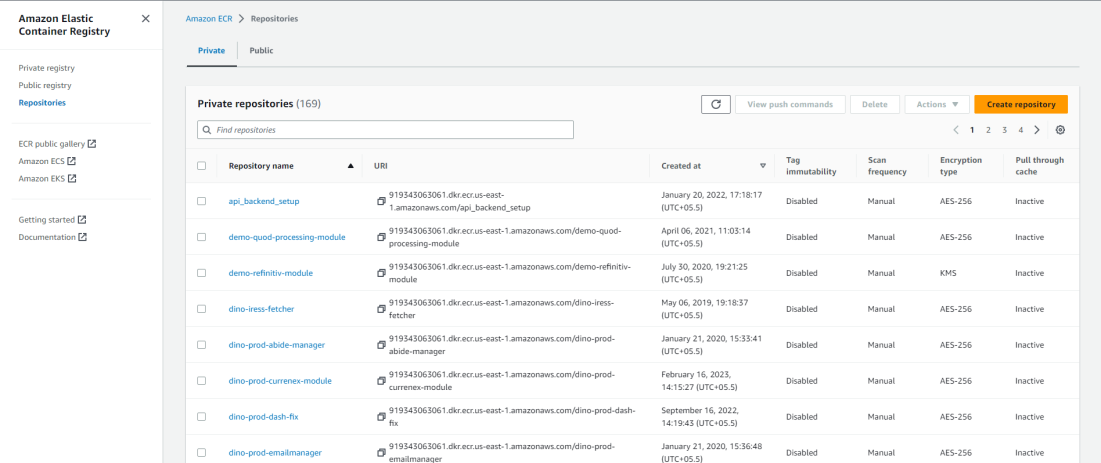
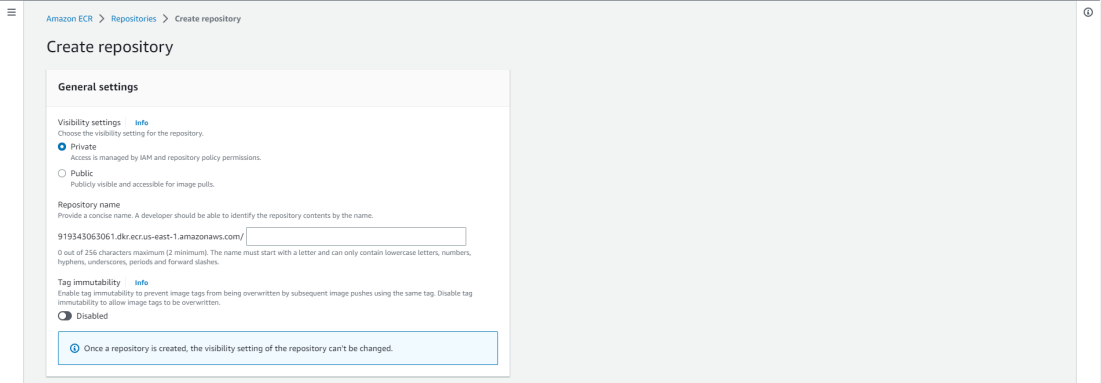
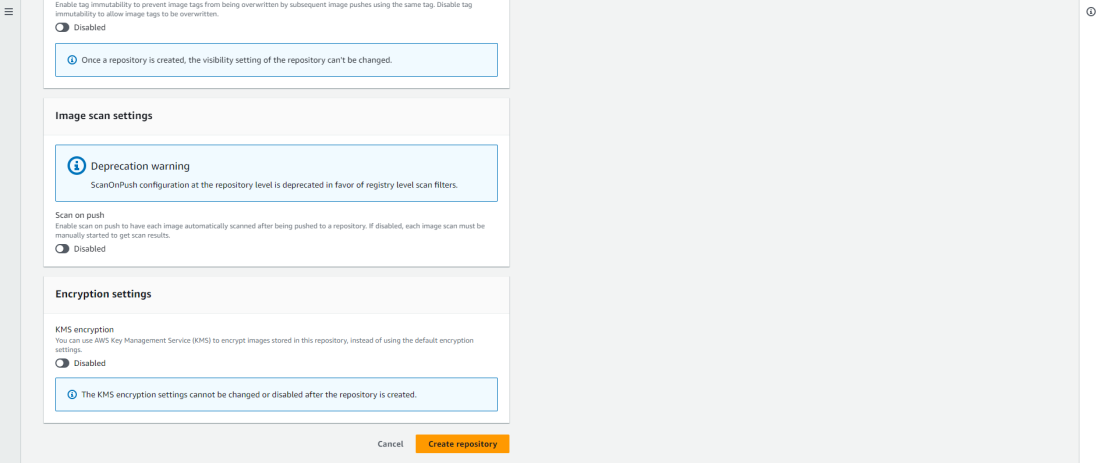
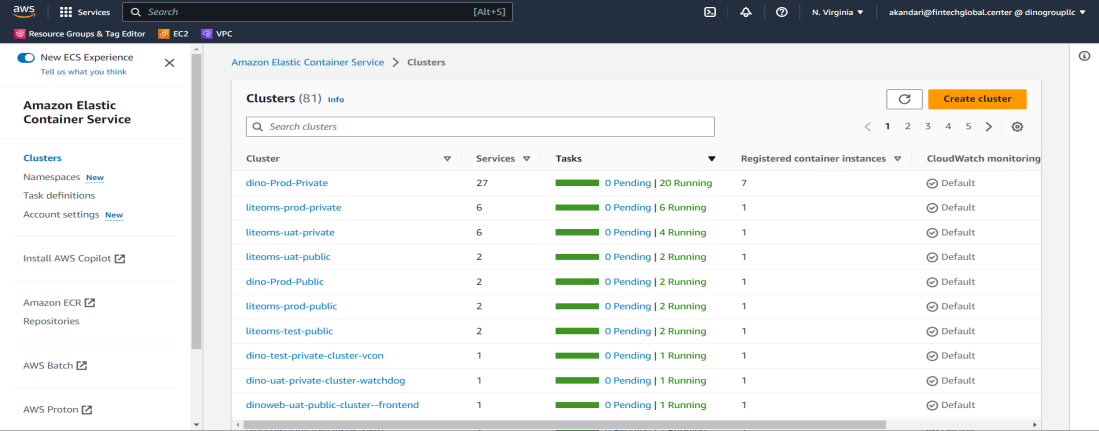
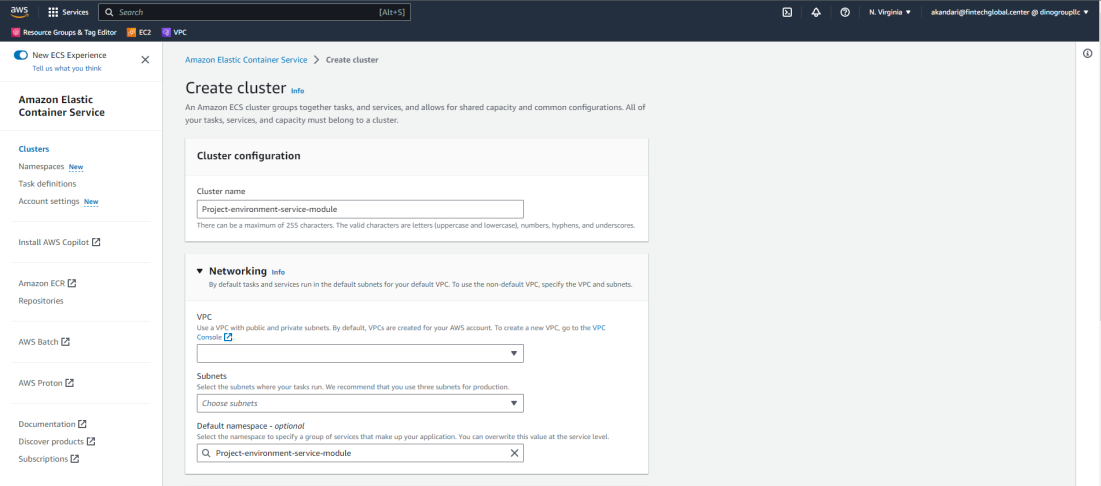
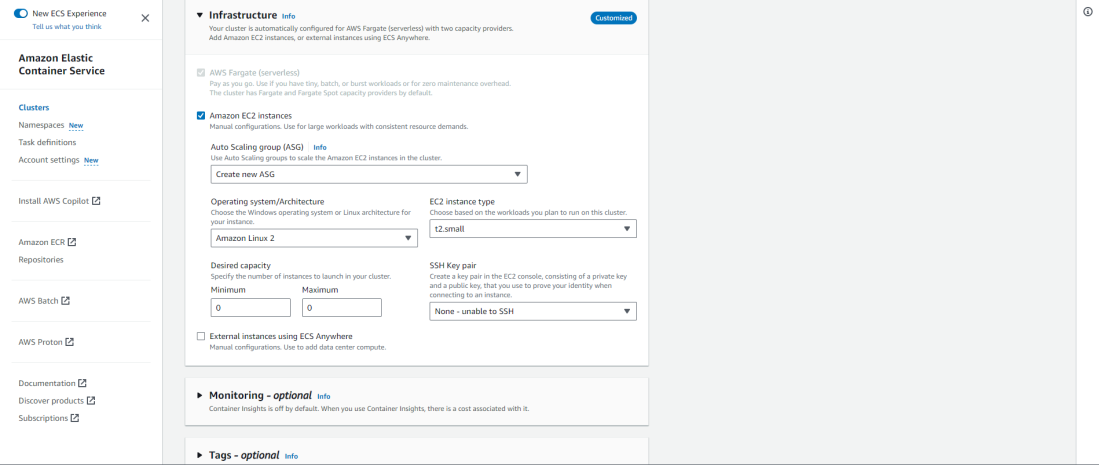
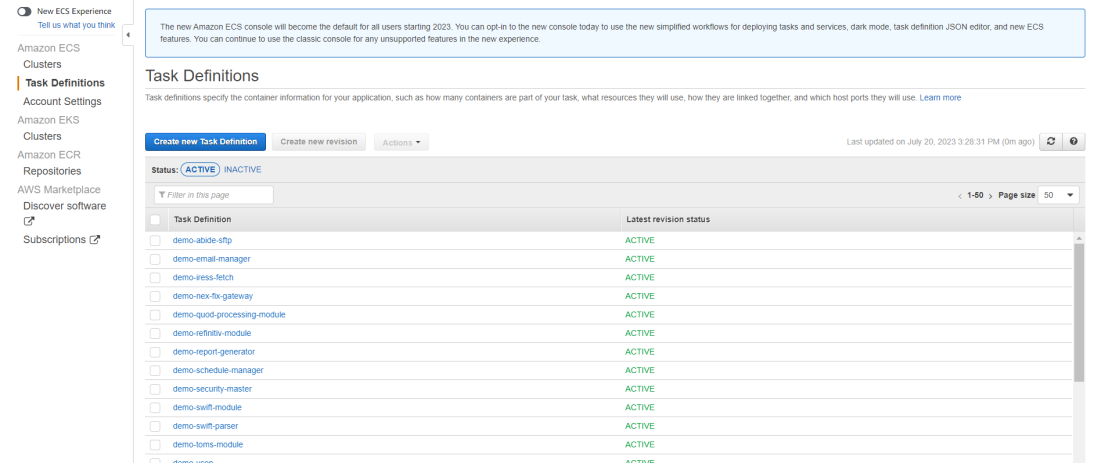
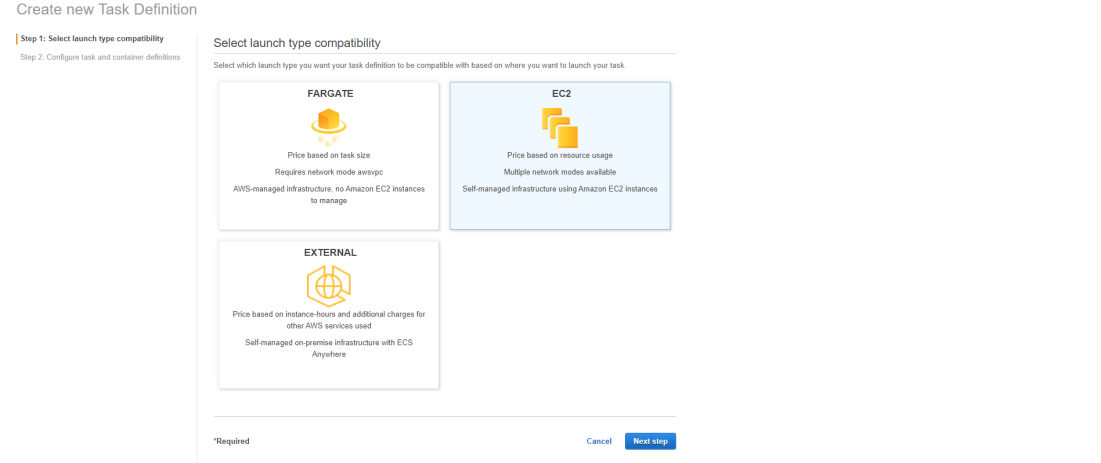
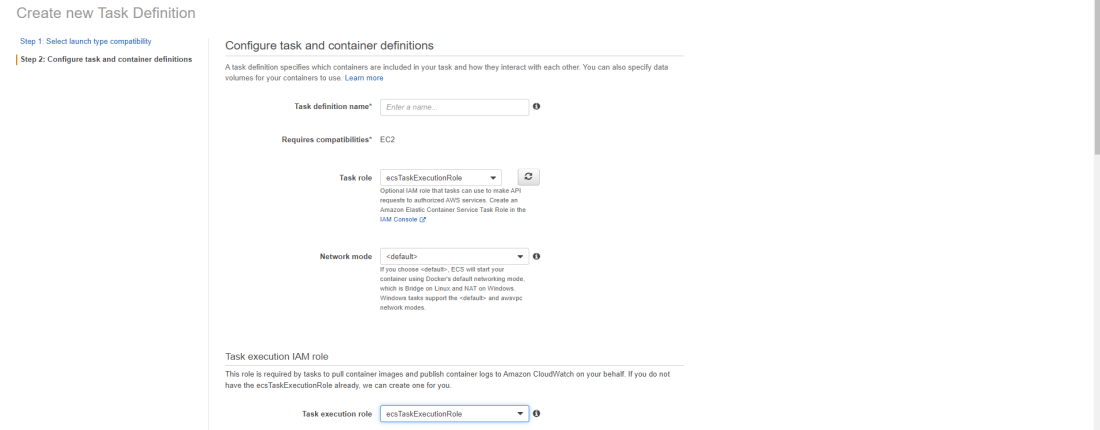
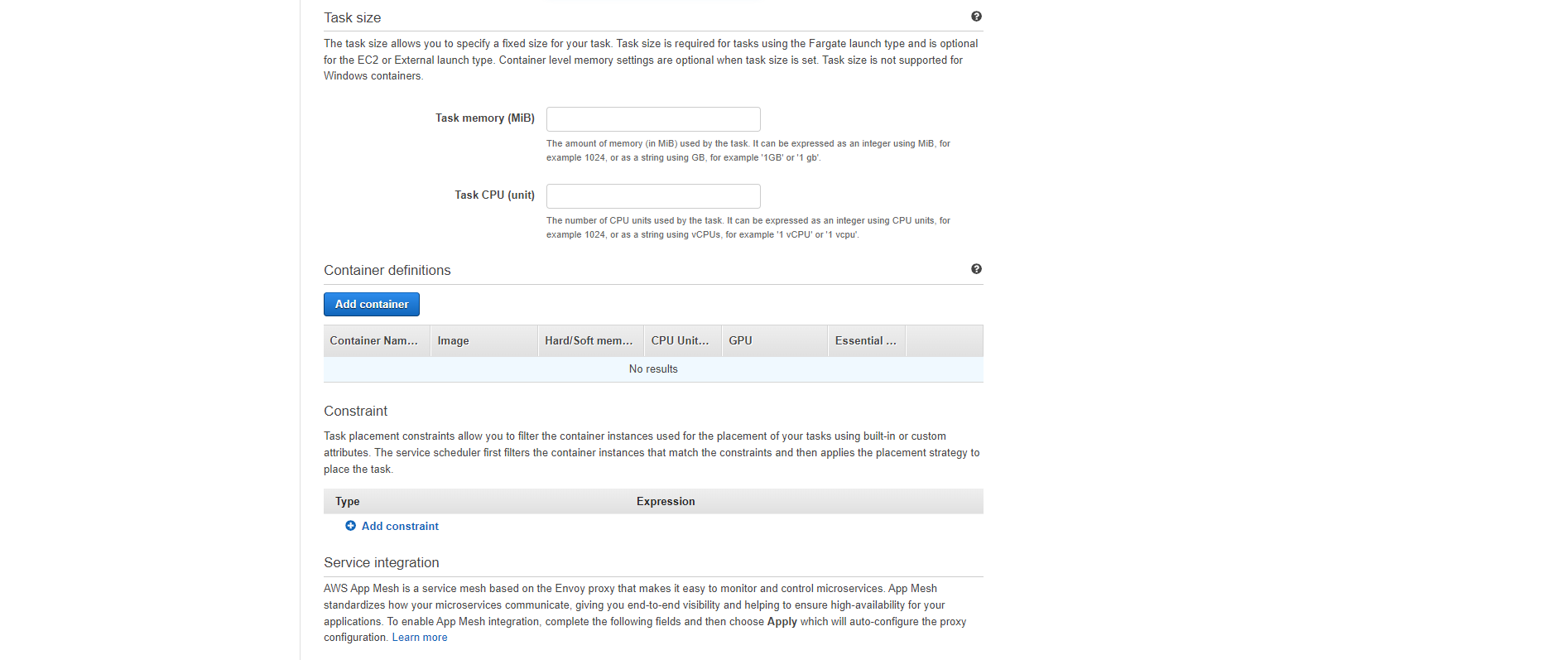
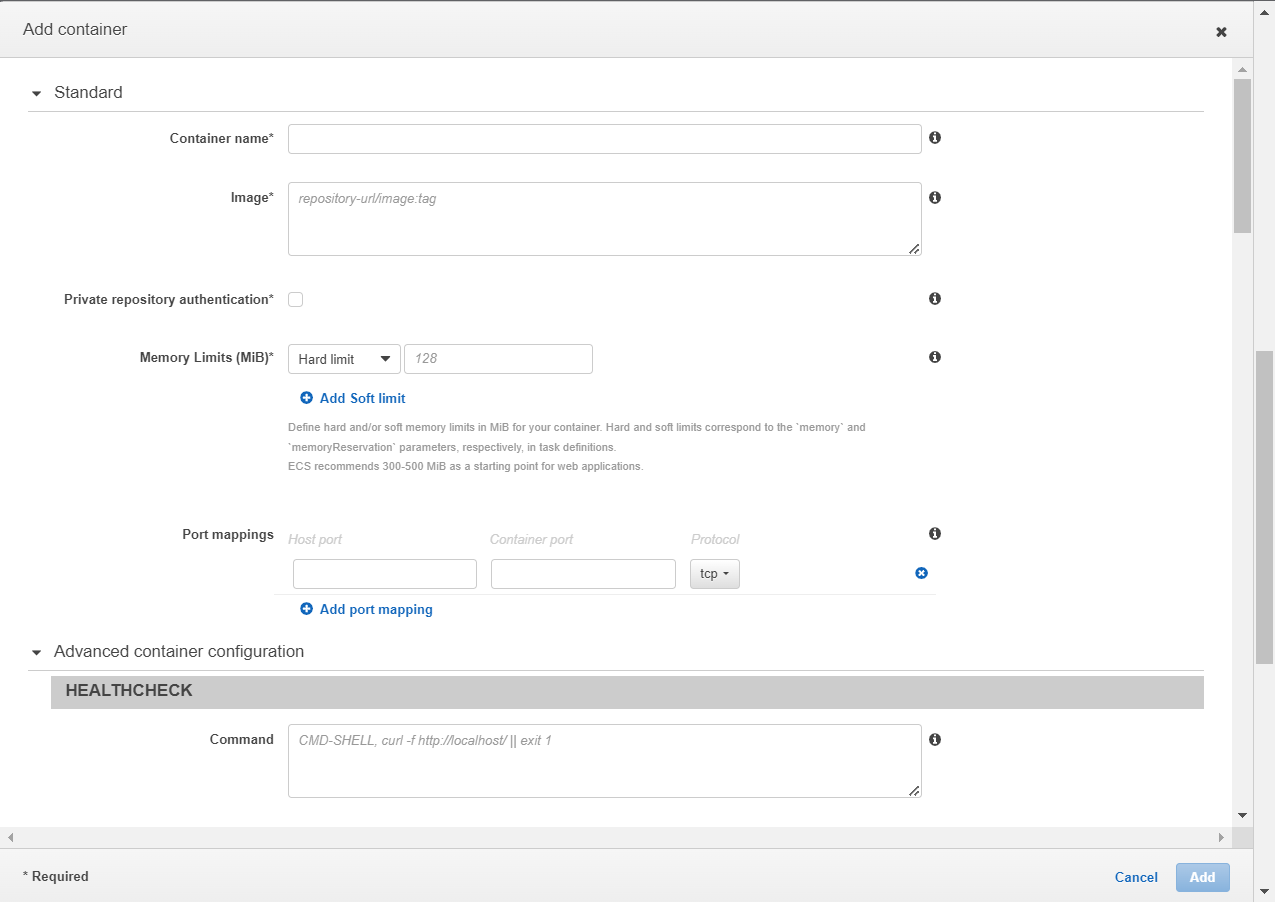
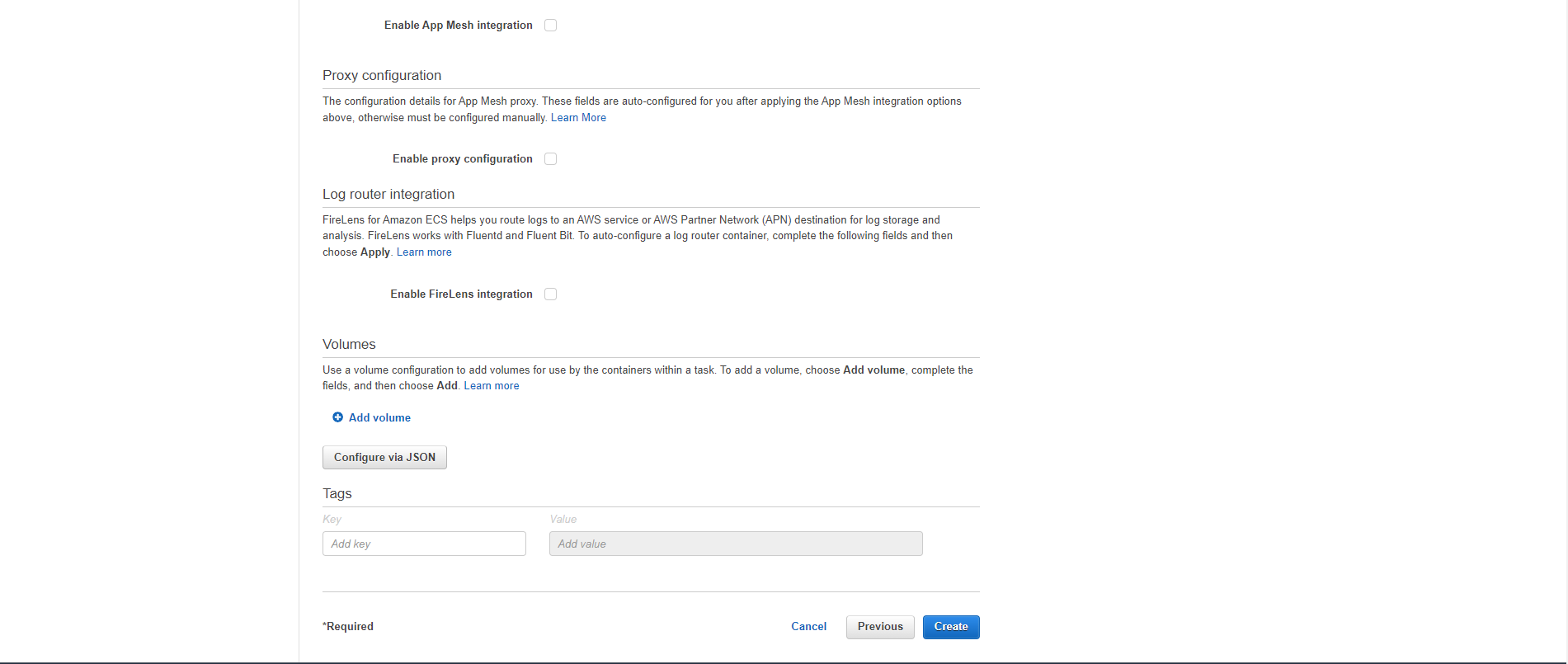
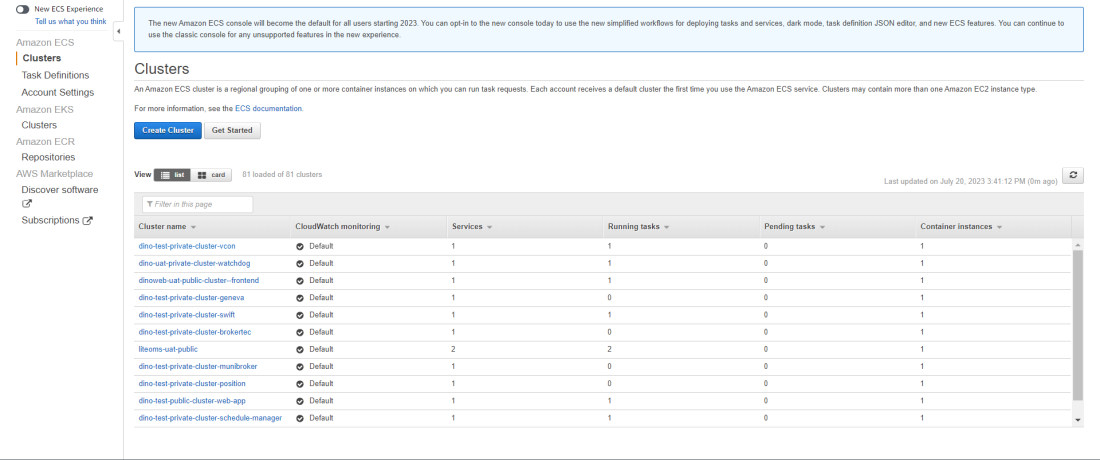
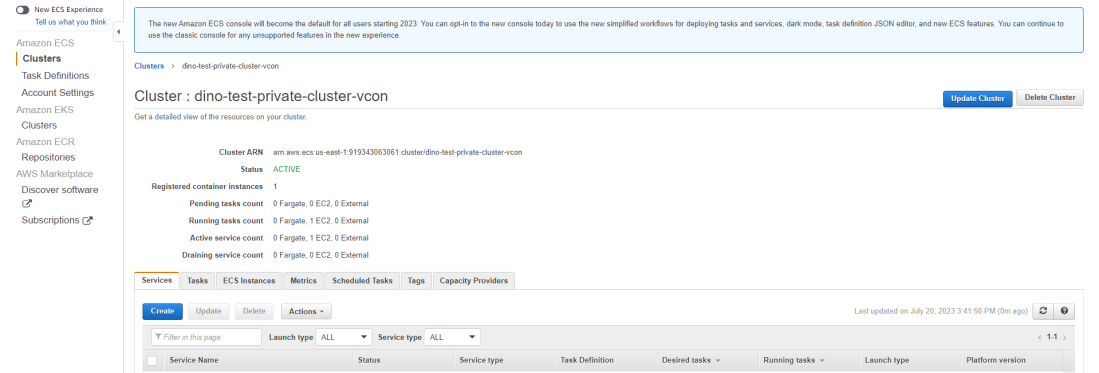
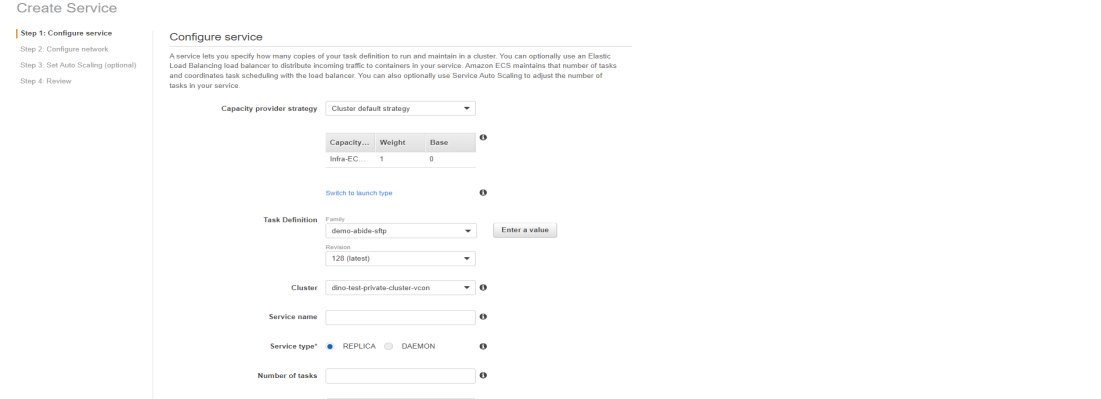
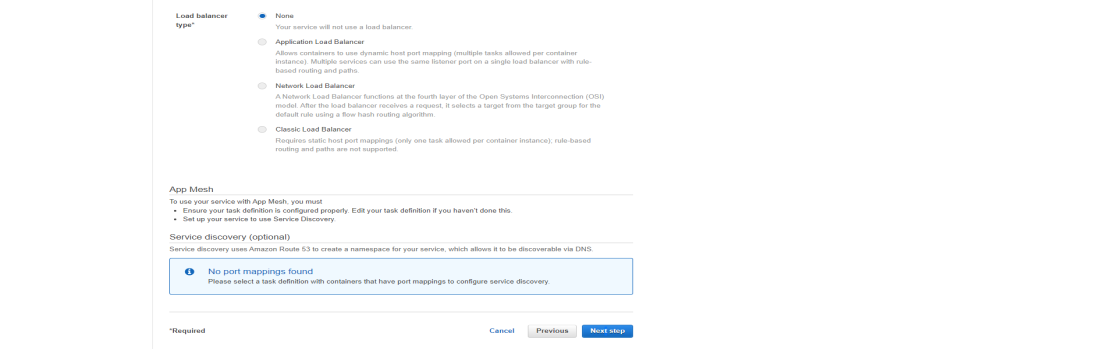
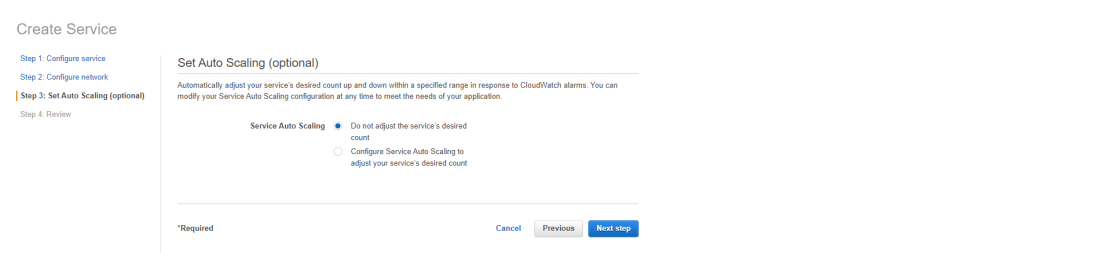
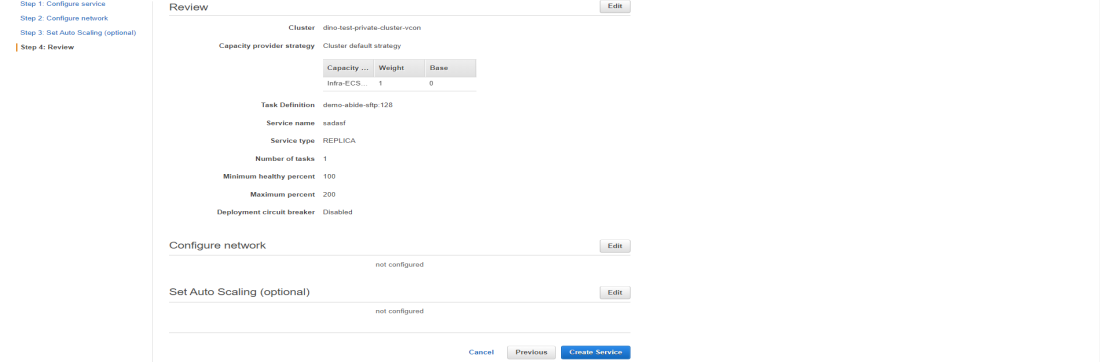
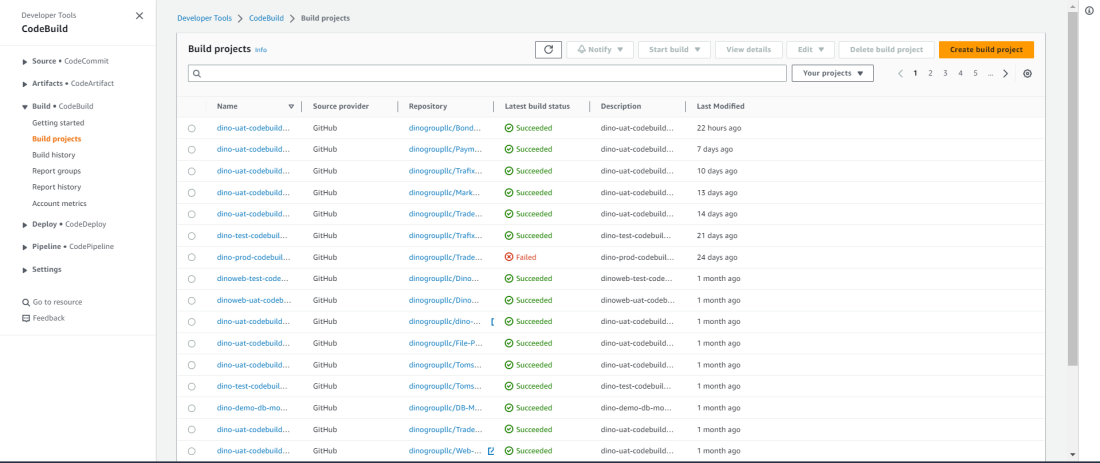
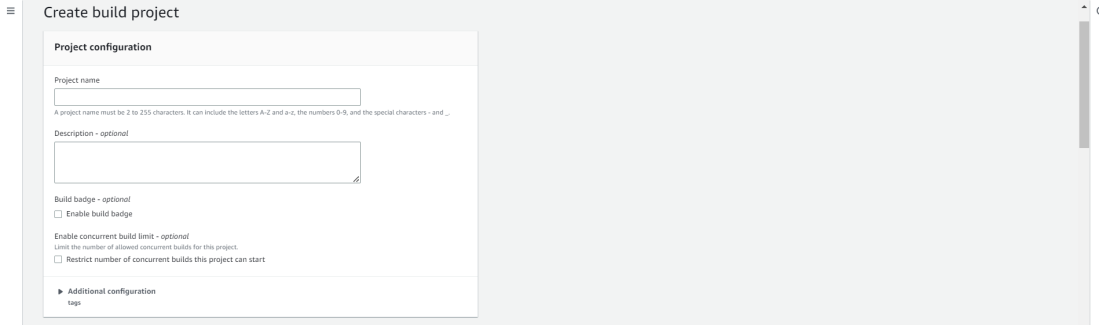
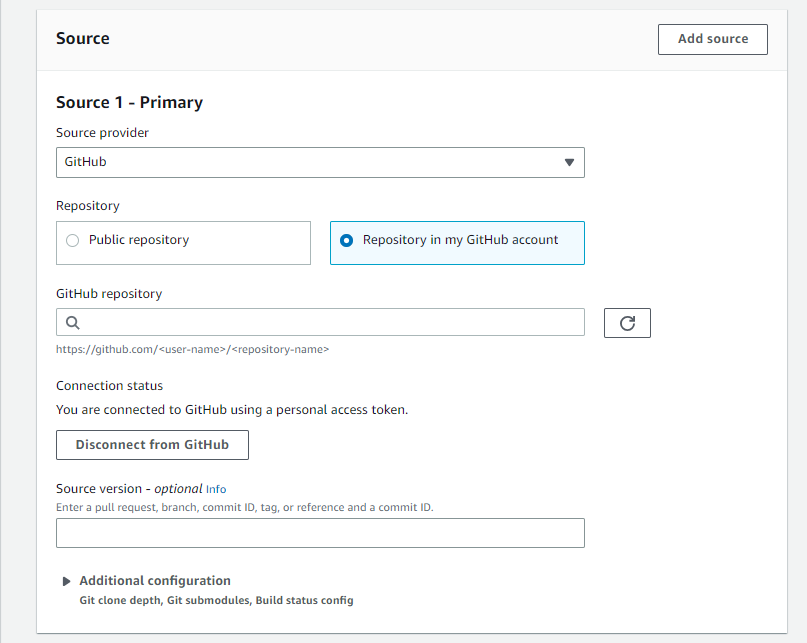
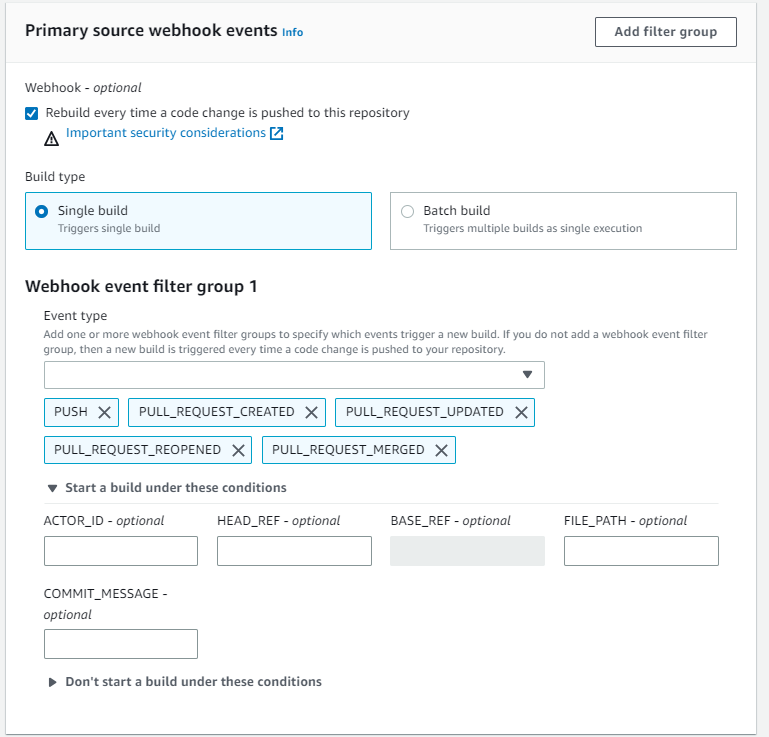
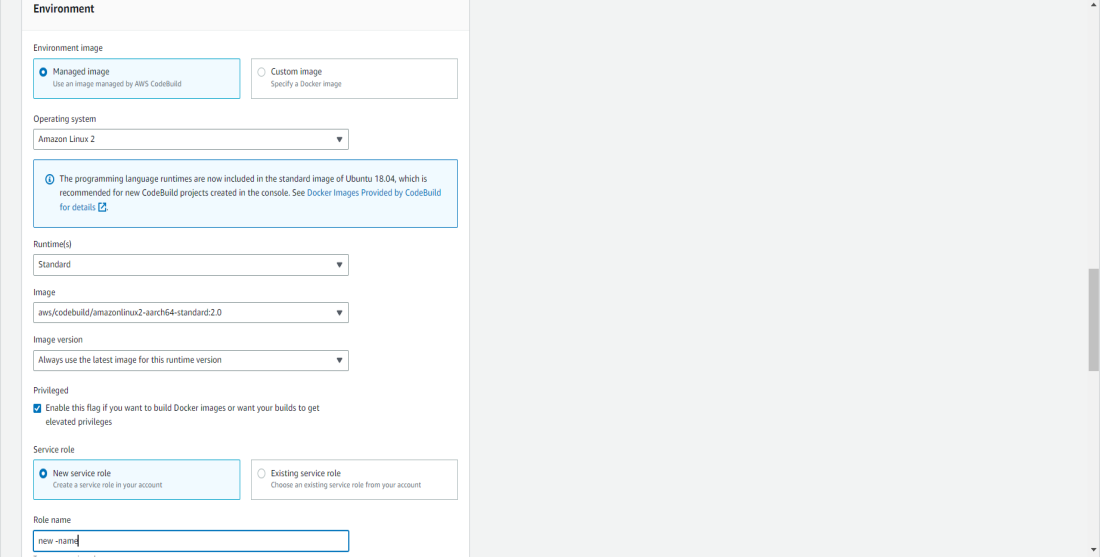
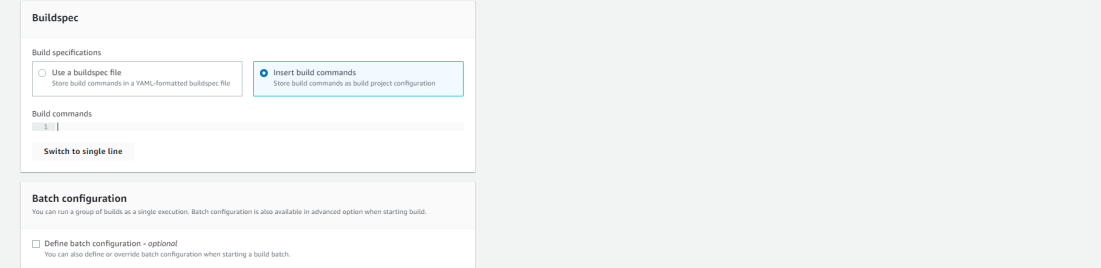
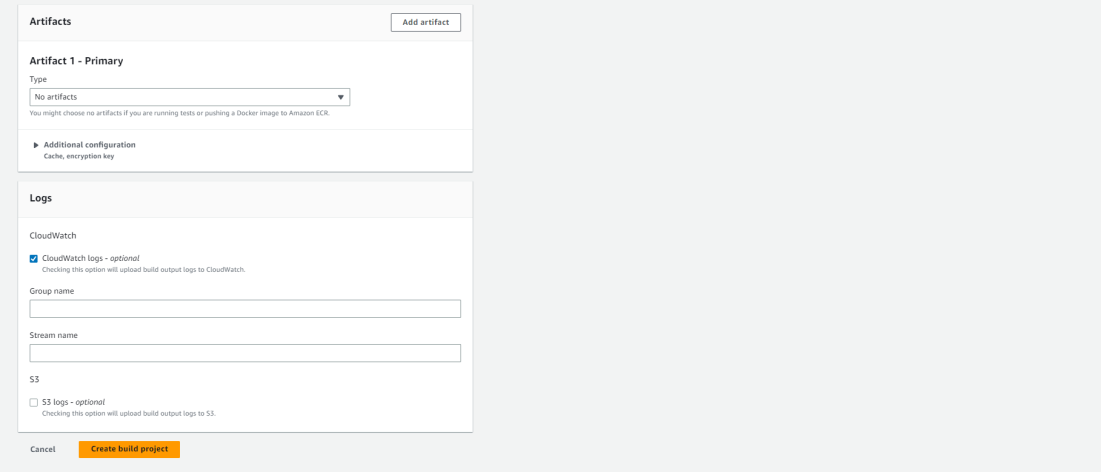
**CICD FOR DINO**

* CICD includes below 3 services of AWS
  + ECS
  + ECR
  + Codebuild

Below are the steps to create ECR, ECS and Codebuild.

* ECR
  + Open ECR ( Elastic container registry)  
    
  + Click on create repository  
    We need to select private repository  
      
    Naming convention should be Project-environment-service-modulename (dino-test-repository-vcon)
  + Disable the image scan settings and Encryption settings  
    Click on Create repository  
    
  + Your private repository will be created copy and save the URI
* ECS  
  In ECS we have to create below services.
  + Cluster
  + Task definition
  + Service
* Cluster
  + Select the new UI from AWS console in toggle button to top-left corner.  
    
  + Click on create cluster  
    
  + Enter cluster name (Project-environment-service-module)  
    Select specific environment VPC (for ICE Bondpoint and TMC, please select Dino-ice-vpc)  
    select proper subnet for private and public module  
    
  + Inside Infrastructure   
    Select **Amazon EC2 instances**  
    Select “**create new ASG**” in Auto scaling group (ASG)  
    Select “**Amazon Linux 2**” in operating system  
    Select Ec2 instance type depending on the module  
    Select Desired capacity as **min 0** and **max 0**Select **SSH key pair** depending on the environment  
    Add proper tags as below  
    Name – cluster name  
    Environment – Test/UAT/Prod  
    Resource Type – Service Name  
    Click on Create  
    
* Task Definition
  + Select the old UI from AWS console in toggle button to top-left corner.  
    Inside **ECS click on Task definition** in left navigation bar  
    click on create new Task Definition  
    
  + Select **Launch Type as EC2** as cluster in which task are deployed are also in EC2.  
    Click on Next.  
    
  + We need to configure the Task and Container definition  
    Enter **Task Name**  
    Select Task role as “**ecsTaskExecutionRole**”  
    Keep the Network mode as **<Default>**  
    Select Task Execution role as “**ecsTaskExecutionRole**”  
    
  + Task size should be left as blank  
    click on **Add container**  
    
  + Enter **Container Name**   
    Enter the **URI** which we saved from ECR in image section.  
    Select memory limit as **soft** and 200  
    If there are any port mapping add   
    Tick mark the **logs to save logs** in cloud watch  
    Click on **ADD**  
    
  + Click on **create**  
    
* Service
  + Click on **cluster** and **select the cluster name** you want to create service for  
    
  + Select the service Tab  
    Click on **Create**
  + Select the **Task definition** for the module to be deployed  
    Select the **revision number**  
    Select the **cluster name**   
    Add **service Name**  
    Service Type as **Replica**  
    Number of Task should be 1 to Run the Task and 0 to Stop the Task  
      
    Click on **Next** step  
    Select the **Load balancer** if any   
    click on **Next** Step   
      
    Do not adjust the **Auto scaling**  
    click on **Next**  
      
    Review the service  
    if all okay click on **Create Service**  
    
* Codebuild
  + Select Build Projects in left navigation bar inside Codebuild.  
    **Click on**
  + Enter **Project Name**  
    enable the **Build Badge**  
    Add **Tags**  
    
  + Inside **Source**   
    Select provider as **Github**Select Repository as Repository in **Github account**  
    Select the module in **Github Repository**  
    Enter Source version as **Branch in Github**  
    
  + Inside primary source **webhook** events  
    Select **Webhook** to automate to start the build when tag is done  
    Build type as single  
    Select all **event type**  
    inside start a build inside head ref enter the tag name   
    
  + Inside Environment  
    Select image as **managed image**  
    Operating system as **Amazon Linux 2**  
    Runtime as **standard**  
    Image as the **latest version**  
    Image version as **latest**  
    Enable the **privileged**  
    Create a **new service role**and enter **Role name**  
    
  + Inside **Buildspec**  
    Select **Insert build** commands  
    Enter the Build spec for the module (Ask developer)  
    
  + Click on **Create Build**  
    
* IAM Permission for service role
  + Select the **IAM role**Click on **Add Permissions** and add **ECS**, **ECR** and **S3** full access to the role  
    