Course Project – Option 1

Deadline: April 4th, 11:59pm ET/8:59pm PT

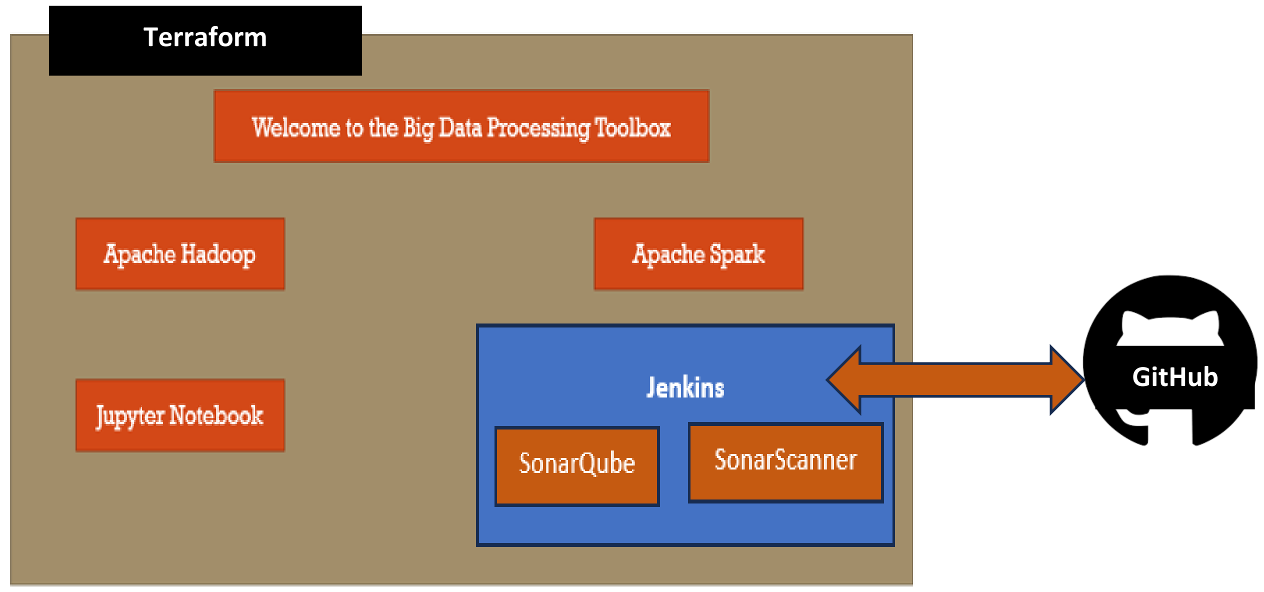
Accept this project by accessing GitHub classroom via the following URL:
https://classroom.github.com/a/ChlFe71N

You are required to submit your GitHub Repository URL on Canvas. A penalty will be applied if you don't do so. If you are working in a team, one submission is sufficient.

Your ReadMe file should include the exact steps that can be used to reproduce a functional version of your solution.

You may choose one peer to work with you on the project. You MUST list your peer's name on Canvas Or at the top of your ReadMe file. Failure to list your peer's name will lead to grading penalty of 10 points to both team members.

BIG DATA PROCESSING TOOLBOX

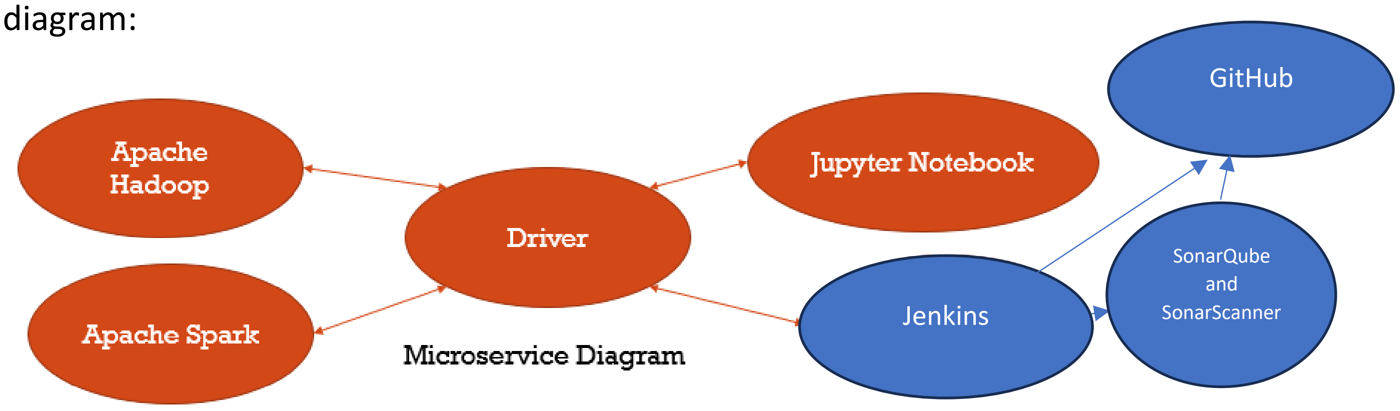


General Description:

You are requested to build a microservice-based application that would allow the users to run Apache Hadoop, Spark, Jupyter Notebooks, SonarQube and SonarScanner without having to install any of them.

Your application is supposed to have one main microservice that acts as the entry point for your overall application.

The expected architecture for your application would look like the following diagram:



- Each circle that is shown in the architecture diagram represents a **microservice that is hosted on a separate docker container** (<u>You may have SonarQube and SonarScanner on the same container or connected to Jenkins via its plugin</u>)
- **ALL** your docker containers should be deployed to **Google Kubernetes Engine.**
- **Your GKE should be coded using Terraform.**
- **Jenkins should be connected to SonarQube and you should be able to kick-off a scan on Jenkins to conduct SonarQube analysis on a public GitHub repository via Jenkins.**

<u>Example Workflow:</u>

Check the Demo video published on Canvas. Please note that the demo doesn't include the full extent of the Jenkins demo and it's up to you to figure out the remaining details.

<u>Important Notes:</u>

- You may reuse any docker images/containers built by others.
- Make sure that your entire application gets installed/run and prepared by the time you demand the user to enter one of the four options.

- No installations outside of Dockerfile (or Kubernetes cluster script) are expected to happen.
- Your application should be runnable without any custom/manual steps outside of launching the Kubernetes cluster along with Docker.
- You SHOULD NOT have any environment variables or configurations that are set manually by the users. However, you can set all environment variables in your Dockerfile and/or Kubernetes cluster scripts.
- For Apache Hadoop, make sure to create **one master node and two worker nodes.**
- Use ReadMe.md file on your repository to list any assumptions, steps and any important information to share.
- Keep any private keys for your GCP account outside of the code on GitHub. Instead, submit them on Canvas along with your repository URL.

Submission Guidelines:

- Post URL for your GitHub Classroom repository to Canvas along with your Peer’s name. Alternatively, you can note your peer’s name at the top of your Readme file.
- Your GitHub repository should have a ReadMe.md file that lists the “exact” steps on how to get this application to work.
- You should record a video demonstrating two elements:
    1. Code Walkthrough while you are explaining your code changes.
    2. Demoing the running application while you are navigating through EVERY functionality that is working in your application. I will use this video to help assessing your grade.
    3. Note: "The demonstration video is crucial for grading. Any functionality not demonstrated in the video may result in a deduction of points."

- Your video size may be large to be uploaded to GitHub. You may use Box to upload the video and add the URL to your ReadMe.md file in your GitHub repository.
    1. Make sure that your video is publicly shared. Private videos won't be visible to the instructor and TAs and therefore, your project grade will be impacted.

Grading Criteria:

- Getting all microservice applications "containerized", have them communicating with the main service, and display expected output: 55% of the total project grade.
- Run SonarScanner on a dummy project and have the results displayed on SonarQube server: 10% of the total grade
- Deploying all containers to Kubernetes Cluster: 10% of the total project grade
- Deploying Kubernetes Cluster to Google Cloud Platform using Terraform: 15% of the total project grade.
- **Submitting Your First Checkpoint Tasks by March 1st, 11:59PM ET/8:59PM PT. (10% of the total project grade)**
    - **You are supposed to submit what is expected to be done by end of week-6 to your GitHub Repository**

Suggested Project Task Schedule (You may run ahead of schedule):

Week	Task
End of Week-6	Build the main web application. Create all docker images and containers
End of Week-8	Create Kubernetes Cluster and Configure it for the docker containers you Created
End of Week-10	1. Write Terraform script to deploy your Kubernetes Cluster to Google Cloud Platform and work out any issues. 2. Record a video for the running version of the application with Code Walkthrough.
End of Week-11	1. Finish your ReadMe.md file to list all your steps, assumptions, and any information you find important to share.

Grading Notes:

- Unlike HW-assignments that has 20% late penalty, late project submissions on Canvas or GitHub will receive 0 points (won't be graded).
- **Not submitting the GitHub video (<u>for both code walkthrough and functionality demo</u>): you will get up to 80% of the maximum grade.**
- Not providing clear details in the ReadMe file on how to run the application (or any variables that need to be updated/replaced): **you will get up to 90% of the maximum grade.**