**Step-by-Step Guide for Setup**

This should walk through a Google Cloud CI/CD pipeline that builds the Maven project, pushes a Docker image to Artifact Registry, and deploys to Google Kubernetes Engine (GKE) using the scripts in the zip file attached. The scripts are all generalized because I could not customize them to git, but most placeholder spots, can be filled in with DFS specific names (i.e. for app titles), or for git repo links.

**1. Setup**

* Google Cloud Project established
* Google Kubernetes Engine (GKE) Cluster set up in your project.
* Artifact Registry repository for Docker images.
* Cloud Build API enabled.
* kubectl, gcloud, and git installed locally (for initial setup).
  + Links to install:
    - Kubectl: <https://kubernetes.io/docs/tasks/tools/>
    - Gcloud: <https://cloud.google.com/sdk/docs/install>
    - Git: <https://git-scm.com/downloads>
* SSH set up in GitHub, with an SSH-Key.
* The zip folder with all the build files attached.

**2. Prepare Your Kubernetes Deployment Manifest**

1. Customize deployment.yaml to our DFS repo. Adjust variables to DFS, IMAGE\_PLACEHOLDER should be the Docker image that connects to the codebase through jib, after it is packaged into the container image.

**YAML**

**3. Prepare Your Cloud Build Pipeline**

1. Create cloudbuild.yaml in your repository root.

**4. Authenticate kubectl and deploy to GKE**

1. deploy.sh created for manual deployment in the event CI/CD automated deployment does not work.

**5. Configure IAM Permissions**

* Ensure the Cloud Build service account has this role:
  + Artifact Registry Writer

**6. Connect Your Repository to Cloud Build**

* In the Google Cloud Console, go to Cloud Build > Triggers.
* Click Create Trigger.
* Select your repository and branch.
* Set the build configuration file to cloudbuild.yaml.

**7. Test the Pipeline**

* Push a commit to the configured branch.
* Monitor the build in Cloud Build.
* Check your GKE cluster to verify the deployment.