สาขาวิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

รายวิชา 523332 วิศวกรรมซอฟต์แวร์ (Software Engineering)

ปฏิบัติการที่ 3: GitHub Board

ให้นักศึกษาแต่ละคนดำเนินการดังต่อไปนี้:

1. สร้าง Repository lab3 บน GitHub ของตนเอง
2. สร้าง Board กำหนด Column
   - Product Backlog: Story
   - Sprint Backlog #1: Story -> To Do -> Doing -> Done
   - Sprint Backlog #2: Story -> To Do -> Doing -> Done
3. สร้าง Story กำหนด labels ทั้ง 2 Sprint ใน Product Backlog และย้ายเข้า Sprint
4. แบ่ง Task กำหนด labels
5. push commit ทำให้ Card เลื่อนไปที่ Column Done

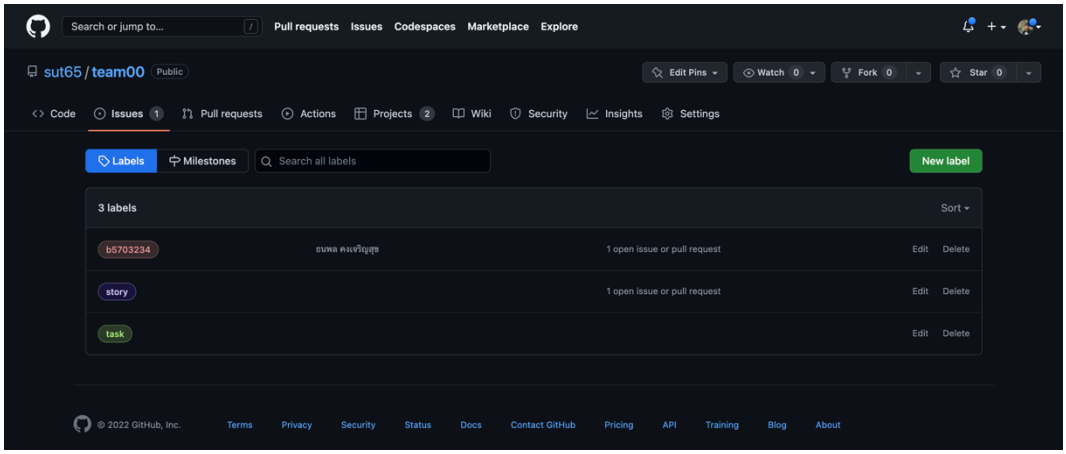
การใช้งาน GitHub เป็น Project Management สำหรับ Agile / Scrum

แนวคิดหลักของการจัดการ Project ใน GitHub สร้างขึ้นรอบ ๆ Issue ซึ่งคล้ายกับ Project Management ตัวอื่น ๆ เช่น Redmime เพื่อปรับให้ Issue ใน GitHub สนับสนุน Scrum นั้นสามารถใช้ป้าย (label) “story” และ “task” ในการกำกับ Issue โดย

- ป้าย story เป็นการบอกว่า Issue ดังกล่าวเป็น User Story และ (ใช้สีม่วง #5319e7)
- ป้าย task เป็นการบอกว่า Issue ดังกล่าวเป็น Task หรือ งาน ที่แยกออกมาจาก User Story (ใช้สีเขียว #adf279)
- ป้ายรหัสนักศึกษาแต่ละคน B63xxxxx (แต่ละคน)

ให้แต่ละกลุ่ม สร้างป้าย story โดยใช้ Color เป็นค่า #5319e7
สร้างป้าย task โดยใช้ Color เป็นค่า #adf279

ตัวอย่าง https://github.com/sut65/team00/labels

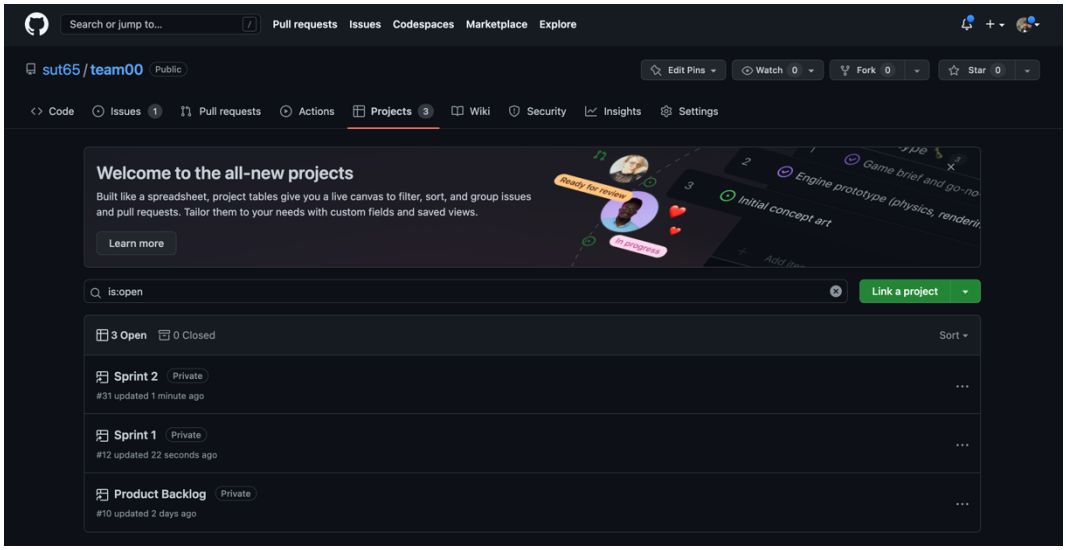


ใน GitHub จะมีระบบที่เรียกว่า “Project” โดย Project จะสามารถแทน บอร์ด เพื่อใช้ตามแนวคิดของ Scrum ได้ โดยบอร์ด จะแบ่งเป็น 2 กลุ่มคือ

1. บอร์ด สำหรับเก็บ Product Backlog
2. บอร์ดสำหรับเก็บ Sprint Backlog #1
3. บอร์ดสำหรับเก็บ Sprint Backlog #2

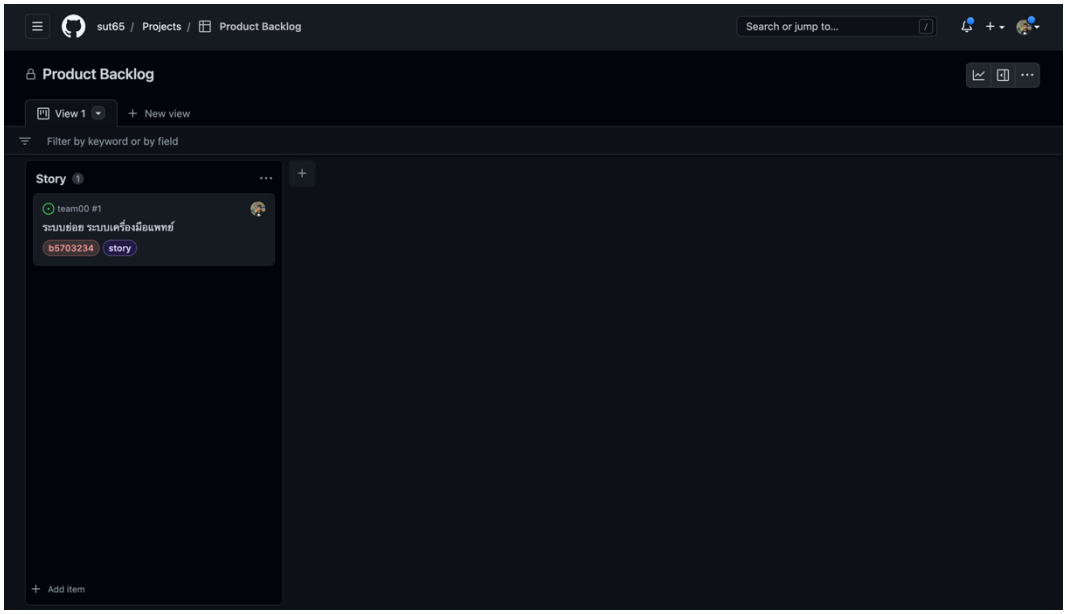
โดยเราจะสร้างบอร์ด “Product Backlog” จำนวน 1 บอร์ด และบอร์ดสำหรับเก็บ Sprint Backlog จำนวน 2 บอร์ดชื่อ “Sprint 1” และ “Sprint 2” ตามลำดับ

https://github.com/sut65/team00/projects?query=is%3Aopen



บอร์ด ชื่อ “Product Backlog” จะเป็น บอร์ด สำหรับเก็บงานค้าง (Backlog แปลว่า “งานค้าง”) ใน บอร์ด จะมี Column เดียวชื่อ Story ไว้เก็บงานค้างของทั้งโครงการในรูปแบบของ User Story

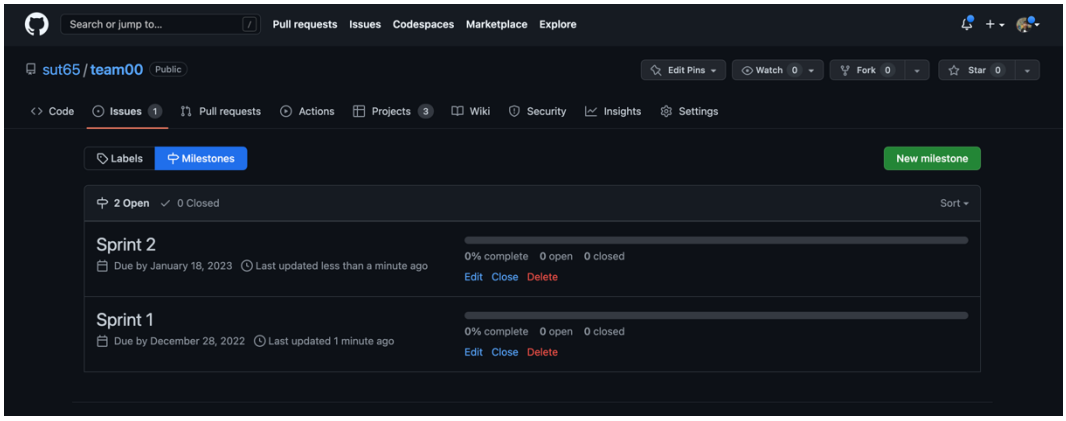
https://github.com/orgs/sut65/projects/10



ส่วน บอร์ด สำหรับ Sprint Backlog จะมีหลาย บอร์ด แยกกันตาม Sprint โดยจะตั้งชื่อตามลำดับของแต่ละ Sprint เช่น **“Sprint 1”, “Sprint 2”** ไปเรื่อย ๆ

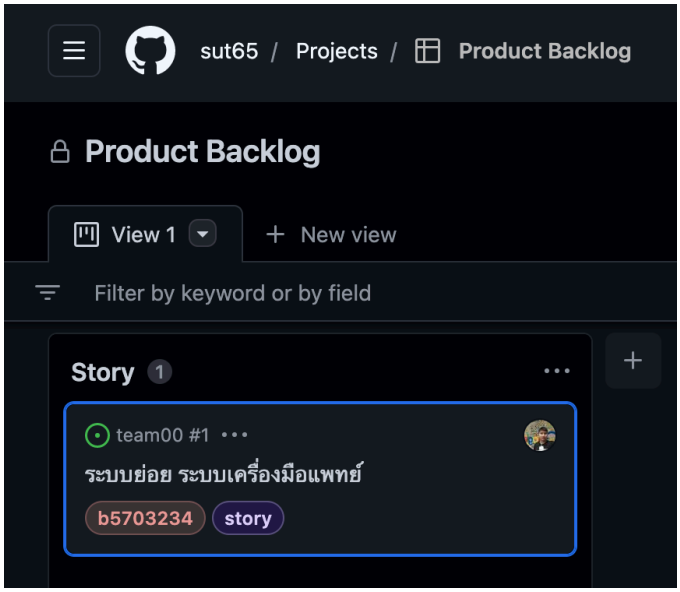
เพื่อให้สามารถ track แต่ละ Sprint ได้อย่างมีประสิทธิภาพ ในแต่ละ Sprint ก็จะมี Milestone ชื่อเดียวกันกำกับอยู่ **ก็คือ บอร์ด ชื่อ “Sprint 1” จะมี Milestone “Sprint 1” กำกับอยู่ด้วย** โดยเราจะโยง Milestone “Sprint 1” เข้ากับทุก ๆ Issue ที่อยู่ในบอร์ด “Sprint 1”

https://github.com/sut65/team00/milestones

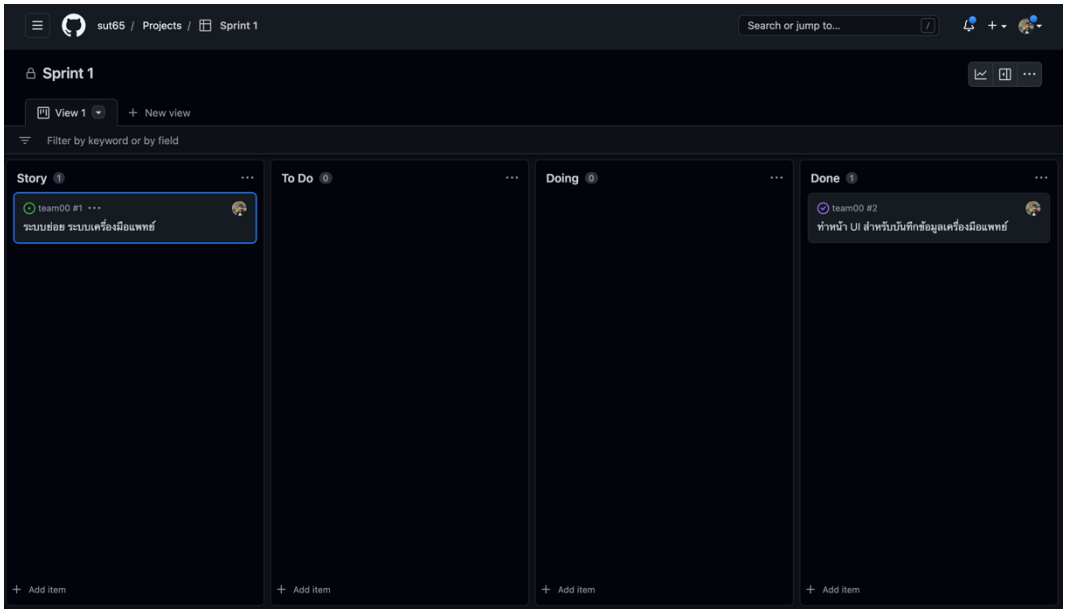


เมื่อสมาชิกแต่ละกลุ่มถูก assign เข้าไปยังแต่ละทีมแล้ว ก็จะสามารถทำหารแก้ไข บอร์ด เองได้โดยแต่ละกลุ่มต้องเข้าไปที่ บอร์ด เพื่อสร้างและตั้งค่า ดังนี้

Product Backlog: https://github.com/orgs/sut65/projects/10



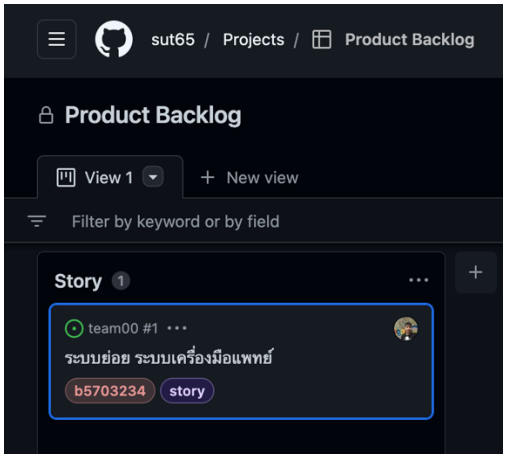
Sprint Backlog: https://github.com/orgs/sut65/projects/12



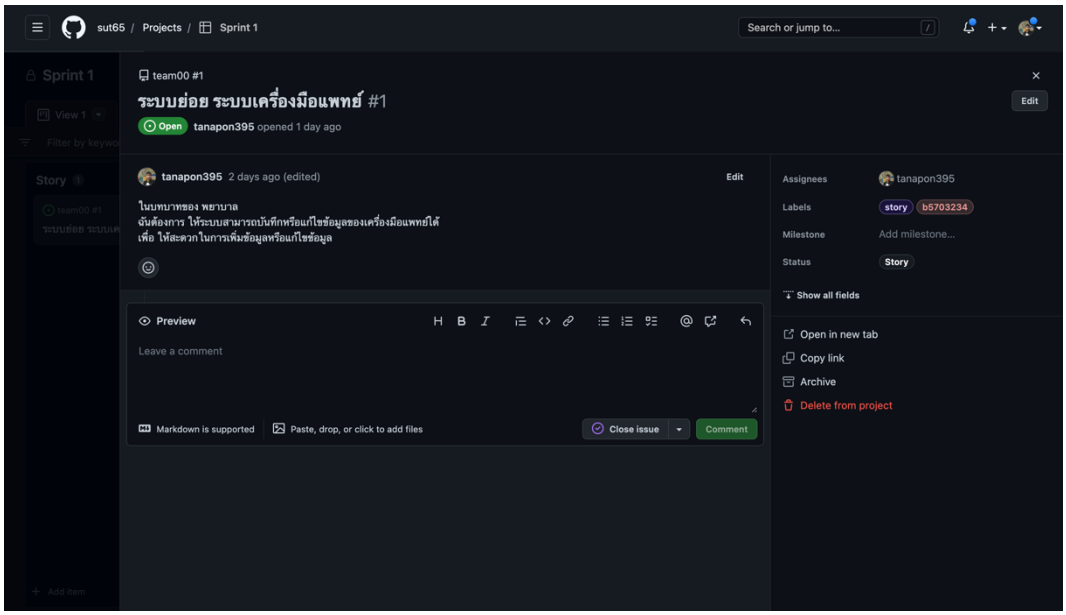
เมื่อเตรียม บอร์ด “Product Backlog”, “Sprint 1” และ “Sprint 2” เรียบร้อยแล้ว ให้ทำการป้อน User Story ใส่ลงใน บอร์ด “Product Backlog” ตรง Column “Story” โดยมีรูปแบบดังนี้

ระบบย่อย <ชื่อระบบย่อย>
ในฐานะของ ...
ฉันต้องการ ...
เพื่อ ...

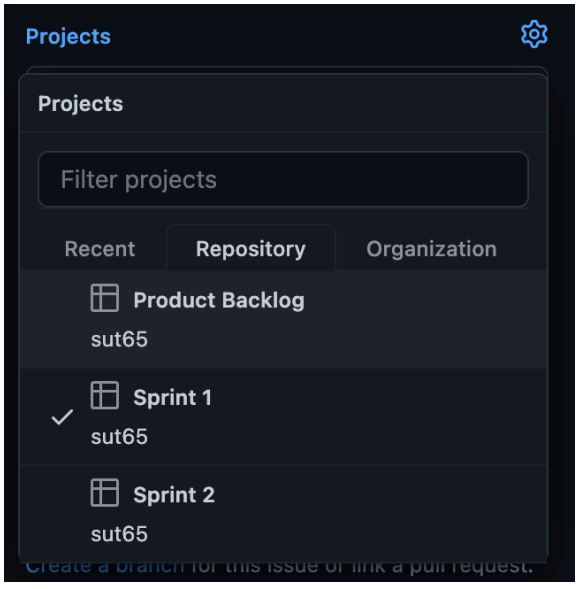
ตัวอย่าง เช่น



เมื่อสร้าง Card แล้วให้ทำการ “**Convert to issue**” เมื่อได้ issue แล้ว ให้ทำการแปะป้าย “story” และป้าย รหัสนักศึกษา



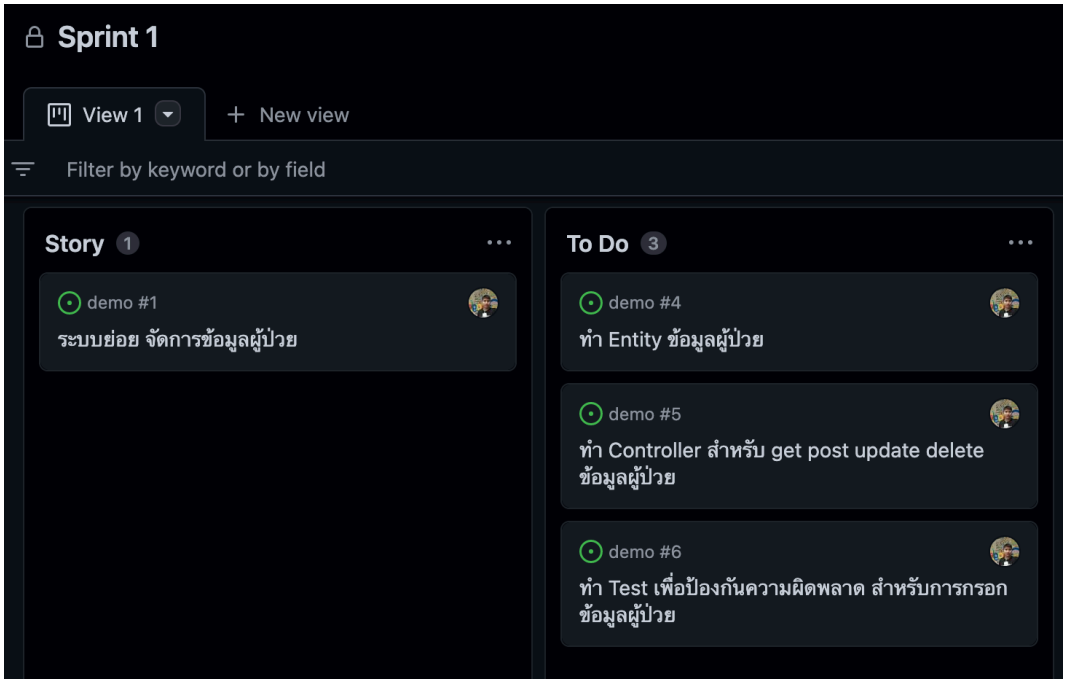
จากนั้นเราสามารถทำการย้าย Story ที่ติดป้ายแล้วจาก “บอร์ด Product Backlog” ไปยังบอร์ดอื่น ๆ เช่น “บอร์ด Sprint 1” เพื่อบอกว่าจะทำ Story นี้ใน Sprint 1 โดยการเปลี่ยนค่า Projects แล้วทำการปลด (uncheck) “Product Backlog” ออก แล้วเลือก “Sprint 1” แทน

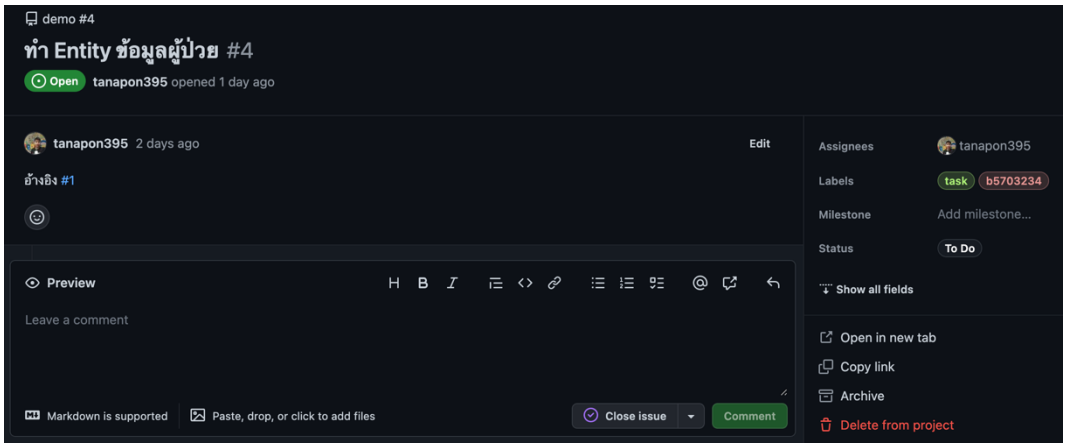


ใน บอร์ด “Sprint 1” จะมีคอลัมน์ “Story” และคอลัมน์ “To Do” ตัวการ์ดที่เป็น User Story จะย้ายจากบอร์ดอื่นมาอยู่ที่คอลัมน์ Story เสมอ และหน้าที่ของทีมคือทำการแบ่ง “story” ให้เป็น “task” โดยการเพิ่ม

การ์ดเข้าไปในคอลัมน์ “To Do” โดยมีข้อกำหนดว่าการ์ดที่ขึ้นต้นด้วยคำกริยา (verb) โดยกำหนดให้เป็นคำกริยา “ทำ”

เมื่อพิมพ์เสร็จจะได้การ์ด จากนั้นให้แปลงการ์ดโดยเลือก “**Convert to issue**” แล้วติดป้าย “task” และรหัสนักศึกษา





จากนั้นให้ทำการย้ายการ์ด จาก คอลัมน์ To Do ไปยัง คอลัมน์ Doing เพื่อเตรียมการเขียนโค้ดในตัวอย่างจะเป็นการทำงานกับ Task หมายเลข #3

ขั้นตอนการเตรียม

1. คัดลอก private key ไฟล์ id_rsa ไปไว้ใน ~/.ssh/id_rsa
   ถ้าไม่มี directory ที่ ~/.ssh สามารถทำการสร้างด้วยคำสั่ง mkdir ~/.ssh
2. เปลี่ยนโหมดไฟล์ id_rsa เป็น 600 ด้วยคำสั่ง chmod 600 ~/.ssh/id_rsa
3. ทำการ clone Git repository ของทีมตนเองจาก GitHub ด้วยคำสั่ง เช่น
   git clone git@github.com:sut65/team00.git
   จะเป็นของกลุ่ม G00 แล้วจะได้ directory เช่น team00 โดยจะทำการเปลี่ยน directory ไปที่ team00 ก่อนที่จะทำงานกับ code
   ด้วยคำสั่ง **cd team00**
4. จากนั้นทำการตรวจสอบว่าเราอยู่ที่ main branch ของ Git repository หรือไม่
   ด้วยคำสั่ง **git status**
   ถ้าไม่ใช่ก็ทำการเปลี่ยน branch ให้เป็น main branch ก่อน
   ด้วนคำสั่ง **git checkout main**
5. สร้าง branch ใหม่โดยตั้งต้นจาก main branch ให้เป็นชื่อ issue-<หมายเลข>
   ด้วยคำสั่ง **git checkout -b issue-3**
   เนื่องจากตัวอย่างเป็นการทำงานเพื่อเขียนโค้ดให้ Issue #3
6. เขียนโค้ดและเพิ่มหรือลดไฟล์เข้า repository ด้วยคำสั่ง
   git add <ไฟล์> หรือ
   git rm <ไฟล์>
7. ทำการ commit ด้วยข้อความ และลงท้ายด้วย close #<หมายเลข> เช่น
   git commit -m “ui module equipment - close #3”
   โดยจะเป็นการ commit ลงไปยัง branch ชื่อ issue-3 (ไม่ใช่การ commit ลง main branch)
8. จากนั้นเป็นการ merge โด้ดจาก branch issue-3 ไปยัง branch main โดนจุดนี้ต้องระวัง เนื่องจากอาจมีเพื่อนในทีมทำการเปลี่ยนโค้ดบน GitHub ไปแล้ว จึงจำเป็นต้อง update โค้ดล่าสุดของเพื่อนลงมาก่อน โดย
   git remote update # ดึงโค้ดลงมาไว้เบื้องหลัง
   git rebase origin/main # ปรับฐาน issue-3 ให้ตรงกับ remote บน GitHub (remote ตอนนี้ชื่อ origin/main)
   git checkout main # สลับมา main
   git merge issue-3 –no-ff # ทำการ merge issue-3 เข้าสู่ main

จากนั้น push

git push origin main

ถ้า push ไม่ขึ้นอีกให้ทำการ rebase อีกรอบ

git remote update

git rebase origin/main

ก่อนทำการ push ซ้ำด้วยคำสั่ง

git push origin main