**รหัสนักศึกษา : 61070143**

**Pre-Lab**

1. Download source code project จากใน Google Class Room ไฟล์ชื่อ sdte-lab6.zip และ Extract ให้เรียบร้อย
2. สร้าง Repository ใน Github โดยให้ตั้งชื่อว่า sdte-lab6 และให้เลือกเป็น Public Repository
3. ทำการ Initial Git, Commit ไฟล์ใน Folder ที่ดาวโหลดมาในข้อ 1 (ยังไม่ต้องแก้ไขไฟล์ใด) และ push ขึ้น Remote repository ที่สร้างในข้อที่ 2
   * + - **git init**
       - **git add .**
       - **git commit -m "First commit”**
       - **git remote add origin REMOTE\_REPO\_URL.git**
       - **git branch -m main**

**A. Git URL : https://github.com/ptydvp/sdte-lab6.git**

**Lab 1 - สร้าง Workflow สำหรับเช็ค Code ด้วย Prettier และตรวจสอบ Test case**

1. สร้างไฟล์ .github/workflows/check-code.yml
2. ใส่ Code จาก <https://gist.github.com/saranonuan/1cee9e2577ce74b31b1bd8ba1a3b50e0> ในไฟล์ .github/workflows/check-code.yml
3. ทำการ Commit และ Push code (**ให้ใช้ branch main เป็น branch หลัก**)
4. เปิด Github Repository ไปที่ menu “Actions” ดูผลลัพธ์และบันทึกผล

**B. Capture Screen ผลการ Run Workflow (ต้องผ่านเป็นสีเขียว  )**

Graphical user interface, text, application, website

Description automatically generated

**Lab 2 - ทดสอบหาก Push Code ที่ไม่ผ่าน/มีปัญหา**

1. ทำการแก้ไขไฟล์ src/server.js บรรทัดที่ 5 ให้ code return เป็นอย่างอื่น เพื่อให้ไม่ผ่าน Test Case
2. ทำการ Commit และ Push code
3. เปิด Github Repository ไปที่ menu “Actions” ดูผลลัพธ์และบันทึกผล

**C. Capture Screen ไฟล์ src/server.js**

**Graphical user interface, text

Description automatically generated**

**D. Capture Screen ผลการ Run Workflow หลัง Push ในข้อที่ 2 (ต้องไม่ผ่านเป็นสีแดง  )**

**Graphical user interface, text, application

Description automatically generated**

1. แก้ไขไฟล์ src/server.js บรรทัดที่ 5 ให้ code return “Hello world!” เหมือนเดิม เพื่อให้ผ่าน Test Case
2. ทำการ Commit และ Push code
3. เปิด Github Repository ไปที่ menu “Actions” ดูผลลัพธ์และบันทึกผล

**E. Capture Screen ผลการ Run Workflow หลัง Push ในข้อที่ 5 (ต้องผ่านเป็นสีเขียว  )**

**Graphical user interface, text, application, website

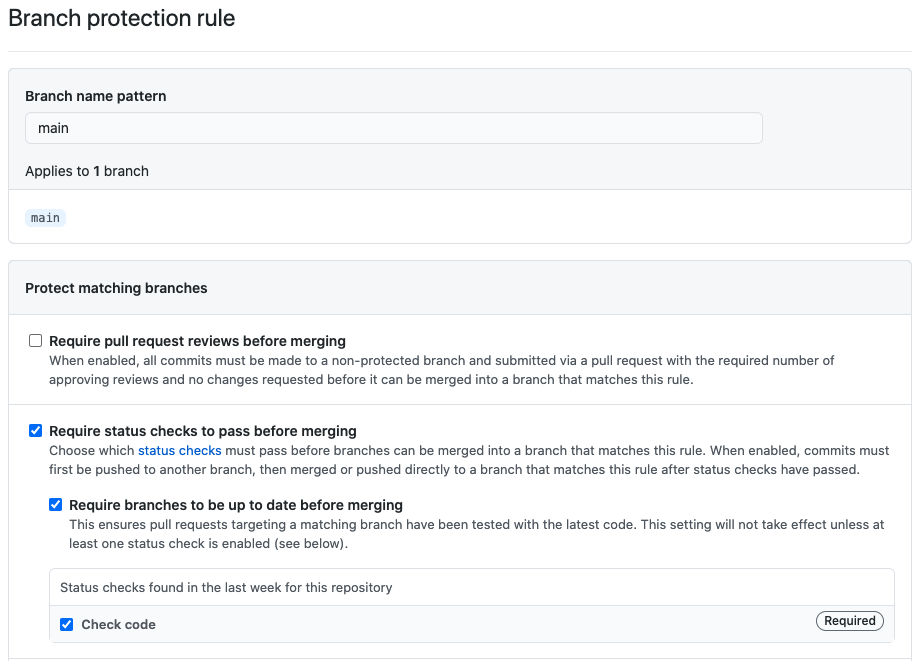
Description automatically generated**

**Lab 3 - ตั้งค่า Protect branch และ Pull request check**

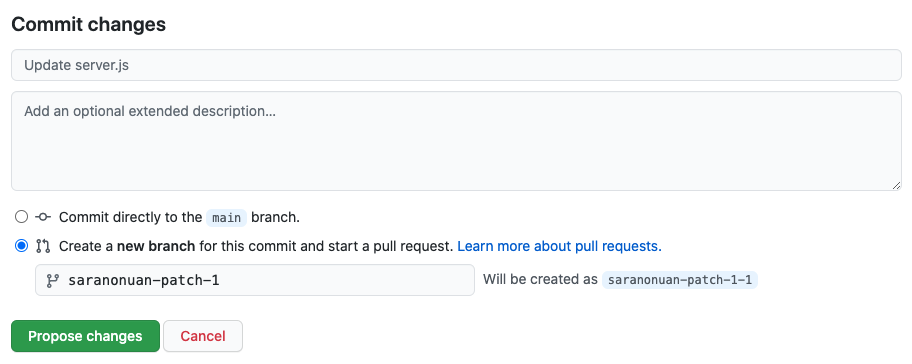
1. เข้าไปที่ เมนู Settings -> Branches -> Branch protection rule กด Add rule
2. ให้ตั้งค่า Branch protection rules สำหรับ branch “**main**” โดย
   1. ต้องผ่าน status check ทั้งหมดก่อน จึงจะ Merge ได้ (Require status checks to pass before merging)

และต้องให้ branch up to date ก่อนที่จะ merge ได้ (Require branches to be up to date before merging)

และต้องรัน Github Action Check Code ผ่าน

* 1. กด Create หรือ Save changes เพื่อบันทึก

1. แก้ไขไฟล์​.github/workflows/check-code.yml โดยเพิ่มตั้งค่า Workflow ให้ทำงานเมื่อมีผู้สร้าง Pull Request ด้วย  
   ตัวอย่าง <https://gist.github.com/saranonuan/743bacbd38b647fb8019baf0ed47d24b>
2. ทำการ Commit และ Push code
3. เปิด Github Repository ไปที่ menu “Actions” ผลการรัน Workflow ยังต้อง Success  อยู่
4. ทดสอบการทำงาน โดยให้ลองแก้ไขไฟล์ src/server.js โดยใช้ Online Editor ให้บรรทัดที่ 5 ให้ return ต่างไปจากเดิม เพื่อให้การทดสอบนั้นไม่ผ่าน   
   แล้วเลือก Commit changes โดยให้เลือกสร้าง Branch ใหม่ เพื่อเปิด Pull Request เป็นการสมมติว่า มีผู้มาแก้ไขแล้วได้สร้าง Pull Request เข้ามา



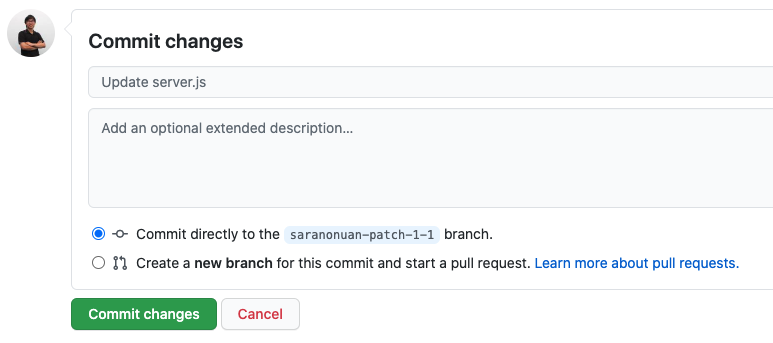
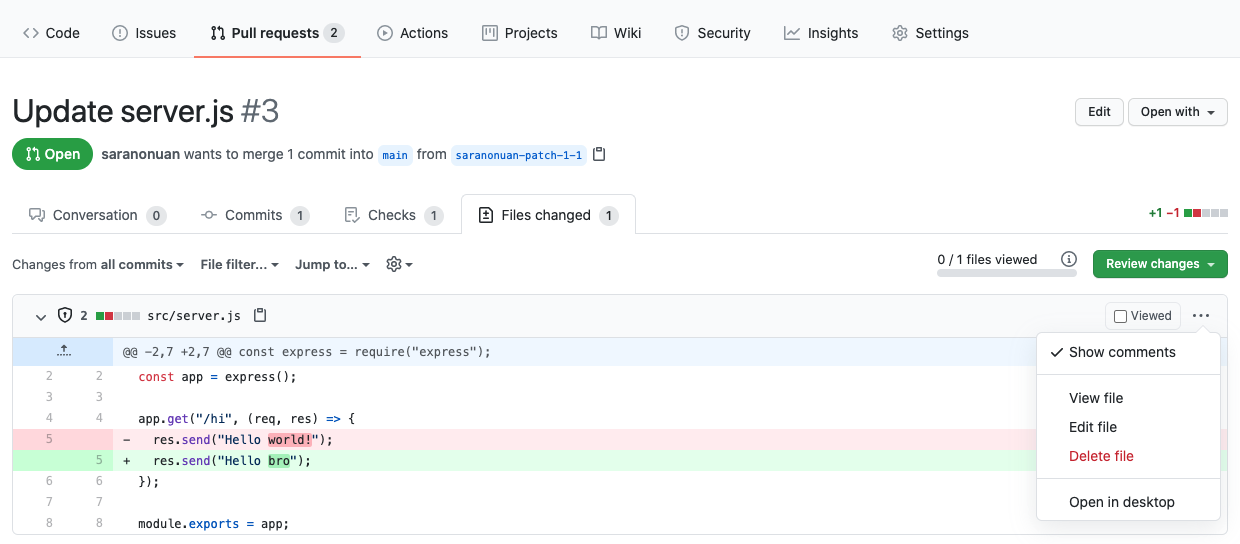
1. รอ Pull request check ทำงานและบันทึกผล

**F. Capture Screen ผลการ Run Check ของ Workflow ในเมนู Conversation ของ Pull-Request (ต้องไม่ผ่าน  ผล check เป็น failed)**

A picture containing text, screenshot, computer

Description automatically generated

1. ไปที่ Tab Files changed แล้วทำการแก้ไขไฟล์ให้กลับเป็นเหมือนเดิม (“Hello world!”) เพื่อให้ผ่าน Test case  
   กด Commit changes บันทึกการแก้ไข เพื่อ trigger workflow ให้ตรวจสอบอีกครั้ง



1. รอ Pull request check ทำงานเพื่อตรวจสอบการแก้ไขในข้อที่ 6 และบันทึกผล

**F. Capture Screen ผลการ Run Check ของ Workflow ในเมนู Conversation ของ Pull-Request (ต้องผ่าน  ผล check เป็น Success)**

Graphical user interface, text, application

Description automatically generated

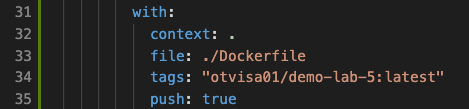
1. กด Confirm Merge

**G. Capture Screen การทำงานของ Workflow (Tab Actions)**

**Graphical user interface, text, application

Description automatically generated**

**Lab 4 - Auto build Docker Image เมื่อมีการ Push เข้า main branch**

1. สร้าง Image repo ใน Docker hub สำหรับเก็บ Image ที่กำลังจะสร้างด้วย Github Actions
2. ใน Github repo ไปที่ menu Settings -> Secrets
3. กด New repository secret และสร้าง Secret ต่อไปนี้
   1. Name “**DOCKER\_USERNAME**” โดยใส่ Value เป็น username ที่ใช้เข้า Docker Hub
   2. Name “**DOCKER\_PASSWORD**” โดยใส่ Value เป็น password ที่ใช้เข้า Docker Hub
4. แก้ไขไฟล์ .github/workflows/check-code.yml   
   ตาม <https://gist.github.com/saranonuan/a03507bcf8877cf230f196a1df6981b6>  
   ซึ่งจะทำให้ Workflow ทำงานเมื่อมีการ push ในทุก branch ยกเว้น main การเช็ค pull-request ทุก branch
5. สร้างไฟล์ .github/workfloes/check-and-build-main.yml  
   ตาม <https://gist.github.com/saranonuan/a03507bcf8877cf230f196a1df6981b6>  
   และให้แทนที่ค่า **DOCKER\_IMAGE\_REPO** ด้วย image repo + tag ที่สร้างในข้อ 1  
   ตัวอย่างเช่น “otvisa01/demo-lab-6:latest”
6. ทำการ Commit และ Push (หาก Push แล้วเกิด Merge reject จากการที่ทำการ Edit ผ่าน Online Editor ใน Lab ที่แล้ว, ให้ทำการ Pull ก่อนด้วยคำสั่ง git pull)
7. เปิด Github Repository ไปที่ menu “Actions” ดูผลลัพธ์และบันทึกผล

**G. Capture Screen ผลการ Run Workflow (ต้องผ่าน  ผล check เป็น Success)**

**Graphical user interface, text, application, website

Description automatically generated**

**H. Capture Screen ใน Docker Hub ให้เห็น Image ที่ Push ขึ้นมาด้วย GitHub Actions**

**Graphical user interface, text, application

Description automatically generated**