echo "Cloning the repository..." git clone --branch main https://${{ secrets.HUB\_TOKEN }}@github.com/chamodi2001/IncidentTracker.git here, it feels like after pushing the images to gitlab, u are accessing EC2 using credentials, then feels like agai u are using the github repo to compose docker images, not the saved images in git lab??

**ChatGPT said:**

Great catch — you're absolutely right to question that!

Let’s clarify what’s happening and why it **still works** — but also how to **improve it for clarity and consistency**.

**🔍 What’s Happening in That git clone Step?**

This line:

bash

CopyEdit

git clone --branch main https://${{ secrets.HUB\_TOKEN }}@github.com/chamodi2001/IncidentTracker.git

Is cloning your GitHub **source code repo** (not the Docker images) into your EC2 instance.

Why?

* So it can access your docker-compose.yml file and any related deployment configuration.
* But **it's not building** the Docker images again from that code — unless your docker-compose.yml uses build: instead of image:.

**✅ What You *Want* to Be Using**

Since you're pushing pre-built Docker images to **GitLab Container Registry**, your EC2 docker-compose.yml **should use those images** — like this:

yaml

CopyEdit

services:

backend:

image: registry.gitlab.com/your-group/backend:main

frontend:

image: registry.gitlab.com/your-group/frontend:main

And your GitHub Actions script does this correctly:

yaml

CopyEdit

docker compose pull

docker compose up -d

This means:

* EC2 is only using the docker-compose.yml from GitHub to **know what images to pull**, and how to run the containers.
* It's **not using the GitHub source code to build** again — unless you accidentally have a build: section in your compose file.

**❌ Potential Confusion: If You Use build: in docker-compose.yml**

If your docker-compose.yml has this:

yaml

CopyEdit

services:

backend:

build: ./backend

Then it would try to build **locally on EC2** from the cloned GitHub code — which is **not what you want**.

**✅ Fix (Optional Clean-up)**

To avoid confusion, you can:

**✅ Option 1: Keep GitHub repo clone — only if docker-compose.yml is needed**

If you need that file:

* Make sure docker-compose.yml uses only **pre-built GitLab images**, not build:.

**✅ Option 2: Don’t even clone GitHub repo — copy just docker-compose.yml to EC2 manually or host it separately.**

* That way, EC2 won’t even look at the GitHub code
* It only works with **GitLab Docker images** and local .env files, configs.

**✅ Recommendation**

If you want to stick with Option 1 (clone repo to get docker-compose.yml), make sure:

yaml

CopyEdit

# docker-compose.yml

services:

backend:

image: registry.gitlab.com/chamodigunathilaka2018/incidenttracker/incident-tracker-server:main

frontend:

image: registry.gitlab.com/chamodigunathilaka2018/incidenttracker/incident-tracker-client:main

No build: references.  
No need to rebuild on EC2.  
You're **just using the GitHub repo to get the config**, not the source.

Let me know if you want to completely **skip cloning the repo on EC2**, and I’ll show you how to deploy with only Docker commands and a hardcoded/hosted docker-compose.yml.