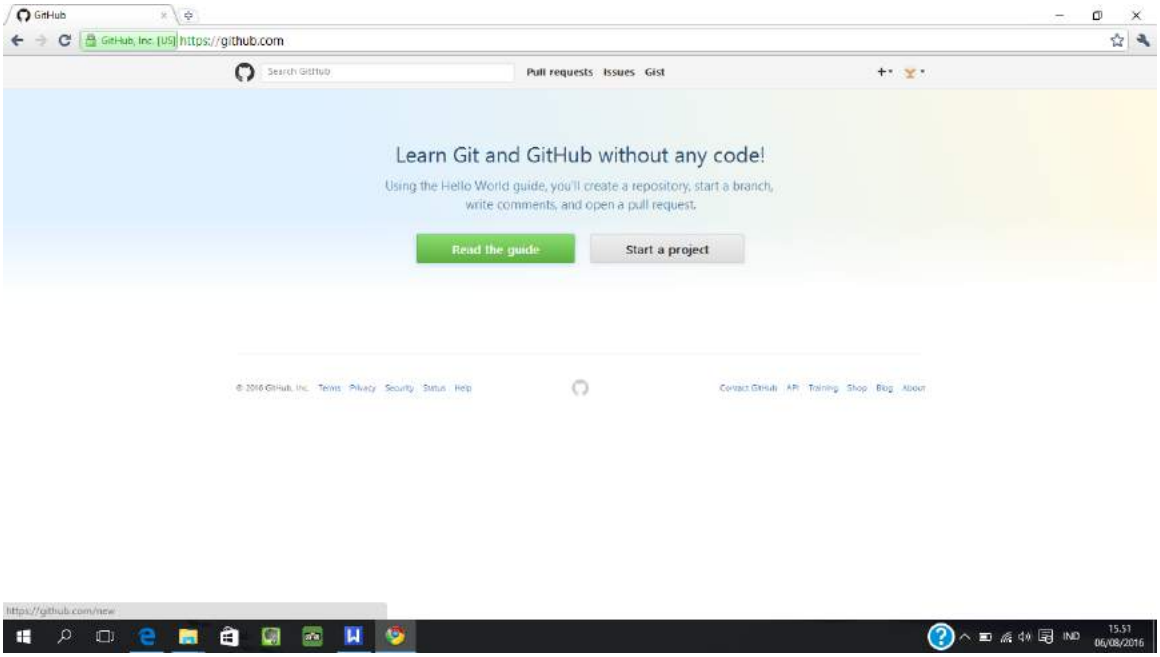
$ git clone urlrepomu

Contoh untuk repo yang barusan saya buat

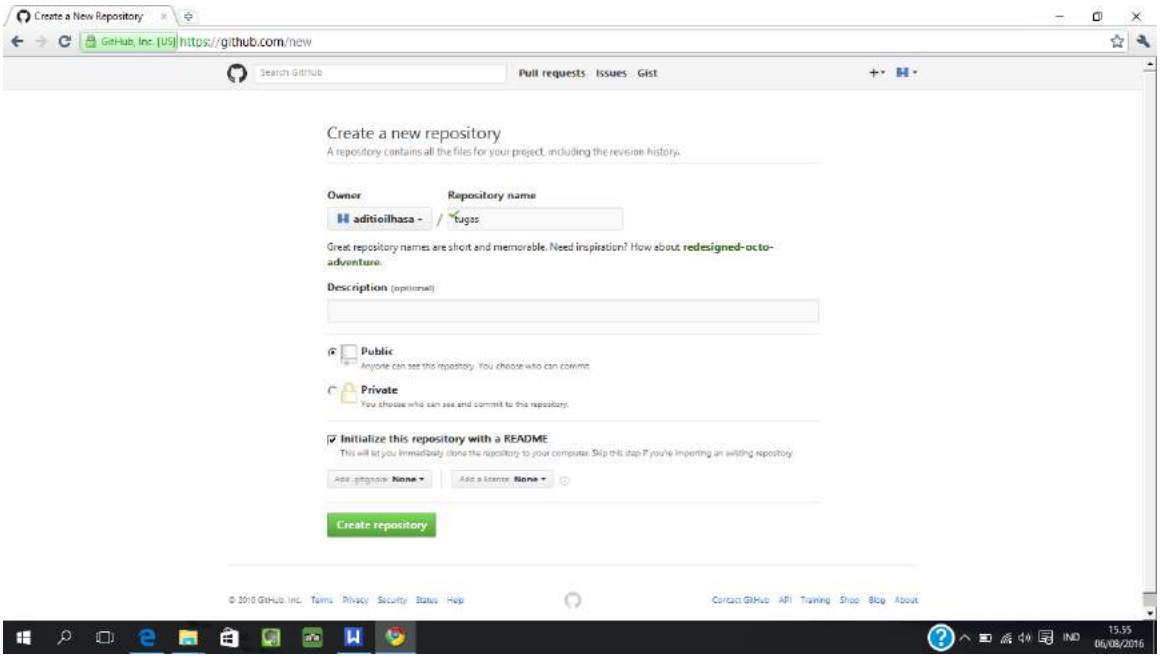
$ git clone https://github.com/prima101112/chatexample.git

C. Cara membuat project baru

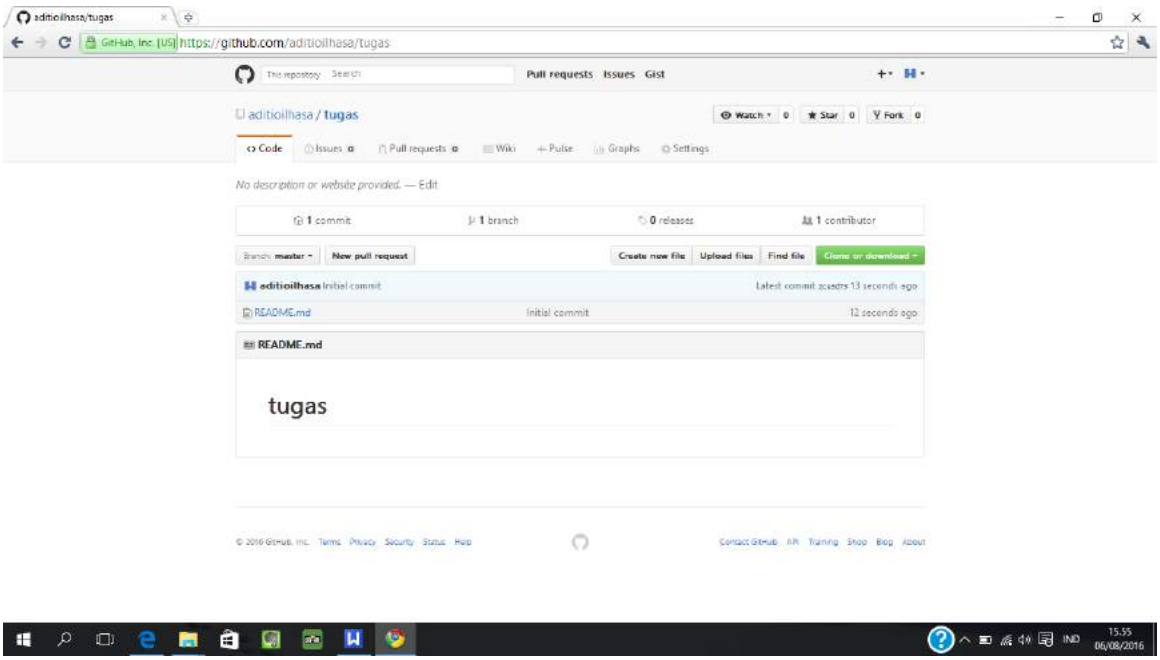
1. klik start a project



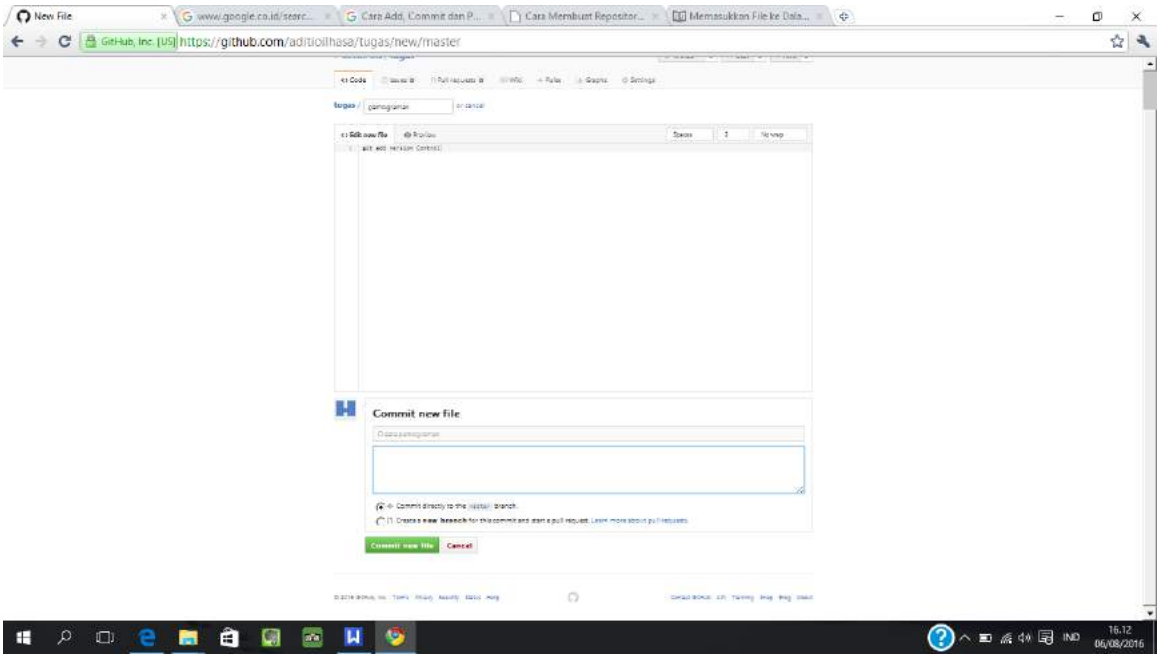
2. Lalu isi repository , pilih public lalu tekan create repository



3. Lalu seperti ini tampilanya

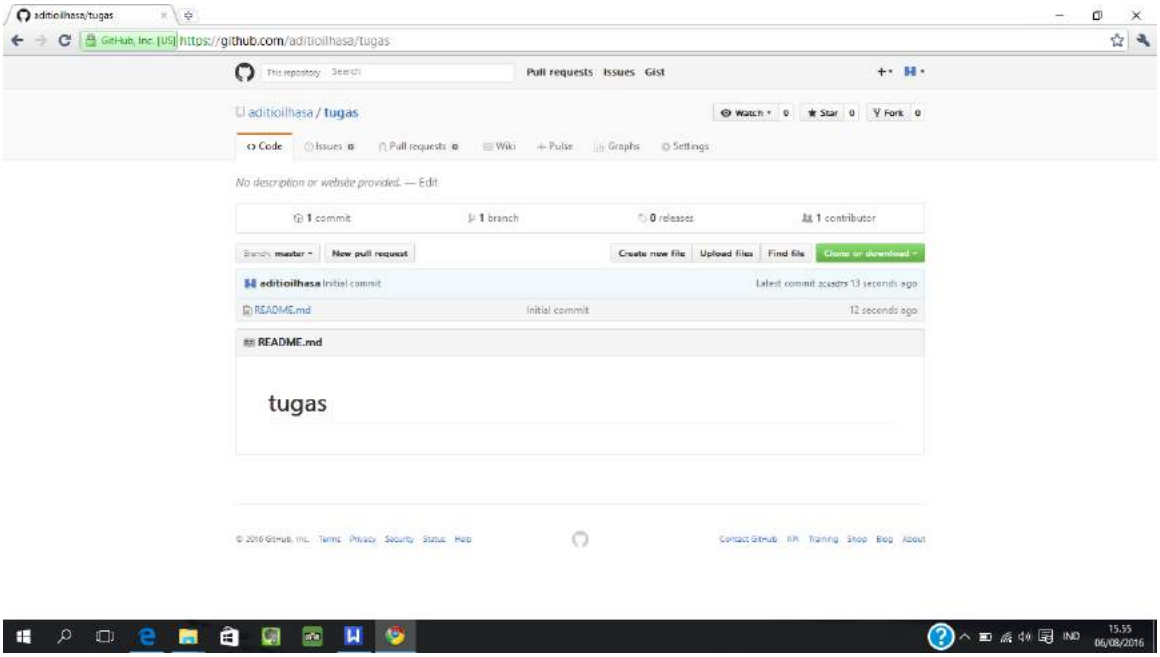


D. Commit new file

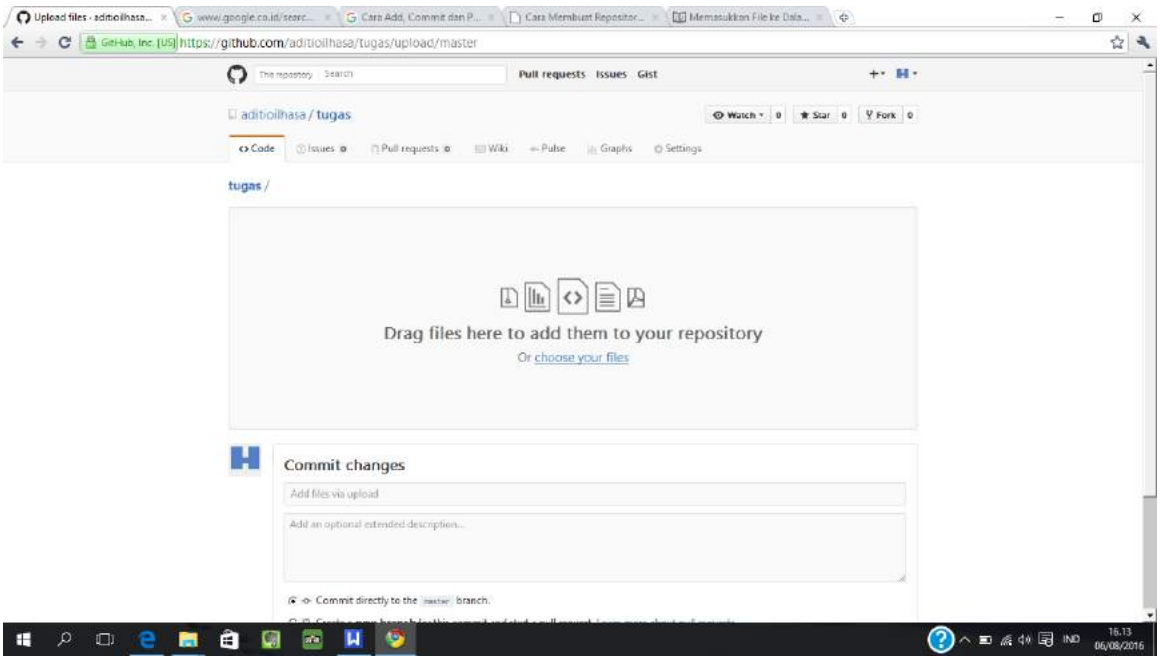


E. Cara Uplod File

1. Klik Uplod files

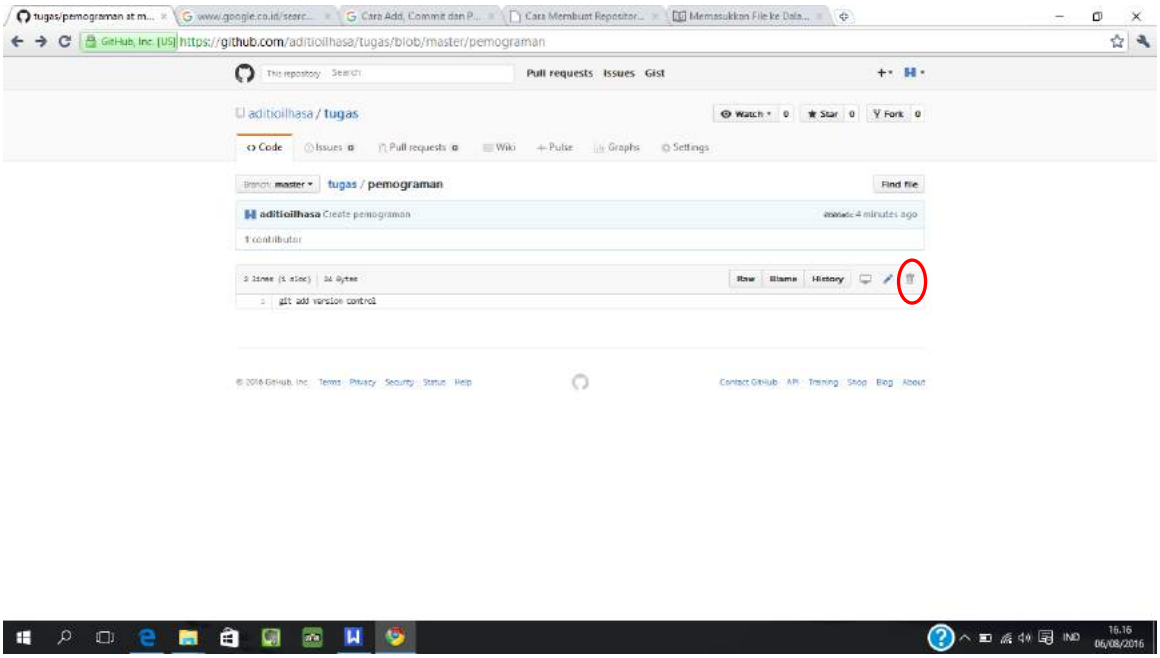


2. Lalu pilih choose your files dan pilih filenya

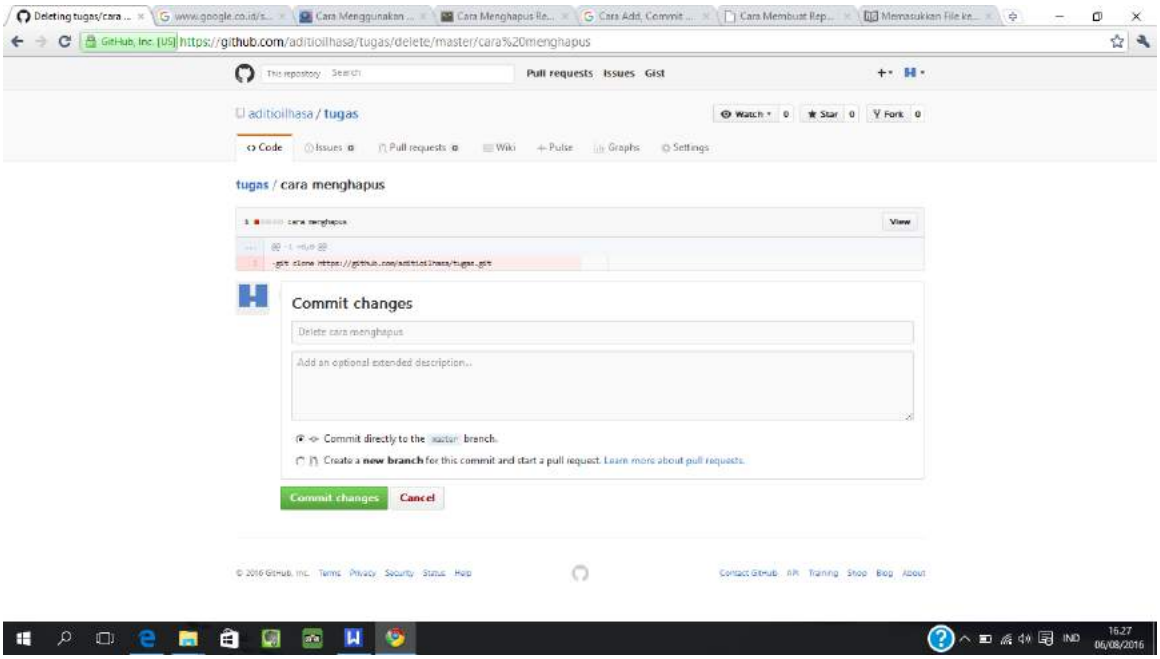


F. Cara menghapus file

1. Ketik icon delete yang dilingkari

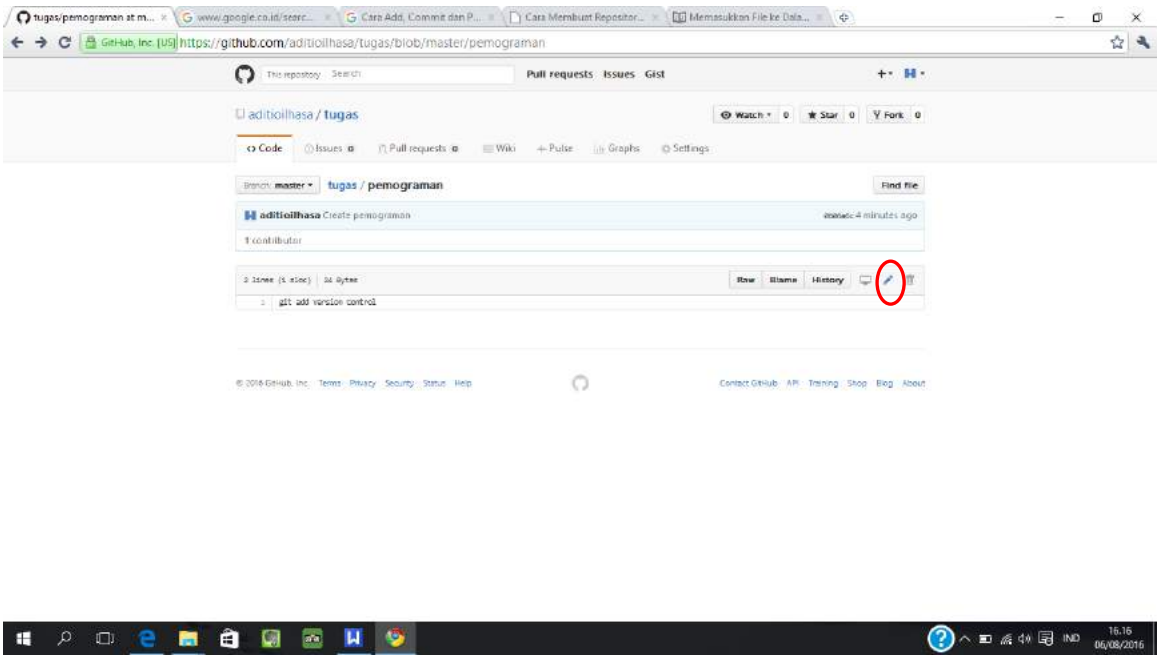


2. Lalu klik commit changes



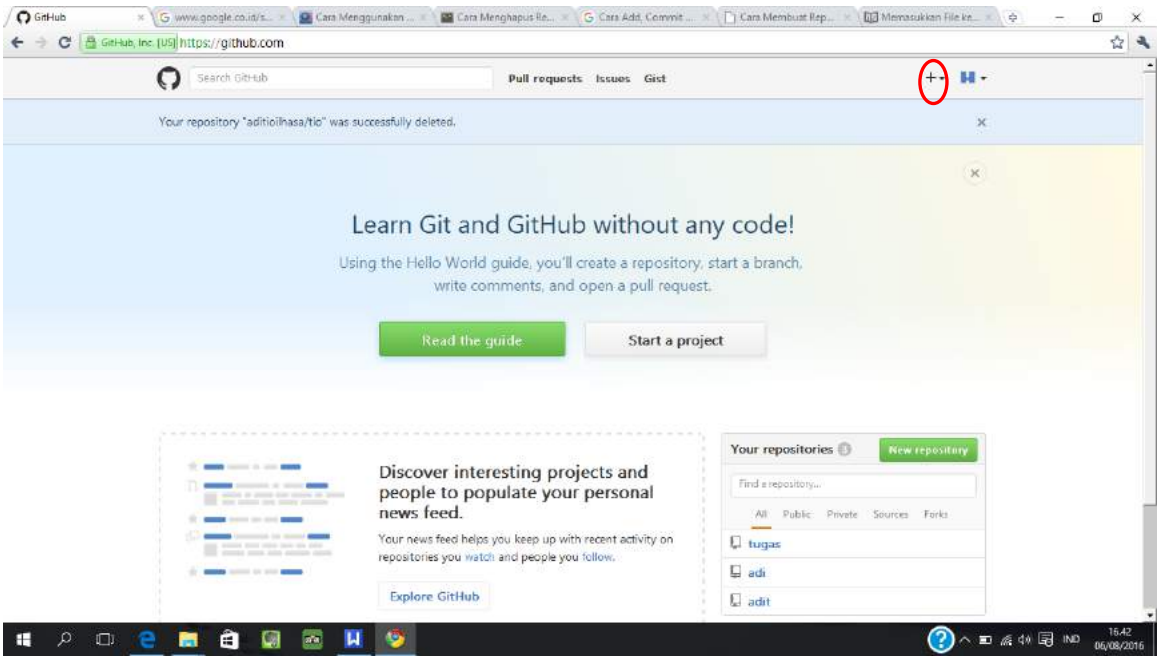
G. Cara mengupdate file

1. Ketik icon edit yang dilingkari

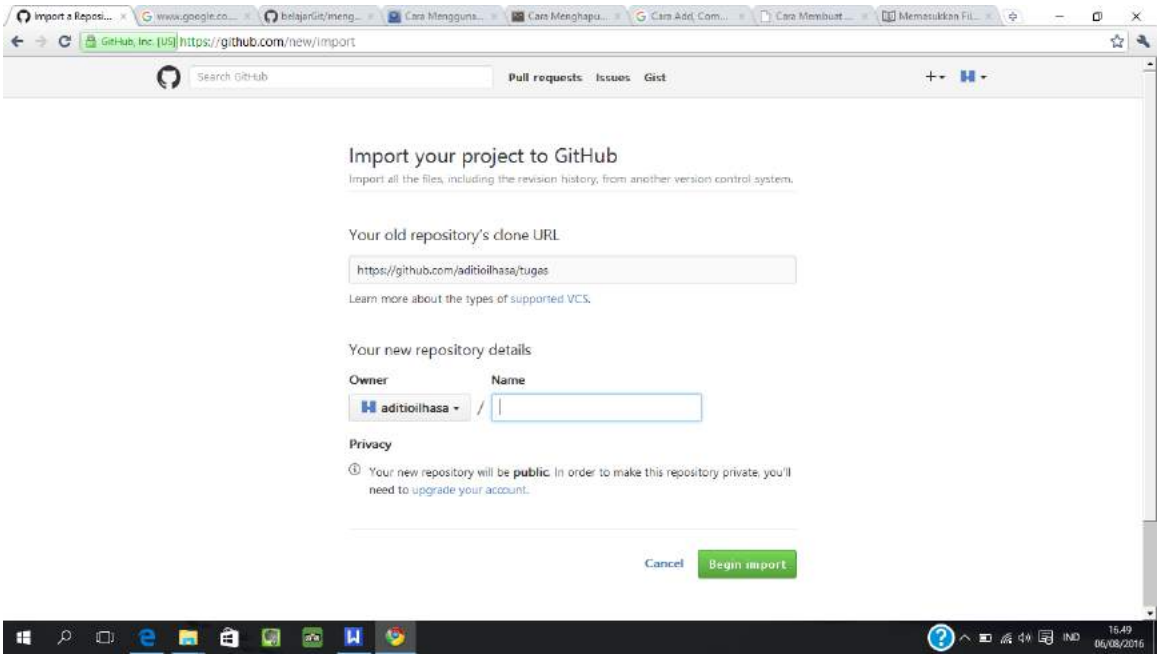


H. Cara cek URL

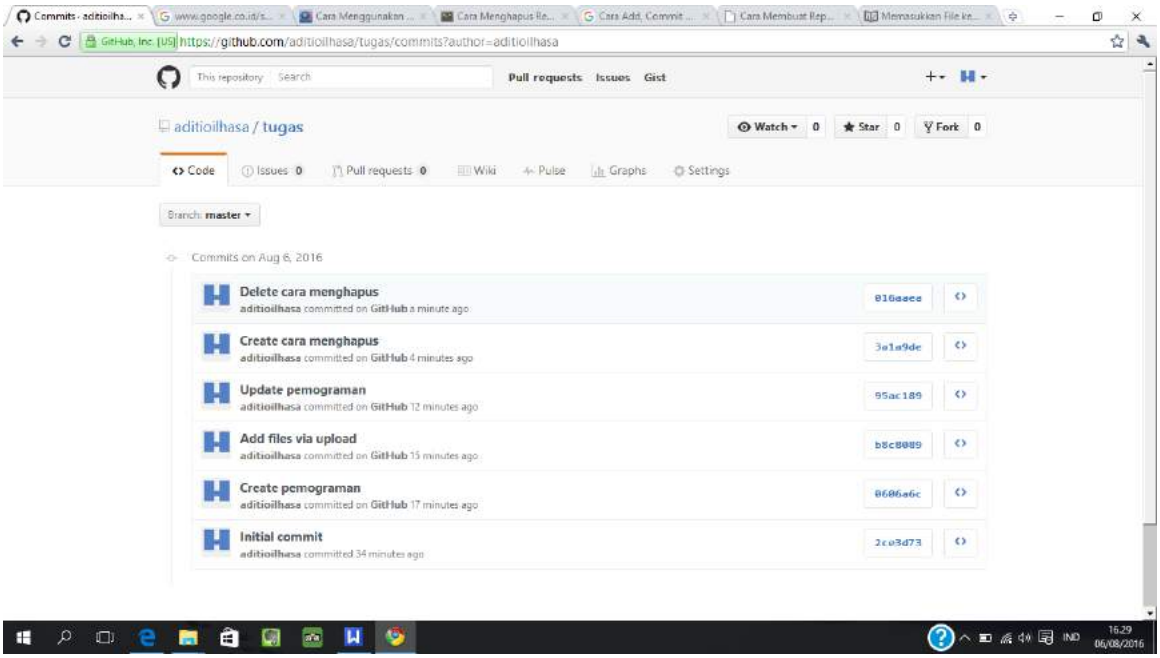
1. Ketik tombol yg dilingkari lalu pilih import repository

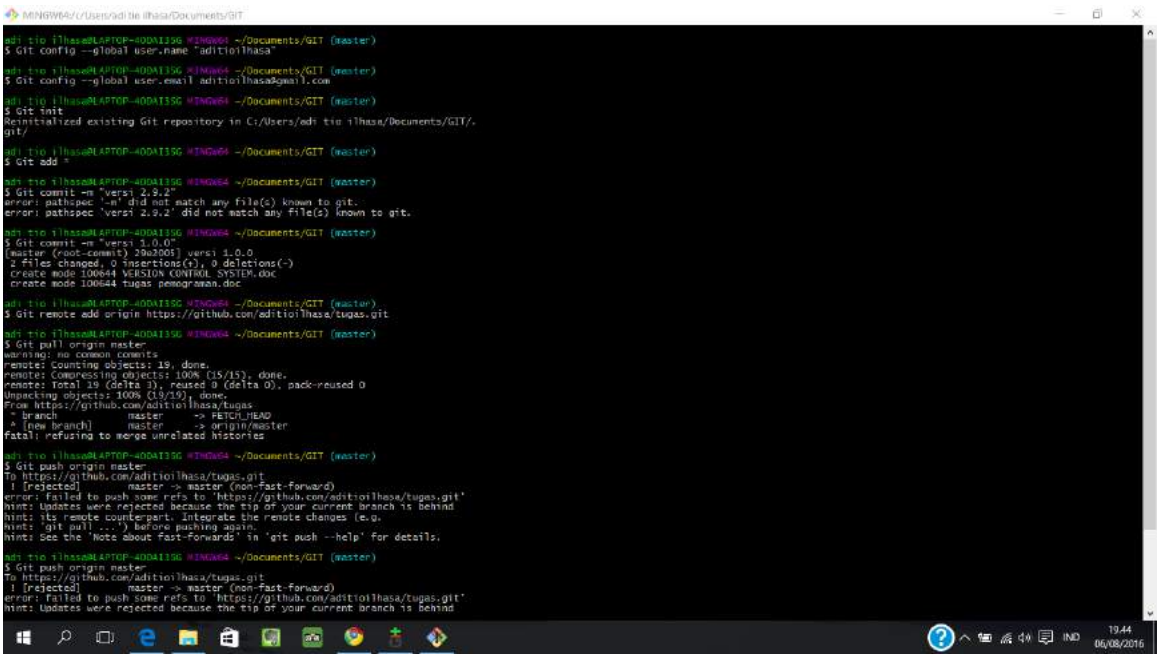


2. Masukan url dan name lalu begin import



History create, add upload, update, delete, commit dan cek url





```
adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git config --global user.name "aditioilhasa"

adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git config --global user.email aditioilhasa@gmail.com

adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git init
Reinitialized existing Git repository in C:/Users/adi tio ilhasa/Documents/GIT/.
git/

adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git add *

adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git commit -m "versi 2.9.2"
error: pathspec '-m' did not match any file(s) known to git.
error: pathspec 'versi 2.9.2' did not match any file(s) known to git.

adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git commit -m "versi 1.0.0"
[master (root-commit) 29e2005] versi 1.0.0
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 VERSION CONTROL SYSTEM.doc
 create mode 100644 tugas pemograman.doc

adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git remote add origin https://github.com/aditioilhasa/tugas.git

adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git pull origin master
warning: no common commits
remote: Counting objects: 19, done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 19 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (19/19), done.
From https://github.com/aditioilhasa/tugas
 * branch            master     -> FETCH_HEAD
 * [new branch]      master     -> origin/master
fatal: refusing to merge unrelated histories

adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git push origin master
To https://github.com/aditioilhasa/tugas.git
 ! [rejected]        master -> master (non-fast-forward)
error: failed to push some refs to 'https://github.com/aditioilhasa/tugas.git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Integrate the remote changes (e.g.
hint: 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.

adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git push origin master
To https://github.com/aditioilhasa/tugas.git
 ! [rejected]        master -> master (non-fast-forward)
error: failed to push some refs to 'https://github.com/aditioilhasa/tugas.git'
hint: Updates were rejected because the tip of your current branch is behind
```

```
adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git pull origin master
warning: no common commits
remote: Counting objects: 19, done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 19 (delta 3), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (19/19), done.
From https://github.com/aditioilhasa/tugas
 * branch            master     -> FETCH_HEAD
 * [new branch]      master     -> origin/master
fatal: refusing to merge unrelated histories

adi tio ilhasa@LAPTOP-4ODAI3SG MINGW64 ~/Documents/GIT (master)
$ Git push origin master
To https://github.com/aditioilhasa/tugas.git
 ! [rejected]        master -> master (non-fast-forward)
error: failed to push some refs to 'https://github.com/aditioilhasa/tugas.git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Integrate the remote changes (e.g.
hint: 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```