

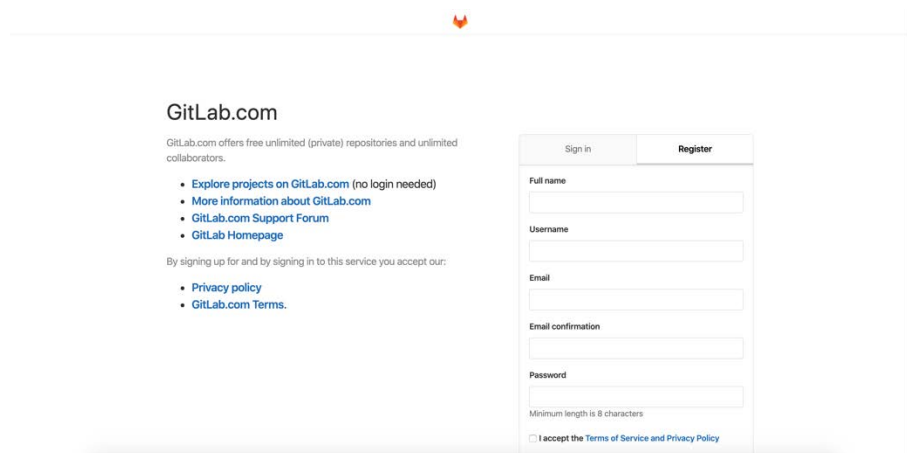
## Lab 03 - วิธีการใช้งาน Gitlab

วันที่ 29 มกราคม 2563

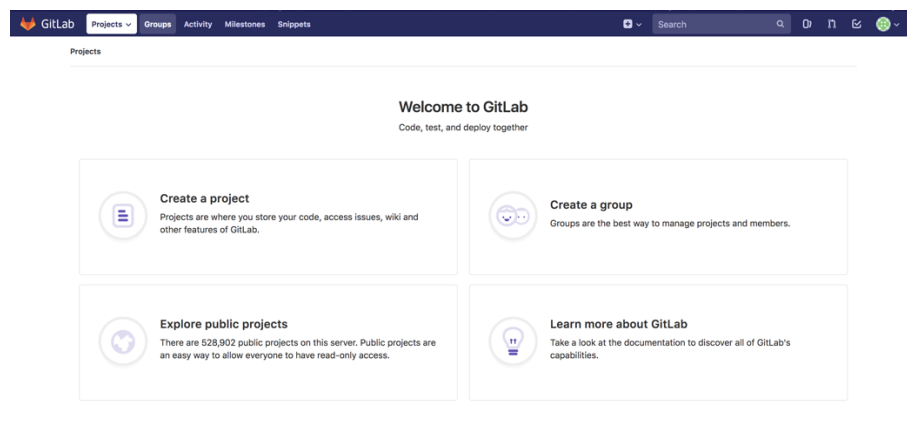
GitLab เป็น Git repository management และเป็น Open-Source เจ้าเดียวที่ถูกนำมาเทียบ โดย GitLab เป็นอีกเครื่องมือหนึ่งที่ได้รับคามนิยม สามารถทำ Code reviews, Issue tracking, activity feeds และ wikis ได้ ที่สามารถรองรับ users ได้มากกว่า 25,000 คนโดยใช้เพียง single GitLab server แล้วก็ยัง สามารถรองรับได้สูงกว่านี้อีกถ้าหากใช้เครื่องที่มีคุณภาพสูงขึ้น หรือเพิ่มเครื่องขึ้นมา

### การใช้งาน GitLab เบื้องต้น

1. การเริ่มต้นใช้งาน GitLab ให้ไปลงทะเบียน/สมัครบัญชีที่เว็บ [https://gitlab.com/users/sign\\_in](https://gitlab.com/users/sign_in)



2. เลือก Create a project



3. ตั้งชื่อโปรเจกต์ในช่อง Project name เลือก Visibility level เป็น Private แล้วกด Create project

**New project**

A project is where you house your files (repository), plan your work (issues), and publish your documentation (wiki), among other things.

All features are enabled for blank projects, from templates, or when importing, but you can disable them afterward in the project settings.

To only use CI/CD features for an external repository, choose **CI/CD for external repo**.

Information about additional Pages templates and how to install them can be found in our [Pages getting started guide](#).

**Tip:** You can also create a project from the command line. [Show command](#)

**Blank project** | Create from template | Import project | CI/CD for external repo

**Project name**  
My awesome project

**Project URL**  
[https://gitlab.com/opor\\_tan/](https://gitlab.com/opor_tan/)

**Project slug**  
my-awesome-project

Want to house several dependent projects under the same namespace? [Create a group](#).

**Project description (optional)**  
Description format

**Visibility Level**

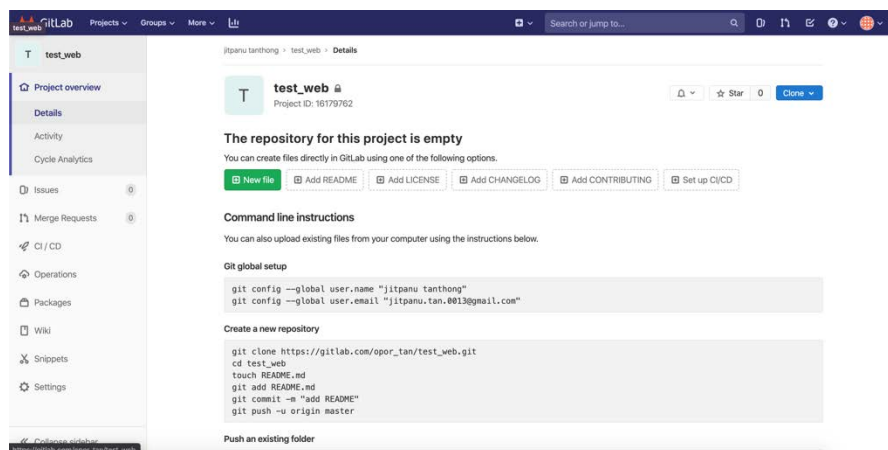
☒ **Private**  
Project access must be granted explicitly to each user.

☐ **Public**  
The project can be accessed without any authentication.

☐ **Initialize repository with a README**  
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

**Create project** | Cancel

จะได้โปรเจกต์หน้าตาอย่างตัวอย่าง ดังนี้



4. กด + Add README เพื่อสร้างไฟล์ Readme.md ซึ่งโดยปกติแล้ว ไฟล์ Readme.md จะเป็นไฟล์ที่ใช้อธิบายรายละเอียดคร่าวๆ เกี่ยวกับโปรเจกต์ซึ่งจะปรากฏขึ้นเป็นหน้าแรกของโปรเจกต์ เมื่อทำการแก้ไขไฟล์ Readme.md เรียบร้อยแล้ว ให้กด Commit change เพื่อบันทึกไฟล์

**New file**

Y master / README.md | Select a template type | FS Soft wrap | text

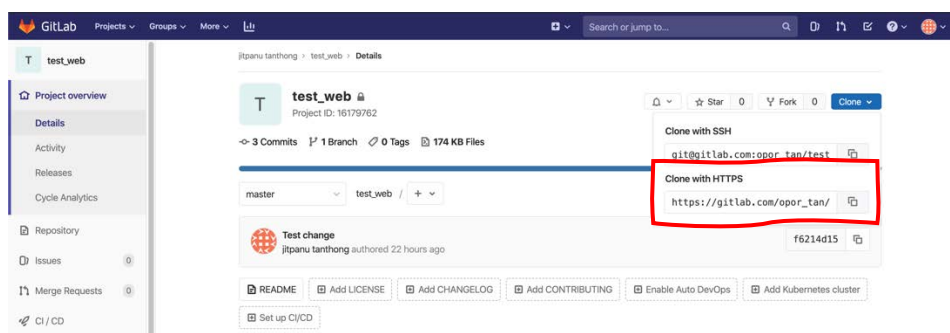
1

**Commit message**  
Add README.md

**Target Branch**  
master

**Commit changes** | Cancel

5. ตัวอย่างหน้าต่างของโปรเจกจะเป็นดังภาพ ต่อมาเราจะทำการคัดลอกโปรเจกที่เราสร้างขึ้นลงสู่คอมพิวเตอร์ของเรา ซึ่งสามารถทำได้โดยการคัดลอก URL จากช่อง Clone with HTTPS



### การคัดลอกโปรเจกจาก GitLab repository

1. สร้าง Folder ใหม่ภายในคอม เพื่อเก็บโปรเจกที่เราจะ clone จาก GitLab เช่น  
D:\gitlab\test\_web
2. เปิด CMD / Terminal
  - a. ใช้คำสั่ง cd เพื่อเข้าไปใน folder ที่สร้างไว้
  - b. ใช้คำสั่ง git --version เพื่อตรวจสอบ version ของ git ถ้าคอมพิวเตอร์ยังไม่ได้ลงโปรแกรม Git ให้ดาวน์โหลดไฟล์จาก <https://git-scm.com/downloads> และทำการติดตั้ง
3. ใช้คำสั่ง git config --global user.name "YOUR\_USERNAME" โดยกรอก Username ตามที่สมัคร
4. ใช้คำสั่ง git config --global user.email "your\_email\_address@example.com" โดยกรอก email ตามที่สมัคร
5. ใช้คำสั่ง git config --global --list เพื่อตรวจสอบข้อมูลที่กรอก

```

opor-MacBook-Pro:git1 admin$ git config --global user.name "opor_tan"
opor-MacBook-Pro:git1 admin$ git config --global user.email "jitpanu.tan.0013@gmail.com"
opor-MacBook-Pro:git1 admin$ git config --global --list
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
user.name=opor_tan
user.email=jitpanu.tan.0013@gmail.com
  
```

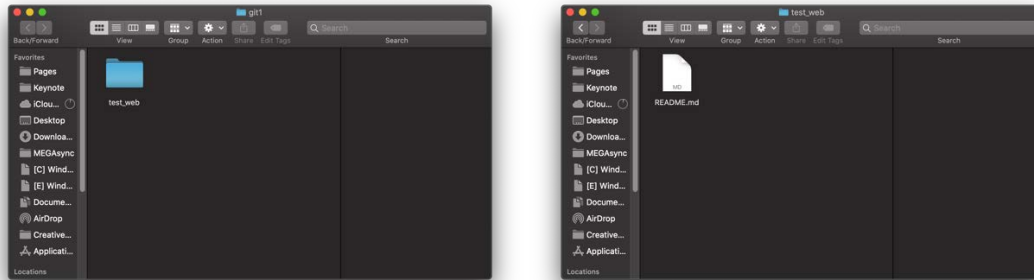
6. ใช้คำสั่ง git init เพื่อสร้างไฟล์ .git ใน folder ที่เราสร้างขึ้นในข้อที่ 1
7. ใช้คำสั่ง git clone HTTPS\_URL โดยใช้ Clone with HTTPS ที่คัดลอกจาก gitlab.com เพื่อ Clone a repository เช่น git clone https://gitlab.com/opor\_tan/test\_web.git

```

[opors-MacBook-Pro:git1 admin$ git init
Initialized empty Git repository in /Users/admin/Desktop/git1/.git/
[opors-MacBook-Pro:git1 admin$ git clone https://gitlab.com/opor_tan/test_web.git
Cloning into 'test_web'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 9 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (9/9), done.

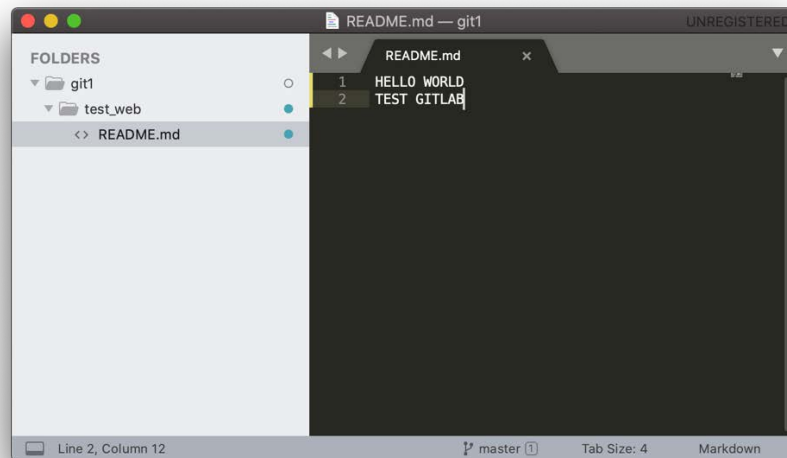
```

เมื่อเราตรวจสอบ Folder ที่สร้างไว้บนเครื่อง จะพบไฟล์ที่สร้างบนเว็บ gitlab.com มาอยู่ภายใน folder



การแก้ไข/อัปเดตไฟล์ในโปรเจกจาก GitLab repository

1. ทำการแก้ไขไฟล์ README.md ด้วย text editor ที่ถนัด เช่น sublime ดังภาพ



2. เมื่อเราเปิด CMD / Terminal ขึ้นมา และใช้คำสั่ง git status เพื่อตรวจสอบการเปลี่ยนแปลง จะพบว่า มีการรายงานว่ามีไฟล์ README.md มีการเปลี่ยนแปลง ดังภาพ

```

[opors-MacBook-Pro:test_web admin$ git commit
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  modified:   README.md

no changes added to commit

```

3. เพื่อทำการ update ไฟล์ต่างๆ บน repository ให้เป็น version ล่าสุด ใช้คำสั่ง `git add <file-name/folder>` เพื่อเลือกไฟล์หรือ folder ที่ต้องการ เช่น `git add README.md`
4. ใช้คำสั่ง `git commit -m "COMMENT TO DESCRIBE THE INTENTION OF THE COMMIT"` เพื่อ commit ไฟล์

```
[opors-MacBook-Pro:test_web admin$ git add README.md
[opors-MacBook-Pro:test_web admin$ git commit -m "Edit file Readme"
[master 6cce585] Edit file Readme
1 file changed, 2 insertions(+)
```

แล้วใช้คำสั่ง `git push origin master` เพื่ออัปโหลดไฟล์ที่แก้ไขขึ้นไปใน gitlab.com

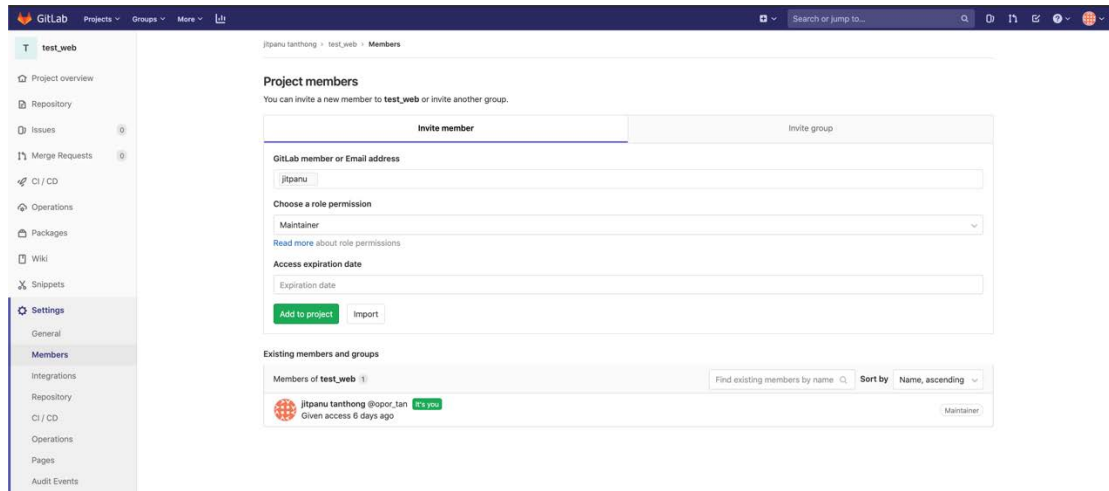
```
[opors-MacBook-Pro:test_web admin$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 274 bytes | 274.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://gitlab.com/opor_tan/test_web.git
98f2b63..6cce585 master -> master
```

เมื่อเราเข้าไปตรวจสอบ ไฟล์ README.md บน gitlab.com จะพบว่าไฟล์ README.md ของเราได้รับการ update แล้ว



## การเพิ่มสมาชิก (Member) ในโปรเจกต์

เนื่องจาก GitLab เป็นเครื่องมือในการจัดการ Code collaboration และ Version control ความสามารถหนึ่งที่ได้ คือ การเพิ่มสมาชิกภายในโปรเจกต์



1. เราสามารถทำการเพิ่มสมาชิกภายในโปรเจกต์เราได้โดย กด Settings -> Members กรอกรหัสของ member ที่ต้องการ แล้วกด Add to Project โดยให้นักศึกษาลองเพิ่มชื่อเพื่อนอย่างน้อย 1 คน เข้าไปในโปรเจกต์ที่ตนเองเป็นเจ้าของอยู่



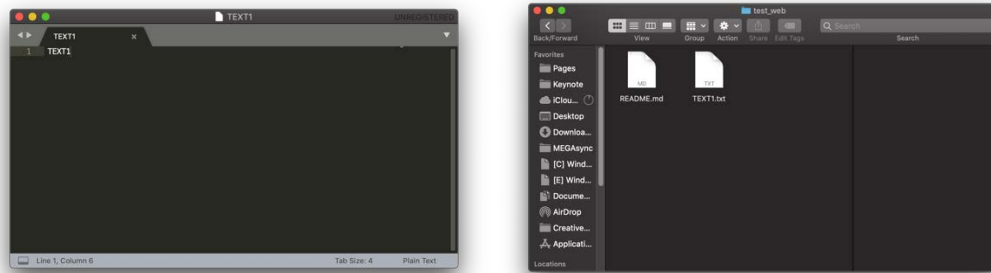
เมื่อทำการเพิ่มสมาชิกแล้ว ใน gitlab.com ของ member ที่เพิ่มเข้าไป จะมีโปรเจกต์เพิ่มเข้ามา



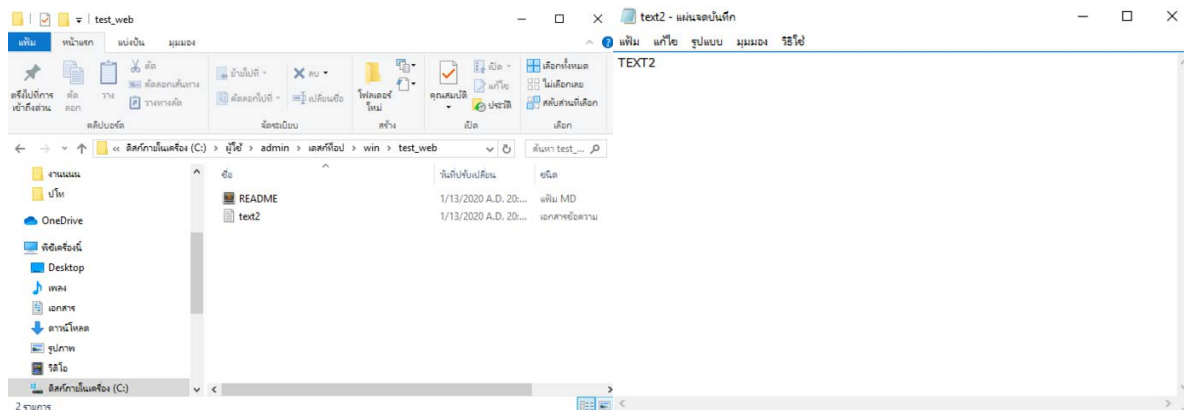
2. ทำการ clone โปรเจกต์ที่เพิ่มเข้ามาลงในคอมพิวเตอร์ สำหรับ user ที่เป็นคนเพิ่ม member ไม่จำเป็นต้อง clone ซ้ำ

## การเพิ่ม/แก้ไขไฟล์ในโปรเจกต์โดยสมาชิก (Merge Commit)

1. ให้ทางฝั่งเจ้าของโปรเจกต์ให้เพิ่มไฟล์ text1.txt ในโปรเจกต์ที่สร้างขึ้น โดยมีข้อความ “TEXT1” ดังภาพ



2. ทางฝั่ง member ให้เพิ่มไฟล์ text2.txt ในโปรเจกต์เดียวกัน โดยมีข้อความ “TEXT2” ดังภาพ



3. ให้ฝั่ง member ทำการ commit และ push ไฟล์ text2.txt ขึ้น repository
4. ให้เจ้าของโปรเจกต์ ทำการ commit และ push ไฟล์ text1.txt ขึ้น repository ตามลำดับ จะพบว่าเกิด error ขึ้นเนื่องจาก repository ประกอบไปด้วยไฟล์ text2.txt ที่ถูก push โดย member ที่ไม่ปรากฏอยู่ใน local repository ของเจ้าของโปรเจกต์

```
opors-MacBook-Pro:test_web admin$ git push origin master
To https://gitlab.com/opor_tan/test_web.git
! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://gitlab.com/opor_tan/test_web.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

5. การแก้ไข ให้เจ้าของโปรเจกต์ ใช้คำสั่ง git pull origin master เพื่ออัปเดตไฟล์ในโปรเจกต์ให้ใหม่ล่าสุดก่อน แล้วจึงค่อยทำการ commit และ push ไฟล์ทางฝั่งเจ้าของขึ้นไป

```

opors-MacBook-Pro:test_web admin$ git add TEXT1.txt
[opors-MacBook-Pro:test_web admin$ git commit -m "add text1"
[master 48f3f18] add text1
[opors-MacBook-Pro:test_web admin$ git push origin master
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 596 bytes | 596.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0)
To https://gitlab.com/opor_tan/test_web.git
d57e5b4..48f3f18 master -> master

```

6. เช่นเดียวกับกับทางฝั่ง member ใช้คำสั่ง git pull origin master เพื่ออัปเดตโปรเจกต์ภายในคอม จะพบไฟล์ที่ทางฝั่งเจ้าของโปรเจกต์อัปเดตเข้ามา

The image shows a GitLab repository named 'test\_web' and a local file explorer window. The GitLab interface displays the 'master' branch with a commit 'add text1' by 'jipanu tanthong'. The file explorer shows the local directory 'test\_web' containing files 'README', 'TEXT1', and 'text2'.

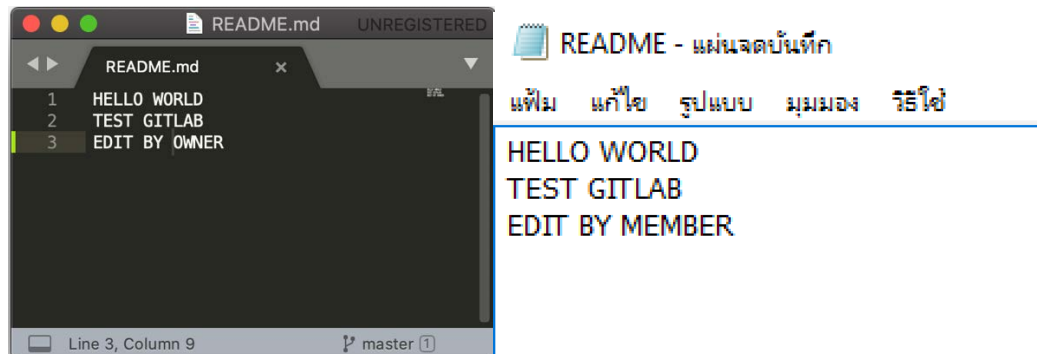
Name	Last commit	Last update
README.md	Edit file Readme	5 days ago
TEXT1.txt	add text2	15 minutes ago
text2.txt	add text1	5 minutes ago

The README.md file content is visible as 'HELLO WORLD TEST GITLAB'.



## กรณีที่เกิดไฟล์เดียวกัน (Conflict)

หากมีสมาชิกในโปรเจกมากกว่า 1 คนทำการแก้ไขไฟล์เดียวกัน ในเวลาเดียวกัน ดังตัวอย่าง หาดเจ้าของโปรเจกและ Member ทำการแก้ไขไฟล์ README.md ใน local repository ของตนเอง ดังนี้



1. หากเจ้าของโปรเจกทำการ commit , push ไฟล์ README.md ที่แก้ไขแล้วก่อน

```

[opors-MacBook-Pro:test_web admin$ git add README.md
[opors-MacBook-Pro:test_web admin$ git commit -m "edit readme"
[master 6c28bee] edit readme
1 file changed, 2 insertions(+), 1 deletion(-)
[opors-MacBook-Pro:test_web admin$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 345 bytes | 345.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://gitlab.com/opor_tan/test_web.git
48f3f18..6c28bee master -> master
opors-MacBook-Pro:test_web admin$

```

<a href="#">README</a> <a href="#">Add LICENSE</a> <a href="#">Add CHANGELOG</a> <a href="#">Add CONTRIBUTING</a> <a href="#">Enable Auto DevOps</a>		
<a href="#">Add Kubernetes cluster</a> <a href="#">Set up CI/CD</a>		
Name	Last commit	Last update
<a href="#">README.md</a>	edit readme	just now
<a href="#">TEXT1.txt</a>	add text2	25 minutes ago
<a href="#">text2.txt</a>	add text1	15 minutes ago
<a href="#">README.md</a>		
HELLO WORLD TEST GITLAB EDIT BY OWNER		





2. ต่อมาให้ member ลอง commit และ push ไฟล์ README.md ของตนเองขึ้น repository จะพบว่า จะเกิด error ขึ้นดังภาพ เนื่องจาก README.md ถูก update ไปก่อนหน้านี้แล้ว

```
C:\Users\admin\Desktop\win\test_web>git add README.md
C:\Users\admin\Desktop\win\test_web>git commit -m "edit readme"
[master 3253a60] edit readme
1 file changed, 2 insertions(+), 1 deletion(-)
C:\Users\admin\Desktop\win\test_web>git push origin master
To https://gitlab.com/opor_tan/test_web.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://gitlab.com/opor_tan/test_web.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

3. ให้ฝั่ง member ใช้ คำสั่ง git pull origin master เพื่อเป็นการ pull README.md เวอร์ชันล่าสุดลงมายัง local repository ของตนเอง


```
C:\Users\admin\Desktop\win\test_web>git pull origin master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://gitlab.com/opor_tan/test_web
 * branch            master       -> FETCH_HEAD
  48f3f18..6c28bee    master       -> origin/master
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
```

4. จะสังเกตได้ว่า บน online repository ไฟล์ README ยังเป็นไฟล์ที่เจ้าของโปรเจกต์อัปเดตขึ้นไป

Name	Last commit	Last update
 README.md	edit readme	28 minutes ago
 TEXT1.txt	add text2	52 minutes ago
 text2.txt	add text1	43 minutes ago
 README.md		
HELLO WORLD TEST GITLAB EDIT BY OWNER		

อย่างไรก็ตาม หาก member ทำการเปิดไฟล์ README ของตนเองจะพบไฟล์ README หน้าตาดังตัวอย่าง

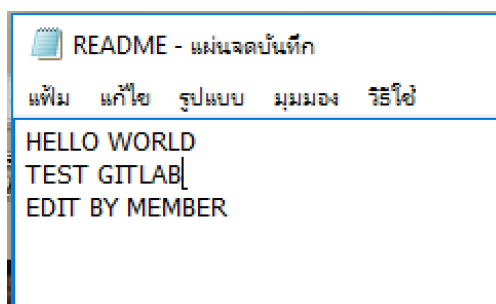
```

 README - แหม่นจตบันเท็ก
แก้ไข  รูปแบบ  มุมมอง  วิดีโอ
-----
HELLO WORLD
TEST GITLAB
<<<<<<< HEAD
EDIT BY MEMBER
=====
EDIT BY OWNER
>>>>>>> 6c28bee7729eac3ec038d4fd15131714a430e926
|
```

ให้สังเกตรูปแบบของเครื่องหมาย <<<, == และ >>>

- โค้ดที่อยู่ระหว่าง <<< และ == คือโค้ดที่ member เป็นคนแก้ไข







- โค้ดที่อยู่ระหว่าง === และ >>> คือโค้ดเจ้าของโปรเจกต์เป็นคนแก้ไข ส่วนตัวเลขต่อท้ายคือหมายเลขของ Commit ที่ทำการ Merge
- เพื่อให้โค้ดทำงานได้เหมาะสม member ก็จะต้องเอาโค้ดที่ตัวเองแก้ไข ไปรวมกับโค้ดของเจ้าของโปรเจกต์เพื่อให้ทำงานได้ เช่น ทำการลบข้อความที่เจ้าของโปรเจกต์แก้ไขออก ดังภาพ



5. หลังจากทำงานแก้ไขโปรแกรมเสร็จแล้ว member สามารถ ทำการ push , commit ได้ตามปกติ

```
C:\Users\admin\Desktop\win\test_web>git add README.md
C:\Users\admin\Desktop\win\test_web>git commit -m "edit readme"
[master b4298b3] edit readme
C:\Users\admin\Desktop\win\test_web>git push origin master
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 679 bytes | 339.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0)
To https://gitlab.com/opor_tan/test_web.git
6c28bee..b4298b3 master -> master
```

และจะพบว่า ไฟล์บน online repository จะกลายเป็นไฟล์เวอร์ชันล่าสุดที่ member เป็นผู้แก้ไข

 <b>edit readme</b> opor_tan2 authored 1 minute ago <span style="float: right;">b4298b39 </span>		
Name	Last commit	Last update
 README.md	edit readme	1 minute ago
 TEXT1.txt	add text2	1 hour ago
 text2.txt	add text1	51 minutes ago
 <b>README.md</b> HELLO WORLD TEST GITLAB EDIT BY MEMBER		

**Conflict** ถือว่าเป็นเหตุการณ์ที่เกิดขึ้นได้เป็นปกติ โดยเฉพาะอย่างยิ่งโปรเจกต์ใหญ่ๆที่มี Developer หลายคนช่วยกันรุมเขียนโค้ด ดังนั้นการแก้ Conflict จึงเป็นหนึ่งในพื้นฐานของการใช้งาน Git ที่นักพัฒนาต้องเข้าใจและจัดการกับมันได้ ไมเช่นนั้นจะเกิดปัญหาอย่างเช่น เผลอไปลบโค้ดของเพื่อน โดยไม่สนใจอะไร เพื่อให้ Conflict หายไป เป็นต้น

สรุปขั้นตอนของการแก้ Conflict มีดังนี้

- Pull จาก Remote ลงเครื่อง
- Conflict เกิดขึ้น
- แก้ไขโค้ดให้เหมาะสม
- Merge Commit
- Push ขึ้น Remote

### แบบฝึกหัด

1. ให้นักศึกษาจับกลุ่ม 3 คน และสร้างใหม่ repository สำหรับทำ Final project โดยกำหนดให้ นศ. 1 คน เป็นเจ้าของ repository และทำการเพิ่มสมาชิกในกลุ่มเข้าไปใน repository ดังกล่าว
2. สร้าง README.md โดยให้ระบุ ชื่อ – นามสกุล รหัสนักศึกษา รวมทั้ง username ของสมาชิกแต่ละคนภายในกลุ่ม
3. ให้นักศึกษาแต่ละคน สร้างไฟล์ responsibility\_6007xxxx.txt (ตามด้วยรหัสนักศึกษาของตนเอง) ไว้จดบันทึกหน้าที่/ความรับผิดชอบของตนเองในโปรเจกต์ ใน local repository ของตนเอง แล้ว push ขึ้นไปยัง main repository
4. ให้แต่ละกลุ่ม สร้างไฟล์ requirement.txt สำหรับเก็บ functional/content requirement ของโปรเจกต์ โดยนศ.ทุกคนจะต้องแก้ไข/เพิ่มเติมข้อความในไฟล์นี้อย่างน้อย คนละ 1 ครั้ง

### เอกสารเพิ่มเติม

Ake Exorcist, มาเรียนรู้ Git แบบง่ายๆกันเถอะ,

<https://blog.nextzy.me/%E0%B8%A1%E0%B8%B2%E0%B9%80%E0%B8%A3%E0%B8%B5%E0%B8%A2%E0%B8%99%E0%B8%A3%E0%B8%B9%E0%B9%89-git-%E0%B9%81%E0%B8%9A%E0%B8%9A%E0%B8%87%E0%B9%88%E0%B8%B2%E0%B8%A2%E0%B9%86%E0%B8%81%E0%B8%B1%E0%B8%99%E0%B9%80%E0%B8%96%E0%B8%AD%E0%B8%B0-427398e62f82>