


LAB 3.1: CREATE A PERSONAL GROUP

Keyboard Time: 1 mins, **Automation Wait Time:** 0 mins

Scenarios: Instructor-Led, Self-Paced

1. While in 'classgroup', near the top right of the page, *Click* **New subgroup** (button)
2. Name the group with the mask `firstname_lastname`  so that it will be unique, easy to remember and easy for others to identify. (For example if your gitlab user id is @supercoolcoder and your avatar URL is <https://gitlab.com/supercoolcoder>, name your subgroup 'supercoolcoder'). From here on in the exercises this will be referred to as 'yourpersonalgroup'
3. *Click* **Create Group**.
4. On the left hand navigation *Click* **Settings**
5. Next to "General", *Click* **Expand**
6. For "Visibility Level", *Check* **Public**.

Must Be Public

Projects that are used by the GitLab Agent must be public when the agent registration is done in a project other than the one the deployment happens from and when the image being sourced is not using a stored docker login secret.

7. **Record or remember** 'yourpersonalgroup' =

IMPORTANT

Throughout the remaining exercises you will replace the text `yourpersonalgroup` with this actual group name

your personal group with this actual group name.

Workshop Version: v1.3.2



LAB 3.2: GITLAB AUTO DEVOPS VIA THE K8S AGENTS

Keyboard Time: 15 mins, **Automation Wait Time:** 15 mins

Scenarios: Instructor-Led, Self-Paced



Runner Based CD Push Through GitLab Agent



When a cluster is connected via the GitLab Agent, kubectl and helm commands can be run in CI jobs. The only special thing to be specified is to use the special GitLab Agent path reference as the Kubernetes Context (stored in KUBE_CONTEXT in labs and for GitLab Auto Deploy).



Target Outcomes



This one Auto DevOps scenario proves out multiple outcomes:

1. Setup a simple application to use Runner Based Push CD to deploy an application to Kubernetes through the cluster connection established by the GitLab Agent. This includes any custom CI/CD that directly uses kubectl and helm commands.
2. Use Auto DevOps (Runner Push Deployment) with the GitLab Agent cluster connection method.
3. Leveraging the Group Level agent configuration that was done in a previous lab. ([Visual Depiction Here](#))



Tip



This configuration also works for any kind of **GitLab Runner Push CD** to the cluster using Helm and kubectl commands, not

Push CD to the cluster using Helm and kubectl commands, not only Auto DevOps.

Configure An Auto DevOps Project

Warning



Before continuing make sure to use DNSChecker.com to check if both the Load Balancer DNS Name and <the Load Balancer IP>.nip.io have propagated through global DNS and wait (or troubleshoot) if they have not.

1. While in 'yourpersonalgroup' Click **New project** (button) and then Click **Import project**
2. On the 'Import project' page, Click **Repository by URL**
3. On the next page, for 'Git repository URL' Paste <https://gitlab.com/guided-explorations/gl-k8s-agent/gl-ci/simply-simple-notes.git>
4. In 'Project name' Type **yourgitlabusername DevOps Security Scanning**
5. Near the bottom of the page Click **Create project** (button)
6. When the import is complete
7. Click **Settings => CI/CD**
8. Next to 'Variables' Click **Expand**
9. Click **Add variable** once for each table row

Key	Value	Protect	Mask
POSTGRES_ENABLED	false	No	No
TEST_DISABLED	1	No	No

10. Click **Settings => CI/CD**

11. Next to 'Auto DevOps', *Click* **Expand**
12. Under 'Auto DevOps' *Check* **Default to Auto DevOps pipeline**
13. Leave 'Deployment strategy' at the default and *Click* **Save changes**
14. On the left navigation *Click* **CI/CD => Pipelines**
15. Only if a pipeline is not already running:
 1. On the upper right of the page *Click* **Run pipeline**
 2. On the 'Run pipeline' page, *Click* **Run pipeline**
16. Watch the pipeline progress by clicking the linked number starting with # under the 'Pipeline' column.
17. To explore the various pipeline jobs by clicking their status icon.
18. To return to the pipelines view, on the left navigation bar, *Click* **CI/CD => Pipelines**
19. **[Automation wait: 15 mins]** wait for the pipeline to complete the 'production' job
20. On the left navigation *Click* **Deployments => Environments**
21. To see the environment deployment status, to the left of 'production' *Click* **[the small right arrow]**
22. To the right of 'production' *Click* **Open** (button)

It can take a while for SSL to register, you can click through the advanced button to see the site if SSL is not working yet.

23. If everything worked as expected, you should see an application page called Simply Simple Notes and should not have any warnings or problems with SSL certificates.

1. Setup a simple application to use Runner Based Push CD to deploy an application to Kubernetes through the cluster connection established by the GitLab Agent.
2. Use Auto DevOps with the GitLab Agent cluster connection method.
3. Leveraging the Group Level agent configuration method.

Workshop Version: v1.3.2

