Subject: Cloud-2 DevSecOps Infrastructure

You are now ready to take a step forward and combine everything you have learned about cloud infrastructure, Kubernetes, CI/CD, and security practices into a single project.

This time, you will build a full DevSecOps pipeline on a managed Kubernetes cluster in the cloud. Forget about local k3s or Vagrant — everything must run in the cloud.

Your mission: provision a Kubernetes cluster with Terraform, configure it for your application, and integrate GitLab CI/CD pipelines with container registry and security tools.

Part 1: Provision the Cloud

- Using Terraform, create a managed Kubernetes cluster on cloud.
- Save the kubeconfig locally so you can manage the cluster from your host machine.
- Ensure that your Terraform scripts can:
  - create the cluster

Part 2: Base Configuration

- Install an ingress controller (nginx-ingress) with Helm to obtain an external IP.
- Create a own namespace for application.
- Deploy your application into this namespace using your own Helm chart.

Requirements:

- The application must include a database, configured with a PersistentVolumeClaim (PVC).
- Use StorageClass to bind persistent volumes for the database.

Part 3: GitLab CI/CD

- Import your project into GitLab.
- Set up GitLab Runner inside the Kubernetes cluster (via Helm).
- Configure the pipeline so that:
  - It builds your application,
  - Pushes the container image to GitLab Container Registry (using predefined variables).

No manual setup inside GitLab Registry should be required. The pipeline push must create and upload the image automatically.

Part 4: Deployment Automation with GitLab Agent

- Create a GitLab Agent for Kubernetes.
- Register it with your cluster and install it via Helm.
- Configure your GitLab pipeline so that after a successful build, your image is automatically deployed into the cluster.
- Store GitLab credentials as a Kubernetes secret inside the cluster.

Part 5: DevSecOps Practices

Your pipeline must also include security scans:

- Add DAST scan in a namespace.
- Configure credentials for GitLab Registry in the namespace as well.
- Deploy DefectDojo to provide a vulnerability management dashboard.
- Connect your security test results to DefectDojo.

Final Deliverables

By the end of this project, you must be able to:

1. Deploy the application with Helm into the cluster.
2. Run a GitLab CI/CD pipeline that:
    - builds your app,
    - pushes it to GitLab Container Registry,
    - deploys it automatically to Kubernetes via GitLab Agent,
    - performs DAST scan,
    - reports vulnerabilities to DefectDojo.

Bonus

- Enable GitLab SAST, SCA, Secret Detection scans.