

Steps for Creating Docker image and Pushing to ECR Repository with the Vulnerability Scanning:

Pre-Requirements:

- Ec2 Instance
- Docker
- Aws-cli

Step1: Connect to the server using the cmd or gitbash :

- Ssh -i "fcra.pem" ubuntu@ip

Step2: Installation of Docker:

- apt-get install docker.io
- systemctl start docker
- systemctl enable docker
- systemctl status docker
-

Step3: Installation of Aws Cli :

- curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
- unzip awscliv2.zip
- apt install unzip -y
- unzip awscliv2.zip
- sudo ./aws/install (To check the path where it installed)
- AWS --version
- AWS configure (enter the id and password details of the iam role which uhave been created)

Step2:

Create a role in AWS with the access of ec2 repository and admin access and create access key:

- for further requirements example below. * *
- Access key - AKIAZ75J3DW74LD37UWL
- Secret access key: oMLCWUVtXIQvEYjylnXhsB2tyM3DqKw5UOXE
- Region :ap-south-1

The screenshot shows the AWS IAM console interface. On the left is a navigation sidebar with 'Identity and Access Management (IAM)' selected. The main content area displays a table of policies, with 'AdministratorAccess' highlighted. Below the table, there are two sections: 'Permissions boundary (not set)' and 'Generate policy based on CloudTrail events'. The latter section includes a 'Generate policy' button and a message stating 'No requests to generate a policy in the past 7 days.'

<input type="checkbox"/>	Policy name ↗	Type	Attached via ↗
<input type="checkbox"/>	AdministratorAccess	AWS managed - job function	Directly
<input type="checkbox"/>	AdministratorAccess-AWSEla...	AWS managed	Directly

► Permissions boundary (not set)

▼ Generate policy based on CloudTrail events

You can generate a new policy based on the access activity for this user, then customize, create, and attach it to this role. AWS uses your CloudTrail events to identify the services and actions used and generate a policy. [Learn more](#) [↗](#)

No requests to generate a policy in the past 7 days.

Step3:

- Git clone [<the project from github>](#) (or we candrag and drop the files using tools)
- Create docker file.
- Vi Dockerfile

```
#Depending on the operating system of the host machines(s) that will build or run the containers, the image specified in the FROM statement may need to be changed.
#For more information, please see https://aka.ms/containercompat

FROM mcr.microsoft.com/dotnet/aspnet:6.0-alpine AS base

WORKDIR /app

EXPOSE 80
EXPOSE 443

ENV ASPNETCORE_ENVIRONMENT=Development
ENV DOTNET_RUNNING_IN_CONTAINER=true

RUN apk add --no-cache icu-libs krb5-libs libgcc libintl libssl1.1 libstdc++ zlib

ENV DOTNET_SYSTEM_GLOBALIZATION_INVARIANT=false

FROM mcr.microsoft.com/dotnet/sdk:6.0-alpine AS build
WORKDIR /src
COPY ["FCRA.Web/FCRA.Web.csproj", "FCRA.Web/"]
COPY ["FCRA.Common/FCRA.Common.csproj", "FCRA.Common/"]
COPY ["FCRA.Repository/FCRA.Repository.csproj", "FCRA.Repository/"]
COPY ["FCRA.Models/FCRA.Models.csproj", "FCRA.Models/"]
COPY ["FCRA.ViewModels/FCRA.ViewModels.csproj", "FCRA.ViewModels/"]
RUN dotnet restore "FCRA.Web/FCRA.Web.csproj"
COPY . .
WORKDIR "/src/FCRA.Web"
RUN dotnet build "FCRA.Web.csproj" -c Release -o /app/build

FROM build AS publish
RUN dotnet publish "FCRA.Web.csproj" -c Release -o /app/publish /p:UseAppHost=false

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "FCRA.Web.dll"]
```

Step4: Create a ECR named Docker-ECR and then click on pushcommands and choose Linux:

- `aws ecr get-login-password --region ap-southeast-1 | docker login -- username AWS --password-stdin 883448062072.dkr.ecr.ap-southeast-1.amazonaws.com`
- `docker build -t fcra .`
- `docker images`
- `docker tag fcra:latest 883448062072.dkr.ecr.ap-southeast-1.amazonaws.com/fcra:latest`
- `docker push 883448062072.dkr.ecr.ap-southeast-1.amazonaws.com/fcra:latest`

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](#).

1. Retrieve an authentication token and authenticate your Docker client to your registry.

Use the AWS CLI:

```
aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 687014092223.dkr.ecr.ap-south-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.

2. 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 newabc .
```

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

```
docker tag newabc:latest 687014092223.dkr.ecr.ap-south-1.amazonaws.com/newabc:latest
```

Close

☐ Then check the Vulnerabilities:

newabc

View push commands

Edit

Images (1)

↺

Delete

Details

Scan

Q Search artifacts

<

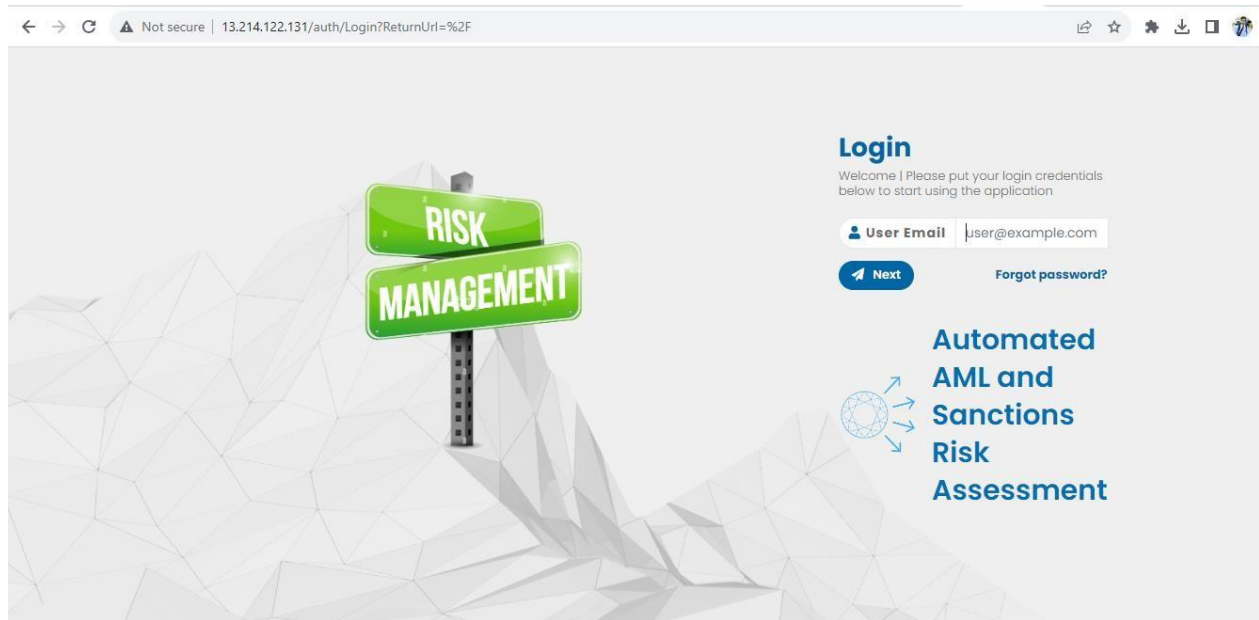
1

>

⚙

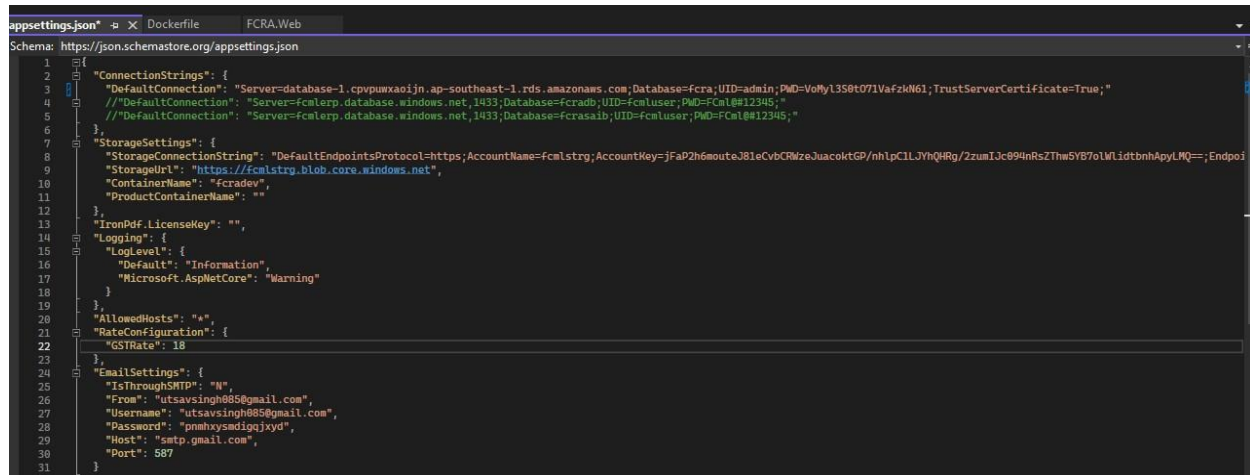
Image tag ▾	Artifact type	Pushed at ▾	Size (MB) ▾	Image URI	Digest	Scan status	Vulnerabi
latest	Image	November 20, 2023, 10:49:37 (UTC+05.5)	92.33	<div><div>Copy URI</div></div>	<div><div>sha256:53b7738f14d37d...</div></div>	Complete	<div><div>✓ None</div></div>

Copy the public ip and paste in the brower.



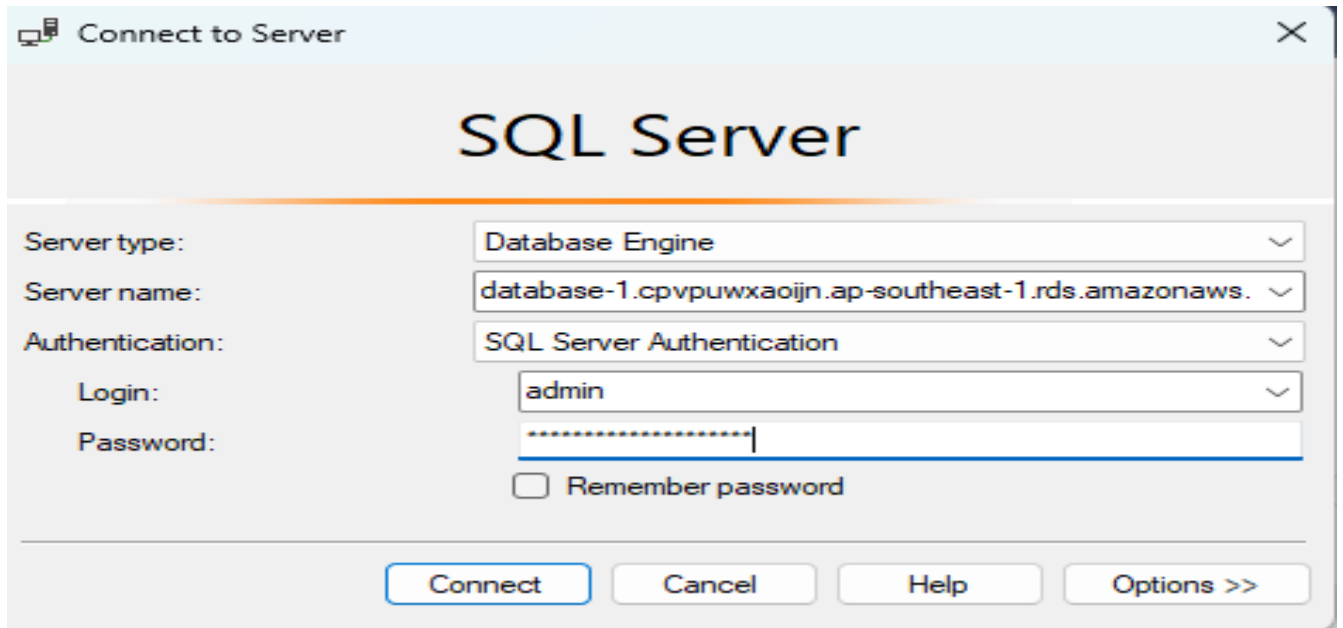
Step6: Create a database for the above ui and attach to it:

- Attach the database server in the local file app.json and connect it to the database.



```
1 {
2   "ConnectionStrings": {
3     "DefaultConnection": "Server=database-1.cpvuwxaiojrn.ap-southeast-1.rds.amazonaws.com;Database=fcra;UID=admin;PWD=VoMyL3S0t071VaFzkN61;TrustServerCertificate=True;"
4     // "DefaultConnection": "Server=fcmlerp.database.windows.net,1433;Database=fcradb;UID=fcmuser;PWD=Fcm1@12345;"
5     // "DefaultConnection": "Server=fcmlerp.database.windows.net,1433;Database=fcrasaib;UID=fcmuser;PWD=Fcm1@12345;"
6   },
7   "StorageSettings": {
8     "StorageConnectionString": "DefaultEndpointsProtocol=https;AccountName=fcm1strg;AccountKey=jFaP2h6moute381eCvbCRMzeJuacoktGP/nhlpCilJYhQH8g/ZzumI3c894nRsZThw5Y87o1wLidtbnhApyLMQ==;EndpointSuffix=core.windows.net",
9     "StorageUrl": "https://fcm1strg.blob.core.windows.net",
10    "ContainerName": "fcradev",
11    "ProductContainerName": ""
12  },
13  "IronPdf.LicenseKey": "",
14  "Logging": {
15    "LogLevel": {
16      "Default": "Information",
17      "Microsoft.AspNetCore": "Warning"
18    }
19  },
20  "AllowedHosts": "*",
21  "RateConfiguration": {
22    "GSTRate": 18
23  },
24  "EmailSettings": {
25    "IsThroughSMTP": "N",
26    "From": "utsavsingh085@gmail.com",
27    "Username": "utsavsingh085@gmail.com",
28    "Password": "pnmhysmdigjxyd",
29    "Host": "smtp.gmail.com",
30    "Port": 587
31  }
32 }
```

Connected to the database using the below details.



Connect to Server

SQL Server

Server type: Database Engine

Server name: database-1.cpvpuwxaoijn.ap-southeast-1.rds.amazonaws.

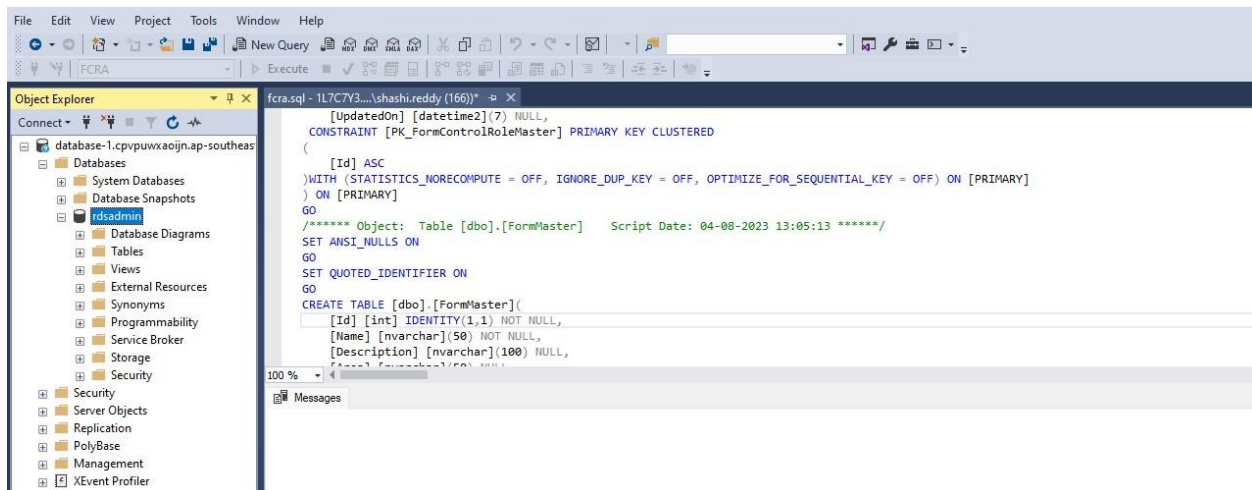
Authentication: SQL Server Authentication

Login: admin

Password:

☐ Remember password

Connect Cancel Help Options >>



```
fcra.sql - 1L7C7Y3...\\shashi.reddy (166)*
[UpdatedOn] [datetime2] (7) NULL,
CONSTRAINT [PK_FormControlRoleMaster] PRIMARY KEY CLUSTERED
(
    [Id] ASC
)WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/***** Object: Table [dbo].[FormMaster]    Script Date: 04-08-2023 13:05:13 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[FormMaster](
    [Id] [int] IDENTITY(1,1) NOT NULL,
    [Name] [nvarchar](50) NOT NULL,
    [Description] [nvarchar](100) NULL,
    [UpdatedOn] [datetime2](7) NULL,
    CONSTRAINT [PK_FormControlRoleMaster] PRIMARY KEY CLUSTERED
    (
        [Id] ASC
    ) ON [PRIMARY]
) ON [PRIMARY]
GO
```

