INDEX

PRACTICAL 1

AIM: Develop an ASP.NET Core MVC based Stateless Web App.

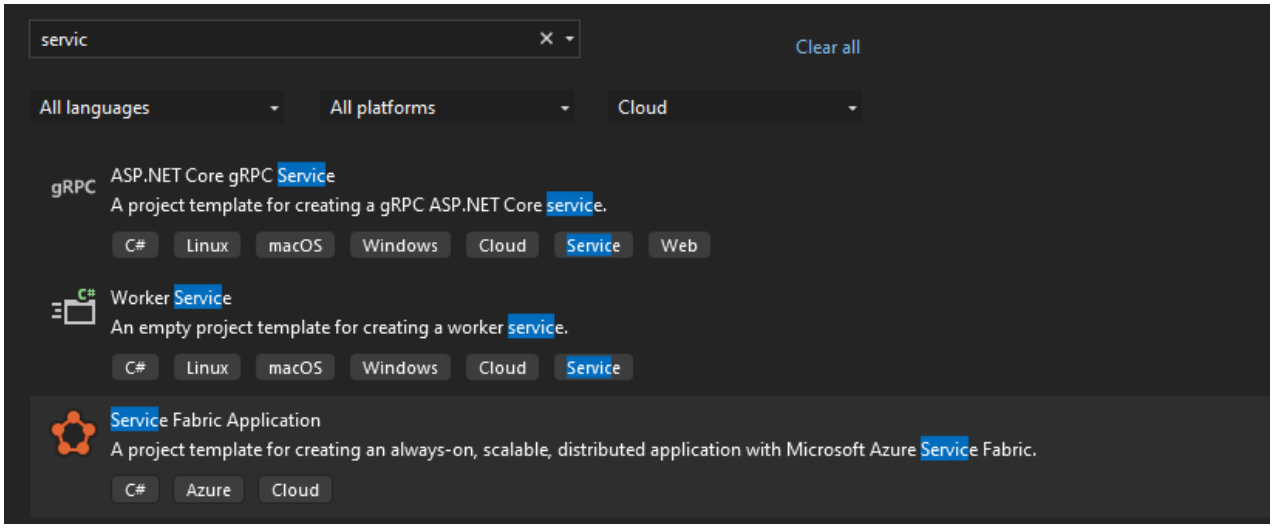
CODE:

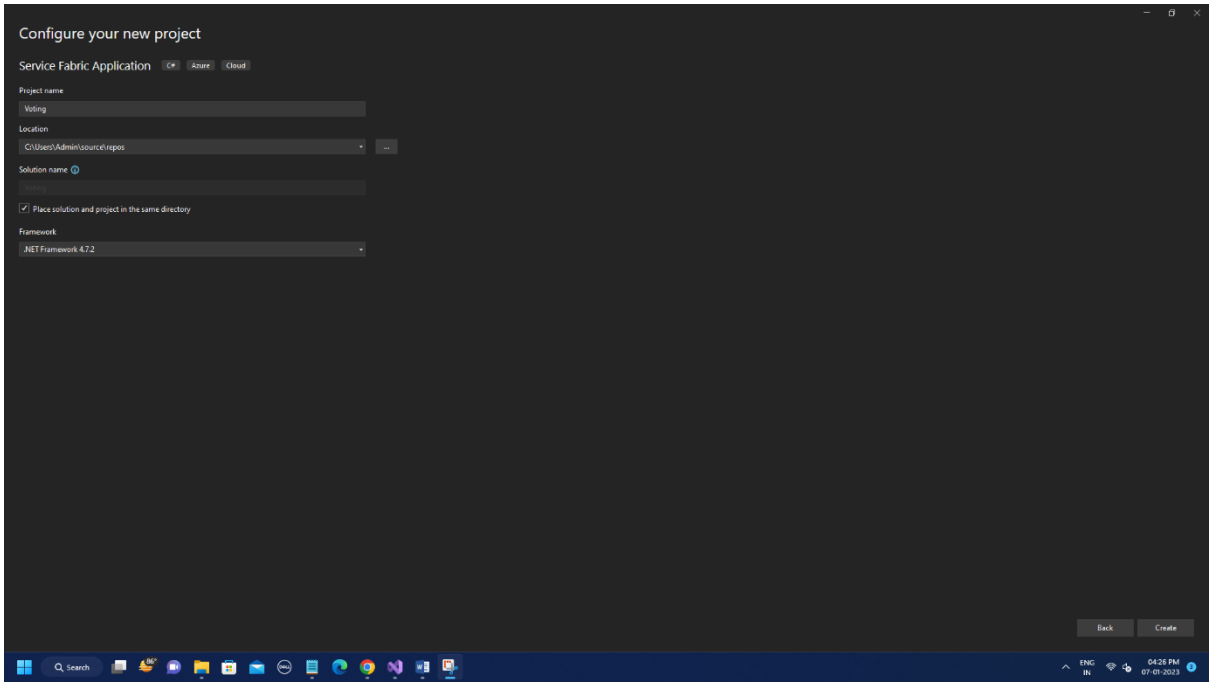
Step 1:

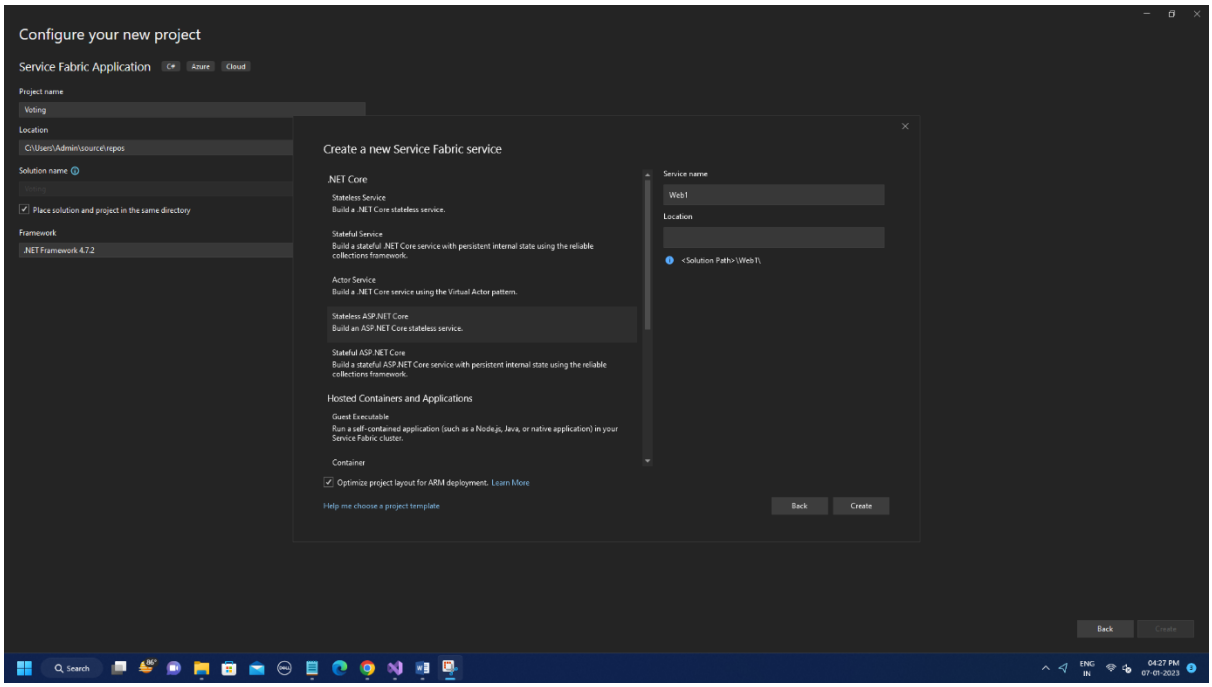
- Create Project Launch Visual Studio as anadministrator.
- Create a project with File >New >Project
- In the New Project dialog, choose Cloud > Service Fabric Application.
- Name the application Voting and click OK.
- On the New Service Fabric Servicepage, chooseStateless ASP.NET Core, name your serviceVotingWeb, then click OK.

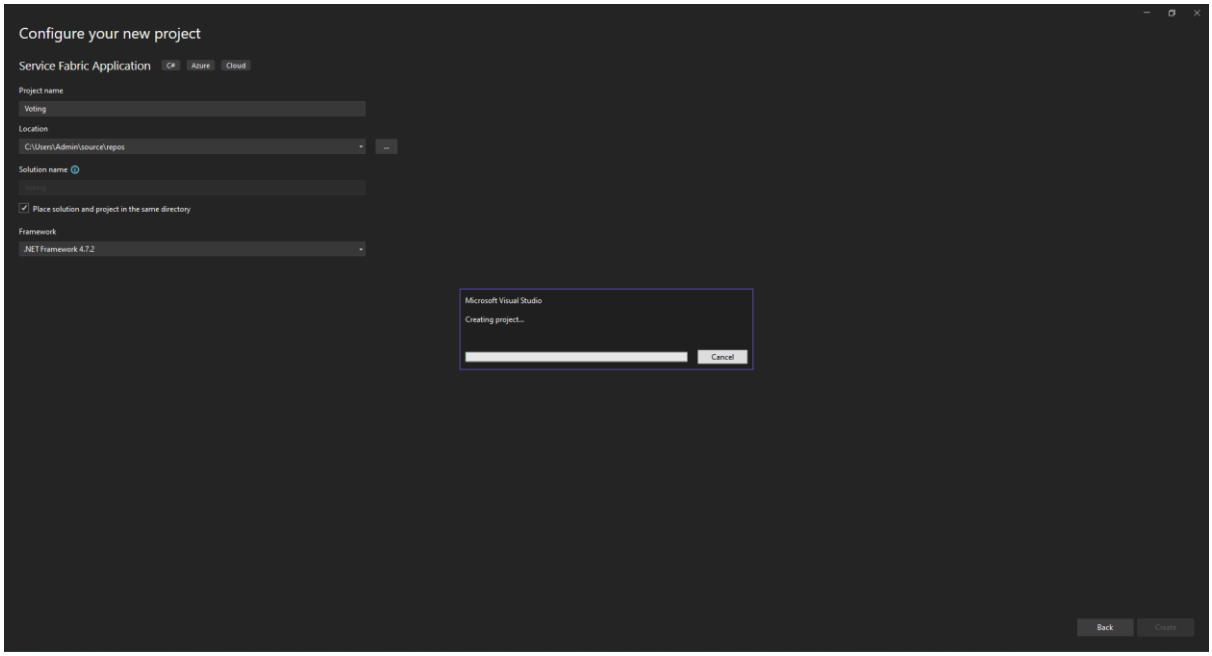
Step 2: The next page provides a set of ASP.NET Core project templates. For this practical, chooseWeb Application (Model-View-Controller), then click OK.

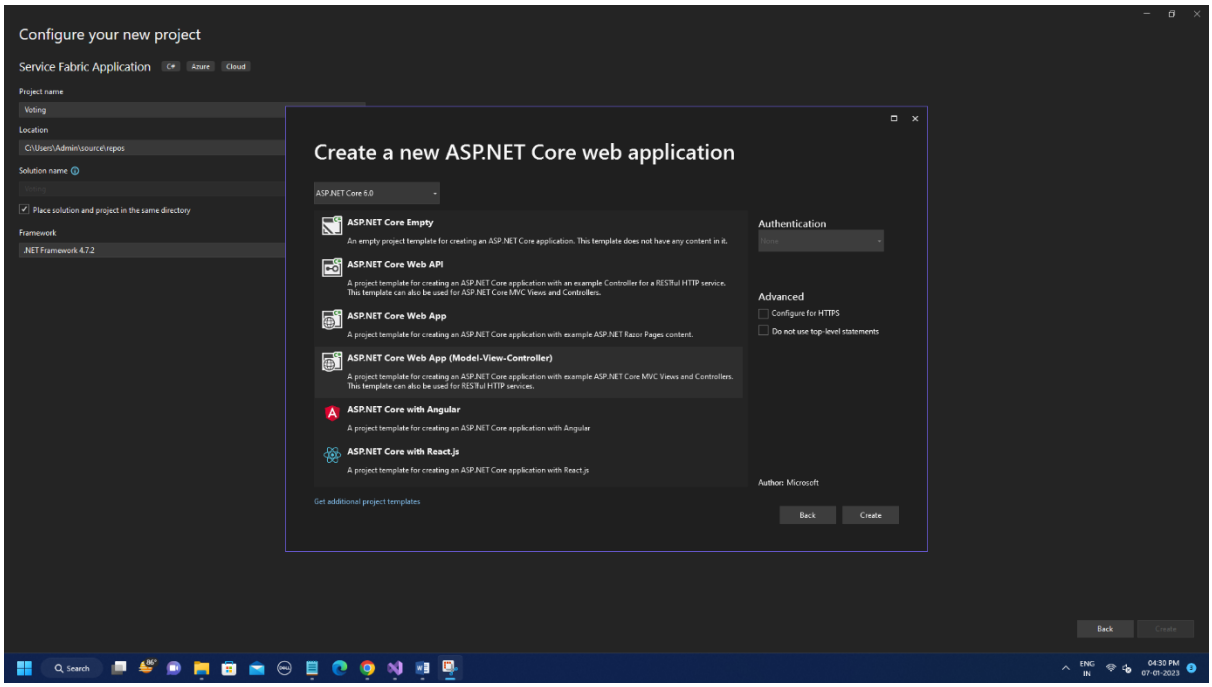
Step 3: Visual Studio creates an application and a service project and displays them in Solution Explorer.

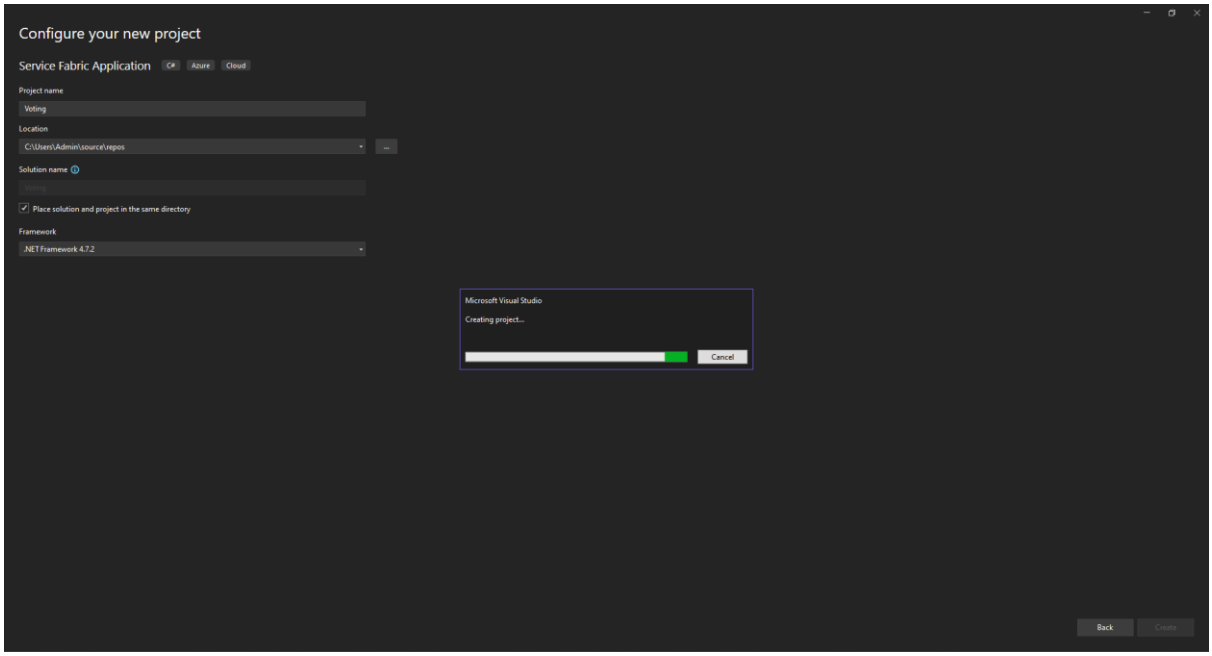



Configure your new project
Service Fabric Application
C#
Azure
Cloud
Project name
Voting
Location
C:\Users\Admin\source\repos
Solution name
Voting
Place solution and project in the same directory
Framework
.NET Framework 4.7.2
Back
Create
Search
ENG
IN
04:26 PM
07-01-2023


Configure your new project
Service Fabric Application
C#
Azure
Cloud
Project name
Voting
Location
C:\Users\Admin\source\repos
Solution name
Voting
Place solution and project in the same directory
Framework
.NET Framework 4.7.2
Create a new Service Fabric service
.NET Core
Stateless Service
Build a .NET Core stateless service.
Stateful Service
Build a stateful .NET Core service with persistent internal state using the reliable collections framework.
Actor Service
Build a .NET Core service using the Virtual Actor pattern.
Stateless ASP.NET Core
Build an ASP.NET Core stateless service.
Stateful ASP.NET Core
Build a stateful ASP.NET Core service with persistent internal state using the reliable collections framework.
Hosted Containers and Applications
Guest Executable
Run a self-contained application (such as a Node.js, Java, or native application) in your Service Fabric cluster.
Container
Optimize project layout for ARM deployment. Learn More
Help me choose a project template
Service name
Web1
Location
<Solution Path>\Web1\
Back
Create
Back
Create
Search
ENG
IN
04:27 PM
07-01-2023


Configure your new project
Service Fabric Application
C#
Azure
Cloud
Project name
Voting
Location
C:\Users\Admin\source\repos
Solution name
Voting
Place solution and project in the same directory
Framework
.NET Framework 4.7.2
Microsoft Visual Studio
Creating project...
Cancel
Back
Create


Configure your new project
Service Fabric Application
C#
Azure
Cloud
Project name
Voting
Location
C:\Users\Admin\source\repos
Solution name
Voting
Place solution and project in the same directory
Framework
.NET Framework 4.7.2
Create a new ASP.NET Core web application
ASP.NET Core 6.0
ASP.NET Core Empty
An empty project template for creating an ASP.NET Core application. This template does not have any content in it.
ASP.NET Core Web API
A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.
ASP.NET Core Web App
A project template for creating an ASP.NET Core application with example ASP.NET Razor Pages content.
ASP.NET Core Web App (Model-View-Controller)
A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.
ASP.NET Core with Angular
A project template for creating an ASP.NET Core application with Angular
ASP.NET Core with React.js
A project template for creating an ASP.NET Core application with React.js
Get additional project templates
Authentication
None
Advanced
Configure for HTTPS
Do not use top-level statements
Author: Microsoft
Back
Create
Back
Create
Search
ENG
IN
04:50 PM
07-01-2023

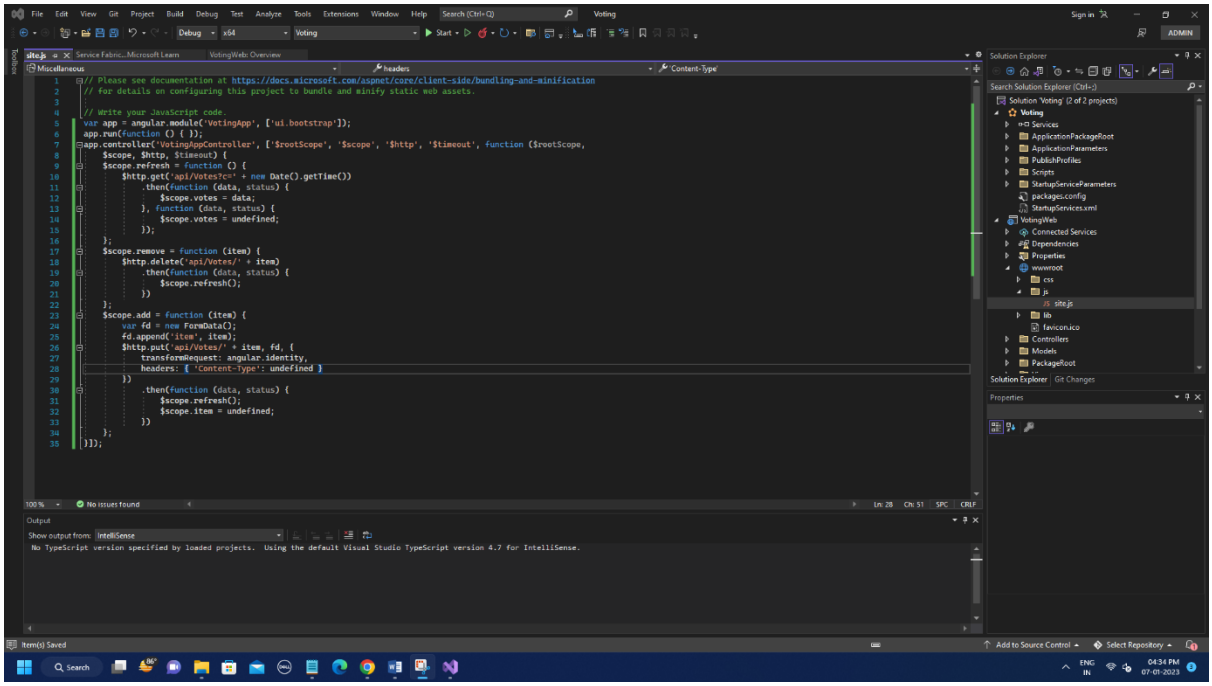


Step 4: 2.1 Update the site.js file Open wwwroot/js/site.js.

Replace its contents with the following JavaScript used by the Home views, then save your changes.

```
var app = angular.module('VotingApp', ['ui.bootstrap']); app.run(function () { });

app.controller('VotingAppController', ['$rootScope', '$scope', '$http', '$timeout', function ($rootScope,
    $scope, $http, $timeout) {

    $scope.refresh = function () {
        $http.get('api/Votes?c=' + new Date().getTime())
            .then(function (data, status) {
                $scope.votes = data;
            }, function (data, status) {
                $scope.votes = undefined;
            });
    };

    $scope.remove = function (item) {
```

```
        $http.delete('api/Votes/' + item)
            .then(function (data, status) {
                $scope.refresh();
            })
    };

    $scope.add = function (item) {
        var fd = new FormData(); fd.append('item', item);
        $http.put('api/Votes/' + item, fd, {
            transformRequest: angular.identity, headers: { 'Content-Type': undefined }
        })
            .then(function (data, status) {
                $scope.refresh();
                $scope.item = undefined;
            })
    };
}]);
```

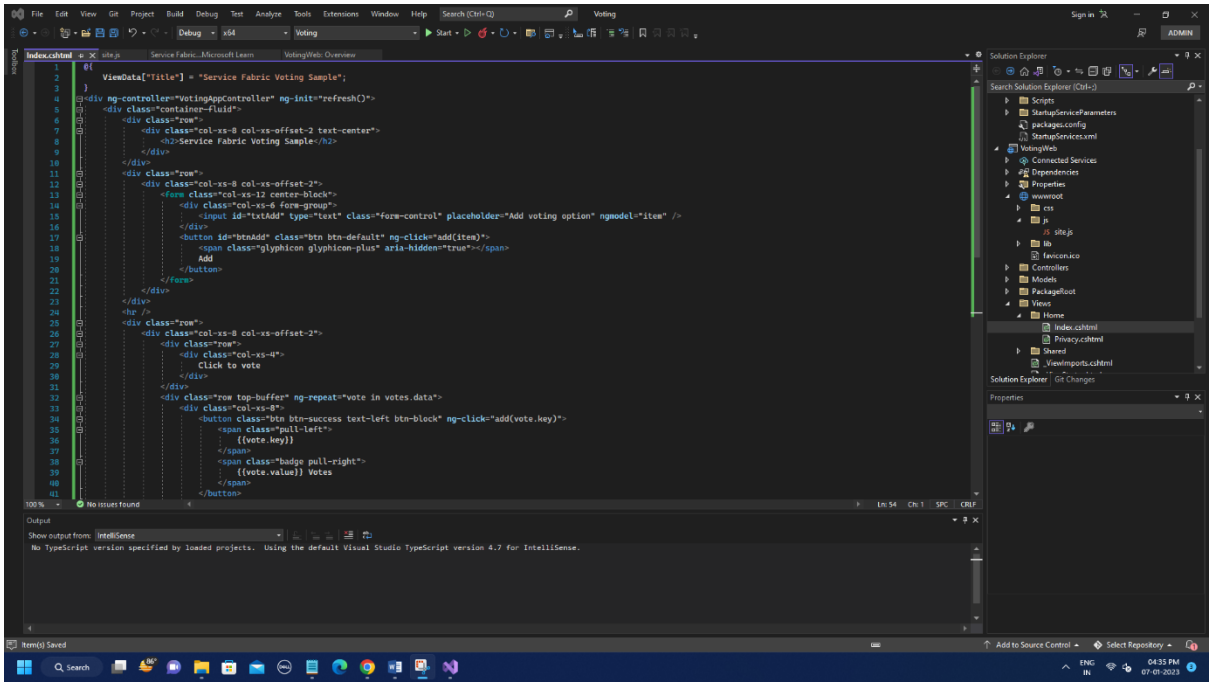


Step 5: 2.2 Update the Index.cshtml file Open Views/Home/Index.cshtml, the view specific to the Home controller.

Replace its contents with the following, then save your changes.

```
@{
   ViewData["Title"] = "Service Fabric Voting Sample";
}
<div ng-controller="VotingAppController" ng-init="refresh()">
   <div class="container-fluid">
      <div class="row">
         <div class="col-xs-8 col-xs-offset-2 text-center">
            <h2>Service Fabric Voting Sample</h2>
         </div>
      </div>
      <div class="row">
         <div class="col-xs-8 col-xs-offset-2">
            <form class="col-xs-12 center-block">
               <div class="col-xs-6 form-group">
                  <input id="txtAdd" type="text" class="form-control" placeholder="Add voting
option" ng- model="item" />
               </div>
               <button id="btnAdd" class="btn btn-default" ng-click="add(item)">
                  <span class="glyphicon glyphicon-plus" aria-hidden="true"></span> Add
               </button>
            </form>
         </div>
      </div>
      <hr />
      <div class="row">
         <div class="col-xs-8 col-xs-offset-2">
            <div class="row">
               <div class="col-xs-4"> Click to vote
```

```
            </div>
          </div>
          <div class="row top-buffer" ng-repeat="vote in votes.data">
            <div class="col-xs-8">
              <button class="btn btn-success text-left btn-block" ng-click="add(vote.key)">
                <span class="pull-left">
                  {{ vote.key }}
                </span>
                <span class="badge pull-right">
                  {{ vote.value }} Votes
                </span>
              </button>
            </div>
            <div class="col-xs-4">
              <button class="btn btn-danger pull-right btn-block" ng-click="remove(vote.key)">
                <span class="glyphicon glyphicon-remove" aria-hidden="true"></span> Remove
              </button>
            </div>
          </div>
        </div>
      </div>
    </div>
</div>
```

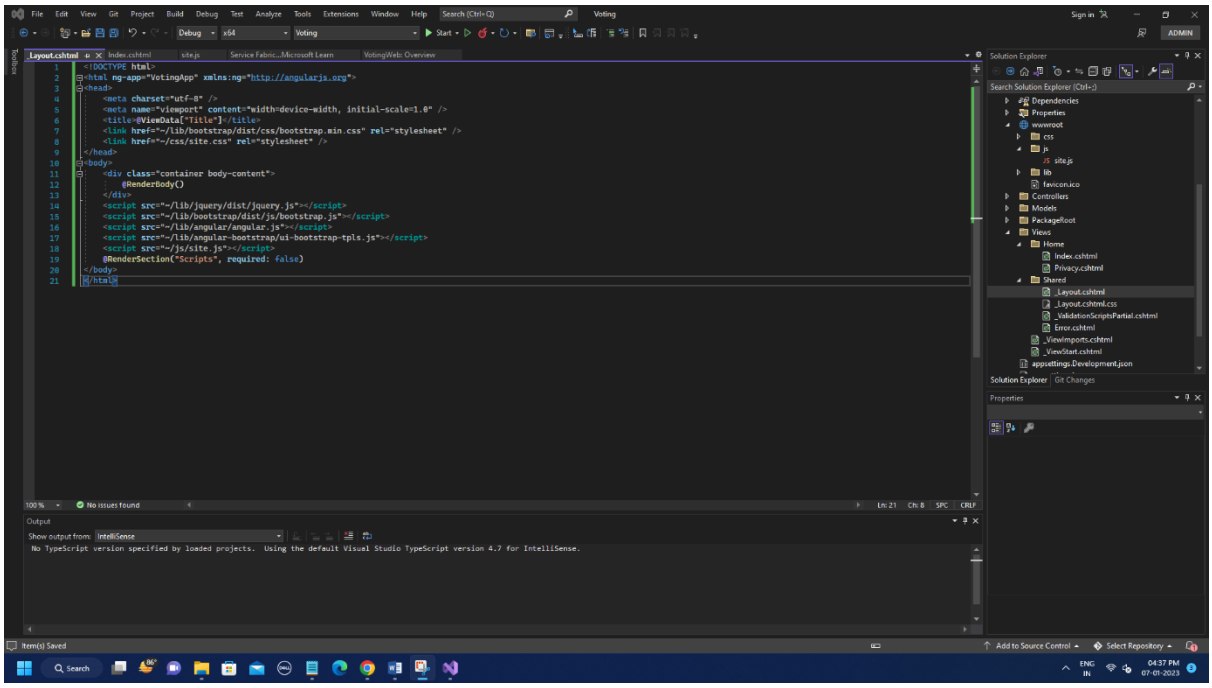


Step 6: Update the _Layout.cshtml file

Open Views/Shared/_Layout.cshtml, the default layout for the ASP.NET app. Replace its contents with the following, then save your changes.

```
<!DOCTYPE html>
<html ng-app="VotingApp" xmlns:ng="http://angularjs.org">
    <head>
        <meta charset="utf-8"/>
        <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
        <title>@ViewData["Title"]</title>

        <link href="~/lib/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet"/>
        <link href="~/css/site.css" rel="stylesheet"/>

    </head>
        <body>
            <div class="container body-content"> @RenderBody()
            </div>
            <script src="~/lib/jquery/dist/jquery.js"></script>
            <script src="~/lib/bootstrap/dist/js/bootstrap.js"></script>
```

```
                <script src="~/lib/angular/angular.js"></script>
                <script src="~/lib/angular-bootstrap/ui-bootstrap-tpls.js"></script>
                <script src="~/js/site.js"></script>
                @RenderSection("Scripts", required: false)
            </body>
</html>
```



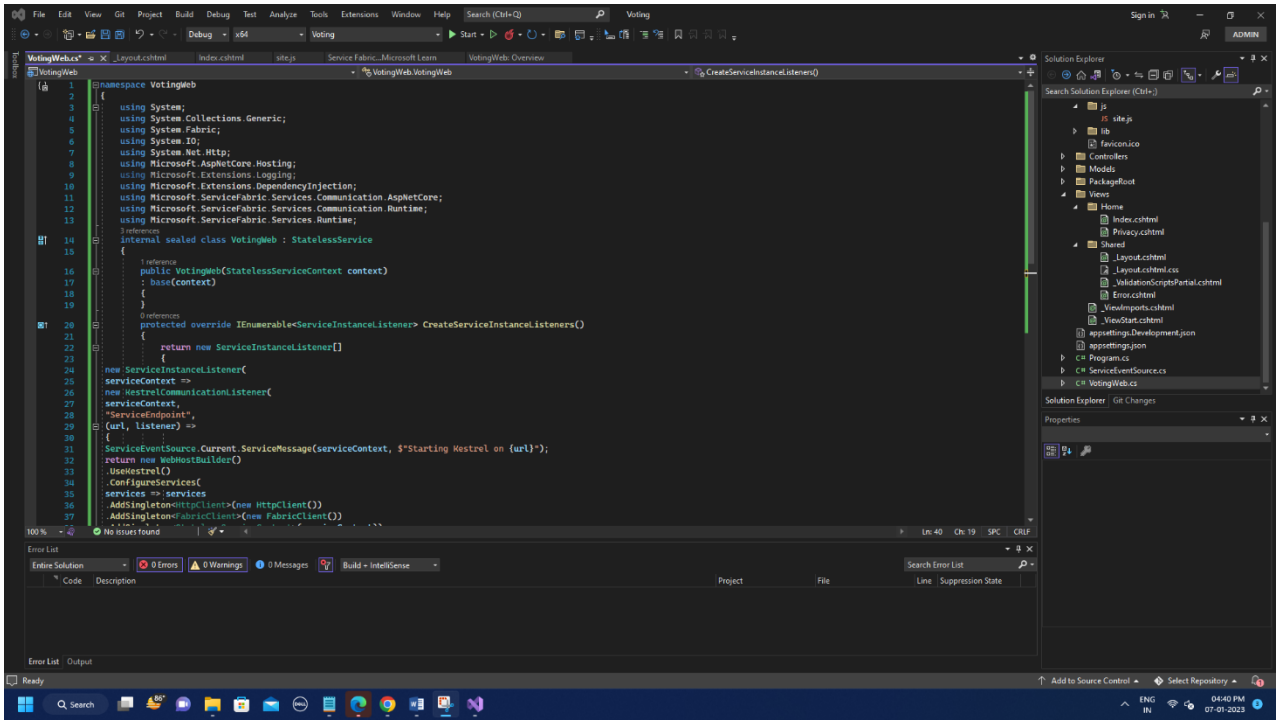
Step 7: Update the VotingWeb.cs file

Open the VotingWeb.cs file, which creates the ASP.NET Core WebHost inside the stateless service using the WebListener web server.Replace the content with the following code, then save your changes.

```
namespace VotingWeb
{
   using System;
   using System.Collections.Generic;
   using System.Fabric;
   using System.IO;
   using System.Net.Http;
```

```
using Microsoft.AspNetCore.Hosting;
using Microsoft.Extensions.Logging;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.ServiceFabric.Services.Communication.AspNetCore;
using Microsoft.ServiceFabric.Services.Communication.Runtime;
using Microsoft.ServiceFabric.Services.Runtime;
internal sealed class VotingWeb : StatelessService
{
   public VotingWeb(StatelessServiceContext context)
   : base(context)
   {
   }
   protected override IEnumerable < ServiceInstanceListener > CreateServiceInstanceListeners()
   {
      return new ServiceInstanceListener[]
      {
         new ServiceInstanceListener(serviceContext =>

            new KestrelCommunicationListener(serviceContext, "ServiceEndpoint",
               (url, listener) => {
                  ServiceEventSource.Current.ServiceMessage(serviceContext, $"Starting Kestrel on {url}");

                  return new WebHostBuilder()
                     .UseKestrel()
                     .ConfigureServices(services => services
                        .AddSingleton < HttpClient > (new HttpClient())
                           .AddSingleton < FabricClient > (new FabricClient())
                           .AddSingleton < StatelessServiceContext > (serviceContext))
                     .UseContentRoot(Directory.GetCurrentDirectory())
                     .UseStartup < Startup > ()
                        .UseServiceFabricIntegration(listener, ServiceFabricIntegrationOptions.None)
```

```
                    .UseUrls(url)
                    .Build();
                }))
        };
    }
internal static Uri GetVotingDataServiceName(ServiceContext context)
    {
        return new Uri($"{context.CodePackageActivationContext.ApplicationName}/VotingData");
    }
  }
}
```



Step 8: Add the VotesController.cs file

Add a controller, which defines voting actions. Right-click on the Controllers folder, then select Add->New item->Visual C#->ASP.NET Core->Class. Name the file VotesController.cs, then click Add.

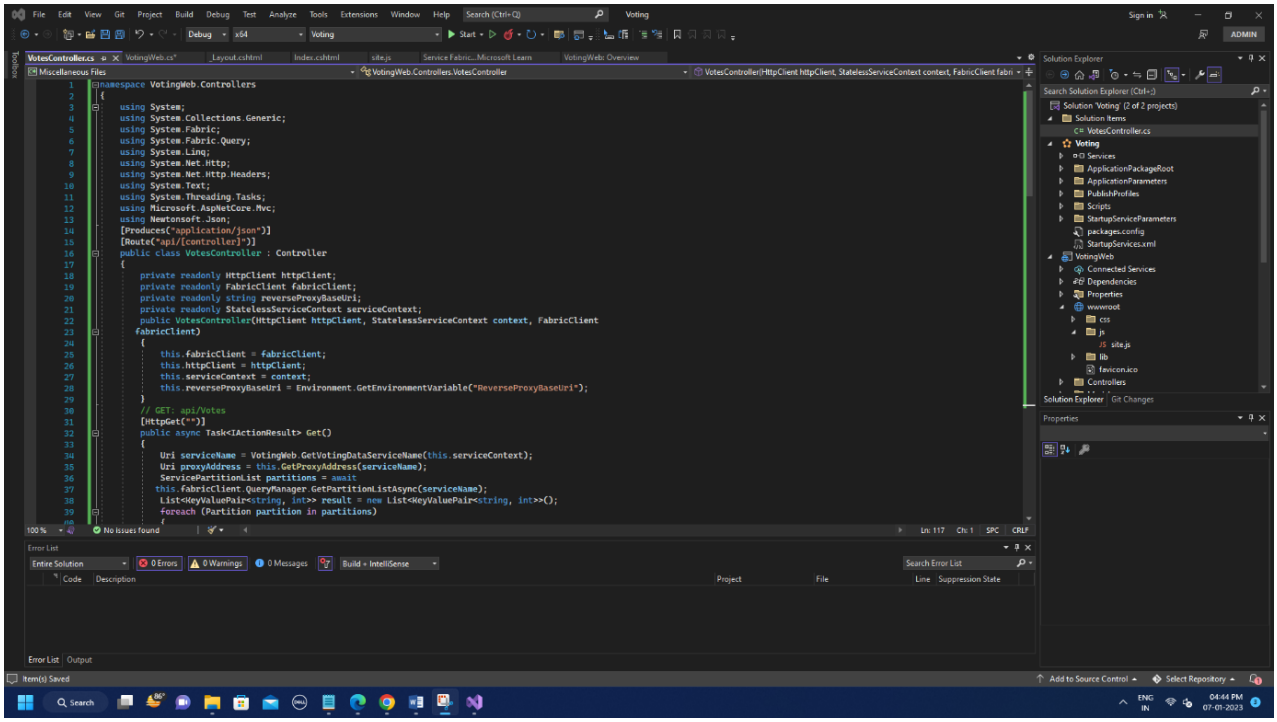
Replace the VotesController.cs file contents with the following, then save your changes this file is modified to read and write voting data from the back-end service.

```
namespace VotingWeb.Controllers {
using System;
using System.Collections.Generic; using System.Fabric;
using System.Fabric.Query; using System.Linq;
using System.Net.Http;
using System.Net.Http.Headers; using System.Text;
using System.Threading.Tasks; using Microsoft.AspNetCore.Mvc; using Newtonsoft.Json;

   [Produces("application/json")][Route("api/[controller]")]
   public class VotesController : Controller
   {
private readonly HttpClient httpClient; private readonly FabricClient fabricClient; private readonly string reverseProxyBaseUri;
private readonly StatelessServiceContext serviceContext;

public VotesController(HttpClient httpClient, StatelessServiceContext context, FabricClient fabricClient)
      {
         this.fabricClient = fabricClient; this.httpClient = httpClient; this.serviceContext = context;
         this.reverseProxyBaseUri = Environment.GetEnvironmentVariable("ReverseProxyBaseUri");
      }

// GET: api/Votes [HttpGet("")]
public async Task < IActionResult > Get()
      {
Uri serviceName = VotingWeb.GetVotingDataServiceName(this.serviceContext); Uri proxyAddress = this.GetProxyAddress(serviceName);
```

```
ServicePartitionList partitions = await this.fabricClient.QueryManager.GetPartitionListAsync(serviceName);
        List < KeyValuePair < string, int >> result = new List < KeyValuePair < string, int >> (); foreach(Partition partition in partitions)
        {
string proxyUrl =

$"{proxyAddress}/api/VoteData?PartitionKey={((Int64RangePartitionInformation) partition.PartitionInformation).LowKey}&PartitionKind=Int64Range";

            using(HttpResponseMessage response = await this.httpClient.GetAsync(proxyUrl))
            {
                if (response.StatusCode != System.Net.HttpStatusCode.OK) {
                    continue;
                }

                result.AddRange(JsonConvert.DeserializeObject < List < KeyValuePair < string, int >>> (await response.Content.ReadAsStringAsync()));
            }
        }

        return this.Json(result);
    }

// PUT: api/Votes/name [HttpPut("{name}")]
public async Task < IActionResult > Put(string name)
    {
Uri serviceName = VotingWeb.GetVotingDataServiceName(this.serviceContext); Uri proxyAddress = this.GetProxyAddress(serviceName);
long partitionKey = this.GetPartitionKey(name); string proxyUrl =

$"{proxyAddress}/api/VoteData/{name}?PartitionKey={partitionKey}&PartitionKind=Int64Range";
```

```
StringContent putContent = new StringContent($"{{ 'name' : '{name}' }}", Encoding.UTF8, "application/json");
        putContent.Headers.ContentType = new MediaTypeHeaderValue("application/json");

        using(HttpResponseMessage response = await this.httpClient.PutAsync(proxyUrl, putContent))
        {
           return new ContentResult()
           {
              StatusCode = (int) response.StatusCode,
                 Content = await response.Content.ReadAsStringAsync()
           };
        }
     }

// DELETE: api/Votes/name [HttpDelete("{name}")]
public async Task < IActionResult > Delete(string name)
     {
Uri serviceName = VotingWeb.GetVotingDataServiceName(this.serviceContext); Uri proxyAddress = this.GetProxyAddress(serviceName);
long partitionKey = this.GetPartitionKey(name); string proxyUrl =

$"{proxyAddress}/api/VoteData/{name}?PartitionKey={partitionKey}&PartitionKind=Int64Range";

        using(HttpResponseMessage response = await this.httpClient.DeleteAsync(proxyUrl))
        {
           if (response.StatusCode != System.Net.HttpStatusCode.OK) {
              return this.StatusCode((int) response.StatusCode);
           }
        }

        return new OkResult();
     }
```

```
/// <summary>
/// Constructs a reverse proxy URL for a given service.
/// Example: http://localhost:19081/VotingApplication/VotingData/
/// </summary>
/// <param name="serviceName"></param>
/// <returns></returns>
private Uri GetProxyAddress(Uri serviceName)
        {
            return new Uri($"{this.reverseProxyBaseUri}{serviceName.AbsolutePath}");
        }

/// <summary>
/// Creates a partition key from the given name.
/// Uses the zero-based numeric position in the alphabet of the first letter of the name (0-25).
/// </summary>
/// <param name="name"></param>
/// <returns></returns>
private long GetPartitionKey(string name)
        {
            return Char.ToUpper(name.First()) - 'A';
        }
    }
}
```



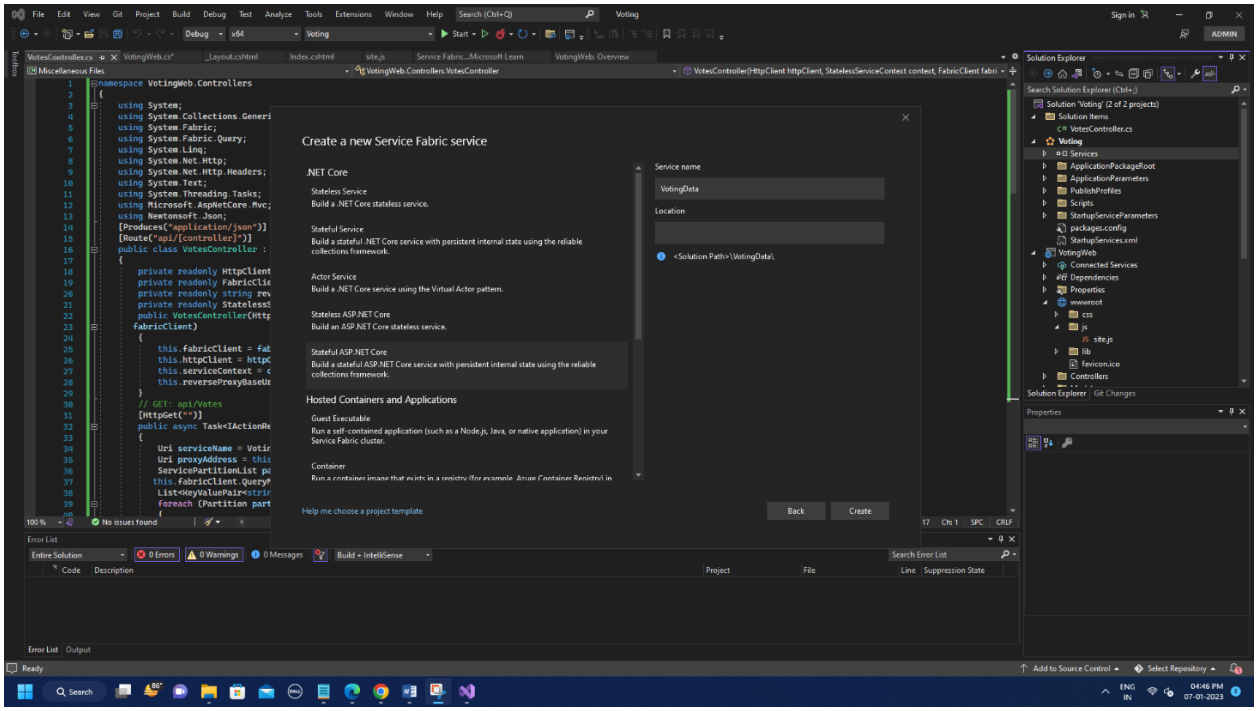
Step 9: Add a stateful back-end service to your application

- In Solution Explorer, right-click Services within the Voting application project and choose Add -> New Service Fabric Service....
- In the New Service Fabric Service dialog, choose Stateful ASP.NET Core, name the service VotingData, then press Create.

Once your service project is created, you have two services in your application. As you continue to build your application, you can add more services in the same way. Each can be independently versioned and upgraded.

The next page provides a set of ASP.NET Core project templates. choose API.

Visual Studio creates the VotingData service project and displays it in Solution Explorer.



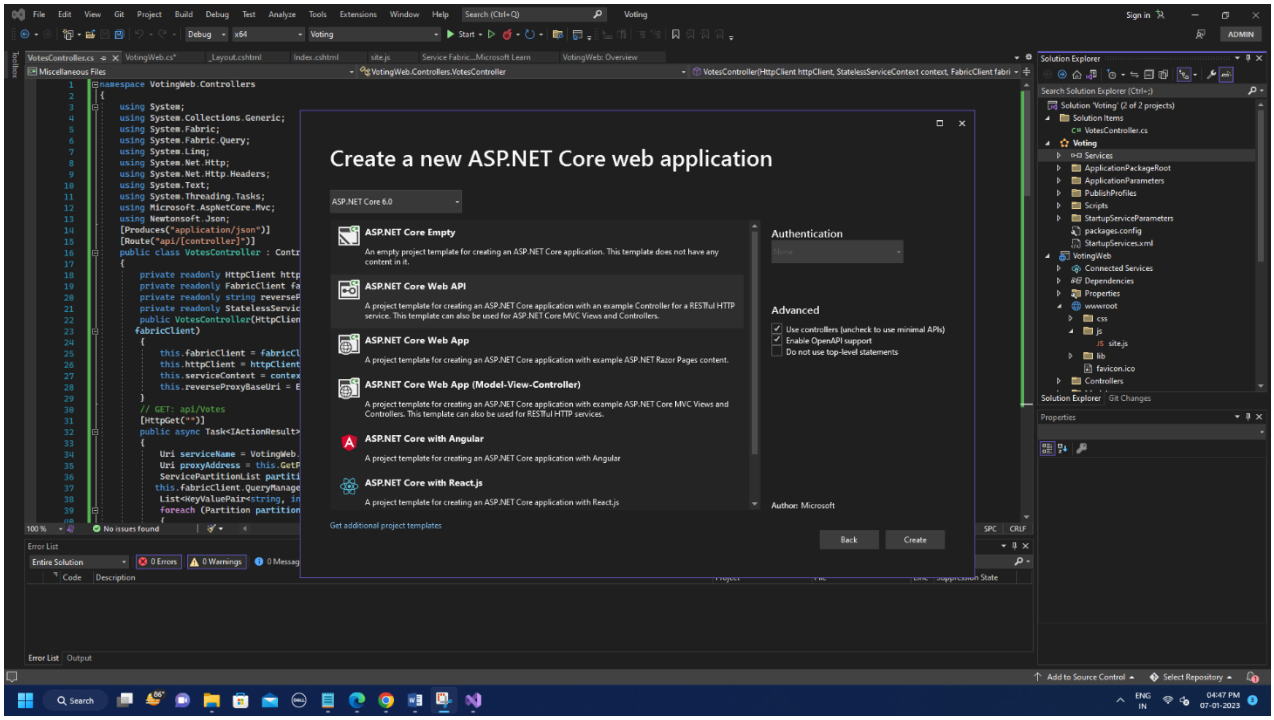
Step 10: Add the VoteDataController.cs file

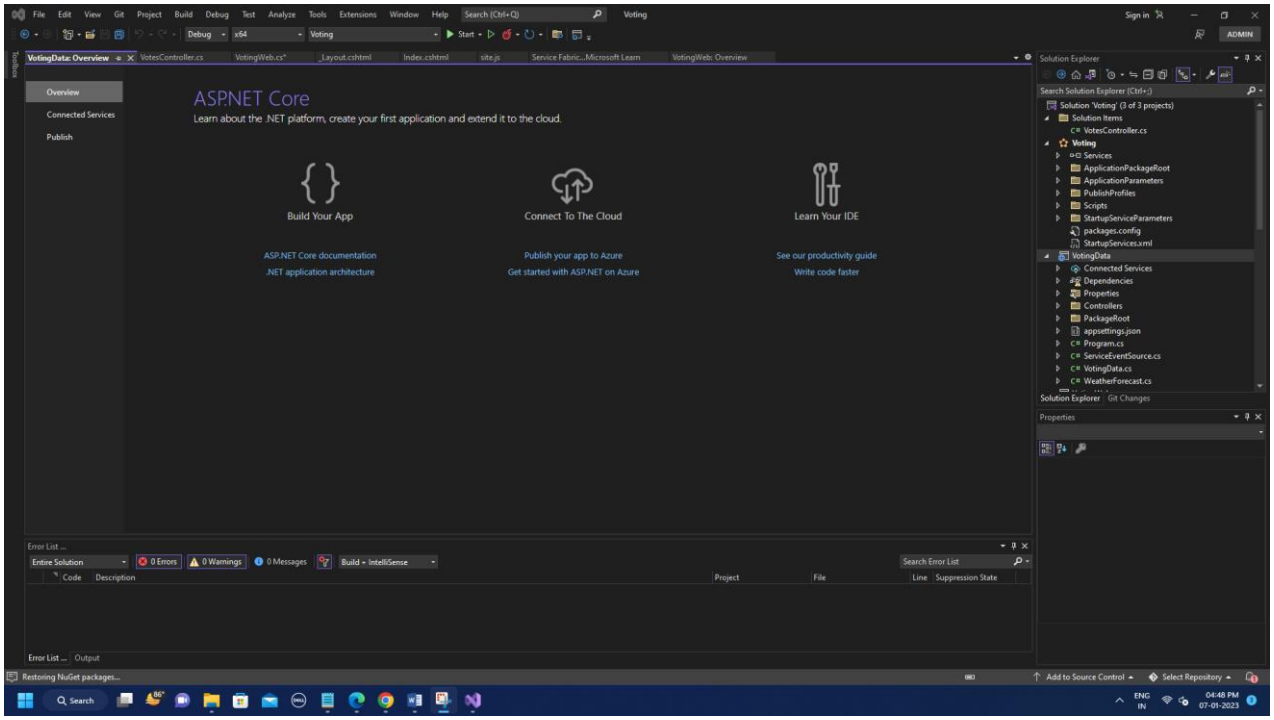
- In the VotingData project, right-click on the Controllers folder, then select Add->New item->Class. Name the file VoteDataController.cs and click Add.
- Replace the file contents with the following, then save your changes.

```
namespace VotingData.Controllers {
using System.Collections.Generic;
using System.Threading;
using System.Threading.Tasks; using Microsoft.AspNetCore.Mvc; using Microsoft.ServiceFabric.Data;
using Microsoft.ServiceFabric.Data.Collections;


   [Route("api/[controller]")]
   public class VoteDataController : Controller
   {
private readonly IReliableStateManager stateManager;
public VoteDataController(IReliableStateManager stateManager)
      {
         this.stateManager = stateManager;
      }
```

```
// GET api/VoteData [HttpGet]
public async Task < IActionResult > Get()
    {
CancellationToken ct = new CancellationToken();
        IReliableDictionary < string, int > votesDictionary =
        await this.stateManager.GetOrAddAsync < IReliableDictionary < string, int >> ("counts");

        using(ITransaction tx = this.stateManager.CreateTransaction())
        {
          var list = await votesDictionary.CreateEnumerableAsync(tx); var enumerator = list.GetAsyncEnumerator();
          List < KeyValuePair < string, int >> result = new List < KeyValuePair < string, int >> ();

          while (await enumerator.MoveNextAsync(ct)) {
            result.Add(enumerator.Current);
          }

          return this.Json(result);
        }
    }

// PUT api/VoteData/name [HttpPut("{name}")]
public async Task < IActionResult > Put(string name)
    {
        IReliableDictionary < string, int > votesDictionary = await this.stateManager.GetOrAddAsync < IReliableDictionary < string, int >> ("counts");
        using(ITransaction tx = this.stateManager.CreateTransaction())
        {
          await votesDictionary.AddOrUpdateAsync(tx, name, 1, (key, oldvalue) => oldvalue + 1); await tx.CommitAsync();
        }
```

```
            return new OkResult();
        }

// DELETE api/VoteData/name [HttpDelete("{name}")]
public async Task < IActionResult > Delete(string name)
        {
            IReliableDictionary < string, int > votesDictionary = await
this.stateManager.GetOrAddAsync < IReliableDictionary < string, int >> ("counts");

            using(ITransaction tx = this.stateManager.CreateTransaction())
            {
                if (await votesDictionary.ContainsKeyAsync(tx, name)) {
                    await votesDictionary.TryRemoveAsync(tx, name); await tx.CommitAsync();
                    return new OkResult();
                }
                else {
                    return new NotFoundResult();
                }
            }
        }
    }
}
```

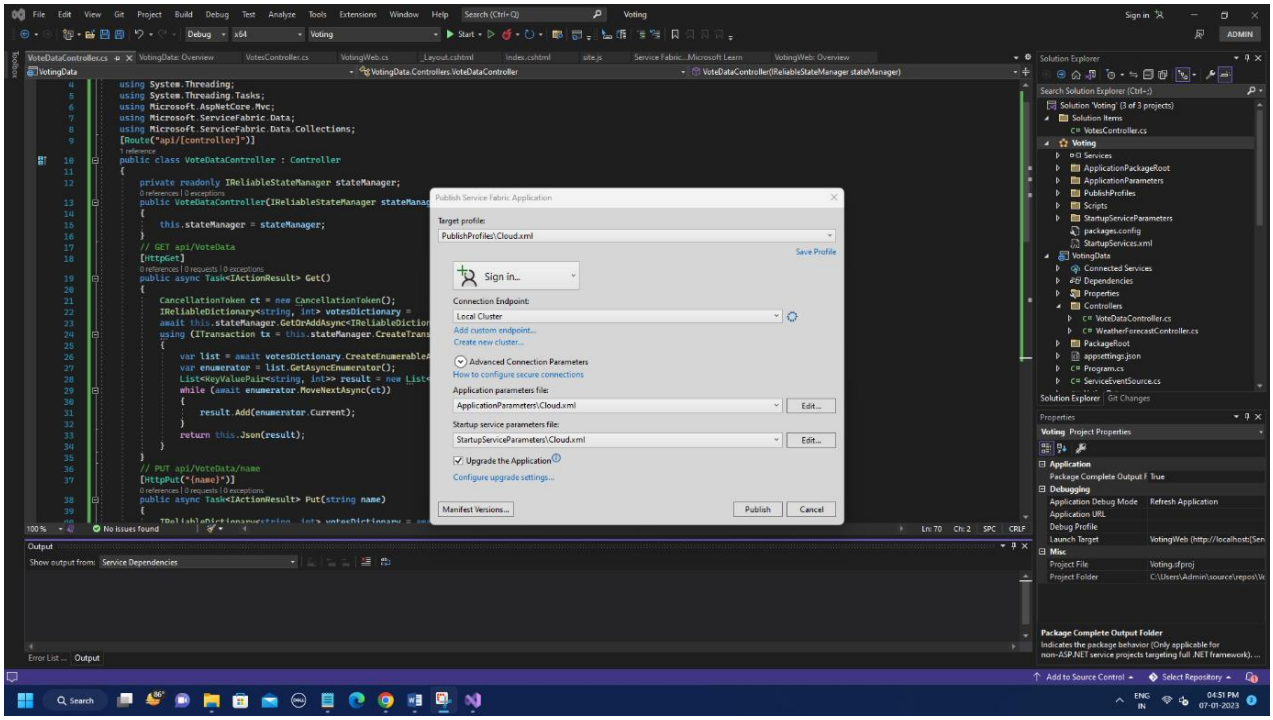


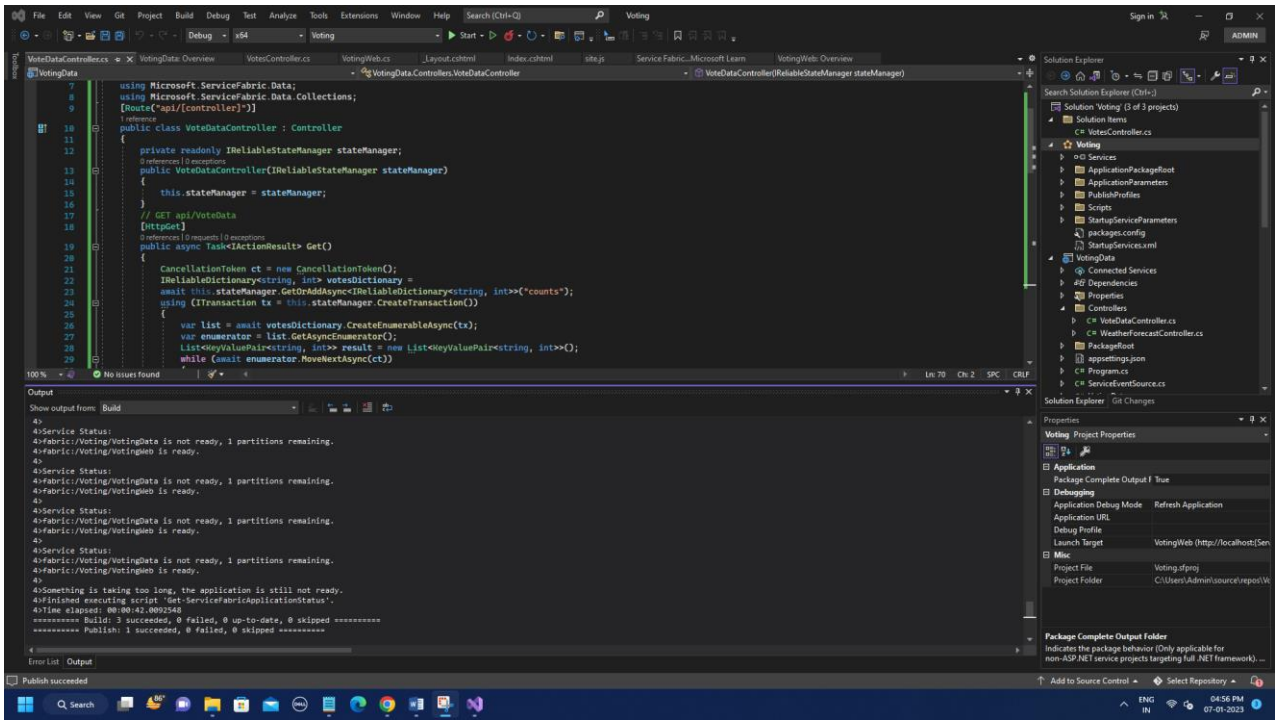


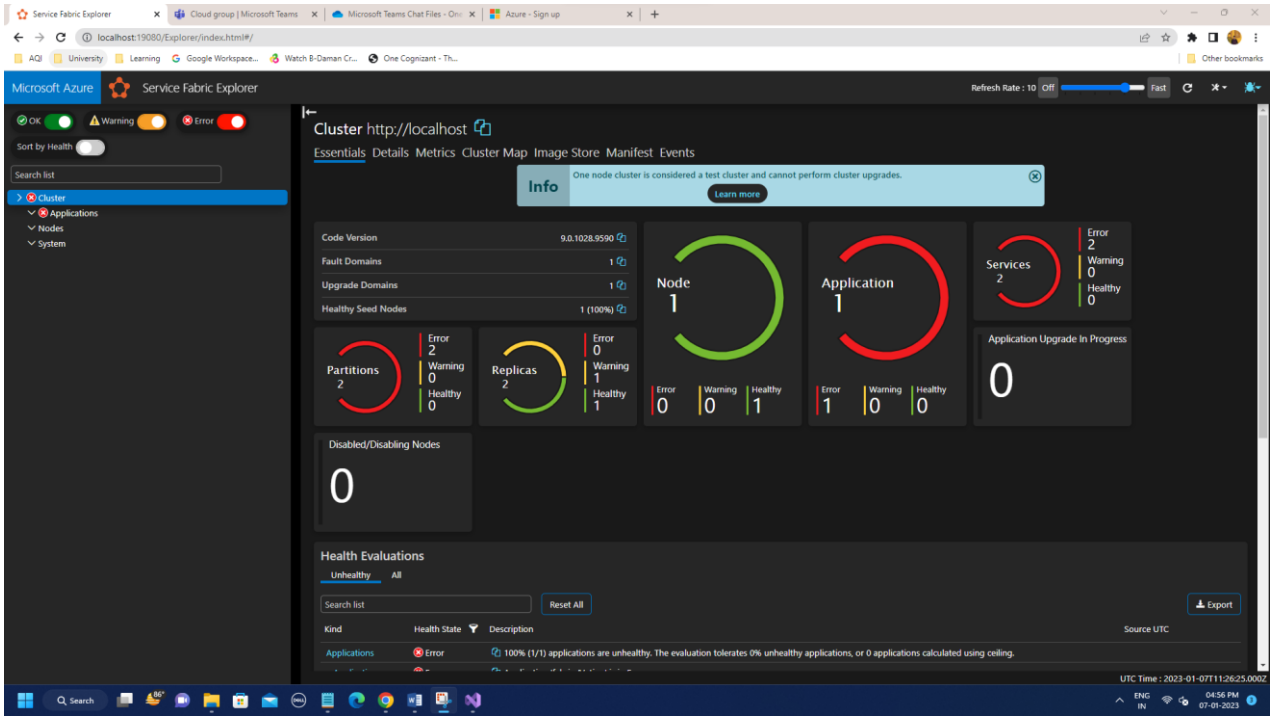
Step 11: Publish Service Fabric application

- In the solution Explorer Right click on the Voting and select Publish. The Publish dialog box appears.
- To open the service fabric explorer, open the Browser and paste the following IP address http://localhost:19080/

- Go to cluster >Application >Voting Type >fabric:/Voting >fabric:/VotingWeb > 8482ef73-476d- 45a6-aca9-b33d75575855(partition) >_Node_0 Instance
- You will see the endpoint in the address section click on that address to run your app






Microsoft Azure
Service Fabric Explorer
Cluster http://localhost
Essentials Details Metrics Cluster Map Image Store Manifest Events
Info
One node cluster is considered a test cluster and cannot perform cluster upgrades.
Learn more
Code Version 9.0.1028.9590
Fault Domains 1
Upgrade Domains 1
Healthy Seed Nodes 1 (100%)
Node 1
Application 1
Services 2
Partitions 2
Replicas 2
Application Upgrade In Progress 0
Disabled/Disabling Nodes 0
Health Evaluations
Unhealthy All
Reset All
Export
Kind
Health State
Description
Source UTC
Applications
Error
100% (1/1) applications are unhealthy. The evaluation tolerates 0% unhealthy applications, or 0 applications calculated using ceiling.

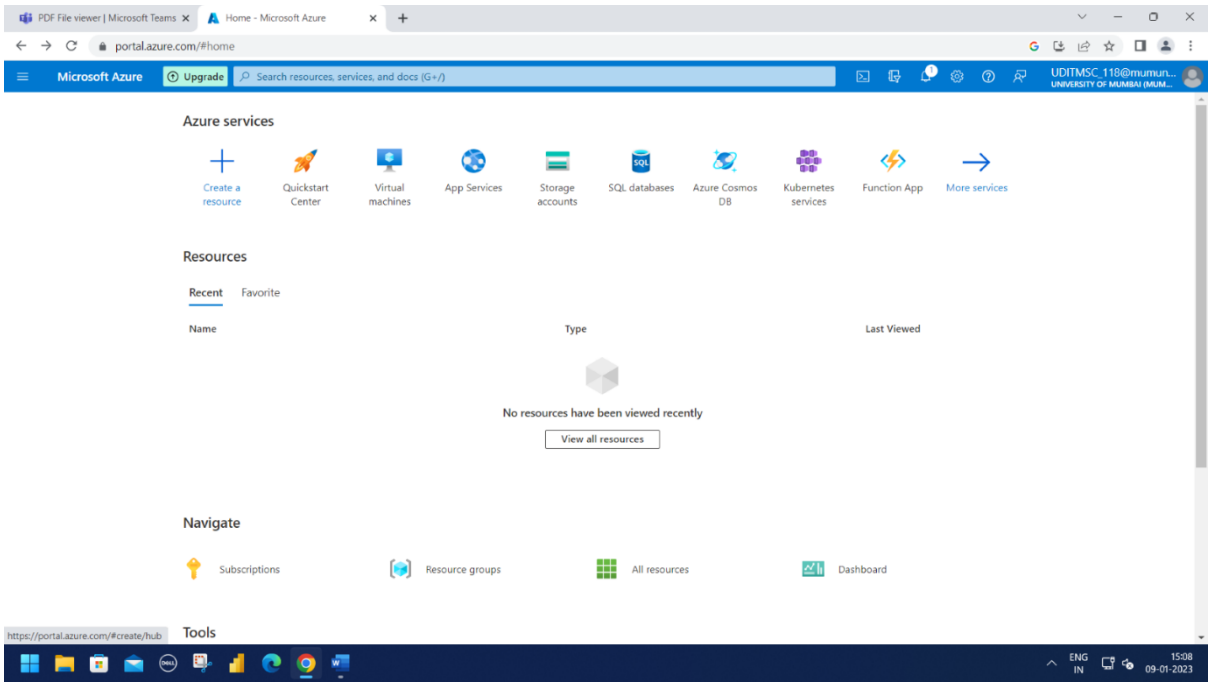
Practical No: 02

Aim: 2A. Create an Azure Kubernetes Service Cluster

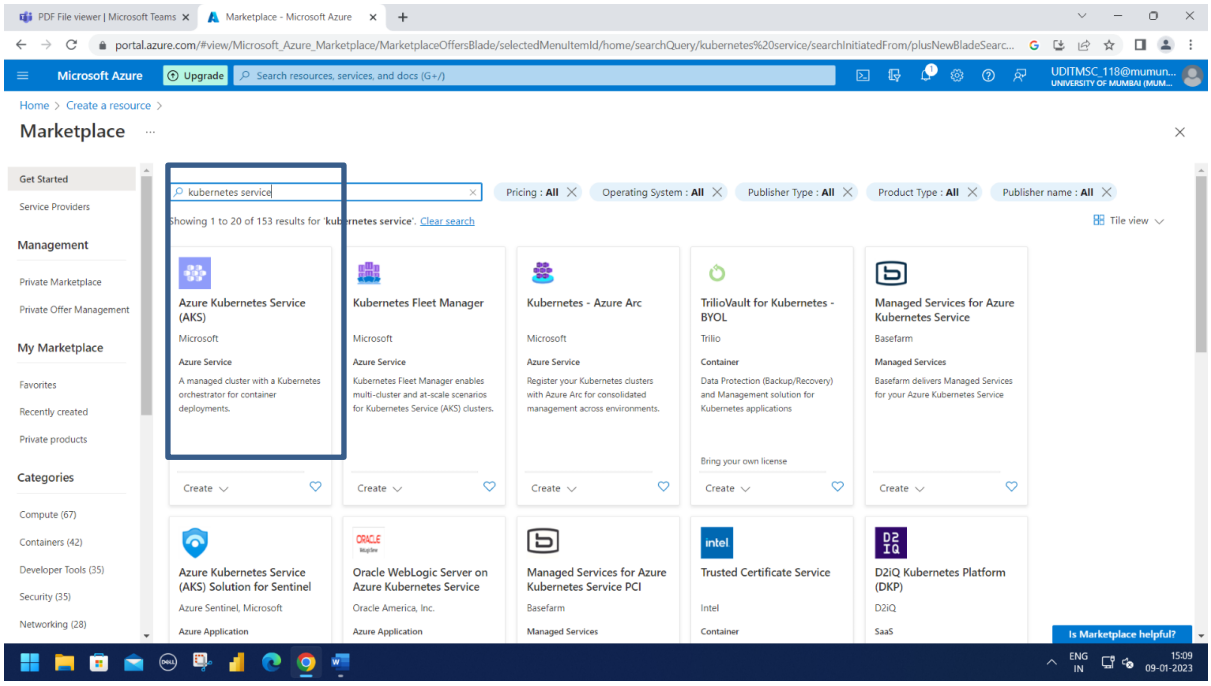
Source Code:

Step 1: Sign in Azure Portal (https://portal.azure.com/)

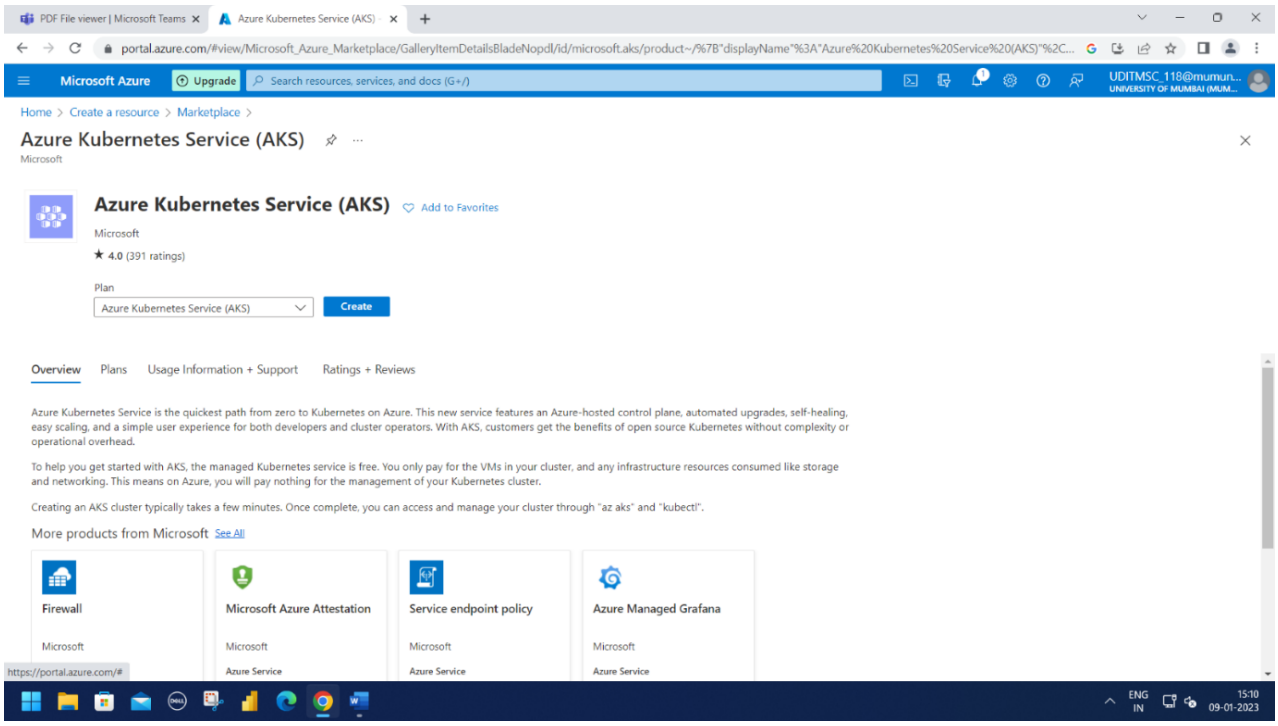
Step 2: Create Resource by clicking on Create Resource option.

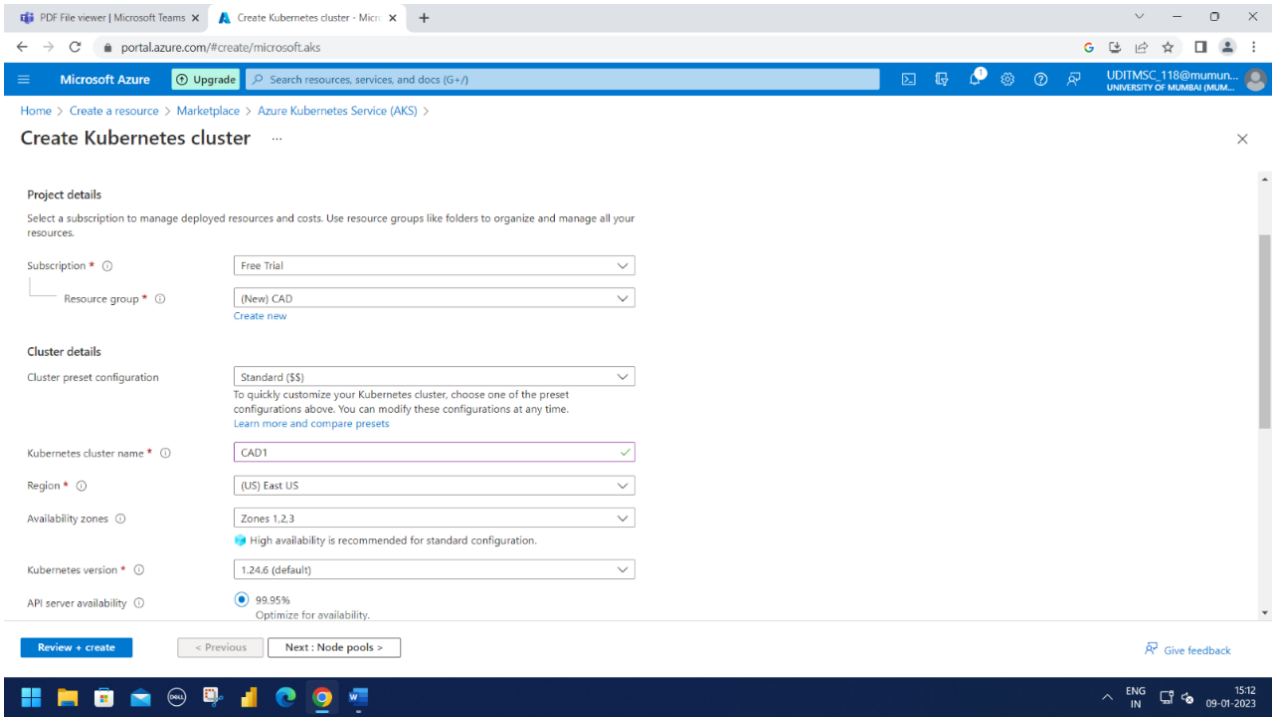


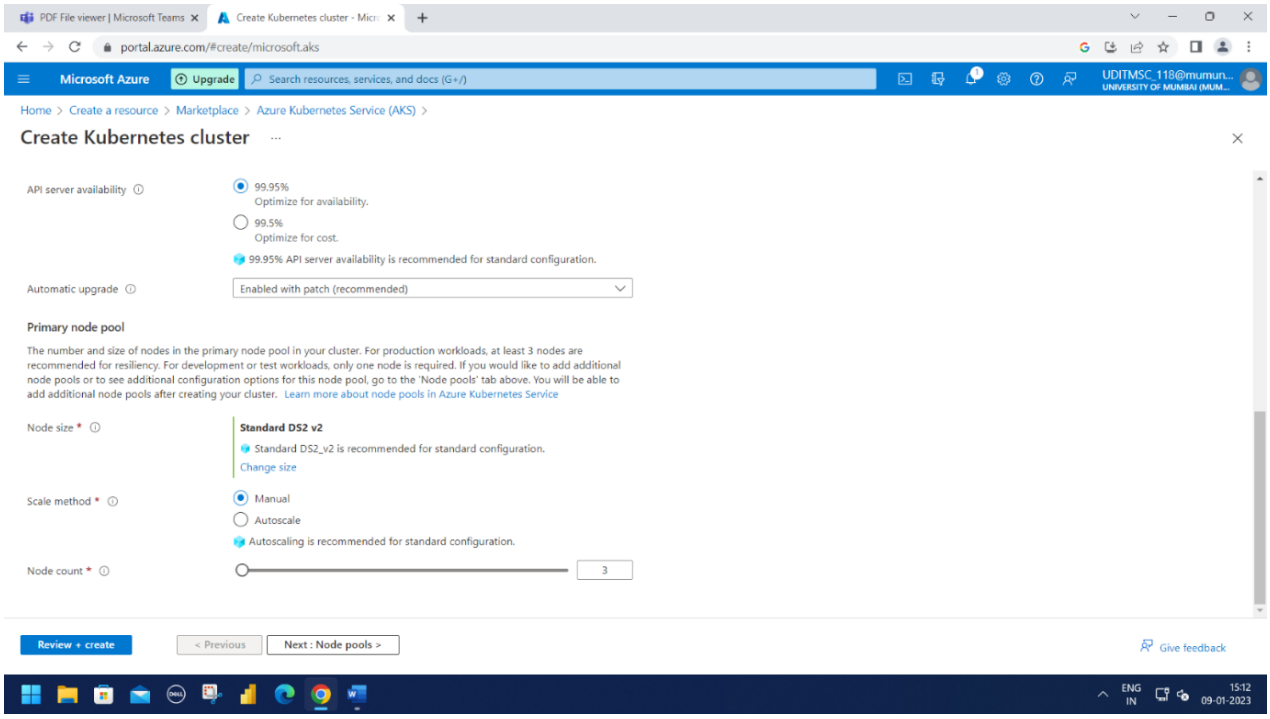
Step 3: Select Kubernetes Services

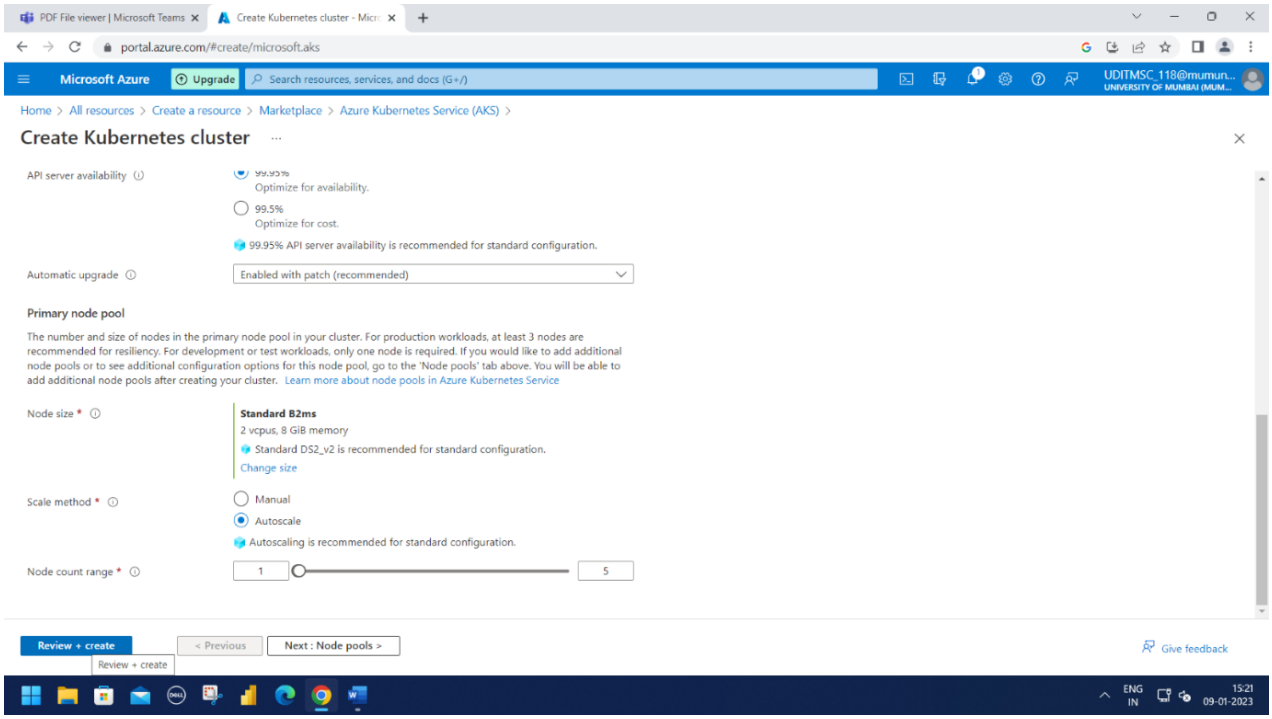


Step 4: Enter the subscription, resource group, kubernetes, Cluster name, Region, Kubernetes version,& DNS Name prefix

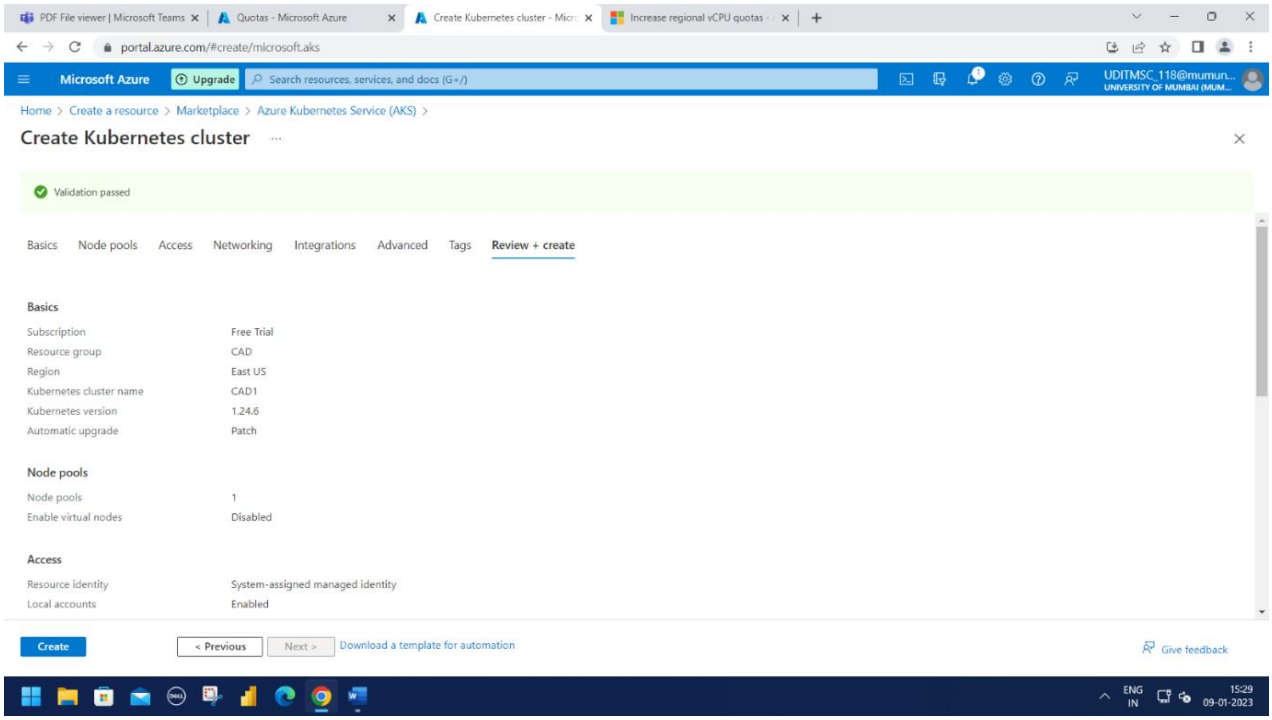



PDF File viewer | Microsoft Teams
Create Kubernetes cluster - Micr
portal.azure.com/#create/microsoft.aks
Microsoft Azure
Upgrade
Search resources, services, and docs (G+/)
UDITMSC_118@mumun...
UNIVERSITY OF MUMBAI (MUM...
Home > Create a resource > Marketplace > Azure Kubernetes Service (AKS) >
Create Kubernetes cluster
Project details
Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.
Subscription *
Free Trial
Resource group *
(New) CAD
Create new
Cluster details
Cluster preset configuration
Standard ($$)
To quickly customize your Kubernetes cluster, choose one of the preset configurations above. You can modify these configurations at any time.
Learn more and compare presets
Kubernetes cluster name *
CAD1
Region *
(US) East US
Availability zones
Zones 1,2,3
High availability is recommended for standard configuration.
Kubernetes version *
1.24.6 (default)
API server availability
99.95%
Optimize for availability.
Review + create
< Previous
Next : Node pools >
Give feedback
ENG
IN
15:12
09-01-2023

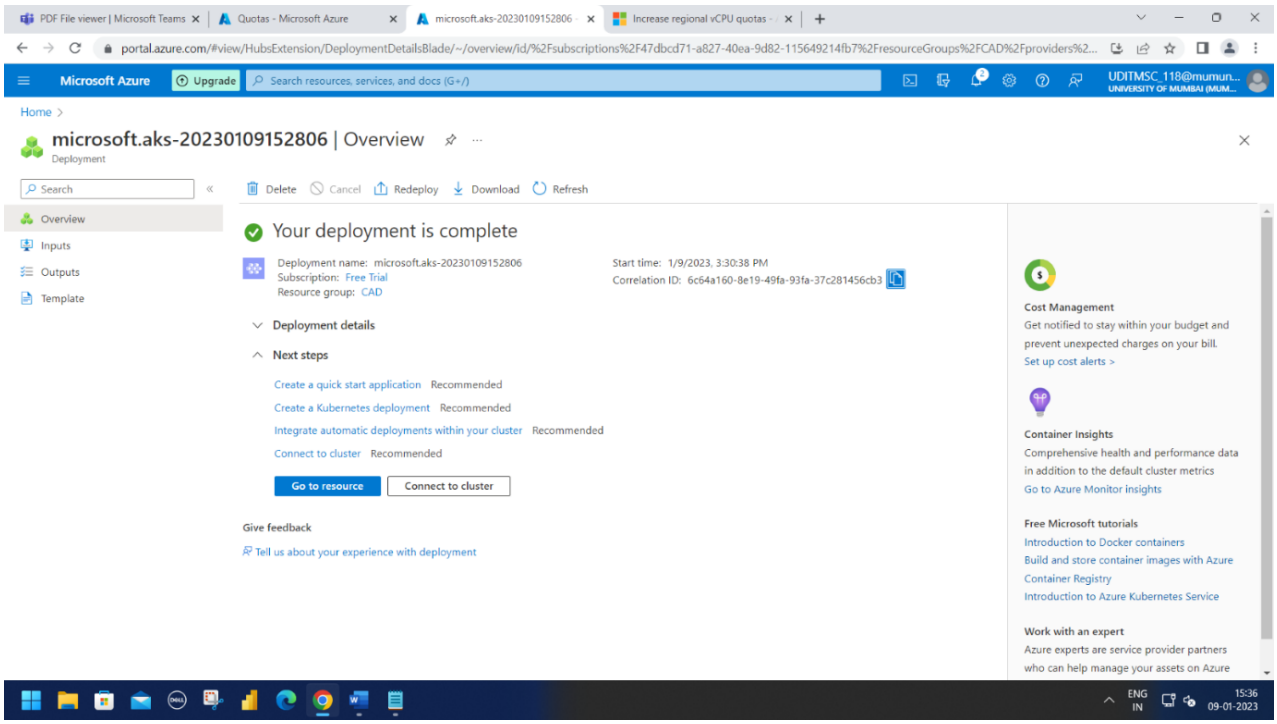

PDF File viewer | Microsoft Teams
Create Kubernetes cluster - Micr
portal.azure.com/#create/microsoft.aks
Microsoft Azure
Upgrade
Search resources, services, and docs (G+/)
UDITMSC_118@mumun...
UNIVERSITY OF MUMBAI (MUM...
Home > Create a resource > Marketplace > Azure Kubernetes Service (AKS) >
Create Kubernetes cluster
API server availability
99.95%
Optimize for availability.
99.5%
Optimize for cost.
99.95% API server availability is recommended for standard configuration.
Automatic upgrade
Enabled with patch (recommended)
Primary node pool
The number and size of nodes in the primary node pool in your cluster. For production workloads, at least 3 nodes are recommended for resiliency. For development or test workloads, only one node is required. If you would like to add additional node pools or to see additional configuration options for this node pool, go to the 'Node pools' tab above. You will be able to add additional node pools after creating your cluster. Learn more about node pools in Azure Kubernetes Service
Node size *
Standard DS2 v2
Standard DS2_v2 is recommended for standard configuration.
Change size
Scale method *
Manual
Autoscale
Autoscaling is recommended for standard configuration.
Node count *
3
Review + create
< Previous
Next : Node pools >
Give feedback
ENG
IN
15:12
09-01-2023



Step 5: Click on Review + Create Button



Step 6: Click on Create Button

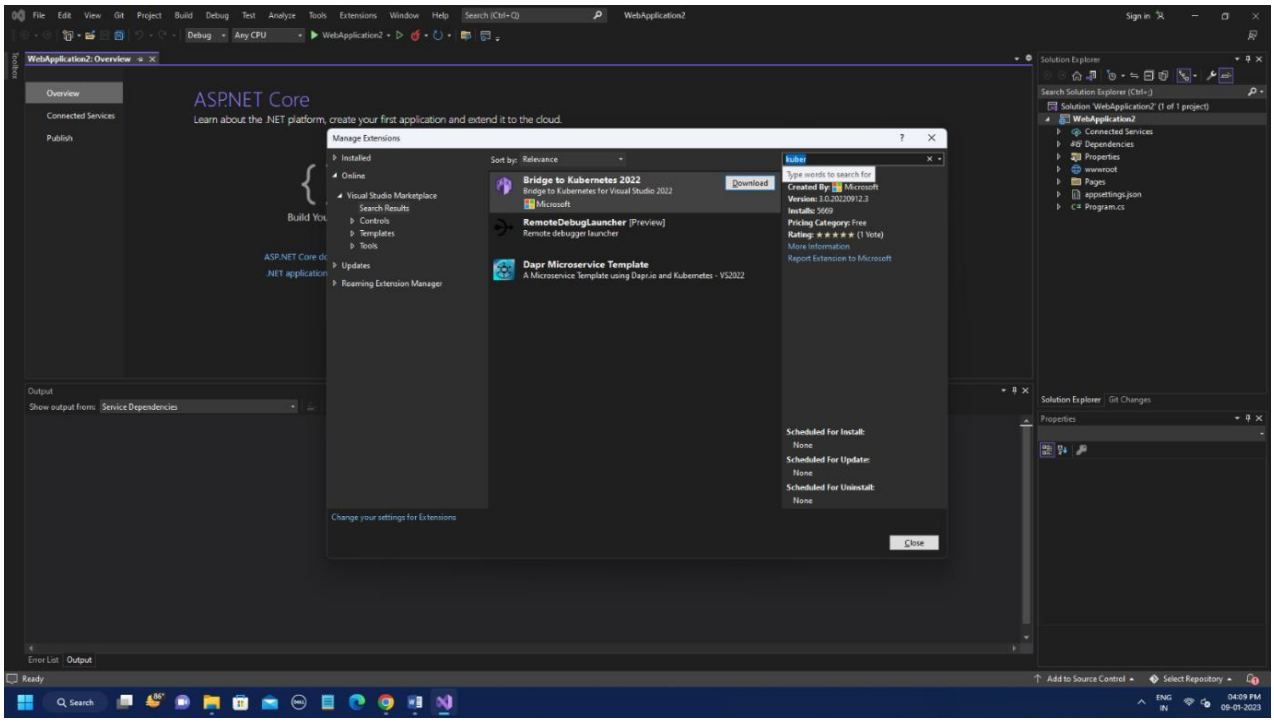


Aim: 2B. Configure Visual Studio to work with an azure kubernetes services Cluster

Source Code:

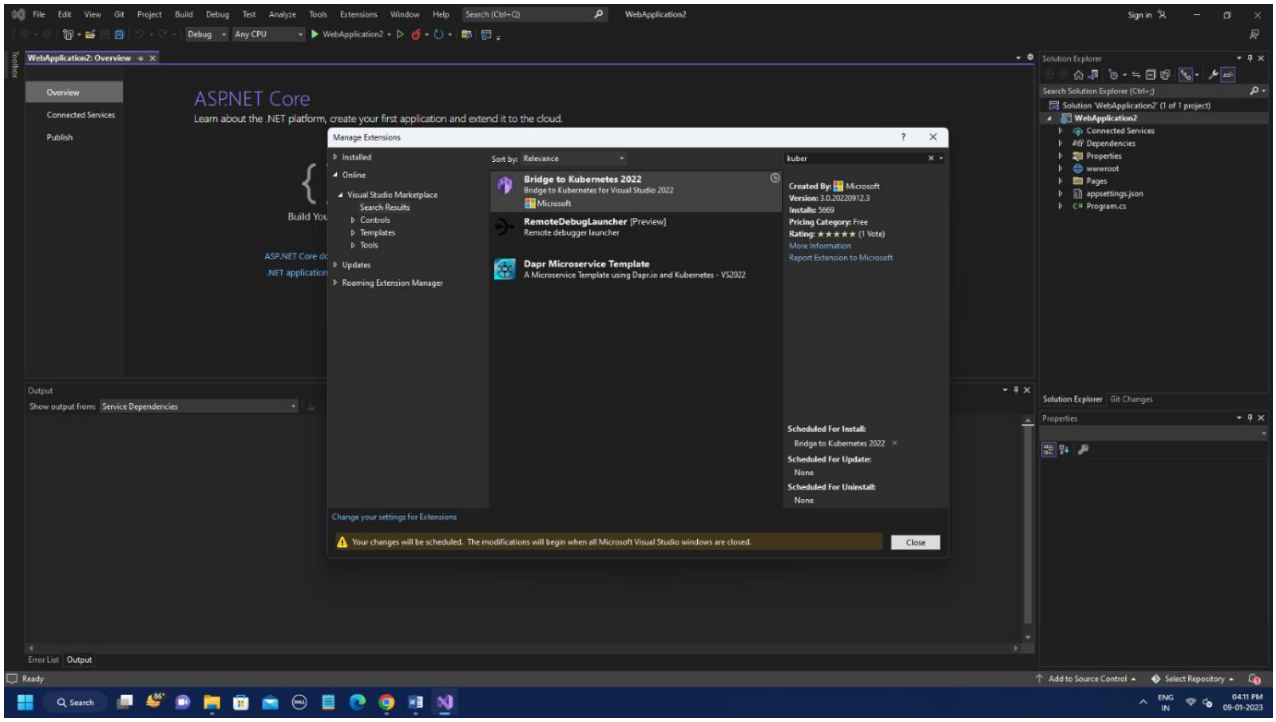
Step 1: Install Visual Studio Enterprise 2022

Step 2: Open Visual Studio 2022 >create a Project >Extensions>Manage Extensions



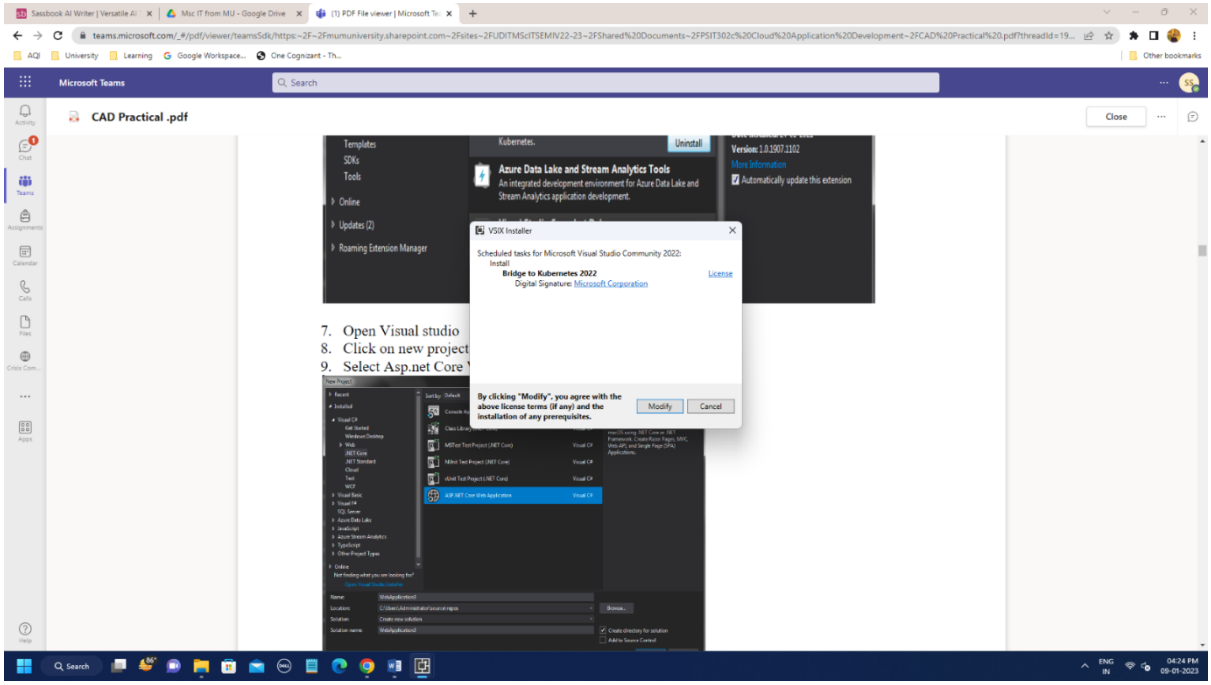
Step 3: Search for Bridge to Kubernetes 2022

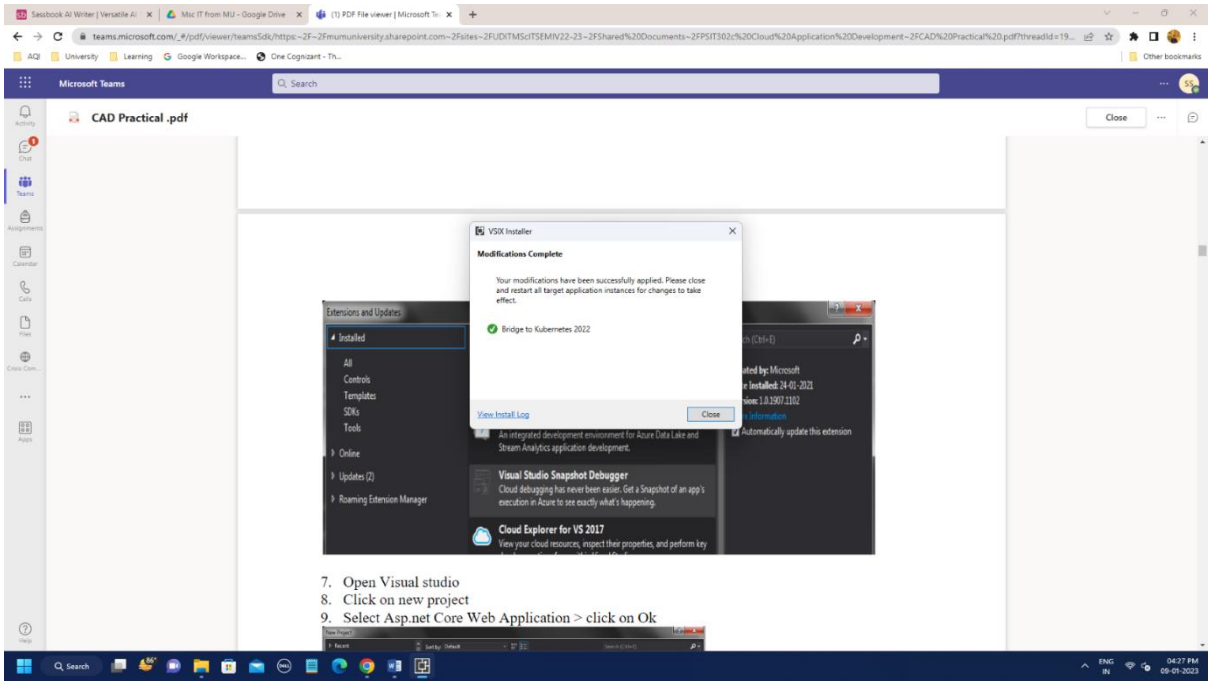
Step 4: Click on Download



Step 5: Close visual studio

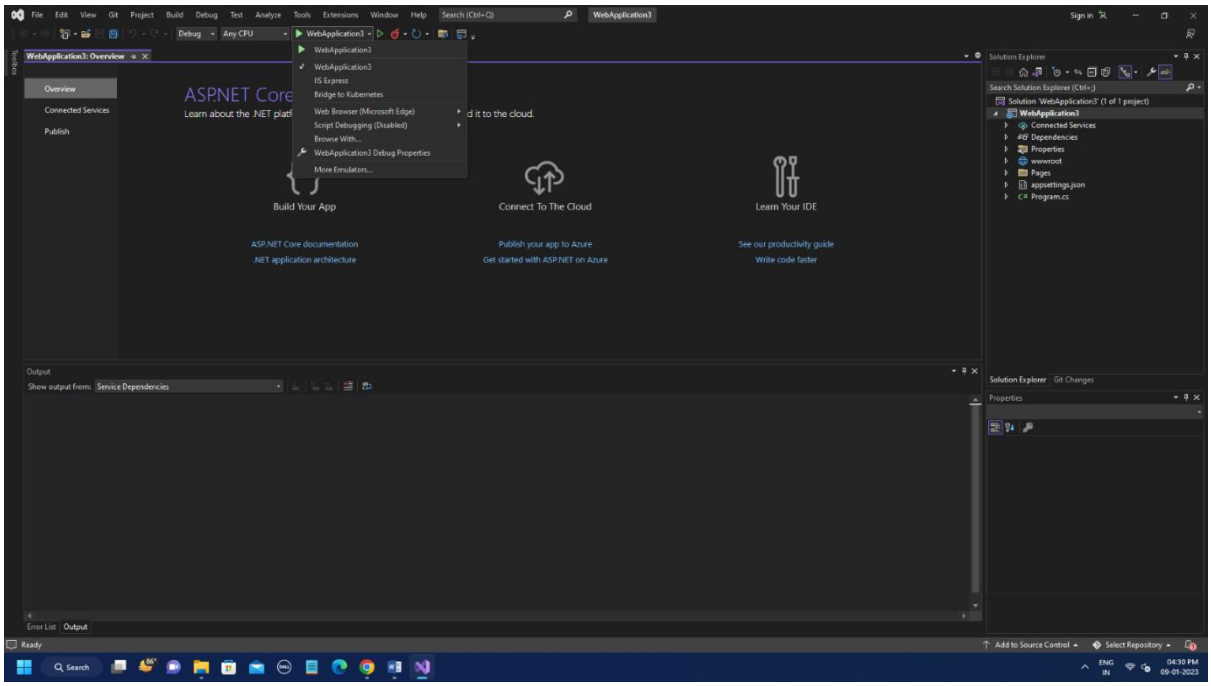
Step 6: click on Modify (install kubernetes tool extension)





Step 7: Open Visual studio

- Click on new project
- Select Asp.net Core Web Application > click on Ok
- Select Model View Control > Click on OK
- Click On Down Arrow of IIS Express
- We can see option of Bridge to Kubernetes



Aim: 2C. Configure Visual Studio Code to work with an azure kubernetes services Cluster

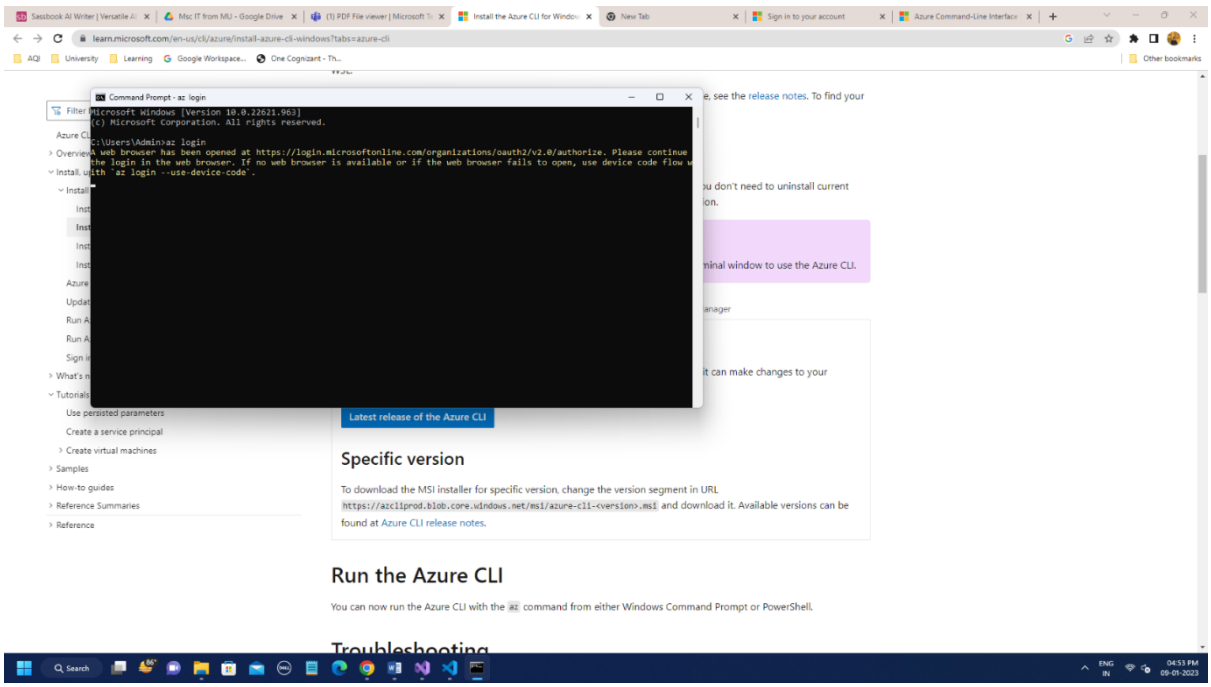
Source Code:

Step 1: Install visual studio code

Step 2: Open extension window

Step 3: Search for Bridge to kubernetes 2022

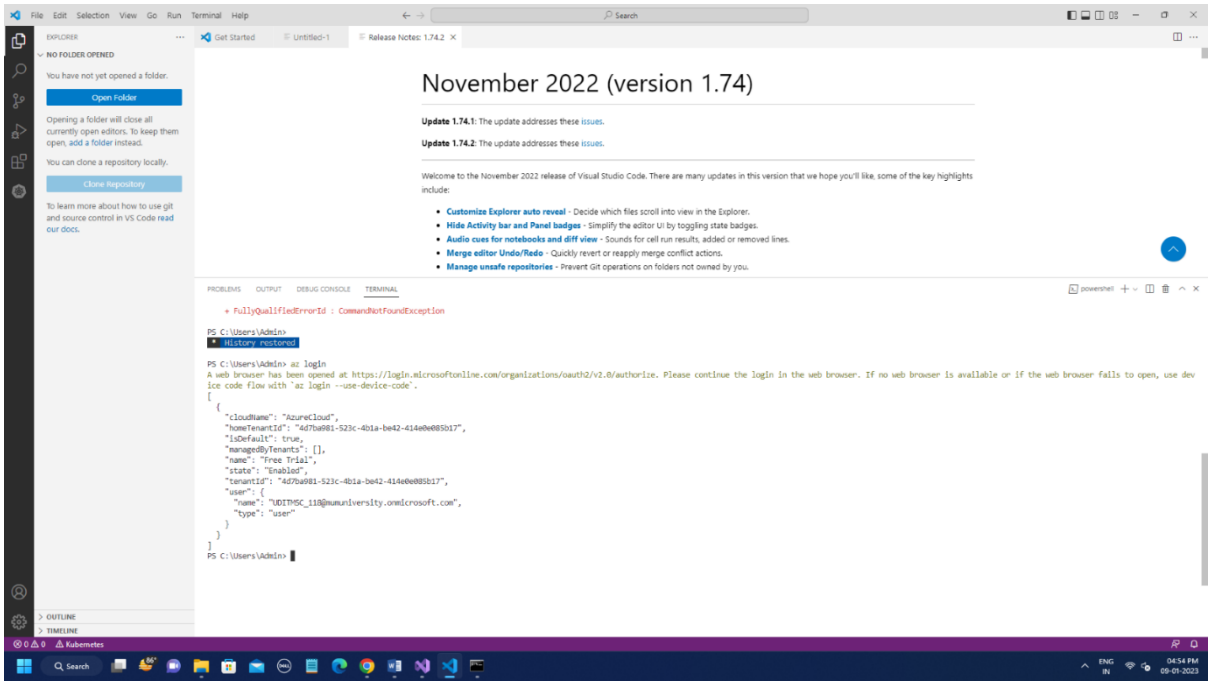
- Click Install
- Search for Azure CLI tools and install
- Go to https://learn.microsoft.com/en-us/cli/azure/install-azure-cli-windows?tabs=azure-cli
- Click on Latest release of the Azure CLI, download the msi and run
- Output on command prompt



Step 4: Close the Visual Studio code and restart application.

Type the following command on the terminal. Output on Visual Studio Code

1.


November 2022 (version 1.74)
Update 1.74.1: The update addresses these issues.
Update 1.74.2: The update addresses these issues.
Welcome to the November 2022 release of Visual Studio Code. There are many updates in this version that we hope you'll like, some of the key highlights include:
Customize Explorer auto reveal - Decide which files scroll into view in the Explorer.
Hide Activity bar and Panel badges - Simplify the editor UI by toggling state badges.
Audio cues for notebooks and diff view - Sounds for cell run results, added or removed lines.
Merge editor Undo/Redo - Quickly revert or reapply merge conflict actions.
Manage unsafe repositories - Prevent Git operations on folders not owned by you.
PS C:\Users\Admin> az login
A web browser has been opened at https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize. Please continue the login in the web browser. If no web browser is available or if the web browser fails to open, use device code flow with `az login --use-device-code`.
"cloudName": "AzureCloud",
"homeTenantId": "4d7ba981-523c-4b1a-be42-414e0e085b17",
"isDefault": true,
"managedByTenants": [],
"name": "Free Trial",
"state": "Enabled",
"tenantId": "4d7ba981-523c-4b1a-be42-414e0e085b17",
"user": {
"name": "UDITMSC_118@nmuniversity.onmicrosoft.com",
"type": "user"
PS C:\Users\Admin>

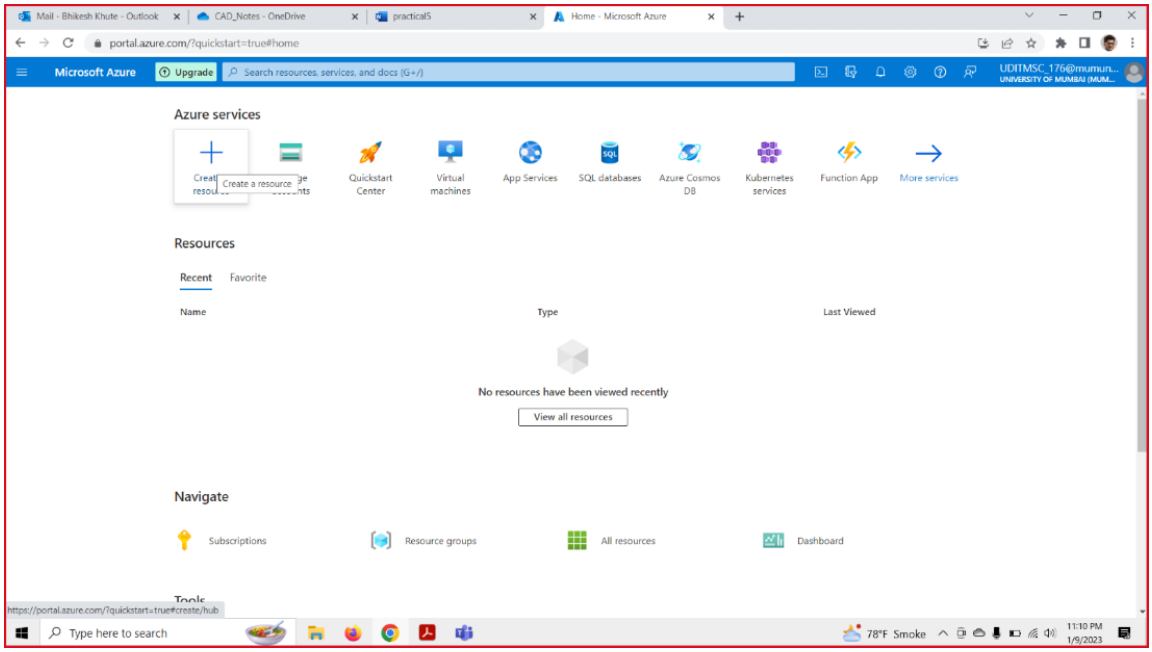
Practical No: 3

Aim: 3a. Create an AKS cluster - From the portal

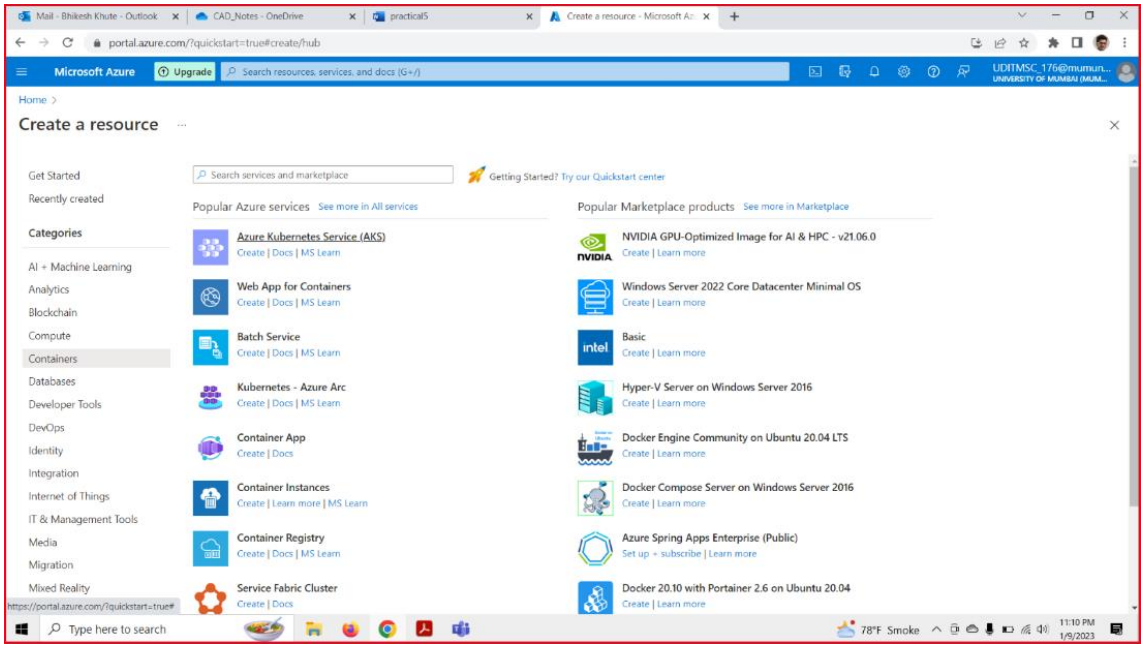
Source Code:

Step 1: Sign in to the Azure portal at https://portal.azure.com.

Step 2: On the Azure portal menu or from the home page, select Create a resource.



Step 3: Select Containers > Kubernetes Service.

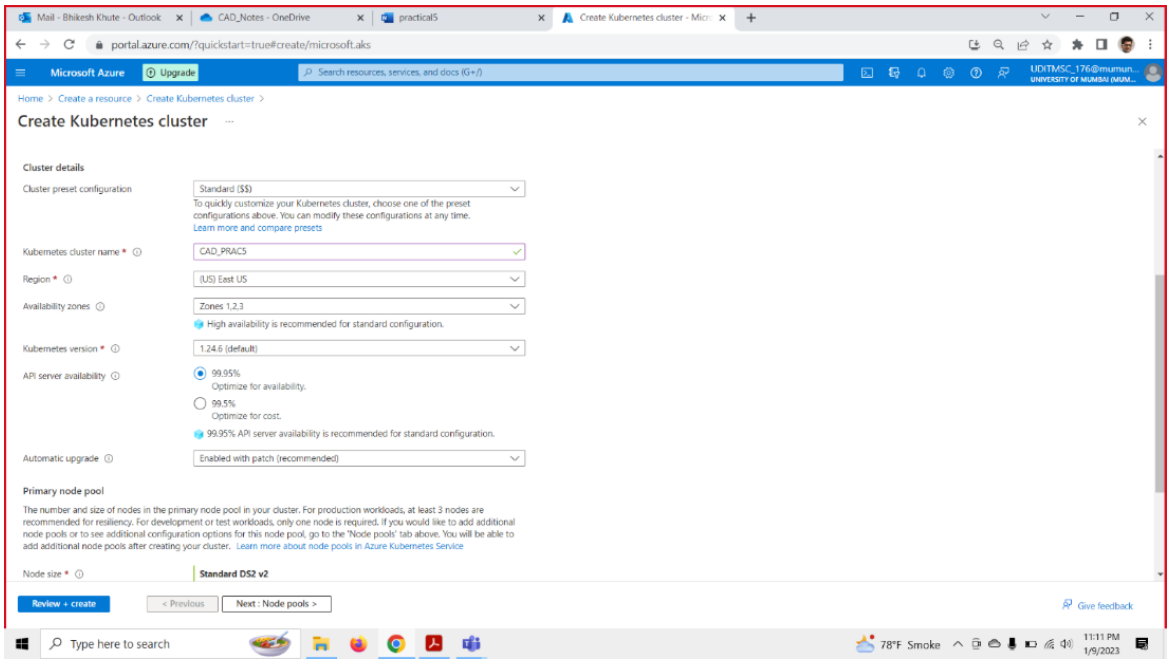


Step 4: On the Basics page, configure the following options:

• **Project details**: Select an Azure Subscription, then select or create an Azure Resource group, such as **CAD_PRACS**.

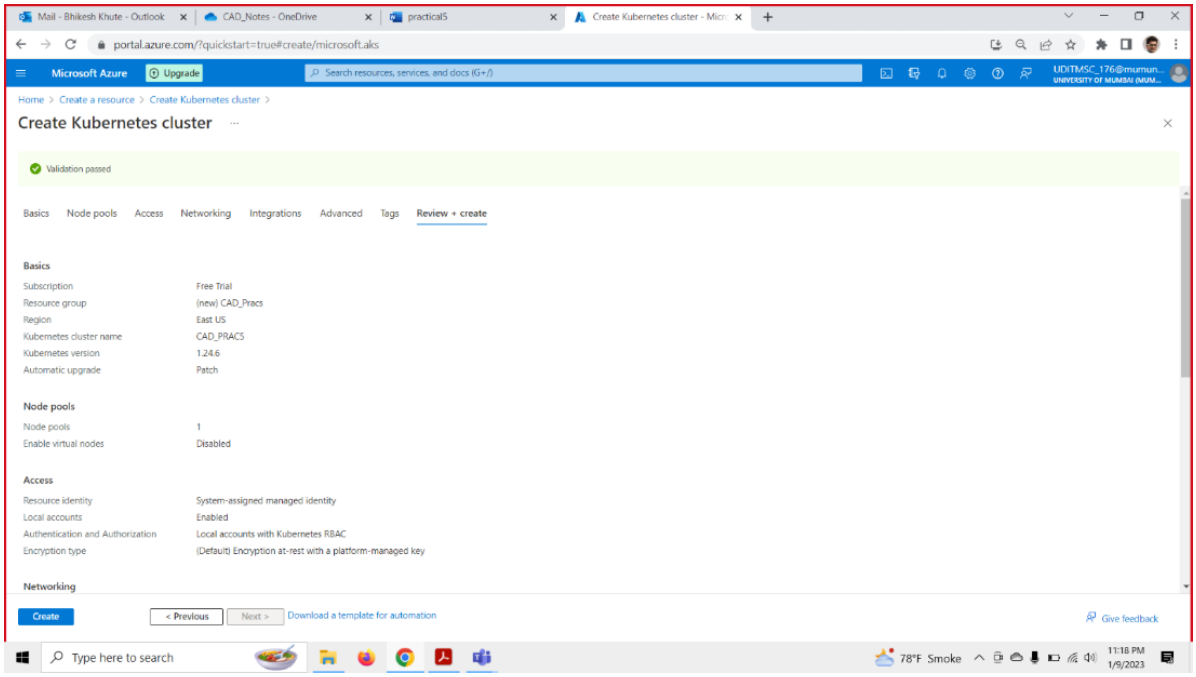
• **Cluster details:** Enter a Kubernetes cluster name, such as **CAD_PRAC5**. Select a Region and Kubernetes version for the AKS cluster.

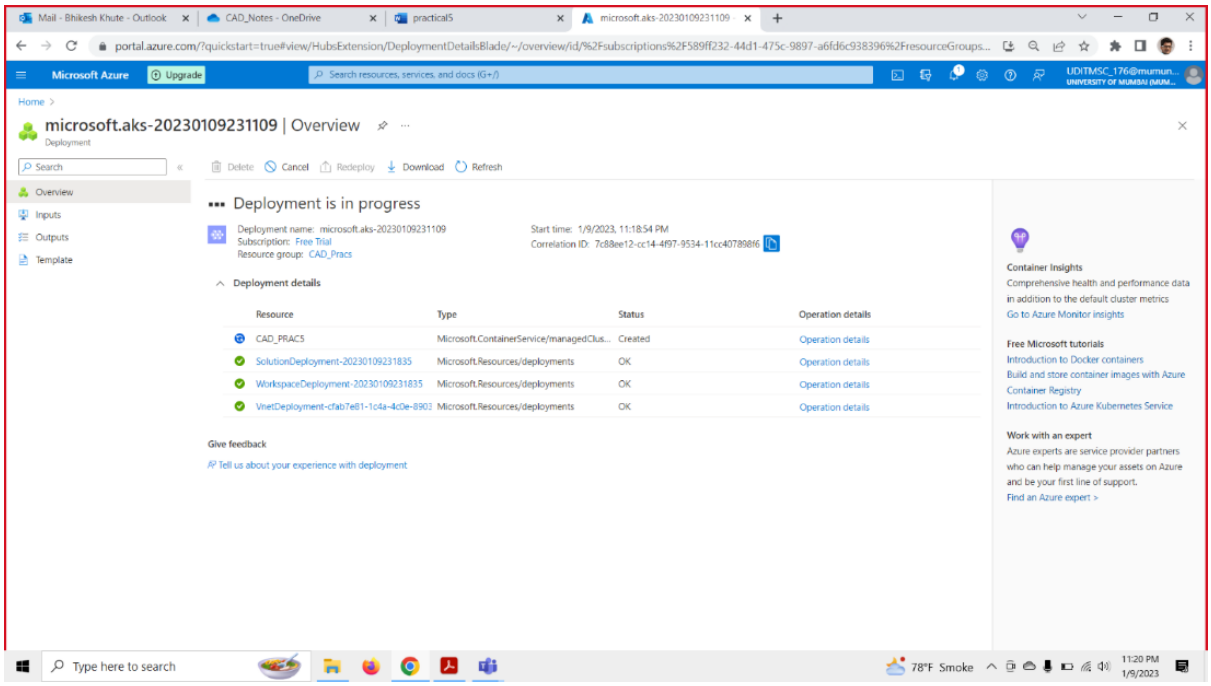
• **Primary node pool:** Select a VM Node size for the AKS nodes. The VM size can't be changed once an AKS cluster has been deployed. - Select the number of nodes to deploy into the cluster. Set Node count to 1. Node count can be adjusted after the cluster has been deployed.

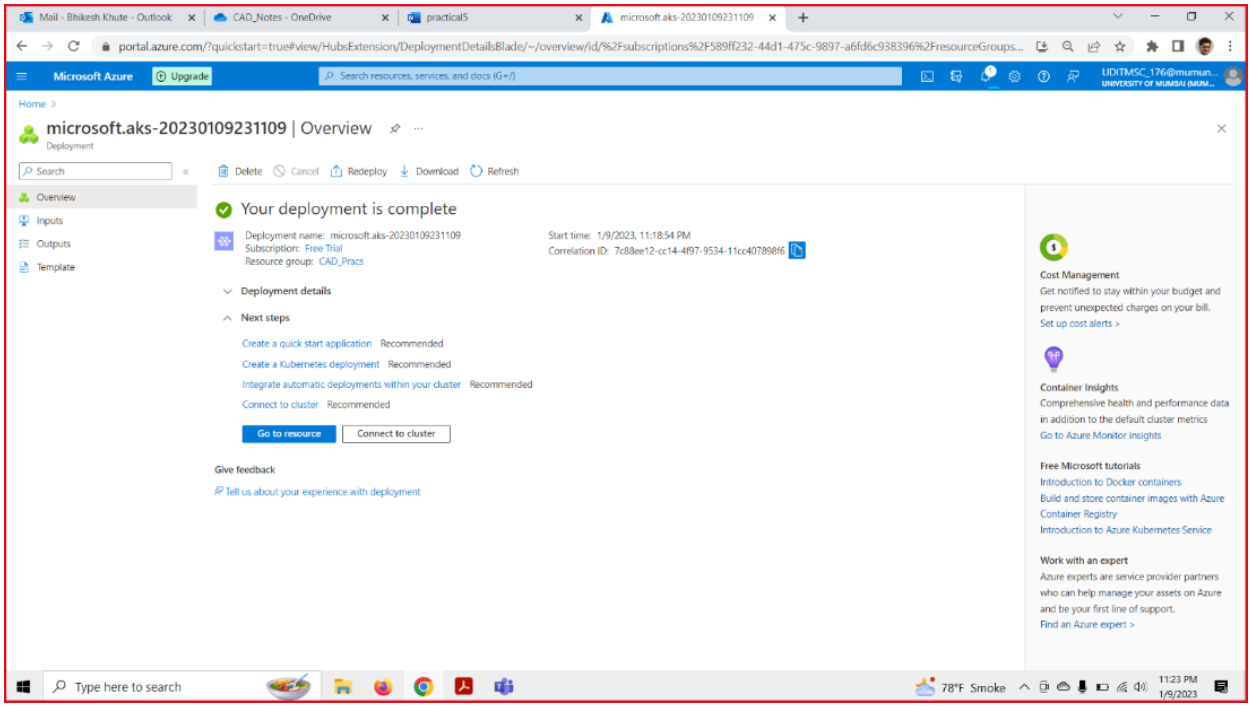


Step 5: On the Node pools page, **keep the default options**. At the bottom of the screen, click Next: Authentication & keep local access or keep it default what is selected. We can change it later.

Step 6: Click Review + create and then Create when validation completes.


Create Kubernetes cluster
Validation passed
Basics Node pools Access Networking Integrations Advanced Tags Review + create
Basics
Subscription Free Trial
Resource group (new) CAD_Pracs
Region East US
Kubernetes cluster name CAD_PRAC5
Kubernetes version 1.24.6
Automatic upgrade Patch
Node pools
Node pools 1
Enable virtual nodes Disabled
Access
Resource identity System-assigned managed identity
Local accounts Enabled
Authentication and Authorization Local accounts with Kubernetes RBAC
Encryption type (Default) Encryption at-rest with a platform-managed key
Networking
Create
< Previous
Next >
Download a template for automation
Give feedback


microsoft.aks-20230109231109 | Overview
Deployment
Delete Cancel Redeploy Download Refresh
Overview
Inputs
Outputs
Template
Deployment is in progress
Deployment name: microsoft.aks-20230109231109
Subscription: Free Trial
Resource group: CAD_Pracs
Start time: 1/9/2023, 11:18:54 PM
Correlation ID: 7c88ee12-cc14-4f97-9534-11cc4078986f
Deployment details
Resource Type Status Operation details
CAD_PRAC5 Microsoft.ContainerService/managedClus... Created Operation details
SolutionDeployment-20230109231835 Microsoft.Resources/deployments OK Operation details
WorkspaceDeployment-20230109231835 Microsoft.Resources/deployments OK Operation details
VnetDeployment-cfab7e81-1c4a-4c0e-8903 Microsoft.Resources/deployments OK Operation details
Give feedback
Tell us about your experience with deployment
Container Insights
Comprehensive health and performance data in addition to the default cluster metrics
Go to Azure Monitor insights
Free Microsoft tutorials
Introduction to Docker containers
Build and store container images with Azure Container Registry
Introduction to Azure Kubernetes Service
Work with an expert
Azure experts are service provider partners who can help manage your assets on Azure and be your first line of support.
Find an Azure expert >

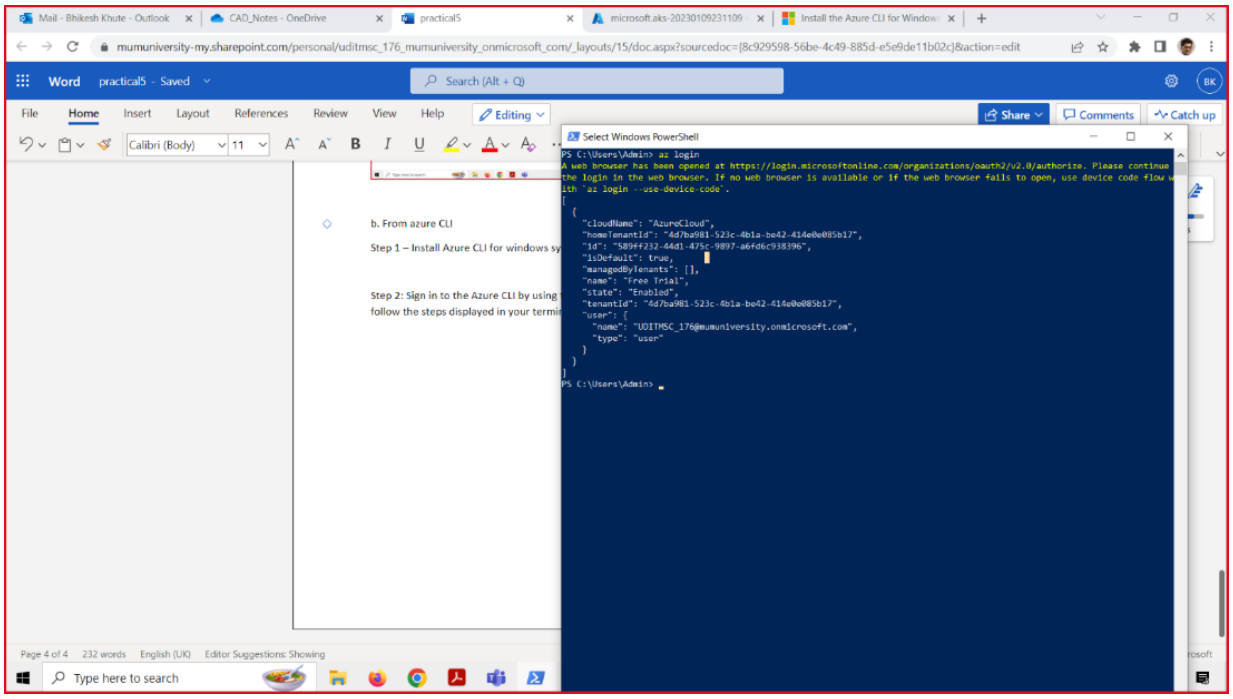

Microsoft Azure
Upgrade
Search resources, services, and docs (G+/)
Home
microsoft.aks-20230109231109 | Overview
Deployment
Search
Overview
Inputs
Outputs
Template
Delete
Cancel
Redeploy
Download
Refresh
Your deployment is complete
Deployment name: microsoft.aks-20230109231109
Subscription: Free Trial
Resource group: CAD_Pracs
Start time: 1/9/2023, 11:18:54 PM
Correlation ID: 7c88ee12-cc14-4f97-9534-11cc40788816
Deployment details
Next steps
Create a quick start application Recommended
Create a Kubernetes deployment Recommended
Integrate automatic deployments within your cluster Recommended
Connect to cluster Recommended
Go to resource
Connect to cluster
Give feedback
Tell us about your experience with deployment
Cost Management
Get notified to stay within your budget and prevent unexpected charges on your bill.
Set up cost alerts >
Container Insights
Comprehensive health and performance data in addition to the default cluster metrics
Go to Azure Monitor insights
Free Microsoft tutorials
Introduction to Docker containers
Build and store container images with Azure Container Registry
Introduction to Azure Kubernetes Service
Work with an expert
Azure experts are service provider partners who can help manage your assets on Azure and be your first line of support.
Find an Azure expert >
Type here to search
78°F Smoke
11:23 PM
1/9/2023

Aim: 3b. Create an AKS cluster – with Azure CLI

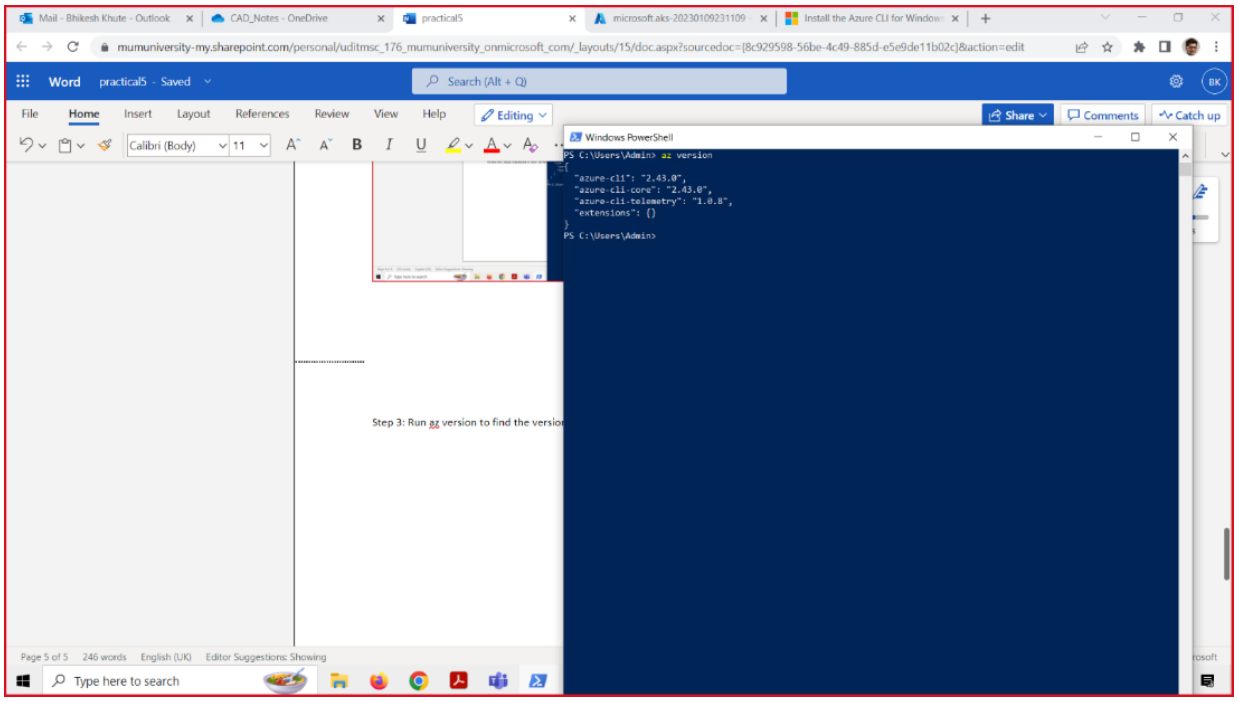
Source Code:

Step 1: Install Azure CLI for windows system.

Step 2: Sign in to the Azure CLI by using the **az login** command. To finish the authentication process, follow the steps displayed in your terminal.



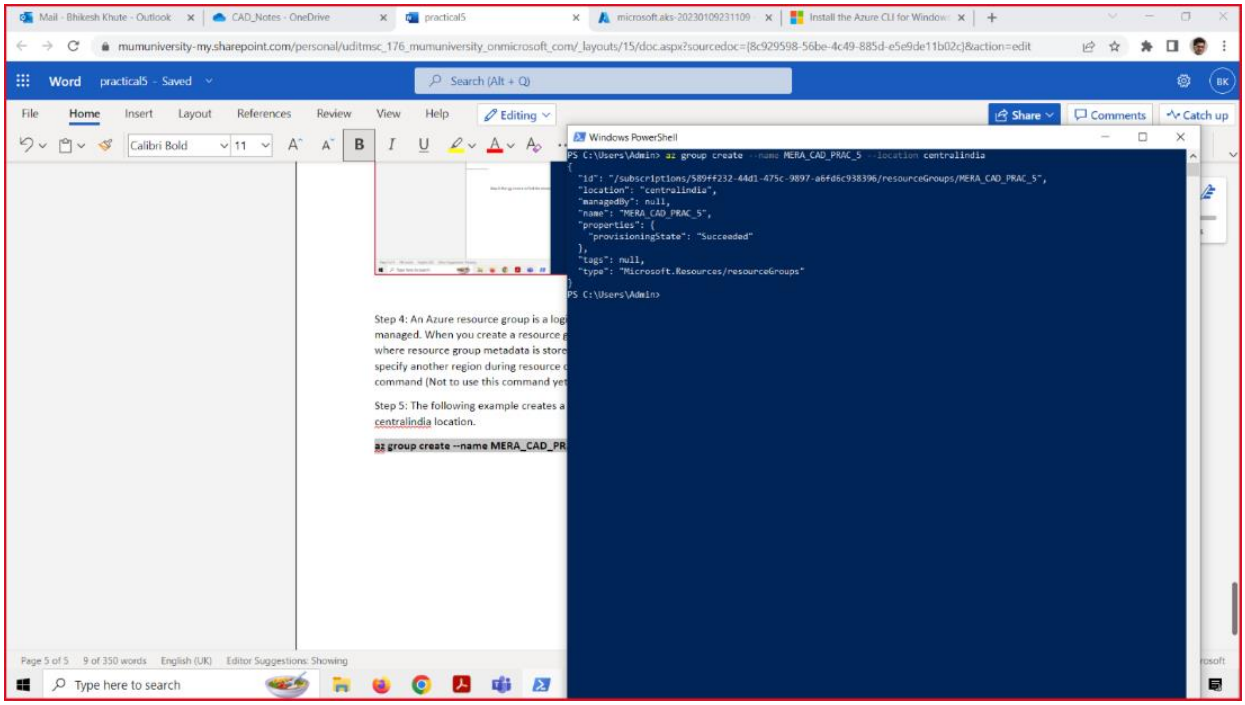
Step 3: Run az version to find the version and dependent libraries that are



Step 4: An Azure resource group is a logical group in which Azure resources are deployed and managed. When you create a resource group, you are asked to specify a location. This location is where resource group metadata is stored, it is also where your resources run in Azure if you don't specify another region during resource creation. Create a resource group using the az group create command (Not to use this command yet. Refer the command in next step)

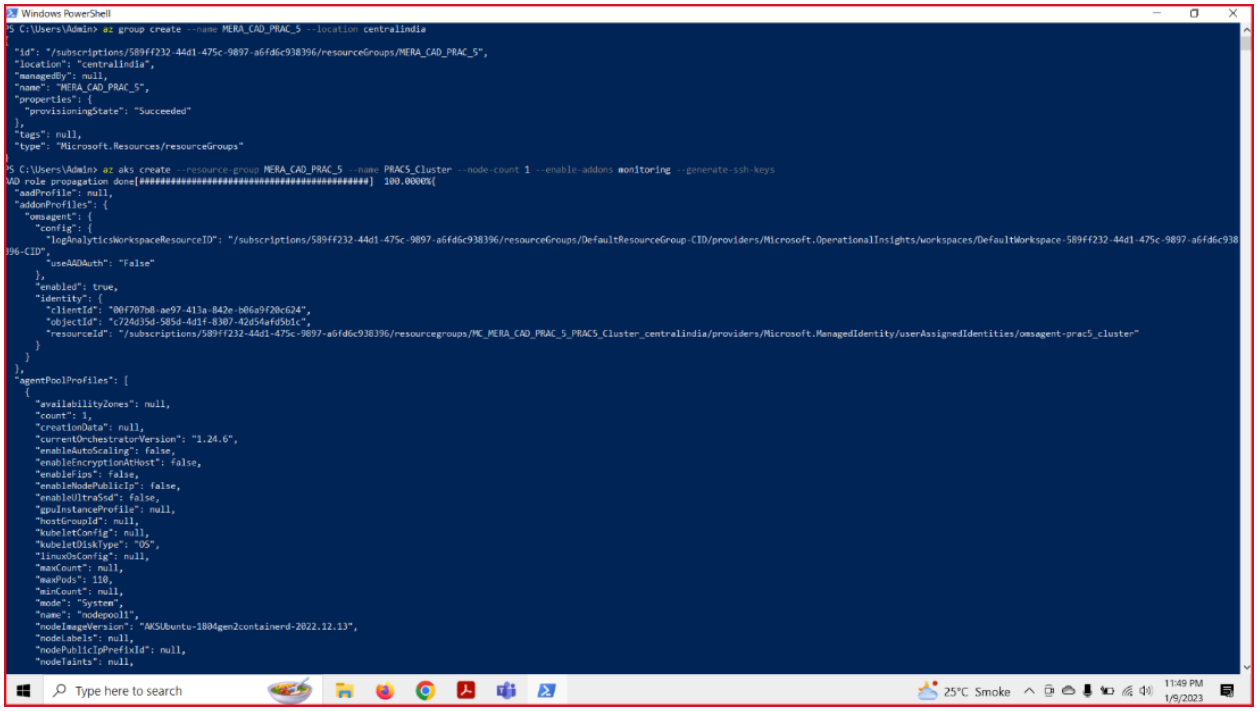
Step 5: The following example creates a resource group named myResourceGroup in the centralindia location.

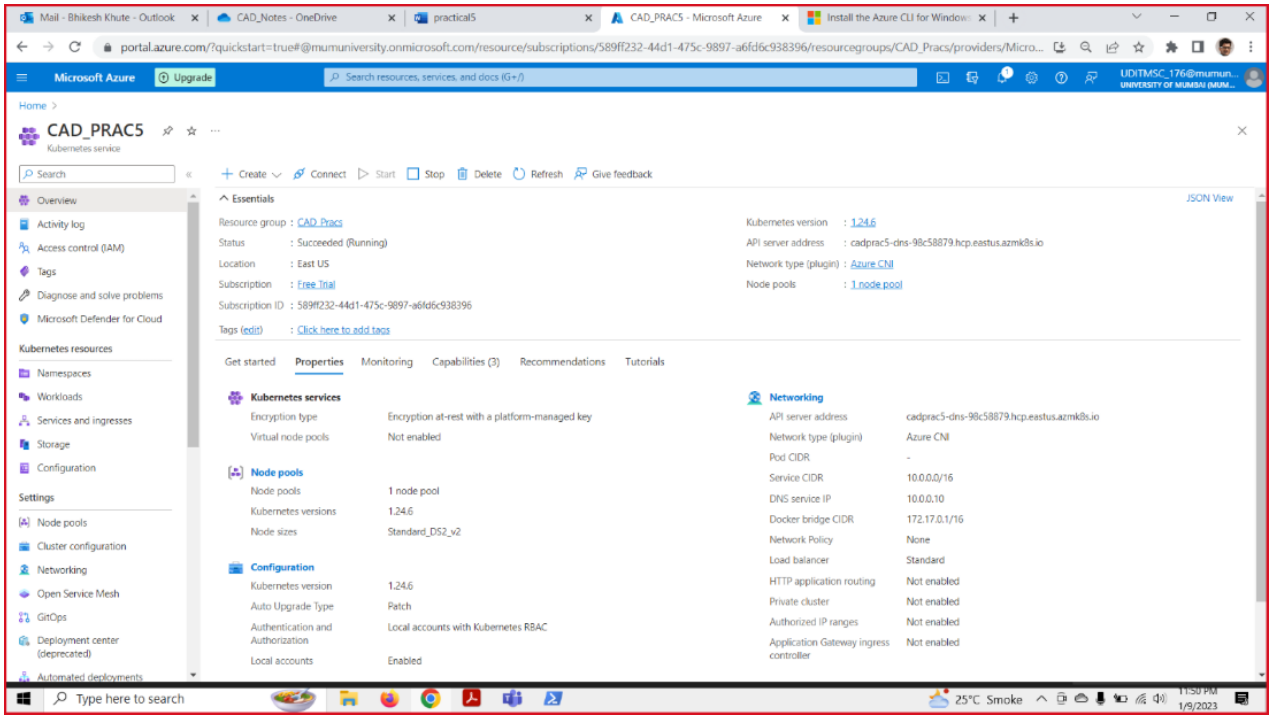
az group create --name MERA_CAD_PRAC_5 --location centralindia



Step 6: Use following command to create an AKS cluster. The following example creates a cluster with one node. This will take several minutes to complete.

az aks create --resource-group MERA_CAD_PRAC_5 --name PRAC5_Cluster --node-count 1 --enable-addons monitoring --generate-ssh-keys


Windows PowerShell
PS C:\Users\Admin> az group create --name MERA_CAD_PRAC_5 --location centralindia
"id": "/subscriptions/589ff232-44d1-475c-9897-a6fd6c938396/resourceGroups/MERA_CAD_PRAC_5",
"location": "centralindia",
"managedBy": null,
"name": "MERA_CAD_PRAC_5",
"properties": {
"provisioningState": "Succeeded"
},
"tags": null,
"type": "Microsoft.Resources/resourceGroups"
PS C:\Users\Admin> az aks create --resource-group MERA_CAD_PRAC_5 --name PRAC5_Cluster --node-count 1 --enable-addons monitoring --generate-ssh-keys
"addonProfiles": {
"omsagent": {
"config": {
"useAADAuth": "False"
},
"enabled": true,
"identity": {
"clientId": "00f707b8-ae97-413a-842e-b06a9f20c624",
"objectId": "c724d35d-585d-4d1f-8307-42d54afd5b1c",
"resourceId": "/subscriptions/589ff232-44d1-475c-9897-a6fd6c938396/resourcegroups/MC_MERA_CAD_PRAC_5_PRAC5_Cluster_centralindia/providers/Microsoft.ManagedIdentity/userAssignedIdentities/omsagent-prac5_cluster"
}
}
},
"agentPoolProfiles": [
{
"availabilityZones": null,
"count": 1,
"creationData": null,
"currentOrchestratorVersion": "1.24.6",
"enableAutoScaling": false,
"enableEncryptionAtHost": false,
"enableFips": false,
"enableNodePublicIp": false,
"enableUltraSsd": false,
"gpuInstanceProfile": null,
"hostGroupId": null,
"kubeletConfig": null,
"kubeletDiskType": "OS",
"linuxOsConfig": null,
"maxCount": null,
"maxPods": 110,
"minCount": null,
"mode": "System",
"name": "nodepool1",
"nodeImageVersion": "AKSUbuntu-1804gen2containerd-2022.12.13",
"nodeLabels": null,
"nodePublicIpPrefixId": null,
"nodeTaints": null,
Type here to search
25°C Smoke
11:49 PM
1/9/2023


Mail - Bhikesh Khute - Outlook
CAD_Notes - OneDrive
practical5
CAD_PRAC5 - Microsoft Azure
Install the Azure CLI for Windows
portal.azure.com/?quickstart=true#@mumuniversity.onmicrosoft.com/resource/subscriptions/589ff232-44d1-475c-9897-a6fd6c938396/resourcegroups/CAD_Pracs/providers/Micro...
Microsoft Azure
Upgrade
Search resources, services, and docs (G+/)
Home
CAD_PRAC5
Kubernetes service
Search
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Microsoft Defender for Cloud
Kubernetes resources
Namespaces
Workloads
Services and ingresses
Storage
Configuration
Settings
Node pools
Cluster configuration
Networking
Open Service Mesh
GitOps
Deployment center (deprecated)
Automated deployments
Create
Connect
Start
Stop
Delete
Refresh
Give feedback
Essentials
JSON View
Resource group : CAD Pracs
Status : Succeeded (Running)
Location : East US
Subscription : Free Trial
Subscription ID : 589ff232-44d1-475c-9897-a6fd6c938396
Tags (edit) : Click here to add tags
Kubernetes version : 1.24.6
API server address : cadprac5-dns-98c58879.hcp.eastus.azmk8s.io
Network type (plugin) : Azure CNI
Node pools : 1 node pool
Get started
Properties
Monitoring
Capabilities (3)
Recommendations
Tutorials
Kubernetes services
Encryption type : Encryption at-rest with a platform-managed key
Virtual node pools : Not enabled
Node pools
Node pools : 1 node pool
Kubernetes versions : 1.24.6
Node sizes : Standard_DS2_v2
Configuration
Kubernetes version : 1.24.6
Auto Upgrade Type : Patch
Authentication and Authorization : Local accounts with Kubernetes RBAC
Local accounts : Enabled
Networking
API server address : cadprac5-dns-98c58879.hcp.eastus.azmk8s.io
Network type (plugin) : Azure CNI
Pod CIDR : -
Service CIDR : 10.0.0.0/16
DNS service IP : 10.0.0.10
Docker bridge CIDR : 172.17.0.1/16
Network Policy : None
Load balancer : Standard
HTTP application routing : Not enabled
Private cluster : Not enabled
Authorized IP ranges : Not enabled
Application Gateway ingress controller : Not enabled
Type here to search
25°C Smoke
1/9/2023

Practical No: 04
Create an API gateway service

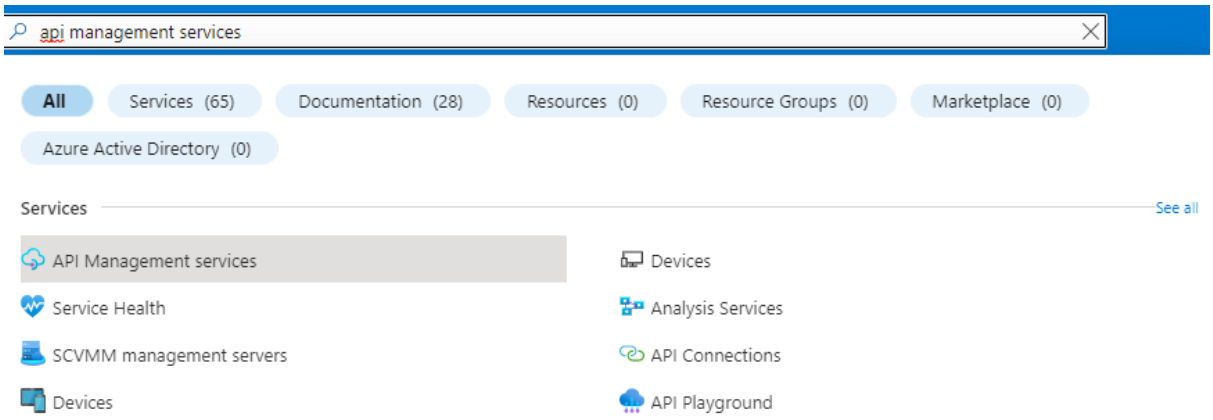
Aim: 4A. Create an API Management Service.

Source Code:

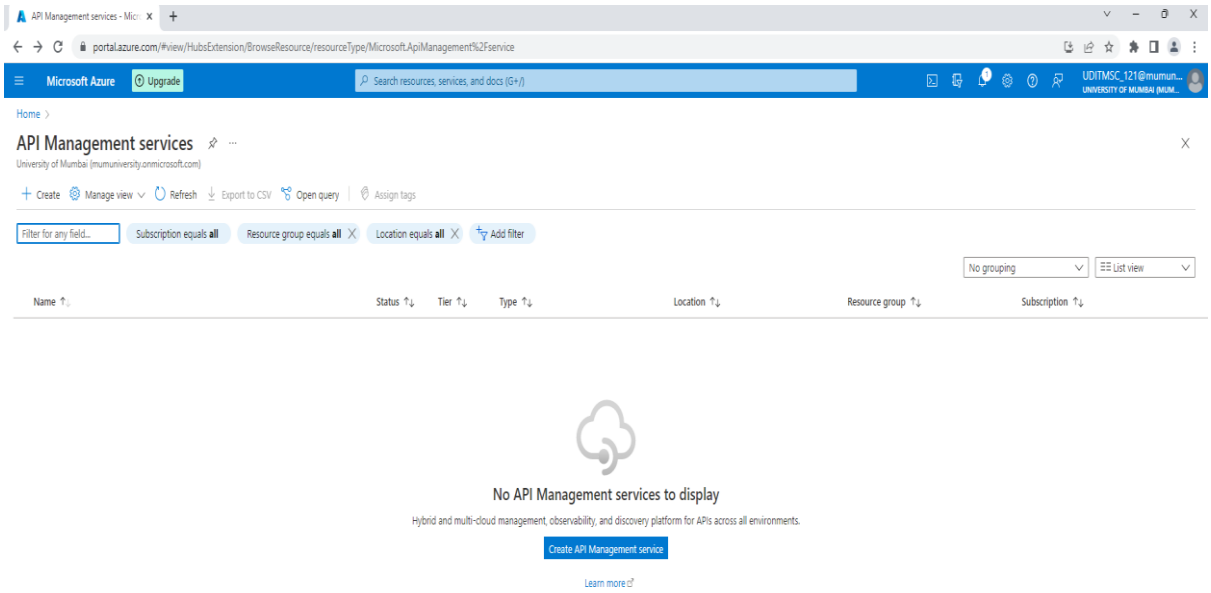
Step 1: Sign-in to your Azure Subscription Portal

Step 2: Search for "API Management", then select API Management in order to create a service instance.

Step 3: Select "API Management service"



Step 4: As API management service page is loaded, select "Create API management service" button.



Step 6: Provide a name. This name sets the URL of the API gateway and portal [Name-AzureManagementService]

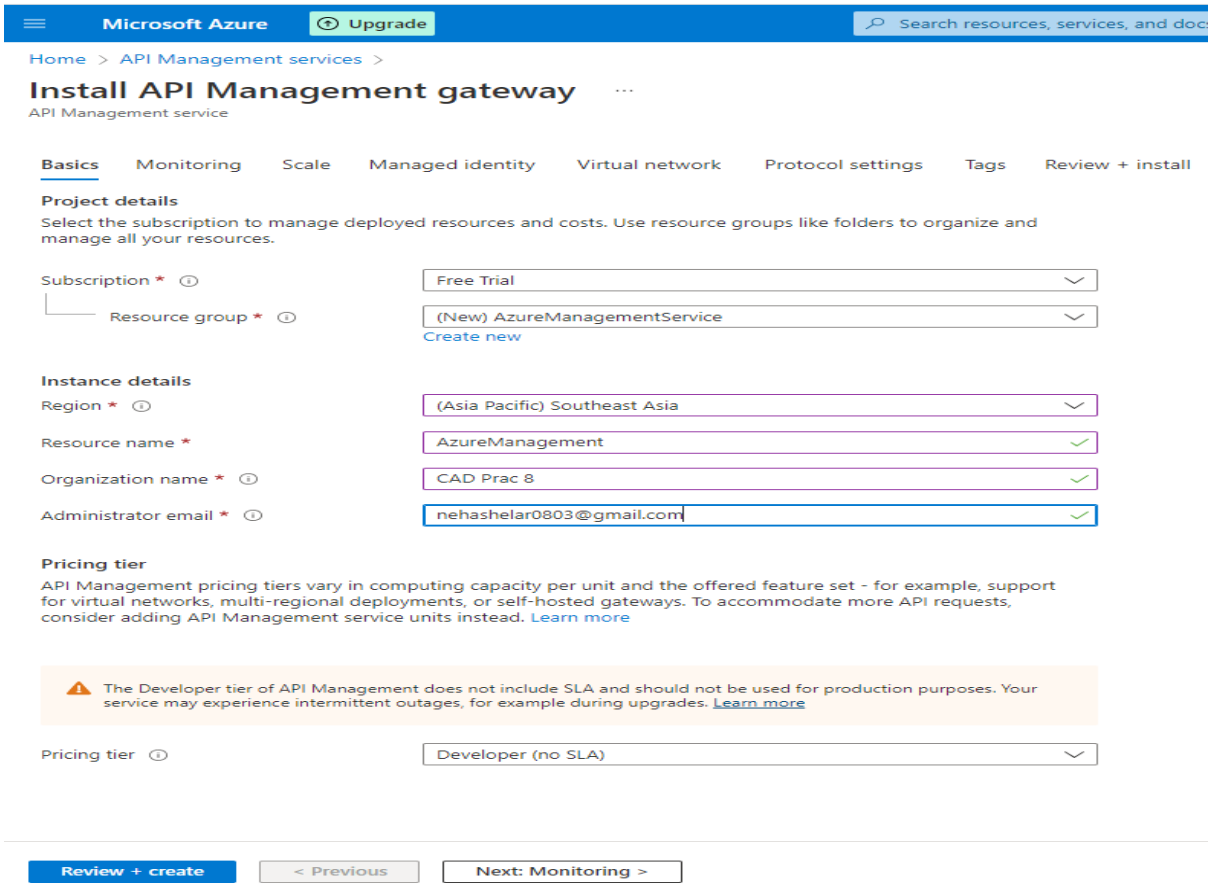
Step 7: Select the subscription and resource group (or create a new resource group), and select the location. [Resource NameAzureManagement, Location- SouthEast Asia]

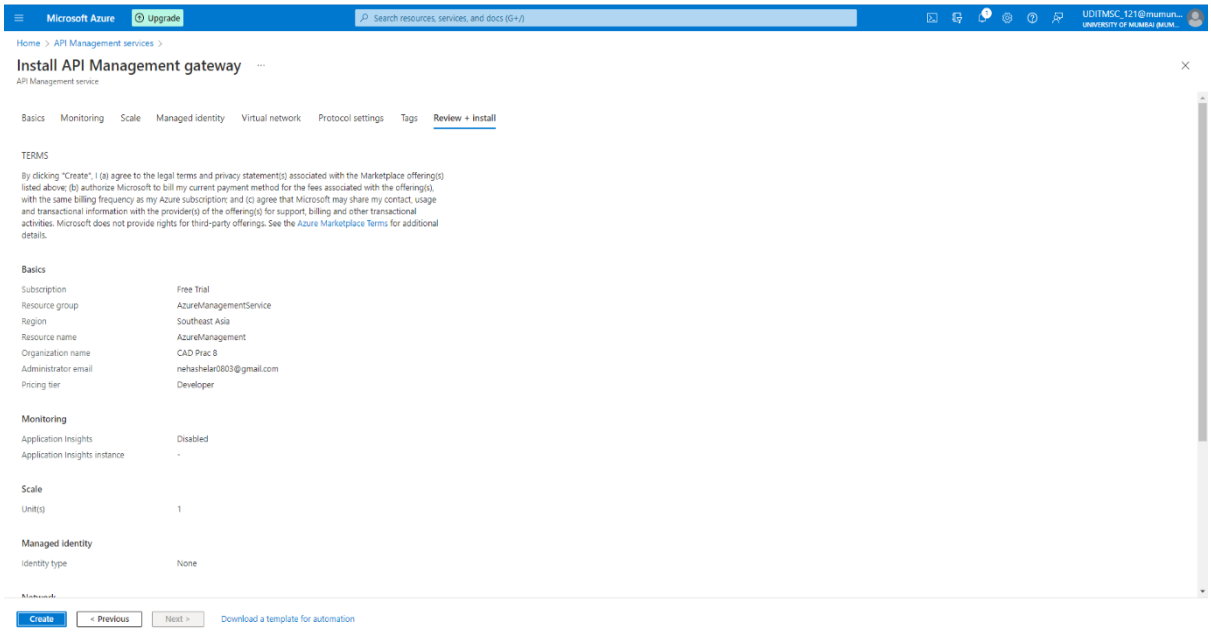
Step 8: Specify the organization name. This name will appear in the developer portal as the organization that publishes the API. [Name-CAD Prac 8]

Step 9: Specify the email address of the administrator. The user who creates the service instance will be the default administrator, so it's best to provide the email address of this user until you want someone else to serve as administrator.

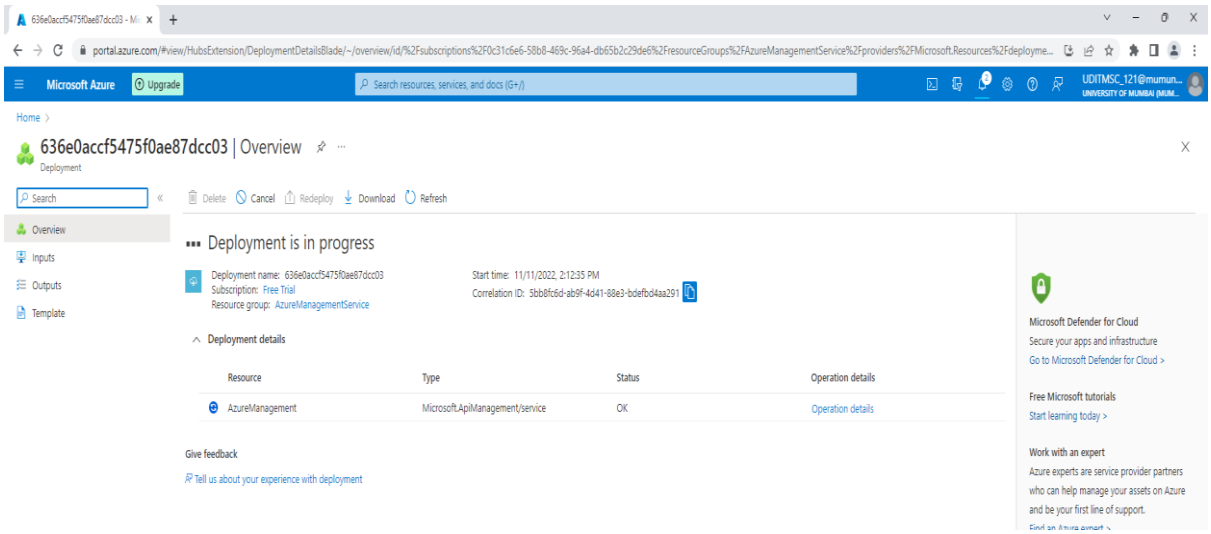
Step 10: Select the pricing tier. The Developer tier is the most comprehensive offering, with sufficient request/response limitations in dev/test scenarios.

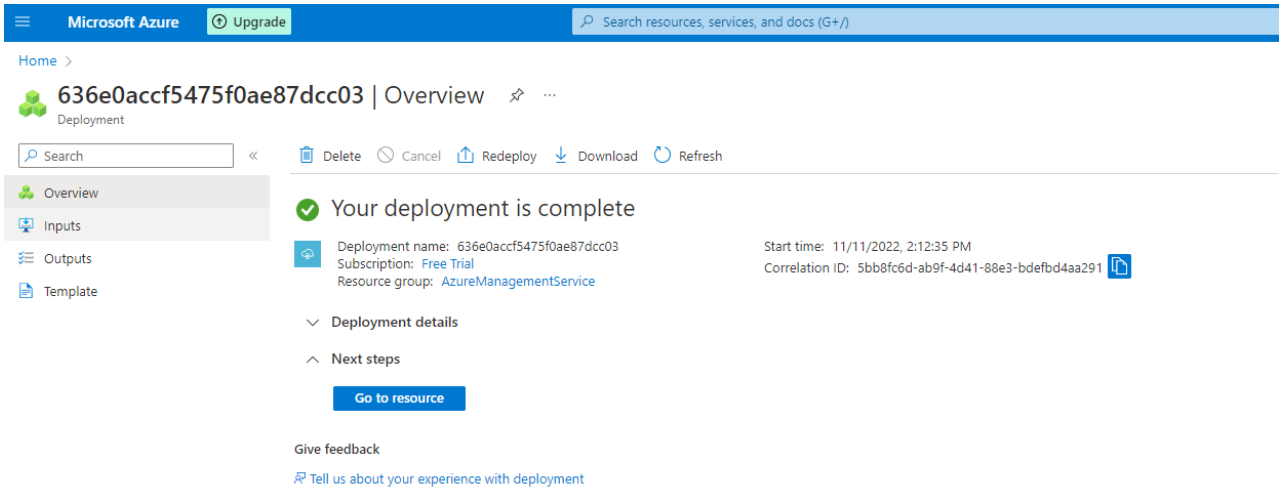
Step 11: After Completing the form click on review + create button to create "Azure API Management service instance".

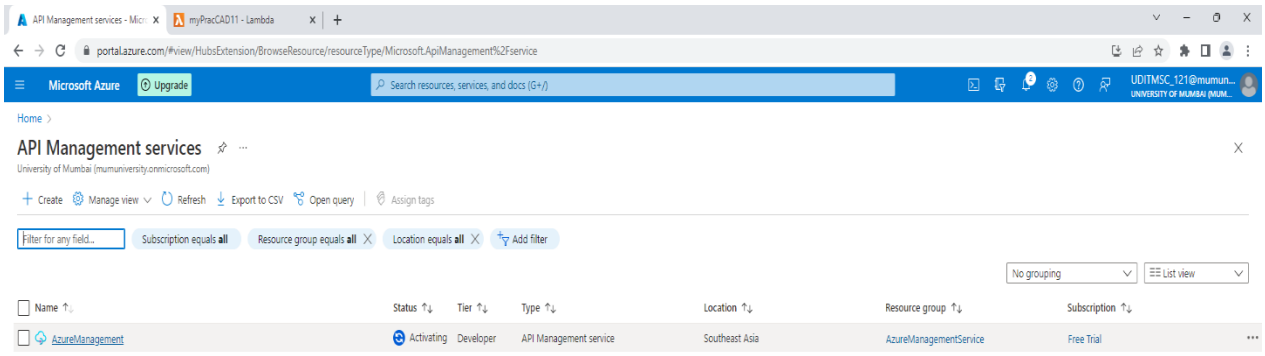


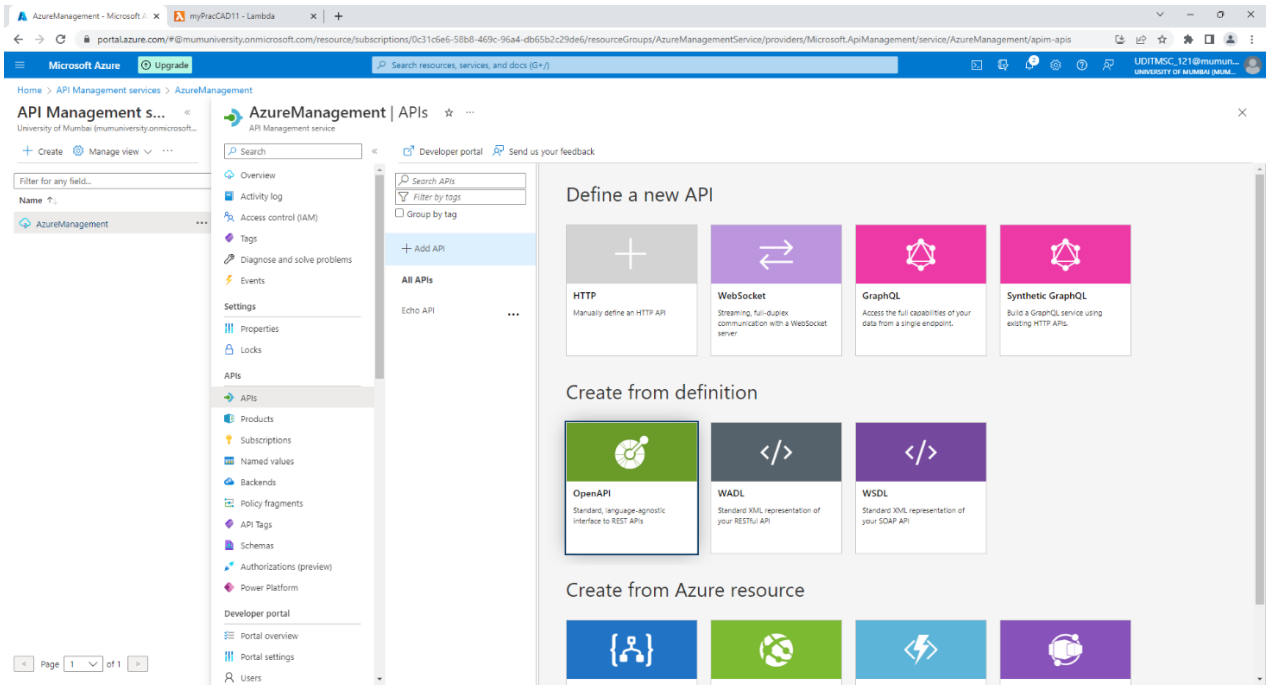


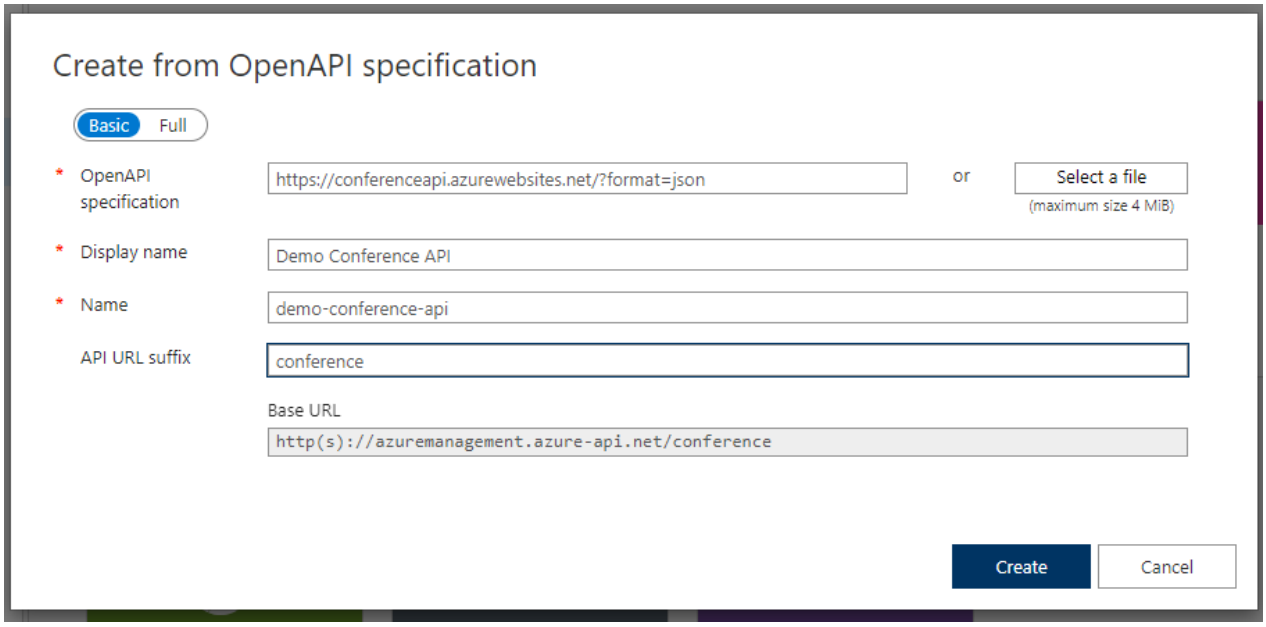
Step 14: After clicking on the create button the API management service instance will go to deployment stage. After the deployment is complete the instance will be shown

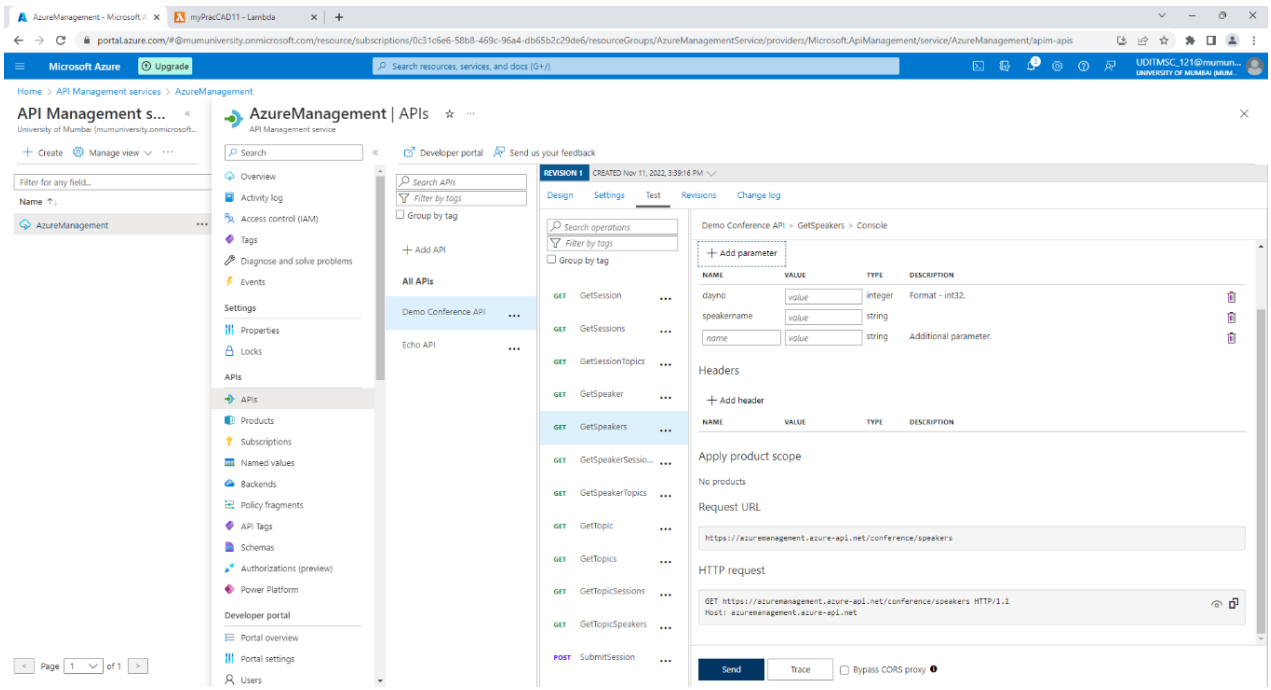



Microsoft Azure
Upgrade
Search resources, services, and docs (G+/)
Home
636e0accf5475f0ae87dcc03 | Overview
Deployment
Search
Overview
Inputs
Outputs
Template
Delete
Cancel
Redeploy
Download
Refresh
Your deployment is complete
Deployment name: 636e0accf5475f0ae87dcc03
Subscription: Free Trial
Resource group: AzureManagementService
Start time: 11/11/2022, 2:12:35 PM
Correlation ID: 5bb8fc6d-ab9f-4d41-88e3-bdefbd4aa291
Deployment details
Next steps
Go to resource
Give feedback
Tell us about your experience with deployment


API Management services
University of Mumbai (mumuniversity.onmicrosoft.com)
Create
Manage view
Refresh
Export to CSV
Open query
Assign tags
Filter for any field...
Subscription equals all
Resource group equals all
Location equals all
Add filter
No grouping
List view
Name
Status
Tier
Type
Location
Resource group
Subscription
AzureManagement
Activating
Developer
API Management service
Southeast Asia
AzureManagementService
Free Trial


AzureManagement | APIs
API Management service
Define a new API
HTTP
WebSocket
GraphQL
Synthetic GraphQL
Create from definition
OpenAPI
WADL
WSDL
Create from Azure resource


Create from OpenAPI specification
Basic
Full
* OpenAPI specification
https://conferenceapi.azurewebsites.net/?format=json
or
Select a file
(maximum size 4 MiB)
* Display name
Demo Conference API
* Name
demo-conference-api
API URL suffix
conference
Base URL
http(s)://azuremanagement.azure-api.net/conference
Create
Cancel


AzureManagement | APIs
API Management s...
Demo Conference API
Echo API
Add parameter
Headers
Add header
Apply product scope
No products
Request URL
HTTP request
Send
Trace

HTTP response

Message Trace Generate definition

```
HTTP/1.1 200 OK
cache-control: no-cache
content-length: 40606
content-type: application/vnd.collection+json
date: Fri, 11 Nov 2022 10:13:17 GMT
expires: -1
pragma: no-cache
vary: Origin
x-aspnet-version: 4.0.30319
x-powered-by: ASP.NET
```

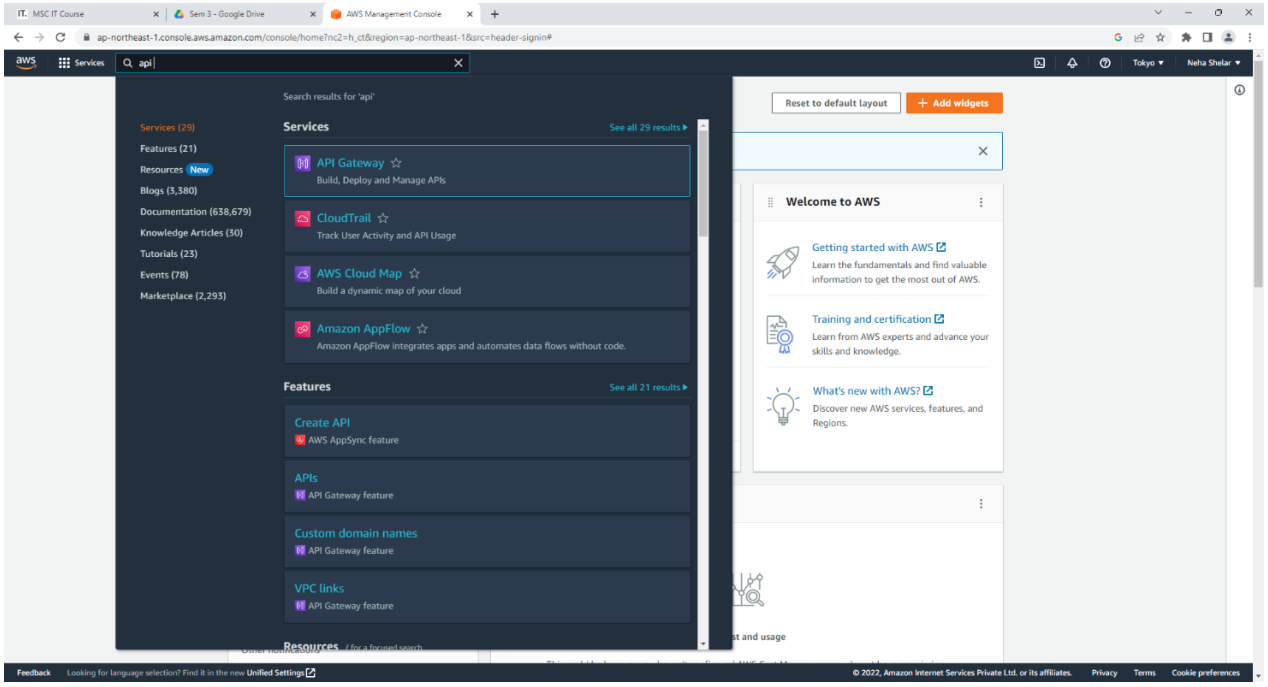

```
{
    "collection": {
        "version": "1.0",
        "href": "https://conferenceapi.azurewebsites.net:443/speakers",
        "links": [],
        "items": [{
                "href": "https://conferenceapi.azurewebsites.net/speaker/1",
```

Aim: 4B. Create an API Gateway Service.

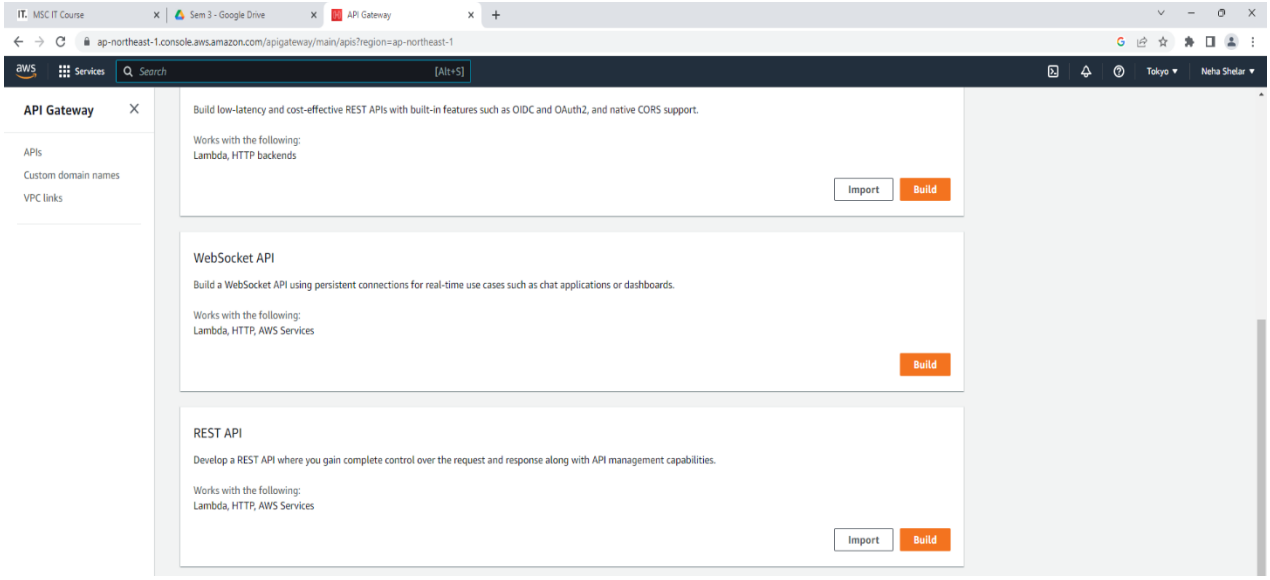
Source Code:

Step 1: Sign in to you AWS account.

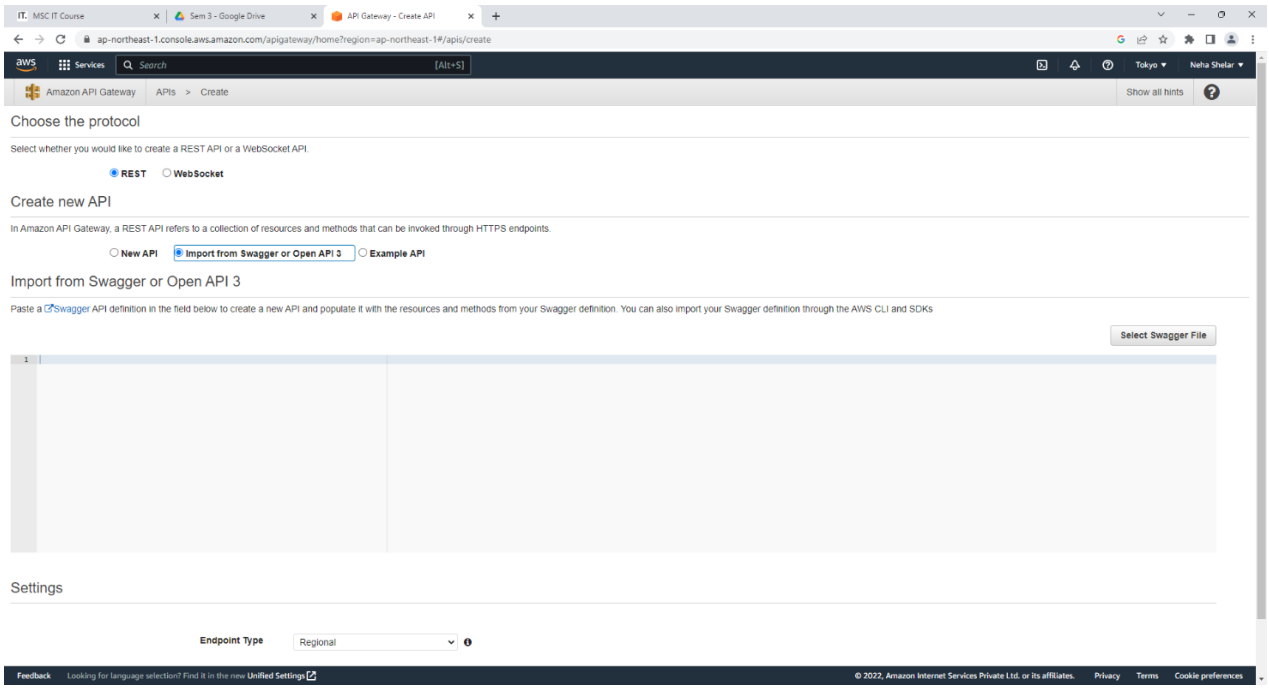
Step 2: search for API Gateway, and create an AWS Gateway instance as shown in Image 1.



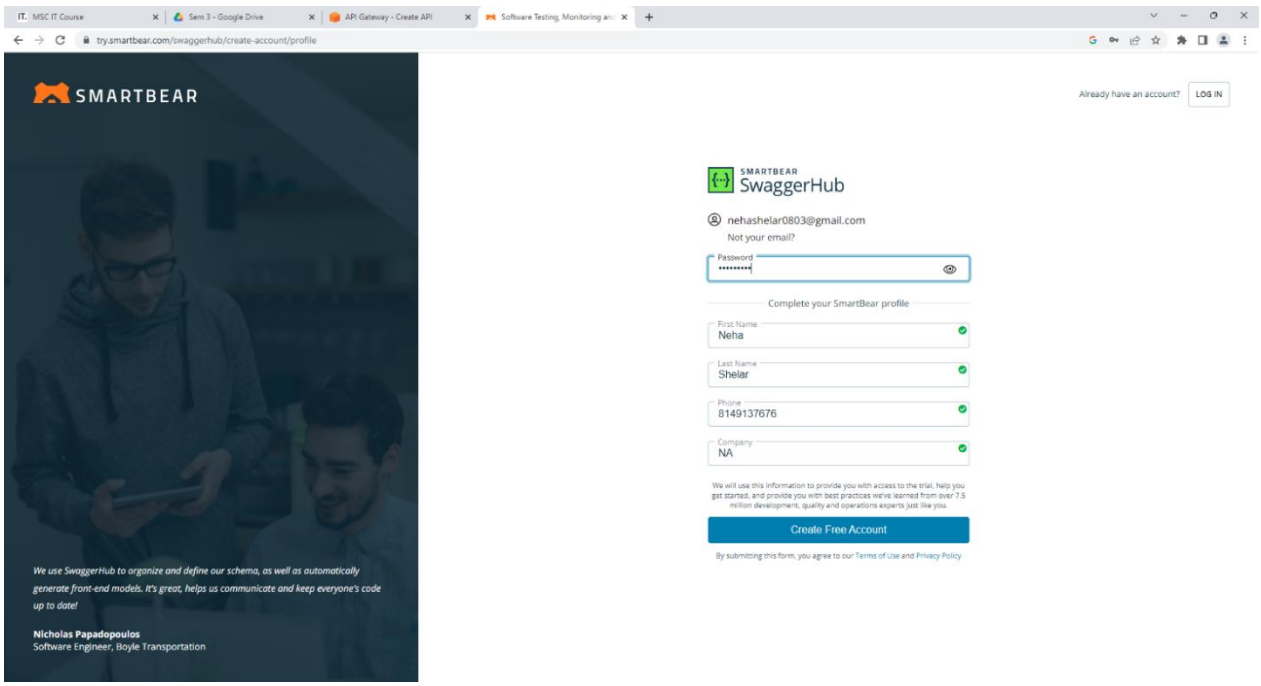
Step 3: Select Import from the REST API selection as shown in image 2.

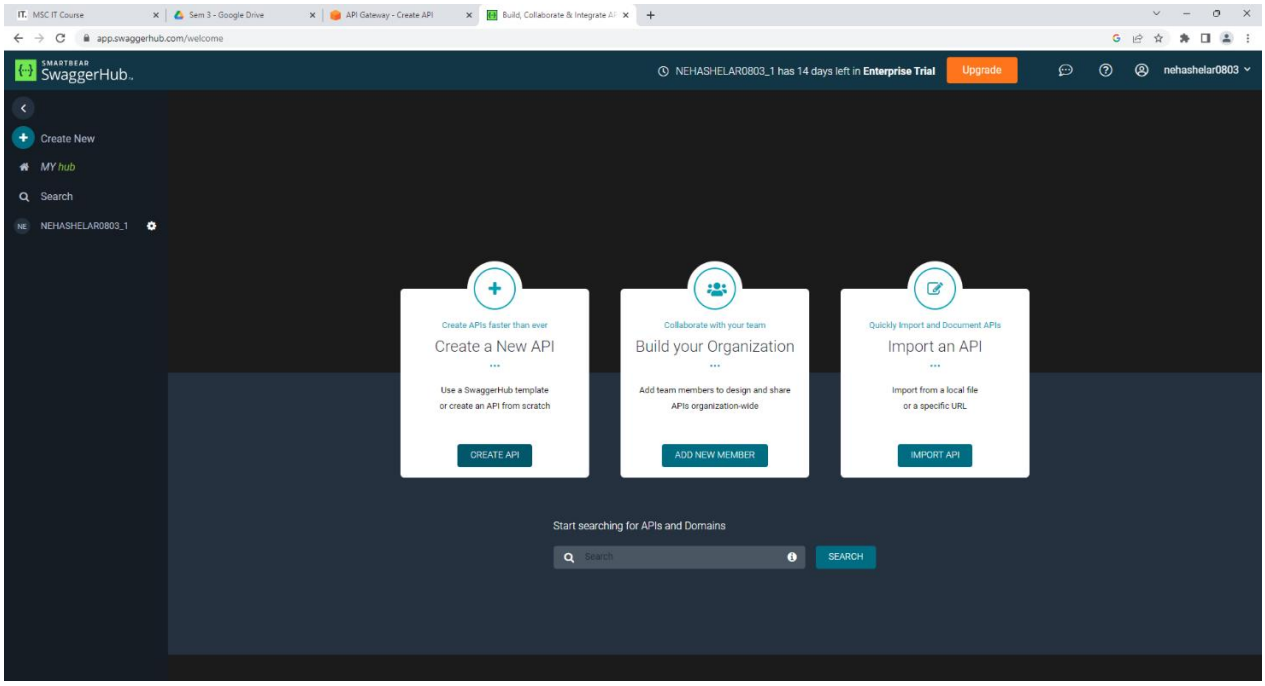


Step 4: Select Import from swagger and copy or select the swagger file as shown in image 3

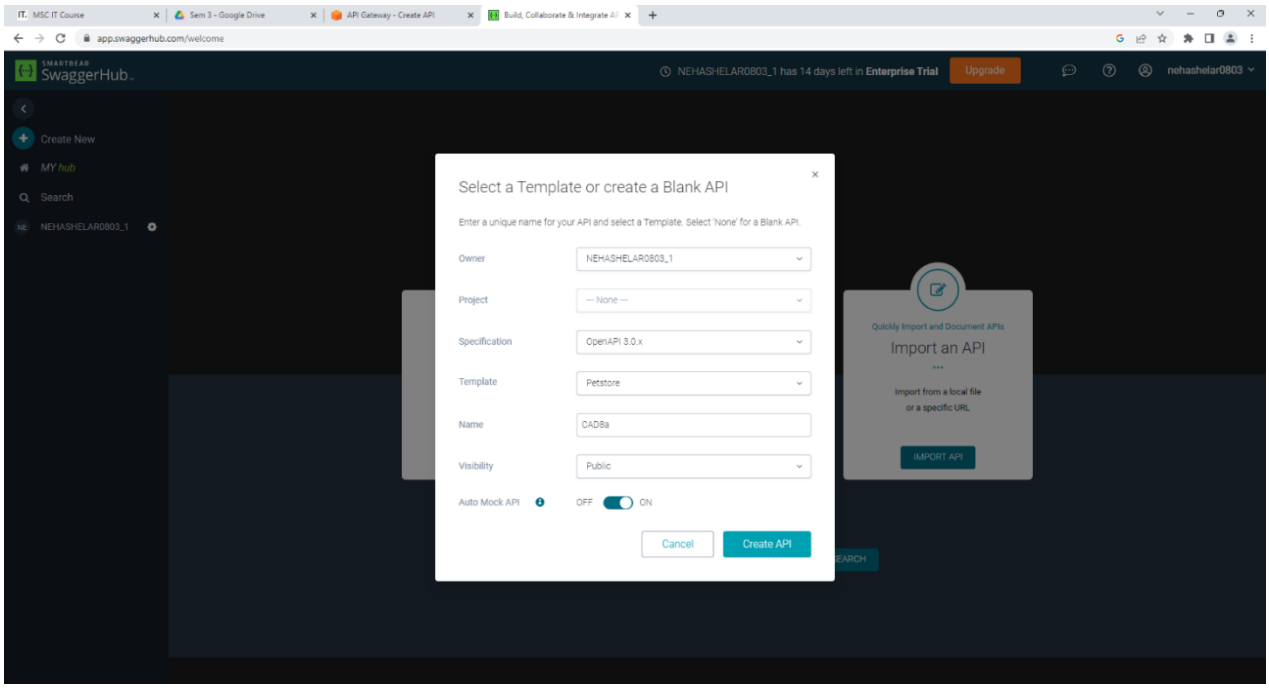


Step 5: To Create an API definition in Swagger. First login to the swagger account. As depicted in image 4.

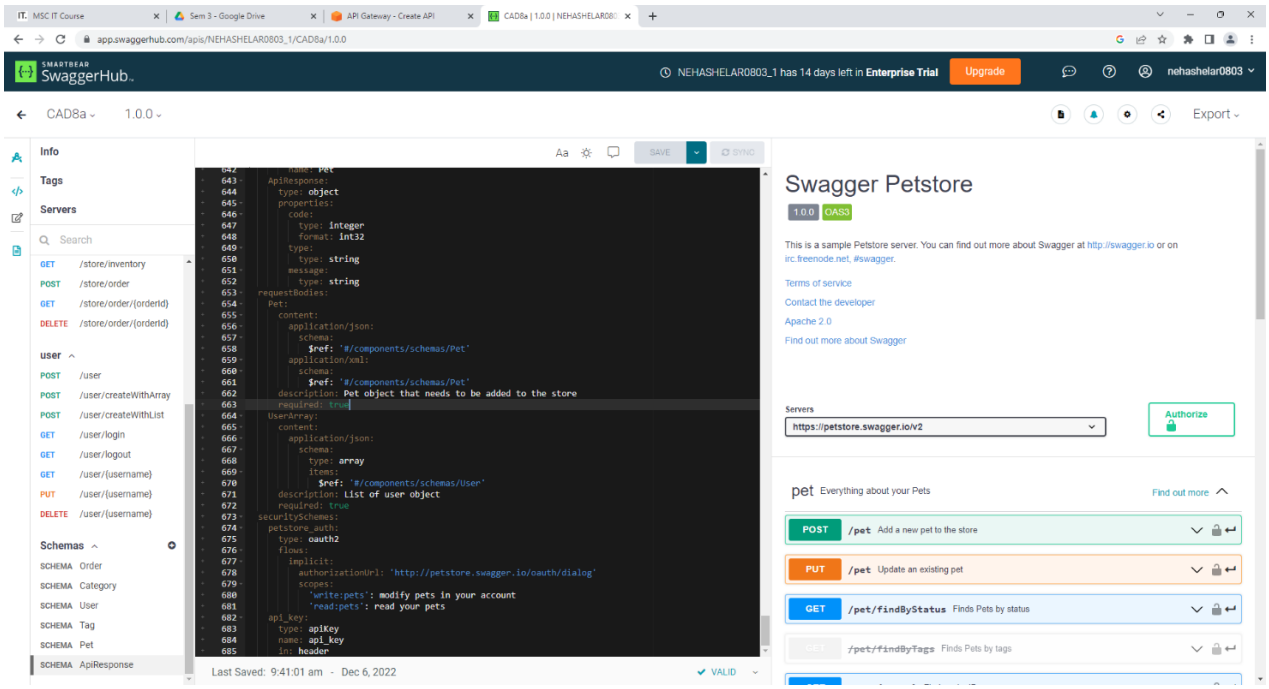




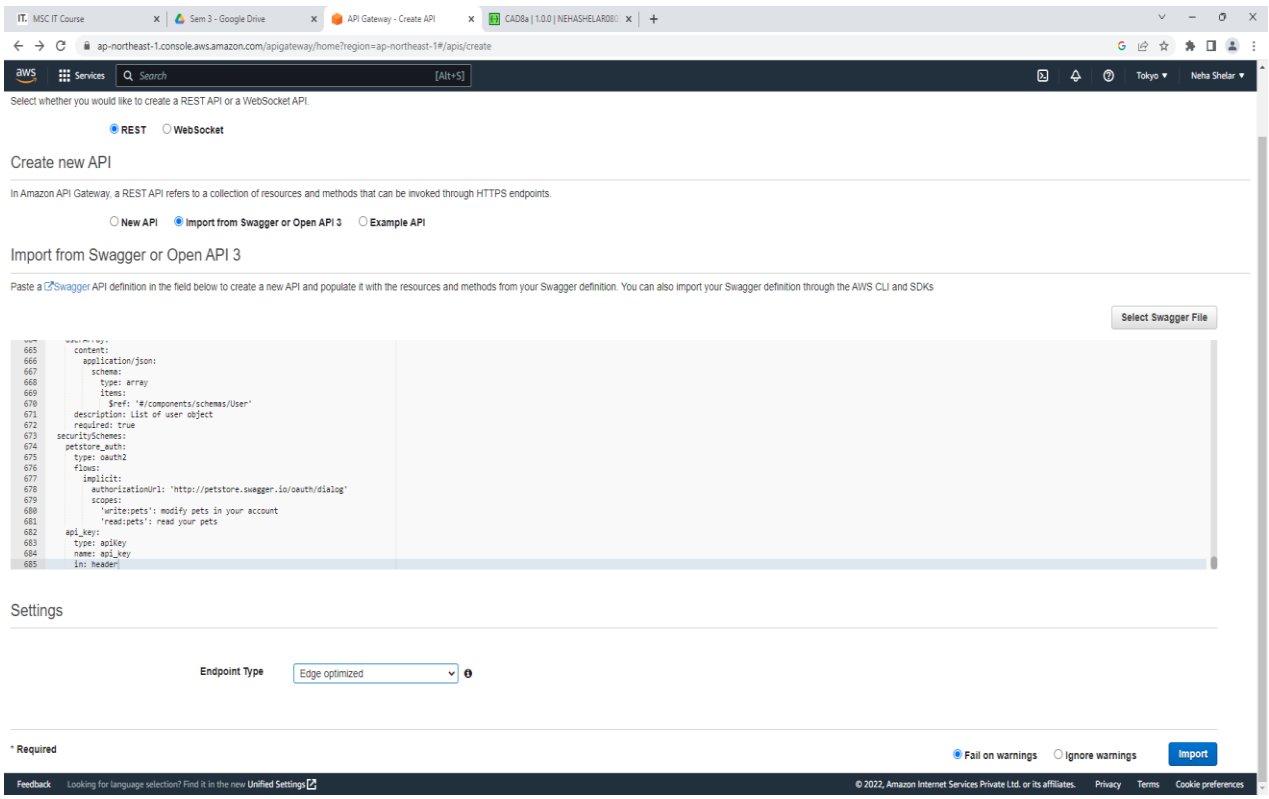
Step 6: Then fill the Form which includes the version, name and other details click on Create API button. As depicted in image 5.



Step 7: Copy the API and definition from swagger as shown in image 6



Step 8: Paste the definition and in In the Settings section, select the endpoint type. Select "Edge Optimized" option and then click on import button



```
openapi: 3.0.0
info:
  description: |
    This is a sample Petstore server.  You can find out more about Swagger at
    [http://swagger.io](http://swagger.io) or on
    [irc.freenode.net, #swagger](http://swagger.io/irc/).
  version: "1.0.0"
  title: Swagger Petstore
  termsOfService: 'http://swagger.io/terms/'
  contact:
    email: apiteam@swagger.io
  license:


    name: Apache 2.0
    url: 'http://www.apache.org/licenses/LICENSE-2.0.html'
servers:
  # Added by API Auto Mocking Plugin
  - description: SwaggerHub API Auto Mocking
    url: https://virtserver.swaggerhub.com/NEHASHELAR0803_1/CAD8a/1.0.0
  - url: 'https://petstore.swagger.io/v2'
tags:
  - name: pet
    description: Everything about your Pets
    externalDocs:
      description: Find out more
      url: 'http://swagger.io'
  - name: store
    description: Access to Petstore orders
  - name: user
    description: Operations about user
```

```
  externalDocs:
    description: Find out more about our store
    url: 'http://swagger.io'
paths:
 /pet:
  post:
    tags:
      - pet
    summary: Add a new pet to the store
    operationId: addPet
    responses:
      '405':
        description: Invalid input
    security:
     - petstore_auth:
         - 'write:pets'
         - 'read:pets'
    requestBody:
      $ref: '#/components/requestBodies/Pet'
  put:
    tags:
      - pet
    summary: Update an existing pet
    operationId: updatePet
    responses:
      '400':
        description: Invalid ID supplied
      '404':
        description: Pet not found
      '405':
```

```
      description: Validation exception
    security:
      - petstore_auth:
          - 'write:pets'
          - 'read:pets'
    requestBody:
      $ref: '#/components/requestBodies/Pet'
/pet/findByStatus:
  get:
    tags:
      - pet
    summary: Finds Pets by status
    description: Multiple status values can be provided with comma separated strings
    operationId: findPetsByStatus
    parameters:
      - name: status
        in: query
        description: Status values that need to be considered for filter
        required: true
        explode: true
        schema:
          type: array
          items:
            type: string
            enum:
              - available
              - pending
              - sold
            default: available
    responses:
```

```
    '200':
      description: successful operation
      content:
        application/json:
          schema:
            type: array
            items:
              $ref: '#/components/schemas/Pet'
        application/xml:
          schema:
            type: array
            items:
              $ref: '#/components/schemas/Pet'
    '400':
      description: Invalid status value
  security:
    - petstore_auth:
        - 'write:pets'
        - 'read:pets'
/pet/findByTags:


 get:
  tags:
    - pet
  summary: Finds Pets by tags
  description: >-
    Muliple tags can be provided with comma separated strings. Use\ \ tag1,
    tag2, tag3 for testing.
  operationId: findPetsByTags
  parameters:
```

```
  - name: tags
    in: query
    description: Tags to filter by
    required: true
    explode: true
    schema:
      type: array
      items:
        type: string
responses:
  '200':
    description: successful operation
    content:
      application/json:
        schema:
          type: array
          items:
            $ref: '#/components/schemas/Pet'
      application/xml:


        schema:
          type: array
          items:
            $ref: '#/components/schemas/Pet'
  '400':
    description: Invalid tag value
security:
  - petstore_auth:
      - 'write:pets'
      - 'read:pets'
```

```
    deprecated: true
'/pet/{petId}':
  get:
    tags:
      - pet
    summary: Find pet by ID
    description: Returns a single pet
    operationId: getPetById
    parameters:
      - name: petId
        in: path
        description: ID of pet to return
        required: true
        schema:
          type: integer
          format: int64
    responses:
      '200':

        description: successful operation
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Pet'
          application/xml:
            schema:
              $ref: '#/components/schemas/Pet'
      '400':
        description: Invalid ID supplied
      '404':
```

```
      description: Pet not found
    security:
      - api_key: []
  post:
    tags:
      - pet
    summary: Updates a pet in the store with form data
    operationId: updatePetWithForm
    parameters:
      - name: petId
        in: path
        description: ID of pet that needs to be updated
        required: true
        schema:
          type: integer
          format: int64
    responses:
      '405':

        description: Invalid input
    security:
      - petstore_auth:
          - 'write:pets'
          - 'read:pets'
    requestBody:
      content:
        application/x-www-form-urlencoded:
          schema:
            type: object
            properties:
```

```
        name:
          description: Updated name of the pet
          type: string
        status:
          description: Updated status of the pet
          type: string
delete:
  tags:
    - pet
  summary: Deletes a pet
  operationId: deletePet
  parameters:
    - name: api_key
      in: header
      required: false
      schema:
        type: string
    - name: petId
      in: path
      description: Pet id to delete
      required: true
      schema:
        type: integer
        format: int64
  responses:

    '400':
      description: Invalid ID supplied
    '404':
      description: Pet not found
```

```
    security:
      - petstore_auth:
          - 'write:pets'
          - 'read:pets'
  '/pet/{petId}/uploadImage':
    post:
      tags:
        - pet
      summary: uploads an image
      operationId: uploadFile
      parameters:
        - name: petId
          in: path
          description: ID of pet to update
          required: true
          schema:
            type: integer
            format: int64
      responses:
        '200':
          description: successful operation
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/ApiResponse'
      security:
        - petstore_auth:
            - 'write:pets'
            - 'read:pets'
      requestBody:
```

```
      content:
        application/octet-stream:
          schema:
            type: string
            format: binary
/store/inventory:
  get:
    tags:
      - store
    summary: Returns pet inventories by status
    description: Returns a map of status codes to quantities
    operationId: getInventory
    responses:
      '200':
        description: successful operation
        content:
          application/json:
            schema:
              type: object
              additionalProperties:
                type: integer
                format: int32
    security:
      - api_key: []
/store/order:
  post:
    tags:
      - store
    summary: Place an order for a pet
    operationId: placeOrder
```

```
    responses:
      '200':
        description: successful operation
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Order'
          application/xml:
            schema:
              $ref: '#/components/schemas/Order'
      '400':
        description: Invalid Order
    requestBody:
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/Order'
      description: order placed for purchasing the pet
      required: true
  '/store/order/{orderId}':
    get:
      tags:
        - store
      summary: Find purchase order by ID
      description: >-
        For valid response try integer IDs with value >= 1 and <= 10.\ \ Other
        values will generated exceptions
      operationId: getOrderById
      parameters:
        - name: orderId
```

```
    in: path
    description: ID of pet that needs to be fetched
    required: true
    schema:
     type: integer
     format: int64
     minimum: 1
     maximum: 10
  responses:
   '200':
    description: successful operation
    content:
     application/json:
      schema:
       $ref: '#/components/schemas/Order'
     application/xml:
      schema:
       $ref: '#/components/schemas/Order'
   '400':
    description: Invalid ID supplied
   '404':
    description: Order not found
 delete:
  tags:
   - store
  summary: Delete purchase order by ID
  description: >-
   For valid response try integer IDs with positive integer value.\ \
   Negative or non-integer values will generate API errors
  operationId: deleteOrder
```

```
    parameters:
      - name: orderId
        in: path
        description: ID of the order that needs to be deleted
        required: true
        schema:
          type: integer
          format: int64
          minimum: 1
    responses:
      '400':
        description: Invalid ID supplied
      '404':
        description: Order not found
/user:
  post:
    tags:
      - user
    summary: Create user
    description: This can only be done by the logged in user.
    operationId: createUser
    responses:
      default:
        description: successful operation
    requestBody:
      content:
        application/json:
          schema:

            $ref: '#/components/schemas/User'
```

```
    description: Created user object
    required: true
/user/createWithArray:
  post:
    tags:
      - user
    summary: Creates list of users with given input array
    operationId: createUsersWithArrayInput
    responses:
      default:
        description: successful operation
    requestBody:
      $ref: '#/components/requestBodies/UserArray'
/user/createWithList:
  post:
    tags:
      - user
    summary: Creates list of users with given input array
    operationId: createUsersWithListInput
    responses:
      default:
        description: successful operation
    requestBody:
      $ref: '#/components/requestBodies/UserArray'
/user/login:
  get:
    tags:
      - user
    summary: Logs user into the system
    operationId: loginUser
```

```
parameters:
  - name: username
    in: query
    description: The user name for login
    required: true
    schema:
      type: string
  - name: password
    in: query
    description: The password for login in clear text
    required: true
    schema:
      type: string
responses:
  '200':
    description: successful operation
    headers:
      X-Rate-Limit:
        description: calls per hour allowed by the user
        schema:
          type: integer
          format: int32
      X-Expires-After:
        description: date in UTC when token expires
        schema:
          type: string
          format: date-time
    content:
      application/json:
        schema:
```

```
            type: string
          application/xml:
            schema:
              type: string
      '400':
        description: Invalid username/password supplied
/user/logout:
  get:
    tags:
      - user
    summary: Logs out current logged in user session
    operationId: logoutUser
    responses:
      default:
        description: successful operation
'/user/{username}':
  get:
    tags:
      - user
    summary: Get user by user name
    operationId: getUserByName
    parameters:
      - name: username
        in: path
        description: The name that needs to be fetched. Use user1 for testing.
        required: true
        schema:

          type: string
    responses:
```

```
    '200':
      description: successful operation
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/User'
        application/xml:
          schema:
            $ref: '#/components/schemas/User'
    '400':
      description: Invalid username supplied
    '404':
      description: User not found
  put:
    tags:
    - user
    summary: Updated user
    description: This can only be done by the logged in user.
    operationId: updateUser
    parameters:
    - name: username
      in: path
      description: name that need to be updated
      required: true
      schema:
        type: string
    responses:
    '400':
      description: Invalid user supplied
    '404':
```

```
        description: User not found
    requestBody:


      content:
        application/json:
          schema:
            $ref: '#/components/schemas/User'
      description: Updated user object
      required: true
  delete:
    tags:
      - user
    summary: Delete user
    description: This can only be done by the logged in user.
    operationId: deleteUser
    parameters:
      - name: username
        in: path
        description: The name that needs to be deleted
        required: true
        schema:
          type: string
    responses:
      '400':
        description: Invalid username supplied
      '404':
        description: User not found
externalDocs:
  description: Find out more about Swagger
  url: 'http://swagger.io'
```

```
components:
  schemas:
    Order:

      type: object
      properties:
        id:
          type: integer
          format: int64
        petId:
          type: integer
          format: int64
        quantity:
          type: integer
          format: int32
        shipDate:
          type: string
          format: date-time
        status:
          type: string
          description: Order Status
          enum:
            - placed
            - approved
            - delivered
        complete:
          type: boolean
          default: false
      xml:
        name: Order
```

```
Category:
  type: object
  properties:

    id:
      type: integer
      format: int64
    name:
      type: string
  xml:
    name: Category
User:
  type: object
  properties:
    id:
      type: integer
      format: int64
    username:
      type: string
    firstName:
      type: string
    lastName:
      type: string
    email:
      type: string
    password:
      type: string
    phone:
      type: string
    userStatus:
```

```
      type: integer
      format: int32
      description: User Status
    xml:
      name: User
  Tag:
    type: object
    properties:
      id:
        type: integer
        format: int64
      name:
        type: string
    xml:
      name: Tag
  Pet:
    type: object
    required:
      - name
      - photoUrls
    properties:
      id:
        type: integer
        format: int64
      category:
        $ref: '#/components/schemas/Category'
      name:
        type: string
        example: doggie
      photoUrls:
```

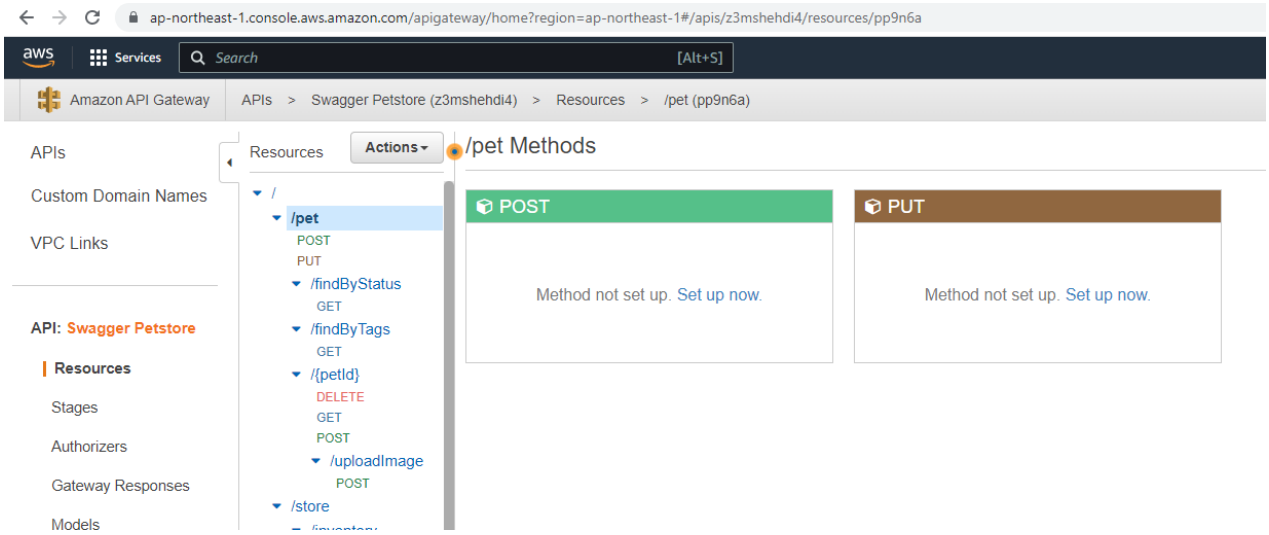
```
      type: array
      xml:
        name: photoUrl
        wrapped: true
      items:
        type: string
    tags:
      type: array
      xml:
        name: tag
        wrapped: true
      items:
        $ref: '#/components/schemas/Tag'
    status:
      type: string
      description: pet status in the store
      enum:
        - available
        - pending
        - sold
  xml:
    name: Pet
ApiResponse:
  type: object
  properties:
    code:
      type: integer
      format: int32
    type:
      type: string
```

```
      message:
        type: string
requestBodies:
  Pet:
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/Pet'
      application/xml:
        schema:
          $ref: '#/components/schemas/Pet'
    description: Pet object that needs to be added to the store
    required: true
  UserArray:
    content:
      application/json:
        schema:
          type: array
          items:
            $ref: '#/components/schemas/User'
    description: List of user object
    required: true
securitySchemes:
  petstore_auth:
    type: oauth2


flows:
      implicit:
        authorizationUrl: 'http://petstore.swagger.io/oauth/dialog'
        scopes:
```

```
        'write:pets': modify pets in your account
        'read:pets': read your pets
  api_key:
    type: apiKey
    name: api_key
    in: header
```

Output:

If the import is successful, we can view the "API Structure in AWS API Gateway" where we can see methods of the API. As shown in image 8

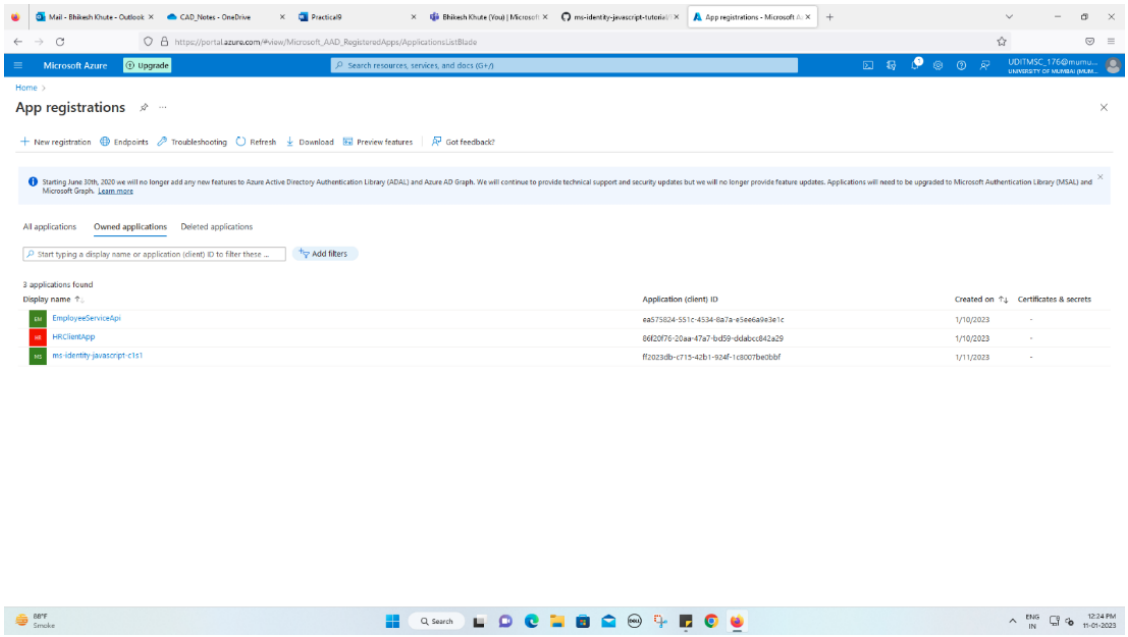


Practical No: 05

Aim: 5a Demonstrate - Securing APIs with Azure Active Directory

Step 1 – Login to Azure Portal and select/search **Azure Active Directory service**.

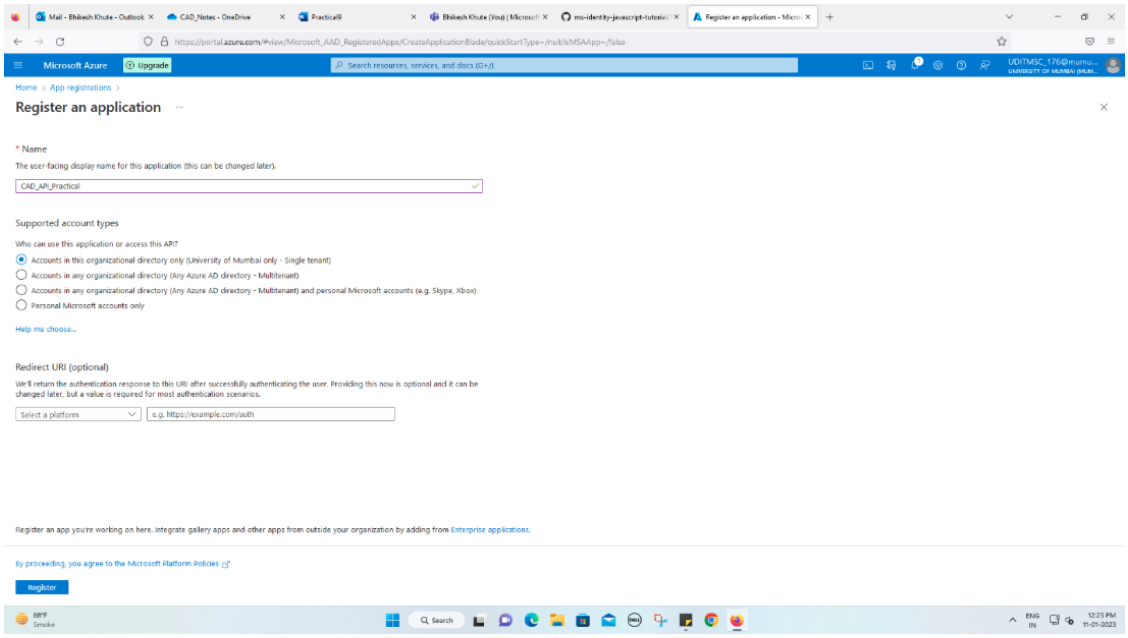
Step 2 - Select the App Registrations blade on the left, then select new registration.



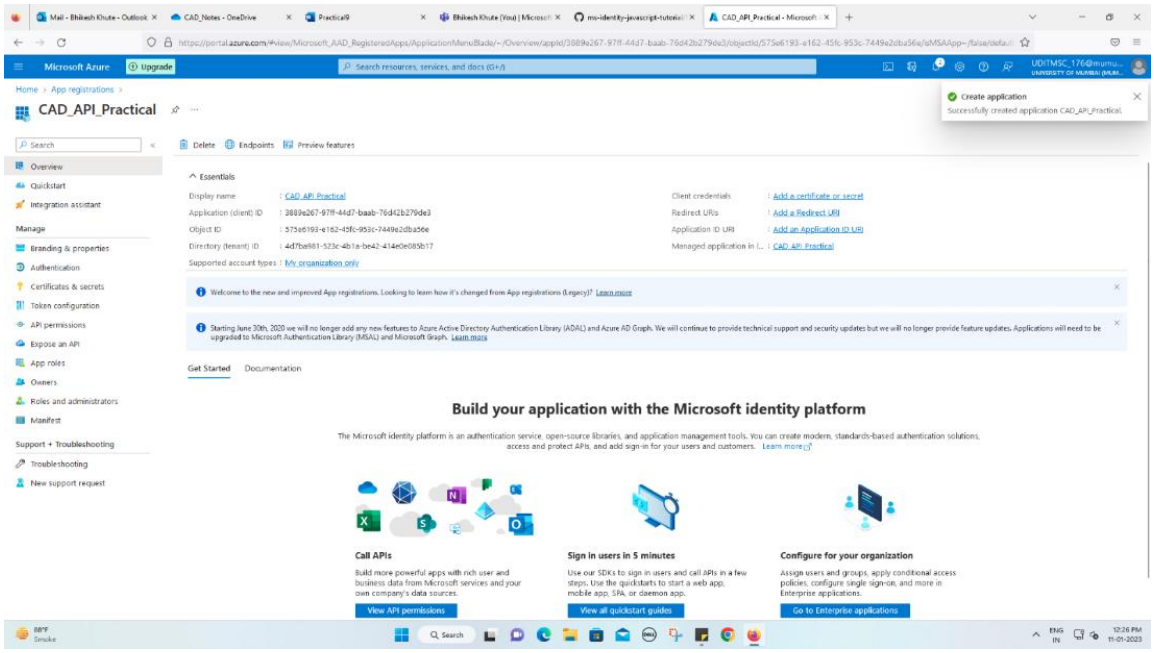
Step 3 – Click on New Registration and

I. add the name of the app as “CAD_API_Practical”.
II. Accounts in this organizational directory only (University of Mumbai only - Single tenant)
III. Ignore the Redirect URI(We’ll see it in the next step)

And click on Register

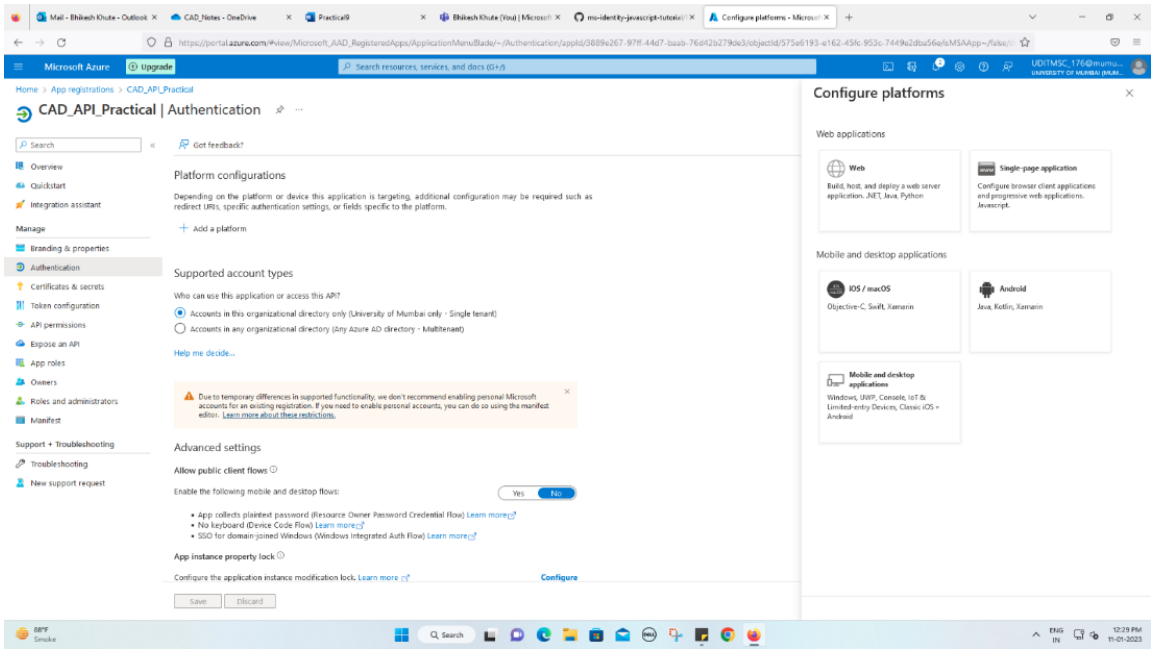


Step 4 – The App is created.

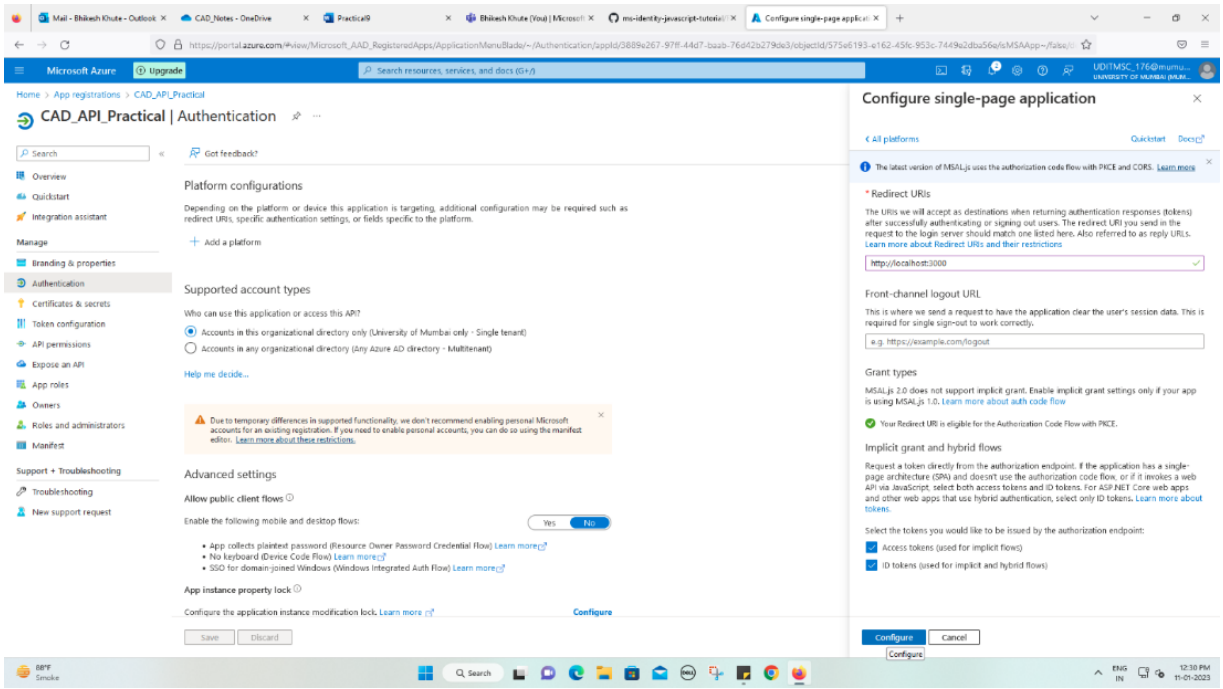


Note down the **Application & Tenant ID** for further use for authentication.

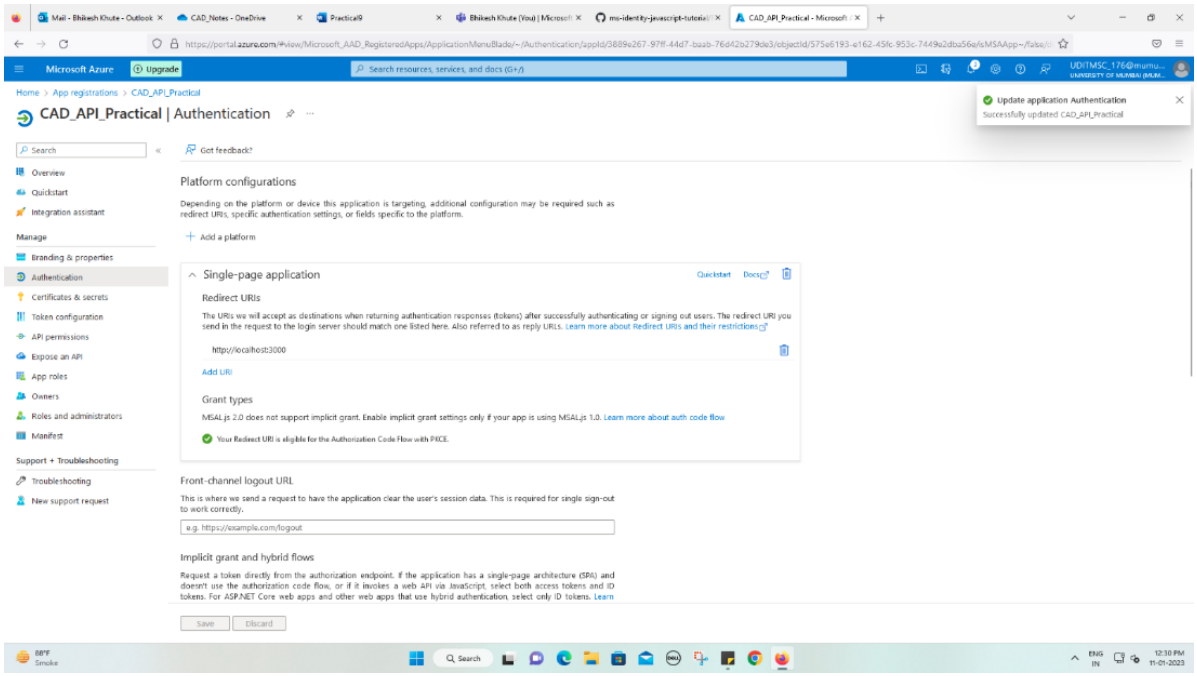
Step 5 – Now click **Add a Redirect URI &** click on +**Add Platform.** Select **Single-page Application**



Step 6 – Add Redirect URI as **http://localhost:3000** & check Access & ID tokens mentioned in the screenshot

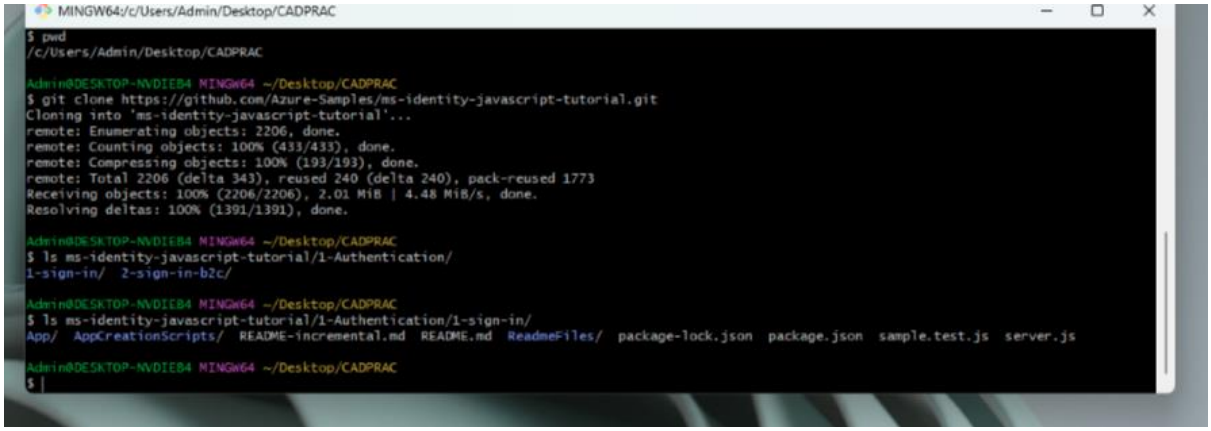


Step 7 – Click on Configure and output is as follow



Step 8 – Now download the application from Github from the below link for secure authentication to the API we created.

git clone https://github.com/Azure-Samples/ms-identity-javascript-tutorial.git



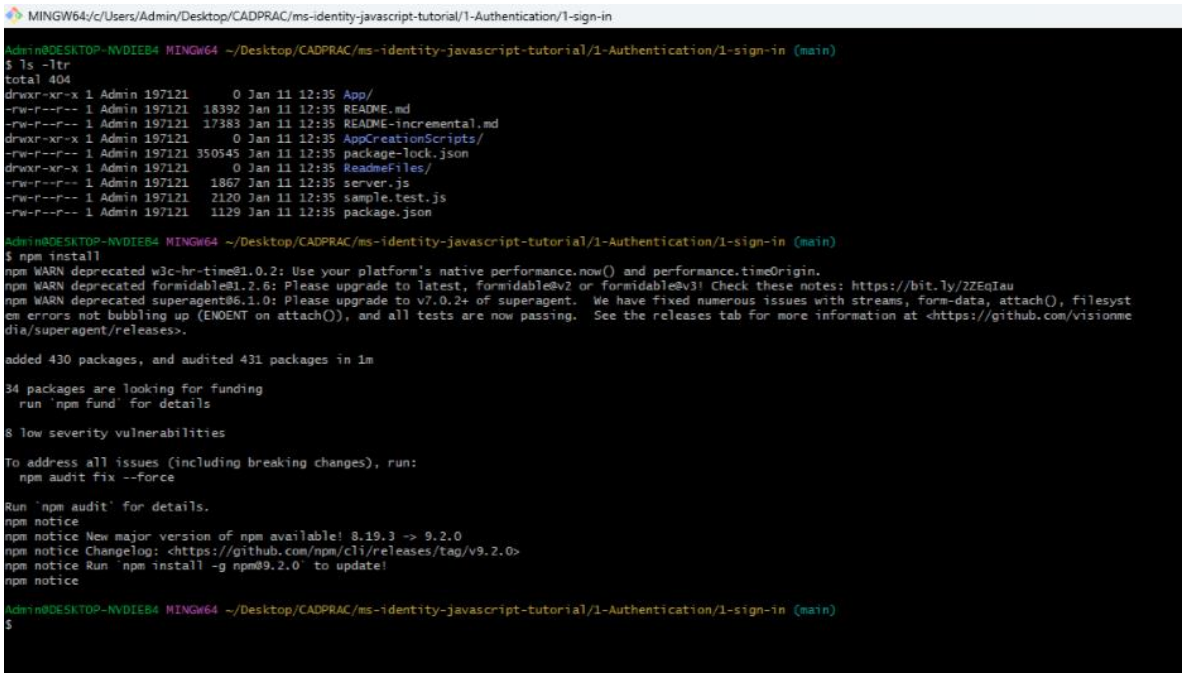
Step 9 – Install NodeJs on the system using the following link -

https://nodejs.org/en/download/

Step 10 – Install the project dependencies using the following command in the mentioned directory below

<folder>/ms-identity-javascript-tutorial/1-Authentication/1-sign-in

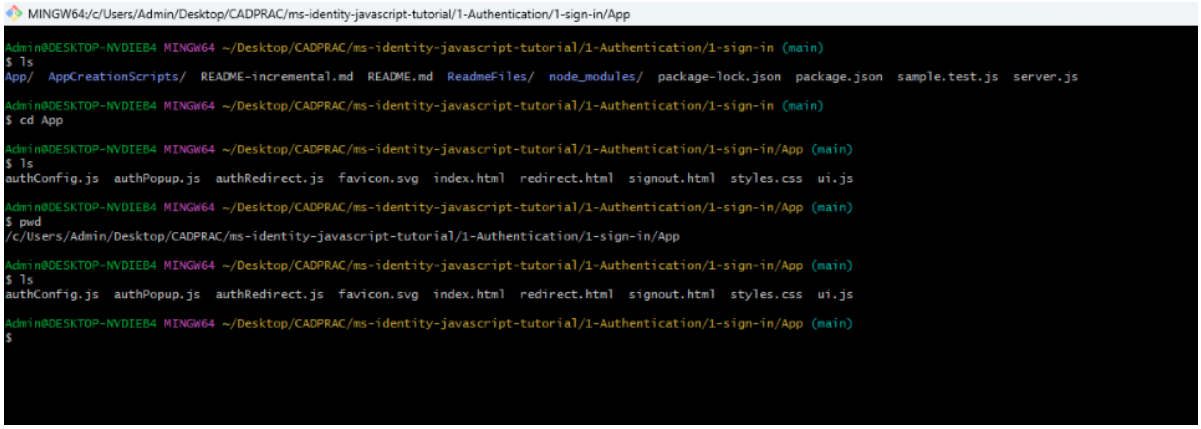
npm install



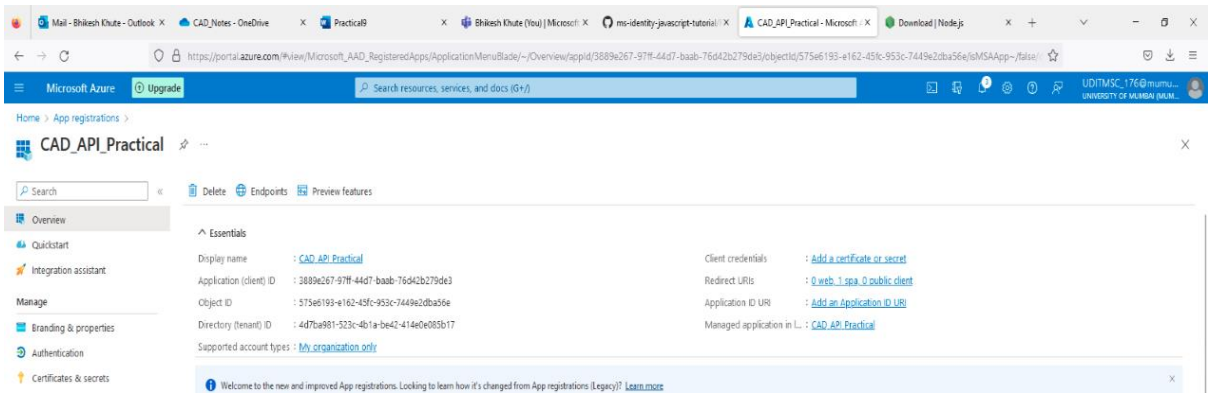
Step 11 – Now Open the App\authConfig.js file to add the App ID & Directory ID for authentication.

Go the path - /ms-identity-javascript-tutorial/1-Authentication/1-sign-in/App

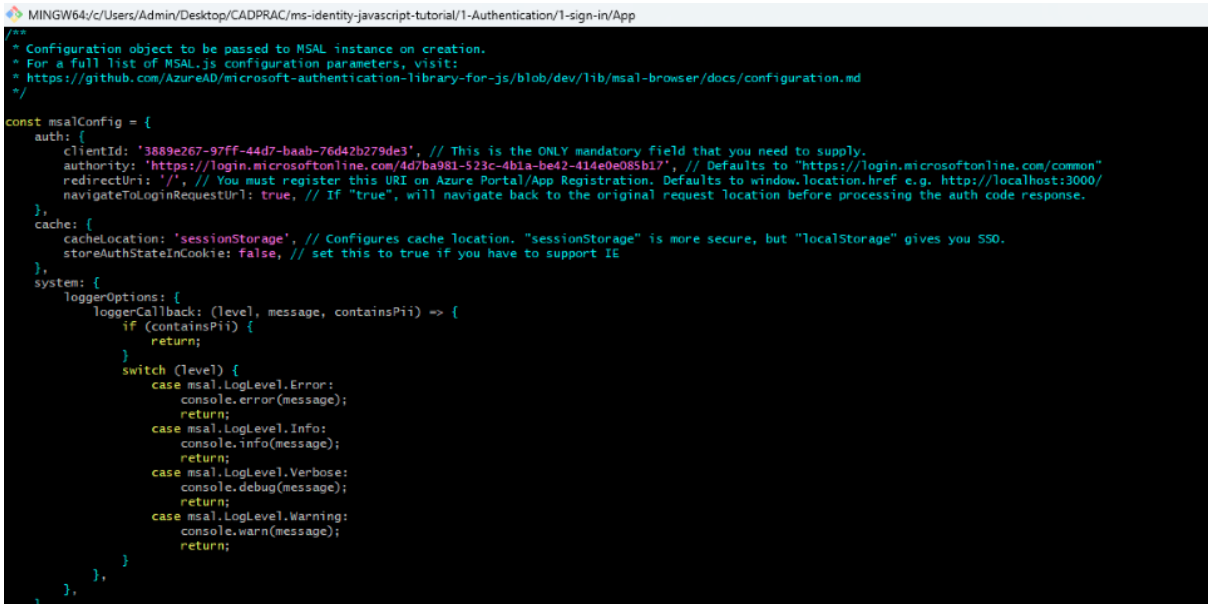
Filname - authConfig.js



Step 12 – Copy the Application ID & Directory ID from the Overview Section

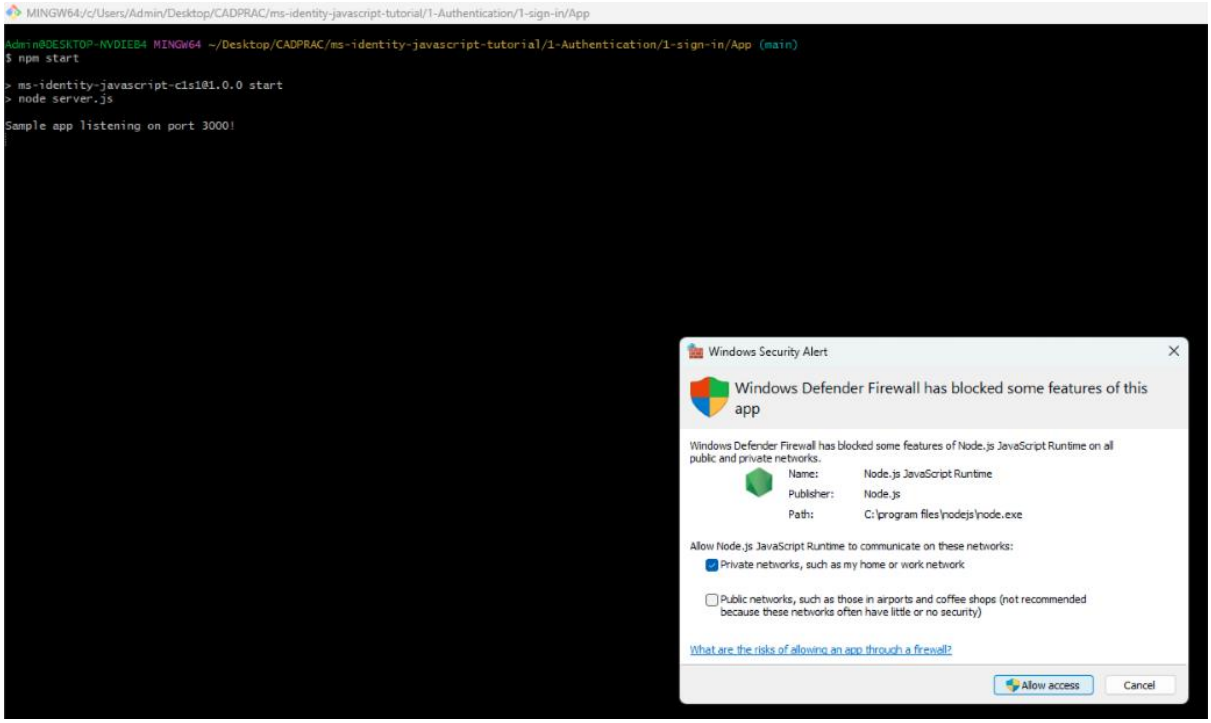


Step 13 – Paste the ID in the authConfig.js file mentioned in the screenshot

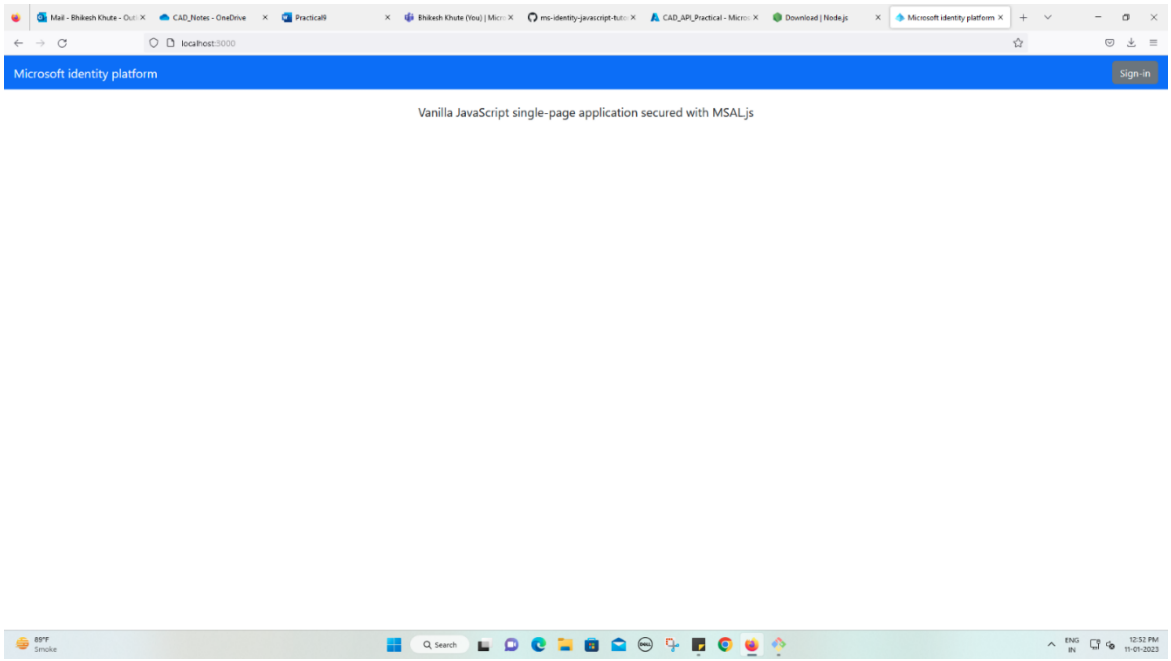


Step 14 – Save the file and run the following command

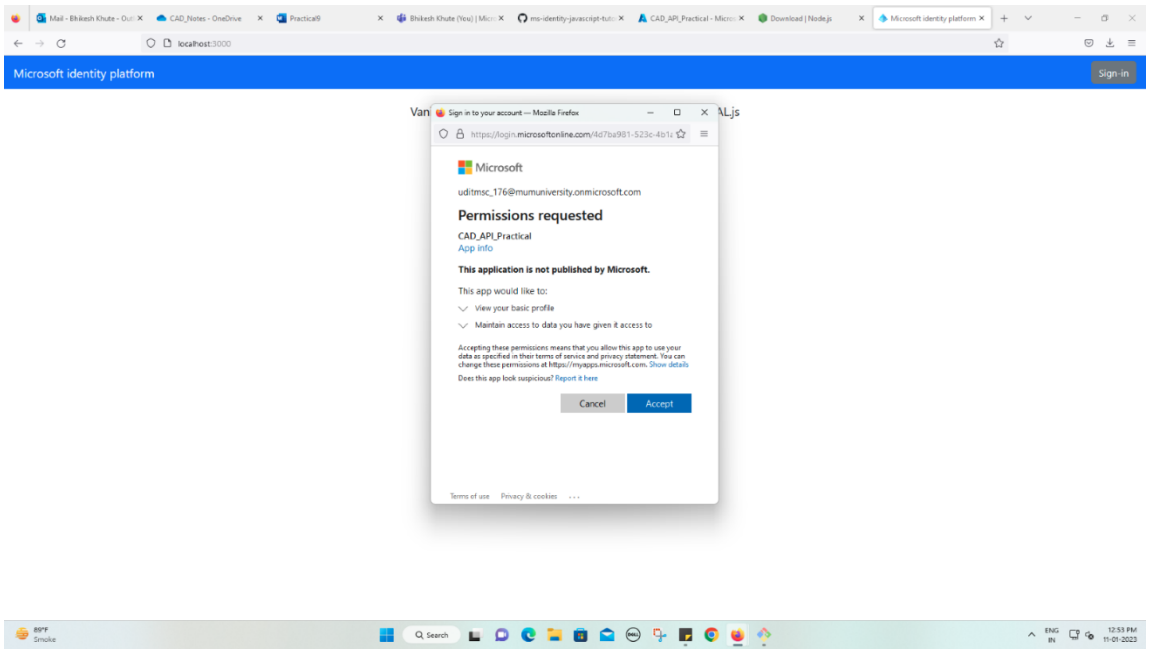
Npm start



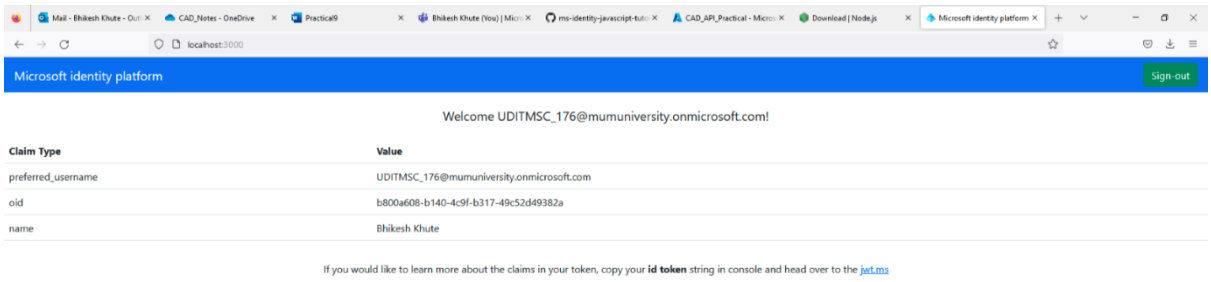
Step 15 – Now go the following URL on the browser – http://localhost:3000



Step 16 – Click on Sign In. A popup will appear



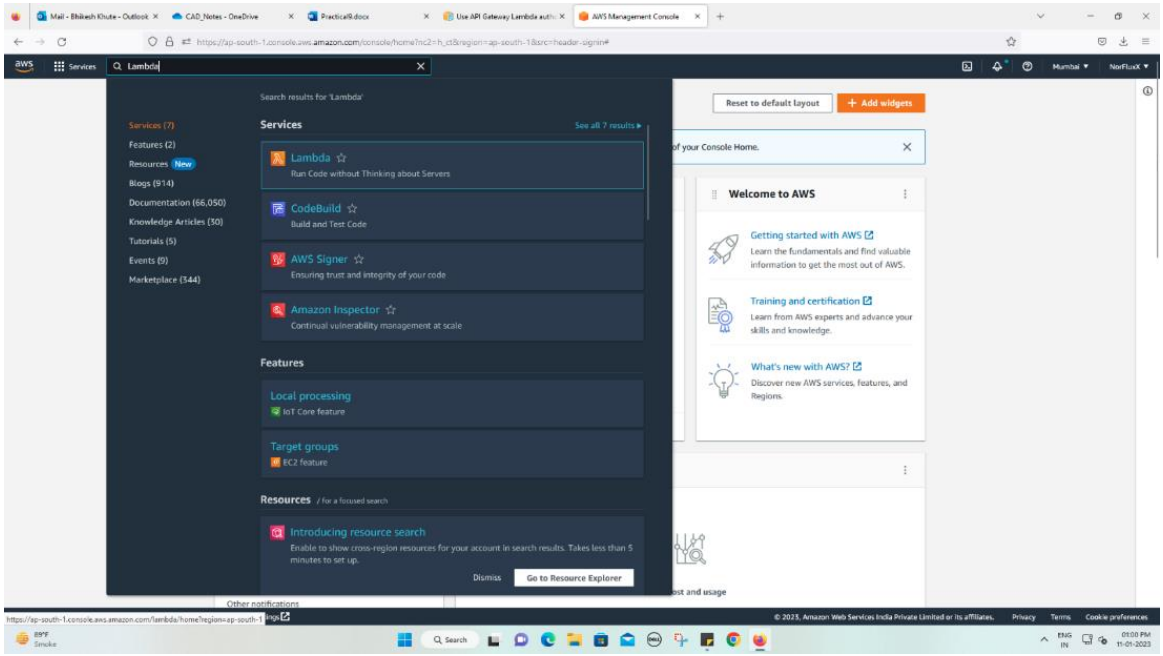
Step 17 – We have successfully signed in using AD securely.



The above following are details about my AD details.

Aim:5B - AWS API Gateway Authorizer

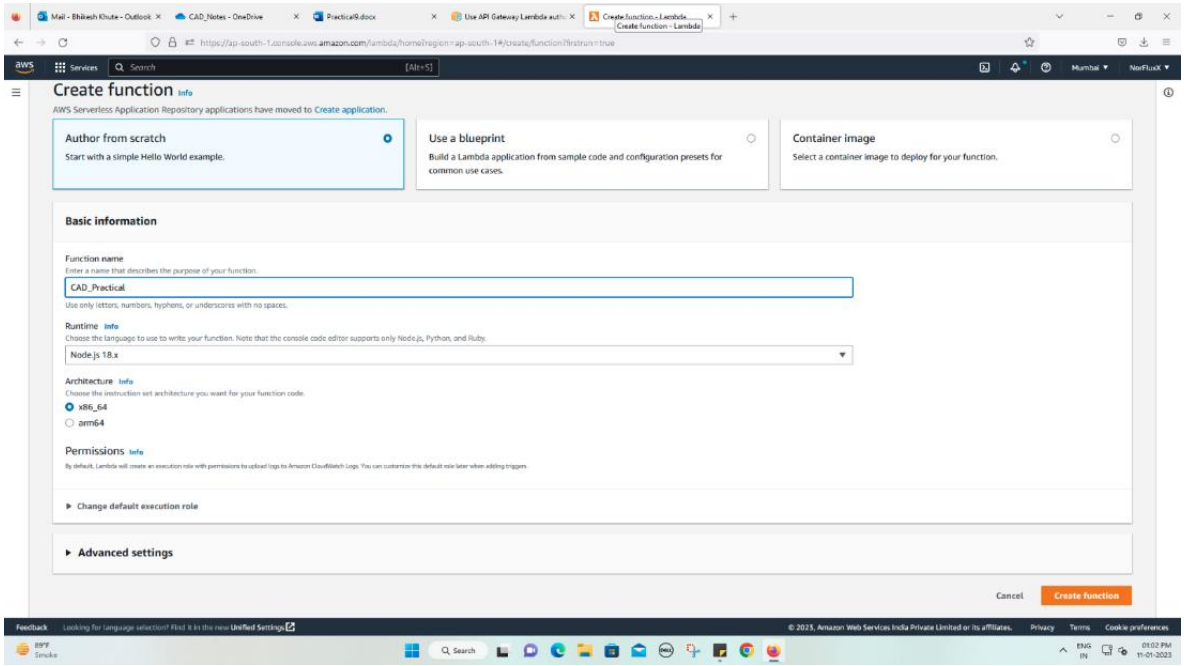
Step 1 – Login to aws.amazon.com and Search for “Lambda”



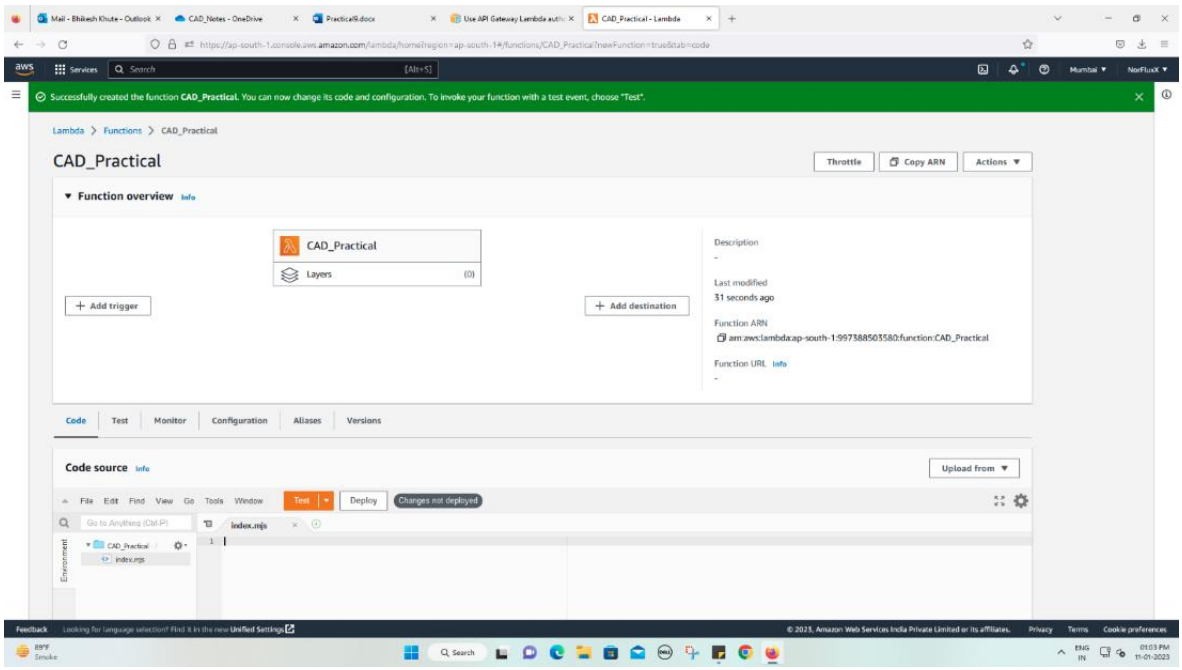
Step 2 – Click on **Create a Function**

Name – CAD_Practical

Leave all others fields as it is and click on **Create a function**



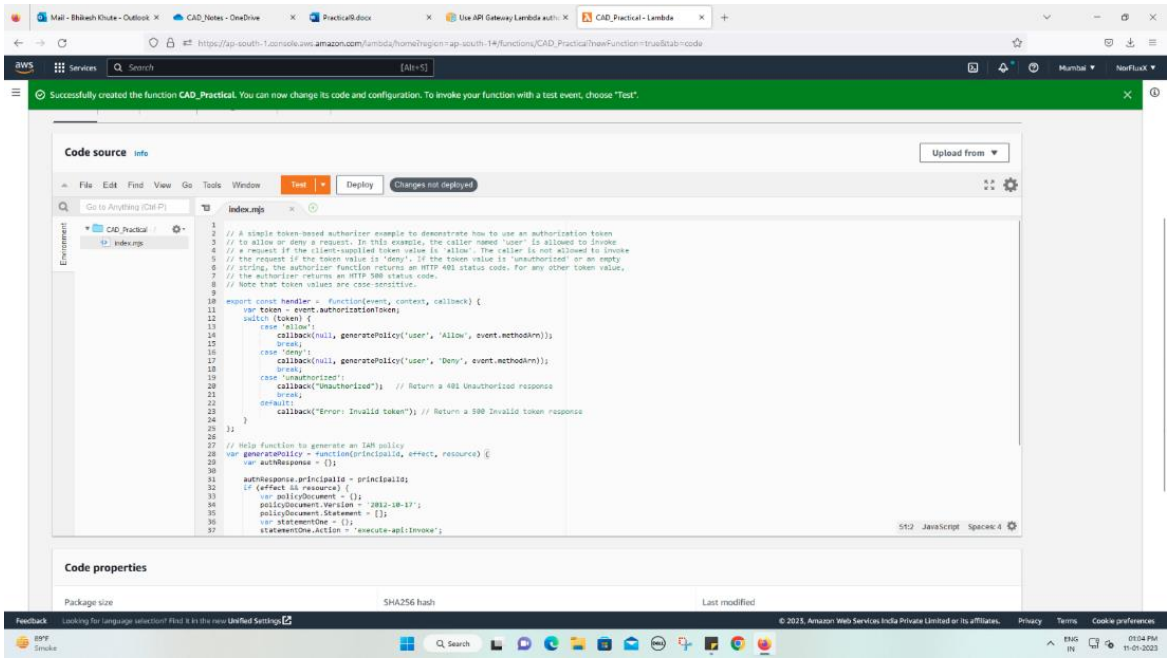
Step 3 – Double click on index.mjs file



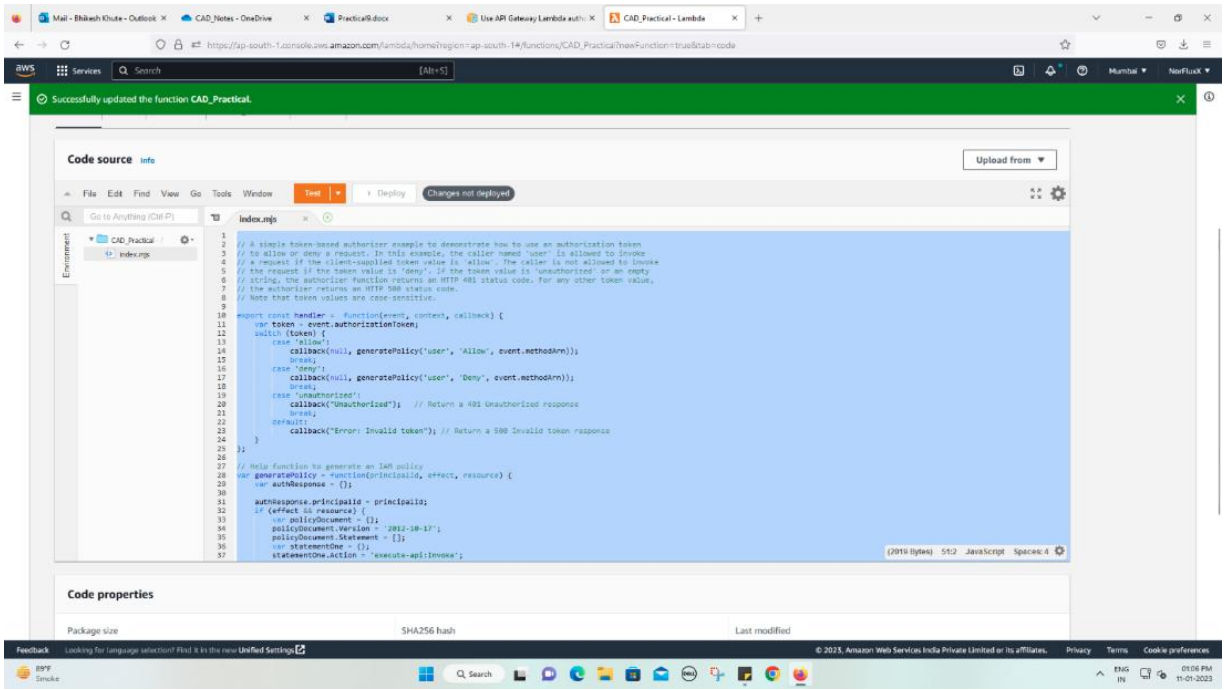
Step 4 – Copy the below and paste the same in the index.mjs editor section.

```
export const handler =  function(event, context, callback) {
    var token = event.authorizationToken;
    switch (token) {
        case 'allow':
            callback(null, generatePolicy('user', 'Allow', event.methodArn));
            break;
        case 'deny':
            callback(null, generatePolicy('user', 'Deny', event.methodArn));
            break;
        case 'unauthorized':
            callback("Unauthorized");   // Return a 401 Unauthorized response
            break;
        default:
            callback("Error: Invalid token"); // Return a 500 Invalid token response
```

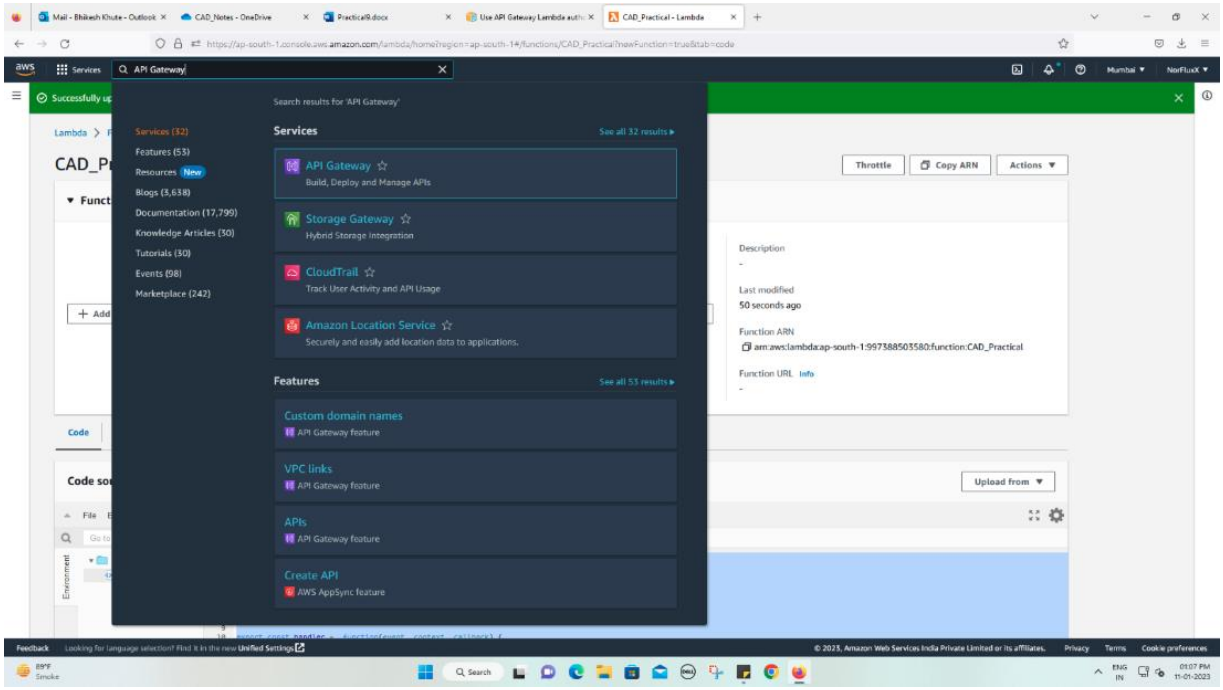
```
    }
};

// Help function to generate an IAM policy
var generatePolicy = function(principalId, effect, resource) {
    var authResponse = {};

    authResponse.principalId = principalId;
    if (effect && resource) {
        var policyDocument = {};
        policyDocument.Version = '2012-10-17';
        policyDocument.Statement = [];
        var statementOne = {};
        statementOne.Action = 'execute-api:Invoke';
        statementOne.Effect = effect;
        statementOne.Resource = resource;
        policyDocument.Statement[0] = statementOne;
        authResponse.policyDocument = policyDocument;
    }

    // Optional output with custom properties of the String, Number or Boolean type.
    authResponse.context = {
        "stringKey": "stringval",
        "numberKey": 123,
        "booleanKey": true
    };
    return authResponse;
}
```



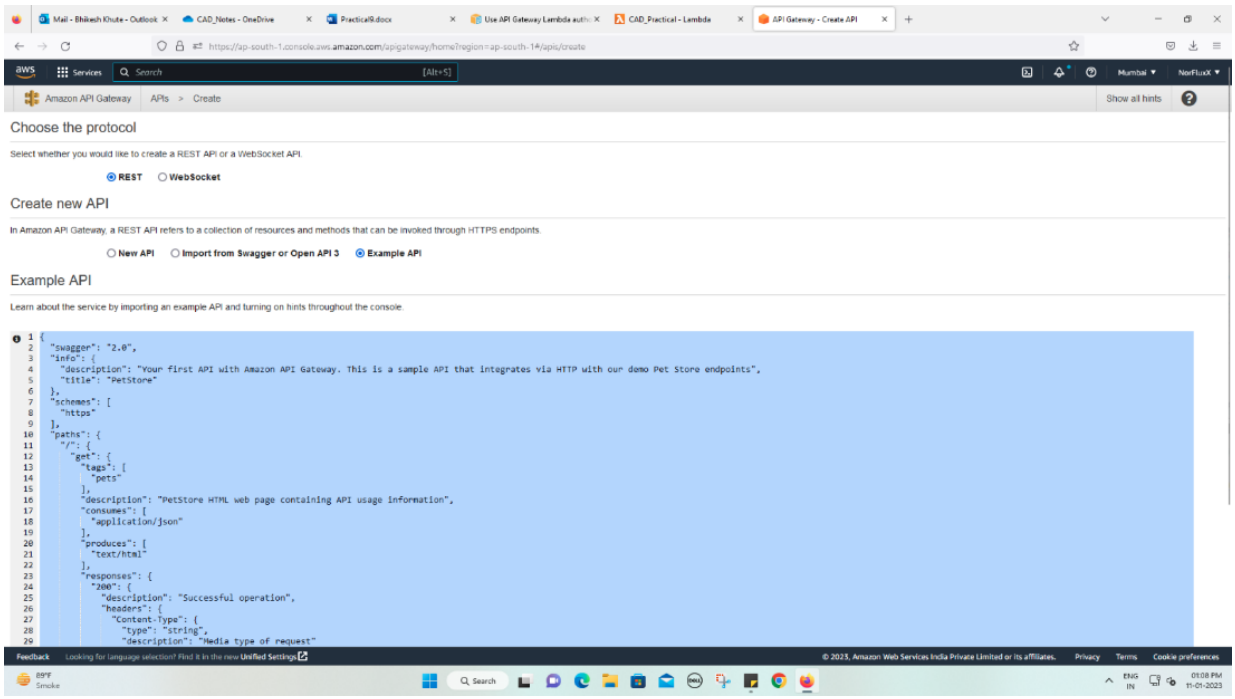
Step 5 – Click on deploy once the code is pasted. The code is successfully deployed



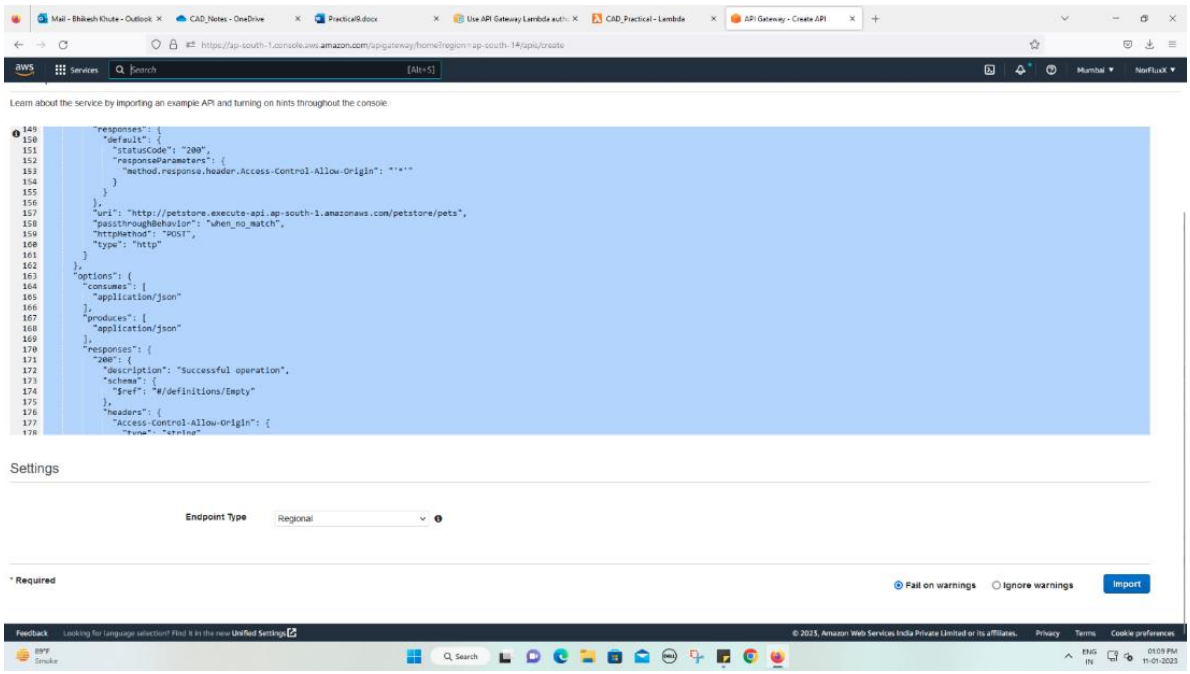
Step 6 – Now search API Gateway in the search bar and open in the new tab



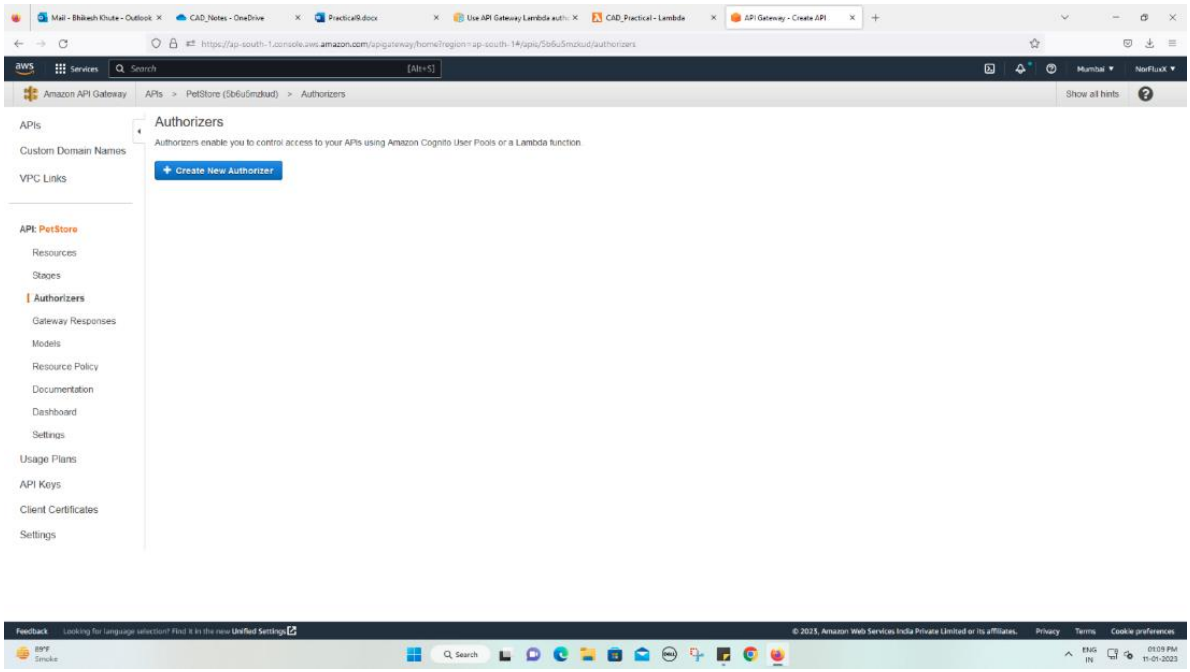
Step 7 – Select Choose an API type - > REST API -> Build



Step 8 – Leave all the fields as it and scroll down to the bottom and click on Import



Step 9 – Now click on the **Authorizers** on the left-hand side as mentioned in the screenshot.



Step 10 – Click on the button - Create New Authorizer and fill in the details as below

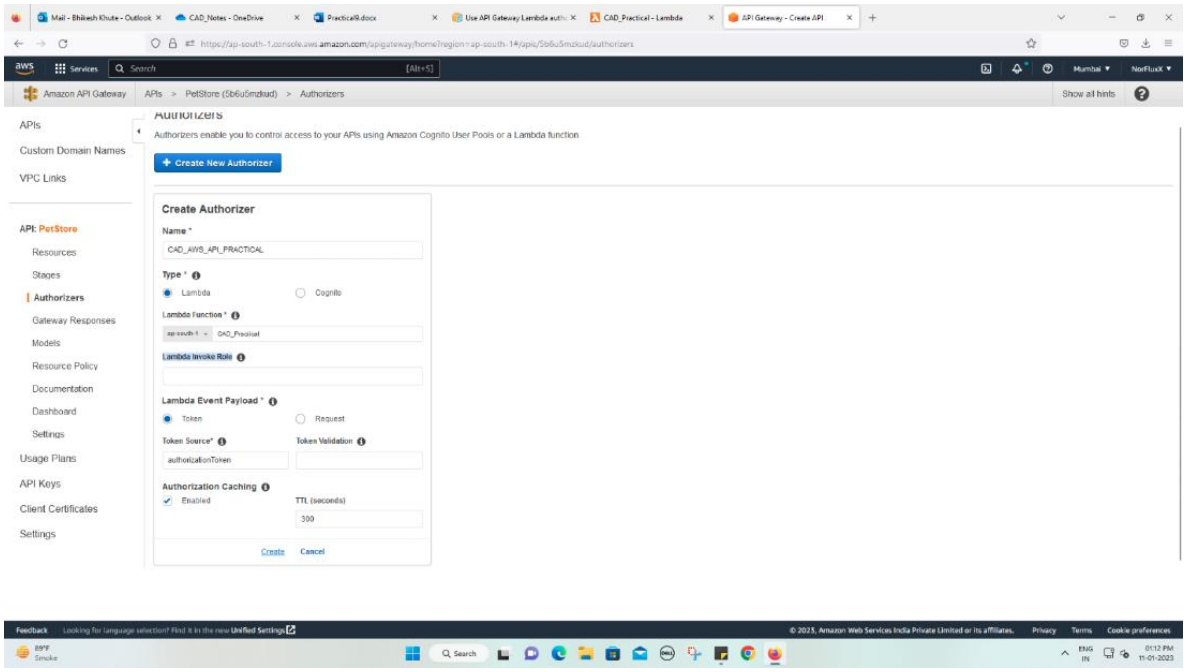
Name – CAD_AWS_API_PRACTICAL

Lambda Function – Automatically CAD_Practical will be displayed and select the same

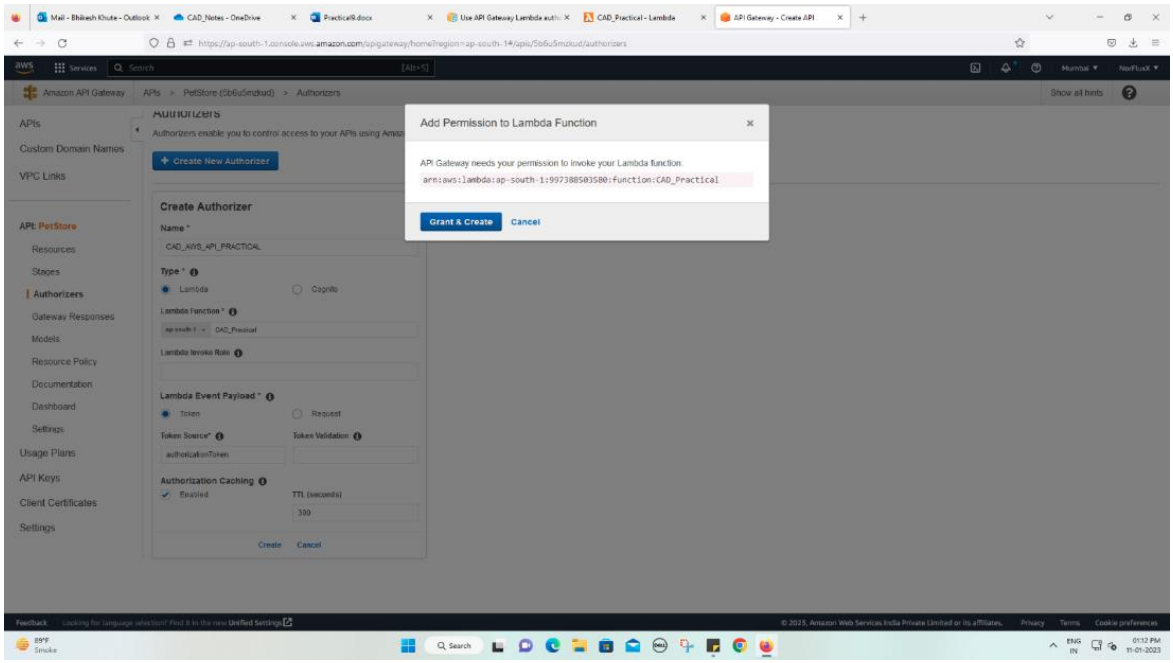
Lambda Invoke Role – Leave it Blank

Token Source – authorizationToken

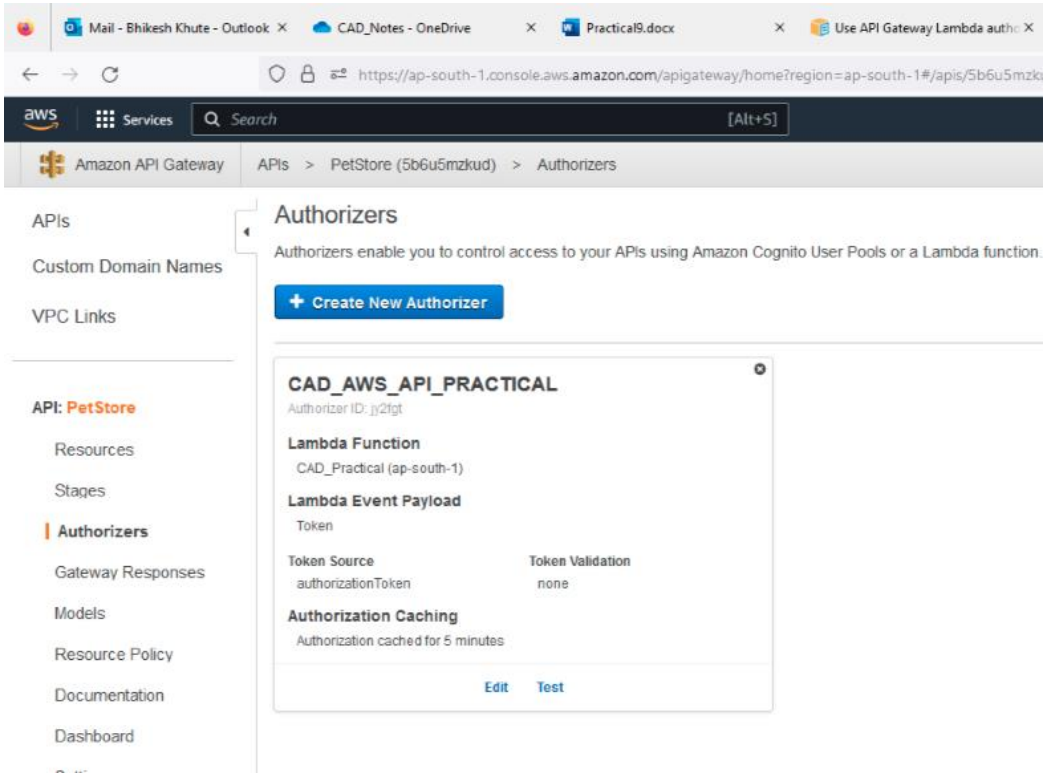
And then click on **Create**



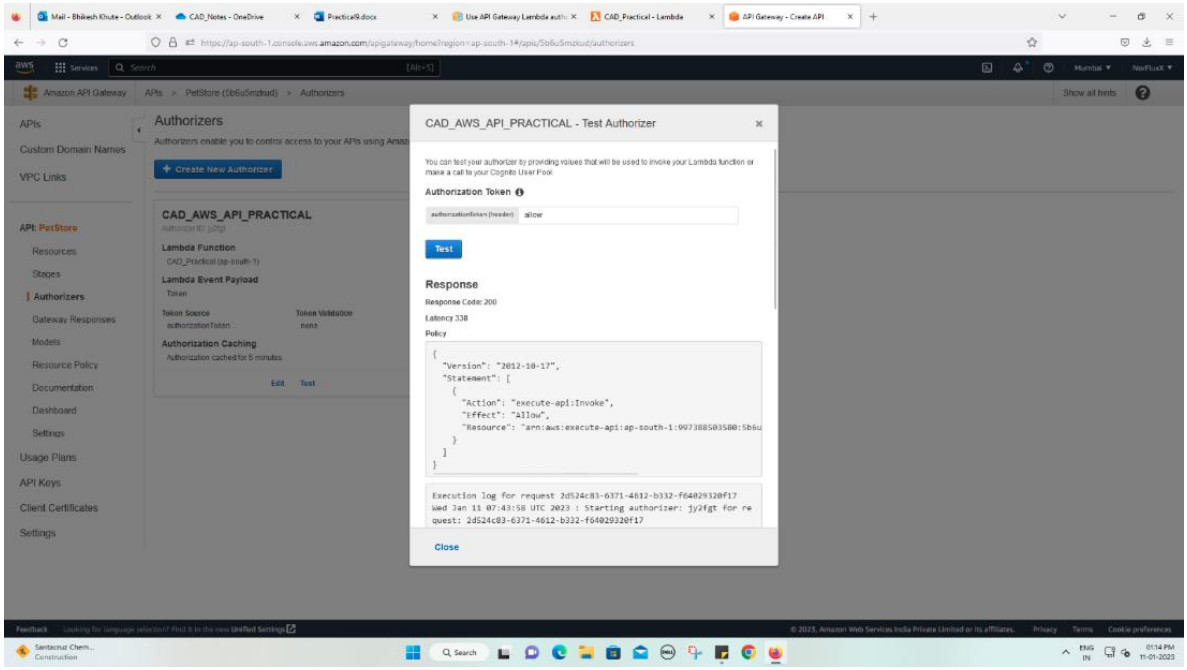
Step 11 – Click on Grant & Create



Step 12 – Now click on the test button



Step 13 – Type **allow** in the box as mentioned in the screenshot



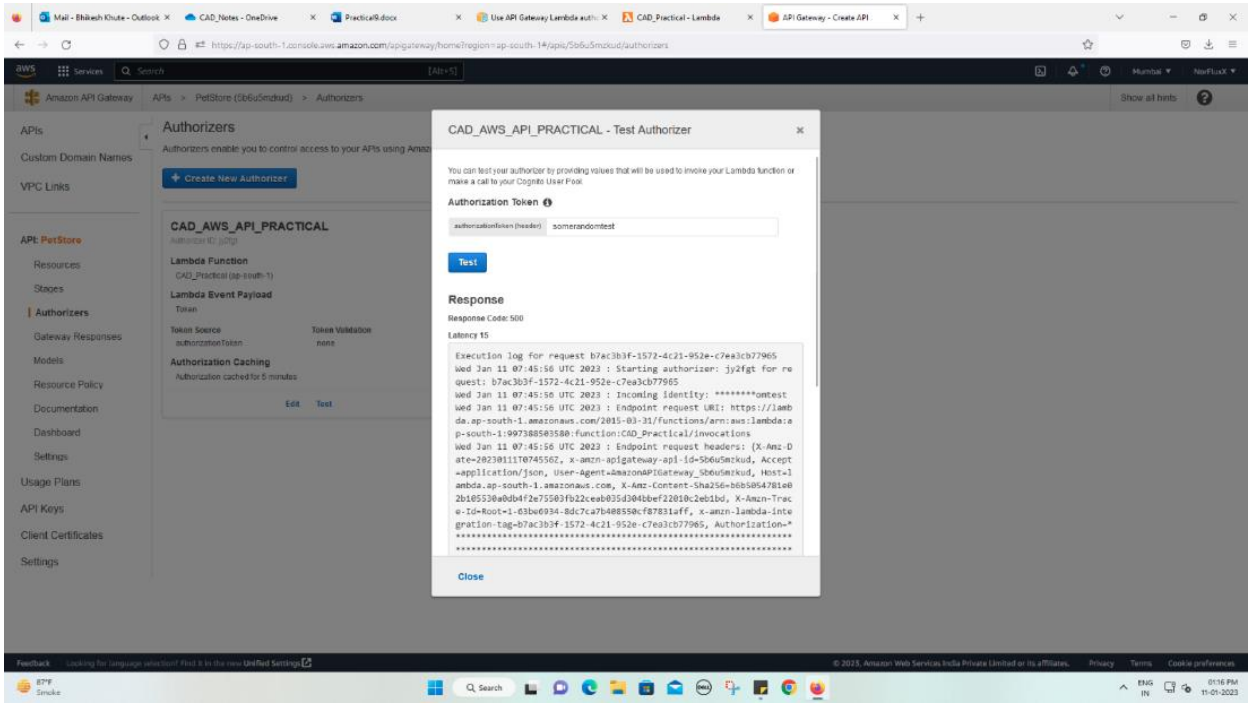
We can see the that function that we created is tokenized. When we input allow, it will check the IAM policy and return policy output of the same.

If we input as **somerandomtest,** then following is the output of the same. An error response will be thrown by the same. As per the function created, it should only be one of the following inputs

'allow':

'deny':

'unauthorized':

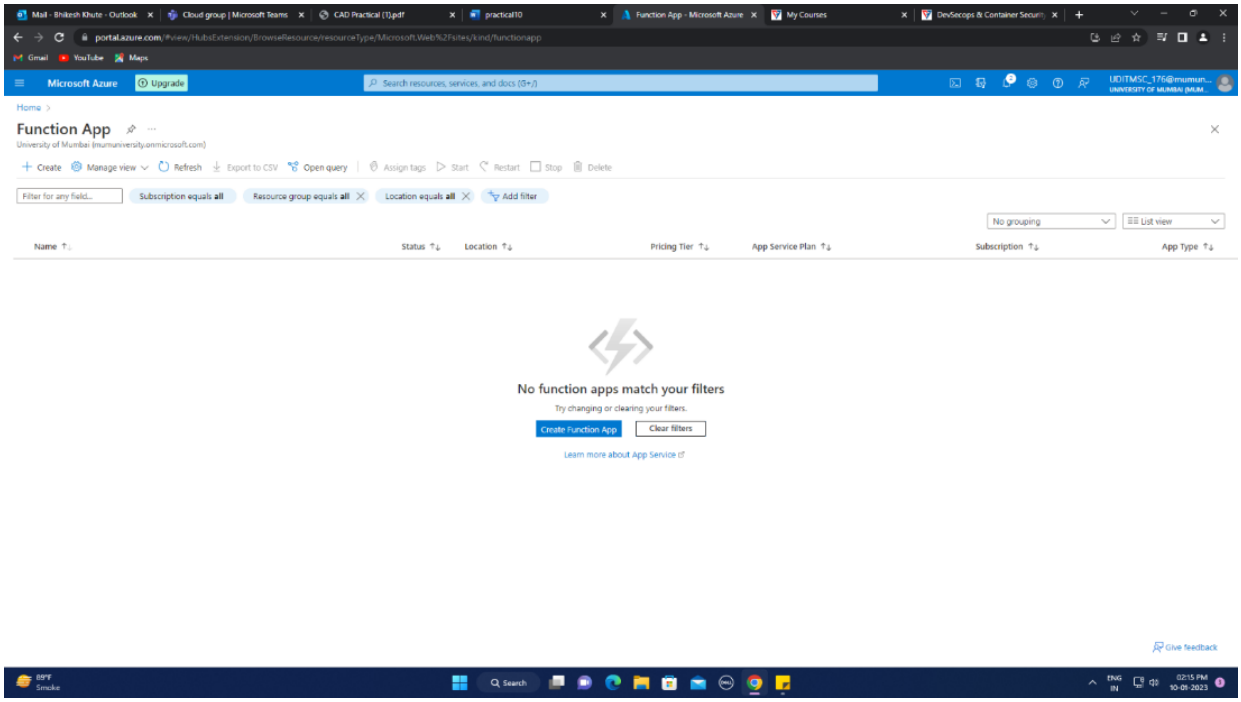


Practical No: 06

Aim: Create a serverless API using Azure functions

Step 1 - Sign in with your Microsoft account to access various Azure functionalities

Step 2 - After Successful sign in, Click on Function app→Click on Create Function App



Step 3 - Click on Resource Group →Create New

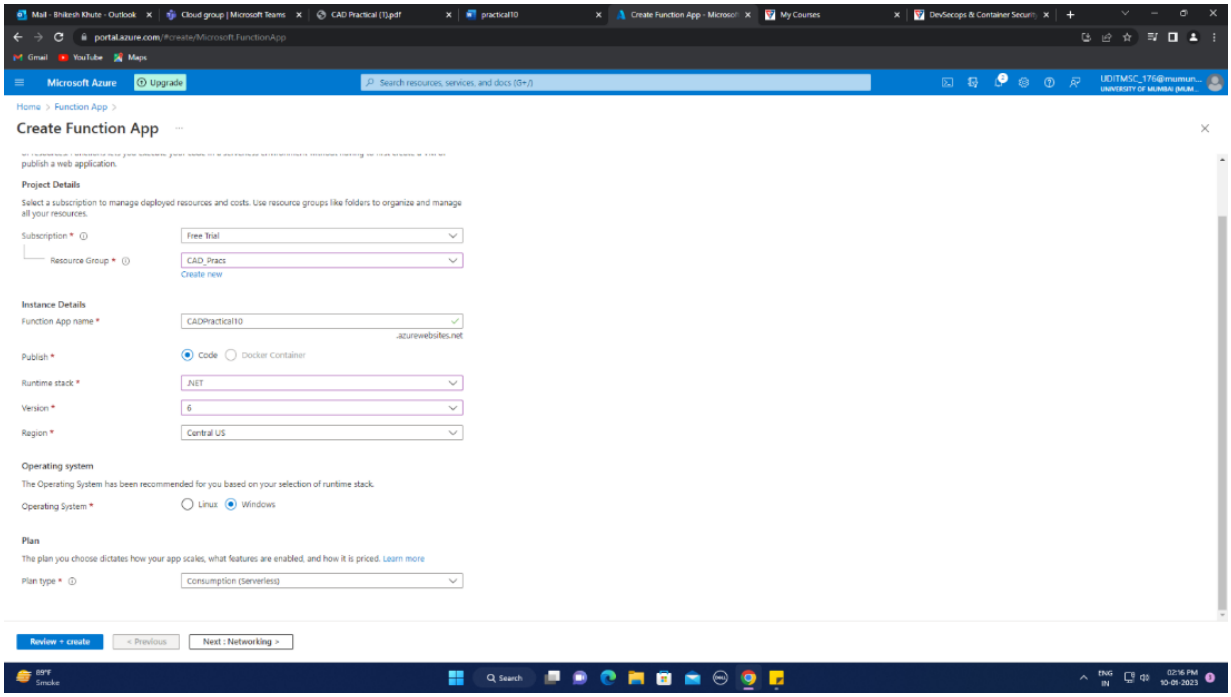
Resource Group Name: CAD10_Practical

Function App Name: CAD10_Practical

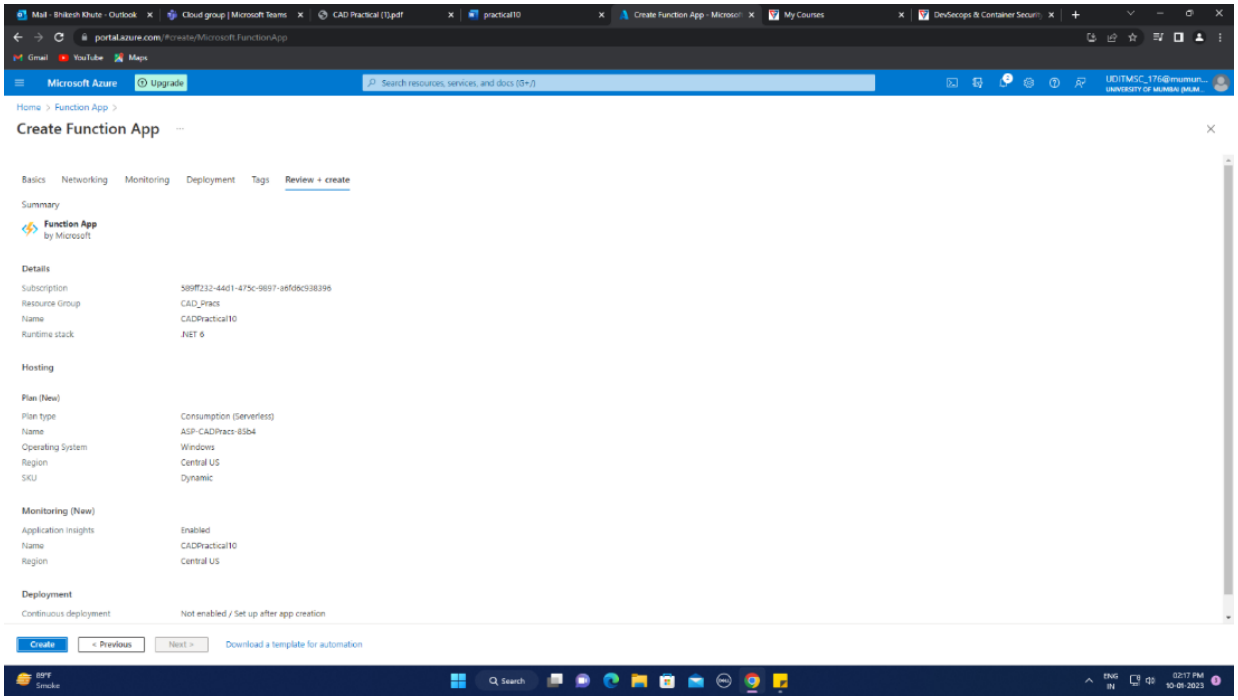
Runtime stack: .NET Core

Version 6

Region: Central US

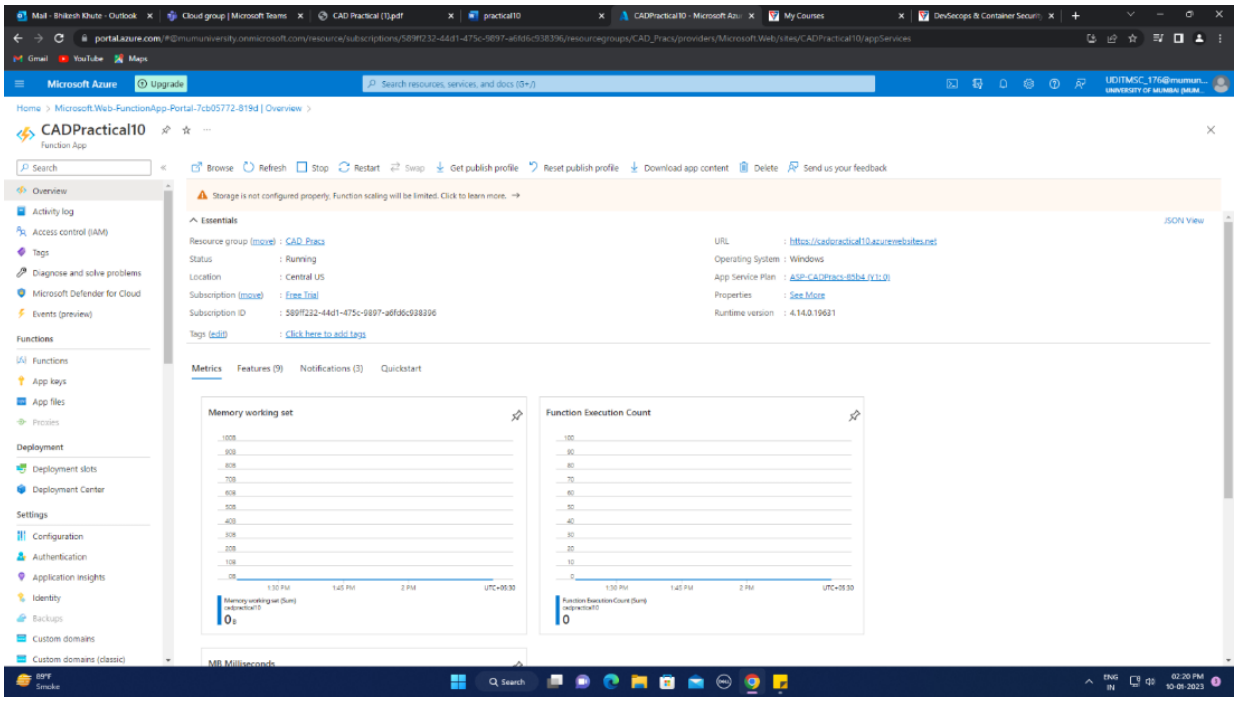


Step 4 - Click on Review and Create:

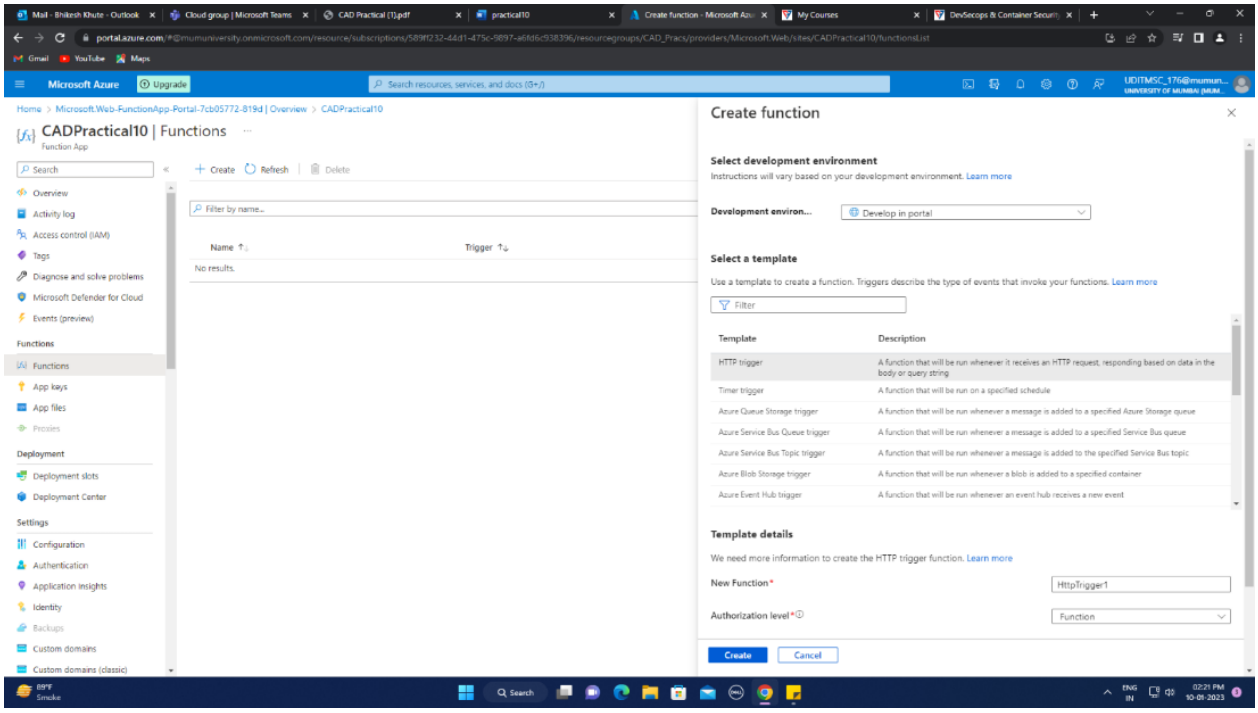


Deployment will automatically start, please wait for it.

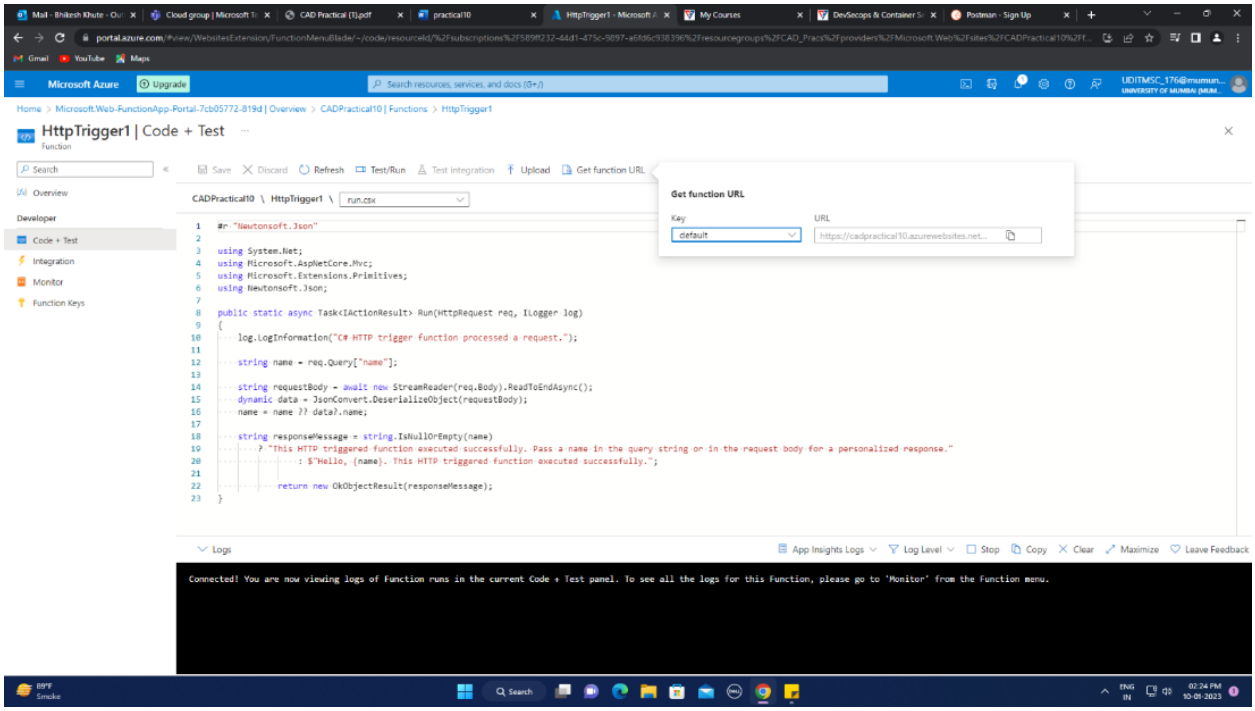
Step 5 – Function is now created. Now, Click on Functions →Add



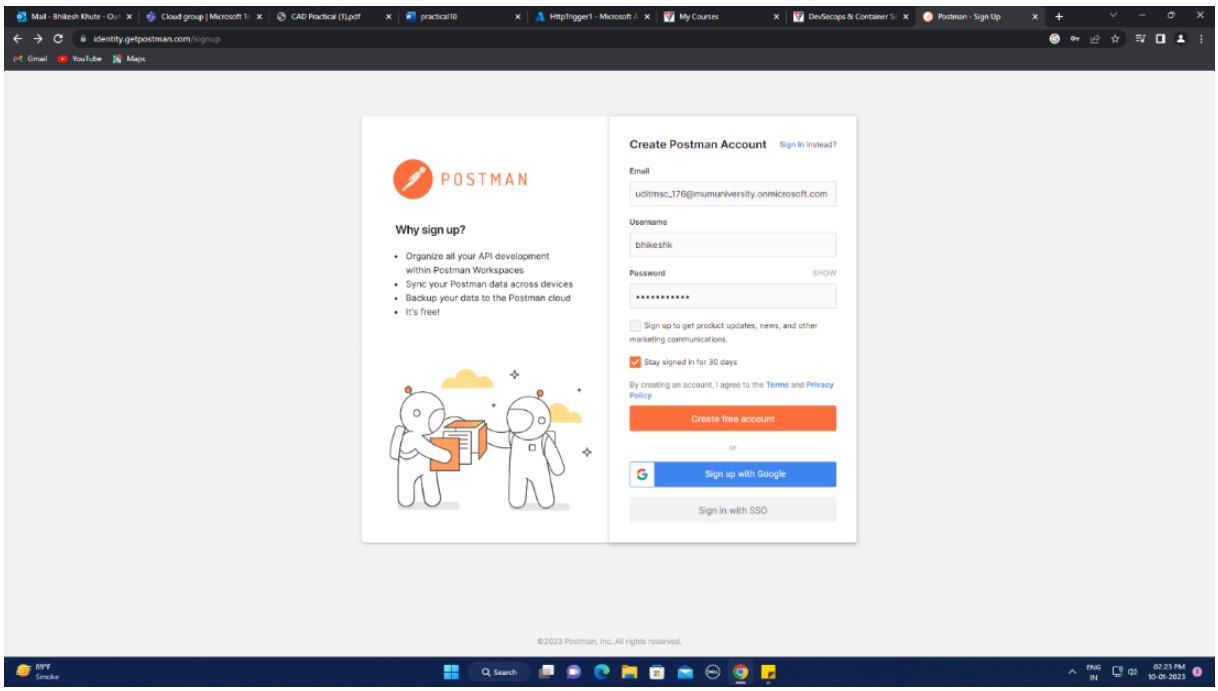
Step 6 - Click on HTTP Trigger and then click on Add & create the same.



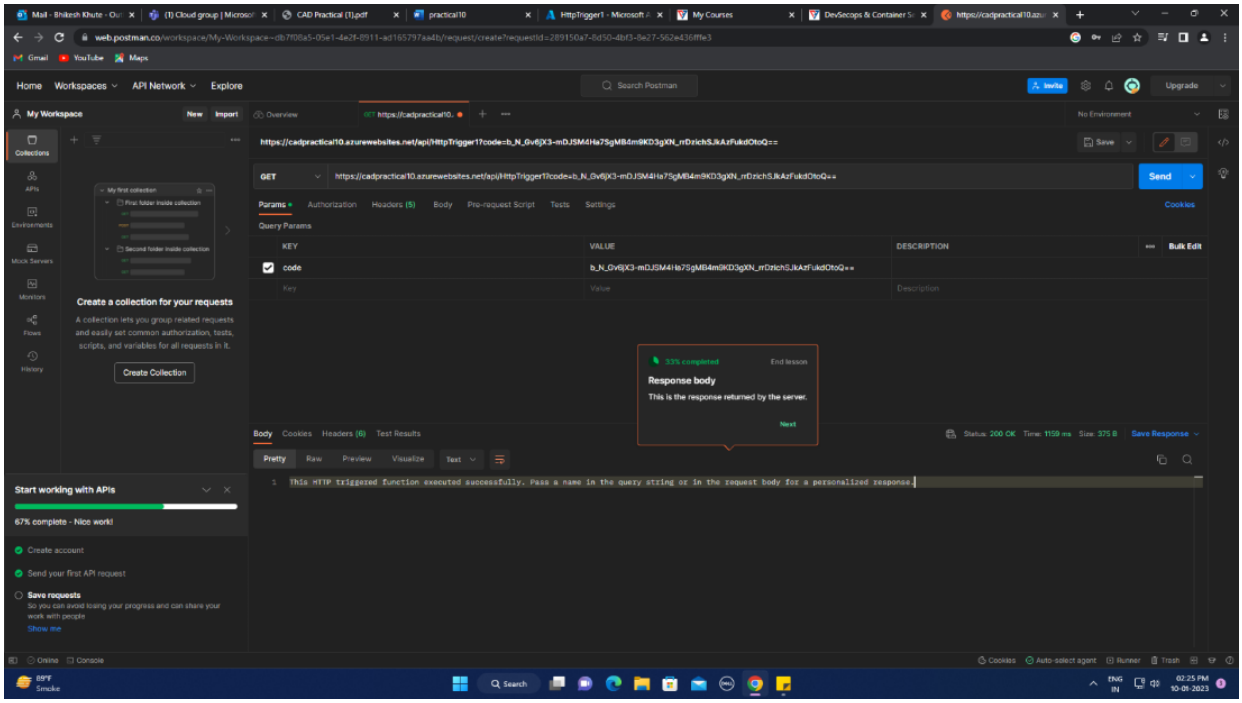
Step 7 – The Http Trigger is now created & the URL too.



Step 8 – Signup on the postman using any SSO to test the http trigger.



Step 9 – Select HTTP Trigger & Click on GET function. Once the GET function is visible, paste the function URL to test the same. Now, we can see in the output of the successful hit request.

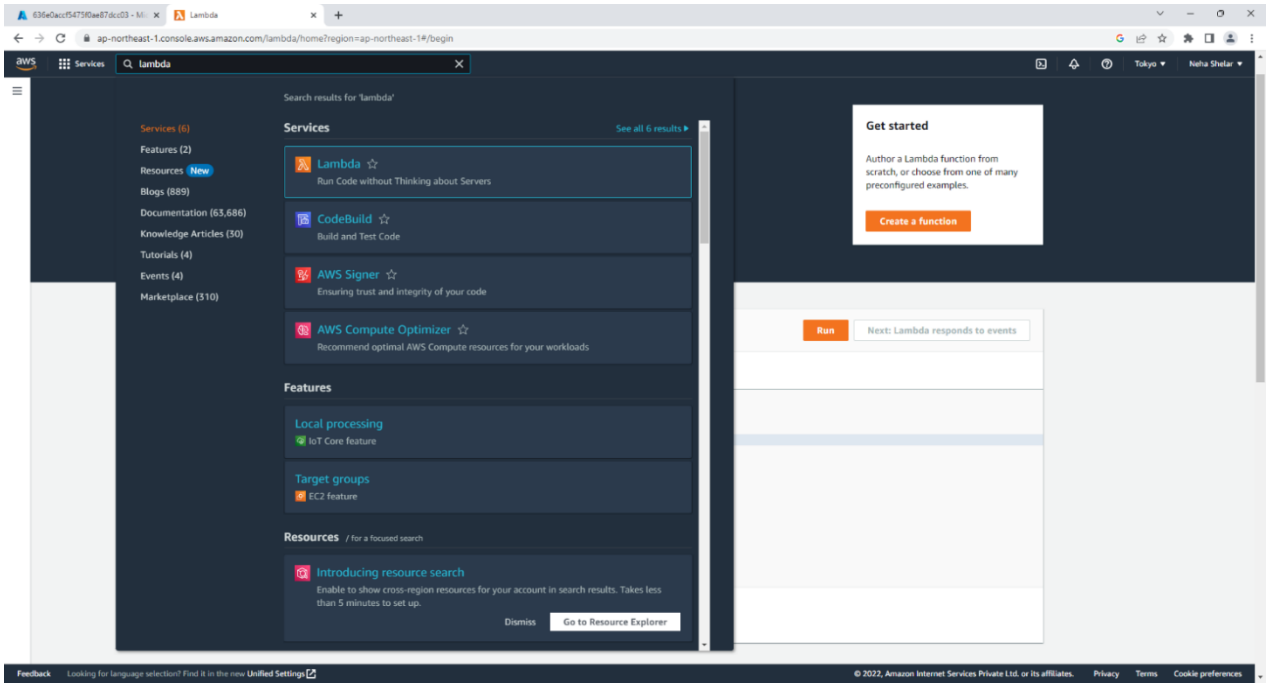


Practical No: 07

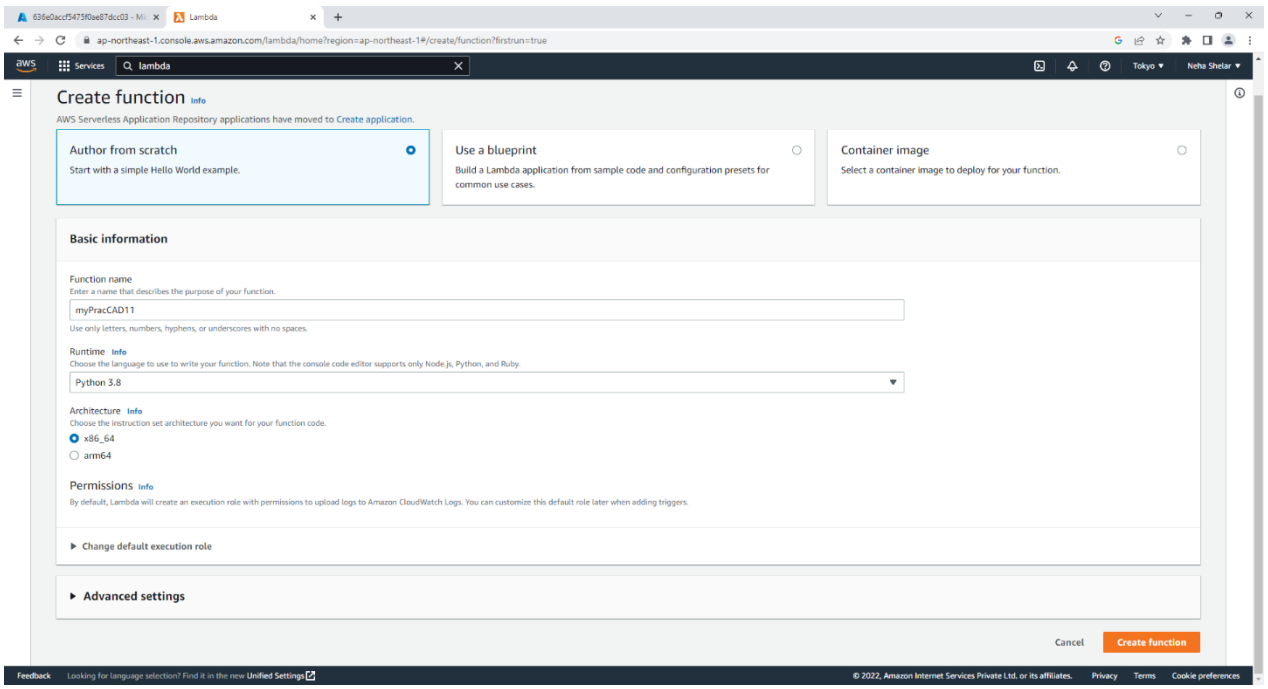
Aim: Create AWS Lambda Function.

To create a Lambda function with the console

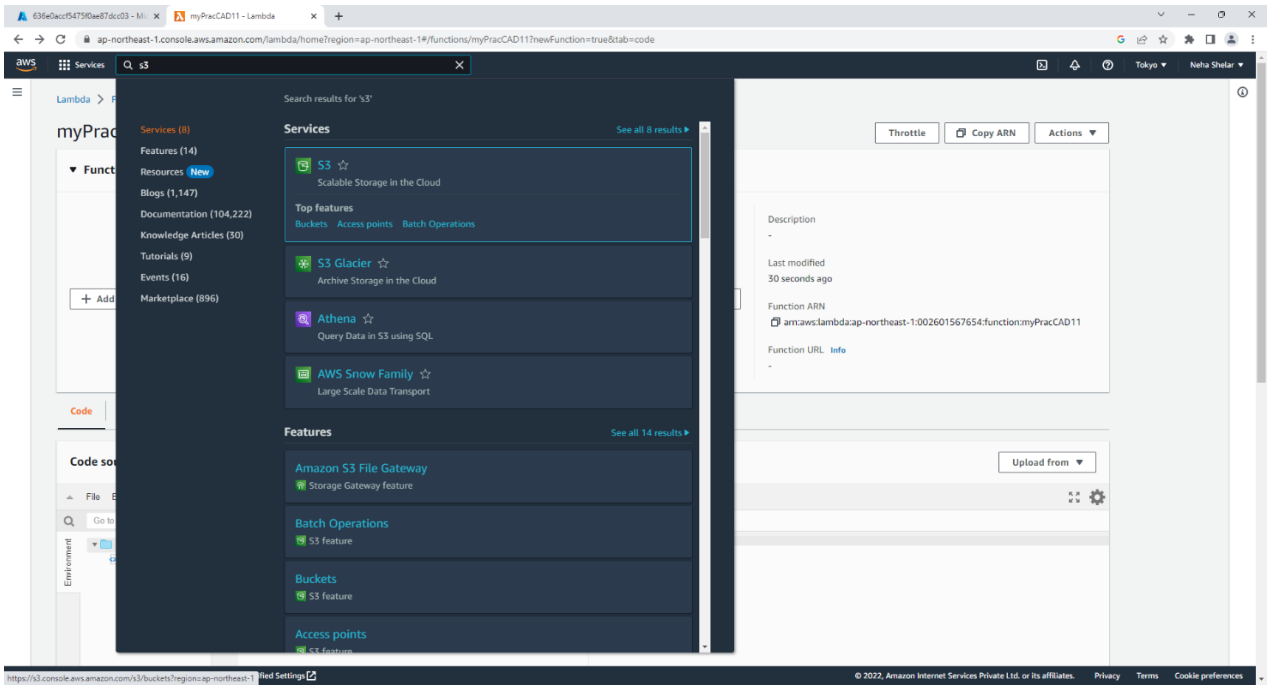
Step 1: Open the Functions page on the Lambda console.

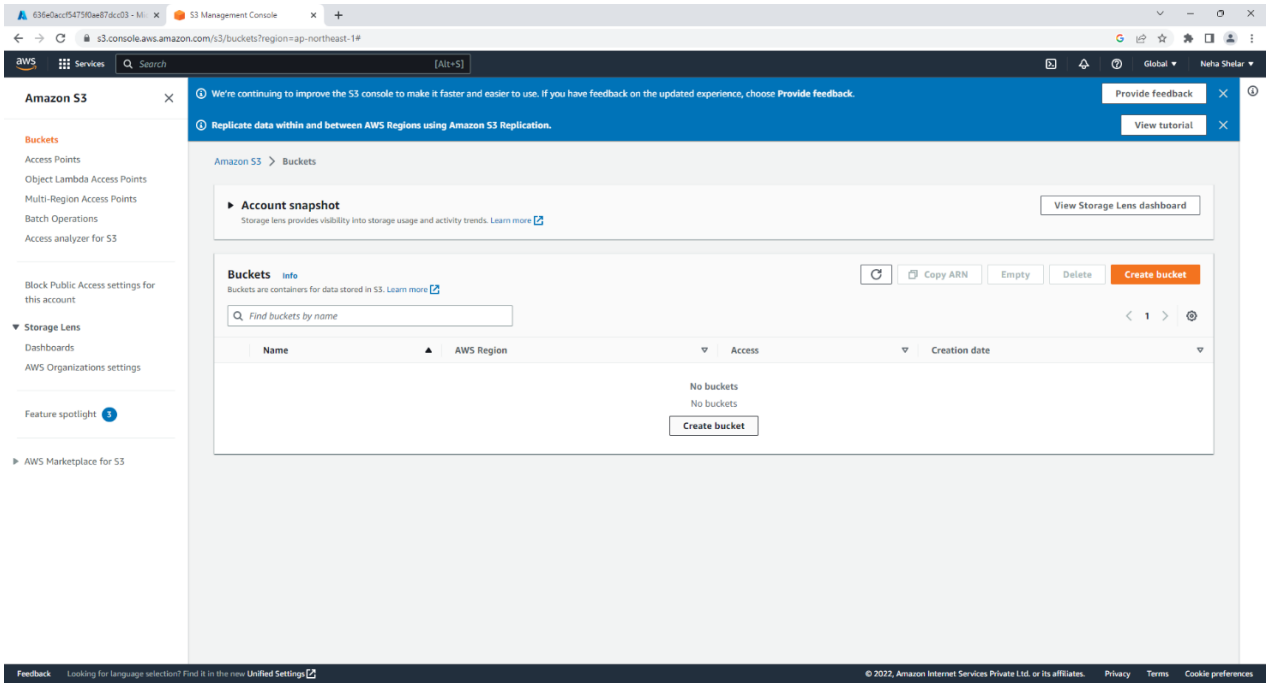


Step 2: Choose Create function.

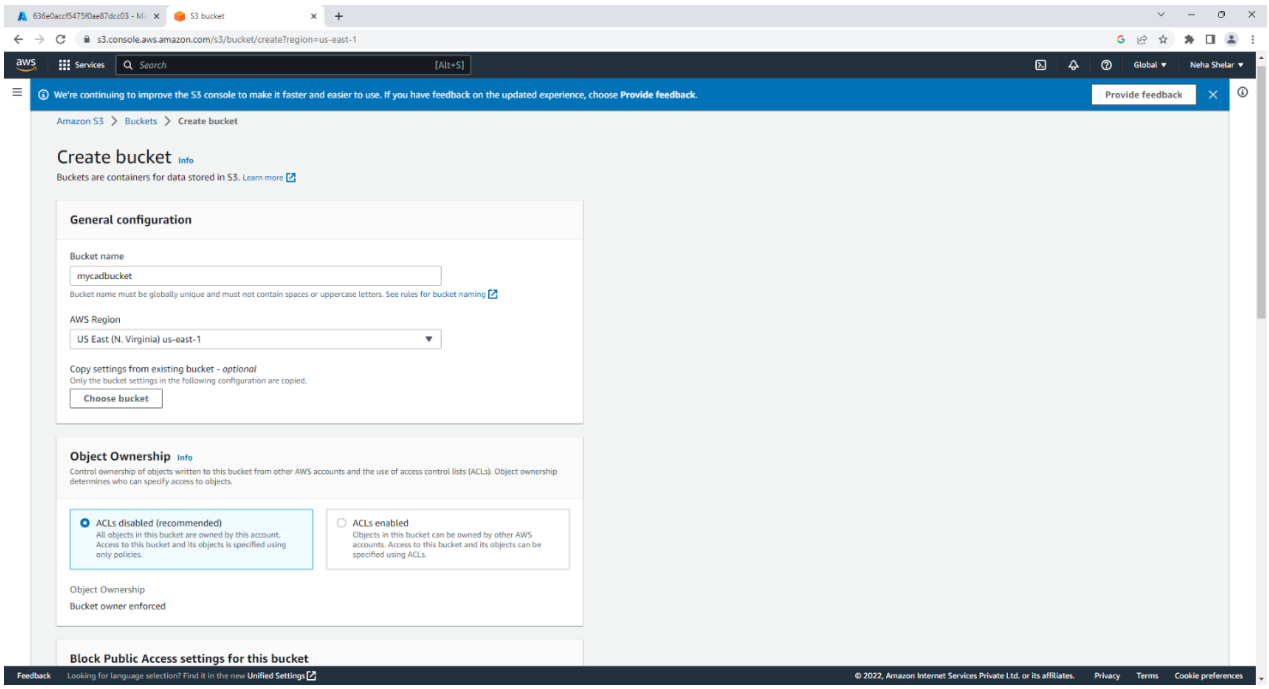


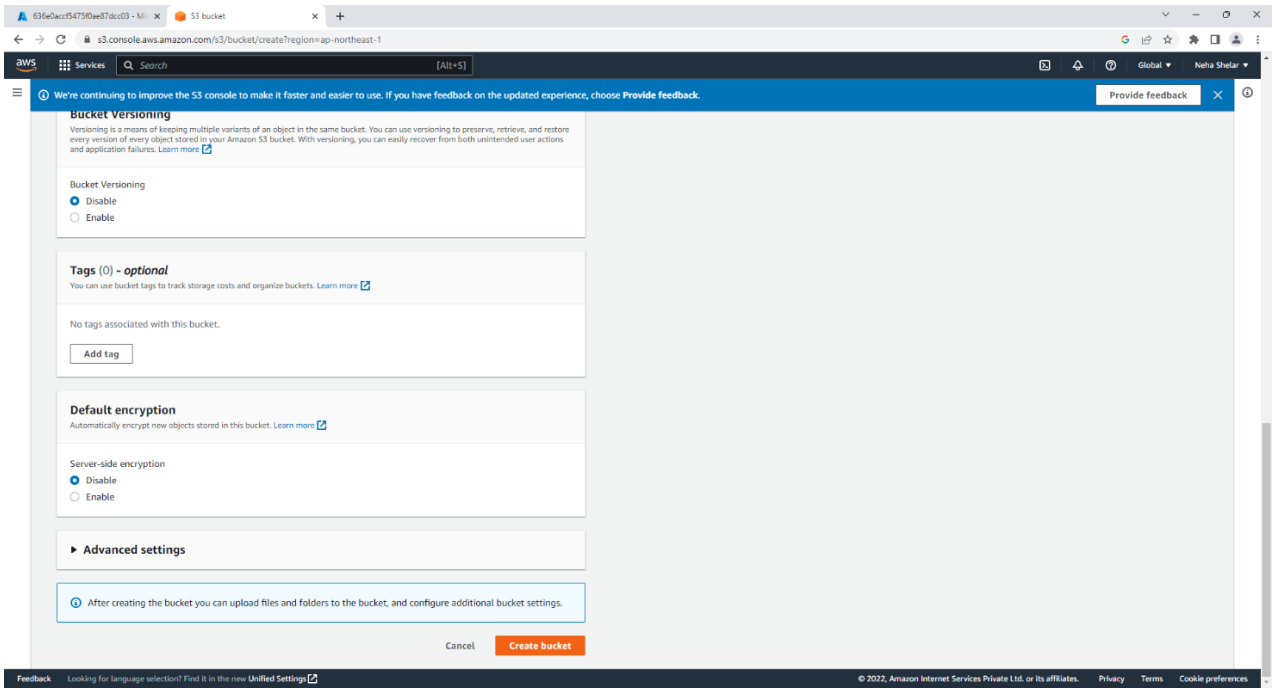
Step 3 - Now search "S3" in the search bar.



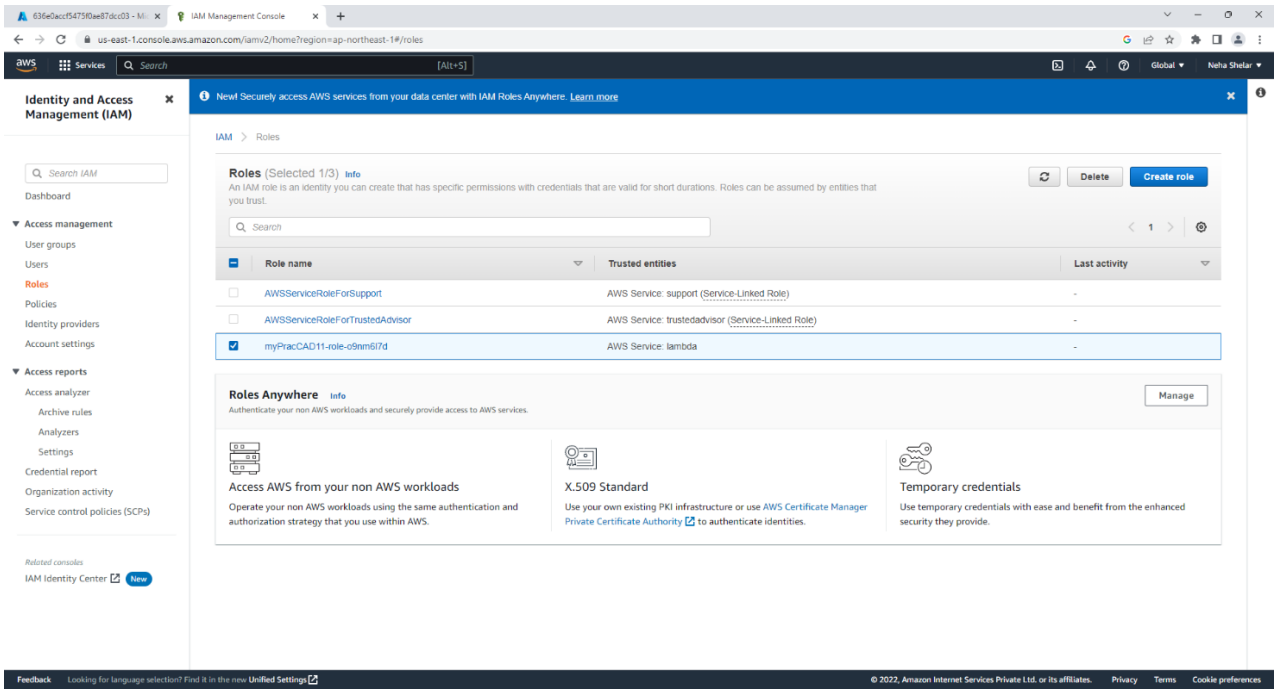


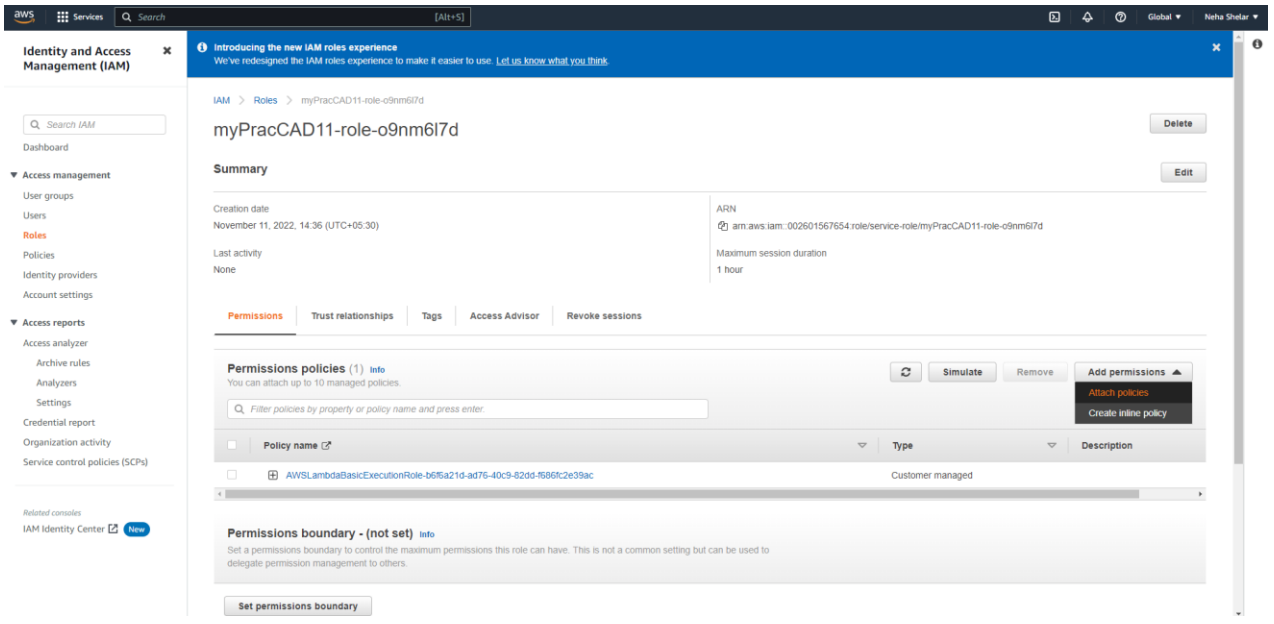
Step 4 – Create a bucket with appropriate bucket name, aws region and encryption.

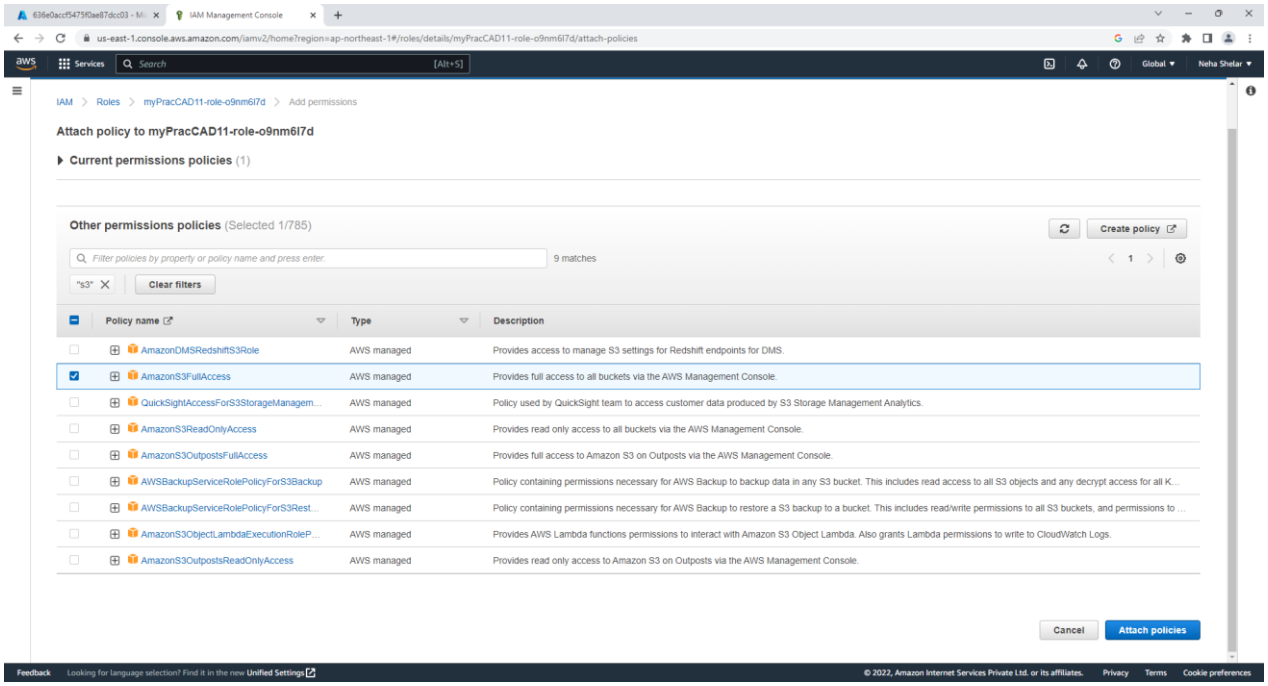


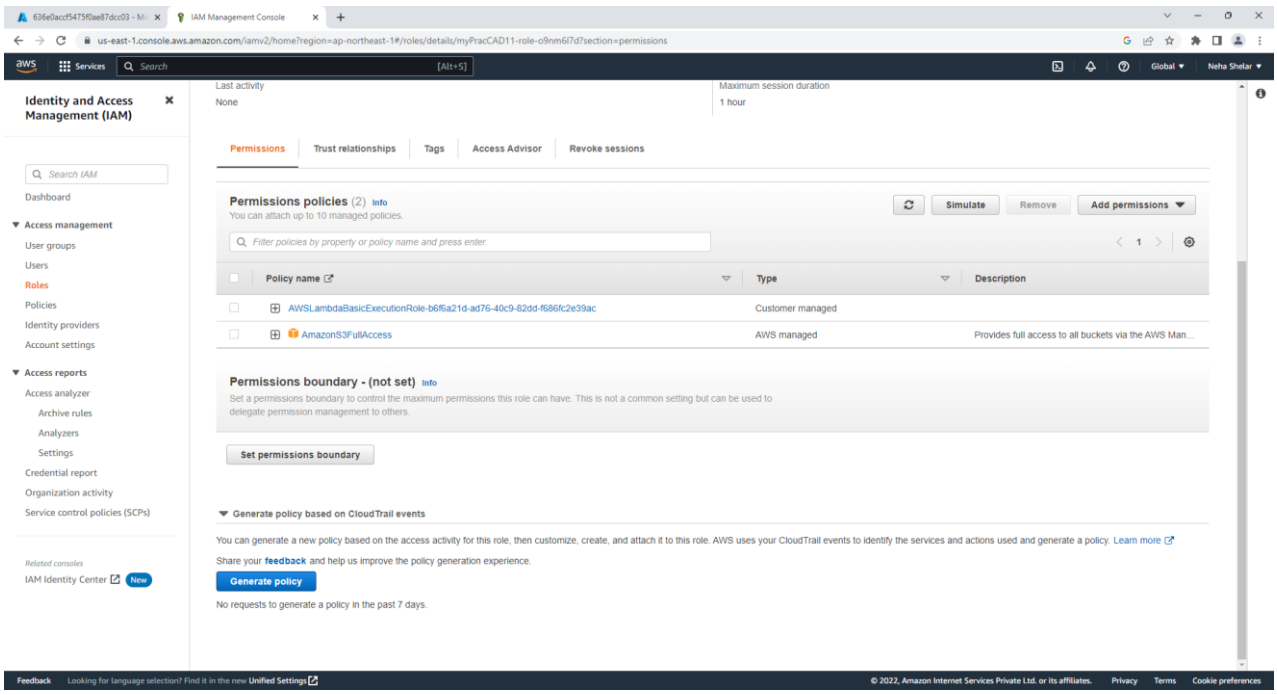


Step 5 - Now go the roles under IAM and select the role and attach policies for s3 bucket access.

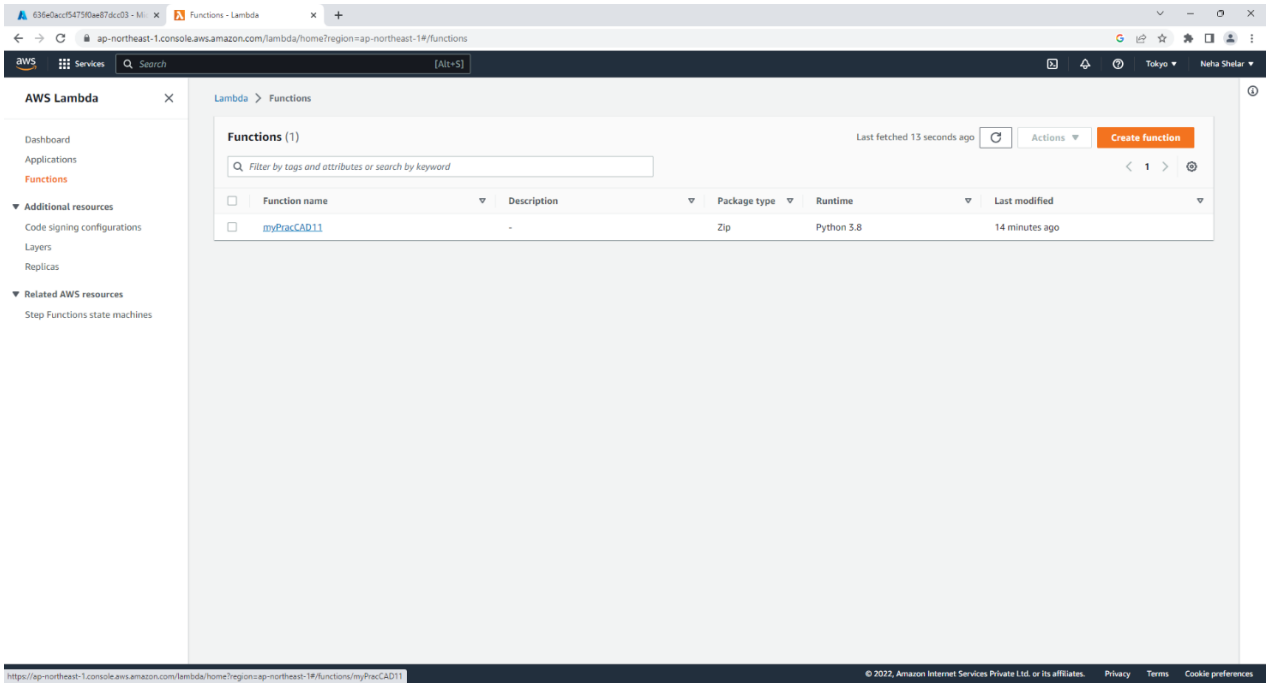




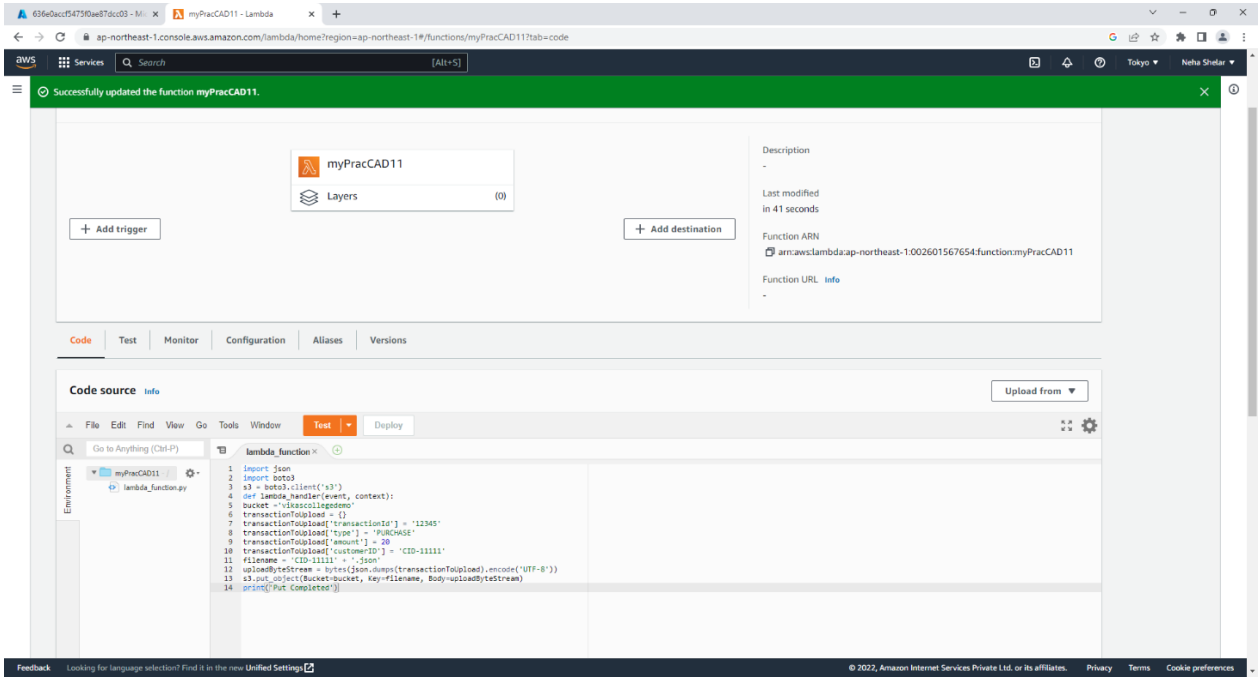


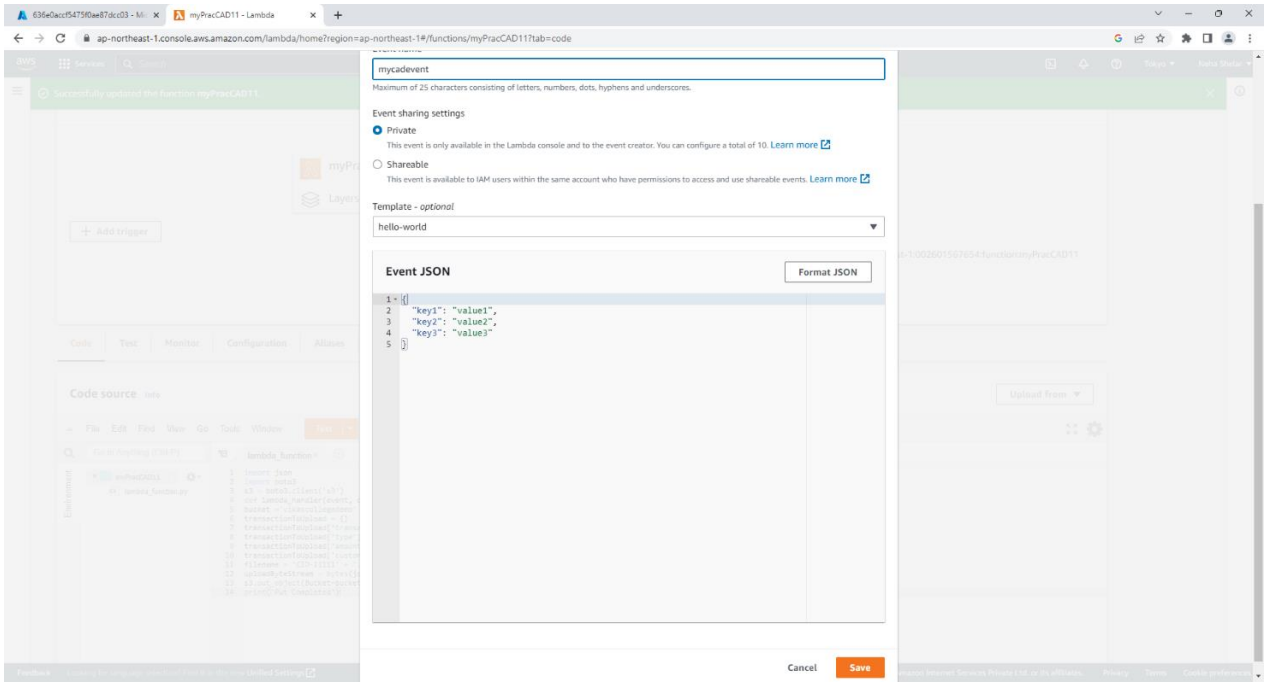


Step 6 - Now go search function and create a function now and paste the code below and deploy the same.



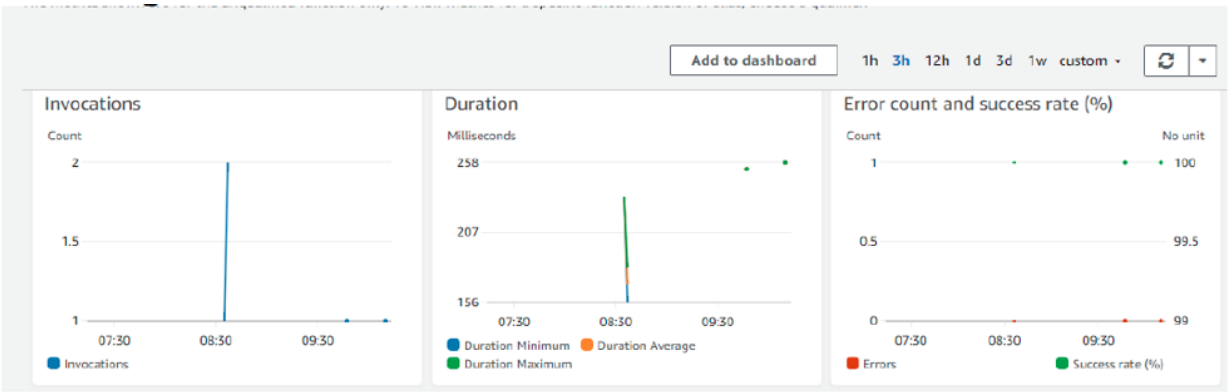

```
import json
import boto3

s3 = boto3.client('s3')
def lambda_handler(event, context):
    bucket='mycadbucket'

    transactionToUpload = {}
    transactionToUpload['transactionId'] = '12345'
    transactionToUpload['type'] = 'PURCHASE'
    transactionToUpload['amount'] = 20
    transactionToUpload['customerID'] = 'CID-11111'
    filename = 'CID-11111' + '.json'
    uploadByteStream = bytes(json.dumps(transactionToUpload).encode('UTF-8'))
    s3.put_object(Bucket=bucket, Key=filename, Body=uploadByteStream)
    print('Put Completed')
```


Successfully updated the function myPracCAD11.
myPracCAD11
Layers
(0)
Add trigger
Add destination
Description
Last modified
in 41 seconds
Function ARN
arn:aws:lambda:ap-northeast-1:002601567654:function:myPracCAD11
Function URL Info
Code
Test
Monitor
Configuration
Aliases
Versions
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
lambda_function
import json
import boto3
s3 = boto3.client('s3')
def lambda_handler(event, context):
bucket ='vlaxcollegedemo'
transactionToUpload = {}
transactionToUpload['transactionId'] = '12345'
transactionToUpload['type'] = 'PURCHASE'
transactionToUpload['amount'] = 20
transactionToUpload['customerID'] = 'CID-11111'
filename = 'CID-11111' + '.json'
uploadByteStream = bytes(json.dumps(transactionToUpload).encode('UTF-8'))
s3.put_object(Bucket=bucket, Key=filename, Body=uploadByteStream)
print('Put Complete')
Feedback
© 2022, Amazon Internet Services Private Ltd. or its affiliates.
Privacy
Terms
Cookie preferences

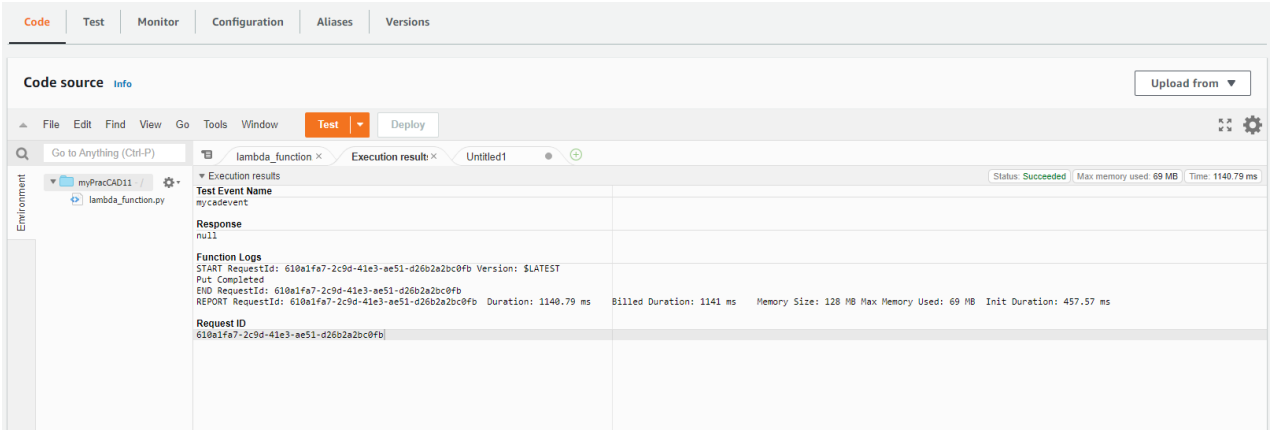

mycadevent
Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.
Event sharing settings
Private
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. Learn more
Shareable
This event is available to IAM users within the same account who have permissions to access and use shareable events. Learn more
Template - optional
hello-world
Event JSON
Format JSON
{
"key1": "value1",
"key2": "value2",
"key3": "value3"
}
Cancel
Save

OUTPUT:

Monitoring Result:



Execution Result:



Practical no. 08

Aim: Build AWS Lambda with AWS API Gateway

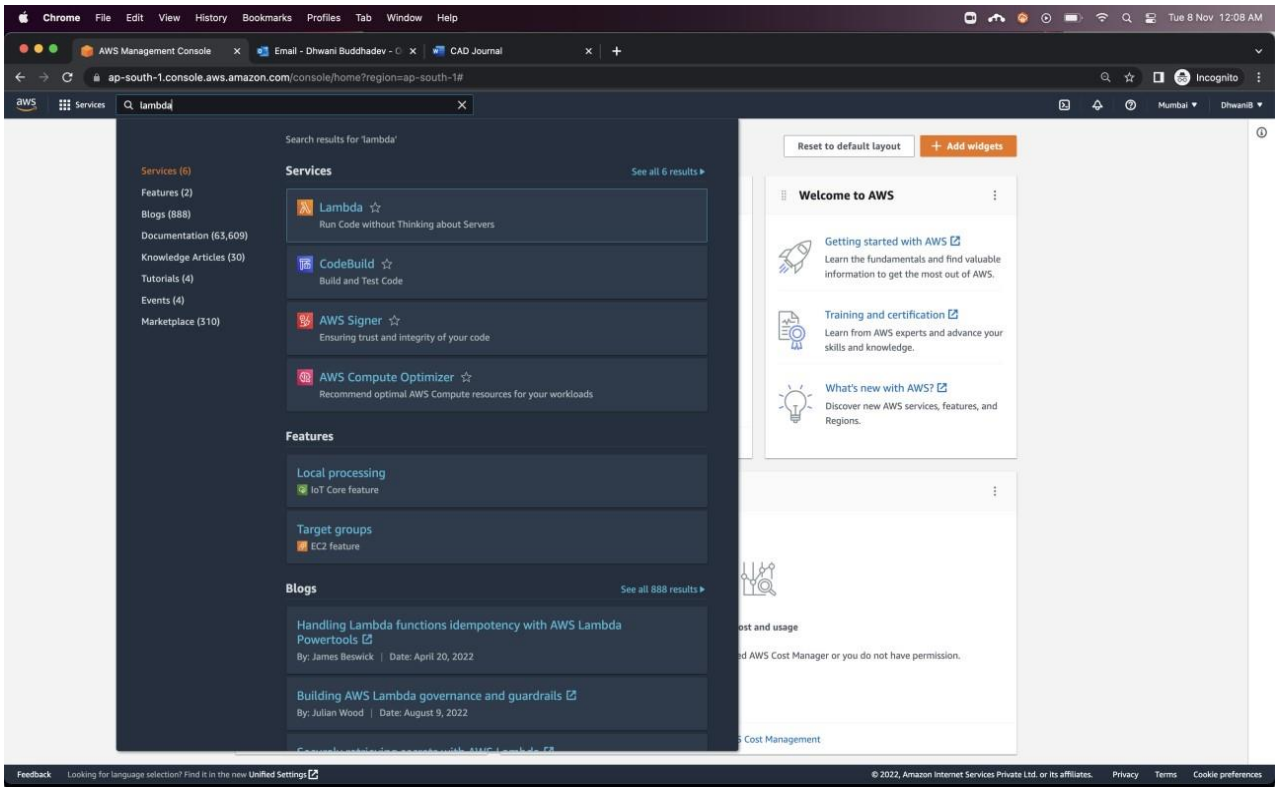
Description of Practical: -

- In this practical we will learn to build AWS Lambda function using API Gateway.
- In order to perform this practical we will take an example.
- The example is EMI Calculator.

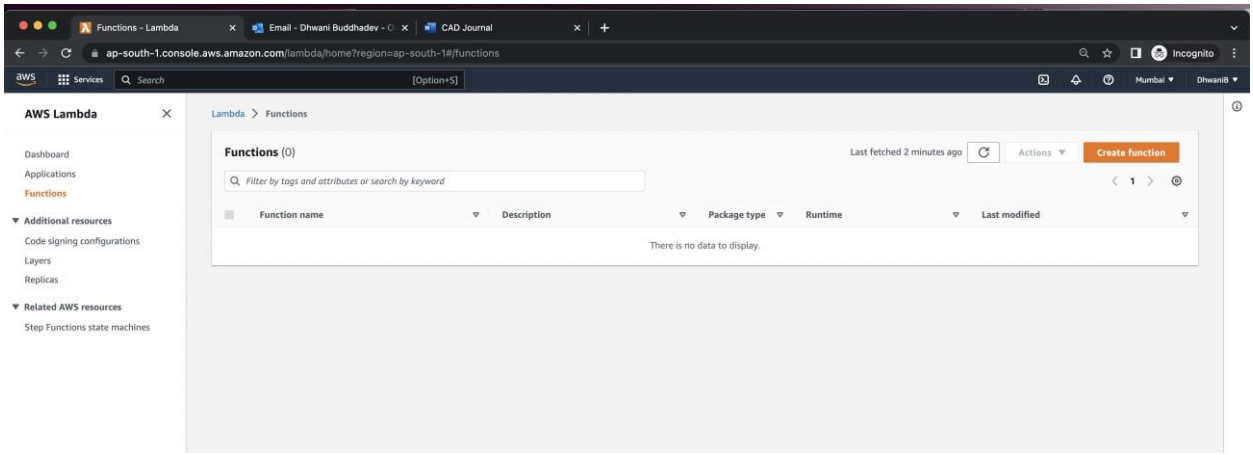
EMI Calculator

Steps

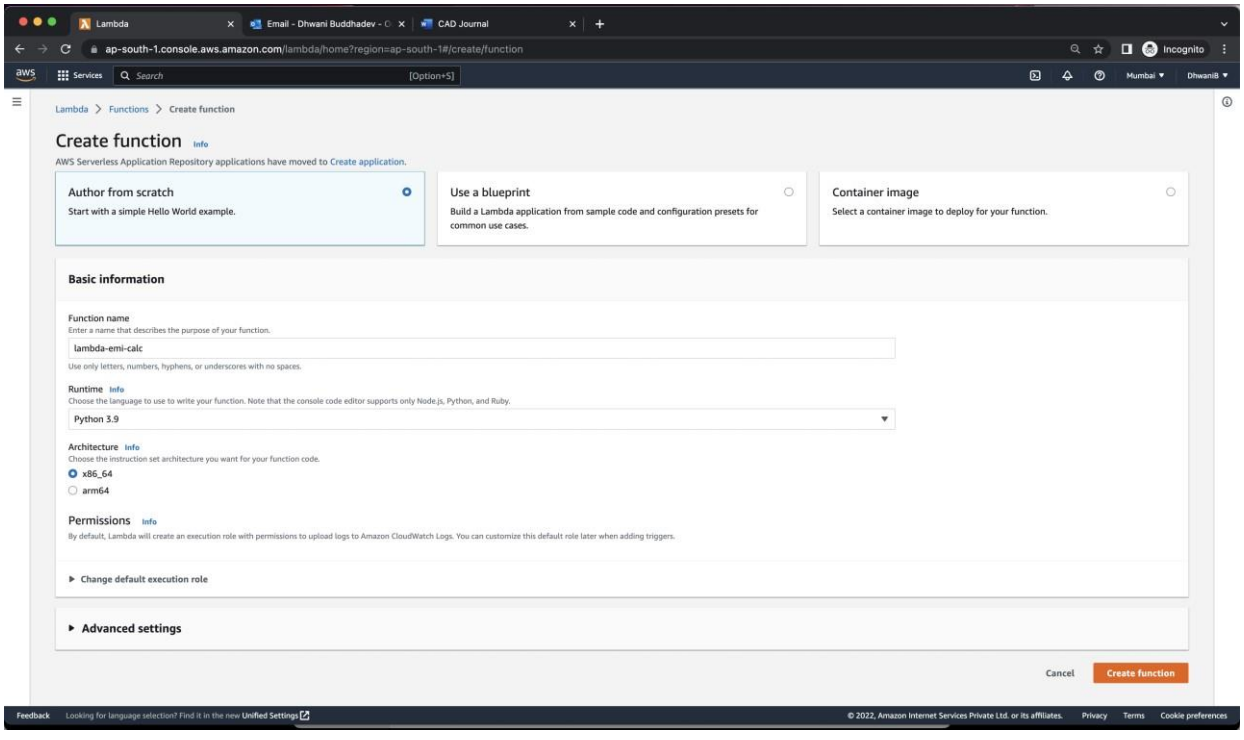
- Login into AWS Console.
- Search for Lambda on the service bar.



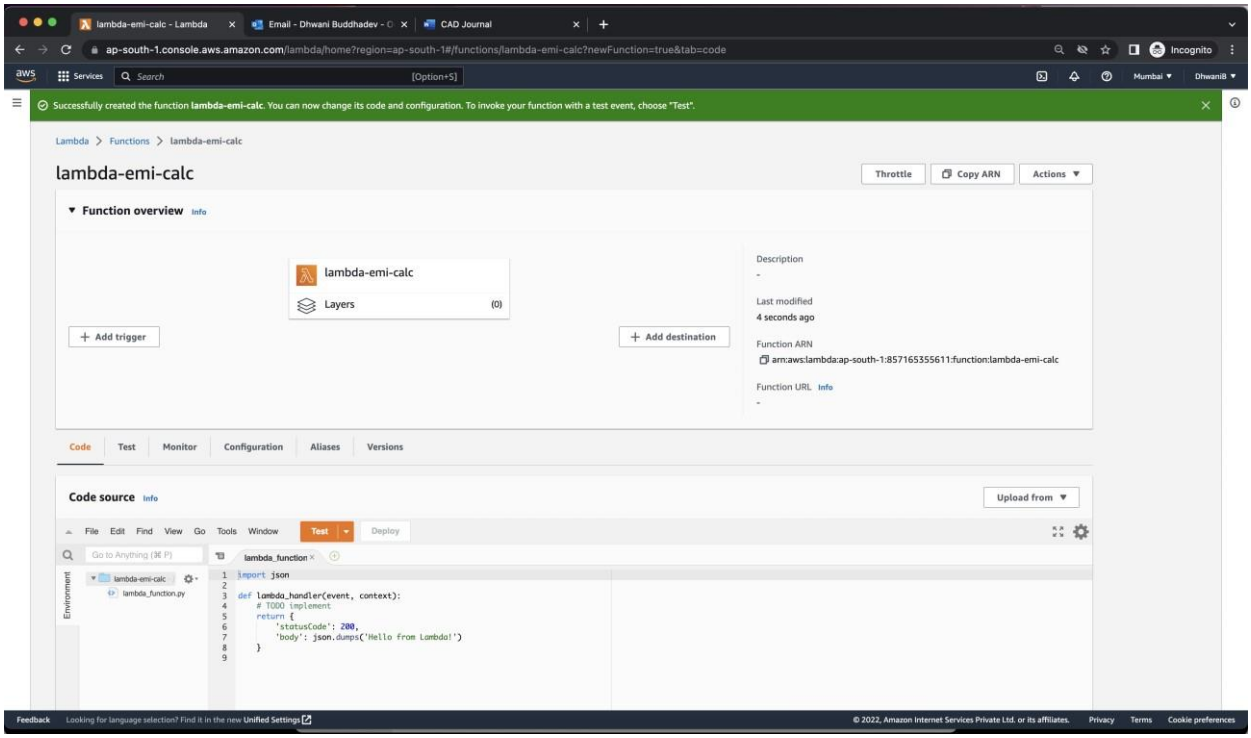
- Here we will create a lambda function for the practical.



- Click on Create Function on the Lambda Tab.



- Here we will choose Author from Scratch.
- After selecting that write the function name as lambda-emi-calc
- We will be running the code on python 3.9 • So, we will choose runtime as python 3.9
- After that click on create function.
- We can see that it has created lambda function with basic default code



- Next step is to clear existing code and write emi calculator python code as below :

import json print('Loading function... Lambda emi calculator using API gateway') def lambda_handler(event, context): #1. Parse out query string params principal = event['queryStringParameters']['p'] rate = event['queryStringParameters']['r'] time = event['queryStringParameters']['t']

principal = int(principal) rate = float(rate) time = int(time)

emi = emi_calculator(principal, rate, time);

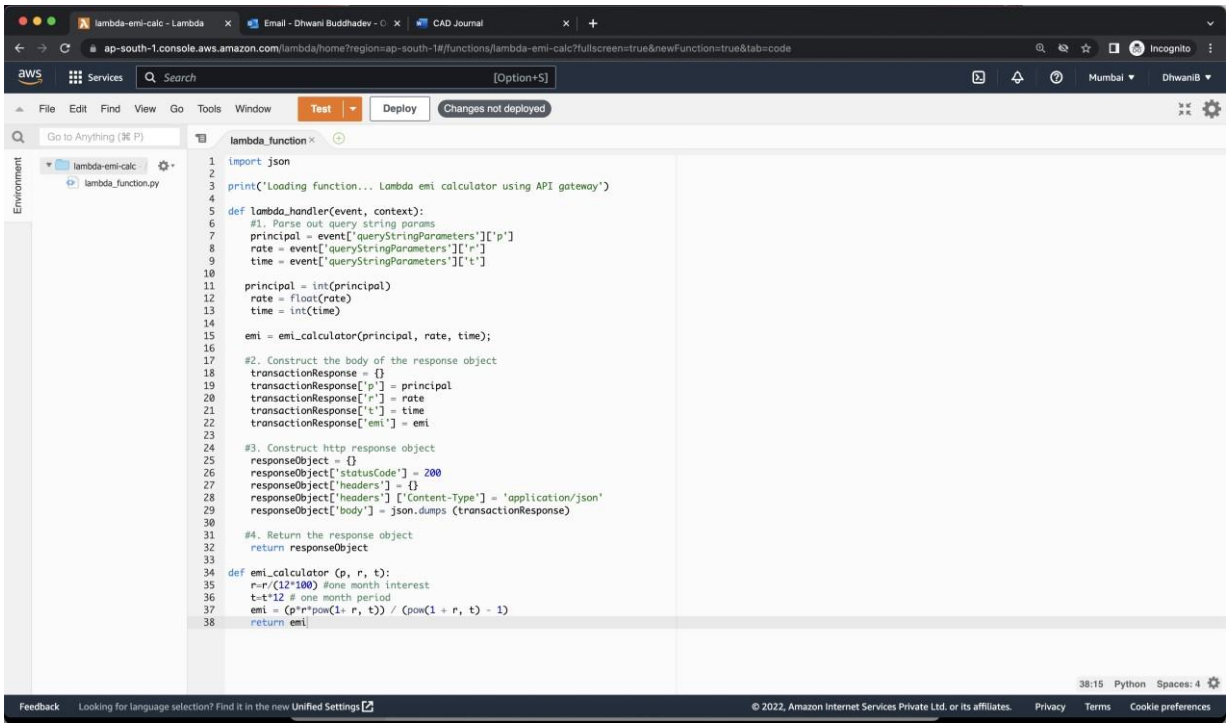
#2. Construct the body of the response object transactionResponse = {} transactionResponse['p'] = principal transactionResponse['r'] = rate transactionResponse['t'] = time transactionResponse['emi'] = emi

---

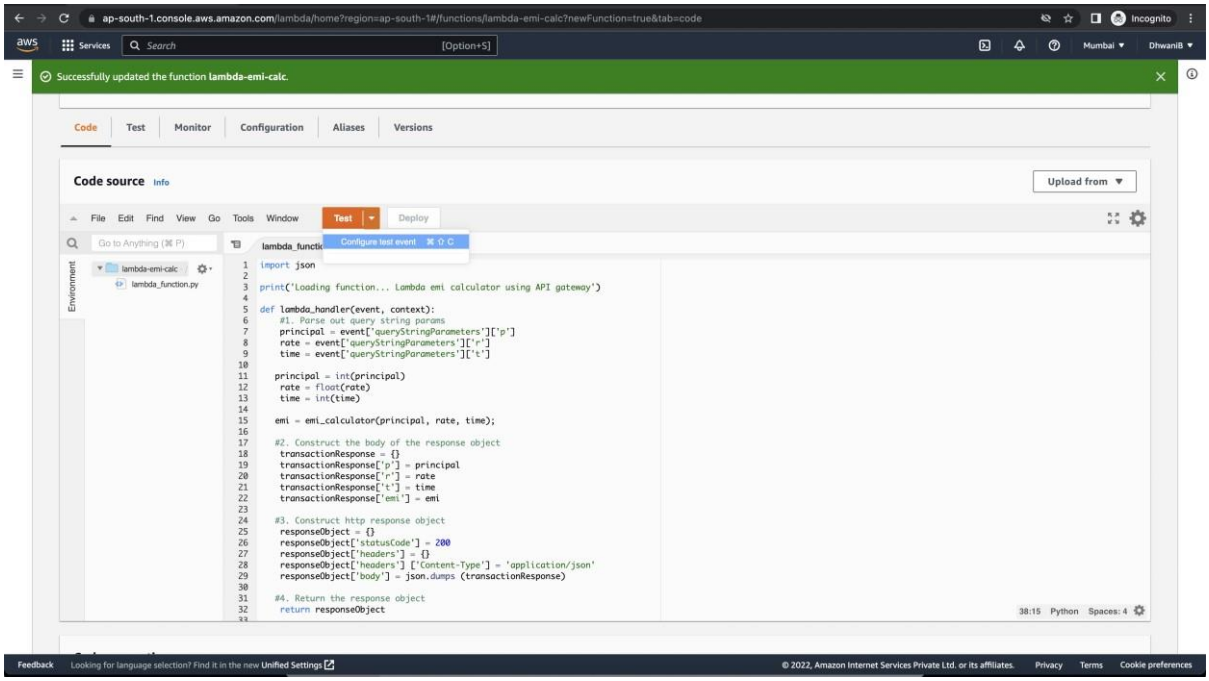
#3. Construct http response object responseObject = {} responseObject['statusCode'] = 200 responseObject['headers'] = {}

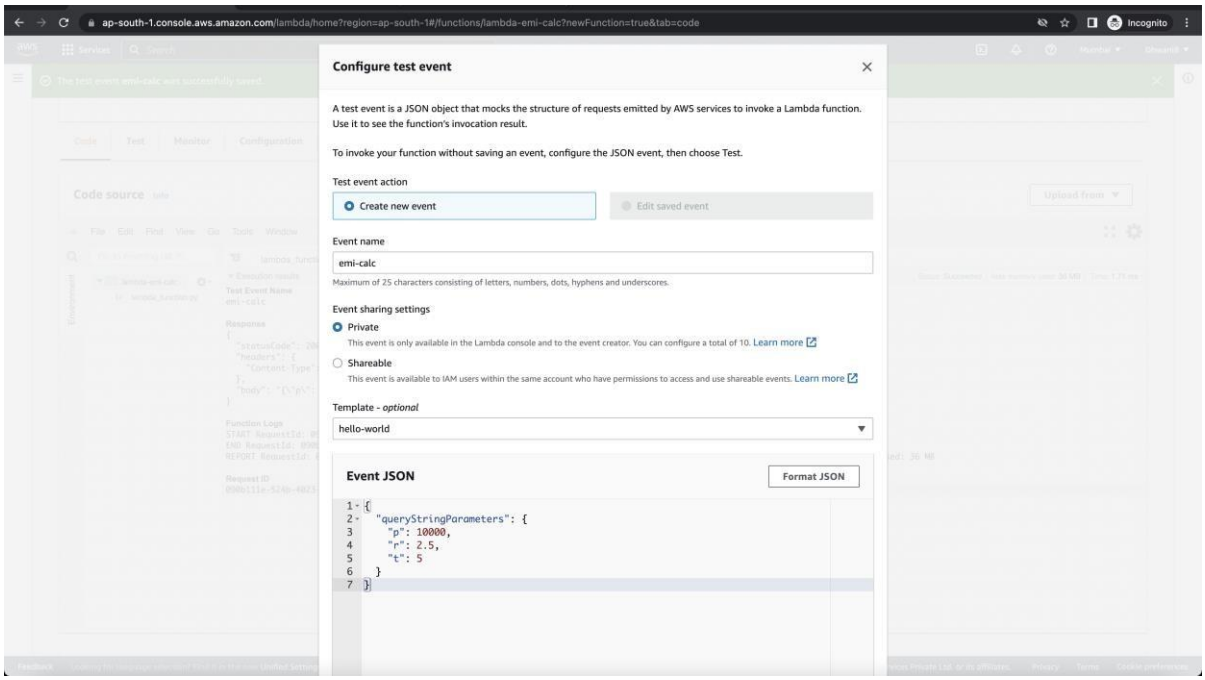
*responseObject['headers'] ['Content-Type'] = 'application/json' responseObject['body'] = json.dumps (transactionResponse) #4. Return the response object return responseObject def emi_calculator (p, r, t): r=r/(12*100) #one month interest t=t*12 # one month period*

*emi = (p*r*pow(1+ r, t)) / (pow(1 + r, t) - 1) return emi*



- Deploy the lambda function
- Then go to dropdown besides test and configure test event with sample input values for testing lambda function code



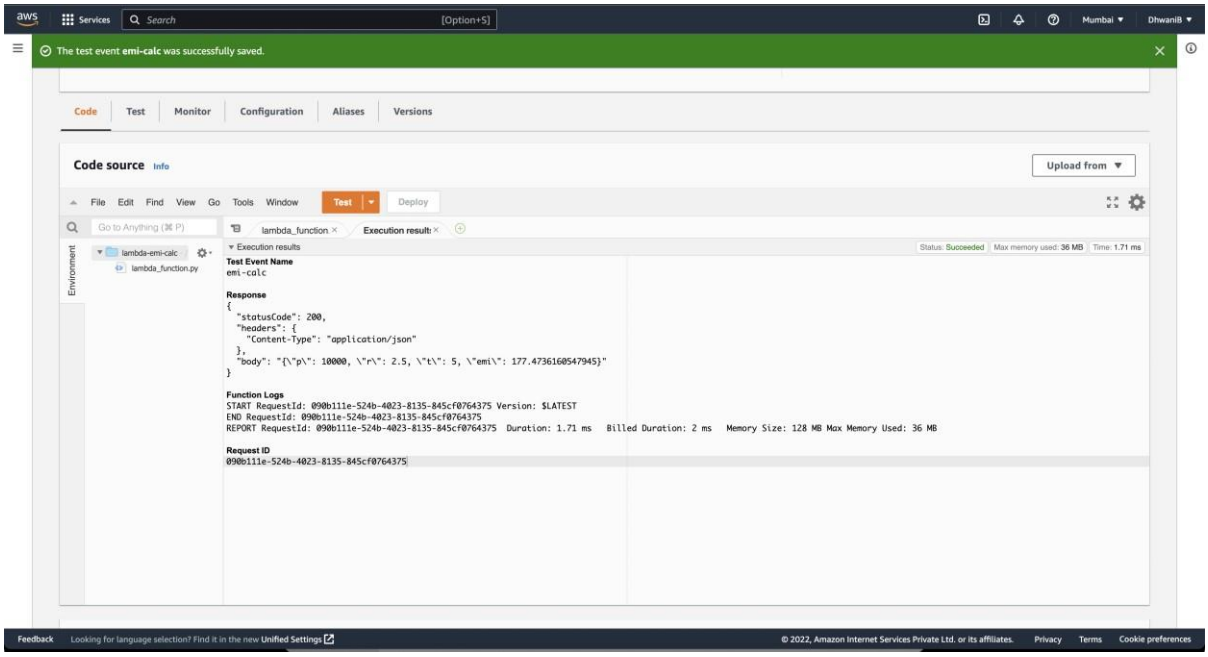


{

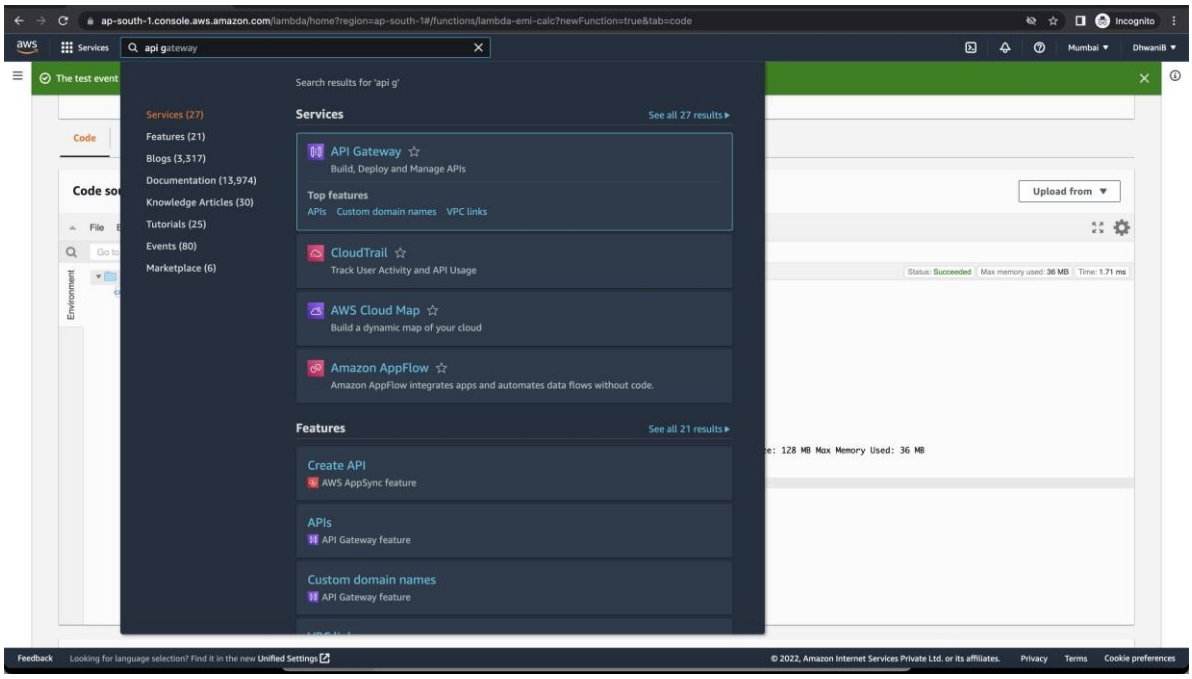
"queryStringParameters": {
"p": 10000,
"r": 2.5,
"t": 5

}
}

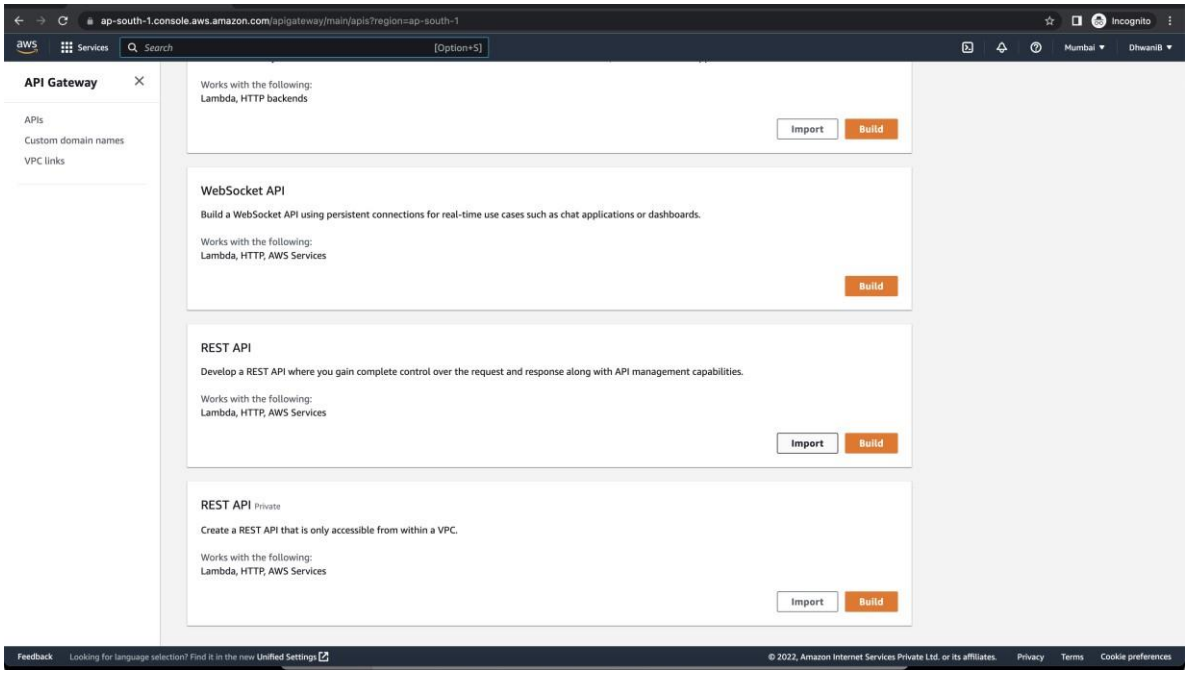
- Test the code and get output without any errors



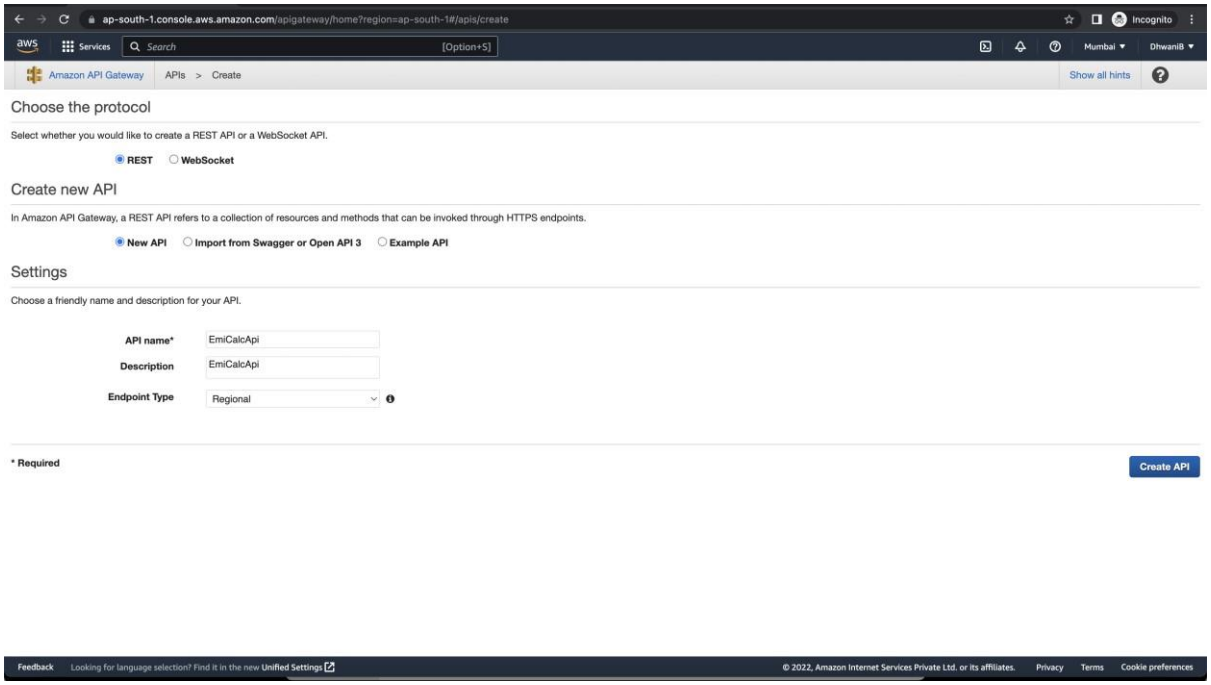
- Go above and search for API gateway on the service bar.



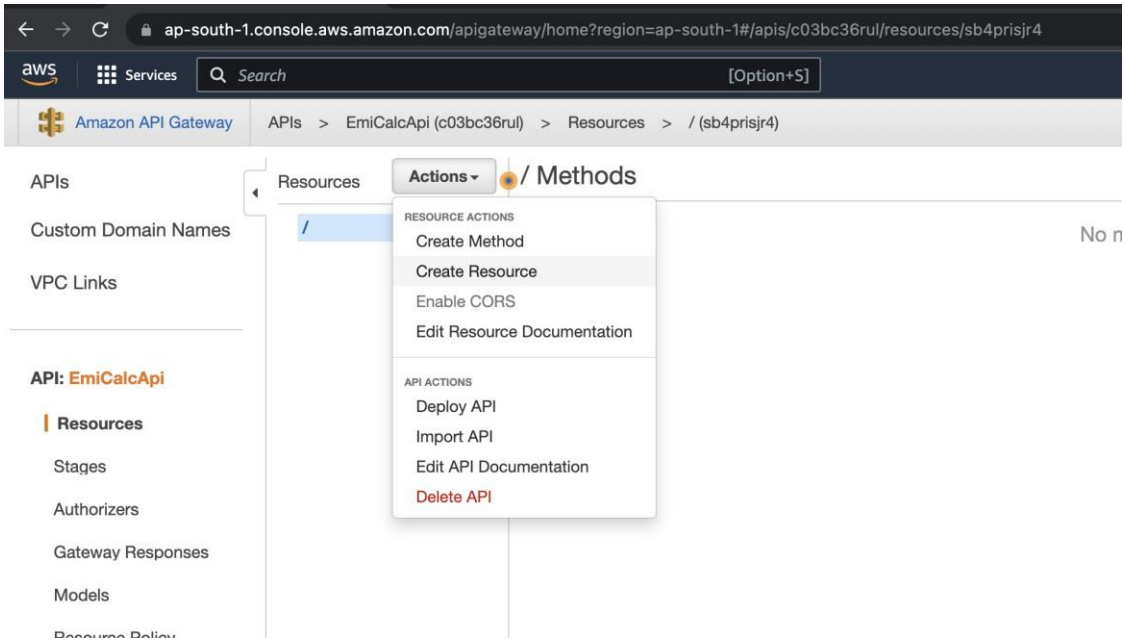
- Then create API gateway
- Choose Rest API and click import



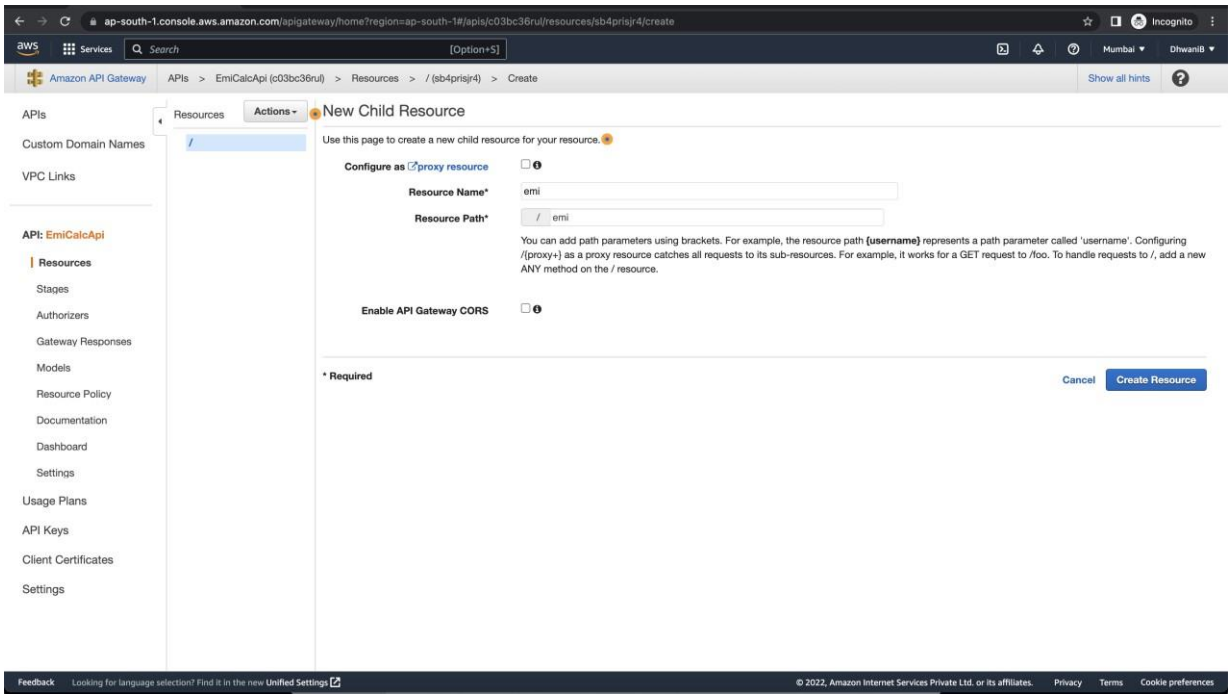
- Create new API
- API name = EmiCalcApi
- Description = EmiCalcApi
- Endpoint type = Regional
- Click on create tab



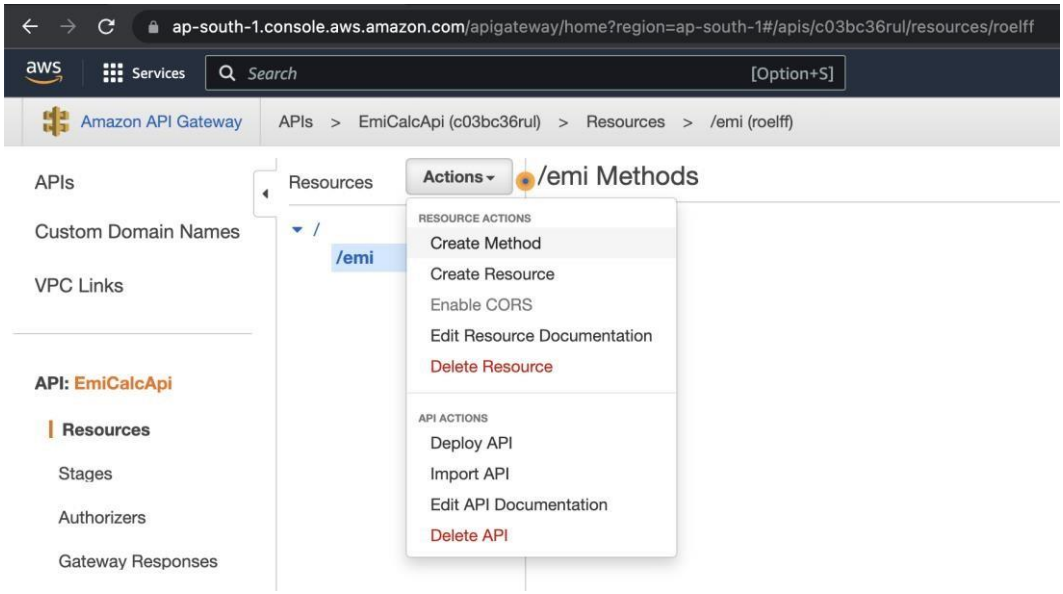
- Next step click on actions then create resource



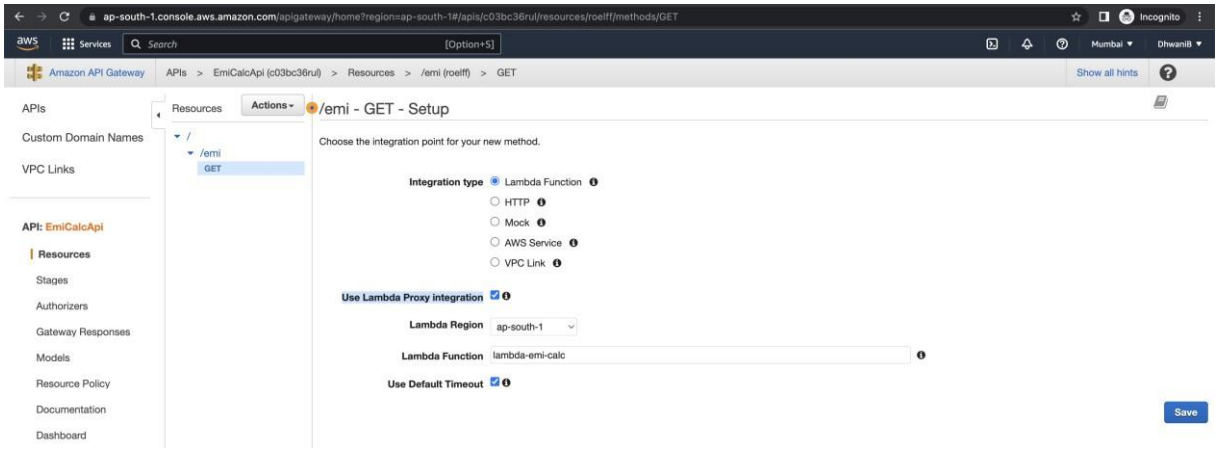
- Enter resource name as emi and click create resource

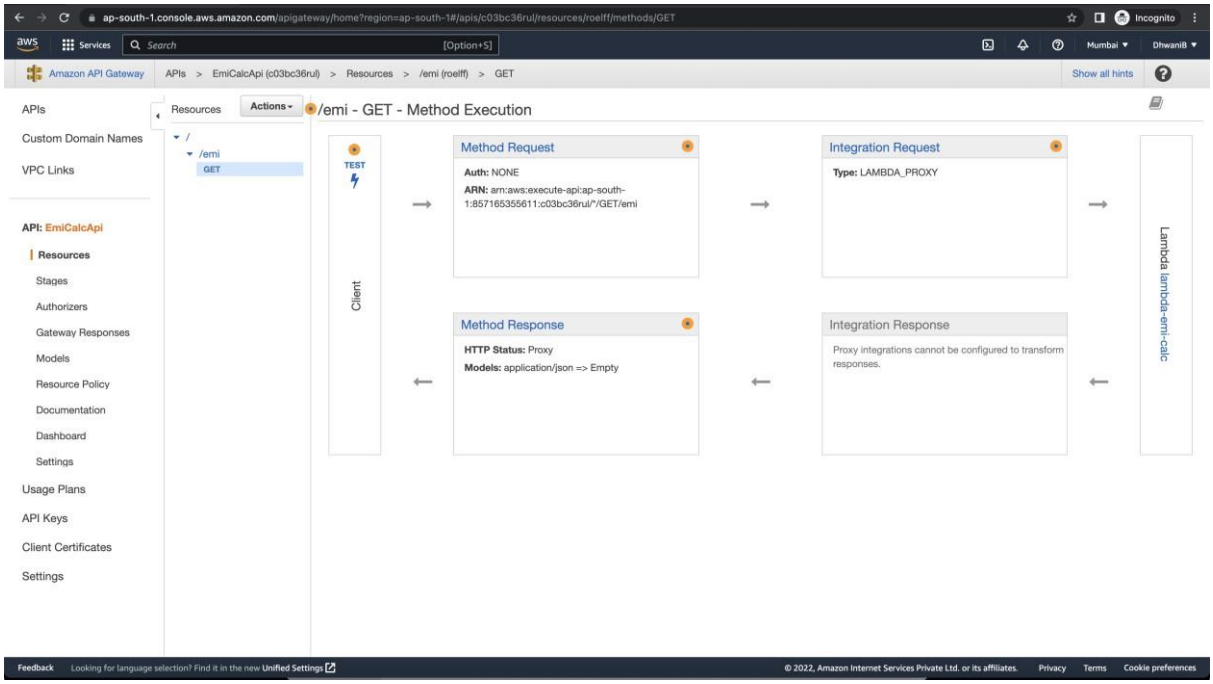


- Now select on actions tab and create method

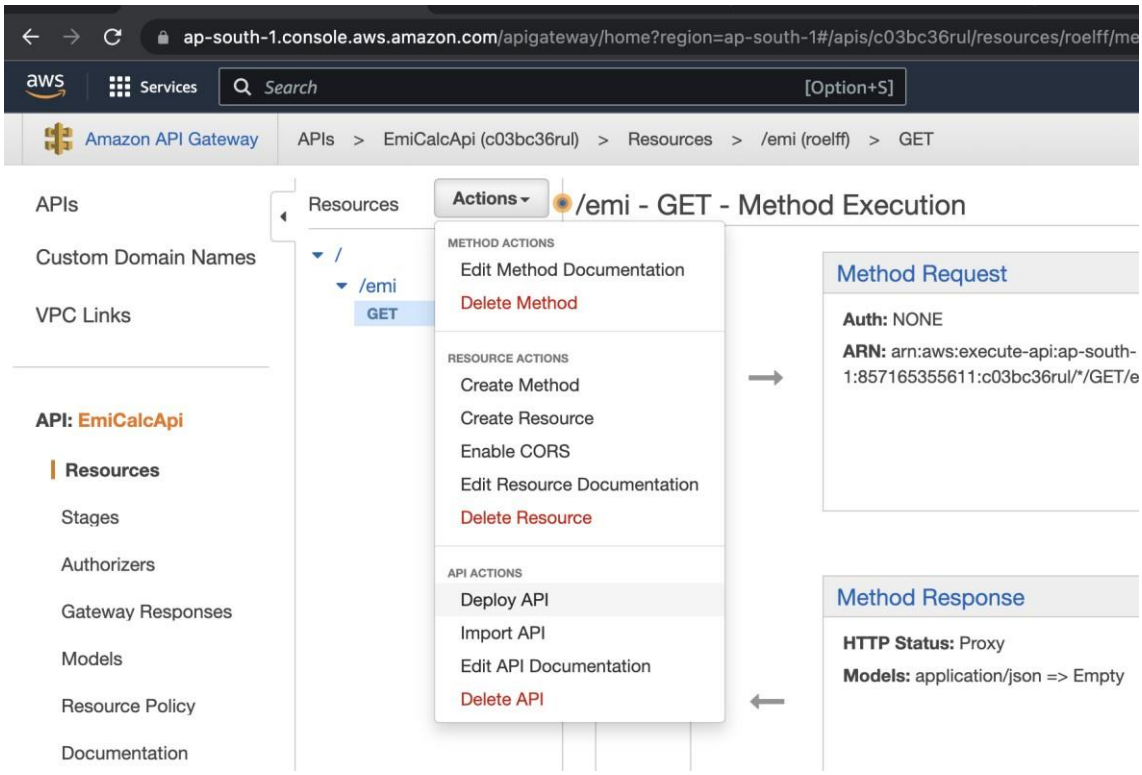


- Use Get parameter
- We are using integration type as Lambda Function
- Enter lambda function same as created above **lambda-emi-calc**
- Check **Use Lambda Proxy integration**
- Save the changes, ok

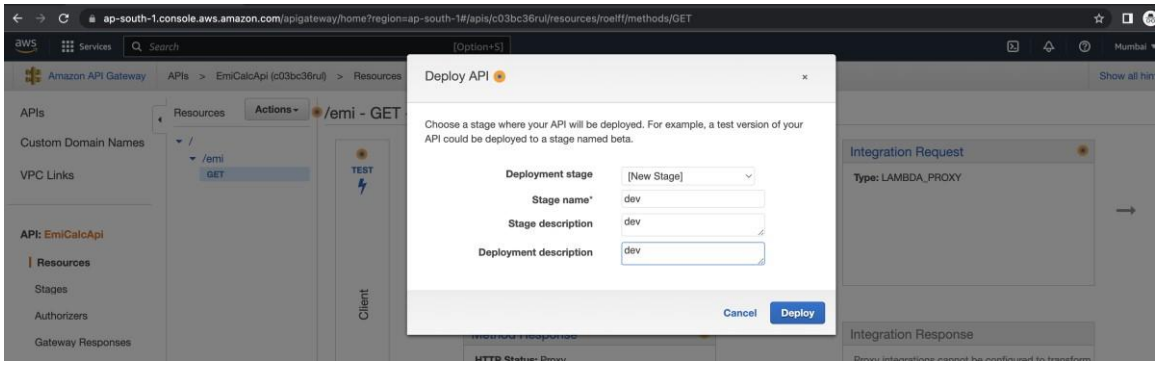


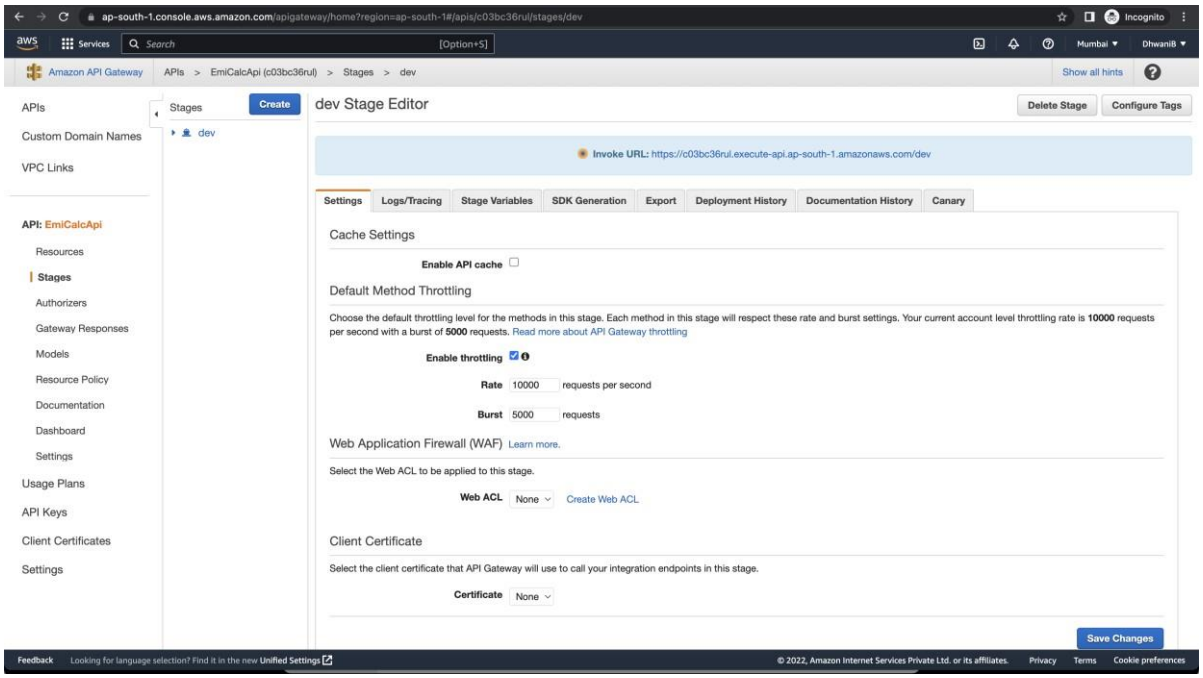


- Next step is to click on actions and then deploy api



- Enter deployment stage as New Stage • Enter stage name as dev
- Stage description as dev
- Deployment description as dev
- After that click on GET method
- Copy the URL and open in browser in new tab





- To calculate the EMI we have to place valid parameters
- Principle = p , Rate = r and Time = t
- Then add rate (r) as certain value 5.8 and the final value time as 10
- Need to add after url - **/emi?p=10000&r=5.8&t=10**
- Click on enter and you will have your required emi