|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC CNTT&TT  KHOA CÔNG NGHỆ THÔNG TIN  **BỘ MÔN: CÔNG NGHỆ PHẦN MỀM**  **--------------------------** | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM  **Độc lập – Tự do – Hạnh phúc**  **-------------------------** |

**DANH SÁCH CHỦ ĐỀ TIỂU LUẬN**

Tên học phần: Vận hành và Bảo trì phần mềm

Ngành học: Kỹ thuật phần mềm

Học kì: I Năm học 2024-2025

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên tiểu luận** | **Số SV thực hiện** | **Tên nhiệm vụ** |
| 1 | Continuous Integration and Delivery – Tích hợp và phát hành liên tục | 3 | * Lý thuyết * Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề. * Giới thiệu tổng quan về DevOps * Tầm quan trọng của Tích hợp liên tục (CI- Continuous Intergration) và Phát hành liên tục (CD - Continuous Deployment) trong vòng đời dự án DevOps. Tính liên tục thể hiện như thế nào trong tiến trình tích hợp? * Trình bày các giai đoạn trong quy trình tích hợp liên tục (CI), phát hành liên tục (CD) và tiến trình DevOps (full). Giới thiệu một số công cụ hỗ trợ quy trình tích hợp liên tục (CI), (CD) và (DevOps) * Trình bày chi tiết về công cụ hỗ trợ tích hợp liên tục CircleCI. So sánh với Jenkins. * Mô tả chi tiết quy trình CI sử dụng CircleCI. * Demo: * Demo quy trình tích hợp liên tục sử dụng CircleCI áp dụng cho dự án Jenkins kết nối tới kho mã nguồn Git&GitHub * Phân tích kết quả demo, làm rõ tính liên tục thể hiện như thế nào. |
| 2 | Code Refactoring - Tái cấu trúc mã nguồn | 3 | Lý thuyết   * Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề. * Trình bày tổng quan về bảo trì phần mềm, tầm quan trọng của bảo trì và đảm bảo chất lượng mã nguồn trong quá trình bảo trì * Giới thiệu một số công cụ trợ giúp hiểu mã nguồn, đảm bảo chất lượng mã nguồn. Ví dụ: code browser, reverse engineering, refactoring code, review code, white box code test, ... * Trình bày chi tiết về refactoring code trong cải tiến chất lượng mã nguồn (mục đích, lợi thế, tính phổ dụng, các ngữ cảnh refactoring, đánh giá chất lượng code sau refactoring...) * Giới thiệu về IDE hỗ trợ các chức năng refactoring code. Trình bày các bước refactoring sử dụng các chức năng refactoring tương ứng.   Demo: Demo quy trình refactoring để cải tiến cấu trúc chương trình cụ thể qua mã nguồn:   * Cấu trúc chương trình của ứng dụng trước khi refactoring và các vấn đề đang tồn tại. * Cấu trúc chương trình sau refactoring và các vấn đề được giải quyết bằng refactoring. * Chỉ rõ các hoạt động refactoring trên cấu trúc chương trình để cải tiến chất lượng mã nguồn, cải tiến cấu trúc chương trình * Chạy các chức năng refactoring tương ứng trên IDE. * Phân tích kết quả cải thiện mã nguồn sau refactoring, ưu điểm và lợi thế của refactory trong sửa đổi bảo trì mã nguồn. |
| 3 | Package and Deploy the software - Đóng gói, phát hành phần mềm | 5 | * Lý thuyết: * Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề. * Trình bày tổng quan về đóng gói và phát hành sản phẩm. * Mô ta sơ lược các loại ứng dụng (ứng dụng mobile, ứng dụng web, ứng dụng desktop, ứng dụng là service, ứng dụng là tool hoặc plugin, ...) và các hình thức đóng gói và phát hành tương ứng. * Mô tả các công cụ được sử dụng để đóng gói, phát hành ứng dụng. * Demo: * Demo quy trình đóng gói và phát hành phần mềm sử dụng các định dạng gói khác nhau, áp dụng cho các kiểu phần mềm khác nhau * Nhận xét và đánh giá về ưu và nhược điểm của quy trình đóng gói và phát hành ứng dụng thủ công. |
| 4 | Xét duyệt mã nguồn - Source code review | 5 | * Lý thuyết:   + Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề.   + Tầm quan trọng của review code trong đảm bảo chất lượng sản phẩm   + Các tiêu chí, tiêu chuẩn cần review   + Phân biệt giữa review thủ công, review tự động. Ưu và nhược điểm của từng cách tiếp cận review   + Giới thiệu một số công cụ review tự động (Codacy, Code climate, Pull Assistant, Codebeat), trợ giúp review thủ công (GitHub, Gerrit, ...).   + Trình bày chi tiết về công cụ được lựa chọn để demo. Mô tả quy trình review code sử dụng công cụ đã lựa chọn. * Demo:   + - Demo quy trình review code tự động kết hợp review thủ công sử dụng các công cụ đã lựa chọn     - Phân tích kết quả review     - Sửa code khắc phục lỗi phát hiện khi review     - Lặp lại quy trình demo. |
| 5 | Test automation – Kiểm thử tự động | 4 | * Lý thuyết:   + Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề.   + Tầm quan trọng của hoạt động test trước khi đóng gói và phát hành ứng dụng đến môi trường người dùng.   + Sự khác biệt giữa test thủ công và test tự động.   + Trình bày chi tiết về test hệ thống và test chấp thuận. Các phương pháp sinh test cases tương ứng.   + Trình bày chi tiết về QTP. Quy trình test sử dụng QTP * Demo: * Demo quy trình test hệ thống, test chấp thuận sử dụng QTP. |
| 6 | Source code management- Git&GitHub – Quản lý mã nguồn sử dụng Git&GitHub | 4 | * Lý thuyết:  1. Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề. 2. Trình bày tổng quan về quản lý mã nguồn, tầm quan trọng của quản lý các phiên bản mã nguồn trong giai đoạn bảo trì phần mềm 3. Giới thiệu một số công cụ quản lý mã nguồn (Git, GitHub, Gerrit, Gitlab, ...). Lý do chọn Git&GitHub. 4. Trình bày chi tiết về kỹ thuật phân nhánh được sử dụng trong quản lý mã nguồn khi làm việc với Git&GitHub. 5. Trình bày sơ lược về tính năng quản lý dự án được hỗ trợ trong GitHub 6. Danh sách các lệnh Git được sử dụng trong quá trình demo.  * Demo:  1. Lập kế hoạch bảo trì phần mềm (dự kiến các phiên bản bảo trì; mô tả sơ lược các chức năng, các sửa đổi tương ứng của mỗi phiên bản; kế phân công công việc (ai, làm gì, thời gian làm, làm trên nhánh nào). 2. Giới thiệu mô hình phân nhánh được sử dụng để demo. 3. Thông kê, theo dõi quy trình bảo trì qua lịch sử commit |
| 7 | Jenkins – tích hợp và phát hành ứng dụng liên tục | 3 | * Lý thuyết:  1. Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề. 2. Tổng quan về DevOps. Quy trình DevOps (core and full) 3. Trình bày chi tiết về Jenkins hỗ trợ trong quản lý cấu hình dự án DevOps, hỗ trợ tích hợp liên tục dự án DevOps 4. Cài đặt và làm việc với Jenkin, thiết lập các tham số cấu hình toàn cụ và cục bộ cho mỗi kiểu dự án của Jenkins  * Demo:  1. Tạo một số kiểu dự án DevOps khác nhau trên Jenkins. 2. Thiết các tham số cấu hình cho từng dự án (tham số cấu hình gồm: các giai đoạn tương ứng của dự án, công cụ sử dụng là gì, các lệnh cần thực thi, ...) 3. Chạy quy trình: test, đóng gói, phát hành và thực thi dự án trên Jenkins theo các tham số cấu hình đã thiết lập. |
| 8 | Điện toán đám mây- Phát hành, vận hành, giám sát vận hành và lập kế hoạch cải tiến phần mềm. | 4 | * Lý thuyết:  1. Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề. 2. Tổng quan về tiến trình DevOps đầy đủ. Giới thiệu sơ lược các công cụ hỗ trợ cho từng giai đoạn của dự án DevOps 3. Tầm quan trọng của hoạt động sau đây trong tiến trình DevOps đầy đủ (full)    1. Phát hành (Deployment)    2. Vận hành (Operation)    3. Giám sát (Monitor) quá trình vận hành trên đám mây,    4. Lập kế hoạch cải tiến phần mềm (Plan) 4. Giới thiệu một số đám mây, đặc biệt tập trung vào đám mây Amazon. Các dịch vụ trên đám mây như saas, paas, iaas, ... 5. Trình bày chi tiết quy trình a), b), c), d) lên đám mây Amazon. 6. Mô tả sơ lược các tham số cần giám sát trong quá trình vận hành ứng dụng, lập kế hoạch cải tiến ứng dụng dựa trên các tham số này.  * Demo:  1. Demo quy trình phát hành ứng dụng lên App Store của Amazon, vận hành ứng dụng, giám sát quá trình vận hành ứng dụng trên đám mây và lập kế hoạch cải tiến chất lượng ứng dụng 2. Phân tích kết quả demo, cải tiến chất lượng mã nguồn dựa trên kế hoạch cải tiến và quay vòng lặp lại quy trình demo. |
| 9 | Build management - Maven | 5 | * Lý thuyết:  1. Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề 2. Trình bày tổng quan về xây dựng và đóng gói ứng dụng (building and packing), tổng quan về quản lý build, tại sao phải quản lý build. 3. Trình bày sơ lược về một công cụ build và hỗ trợ quản lý build áp dụng cho các dự án sử dụng các ngôn ngữ lập trình khác nhau. 4. So sánh, đánh giá các công cụ quản lý build gồm: Maven, Ant, Gradle áp dụng cho các dự án Java 5. Trình bày ch i tiết về Maven. Làm việc với các lệnh của Maven  * Demo  1. Demo quy trình build tự động và build liên tục sử dụng Maven áp dụng cho dự án Jenkins kết nối đến kho quản lý mã nguồn trên GitHub 2. Tích hợp một số plugin và các phụ thuộc hỗ trợ các hoạt động kiểm thử (đơn vị, hệ thống), tìm lỗi (findbugs, spotbugs, …), xét duyệt mã nguồn và phân tích mã nguồn tĩnh (static code analysis), …. 3. Hiển thị và Phân tích kết quả build. |
| 10 | Continuous deployment -pipelines | 5 | * Lý thuyết:   + Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề   + Trình bày tổng quan về DevOps   + Trình bày chi tiết về quy trình triển khai liên tục (Continuous Deployment - CD) .   + Giới thiệu một số công cụ trợ giúp CD, ví dụ: Jenkins pipeline, Codefresh,Azure pipelines, Buddy, ...   + Xây dựng đường ống phát hành liên tục (CD) sử dụng Jenkins pipeline. * Demo: Demo quy trình phát hành liên tục dự án DevOps sử dụng Jenkins pipeline. |
| 11 | Repository management- Nexus | 4 | * Lý thuyết:   + Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề   + Trình bày tổng quan về các chức năng quản lý kho artifacts. Tầm quan trọng của Artifacts Repository Management.   + Giới thiệu một số công cụ quản lý kho (ví dụ Maven, Nexus, ...) Phân tích ưu và nhược điểm của công cụ.   + Trình bày chi tiết về Nexus, cách định dạng gói mà Nexus quản lý. Quy trình làm việc với kho Nexus.   + Tích hợp Nexus với Jenkins. * Demo:   + Demo quy trình đóng gói và phát hành dự án jenkins (gắn với mã nguồn trên kho Github) lên kho Nexus.   + Phân tích kết quả demo. |
| 12 | Quản lý mã nguồn - Git/Bitbucket | 3 | * Lý thuyết:   + Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề   + Giới thiệu về Git&Bitbucket hỗ trợ trong quản lý mã nguồn.   + Mô tả chi tiết về mô hình phân nhánh được sử dụng để quản lý mã nguồn của dự án   + Lập kế hoạch phát hành các phiên bản phần mềm dự kiến và kế hoạch bảo trì, sửa mã tương ứng cho từng phiên bản dự kiến   + Danh sách các lệnh Git ứng được sử dụng trong quá trình triển khai kế hoạch bảo trì * Demo:   + Demo quy trình bảo trì sửa mã theo bản kế hoạch đã lập mô hình phân nhánh được sử dụng để demo   + Phân công, giám sát quá trình sửa đổi và gán nhãn các phiên bản tương ứng. Thống kê các đóng góp của từng thành viên tham gia cộng tác phát triển mã. |
| 13 | FindBugs in source code – FindBugsTM Eclipse plugin | 5 | * Lý thuyết:   + Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề   + Trình bày tầm quan trọng của hoạt động đảm bảo chất lượng mã nguồn. Giới thiệu một số kỹ thuật trợ giúp đánh giá, đảm bảo chất lượng mã nguồn (ví dụ: test, thanh tra, phân tích mã nguồn, review code, QA, smell code, ...).   + Giới thiệu về các bug templates trong đảm bảo chất lượng mã nguồn. Cơ sở (các tiêu chí, tiêu chuẩn, các hướng dẫn code) để review code   + Sự khác biệt giữa verify code tự động và verify code thủ công. So sánh ưu và nhược điểm của từng cách tiếp cận này   + Giới thiệu các công cụ hỗ trợ verify code tự động, đảm bảo chất lượng mã (ví dụ: Codacy, CodeFactor, Code Climate, Pull Assistant, .... ). Trình bày chi tiết về công cụ (các công cụ) được lựa chọn sử dụng để demo.   + Mô tả quy trình find bugs in source code sử dụng công cụ đã lựa chọn. * Demo 1:   + Demo quy trình find bugs code thủ công sử dụng Findbugs plugin   + Đánh giá kết quả verify code * Demo 2:   + Demo quy trình verify code tự động sử dụng công cụ lựa chọn trên dự án Jenkins   + Phân tích kết quả verify   + Sửa code khắc phục lỗi bugs   + Lặp lại quy trình find Bugs |
| 14 | Test tự động - Selelium | 4 | * Lý thuyết:   + Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề   + Tầm quan trọng của hoạt động test hệ thống và test chấp thuận trước khi đóng gói và phát hành ứng dụng.   + Sự khác biệt giữa test thủ công và test tự động.   + Giới thiệu một số công cụ hỗ trợ test hệ thống và test chấp thuận. Trình bày chi tiết về Selenium.   + Phương pháp sinh test cases   + Trình bày chi tiết về Selenium. Các bước test tự động sử dụng Selenium. * Demo:   Demo quy trình test hệ thống, test chấp thuận sử dụng Selenium trên dự án Jenkins kết nối với kho mã nguồn GitHub. |
| 15 | Quản lý cấu hình - Jenkins | 3 | * Lý thuyết:   + Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề   + Trình bày tổng quan về quản lý cấu hình phần mềm, tầm quan trọng của quản lý cấu hình phần mềm   + Giới thiệu về Jenkins hỗ trợ trong quản lý cấu hình và tích hợp liên tục (CI).   + Quy trình tích hợp liên tục (CI) và cấu hình dự án jenkins tương ứng với quy trình CI * Demo   + Lập kế hoạch demo quy trình thiết lập cấu hình và quản lý cấu trình trên một số kiểu dự án của jenkins sử dụng quy trình CI   + Demo quy trình CI trên dự án Jenkins sử dụng kế hoạch đã thiết lập. |
| 17 | Code Hosting | 5 | * Lý thuyết:   + Nêu được tầm quan trọng, lý do lựa chọn chủ đề, phạm vi của chủ đề   + Giới thiệu chung về code hosting; Các providers cung cấp hosting   + Trình bày chi tiết về quy trình code hosting (Make code ready; Hosting choosen; Take a domain; Publish source code to hosting) * Demo: * Quy trình code hosting cho dự án mã nguồn cụ thể sử dụng:   + - Git (cấu hình trên server để tự động tải code)     - FTP (File Transfer Protocol)     - Upload file. * Vận hành, bảo trì code sau khi publish lên hosting. |
| 18 | FindBugs – Tìm lỗi trong mã nguồn |  | * Lý thuyết * Bugs là gì? Tại sao cần tìm bugs trong mã nguồn * Các mẫu bugs? Liệt kê danh sách các bug templates * Find bugs dựa trên cơ chế nào, tìm bug thủ công và tìm bug tự động sử dụng công cụ trợ giúp * Liệt kê các công cụ trợ giúp tìm bugs. So sánh, đánh giá ưu/nhược điểm của từng công cụ. Trình bày cụ thể về công cụ được sử dụng trong demo * Demo * Giới thiệu dự án mã nguồn cần find bugs * Trình bày về công cụ findbugs được sử dụng * Thực hiện findbugs * Phân tích kết quả, gỡ rối và thực hiện lại quy trình demo * Tham khảo:   Links:  <http://findbugs.sourceforge.net/manual/index.html>  <http://findbugs.sourceforge.net/publications.html> Ví dụ: FindBugs (1.2.1-dev-20070506) Analysis for jdk1.7.0-b12: =>See link: <https://findbugs.cs.umd.edu/demo/jdk7/index.html> Using the FindBugs™ Eclipse plugin:Link: <http://findbugs.sourceforge.net/manual/eclipse.html>=> FindBugs đã không được bảo trì trong thời gian dài, do vậy ta dùng Spotbugs là một plugin của Maven để thay thế:Spotbugs đang sử dụng FindBugs 3.0.1 (see link: <https://gleclaire.github.io/findbugs-maven-plugin/>)<https://spotbugs.github.io/>Spotbug: Có hơn 400 bug patterns can be fount in link:<https://spotbugs.readthedocs.io/en/latest/bugDescriptions.html>See link để thực hành demo trên dự án cụ thể:<https://www.baeldung.com/intro-to-findbugs> |
| 19 | Reverse-Engineering |  | * Lý thuyết * Demo |
| 21 | Amazon Web Service (AWS) |  | * Lý thuyết * Demo |
| 23 | Docker for DevOps and Developer |  | * Lý thuyết * Demo |

**Lãnh đạo bộ môn Nhóm giảng viên biên soạn**

**YÊU CẦU CÔNG VIỆC:**

* **Về Demo: Mỗi demo cần trình bày:**

- Giới thiệu về dự án/mã nguồn của ứng dụng được sử dụng để demo

- Mục đích demo, kịch bản demo

- Các công cụ sử dụng trong tiến trình demo: download, cài đặt, thiết lập các tham số môi trường (nếu có), mục đích sử dụng của từng công cụ trong kịch bản demo.

- Các bước tiến hành kịch bản demo

- Phân tích kết quả demo. Nhận xét về ưu và nhược điểm của quy trình demo (nếu có)

- So sánh kết quả demo (nếu chủ đề sử dụng nhiều công cụ demo/nhiệm vụ)

* **Về Slide:**
  + Từ 12 đến 15 slide (trình bày tối đa 15 phút)
  + Bảng phân công nhiệm vụ/nhóm
  + Nêu bật được các kết quả đạt được, gồm:
    - Lý do chọn chủ đề, tính cấp thiết của chủ đề, phạm vi của chủ đề
    - Các lý thuyết cần tìm hiểu
    - Các công cụ được sử dụng
    - Kịch bản demo
    - Nhận xét, đánh giá và công việc cần làm trong tương lai.
* **Về Báo cáo:**
  + Báo cáo tối thiểu 45 trang:
    - Hình thức: định dạng giống như báo cáo chuyên ngành
    - Bảng phân công công việc
    - Nội dung: Tổ chức thành 3 chương:
      * Chương 1: Tổng quan

=> chương này trình bày (1) tổng quan về chủ đề; (2) lý do lựa chọn/tính cấp thiết của chủ đề/tầm quan trọng của chủ đề/vị trí của chủ đề trong vòng đời phần mềm; (3) phạm vi của chủ đề; (4) Các nội dung cần triển khai.

* + - * Chương 2: Cơ sở lý thuyết

=> Trình bày các lý thuyết liên quan đến chủ đề (tham khảo phần nội dung công việc)

* + - * Chương 3: Demo

=> trình bày tất cả các thông tin liên quan đến demo (như phần yêu cầu demo)

* **Về tài liệu tham khảo:**
  + Tài liệu tham khảo cần chỉ rõ nguồn gốc, đảm bảo độ tin cậy, tính đúng đắn
  + Cần trích dẫn từ các tài liệu Tiếng Anh.
* **Về phần công việc tương ứng với từng chủ đề:**
  1. Sinh viên có thể bổ sung thêm các phần nội dung khác nếu thấy cần thiết, phù hợp với quy mô (cỡ nhóm) và tính khó của chủ đề.
  2. Dựa trên phần công việc của chủ đề, điền phần công việc của từng thành viên trong nhóm vào cột: Tên nhiệm vụ
* **Về sản phẩm cần nộp cho Giáo viên, mang đi thi và nộp lại phòng khảo thí sau khi kết thúc báo cáo:**
  1. Cuốn báo cáo
  2. Đĩa CD đính kèm cuốn báo cáo: Trong CD gồm:
     + File mềm báo cáo
     + File Slides tóm tắt kết quả đạt được.
     + File phân công công việc cho từng thành viên trong nhóm
     + Demo + file read me: hướng dẫn các bước chạy demo.
* **Về các tiêu chí chấm điểm:**

|  |  |
| --- | --- |
| **NỘI DUNG** | **THANG ĐIỂM** |
| Tính đúng đắn về kiến thức liên quan đến chủ đề | 2 |
| Khả năng lưu loát trong thuyết trình | 2 |
| Khả năng làm việc nhóm | 2 |
| Mức đạt được các nội dung theo kế hoạch | 3 |
| Khả năng đọc sách ngoại ngữ tham khảo; Khả năng tổng hợp kiến thức, viết báo cáo. | 1 |
| **TỔNG ĐIỂM** | **10** |

=> Lớp trưởng tổng hợp kết quả các nhóm + file này sau khi đã điền cột nhiệm vụ của từng thành viên trong nhóm. Lớp trưởng nộp các kết quả cho mình (Giáo viên) trước ngày thi 2 ngày (gửi qua email). Nếu file phân công nhiệm vụ từng thành viên không nộp, mình sẽ fix cứng phần công việc của từng người và dựa vào đó để cho điểm khi báo cáo.