MODUL 2

GIT
STRING
THREAD

Oleh :

1. Afifah Asmar Sari (5114100154)
2. Ilyas Bintang P. (5114100157)

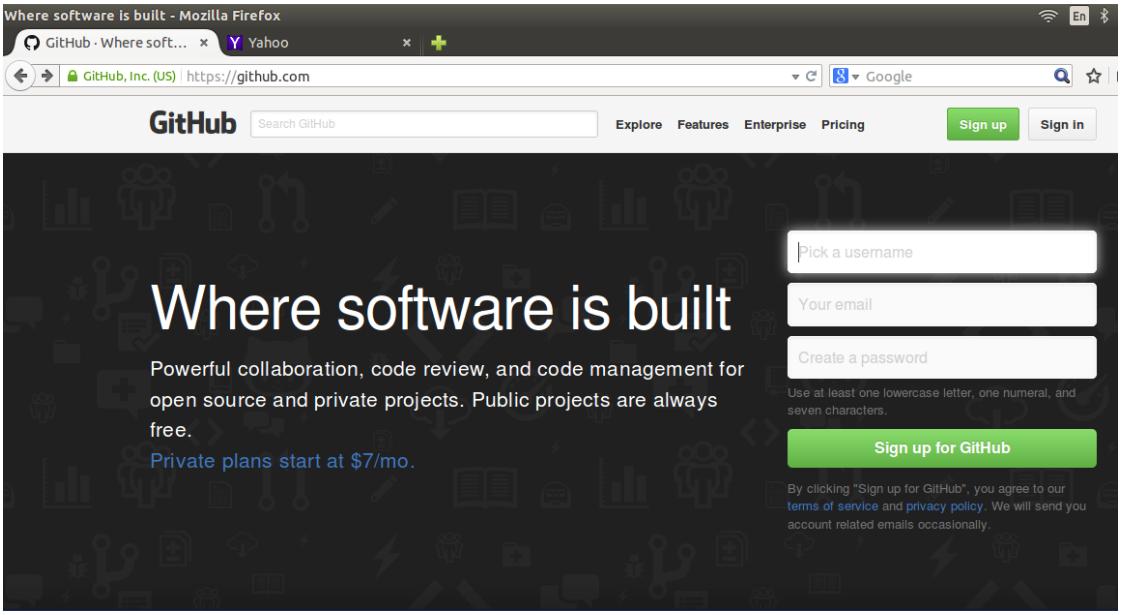
GIT, STRING, THREAD

A. GIT

Sebelum mengenal lebih jauh mengenai git, kita harus tahu github terlebih dahulu. Github merupakan sebuah jejaring sosial bagi para software developer. Github ditargetkan untuk para developer yang bekerja dalam tim dan tidak berada di satu tempat. Alamat dari github adalah github.com. Karena dia disebut sebagai sebuah jejaring sosial maka untuk dapat menggunakannya kita harus memiliki akun terlebih dahulu. Berikut langkah-langkah awal menggunakan github :

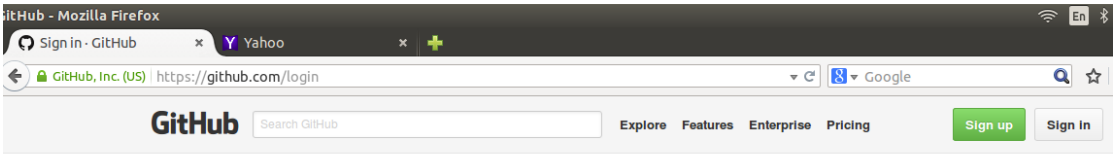
1. Sign-Up

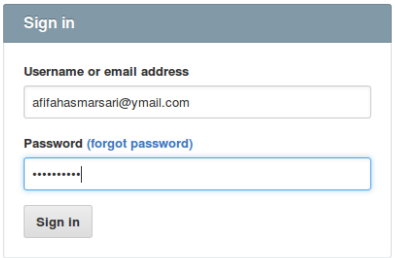
   Kita perlu mendaftar dengan mengisikan data-data tertentu terlebih dahulu jika belum memiliki akun pada github, berikut adalah bentuk tampilan awal ketika mengakses github.com



2. Sign-In

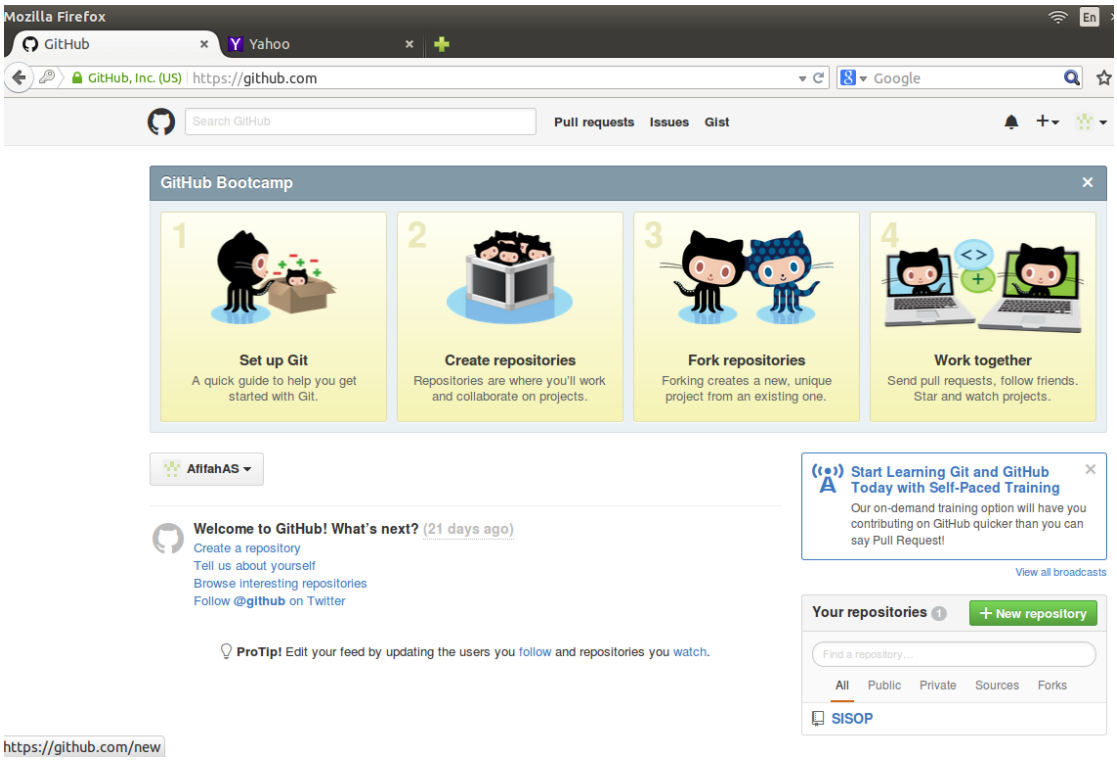
   Ketika kita sudah memiliki akun, maka hal yang dilakukan setiap kali kita ingin masuk pada github hanya dengan sign-in





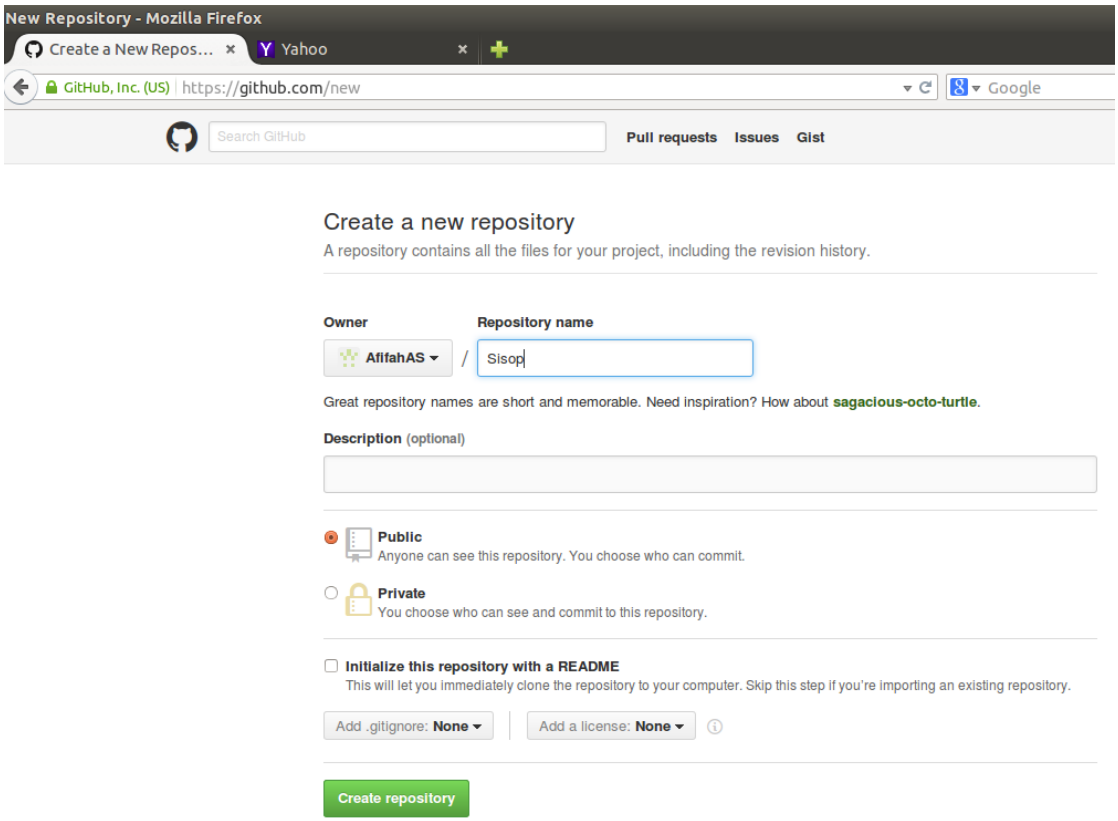
3. Home

Berikut tampilan awal setelah masuk pada akun anda



4. Membuat repository

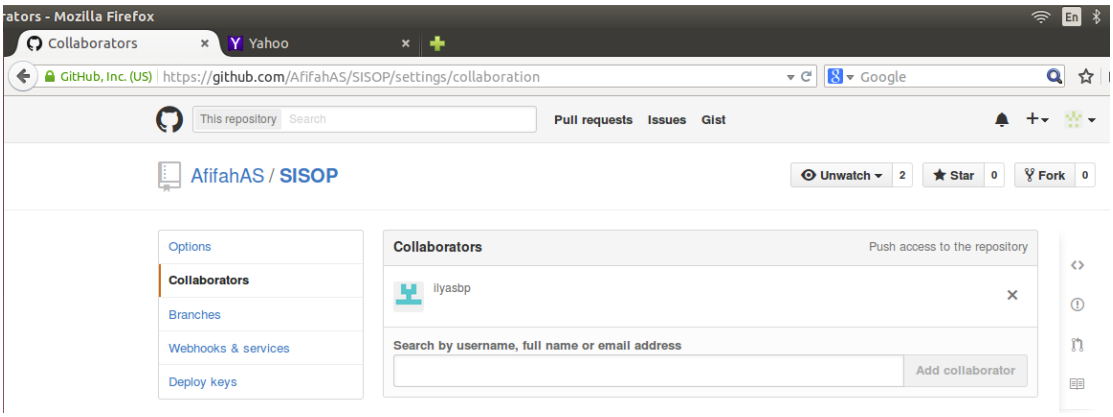
Hal selanjutnya adalah membuat repository dengan memilih menu new repository pada bagian kanan bawah, maka selanjutnya akan muncul seperti gambar dibawah ini



Cantumkan nama repository yang diinginkan, atur hak akses serta tambahkan deskripsi jika diperlukan.

5. Mengatur collaborators

   Karena github ditargetkan untuk developer yang bekerja dalam tim, maka collaborators perlu diatur guna mengikutsertakan user lain dalam project yang tengah dikerjakan. Caranya dengan membuka repository terlebih dahulu, kemudian pilih menu setting, selanjutnya tambahkan user yang dikehendaki. Pastikan user tersebut memiliki akun pada github



Cara untuk sinkronisasi dan mengunduh file pada repository :

1. Setting proxy (jika menggunakan proxy)

   Syntax :

   ```
   export http_proxy=”http://email:password@proxy:port”
   ```

   ```
   export https_proxy=”https://email:password@proxy:port”
   ```

2. Install tool git

   ```
   sudo –E apt-get install git
   ```

3. Fungsi git clone (sinkronisasi komputer dengan repository)

   ```
   git clone alamat_repository
   ```

4. Git pull (mengunduh file keseluruhan pada repository)

   ```
   git pull
   ```

Cara untuk mengupload file pada repository :

1. Memindahkan file pada folder repository yang telah disinkronisasikan sebelumnya
2. Pindah direktori ke folder repository

   ```
   cd nama_direktori
   ```

3. Git add

   ```
   git add nama_file.ekstensi
   ```

4. Git commit

   ```
   git commit –m “comment”
   ```

5. Git push

```
git push
```

B. STRING

Perintah pada string yang dipelajari kali ini adalah cara untuk membagi sebuah string utuh menjadi beberapa sub string. Setiap kali perintah string digunakan, jangan lupa untuk selalu menggunakan library :

```
#include<string.h>
```

Ada dua buah perintah yang dapat digunakan untuk membagi string, yakni sebagai berikut :

- strstr()

  Sebelum memanggil perintah strstr(), kita perlu membuat sebuah variable pointer yang menunjuk tipe data char. Dimana variable ini berfungsi untuk menyimpan hasil dari perintah strstr().

  Inisialisasi :

  ```
  char *variable
  ```

  Perintah ini pada dasarnya digunakan untuk mencari kemungkinan substring pertama dengan pembatas yang telah ditentukan. Pembatas yang dimaksud adalah sebuah karakter yang terdapat pada string dan nantinya digunakan sebagai penanda berakhirnya pencarian dari sub string kemungkinan pertama. Maka output yang dihasilkan nantinya adalah sebuah substring sebelum pembatas.

  Syntax :

  ```
  strstr(string_utuh,pembatas)
  ```

  Selanjutnya, apabila kita ingin menampilkan hasilnya, maka kita hanya perlu menampilkan isi dari variable pointer yang telah dibuat sebelumnya.

- strtok()

  Sama halnya dengan strstr(), sebelum memanggil fungsi strtok(), kita memerlukan sebuah variable pointer yang menunjuk tipe data character guna menyimpan hasil pembagian string. Inisialisasinya sama dengan strstr() yang sebelumnya telah dijelaskan.

  Perbedaan perintah ini dengan perintah strstr() adalah perintah ini akan memindai seluruh kemungkinan substring yang ada berdasarkan pembatas yang telah ditentukan.

  Syntax :

  ```
  strtok(string_utuh,pembatas)
  ```

Berikut adalah contoh penggunaan dari kedua perintah string diatas :

Source Code

```
afifahs@afifahs-Lenovo-G40-70: ~/SistemOperasi/PRAKTIKUM/2
  GNU nano 2.2.6                File: string.c

#include<stdio.h>
#include<string.h>

int main()
{
        char input[50];
        printf("Masukkan string: ");
        scanf("%[^\n]",input);
        printf("%s\n",input);

        char *substr = strstr(input,"o"); //mencari kemungkinan pertama
        printf("Hasil strstr(): %s\n", substr);

        char *token = strtok(input,"o");
        while(token!=NULL){
                printf("Hasil strtok(): %s\n", token);
                token = strtok(NULL,"o");
        }

        return 0;
}
```

Executed Program

```
afifahs@afifahs-Lenovo-G40-70: ~/SistemOperasi/PRAKTIKUM/2
afifahs@afifahs-Lenovo-G40-70:~/SistemOperasi/PRAKTIKUM/2$ gcc -o string string.
c
afifahs@afifahs-Lenovo-G40-70:~/SistemOperasi/PRAKTIKUM/2$ ./string
Masukkan string: Hello World
Hello World
Hasil strstr(): o World
Hasil strtok(): Hell
Hasil strtok():  W
Hasil strtok(): rld
```

C. THREAD

Thread merupakan proses yang sesungguhnya. Di dalam sebuah proses, terdapat thread dan sesungguhnya thread inilah yang dijalankan. Setiap thread akan memiliki fungsi tersenidri yang dijalankan. Fungsi yang dijalankan oleh thread harus memiliki return void* begitu juga parameternya harus void*. Kita memerlukan sebuah library yang harus di definisikan sebelum kita memanggil fungsi pthread.

Library :

```
#include<pthread.h
```

Pertama kali kita perlu membuat thread terlebih dahulu.

Inisialisasi :

```
pthread_t nama_thread
```

Selanjutnya barulah kita memanggil thread.

Syntax :

```
pthread_create(id_thread,atribut,nama_fungsi,???)
```

Pada dasarnya proses berjalannya thread bergantung pada sistem operasi, namun kita dapat membuat sebuah thread menunggu thread yang lain selesai yakni dengan menggunakan perintah pthread_join. Parameter pertama berisi thread yang menunggu, dan parameter kedua berisi indikator kapan thread yang menunggu dijalankan.

Syntax :

```
pthread_join(nama_thread,indikator)
```

Saat melakukan compile jangan lupa menambahkan –lpthread pada akhir perintah.

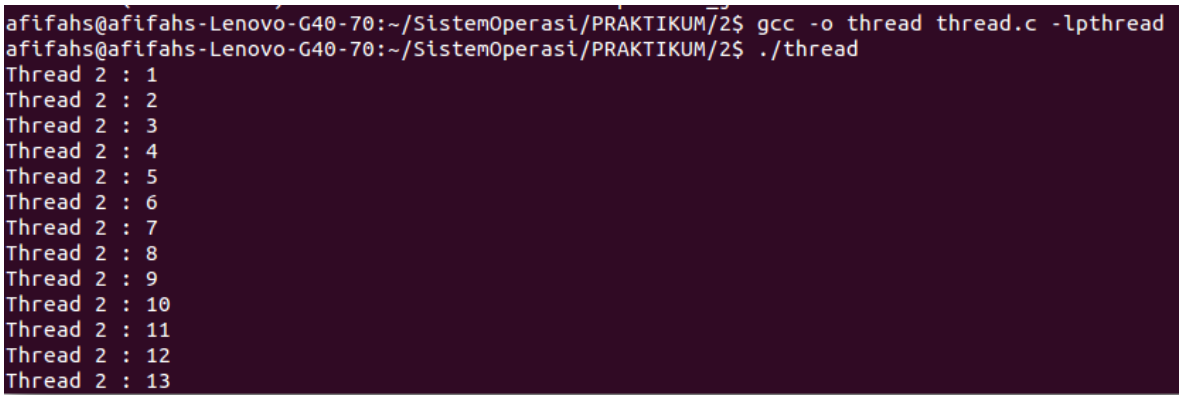
Compile :

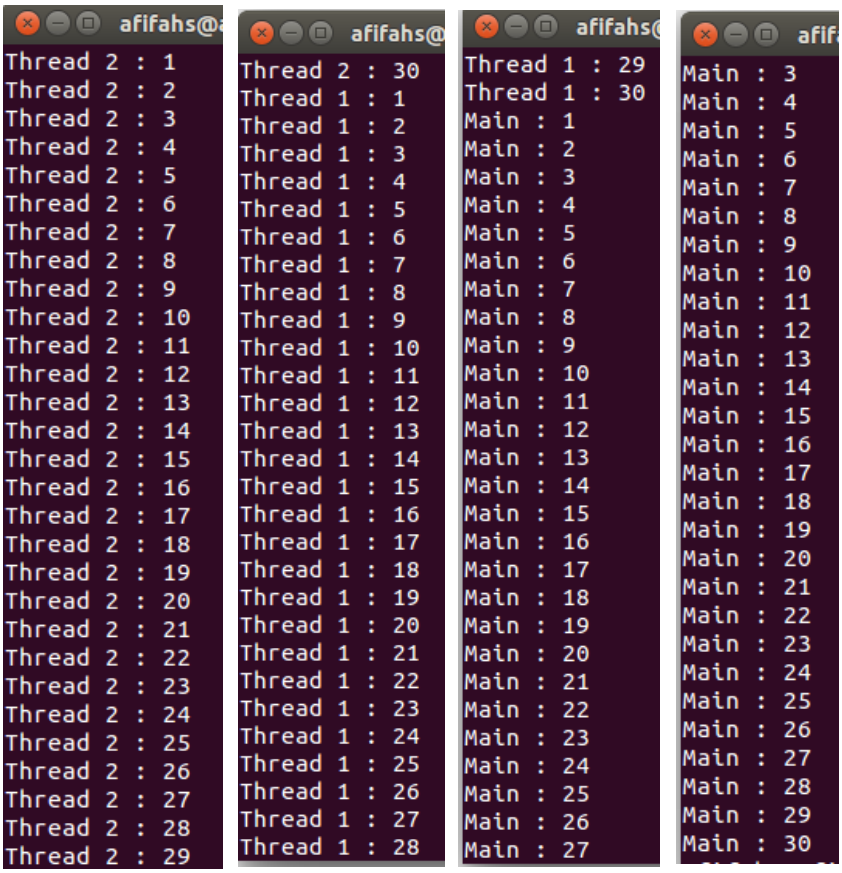
```
gcc –o nama_eksekusi nama_file -lpthread
```

Berikut adalah contoh kasus dimana terdapat 2 thread yang memiliki fungsi masing2 serta sebuah fungsi pada main. Selanjutnya, kita mengatur agar thread 2 berjalan terlebih dahulu, dilanjutkan thread 1, dan yang terkahir berjalan adalah fungsi pada main.

Source code

Executed Program




Thread 2 : 1
Thread 2 : 2
Thread 2 : 3
Thread 2 : 4
Thread 2 : 5
Thread 2 : 6
Thread 2 : 7
Thread 2 : 8
Thread 2 : 9
Thread 2 : 10
Thread 2 : 11
Thread 2 : 12
Thread 2 : 13
Thread 2 : 14
Thread 2 : 15
Thread 2 : 16
Thread 2 : 17
Thread 2 : 18
Thread 2 : 19
Thread 2 : 20
Thread 2 : 21
Thread 2 : 22
Thread 2 : 23
Thread 2 : 24
Thread 2 : 25
Thread 2 : 26
Thread 2 : 27
Thread 2 : 28
Thread 2 : 29
Thread 2 : 30
Thread 1 : 1
Thread 1 : 2
Thread 1 : 3
Thread 1 : 4
Thread 1 : 5
Thread 1 : 6
Thread 1 : 7
Thread 1 : 8
Thread 1 : 9
Thread 1 : 10
Thread 1 : 11
Thread 1 : 12
Thread 1 : 13
Thread 1 : 14
Thread 1 : 15
Thread 1 : 16
Thread 1 : 17
Thread 1 : 18
Thread 1 : 19
Thread 1 : 20
Thread 1 : 21
Thread 1 : 22
Thread 1 : 23
Thread 1 : 24
Thread 1 : 25
Thread 1 : 26
Thread 1 : 27
Thread 1 : 28
Thread 1 : 29
Thread 1 : 30
Main : 1
Main : 2
Main : 3
Main : 4
Main : 5
Main : 6
Main : 7
Main : 8
Main : 9
Main : 10
Main : 11
Main : 12
Main : 13
Main : 14
Main : 15
Main : 16
Main : 17
Main : 18
Main : 19
Main : 20
Main : 21
Main : 22
Main : 23
Main : 24
Main : 25
Main : 26
Main : 27
Main : 3
Main : 4
Main : 5
Main : 6
Main : 7
Main : 8
Main : 9
Main : 10
Main : 11
Main : 12
Main : 13
Main : 14
Main : 15
Main : 16
Main : 17
Main : 18
Main : 19
Main : 20
Main : 21
Main : 22
Main : 23
Main : 24
Main : 25
Main : 26
Main : 27
Main : 28
Main : 29
Main : 30