# BÀI THỰC HÀNH SỐ 3-1 – GROUP PROJECT 3-1

# SOFTWARE CONFIGURATION MANAGEMENT

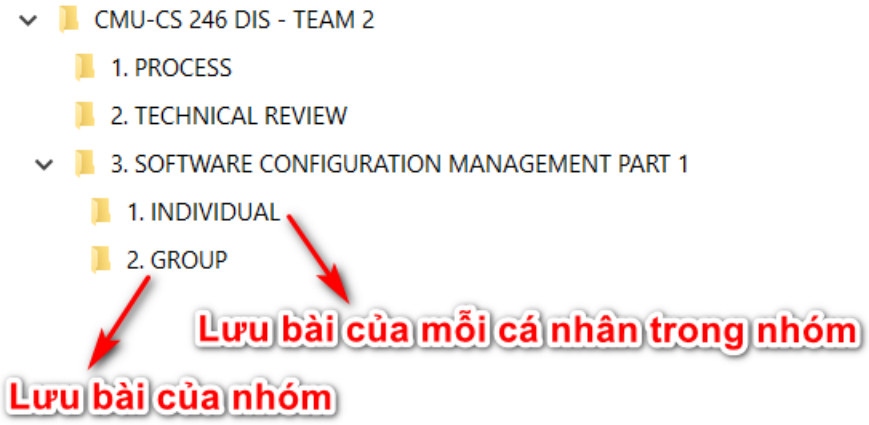
***Định nghĩa:***

Quản lý cấu hình phần mềm được định nghĩa là một quy trình để quản lý, tổ chức và kiểm soát một cách có hệ thống các thay đổi trong tài liệu, mã và các thực thể khác trong Vòng đời phát triển phần mềm. Nó được viết tắt là quy trình SCM trong ngành công nghệ phần mềm

***Yêu cầu minh chứng khi thực hành***

Để giám sát được kết quả làm việc của sinh viên, yêu cầu sinh viên khi thực hiện bài thực hành phải thực hiện một số thao tác sau:

- Tạo thư mục để lưu bài làm như hình bên dưới (Lưu trên drive)
- Trong quá trình thực hiện bài tập, sinh viên phải chụp màn hình (chụp toàn màn hình – không cắt bớt) từng bước làm và lưu vào file word có ghi chú bên dưới để giảng viên có thể kiểm tra (xem mẫu ở cuối file).



# PHẦN I. BÀI CÁ NHÂN

## 1. Mô tả

Để quản lý việc phát triển một dự án phần mềm, công ty ABC tổ chức một hệ thống SCM (Software Configuration Management) với phần mềm **SVN** dùng để lưu trữ các phiên bản (revision) của tất cả các tệp tài liệu, mã nguồn trong dự án. Mô hình SCM của công ty được thiết kế như sau:

- Mỗi lập trình viên trong nhóm phát triển đều cài đặt **VisualSVN Server** trên máy tính cá nhân, tạo các **repository riêng biệt** cho dự án mà mình phụ trách.
- Lập trình viên có thể thao tác với mã nguồn bằng công cụ lập trình như **NetBeans** tích hợp plugin **SVN**, hoặc sử dụng công cụ hỗ trợ như **TortoiseSVN** để thực hiện các thao tác với các tệp tài liệu văn bản.

## 2. Yêu Cầu Thực Hiện

Sinh viên sẽ sử dụng lại **dự án Calculator** đã thực hiện ở Group Project 1, thực hiện quản lý dự án theo mô hình SCM sử dụng công cụ **NetBeans**, **VisualSVN Server** và **TortoiseSVN**, với các yêu cầu:

- Cài đặt hệ thống máy chủ quản lý mã nguồn (VisualSVN Server).
- Tạo repository, tài khoản người dùng, phân quyền truy cập.
- Kết nối NetBeans với SVN để thực hiện các thao tác quản lý mã nguồn (checkout, commit...).
- Sử dụng TortoiseSVN để quản lý các file tài liệu liên quan.
- Thao tác với các phiên bản code như checkout revision cũ hơn, merge...

## 3. Hướng Dẫn Cụ Thể

*Bước 1: Cài đặt VisualSVN Server*

1. Truy cập trang: https://www.visualsvn.com/server/download/
2. Tải bản **VisualSVN Server Standard Edition** (miễn phí).
3. Cài đặt VisualSVN Server với các tùy chọn mặc định.

4. Mở ứng dụng VisualSVN Server Manager.

*Bước 2: Tạo Repository trên VisualSVN Server*

1. Chuột phải vào "Repositories" → chọn **Create New Repository**.
2. Đặt tên repository là **Calculator**.
3. Chọn cấu hình với chuẩn cấu trúc thư mục: **trunk/branches/tags**.
4. Tạo xong sẽ có URL dạng: https://localhost/svn/Calculator.

*Bước 3: Tạo User, Group và Phân quyền*

1. Trong VisualSVN Server, chuột phải vào **Users** → chọn **Create New User**.
2. Tạo user student01 với mật khẩu riêng.
3. Tương tự tạo thêm **Groups** nếu cần quản lý theo nhóm.
4. Chuột phải vào repository → **Properties** → tab **Security** → cấp quyền **Read/Write** cho user hoặc group tương ứng.

*Bước 4: Cấu hình SVN trên NetBeans*

1. Mở **NetBeans IDE**.
2. Vào menu **Team → Subversion → Checkout**.
3. Nhập URL SVN repo: https://localhost/svn/Calculator, điền username/password vừa tạo.
4. Chọn thư mục **trunk/** để checkout về máy.
5. Sau khi tải về, NetBeans sẽ hỏi có muốn mở project không → chọn Yes.

*Bước 5: Commit (Check-in) Dự Án Calculator*

1. Thực hiện các chỉnh sửa trong mã nguồn nếu cần.
2. Chuột phải vào project → chọn **Subversion → Commit**.
3. Nhập nội dung log → chọn các file → nhấn **Commit**.

*Bước 6: Sử dụng TortoiseSVN*

*I. Cài đặt TortoiseSVN*

1. Tải về TortoiseSVN từ: https://tortoisesvn.net/downloads.html

2. Chạy file cài đặt và làm theo hướng dẫn.
3. Sau khi cài đặt xong, khởi động lại máy tính để hoàn tất tích hợp với Windows Explorer.

*II. Sử dụng TortoiseSVN để thao tác với tài liệu*

1. Tạo một folder mới trên máy, chuột phải vào → **SVN Checkout**.
2. Dán URL SVN như ở bước trên → nhấn OK.
3. Đặt vào thư mục một file tài liệu bất kỳ (PDF, DOCX,...).
4. Chuột phải vào file → **SVN Commit** để đẩy lên server.
5. Thực hiện sửa file → **SVN Update** để kiểm tra cập nhật.
6. Có thể dùng **Show Log** để xem lịch sử phiên bản.

*Bước 7: Checkout Revision Cũ hơn*

1. Chuột phải vào file/project → chọn **TortoiseSVN → Show Log**.
2. Chọn một phiên bản cũ → chuột phải → **Revert to this revision** hoặc **Update to revision**.

*Bước 8: Thực hiện Merge*

1. Tạo một nhánh trong repository (branch).
2. Commit một thay đổi vào branch.
3. Dùng tính năng **TortoiseSVN → Merge** để hợp nhất lại vào **trunk**.

**Lưu ý:** Sinh viên cần ghi lại toàn bộ quá trình làm bài vào một file báo cáo (PDF), trong đó đính kèm hình ảnh các bước đã thực hiện để nộp cùng source code dự án, upload lên thư mục 1. INDIVIDUAL (*tên file là Họ và Tên của mình)*

### ☙ Dùng TortoiseSVN khi:

- Làm việc nhóm, cần thao tác toàn cục với repository.
- Quản lý dự án ngoài NetBeans (hoặc nhiều IDE khác nhau).
- Muốn thao tác nhanh trên hệ điều hành mà không cần mở IDE.
- Cần xử lý các xung đột, merge, revert mạnh mẽ.

→ **TortoiseSVN** là công cụ **toàn diện, mạnh mẽ, độc lập IDE**

### ☙ Dùng SVN trong NetBeans khi:

- Đang làm việc cá nhân với project Java trong NetBeans.
- Muốn thao tác nhanh commit/update ngay trong IDE.
- Không cần quá nhiều thao tác nâng cao với repository.

→ **SVN trong NetBeans** thì **nhanh, tiện lợi cho cá nhân khi đang lập trình**.

# PHẦN II. BÀI TẬP NHÓM
# QUẢN LÝ DỰ ÁN PHẦN MỀM TRÊN GITHUB/GITLAB

Công ty ABC tổ chức một hệ thống **Software Configuration Management** (SCM) sử dụng GitHub/GitLab để lưu trữ tất cả các phiên bản (revision) của các file mềm dự án **Calculator**. Mô hình SCM của công ty được thiết kế như sau:

***Mục Tiêu:***

Nhóm của bạn sẽ phát triển thêm các tính năng cho ứng dụng **Calculator** đã thực hiện trong **Group Project 1**. Bạn sẽ sử dụng **GitHub/GitLab** để quản lý mã nguồn, tài liệu, test case và làm việc nhóm. Mỗi thành viên trong nhóm sẽ tham gia vào các công việc được phân công sau:

***Các Công Việc Cần Thực Hiện:***

**1. Cập Nhật và Mở Rộng Chức Năng Ứng Dụng Calculator:**

Ứng dụng **Calculator** hiện tại chỉ hỗ trợ các phép toán cơ bản và một số phép toán nâng cao cơ bản. Nhóm của bạn cần bổ sung thêm các tính năng sau:

***Các phép toán khoa học:***

- Giai thừa (n!).
- Logarit ($\log_{10} x$, $\ln x$).
- Các hàm lượng giác: sin(x), cos(x), tan(x), cot(x).
- Chuyển đổi đơn vị góc: Độ ↔ Radian.

***Lịch sử tính toán:***

- Lưu trữ lịch sử phép tính vào file **JSON** hoặc **text** (thay vì chỉ lưu trong bộ nhớ).
- Cho phép người dùng tìm kiếm lại phép tính cũ trong lịch sử.
- Cho phép người dùng xóa từng dòng lịch sử.

***Giao diện người dùng:***

- Thêm chế độ **Dark mode** và **Light mode**.

- Cho phép người dùng chọn phông chữ và màu sắc giao diện.
- Hỗ trợ **keyboard shortcuts** (sử dụng bàn phím để nhập phép tính).

***Các chức năng điều khiển:***

- Thêm nút **CE (Clear Entry)** để xóa một số duy nhất mà không làm thay đổi phép tính.
- Thêm chức năng ← **(Backspace)** và → **(Forward)** để xóa một ký tự ở hai chiều con trỏ.

***Các tính năng bổ sung khác:***

- **Copy/Paste** kết quả vào clipboard.
- Hỗ trợ tính toán theo biểu thức phức tạp như: "5 + (3 * 2) - √9".

***Xử lý lỗi nâng cao:***

- Kiểm thử các tình huống lỗi đầu vào như phép tính vượt quá giới hạn, hay phép toán không hợp lệ (ví dụ: chia cho 0, căn bậc hai của số âm).
- Thêm thông báo lỗi chi tiết khi nhập sai dữ liệu (ví dụ: "5++2", "√-9").

**2. Quản Lý Dự Án trên GitHub/GitLab:**

Để quản lý mã nguồn và các tài liệu liên quan, nhóm của bạn sẽ sử dụng **GitHub/GitLab** theo các bước sau:

***2.1. Tạo Repository trên GitHub/GitLab:***

- Nhóm trưởng tạo một repository cho dự án **Calculator** đã mở rộng trên **GitHub/GitLab**.
- Mời các thành viên trong nhóm tham gia vào dự án và thêm giảng viên (huyndq@duytan.edu.vn) để kiểm tra.
- Đảm bảo mỗi thành viên đều có quyền **push** và **pull** mã nguồn từ repository.

***2.2. Đồng Bộ Dự Án Calculator Lên GitHub/GitLab:***

- **Upload toàn bộ thư mục Project** lên **GitHub/GitLab**.
- Tất cả các tài liệu liên quan (thiết kế, test case, code) đều được đưa vào repository này.

### *2.3. Bổ sung chức năng vào Calculator:*

- Mỗi thành viên thực hiện một phần công việc như bổ sung phép toán, phát triển giao diện, cập nhật lịch sử tính toán, v.v.
- Sau khi hoàn thành công việc, các thành viên thực hiện **commit** và **push** các thay đổi lên **GitHub/GitLab**.

### *2.4. Merge các thay đổi từ các thành viên:*

- Nhóm trưởng thực hiện **merge** mã nguồn và tài liệu từ các thành viên.
- Đảm bảo rằng tất cả các tính năng mới được tích hợp đúng vào dự án chính.

### *2.5. Quản Lý Test Case và Kiểm Thử:*

- Cập nhật lại các tài liệu thiết kế và test case đã làm ở **Group Project 1**. Đặt tên tài liệu là **Tên tài liệu_Ver2.0**.
- Phân công các thành viên thực hiện kiểm thử các chức năng mới (các phép toán bổ sung, tính năng giao diện, lịch sử tính toán).
- Sau khi kiểm thử xong, **commit** và **push** kết quả kiểm thử lên **GitHub/GitLab**.

## 3. Công Việc Của Các Thành Viên:

Các thành viên trong nhóm (ngoại trừ nhóm trưởng) thực hiện các công việc sau:

### *3.1. Kết nối đến repository của nhóm:*

- **Clone repository** mà nhóm trưởng đã tạo trên **GitHub/GitLab**.

### *3.2. Thực hiện cập nhật tài liệu:*

- Tải xuống các tài liệu mà nhóm trưởng phân công, **cập nhật** và **check-in** lên repository.

### *3.3. Code các chức năng bổ sung:*

- Thực hiện code các tính năng bổ sung theo yêu cầu.
- **Push** các thay đổi mã nguồn lên repository sau khi hoàn thành.

### *3.4. Kiểm thử và commit kết quả:*

- Tiến hành kiểm thử các tính năng bổ sung (dựa trên **Test Case**) và ghi kết quả vào tài liệu **Test Case**.
- **Check-in** tài liệu **Test Case** lên repository.

### *3.5. Sử dụng Git:*

- Sử dụng **Git** để **checkout**, **update**, **commit**, **push** mã nguồn.
- Sử dụng **TortoiseSVN** để **checkout**, **update**, **commit** tài liệu liên quan.

## 4. Yêu Cầu Bài Tập Nhóm:

- **Minh Chứng Quy Trình SCM:** Mỗi thành viên cần thực hiện các thao tác **SCM** theo đúng quy trình **Checkout**, **Commit**, **Push**, và **Merge** trên **GitHub/GitLab**.
    - **Minh chứng các thao tác SCM**: Cung cấp các log hoặc screenshot cho giảng viên để chứng minh rằng các thao tác **commit**, **push**, **merge** được thực hiện đúng.
    - **Tránh conflict**: Đảm bảo rằng các bạn **check-in code** và tài liệu lên **GitHub/GitLab** thường xuyên và tránh tình huống **conflict** khi làm việc đồng thời.
- **Test Case phải được cập nhật đầy đủ** cho tất cả các chức năng bổ sung và kiểm thử phải được thực hiện theo tài liệu.
- **Minh chứng quy trình hợp tác nhóm**: Các thành viên cần **minh chứng** việc tham gia vào các công việc nhóm, bao gồm các hoạt động như **clone repository**, **commit**, **push**, **merge**, và các hoạt động kiểm thử, quản lý tài liệu.
- **Video Demo (tùy chọn):** Nếu có, nộp video demo về ứng dụng và giải thích các tính năng mới.

**Lưu Ý:**

- **Cẩn thận với việc đồng bộ mã nguồn:** Đảm bảo rằng tất cả các thành viên đã **commit** và **push** các thay đổi lên repository đúng hạn.
- **Quy trình Merge hợp lý:** Tránh xung đột mã nguồn và đảm bảo rằng tất cả các thay đổi đã được **merge** một cách chính xác và đầy đủ.
- **Đảm bảo chất lượng mã nguồn:** Kiểm tra kỹ các tính năng mới để đảm bảo rằng ứng dụng hoạt động đúng như yêu cầu.
- Nhóm làm báo cáo theo mẫu dưới đây, và nộp lên thư mục 2.Group

# MẪU BÁO CÁO NHÓM

## Chú ý làm bằng Tiếng Anh để nộp

1. **Giới thiệu dự án:**
    - Tóm tắt mục tiêu của dự án **Calculator** và các tính năng mới được bổ sung.
    - Mô tả về các công nghệ sử dụng (Java, GitHub/GitLab, v.v.).
2. **Quy trình làm việc nhóm:**
    - Mô tả cách thức nhóm đã tổ chức công việc.
    - Phân công công việc giữa các thành viên trong nhóm.
    - Các công việc đã hoàn thành và cách nhóm phối hợp làm việc qua **GitHub/GitLab**.
3. **Các thao tác SCM:**
    - Mô tả chi tiết các thao tác SCM được thực hiện (checkout, commit, push, merge) với **GitHub/GitLab**.
    - Minh chứng quá trình **commit**, **push** của từng thành viên qua log hoặc screenshot.
    - Giải thích cách nhóm đã tránh được **conflict** trong khi làm việc đồng thời và cách xử lý khi gặp xung đột mã nguồn.
4. **Quản lý tài liệu:**
    - Mô tả các tài liệu liên quan (thiết kế, test case) đã được quản lý trên GitHub/GitLab.
    - Cung cấp ví dụ về cách các tài liệu được cập nhật và lưu trữ trong repository.

5. **Kiểm thử và kết quả:**
    - Báo cáo về các bài kiểm thử cho các tính năng bổ sung, bao gồm các test case đã thực hiện.
    - Minh chứng kết quả kiểm thử thông qua tài liệu **Test Case** hoặc báo cáo kiểm thử.
6. **Các vấn đề gặp phải và giải pháp:**
    - Liệt kê các vấn đề kỹ thuật, xung đột mã nguồn, hoặc các khó khăn khác trong quá trình làm việc nhóm.
    - Giải pháp đã áp dụng để giải quyết các vấn đề đó.
7. **Video Demo (tùy chọn):**
    - Nếu có, nhóm có thể thêm video demo về ứng dụng Calculator với các tính năng mới được phát triển.
    - Giải thích ngắn gọn về các tính năng mới và cách sử dụng ứng dụng.

**Nộp báo cáo:**

- **Tệp báo cáo** phải được nộp lên hệ thống quản lý học tập (LMS) hoặc gửi qua email theo yêu cầu của giảng viên.
- Báo cáo cần được **định dạng chuyên nghiệp**, rõ ràng và đầy đủ thông tin.