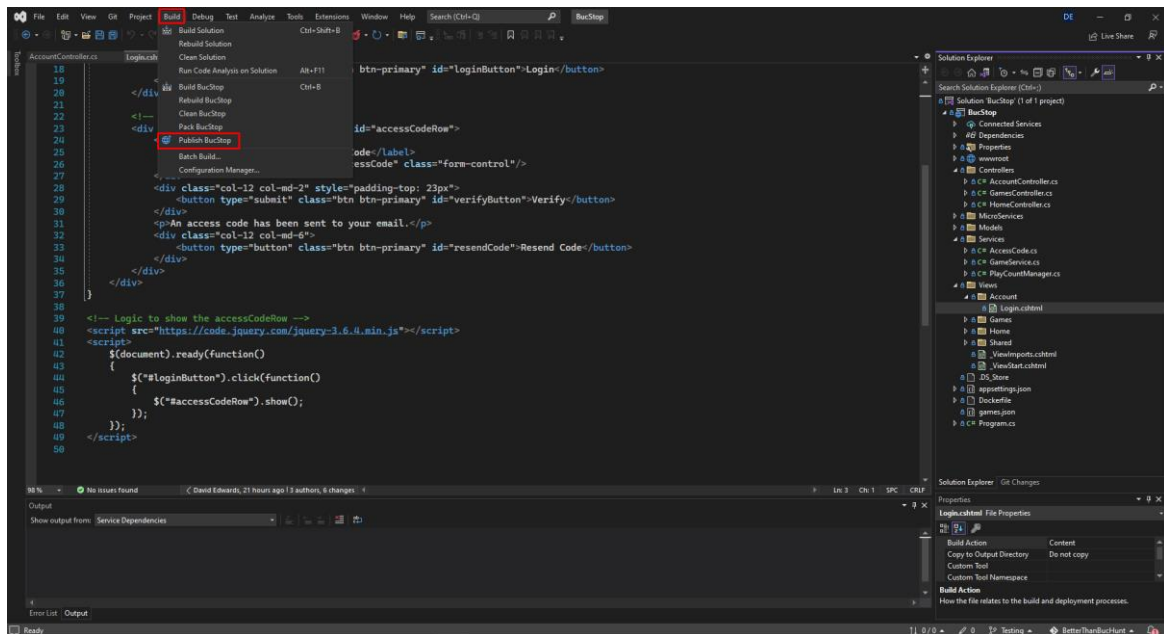


Using AWS and Docker for Deployment

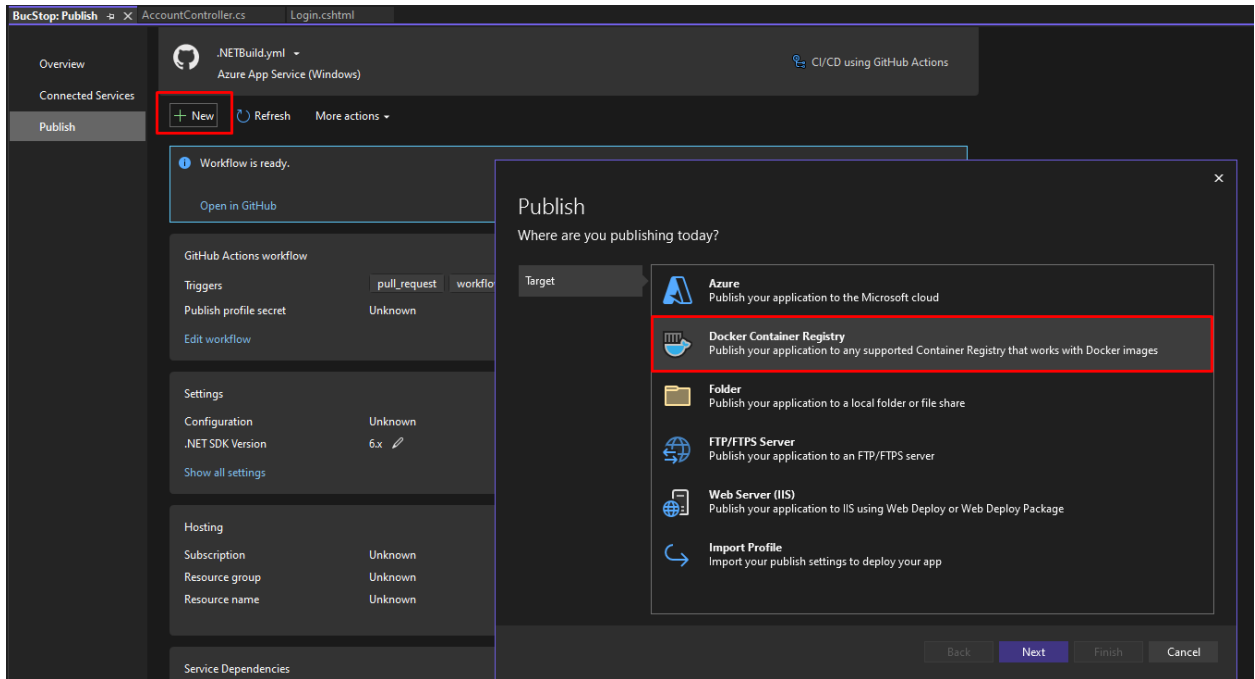
Before continuing, we have a few things we need to get done first.

- Create a Docker account at <https://hub.docker.com>, this will be used to move your program into docker.
- Download Docker Desktop, it is needed to publish with Visual Studio.
- Create an AWS account at <https://portal.aws.amazon.com/billing/signup>

1. In Visual Studio, while in your project navigate to the top bar and click “Build” and then “Publish <Project name>”.

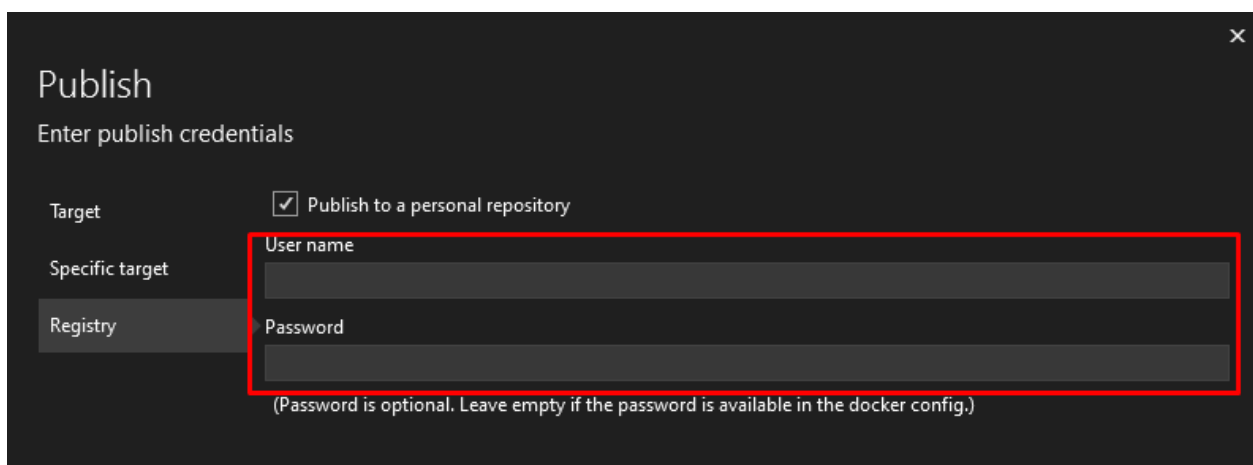


- Since this is the first time you are publishing the project, you have to press “New” and then choose your target on where you want to publish, in this case it will be “Docker”. Once you press next, you will have three options to choose from, choose the one that says “Docker Hub”.

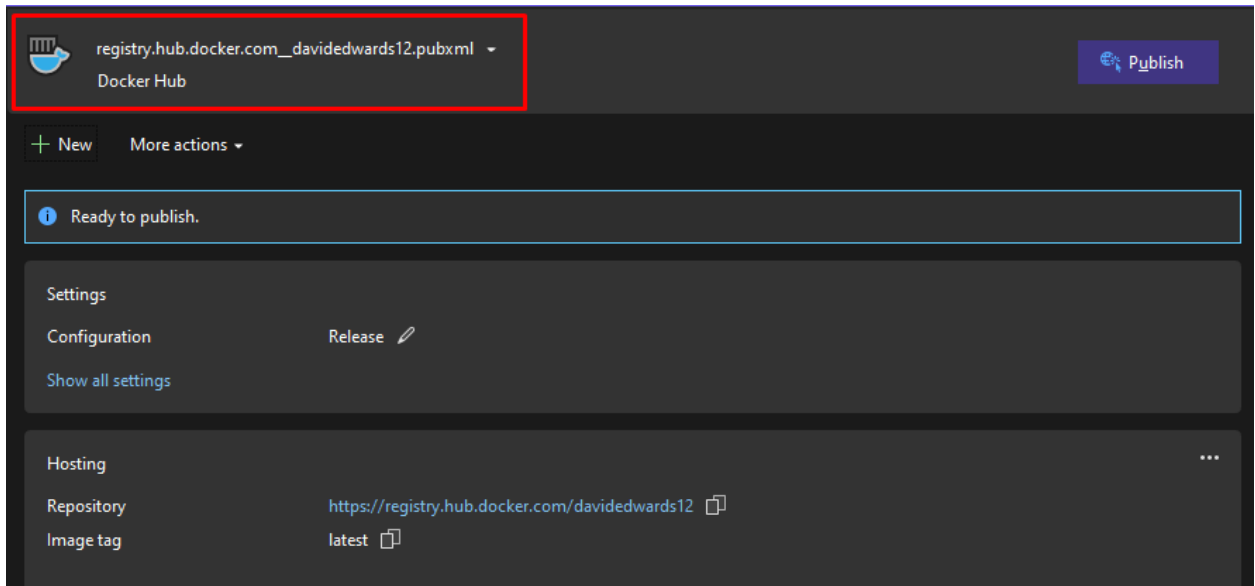


Note: You might have to install some packages in your VS if you do not have the required packages. If this is the case, go ahead and install them and then continue.

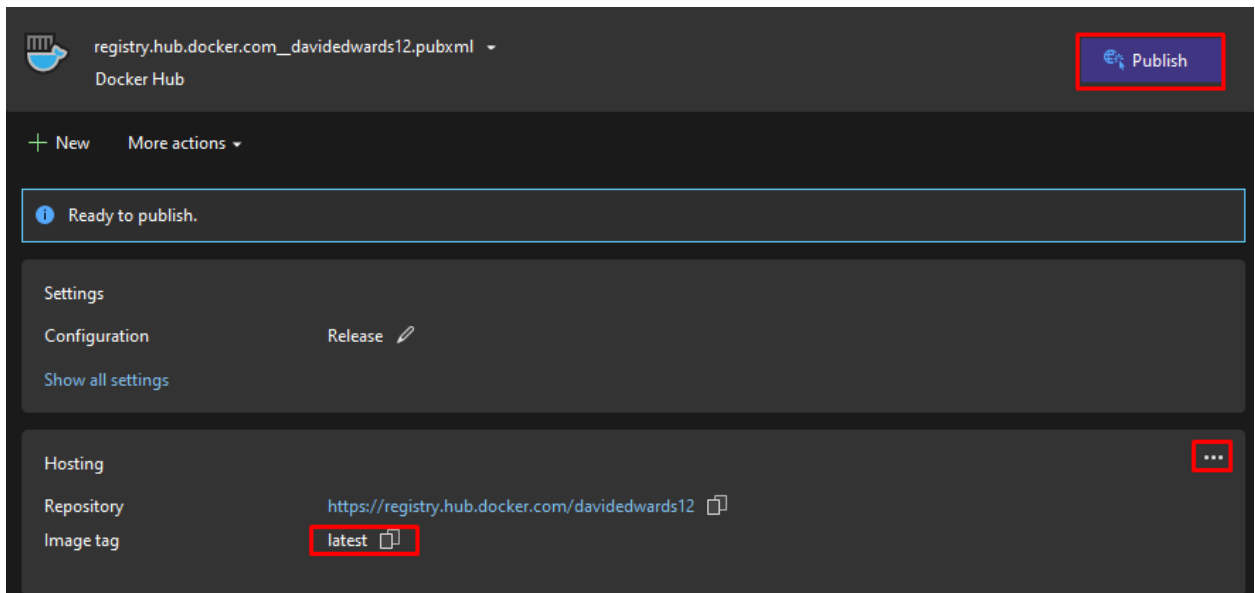
- Next, log in into your Docker Hub account that you created at the start. Once you log in, you can press “Close.”



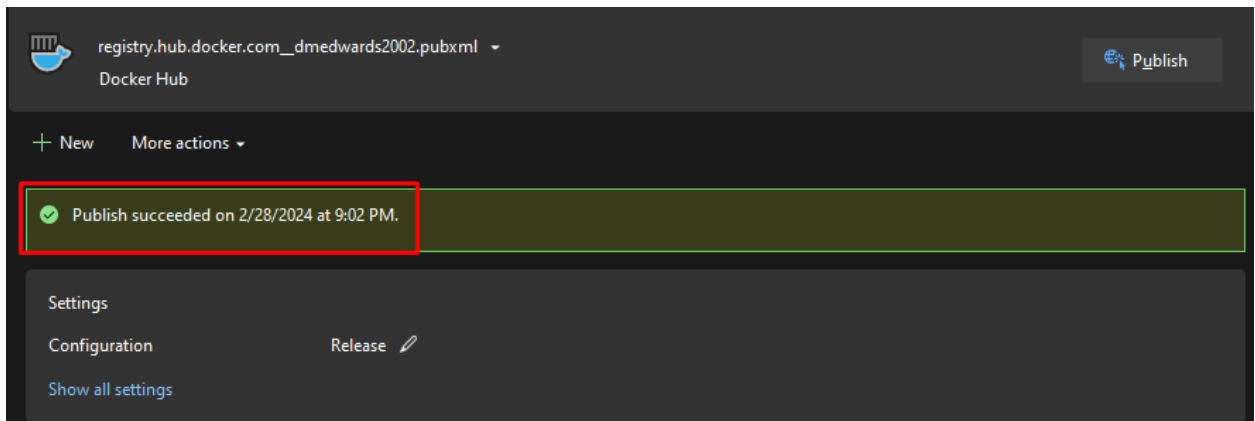
- Back in the publish tab, you should see that you are now logged into your Docker Hub account and can publish to your Docker Hub.



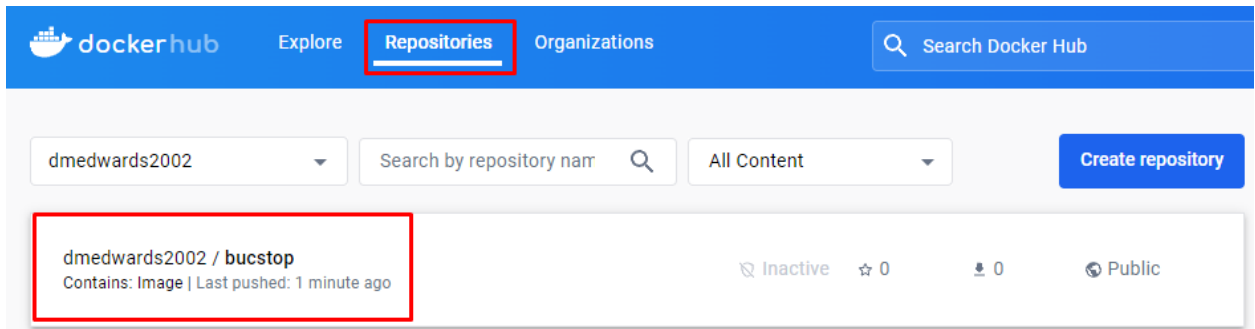
- Now we can publish the project! Here, you can change the image tag by pressing the three little dots to the side (I think of the image tag as the version of the project, i.e., 1.0.0, 1.0.1, etc..). You can leave it latest if you want to, just know that it replaces the last latest you had. Once you have the image tag set to whatever you want, you can now press the “Publish button” at the top (if you have not downloaded Docker Desktop by this point, you will need to now as VS requires it to publish).



6. Once you have published successfully, you will get this pop up that confirms it.



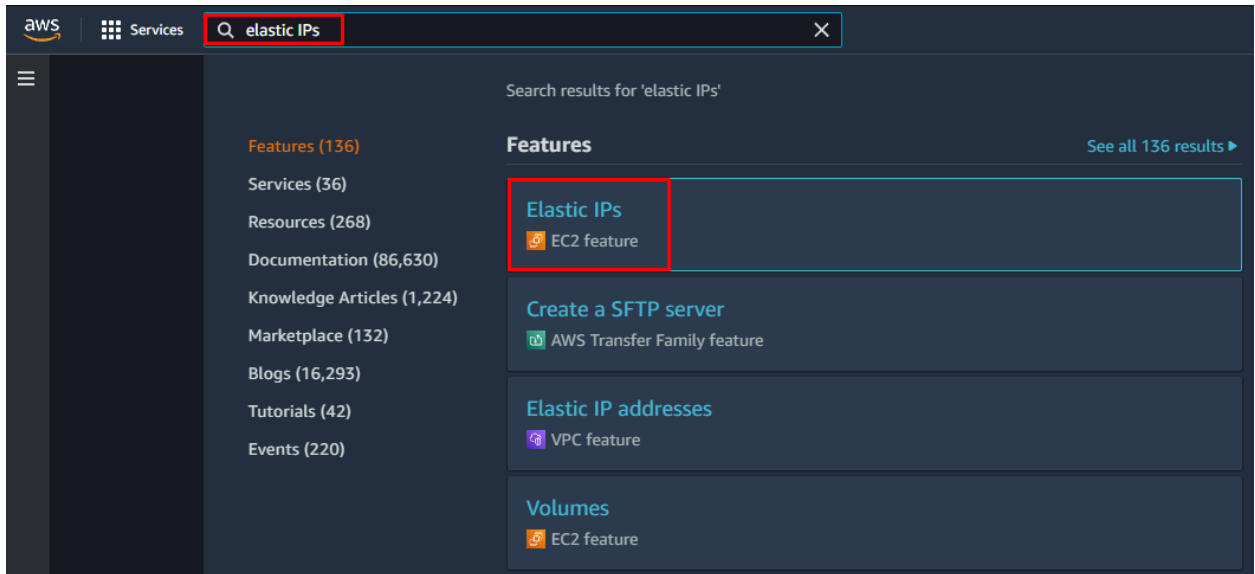
7. Once you have that, you can go to your Docker Hub account and under repositories you should see your newly published project.



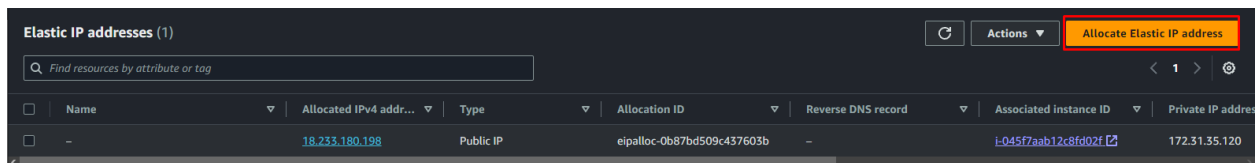
With this in mind, you can now deploy two different ways: either locally through the Docker Desktop application that we downloaded earlier or go through a public IP address with AWS. For this demo, we will be using AWS and an elastic IP address (more on that later).

Setting Up AWS


1. Assuming you have already made an AWS account, head over to the search bar at the top and search for “Elastic IPs”, we will use this later.



2. In the top right, click on “Allocate Elastic IP address.”



3. I recommend leaving everything as it is and pressing “Allocate” at the button. You will be given an IP address to use when we set up our virtual server.

 Services

[Alt+S]

Elastic IP address settings [Info](#)

Network border group [Info](#)

Public IPv4 address pool

- ☒ Amazon's pool of IPv4 addresses
- ☐ Public IPv4 address that you bring to your AWS account with BYOIP. (option disabled because no pools found) [Learn more](#)
- ☐ Customer-owned pool of IPv4 addresses created from your on-premises network for use with an Outpost. (option disabled because no customer owned pools found) [Learn more](#)

Global static IP addresses

AWS Global Accelerator can provide global static IP addresses that are announced worldwide using anycast from AWS edge locations. This can help improve the availability and latency for your user traffic by using the Amazon global network. [Learn more](#)

Create accelerator

Tags - optional

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

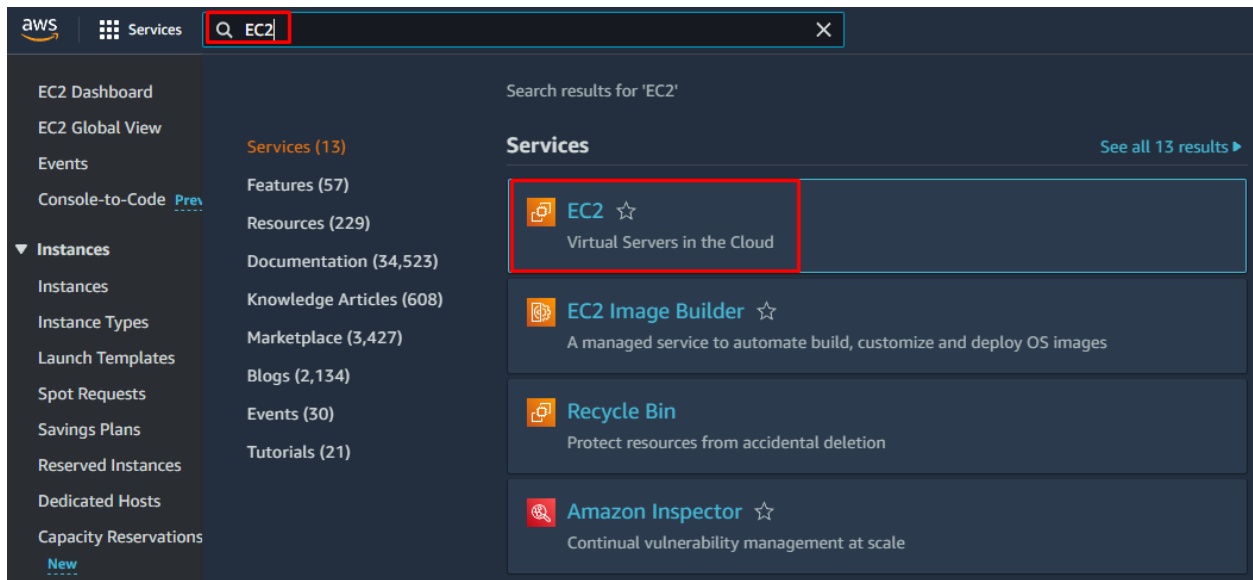
Add new tag

You can add up to 50 more tag

Cancel

Allocate

4. Next, we search for “EC2.” This is where we set up our virtual server.



5. On that page, you will want to press “Launch instance.”

6. Now we are setting up the instance. Name it whatever it want, and for this we will be using Amazon Linux

Name and tags Info

Name

Test

Add additional tags

▼ Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Q Search our full catalog including 1000s of application and OS images

Recents

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

SUSE Li

SUSE

Q

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI

ami-0440d3b780d96b29d (64-bit (x86), uefi-preferred) / ami-0f93c02efd1974b8b (64-bit (Arm), uefi)

Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible ▼


7. Scrolling down, you can use a Key Pair to secure the instance more. It is not necessary but recommended. Then, in the network settings, check all the boxes that I have.

▼ **Key pair (login)** [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

etsu2024

 [Create new key pair](#)

▼ **Network settings** [Info](#)

Edit

Network [Info](#)

vpc-063dc5898d57461d0

Subnet [Info](#)

No preference (Default subnet in any availability zone)

Auto-assign public IP [Info](#)

Enable

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group

☐ Select existing security group

We'll create a new security group called 'launch-wizard-8' with the following rules:

☒ Allow SSH traffic from

Helps you connect to your instance

Anywhere

0.0.0.0/0

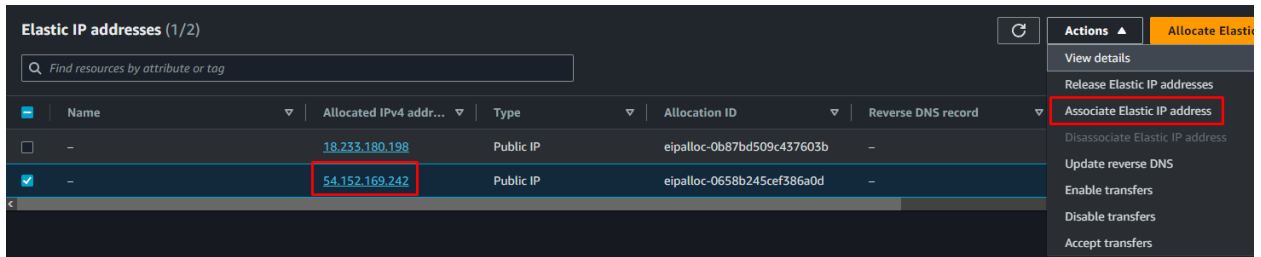
☒ Allow HTTPS traffic from the internet

To set up an endpoint, for example when creating a web server

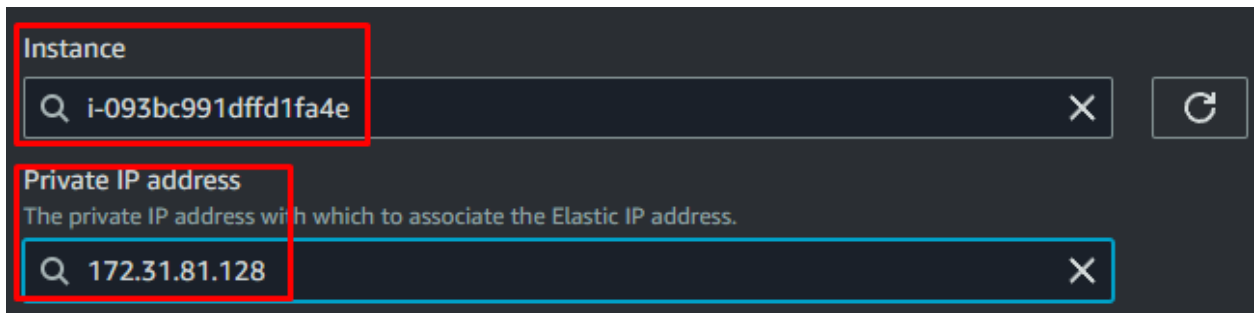
☒ Allow HTTP traffic from the internet

To set up an endpoint, for example when creating a web server

8. Finally, press “Launch Instance” on the side.
9. Before you connect to the instance, we can now add the elastic IP to the instance. Go back to the Elastic IPs and select the one you got and press “Actions” and then “Associate Elastic IP Address.”



- Choose the instance that you just created and select the provided private IP, then press Associate.



- When you go back to your instances, you should now see that the Elastic IP is now associated with the instance that you selected.
- Connect to your instance with all the settings that are provided.
- Now that we are connected to the instance, we have to install docker on the instance to be able to run the commands. We use commands “sudo yum update -y” to make sure everything is up to date and then use “sudo yum install docker”

```
[ec2-user@ip-172-31-81-128 ~]$ sudo yum update -y
Last metadata expiration check: 0:10:55 ago on Thu Feb 29
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-81-128 ~]$ sudo yum install docker
```

- Run the command “sudo service docker start” to start the Docker service.
- Now run the command “sudo su” to get into the root account. This is where we will do all the Docker things.
- We now pull in our Docker image that we published using VS. To do this, we run the command “docker pull \$username/name of project:tag” in my case, this will be “docker pull dmedwards2002/bucstop:latest.” To make sure you got the correct one, you can run “docker images” and it will show all the images you have in the instance.

```
[root@ip-172-31-81-128 ec2-user]# docker pull dmedwards2002/bucstop:latest
latest: Pulling from dmedwards2002/bucstop
5d0aeceef7ee: Pull complete
7c2bfda75264: Pull complete
950196e58fe3: Pull complete
ecf3c05ee2f6: Pull complete
819f3b5e3ba4: Pull complete
20ae32215d86: Pull complete
4f4fb700ef54: Pull complete
58b9c250b831: Pull complete
Digest: sha256:d9ee3fd5ac8ff0ee3686a99dcf5a84273c84df536737221480dadba855b35fc
Status: Downloaded newer image for dmedwards2002/bucstop:latest
docker.io/dmedwards2002/bucstop:latest
[root@ip-172-31-81-128 ec2-user]# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
dmedwards2002/bucstop	latest	5a80cb85901b	54 minutes ago	218MB

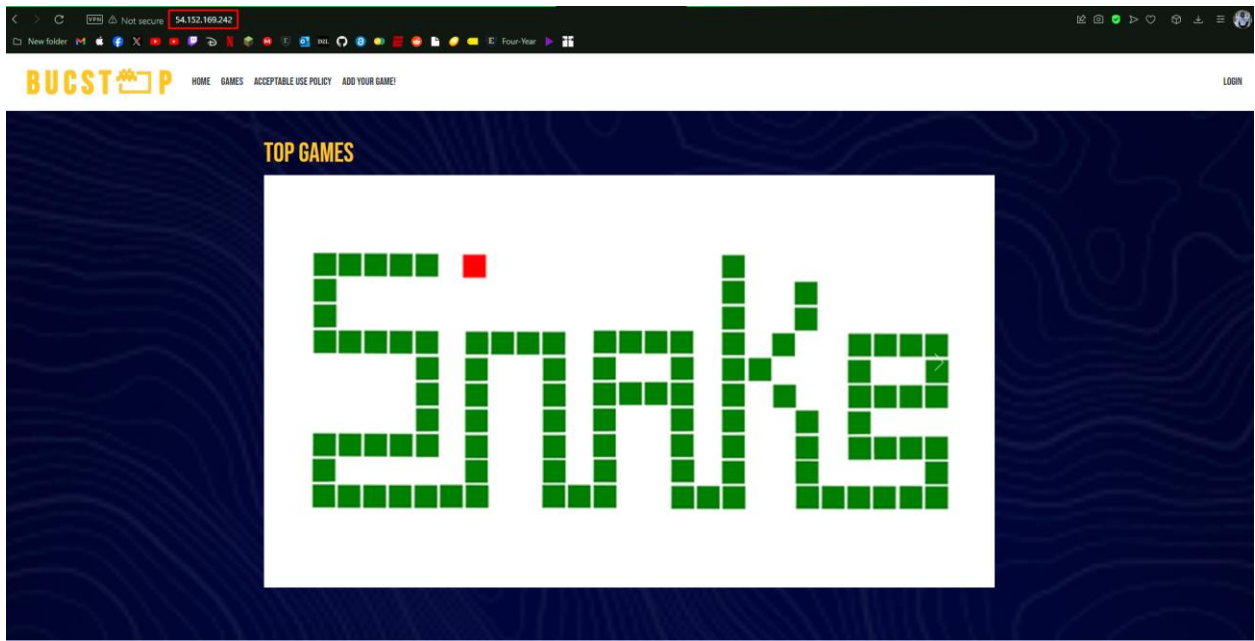
17. Next we run the image using “docker run -d -p<port>:<port> \$username/name of project:tag,” for me it will look like “docker run -d -p80:80 dmedwards2002/bucstop:latest.”

```
docker run -d -p80:80 dmedwards2002/bucstop:latest
```

18. To make sure that the image is running, you can do “docker container ls” and it will list all of your active containers.
19. Now, to make sure it all worked, you can copy and paste that Elastic IP address into your address bar and press enter! If you do not remember what the IP is, then at the bottom of the instance, it will tell you what your public IP address is.

```
i-093bc991dffd1fa4e (Test)
PublicIPs: 54.152.169.242 PrivateIPs: 172.31.81.128
```

20. And now you can give this IP to anyone while you have the container running, and they will be able to see your project!



Note: The container will continue running and that uses resources, resources that are limited due to AWS having a free version and if you go over the limits then you start paying. To stop this from happening, be sure to pause the container whenever you are done running it. You can also stop the container, but that gets rid of it entirely so pausing does the job just fine. To do this, enter the command “docker container pause <container name>”, for me it looks like “docker container pause exciting_lamport.” AWS comes up with crazy names.

```
[root@ip-172-31-81-128 ec2-user]# docker container ls
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
1a8608934226   dmedwards2002/bucstop:latest       "dotnet BucStop.dll"    8 minutes ago Up 8 minutes  0.0.0.0:80->80/tcp, :::80->80/tcp, 443/tcp exciting_lamport
[root@ip-172-31-81-128 ec2-user]# docker container pause exciting_lamport
exciting_lamport
[root@ip-172-31-81-128 ec2-user]# docker container ls
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
1a8608934226   dmedwards2002/bucstop:latest       "dotnet BucStop.dll"    8 minutes ago Up 8 minutes  0.0.0.0:80->80/tcp, :::80->80/tcp, 443/tcp exciting_lamport
[root@ip-172-31-81-128 ec2-user]#
```

That is how you utilize AWS and Docker to deploy your project. Hope this helps you in figuring it all!