**Individual Cloud Journey Week 3**

Aniketh Satyanarayana

W1607589

For the week 3 of the individual cloud journey, I chose to use the AWS Elastic Container Registry.

For this purpose, I went about it doing things the usual way.i.e. first building an application and building an image of it using Docker.

After this, an effort was made to establish connection between Docker and AWS.

Amazon Elastic Container Registry (ECR) is a fully managed Docker container registry that makes it easy to store, share, and deploy container images.

For the sake of later convnience and easy identification, I used the same names for the image as well as the repository.

Steps followed and things to take care of:

* First of all, a repository is to be created in the aws ecr registry.
* Then we will have to create the docker imahe locally and run the container.
* Then push it to the aws system.
* AWS ECR - Repositories

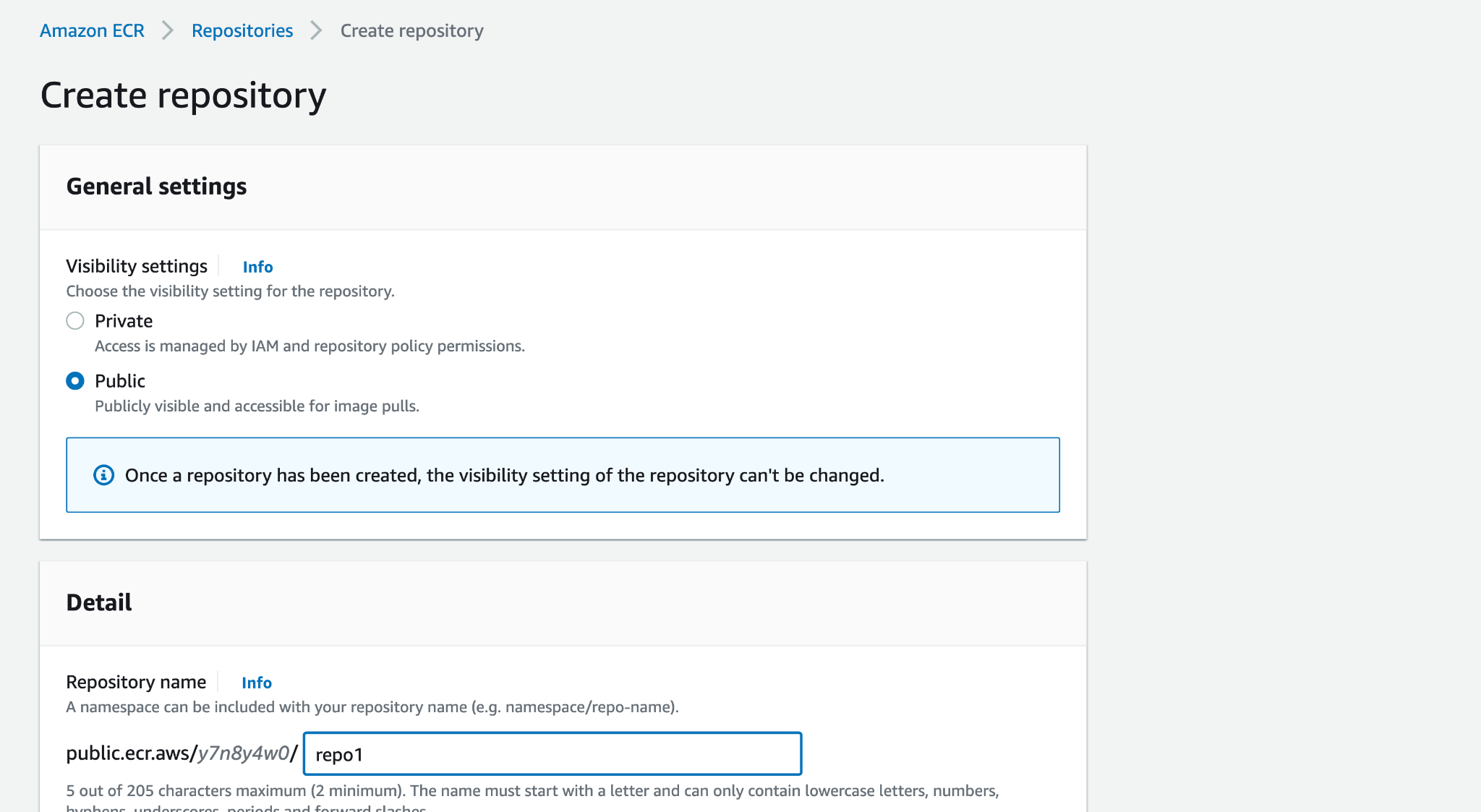
Command to create a repo is :

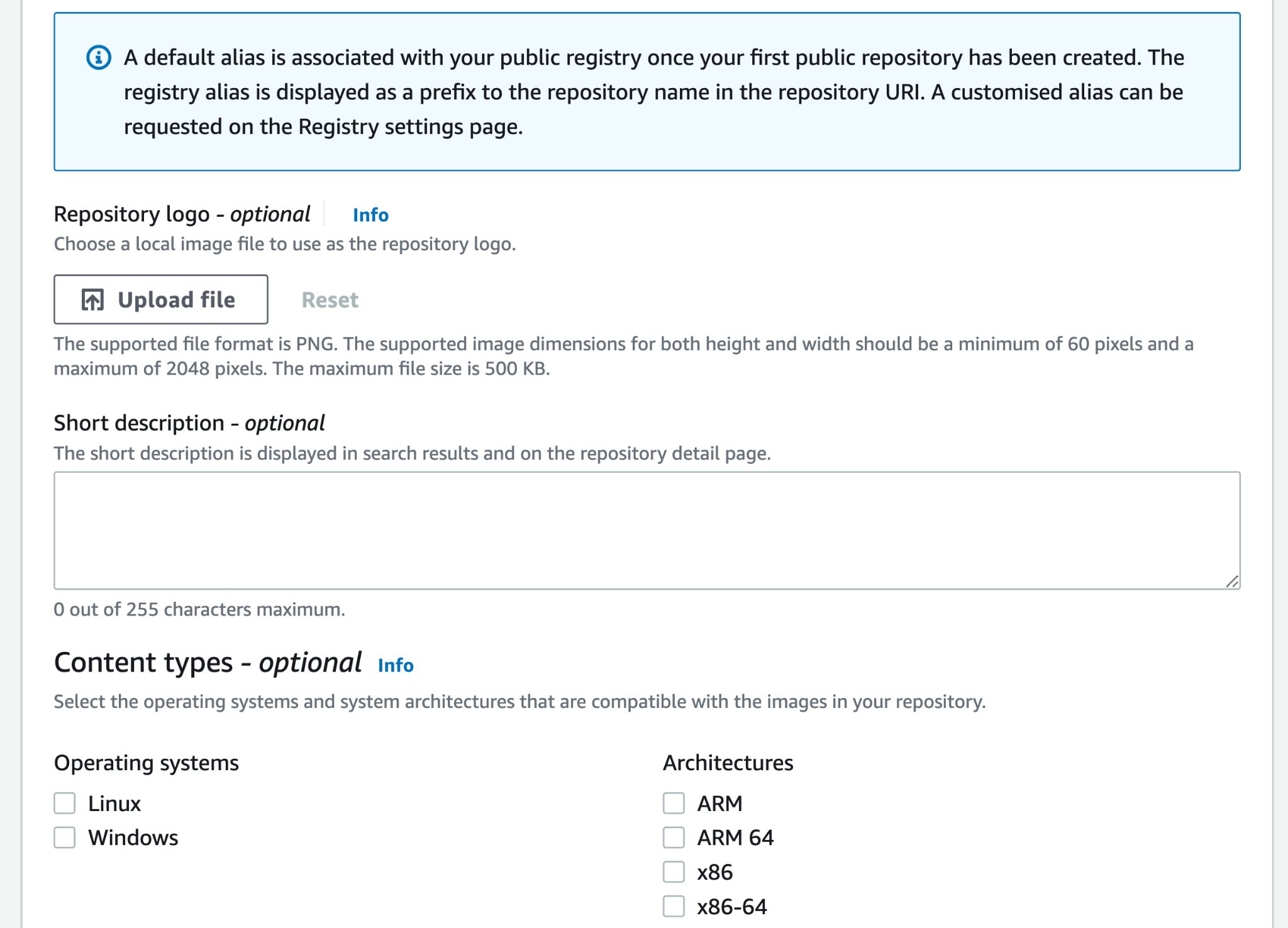
aws ecr create-repository --repository-name <repo\_name> --region <region\_name>.

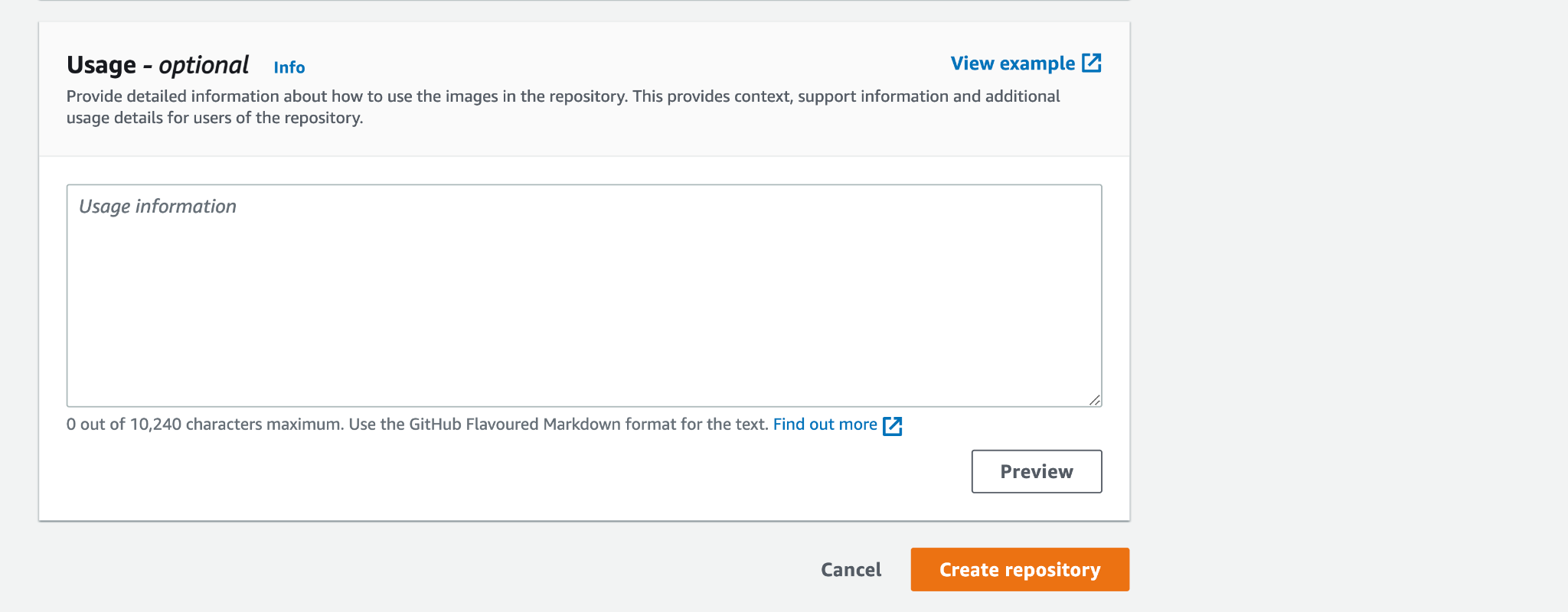
or use the GUI to create a repository on the AWS ECR platform:

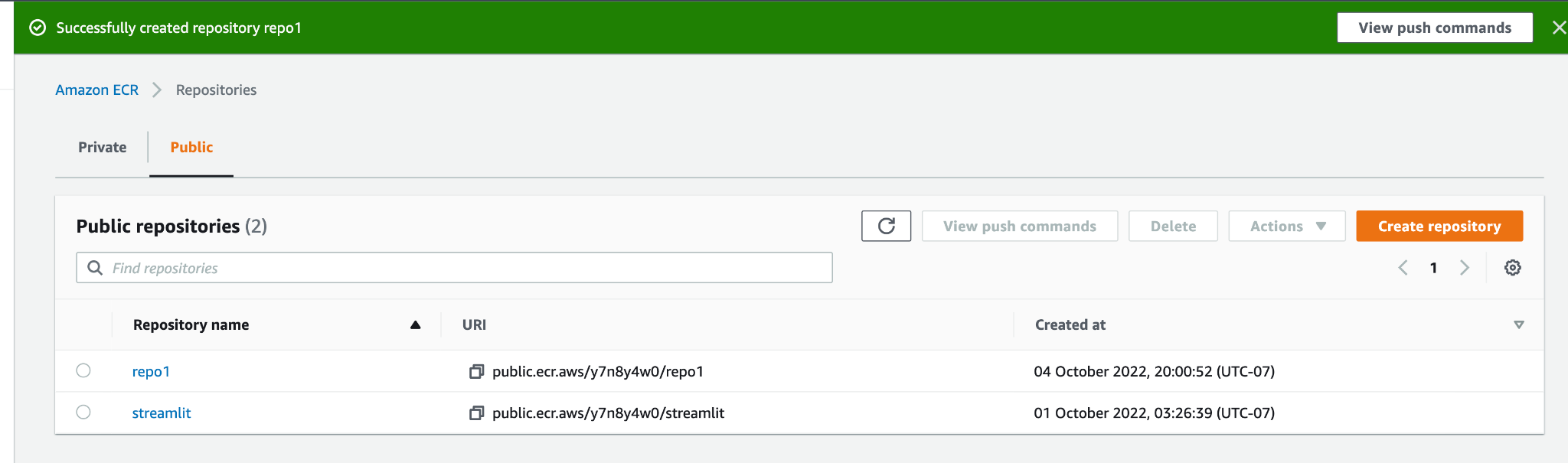
Graphical user interface, application

Description automatically generated









Now, the repository is created.

Clicking on the repo name leads to ‘View push commands’.

We can view the push commands which is a set of 4 different strings that can be used to make an external app connect with the AWS system. (Interact with the aws.

The docker images created on Docker can be eventually pushed to AWS registry using this string.

Graphical user interface, text, application, email

Description automatically generated

1)The next step here is to retrieve an authentication token and authenticate the Docker client with the registry.

The generic command for this is:

**aws ecr-public get-login-password –region region | docker login ---username AWS –password-stdin public.ecr.aws/y7n8y4w0**

which I replace with the specific command that has my AWS credentials:

(region – us-east-1 and AWS account ID 689510502173)- Eg: parameter ‘public.ecr.aws/y7n8y4w0’ with my unique credential i.e. AWS account ID.

This replacement goes for all the future commands form hereon.

**aws ecr-public get-login-password -–region us-east-1 | docker login –username AWS –password-stdin 689510502173.dkr.ecr.us-east-1.amazonaws.com**

Graphical user interface, text

Description automatically generated

Confirmation is obtained in the terminal in the form of ‘login succeeded.’ That’s when I can move on to the next step.

2)In the next step, I build the docker image using ‘docker build -t repo1 .’

3)Tagging the image :

docker tag <imageId > 689510502173.dkr.ecr.us-east-1.amazonaws.com

docker tag 6916f47c8852 689510502173.dkr.ecr.region.amazonaws.com/repo1:latest

4)Pushing the image **docker push *aws\_account\_id*.dkr.ecr.*region*.amazonaws.com/*my-repository:tag***

***docker push*** 689510502173.dkr.ecr.region.amazonaws.com/repo1:latest

Graphical user interface, text, application, letter

Description automatically generated

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

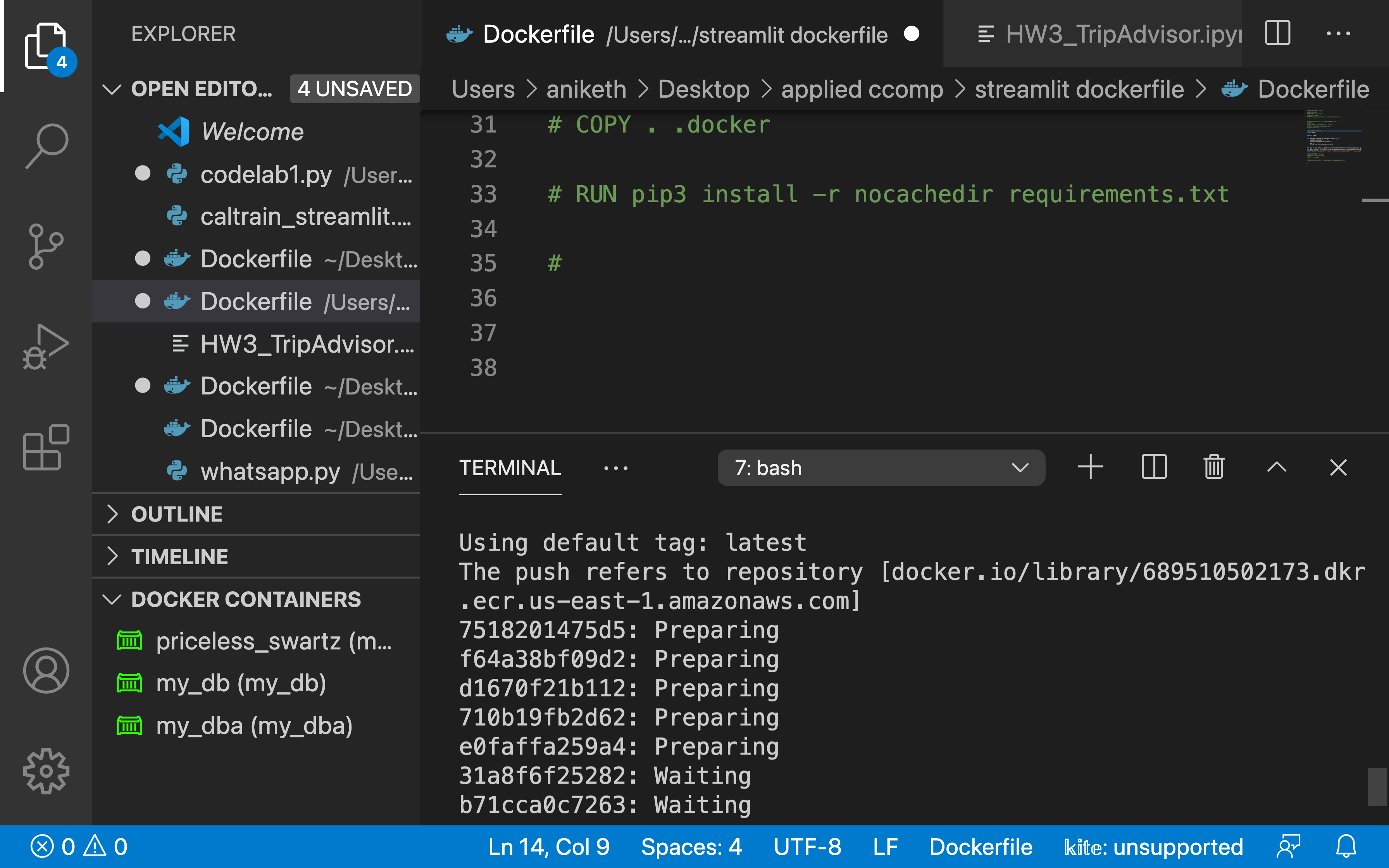
Description automatically generated

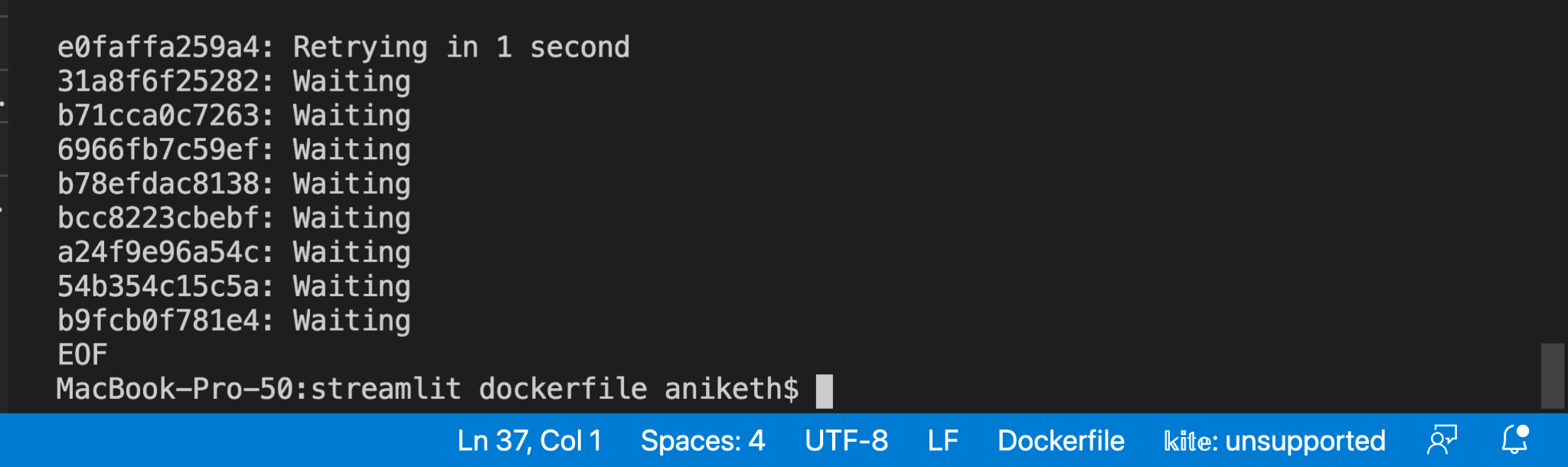
The repository repo1 has the image pushed into it.

Graphical user interface, application, Teams

Description automatically generated

Problems encountered: Some things like:





login not allowed. This needs us to fetch an IAM role for the user out there.

We do this by downloading the secret credentials by click on the IAM Page.

No such image found. Push failed. Cannot lookup host.

All of these were tackled by proper usage of the commands.

**Commands summarized:**

1)Build an image locally. docker build -t repo1 .

2)Run the container. docker run -d –p 8888:8888 repo1

3)Connect to AWS

aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin aws\_account\_id.dkr.ecr.region.amazonaws.com

aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 689510502173.dkr.ecr.us-east-1.amazonaws.com

4)Tag the image

docker tag 6916f47c8852 689510502173.dkr.ecr.region.amazonaws.com/repo1:latest

5)Push the image to the aws

docker push 689510502173.dkr.ecr.us-east-1.amazonaws.com/dockmetaws:latest

**References:**

<https://stackoverflow.com/questions/70828205/pushing-an-image-to-ecr-getting-retrying-in-seconds>

<https://stackoverflow.com/questions/41984399/denied-requested-access-to-the-resource-is-denied-docker>

<https://www.youtube.com/watch?v=D8ym8RP1yvo>

<https://www.youtube.com/watch?v=Jc5GI3v2jtE>

<https://www.youtube.com/watch?v=zs3tyVgiBQQ>

<https://docs.aws.amazon.com/AmazonECR/latest/userguide/repository-create.html>

**Work -Plan for next week:**

1)To integrate EC2 / ECS with the ECR and create a running instance that can serve the app.

2)Explore aws Glue, sagemaker and chatbot.

Thanks for reading!

Run server with old db

/admin error

Change db