CIS 178 - Build Automation for DevOps and QA

COLLEGE:
Merritt College

ORIGINATOR: Brown, Courtney

DIVISION/DEPARTMENT:
Merritt - Division II/M - Technology

STATE CONTROL NUMBER: CCC000614955

DATES:
BOARD OF TRUSTEES APPROVAL DATE: 02/25/2020
STATE APPROVAL DATE: 03/11/2020
CURRICULUM COMMITTEE APPROVAL DATE: 12/12/2019
REQUISITE VALIDATION: 11/05/2019

CURRENT EFFECTIVE DATE: 01/01/2021

1. REQUESTED CREDIT CLASSIFICATION:

D - Credit - Degree Applicable

N - Not Basic Skills

1 - Program Applicable

2. DEPT/COURSE NO:

CIS 178

3. COURSE TITLE:

Build Automation for DevOps and QA

4. COURSE:

Course
MC Course Modification

TOP NO. 0707.30 - Computer Systems Analysis*

5. UNITS:

Variable No

Units (Min) 4.000

Min Total

Hours

Lecture Hours (Min) 3.000

52.5

Lab/Studio/Activity Hours (Min) 3.000

52.5

6. SELECTED TOPIC:

NO. OF TIMES OFFERED AS SELECTED TOPIC:

AVERAGE ENROLLMENT:

7. JUSTIFICATION FOR COURSE:

Every company that creates software needs a way to test it. This course covers current tools and techniques for software quality assurance, building and releasing software systems. These same tools are used in DevOps, the automation of infrastructure management. A search for DevOps on Indeed.com returns approximately 10,000 full-time and contract jobs with salries that range from $75,000 through $120,000. This course provides the skills described in those job descriptions and builds on basic skills learned through courses in Linux, introductory programming and Software Engineering. It provides experience with current software systems and best practices. This course aligns the Merritt Cybersecurity program Center For Academic Excellence 2-year Cyber Defense (CAE2Y-CD) national standard for cybersecurity education enabling articulation to 4 year programs.

8. COURSE/CATALOG DESCRIPTION

Design and integration of applications development (Dev) tools and Operations tools (Ops) into automated control systems: monitoring of Source Code Management (SCM) repositories for changes to initiate automate software build, infrastructure provisioning, or configuration updates. Creation of Virtual Machines suitable for on-the-job use through hands-on project based learning. Apply best practices for toolchain configuration, monitoring, and testing software for Quality Assurance (QA).

9. OTHER CATALOG INFORMATION

a. Modular: No

If yes, how many modules:

b. Open entry/open exit: No

c. Grading Policy: Both Letter Grade or Pass/No Pass

d. Eligible for credit by Exam: No

e. Repeatable according to state guidelines: No

f. Required for degree/certificate (specify):

g. Meets GE/Transfer requirements (specify): See GE tab

h. C-ID Number:

Expiration Date:

i. Are there prerequisites/corequisites/recommended preparation for this course? Yes

10. LIST STUDENT PERFORMANCE OBJECTIVES (EXIT SKILLS):

If an Objective cannot be deleted, make sure a Content-Review found in the Content Validation Page is not using that objective.

Objectives

1. **Utilize the contents of a repository to build versioned production software.**
2. **Implement, exercise and manage QA and regression tests for known defects and issues.**
3. **Perform continuous integration tests on the contents of a repository.**
4. **Build a virtual machine that is a server and accumulates all the tools used for assignments.**

11. COURSE CONTENT:

LECTURE CONTENT:

A. Overview & Definitions of Software Quality, QA, Build and Release (10%)
B. Software Build Methodology and Tools (15%)
C. Constructing a Test Plan (15%)
D. Test Execution & Documentation (15%)
E. Release & Version Tagging (15%)
F. Advanced Automation & Continuous Integration (15%)
G. Distributed Builds (15%)

LAB CONTENT:

1.

Make Build and Run Native Application Make is a tool that is used to compile software. It is the most widely used build technology in software development. In this assignment the student is introduced to make by using it to build one category of software, the native application. Native applications are some of the most common category encountered in software development. The student learns about this important category, how to build it manually, how to build it with automation, and how to test it with automation. (33%)

2.

QA Metrics - Code Coverage: Game-Of-Life The student links their VM to a software repository hosted on Github. They write scripts that cause their build automation server to retrieve the source code for a complex repository-hosted project such as "Game of Life" and build the software. They then write scripts to generate a measurement of how much of the code is covered by a test case. The students learn about different quality metrics, tools to measure them, and generating readable reports on overall quality of the software under development. (33%)

3.
Ant Build and Run Java JVM Application Java is one of the most widely used languages for software development. The tool commonly used to build Java applications is called ANT. The student is given Java source code and required to write instructions for Ant on how to build the Java application. In order to do this they must: 1. Create the appropriate directory structure and extract the source code into the correct place. 2. Unzip the threesData.bin file and put it in the threesData.bin file appropriate directory. 3. Write a build.xml that builds a binary named CountThrees from the source code accounting for any dependencies 4. Create a build.xml target named test with the appropriate dependencies that when invoked will execute the program 5. Upload only the build.xml using the assignment link. (34%)

12. METHODS OF INSTRUCTION (List methods used to present course content):

- Lecture
- Lab
- Observation and Demonstration
- Discussion
- Projects
- Directed Study
- Threaded Discussions

Other Methods:

13. ASSIGNMENTS

Out-of-class Assignments (List all assignments, including library assignments. Requires two (2) hours of independent work outside of class for each unit/weekly lecture hour. Outside assignments are not required for lab-only courses, although they can be given.)

Override Outside Class Hours: No

Outside-of-Class Hours (Min) 6.000

Outside-of-Class Hours (Max) 0.000

Override Outside-of-Class Hours (Min) 0.000

Override Outside-of-Class Hours (Max) 0.000

Out of class Assignment
Build a Virtual Machine & install Command Line Tools The student begins assembling their bare Continuous Integration (CI) server by installing a basic system with ssh access. This server accumulates all the tools used for assignments. On completion of this course the student has a stand-alone Virtual Machine (VM) that they can take with them to demonstrate proficiency in interviews, or use on the job in their professional work.

Install Tomcat and the Jenkins Continuous Integration (CI) Server The majority of web applications are deployed using Tomcat which is a Java application container; an application that is used to host other

applications. Into this container they install one of the common build automation tools Jenkins. This forms the core tool used for most other assignments. The student learns about the typical web application infrastructure that they will encounter by building it themselves. They get first hand knowledge of the kind of manual systems configuration that they will later learn how to automate.

Autotools, Building & Installing Git The GNU build system, also known as the Autotools, is a suite of programming tools designed to assist in making source code packages portable to many Unix-like systems. It can be difficult to make a software program portable: the C compiler differs from system to system; certain library functions are missing on some systems; header files may have different names. One way to handle this is to write conditional code, with code blocks selected by means of preprocessor directives (#ifdef); but because of the wide variety of build environments this approach quickly becomes unmanageable. Autotools is designed to make this problem more manageable. The student also learns how to interact with software repositories like Github by building and using their own client tool git. This assignment teaches the student about issues in cross-platform software and what tools and best practices they can use to mitigate those issues.

14. STUDENT ASSESSMENT: (Grades are based on):

- ESSAY (Includes "blue book" exams and any written assignment of sufficient length and complexity to require students to select and organize ideas, to explain and support the ideas, and to demonstrate critical thinking skills.)
- COMPUTATION SKILLS
- NON-COMPUTATIONAL PROBLEM SOLVING (Critical thinking should be demonstrated by solving unfamiliar problems via various strategies.)
- SKILL DEMONSTRATION
- MULTIPLE CHOICE

OTHER (Describe):

15. TEXTS, READINGS, AND MATERIALS

A. Textbooks:
YesNo37
Smart, John Ferguson. *Jenkins: The Definitive Guide*. 1 edition O'Reilly, 2011.
Humble, Jez, Farley Dave. *Continuous Integration*. 1 edition Addison Wesley, 2011.

VirtualBox. Software. N/A edition. Oracle

Online site with live documentation and wiki https://wiki.jenkins-ci.org/display/JENKINS/Home

*Date is required: Transfer institutions require current publication date(s) within 5 years of outline addition/update.

B. Additional Resources:
Library/LRC Materials and Services:

The instructor, in consultation with a librarian, has reviewed the materials and services of the College Library/LRC in the subject areas related to the proposed new/updated course

Print Materials were reviewed? Yes

Non-Print Materials were reviewed? No

Online Materials were reviewed? Yes

Services were reviewed? Yes

Specific materials and/or services needed have been identified and discussed. Librarian comments:

C. Readings listed in A and B above are: (See definition of college level):
YesNo39
Primarily college level

16. DESIGNATE OCCUPATIONAL CODE:

B - Advanced Occupational

17. LEVEL BELOW TRANSFER:

Y - Not applicable

18. CALIFORNIA CLASSIFICATION CODE:

Y - Credit Course

19. NON CREDIT COURSE CATEGORY:

Y - Not Applicable, Credit course

20. FUNDING AGENCY CATEGORY:

Y - Not Applicable (funding not used to develop course)

REQUISITES AND ADVISORIES

RECOMMENDED PREPARATION:
CIS 005 Introduction to Computer Science or CIS 006 Introduction to Computer Programming and CIS 072 Systems and Network Administration CIS 108 Scripting for Systems Automation and Data Analysis

STUDENT LEARNING OUTCOMES

1. **Plan and implement continuous integration tests on the contents of a repository,**

   Assignments and projects that use an automation tool to retrieve software, configure the environment, launch tests and collet results.

2. **Identify software quality issues and manage QA and regression tests for known defects and issues.**

Assignments and projects that include specific tests to be run on a repository and the metrics to be calculated and displayed.

3. **Select code and functional elements from repository for inclusion in production release candidates.**

   Quiz/exam questions, project, assignment and laboratory exercising on selecting components to include in a software build.

4. **Interpret and manage build identification tags that distinguish between specific software versions.**

   Create projects that monitor repository versions and initiate action when a change is recognized.