

React App Containerized and Pushed to Amazon ECR locally and using Jenkins for complete automation

Part 1 - prepare docker image

Forked project to my repo:

<https://github.com/yrenamm/nfx-react-clone>

Create new working repo:

- **mkdir nfx-react-clone**

Clone repo to my local pc

- **git clone https://github.com/yrenamm/nfx-react-clone.git**

Remove yarn.lock

- **rm -rf yarn.lock**

Install dependencies listed in the package.json file of your project. (Executing it within the directory of your project where the package.json file is located, yarn.lock must be created)

- **npm use 16.10.0**
- **yarn install**

Run application:

- **yarn start**

Build docker image:

- **docker build -t yrenamm/nfx-react .**

Run docker container

- **docker run -itd -p 3000:80 yrenamm/nfx-react:latest**
- **docker ps**

<http://localhost:3000/>

Stop docker container:

- **docker stop 8767cdf2f048**

Remove container:


- **docker remove 8767cdf2f048**
- **docker ps** (will not appear)

Part 2 - deploy docker image to AWS

Search for AWS Elastic Container Registry

Elastic Container Registry x +

← → ↻ us-east-1.console.aws.amazon.com/ecr/create-repository?region=us-east-1

 Services [Alt+S]

Console Home IAM S3 EC2 VPC

Amazon ECR > Repositories > Create repository

Create repository

General settings

Visibility settings [Info](#)

Choose the visibility setting for the repository.

☒ **Private**
Access is managed by IAM and repository policy permissions.

☐ **Public**
Publicly visible and accessible for image pulls.

Repository name

Provide a concise name. A developer should be able to identify the repository contents by the name.

384874606381.dkr.ecr.us-east-1.amazonaws.com/

15 out of 256 characters maximum (2 minimum). The name must start with a letter and can only contain lowercase letters, numbers, hyphens, underscores, periods and forward slashes.

Tag immutability [Info](#)

Enable tag immutability to prevent image tags from being overwritten by subsequent image pushes using the same tag. Disable tag immutability to allow image tags to be overwritten.

☐ **Disabled**


 Once a repository is created, the visibility setting of the repository can't be changed.

Image scan settings



Deprecation warning

ScanOnPush configuration at the repository level is deprecated in favor of registry level scan filters.

Scan on push

Enable scan on push to have each image automatically scanned after being pushed to a repository. If disabled, each image scan must be manually started to get scan results.

☒ Enabled

VPC

amazon ECR > Repositories > nfx-react-clone

nfx-react-clone

Images (0)

Search artifacts

Image tag

Push commands for nfx-react-clone

macOS / Linux

Windows

Make sure that you have the latest version of the AWS CLI and Docker installed. For more information, see [Getting Started with Amazon ECR](#).

Use the following steps to authenticate and push an image to your repository. For additional registry authentication methods, including the Amazon ECR credential helper, see [Registry Authentication](#).

- Retrieve an authentication token and authenticate your Docker client to your registry.
Use the AWS CLI:

```
aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 384874606381.dkr.ecr.us-east-1.amazonaws.com
```

Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.
- Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions [here](#). You can skip this step if your image is already built:

```
docker build -t nfx-react-clone .
```
- After the build completes, tag your image so you can push the image to this repository:

```
docker tag nfx-react-clone:latest 384874606381.dkr.ecr.us-east-1.amazonaws.com/nfx-react-clone:latest
```
- Run the following command to push this image to your newly created AWS repository:

```
docker push 384874606381.dkr.ecr.us-east-1.amazonaws.com/nfx-react-clone:latest
```

Close

Retrieve an authentication token and authenticate your Docker client to your registry.
Use the AWS CLI:

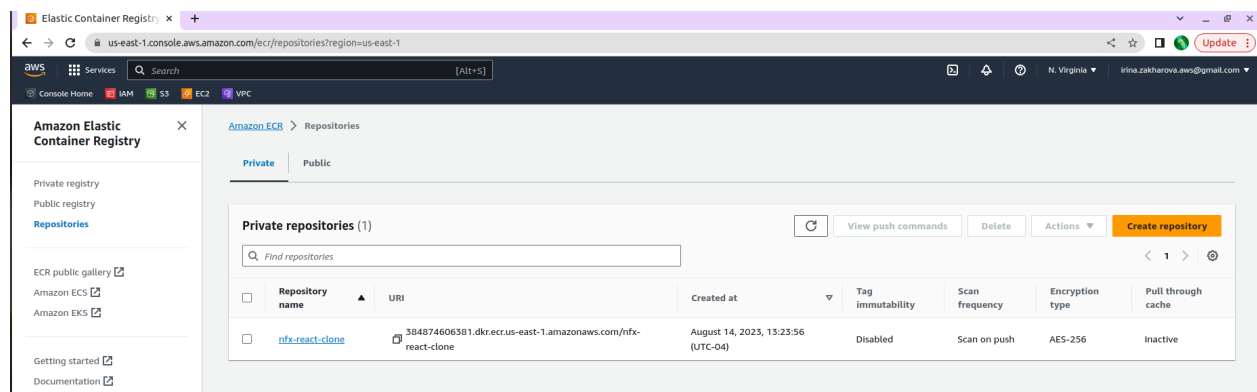
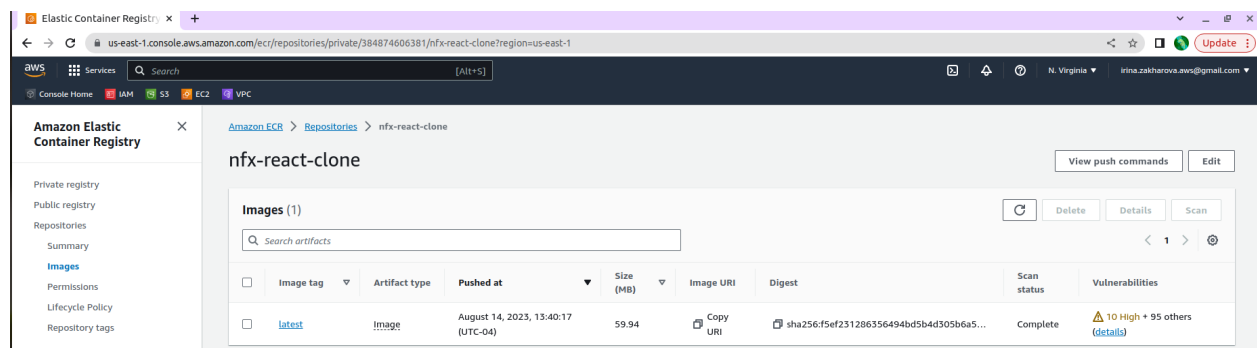
- `aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 384874606381.dkr.ecr.us-east-1.amazonaws.com`

After the build completes, tag your image so you can push the image to this repository:

- `docker tag yrenamm/nfx-react:latest 384874606381.dkr.ecr.us-east-1.amazonaws.com/nfx-react-clone:latest`

Run push command:

- `docker push 384874606381.dkr.ecr.us-east-1.amazonaws.com/nfx-react-clone:latest`



Part 3 - Automation with Jenkins

Work with .git

- `rm -rf .git`
- `git status`
- `git init`
- `git status`
- `git add .`
- `git commit`

Create new repository **nfx-react** and:

<https://github.com/yrenamm/nfx-react>

- **git remote add origin https://github.com/yrenamm/nfx-react.git**
- **git branch -M main**
- **git push -u origin main**

Create Jenkins job:

Jenkins -> New Item -> nfx-react-app -> pipeline:

The screenshot shows the Jenkins configuration page for a new pipeline job named 'nfx-react-app'. The breadcrumb trail at the top is: Dashboard > advanced-July29th-Cohort > irinamm-workspace > nfx-react-app > Configuration. On the left, the 'Configure' section has three tabs: 'General', 'Advanced Project Options', and 'Pipeline', with 'Pipeline' being the active tab. The main configuration area is titled 'Pipeline' and contains the following fields:

- Definition:** A dropdown menu set to 'Pipeline script from SCM'.
- SCM:** A dropdown menu set to 'Git'.
- Repositories:** A section containing:
 - Repository URL:** A text input field with the value 'https://github.com/yrenamm/nfx-react.git'.
 - Credentials:** A dropdown menu set to '- none -' with an 'Add +' button below it.
 - Advanced:** A dropdown menu set to 'Advanced'.
 - Add Repository:** A button.
- Branches to build:** A section containing:
 - Branch Specifier (blank for 'any'):** A text input field with the value '*/main'.
- Script Path:** A text input field with the value 'Jenkinsfile'.
- Lightweight checkout:** A checkbox that is checked.
- Pipeline Syntax:** A link.

Jenkinsfile:

```

1 pipeline {
2   agent any
3   options {
4     timeout(time: 20, unit: 'MINUTES')
5   }
6   stages{
7     // NPM dependencies
8     stage('pull npm dependencies') {
9       steps {
10        sh 'npm install'
11      }
12    }
13    stage('build Docker Image') {
14      steps {
15        script {
16          // build image
17          docker.build("384874606381.dkr.ecr.us-east-1.amazonaws.com/nfx-react-clone:v1.0.0.RELEASE")
18        }
19      }
20    }
21    stage('Trivy Scan (Aqua)') {
22      steps {
23        sh 'trivy image --format template --output trivy_report.html 384874606381.dkr.ecr.us-east-1.amazonaws.com/nfx-react-clone:v1.0.0.RELEASE'
24      }
25    }
26    stage('Push to ECR') {
27      steps {
28        script{
29          //https://<AwsAccountNumber>.dkr.ecr.<region>.amazonaws.com/netflix-app', 'ecr:<region>:credentialsId>
30          docker.withRegistry('https://384874606381.dkr.ecr.us-east-1.amazonaws.com/nfx-react-clone', 'ecr:us-east-1:ira-ecr') {
31
32            // build image
33            def myImage = docker.build("384874606381.dkr.ecr.us-east-1.amazonaws.com/nfx-react-clone:v1.0.0.RELEASE")
34            // push image
35            myImage.push()
36          }
37        }
38      }
39    }
40  }
41 }
42 }

```

AWS create access key:

The screenshot shows the AWS IAM console interface. At the top, there's a navigation bar with 'IAM' selected. Below it, a green banner reads 'Access key created' with a message: 'This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.'

The main content area is titled 'Retrieve access keys' and shows the details for a newly created access key for the user 'ira_jenkins'. It includes a table with two columns: 'Access key' and 'Secret access key'. The 'Access key' value is 'AKIAVTHCQN4WW7YNMJEN' and the 'Secret access key' is masked with dots, with a 'Show' link next to it.

Below the table, there's a section titled 'Access key best practices' with a list of recommendations:

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

A link to 'best practices for managing AWS access keys' is provided. At the bottom right, there are two buttons: 'Download .csv file' and 'Done'.

Jenkins add credential: (AWS credentials as a type)

jenkins-ibtlearning.com/manage/credentials/store/system/domain/_/newCredentials

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

Kind
AWS Credentials

Scope
Global (Jenkins, nodes, items, all child items, etc)

ID
ira-ecr

Description
ira ecr (AWS)

Access Key ID
AKIAVTHCQN4WW7YNMJEN

Secret Access Key

Please specify the Secret Access Key

Run Jenkins job and check in AWS:

us-east-1.console.aws.amazon.com/ecr/repositories/private/384874606381/nfx-react-clone?region=us-east-1

Amazon Elastic Container Registry

Private registry
Public registry
Repositories
Summary
Images
Permissions
Lifecycle Policy
Repository tags

ECR public gallery
Amazon ECS
Amazon EKS

Amazon ECR > Repositories > nfx-react-clone

nfx-react-clone

View push commands Edit

Images (2)

Search artifacts

	Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulnerabilities
<input type="checkbox"/>	v1.0.0.RELEASE	Image	August 14, 2023, 16:10:40 (UTC-04)	59.94	Copy URI	sha256:b977d6a56b484790b6129d3b0687...	Complete	10 High + 95 others (details)
<input type="checkbox"/>	latest	Image	August 14, 2023, 13:40:17 (UTC-04)	59.94	Copy URI	sha256:f5ef231286356494bd5b4d305b6a5...	Complete	10 High + 95 others (details)

1. Fork or Clone Repository
2. cd into the app directory
3. Install dependencies
use Yarn install
4. Run app locally
use Yarn start
5. Check the app on your Browser using localhost or server ip and the port it is listening on.

Dockerize Application

6. Create Dockerfile for your react app, use nginx to serve the static assets
7. Build Docker Image by using the docker build command
 - a. `docker build -t repoName/imageName:tag .`
8. Run the container by using the docker run command to confirm application still works
 - a. `docker run -d -p 3000:80 nameOfImage`
 - b. Go to Browser and confirm your containerised app works using your localhost or server IP and the port e.g localhost:3000
9. Log onto your AWS Console and type ECR on the search bar and click on it. ECR = Elastic Container Registry
10. Create a repository,
 - a. click on private
 - b. type in repo name
 - c. Enable Scan on Pushand Click on Create
11. Click or enter into your repository and click on the view push commands
12. Go to your terminal and follow each steps provided by aws
13. Congratulations, You have made your first ECR push

AUTOMATE THE ENTIRE STEPS TO PUSH YOUR IMAGE TO ECR USING JENKINS

1. Create an ECR key in Jenkins Key Storage using your aws access and secret key
2. Create a Jenkinsfile or Add one
3. Go to your Jenkins workspace and Create a Pipeline Job
4. Build your Job
5. Go to ECR and confirm Image was succesfully pushed