**Docker Assignment**

**Assignment -1**

Containerize a Python Flask Application

1. Create a Python Flask application:

* Create a new directory for your project.
* Inside the directory, create a new file named app.py.
* Write a simple Python Flask application that returns "Hello, Docker!" when accessed at the root URL ("/").
* Make sure to include the necessary Flask imports and define a Flask app.

2. Create a Dockerfile:

* Inside the project directory, create a new file named Dockerfile.

3. Create a requirements.txt file:

* In the project directory, create a file named requirements.txt.
* Add any necessary dependencies for your Flask application. For example, you can add Flask to the requirements.txt file.

4. Build the Docker image:

* Open a terminal or command prompt and navigate to the project directory.

5. Run the Docker container:

* Once the image is built, you can run a container based on that image.

6. Test the Flask application:

* Open a web browser and access http://localhost:5000.
* You should see the message "Hello, Docker!" displayed in the browser.

**Assignment -2**

The assignment is to containerize and run a sample application in multiple formats.

The sample app that is to be used for this is **Webanalyze.** ([webanalyze](https://apc01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fgithub.com%2Frverton%2Fwebanalyze&data=05%7C01%7C%7C32ef11afe4f944d1d26508db3a6ccfe4%7C7742820587ff4048a64591b337240228%7C0%7C0%7C638168011240620495%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C3000%7C%7C%7C&sdata=1Oo2S%2BpHTF51%2BDavHmEK6ItfL5%2Fc6NoXy5kCeeNKrbs%3D&reserved=0))

* Write a Dockerfile to containerize this application/software.
* Run the resulting docker image.
* Capture the output/log of the docker container into your host file system. (Persistence)
* Have multiple instances (minimum 5) of the docker container run for different websites parallely. (Docker Compose)
* Store the application logs for multiple instances.
* Use the product page websites of Meta, Google, Amazon, Apple, and Netflix as inputs for your 5 instances.
* For the standalone container, use the Happiest Minds website as input.
* Once all the above is completed, deploy the same in a docker swarm stack.

**Assignment-3**

* Pick a project you’re currently working on or have So let’s say you have an application that’s a react web app. It uses nodes, yarn, etc. And all of it is on your local machine.
* Create a Docker image that can act as your development environment as opposed to using dependencies on your local machine So in this case, you’d create a Dockerfile.
* Use the development image, The goal here is to use the Node version inside of the Docker file/container instead of using the ones on your local machine.

**YOUR TITLE HERE**

**Assignment-4**

Multi-Container Application with Docker Compose:

* Create a multi-container application using Docker Compose.
* Define the services, networks, and volumes in a docker-compose.yml file.
* Include different components such as a web server, database, and caching layer in separate containers.
* Configure the intercommunication between containers and run the application.

**Assignment-3**

Containerized WordPress:

* Containerize the WordPress content management system using Docker.
* Use separate containers for the WordPress application and the database (e.g., MySQL or MariaDB).
* Mount volumes to persistently store WordPress files and the database data.
* Configure and run the containers to host a functional WordPress site.

1.) CICD Pipeline To Deploy To Kubernetes Cluster Using Jenkins: <https://youtu.be/XE_mAhxZpwU>

2.) Deploying Java Applications with Docker and Kubernetes: <https://youtu.be/0GgBi8yNQT4>

3.) Deploying Node.js Application In Kubernetes : <https://youtu.be/CfPRbdT-wXo>

4.) Continuous Deployment of Node JS app with GitLab CI : <https://youtu.be/j1cj5oK8t6c>

5.) Build and Push Docker Image to GitLab Container Registry : <https://youtu.be/AR29V1wWjjk>