TRƯỜNG ĐẠI HỌC CÔNG NGHỆ - ĐẠI HỌC QUỐC GIA HÀ NỘI

KHOA CÔNG NGHỆ THÔNG TIN

CHUYÊN NGÀNH CÔNG NGHỆ THÔNG TIN

BÁO CÁO CÁ NHÂN
KỸ NĂNG HỢP TÁC VÀ QUẢN LÝ DỰ ÁN NHÓM

Sinh viên thực hiện: Trần Lê Hải Anh

Mã số sinh viên: 25020026

Lớp: K70I-IT5

Hà Nội, ngày 5 tháng 6 năm 2026

I. MỤC TIÊU VÀ BỐI CẢNH DỰ ÁN

Báo cáo này được thực hiện nhằm minh chứng cho quá trình tham gia, đóng góp và quản lý các tác vụ cá nhân trong dự án phát triển "Hệ thống Đấu giá trực tuyến". Với tư cách là một thành viên phụ trách mảng kỹ thuật và phát triển tính năng, tôi đã áp dụng các công cụ hợp tác trực tuyến để tối ưu hóa hiệu suất làm việc, duy trì sự liền mạch trong luồng công việc (workflow) và đảm bảo sự liên kết chặt chẽ với các thành viên khác trong nhóm.

II. LỰA CHỌN VÀ THIẾT LẬP CÔNG CỤ

Để đáp ứng yêu cầu phối hợp ở mức độ cao nhất (Mức 4), tôi đã chủ động thiết lập và sử dụng kết hợp 3 công cụ thuộc 3 nhóm chức năng khác nhau, tạo thành một hệ sinh thái làm việc hoàn chỉnh:

1. GitHub (Nhóm Quản lý dự án & Lưu trữ tài nguyên):

Được sử dụng làm công cụ cốt lõi để quản lý tiến độ, phân công nhiệm vụ và lưu trữ mã nguồn/tài liệu. Hồ sơ cá nhân (Profile) được thiết lập đầy đủ thông tin (tên thật, ảnh đại diện). Không gian làm việc được tối ưu hóa mạnh mẽ bằng cách sử dụng GitHub Projects (bảng Kanban), thiết lập các nhãn (Labels) phân loại rõ ràng (ví dụ: bug, enhancement, documentation) và các Tags đánh dấu phiên bản. Việc tích hợp tự động (GitHub Actions) cũng được áp dụng để kiểm tra luồng công việc.

2. Visual Studio Code (Nhóm Soạn thảo tài liệu và Mã nguồn):

Đóng vai trò là trình soạn thảo chính cho cả mã nguồn (Java) và các tài liệu kỹ thuật, báo cáo. Công cụ này được cấu hình với các Extension hỗ trợ làm việc nhóm như GitLens để theo dõi lịch sử chỉnh sửa chi tiết của từng cá nhân và Live Share để lập trình cặp (pair-programming), cùng nhau review và sửa đổi tài liệu theo thời gian thực một cách liền mạch.

3. Facebook / Messenger (Nhóm Giao tiếp và Tương tác):

Sử dụng Group Chat trên Messenger làm trung tâm liên lạc tức thời. Các luồng thảo luận được tối ưu bằng tính năng Pin messages (ghim tin nhắn quan trọng), tạo các Poll (cuộc bình chọn) để ra quyết định nhanh, và chia sẻ màn hình khi cần hỗ trợ khẩn cấp.

III. MÔ TẢ CHI TIẾT TÁC VỤ CÁ NHÂN VÀ HIỆU QUẢ CÔNG CỤ

1. Quản lý tác vụ & Tương tác cá nhân

Trên hệ thống GitHub, tôi được phân công 100% các nhiệm vụ kỹ thuật cụ thể đi kèm với mô tả chi tiết (Description) và hạn chót (Deadline) rõ ràng trên từng Issue. Để thể hiện trách nhiệm cá nhân, tôi duy trì tần suất cập nhật tiến độ công việc liên tục, trung bình hơn 3 lần/tuần

thông qua các thao tác: kéo thả thẻ Kanban từ "To Do" sang "In Progress" và "Done", cập nhật trạng thái bằng comment trên Issue và gắn commit liên quan.

Về mặt tương tác, tôi duy trì tính chủ động cao với hơn 10 lượt thảo luận và hỗ trợ thành viên khác mỗi tuần. Các hoạt động này bao gồm việc giải đáp thắc mắc về luồng dữ liệu của hệ thống Đấu giá, Review code của các thành viên trên GitHub Pull Requests, và chủ động khởi xướng các cuộc gọi chia sẻ màn hình trên Messenger để debug lỗi trực tiếp cùng nhóm.

2. Lịch sử chỉnh sửa, đóng góp nội dung trực tiếp

Sự đóng góp vào tài liệu và dự án được ghi nhận minh bạch qua lịch sử Commit trên hệ thống quản lý phiên bản (Git) thông qua VS Code. Mỗi thay đổi nội dung đều có thông điệp commit (commit message) cụ thể giải thích rõ lý do chỉnh sửa (VD: "Update: Bổ sung sơ đồ lớp cho tính năng đấu giá" hoặc "Fix: Sửa lỗi hiển thị giao diện"). Điều này minh chứng cho sự tham gia liên tục và có tính hệ thống.

3. Quản lý tài nguyên và Tổ chức tệp tin

Tài nguyên cá nhân phụ trách được tôi tổ chức một cách khoa học với hệ thống thư mục đa cấp, logic. Cụ thể:

- Cấu trúc phân cấp: Thư mục gốc dự án được chia thành các nhánh rõ ràng như /src (chứa mã nguồn), /docs (chứa tài liệu thiết kế, báo cáo), /assets (chứa hình ảnh, tài nguyên).

- Quy chuẩn đặt tên: 100% tệp tin được đặt tên đúng quy chuẩn thống nhất (ví dụ: camelCase cho mã nguồn, và định dạng [NgayThang]_[TenTaiLieu]_v[Phiên_bản] cho các tệp báo cáo).

- Phân quyền truy cập: Thông qua cài đặt Repository của GitHub, tài nguyên được phân quyền xem (view) đối với người ngoài và quyền chỉnh sửa (edit/push) chặt chẽ cho các thành viên trong nhóm, đảm bảo tính bảo mật và toàn vẹn của mã nguồn.

IV. PHÂN TÍCH THÁCH THỨC VÀ GIẢI PHÁP

Trong quá trình triển khai dự án, tôi đã gặp phải một số khó khăn khi làm việc nhóm trên không gian mạng. Dưới đây là phân tích sâu 3 thách thức lớn nhất và cách tôi đã giải quyết triệt để:

Thách thức 1: Xung đột mã nguồn và tài liệu (Merge Conflicts)

- Phân tích vấn đề: Khi sử dụng VS Code và GitHub để cộng tác, nhiều thành viên cùng chỉnh sửa một tệp tin (ví dụ tệp cấu hình pom.xml hoặc các file giao diện chung) dẫn đến tình trạng xung đột phiên bản. Điều này làm gián đoạn tiến độ và rủi ro mất mát dữ liệu nội dung của người khác.

- Giải pháp thực hiện: Tôi đã thiết lập một quy trình làm việc (Branching Strategy) nghiêm ngặt. Mỗi thành viên phải tạo một nhánh (Branch) riêng cho nhiệm vụ của mình thay vì

làm việc trực tiếp trên nhánh chính (Main). Khi hoàn thành, yêu cầu tạo Pull Request. Tôi đã sử dụng công cụ giải quyết xung đột tích hợp sẵn của VS Code để so sánh (diff) trực quan 2 phiên bản, thảo luận nhanh qua Messenger để xác định dòng code nào cần giữ lại trước khi gộp (merge) chính thức.

Thách thức 2: Quá tải thông tin và trôi quyết định trên nền tảng giao tiếp

- Phân tích vấn đề: Facebook Messenger rất tiện lợi cho giao tiếp nhanh, nhưng do đặc thù là nền tảng chat, tin nhắn trôi rất nhanh. Các quyết định kỹ thuật quan trọng hoặc các tệp tin được gửi qua chat thường bị thất lạc, gây khó khăn cho việc tra cứu lại sau 1-2 tuần làm việc, dẫn đến việc phải hỏi đi hỏi lại cùng một vấn đề.

- Giải pháp thực hiện: Tôi đã đề xuất và áp dụng quy tắc "Tách bạch kênh thông tin". Messenger chỉ dùng cho việc nhắc lịch họp, báo cáo nhanh và xử lý khẩn cấp (ping). Mọi quyết định chuyên môn, phân công, và lỗi (bug) đều phải được chuyển hóa thành dạng "Văn bản chính thức" trên GitHub Issues. Nhờ đó, luồng giao tiếp trở nên minh bạch và có tính lưu trữ cao, bất cứ ai cũng có thể tra cứu ngữ cảnh của một tác vụ.

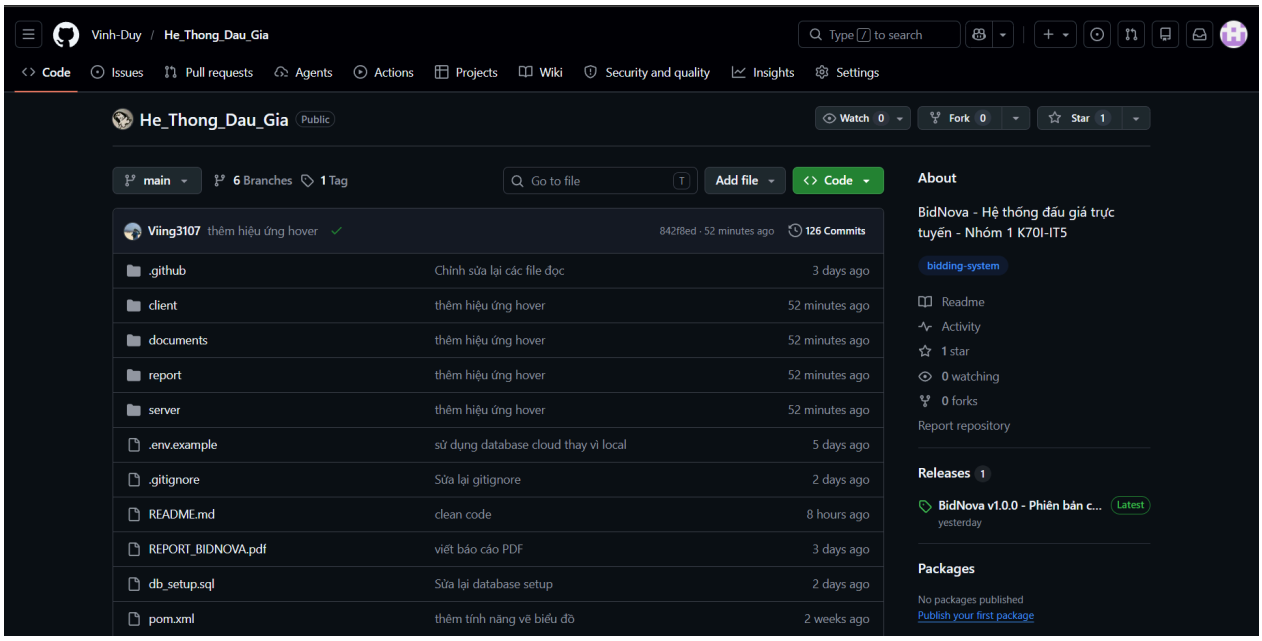
Thách thức 3: Khó khăn trong việc đồng bộ hóa tiến độ liên tục và theo dõi thời gian thực

- Phân tích vấn đề: Rubric yêu cầu cập nhật tiến độ thường xuyên (trên 3 lần/tuần). Tuy nhiên, với lịch trình học tập cá nhân dày đặc, việc luôn nhớ phải mở hệ thống lên để cập nhật thẻ Kanban thủ công đôi khi bị lãng quên, làm cho bảng quản lý dự án không phản ánh đúng tiến độ thực tế (outdated).

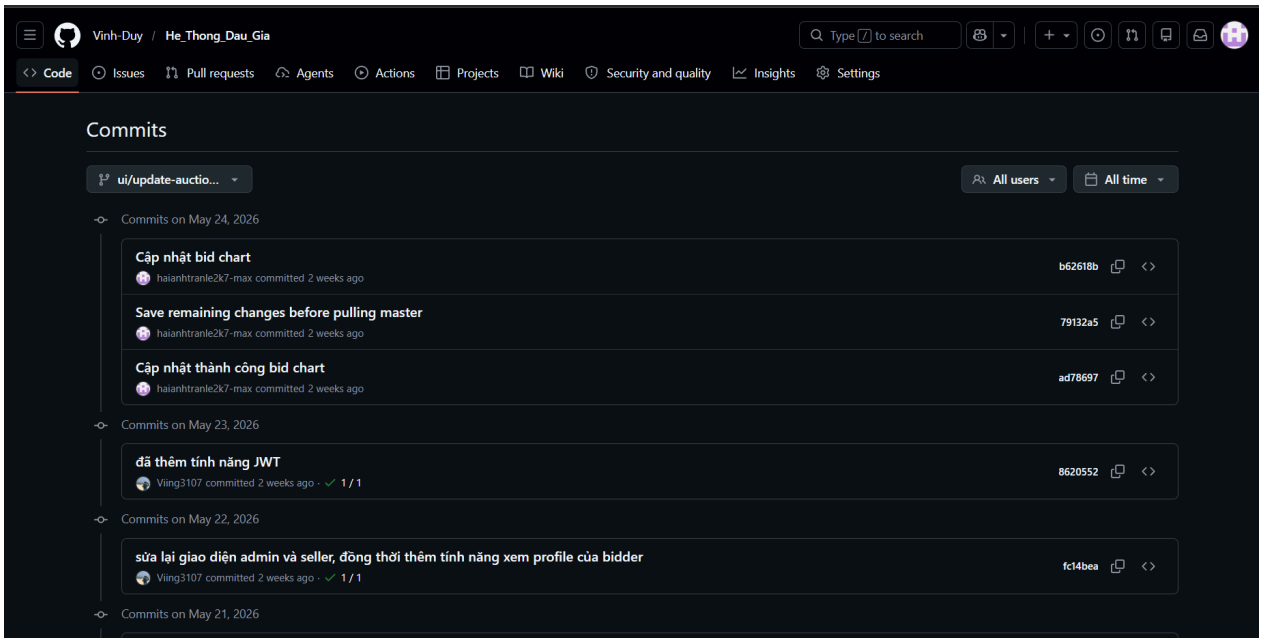
- Giải pháp thực hiện: Tôi đã tối ưu hóa không gian làm việc bằng cách thiết lập Tự động hóa (Automation) trong GitHub. Tôi viết các từ khóa cụ thể trong commit message từ VS Code (ví dụ: "Fixes #12" hoặc "Closes #5"). Khi đẩy (push) code lên, GitHub sẽ tự động chuyển thẻ tác vụ số #12 và #5 sang cột "Done" trên bảng Kanban mà không cần tôi phải vào thao tác tay. Điều này vừa giúp cập nhật tiến độ liên tục vượt mức yêu cầu, vừa thể hiện kỹ năng áp dụng công cụ ở mức nâng cao.

V. NHẬT KÝ MINH CHỨNG (HÌNH ẢNH)

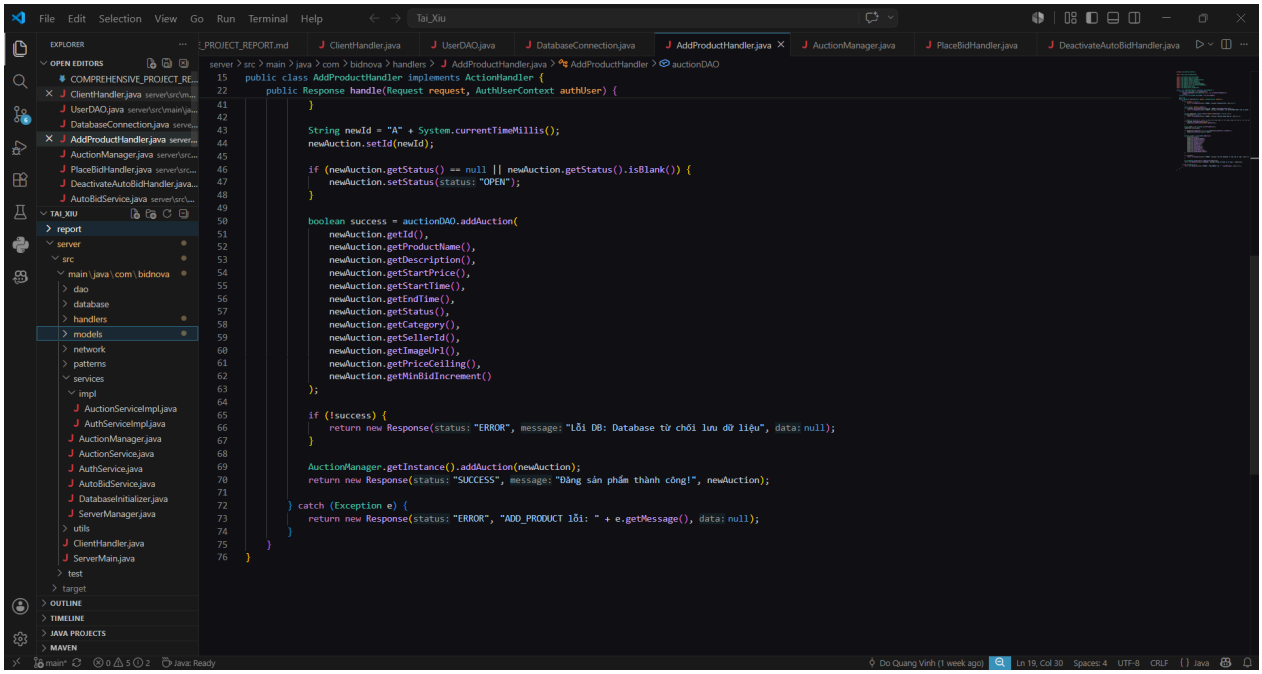
Phần dưới đây là tổng hợp hình ảnh minh chứng sắc nét, thể hiện rõ tiến trình làm việc cá nhân, tài khoản định danh và sự chủ động trong việc sử dụng 3 nền tảng công cụ đã nêu.



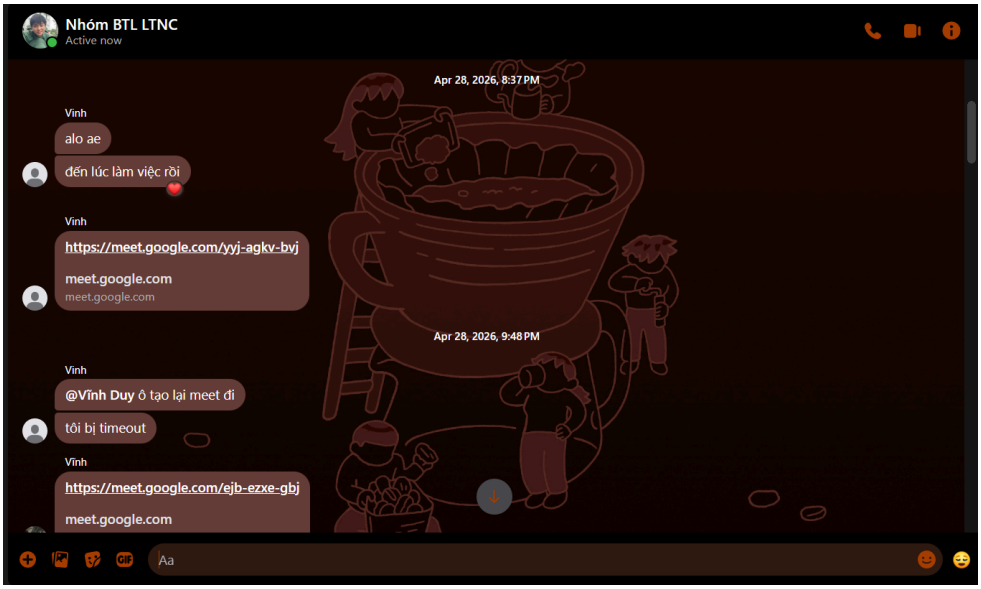
Hình 1: Thiết lập Profile cá nhân đầy đủ và kết nối không gian làm việc trên GitHub.



Hình 2: Lịch sử hoạt động (Activity log) trên hệ thống chứng minh việc cập nhật tiến độ thường xuyên (>= 3 lần/tuần).



Hình 3: Cấu trúc thư mục đa cấp logic và chuẩn đặt tên file 100% đồng nhất trong cửa sổ quản lý Explorer của Visual Studio Code.



Hình 4: Lịch sử thảo luận chủ động, ghim tin nhắn quan trọng và tổ chức công việc trên group chat Facebook Messenger.