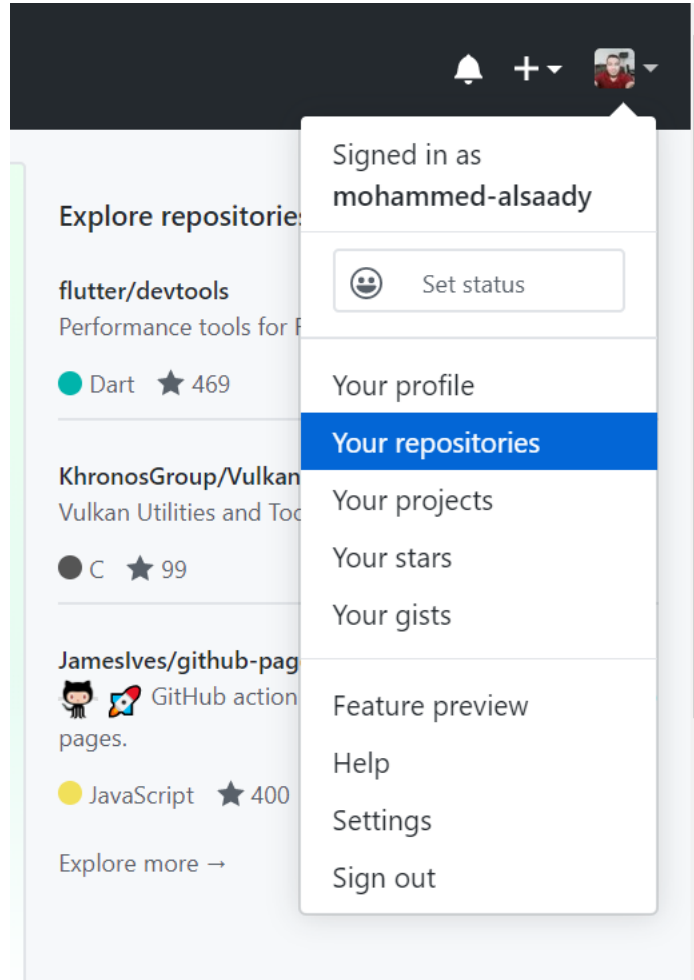


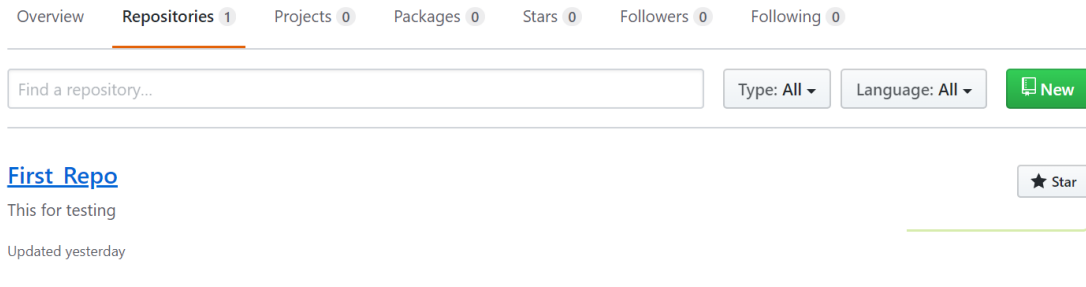
## الفصل الثاني

في الفصل الأول علمنا مستودع (Repository) على Github , الآن كيف نتعامل معها .

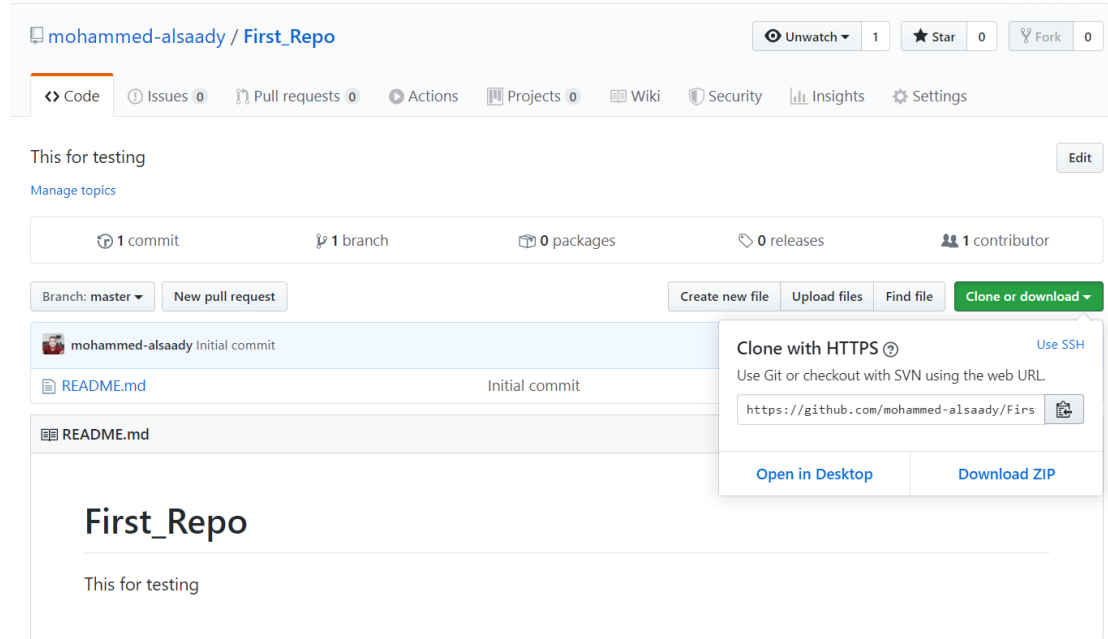
بالبداية سوف نفتح Github , ومن القائمة على اليمين نختار (Your Repositories) . (



حتى نفتح لنا صفحة المستودعات



سوف أختار المستودع ( First\_Repo ) بالنقر عليه , على اليمين سوف تجد ( Clone or Download ) ننقر على السهم الصغير الذي بجوارهما لتظهر النافذة :



لاحظ في الصورة أعلاه يعطيك عدة خيارات أما تنزل المستودع على شكل ملف مضغوط , أو تفتحه , وفوقه تجد كلمة (Use SSH) هذه لو رغبت أن تتعامل بين Git الموجود على حاسوبك و Github بالأوامر وبأمان عالي فعليك استخدامه لكن ! هناك بعض الإعدادات التي يجب أن تعلمها أولاً , لنعملها .

## بروتوكول SSH

هو اختصار لكلمتين ( Secure Shell ) وهو بالأصل عبارة عن بروتوكول لتبادل المعلومات بين حاسوبك الشخصي والسيرفر (هنا سوف نتعامل مع سيرفر Github) , ويستعمل مفتاحين للتحقق هما ( Public Key ) و ( Private Key ) طيب , نترك Github الان ونرجع الى Git Bash نفتحه , ونعمل الخطوات الأربعة التالية :

**الخطوة الأولى -** نصنع أو نولد المفتاح العام والخاص لـ SSH :

داخل Git Bash نكتب الأمر التالي :

```
ssh-keygen -t rsa -b 4096 -C "yourEmail"
```

المقصود بـ Your Email هو الأيميل الذي سجلت فيه بـ Github .

```
MINGW64:/c/Users/lenovo
lenovo@Mohammed MINGW64 ~
$ ssh-keygen -t rsa -b 4096 -C "mohammedeydan@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/lenovo/.ssh/id_rsa):
Created directory '/c/Users/lenovo/.ssh'.
Enter passphrase (empty for no passphrase):
```

سوف يطلب منك تحديد الملف الذي تريد خزن فيه المفاتيح , سوف نتركه على حاله الأفتراضي وننقر على ( Enter ) . بعدها سوف يطلب منك الباسورد , ندخلها , ثم يطلب تأكيدها فنكتبها مرة ثانية .

ملاحظة : عندما تكتب الباسورد تظهر الشاشة وكأنك لم تكتب شي .

في الأمر أعلاه أن SSH انشأ مجلد باسم (.ssh) في المسار ( C:/Users/UserName/.ssh ) , وحتى تتأكد تقدر تستخدم الأمر التالي الذي يستعرض مافي المجلد (.ssh)

Ls .ssh

```
MINGW64:/c/Users/lenovo
lenovo@Mohammed MINGW64 ~
$ ls .ssh
id_rsa id_rsa.pub
lenovo@Mohammed MINGW64 ~
$
```


لاحظ فيه ملفين هما (id\_rsa.pub) هو المفتاح الرئيسي , والثاني (id\_rsa) المفتاح الخاص, وهما عبارة عن ملفين نصيين , يكون محتواهم مشفر , لنشاهد مافي داخلهما .

نذهب الى القرص C ثم Users وبعدها نختار اسم المستخدم الذي نعمل عليه (مثلا أنا استخدم Lenovo ) وبداخله نبحث عن المجلد .ssh

This PC > Windows (C:) > Users > lenovo > .ssh

Name	Date modified	Type	Size
id_rsa	31/01/2020 01:29	File	4 KB
id_rsa	31/01/2020 01:29	Microsoft Publisher ...	1 KB

لو فتحنا أحدهما بواسطة (NotePad) سيكون بهذا الشكل المشفر:



```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDAyE42Y
+UpjiYn6/8dS9So/sFVySEK/R4M5iHMqHYFBo/gX5YH7n3fds2yYEAfr
UjyhnuwDWPCT05agmzRtUv8IH4JOW09QlgbjMAD1lSr6IdMdGBre/wc1f
pYhS+n8RYWRQ7VAwICIGMATDwuX/90CQPvgEIZEB1M5pwMO7K1ub
+Xv1EbgKsFIYAz02KcP0uXkg
+z8vkD7CYX4n9pjr2mzYO7a4xYTYIh93HVteJ3VmYmLRHnP05MZKSbOi8
GjR+4F1JypRy+b7enA/ke8Mj0OWO4qajG0NRgdZDBT65/
+QonZBLflYooy7NBncxvIYgmM7OTqHri56pyHZB/Iztk7+L5p5mii3a
+m1Uz/RuZML41YygGpwjO2jH8AKCO0VHV
+Cr6n7ldYjsswqKzi9glLMhDyuNFF1HEc9wrS/uWmb7R8pDP7jDW9zEwG
UTOu10cNFwtapKxUgHdvPED6KovK1lNWUgUdQ9wP6cG2vCtDyGHqnU3YP
dQMNSdUcasTL0GvtNTZwLu+G2EXQ
+SvR9VfJrpMR0OJbT897R8wzj7AXTeRUimplQBwBzp1n
+axn3W/gzIxWEgEjRZHuoZFKuH9dVGXx1m8dOl2y9C2X2ido9+apKui
+9HFQwDJcbjHwh7RJPEUSPyIXRf/o2tzAHSDnV7fZE4KIu86NtqrKRSQ=
```

### الخطوة الثانية : تسجيل المفتاح الخاص ب SSH-Agent

نفتح الـ Git Bash , ونكتب فيه الأمر التالي لتشغيل ssh-agent , لكن أنتبه للعلامة ( ` ) الموجودة في أقصى يسار لوحة المفاتيح .

```
Eval `ssh-agent -s`
```

كما في الصورة أدناه :

```
MINGW64:/c/Users/lenovo
lenovo@Mohammed MINGW64 ~
$ eval `ssh-agent -s`
Agent pid 1024

lenovo@Mohammed MINGW64 ~
$ |
```

وحتى نضيف المفتاح الخاص الذي هو في الملف (id\_rsa) نستخدم أمر الأضافة ssh-add , سوف يطلب منا باسورد Github .  
بهذا الشكل :

```
MINGW64:/c/Users/lenovo
lenovo@Mohammed MINGW64 ~
$ eval `ssh-agent -s`
Agent pid 1024

lenovo@Mohammed MINGW64 ~
$ ssh-add ~/.ssh/id_rsa
Enter passphrase for /c/Users/lenovo/.ssh/id_rsa:
Identity added: /c/Users/lenovo/.ssh/id_rsa (mohammedeydan@gmail.com)

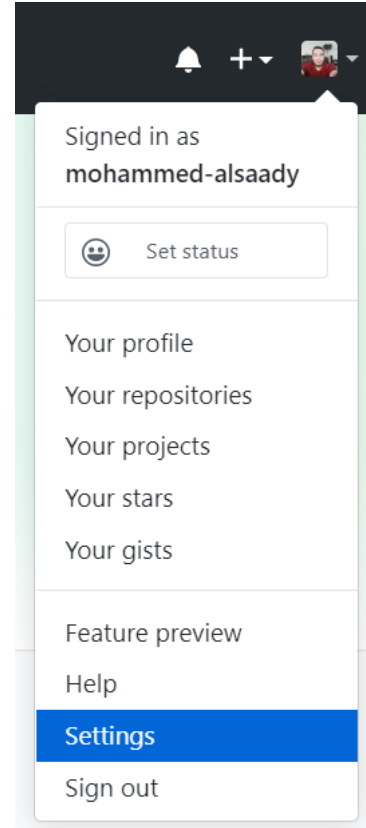
lenovo@Mohammed MINGW64 ~
$ |
```

مبروك لقد سجلت المفتاح الخاص , بقي المفتاح العام .

**الخطوة الثالثة :** نسجل المفتاح العام في Github .

تتذكر في الخطوة الأولى الملف العام هو (Id\_rsa.pub) وفتحناه بالنوت باد , نحتاج ان ننسخ كل محتوياته لكن ! كيف أميز بين المفتاح الرئيسي والخاص !

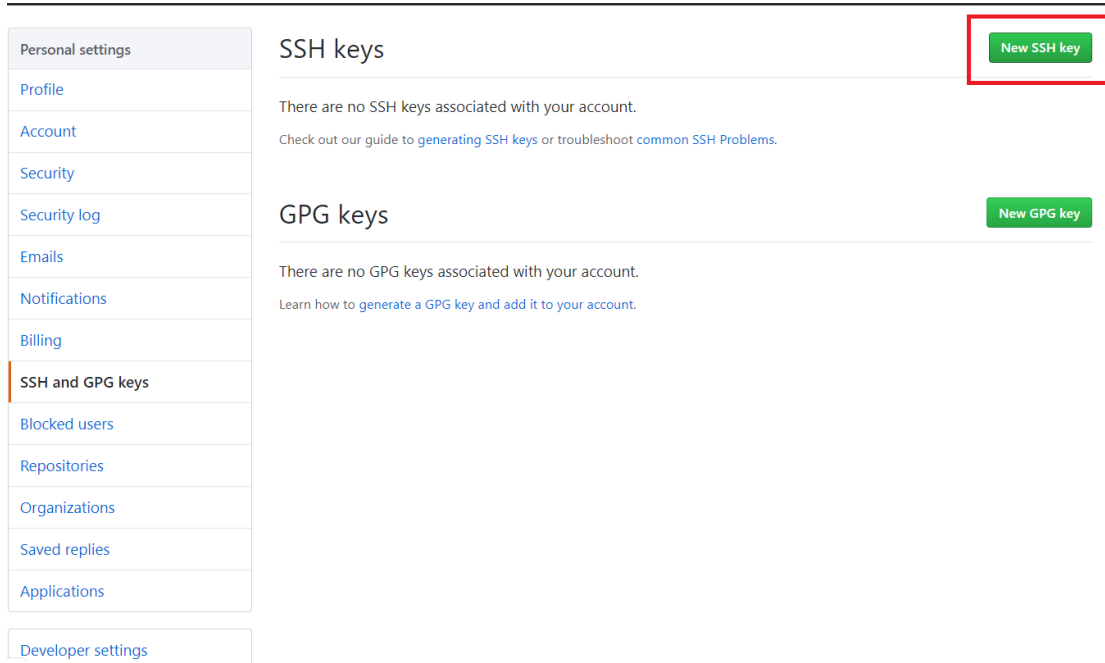
الأمر بسيط الملف الذي تفتحه وتجده أول سطر فيه ( Begin OpenSSH Private Key ) فهذا المفتاح الخاص , اما المفتاح العام تجد في آخر سطر أيميلك .  
الآن نذهب الى Github , ومن القائمة اعلى اليمين نختار ( Setting )



ستفتح لنا صفحة الإعدادات , من القائمة نختار (SSH and GPG Keys)

Personal settings	Public profile
Profile	<b>Name</b> <input type="text" value="Mohammed Alsaady"/> <small>Your name may appear around GitHub where you contribute or are mentioned. You can remove it at any time.</small>
Account	<b>Public email</b> <input type="text" value="Select a verified email to display"/>
Security	<small>You have set your email address to private. To toggle email privacy, go to <a href="#">email settings</a> and uncheck "Keep my email address private."</small>
Security log	<b>Bio</b> <input type="text" value="Developer ."/>
Emails	<small>You can @mention other users and organizations to link to them.</small>
Notifications	<b>URL</b> <input type="text"/>
Billing	<b>Company</b> <input type="text"/>
SSH and GPG keys	<small>You can @mention your company's GitHub organization to link it.</small>
Blocked users	
Repositories	
Organizations	
Saved replies	
Applications	
Developer settings	

لتفتح لنا الصفحة ومنها ننقر على ( New SSH Key )



Personal settings

- Profile
- Account
- Security
- Security log
- Emails
- Notifications
- Billing
- SSH and GPG keys**
- Blocked users
- Repositories
- Organizations
- Saved replies
- Applications

Developer settings

### SSH keys

There are no SSH keys associated with your account.

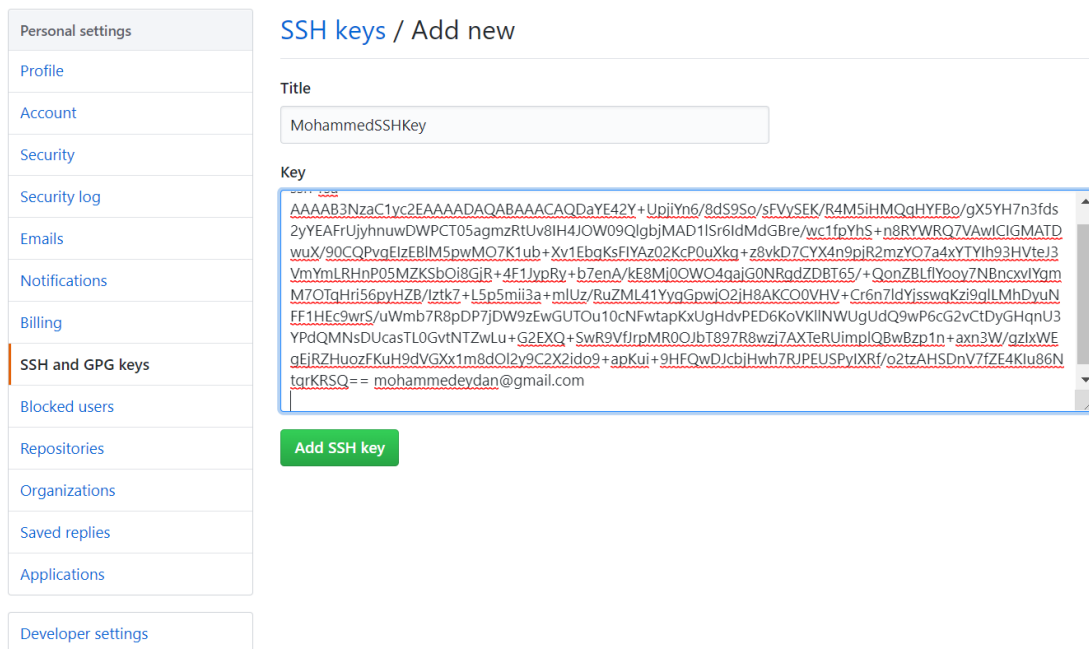
Check out our guide to [generating SSH keys](#) or [troubleshoot common SSH Problems](#).

### GPG keys

There are no GPG keys associated with your account.

Learn how to [generate a GPG key](#) and add it to your account.

سوف يطلب منا عنوان المفتاح بإمكانك تسميته أي شيء مثلًا أنا سوف أسميه (MohammedSSHKey) والصق المفتاح الذي نسخته في حقل Key .



Personal settings

- Profile
- Account
- Security
- Security log
- Emails
- Notifications
- Billing
- SSH and GPG keys**
- Blocked users
- Repositories
- Organizations
- Saved replies
- Applications

Developer settings

### SSH keys / Add new

Title

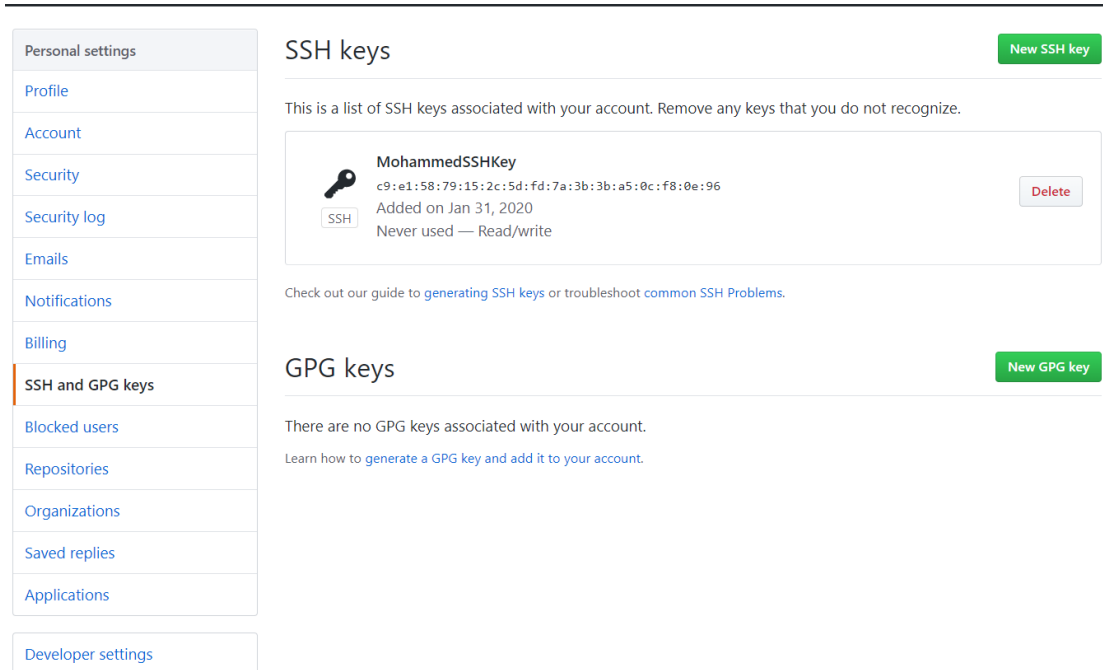
MohammedSSHKey

Key

```
AAAAB3NzaC1yc2EAAAADAQABAAQDAIE42Y+UpjiYn6/8dS9So/sFVySEK/R4M5iHMqgHYFBo/gX5YH7n3fds  
2yYEAfrUjyhnuwDWPCT05agmzRtUv8IH4JOW09QgjbMAD1ISr6idMdGBre/wc1fpYhS+n8RYWRQ7VAwICIGMATD  
wuX/90CQPvgElzEBIM5pwMQ7K1ub+Xv1EbgKsFIYAz02KcP0uXkq+z8vkD7CYX4n9piR2mzYO7a4xTYIh93HVteJ3  
VmYmLRHnPO5MZKSbOi8GIR+4F1JypRy+b7enA/ke8Mi0OWO4gajG0NRqdZDBT65/+QonZBLfYooy7NBncxvYgm  
M7OTqHri56pyHZB/lztk7+LSp5mii3a+mLUz/RuZML41YyqGpwO2jH8AKCO0VHV+Cr6n7ldYjsswqKzi9gLLMhDyuN  
FF1HEc9wrS/uWmb7R8pDP7jDW9zEwGUTOu10cNFwtapKxUgHdvPED6KoVKIINWUgUdQ9wP6cG2vCtDyGHqnU3  
YPdQMNsDUcasTL0GvntZwLu+G2EXQ+SwR9VfJrpMROOJbT897R8wzj7AXTeUimplQBwBzp1n+axn3W/gzlxWE  
gEjRZHuoZFkuH9dVGXx1m8dOl2y9C2X2ido9+apKui+9HFQwDjcbjHwh7RJPEUSPyIXRf/o2tzAHSdNv7fZE4Klu86N  
trKRsq== mohammedeydan@gmail.com
```

Add SSH key

ثم انقر على Add SSH Key , سوف يطلب منك إدخال الباسورد , أكتبها وانقر على Confirm Password , بعدها سوف يضيف المفتاح وتظهر لك الصفحة هذه .



The screenshot shows the GitHub 'SSH keys' page. On the left is a sidebar with navigation links: Personal settings, Profile, Account, Security, Security log, Emails, Notifications, Billing, SSH and GPG keys (highlighted), Blocked users, Repositories, Organizations, Saved replies, Applications, and Developer settings. The main content area is titled 'SSH keys' and includes a 'New SSH key' button. Below the title, it states: 'This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.' A table lists one key: 'MohammedSSHKey' with a key icon, a long alphanumeric ID, 'Added on Jan 31, 2020', and 'Never used — Read/write'. A 'Delete' button is next to it. Below the table is a link to a guide: 'Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH Problems](#).' Below this is the 'GPG keys' section with a 'New GPG key' button. It states: 'There are no GPG keys associated with your account.' and provides a link: 'Learn how to generate a GPG key and add it to your account.'

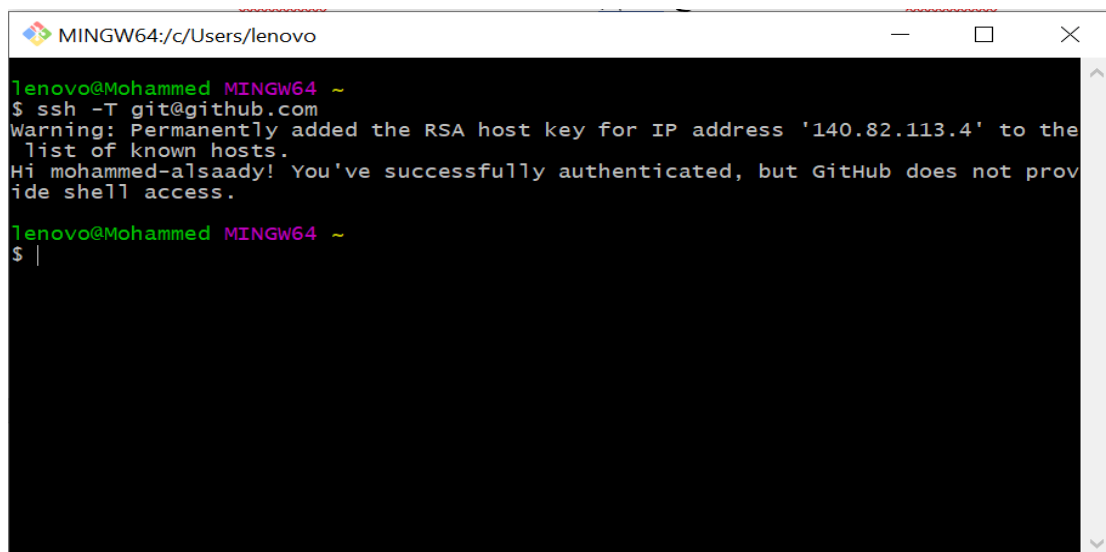
مبروك أكملت تسجيل المفتاح العام , بقيت لنا خطوة نتأكد من الأتصال بين Git والـ Github .

أيضا سوف يصلك إيميل من Github يخبرك بأن أحدهم أضاف مفتاح SSH .

**الخطوة الرابعة : فحص الأتصال بين Git والـ Github**

نفتح الـ Git Bash ونكتب الأمر التالي : `ssh -T git@github.com`

في بداية انشاء الاتصال ربما يسألك ان كنت عملت كل شي تكتب (yes) وتنقر على . Enter



```
MINGW64:/c/Users/lenovo
Tenovo@Mohammed MINGW64 ~
$ ssh -T git@github.com
Warning: Permanently added the RSA host key for IP address '140.82.113.4' to the list of known hosts.
Hi mohammed-alsaady! You've successfully authenticated, but GitHub does not provide shell access.
Tenovo@Mohammed MINGW64 ~
$ |
```

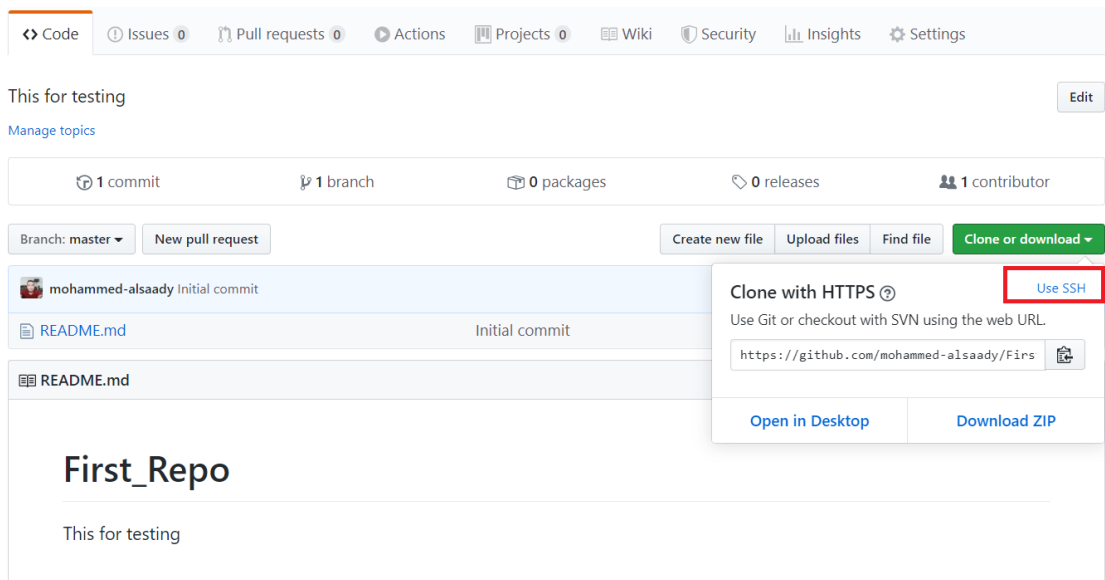


إذا وجدت العبارة (You've successfully) معناه مبروك لقد تم الاتصال بنجاح .

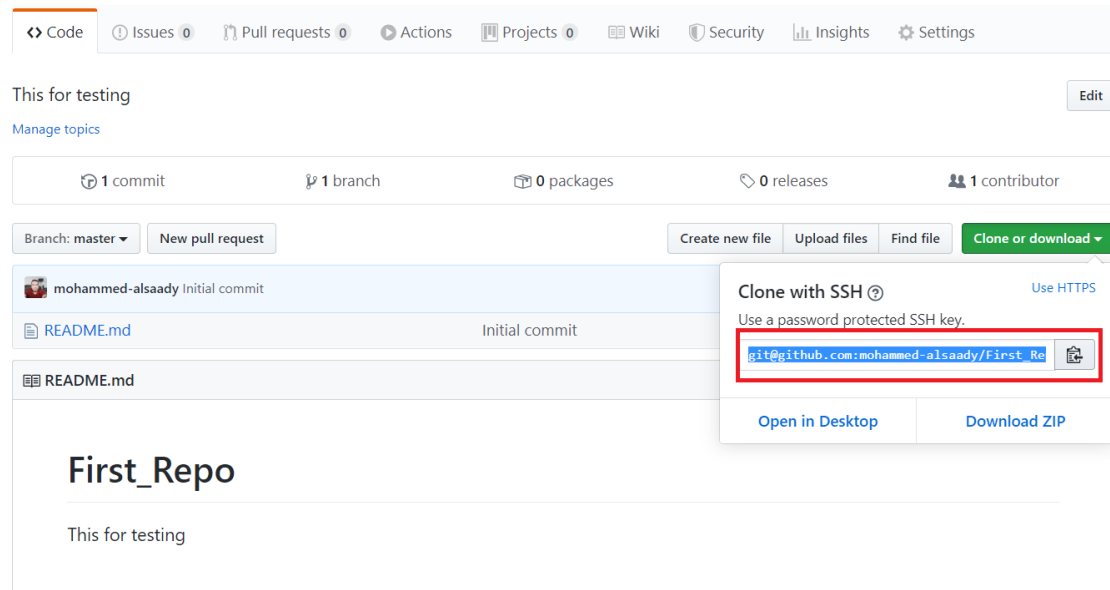
الآن وبعد أن أكملنا جميع الإعدادات بقي لدينا أننا سوف نعمل مجلد خاص بالمشاريع التي نعمل عليها (وهو اختياري) مثلاً أنا سوف أنشأ مجلد على القرص D باسم (Projects) من أجل تنزيل فيه المستودع (First\_Repo) ولعمل ذلك سوف افتح Git Bash وانتقل إلى القرص D وبعدها أعمل مجلد (Projects) و أدخل فيه من خلال الأوامر التالية :

```
MINGW64:/d/Projects
Tenovo@Mohammed MINGW64 ~
$ pwd
/c/Users/Tenovo
Tenovo@Mohammed MINGW64 ~
$ cd d:
Tenovo@Mohammed MINGW64 /d
$ pwd
/d
Tenovo@Mohammed MINGW64 /d
$ mkdir Projects
Tenovo@Mohammed MINGW64 /d
$ cd Projects
Tenovo@Mohammed MINGW64 /d/Projects
$ pwd
/d/Projects
Tenovo@Mohammed MINGW64 /d/Projects
$
```

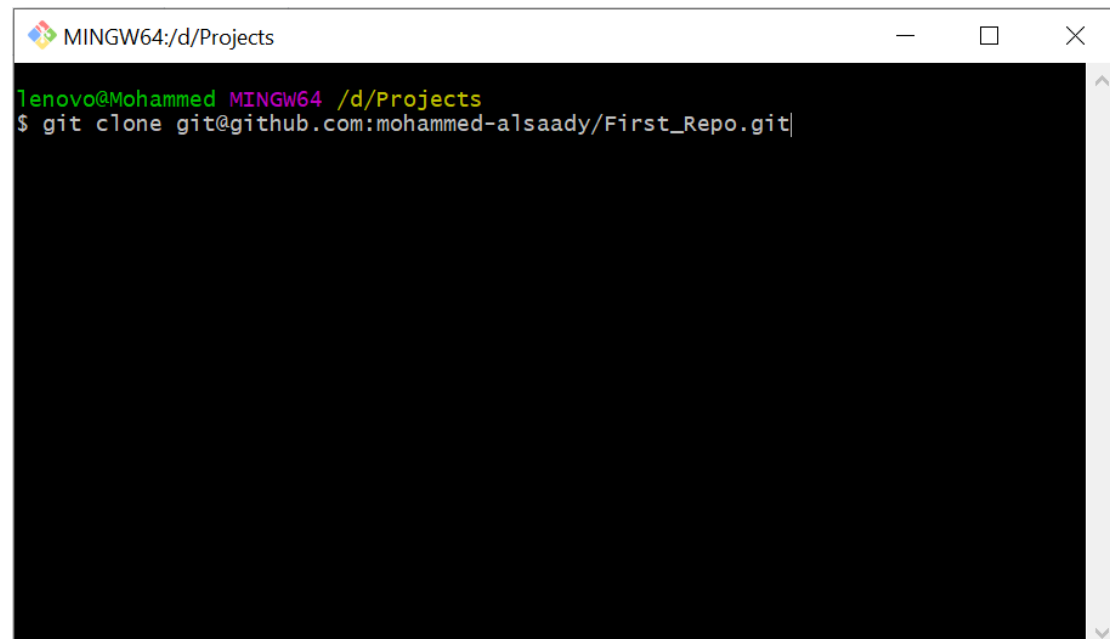
الآن أذهب إلى Github وافتح المستودع (First\_Repo) ومن (Clone or Download)



الآن صار لديك أما ان تستعمل https أو البروتوكول SSH , أنا سوف أستخدم SSH , كما تلاحظ في الصورة أعلاه تنقر على Use SSH , وأنتبه للرابط كيف يتغير عند استخدامك Https أو SSH . بعدها ننسخ الرابط



ونرجع الى لـ Git Bash ونعمل Clone من خلال الأمر : **git clone link**



وننقر على Enter , سوف يطلب منك كلمة السر التي وضعتها عند انشاء المفتاح العام ( نحن استخدمنا نفس باسورد Git Hub ) بعدها سوف يقوم بتنزيل المستودع داخل المجلد Projects .

```
MINGW64:/d/Projects
Tenovo@Mohammed MINGW64 /d/Projects
$ git clone git@github.com:mohammed-alsaady/First_Repo.git
Cloning into 'First_Repo'...
Enter passphrase for key '/c/Users/lenovo/.ssh/id_rsa':
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.

Tenovo@Mohammed MINGW64 /d/Projects
$
```

الآن لو فتحت المجلد Projects سوف تجد مجلد جديد اسمه ( First\_Repo ) , هذا المجلد يسمى ( Working Directory ) وبعض الأحيان يسمى ( Working Tree ) وتكون بداخله جميع مجلدات وملفات المشروع الذي نعمل عليه .

داخل هذا المجلد ملف واحد فقط هو ( README.md ) , طيب الآن لندخل عليه من Git bash وأيضا سوف نستعرض الملفات والمجلدات الظاهرية من خلال الأمر ( ls ) أما إذا أردنا استعراض جميع الملفات والمجلدات المخفية فأننا سوف نستخدم الأمر ( ls -a ) كما في الصورة أدناه .

```
MINGW64:/d/Projects/First_Repo
Tenovo@Mohammed MINGW64 /d/Projects
$ pwd
/d/Projects

Tenovo@Mohammed MINGW64 /d/Projects
$ cd First_Repo/

Tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ pwd
/d/Projects/First_Repo

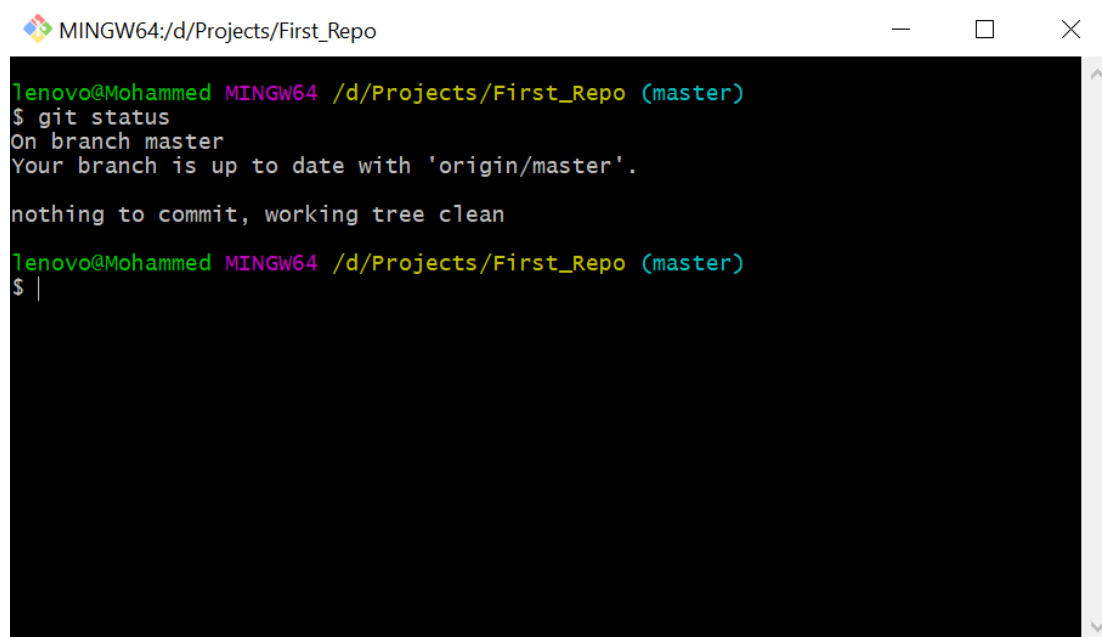
Tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ ls
README.md

Tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ ls -a
./ ../ .git/ README.md

Tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$
```

إذا لاحظت بالأمر الأخير ( ls -a ) أستعرض لنا مجلد مخفي باسم (.git) ! شنو هذا ؟؟؟

هذا عبارة عن مجلد يستخدمه Git حتى يحفظ بي أي تغييرات تصير بداخل المستودع أو المجلد ( First\_Repo ) سوف نتكلم عنه بالتفصيل فيما بعد , لكن الان دعنا نحص حالة الملفات الموجودة في المستودع ( First\_Repo ) الذي أنزلناه من خلال الامر التالي : **git status**



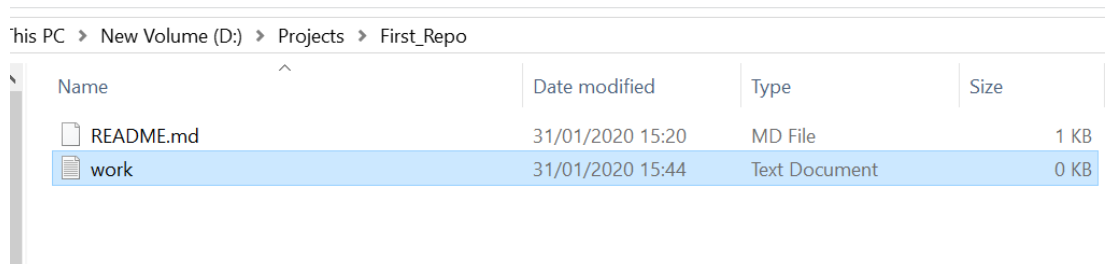
```
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

إذا لاحظت أن git في الصورة أعلاه يخبرنا ليس هناك أي تغيير وأن المجلد ( First\_Repo ) الذي يسميه ( Working Tree ) ليس فيه أي تغيير .

لنفتح المجلد ( First\_Repo ) من خلال ( Windows Explorer ) ونعمل ملف نصي جديد فيه وليكن اسمه ( work.txt )



Name	Date modified	Type	Size
README.md	31/01/2020 15:20	MD File	1 KB
work	31/01/2020 15:44	Text Document	0 KB

الآن لنعيد الأمر `git status` ونرى :

```
MINGW64:/d/Projects/First_Repo
tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
   work.txt

nothing added to commit but untracked files present (use "git add" to track)
tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

أعلاه Git يخبرنا هناك ملف باسم (`work.txt`) لن يراقبه ان لم نضعه في الفهرس (Index) ! لحظة شنو الفهرس ؟

الآلية التي يستخدمها الـ Git تكون على ثلاث مراحل :

**المرحلة الأولى – Working Directory** ويكون فيه جميع الملفات سواء التي يراقبها الـ git او لا .

**المرحلة الثانية –** نصنع قائمة بالملفات التي نريد من الـ git مراقبتها وهي تسمى بمرحلة **Staging Area** .

**المرحلة الثالثة –** نضيف الملف الى المستودع المحلي الذي ينشأه git بصورة تلقائية بعد المرور بهذه المراحل التي يمر بها الملف الجديد يصير بإمكاننا رفعه على Github .

طيب , كيف ننقل الملف من مرحلة **Working Directory** الى مرحلة **Staging Area** ؟ بأمر واحد بسيط وهو : `git add filename.extension` وكما في الصورة التالية :

```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git add work.txt

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   work.txt

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$
```

لاحظ عندما استخدمت الأمر (git status) فإن الـ Git يخبرنا هناك ملف اسمه (work.txt) في مرحلة (Staging Area) .

الآن نريد نقله الى المستودع المحلي فأننا سوف نعمل معه ملاحظة أو رسالة تخبر الآخرين اننا عملنا كذا وكذا داخل المشروع من خلال الأمر : **git commit -m "message"** , حيث ان -m تخبر الـ git أن هذا الامر سوف يستخدم علامات الاقتباس , للنفذ الأمر ونرى .

```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git commit -m "This first change in my project! "

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'lenovo@Mohammed.(none)')

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$
```

لكن ! لم ينفذ الأمر !!! الـ Git يطلب مني أن أعرفه من أنا قبل ان أكمل الأمر .

طيب لتعرف عليه من خلال الأمرين : `git config --global user.name "your name"` والأمر : `git config --global user.email "your email"` , وكما في الصورة أدناه :

```
MINGW64:/d/Projects/First_Repo
tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git config --global user.name "Mohammed-ALsaady"

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git config --global user.email "mohammedeydan@gmail.com"

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$
```

الان لنعيد الأمر (`git commit -m "message"`) :

```
MINGW64:/d/Projects/First_Repo
tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git config --global user.name "Mohammed-ALsaady"

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git config --global user.email "mohammedeydan@gmail.com"

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git commit -m "This first change in my project! "
[master ad9581d] This first change in my project!
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 work.txt

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (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

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

الان صار كل شي تمام , لكن ! كيف نرفع التغييرات التي حدثت على github , الامر بسيط أولاً يجب ان نحدد المستودع على github الذي نريد نرفع التعديلات عليه ( ربما يكون لديك اكثر من مستودع ) وحسب الأمر : `git remote`

```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git remote
origin

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

تمام , بصورة افتراضية المستودع الذي عملناه على الـ Github يختصر اسمه باسم ( Origin ) في حالة عملنا له clone نستخدم الامر التالي : **git remote show origin** ربما يطلب منك ادخال كلمة السر .

```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git remote
origin

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git remote show origin
Warning: Permanently added the RSA host key for IP address '140.82.114.4' to the
list of known hosts.
Enter passphrase for key '/c/Users/lenovo/.ssh/id_rsa':
* remote origin
Fetch URL: git@github.com:mohammed-alsaady/First_Repo.git
Push URL: git@github.com:mohammed-alsaady/First_Repo.git
HEAD branch: master
Remote branch:
  master tracked
Local branch configured for 'git pull':
  master merges with remote master
Local ref configured for 'git push':
  master pushes to master (fast-forwardable)

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

كما تشاهد من الشاشة موجود في المستودع ( First\_Repo ) الموجود على Github .

الآن نريد إجراء التحديث على المستودع ( First\_Repo ) على Github من خلال الأمر التالي : **git push origin master** كما في الصورة ادناه , سوف يطلب منك كلمة السر يجب ادخالها :



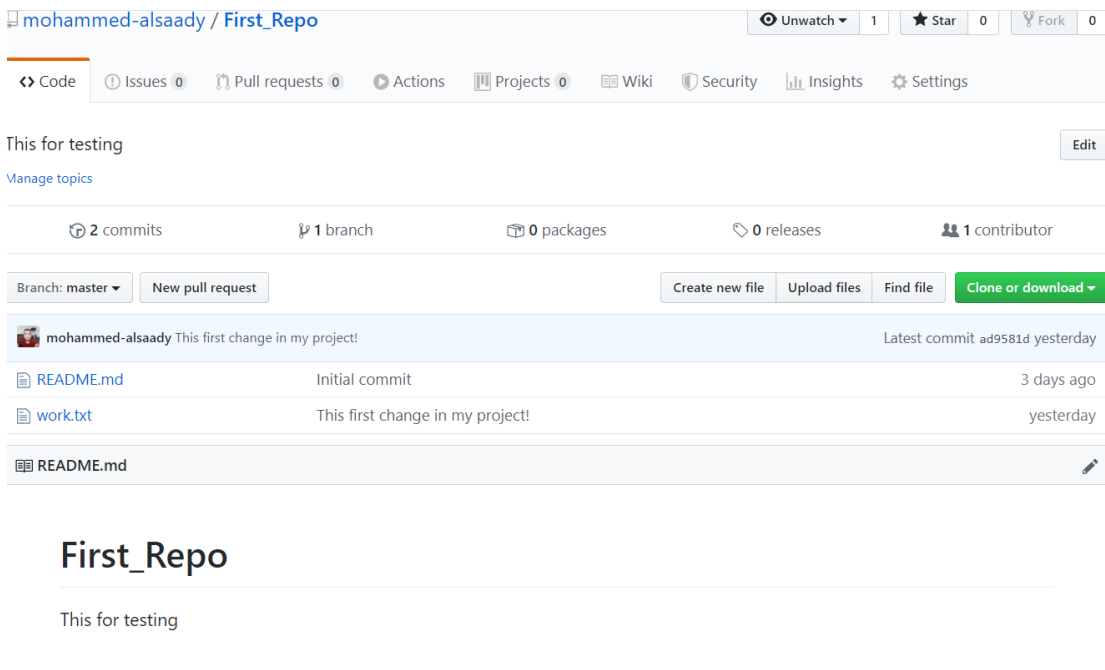
```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git push origin master
Enter passphrase for key '/c/Users/lenovo/.ssh/id_rsa':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 300 bytes | 60.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:mohammed-alsaady/First_Repo.git
 a56e190..ad9581d  master -> master

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$
```

مبروك! تم رفع كل التحديثات التي علمتها على مشروع على Github , لكن ماهو  
origin و master ؟

Master هي branch سوف نتطرق لها فيما بعد , أما origin فأنه الاسم المختصر  
للمستودع البعيد الموجود على github الذي علمنا له Clone فيما سبق.

وحتى نتأكد أذهب الى Github وأعمل Refresh للصفحة ستجد الملف work.txt  
موجود .



لاحظ الرسالة الموجودة بجانب الملف (work.txt) حتى تفهم لماذا كتبناه , طيب  
لنرجع الى Git Bash وننفذ الامر **git status** لنرى ماذا حدث .

```
MINGW64:/d/Projects/First_Repo
tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$
```

سوف تظهر الرسالة أعلاه تخبرنا بأنه تم تحديث كل شيء وليس هناك ما يحتاج لعمل (Commit) .

طيب , لو أننا فتحنا الملف work.txt الموجود على الحاسوب وكتبنا فيه شيء وعملنا . git status

```
work - Notepad
File Edit Format View Help
ICT Taskforce Initiative .

Windows (CRLF) Ln 1, Col 27 100%
```

ثم أحفظ , بعدها اذهب الى gitbash وكتبت فيه :

```
MINGW64:/d/Projects/First_Repo
tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   work.txt

no changes added to commit (use "git add" and/or "git commit -a")

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

يخبرنا git بأن الملف work.txt قد حصلت فيه بعض التعديلات ويجب ان تضيفه الى الفهرس (staged) , نكتب امر الإضافة \* git add حتى يضيف جميع الملفات الغير مفهرسه الى الفهرس بدل ما نظيف ملف واحد :

```
MINGW64:/d/Projects/First_Repo
tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git add *

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   work.txt

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

سؤال لو أردت أرجاع الملف من Staging Area الى Working Directory ماذا افعل ؟ الأمر بسيط جداً , أستخدم الامر : **git restore --staged work.txt** طيب , الان الخطوة التالية هي إضافة الملف من الفهرس الى المستودع المحلي وتكون من خلال الأمر git commit ويكون بهذا الشكل :

```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git commit -m "I had added a new text to the working file!"
[master 65b67c5] I had added a new text to the working file!
1 file changed, 1 insertion(+)

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (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

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

الأمور لحد الان تمام , لكن بقى نرفع التحديثات الى Github من خلال الامر `git push`.

```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git push origin master
Enter passphrase for key '/c/Users/lenovo/.ssh/id_rsa':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 334 bytes | 334.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:mohammed-alsaady/First_Repo.git
   ad9581d..65b67c5  master -> master

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

الان لو ذهبت الى الـ github وفتحت المستودع First\_Repo سوف تجد

mohammed-alsaady / First\_Repo

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

This for testing Edit

Manage topics

3 commits 1 branch 0 packages 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

mohammed-alsaady I had added a new text to the working file! Latest commit 65b67c5 3 minutes ago

README.md	Initial commit	3 days ago
work.txt	I had added a new text to the working file!	3 minutes ago

README.md

طيب السؤال هنا : اين اشاهد التعديلات التي جرت على مشروعي ؟ بصورة بسيطة  
تقدر تشوف أي تغييرات صارت على مشروعك من خلال الامر **git log**

```

MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git log
commit 65b67c5d8b9b76cc37634e6440a8860a60a3f67e (HEAD -> master, origin/master, origin/HEAD)
Author: Mohammed-Alsaady <mohammedeydan@gmail.com>
Date: Sun Feb 2 00:26:39 2020 +0300

    I had added a new text to the working file!

commit ad9581d04ef4ee2517813b55edbe039d67c73713
Author: Mohammed-Alsaady <mohammedeydan@gmail.com>
Date: Fri Jan 31 16:09:25 2020 +0300

    This first change in my project!

commit a56e1908eccc9e2e6fbef1cc0e33b2bcbccd2bfc
Author: Mohammed Alsaady <60448608+mohammed-alsaady@users.noreply.github.com>
Date: Thu Jan 30 00:20:52 2020 +0300

    Initial commit

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$

```

لاحظ أعطاك كل شي وبالتفصيل وصفها على شكل Commit (باللون الأصفر) ولكل واحد معرف خاص بها و تعني هذا يمثل تعديل صار على كل المشروع .

طيب , لو أردت استعراض جميع Commits بصورة بسطر واحد , بهذه الحالة  
تقدر تستخدم الأمر : **git log --online --decorate**

```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git log --oneline --decorate
65b67c5 (HEAD -> master, origin/master, origin/HEAD) I had added a new text to t
he working file!
ad9581d This first change in my project!
a56e190 Initial commit

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$
```

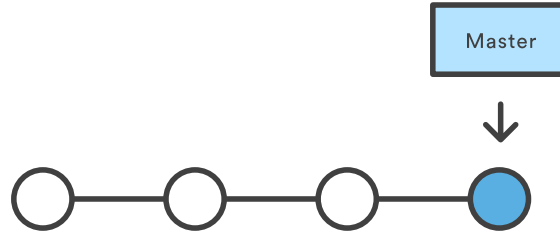
طيب ماذا لو كانت ال Commit كثيرة وأردت استعراض عدد معين فقط ؟ في هذه الحالة تقدر تحدد العدد الذي تريد عرضه من خلال إضافة المعامل `--max-count=` مثلا اريد استعرض اثنين فقط :

```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git log --oneline --decorate --max-count=2
65b67c5 (HEAD -> master, origin/master, origin/HEAD) I had added a new text to t
he working file!
ad9581d This first change in my project!

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$
```

تردد كثيراً مصطلح ( Branch ) ماهو ؟

باللغة العربية يعني فرع , لكن كمفهوم ( Concept ) في git يعني خط مستقل من التعديلات التي جرت او تجرى على مشروع ما ومن أمثلتها ( Master )! أعتقد لازال غير مفهوم لناخذ الشكل أدناه ونرى:



في الشكل أعلاه لدينا Branch أسمها ( Master ) ترتبط بها دوائر ( كل دائرة مرتبطة بالأخرى ) فالدائرة الأولى تمثل إضافة النص الى الملف work.txt اما الدائرة الثانية تمثل في إضافة الملف work.txt وهكذا .

إذا لاحظت وجود كلمة Head وهو عبارة عن مؤشر يُوْشر على Branch الحالية التي نستخدمها .

طيب ممكن استعرض Branches الموجودة في مشروعي , بأستخدام الأمر التالي : **git branch**

```

MINGW64:/d/Projects/First_Repo
tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git branch
* master

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$

```

بصورة أفتراضية الـ git يعمل branch واحدة أسمها ( master ) على حاسوبك وعلى المستودع الموجود على github , طيب ماذا لو أستعرضنا الـ branches الموجودة على المستودع First\_Repo في Github , بأستخدام الامر التالي : **git**

**branch --remote**

```
MINGW64:/d/Projects/First_Repo
tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git branch
* master

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git branch --remote
origin/HEAD -> origin/master
origin/master

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$
```

طيب لو أردت مشاهدة ماهي التعديلات على الملف work.txt قبل أن أنقله الى Staging Area؟ بصورة بسيطة افتح الملف work.txt وأمسح ما فيه وأكتب اسمك واحفظه , بعدها افتح gitbash واكتب الامر Git Status :

```
MINGW64:/d/Projects/First_Repo
tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   work.txt

no changes added to commit (use "git add" and/or "git commit -a")

tenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

الان أريد اعرض التغيرات التي صارت على work.txt , يكون من خلال الامر **git diff filename** كما في الصورة أدناه :



```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git diff work.txt
diff --git a/work.txt b/work.txt
index c953b3c..9d17c85 100644
--- a/work.txt
+++ b/work.txt
@@ -1,1 @@
-ICT Taskforce Initiative .
\ No newline at end of file
+Mohammed Alsaady
\ No newline at end of file

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

لاحظ أنه وضع علامة سالبة قبل ( ICT Taskforce initiative ) ( باللون الأحمر ) التي مسحتها , وعلامة + قبل اسمي ( باللون الأخضر ) ,  
طيب أنقل الملف الى الفهرس من خلال الامر ( git add ) :

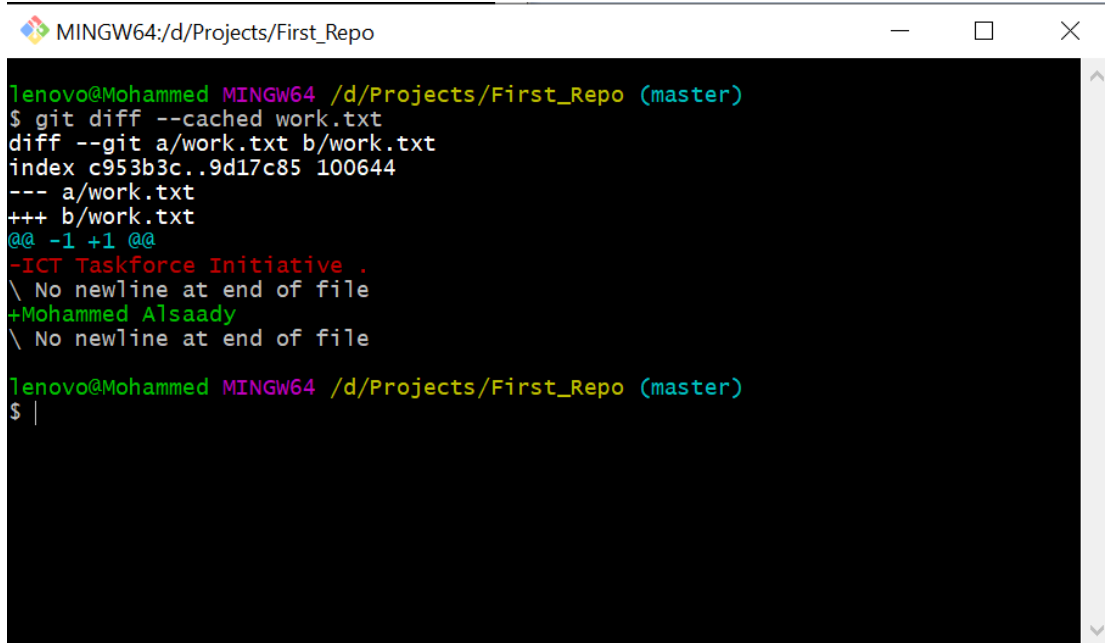
```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git add work.txt

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   work.txt

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```

ماذا لو أردت ان أرى التغييرات التي صارت على الملف work وهو في مرحلة Staging واقارنها مع الملف وهو في مرحلة Commit ؟ الامر بسيط نستخدم المعامل **-cached**



```
MINGW64:/d/Projects/First_Repo
lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ git diff --cached work.txt
diff --git a/work.txt b/work.txt
index c953b3c..9d17c85 100644
--- a/work.txt
+++ b/work.txt
@@ -1,1 @@
-ICT Taskforce Initiative .
\ No newline at end of file
+Mohammed Alsaady
\ No newline at end of file

lenovo@Mohammed MINGW64 /d/Projects/First_Repo (master)
$ |
```