Contents

[1 Pre-Requisite: 2](#_Toc41671352)

[2 Technology Stack 2](#_Toc41671353)

[3 Architecture 3](#_Toc41671354)

[4 Code Deployment 3](#_Toc41671355)

[5 Database 8](#_Toc41671356)

[6 ECS Cluster using EC2: 13](#_Toc41671357)

[6.1 Creating Cluster 13](#_Toc41671358)

[6.2 Console view 17](#_Toc41671359)

[6.2.1 EC2 17](#_Toc41671360)

[6.2.2 ENI 18](#_Toc41671361)

[6.2.3 Security Groups 19](#_Toc41671362)

[6.2.4 Subnets 19](#_Toc41671363)

[6.2.5 Route Tables 19](#_Toc41671364)

[7 Create Task Definition 20](#_Toc41671365)

[8 Creating Service in Cluster 24](#_Toc41671366)

[8.1 Cluster Created 36](#_Toc41671367)

[9 Application Run 36](#_Toc41671368)

[10 SNS Integration 36](#_Toc41671369)

[11 CloudWatch Integration 37](#_Toc41671370)

[12 Cost Calculator 37](#_Toc41671371)

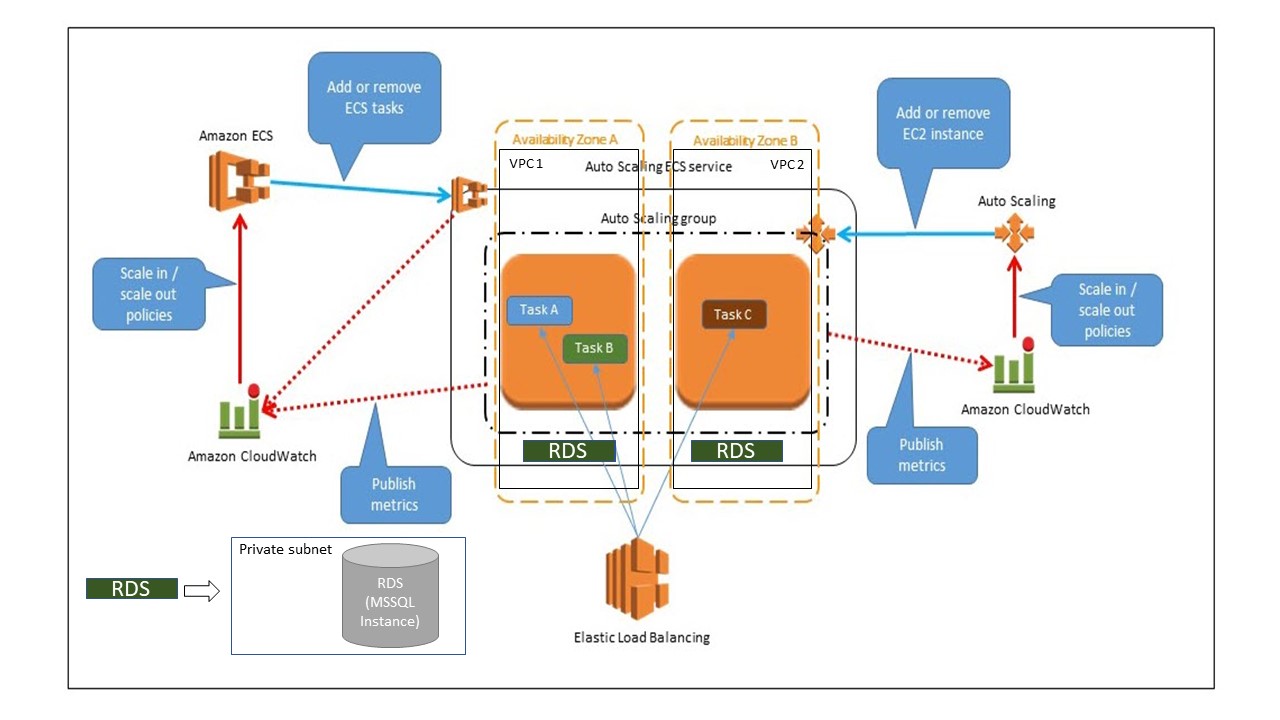
# Pre-Requisite:

* Aws Toolkit
* Visual Studio 2019
* Docker Desktop For Windows
* AWS Account

# Technology Stack

* Asp.Net Core technology will be one of the best to use as core framework supports containerizations for all platforms such as windows, linux and macOS.
* .Net Core is build to live in container.
* Also, micro services architecture could be implemented using this core framework.

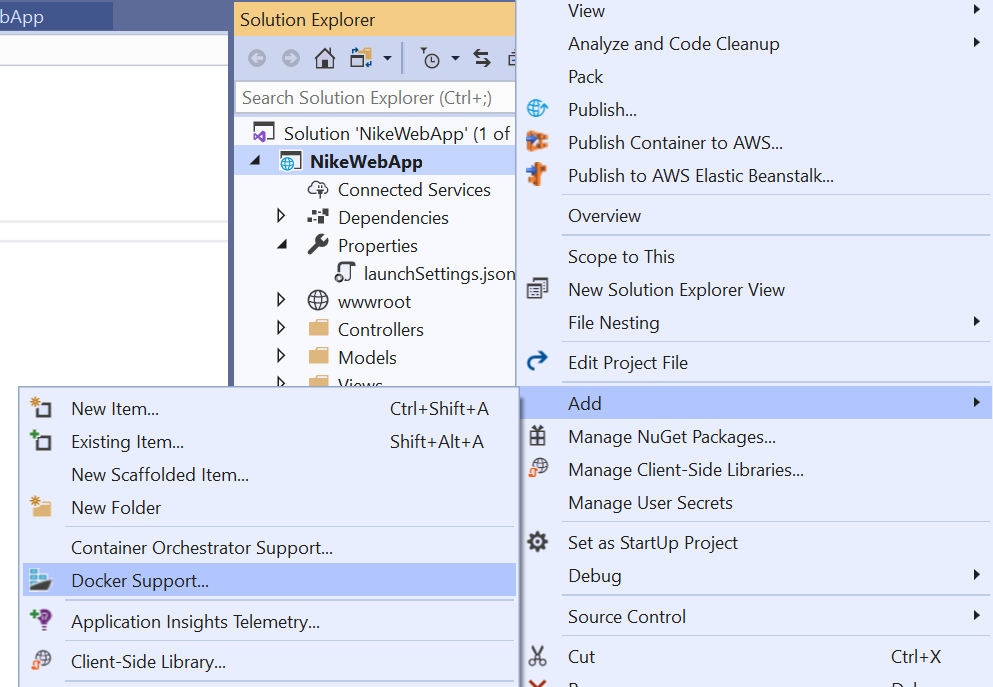
# Architecture

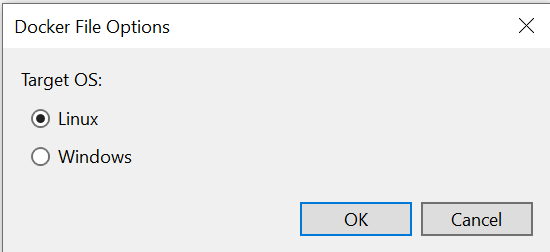


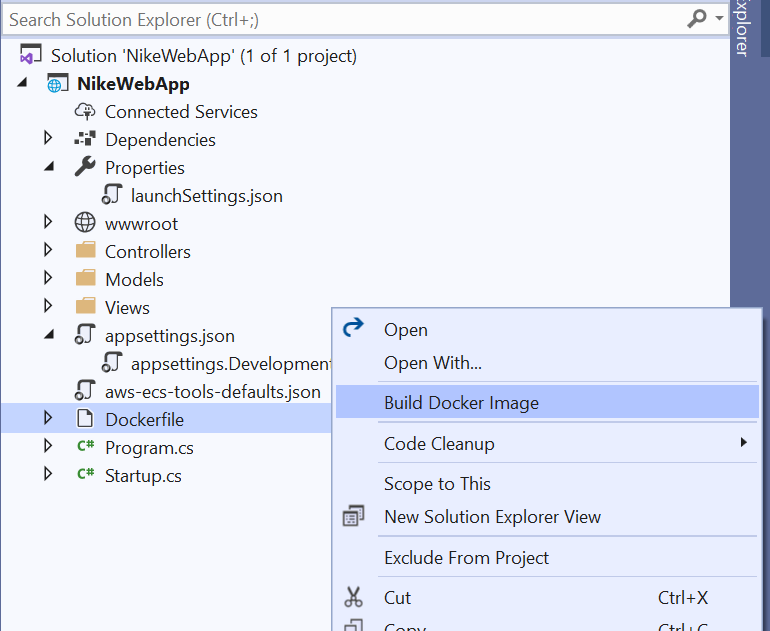
# Code Deployment

Following are the steps to create deployments:

1. Add Docker support to WebApp project:

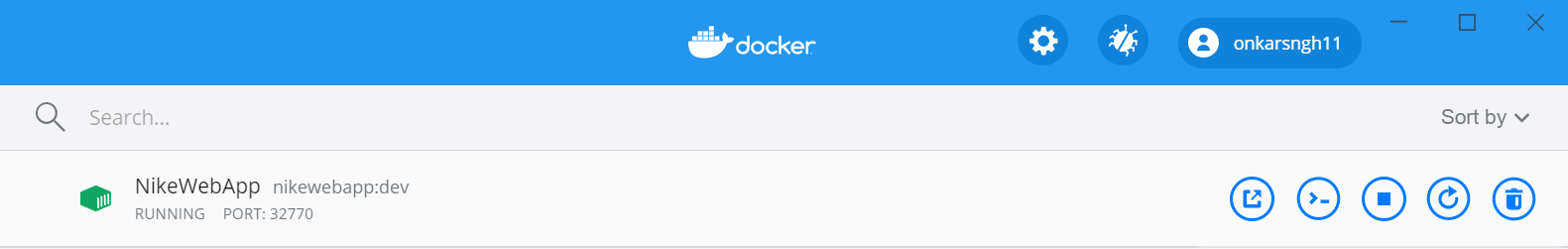


1. Select Linux:   
     
     
   This will generate a DockerFIle.
2. Build Docker Image:

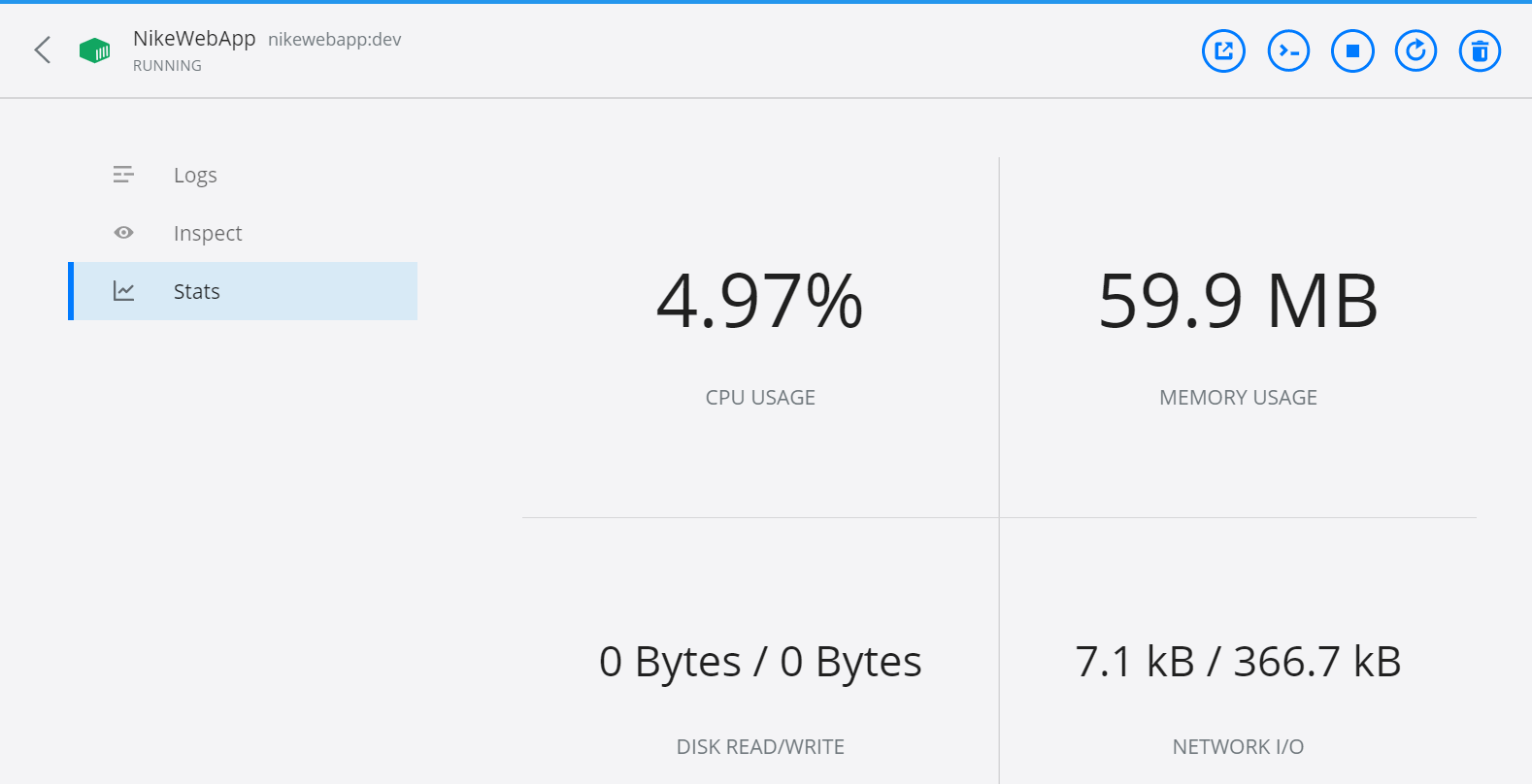


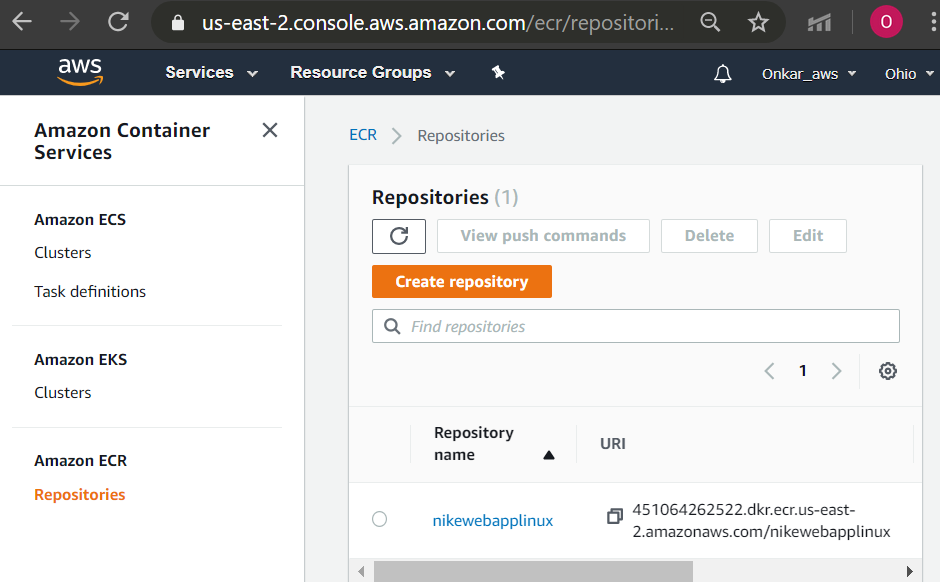
1. Select Docker Profile and Run the solution to verify if it’s showing up in docker desktop dashboard.

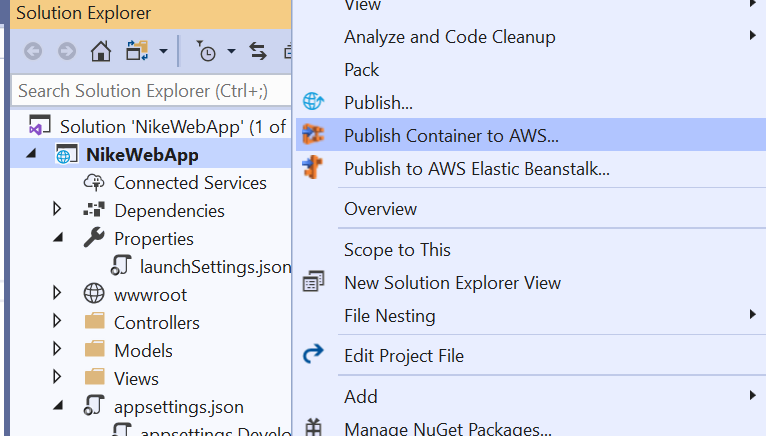


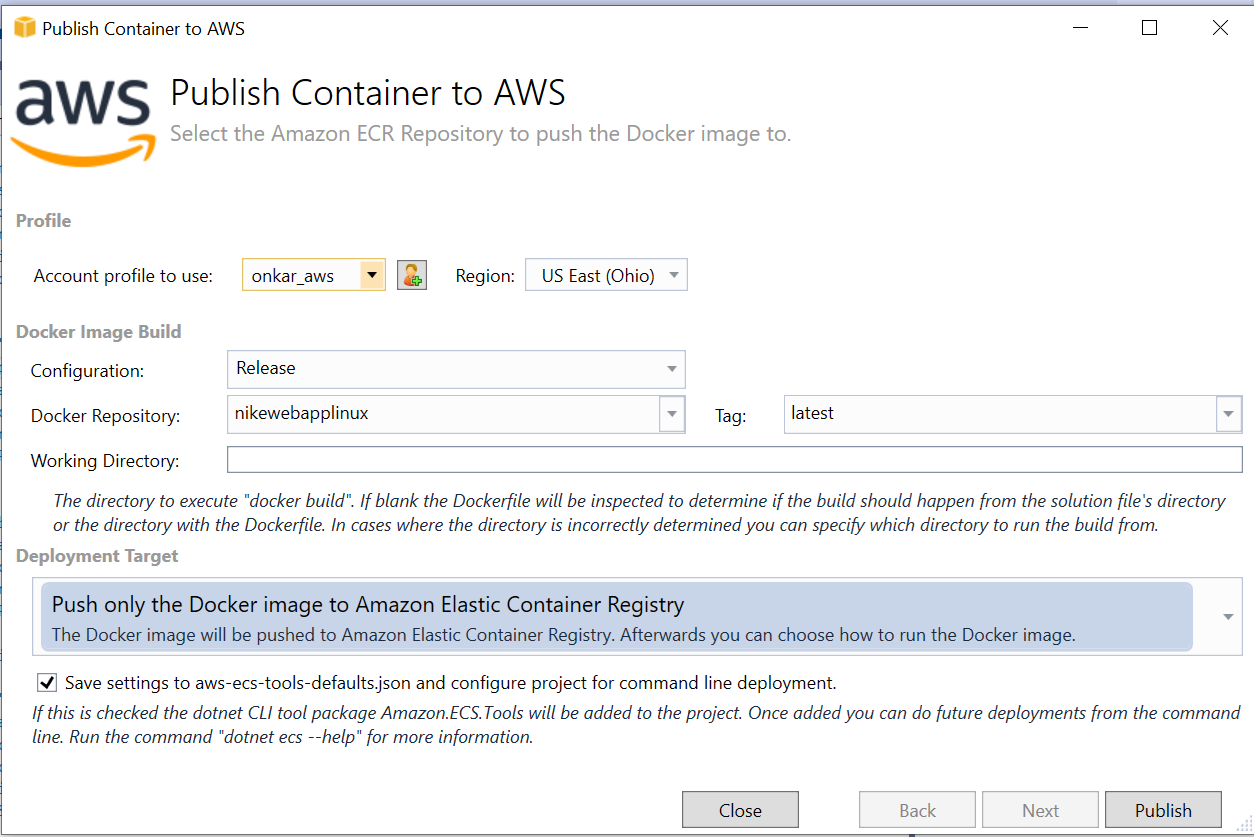
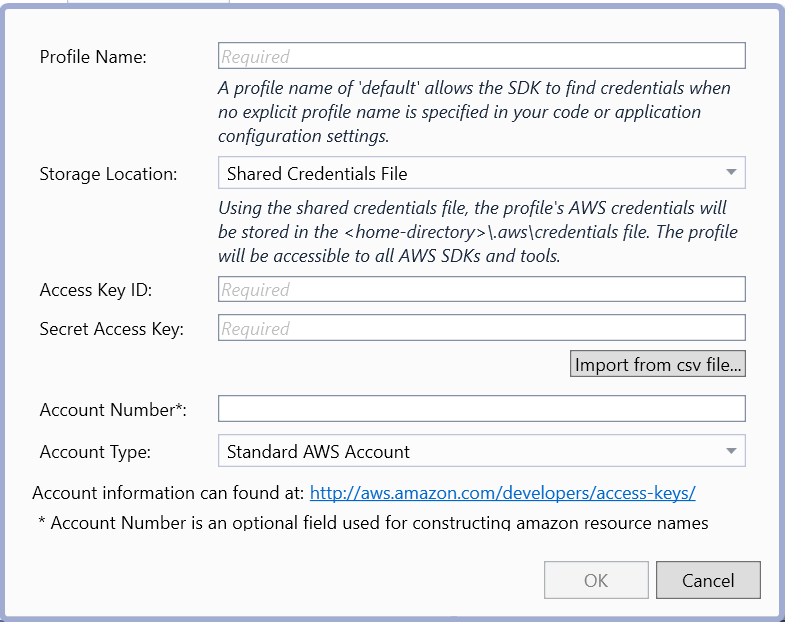
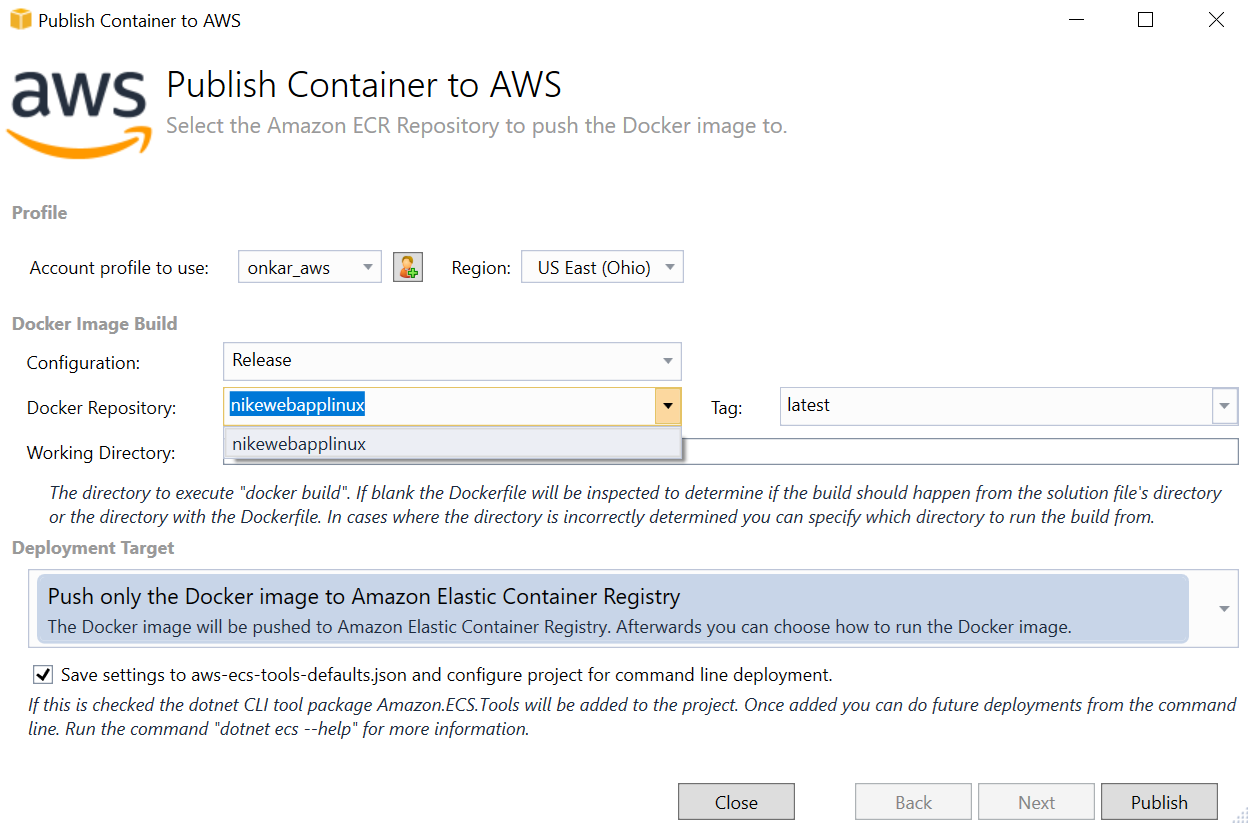


1. Check for resource usage for estimation by clicking on this instance and going to Stats. This will provide a general idea of how much hardware resources application will use.



1. Create ECR Repository from aws console:  
     
   
2. Now, right Click on WebApp:



1. From dialog box, click user icon:  
     
   
2. Enter your AWS account details:  
     
   
3. Select available ECR repository previously created and select “Push Only the Docker Image to Amazon ECR” in deployment target.  
     
   
4. Click Publish.

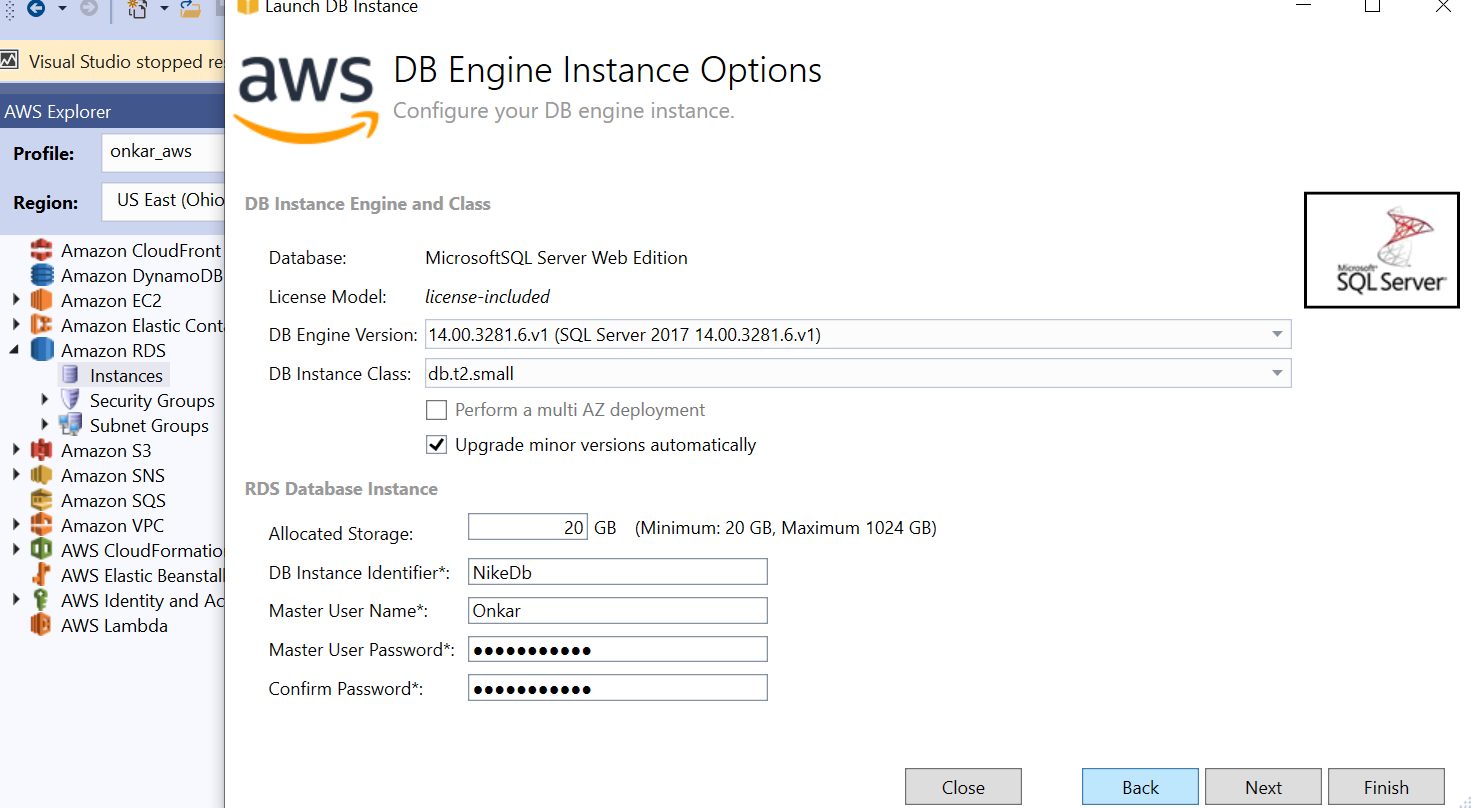
**Key Points:**

1. Visual studio 2019 along with AWS Toolkit, makes deployment easier and more manageable by generating json configuration of deployment for future use
2. Docker Support adds “Dockerfile” which can be used and modified according to need.
3. ECR Repository is ideal for storing Docker images as it is primarily made for that purpose.

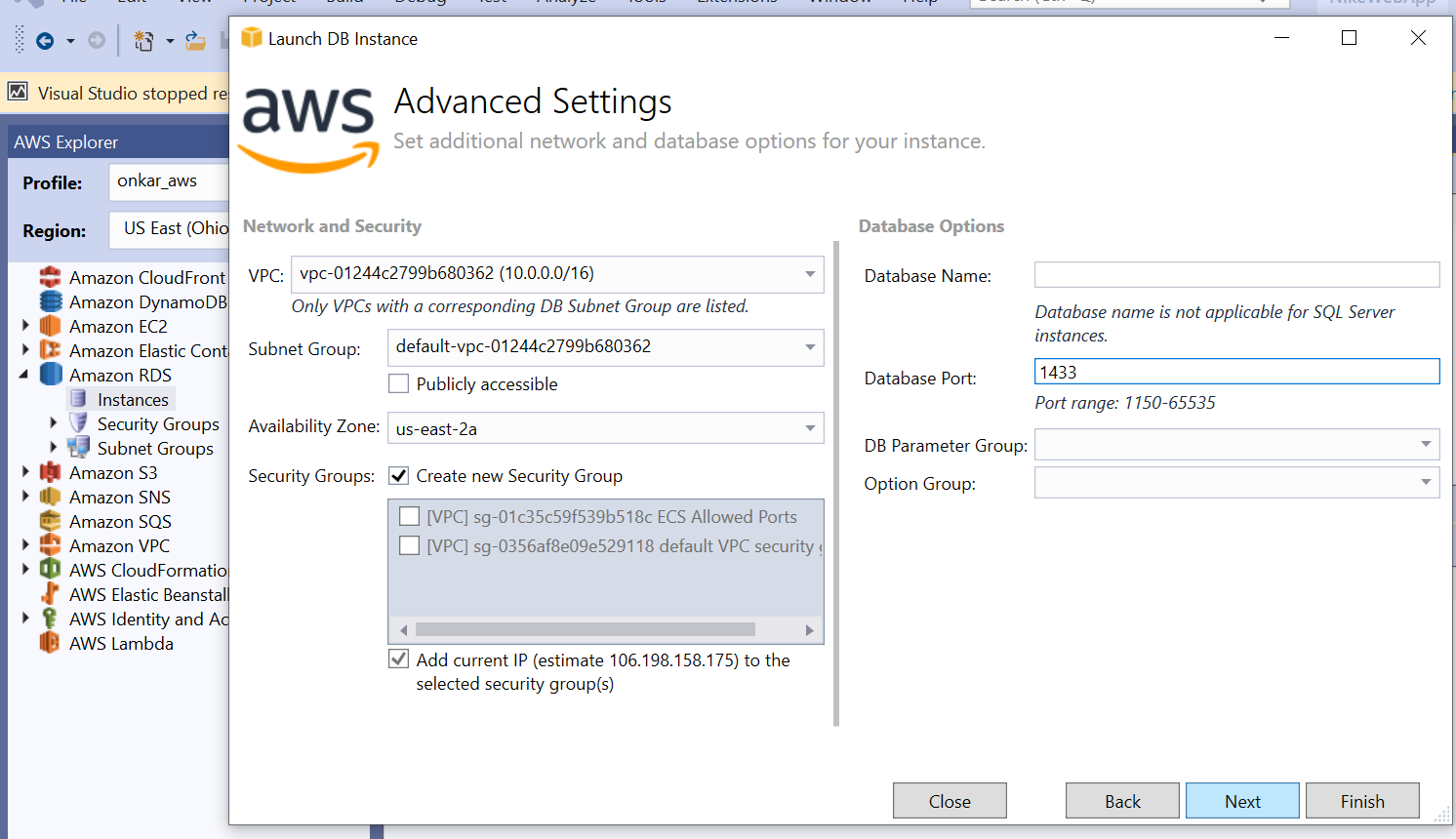
# Database

**RDS** **:-** The Amazon Relational Database Service (**RDS AWS**) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, re-sizable capacity in an industry-standard relational database and manages common database administration tasks.

1. Select Amazon RDS -> Launch instance ->SQL WebServer instance



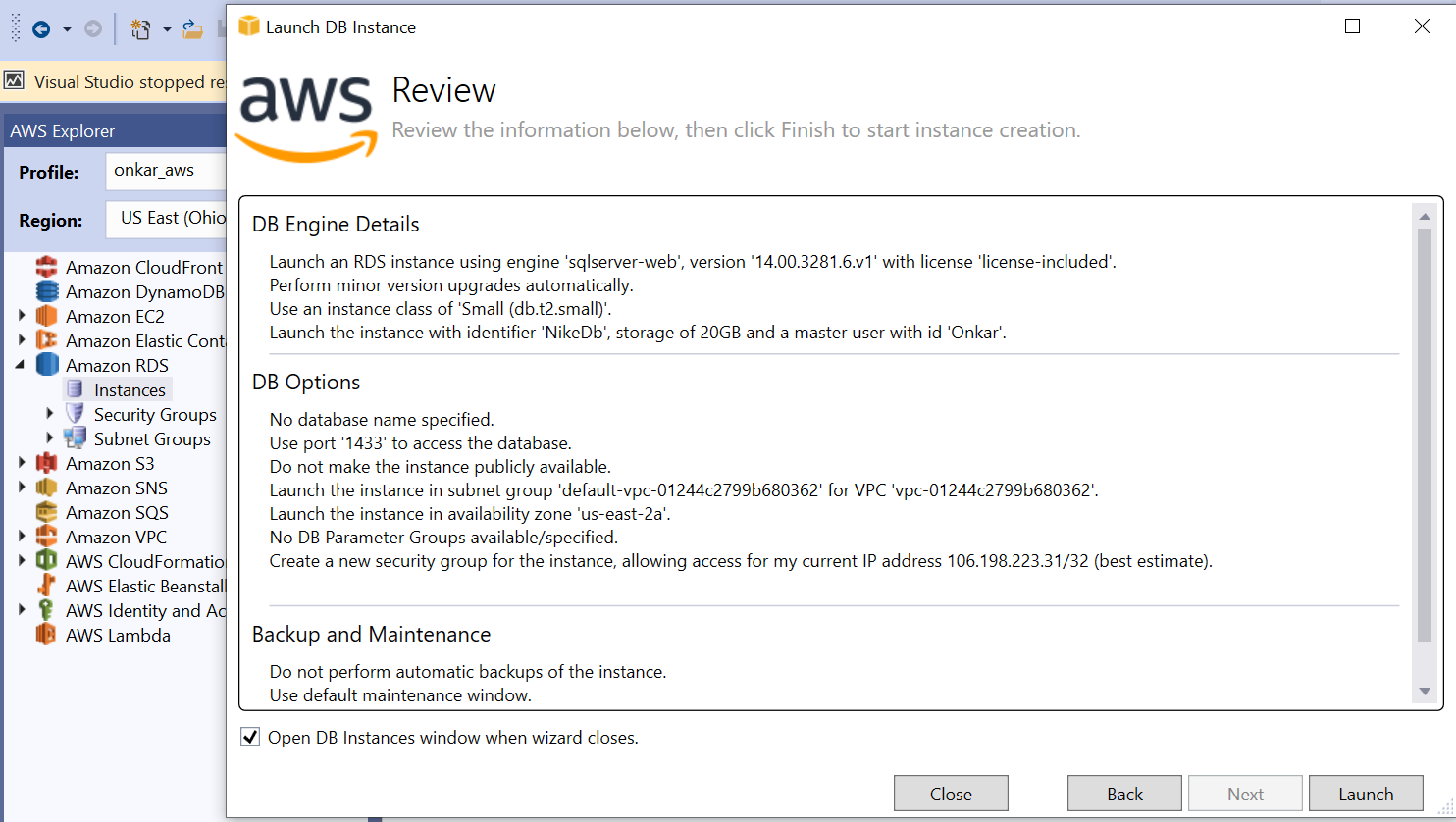
1. Click Next



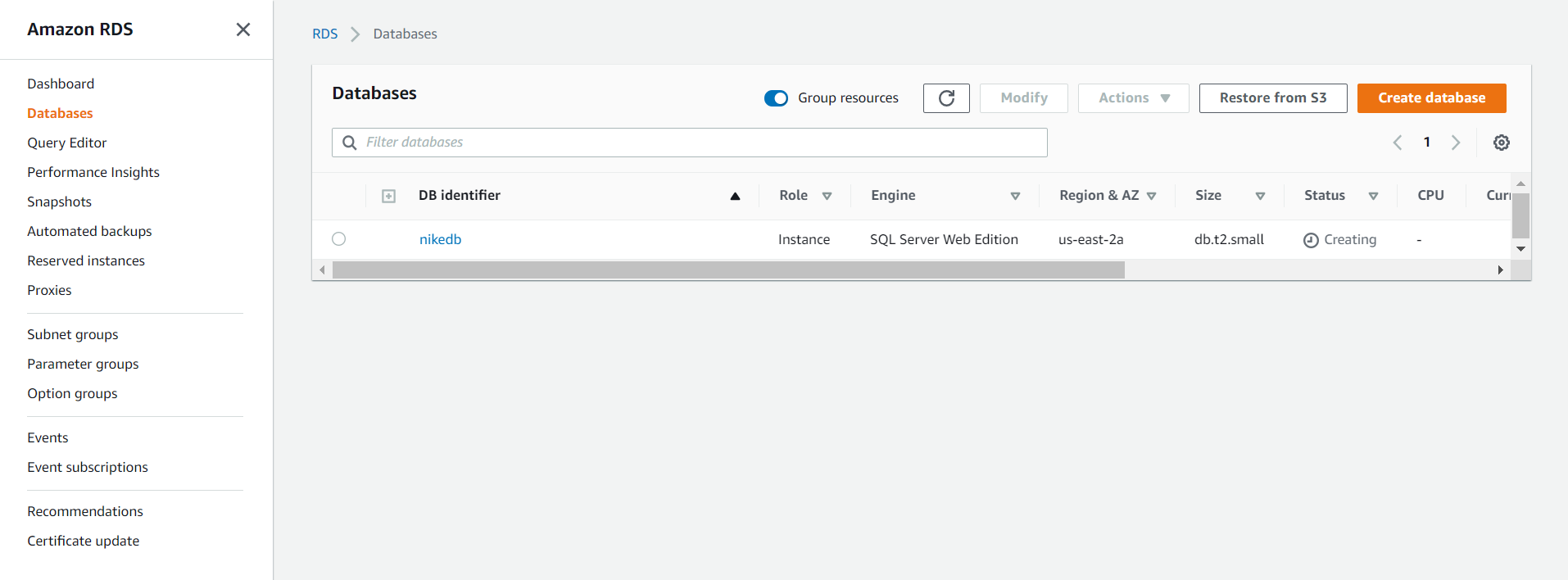
1. Click next



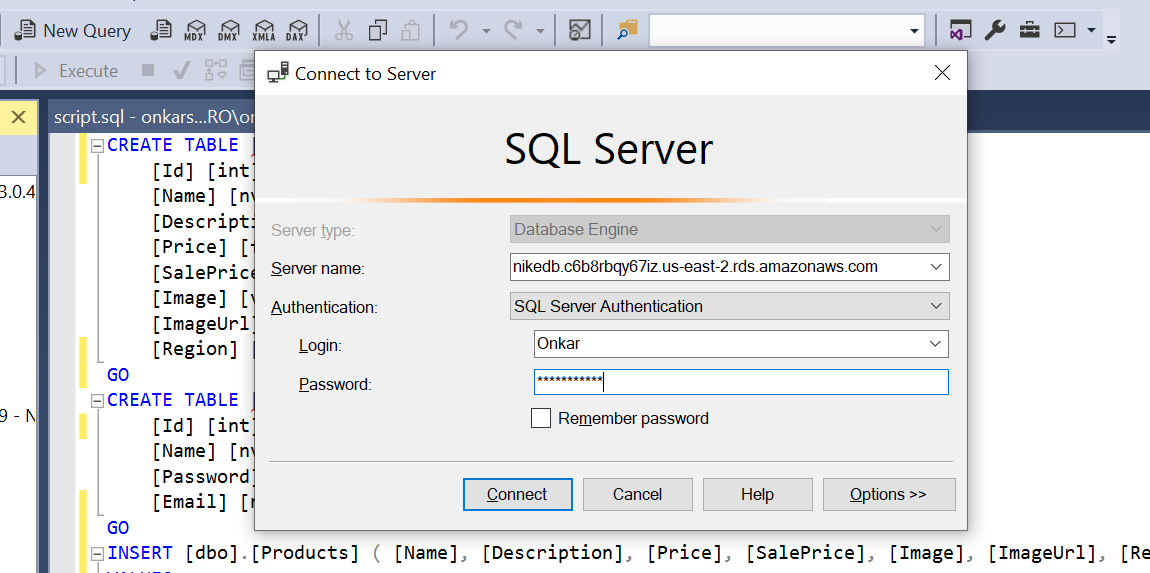
1. Click next



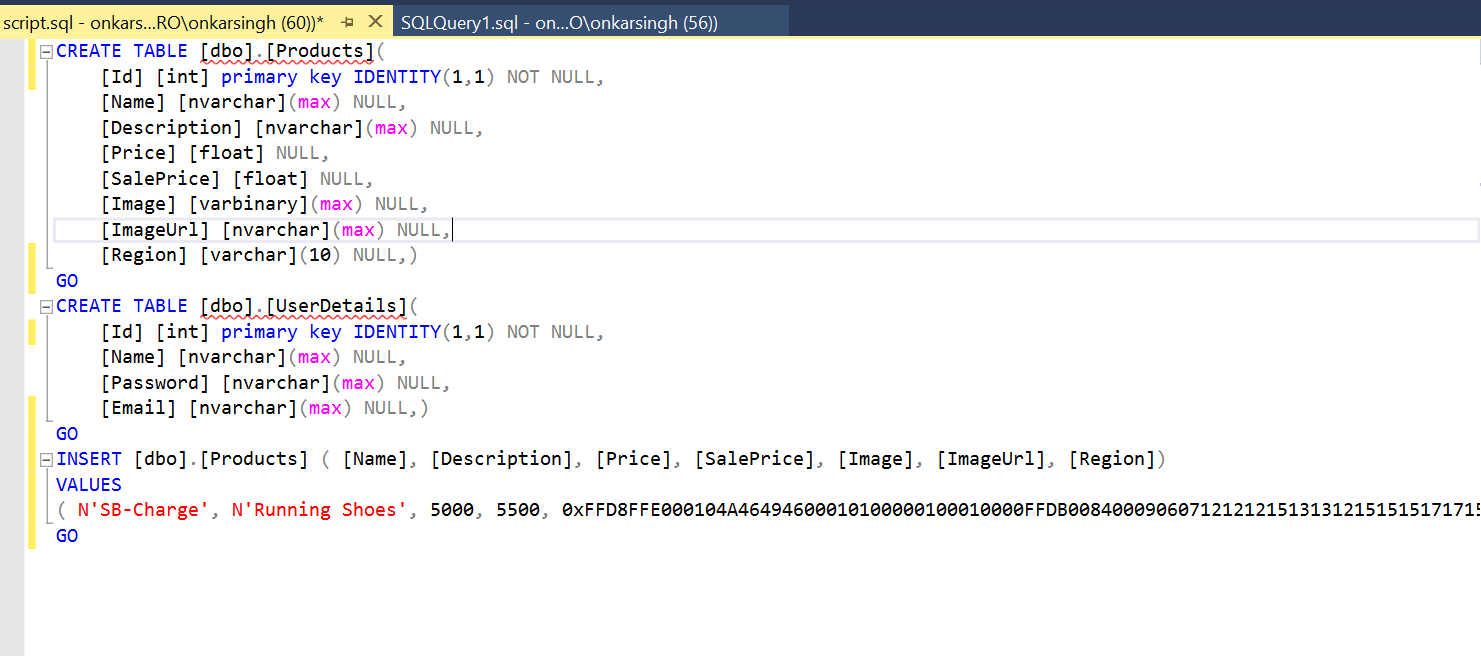
1. Verify in console



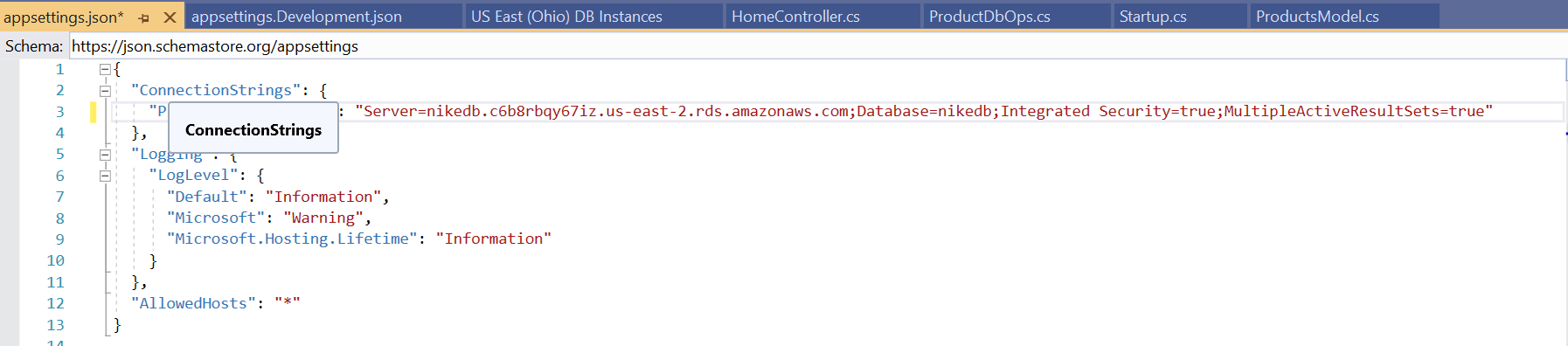
1. Create SQL Server Db Instance and connect with SQL and Credentials:



1. Run required scripts:

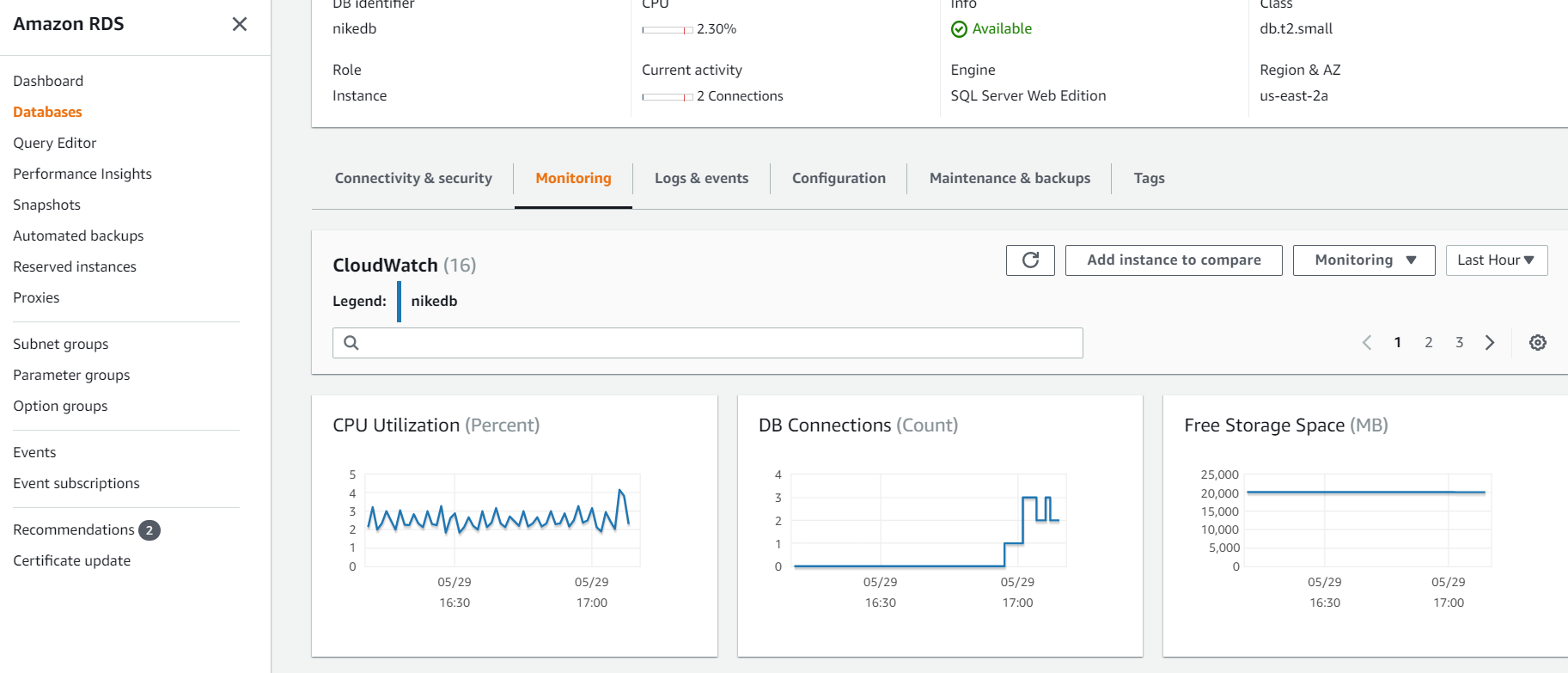


1. Add connection string to appsettings.json:



For actual working, I used my master username and password.

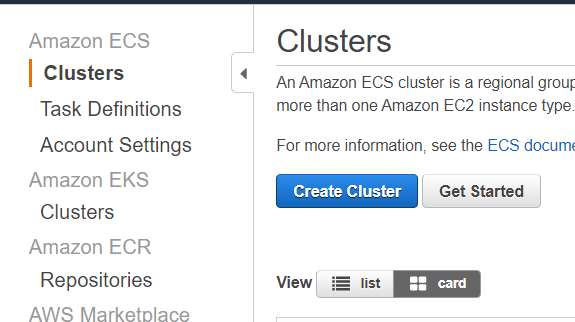
1. CloudWatch integration for RDS:

Video referred : <https://www.youtube.com/watch?v=z-N0z5K_WFI>

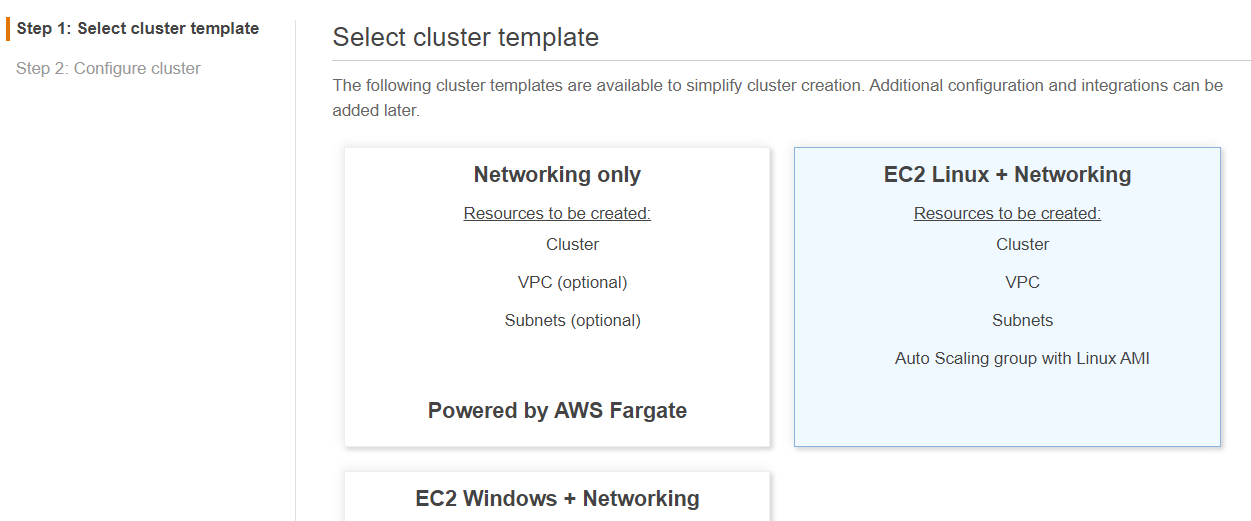
# ECS Cluster using EC2:

## Creating Cluster

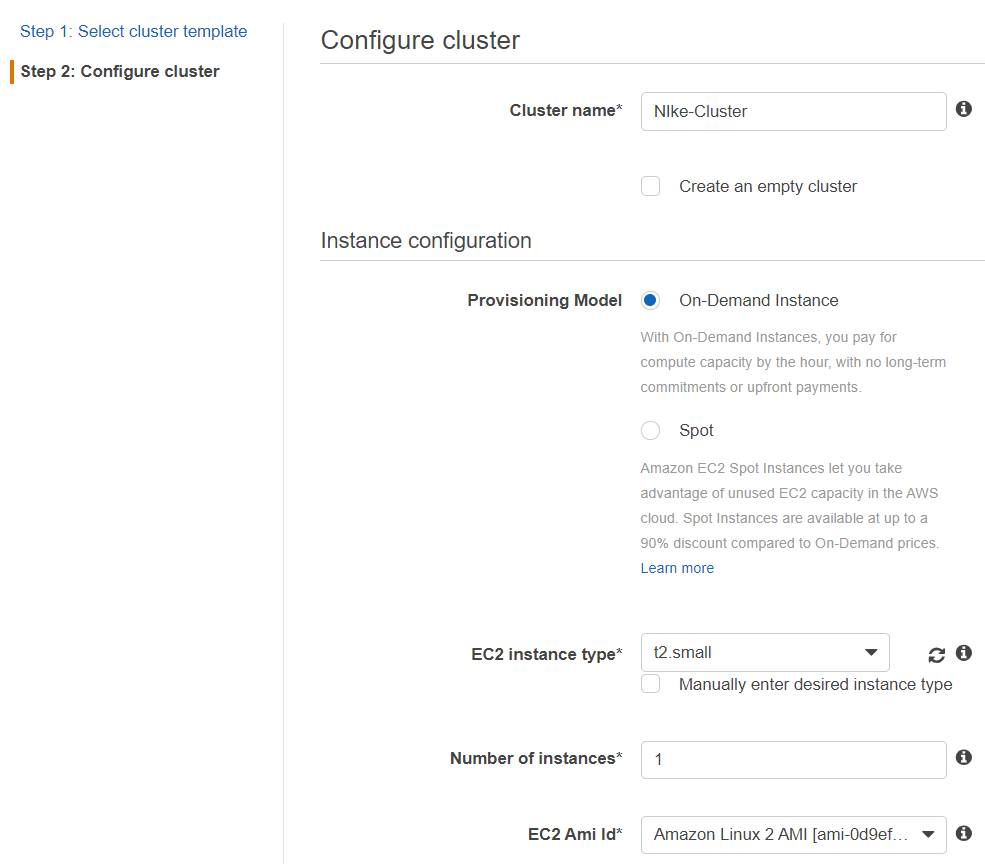
1. In new tab, Create Cluster from ECS home screen:

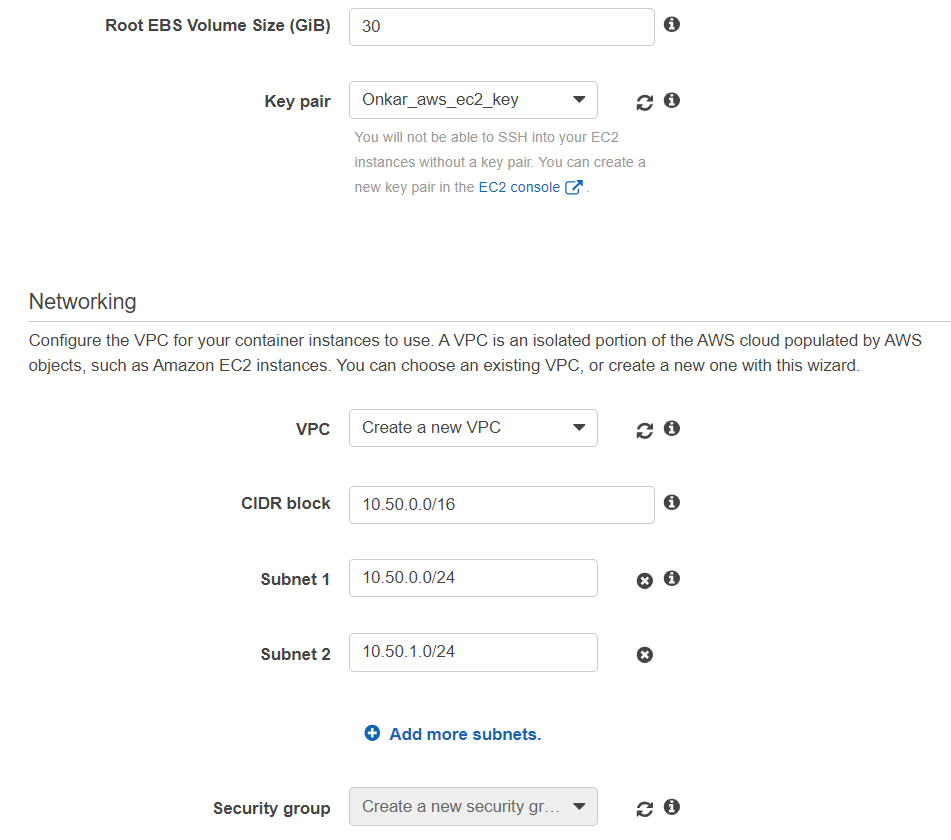


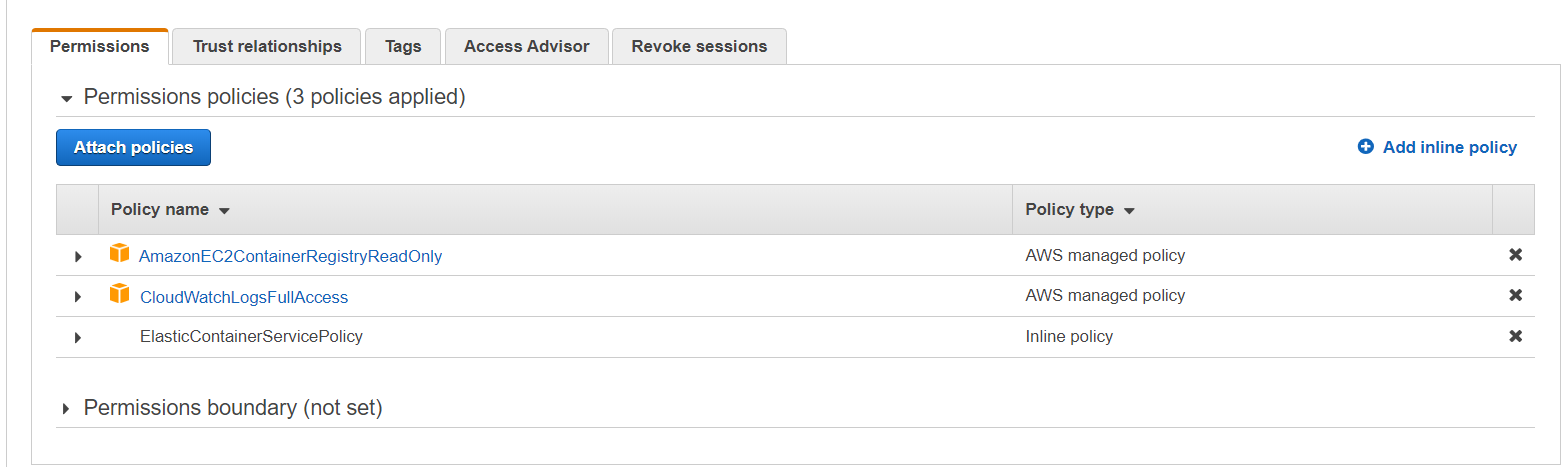
1. Select highlighted:

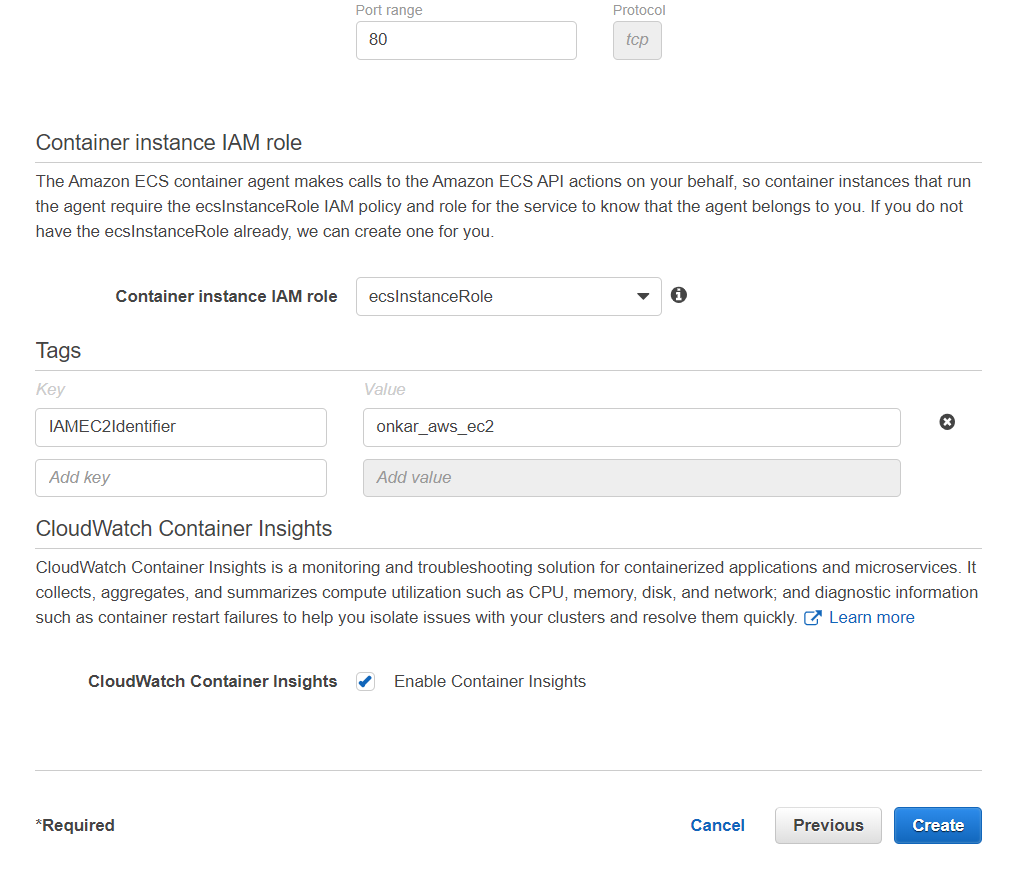


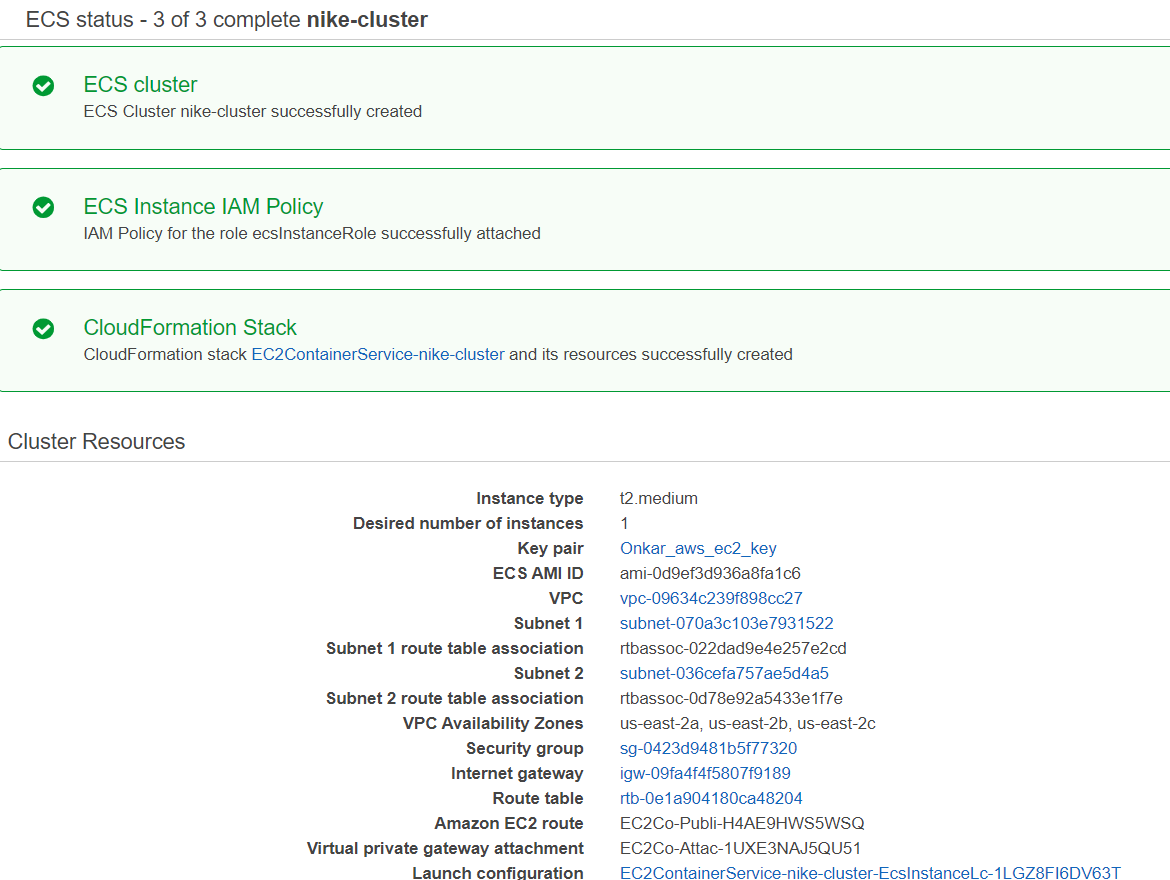
1. Specify Configurations as shown:





1. Added IAM Role and attached policy  
   



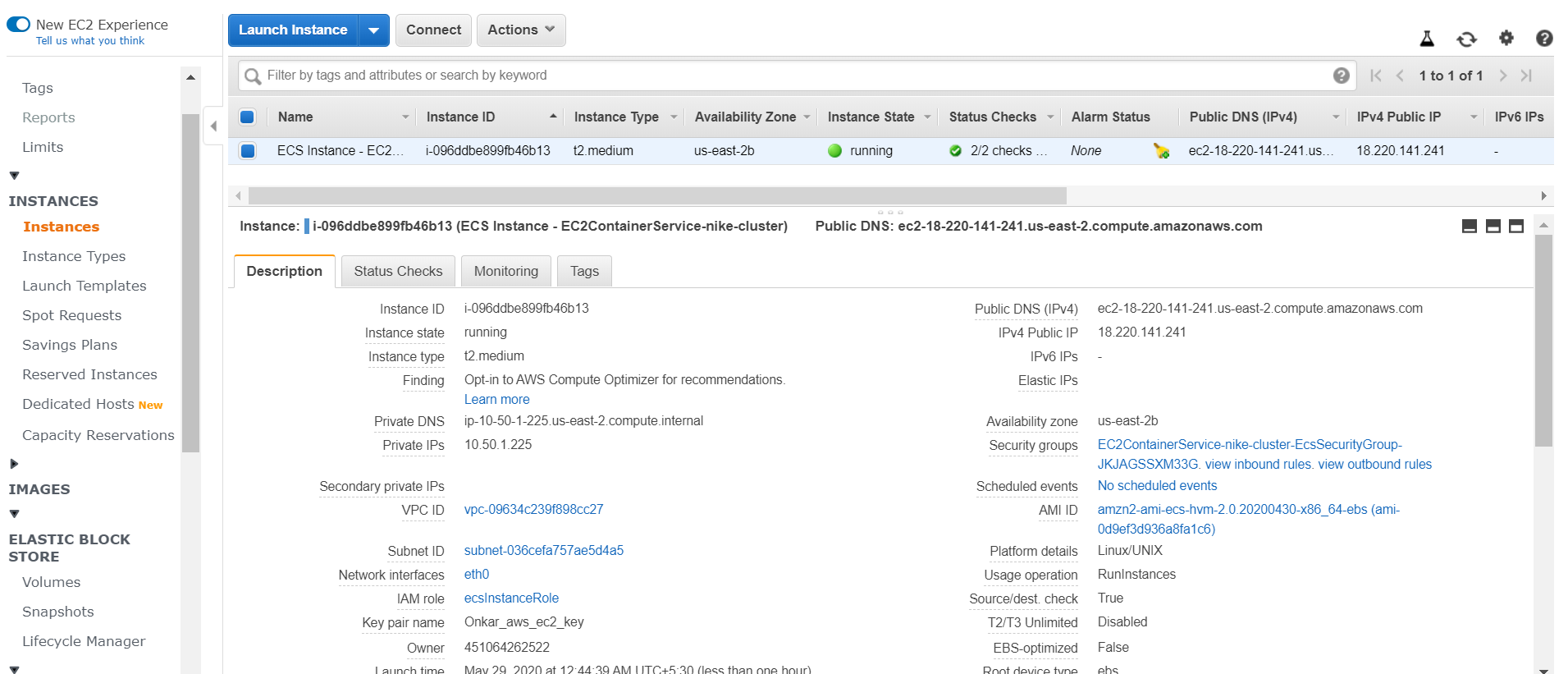
1. Create Cluster:  
     
   

## Console view

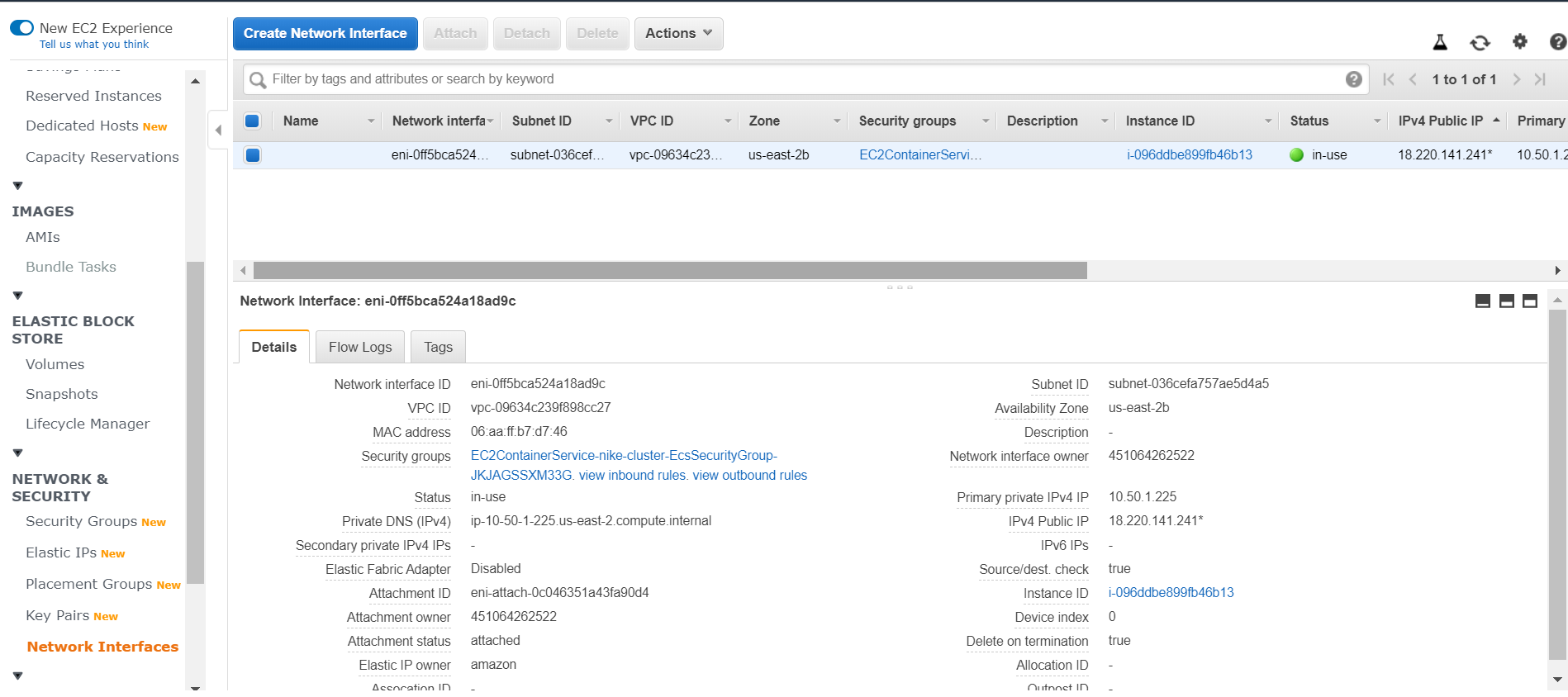
After creating Cluster ec2, VPC, Subnets, Security Groups are created automatically as shown below:

### EC2

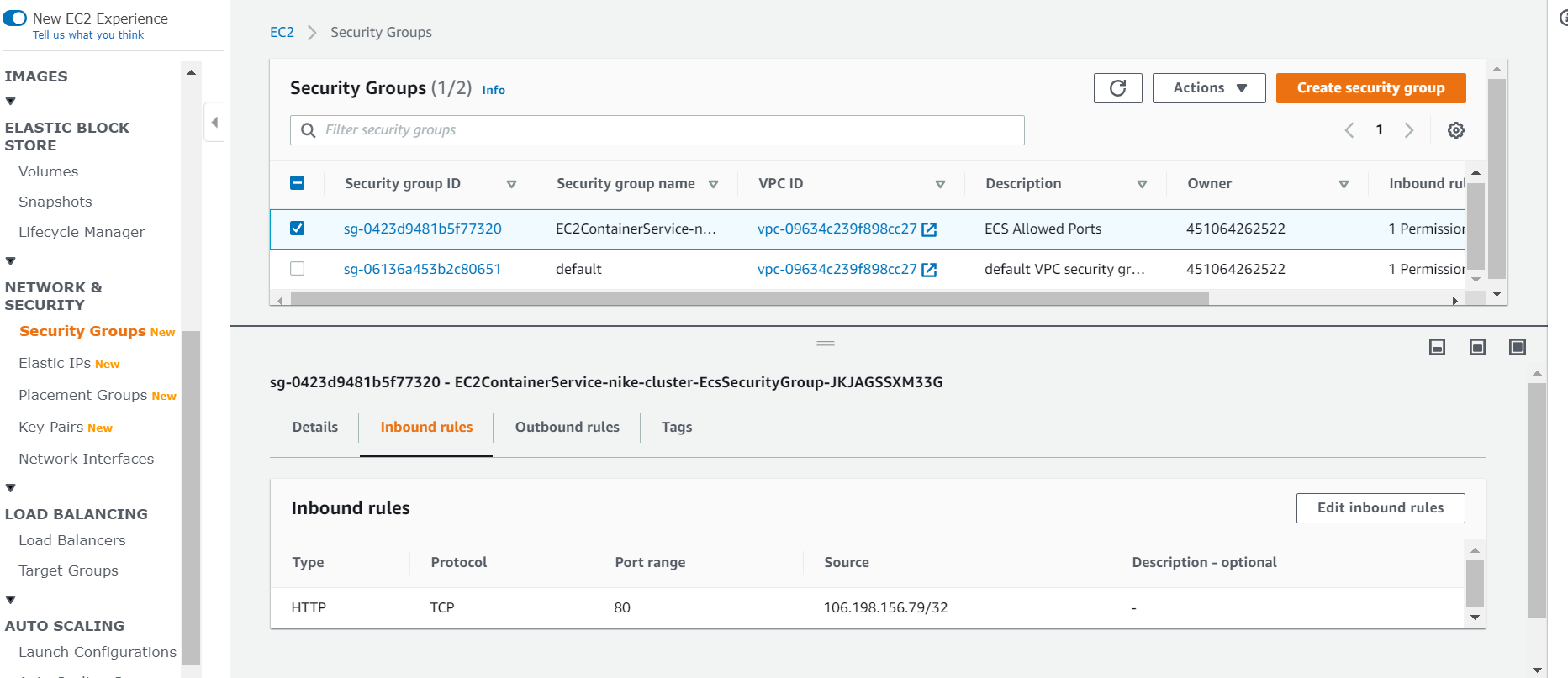
**EC2 Instance:-** Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, scalable compute capacity in the cloud.



### ENI

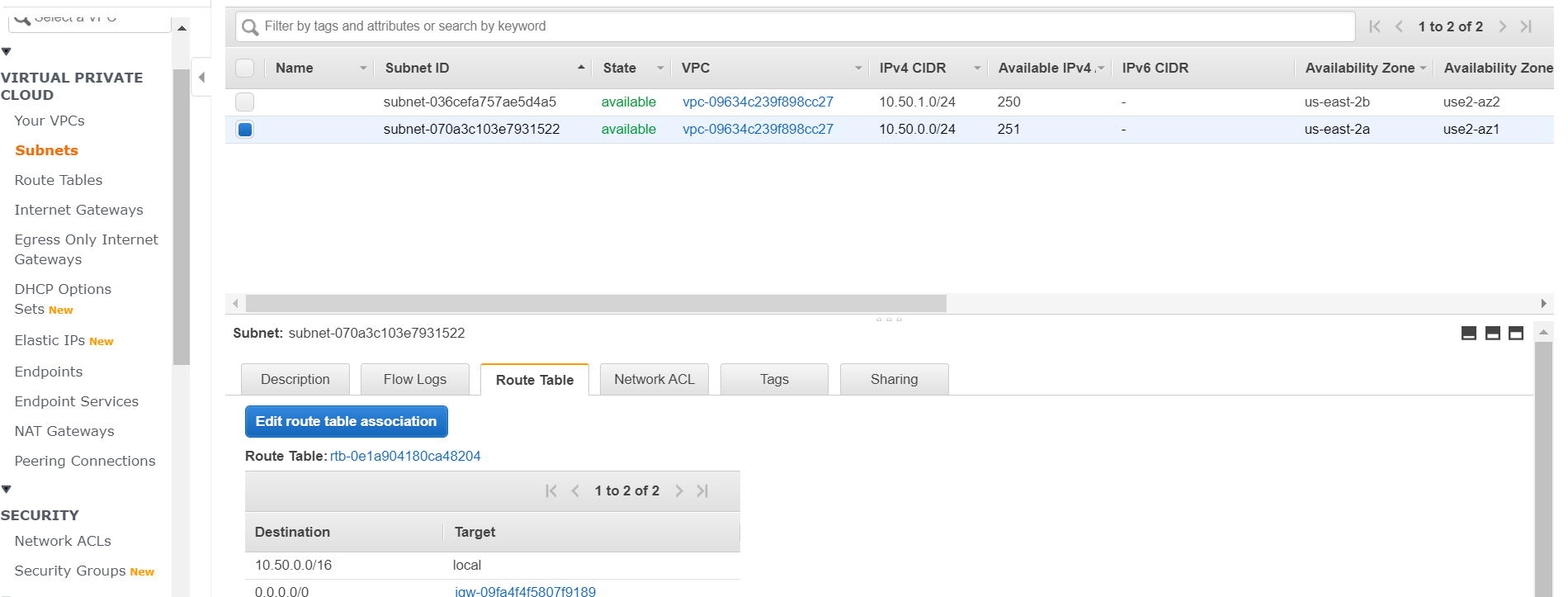


### Security Groups

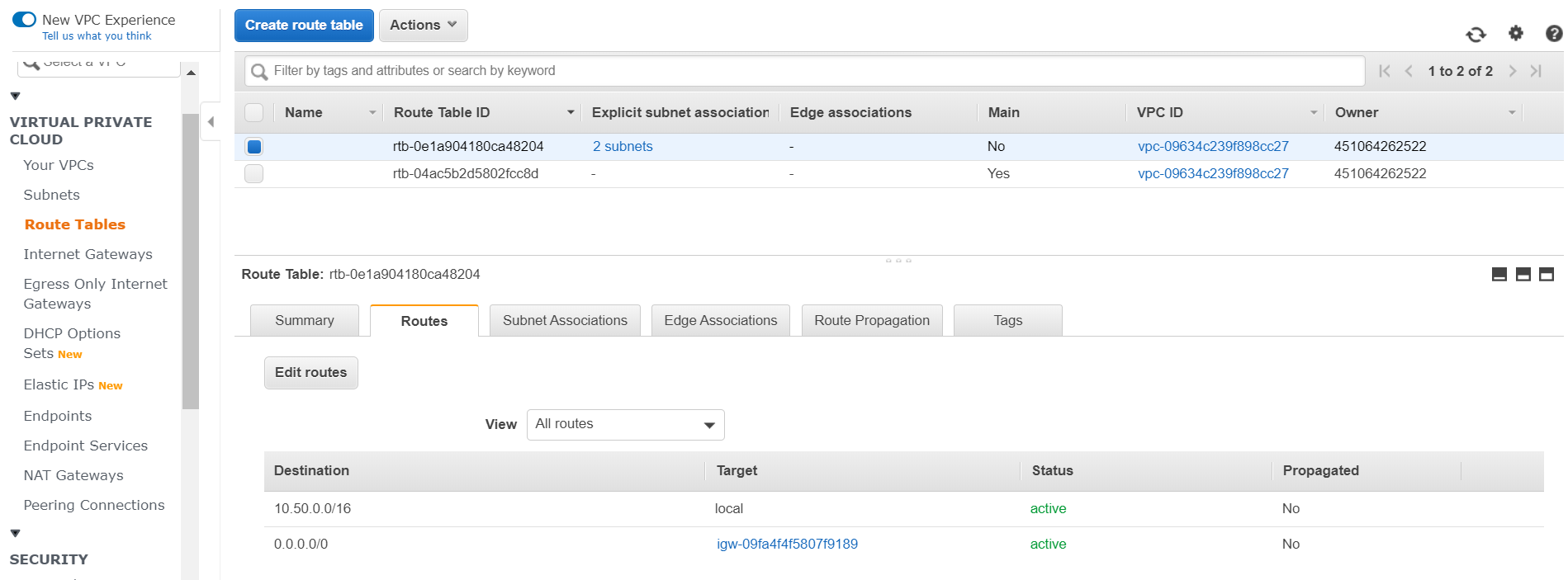


### Subnets

Subnets need to be configured one for private and one for public. This is done by routinas follows:

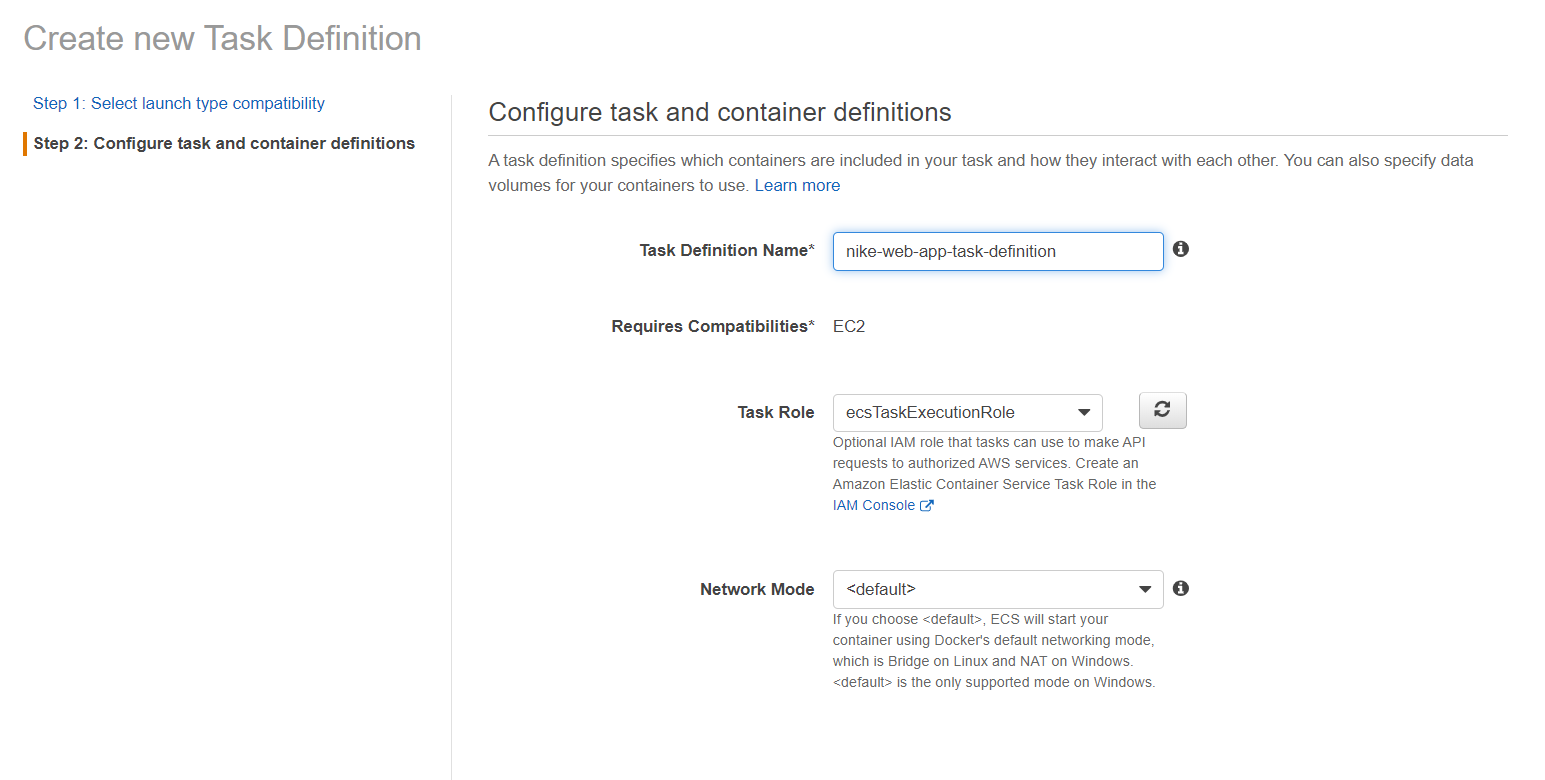


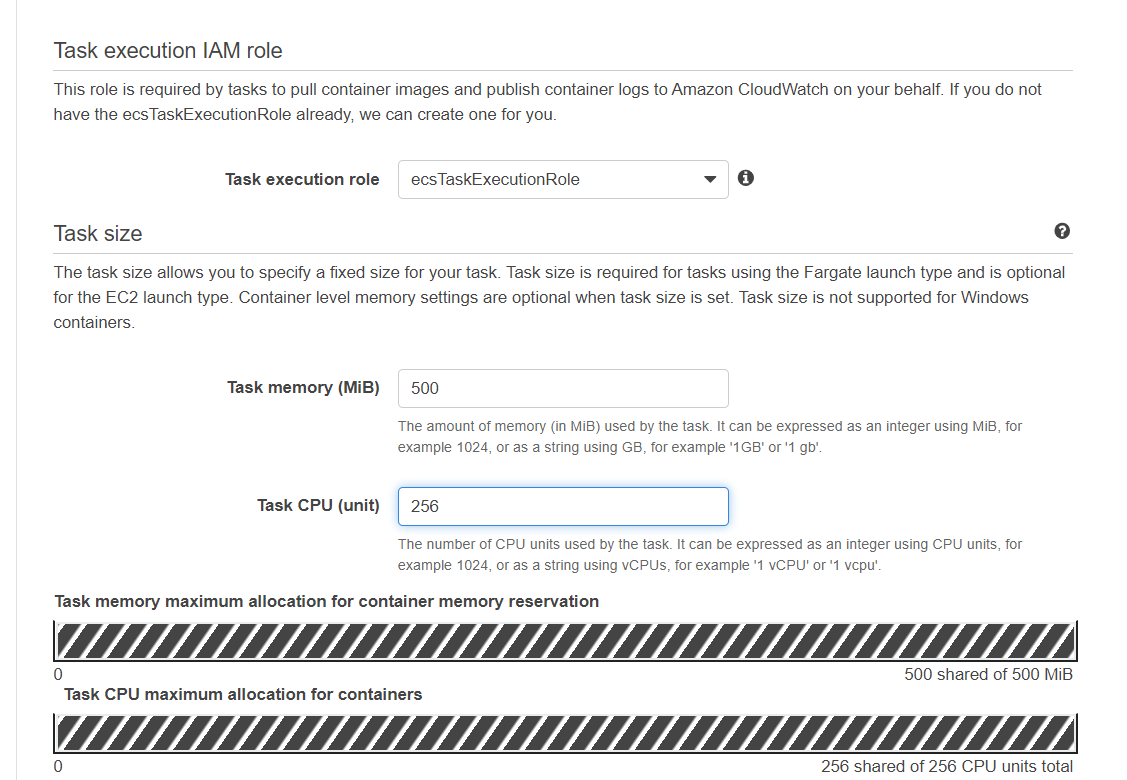
### Route Tables

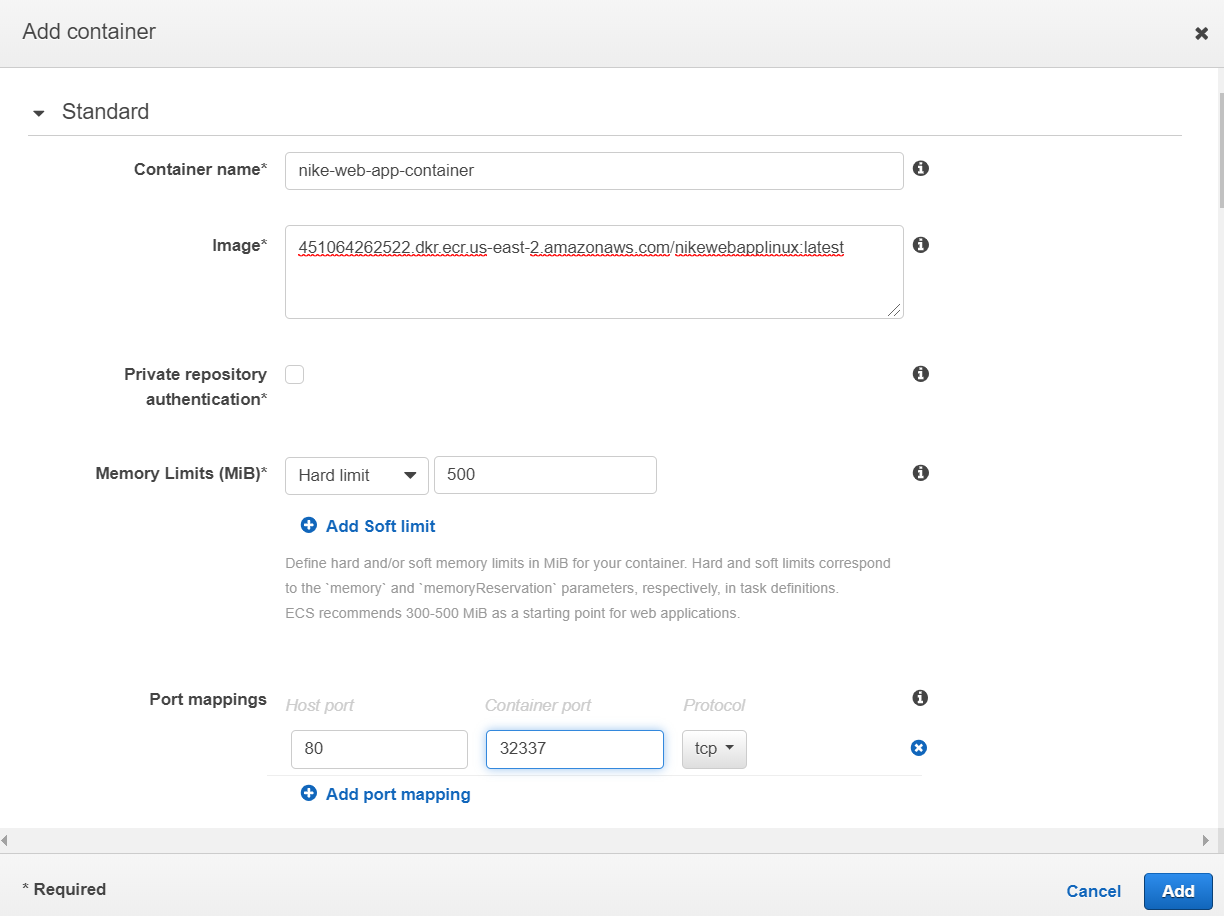


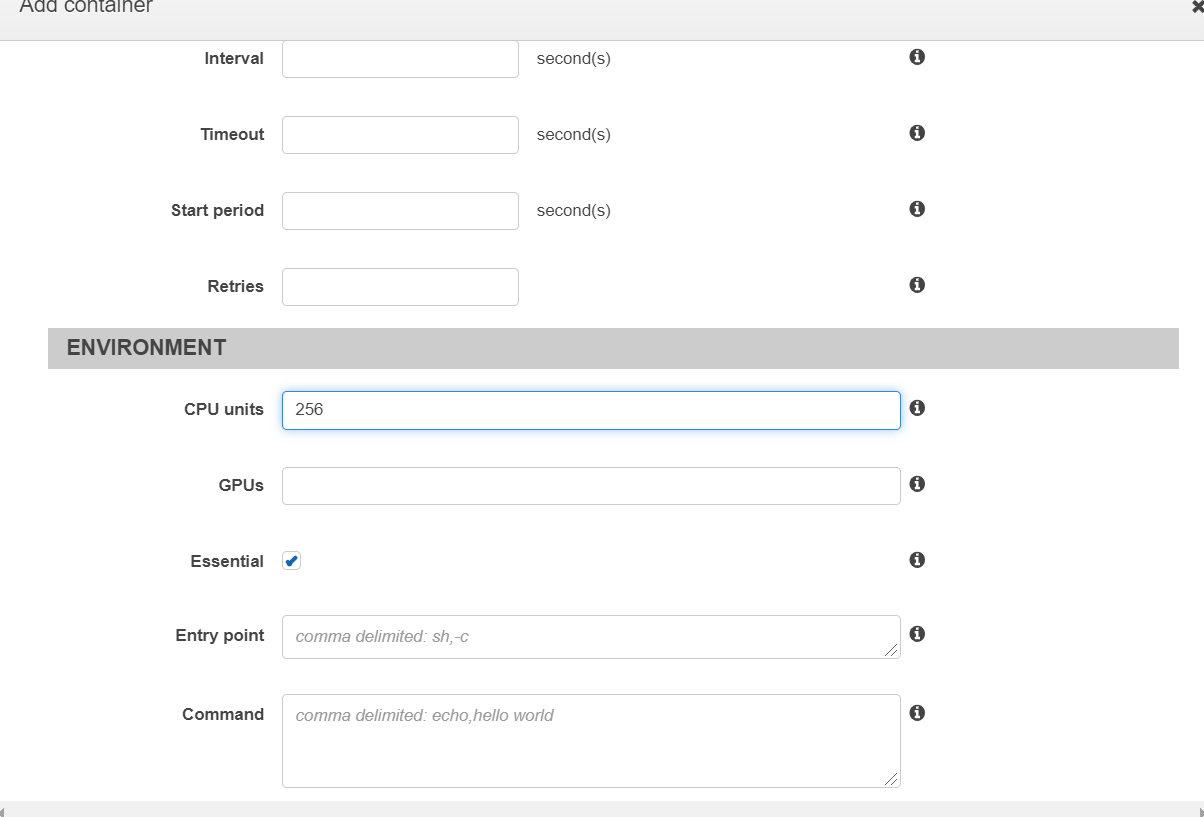
# Create Task Definition

Following are the steps to create task definition:

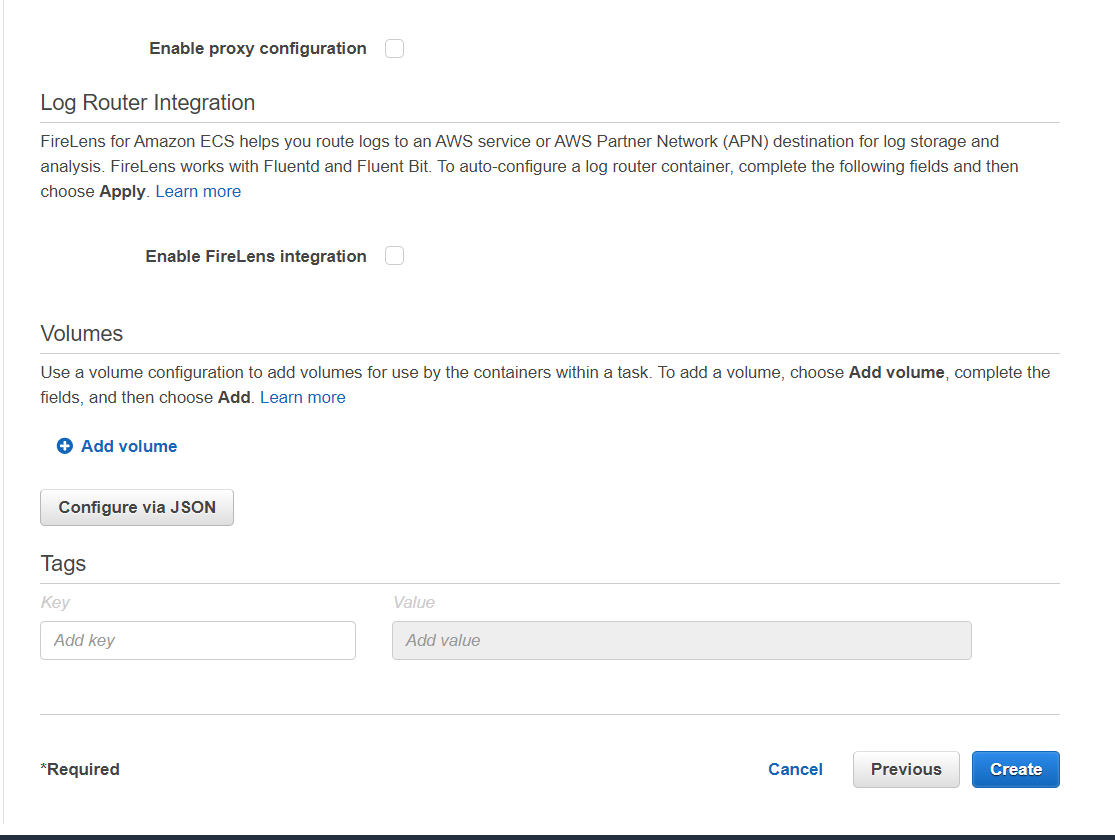






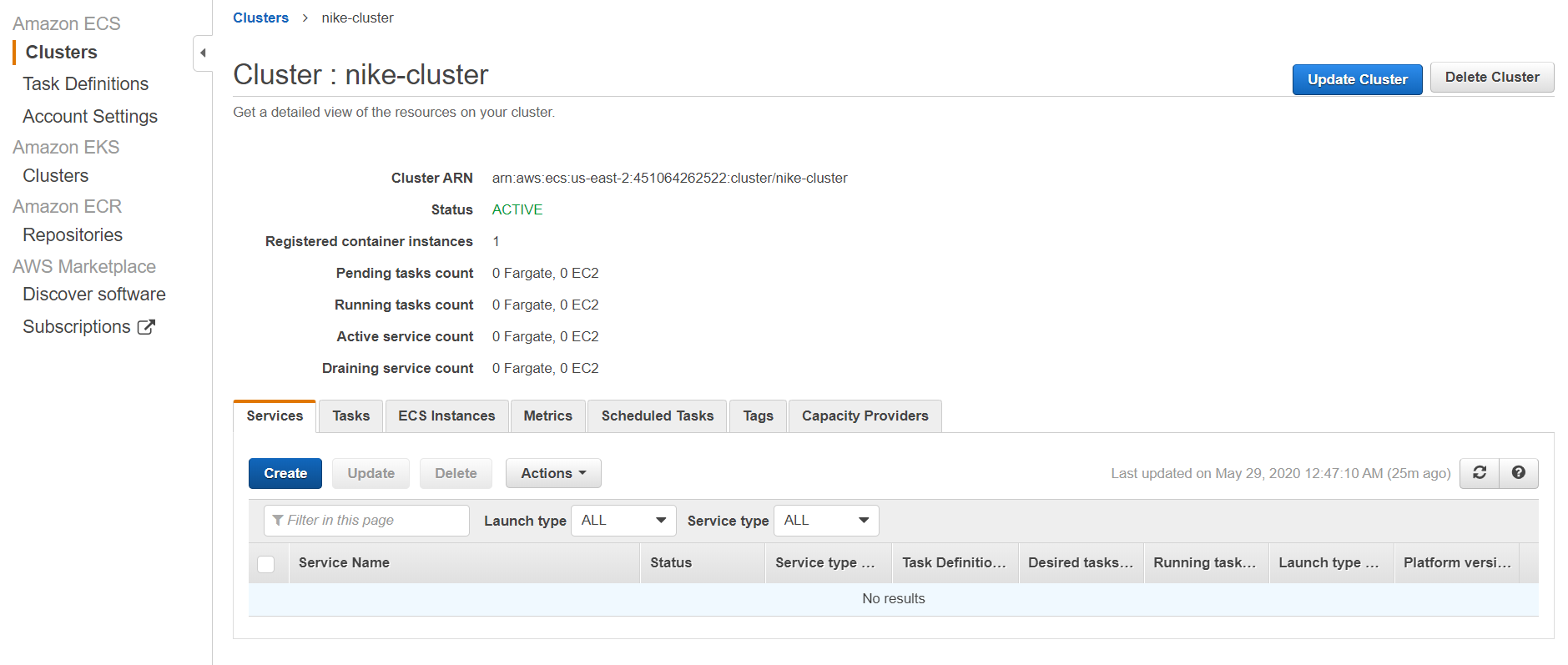




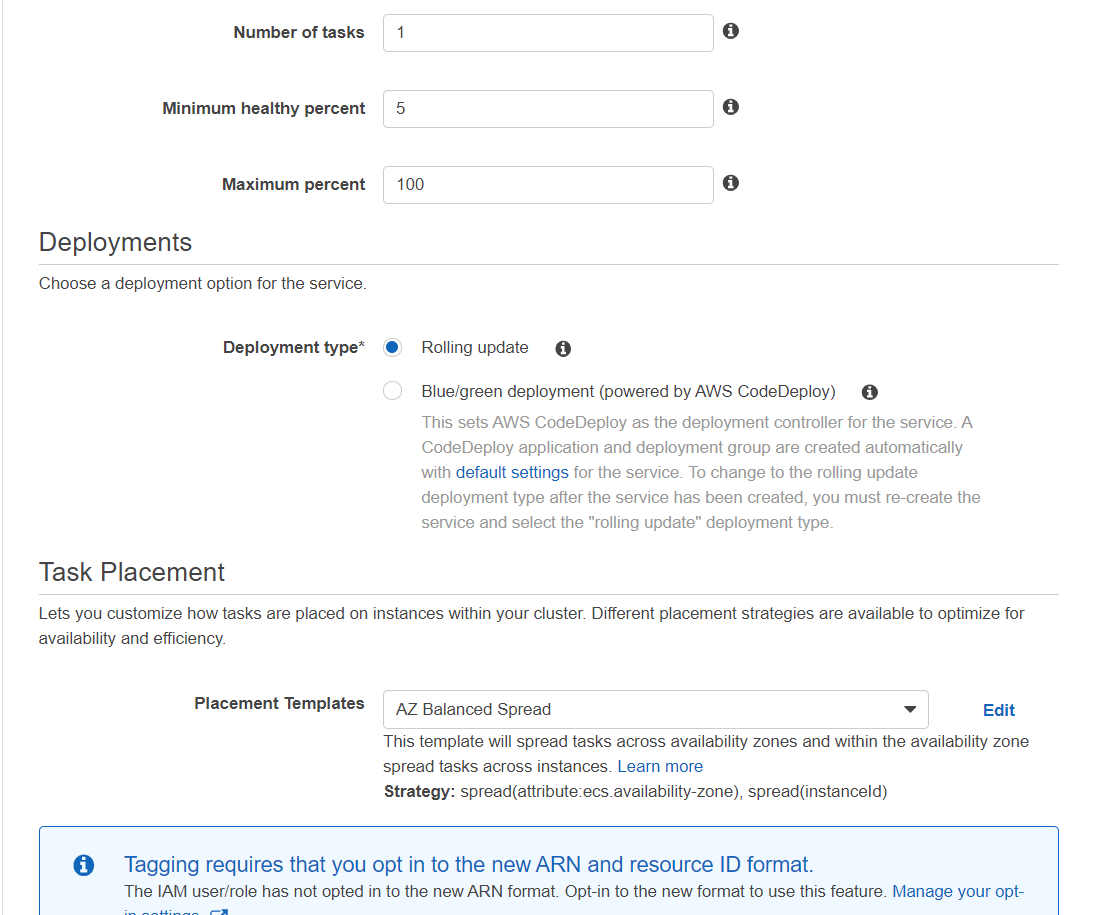


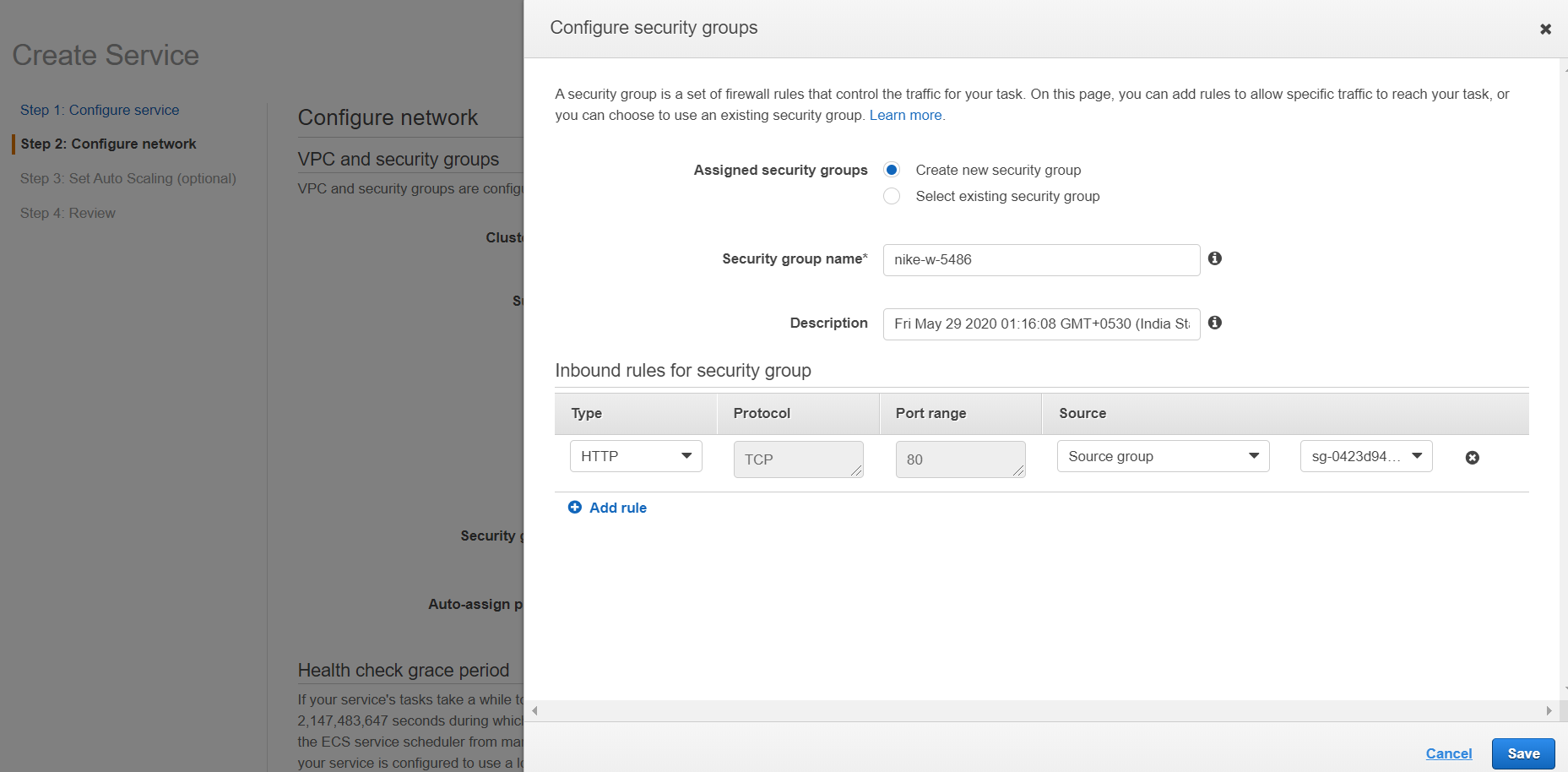
# Creating Service in Cluster

Following are the steps to create service:

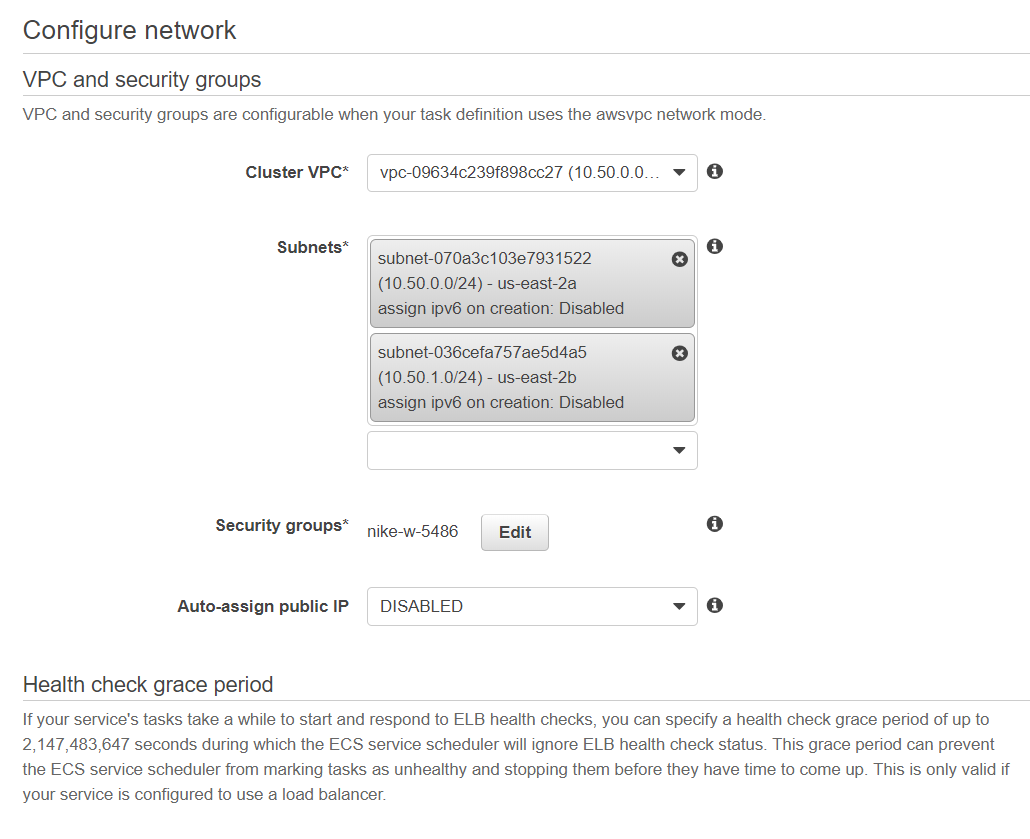


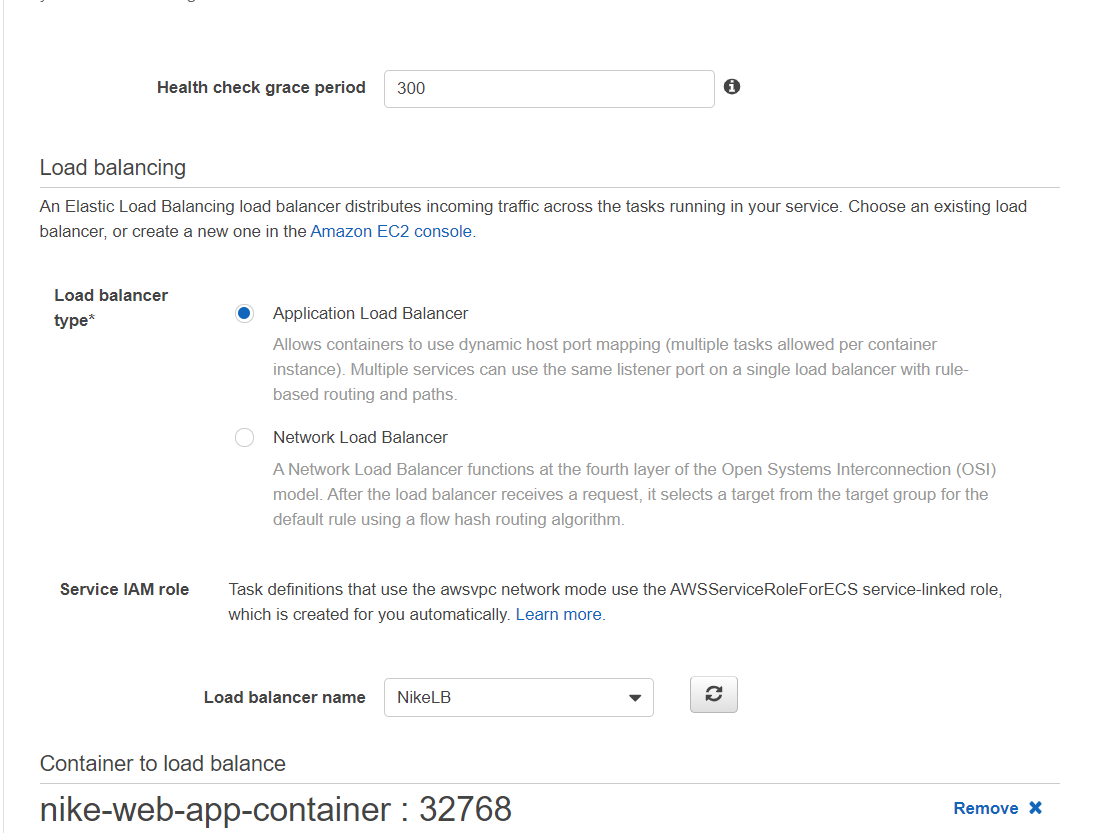


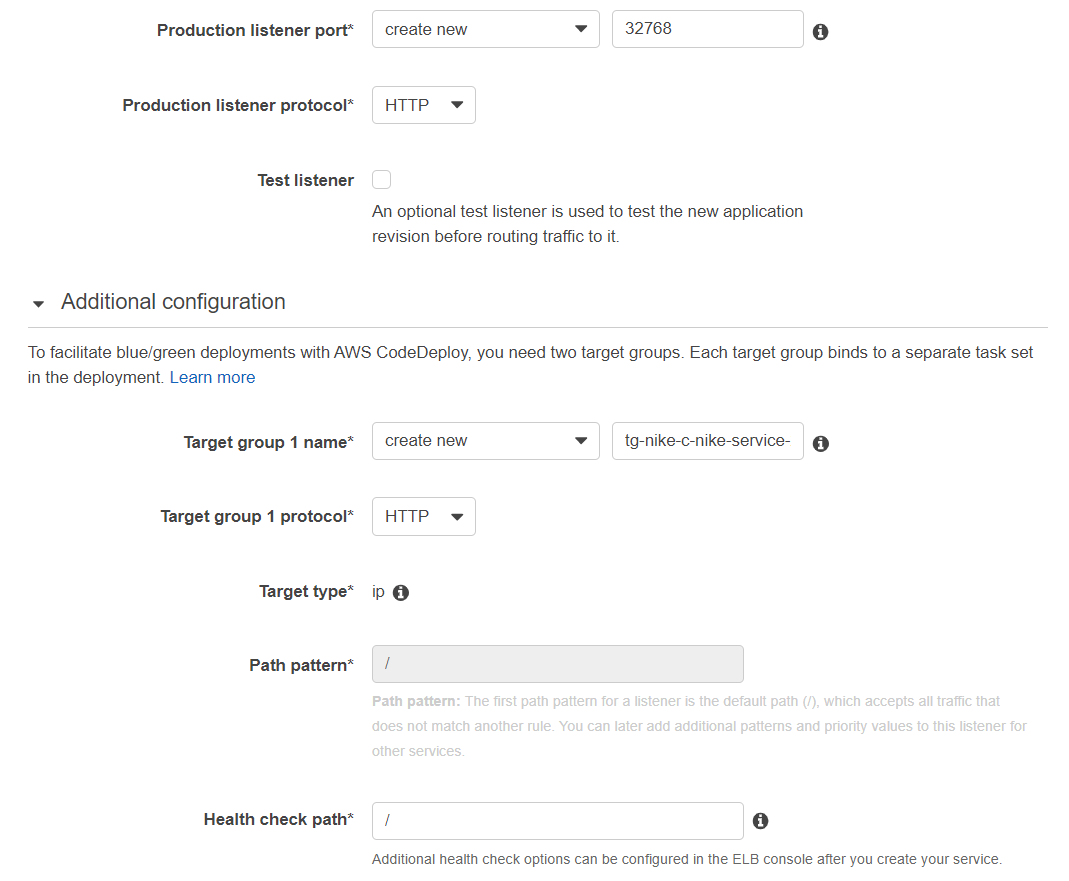


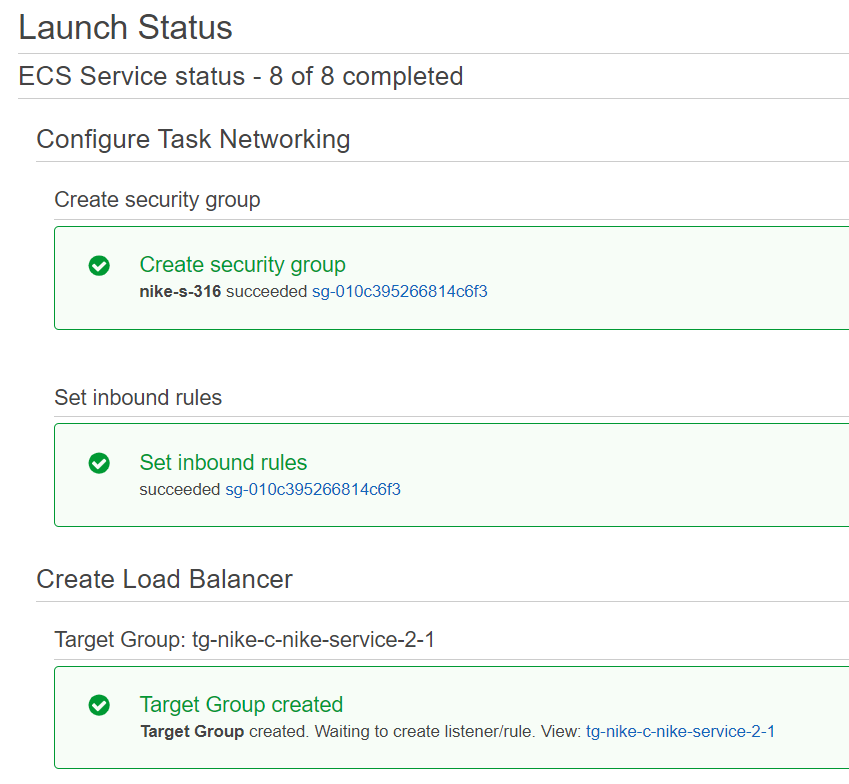
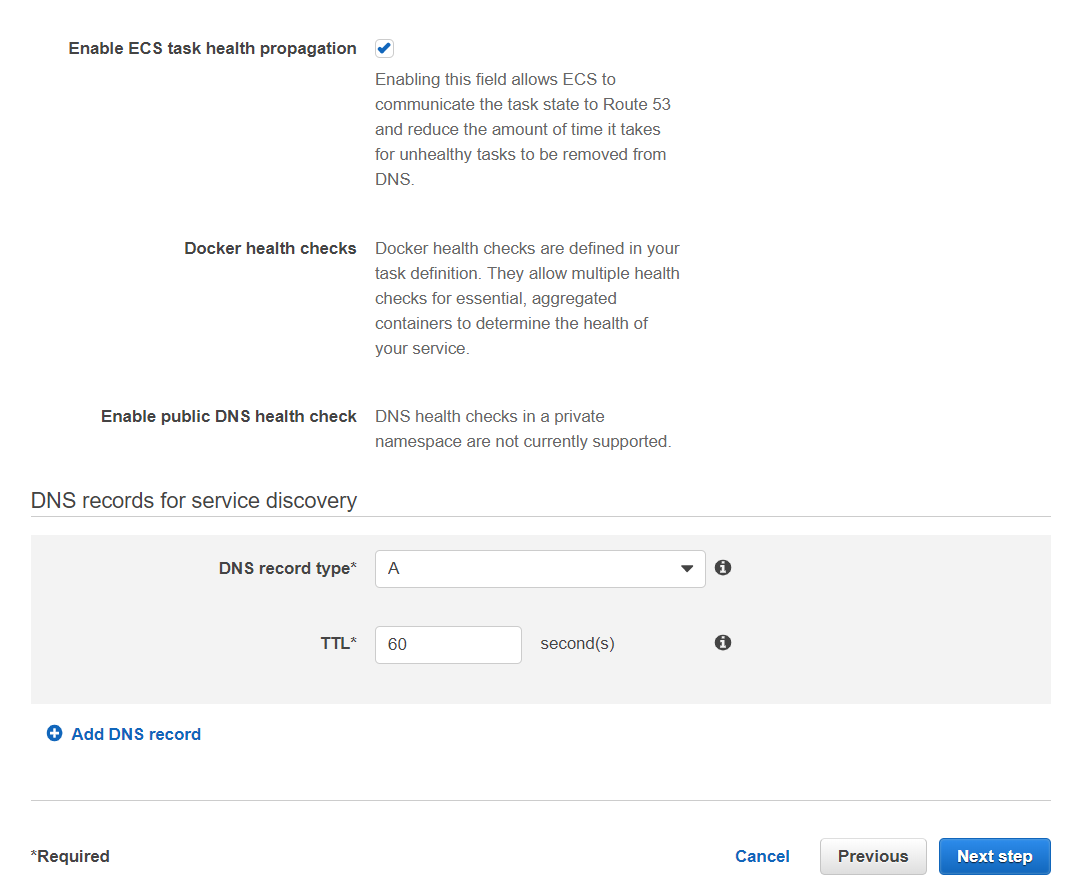
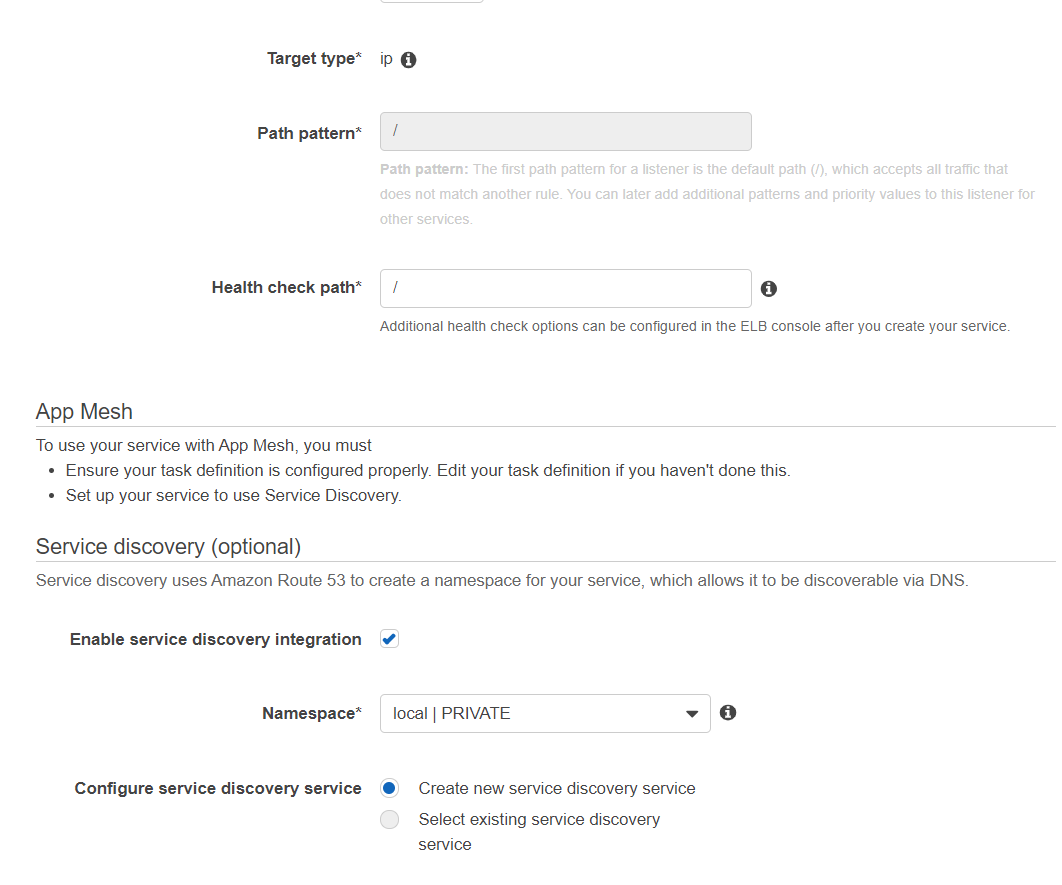


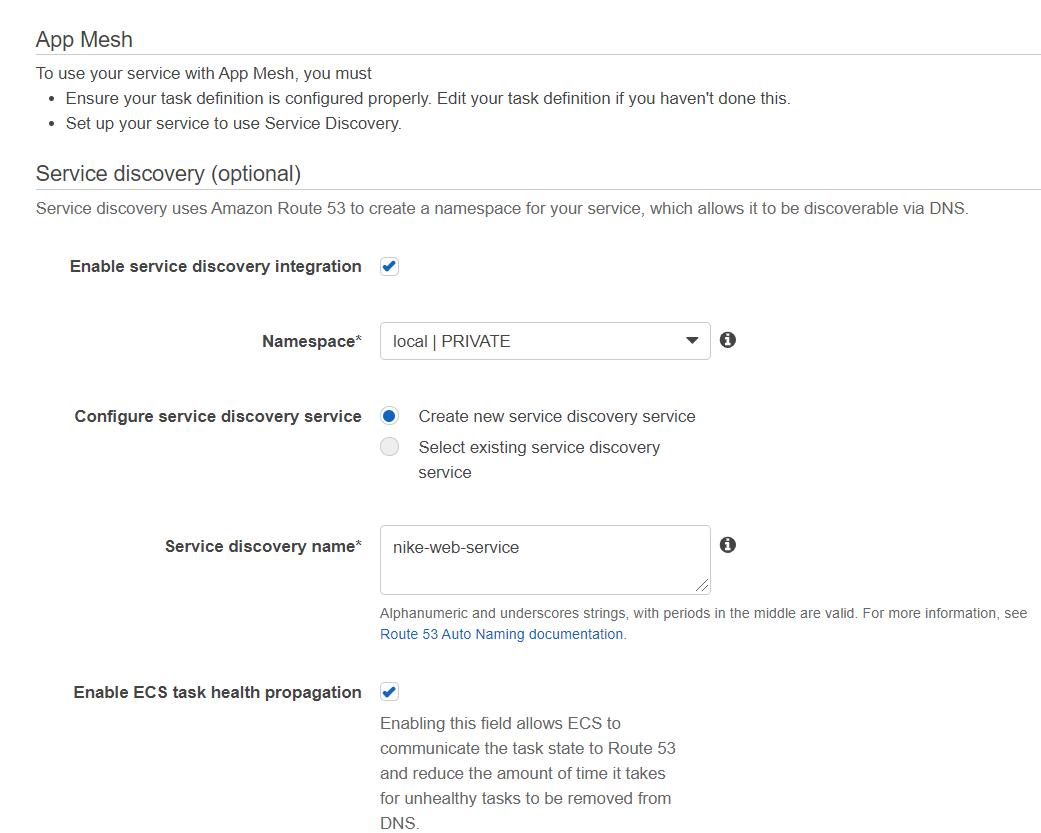
Security groups can be set to one’s machine ip to secure. But, in this demo, it’s set for public access.

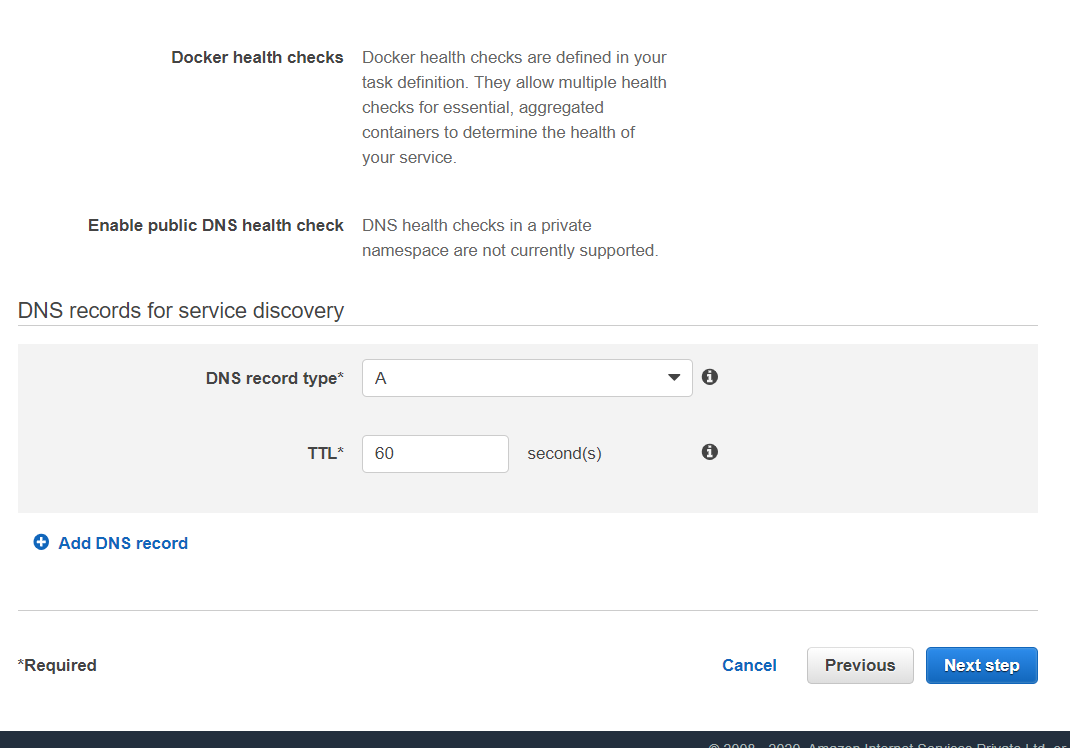


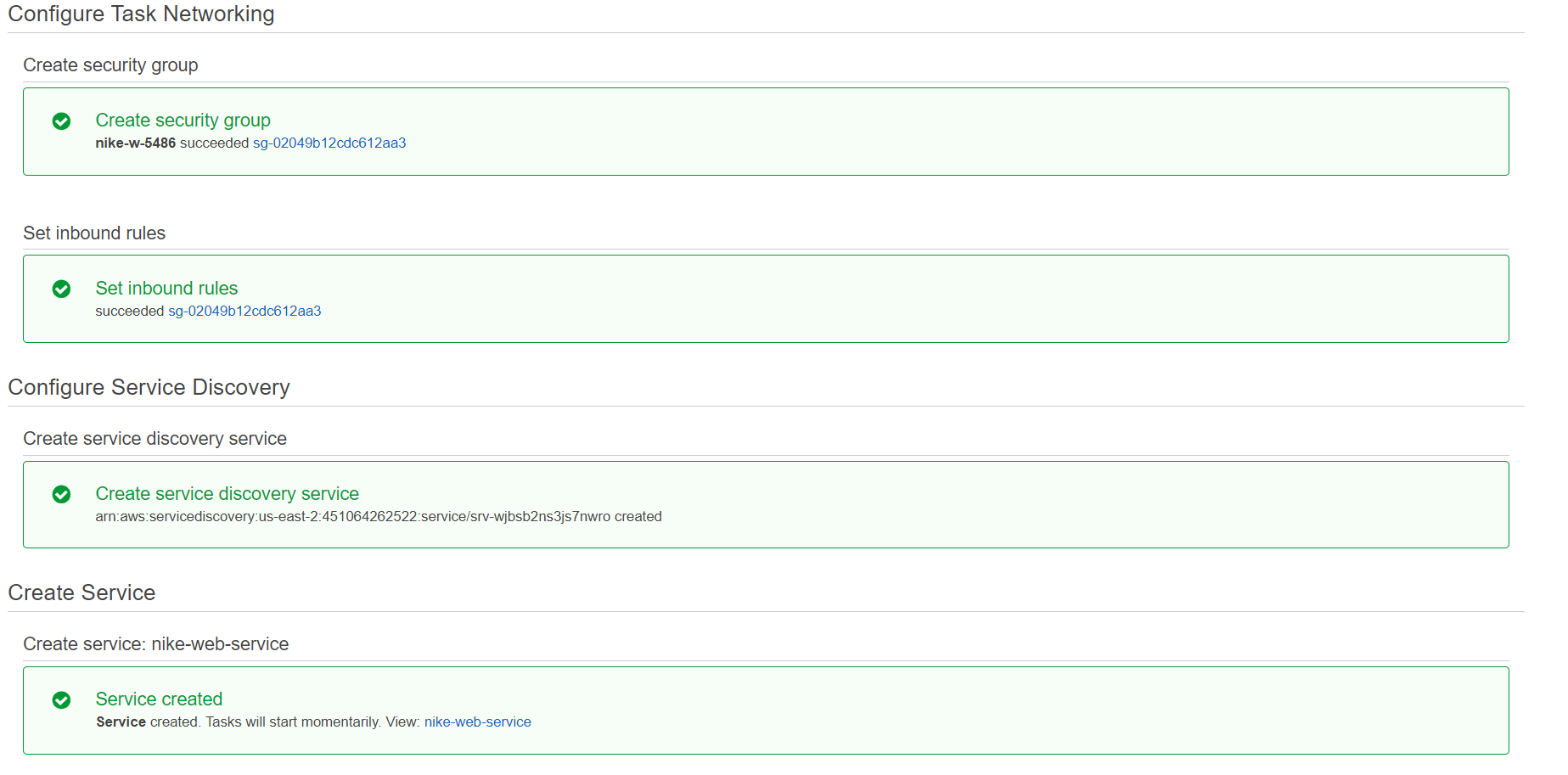
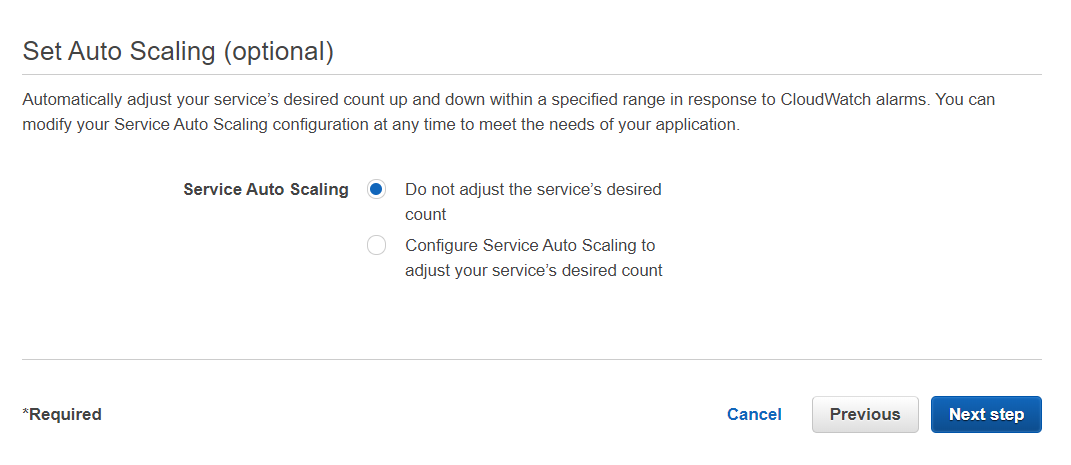




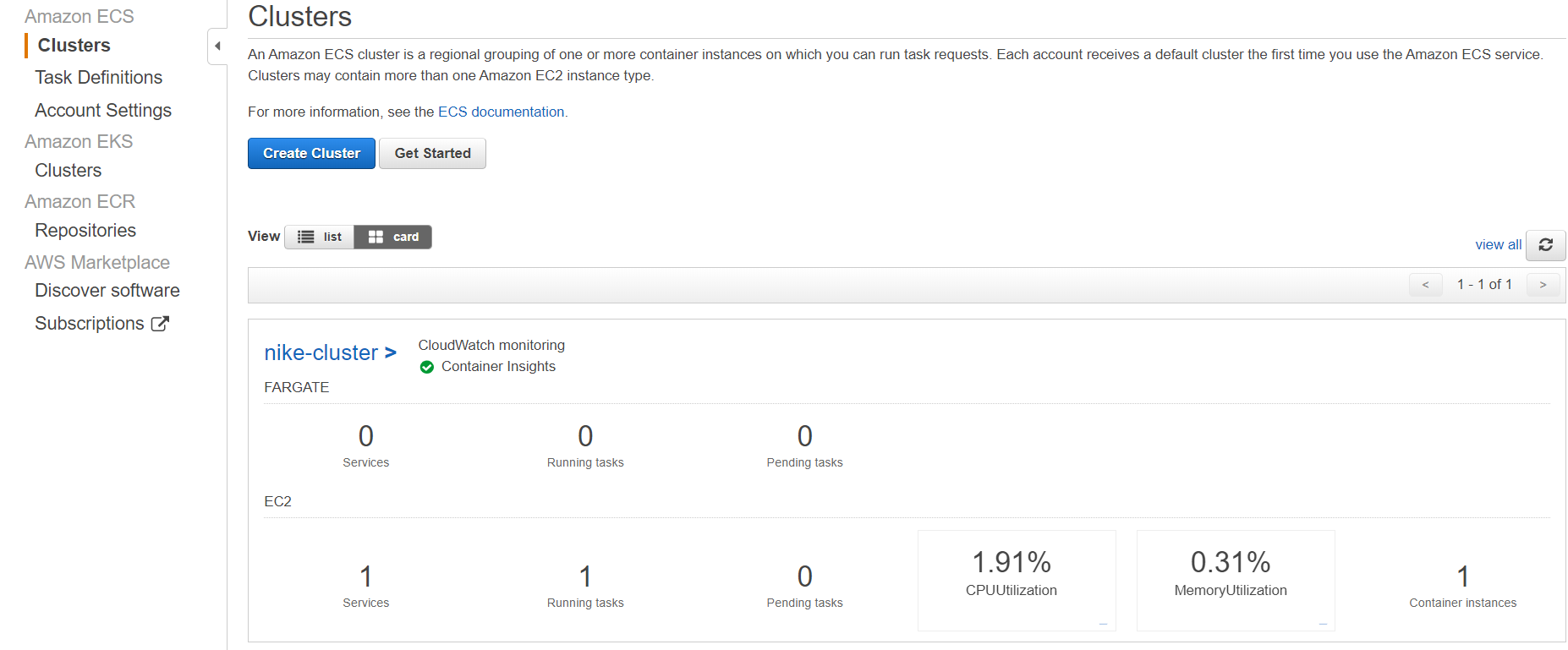




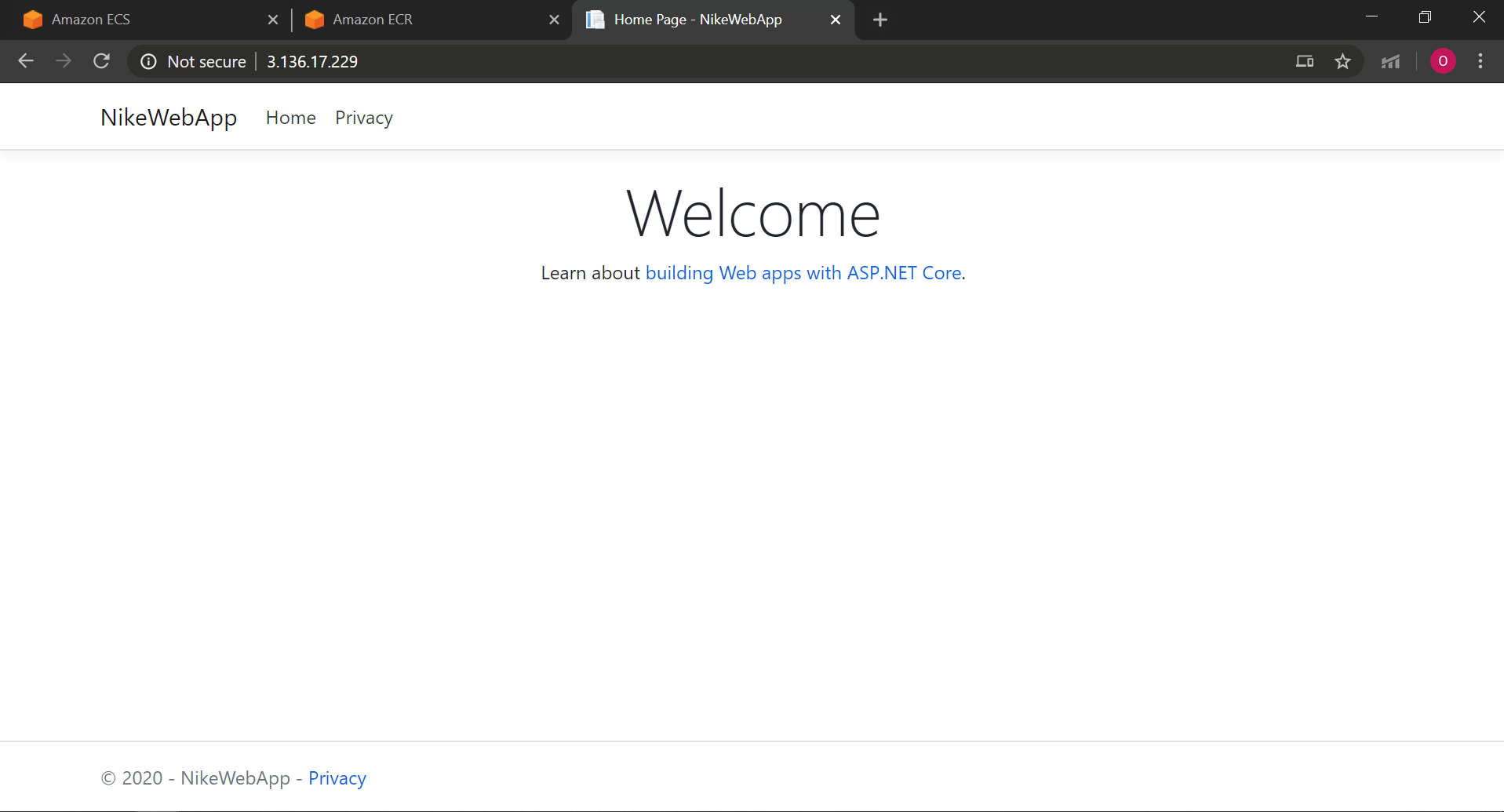


## Cluster Created

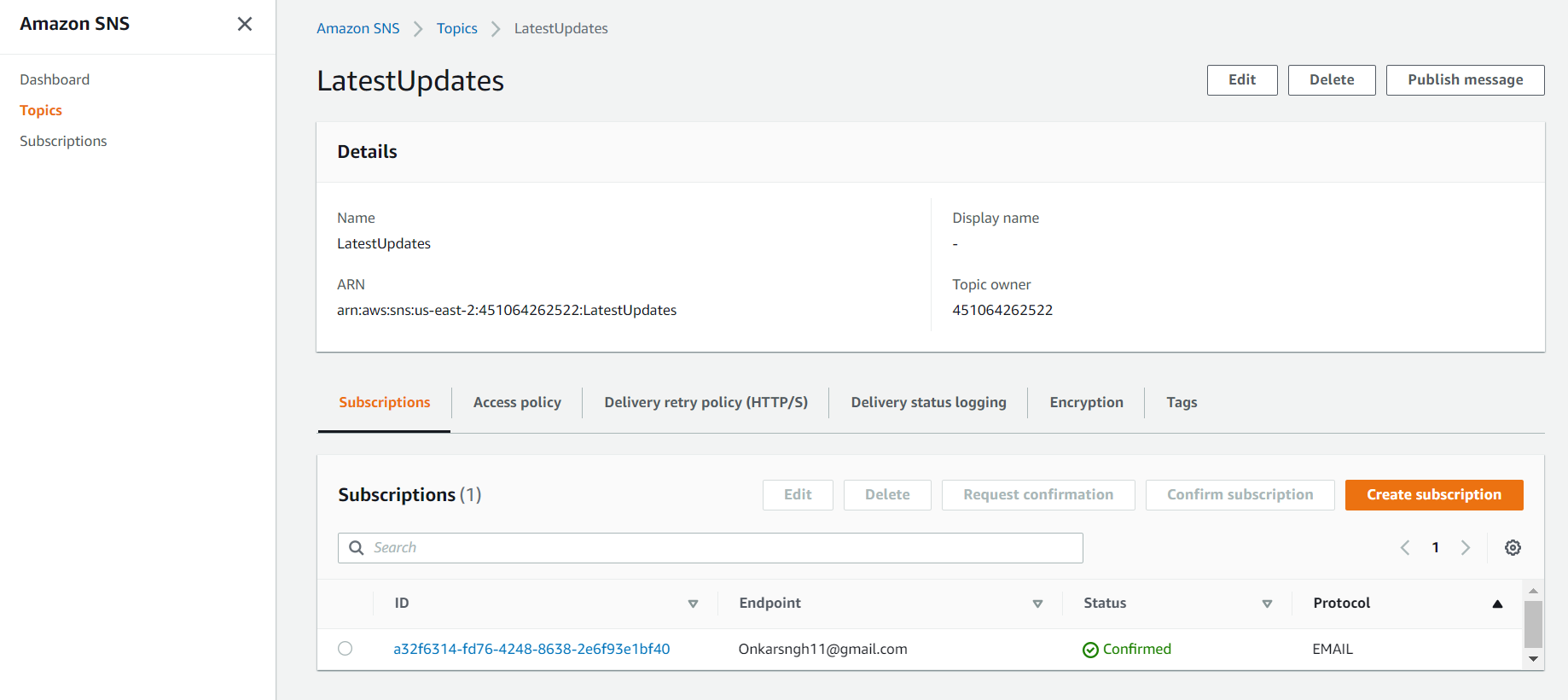


# Application Run



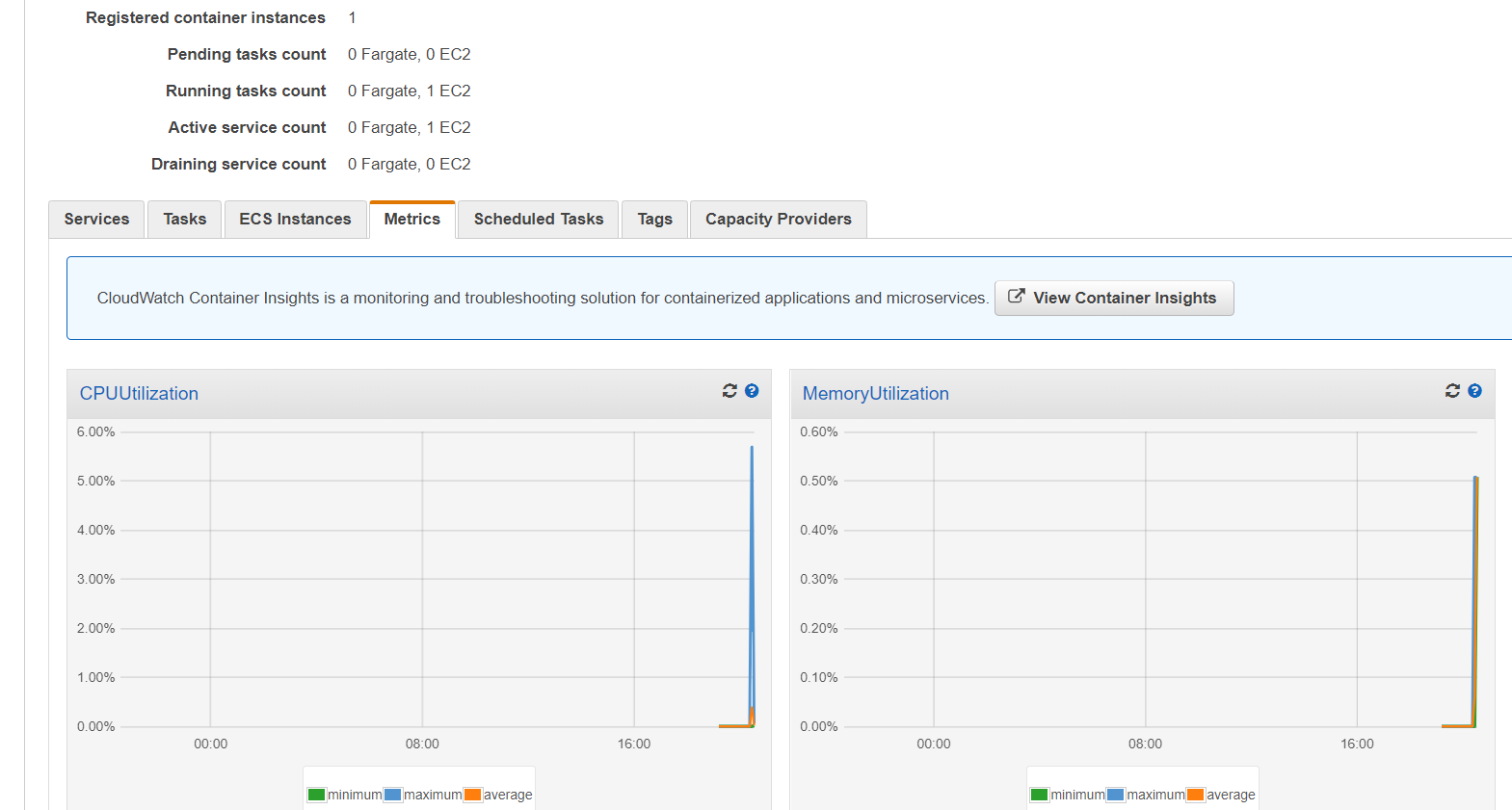
# SNS Integration

Added Subscriptions manually as of now but this can be configured from application parameters using json.



# CloudWatch Integration

**Cloud Watch :-** Tomonitor the health of the application.



# Cost Calculator

