ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

ĐỒ ÁN CHUYÊN NGÀNH
HƯỚNG CÔNG NGHỆ PHẦN MỀM (CO4029)

Báo cáo Đồ án – version 0.2

GRADING SYSTEM

Giảng viên hướng dẫn: Lê Đình Thuận
Sinh viên thực hiện: Lê Minh Châu 2010947

Tp. Hồ Chí Minh, Tháng 3/2024

Mục lục

1 Tổng quan dự án

1.1 Bối cảnh

Trong bối cảnh nhu cầu tìm hiểu về ngành khoa học máy tính nói chung hiện nay vẫn đang rất cao. Điều này gây ra áp lực không nhỏ đối với người giảng dạy do việc số người tham gia quá cao sẽ gây khó khăn trong việc quản lý, đánh giá cũng như đảm bảo chất lượng học viên. Ngoài việc người giảng dạy phải update tài liệu lớp học liên tục do đặc điểm ngành nghề, việc chấm điểm, đánh giá gian lận nhằm đảm bảo chất lượng đầu ra cũng tiêu tốn rất nhiều thời gian và công sức của người giảng dạy.

Đối với vấn đề gian lận, trong hội thảo "Đánh giá hoạt động học tập trong dạy học trực tuyến: Từ thiết kế đến triển khai" do Ban Đại Học, ĐHQG-HCM tổ chức ngày 26/08/2021, Theo TS Nguyễn Tấn Đại, trước khi thi trực tuyến, ProctorU - công ty cung cấp dịch vụ theo dõi người dự thi và kiểm tra ID, đã bắt quả tang sinh viên gian lận chỉ dưới 1% trong số 340.000 bài thi từ tháng 1-3/2020. Khi áp dụng thi trực tuyến có giám thị theo dõi từ xa, tỷ lệ sinh viên gian lận đã tăng trên 8% trong 1,3 triệu bài thi từ tháng 4-6/2020.

Đối với ngành khoa học máy tính, gian lận có thể là copy code của một ai đó, hoặc sử dụng công cụ (tiêu biểu có thể kể đến AI) trong các bài kiểm tra. Điều này sẽ làm giảm chất lượng giáo dục cũng như chất lượng học viên, và nhiều hệ lụy khác, không chỉ ảnh hưởng cá nhân đó mà cả tổ chức giảng dạy.

Mục tiêu của dự án:

- Giảm thiểu tình trạng gian lận, tăng chất lượng đầu ra của học viên
- Giảm thời gian xử lý của người giảng dạy trong việc chấm điểm và đánh giá gian lận
- Có thể sử dụng bởi nhiều cơ sở giảng dạy khác nhau
- Có thể sử dụng với một server vừa phải

1.2 Ý tưởng

Grading System là hệ thống chấm điểm bài kiểm tra trong ngành khoa học máy tính bằng cách sử dụng quá trình CI của các công cụ lưu trữ mã nguồn nhằm tiết kiệm thời gian xử lý một lượng lớn mã nguồn, đồng thời sử dụng những hệ thống đánh giá gian lận được phát triển và sử dụng bởi các tổ chức tin cậy để đánh giá bài làm của học viên, sau đó sẽ được đánh giá lại bởi giảng viên nhằm xác định chính xác bài làm gian lận và có biện pháp xử lý phù hợp.

Grading System sẽ tiết kiệm thời gian cho người dùng bằng việc tận dụng sức mạnh máy tính tương tự như một số online judge. Thông thường, quy trình hoạt động của các online judge là người dùng sẽ viết code trên text editor, sau khi hoàn thành sẽ gửi bài làm đến server. Server sẽ sử dụng IDE của mình để run bài làm với các testcase được tạo sẵn, sau đó trả về kết quả cho người dùng. Đối với quy trình này, người dùng không cần bỏ công sức trong việc chấm điểm, tuy nhiên, với một hệ thống nhỏ, việc run code trên IDE có thể khiến server bị quá tải, do việc run code tiêu tốn rất nhiều tài nguyên máy tính. Mã nguồn càng lớn, hệ thống càng tiêu

tốn nhiều tài nguyên. Ngoài ra, lượng request đến server cũng là một vấn đề cần giải quyết.

Đặc điểm của hệ thống Grading System là tận dụng khả năng của các hệ thống khác sẵn có, được phát triển qua nhiều năm và được tin tưởng bởi cộng đồng phát triển nhằm nâng cao sự ổn định, chính xác và giảm tải cho server cục bộ, cũng như công sức phát triển hệ thống nhưng vẫn đạt được chất lượng nhất định. Đồng thời tiết kiệm thời gian của người dùng bằng việc tự động hóa một số công việc trong quá trình chấm điểm bài kiểm tra của học viên.

Cơ chế hoạt động của hệ thống được mô tả như sau:

1. Người dùng sử dụng công cụ lưu trữ mã nguồn tạo repository chứa đề, testcase mẫu, các file workflow tương ứng với các ngôn ngữ được cho phép (có thể tự tạo, hoặc chọn template có sẵn của hệ thống).
2. Học viên sẽ lấy repository mà giảng viên đã tạo, chọn ngôn ngữ thích hợp, có thể config lại tùy theo nhu cầu cá nhân, giải quyết bài toán và commit lại vào repository của mình.
3. Sau khi học viên commit, quá trình CI sẽ tự động chạy dựa trên file workflow và gửi kết quả về hệ thống.
4. Hệ thống sử dụng công cụ đánh giá gian lận kiểm tra bài làm của học viên, xử lý và trả về kết quả và thông tin chi tiết (điểm, testcase passed, cheating status) cho người dùng qua email dưới dạng file csv.

Một số ưu điểm dựa trên chiến lược phát triển của hệ thống:

- Ổn định, đáng tin cậy
  Hệ thống sử dụng các hệ thống sẵn có để chạy và đánh giá bài làm của học viên. Các hệ thống này thường có quy mô lớn và được cộng đồng tin tưởng. Do đó, có thể đảm bảo được việc hạn chế downtime, đồng thời giảm lỗi từ phía server khi compile bài làm.
- Giảm tải cho server
  Bài làm của học viên được compile và lưu trữ trên các công cụ lưu trữ mã nguồn. Hệ thống không cần tốn chi phí cho những hành động này. Vì vậy, những tài nguyên đó có thể sử dụng để thực hiện các hoạt động khác.
- Hỗ trợ đa ngôn ngữ
  Các bài làm có thể sử dụng nhiều loại ngôn ngữ khác nhau, tùy từng học viên. Đối với một số môn học (DSA, ...), ngôn ngữ thường không chiếm vai trò quá quan trọng, do đó, học viên có thể sử dụng ngôn ngữ phù hợp để làm bài sẽ giúp hỗ trợ việc học tốt hơn.
- Tự động hóa
  Hầu hết các quá trình chấm điểm và đánh giá gian lận bài kiểm tra đều sử dụng công cụ tự động hóa, giảm thiểu công sức cần phải bỏ ra của giảng viên.

1.3 Phương hướng phát triển

Lúc trước, việc gian lận có thể là copy bài làm, hoặc cùng hợp tác để vượt qua kì thi. Các công cụ hiện có thường có thể đánh giá được do đã được phát triển qua nhiều năm để đối phó với tình trạng này.

Tuy nhiên, hiện nay, rất nhiều công cụ khác rất có tiềm năng để vượt qua được những công cụ đánh giá gian lận hiện tại, tiêu biểu là AI. Dù vậy, hệ thống vẫn có thể khả thi do hiện tại, các đoạn code mà AI sinh ra thường không quá khác biệt. Nếu có một vài học viên sử dụng, hệ thống vẫn có thể nhận biết được.

Nhưng tiềm năng của AI đối với việc này hoàn toàn không thể dự đoán được. Một vấn đề phát sinh, nếu AI có thể sinh mã hoàn toàn khác giữa các học viên, mọi chuyện sẽ phức tạp hơn nhiều. Hệ thống lúc này cần phải có công cụ hiệu quả hơn để đối phó với tình huống này. Cũng chính vì vậy, hệ thống được xây dựng theo hướng có thể sử dụng nhiều công cụ đánh giá khác nhau thay vì thiết kế riêng biệt dành riêng cho hệ thống. Đây chính là thách thức lâu dài mà hệ thống cần phải phát triển thêm.

Ngoài ra, hệ thống được thiết kế để sử dụng công cụ đánh giá gian lận của bên thứ ba. Do đó, trong tương lai hệ thống hoàn toàn có thể áp dụng một công cụ khác được thiết kế không phải dành riêng cho ngành khoa học máy tính. Việc mở rộng khả năng xử lý gian lận sang nhiều lĩnh vực khác cũng là hướng phát triển quan trọng của hệ thống này.

2 Công cụ liên quan

2.1 Git, Github & Github Action

Git

Git là một hệ thống quản lý phiên bản phân tán (Distributed Version Control System – DVCS), nó là một trong những hệ thống quản lý phiên bản phân tán phổ biến nhất hiện nay. Git cung cấp cho mỗi lập trình viên kho lưu trữ (repository), lưu lại tất cả các file trong toàn bộ dự án và ghi lại toàn bộ lịch sử thay đổi của file. Mỗi sự thay đổi được lưu lại sẽ được và thành một version (phiên bản).

Github

Github là nền tảng lưu trữ cloud-based được thiết kế đặc biệt cho các dự án phần mềm. Github có đầy đủ những tính năng của Git, ngoài ra nó còn bổ sung những tính năng về social để các developer tương tác với nhau.

CI/CD

Continuous Integration / Continuous Deployment (CI/CD) là quá trình tự động quá việc kiểm tra mã nguồn (test), xây dựng (build) và triển khai (deploy) của dự án.
Continuous Intergration (CI): là quá trình tự động hóa trong phát triển phần mềm, cho phép các thành viên trong đội kiểm tra và hợp nhất các mã nguồn vào Repository chung. Quá trình CI được diễn ra như sau:

1. Tích hợp mã nguồn: Các lập trình viên commit và push code của mình lên repository.
2. Kích hoạt CI: CI server giám sát repository và kiểm tra liên tục sự xem có sự thay đổi nào hay không. Khi phát ra sự thay đổi, CI server bắt đầu quá trình CI.
3. Build code: Với code mới nhất trên repository, CI server sẽ build mã nguồn này thành một phần mềm hoàn chỉnh.
4. Chạy các test case: Sau khi quá trình build hoàn tất, công cụ CI server bắt đầu chạy kiểm tra đơn vị (unit test), kiểm tra tích hợp (integration test), kiểm tra giao diện (UI test) hoặc các kiểm tra khác tùy thuộc vào yêu cầu của dự án.
5. Đưa ra thông báo: Nếu có lỗi trong quá trình kiểm tra, CI server sẽ đưa ra thông báo lỗi cho các lập trình viên biết. Các thông báo này thường được gửi qua email, chat...
6. Hợp nhất mã nguồn: Nếu không có lỗi, CI server sẽ tự động hợp nhất mã nguồn của các lập trình viên vào một phiên bản mới nhất của phần mềm. Phiên bản mới này được lưu trữ trong kho chung của dự án và có thể được triển khai vào các môi trường khác nhau của dự án.

Continuous Deployment (CD): Là quá trình triển khai tự động sau khi quá trình CI được diễn ra thành công. Quá trình CD sẽ được kích hoạt để triển khai ứng dụng của chúng ta vào các môi trường của dự án. Các gói phần mềm đã build thành công được triển khai bằng cách sử dụng các công cụ tự động hoặc được triển khai thủ công.

Github Action

Github Action là nền tảng miễn phí do Github cung cấp để tự động hóa quá trình CI/CD, cho phép người dùng tự định nghĩa các file workflows, tự động hóa các quy trình, hoạt động trong phát triển phần mềm. Một số đặc điểm nổi bật:

- Tích hợp sẵn với Github: Vì Github Actions được phát triển bởi Github nên nó được tích hợp sẵn với nền tảng Github. Do đó, người dùng có thể sử dụng các tính năng của Github Actions trực tiếp từ trang web của Github.
- Hỗ trợ chạy trên nhiều hệ điều hành: Github Actions có thể chạy trên nhiều hệ điều hành khác nhau: Windows, macOS, Ubuntu... Người dùng có thể kiểm tra ứng dụng của mình trên nhiều nền tảng khác nhau và đảm bảo rằng nó hoạt động đúng cách trên mọi nền tảng.
- Hỗ trợ chạy trên các môi trường ảo: Cho phép người dùng thực hiện trong các môi trường ảo khác nhau, người dùng có thể kiểm tra ứng dụng của mình trên nhiều môi trường đấy một cách dễ dàng. Người dùng có thể định nghĩa ngôn ngữ, các version của nó để run code.
- Tích hợp các công cụ bên thứ ba: Các công cụ bên thứ ba có thể được tích hợp vào quá trình CI/CD. Ví dụ, ta có thể sử dụng Github Actions để triển khai ứng dụng của mình lên AWS hoặc Azure.

Người dùng có thể sử dụng Github Hosted để chạy chương trình trên Github hoặc sử dụng Self Hosted Runner nếu có thêm các depedency đặc biệt.

Trong ***Grading System***, người dùng có thể sử dụng Github như một công cụ lưu trữ, tạo môi trường và run code bằng các file workflows. Số điểm của học viên được tính dựa vào kết quả của quá trình CI trên Github.

2.2 Công cụ chống gian lận

Quá trình CI trên Github chỉ có thể phát hiện lỗi, kiểm tra các testcase, ... và hoàn toàn không thể phát hiện gian lận. Do đó, hệ thống cần sử dụng một công cụ khác phát hiện gian lận riêng dựa trên những đoạn mã đủ tiêu chuẩn trên Github.

Một trong những công cụ nổi bật chống gian lận là MOSS (Measure Of Software Similarity), được phát triển bởi đại học Stanford. Ứng dụng chính của MOSS là phát hiện đạo văn trong các lớp học lập trình. Một trong những ưu điểm nổi bật của MOSS là khả năng phát hiện những đoạn mã cơ bản, phổ biến trong nội bộ lớp học, hoặc cho phép những đoạn mã mẫu có thể copy để sử dụng. Ngoài ra, MOSS cũng có thể phát hiện được những trường hợp có thay đổi tên biến, tên hàm, thêm comment, ... nhằm đánh lạc hướng người kiểm tra.

Tuy nhiên, MOSS không phải là công cụ tự động hóa hoàn toàn quá trình kiểm tra gian lận. MOSS chỉ có thể chỉ ra và đánh dấu những điểm bất thường. Việc quyết định gian lận hay không vẫn cần người giảng dạy thực hiện. Dù vậy, MOSS tiết kiệm rất nhiều thời gian cho giảng viên trong quá trình điều tra gian lận giữa các học viên.

2.3 Công cụ trong hệ thống Grading System

Các công cụ trên chỉ là một vài trong số những công cụ có thể sử dụng cho hệ thống. Như đã đề cập ở trên, hệ thống được phát triển theo hướng có thể lựa chọn công cụ khác (Gitlab thay cho Github, sử dụng công cụ đánh giá gian lận khác, ...), có thể tích hợp vào cùng hệ thống (có thể sử dụng cả Gitlab và Github cho Repo và testing, intergrate nhiều công cụ chống gian lận khác nhau vào cùng một hệ thống, ...) hoặc sử dụng một hệ thống testing và cheating được thiết kế phù hợp với nghiệp vụ riêng. Đây cũng là một trong những hướng phát triển lâu dài của hệ thống Grading System nhằm hướng đến thích nghi với quá trình phát triển cũng như có thể sử dụng trong nhiều lĩnh vực khác nhau.

Về cơ bản, quá trình hoạt động của Grading System khi tích hợp các công cụ như sau:

1. Người dùng sẽ tạo Repo đề kiểm tra trên nền tảng lưu trữ của họ (ở đây là Github).
2. Học viên sẽ hoàn thành đề thi và nộp lại hệ thống.
3. Hệ thống sẽ tự động sử dụng quá trình CI của Github để chạy mã.
4. Sau khi có kết quả, người dùng sẽ chọn lọc và gửi những đoạn mã để kiểm tra gian lận, có thể là loại những bài làm compile lỗi, ...
5. Người dùng dựa trên kết quả trả về, review lại các bài làm và đánh giá.

Như vậy, hệ thống Grading System không phải là một hệ thống tự động hoàn toàn. Dù vậy, nếu so với quá trình chấm bài truyền thống, sử dụng Grading System sẽ giảm thiểu công sức, thời gian phải bỏ ra khi mà hầu hết các quá trình đều đã được tự động hóa. Hơn nữa, những quá trình cần sự can thiệp của con người trong hệ thống đều là những quá trình cần thiết nhằm đảm bảo việc chấm gian lận thực sự là đúng đắn, do công cụ chỉ hỗ trợ con người, nó không thể hoàn toàn thay thế được con người, đặc biệt là trong những trường hợp phức tạp như việc này, ít nhất là cho đến thời điểm hiện tại.

3 Activity

Hệ thống sẽ cung cấp hai lựa chọn là Student host và Teacher host.

3.1 Student host activity

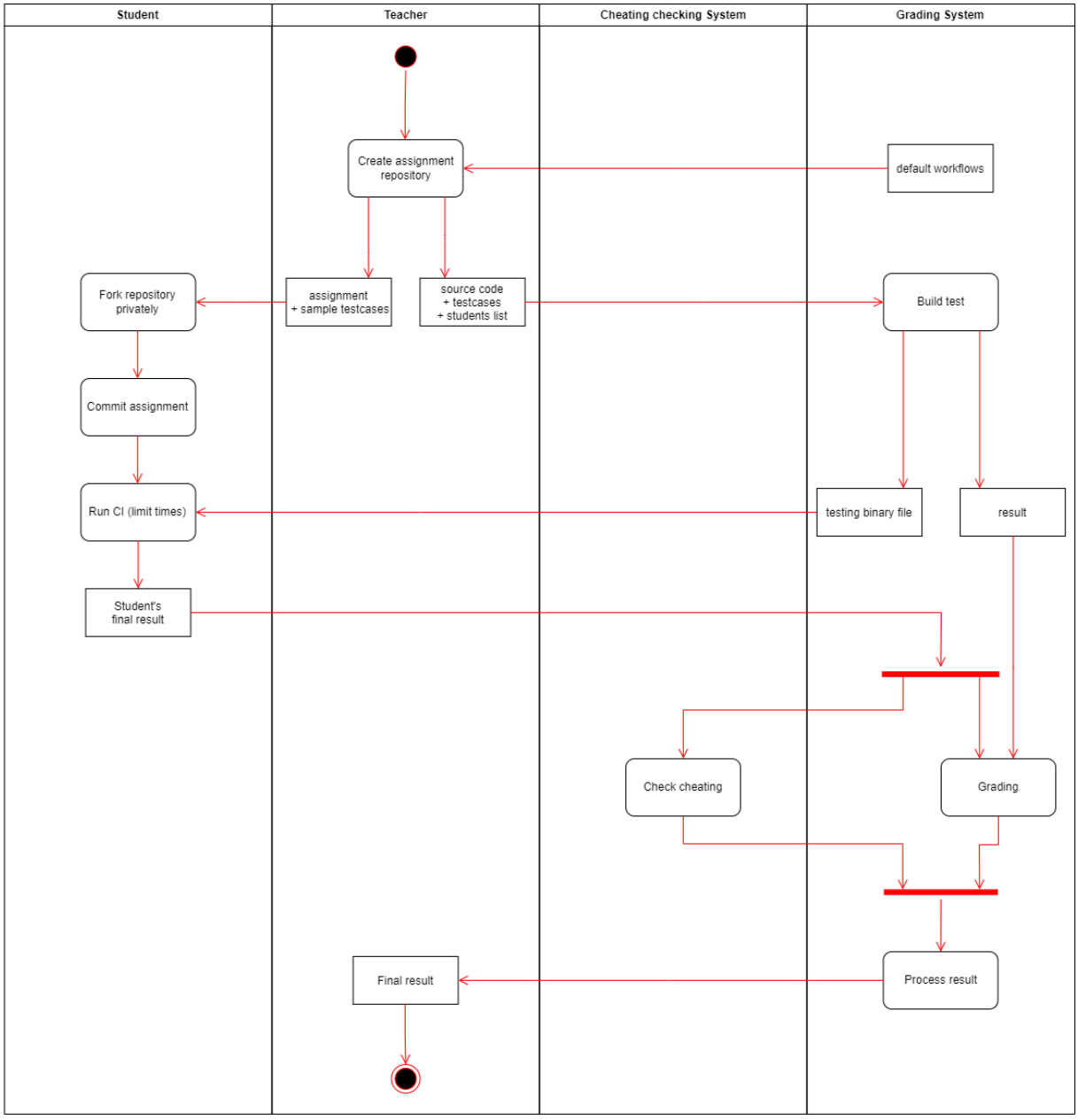
3.1.1 Studen host

Đối với Student host, hệ thống sẽ sử dụng quá kết quả của quá trình CI bên phía Github của học viên và đánh giá thông qua kết quả của source code mà người dùng cung cấp. Luồng thực thi của hệ thống như sau:

1. Người dùng sẽ tạo một repo đề dành cho học viên, danh sách học viên, source code cho bài kiểm tra và testcases để cung cấp cho hệ thống.
2. Hệ thống sẽ run file test tạo ra hai file: một file chứa testcase được build thành một binary file, file còn lại là kết quả của source code được người dùng cung cấp.
3. Học viên sẽ fork privately repo đề, hoàn thành bài kiểm tra và commit kết quả lên repo cá nhân. Người dùng có thể giới hạn số lần commit của học viên. Kết quả của quá trình CI có thể ẩn đối với học viên, và sẽ được gửi đến server cùng source code của lần commit đó.
4. Sau khi đến hạn nộp bài, hệ thống sẽ gửi source code của học viên đến công cụ kiểm tra gian lận, kết quả quá trình CI của học viên sẽ được so sánh với kết quả run bởi source code do người dùng cấp và đánh giá kết quả.
5. Kết quả của quá trình chấm gian lận và so sánh với kết quả của người dùng sẽ được xử lý, tổng hợp và gửi lại cho người dùng.

3.1.2 Student host activity diagram

Quá trình hoạt động của Student host được mô tả bằng sơ đồ sau:



Hình 1: *System Activity diagram - Student host*

3.2 Teacher host activity

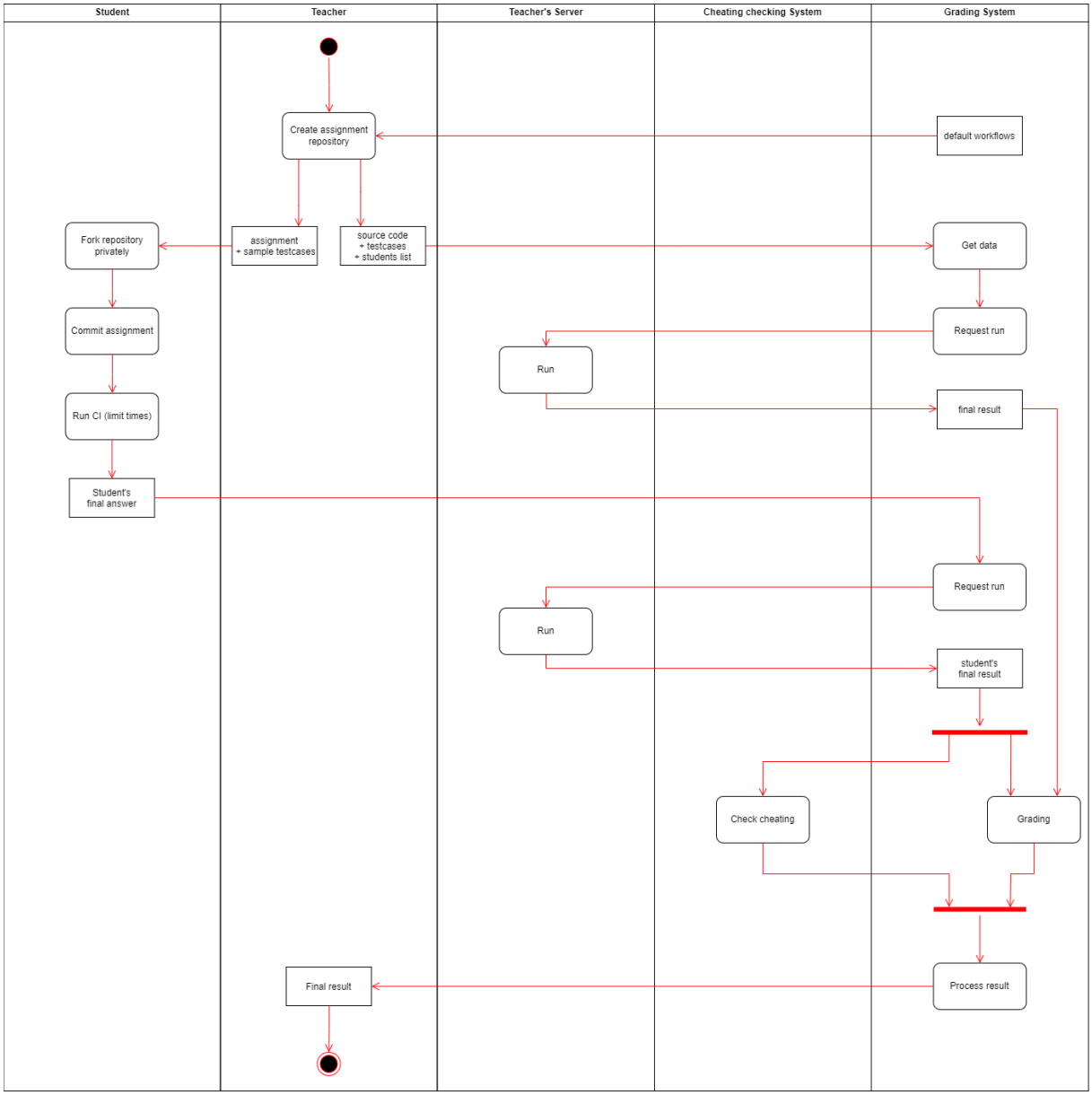
3.2.1 Teacher host

Đối với Teacher host, hệ thống sẽ sử dụng API IDE của người dùng cung cấp để run các source code của người dùng và của học viên. Kết quả vẫn sẽ được lấy và đánh giá tương tự như Student host. Luồng thực thi của hệ thống như sau:

1. Người dùng sẽ tạo một repo đề dành cho học viên, danh sách học viên, source code cho bài kiểm tra và testcases để cung cấp cho hệ thống.
2. Học viên sẽ fork privately repo đề, hoàn thành bài kiểm tra và commit kết quả lên repo cá nhân. Người dùng có thể giới hạn số lần commit của học viên.
3. Hệ thống sẽ chuyển tiếp request đến server IDE của người dùng, run source code của học viên và lấy lại kết quả. Source code của lần commit đó sẽ được gửi đến hệ thống. Kết quả này sẽ không được gửi lại cho học viên.
4. Sau khi đến hạn nộp bài, hệ thống sẽ gửi source code của học viên đến công cụ kiểm tra gian lận, kết quả run source code của học viên sẽ được so sánh với kết quả run bởi source code do người dùng cấp và đánh giá kết quả.
5. Kết quả của quá trình chấm gian lận và so sánh với kết quả của người dùng sẽ được xử lý, tổng hợp và gửi lại cho người dùng.

3.2.2 Teacher host activity diagram

Quá trình hoạt động của Teacher host được mô tả bằng sơ đồ sau:



Hình 2: *System Activity diagram - Teacher host*

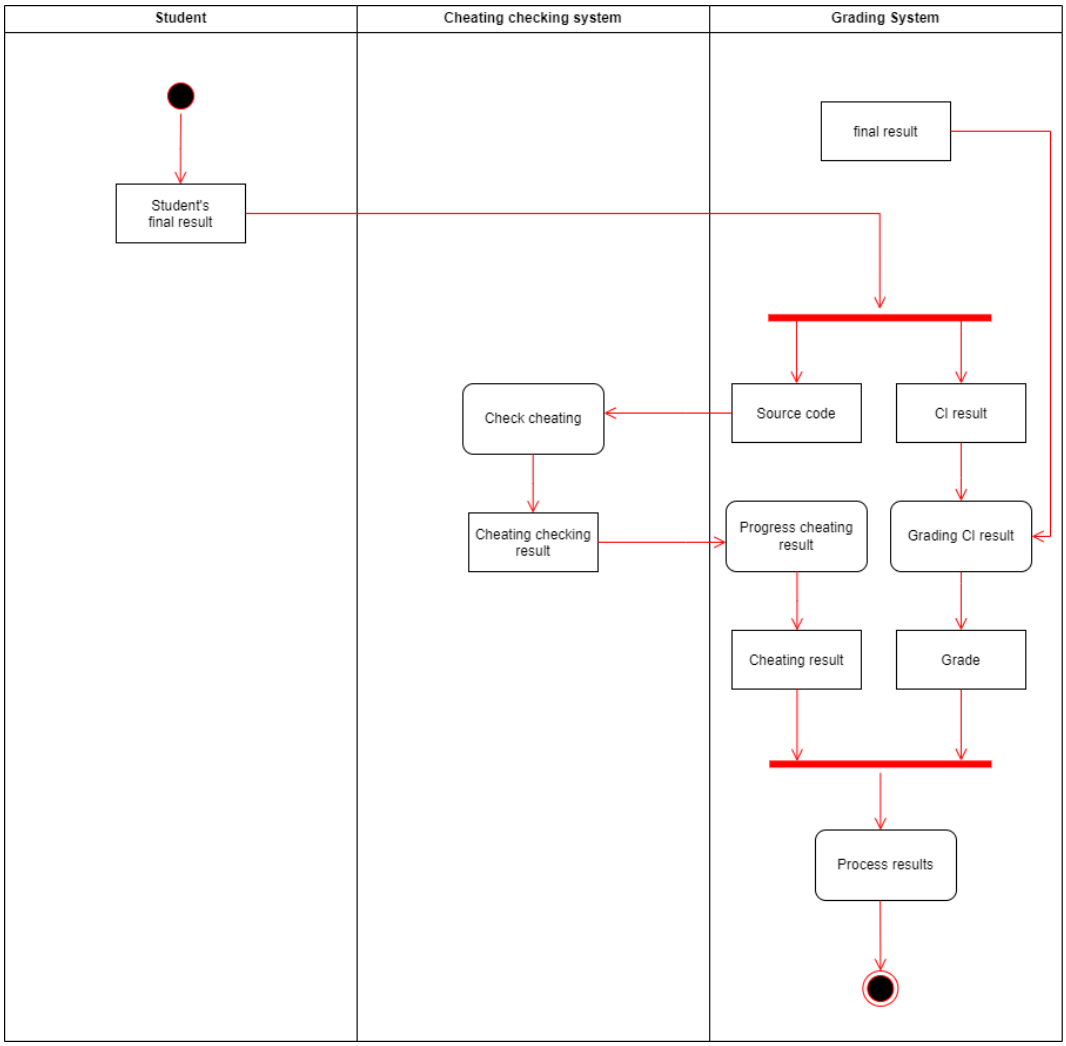
3.3 Grading and checking activity

3.3.1 Grading and Checking

Grading and Checking là module chính của hệ thống. Mặc định, hệ thống sẽ chấm theo phần trăm testcase passed. Tuy nhiên, nếu người dùng có phương pháp chấm điểm khác, hệ thống có thể sử dụng cách chấm đó. Ngoài ra, người dùng cũng có thể chọn công cụ đánh giá gian lận mà mình tin tưởng. Luồng thực thi của module như sau:

1. Module sẽ nhận kết quả của người dùng cùng với kết quả và source code của học viên.
2. Source code của học viên sẽ được gửi đến công cụ đánh giá gian lận, đồng thời, kết quả từ source code của học viên sẽ được so sánh với kết quả từ source code của người dùng.
3. Kết quả kiểm tra gian lận và điểm số sẽ được xử lý và gửi lại cho người dùng.

3.3.2 Grading and Checking activity diagram



Hình 3: *Grading and Checking result activity diagram*

4 Functional & Non-Functional Requirements

4.1 Functional Requirements

- Chọn kế hoạch (Choose Planning): Tùy theo nhu cầu sử dụng, điều kiện, ... mà người dùng có thể chọn kế hoạch thực thi cho bài kiểm tra của mình (Student host và Teacher host).
- Hướng dẫn (View instructions): Xem thông tin, hướng dẫn chi tiết về các plan.
- Tạo không gian kiểm tra (Create asignment space): Người dùng sẽ tạo nơi lưu trữ các thông tin về assignment cùng các thông tin liên quan như source code, kết quả, testcases, ...
- Upload tài liệu (Upload file): Upload các tài liệu liên quan đến bài kiểm tra phục vụ cho quá trình chấm điểm và đánh giá gian lận (source code, testcases, students list).
- Tạo workflow (Generate workflow): các file workflow là các instructions để tạo môi trường, run code, request API, ... Hệ thống sẽ cung cấp một số template, người dùng có thể sử dụng template hoặc customize template thành một file workflow, một số trường trong file workflow buộc phải giữ nguyên.
- Xem thống kê (Statistic): Xem thông tin thống kê về các bài làm, điểm số, tình trạng gian lận, ...
- Đối với Teacher host:
  - Cấp server (Explore server): Người dùng sẽ cung cấp API run code cho hệ thống, hệ thống sẽ dựa trên API này để gửi source code và nhận lại result.
- Tạo file binary (Generate binary file): Nhận file test của người dùng và tạo một file thực thi, khi học viên commit sẽ sử dụng file này để run code.
- Chấm điểm (Grading): Nhận kết quả commit của học viên và file result run bởi source code của người dùng và đánh giá kết quả học viên.
- Chấm gian lận (Detect cheating): Kiểm tra mức độ gian lận trong các bài làm của học viên. Các bài làm này sẽ được gửi đến công cụ chấm gian lận / công cụ kiểm tra gian lận của hệ thống.
- Xử lý kết quả (Process result): Kết quả gian lận và điểm số sẽ được xử lý, gộp lại thành một file csv, cùng với biểu đồ thống kê.
- Thông báo kết quả: Khi có kết quả trả về, hệ thống sẽ gửi kết quả đến email của người dùng.

4.2 Non-Functional Requirements

Tương thích

- Tương thích với trình duyệt Chrome, Edge.

Đơn giản

- Mỗi chức năng không quá 4 thao tác.
- Người dùng có thể sử dụng hệ thống sau khi xen hướng dẫn sử dụng.

Chịu tải

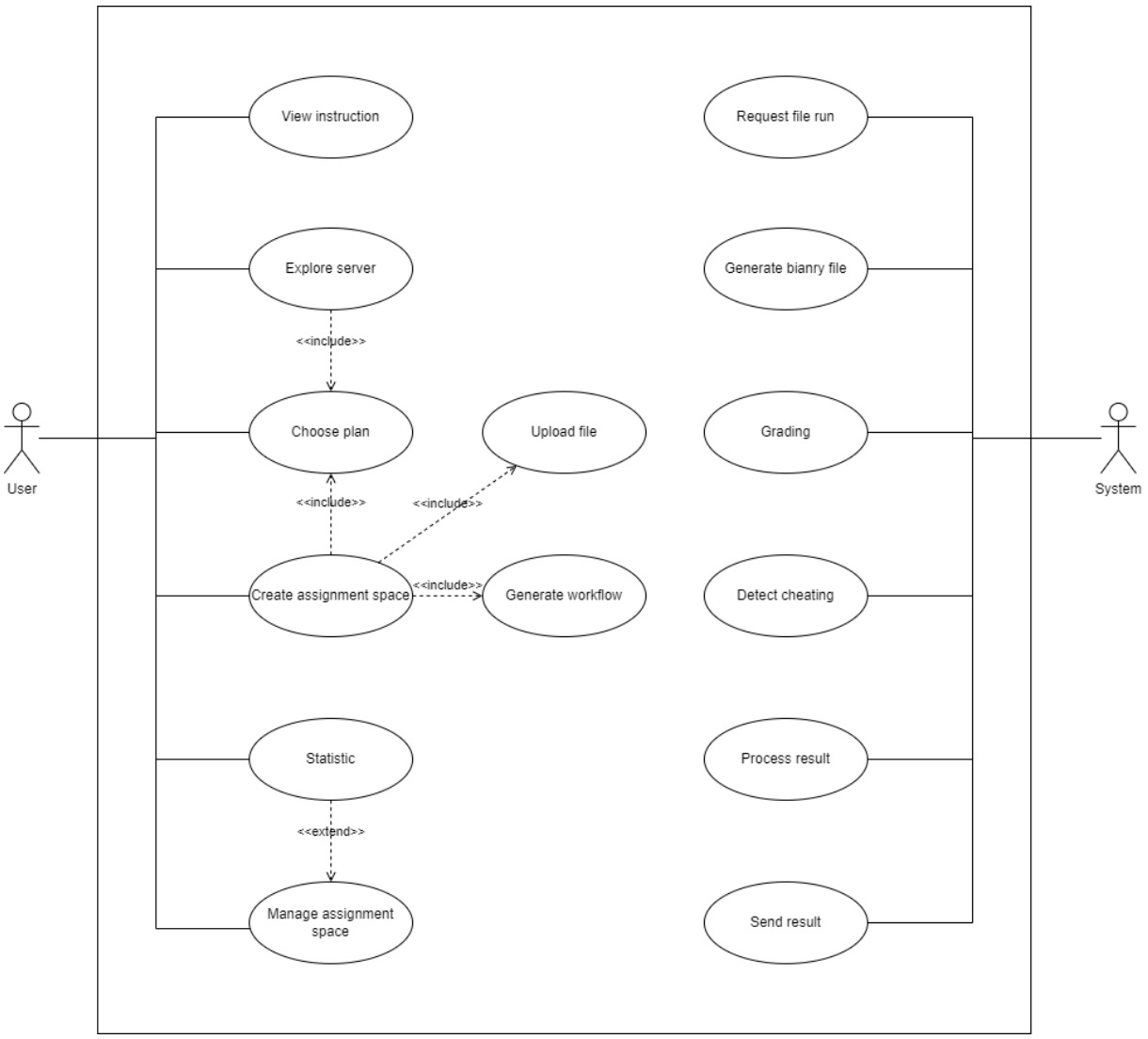
- Đáp ứng được với lớp học tối đa 100 người.

Localization

- Hệ thống hỗ trợ ngôn ngữ Tiếng Việt.
- Hệ thống hiển thị giờ dựa trên đồng hồ 24 giờ.
- Hệ thống hiển thị ngày theo định dạng dd/mm/yyyy.

5 Usecases

5.1 System diagram



Hình 4: *System diagram*

5.2 Usecase senario

5.2.1 Choose plan

Use-case name	**Choose plan**
Create by	Lê Minh Châu
Date created	28/03/2024
Actor	User
Description	Người dùng chọn plan phù hợp với bài kiểm tra cũng như điều kiện hiện có
Trigger	Người dùng chọn vào một trong các thẻ plan
Pre-condition	Người dùng truy cập vào trang web Grading System hoặc người dùng đang tạo một assignment space
Post-condition	Người dùng hiểu được cơ chế và chọn được plan phù hợp với bài kiểm tra của mình
Normal flow	1. Người dùng truy cập vào Grading System 2. Hệ thống trả về trang thông tin sơ bộ của các plan 3. Người dùng chọn plan cần tìm hiểu, xem thông tin chi tiết về plan đã chọn 4. Người dùng chọn "Use plan"
Alternative flow	Tại bước 1 1.1. Người dùng đang tạo một assignment space mới 2.1. Hệ thống trả về thông tin sơ bộ của các plan 3.1. Người dùng chọn plan cần tìm hiểu, xem thông tin chi tiết về plan đã chọn 4.1. Người dùng chọn "Use plan" Tại bước 3 3.2. Người dùng có thể chọn "Use plan" trực tiếp khi hover vào thẻ plan
Exception	Không

Bảng 1: Đặc tả cho usecase Choose plan

5.2.2 View instruction

Use-case name	**View instructions**
Create by	Lê Minh Châu
Date created	28/03/2024
Actor	User
Description	Người dùng có thể chọn chức năng này để xem các chỉ dẫn khi cần thiết
Trigger	Người dùng chọn "View instruction"
Pre-condition	Không
Post-condition	Người dùng có thể tìm được thông tin cần thiết cho vấn đề của mình
Normal flow	1. Người dùng truy cập vào Grading system 2. Người dùng chọn "View instructions" 3. Hệ thống trả về một số hướng dẫn cần thiết và các đường liên kết liên quan 4. Người dùng tìm kiếm hướng dẫn phù hợp với trường hợp của mình
Alternative flow	1.1. Người dùng có thể đang truy cập vào bất kì trang nào của hệ thống 2.1. Người dùng chọn "View instructions" 3.1. Hệ thống trả về một số hướng dẫn cần thiết và các đường liên kết liên quan 4.1. Người dùng tìm kiếm hướng dẫn phù hợp với trường hợp của mình
Exception	Không

Bảng 2: Đặc tả cho usecase View instruction

5.2.3 Create assignment space

Use-case name	**Create assignment space**
Create by	Lê Minh Châu
Date created	28/03/2024
Actor	User
Description	Khi có assignment mới, người dùng có thể sử dụng chức năng này để tạo một không gian assignment mới phù hợp với bài kiểm tra của mình
Trigger	Người dùng chọn "New space"
Pre-condition	Người dùng đã đăng nhập vào hệ thống
Post-condition	Người dùng tạo thành công space mới phù hợp với bài kiểm tra của mình
Normal flow	1. Người dùng chọn "New space" 2. Hệ thống chuyển hướng người dùng đến trang tạo space 3. Người dùng cung cấp các thông tin cần thiết được yêu cầu bởi hệ thống để tạo một space mới 4. Người dùng nhấn "Create space" 5. Hệ thống sẽ chuyển hướng người dùng đến space đã được tạo
Alternative flow	Tại bước 4 3.1. Người dùng nhấn "Cancel" 4.1. Hệ thống sẽ hiện popup yêu cầu người dùng xác nhận hành động hủy 5.1. Người dùng xác nhận hành động của mình 6.1. Hệ thống chuyển hướng người dùng đến trang chủ của hệ thống Tại bước 5.1 5.2. Người dùng không xác nhận hành động của mình 6.2. Hệ thống đóng popup để người dùng tiếp tục tạo space
Exception	Tại bước 4, nếu người dùng không cung cấp đủ các thông tin yêu cầu, hệ thống không tạo space và sẽ cảnh báo lỗi các trường thông tin còn thiếu

Bảng 3: Đặc tả cho usecase Create assignment space

5.2.4 Generate workflow

Use-case name	**Generate workflow**
Create by	Lê Minh Châu
Date created	28/03/2024
Actor	User
Description	Người dùng chọn file workflow, hoặc chỉnh sửa file workflow phù hợp với nhu cầu cũng như yêu cầu của bài kiểm tra
Trigger	Người dùng chọn các option workflow khi tạo hoặc chỉnh sửa space
Pre-condition	Người dùng đang tạo hoặc chỉnh sửa assignment space
Post-condition	Người dùng tạo được file workflow phù hợp với yêu cầu bài kiểm tra
Normal flow	1. Người dùng chọn "Use template" dành cho workflow 2. Hệ thống hiện popup các template của các workflow 3. Người dùng chọn template workflow tùy theo nhu cầu sử dụng và chọn "Use template" 4. Hệ thống lưu lựa chọn và đóng popup
Alternative flow	Tại bước 1 1.1. Người dùng chọn "New workflow" 2.1. Hệ thống hiện popup edit cho workflow 3.1. Người dùng chỉnh sửa file workflow và chọn "Confirm" 4.1. Hệ thống lưu các chỉnh sửa và đóng popup Tại bước 3 3.2. Người dùng chọn "Edit" 4.2. Người dùng chỉnh sửa và chọn "Save" 5.2. Hệ thống lưu các chỉnh sửa và đóng popup Tại bước 3, 3.1, 4.2 Người dùng nhấn "Cancel", hệ thống đóng popup
Exception	Tại 3.1, 4.2 Nếu người dùng không cung cấp đủ thông tin hoặc sai định dạng, hệ thống sẽ báo lỗi tại nơi gây lỗi và yêu cầu người dùng chỉnh sửa lại

Bảng 4: Đặc tả cho usecase Generate workflow

5.2.5 Upload file

Use-case name	**Upload file**
Create by	Lê Minh Châu
Date created	28/03/2024
Actor	User
Description	Người dùng cung cấp các file cần thiết mà hệ thống yêu cầu để tạo assignment space
Trigger	Người dụng chọn "Upload"
Pre-condition	Người dùng đã đăng nhập và đang tạo hoặc chỉnh sửa assignment space
Post-condition	Người dùng tải lên các file cần thiết để hệ thống tạo hoặc chỉnh sửa assignment space
Normal flow	1. Người dùng chọn "Upload" 2. Hệ thống hiện popup upload file cho người dùng 3. Người dùng chọn các file cần thiết để tải lên hệ thống và chọn "Confirm" 4. Hệ thống lưu các file và đóng popup
Alternative flow	Tại bước 1, người dùng chọn "Upload with binary" 2.1. Hệ thống hiện popup upload file cho người dùng 3.1. Người dùng chọn các file cần thiết để tải lên hệ thống và chọn "Confirm" 4.1. Hệ thống lưu các file và đóng popup Tại bước 2, 3 Người dùng chọn "Cancel", hệ thống không lưu các file và đóng popup
Exception	Tại bước 3 Người dùng không tải lên đủ các file yêu cầu, hệ thống sẽ báo lỗi thiếu file và yêu cầu người dùng cung cấp thêm file

Bảng 5: Đặc tả cho usecase Upload file

5.2.6 Manage assignment spaces

Use-case name	**Manage assignment spaces**
Create by	Lê Minh Châu
Date created	28/03/2024
Actor	User
Description	Người dùng thực hiện các thao tác xem, sửa, xóa, các thông tin liên quan đến space đã tạo
Trigger	Người dùng chọn "My spaces"
Pre-condition	Người dùng đã đăng nhập vào Grading System
Post-condition	Người dùng có thể thực hiện các thao tác quản lý các space đã tạo trước đó
Normal flow	1. Người dùng chọn "My spaces" 2. Hệ thống chuyển người dùng đến trang thông tin các space mà người dùng đã tạo 3. Người dùng chọn space cần quản lý trong danh sách space của mình 4. Người dùng thực hiện các thao tác quản lý space và chọn "Save" 5. Hệ thống lưu các thay đổi và điều hướng người dùng về trang chính
Alternative flow	Tại bước 4 4.1. Người dùng chọn "Cancel" 5.1. Hệ thống không lưu các thao tác người dùng vừa thực hiện và điều hướng người dùng về tranh chính
Exception	Tại bước 4 Các thông tin quản lý sẽ gây lỗi được cảnh báo đến người dùng và yêu cầu người dùng chỉnh sửa lại

Bảng 6: Đặc tả cho usecase Manage assignment spaces

5.2.7 Statistic

Use-case name	**Statistic**
Create by	Lê Minh Châu
Date created	29/03/2024
Actor	User
Description	Người dùng xem các thông tin mang tính thống kê về bài kiểm tra sau khi hoàn thành
Trigger	Người dùng chọn "Statistic" tại một assignment space
Pre-condition	Người dùng cần đăng nhập, đã tạo và hoàn thành ít nhất một assignment space
Post-condition	Người dùng xem được thông tin thống kê về điểm số, gian lận của các bài làm trong space
Normal flow	1. Người dùng truy cập vào trang quản lý lớp học 2. Người dùng chọn "Statistic" trong trang quản lý lớp học 3. Hệ thống hiện thông tin thống kê về bài kiểm tra dưới dạng bảng
Alternative flow	Tại bước 3, nếu assignment vẫn chưa đến hạn 3.1. Hệ thống thông báo chưa có thông tin thống kê về bài kiểm tra Tại bước 2 4.2. Người dùng chọn "Chart" 5.2. Hệ thống hiện thông tin về bài kiểm tra dưới dạng biểu đồ
Exception	Không

Bảng 7: Đặc tả cho usecase Statistic

5.2.8 Explore server

Use-case name	**Explore server**
Create by	Lê Minh Châu
Date created	29/03/2024
Actor	User
Description	Người dùng cung cấp API server cho hệ thống để run các bài làm khi người dùng chọn plan "Teacher host"
Trigger	Người dùng chọn "Explore server" khi đang tạo hoặc chỉnh sửa space
Pre-condition	Người dùng chọn "Teacher host", đang tạo hoặc chỉnh sửa assignment space
Post-condition	Người dùng thiết lập API IDE server cho hệ thống thành công
Normal flow	1. Người dùng chọn "Explore" trong mục "Explore server" 2. Hệ thống hiện popup lấy API 3. Người dùng cung cấp API cho hệ thống 4. Người dùng chọn "Confirm" 5. Hệ thống kiểm tra, lưu API và đóng popup
Alternative flow	Tại bước 4, người dùng nhấn "Cancel" 5.1. Hệ thống đóng popup
Exception	Tại bước 5, nếu hệ thống kiểm tra API không thành công 5.1. Thông báo lỗi đến người dùng

Bảng 8: Đặc tả cho usecase Explore server

5.2.9 Generate binary file

Use-case name	**Generate binary file**
Create by	Lê Minh Châu
Date created	29/03/2024
Actor	System
Description	Hệ thống tạo ra file binary từ file test và source code do người dùng cung cấp
Trigger	Người dùng upload file source code và file test
Pre-condition	Người dùng upload thành công file source code và file test, plan được chọn là "Student host"
Post-condition	Hệ thống tạo ra một file binary dùng để run mỗi khi có request từ học viên
Normal flow	1. Người dùng upload source code và file test 2. Hệ thống kiểm tra source code và file test 3. Hệ thống run các file tạo binary
Alternative flow	Tại bước 1, nếu người dùng upload binary 2.1. Hệ thống kiểm tra file binary 3.1. Hệ thống lưu file binary
Exception	Tại bước 3, hệ thống không thể tạo binary file 4.2 Hệ thống báo lỗi đến người dùng Tại 2.1, nếu kiểm tra thất bại (không phải binary file) 3.3. Hệ thống thông báo lỗi đến người dùng

Bảng 9: Đặc tả cho usecase Generate binary file

5.2.10 Request run

Use-case name	**Request run**
Create by	Lê Minh Châu
Date created	31/03/2024
Actor	System
Description	Hệ thống yêu cầu máy server của người dùng run các bài làm của học viên thông qua các API được cung cấp bởi người dùng
Trigger	Học viên commit bài làm của mình lên repository cá nhân trên Github
Pre-condition	Người dùng cung cấp đầy đủ API, học viên commit thuộc assignment space của người dùng đang sử dụng Teacher host plan
Post-condition	Hệ thống nhận được kết quả run code với source code của học viên
Normal flow	1. Học viên commit bài làm lên repository cá nhân 2. Hệ thống nhận bài làm của học viên và gửi qua server của người dùng thông qua API 3. Hệ thống nhận lại kết quả run source code của học viên từ phía server của người dùng
Alternative flow	
Exception	Tại bước hai, nếu học viên commit sau hạn cuối hoặc commit quá số lần quy định của người dùng, hệ thống sẽ không gửi bài làm của học viên

Bảng 10: Đặc tả cho usecase Request run

5.2.11 Grading

Use-case name	**Grading**
Create by	Lê Minh Châu
Date created	31/03/2024
Actor	System
Description	Hệ thống nhận kết quả từ run source code của học viên và kết quả từ source code của người dùng, so sánh và chấm điểm
Trigger	Nhận được kết quả của học viên
Pre-condition	Hệ thống phải có kết quả từ cả source code của người dùng và học viên
Post-condition	Trả về điểm của học viên sau khi so sánh với kết quả của người dùng
Normal flow	1. Sau khi nhận kết quả của học viên, hệ thống sẽ tự kích hoạt Grading 2. Hệ thống lấy kết quả từ source code người dùng và source code của học viên để so sánh và chấm điểm 3. Hệ thống trả về danh sách điểm từ kết quả input
Alternative flow	
Exception	Tại bước 1, nếu kết quả run source code từ người dùng chưa có, hệ thống sẽ tạm thời lưu kết quả của học viên Tại bước 2, nếu học viên không có bài làm, hệ thống sẽ mặc định điểm số của học viên là 0 Tại bước 2, nếu học viên có nhiều bài làm, hệ thống sẽ dựa trên yêu cầu của người dùng (cao nhất, lần cuối nộp bài, scale, ...) để update điểm cho học viên

Bảng 11: Đặc tả cho usecase Grading

5.2.12 Detect cheating

Use-case name	**Detect cheating**
Create by	Lê Minh Châu
Date created	31/03/2024
Actor	System
Description	Hệ thống sau khi có source code của học viên sẽ gửi dụng công cụ phát hiện gian lận để kiểm tra
Trigger	Đến hạn cuối của bài kiểm tra
Pre-condition	Có ít nhất hai bài làm của học viên được commit trong cùng một assignment space
Post-condition	Hệ thống nhận về kết quả chấm gian lận từ công cụ chuyên dụng
Normal flow	1. Khi đến hạn cuối của bài kiểm tra, hệ thống sẽ tự lọc các source code của học viên 2. Hệ thống sử dụng API của công cụ chấm gian lận để đánh giá bài làm 3. Hệ thống nhận về kết quả chấm gian lận
Alternative flow	
Exception	Tại bước 1, nếu học viên có nhiều bài làm, hệ thống sẽ chọn source code dựa trên tiêu chí của người dùng (bài cuối cùng, bài điểm cao nhất, ...)

Bảng 12: Đặc tả cho usecase Detect cheating

5.2.13 Process result

Use-case name	**Process result**
Create by	Lê Minh Châu
Date created	31/03/2024
Actor	System
Description	Hệ thống xử lý kết quả nhận được từ việc chấm điểm và đánh giá gian lận
Trigger	Hệ thống nhận về kết quả điểm số và đánh giá gian lận
Pre-condition	Hệ thống có kết quả điểm số và đánh giá gian lận
Post-condition	Hệ thống tạo ra danh sách điểm số và đánh giá gian lận
Normal flow	1. Hệ thống nhận kết quả điểm số và đánh giá gian lận 2. Hệ thống tạo danh sách các bài làm, điểm số, và khả năng gian lận giảm dần 3. Hệ thống xuất ra file báo cáo
Alternative flow	
Exception	

Bảng 13: Đặc tả cho usecase Process result

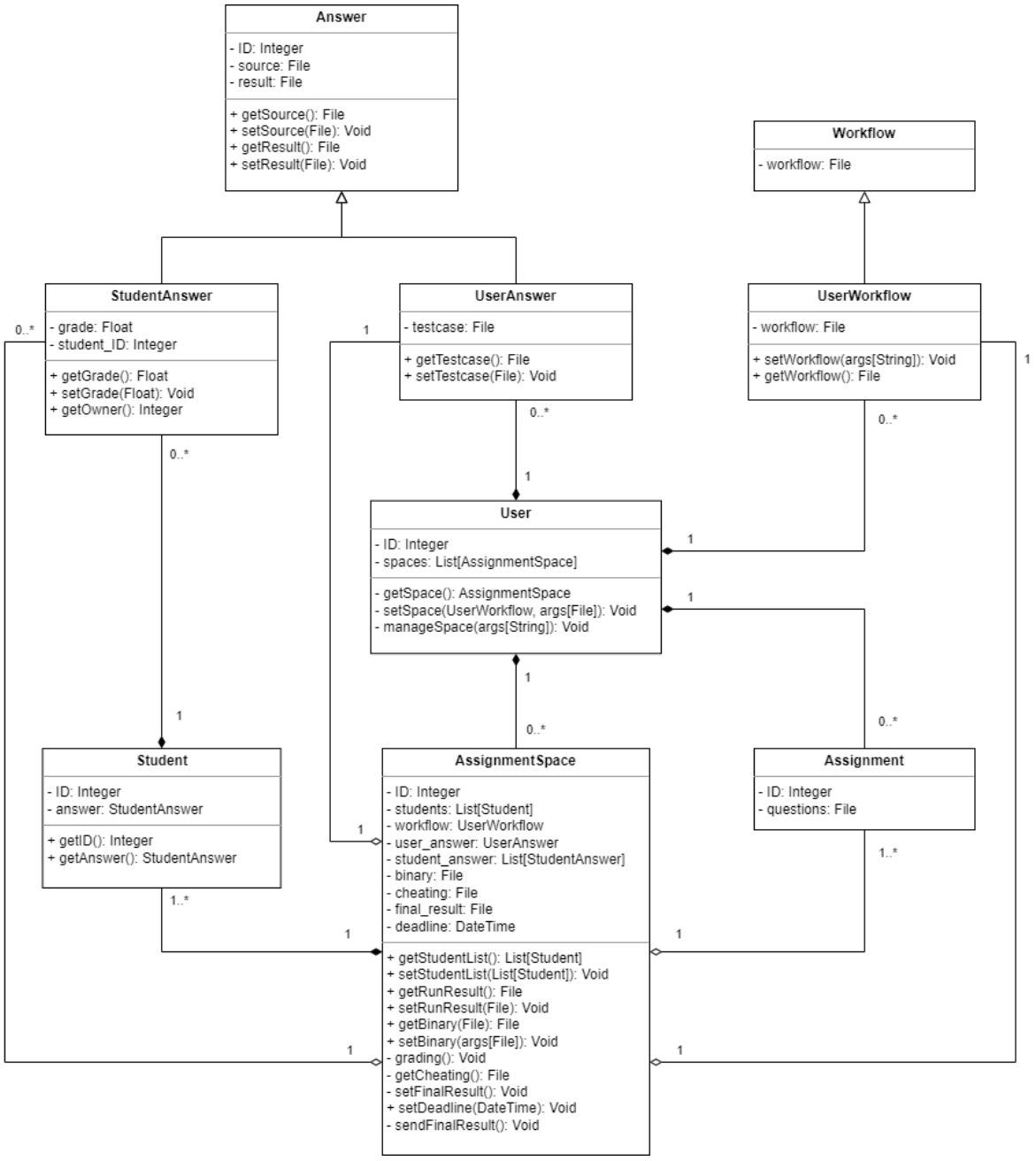
5.2.14 Send result

Use-case name	**Send result**
Create by	Lê Minh Châu
Date created	31/03/2024
Actor	System
Description	Hệ thống gửi kết quả đến người dùng thông qua mail
Trigger	Có kết quả của Process result
Pre-condition	Hệ thống cần có mail của người dùng
Post-condition	Hệ thống gửi kết quả đến người dùng qua địa chỉ mail
Normal flow	1. Hệ thống nhận kết quả của quá trình Process result 2. Hệ thống chuẩn bị mail theo template có sẵn 3. Hệ thống đính kèm các file 4. Hệ thống gửi mail đến người dùng
Alternative flow	
Exception	Nếu người dùng không cung cấp địa chỉ mail nhận, hệ thống sẽ mặc định mail đăng kí tài khoản là mail nhận

Bảng 14: Đặc tả cho usecase Send mail

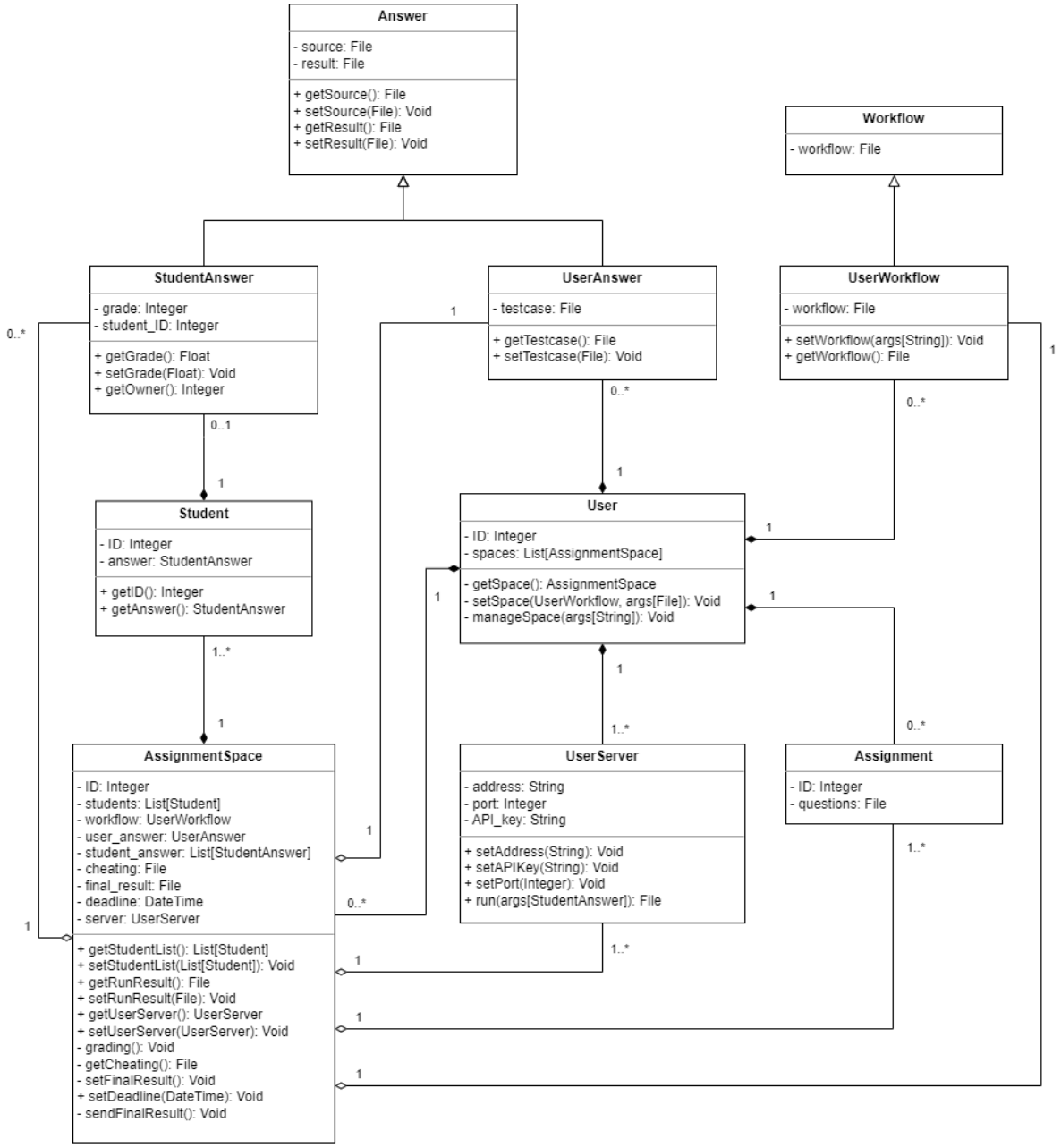
6 Class diagram

6.1 Student host



Hình 5: *Student host class diagram*

6.2 Teacher host



Hình 6: *Teacher host class diagram*

7 Design

Tài liệu

[1] Cheating rate does not decrease

[2] How to use Github Action

[3] Github Action for multiple language

[4] CI/CD and Github Action

[5] Measure of Software Similarity - MOSS

[6] Set up automatic testing with Github Action