AIDE 501 DevOps 26/27

Assignment 1

Cloud Computing and Containers

Assignment objectives

The objective of this assignment is to develop, containerize, publish, and deploy a web application through the entire containerization and cloud deployment lifecycle.

1. Develop a meaningful web application using React.js for the frontend and Node.js with Express.js for the backend and a database under Windows OS.

- At least three functional pages or views.
- Navigation between pages.
- A responsive user interface.
- Some interactive functionality.
- A meaningful application title and content.
- A backend and a database.
- Execute the application locally.
- Test the application using a web browser.
- Verify that all pages and interactive elements work correctly.

2. Docker and containers.

- Create a Docker image using a Dockerfile.
- Run and test a Docker container.
- Access the website through a browser.
- Create a docker-compose.yml file to manage the application.
- The application must contain at least three services:
    - Frontend container
    - Backend container
    - Database container

3. Publish Docker images.

- Students must publish the same application image to THREE different container registries:
  - Docker Hub
    - Create an account at Docker Hub.
    - Authenticate, tag, push.
  - GitHub Container Registry
    - Create a GitHub account.
    - Generate a Personal Access Token (classic) with the required package permissions.
    - Authenticate, tag, push.
  - GitLab Container Registry
    - Create an account at GitLab.
    - Create a GitLab project and configure its container registry.
    - Authenticate using an appropriate access token, tag, push.
- Students must create an account on each platform and configure an image repository.
- Students must configure their repositories as private and provide appropriate authentication and authorization.

4. Install and configure a Linux virtual machine using VirtualBox.

- Run the same Docker image under a Linux operating system installed on the same physical laptop.
- Install Docker Engine inside Linux OS and verify Docker installation.
- Pull the image from each of the three registries and run it under Linux OS.
- Deploy and execute Docker containers under Linux.

5. Create and configure an AWS account to deploy the React application on an Amazon EC2 virtual machine in the AWS cloud.

- Students must check their account's EC2 eligibility and estimated costs before launching resources.
- Launch an EC2 instance
  - Select an Ubuntu Server AMI.
  - Choose an eligible instance type.
  - Configure instance storage.
  - Create or select a key pair.
  - Configure the security group.

    - Launch the instance.
    - Obtain the public IPv4 address.
    - Security group requirements:
        - Protocol Port Purpose
        - SSH 22 Remote administration
        - HTTP 80 Access to the website
- Connect to EC2 From Windows PowerShell using SSH.
- Install Docker on EC2 Inside the EC2 Ubuntu instance.
- Retrieve Docker images from different registries directly from the EC2 instance.
    - If an image is private, students must authenticate first.
- Students must run the image pulled from each registry.
- They must then access the application from their Windows browser:
    - http://EC2_PUBLIC_IP
- Students must repeat the deployment using images from GHCR and GitLab. To avoid port conflicts, they can stop the previous container before running the next one, or use different host ports and corresponding security group rules.
- Important AWS resource cleanup at the end of the assignment:
    - Stop and terminate their EC2 instance.
    - Check whether any EBS volumes remain and delete unused volumes.
    - Release any allocated Elastic IP addresses.
    - Verify that no other billable resources remain.
    - Check AWS billing and Free Tier usage.

6. demonstrate the usage of API.

7. Security considerations.

8. Bonus work – DevOps challenges.

- Configure GitHub Actions to automatically build and publish the Docker image to GHCR whenever new code is pushed to GitHub.
- Scan images before publication.
- Publish the same image to Amazon ECR and deploy it on EC2.
- Container management:
    - Container restart policies.
    - Container logs.
    - Container resource monitoring.
    - Image versioning.
    - Container removal and recreation.