**WheelsOnDemand-github-actions-terraform-ecs**

1. **Create a GitHub repository**

I’ve named mine “WheelsOnDemand-github-actions-terraform-ecs”

1. **Update the .gitignore file**

Copy and paste the .gitignore file from the shared repository into it

git add .

git commit -m “update gitignore file”

git push

1. **Add the Terraform code into the repository**

Download the Terraform code: <https://github.com/Silas-cloudspace/cicd-projects>

Unzip it and paste the folder into the “WheelsOnDemand-github-actions-terraform-ecs” folder in your local computer

git add .

git commit -m “add iac files”

git push

1. **Create the remote backend**

Now we will create an S3 bucket to store the Terraform state and a DynamoDB table to lock the Terraform state.

Navigate to remote\_backend folder

cd remote\_backend

terraform init

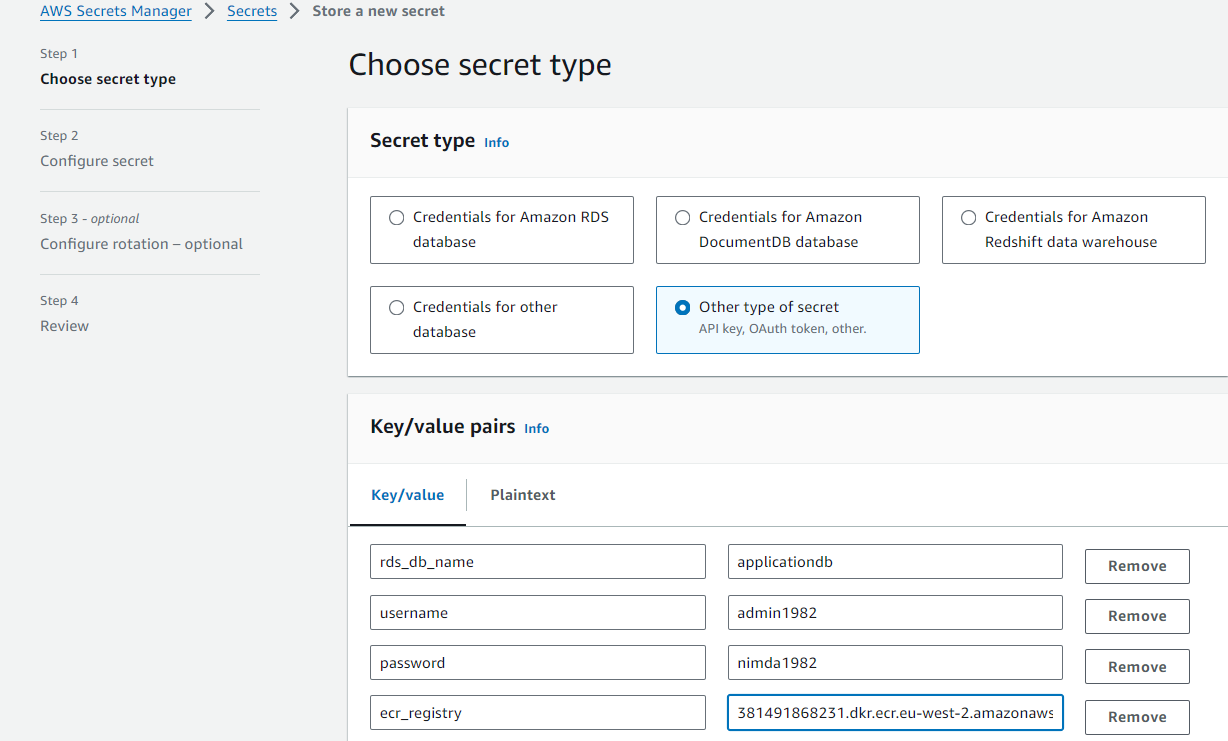
terraform apply

1. **Create Secrets in AWS Secrets Manager**

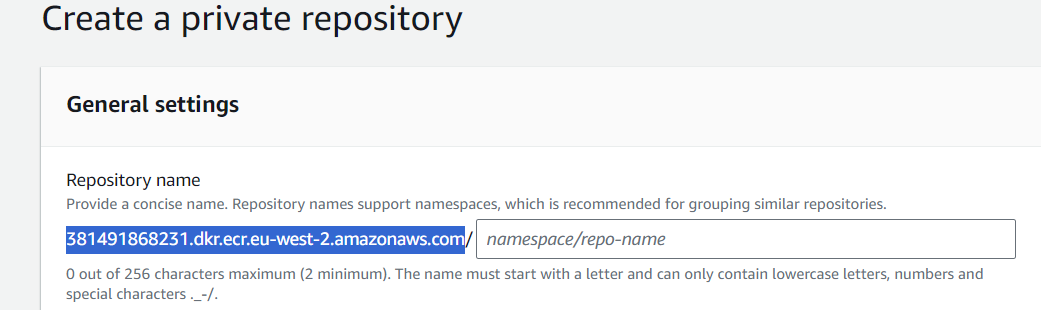
We will now add the value for our RDS database name, username and password, and also our ECR registry as secrets in Secrets Manager.

Go to AWS console and search for Secrets Manager

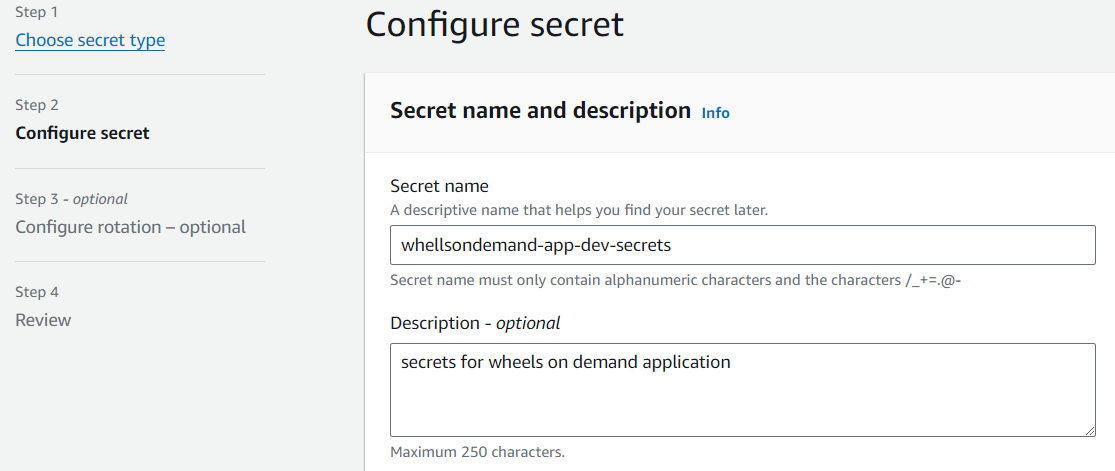
Click on “Store a new secret”



In order to get your Elastic Container Registry, open a new AWS tab, go to ECR and click on “Create repository”



Copy the ECR value and paste it on secrets manager.



You can click next on the following steps.

1. **Register a Domain name**

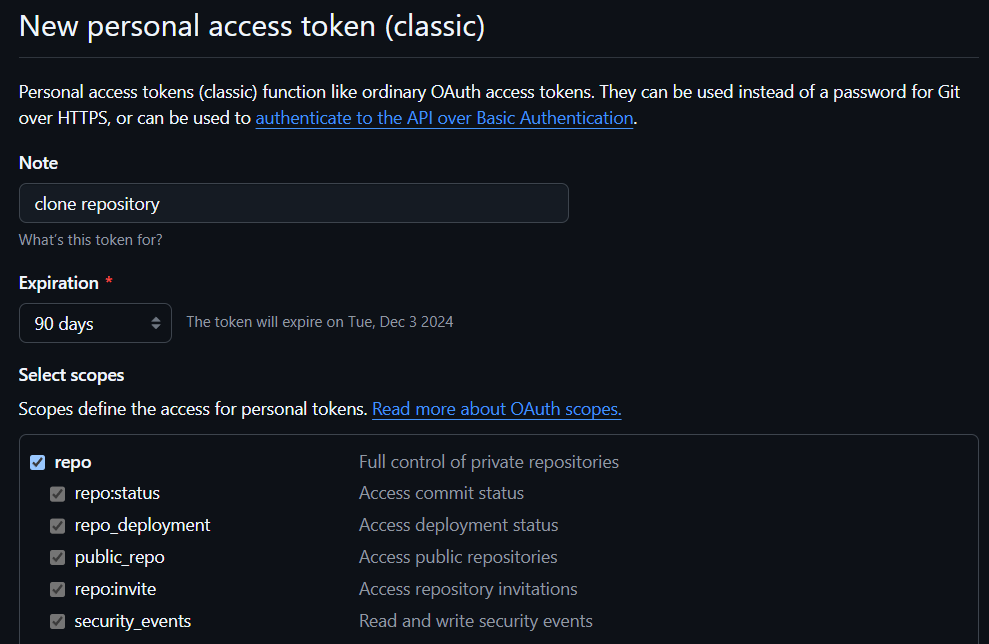
Go to AWS Route 53 and create a new domain name for yourself. It will cost you around 14 dollars.

1. **Create a personal access token on github**

This token will be used by docker to clone the application codes repository when we build our docker image

Github -> select your profile -> settings -> Developer settings -> Personal access tokens -> Tokens (classic) - > Generate new token -> Generate new token classic

Edit it as you see in the following example:



Remember to copy your personal access token and save it anywhere

ghp\_POIRp13LW1Zw8zFtmAJmIKuC5j1FMN2tkauN

1. **Create GitHub repository Secrets**

Now we will create the repository secrets that the GitHub Action job need to build our cicd pipeline for this project.

Go to your GitHub repository

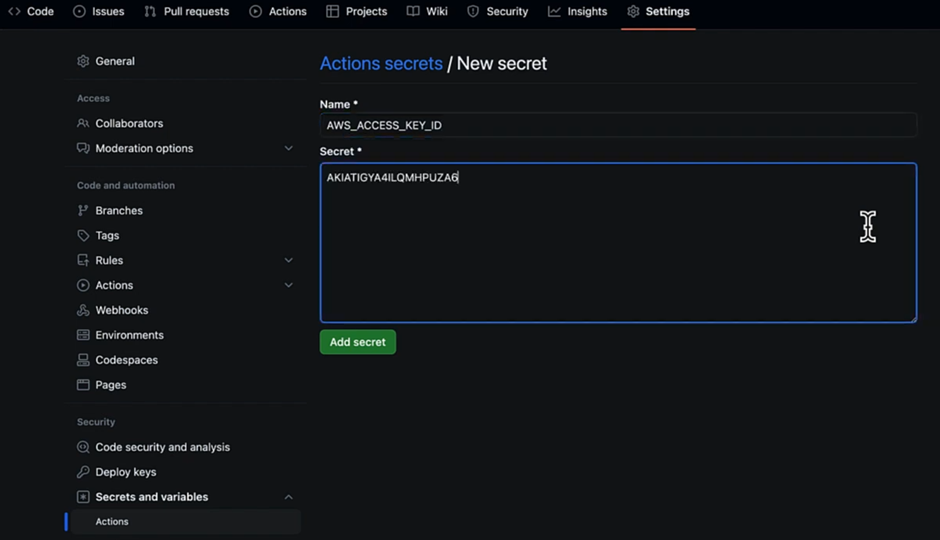
Click on settings

Navigate to “Secrets and variables”

Choose “actions”

Click on “New repository secret”

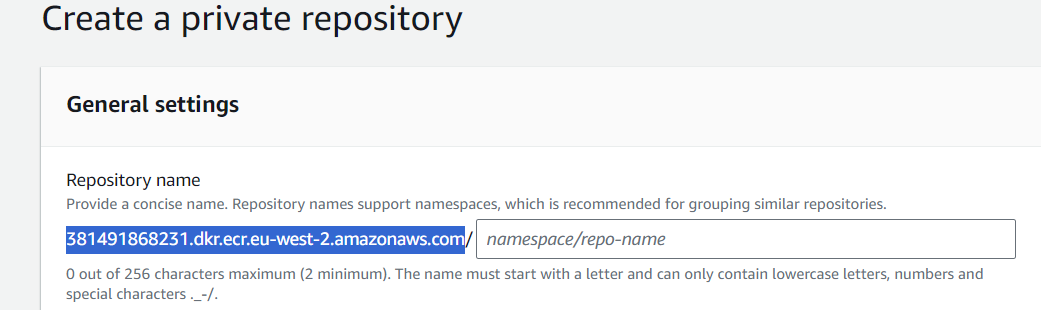
Add 7 secrets:



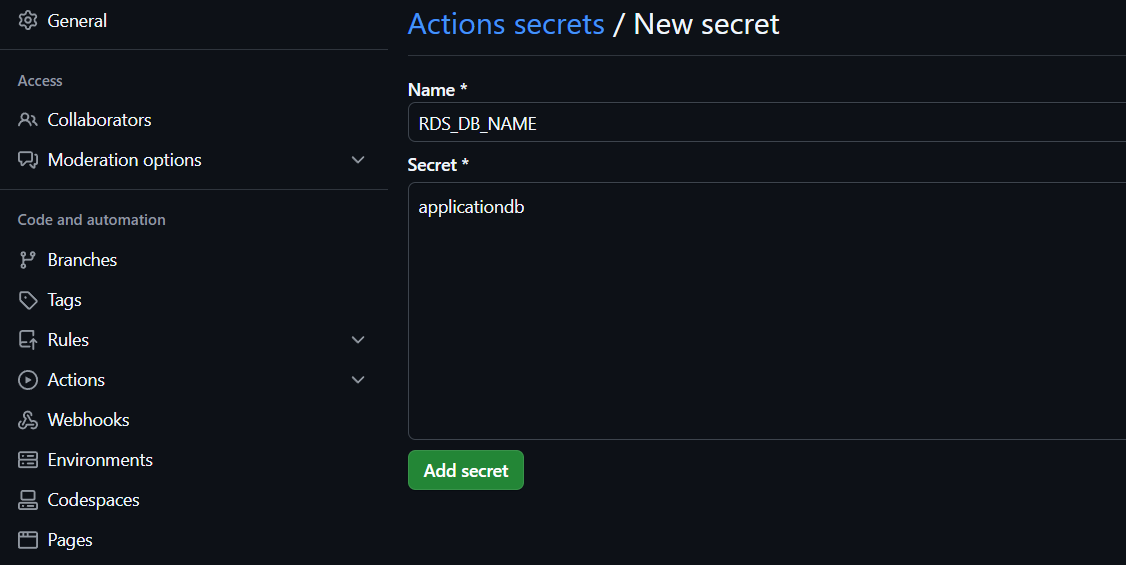
Above you can find an example

* AWS\_ACCESS\_KEY\_ID - “Your AWS access key id”
* AWS\_SECRET\_ACCESS\_KEY – “Your secret AWS access key”
* ECR\_REGISTRY –

In order to get your Elastic Container Registry, open a new AWS tab, go to ECR and click on “Create repository”



* PERSONAL\_ACCESS\_TOKEN – “The personal access token we created on point VII”
* RDS\_DB\_NAME – Go to AWS console > Secrets Manager > select the secret we create before > Under secret value, click on “retrieve secret value”. Paste the rds\_db\_name “applicationdb” as the secret on GitHub.



* RDS\_DB\_USERNAME - RDS\_DB\_NAME – Go to AWS console > Secrets Manager > select the secret we create before > Under secret value, click on “retrieve secret value”. Paste the rds username you choose as the secret on GitHub.
* RDS\_DB\_PASSWORD – RDS\_DB\_NAME – Go to AWS console > Secrets Manager > select the secret we create before > Under secret value, click on “retrieve secret value”. Paste the rds password you choose as the secret on GitHub.

1. **Create the GitHub Action Workflow file**

Navigate to “WheelsOnDemand-github-actions-terraform-ecs” on VS Code

mkdir -p .github/workflows

cd .github/workflows

touch deploy\_pipeline.yml

1. **Create a GitHub Actions Job to Configure AWS Credentials**

We will now create the job responsible for configuring our IAM credentials to verify our access to AWS and authorize our GitHub Actions job to create new resources in our AWS account.