**AFS-205 Course Design Guide**

\*The contents of this course design guide will be built in Canvas, exactly as written.

|  |  |
| --- | --- |
| **Learning Experience Designer** | Margot Cassidy |
| **SME** | Dr Ph.D Camerounan Kenne |
| **Program Director** | TBD |
| **Development Dates** | Create content 5/1-7/5, Build course in Canvas 7/6-7/10, QC 7/10, Publish 7/13 |
| **Course Start Date**  **(see Wrike)** | 6/1/20 |
| **Syllabus - Course Details (see Safe Place)** | |
| **Course Title** | Web Infrastructure & Server Deployment - NGINX, Apache, & Docker |
| **Course Number** | AFS-205 |
| **Credit Hours** | 3 credits  15 didactic; 60 lab; 30 homework = 105hrs  [Sample Credit Matrix](https://docs.google.com/spreadsheets/d/1yfzQOV25pcXcitFfVLfagQqvSoN-esrFJfCCqAJ8c5Y/edit#gid=1225069450)  ***Didactic Hrs***.   * Zoom - live classes   + Required Wks 1, 3, 5 and 7; **60 minutes** * Slack - support, community, collaboration, sharing of info (free - browser-based) * Scrimba video lectures - present content; practice coding; triple the time of the video length for practice   ***Homework Hrs***   * Slack * Research Help Resources - post them in Slack to assist your peers * Reflection assignment - wk 8 * Coding challenges   ***Lab Hours***   * Badges/Portfolium   + Git and GitHub * Cerego - Learn its * VisualStudio Code text editor assignments and projects |
| **Prerequisite(s)** | FSW100, FSW-115 |
| **Corequisite(s)** | NA |
| **Course Description** | Students will learn how to create servers and reverse proxies using web servers (e.g. NGINX, Apache) and common web communication protocols. This course also covers server deployment with Docker, from initial configuration to the cloud, so students might learn how to create, deploy, and run applications.). Required text/materials: Web Infrastructure Interactive Lessons powered by V School (included in weekly course materials). |
| **Required Textbook(s) (APA citation)** | * Web Development Interactive Lessons |
| **VitalSource VBID (ISBN)** | NA |
| **Cerego Series LTI URL** |  |
| **SIS ID**  **(CampusVue or registrar)** |  |
| **Instructional Methods** | The instructional methods utilized in this course include:   * Learn Its * Practice * Assignments/Projects * Live Class   You are expected to participate in and contribute to all class activities. You can expect that these activities require the following number of hours each week:   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Week** | **Learn It** | **Practice** | **Projects** | **Live Class** | | 1 | 2 | 2 |  | 1 | | 2 | 2 | 2 |  |  | | 3 | 2 | 2 |  | 1 | | 4 | 2 | 2 |  |  | | 5 | 2 | 2 |  | 1 | | 6 | 2 | 2 |  |  | | 7 | 2 | 2 |  | 1 | | 8 | 0 | 0 |  |  | |

|  |  |
| --- | --- |
| Syllabus - Outcomes and Objectives | |
| **Course Outcomes and Weekly Objectives**   * Build a Docker ecosystem using multiple microservices for a specific role or objective. * Create containers in Docker and Docker Desktop to host web services. * Apply BusyBox and parallel instances with Apache server. * Create routes to understand middleware and NGINX server. * Reverse proxy servers in the Docker ecosystem. * Deploy servers and host websites with Apache server, Docker, and NGINX.   WEEK 1: **Introduction to Docker**   * Explain the concepts of NGINX and Docker. * Explain how Docker servers are utilized in web development. * Build web structure in Docker and NGINX.   WEEK 2:  **Introduction to NGINX Servers**   * Explain NGINX and its benefits in server development. * Use APIs to inject dependencies with Postman. * Create NGINX layout and build containers using Docker. * Create container logs in Docker.   WEEK 3: **Docker Development and Configuration**   * Create a request body and transfer data to the container. * Configure a port to proxy HTTP server. * Secure a container in Docker and create data with Redis. * Create an ID container and identify microservices such as NGINX. * Collect ID containers in Docker ecosystem.   WEEK 4: **Apache Server Configuration and Security**   * Deploy Apache server in Docker. * Create a middleware route with Apache. * Configure Apache security. * Explain the difference between Apache server and NGINX. * Create reverse proxies with Apache.   WEEK 5: **Performance and Configuration**   * Create reverse proxies with Apache, NGINX, and Docker. * Build an Apache (httpd) web server inside a container. * Create an action API script. * Create a directory to hold Dockerfile. * Use a form to add, delete, and update data, using HTTP requests.   WEEK 6: **Configuring Web Services**   * Create an NGINX controller. * Create Apache security in Docker. * Create reverse proxy security and hosting. * Deploy Apache server and NGINX. * Handle errors in Docker.   WEEK 7: **Capstone Project**   * Reverse proxy security with HTTP, HTTPS and HTTPD. * Implement transfer protocol of NGINX or Apache server. * Explain configuration, deployment, and process of web development. * Review important terms and concepts of the course.   WEEK 8: **Reflection**   * Reflect on the concepts learned in the course. | |

# **Point Breakdown (**this can change**)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Assignment** | **Points per Item** | **Total Points** | **Grade Category** |
| Learn Its x 6 | Varies | 160 | Learn Its |
| Practice Assignments/Activities | --- | 350 | Assignments |
| Projects |  | 350 | Projects |
| Live Class Participation x 4 | 35 | 140 | Participation |
|  |  |  |  |
|  |  |  |  |
|  |  | **Total: 1000** |  |

# **Category Breakdown**

|  |  |
| --- | --- |
| **Grade Category** | **Percentage** |
| Learn Its | 16% |
| Participation | 14% |
| Practice | 35% |
| Projects | 35% |
|  | **Total: 100%** |

|  |  |
| --- | --- |
| **Note to the Instructor** | |
| Standard Note to the Instructor verbiage will go at the top of this page.  **Peer Mentor Program**  Keep an eye out for students that are excelling in your courses, and nominate them for the Peer Mentor program. Peer mentors are paid, part-time student workers who provide support, encouragement, and information about being a student.  Contact your PD and/or [nick.keeling@bryanuniversity.edu](mailto:nick.keeling@bryanuniversity.edu) if you’ve identified an excelling student you’d like to nominate.  **LAB HOURS (office hours)**  During the off-weeks of live events (weeks 2, 4, 6 and 8), you need to be available for at least two hours. In other words, set up an open Zoom session during the day(s)/time(s) you specify, and promote these “Open Lab Hours” to students, so that they can pop-in and ask questions, get help on coding problems, etc. -- it’s their “lab time” with you.  **LIVE EVENTS**  Live events should always be recorded, so be sure to press “Record” when starting class. You do this so you can post the recording in Slack, for those students who may have missed the live event.  Week 1 Live Event Topic   * Welcome & Introductions (5-10mins) * Course Overview - objectives, material, assignments (5-10mins) * Week 1 Overview - WebSite Ecosystem Structural Docker vs Microsystem, Servers, Apache, NGINX, RET and API hosting (10-15mins) * Demonstrate basic server code and functionality (5-10mins) * Discuss & Demonstrate Modularization and Node Modules & Patterns (15-20mins) * Q & A (5-10mins     Week 3 Live Event Topics  Apache- installation & explanation (5mins)   * Docker - installation & show example(s) of basic server code written using Express. (15-20mins) * NGINX - installation * Discussion - HTTP, APIs, REST, GET AND SETTING SERVER, DEPLOY, CONFIGURATION (5-10mins) * Security controller REVERSE PROXY * Concept of Programing - installation, discussion of functionality, explanation of interface, demonstration of GET Request. (20-30mins) * Q & A (5-10mins)   Week 5 Live Event Topics   * Discussion - Apache Server, NGINX hosting, the security controller, Morgan, URL Parameters & Queries. (10-15mins) * Create and/or Demonstrate performance Server functionality, including performing HTTP Requests (GET, GET One, PUT, POST, DELETE) using application database and Postman (25-30mins) * Discuss all the technical specifications * Discuss Capstone (5-10mins) * Q & A (5-10mins)   Week 7 Live Event Topics   * Reverse Proxy & load Balancing * Importance of protocol and backend server * Configuration AWS vs Apache Server   **SLACK**  This is your go-to tool for building community in your class, communicating to students, and for students to communicate with each other. They should get used to using this tool, as it’s an industry standard!  As you’re monitoring the communication and collaboration between students, make note of a student(s) that is expressing effective written communication and/or teamwork. It is likely that in this first class, there won’t be too much of it, however, if a student stands out in these two areas, go ahead and complete this pathway requirement in Portfolium.  Best practices for faculty:   * Ensure students have added the #BryanWebDevProgram channel, to connect all students in the BU web dev program. They were instructed to do this and setup slack in their UNV-101T course. * Create a channel just for this course using the following naming convention: #CourseCode-StartDate (e.g. #FSW-100-8.19.19)   1. Post live event recordings for anyone that missed live class.   2. Post course announcements or any resources you think will be helpful that week * Create a Questions channel just for your course (e.g. #FSW-100Questions-8.19.19) - Students should know that if they have a question, they should follow these steps:   1. Search Google, ask a classmate, then ask you * Set up important reminders with the /remind command. One example is having a reminder fire every weekday afternoon with the 3 needed commands for the students to push their code to github. * Have separate slack channel for students and staff, with added channels for every class. * Set a time limit on your notifications in settings (ex: off between 9pm and 7am). Allow for @username direct messages to alert you between your open hours.   **PRACTICE EXERCISES & PROJECTS**  Setup your own GitHub account with your Bryan University email address. Make sure that students add your email to their GitHub, so that you can be notified when they’ve shared something in their repository.  Web development requires a lot of manual grading, so be sure to setup notifications and/or enough time to provide timely feedback once a student submits work. And, because the industry-standard is to push projects to GitHub, you will have to access GitHub a lot, in order to grade most students’ work. Below is some guidance on the grading of the practice exercises and projects in this course:  Week 1: Answer Guidance | |

[Course Design Guide Help Document](https://docs.google.com/document/d/1ccwyf4rV5Jo1gBF7nMq_OekTQ1JFYcHBectVePxYyPA/edit?usp=sharing)

|  |  |
| --- | --- |
| **Week 1: Introduction to Docker** | |
| Weekly Notes/Brainstorming area  Docker containers run natively in windows Server 2016, Windows server 2019 and window 10. These labs are based on the latest release of windows and Docker with provide the best experience to containerized windows applications   * [Install Docker Desktop on window 10](https://hub.docker.com/editions/community/docker-ce-desktop-windows) | |
| **Objectives** | |
| **Step 1: Acknowledge Syllabus** |  |
| **Step 2: Why Learn This?**  **Type:** Header/**Content Page**/Assignment/Quiz | **Due Date:** Week 1, Day 3 | **Points:** X | **Grade Category:** X | **Requirement:** View | **Submission Type: None**/File Upload/Text Entry/External Tool | **Alignment**  Objectives: |
| Image result for nginx and docker logo Image result for nginx and docker logo Monitor Apache Server Load & Page Statistics using Apache Mod_status  ~~Docker Nginx Web Server - file sharing/copying & Dockerfile~~  Thus far on your journey through Bryan University's Web Development program, we have focused primarily on Web Development, building what everyone sees on a website. But what goes-on-behind the scenes: what happens to our username and password or credit card information when we enter them? How does a website return the list of items I am interested in?  The next part of your web development education will focus on understanding the server part, which handles the functionality of web applications. In this course, we will focus on different levels of framework programming, one portion of Docker, NGINX, Apache server etc... And since we are learning the MERN Stack, we focus on the APIs and the E Unix and Linux capability. Servers act as an intermediary between the User and the source of a website’s data, generally stored in a database.  Map of NGINX Event-Driven Architecture  Kubernetes vs. Docker: A Primer - Container Journal  **NGINX MASTER ARCHITECTURE STRUCTURAL FUNCTIONALITY OF THE SYSTEM** | |
| **Step 3: Participate in the Week 1 Live Event**  **Type:** Header/Content Page/**Assignment**/Quiz | **Due Date:** Week 1, Day2 | **Points:** 35 | **Grade Category:** Participation | **Requirement:** MarkDone | **Submission Type: None**/File Upload/Text Entry/External Tool |  |
| Active participation is essential to your overall learning. It is important that you come to class prepared, ready to learn, and be actively engaged in class activities and discussions with your instructor and classmates.  To access your live class sessions, click **Web Meeting**. | |
| **Step 4: Review of Servers**  **Type:** Header/Content Page/**Assignment**/Quiz | **Due Date:** Week 1, Day3 | **Points:** X | **Grade Category:** Resources | **Requirement:** MarkDone | **Submission Type: None**/File Upload/Text Entry/External Tool | **Alignment**  Objectives: |
| **~~Web Structure. Apache Server Nginx & Docker Programming~~**  Add content about what Servers are  The differentiation between Apache Server and NGinx and Docker is quite simple:   * Structural web developers build websites and hosting to different levels of server **looks**. * Backend developers build how a website **works**.   Obviously there is much more to understanding each type of programming. This course will focus on providing you a thorough understanding of server development, part of backend programming.   * The term ‘server’ can be vague and is often misunderstood. Read these articles to gain a better perspective of servers:<https://www.lifewire.com/servers-in-computer-networking-817380> * [https://docs.docker.com/toolbox/toolbox\_install\_window](https://docs.docker.com/toolbox/toolbox_install_window/) * [Releases · docker/toolbox](https://github.com/docker/toolbox/releases) * Docker Introduction fundamental   Next, review the Pause and Practice videos from FSW-125   * [Intro to Server Side Programming](https://drive.google.com/file/d/1aYGwzrhykCMw-ASrNEN_BoEBqdJCFSNI/view?usp=sharing) * [HTTP Review](https://drive.google.com/file/d/1QjMlo6L0NG6sqrZcDbINDQ05gV54WyhV/view?usp=sharing) | |
| **Step 5: Introduction to Docker**  **Type:** Header/Content Page/**Assignment**/Quiz | **Due Date:** Week 1, Day4 | **Points:** X | **Grade Category:** Assignment| **Requirement:** MarkDone | **Submission**  **Type: None**/File Upload/Text Entry/External Tool | |
| Read Introduction to Docker PPT  Install Docker for Windows: [Docker Installation Procedures](https://drive.google.com/file/d/1-vXes3BSEWr5Wysuw8VL_-1BOsdJzX6Z/view?usp=sharing)  How to Use Docker with VisualStudio Code - video |  |
| **Step 6: Complete the Week 1 Learn It: Intro to Docker**  **Due Date:** Week 1, Day 6 | **Points:** 30 | **Submission Type:** External tool  **LTI URL from Cerego:** https://cerego.com/lti/study/970685  Watch this video for step-by-step instructions that show you the best way to start a Learn It:  <iframe src="https://bryanuniversity.instructuremedia.com/embed/fee826eb-9e0e-413e-98f5-99951c9c13b5" width="560px" height="320px" allowfullscreen="allowfullscreen"></iframe>  Return to this Learn It, work on it throughout the week, and you'll see your assignment progress continue to increase. When you reach 100% Assignment Progress, you can stop since you'll have earned 100% on the Learn It.  [This video](https://bryanuniversity.instructuremedia.com/embed/47ce6277-a87e-4f00-8b18-710a3104751a) shows how to work on this Learn It throughout the week to improve your progress and eventually earn 100%.  Your grade on this Learn It will update automatically in the gradebook each time you make progress. So, after your first study session, you may see a grade of only 10-15%. Don't panic! This is normal. As you review the Learn It questions throughout the week, your progress will keep increasing and your grade will keep going up! Keep working on it until you reach 100%.  **Important Note:** You **cannot** reach 100% by studying **on one day only**. You must work on this Learn It several days this week, if you want to earn 100%.  If you have questions, refer to the [Student Guide to Learn Its](https://docs.google.com/document/d/1_i2VSa37EZdTOaOGUlHvMzpyihl7b67pauEHP7n4IDg/edit?usp=sharing). | |
| **Step 7: Complete the Pause and Practice: Set Up Docker Ecosystem**  **Type:** Header/Content Page/**Assignment**/Quiz | **Due Date:** Week 1, Day 7 | **Points:** 35 | **Grade Category:** Assignments | **Requirement:** Mark Done | **Submission Type: None**/File Upload/Text Entry/External Tool | |
| Videos?  **Purpose:**  The purpose of this assignment is to set up your Docker ecosystem.  **Getting Started:**  1. Create a repository account in Docker hub.  2. Create a user name and password when signing to Docker.com  Determine which aspects are the most important to consider to establish “Docker Server and Container” maximization during the project and the first reading of Apache and Nginx.  **Install Server and Docker Ecosystem**   * Install Docker   + First create an account and register as first-time user at <https://www.docker.com/> * Install Nginx - instructions to come * Install Apache Server - instructions to come   **Submission Instructions:**  Screencast only  **1-2 hrs** | |
| **Step 8: Engage in Slack during Week 1**  **Type:** Header/**Content Page**/Assignment/Quiz | **Due Date:** Week 1, Day 7 | **Points:** X | **Grade Category:** X | **Requirement:** Mark Done| **Submission Type: None**/File Upload/Text Entry/External Tool | |
| * What is the difference between Nginx server and Apache server? * Which aspects are the most important to consider to establish “Docker Server and Container” maximization during the project and the first reading of Apache and Nginx?   + For each aspect/area of importance identified, identify key/relevant/critical items and compile facts, C:/dockFiles, elements for more in-depth analysis and record in comparative matrices and relationship between the HTTP protocol and other existing protocol. Use tables to support suppositions, insight, observations and conclusions. * What was your experience like manually scaling applications of servers and installation processing? | |
| **GO MOBILE** | |

|  |  |
| --- | --- |
| Week 2: Introduction to NGINX Servers | |
| Blank section | |
| **Objectives**   * Introduction to NGINX in collaboration with Docker Ecosystem * Explain NGInx server hosting system and its benefits in server development. * Use Application Programming Interfaces (API). * Explain REpresentational State Transfer (REST). * NGINX Controller * NGINX WAF * NGINX Amplify * NGINX High performance balancer | |
| **Complete the Pause and Practice: Command Line**  **Type:** Header/**Content Page**/Assignment/Quiz | **Due Date:** Week 2, Day 3 | **Points:** X | **Grade Category:** X | **Requirement:** Mark Done| **Submission Type: None**/File Upload/Text Entry/External Tool | |
| **NGINX Plus and AWS | NGINX**  **High Performance Applications delivery for Microservices**  Improve your Load Balancer performance with SaltStack and Nginx  **Video**  **Read:** [**https://drive.google.com/file/d/1CqjlAi1madjdQg0A4SgstLQkTRD64XAS/view?usp=sharing**](https://drive.google.com/file/d/1CqjlAi1madjdQg0A4SgstLQkTRD64XAS/view?usp=sharing) | |
| **Engage in Slack during Week 2**  **Type:** Header/**Content Page**/Assignment/Quiz | **Due Date:** Week 2, Day 3 | **Points:** X | **Grade Category:** X | **Requirement:** Mark Done| **Submission Type: None**/File Upload/Text Entry/External Tool | |
| Explain the concept of running a group of containerized applications that interact with each other via HTTP. What did you find difficult about it? Did you find any resources to help you? (If so, share them).  ***Remember:*** Slack is a good way to exemplify your Teamwork and Communication skills, needed to complete your Portfolium badging pathway! | |
| **Complete the Week 2 Learn It:** Introduction to NGINX Servers  **Due Date:** Week 2, Day 5 | **Points:** 25 | **Submission Type:** External Tool  **LTI URL from Cerego:**  https://cerego.com/lti/study/970689 | |
| **Review Your Week 2 Support Resources**  **Type:** Header/**Content Page**/Assignment/Quiz | **Due Date:** Week 2, Day 7 | **Points:** X | **Grade Category:** X | **Requirement:** View| **Submission Type: None**/File Upload/Text Entry/External Tool    **Windows Server**, **Window Application**  Download [docker-signa](https://github.com/moby/docker-signal)  [Automatically start containers](https://docs.docker.com/config/containers/start-containers-automatically/)   * [Limit a container’s resources](https://docs.docker.com/config/containers/resource_constraints/) * [Configure storage drivers](https://docs.docker.com/storage/storagedriver/select-storage-driver/) * [Container security](https://docs.docker.com/engine/security/) * [NGINX Installation](http://nginx.org/en/docs/windows.html) * [NGINX Products](https://www.nginx.com/products/)  Operator exclusive options[🔗](https://docs.docker.com/engine/reference/run/#operator-exclusive-options) Only the operator (the person executing docker run) can set the following options.   * [Detached vs foreground](https://docs.docker.com/engine/reference/run/#detached-vs-foreground)   + [Detached (-d)](https://docs.docker.com/engine/reference/run/#detached--d)   + [Foreground](https://docs.docker.com/engine/reference/run/#foreground) * [Container identification](https://docs.docker.com/engine/reference/run/#container-identification)   + [Name (--name)](https://docs.docker.com/engine/reference/run/#name---name)   + [PID equivalent](https://docs.docker.com/engine/reference/run/#pid-equivalent) * [IPC settings (--ipc)](https://docs.docker.com/engine/reference/run/#ipc-settings---ipc) * [Network settings](https://docs.docker.com/engine/reference/run/#network-settings) * [Restart policies (--restart)](https://docs.docker.com/engine/reference/run/#restart-policies---restart) * [Clean up (--rm)](https://docs.docker.com/engine/reference/run/#clean-up---rm) * [Runtime constraints on resources](https://docs.docker.com/engine/reference/run/#runtime-constraints-on-resources)   [Runtime privilege and Linux capabilities](https://docs.docker.com/engine/reference/run/#runtime-privilege-and-linux-capabilities)  What’s next?  [NGINX SERVICES](https://www.nginx.com/products/)     * [NGINX Solution for Apache ProxyPassReverse](https://www.nginx.com/resources/wiki/start/topics/examples/likeapache/) * [Like Apache: .htaccess](https://www.nginx.com/resources/wiki/start/topics/examples/likeapache-htaccess/) * [Separating Error Logs per Virtual Host](https://www.nginx.com/resources/wiki/start/topics/examples/separateerrorloggingpervirtualhost/) * [IMAP Proxy Example](https://www.nginx.com/resources/wiki/start/topics/examples/imapproxyexample/) * [Using a Perl Script as the IMAP Auth Backend](https://www.nginx.com/resources/wiki/start/topics/examples/imapauthenticatewithapacheperlscript/) * [NGINX CODE LAYOUT](http://nginx.org/en/docs/dev/development_guide.html#code_layout)   Windows [NGINX INSTALLATION](http://nginx.org/)  [NGINX Service for Windows](https://github.com/InvGate/winginx/) which uses [NSSM](http://nssm.cc/) as a wrapper for service behaviour.  1. [Starting, Stopping, and Reloading Configuration](http://nginx.org/en/docs/beginners_guide.html#control) 2. [Configuration File’s Structure](http://nginx.org/en/docs/beginners_guide.html#conf_structure) 3. [Serving Static Content](http://nginx.org/en/docs/beginners_guide.html#static) 4. [Setting Up a Simple Proxy Server](http://nginx.org/en/docs/beginners_guide.html#proxy) 5. [Setting Up FastCGI Proxying](http://nginx.org/en/docs/beginners_guide.html#fastcgi) | |
| **Complete Project 1: Create Docker Containers**  **Type (choose):** **Assignment**/Quiz | **Due Date:** Week 2, Day 7 | **Points:** 50 | **Grade Category:** Assignments | **Submission Type (choose):** **File Upload** / External Tool / Text Entry | **Alignment**  Objectives: |
| **Purpose:**  In this assignment you will create your own NGINX layout. It should have the HEADER, HTTP, MAIL and the STREAM.  **Getting Started:**  You will be building two containers using Docker. These containers can be built starting from an official base image. **Official Images** has a list of different official base images. Start from the image that is suitable for you.   * **Server:**   + Create a volume named "ServerDocker" or something similar.   + The server container will mount "ServerDocker" in "/serverdata".   + This container runs a server application which will create a file of size 1KB with random text data in "/serverdata" and then transfer the file to the client along with the checksum.   + The server application itself can be built using any language you are comfortable with. But, the container should include all the packages that are required to run your application. Choose your base image wisely and install only the necessary packages.   + The port on which the server runs must be specified as a command line argument when we run docker. * **Client:**   + Create a volume named "ServerDocker" or something similar.   + The client container will mount "clientvol" in "/clientdata".   + The client container runs an application that connects to the server, receives the file that the server sends and saves it in "/clientdata".   + Verify that the file is received properly at the clientside by verifying the checksum.   + The client application again can be written in any language that you are comfortable with, but the container should include all the necessary packages. Choose your base image wisely and install only necessary packages.     **Include files**  The following two #include statements must appear at the beginning of every nginx file:  · #include <ngx\_config.h>  · #include <ngx\_core.h>  In addition to that, HTTP code should include  · #include <ngx\_http.h>  Mail code should include  · #include <ngx\_mail.h>  Stream code should include  · #include <ngx\_stream.h>  If you need additional resources, see: [**GET START**](https://docs.docker.com/get-started/)  **Submission Instructions:**  Screencast and URL  1-2 hrs | |

|  |  |
| --- | --- |
| Week 3: Docker Development and Configuration | |
|  | |
| **Participate in the Week 3 Live Event**  **Type:** Header/Content Page/**Assignment**/Quiz | **Due Date:** Week 3, Day2 | **Points:** 25 | **Grade Category:** Participation | **Requirement:** MarkDone | **Submission Type: None**/File Upload/Text Entry/External Tool | **Alignment**  Objectives: |
| Standard boilerplate (see week 1) | |
| **Engage in Slack during Week 3** | |
| Be an active member of the coding community. Participate by:  Explain the process of running a group of containerized applications that interact with each other via volumes, such as Docker Nginx hosting and Apache server. | |
| **Complete the Week 3 Learn It:** Docker Development and Configuration  **Due Date:** Week 3, Day 6 | **Points:** 35 | **Submission Type:** External Tool (Cerego) (35)  **LTI URL from Cerego:** https://cerego.com/lti/study/970690 | |
| **Complete the Pause and Practice:**  **Type:** Header/Content Page/**Assignment**/Quiz | **Due Date:** Week 3, Day 6 | **Points:** 35 | **Grade Category:** Assignment | **Requirement:** Mark Done| **Submission Type: None**/File Upload/Text Entry/External Tool |  |
| Powerpoint: <https://drive.google.com/file/d/11WblAaJBws6sUz7pwdRDBtLfPm6QUg7q/view?usp=sharing>  **Child commands**[**🔗**](https://docs.docker.com/engine/reference/commandline/config/#child-commands)   |  |  | | --- | --- | | Command | Description | | [docker config create](https://docs.docker.com/engine/reference/commandline/config_create/) | Create a config from a file or STDIN | | [docker config inspect](https://docs.docker.com/engine/reference/commandline/config_inspect/) | Display detailed information on one or more configs | | [docker config ls](https://docs.docker.com/engine/reference/commandline/config_ls/) | List configs | | [docker config rm](https://docs.docker.com/engine/reference/commandline/config_rm/) | Remove one or more configs | |  |
| **Complete Assignment: Server Configuration**  **Type:** Header/Content Page/**Assignment**/Quiz | **Due Date:** Week 3, Day 7 | **Points:** 35 | **Grade Category:** Assignment | **Requirement:** Mark Done| **Submission Type: None**/File Upload/Text Entry/External Tool | **Alignment**  Objectives: |
| **Purpose:**  This assignment introduces container orchestration with docker-compose and relevant concepts such as docker network.  **Getting Started:**  **Apache Server**  You will create an .htaccess file to be placed in the root of your public HTML directory or c drive. Through the configuration directives in the .htaccess file, you will accomplish the following:  1. Configure a custom error document response for HTTP response status of 404 ("Not Found").  2. Set expiration headers so that CSS, JavaScript, and image (JPEG, GIF, PNG) files are cached.   * Mime types are: image/jpeg, image/gif, and image/png   3. Set an output filter so that HTML, CSS, and JavaScript files are compressed before they are sent to the web browser.  Mime types are: text/html, text/CSS, text/JavaScript, application/JavaScript  **Submission:**  Screencast video & Github | |
| **Review Your Week 3 Resources** |  |
| **Objectives**   * **3.1** * Network Container * Security Configuration * Environment variables * Configuration files * Command line * Introduction to AWS * AWS setting and configuration * Docker and AWS * AWS architecture * Differences between Docker and AWS * Docker Configuration commands  What’s next?[🔗](https://docs.docker.com/engine/swarm/#whats-next)Swarm mode key concepts and Guideline  * Learn swarm mode [key concepts](https://docs.docker.com/engine/swarm/key-concepts/) * Get started with the [Swarm mode tutorial](https://docs.docker.com/engine/swarm/swarm-tutorial/). * [Category: Container Security & Orchestration](https://www.f5.com/company/blog/container-security-basics-orchestration) * [AWS Container Security Workbook: A Best Practices Guide](https://www.alertlogic.com/solutions/container-security/)  Swarm mode CLI commands[🔗](https://docs.docker.com/engine/swarm/#swarm-mode-cli-commands) **Explore swarm mode CLI commands**   * [swarm init](https://docs.docker.com/engine/reference/commandline/swarm_init/) * [swarm join](https://docs.docker.com/engine/reference/commandline/swarm_join/) * [service create](https://docs.docker.com/engine/reference/commandline/service_create/) * [service inspect](https://docs.docker.com/engine/reference/commandline/service_inspect/) * [service ls](https://docs.docker.com/engine/reference/commandline/service_ls/) * [service rm](https://docs.docker.com/engine/reference/commandline/service_rm/) * [service scale](https://docs.docker.com/engine/reference/commandline/service_scale/) * [service ps](https://docs.docker.com/engine/reference/commandline/service_ps/) * [service update](https://docs.docker.com/engine/reference/commandline/service_update/)   [docker](https://docs.docker.com/search/?q=docker), [container](https://docs.docker.com/search/?q=container), [cluster](https://docs.docker.com/search/?q=cluster), [swarm](https://docs.docker.com/search/?q=swarm)  Container identification[🔗](https://docs.docker.com/engine/reference/run/#container-identification)  Name (--name)[🔗](https://docs.docker.com/engine/reference/run/#name---name) The operator can identify a container in three ways:   |  |  | | --- | --- | | **Identifier type** | **Example value** | | **UUID long identifier** | “f78375b1c487e03c9438c729345e54db9d20cfa2ac1fc3494b6eb60872e74778” | | **UUID short identifier** | “f78375b1c487” | | **Name** | “evil\_ptolemy” | | |
| **Supported networks :**   |  |  | | --- | --- | | **Network** | **Description** | | **none** | No networking in the container. | | **bridge** (default) | Connect the container to the bridge via veth interfaces. | | **host** | Use the host's network stack inside the container. | | **container**:<name|id> | Use the network stack of another container, specified via its *name* or *id*. | | **NETWORK** | Connects the container to a user created network (using docker network create command) |   More Resources:   * [Docker Configuration create](https://docs.docker.com/engine/swarm/configs/) * [101 AWS Security Tips & Quotes, Part 1: Essential Security Practices](https://www.threatstack.com/blog/101-aws-security-tips-quotes-part-1-essential-security-practices) * [Secure your AWS account](https://www.msp360.com/resources/blog/amazon-s3-backup-security-guide/) | |

|  |  |
| --- | --- |
| Week 4: Apache Server Configuration and Security | |
| Weekly Notes/Brainstorming area  **What You Will Learn:** [Docker and Apache] | |
| **Objectives**   * **4.1** Installing apache on the linux platform * Install Apache from the source * Docker handle Apache * What is virtual Host * Configure your very first Production Web Server * Using mod\_evasive to rebutting the Dos attack * Using mod\_security to enhance apache security * Docker [handlers in Apache](https://www.guru99.com/apache.html#13) * [How to run NGINX with Apache](https://www.guru99.com/apache.html#14) * [How to Secure Apache Web Server?](https://www.guru99.com/apache.html#15) * [Hiding Apache version and OS information](https://www.guru99.com/apache.html#16) * [Hiding Apache version and OS information](https://www.guru99.com/apache.html#16) * [Disable Directory Listing](https://www.guru99.com/apache.html#17), start and shooting down. | |
| **Engage in Slack During Week 4**  **Type:** Header/**Content Page**/Assignment/Quiz | **Due Date:** Week 4, Day 3 | **Points:** X | **Grade Category:** X | **Requirement:** Mark Done| **Submission Type: None**/File Upload/Text Entry/External Tool | |
| Explain how to use 3rd party services, such as databases, inside containers as part of your project.  ***Remember:*** Slack is a good way to exemplify your Teamwork and Communication skills, needed to complete your Portfolium badging pathway! | |
| **Complete the Week 4 Learn It:** Apache Server Configuration & Security  **Due Date:** Week 4, Day 5 | **Points:** 25 | **Submission Type:** External Tool  **LTI URL from Cerego:** https://cerego.com/lti/study/970691 | |
| **Complete the Pause and Practice:**  **Type:** Header/Content Page/**Assignment**/Quiz | **Due Date:** Week 2, Day 6 | **Points:** 35 | **Grade Category:** Assignment | **Requirement:** Mark Done| **Submission Type: None**/File Upload/Text Entry/External Tool | |
| Time on Task: Apache vs Nginx: Practical Considerations |  |
|  |  |
|  | |
| **Complete Project 2: Creating an HTTP Connection**  **Type (choose):** Assignment/Quiz | **Due Date:** Week 4, Day 7 | **Points:** 75 | **Grade Category:** Assignments | **Submission Type (choose):** File Upload / External Tool / Text Entry | **Alignment**  Objectives: |
| **Purpose:**  In this assignment, you will redirect Docker to the Apache directory for your favorite book.  **Getting Started:**   1. Create a redirect or rewrite rule so that the URL (where your username is substituted in place of GitHub folder):   Library site:[http://AFS-205students.dce.BryanUniversity.edu/Bryanlibrary/myfavoritebook](http://fsw-205students.dce.bryanuniversity.edu/Brianlibrary/myfavoritebook)  redirects (HTTP 301 or 302) to a page that contains information about a favorite book of yours.  The page that is redirected to should be an external URL (e.g. from Wildcat, Amazon, B&N, Borders, publisher site, etc.).   1. Use "Redirect" unless you want to take on the more difficult (but more flexible) "Rewrite" configuration.   For example, see  [http://AFS-205students.dce.bryanuniversity.edu/~NAME](http://fsw-205students.dce.brianuniversity.edu/~NAME) OFYOURFOLDER/myfavoritebook  Keep in mind that you'll need the full path string on the URL. So, if I wanted http://fswstudents.dce.bryanuniversity.edu/library to redirect to  <https://sites.google.com/bryanuniversity.edu/resourcecenter/library>, the following line in the .htaccess file in my public\_html folder would be needed:  Redirect 301 /~library/book http://library.bryanuniverisyt.edu  **Submission**  Screencast & URL  Submit the URL for your "myfavorite" book redirect.  Understanding of Apache Server and HTTP status code will help to recognize the commonly encounter errors. In your screencast, explain each of these status code errors.   * Error 400? * Error 401? * Error 403? * Error 404? * Error 500? * Error 502? * Error 503?   1.5 hrs | |
| **Review Your Week 4 Support Resources**  **Type:** Header/**Content Page**/Assignment/Quiz | **Due Date:** Week 4, Day 7 | **Points:** X | **Grade Category:** X | **Requirement:** View| **Submission Type: None**/File Upload/Text Entry/External Tool |  |
| All of these resources were introduced in your various lessons this week. I’ve compiled them here for your convenience. Take the time to review and explore them; and, bookmark them! They’ll likely be beneficial in every week of this course, and throughout your program:   * Pluralsight resources: * [**Compiling and Installing**](http://httpd.apache.org/docs/2.4/install.html) * [Install Apache 2 (For Windows)](https://www3.ntu.edu.sg/home/ehchua/programming/howto/Apache_HowToInstall.html#zz-1.) * [Configuring Apache 2](https://www3.ntu.edu.sg/home/ehchua/programming/howto/Apache_HowToInstall.html#zz-2.) * [Start/Shutdown Apache 2](https://www3.ntu.edu.sg/home/ehchua/programming/howto/Apache_HowToInstall.html#zz-3.) * [If Things Go Wrong…](https://www3.ntu.edu.sg/home/ehchua/programming/howto/Apache_HowToInstall.html#zz-4.) * [Create bash function](https://linuxtechlab.com/12-bash-scripting-io-redirection/) * [Apache Installation](https://httpd.apache.org/docs/) | |

|  |  |
| --- | --- |
| Week 5: Performance and Configuration | |
| Weekly Notes/Brainstorming are  Handling Requests: Nginx vs Apache  The biggest difference between Apache and Nginx is in the underlying architecture of the way they handle requests.  Apache processes requests with MPM-s or [Multi-Processing-Modules](https://httpd.apache.org/docs/2.4/mpm.html), which is “responsible for binding to network ports on the machine, accepting requests, and dispatching children to handle the requests.” The Importance of Htaccess and AskApache Protocol and configuration. | |
| **Objectives**   * **5.1** * [Htaccess - Evolved](https://www.askapache.com/htaccess/#Htaccess_Evolved) * [AskApache Htaccess Journey](https://www.askapache.com/htaccess/#AskApache_Htaccess_Journey) * [What Is .htaccess](https://www.askapache.com/htaccess/#What_Is_htaccess)   + [Creating Htaccess Files](https://www.askapache.com/htaccess/#Creating_Htaccess_Files)   + [Htaccess Scope](https://www.askapache.com/htaccess/#Htaccess_Scope) * [Htaccess File Syntax](https://www.askapache.com/htaccess/#Htaccess_File_Syntax) * [Htaccess Directives](https://www.askapache.com/htaccess/#Htaccess_Directives) * [Main Server Config Examples](https://www.askapache.com/htaccess/#Main_Server_Config_Examples) * [Real wp-comments-post.Nginx](https://www.askapache.com/htaccess/#Real_wp-comments-post-php) * [HTTP PROTOCOL](https://www.askapache.com/htaccess/#HTTP_PROTOCOL) * [SPECIFY CHARACTERS](https://www.askapache.com/htaccess/#SPECIFY_CHARACTERS) * [BAD Content Length](https://www.askapache.com/htaccess/#BAD_Content_Length) * [BAD Content Type](https://www.askapache.com/htaccess/#BAD_Content_Type) * [Missing HTTP\_HOST](https://www.askapache.com/htaccess/#Missing_HTTP_HOST) * [Bogus Graphics Exploit](https://www.askapache.com/htaccess/#Bogus_Graphics_Exploit) * [No UserAgent, Not POST](https://www.askapache.com/htaccess/#UserAgent_POST) | |
| **Participate in the Week 5 Live Event**  **Type:** Content Page/Assignment/Quiz | **Due Date:** Week X, Day X | **Points:** X | **Grade Category:** X | **Requirement:** X | **Submission Type:** None/File Upload/Text Entry/External Tool | **Alignment**  Objectives: |
|  | |
| **Engage in Slack during Week 5**  **Type:** Header/**Content Page**/Assignment/Quiz | **Due Date:** Week 5, Day 3 | **Points:** X | **Grade Category:** X | **Requirement:** Mark Done| **Submission Type: None**/File Upload/Text Entry/External Tool | |
|  | |
| **Complete the Week 5 Learn It:** Performance and Configuration  **Due Date:** Week 5, Day 7 | **Points:** 25 | **Submission Type:** External Tool  **LTI URL from Cerego:** https://cerego.com/lti/study/970692 | |
| **Complete the Pause and Practice:**  **Type (choose):** Assignment/Quiz | **Due Date:** Week 5, Day 7 | **Points:** X | **Grade Category:** Assignments | **Submission Type (choose):** File Upload / External Tool / Text Entry | **Alignment**  Objectives: |
| Nginx & Apache Installation PPT  Apache & Nginx servers configuration PPT | |
| **Complete Assignment: Reverse Proxy 1**  **Type (choose):** Assignment/Quiz | **Due Date:** Week 5, Day 7 | **Points:** X | **Grade Category:** Assignments | **Submission Type (choose):** File Upload / External Tool / Text Entry | **Alignment**  Objectives: |
| **Purpose:** This part introduces reverse proxy web server and hosting concepts with Apache and Nginx.  **Getting Started:**  **Create a** Redirect HTTP to HTTPS in Nginx. Considering how we have set up our Nginx server, there are two ways we configure this,   * Redirect all HTTP traffic to HTTPS, * Redirect HTTP traffic for a single virtual host (website).   Before you make any changes for this, you need to make sure that you have a valid SSL certificate configured in Nginx. Follow these steps:   1. Write a Docker file to create and copy a directory. Build it using Nginx modules.   FROM NGINX:2.7-slim WORKDIR /app COPY. /app docker build –tag  (check where the docker volumes are stored)   1. Use this Dockerfile to create the reverse proxy image. It will use the nginx.conf after copying it to the proxy container:   FROM nginx: alpine  COPY nginx.conf /etc/nginx/nginx.conf  We're using Nginx 1.15.7. To check the version, we can add the following to the Dockerfile because the alpine docker image doesn't have bash installed by default:  RUN apk update && apk add bash  **Submission**  Screencast & Github 1.5 hrs | |
| **Complete Assignment: Reverse Proxy 2**  **Type (choose):** Assignment/Quiz | **Due Date:** Week 5, Day 7 | **Points:** X | **Grade Category:** Assignments | **Submission Type (choose):** File Upload / External Tool / Text Entry | **Alignment**  Objectives: |
| Now that you understand how to configure a reverse proxy server, complete this redirection.Redirecting all HTTP traffic to HTTPS Open the Nginx configuration file, (normally its /etc/nginx/nginx.conf or it can also be /etc/nginx/conf/default.conf or /etc/nginx/conf/virtual.conf).  Under the section for port 80 for Nginx configuration, edit the file with following:  **server {**  **listen 80 default\_server;**  **server\_name \_;**  **return 301 https://$host$request\_uri;**  **}** Redirecting HTTP traffic for a single virtual host (website) To make a change to a single virtual block/ website, open configuration file and make the following change to the website for which you need to redirect http to https in Nginx:  **server {**  **listen 80;**  **server\_name linuxtechlab.com www.linuxtechlab.com;**  **return 301 https://linuxtechlab.com$request\_uri;**  }  Screenshot  30 ms | |
| **Review Your Week 5 Resources** |  |
| Include any additional, supplemental resources, if applicable  Pluralsight resources:   * [nginx for Windows](http://nginx.org/en/docs/windows.html) * [How nginx processes a request](http://nginx.org/en/docs/http/request_processing.html) * [Server names](http://nginx.org/en/docs/http/server_names.html) * [Using nginx as HTTP load balancer](http://nginx.org/en/docs/http/load_balancing.html) * [Configuring HTTPS servers](http://nginx.org/en/docs/http/configuring_https_servers.html) * [How nginx processes a TCP/UDP session](http://nginx.org/en/docs/stream/stream_processing.html) * [Scripting with njs](http://nginx.org/en/docs/njs/index.html) * [Chapter “nginx” in “The Architecture of Open Source Applications”](http://www.aosabook.org/en/nginx.html) * [Controlling nginx](http://nginx.org/en/docs/control.html) * [Connection processing methods](http://nginx.org/en/docs/events.html) * [Setting up hashes](http://nginx.org/en/docs/hash.html) * [A debugging log](http://nginx.org/en/docs/debugging_log.html) * [Logging to syslog](http://nginx.org/en/docs/syslog.html) * [Configuration file measurement units](http://nginx.org/en/docs/syntax.html) * [Command-line parameters](http://nginx.org/en/docs/switches.html) * [nginx for Windows](http://nginx.org/en/docs/windows.html) * [Apache Configuration of Multiple Processor Modules(MPM)](https://httpd.apache.org/docs/2.4/mpm.html) | |

|  |  |
| --- | --- |
| Week 6: Configuring Web Services | |
| Weekly Notes/Brainstorming area | |
| **Objectives**   * **6.1** * **Nginx controller** * **AWS controller** * **Apache controller** * **Docker controller** * **Security** * **Principle of handling Errors** * **Docker balandoad performance** | |
| **Complete the Pause and Practice:**  **Type:** Content Page/Assignment/Quiz | **Due Date:** Week X, Day X | **Points:** X | **Grade Category:** X | **Requirement:** X | **Submission Type:** None/File Upload/Text Entry/External Tool | **Alignment**  Objectives: |
|  | |
| **Complete the Pause and Practice:**  **Type:** Content Page/Assignment/Quiz | **Due Date:** Week X, Day X | **Points:** X | **Grade Category:** X | **Requirement:** X | **Submission Type:** None/File Upload/Text Entry/External Tool | **Alignment**  Objectives: |
| Time on Task: | |
| **Engage in Slack during Week 6** | |
| Add suggested discussion topics/prompts here, if applicable | |
| **Complete the Week 6 Learn It:** Configuring Web Services  **Due Date:** Week 6, Day 6 | **Points:** 20 | **Submission Type:** External Tool  **LTI URL from Cerego:** https://cerego.com/lti/study/970693 | |
| **Complete Project 3: Let’s Encrypt**  **Type (choose):** Assignment/Quiz | **Due Date:** Week 6, Day 7 | **Points:** X | **Grade Category:** Assignments | **Submission Type (choose):** File Upload / External Tool / Text Entry | **Alignment**  Objectives: |
| **Purpose:**  In this project, you will configure TLS/SSL certificates for both the domains hosted on Apache. We’ll obtain the certificates through [Let’s Encrypt] ([https://letsencrypt.org](https://letsencrypt.org/)]. Nginx supports SSL termination so we can set up SSL without modifying Apache’s configuration files.  **Getting Started:**  **Setting Up HTTPS Websites with Let’s Encrypt**   1. Create files in Nginx and Apache folders 2. Open the Nginx or Apache file. Change website server with BryanUniversity.edu intranet and internet website. 3. Search BryanUniversity DNS and IP address. Change the configuration in the listing server below.   /etc/nginx/sites-available/apache  server {  listen 80;  server\_name foobar.net www.foobar.net;  location / {  proxy\_pass\_http://your\_server\_ip:8080;  proxy\_set\_header Host $host;  proxy\_set\_header X-Real-IP $remote\_addr;  proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  proxy\_set\_header X-Forwarded-Proto $scheme;  }  }  server {  listen 80;  server\_name test.io www.test.io;  location / {  proxy\_pass http://your\_server\_ip:8080;  proxy\_set\_header Host $host;  proxy\_set\_header X-Real-IP $remote\_addr;  proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  proxy\_set\_header X-Forwarded-Proto $scheme;  }  }   1. Save the file and perform a configuration test:   $ sudo nginx -save  Submission  Screencast  Time on Task: 2h | |
| **Review Your Week 6 Resources**  Apache HTTP Server provides very comprehensive and flexible logging capabilities. This document describes how to configure its logging capabilities, and how to understand what the logs contain.   * [Security Warning](https://httpd.apache.org/docs/1.3/logs.html#security) * [Error Log](https://httpd.apache.org/docs/1.3/logs.html#errorlog) * [Access Log](https://httpd.apache.org/docs/1.3/logs.html#accesslog)   + [Common Log Format](https://httpd.apache.org/docs/1.3/logs.html#common)   + [Combined Log Format](https://httpd.apache.org/docs/1.3/logs.html#combined)   + [Multiple Access Logs](https://httpd.apache.org/docs/1.3/logs.html#multiple)   + [Conditional Logging](https://httpd.apache.org/docs/1.3/logs.html#conditional) * [Log Rotation](https://httpd.apache.org/docs/1.3/logs.html#rotation) * [Piped Logs](https://httpd.apache.org/docs/1.3/logs.html#piped) * [Virtual Hosts](https://httpd.apache.org/docs/1.3/logs.html#virtualhosts) * [Other Log Files](https://httpd.apache.org/docs/1.3/logs.html#other)   + [PID File](https://httpd.apache.org/docs/1.3/logs.html#pidfile)   + [Script Log](https://httpd.apache.org/docs/1.3/logs.html#scriptlog)   + [Rewrite Log](https://httpd.apache.org/docs/1.3/logs.html#rewritelog) | |

|  |  |
| --- | --- |
| Week 7: Capstone Project | |
| Weekly Notes/Brainstorming area | |
| **Objectives**   * **7.1** * **7.2** * **7.3** | |
| **Participate in the Week 7 Live Event**  **Type:** Content Page/Assignment/Quiz | **Due Date:** Week X, Day X | **Points:** X | **Grade Category:** X | **Requirement:** X | **Submission Type:** None/File Upload/Text Entry/External Tool | **Alignment**  Objectives: |
| Provide guidance on what you think should be covered during this last class: | |
| **Engage in Slack during Week 7** | |
| Reflect on lessons you learned in your Capstone Project.   * What worked well? * What parts of the project are you particularly pleased with, and why? * What parts of the project need additional work? * If you were to approach this project again, what would you do the same? What would you do differently? | |
| **Complete the Learn It Recharge** | |
| **Complete the Capstone Project:**  **Type (choose):** Assignment/Quiz | **Due Date:** Week 7, Day 7 | **Points:** X | **Grade Category:** Assignments | **Submission Type (choose):** File Upload / External Tool / Text Entry | **Alignment**  Objectives: |
| **Final Project Report**  The report can be written or be a screencast. If written, about 3-4 pages (including pictures); if screencast, about 5 minutes. Please cover each of the 5 simple planes of user experience of NGINX DOCKER AND APACHE WEBSER.   1. Visit Bryan library web site: <https://sites.google.com/bryanuniversity.edu/resourcecenter/library> 2. Describe the implementation details:  * Name of the site * **Strategy.** A brief statement about the purpose and goals of the site. * **Define the audiences** (primary and secondary) of the site and briefly describe them. (1 to 2 sentences for each audience) Marketing hosting product or support services * **Scope and Structure.** What features are critical for the initial site (a list is all you need here) and how do they relate to NGINX AND APACHE SERVER   Here, I'm more interested in seeing this represented not in a "pretty" form, but in a rough form that you did while working out ideas for the site. Including a picture from a quick sketch on paper or whiteboard is enough here -- no need to spend additional time making your scope and structure neat and tidy. The idea here is that this section should be present, but can be "rough" -- convince us you went through this step!   1. **Skeleton.**  * Provide one skeleton (possibly two if you developed alternatives). Again, include a picture from a quick sketch on paper or whiteboard to show the importance of Docker and Container Images. The idea here is that this section should be present, but can be "rough" -- convince us you went through this step! * Include something in the container ID, such as “Hello World.”  1. **Surface.**  * Your finished project will show the "surface" of the site, or a single page with some content. * Include links to the three pages of your site where you implemented the transfer protocol of NGINX or APACHE Hosting server. * If there are things you want to mention about the "surface," do so here, but nothing is expected other than the links!  1. Implementation and Future Directions.  * Briefly discuss the implementation of the site. Discuss NGINX, APACHE, Docker as an Ecosystem. For example: * How did you go about building or create a Docker file? Did you use any VM, Hosting frameworks or microservices libraries? How did you organize the underlying resources needed for the site? What is the "PROTOCOL" component of the site? * Future Directions. Briefly discuss the future direction of the site. What ideas do you have for future work and/or expansion?   Time - 4 hrs  Time on Task:  **Rubric Example**   |  |  |  |  | | --- | --- | --- | --- | | **CRITERION** | **Does not Meet Expectations (24)** | **Needs Improvement (86)** | **Meets Expectations** | | Website has at least 3 fully responsive/designed pages  (30 possible) | Website has fewer than 3 pages, and not all pages are fully responsive or designed  (5 points) | Website has at least 3 pages, but not all pages are fully responsive or designed  (20 points) | Website has at least 3 fully responsive/designed pages  (30 points) | | Website includes the 4 basic sections: navbar, header, footer and main section  (10 possible) | Website only includes 2 of the 4 basic sections.  (2) | Website includes at least 3 of the 4 basic sections.  (7) | Website includes the 4 basic sections: navbar, header, footer and main section.  (10) | | Website includes icons and external fonts  (10 possible) | Website does not incorporate icons. More than 2 fonts are used; Site is too busy.  (2) | Website includes icons and external fonts. More than 2 fonts are used; Site is too busy.  (7) | Website includes icons and external fonts appropriately. No more than 2 fonts used; Site is not too busy.  (10) | | Website includes images, list(s), and table(s)  (20 possible) | Website does not include all elements: images, list(s), or table(s). And/or, elements included are not appropriate (e.g. images are out of place, element used is wrong for the type of content).  (5) | Website includes images, list(s), and table(s), but are not appropriate (e.g. images are out of place, element used is wrong for the type of content). And/or, not all elements are included.  (15) | Website includes images, list(s), and table(s) appropriately.  (20) | | Website includes at least one meaningful transition (e.g. hovering)  (15 possible) | Website does not include a transition.  (3) | Website includes at least one transition, but it’s not meaningful.  (10) | Website includes at least one meaningful transition (e.g. hovering).  (15) | | CSS box model and style sheet is applied appropriately  (10 possible) | CSS box model or style sheet is not applied.  (2) | CSS box model and style sheet is applied, but not appropriately.  (7) | CSS box model and style sheet is applied appropriately.  (10) | | Website is coded appropriately, to include inline and block elements, divs, spans, semantic HTML, CSS box model requirements, appropriate responsive code (e.g. media queries, flexbox, CSS grid)  (30pts possible) | Website is not coded appropriately, and/or doesn’t include all the requirements: inline and block elements, divs, spans, semantic HTML, CSS box model requirements, and responsive code (e.g. media queries, flexbox, CSS grid).  (5) | Website code includes inline and block elements, divs, spans, semantic HTML, CSS box model requirements, and responsive code (e.g. media queries, flexbox, CSS grid), but most, not all is coded appropriately.  (20) | Website is coded appropriately, to include inline and block elements, divs, spans, semantic HTML, CSS box model requirements, appropriate responsive code (e.g. media queries, flexbox, CSS grid).  (30) | | |

|  |  |
| --- | --- |
| Week 8: Reflection | |
| Weekly Notes/Brainstorming area | |
| **Objectives**   * **8.1** * **8.2** * **8.3** | |
| **Complete the Reflection Assignment**  **Type (choose):** Assignment/Quiz | **Due Date:** Week 7, Day 7 | **Points:** X | **Grade Category:** Assignments | **Submission Type (choose):** File Upload / External Tool / Text Entry | **Alignment**  Objectives: |
| Reflection Instructions  Reflect on the skills you gained these past 8 weeks, as well as any other important takeaways you have from this class and how they apply to your future goals.  Respond to the following:   * In this course you learned how to create a Docker server and an NGINX hosting system with Apache server. What was the impact of this course on your future career goals? * What was difficult or surprising to you about NGINX, Apache and Docker, and why? * Did this course inspire you to want to continue using Docker? Why or why not?   Complete your reflection in any format you choose. Here are some options:  A website summary (a mix of text and images): Push your code to Github, then copy/paste your URL here to submit it.  A typed summary (1-2 paragraphs)  A video summary (1-3 minutes): Record yourself as you reflect. You can use any recording device that is convenient for you. I recommend the Canvas Media Recording Tool. It is the easiest and most convenient tool to use because you record directly from this assignment page. Click Here for How-To instructions.  Studio's Video Recording Tool is mobile friendly. Click to access the Video Tutorial  and/or the Quick Reference Guide  A comic strip or picture collage summary (5-10 graphics)  Upload your assignment by the due date. | |
| **Resubmit for Mastery** | |
|  |  |
| **Complete the End-of-Class Survey** | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Course Development** | **Due Date** | **Completed** | **Notes/Comments** |
| Draft Weekly Topics and Learning Objectives |  |  |  |
|  |  |  |  |
| **Milestone 1** | | |  |
| Learn Its Wks 1-2 |  |  |  |
| Learn Its Wks 3-4 |  |  |  |
| Learn Its Wks 5-6 |  |  |  |
|  |  |  |  |
| **Milestone 2** | | |  |
| Video Lectures/Content Outline Wks 1 & 2 |  |  |  |
| * Practice Activities * Independent Project 1 instructions and answer guidance |  |  |  |
| Wk 1 & 2 lessons and screencasts |  |  |  |
| Video Lectures & Practice Activity Ideas Wk 3-4 |  |  |  |
| * Independent Project 2 instructions and answer guidance |  |  |  |
| Wk 3-4 gaps and Practice Activity student instructions |  |  |  |
| Video Lectures & Practice Activity Ideas Wk 5-6 |  |  |  |
| * Independent Project 3 instructions and answer guidance |  |  |  |
| Wk 5-6 gaps and Practice Activity student instructions |  |  |  |
| Week 7: Capstone Project |  |  |  |
| * Instructions and answer guidance |  |  |  |
| Wk 8 reflection and final gap analysis |  |  |  |
| **Milestone 3** | | |  |
| Build course in Canvas |  |  |  |
| Final QC & build of Learn Its in Cerego |  |  |  |
| Final Drafts |  |  |  |
|  |  |  |  |
| **Milestone 4** | | |  |
| Internal QC |  |  |  |
| Publish |  |  |  |
| Faculty Course Setup |  |  |  |
| Student Access |  |  |  |
| Go Live |  |  |  |