**ใบงานที่ 2**

**Git and GitHub**

**ชื่อ - นามสกุล..........นายศรสิวะพงษ์...โพธิวงศ์......รหัสนักศึกษา.................2661031741113............................**

**ทฤษฎีที่เกี่ยวข้อง**

Git คือ ซอร์ฟแวร์ตัวหนึ่งที่เราสามารถติดตั้งในเครื่องเราได้และจำเป็นต้องมีหากเราต้องการใช้งานคุณสมบัติของมัน หน้าที่หลักของ Git คือการทำส่งที่เรียกว่า “ระบบควบคุมเวอร์ชั่น”(Version Control) นั้คือ Git จะคอยดูแลการเปลี่ยนแปลงของโปรเจคงานของเราในภาพรวมทั้งหมดเช่นถ้าเรากำหนดให้มันดูแลงานใน Folder WorkShop1 อะไรก็ตามที่มีการเปลี่ยนแปลงไปจะถูก Git ดูแลให้ เช่น สร้างประวัติการแก้ไข, ย้อนดูโค้ดเก่า, แตกสาขาเพื่อทดลองฟีเจอร์ใหม่ๆ ได้ทั้งหมดบนเครื่องของเราเอง โดยไม่จำเป็นต้องเชื่อมต่ออินเทอร์เน็ต การกระทำหลักใน Git คือการ "commit" ซึ่งเปรียบเสมือนการสร้าง "จุดเซฟ" (save point) ให้กับโค้ดที่เรากำลังทำงานด้วย ณ เวลานั้นๆ

GitHub คือ แพลตฟอร์มบนเว็บไซต์ (Website/Service) ที่ให้บริการ "โฮสติ้งสำหรับ Git" ซึ่งทำหน้าที่เก็บไฟล์งานของเราบนโลกออนไลน์ดังนั้นเราต้องมีอินเทอร์เน็ตเพื่อเข้าถึง GitHub เพื่ออัปโหลด (push) โค้ดที่ของเราที่กดบันทึกไว้บนเครื่องเราเองและบันทึกไปที่ GitHub นอกจากนี้เรายังสามารถดาวน์โหลด (pull) โค้ดเก่าๆ ที่เราส่งไปเก็บไว้ใน GitHub ลงมาที่เครื่องเราได้ตลอดด้วย และยังสามารถดาวน์โหลดโค้ดของคนอื่นที่ได้เผยแพร่ไว้ลงมาทดลองหรือพัฒนาต่อก็ได้เช่นกัน

ตัวอย่าง GitHub ฟีเจอร์ที่ส่งเสริมการทำงานร่วมกัน เช่น:

* **Pull Requests:** ช่องทางสำหรับเสนอการแก้ไขโค้ดให้เจ้าของโปรเจกต์พิจารณา
* **Issues:** ใช้ติดตามบั๊ก, สิ่งที่ต้องทำ, หรือข้อเสนอแนะต่างๆ
* **Actions:** ระบบอัตโนมัติสำหรับทดสอบและนำโค้ดไปใช้งาน (CI/CD)
* **Social Coding:** สามารถติดตาม (follow) นักพัฒนาคนอื่น, ให้ดาว (star) โปรเจกต์ที่ชอบ และเป็น Portfolio โชว์ผลงานของเราได้

**ตารางเปรียบเทียบ Git vs GitHub**

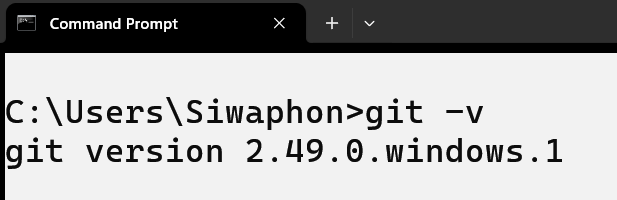
|  |  |  |
| --- | --- | --- |
| คุณสมบัติ | Git | GitHub |
| ประเภท | ซอฟต์แวร์ (Software) | แพลตฟอร์ม/บริการบนเว็บไซต์ (Platform/Service) |
| หน้าที่หลัก | ระบบควบคุมเวอร์ชัน (Version Control) | โฮสติ้งสำหรับ Git Repository (Code Hosting) |
| สถานที่ทำงาน | บนเครื่องคอมพิวเตอร์ของผู้ใช้ (Local) | บนเซิร์ฟเวอร์คลาวด์ (Remote/Cloud) |
| การทำงาน | ทำงานแบบ Offline ได้ | ต้องเชื่อมต่ออินเทอร์เน็ตเพื่อใช้งาน |
| ผู้พัฒนา/เจ้าของ | Linus Torvalds (เป็น Open Source) | Microsoft Corporation |
| การใช้งานหลัก | ผ่าน Command Line หรือโปรแกรม GUI | ผ่านหน้าเว็บไซต์ (Web Interface) |
| เป้าหมาย | ติดตามการเปลี่ยนแปลงของโค้ดและจัดการประวัติ | ทำงานร่วมกับผู้อื่น, สำรองโค้ด, และแสดงผลงาน |
| ความจำเป็น | สามารถใช้ Git โดยไม่มี GitHub ได้ | แทบจะเป็นไปไม่ได้ที่จะใช้ GitHub โดยไม่มี Git |
| ฟีเจอร์เด่น | commit, branch, merge, rebase | Pull Requests, Issues, Actions, Fork, Social Coding |

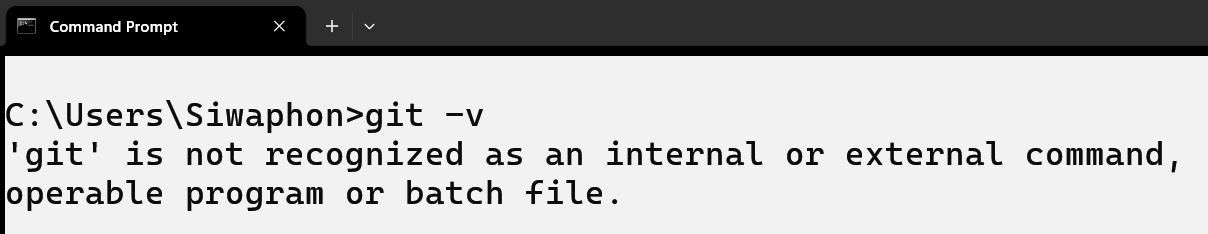
Git และ GitHub ไม่ใช่แค่ "เครื่องมือ" แต่เป็น "ทักษะพื้นฐาน" ที่นักพัฒนาทุกคนต้องมีในปัจจุบันเพราะนิยมใช้ในการบริหารจัดการ การพัฒนาโปรเจคที่มีการแยกการทำงานโดยนักพัฒนาหลายคน ทั้งนักพัฒนาส่วนใหญ่มักใช้ GitHub เพื่อวัตถุประสงค์ดังต่อไปนี้

* **เพื่อให้เป็นที่ยอมรับในวงการ:** บริษัทส่วนใหญ่ใช้ Git และ GitHub เป็นเครื่องมือหลักในการพัฒนาซอฟต์แวร์ การใช้งานเป็นจึงเป็นเหมือนใบเบิกทางในการสมัครงาน
* **เพื่อสร้างโปรไฟล์ให้โดดเด่น:** การมีโปรไฟล์ GitHub ที่มีโปรเจกต์น่าสนใจและมีการเคลื่อนไหวอยู่เสมอ จะช่วยให้เราโดดเด่นกว่าผู้สมัครคนอื่นๆ
* **เพื่อเรียนรู้และเติบโต:** การเข้าไปมีส่วนร่วมในโปรเจกต์ Open Source บน GitHub เป็นวิธีที่ดีที่สุดในการเรียนรู้จากโค้ดของโปรแกรมเมอร์เก่งๆ ทั่วโลกและพัฒนาฝีมือของตัวเอง

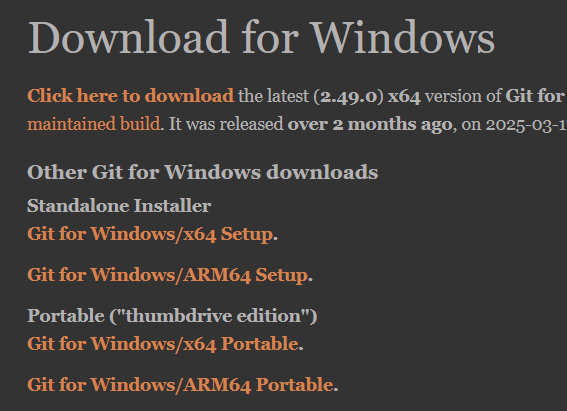
**ภาคปฎิบัติ**

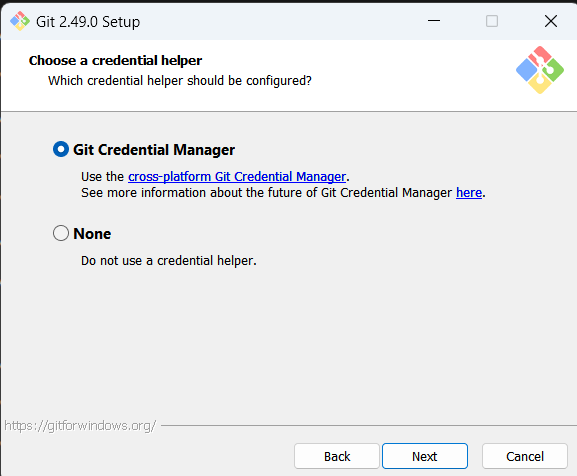
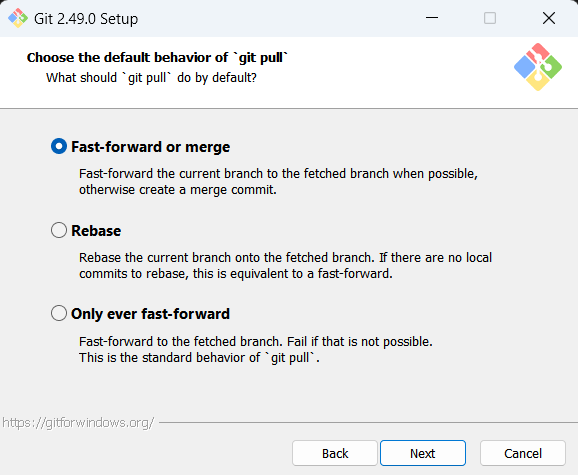
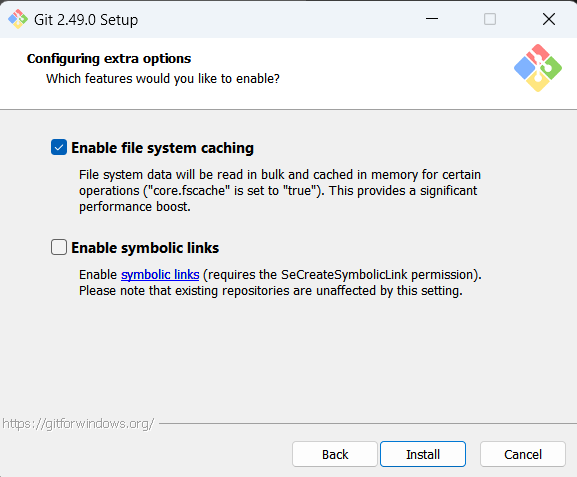
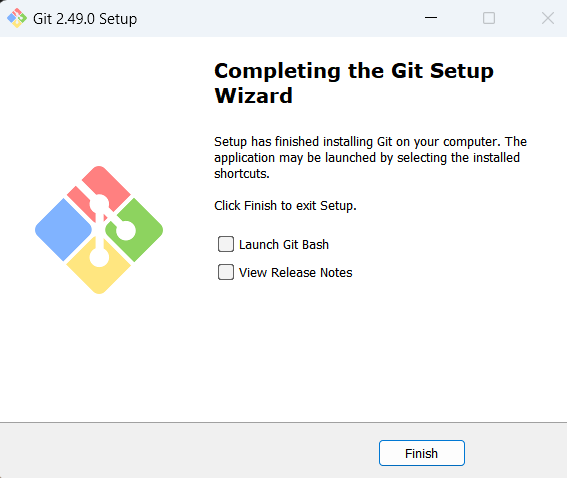
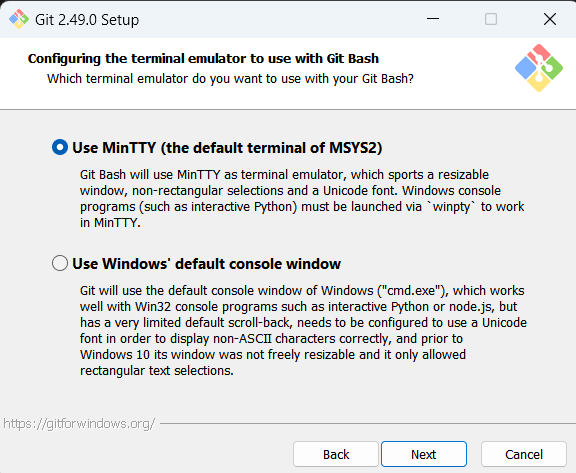
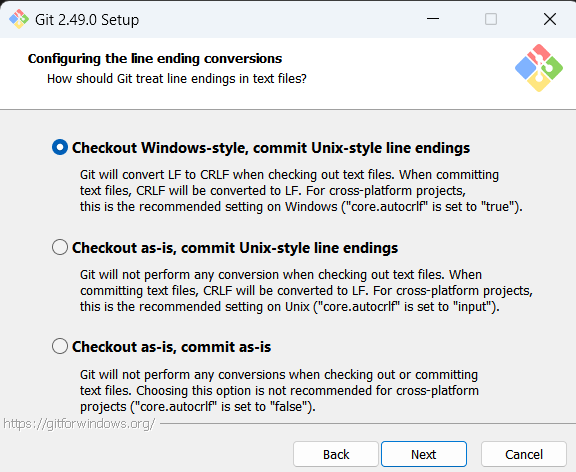
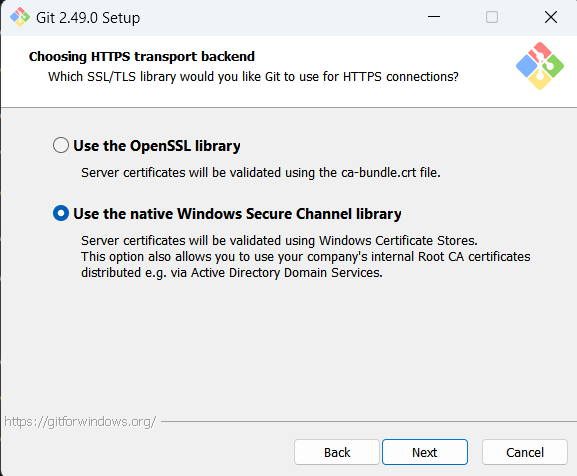
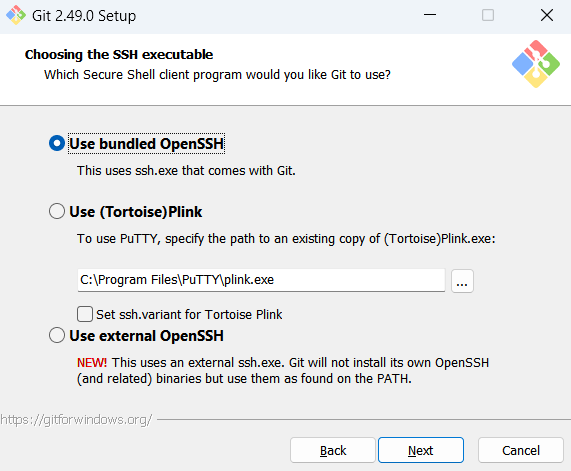
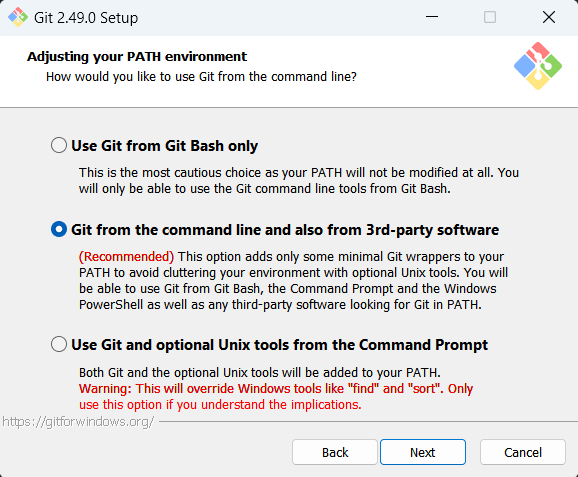
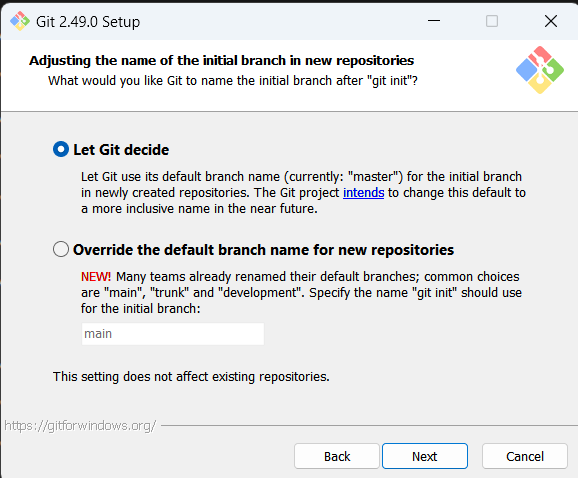
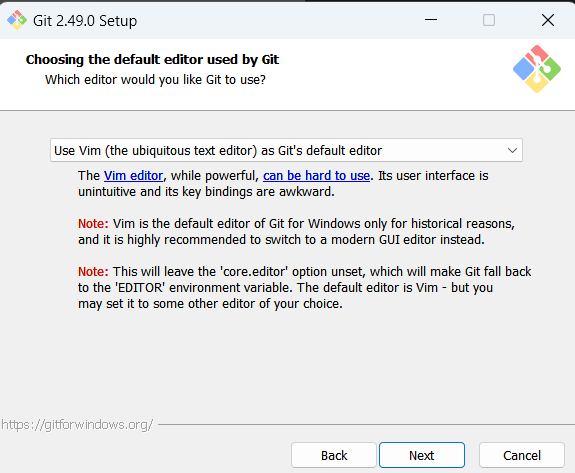
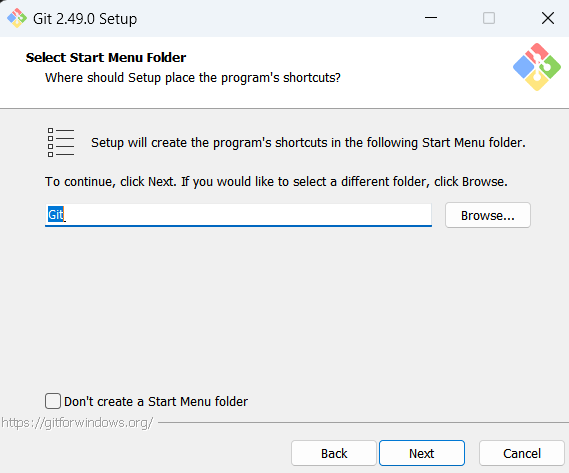
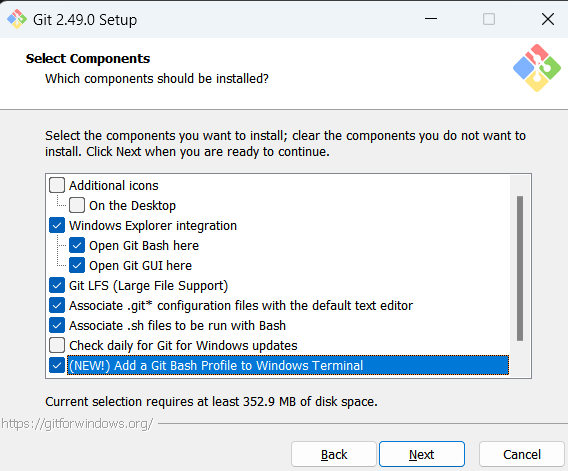
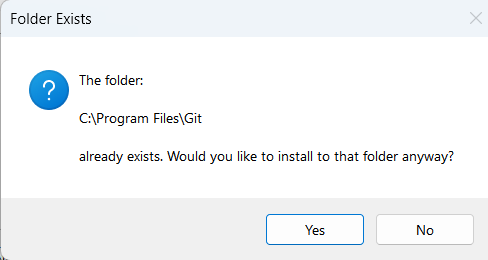
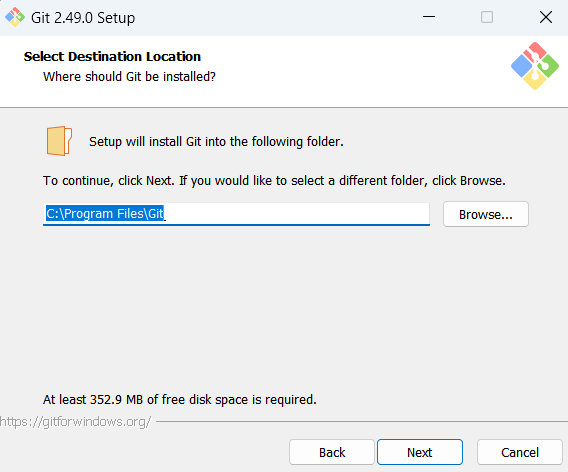
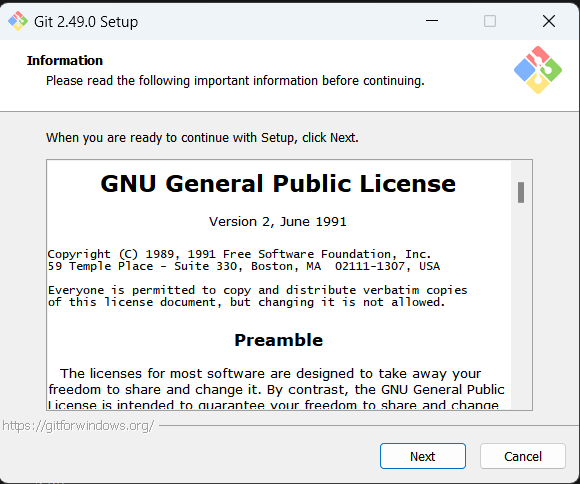
1. การตรวจสอบและติดตั้ง Git หากใครได้ทำใบงาน “Mobile Application Development by Flutter” ผ่านมาแล้วจะมีการให้ติดตั้ง Git ที่ดาวน์โหลดจากเว็บ [https://git-scm.com/ ไป](https://git-scm.com/%20%20ไป)แล้วในกรณีที่ต้องการตรวจสอบให้เปิด Terminal ขึ้นมาแล้วพิมพ์คำสั่ง git -v หากให้ผลลัพท์เป็น Version ออกมานั้นหมายถึงได้มีการติดตั้ง Git ในเครื่องเราแล้ว



แต่ในกรณีที่ไม่พบ Git ในเครื่องเราจะแสดงข้อความแบบนี้ หมายความว่ายังไม่ได้ติดตั้ง Git ไว้ในเครื่องเราให้ทำการดาวน์โหลดและติดตั้งในเรียบร้อย

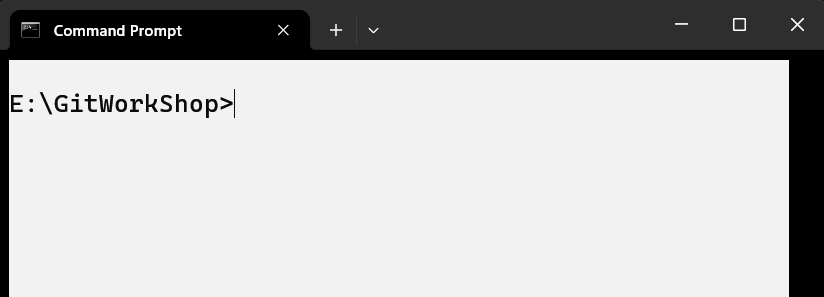
**โดยมีขั้นตอนการดาวน์โหลดและติดตั้งดังนี้**

* 1. Download ที่ <https://git-scm.com/downloads> เลือก Windows เลือก Standalone Installer และเลือกรุ่น 32-bit หรือ 64-bit ตาม Windows ของเรา
  2. ติดตั้งให้เรียบร้อย

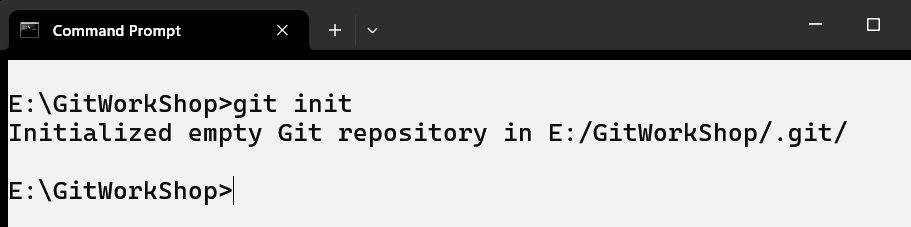


1. ในการใช้งาน Git เราจะต้องบอกให้ Git รู้ก่อนว่าเราจะให้มันค่อยดูแลงานเราที่ Folder ไหนโดยเมื่อเราสร้างโปรเจคขึ้นมาด้วยภาษาใดก็ตาม หรือสร้างด้วยคำสั่งของ Window ตามปกติก็ตามเมื่อเราต้องการจะบอก Git ให้ดูแล Folder นี้เราจะต้องใช้คำสั่ง git init ใน terminal โดยต้องอยู่ใน path ของ Folder ที่เราต้องการ โดยมีขั้นตอนตามนี้

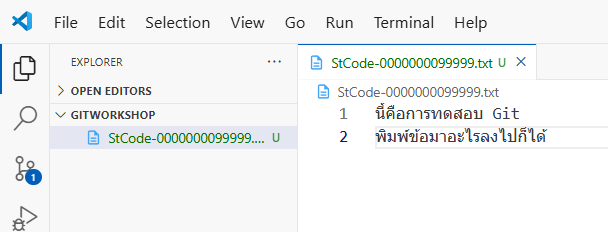
2.1 สร้าง Folder ขึ้นมาสำหรับทดสอบ Git ชื่อ GitWorkShop เข้าไปใน Folder GitWorkShop คลิ๊กขวาที่พื้นที่ว่างเลือกเมนู Open in Terminal สังเกตุว่าจะระบุชื่อ Folder ของเรา(GitWorkShop)ใน Path ของ Command



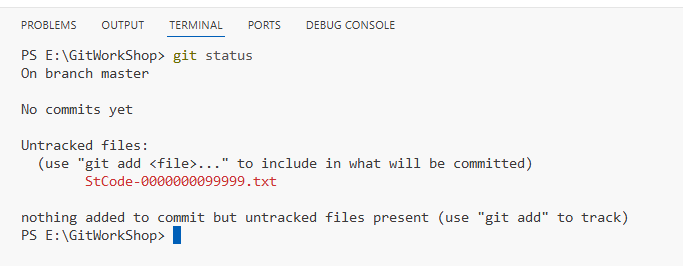
2.2 บอกให้ Git รู้ว่าจะทำงานกับ Folder นี้โดยพิมพ์คำสั่ง git init ใน Terminal



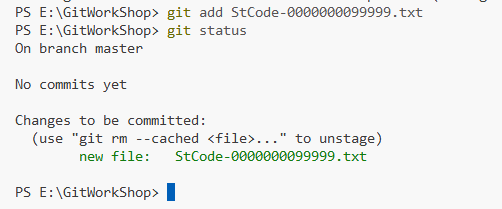
2.3 เปิด Folder GitWorkShop นี้ใน VScode ซึ่งจะเปิดผ่าน Terminal ก็ได้โดยพิมพ์คำสั่ง code . หรือเปิดแบบปกติผ่านโปรแกรม VScode ก็ได้ตามที่ถนัด จากนั้นให้สร้างไฟล์ “StCode-รหัสนักศึกษา.txt” จากนั้นพิมพ์ข้อความอะไรลงไปก็ได้หรือไม่พิมพ์ก็ได้

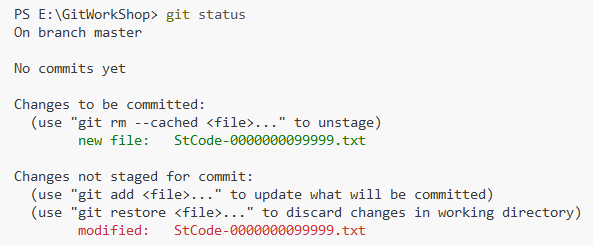


2.4 เปิด Terminal ใน VScode หรือใช้ Command เดิมที่เคยเปิดก็ได้พิมพ์คำสั่ง git status สั่งเกตุข้อความ “Untracked files” เนื่องจาก Git พบว่าเรายังไม่เคย Add ไฟล์เข้ามาทีดังนั้นไฟล์ที่ทำงานอยู่จึงยังไม่พบว่ามีการเปลี่ยนแปลง

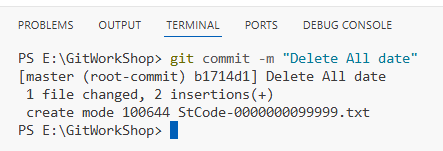


2.5 เพิ่มไฟล์ให้ Git โดยใช้คำสั่ง git add ตามด้วยชื่อไฟล์(หากใช้ \*.\* หมายถึงทุกไฟล์) จากนั้นใช้คำสั่ง git status อีกครั้งสังเกตุความแตกต่างจากข้อ 2.4 นั้นหมายถึงตอนนี้ Git ได้คอยติดตามดูการเปลี่ยนแปลงของไฟล์ที่เรากำหนดแล้ว



2.6 ทดลองเพิ่มข้อความในไฟล์ หรือลบข้อความออกจากไฟล์ บันทึกไฟล์และใช้คำสั่ง git status อีกครั้ง จะพบว่าตอนนี้ Git สามารถตรวจสอบได้แล้วว่ามีการเปลี่ยนแปลงเกิดขึ้นกับไฟล์ที่เรากำหนด

2.7 ตอนนี้การเปลี่ยนแปลงที่เกิดขึ้นที่ Git ตรวจสอบได้แต่ยังไม่ได้ถูกจัดเก็บแบบถูกต้องหมายถึงตอนนี้ข้อมูลที่มีการเปลี่ยนแปลงจะถูกจัดเก็บในส่วน Staging changes ตามกระบวนการทำงานของ Git เพียงแต่ Git ตรวจสอบได้ว่ามีการเปลี่ยนแปลงเท่านั้น สำหรับไฟล์ต้นฉบับนั้นจะมีการบันทึกแล้วจริงๆ แต่การใช้คุณสมบัติของ Git ยังไม่ครบหากยังไม่พบคำสั่ง Commit ดังนั้นหากต้องการให้ Git ทำงานได้ครบจริงเราจะต้องใช้คำสั่ง git commit -m “ในส่วนนี้ใส่คำอธิบาย”

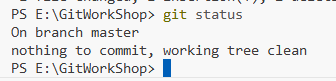


ค่า Has ของไฟล์

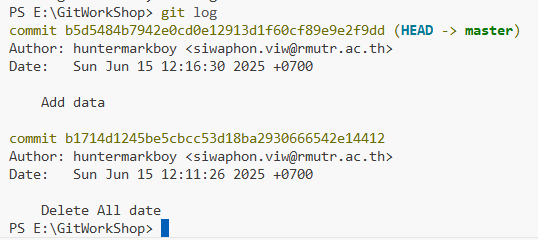
ใช้ตรวจสอบการ

เปลี่ยนแปลง

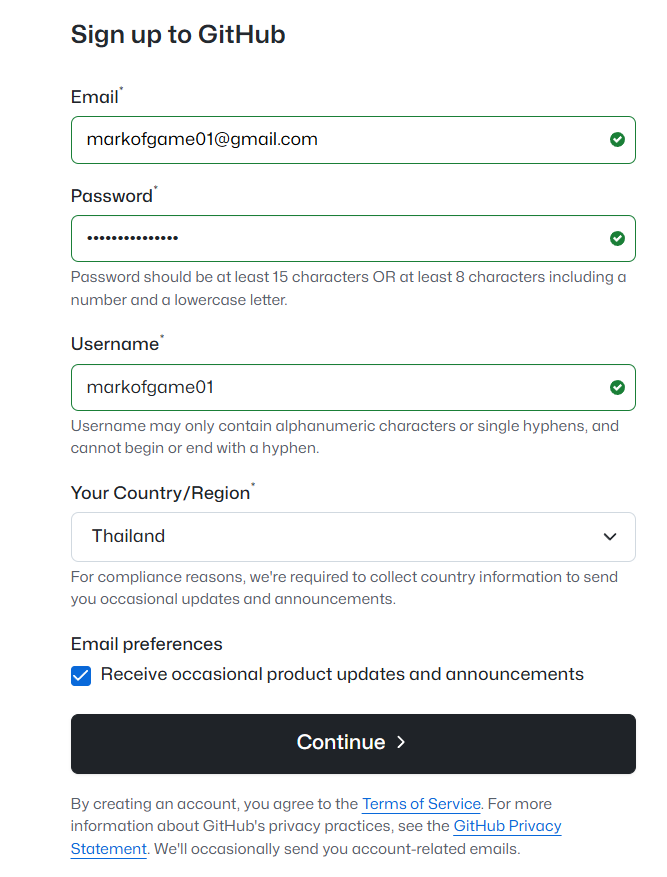
เมื่อเราใช้คำสั่ง git status จะพบข้อความ nothing to commit หมายถึงไม่พบการเปลี่ยนแปลงอะไรใหม่



2.8 เราสามารถตรวจสอบการกระทำที่ผ่านมาทั้งหมดได้โดยใช้ git log จะพบการใช้คำสั่งเรืองตามวันเดือนปี เวลา ตรงนี้แต่ละคนอาจไม่เหมือนกันขึ้นอยู่กับว่าทำอะไรไปบ้าง ซึ่งจะเห็นว่าข้อความคอมเม้นที่เราใส่เข้าไปจะเป็นส่วนสำคัญที่จะบอกว่าเราทำอะไรลงไปบ้าง



1. การติดตั้ง GitHub Desktop มาเป็นตัวช่วยในการใช้งาน Git ซึ่งจะต้องสมัคร Account ก่อนโดยเข้าไปที่ https://github.com/ เลือก Sign Up

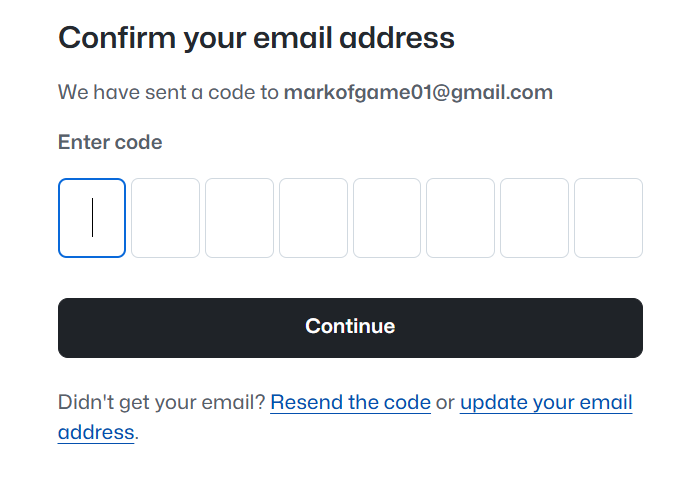
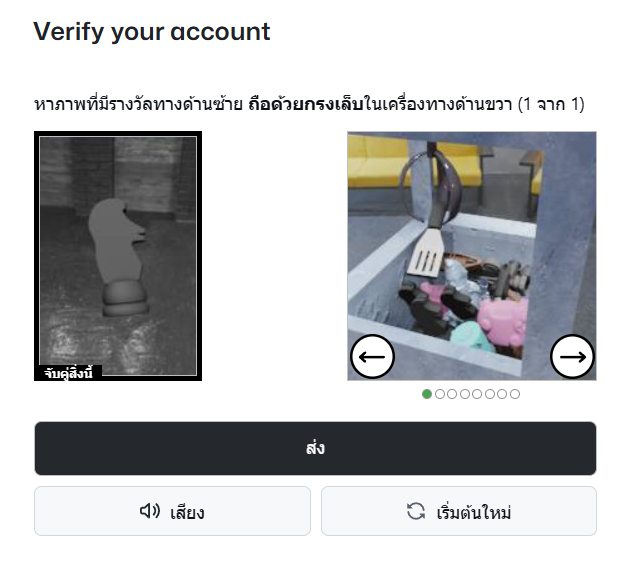
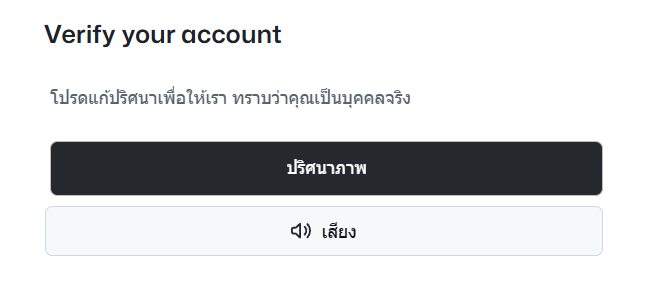


ใส่ Email ที่ใช้งานได้จริง

กำหนดรหัสผ่าน 8 ตัวขึ้นไป ประกอบด้วย อักษรตัวใหญ่+ตัวเล็ก+ตัวเลข หรืออักรพิเศษและต้องไม่อยู่ในดิกชั่นนารีระบบ

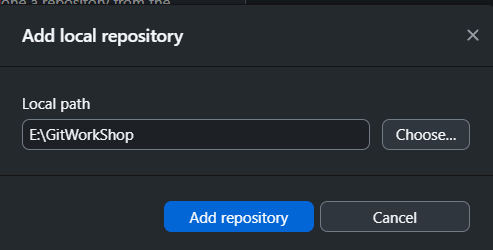
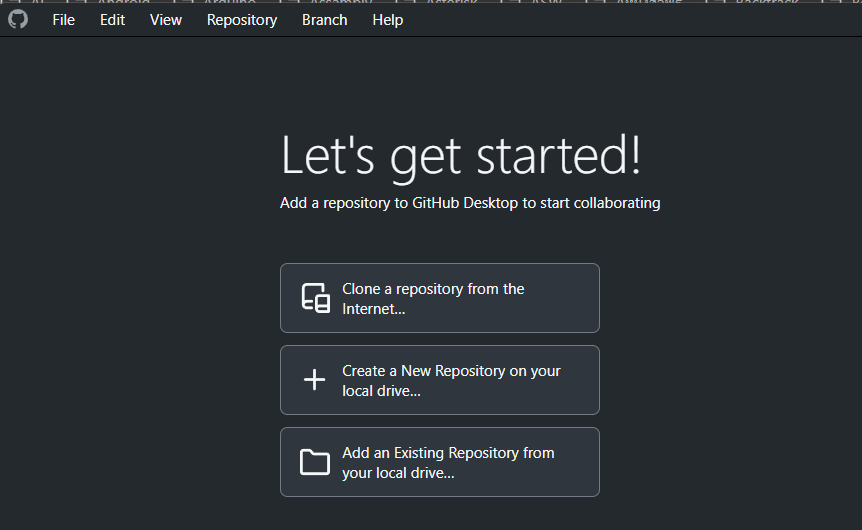
กำหนด Username ในการเข้าระบบใช้ได้ทั้ง Use และ Email

1. การยืนยันระบบจะให้เลือกจะยืนยันด้วยภาพหรือเสียง แนะนำให้เลือกยืนยันด้วยภาพ



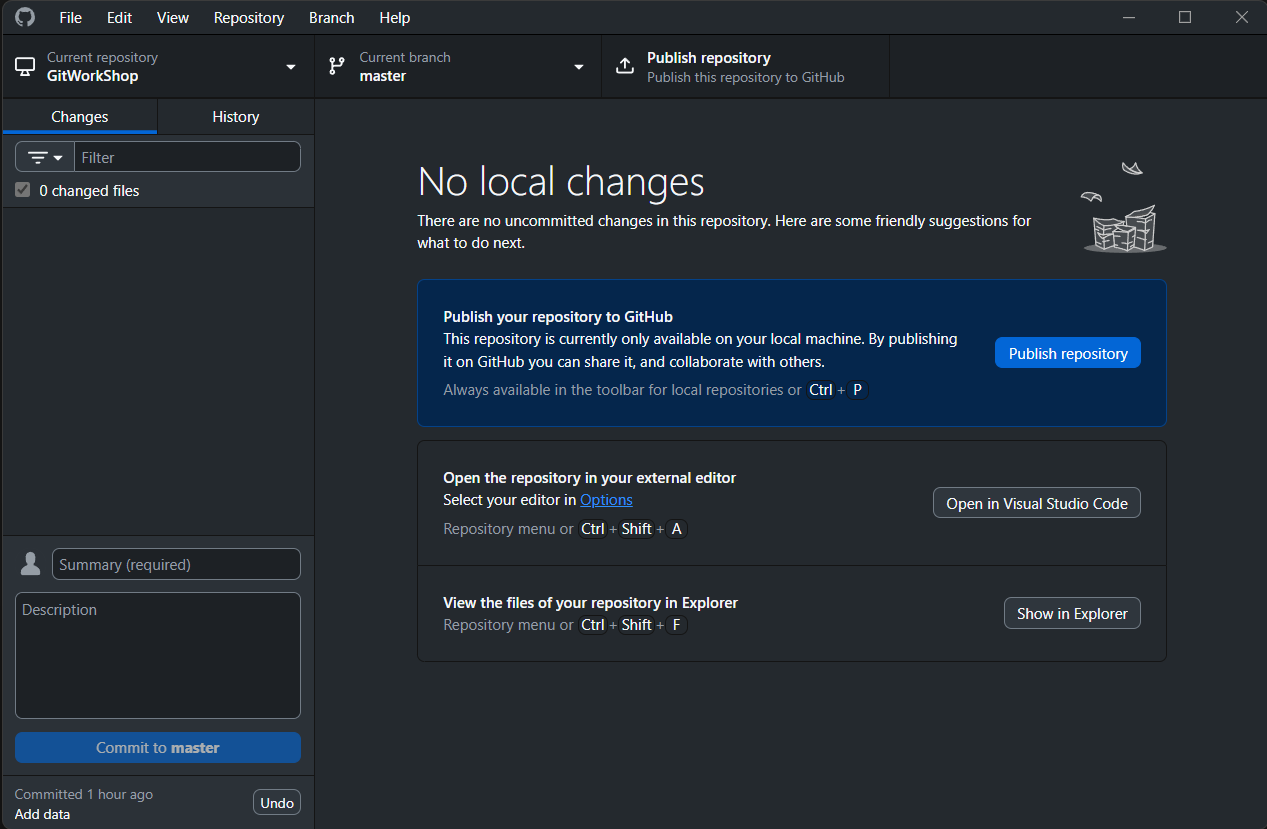
ระบบจะส่งรหัสยืนยันไปที่ Email ของเราที่ใช้สมัครนำมากรอกให้ถูกต้อง

1. จะแสดงหน้าเข้าระบบให้ใส่ Username หรือ Email และรหัสผ่าน เมื่อเข้าระบบสำเร็จให้เลื่อนข้อความเพจมาด้านล่างสุดจะพบส่วนของการดาวน์โหลด GitHub Desktop ให้ทำการดาวน์โหลดและติดตั้งให้เรียบร้อย
2. เมื่อเปิดโปรแกรมขึ้นมาจะพบเมนู 3 -4 เมนูให้เลือกเมนู “Add an Exiting Repository from your local drive” เพราะเราได้สร้าง Folder งานไว้ก่อนหน้าแล้วก็คือ GitWorkShop



เลือก Path ไปยัง GitWorkShop

1. จะพบหน้าแรกของ GitHub Desktop

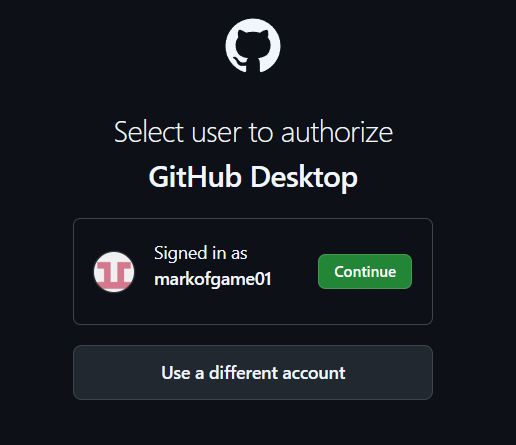
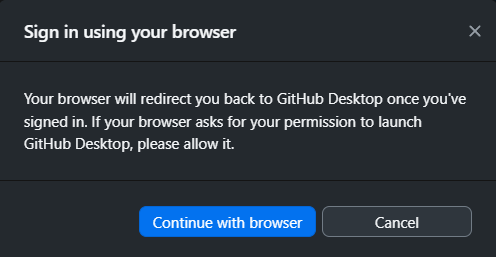
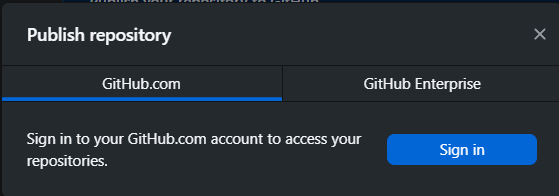
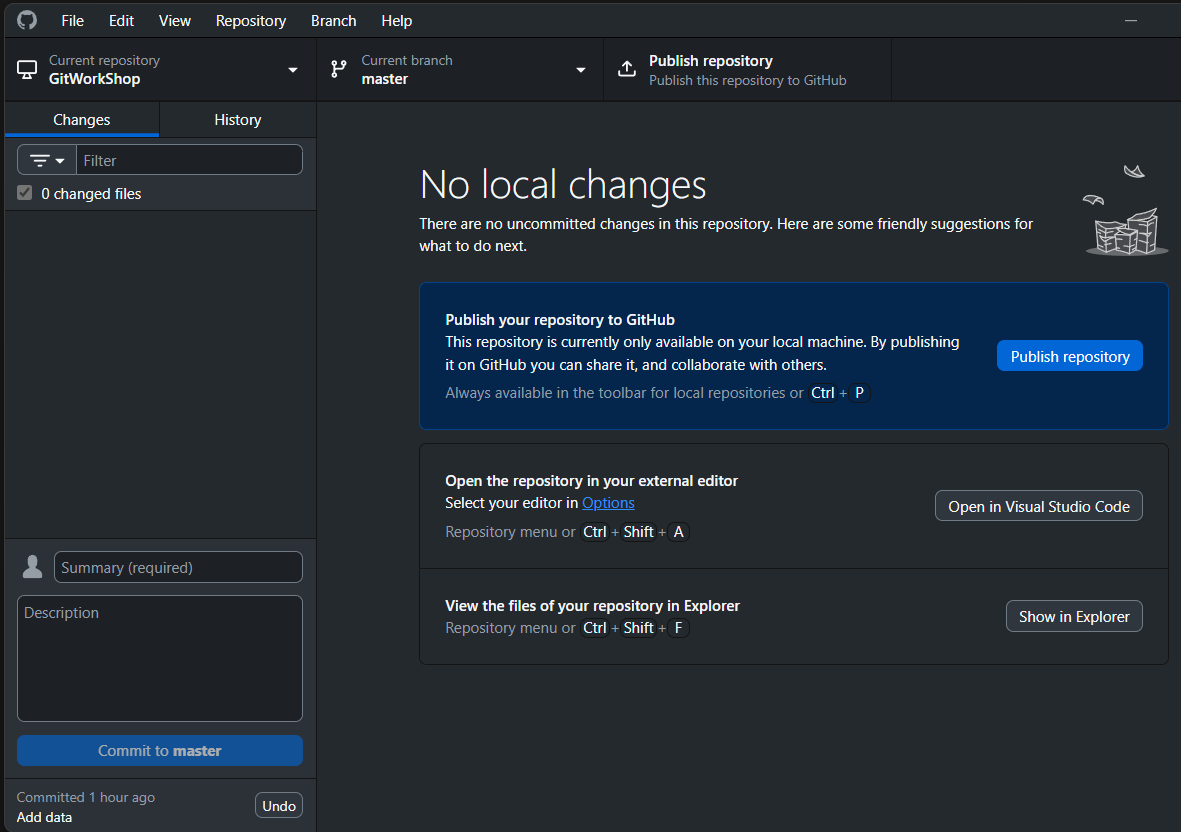


ชื่อ Folder งานของเราหรือเรียกว่า Repository

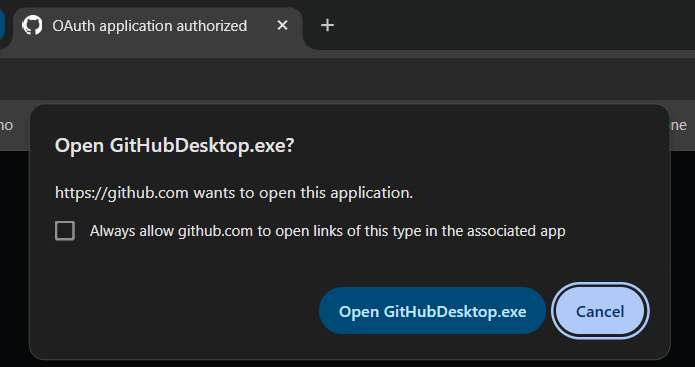
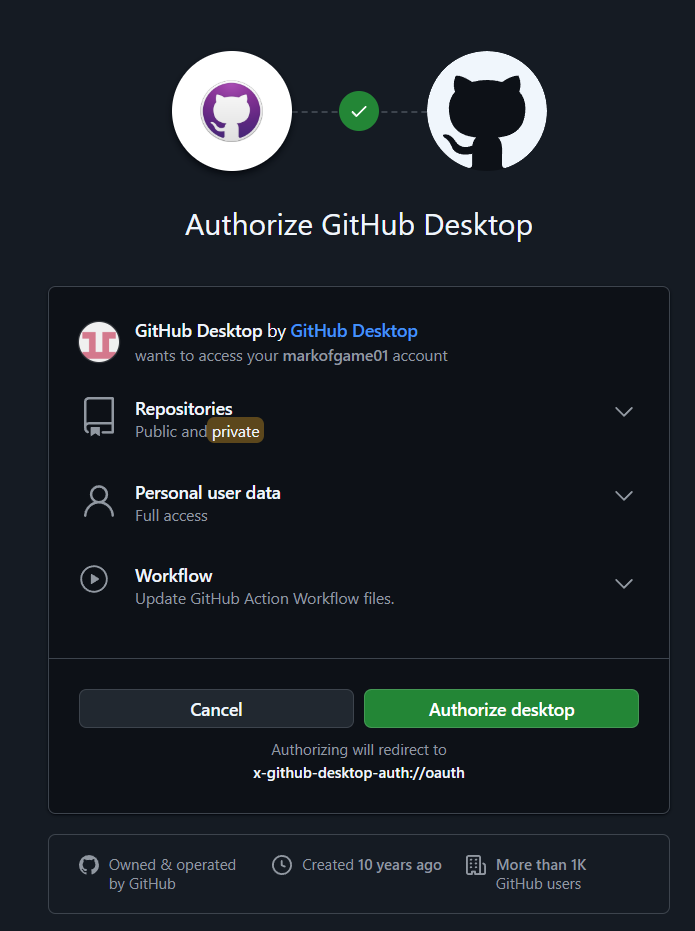
แสดงรายการเปลี่ยนแปลงและประวัติ

ระบุคอมเม้นในการเปลี่ยนแปลงและปุ่มส่งคำสั่ง Commit

1. การเริ่มใช้งาน GitHub Desktop จะต้องเชื่อมต่อกับ GitHub Server ซะก่อนด้วย Username และ Password ที่ได้สมัครไปก่อนหน้านี้

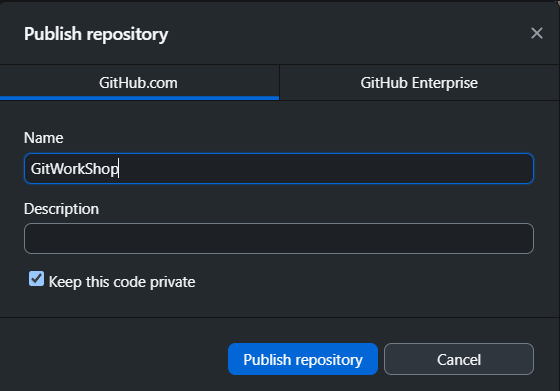


ในขั้นตอนการสมัครเราได้ Sign In เข้าระบบไว้แล้วก็จะพบ User ของเราให้กดปุ่ม Continue ไปได้เลยหากของใครถาม Username / Password อยู่ก็สามารถกรอกแล้วกดเข้าระบบได้เลยเช่นกัน

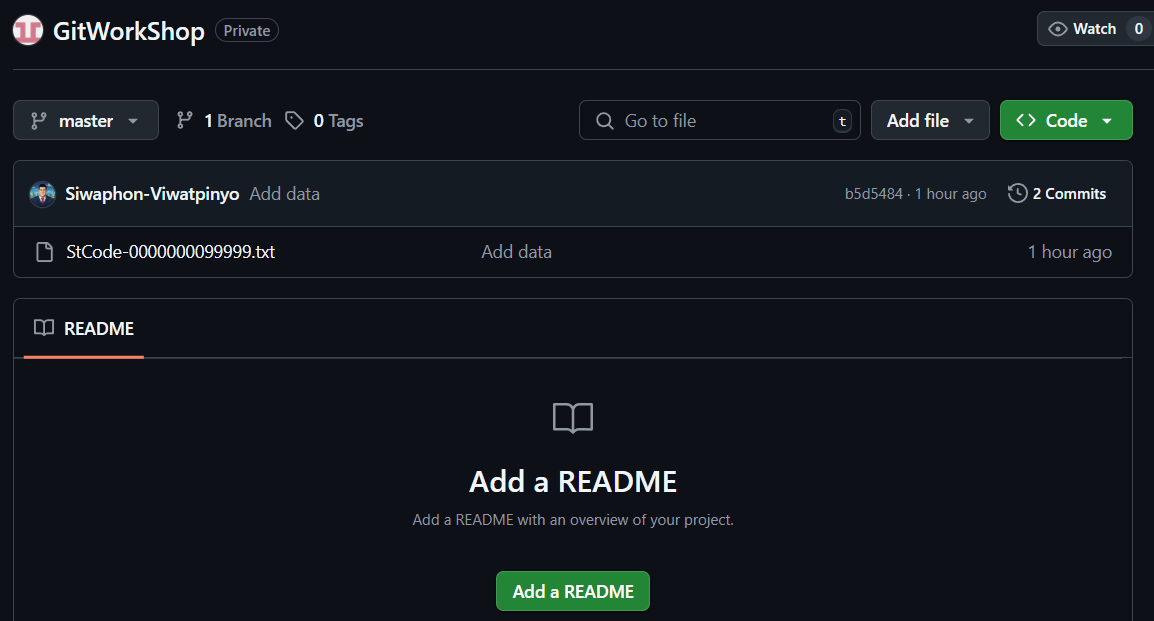
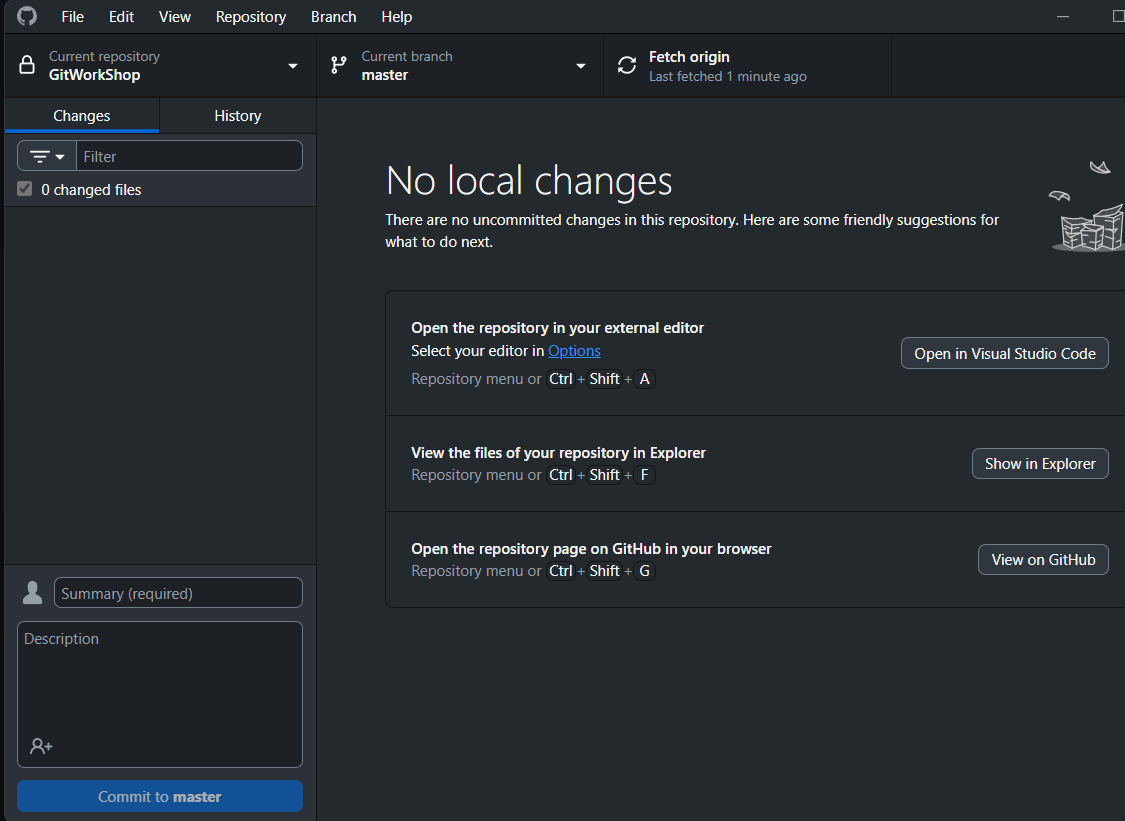


เลือก Repository เป็น Publish

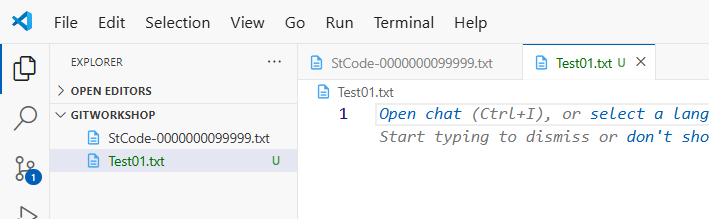
1. จากนั้นจะกลับมาที่ GitHub Desktop เพื่อให้ตั้งชื่อ Repositories เราสามารถใช้ชื่อเดียวกับ Folder ก็ได้หรือตั้งใหม่ก็ได้ในใบงานนี้ให้ใช้ชื่อเดียวกับ Folder ได้เลยและกดปุ่ม Publish Repositories



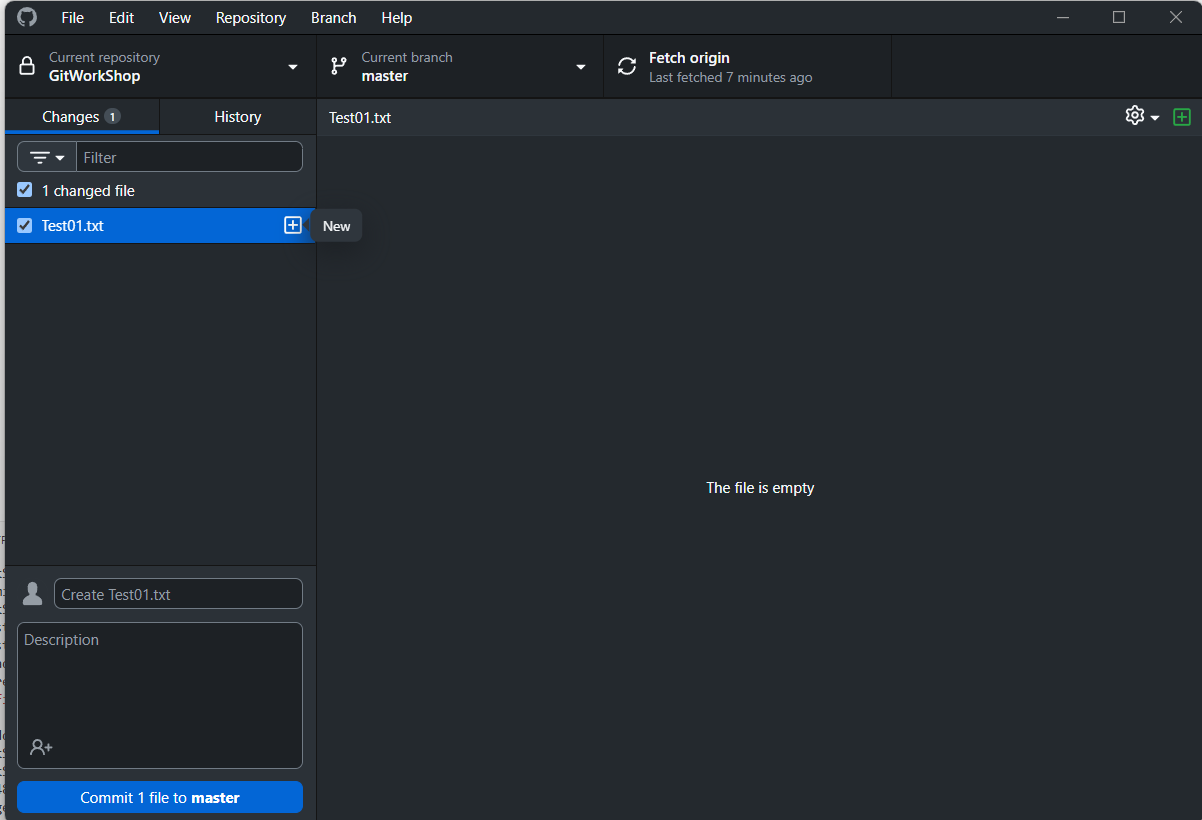
เมื่อกลับสู่หน้าหลักให้เลือกเมนู View on GitHub จะพบไฟล์ StCode-รหัสนักศึกษา.txt



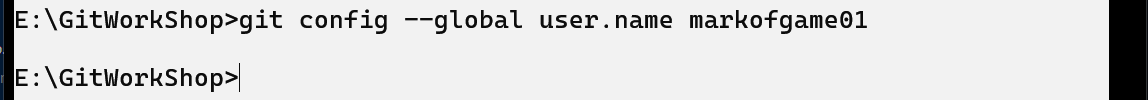
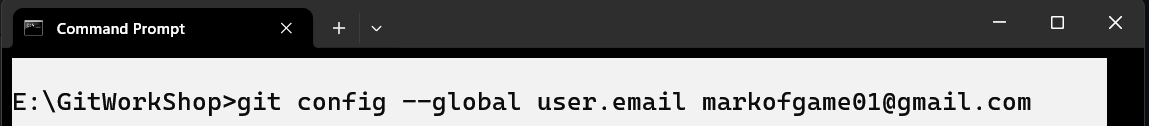
1. กลับไปที่ VScode ทดลองเพิ่มไฟล์ Test01.txt เข้าไปใน GitWorkShop



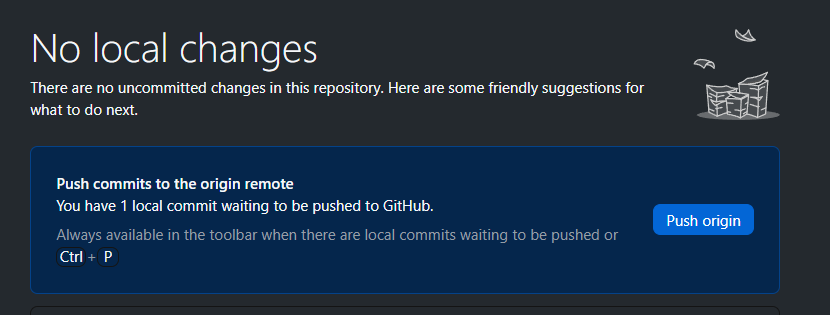
1. กลับไปที่ GitHub Desktop จะพบว่าในรายการเปลี่ยนแปลง(Changes) แสดงรายการว่ามีไฟล์ Test01.txt ขึ้นมาในส่วนนี้เราสามารถเลือกได้ว่าจะ Commit ไฟล์ไหนบ้างหรือทุกไฟล์ จากนั้นเมื่อเราต้องการ Commit ให้ระบุ Comment ในส่วน Description จะระบุหรือไม่ก็ได้ และกดปุ่ม Commit



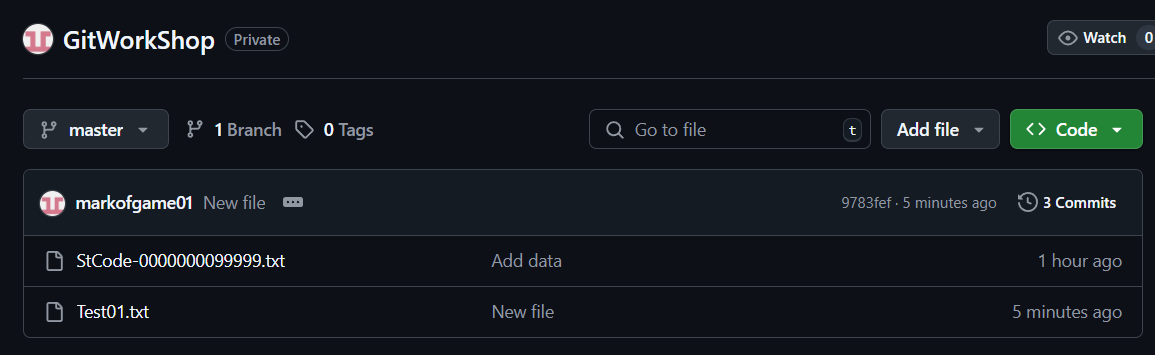
1. สำหรับครั้งแรกโปรแกรมจะแจ้งให้เรา Author identity โดยพิมพ์คำสั่งต่อไปนี้ใน Terminal ใน Folder งานของเราโดยใช้ Email และ Username ตามที่เราสมัครไปก่อนหน้านี้จากนั้นกลับไปกดปุ่ม Commit อีกครั้ง



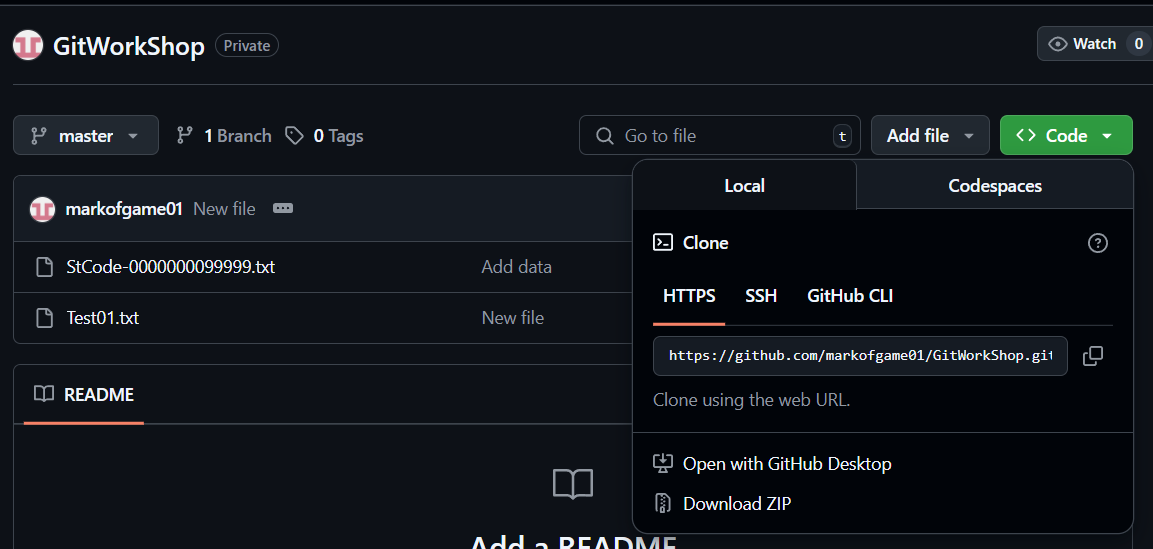
1. ในขั้นตอนข้อ 12 คำสั่ง Commit มีผลกับข้อมูลในเครื่องเราแต่ยังไม่ได้ถูกส่งไปเก็บที่ GitHub ขั้นตอนไปเมื่อเราต้องการส่งข้อมูลขึ้น GitHub จะต้องกดปุ่ม Push origin ก่อนเสมอ



เมื่อโปรแกรมทำงานเสร็จให้กลับไปที่หน้าเว็บ GitHub และรีเฟรสเพื่อตรวจสอบไฟล์ว่าถูกอัพขึ้นไปยัง GitHub Server หรือยัง



1. การส่งงานด้วย Git URL เมื่อนักศึกษาทำใบงานทุกครั้งต้องส่งงานไปเก็บใน Repo ของตนเองและส่ง Git URL เข้าระบบเพื่อให้อาจารย์สามารถตรวจงานได้โดยสามารถคัดลองได้จากเมนู Code > Local > HTTPS

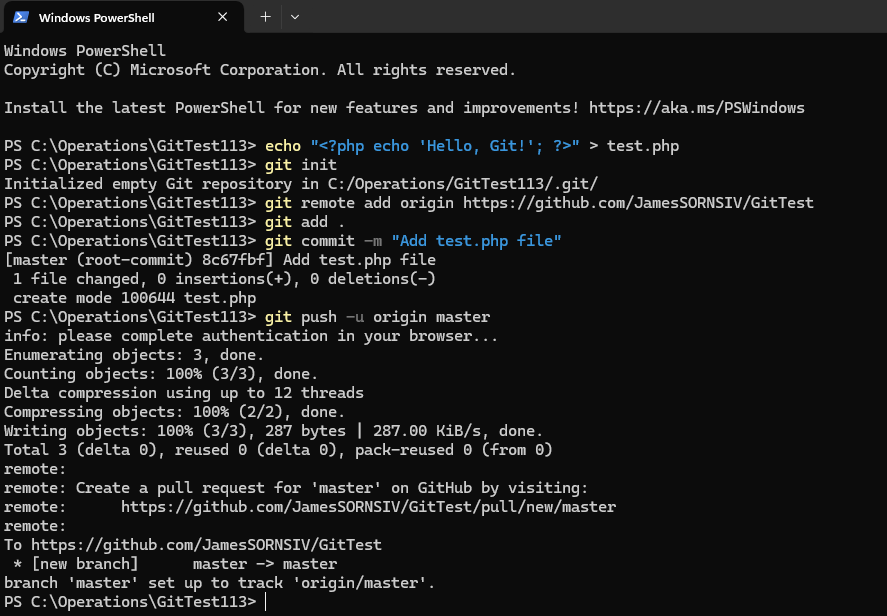


1. ให้นักศึกษาส่ง Git URL เข้า Team ในหัวข้องานที่อาจารย์กำหนดโดยใส่ไว้ในส่วนนี้   
   Git URL : https://github.com/JamesSORNSIV/GitWorkShop.git.
2. ให้นักศึกษาสร้าง Repositories ใหม่ชื่อ GitTest จากนั้นให้สร้าง Folder GitTest........(รหัสนักศึกษา 3 ตัวหลัง) และสร้างไฟล์ชื่อ test.php และทำการใช้คำสั่งตามที่ระบบแนะนำเพื่อส่งไฟล์ดังกล่าวไปเก็บที่ Git Repositories ที่ชื่อ GitTest

Git URL : https://github.com/JamesSORNSIV/GitTest

**รูปคำสั่งที่ใช้ทั้งหมดเพื่อส่งไฟล์ขึ้น Git Repo**

**บันทึกรูปหรือคำสั่ง..............................................................................................................................................................**

****

1. การลบ Git Repositories ให้ทำที่เว็บ Github และเข้าระบบให้เรียบร้อย เลือกที่ Repositories ที่ต้องการลบ จากนั้นเลือกเมนู Settings เลื่อนลงมาล่างสุด เลือก Delete this repository จากนั้นเลือก I want Delete this repository และเลือก I have read and understand there effects จากนั้นต้องระบุข้อความตามที่ระบบแจ้ง เช่น ในตัวอย่างจะเป็น markofgame01/GitTest เมื่อมั่นใจว่าพิมพ์ทุกคำถูกต้องแล้วให้กดปุ่ม Delete this repositoryก็จะเป็นการลบ Repo ที่ไม่ต้องการออกไปแต่ต้องระวังเพราะเมื่อลบแล้วก็นำกลับคืนไม่ได้ แต่งานที่อยู่ในเครื่องเราจะไม่ได้หายไปดู ดังนั้นก็จะสามารถส่งกลับเข้ามาใน Git Server ได้เสมอ