|  |  |
| --- | --- |
| UNIVERSITY OF INFORMATION TECHNOLOGY & COMMUNICATIONS  FACULTY OF INFORMATION TECHNOLOGY  **DEPARTMENT: SOFTWARE ENGINEERING**  **--------------------------** | SOCIALIST REPUBLIC OF VIETNAM  **Independence – Freedom – Happiness**  **-------------------------** |

**LIST OF ESSAY TOPICS**

Course Name: Software Operation and Maintenance

Major: International Information Technology

Semester: Academic Year 2025-2026

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Essay Title** | **Number of Students** | **Task Name** |
| 1 | Continuous Integration and Delivery | 2-3 | * Theory * State the importance, reasons for choosing the topic, and scope of the topic. * Overview of DevOps * The importance of Continuous Integration (CI) and Continuous Deployment (CD) in the DevOps project lifecycle. How is continuity expressed in the integration process? * Present the stages in the continuous integration (CI), continuous delivery (CD), and full DevOps process. Introduce some tools to support the continuous integration (CI), (CD), and (DevOps) process * Present details about the CircleCI continuous integration support tool. Compare with Jenkins. * Describe in detail the CI process using CircleCI. * Demo: * Demo of the continuous integration process using CircleCI applied to a Jenkins project connected to Git&GitHub source code repository * Analyze the demo results, clarifying how continuity is expressed. |
| 2 | Code Refactoring | 2-3 | Theory   * State the importance, reasons for choosing the topic, and scope of the topic. * Overview of software maintenance, the importance of maintenance and ensuring code quality during maintenance * Introduce some tools to help understand the source code and ensure code quality. For example: code browser, reverse engineering, refactoring code, review code, white box code test, ... * Present details about code refactoring in improving code quality (purpose, advantages, universality, refactoring contexts, evaluating code quality after refactoring...) * Introduce IDEs that support code refactoring functions. Present the steps of refactoring using the corresponding refactoring functions.   Demo: Demo of the refactoring process to improve the structure of a specific program through source code:   * The program structure of the application before refactoring and the existing problems. * The program structure after refactoring and the problems solved by refactoring. * Specify the refactoring activities on the program structure to improve code quality and program structure * Run the corresponding refactoring functions on the IDE. * Analyze the results of code improvement after refactoring, the advantages and benefits of refactoring in code maintenance modifications. |
| 3 | Package and Deploy the software | 2-3 | * Theory: * State the importance, reasons for choosing the topic, and scope of the topic. * Overview of packaging and releasing products. * Briefly describe the types of applications (mobile applications, web applications, desktop applications, service applications, tool or plugin applications, ...) and the corresponding packaging and release methods. * Describe the tools used to package and release applications. * Demo: * Demo of the software packaging and release process using different package formats, applied to different types of software * Comment and evaluate the advantages and disadvantages of the manual application packaging and release process. |
| 4 | Source code review | 2-3 | * Theory:   + State the importance, reasons for choosing the topic, and scope of the topic.   + The importance of code review in ensuring product quality   + Criteria and standards for review   + Distinguish between manual review and automated review. Advantages and disadvantages of each review approach   + Introduce some automatic review tools (Codacy, Code climate, Pull Assistant, Codebeat), and manual review support tools (GitHub, Gerrit, ...).   + Present details about the tool selected for the demo. Describe the code review process using the selected tool. * Demo:   + - Demo of the automated code review process combined with manual review using the selected tools     - Analyze review results     - Fix code to correct errors detected during review     - Repeat the demo process. |
| 5 | Test automation | 2-3 | * Theory:   + State the importance, reasons for choosing the topic, and scope of the topic.   + The importance of testing before packaging and releasing the application to the user environment.   + The difference between manual testing and automated testing.   + Present details about system testing and acceptance testing. Corresponding test case generation methods.   + Present details about QTP. Testing process using QTP * Demo: * Demo of the system testing and acceptance testing process using QTP. |
| 6 | Source code management - Git&GitHub | 2-3 | * Theory:  1. State the importance, reasons for choosing the topic, and scope of the topic. 2. Overview of source code management, the importance of managing source code versions during software maintenance 3. Introduce some source code management tools (Git, GitHub, Gerrit, Gitlab, ...). Reasons for choosing Git&GitHub. 4. Present details about the branching techniques used in source code management when working with Git&GitHub. 5. Briefly present the project management features supported in GitHub 6. List of Git commands used during the demo.  * Demo:  1. Plan software maintenance (expected maintenance versions; briefly describe the functions and corresponding modifications of each version; work assignment plan (who, what, time to do, which branch to work on). 2. Introduce the branching model used for the demo. 3. Statistics and tracking of the maintenance process through commit history |
| 7 | Jenkins – continuous integration and application release | 2-3 | * Theory:  1. State the importance, reasons for choosing the topic, and scope of the topic. 2. Overview of DevOps. DevOps process (core and full) 3. Present details about Jenkins support in DevOps project configuration management, supporting continuous integration of DevOps projects 4. Install and work with Jenkins, set up global and local configuration parameters for each type of Jenkins project  * Demo:  1. Create several different types of DevOps projects on Jenkins. 2. Set configuration parameters for each project (configuration parameters include: corresponding stages of the project, what tools are used, what commands need to be executed, ...) 3. Run the process: test, package, release, and execute the project on Jenkins according to the configured parameters. |
| 8 | Cloud computing - Releasing, operating, monitoring operations, and planning software improvements. | 2-3 | * Theory:  1. State the importance, reasons for choosing the topic, and scope of the topic. 2. Overview of the full DevOps process. Briefly introduce the tools that support each stage of the DevOps project 3. The importance of the following activities in the full DevOps process    1. Deployment    2. Operation    3. Monitor the operation process on the cloud    4. Plan software improvements 4. Introduce some clouds, especially focusing on the Amazon cloud. Cloud services such as saas, paas, iaas, ... 5. Present in detail the process a), b), c), d) to the Amazon cloud. 6. Briefly describe the parameters to be monitored during application operation, and plan application improvements based on these parameters.  * Demo:  1. Demo of the process of releasing the application to the Amazon App Store, operating the application, monitoring the application's operation on the cloud, and planning for application quality improvement 2. Analyze demo results, improve code quality based on the improvement plan, and repeat the demo process. |
| 9 | Build management - Maven | 2-3 | * Theory:  1. State the importance, reasons for choosing the topic, and scope of the topic 2. Present an overview of building and packing applications, an overview of build management, and why build management is necessary. 3. Briefly present a build tool and build management support for projects using different programming languages. 4. Compare and evaluate build management tools including: Maven, Ant, Gradle applied to Java projects 5. Present details about Maven. Working with Maven commands  * Demo  1. Demo of the automatic and continuous build process using Maven applied to a Jenkins project connected to a source code repository on GitHub 2. Integrate some plugins and dependencies to support testing activities (unit, system), bug finding (findbugs, spotbugs, ...), code review and static code analysis, ... 3. Display and Analyze build results. |
| 10 | Continuous deployment -pipelines | 2-3 | * Theory:   + State the importance, reasons for choosing the topic, and scope of the topic   + Present an overview of DevOps   + Present details about the Continuous Deployment (CD) process.   + Introduce some CD support tools, for example: Jenkins pipeline, Codefresh, Azure pipelines, Buddy, ...   + Build a continuous delivery pipeline (CD) using Jenkins pipeline. * Demo: Demo of the continuous delivery process of a DevOps project using Jenkins pipeline. |
| 11 | Repository management- Nexus | 2-3 | * Theory:   + State the importance, reasons for choosing the topic, and scope of the topic   + Present an overview of artifact repository management functions. Importance of Artifacts Repository Management.   + Introduce some repository management tools (e.g. Maven, Nexus, ...) Analyze the advantages and disadvantages of the tool.   + Present details about Nexus, how Nexus manages package formats. Working process with Nexus repository.   + Integrate Nexus with Jenkins. * Demo:   + Demo of the process of packaging and releasing a Jenkins project (linked to source code on the Github repository) to the Nexus repository.   + Analyze demo results. |
| 12 | Source code management - Git/Bitbucket | 2-3 | * Theory:   + State the importance, reasons for choosing the topic, and scope of the topic   + Introduce Git&Bitbucket support in source code management.   + Describe in detail the branching model used to manage the project's source code   + Plan the release of expected software versions and the corresponding maintenance and code correction plan for each expected version   + List of Git commands used during the implementation of the maintenance plan * Demo:   + Demo of the code maintenance and modification process according to the branching model plan used for the demo   + Assign, monitor the modification process, and label the corresponding versions. Statistics of the contributions of each member participating in collaborative code development. |
| 13 | FindBugs in source code – FindBugsTM Eclipse plugin | 2-3 | * Theory:   + State the importance, reasons for choosing the topic, and scope of the topic   + Present the importance of source code quality assurance activities. Introduce some techniques to help evaluate and ensure source code quality (e.g., test, inspection, source code analysis, code review, QA, smell code, ...).   + Introduce bug templates in source code quality assurance. Basis (criteria, standards, code guidelines) for code review   + The difference between automatic code verification and manual code verification. Compare the advantages and disadvantages of each approach   + Introduce tools to support automatic code verification and ensure code quality (e.g., Codacy, CodeFactor, Code Climate, Pull Assistant, ....). Present details about the tool(s) selected for the demo.   + Describe the process of finding bugs in source code using the selected tool. * Demo 1:   + Demo of the manual bug finding process using the Findbugs plugin   + Evaluate code verification results * Demo 2:   + Demo of the automatic code verification process using the selected tool on the Jenkins project   + Analyze verification results   + Fix code to fix bugs   + Repeat the bug finding process |
| 14 | Automated testing - Selenium | 2-3 | * Theory:   + State the importance, reasons for choosing the topic, and scope of the topic   + The importance of system testing and acceptance testing before packaging and releasing the application.   + The difference between manual testing and automated testing.   + Introduce some tools to support system testing and acceptance testing. Present details about Selenium.   + Test case generation method   + Present details about Selenium. Steps for automated testing using Selenium. * Demo:   Demo of the system testing and acceptance testing process using Selenium on a Jenkins project connected to a GitHub code repository. |
| 15 | Configuration Management - Jenkins | 2-3 | * Theory:   + State the importance, reasons for choosing the topic, and scope of the topic   + Present an overview of software configuration management, the importance of software configuration management   + Introduce Jenkins support in configuration management and continuous integration (CI).   + Continuous Integration (CI) process and Jenkins project configuration corresponding to the CI process * Demo   + Plan a demo of the configuration setup and management process on several types of Jenkins projects using the CI process   + Demo of the CI process on a Jenkins project using the established plan. |
| 16 | Software maintenance and support techniques | 2-3 | * Theory:   + State the importance, reasons for choosing the topic, and scope of the topic   + Software process models (waterfall, iterative and incremental, Agile&Scrum, Agile & Lean - DevOps). State the advantages and disadvantages of each approach. Advantages and development trends of DevOps in software engineering   + The relationship between operations and software development through the DevOps approach. Present an overview of DevOps (full) and DevOps (core); tools to support each stage in the DevOps process   + Present how to package software for release using different package formats and different forms of packaging.   + Classify software maintenance to build expected software versions. Describe each version. Plan source code modifications to obtain the expected versions.   + Software maintenance - supporting techniques. Example: code browser, refactoring code, reverse engineering, ... * Demo:   + Demo of software maintenance steps and source code version management using Git&GitHub:   + Demo of software packaging with different package formats. |
| 17 | Code Hosting | 2-3 | * Theory:   + State the importance, reasons for choosing the topic, and scope of the topic   + General introduction to code hosting; Providers offering hosting   + Detailed presentation of the code hosting process (Make code ready; Hosting chosen; Take a domain; Publish source code to hosting) * Demo: * Code hosting process for a specific source code project using:   + - Git (configured on the server to automatically download code)     - FTP (File Transfer Protocol)     - Upload file. * Operating and maintaining code after publishing to hosting. |
| 18 | FindBugs – Finding bugs in source code | 2-3 | * Theory * What are bugs? Why is it necessary to find bugs in source code * Bug patterns? List of bug templates * How does FindBugs work, manual bug finding and automatic bug finding using supporting tools * List supporting tools for finding bugs. Compare and evaluate the advantages/disadvantages of each tool. Present details about the tool used in the demo * Demo * Introduce the source code project to find bugs * Presentation about the findbugs tool used * Perform findbugs * Analyze the results, troubleshoot, and redo the demo process * References:   Links:  <http://findbugs.sourceforge.net/manual/index.html>  <http://findbugs.sourceforge.net/publications.html> Example: 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 has not been maintained for a long time, so we use Spotbugs as a Maven plugin to replace it:Spotbugs is using FindBugs 3.0.1 (see link: https://gleclaire.github.io/findbugs-maven-plugin/)<https://spotbugs.github.io/>Spotbug: There are more than 400 bug patterns that can be found in link:<https://spotbugs.readthedocs.io/en/latest/bugDescriptions.html>See link for demo practice on a specific project:<https://www.baeldung.com/intro-to-findbugs> |

**JOB REQUIREMENTS:**

* **About Demo: Each demo needs to present:**

- Introduction to the project/source code of the application used for the demo

- Demo purpose, demo scenario

- Tools used in the demo process: download, install, set environment parameters (if any), purpose of using each tool in the demo scenario.

- Steps to conduct the demo scenario

- Analyze demo results. Comments on the advantages and disadvantages of the demo process (if any)

- Compare demo results (if the topic uses multiple demo tools/tasks)

* **About Slide:**
  + From 12 to 15 slides (maximum presentation time of 15 minutes)
  + Task assignment table/group
  + Highlight the achieved results, including:
    - Reasons for choosing the topic, urgency of the topic, scope of the topic
    - Theories to learn
    - Tools used
    - Demo scenario
    - Comments, evaluations, and future work.
* **About the Report:**
  + Report minimum 45 pages:
    - Format: similar to a specialized report
    - Work assignment table
    - Content: Organized into 3 chapters:
      * Chapter 1: Overview

=> This chapter presents (1) an overview of the topic; (2) the reason for choosing/urgency of the topic/importance of the topic/position of the topic in the software lifecycle; (3) the scope of the topic; (4) The contents to be deployed.

* + - * Chapter 2: Theoretical basis

=> Present the theories related to the topic (refer to the work content section)

* + - * Chapter 3: Demo

=> Present all information related to the demo (as in the demo requirements section)

* **About references:**
  + References need to clearly state the origin, ensuring reliability and accuracy
  + Citations from English documents are required.
* **Regarding the work corresponding to each topic:**
  1. Students can add other content sections if they deem it necessary, appropriate to the scale (group size) and difficulty of the topic.
  2. Based on the work of the topic, fill in the work of each member in the group in the column: Task name
* **Regarding the products to be submitted to the Teacher, taken to the exam, and returned to the testing room after the report ends:**
  1. Report document
  2. CD attached to the report: The CD includes:
     + Soft copy of the report
     + File Slides summarizing the results achieved.
     + File assigning work to each member of the group
     + Demo + read me file: instructions on how to run the demo.
* About the scoring criteria:

|  |  |
| --- | --- |
| **CONTENT** | **SCORE** |
| Accuracy of knowledge related to the topic | 2 |
| Fluency in presentation | 2 |
| Teamwork ability | 2 |
| Level of achievement of the contents according to the plan | 3 |
| Ability to synthesize knowledge and write reports. | 1 |
| **TOTAL SCORE** | **10** |