ISEC6000
Secure DevOps

Curtin University

Assignment 1

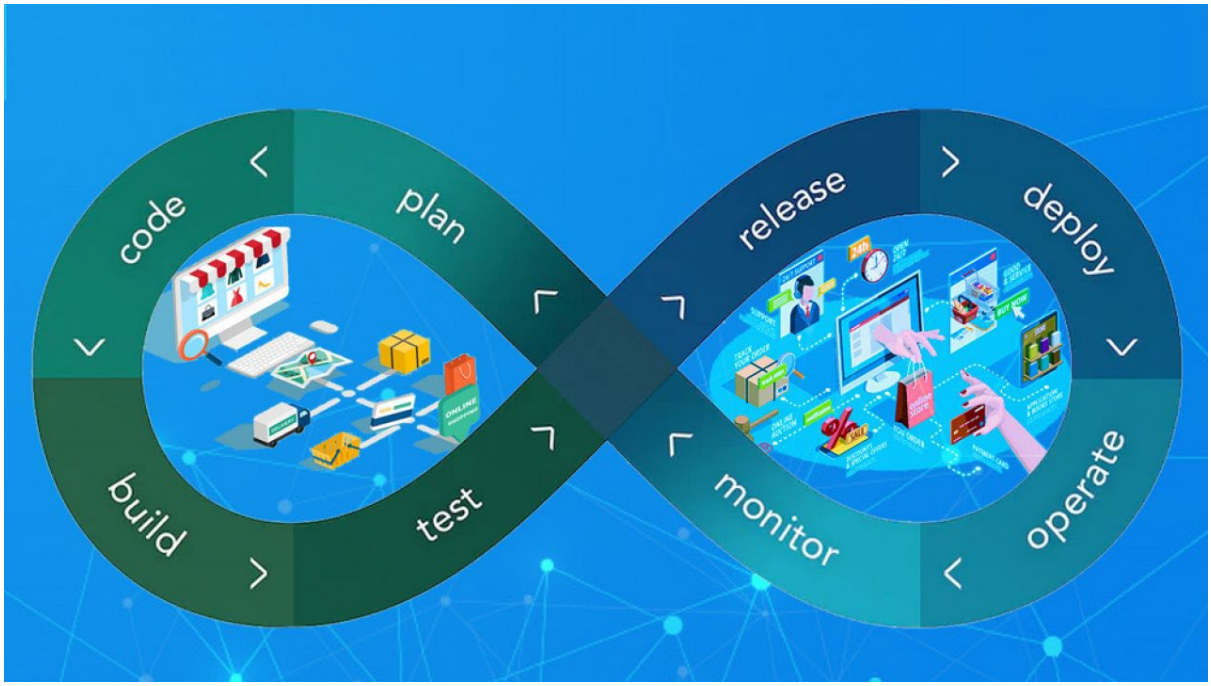
Building and Securing a Microservices E-commerce Application

Weighting Percentage: 25% of the unit

Submission: Please follow the descriptions as given in this document

Deadline: 5 Sept 2023 11.59PM

Late Submission: late submission attracts 5% raw penalty per day up to 5 days (i.e., 10 Sept 2023 11:59PM)



1. Outline

Imagine you are part of a cutting-edge technology team responsible for designing, deploying, and securing a modern e-commerce application. This application will revolutionize online shopping by incorporating the latest advancements in technology, security, and DevOps practices. Your journey begins by embracing the knowledge gained in Weeks 1 to 5 and embarking on a hands-on experience that will shape your expertise in Docker containerization, Kubernetes orchestration, integrating security, and mastering the Google Kubernetes Engine (GKE) environment.

During this project, you will also skilfully construct, configure, deploy, and uphold a comprehensive Saleor application suite on a Linux server, meticulously employing Docker Compose. Saleor, an open-source e-commerce platform, entails a Python API backend, a React (Javascript) frontend with a dashboard, a PostgreSQL database backend supplemented by an in-memory Redis cache, and other indispensable supporting services. Drawing upon the knowledge assimilated thus far in this Unit, you will navigate and surmount the intricate architectural intricacies of Saleor, unveiling your prowess in a pragmatic and tangible manner.

Fig. 1. Microservice platform

2. Project Tasks

Task 1: Set Up Initial Infrastructure [20]

1. Create a Kubernetes Cluster on GKE (or equivalent tool)
    a. Log in to your Google Cloud Console.
    b. Navigate to the Kubernetes Engine section and click "Create Cluster."

c. Configure your cluster settings, such as the cluster name, location, and node pool configuration.
d. Choose the desired Kubernetes version.
e. Click "Create" to provision your GKE cluster.

2. Install and configure kubectl to manage your Kubernetes cluster.
    a. After creating the GKE cluster, you will need to configure your local environment to use kubectl to interact with this cluster.
    b. In the Google Cloud Console, navigate to the "Kubernetes Engine" > "Clusters" section and click the "Connect" button next to your cluster.
    c. Follow the instructions to authenticate kubectl with your GKE cluster.

3. Set up a private GitHub repository to store your project files.
    a. Log in to your GitHub account (or create one if you don't have it).
    b. Click on the '+' icon at the top right corner of the GitHub dashboard and select "New repository."
    c. Choose a meaningful repository name for your ISEC6000 Secure DevOps project.
    d. Select "Public" for the repository visibility.
    e. You need to add a description and choose whether to initialize the repository with a README. (Bonus marks if you have a proper README file.
    f. Click "Create repository."
    g. Push the initial setup code to the repository.

Task 2: Microservices Architecture and Deployment [50]

1. Begin by acquainting yourself with the core projects and their purposes:
    a. **Saleor API**: Explore the functionalities at https://github.com/saleor/saleor .
    b. **Saleor storefront**: Understand the frontend mechanics at https://github.com/saleor/react-storefront .
    c. **Saleor dashboard**: Dive into the dashboard intricacies at https://github.com/saleor/saleor-dashboard .
    d. **Saleor platform**: Access the repository at https://github.com/saleor/saleor-platform , which contains essential Docker Compose elements for configuring, building, and executing Saleor components. Note that this repository references the three aforementioned repos using Git submodules.
2. Create a personal account on Github.com and proceed to fork the Saleor platform repository.
3. Follow the step-by-step guidelines outlined in the Saleor platform repository to effectively run a Saleor stack enriched with sample data.
4. Tailor the Compose file to ensure optimal functionality:
    a. Configure the React Storefront to operate on port 3009.
    b. Assign port 9003 for the Saleor Dashboard.
    c. Initiate the stack and verify the successful launch of all services:
        - Saleor React Storefront: Accessible at http://<Your-Linux-Server-IP>:3009.

- Saleor Dashboard: Reachable via http://<Your-Linux-Server-IP>:9003.

5. Commit your modifications and push them to the forked repository, appending the tag isec6000-assignment1.

Task 3: Implementing Security Measures [10]

1. Container Security:
   a. Ensure secure configuration of containers.
   b. Implement container image vulnerability scanning using tools like Trivy.

Task 4: Architecture Visualization [20]

1. Present an architecture diagram that encapsulates the essential functionalities of each Saleor service, accompanied by their interactions within the stack. You have the creative freedom to choose diagram style and tools that best convey your design, with the understanding that clarity and conciseness are pivotal. The architecture diagram should encompass the following aspects:
   - Illustrate the volumes and networks utilized by different Saleor services.
   - Depict how the different Saleor services communicate with each other.
   - Highlight the exposed ports associated with each container.

3. Submission

You need to submit a pdf document containing the tasks as mentioned below. The documentation should be logically organized, providing clear instructions for each pivotal task. The pdf file name should be in this format: **P1_StudentID**.For example, if your student ID is 123456, then the filename should be **P1_123456.pdf**. (Note: Fail to these instructions will result in a **penalty of 50%**).

A. For **Task 1** and **Task 2**: Link to your forked repo -

- The repo must be public so that your work can be reviewed and marked.
- Make two Github repos. The two repos must contain the tag `isec6000-assignment1-task1` and `isec6000-assignment1-task2` (respectively to Task 1 and Task 2).
- It must be possible to run a Saleor stack following the Readme instructions that is present in your github repo.

B. For **Task 3** and **Task 4**: Include the related contents in the assignment 1 pdf file.

4. Bonus Marks

a. Bonus marks if you have a proper README file.

b. Clear and comprehensive documentation enhancing the overall user experience and understanding of your project.