Assignment: Build, Deploy, and Automate a Kubernetes Application with GitHub Actions and Argo CD

Objective:

In this assignment, you will:

1. Create a sample application.
2. Set up CI/CD pipelines for building, deploying, and automatically updating the application in a Kubernetes cluster.

Instructions:

1. Create a GitHub Repository for the Sample Application

- Choose a language you're comfortable with (Node.js, Python, Go, etc.).
- Write a simple application in this language, for example, an HTTP server that responds with "Hello, World!".
- Push your code to a **public GitHub repository** named sample-app.

2. Write a Dockerfile for the Application

- Create a Dockerfile at the root of your sample-app repository.
- The Dockerfile should:
  - Copy the necessary application files.
  - Install any dependencies.
  - Expose the required port.
  - Start the application when the container runs.
- Push the Dockerfile to the repository.

3. Set Up GitHub Actions to Build and Push the Docker Image

- Create a **GitHub Action** workflow file (.github/workflows/docker-publish.yml) in the sample-app repository.
- The workflow should:
  - Trigger on each push to the main branch.
  - Build the Docker image.
  - Log in to Docker Hub (store Docker credentials in GitHub Secrets).
  - Push the Docker image to Docker Hub with the latest tag.

4. Create an Infrastructure Repository for the Helm Chart

- Create a new GitHub repository named `sample-app-infra`.
- Write a **Helm chart** for your application, specifying:
  - Deployment, Service, and other Kubernetes resources.
  - Docker image configuration to pull from Docker Hub.
- Push the Helm chart to this `sample-app-infra` repository.

5. Install Argo CD on Your Local Kubernetes Cluster

- Set up a **local Kubernetes cluster** (e.g., using Minikube, Kind, or k3s).
- Install Argo CD in this cluster.
- Expose Argo CD's UI and log in to configure and manage deployments.

6. Connect Argo CD to Your Infrastructure Repository

- In the Argo CD UI or CLI:
  - Connect to your `sample-app-infra` repository.
  - Configure it to automatically deploy your Helm chart in the local Kubernetes cluster.
- Verify that Argo CD successfully pulls the Helm chart and deploys the application.

7. Update GitHub Actions to Deploy on Kubernetes via Argo CD

- Modify the **GitHub Actions** workflow in `sample-app` to trigger a deployment update:
  - After pushing a new Docker image to Docker Hub, trigger Argo CD to update the application in the Kubernetes cluster.

Deliverables:

- GitHub repositories for `sample-app` and `sample-app-infra`.
- Working Dockerfile and Helm chart in each respective repository.
- GitHub Actions workflow files for building, pushing, and triggering deployments.
- A running instance of your application in the local Kubernetes cluster, managed by Argo CD.