BÀI TẬP THỰC HÀNH SỐ 1

Học phần: CSE485 - Công nghệ Web

Bài 1: Thực hành kỹ năng làm việc nhóm trên Git/Github (Gitlab/Bitbucket)

Note: Git và Docker là những công cụ mà một Developer cần (bắt buộc) phải trang bị để sử dụng trong môi trường doanh nghiệp (theo yêu cầu của các dự án). SV nên chủ động tự học, tự nghiên cứu.

Bài tập: **Hợp tác phát triển mã với Git và GitHub**

Bạn và nhóm của bạn sẽ làm việc trong một dự án để phát triển một ứng dụng web đơn giản sử dụng HTML, CSS, JavaScript và PHP. Bạn sẽ sử dụng Git và GitHub để cộng tác trong dự án.

Yêu cầu:

- Mỗi thành viên trong nhóm nên có tài khoản GitHub (hoặc thay thế với GitLab/Bitbucket).
- Mỗi thành viên trong nhóm nên có một trình soạn thảo văn bản hoặc môi trường phát triển tích hợp (IDE) để chỉnh sửa mã.
- Mỗi thành viên trong nhóm nên cài đặt Git trên máy cục bộ của mình.

Hãy tưởng tượng bạn và bạn bè của bạn đang cùng nhau thực hiện một dự án lớn. Bạn đang xây lâu đài từ các khối vuông! Mỗi người trong số các bạn có một công việc phải làm, chẳng hạn như xây tường, tháp hoặc hào. Nhưng bạn cần đảm bảo rằng tất cả các bạn đang làm việc cùng nhau và mọi người đều biết chuyện gì đang xảy ra.

Trong GitHub, khi bạn và bạn bè cùng làm việc trên một dự án, mỗi người sẽ có máy tính riêng để viết và lưu mã của mình. Nhưng bạn cũng có một không gian dùng chung trên internet nơi bạn lưu trữ tất cả mã của mình, được gọi là kho lưu trữ.

Một trong những cách bạn có thể làm việc cùng nhau trên GitHub là sử dụng tính năng "Hoạt động". Đây giống như một bảng thông báo lớn, nơi mọi người có thể thấy những người khác đang làm gì. Đó là một cách để đảm bảo tất cả các bạn đang làm việc cùng nhau và mọi người đều biết chuyện gì đang xảy ra.

Dưới đây là một số ví dụ về các hoạt động bạn có thể thấy trên GitHub:

- Ai đó có thể tạo một nhánh mới cho một tính năng mới mà họ đang thực hiện.
- Ai đó có thể thay đổi một số mã và tạo yêu cầu kéo để người khác xem xét.
- Ai đó có thể phê duyệt yêu cầu kéo mà người khác đã thực hiện.
- Ai đó có thể để lại nhận xét về yêu cầu kéo, đưa ra phản hồi về những thay đổi đã được thực hiện.

Tất cả những hoạt động này giống như những công việc khác nhau mà bạn và bạn bè của bạn có khi xây dựng lâu đài của mình. Mỗi bạn đang làm một việc khác nhau, nhưng tất cả các bạn cần biết những người khác đang làm gì để có thể làm việc cùng nhau.

Tính năng Hoạt động trên GitHub là một cách để bạn và bạn bè của bạn cập nhật những gì mọi người đang làm. Bạn có thể xem những thay đổi nào đang được thực hiện đối với mã, ai đang làm việc với cái gì và mọi người đang đưa ra phản hồi gì. Đó là một cách để đảm bảo rằng tất cả các bạn đều thống nhất và mọi người đều biết chuyện gì đang xảy ra.

Vì vậy, tóm lại, các hoạt động trên GitHub giống như một bảng thông báo lớn, nơi mọi người có thể xem những gì người khác đang làm. Đó là một cách để đảm bảo tất cả các bạn đang làm việc cùng nhau và mọi người đều biết chuyện gì đang xảy ra. Nó giống như xây dựng một lâu đài bằng những khối vuông với bạn bè của bạn và đảm bảo rằng mọi người đều biết công việc của họ là gì và những người khác đang làm gì.

Hướng dẫn:

1. Thiết lập Dự án mới (Kho lưu trữ) [Trưởng nhóm]
   - Chọn một thành viên trong nhóm để tạo kho lưu trữ GitHub mới cho dự án (Trưởng nhóm). Yêu cầu tạo kho có tên: **CSE485_2023**.
   - Đặt kho lưu trữ ở chế độ công khai hoặc riêng tư, tùy theo sở thích của nhóm bạn
   - Thêm các thành viên trong nhóm vào kho lưu trữ.
2. Sao chép kho lưu trữ vào máy cục bộ [Mỗi thành viên]
   - Truy cập kho lưu trữ trên GitHub.
   - Nhấp vào nút "Code" và sao chép URL của kho lưu trữ.
   - Mở một công cụ lệnh trên máy cục bộ.
   - Điều hướng đến thư mục mà bạn muốn sao chép (lưu) kho lưu trữ.
   - Chạy lệnh **git clone** theo sau là URL kho lưu trữ.
     Ví dụ: **git clone** https://github.com/kitudu/TLU_CSE485
3. Thảo luận trong nhóm và quyết định cấu trúc dự án cơ bản cho ứng dụng web của bạn (ví dụ: cấu trúc thư mục cho các thành phần lưu trữ khác nhau, qui ước đặt tên tệp tin ...).
4. Tạo nhánh (branch) [Mỗi thành viên]
   - Tạo một nhánh để mỗi thành viên trong nhóm làm việc. Mỗi thành viên trong nhóm nên làm việc trên nhánh riêng của riêng mình.
   - Chạy lệnh **git checkout -b** theo sau là tên nhánh của bạn để tạo một nhánh mới. (hoặc sử dụng lệnh **git branch <tennhanh>** để tạo nhánh mới và **git checkout <tennhanh>** để chuyển sang làm việc với nhánh bất kì.

5. Thực hiện nhiệm vụ theo phân công/thỏa thuận [Mỗi thành viên]
    - Mỗi thành viên trong nhóm nên thực hiện một phần khác nhau của ứng dụng web.
    - Viết mã cho phần ứng dụng web trong (các) tệp thích hợp trong kho lưu trữ cục bộ của mình.
    - Đảm bảo bao gồm các chú thích trong mã để giải thích mã của bạn làm gì.
    - Kiểm tra mã để đảm bảo mã hoạt động như mong đợi.
6. Xác nhận thay đổi và đẩy lên nhánh [Mỗi thành viên]
    - Chạy lệnh **git add** theo sau là danh sách (các) tệp có thay đổi (sửa đổi) hoặc sử dụng **git add .** để quét tất cả
    - Chạy lệnh **git commit** theo sau là một thông báo giải thích những thay đổi bạn đã thực hiện.
    - Chạy lệnh **git push** để đẩy các thay đổi từ local lên nhánh trên GitHub.
7. Hợp nhất các thay đổi từ nhánh của mỗi thành viên trong nhóm vào nhánh chính bằng yêu cầu kéo (pull). [Mỗi thành viên]
    - Truy cập trang kho lưu trữ trên GitHub.
    - Nhấp vào tab "Pull requests".
    - Nhấp vào nút "New pull request".
    - Chọn nhánh của bạn làm nhánh "compare" và nhánh chính làm nhánh "base".
    - Để lại một chú thích, mô tả những thay đổi bạn đã thực hiện.
    - Nhấp vào nút "Create pull request".
    - Các thành viên trong nhóm của bạn sẽ xem xét mã và để lại chú thích nếu cần.
    - Khi mọi người đã phê duyệt pull request, chủ sở hữu kho lưu trữ nên hợp nhất các thay đổi vào nhánh chính.

Hãy tưởng tượng bạn và bạn bè của bạn đang cùng nhau xây dựng một ngôi nhà trên cây. Mỗi bạn có một công việc phải làm, chẳng hạn như cắt gỗ, đóng đinh hoặc sơn. Sau khi hoàn thành công việc của mình, bạn muốn cho bạn bè biết để họ có thể kiểm tra công việc của bạn và đảm bảo mọi thứ diễn ra suôn sẻ.

Trong GitHub, khi bạn và bạn bè cùng làm việc trên một dự án, mỗi người sẽ có máy tính riêng để viết và lưu mã của mình. Nhưng bạn cũng có một không gian dùng chung trên internet nơi bạn lưu trữ tất cả mã của mình, được gọi là kho lưu trữ.

Khi bạn viết xong mã của mình và hài lòng với cách nó hoạt động, bạn muốn chia sẻ mã đó với bạn bè để họ có thể kiểm tra công việc của bạn và đảm bảo mọi thứ đang diễn ra suôn sẻ, giống như với ngôi nhà trên cây của bạn.

Đó là lúc yêu cầu kéo xuất hiện. Yêu cầu kéo giống như yêu cầu bạn bè của bạn kiểm tra công việc của bạn và đưa ra phản hồi cho bạn. Khi bạn tạo một yêu cầu kéo, bạn đang nói với bạn bè của

mình rằng bạn có một số mã mới muốn thêm vào dự án và bạn yêu cầu họ xem xét mã đó và đưa ra ý kiến của họ.
Bạn bè của bạn có thể xem mã của bạn và để lại nhận xét nếu họ nghĩ cần phải thay đổi hoặc cải thiện điều gì đó. Sau khi mọi người đồng ý rằng mã của bạn tốt, ai đó có thể hợp nhất mã đó vào dự án, nghĩa là mã đó trở thành một phần vĩnh viễn của mã mà mọi người có thể sử dụng.
Vì vậy, tóm lại, yêu cầu kéo giống như yêu cầu bạn bè của bạn kiểm tra công việc của bạn và cung cấp cho bạn phản hồi trước khi bạn thêm mã của mình vào dự án. Đó là một cách để đảm bảo mọi thứ hoạt động trơn tru và mọi người đều hài lòng với mã trước khi nó trở thành một phần lâu dài của dự án.

8. Khắc phục mọi xung đột hợp nhất phát sinh và đảm bảo rằng ứng dụng web hoạt động như mong đợi sau khi hợp nhất.

Hãy tưởng tượng bạn và bạn của bạn đang cùng vẽ một bức tranh. Cả hai bạn đang vẽ các phần khác nhau của bức tranh và cả hai bạn đều có khoảng thời gian tuyệt vời. Nhưng khi bạn thử đặt các bức tranh của mình lại với nhau, bạn nhận thấy rằng cả hai bạn đã vẽ trên cùng một phần của bức tranh.
Đó là một cuộc xung đột hợp nhất! Đó là khi hai người đang cố gắng làm cùng một việc vào cùng một thời điểm và công việc của họ mâu thuẫn với nhau.
Trong GitHub, khi bạn và bạn bè cùng làm việc trên một dự án, mỗi người sẽ có máy tính riêng để viết và lưu mã của mình. Nhưng bạn cũng có một không gian dùng chung trên internet nơi bạn lưu trữ tất cả mã của mình, được gọi là kho lưu trữ.
Khi bạn và bạn bè của bạn đang làm việc trên cùng một phần mã cùng lúc, bạn có thể vô tình tạo ra xung đột hợp nhất. Ví dụ: nếu bạn và bạn của bạn đều cố gắng thay đổi cùng một dòng mã, GitHub sẽ không biết nên giữ thay đổi nào và loại bỏ thay đổi nào. Nó sẽ cần ai đó nói cho nó biết những thay đổi nào cần giữ lại và những thay đổi nào cần loại bỏ.
Đây là một chút giống như bức tranh của bạn. Nếu bạn và bạn của bạn cùng vẽ trên cùng một phần của bức tranh, bạn sẽ cần nói chuyện với nhau và quyết định giữ lại phần nào trong bức tranh của mình và thay đổi phần nào.
Để khắc phục xung đột hợp nhất trong GitHub, bạn và bạn bè của mình sẽ cần thảo luận với nhau và quyết định phần nào của mã sẽ được giữ lại và phần nào sẽ thay đổi. Khi bạn đã đưa ra quyết định, ai đó có thể cho GitHub biết những thay đổi nào cần giữ lại và những thay đổi nào cần loại bỏ. Sau đó, bạn có thể tiếp tục làm việc với dự án của mình cùng nhau!
Vì vậy, tóm lại, xung đột hợp nhất giống như khi bạn và bạn của bạn vô tình vẽ lên cùng một phần của khung vẽ và bạn cần nói chuyện với nhau để quyết định giữ lại phần nào và thay đổi phần nào.

Trong GitHub, đó là khi hai người đang cố gắng làm việc trên cùng một phần mã cùng lúc và những thay đổi của họ xung đột với nhau. Đó là điều có thể khắc phục bằng giao tiếp và cộng tác!

9. Tạo thẻ phát hành (release tag) cho phiên bản cuối cùng của ứng dụng web.

- Truy cập trang kho lưu trữ trên GitHub.
- Nhấp vào tab "Releases".
- Nhấp vào nút "Create a new release".
- Đặt tên cho bản phát hành của bạn và thêm mô tả.
- Nhấp vào nút "Publish release".

Hãy tưởng tượng bạn và bạn bè của bạn đang cùng nhau giải một câu đố lớn. Bạn đã làm việc với nó trong một thời gian dài, và cuối cùng bạn cũng hoàn thành! Nhưng trước khi bạn tháo khối hình ra và đặt lại vào hộp, bạn muốn chụp ảnh nó để bạn có thể nhớ nó tuyệt vời như thế nào.

Trong GitHub, khi bạn và bạn bè cùng làm việc trên một dự án, mỗi người sẽ có máy tính riêng để viết và lưu mã của mình. Nhưng bạn cũng có một không gian dùng chung trên internet nơi bạn lưu trữ tất cả mã của mình, được gọi là kho lưu trữ.

Khi bạn hoàn thành dự án của mình, bạn có thể muốn tạo một thẻ phát hành. Thẻ phát hành giống như chụp ảnh câu đố đã hoàn thành của bạn. Đó là một cách để ghi nhớ dự án trông như thế nào tại một thời điểm cụ thể, trong trường hợp bạn muốn quay lại dự án sau.

Để tạo thẻ phát hành, bạn cần quyết định khi nào bạn muốn chụp "ảnh" dự án của mình. Đó có thể là khi bạn hoàn thành một tính năng lớn, khi bạn sửa một lỗi hoặc khi bạn chỉ muốn đánh dấu một thời điểm cụ thể. Khi đã quyết định, bạn có thể tạo thẻ phát hành trong GitHub.

Khi bạn tạo thẻ phát hành, bạn đang yêu cầu GitHub chụp "bức ảnh" về dự án của bạn tại thời điểm cụ thể đó. Bạn có thể đặt tên cho thẻ phát hành của mình, chẳng hạn như "Phiên bản 1.0" hoặc "Bản phát hành bản sửa lỗi". Bạn cũng có thể thêm mô tả về những gì đã thay đổi trong dự án của mình kể từ thẻ phát hành cuối cùng.

Khi bạn đã tạo thẻ phát hành, nó sẽ được lưu trong GitHub mãi mãi. Nó giống như chụp ảnh câu đố của bạn và cho vào album ảnh để bạn có thể nhớ nó tuyệt vời như thế nào. Bạn luôn có thể quay lại thẻ phát hành của mình sau và xem dự án của bạn trông như thế nào tại thời điểm cụ thể đó.

Vì vậy, tóm lại, thẻ phát hành giống như chụp ảnh bức tranh ghép hình đã hoàn thành của bạn. Đó là một cách để ghi nhớ dự án của bạn trông như thế nào tại một thời điểm cụ thể, trong trường hợp bạn muốn quay lại dự án đó sau. Trong GitHub, đó là khi bạn tạo một điểm đánh dấu đại diện cho một phiên bản cụ thể của dự án của bạn và lưu nó vào kho lưu trữ mãi mãi.

Bài tập 2: Tạo mô hình dự án Quản lý sinh viên và lưu vào thư mục **BTTH01** trong kho nói trên

- Lớp Sinh viên (Student)
- Lớp Danh sách Sinh viên (StudentDAO)

Ví dụ:

Triển khai các thao tác CRUD cho danh sách sinh viên sử dụng lập trình hướng đối tượng (OOP) trong PHP:

Định nghĩa một lớp **Student** với ba thuộc tính **(id, name, age)** và các phương thức getters và setters cho mỗi thuộc tính. Nó cũng định nghĩa một lớp **StudentDAO** với một mảng để lưu trữ danh sách sinh viên và các phương thức cho các thao tác CRUD **(create, read, update, delete và getAll)**.

Bài tập 3: Tạo file Danh sách sinh viên tùy chọn (.txt, .csv) với cấu trúc tự đề xuất

Ví dụ:

id,name,age,grade

1,John Smith,12,6

2,Jane Doe,13,7

3,Michael Johnson,11,5

4,Emily Davis,12,6

5,James Rodriguez,14,8

Bài tập 4: Viết ứng dụng với các chức năng

- Đọc dữ liệu từ file và lưu vào danh sách sinh viên để Hiển thị lên trang Web
- Xây dựng FORM để nhập dữ liệu Sinh viên mới và lưu lại và file đã cho ở Bước 3 (Kiểm tra sự trùng lặp).

HẾT