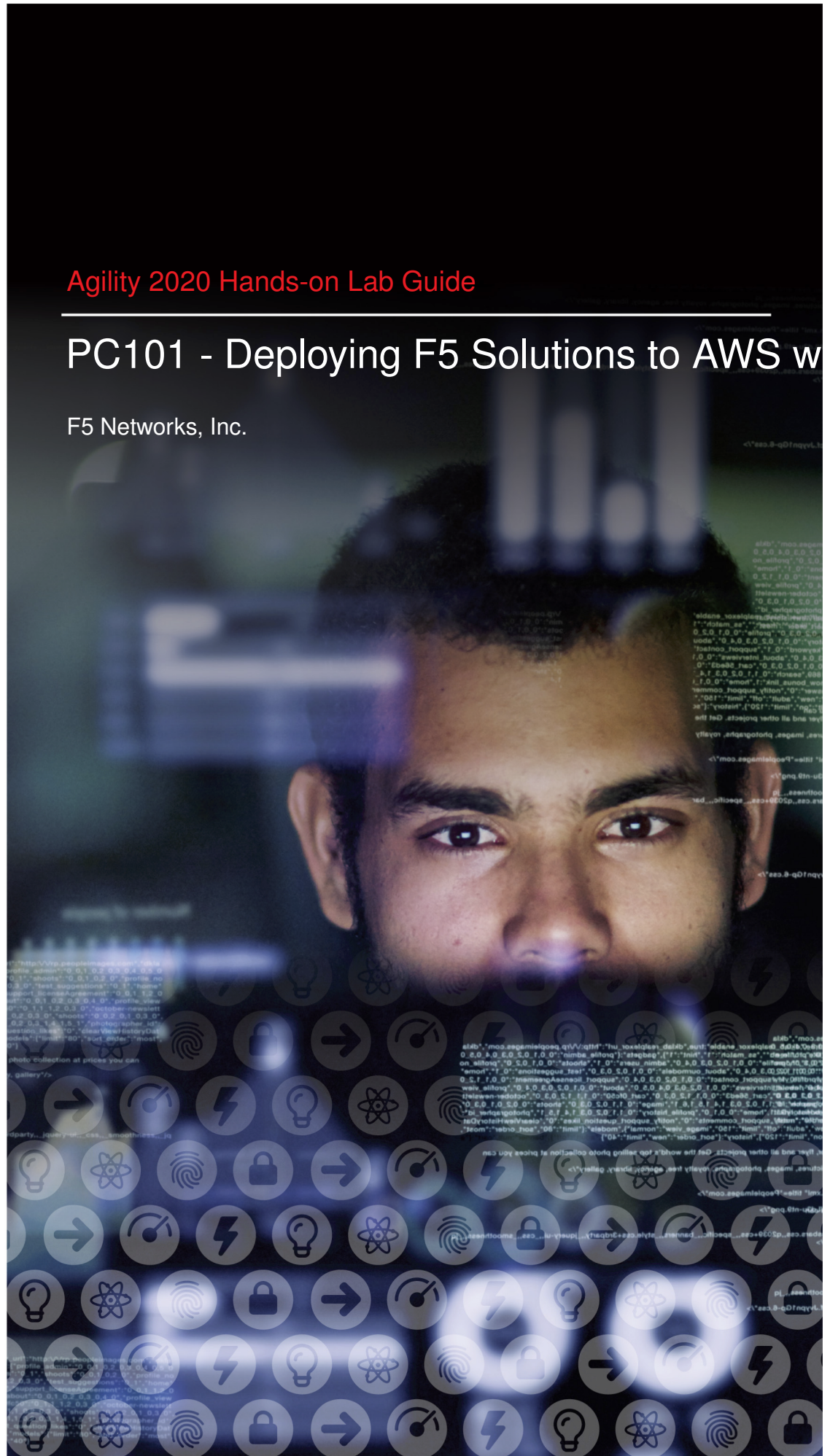




Agility 2020 Hands-on Lab Guide

PC101 - Deploying F5 Solutions to AWS w

F5 Networks, Inc.



Contents:

| | | |
|----------|-----------------------------------------------------------------------------|----------|
| 1 | PC 101 - Deploying F5 Solutions to AWS with CloudFormation Templates | 5 |
| 1.1 | Prerequisites | 5 |
| 1.2 | Deploying an F5 CloudFormation Template to AWS | 18 |
| 1.3 | Explore the AWS Virtual Private Cloud | 22 |
| 1.4 | F5 Admin SSH and Configuration Utility (WebUI) Access | 29 |
| 1.5 | Creating High Availability Virtual Servers in AWS | 42 |
| 1.6 | Delete Environment in Fell Swoops | 46 |

PC 101 - Deploying F5 Solutions to AWS with CloudFormation Templates

This is a self-guided introductory lab that covers:

- Deploying F5 solutions to AWS with Terraform and CloudFormation Templates
- Service Discovery iApp - Dynamically populate pool members with the AutoDiscovery iApp.
- Cross Availability Zone HA with F5
- Application Services Extension - Automate the creation of virtual servers and associated services using a declarative model.

1.1 Prerequisites

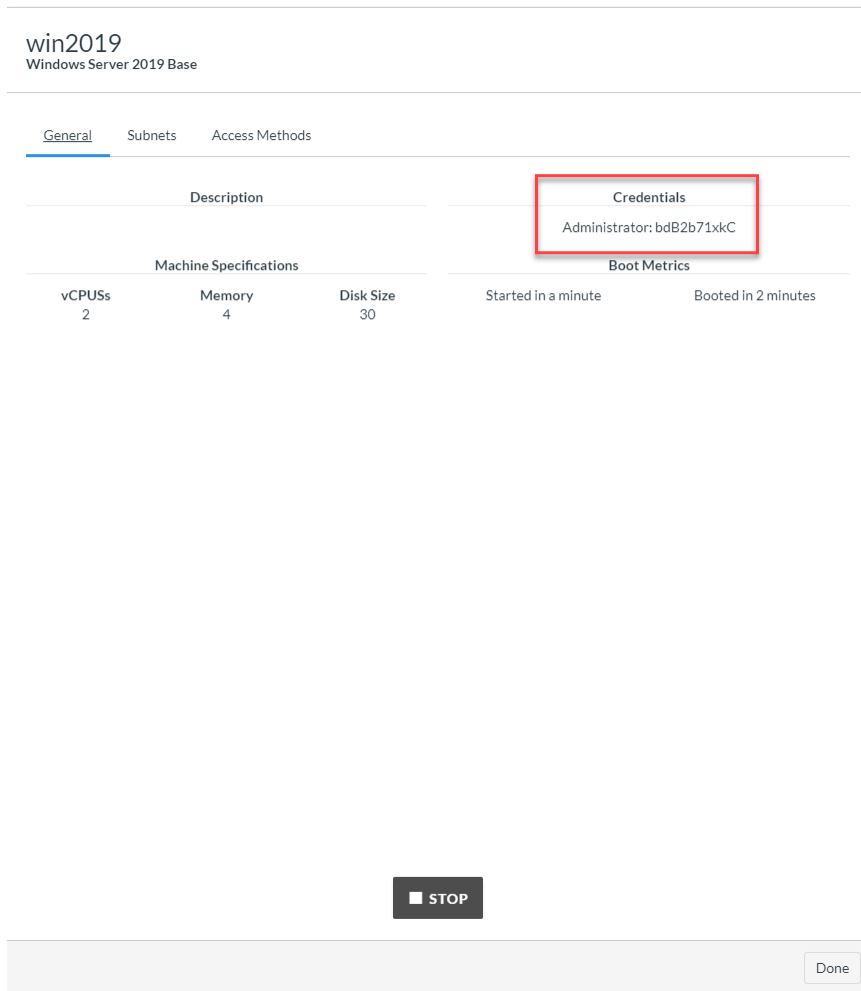
In this lab we'll prepare our AWS environment with the prerequisites required to launch F5 CloudFormation templates.

- Create AWS account and login to the AWS console.
- Accept the F5 license subscription from the AWS Marketplace.
- Create and upload SSH keys to AWS.
- Create the infrastructure prerequisites necessary to launch an F5 CloudFormation template.

1.1.1 Login to F5's lab platform (Unified Demo Framework) and launch a remote desktop session

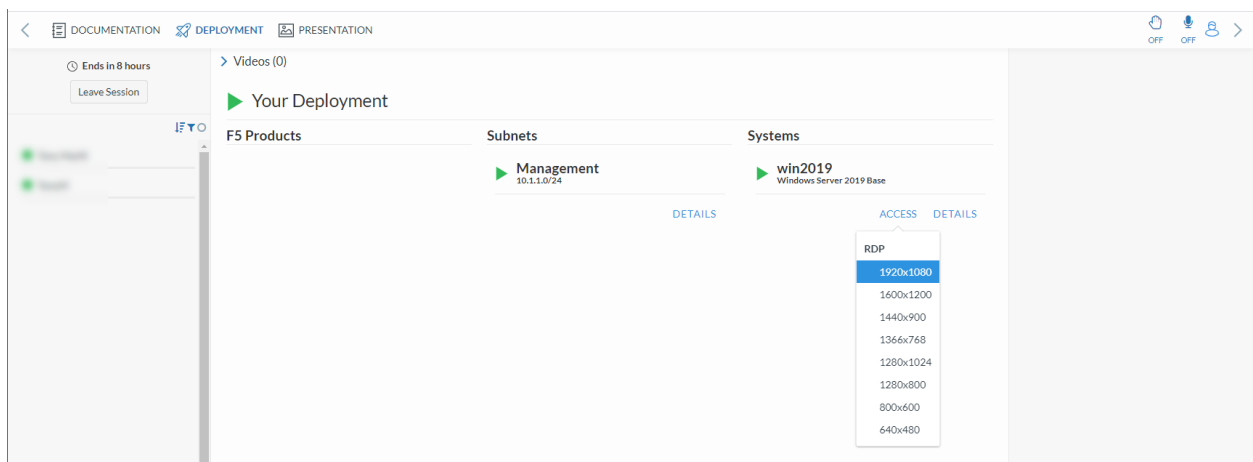
Follow the link in your email invite and login to the lab environment.

Your Deployment => Systems => win2019 => DETAILS. Copy the Administrator password to your clipboard. The password starts immediately after Administrator:

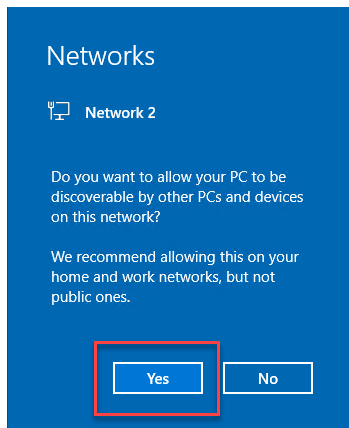


Click [Done] in the lower right-hand corner.

Under win2019, from the ACCESS drop-down, select RDP and resolution. The lab looks best in FHD (1920 x 1080). This will download the rdp file to your computer. Launch the RDP file and login via RDP using the Administrator credentials. Copy and paste credentials from your clipboard.



When prompted with the blue “Networks” message click “Yes”.



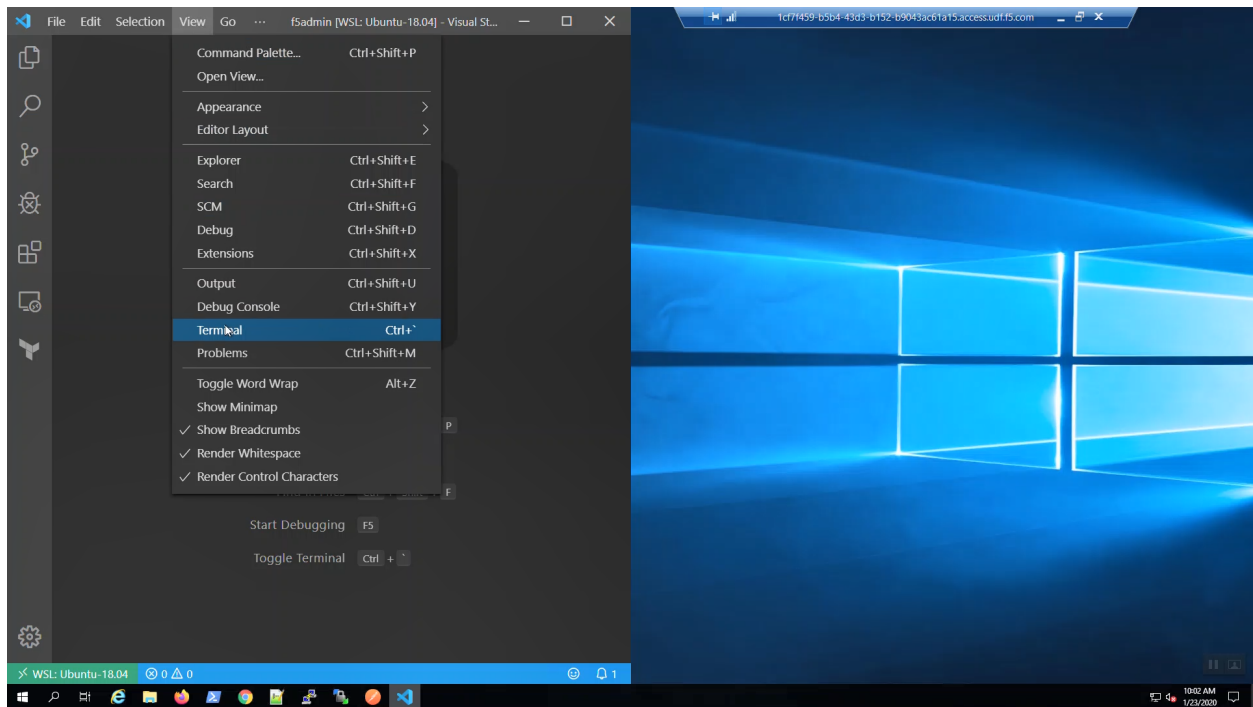
Attention: Before proceeding, wait 30 seconds for the Visual Studio Code and Postman applications to start automatically.

You can run the entire lab from within the Windows jump host.

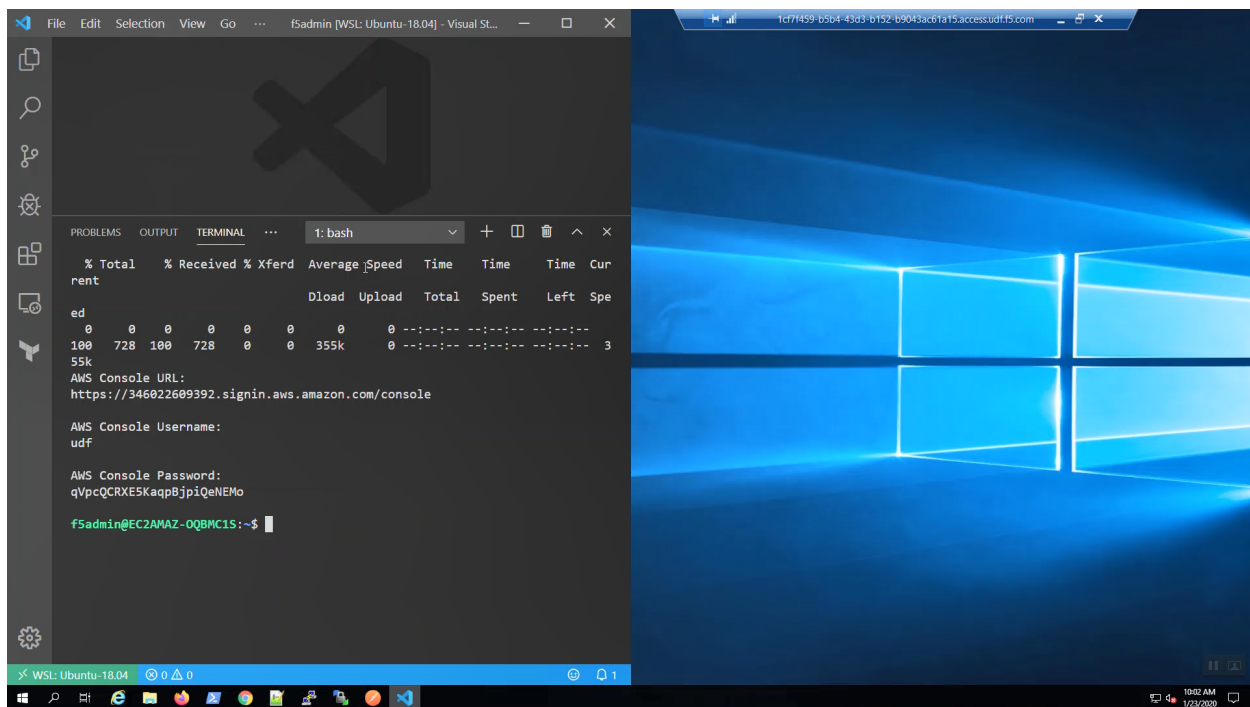
Visual Studio Code => View => Terminal

You will see:

- AWS Console URL
- AWS Console Username
- AWS Console Password



Maximize the Terminal Window by clicking on the ^ in the upper-right-hand corner of the Terminal.



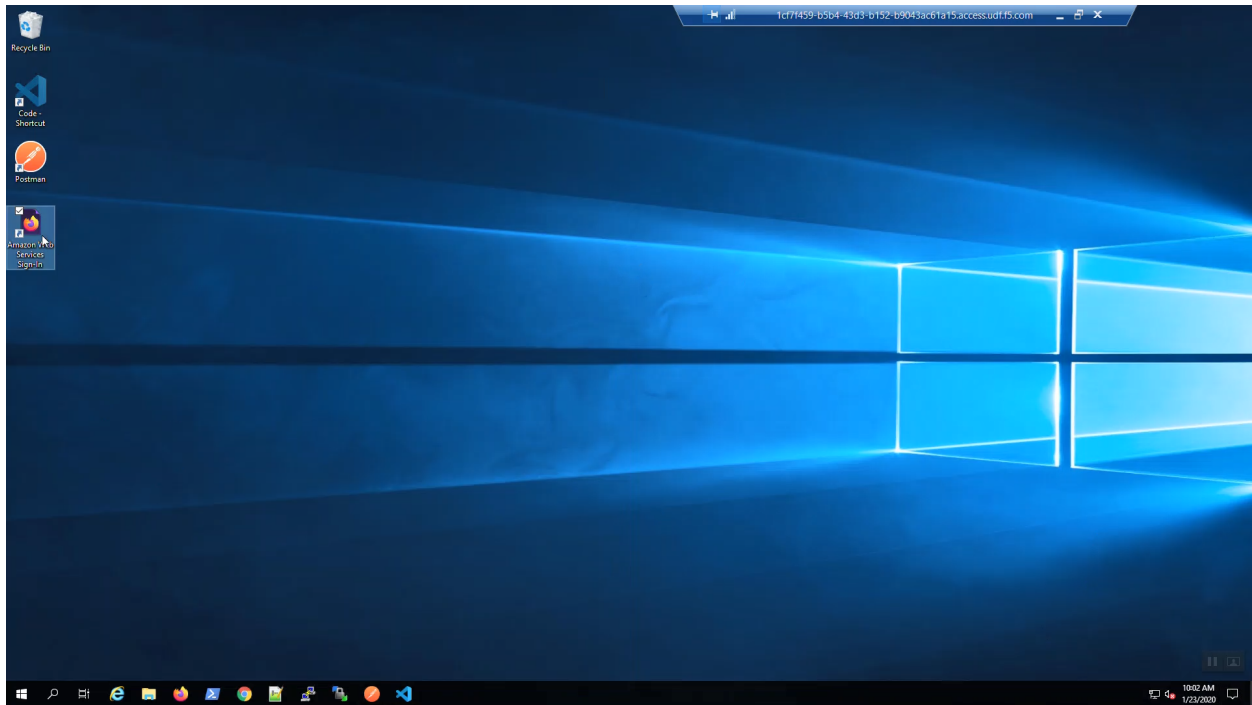
1.1.2 Login to AWS Console

A URL shortcut has been auto-generated on the Windows desktop: “Amazon Web Services Sign-In”.

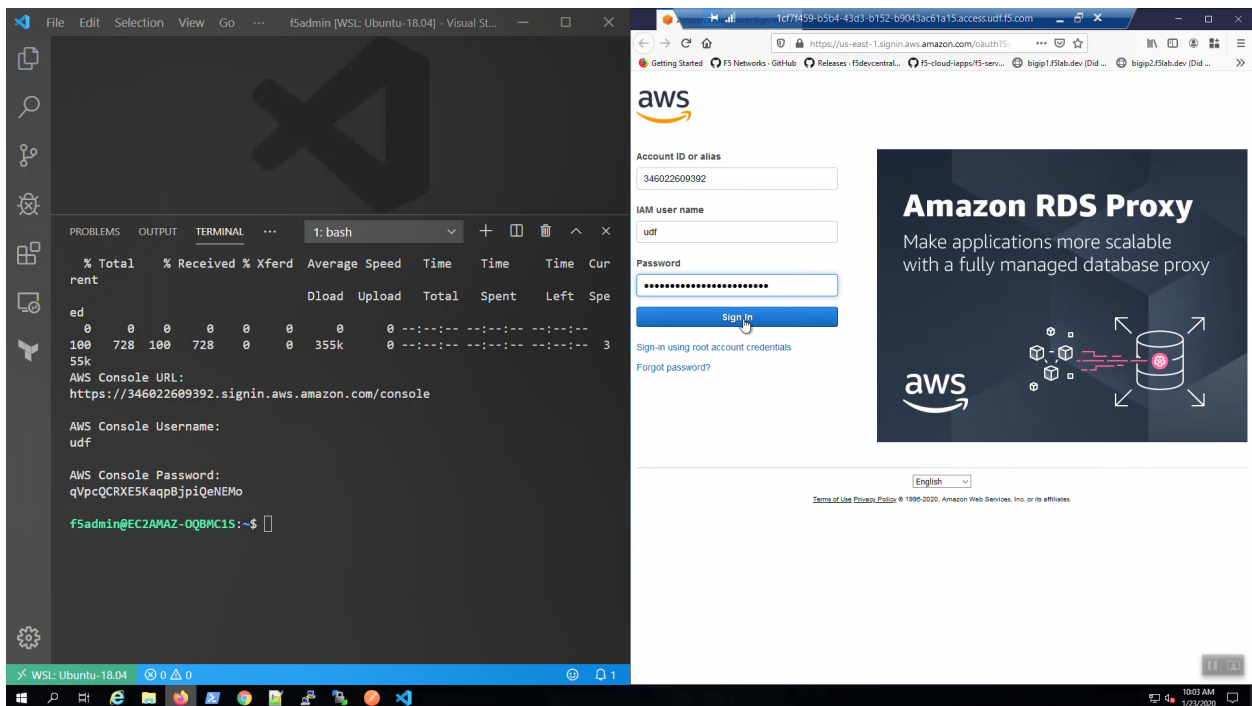
First, launch Firefox from the taskbar, then click on the “Amazon Web Services Sign-In” URL shortcut on the Desktop.

If the Firefox resolution in your RDP session renders components off-screen, try to first launch Firefox from the taskbar *before* you click on the “Amazon Web Services Sign-In” URL shortcut on the Desktop.

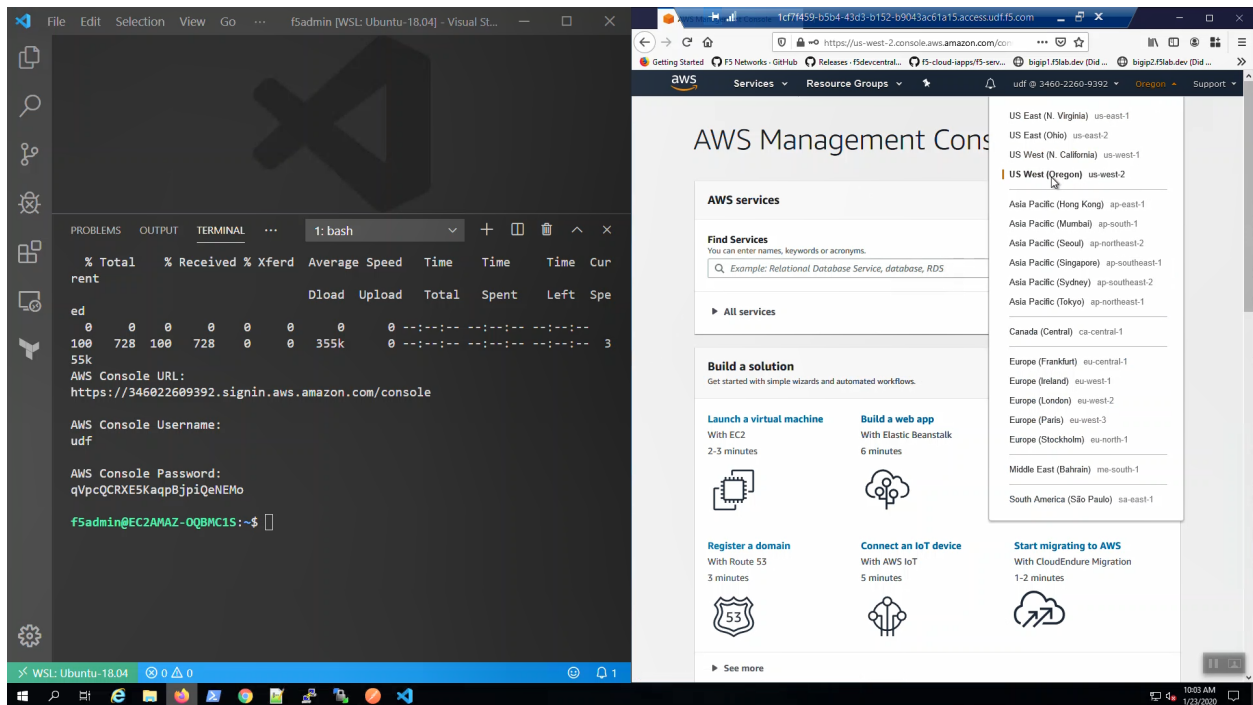
Alternatively, you can CTRL+click the “AWS Console URL:” https shortcut in the Visual Studio Code terminal.



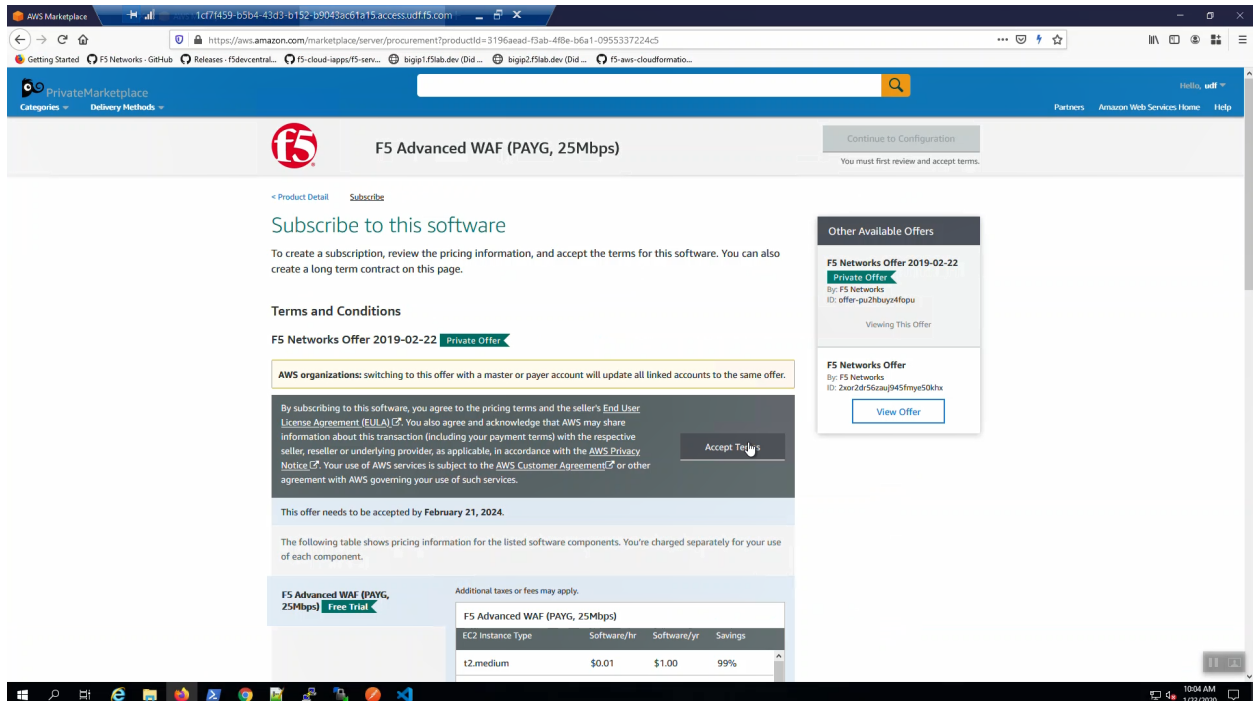
Login to the AWS web console with the credentials shown in your terminal.



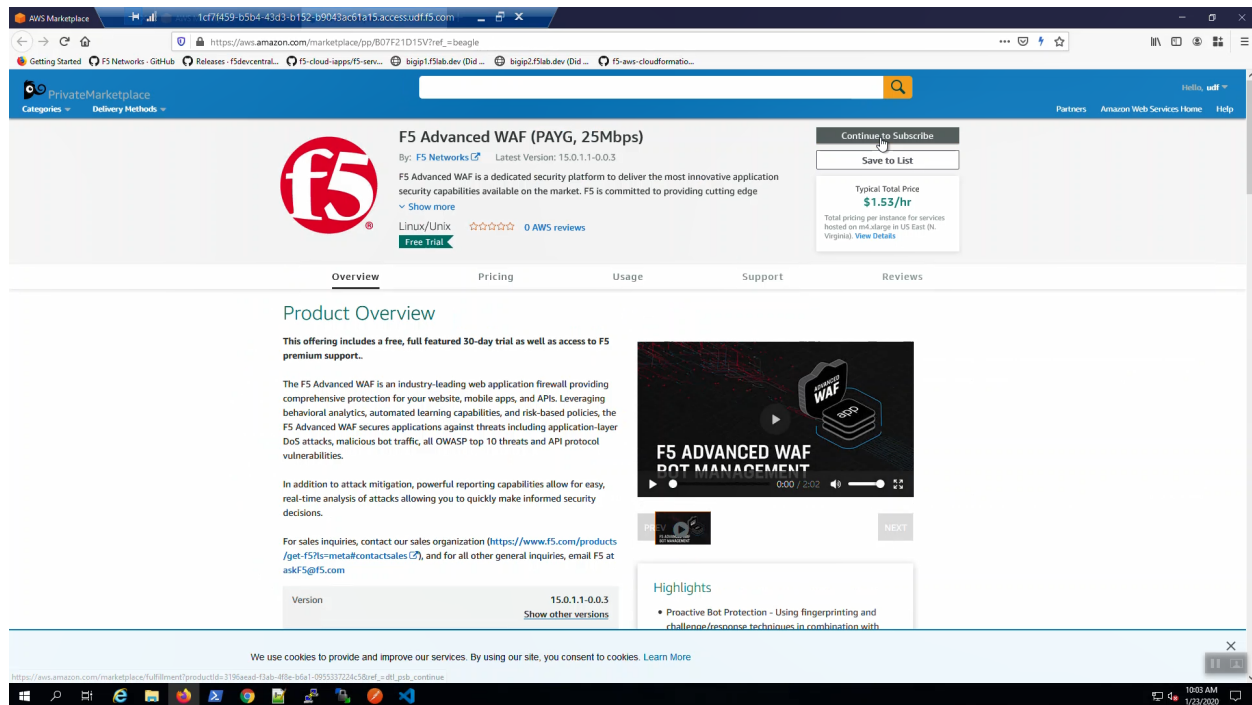
In the upper-right-hand corner, choose US-West (Oregon) us-west-2 region.



“Services” => type “marketplace” in the search window. Select “AWS Marketplace Subscriptions” from the search results. Right-click on “Manage subscriptions” to open a new tab. => “Discover products” Type “f5 advanced 25mbps” in the search box. => “F5 Advanced WAF (PAYG, 25Mbps)” => “Continue to Subscribe” => “Accept Terms”



Track “Effective date” and “Expiration date”. When they are no longer “Pending” you can proceed.



1.1.3 Create an AWS VPC with Terraform

From the Visual Studio Code Terminal, clone the github repository for this lab and change to the working directory.

Attention: For a smooth ride, always invoke commands from inside the cloned git repository (f5agility2020-pc101). To check you're in the right place, you can run the command `pwd` and the output should read `/home/f5admin/f5agility2020-pc101`

```
git clone https://github.com/TonyMarfil/f5agility2020-pc101.git
cd f5agility2020-pc101/
```

The screenshot shows a VS Code terminal window with the following content:

```
f5admin [WSL: Ubuntu-18.04] - 1c77459-b5b4-43d3-b152-b9043ac61a15.access.udf.f5.com
1: bash

% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed

AWS Console URL:
https://346022609392.signin.aws.amazon.com/console

AWS Console Username:
udf

AWS Console Password:
qVpcQCRXE5KaqpBjpiQeNEMo

f5admin@EC2AMAZ-QQBMC1S:~$ git clone https://github.com/TonyMarfil/f5agility2020-pc101.git
Cloning into 'f5agility2020-pc101'...
remote: Enumerating objects: 53, done.
remote: Counting objects: 100% (53/53), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 53 (delta 13), reused 50 (delta 10), pack-reused 0
Unpacking objects: 100% (53/53), done.
f5admin@EC2AMAZ-QQBMC1S:~$ cd f5agility2020-pc101/
f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$
```

Run the start.sh script to set environment variables and make the ./scripts directory executable

```
source ./start.sh
```

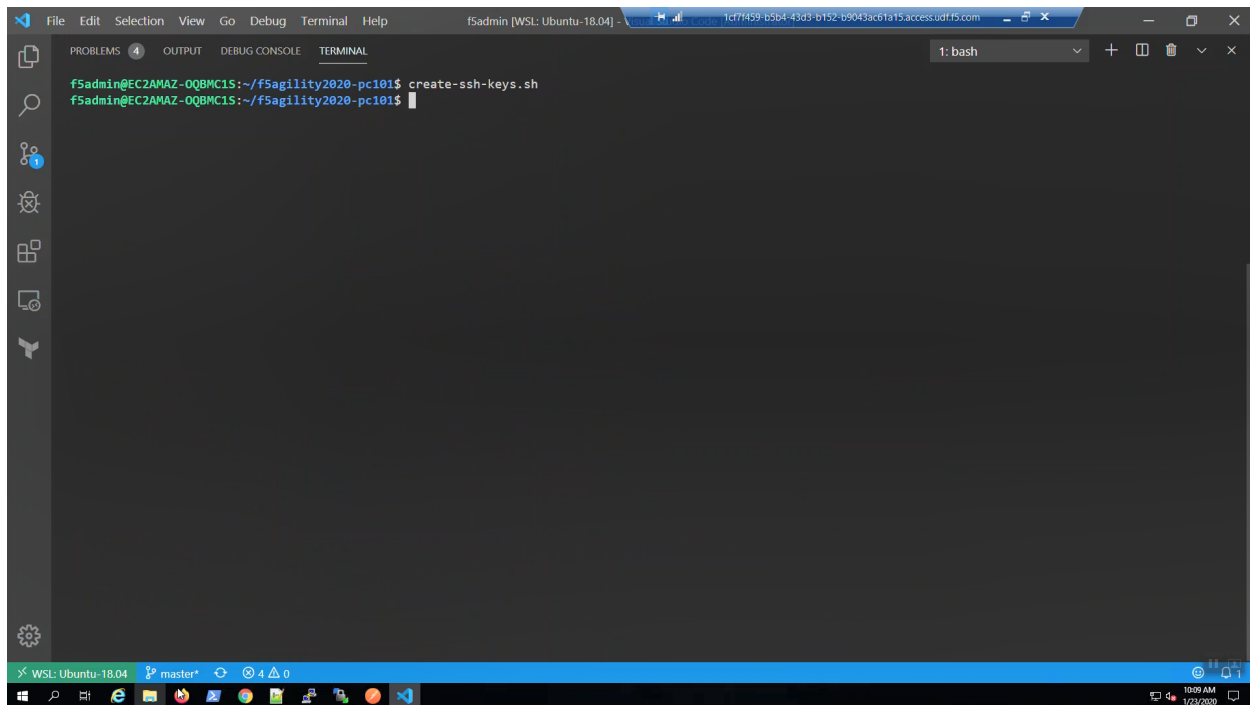
The screenshot shows a VS Code terminal window with the following content:

```
f5admin [WSL: Ubuntu-18.04] - 1c77459-b5b4-43d3-b152-b9043ac61a15.access.udf.f5.com
1: bash

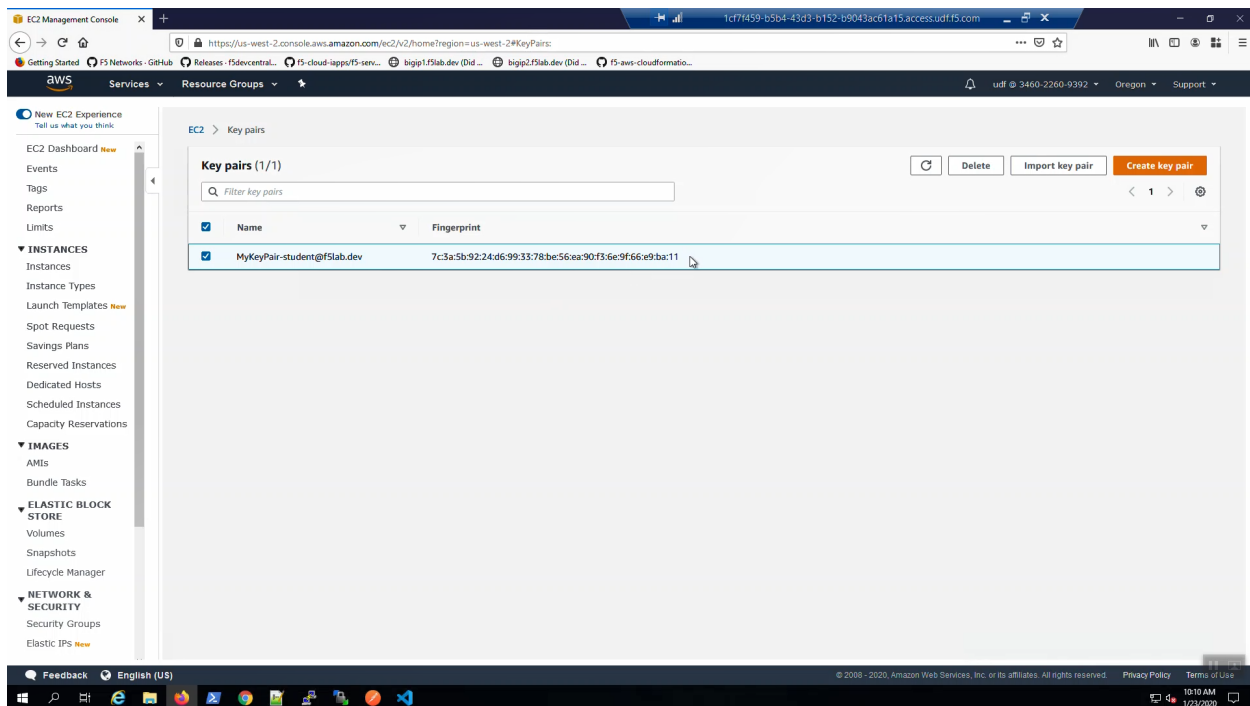
f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$ source ./start.sh
f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$
```

Create an SSH key and upload to your AWS account. We'll later use this key to connect to our F5 instances.

```
create-ssh-keys.sh
```

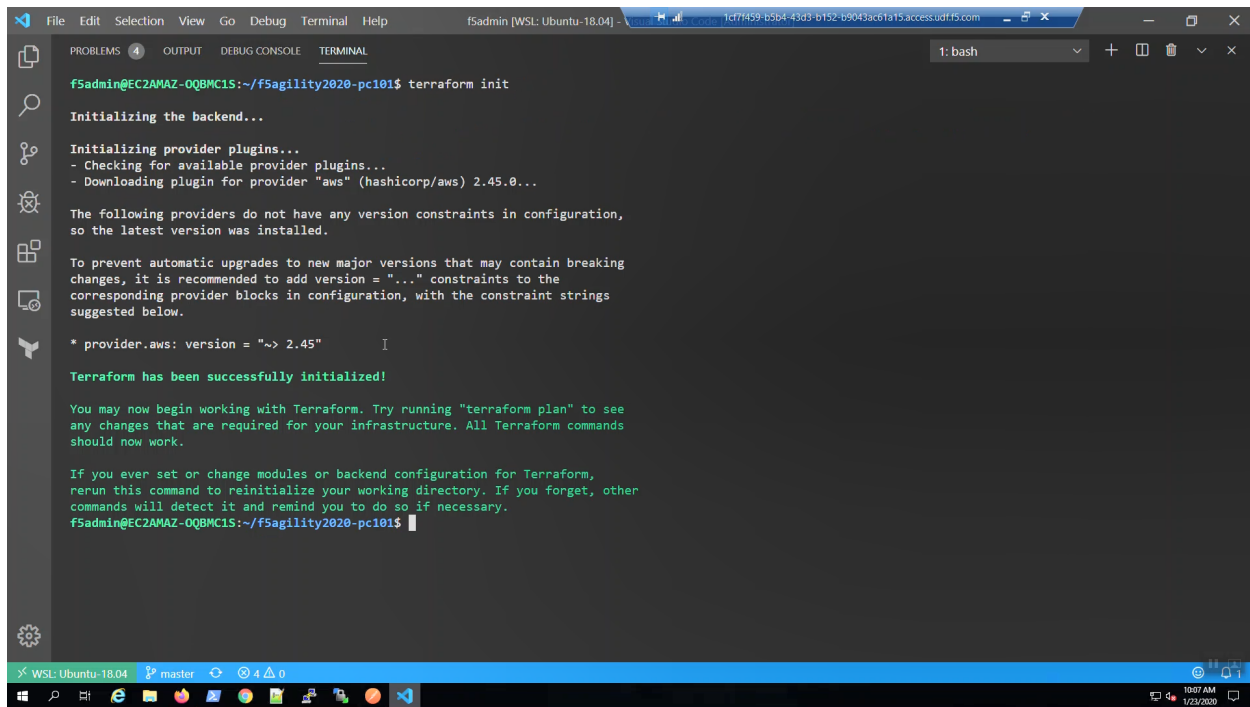


From the AWS Console => Services => EC2 => Key pairs. Confirm your ssh key was created.



Initialize Terraform modules.

```
terraform init
```



```
f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$ terraform init

Initializing the backend...

Initializing provider plugins...
- Checking for available provider plugins...
- Downloading plugin for provider "aws" (hashicorp/aws) 2.45.0...

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, it is recommended to add version = "... constraints to the
corresponding provider blocks in configuration, with the constraint strings
suggested below.

* provider.aws: version = "~> 2.45"

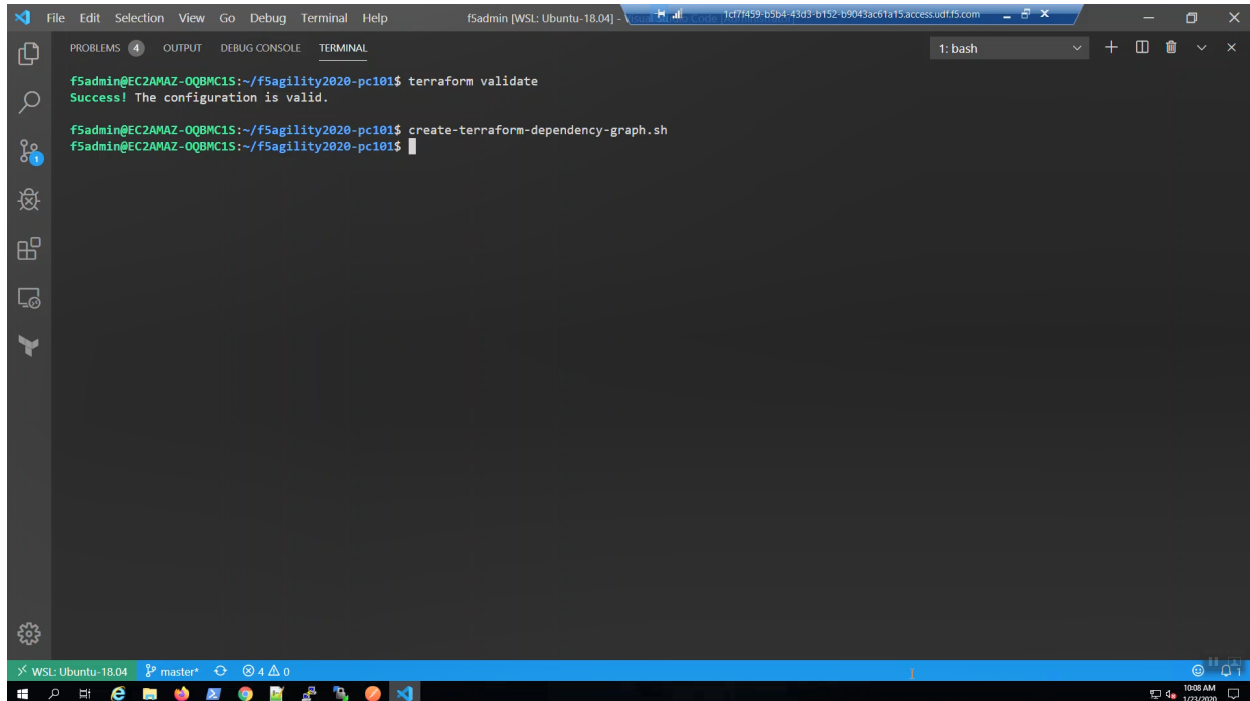
Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$
```

Validate Terraform files and create a terraform dependency graph.

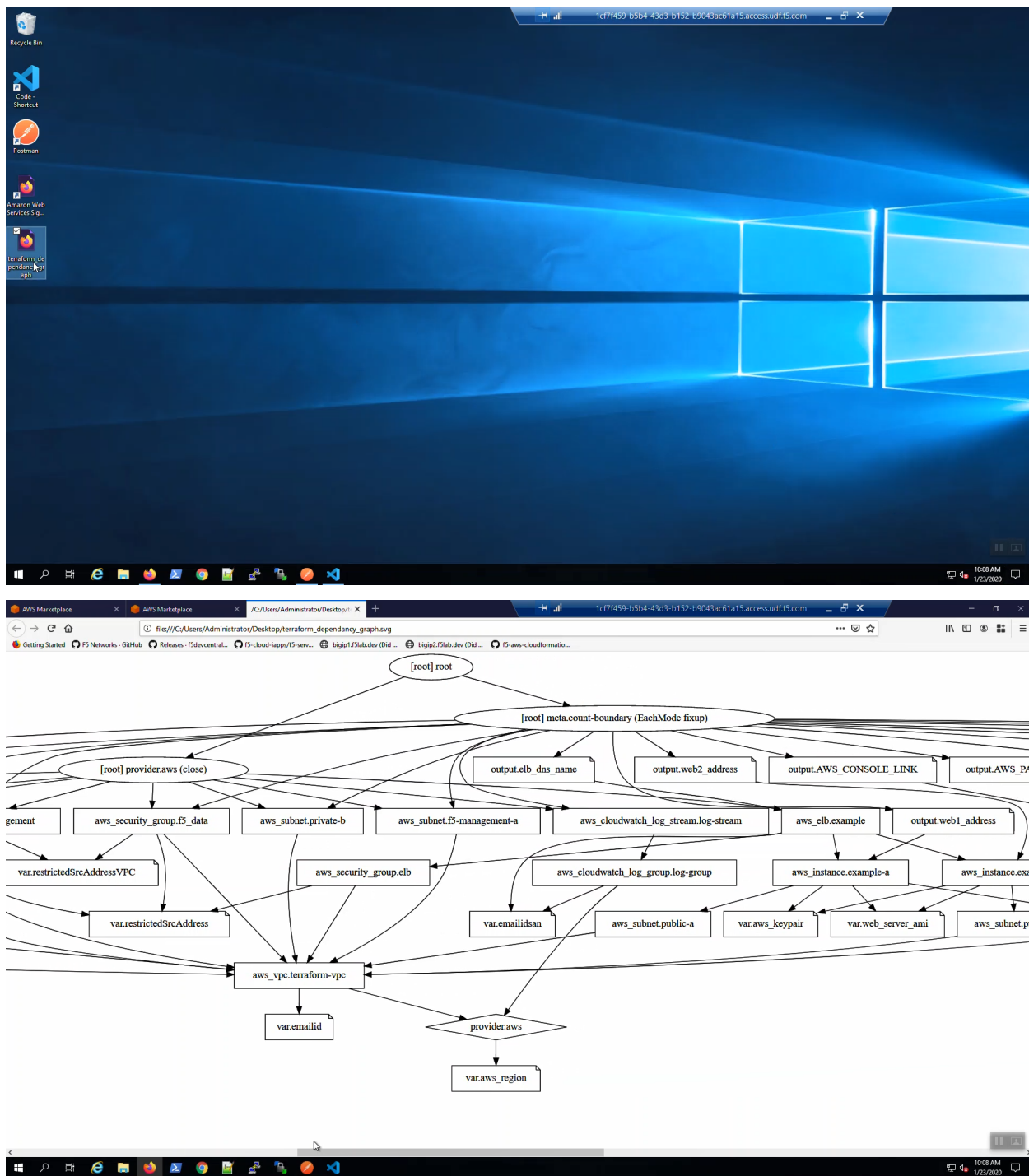
```
terraform validate
create-terraform-dependency-graph.sh
```



```
f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$ terraform validate
Success! The configuration is valid.

f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$ create-terraform-dependency-graph.sh
f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$
```

From the Windows desktop, click on the “terraform_dependency_graph” URL shortcut. Review in your browser. Terraform creates a dependency of all of the objects in your environment. This is one of the major advantages to using a declarative tool for building infrastructure and services.



From the Visual Studio Code Terminal

```
terraform plan -var 'bigip_admin_password=f5letmein'  
terraform apply -var 'bigip_admin_password=f5letmein' -auto-approve
```

The screenshot shows a Visual Studio Code window with a terminal tab active. The terminal is running a command prompt on a Windows Subsystem for Linux (WSL) environment, specifically Ubuntu 18.04. The user is 'fsadmin' and the current directory is '~/f5agility2020-pc101'. The command entered is 'terraform plan -var 'bigip_admin_password=f5letmein''. The output shows that Terraform is refreshing its state in memory and then calculating the plan. The plan is for a 'terraform' provider. The terminal window has a dark theme and a sidebar on the left with icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The status bar at the bottom indicates the WSL environment and the current file.

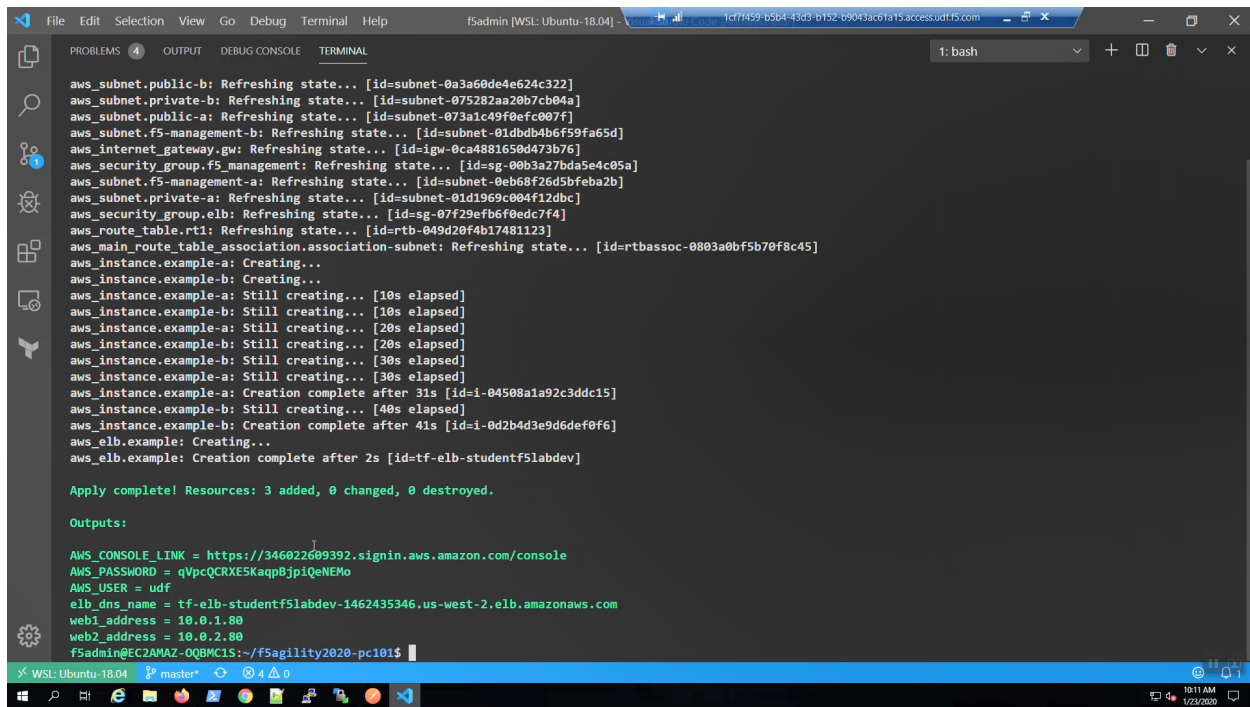
```
f5admin@EC2AMAZ-QQ8MC1S:~/f5agility2020-pc101$ terraform plan -var 'bigip_admin_password=f5letmein'
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

-----

f5admin@EC2AMAZ-QQ8MC1S:~/f5agility2020-pc101$
```

The screenshot shows the same Visual Studio Code window with the terminal tab active. The user has entered the command 'terraform apply -var 'bigip_admin_password=f5letmein' -auto-approve'. The output shows the Terraform engine creating various AWS resources. The resources are created in a sequence: 'aws_vpc.terraform-vpc', 'aws_cloudwatch_log_group.log-group', 'aws_cloudwatch_log_stream.log-stream', 'aws_vpc.terraform-vpc', 'aws_security_group.instance', 'aws_subnet.public-b', 'aws_subnet.private-a', 'aws_subnet.f5-management-a', 'aws_subnet.f5-management-b', 'aws_security_group.elb', 'aws_security_group.f5_management', 'aws_security_group.f5_data', 'aws_subnet.private-b', 'aws_internet_gateway.gw', 'aws_internet_gateway.gw', 'aws_subnet.public-a', 'aws_security_group.elb', 'aws_subnet.public-b', and 'aws_route_table.r1'. The terminal window has a dark theme and a sidebar on the left with icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The status bar at the bottom indicates the WSL environment and the current file.

```
f5admin@EC2AMAZ-QQ8MC1S:~/f5agility2020-pc101$ terraform apply -var 'bigip_admin_password=f5letmein' -auto-approve
aws_vpc.terraform-vpc: Creating...
aws_cloudwatch_log_group.log-group: Creating...
aws_cloudwatch_log_group.log-group: Creation complete after 0s [id=studentf5labdev]
aws_cloudwatch_log_stream.log-stream: Creating...
aws_cloudwatch_log_stream.log-stream: Creation complete after 0s [id=log-stream]
aws_vpc.terraform-vpc: Creation complete after 1s [id=vpc-078b314202bbdff7]
aws_security_group.instance: Creating...
aws_subnet.public-b: Creating...
aws_subnet.private-a: Creating...
aws_subnet.f5-management-a: Creating...
aws_subnet.f5-management-b: Creating...
aws_security_group.elb: Creating...
aws_security_group.f5_management: Creating...
aws_security_group.f5_data: Creating...
aws_subnet.private-b: Creating...
aws_internet_gateway.gw: Creating...
aws_internet_gateway.gw: Creation complete after 0s [id=igw-0ca4881650d473b76]
aws_subnet.public-a: Creating...
aws_security_group.elb: Creation complete after 1s [id=sg-07f29efb6f0edc7f4]
aws_subnet.public-b: Creation complete after 1s [id=subnet-0a3a60de4e624c322]
aws_route_table.r1: Creating...
```



```
aws_subnet.public-b: Refreshing state... [id=subnet-0a3a60de4e624c322]
aws_subnet.private-b: Refreshing state... [id=subnet-075282aa20b7cb04a]
aws_subnet.public-a: Refreshing state... [id=subnet-073a1c49f0efc007f]
aws_subnet.f5-management-b: Refreshing state... [id=subnet-01dbdb4b6f59fa65d]
aws_internet_gateway.gw: Refreshing state... [id=igw-0ca4881650d473b76]
aws_security_group.f5_management: Refreshing state... [id=sg-00b3a27bda5e4c05a]
aws_subnet.f5-management-a: Refreshing state... [id=subnet-0eb68f26d5bfeba2b]
aws_subnet.private-a: Refreshing state... [id=subnet-01d1969c004f12dbc]
aws_security_group.elb: Refreshing state... [id=sg-07f29efb6f0edc7f4]
aws_route_table.rtl: Refreshing state... [id=rtb-049d20f4b17481123]
aws_main_route_table_association.association-subnet: Refreshing state... [id=rtbassoc-0803a0bf5b70f8c45]
aws_instance.example-a: Creating...
aws_instance.example-b: Creating...
aws_instance.example-a: Still creating... [10s elapsed]
aws_instance.example-b: Still creating... [10s elapsed]
aws_instance.example-a: Still creating... [20s elapsed]
aws_instance.example-b: Still creating... [20s elapsed]
aws_instance.example-a: Still creating... [30s elapsed]
aws_instance.example-b: Still creating... [30s elapsed]
aws_instance.example-a: Creation complete after 31s [id=i-04508a1a92c3ddc15]
aws_instance.example-b: Still creating... [40s elapsed]
aws_instance.example-b: Creation complete after 41s [id=i-0d2b4d3e9d6def0f6]
aws_elb.example: Creating...
aws_elb.example: Creation complete after 2s [id=tf-elb-studentf5labdev]

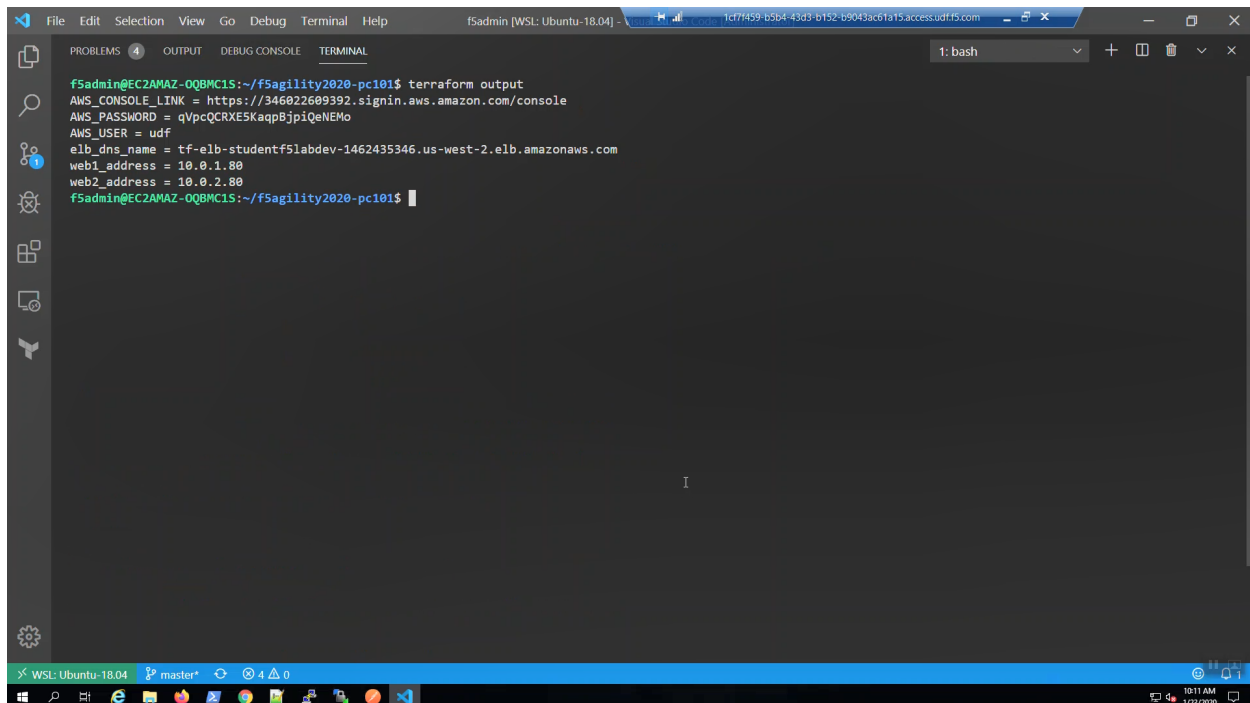
Apply complete! Resources: 3 added, 0 changed, 0 destroyed.

Outputs:

AWS_CONSOLE_LINK = https://346022609392.signin.aws.amazon.com/console
AWS_PASSWORD = qVpcQRXESKaqpBjpiQeNEMo
AWS_USER = udf
elb_dns_name = tf-elb-studentf5labdev-1462435346.us-west-2.elb.amazonaws.com
web1_address = 10.0.1.80
web2_address = 10.0.2.80
fsadmin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$
```

Review the terraform output when complete. You can always get the terraform output details again by invoking from the terminal:

```
terraform output
```



```
fsadmin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$ terraform output
AWS_CONSOLE_LINK = https://346022609392.signin.aws.amazon.com/console
AWS_PASSWORD = qVpcQRXESKaqpBjpiQeNEMo
AWS_USER = udf
elb_dns_name = tf-elb-studentf5labdev-1462435346.us-west-2.elb.amazonaws.com
web1_address = 10.0.1.80
web2_address = 10.0.2.80
fsadmin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$
```

1.2 Deploying an F5 CloudFormation Template to AWS

Deploy an F5 CloudFormation template to AWS using the supporting AWS infrastructure and services created in the previous lab.

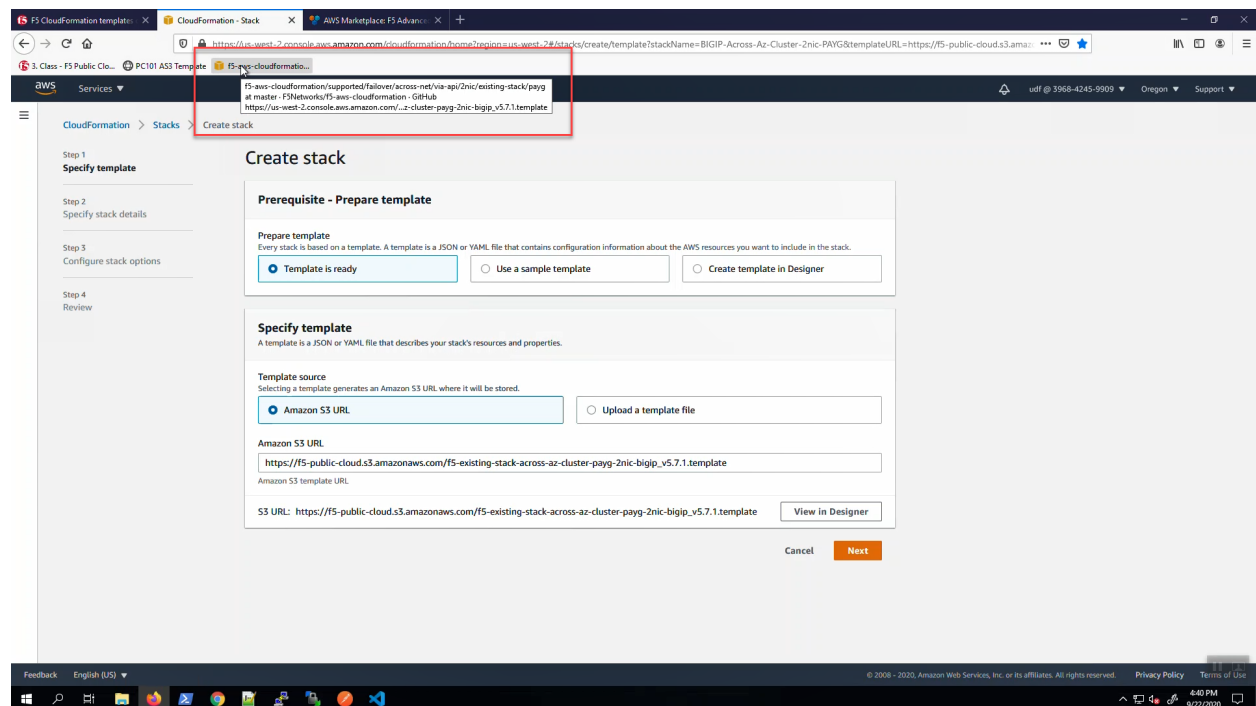
1.2.1 F5 CloudFormation templates on Github

Open a new tab in the web browser. Click on the “PC101 AS3 Template” URL bookmark in the bookmark toolbar. An AS3 Declaration is a single JSON-formatted file that defines your virtual server(s) and all accompanying configuration objects. Some interesting configuration objects are highlighted in the screenshot below:

- We use AWS tags to lookup the application pool members dynamically. You do not hardcode the IP addresses of pool members.
- An HTTP/2 profile is enabled.
- The Virtual Server Address is a “catch-all” wildcard address: 0.0.0.0/0.

Open another new tab in the web browser. Click on the “f5-aws-cloudformation” URL bookmark in the bookmark toolbar. Supported F5 CloudFormation templates are hosted on the official F5 GitHub repo: <https://github.com/F5Networks/f5-aws-cloudformation>.

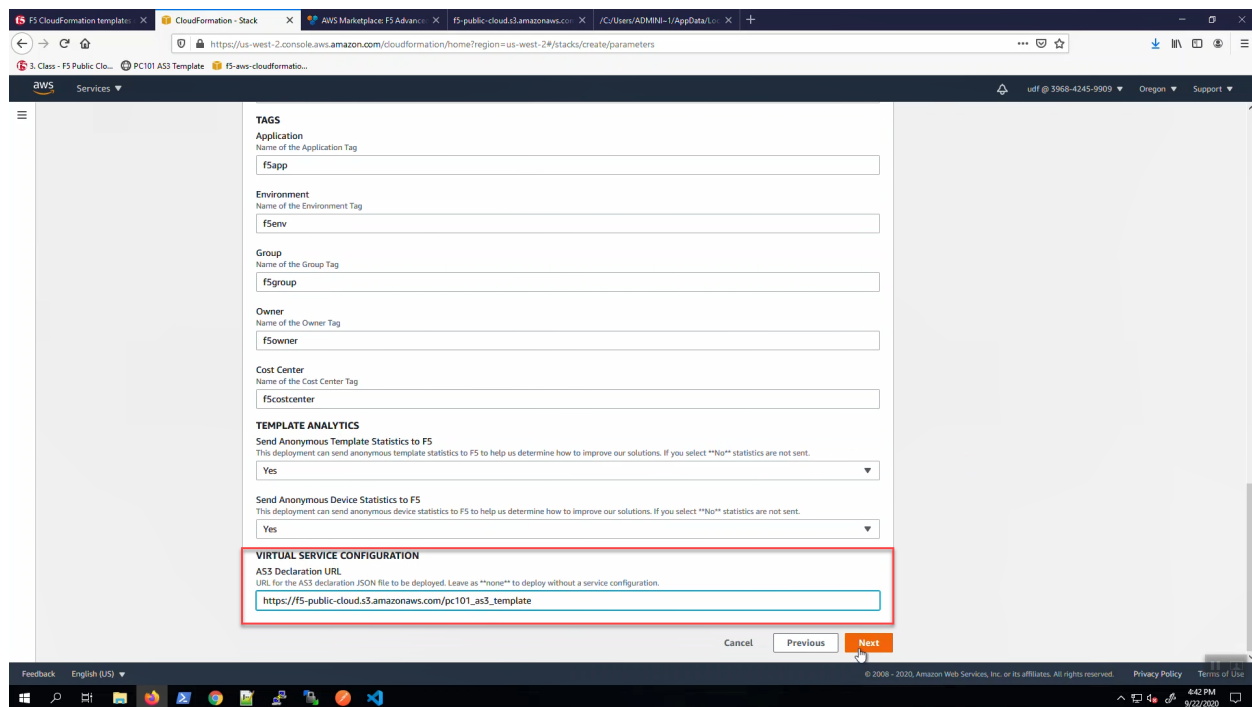
Scroll down and click on “Next”



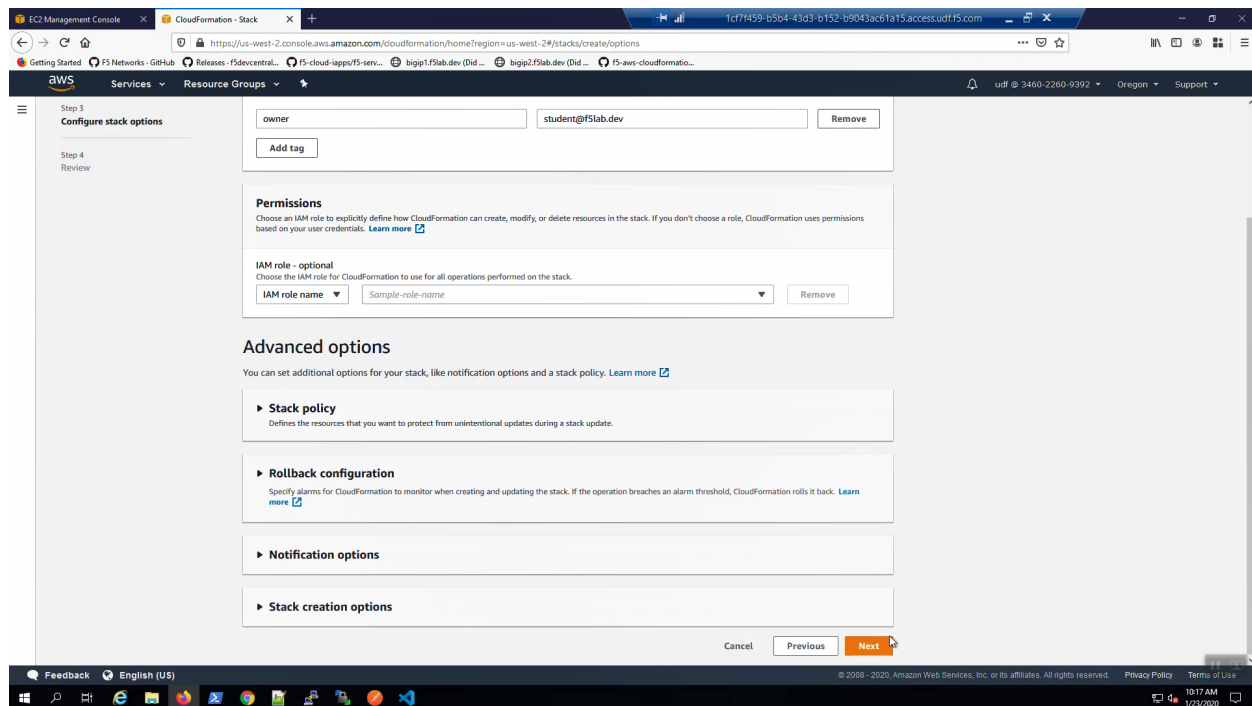
Fill in the CloudFormation template parameters. Leave defaults where not explicitly called out below.

| Parameter | Value |
|--------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| NETWORKING CONFIGURATION | |
| VPC | (10.0.0.0/16)(terraform_student@f5lab.dev) |
| Management Subnet AZ1 | (10.0.101.0/24)(management) |
| Management Subnet AZ2 | (10.0.102.0/24)(management) |
| Subnet1 in AZ1 | (10.0.1.0/24)(public) |
| Subnet1 in AZ2 | (10.0.2.0/24)(public) |
| INSTANCE CONFIGURATION | |
| BIG-IP Image Name | AdvancedWaf25Mbps |
| SSH Key | MyKeyPair-student@f5lab.dev |
| Source Address(es) for Management Access | 0.0.0.0/0 |
| Source Address(es) for Web Application Access (80/443) | 0.0.0.0/0 |
| NTP Server | 0.pool.ntp.org |
| Timezone (Olson) | UTC |
| BIG-IP Modules | ltm:nominal,avr:nominal |
| VIRTUAL SERVICE CONFIGURATION | |
| AS3 Declaration URL | https://f5-public-cloud.s3.amazonaws.com/pc101_as3_template |

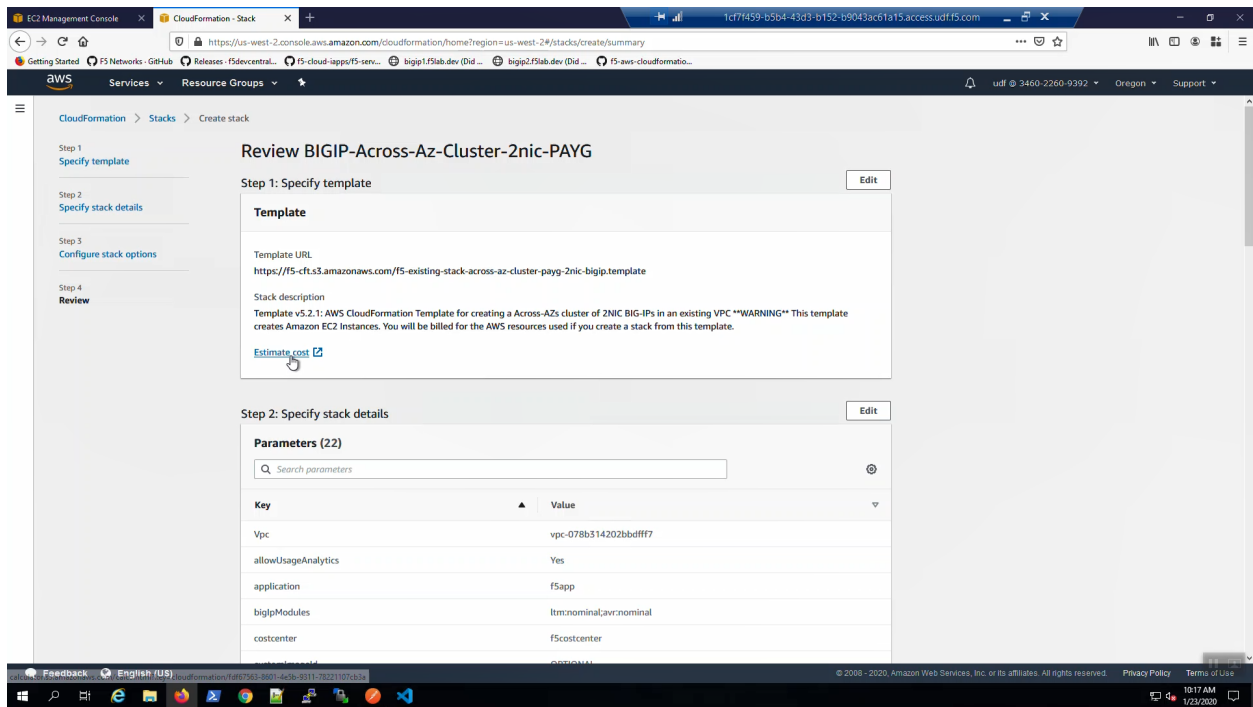
The screenshot shows the AWS CloudFormation console interface. The main heading is 'Specify stack details'. On the left, there is a sidebar with a list of steps: 'Step 1: Specify template', 'Step 2: Specify stack details' (which is the active step), 'Step 3: Configure stack options', and 'Step 4: Review'. The main content area is titled 'Specify stack details' and contains a form for configuring the stack. The form has a section for 'Stack name' with a text input field containing 'BIGIP-Across-Az-Cluster-Znic-PAYG'. Below this is a section for 'Parameters' with a note: 'Parameters are defined in your template and allow you to input custom values when you create or update a stack.' Under the 'Parameters' section, there is a sub-section titled 'NETWORKING CONFIGURATION'. This section contains five dropdown menus: 'VPC' (set to 'vpc-078b314202bdf7f7 (10.0.0.0/16) (terraform_student@f5lab.dev)'), 'Management Subnet AZ1' (set to 'subnet-0eb68f26d5bfeba2b (10.0.101.0/24) (management)'), 'Management Subnet AZ2' (set to 'subnet-01dbdb4b6f59fa65d (10.0.102.0/24) (management)'), 'Subnet1 in AZ1' (set to 'Public or External subnet'), and 'Subnet1 in AZ2' (set to 'Public or External subnet'). At the bottom of the parameters section, there is a checkbox labeled 'Provision Public IP addresses for the BIG-IP interfaces' with a note: 'Whether or not to provision Public IP Addresses for the BIG-IP Network interfaces. By Default no Public IP addresses are provisioned.' The console also shows a top navigation bar with 'AWS' logo, 'Services', 'Resource Groups', and a user profile 'udf @ 3460-2260-9392' in 'Oregon'. The bottom of the console shows a Windows taskbar with various application icons and a system clock showing '10:15 AM 1/23/2020'.



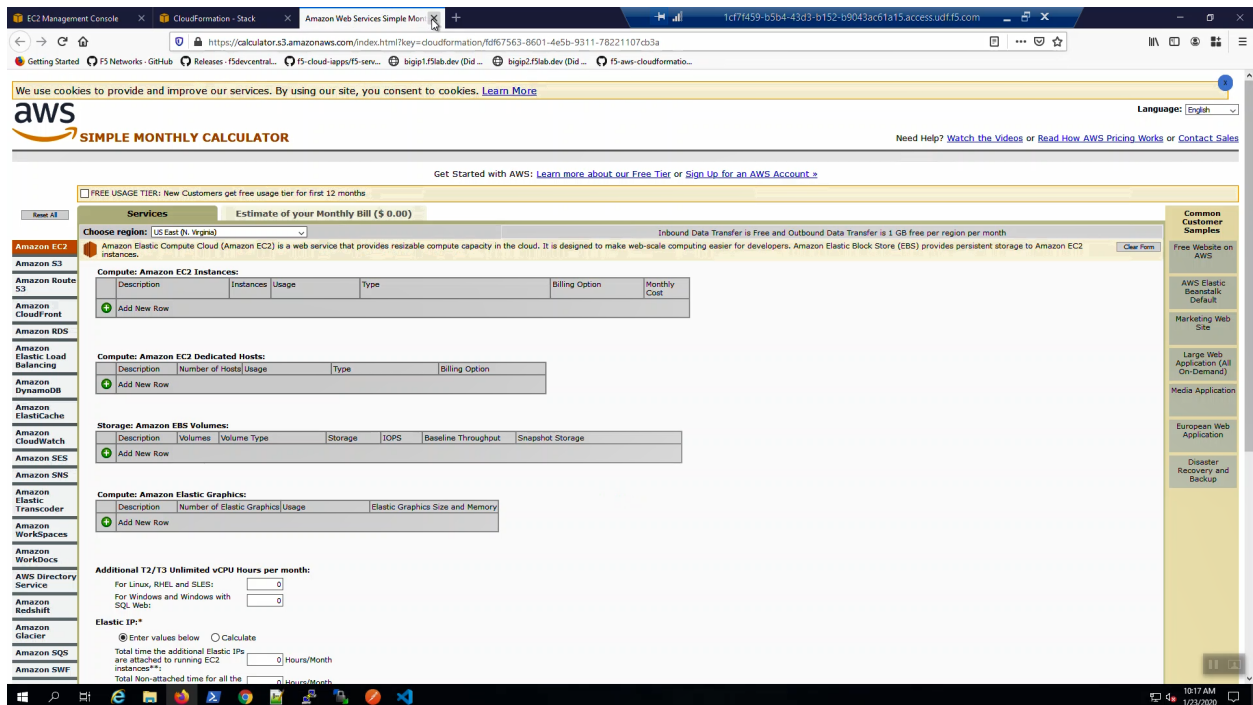
Scroll to the bottom of the CloudFormation template and click [Next].



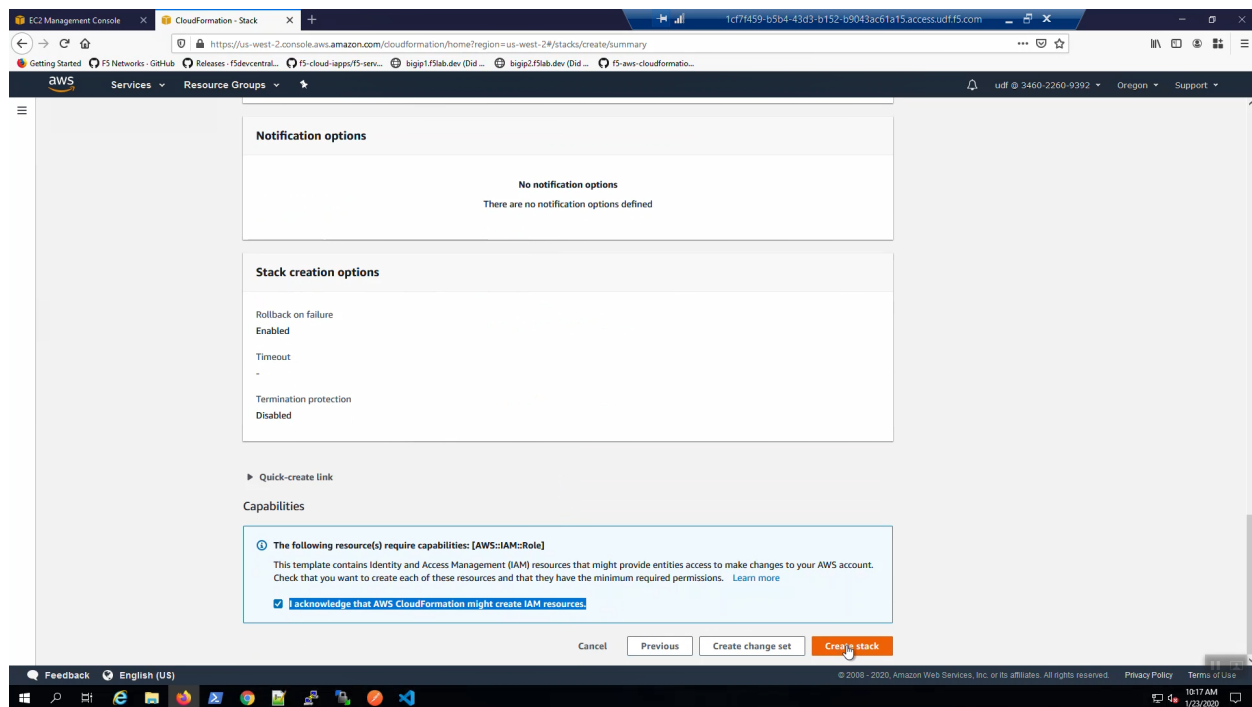
Click on “Estimate cost”.



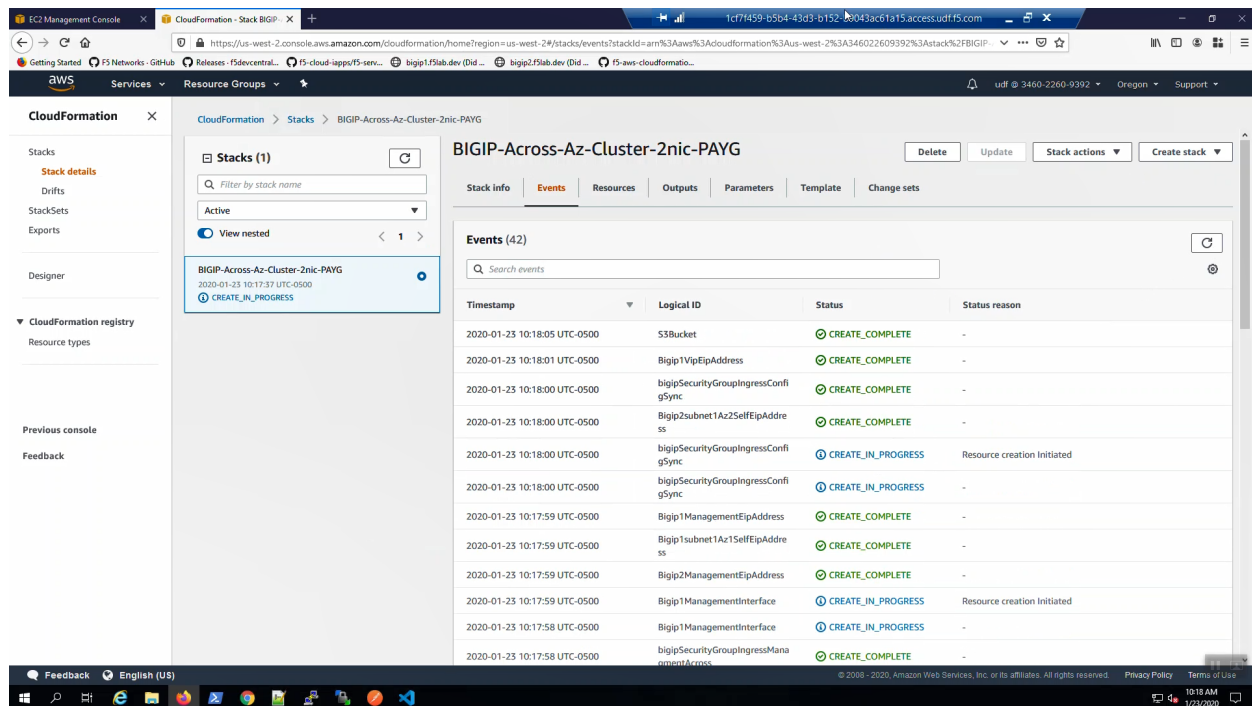
You can estimate the cost of running this CloudFormation stack with the AWS Simple Monthly Calculator. Review and close this screen.



Scroll to the bottom of the CloudFormation template. Acknowledge that AWS CloudFormation might create IAM resources and [Create stack].



Watch as your resources are built in AWS by refreshing the CloudFormation Events screen.

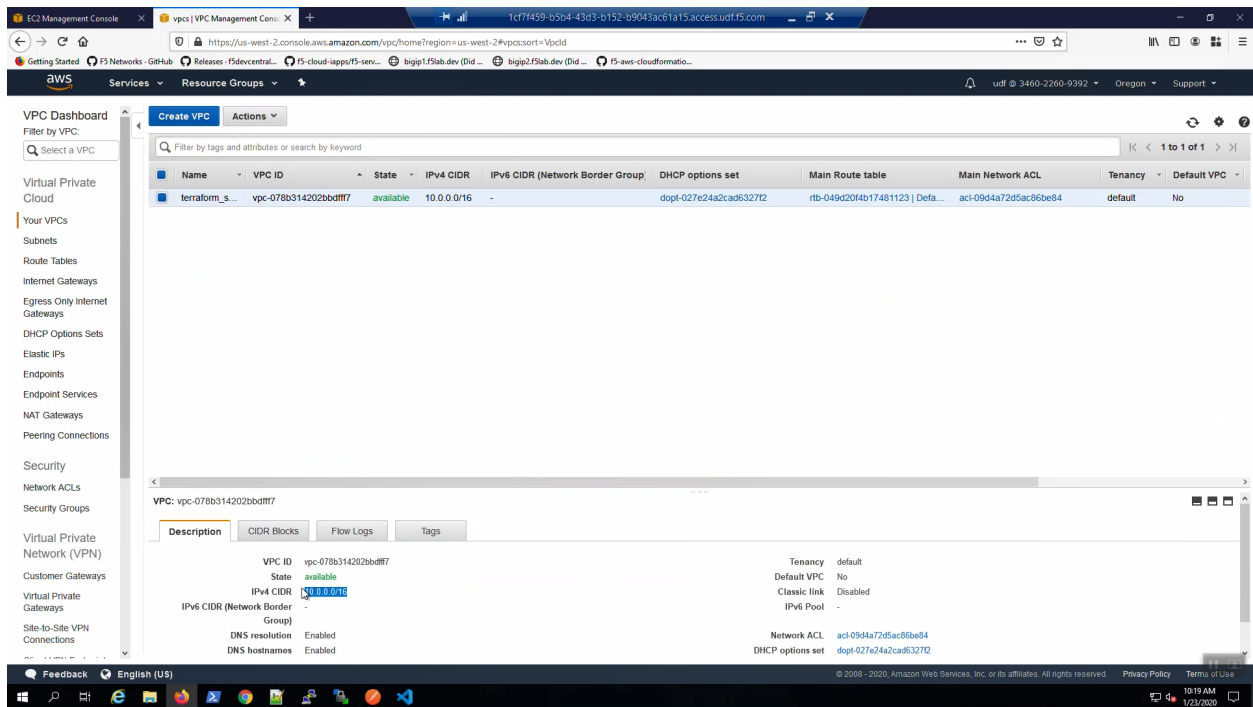


1.3 Explore the AWS Virtual Private Cloud

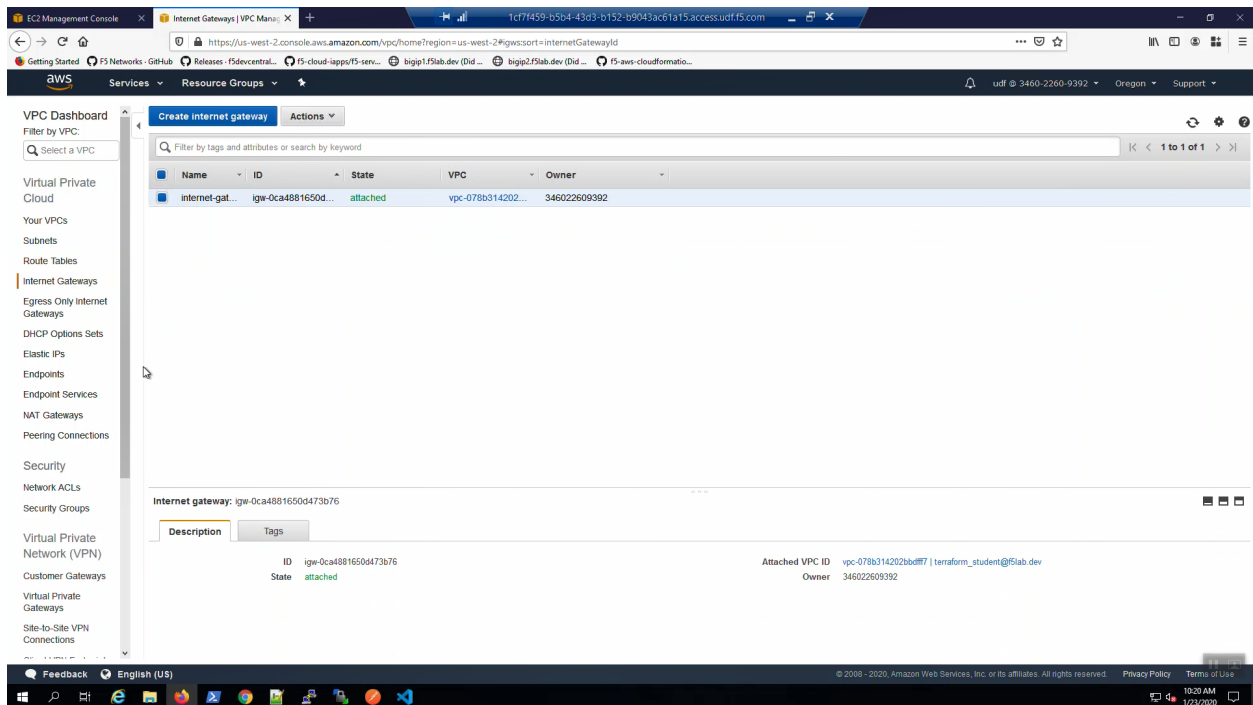
The F5 environment will be ready to protect our web applications shortly. In the meantime, let's explore the AWS Virtual Private Cloud we've created to familiarize ourselves with some basic AWS networking concepts.

1.3.1 Explore AWS Environment

Services => VPC => Your VPCs. A VPC (Virtual Private Cloud) was created by Terraform. All of our supporting infrastructure will run inside this VPC.



Services => VPC => Internet Gateways. An Internet Gateway was created to allow instances in our VPC to connect to the Internet.



Services => VPC => Subnets. Terraform created management and traffic subnets (both public and private)

across two Availability Zones: us-west-2a and us-west-2b.

The screenshot shows the AWS Management Console's VPC Dashboard. The 'Subnets' tab is active, displaying a table of subnets. The 'management' subnet in the us-west-2a Availability Zone is highlighted. Below the table, the details for this subnet are shown, including its ID, VPC, IP address range, and associated resources like the route table and network ACL.

| Name | Subnet ID | State | VPC | IPV4 CIDR | Available IPv4 | IPV6 CIDR | Availability Zone | Availability Zone ID | Network Border Gr | Route table |
|------------|--------------------------|-----------|-----------------------|---------------|----------------|-----------|-------------------|----------------------|-------------------|----------------------|
| private | subnet-01d1969c004f12dbc | available | vpc-078b314202bcb0ff7 | 10.0.100.0/24 | 251 | - | us-west-2a | usw2-az2 | us-west-2 | rtb-049d20f4b1748112 |
| management | subnet-01c0db4b6f59fa65d | available | vpc-078b314202bcb0ff7 | 10.0.102.0/24 | 250 | - | us-west-2b | usw2-az1 | us-west-2 | rtb-049d20f4b1748112 |
| public | subnet-073a1c490d0c007f1 | available | vpc-078b314202bcb0ff7 | 10.0.1.0/24 | 247 | - | us-west-2a | usw2-az2 | us-west-2 | rtb-049d20f4b1748112 |
| private | subnet-075282aa20b7c04a4 | available | vpc-078b314202bcb0ff7 | 10.0.200.0/24 | 251 | - | us-west-2b | usw2-az1 | us-west-2 | rtb-049d20f4b1748112 |
| public | subnet-0a3a60de4e624c322 | available | vpc-078b314202bcb0ff7 | 10.0.2.0/24 | 247 | - | us-west-2b | usw2-az1 | us-west-2 | rtb-049d20f4b1748112 |
| management | subnet-0eb6826d5bfeba2b | available | vpc-078b314202bcb0ff7 | 10.0.101.0/24 | 250 | - | us-west-2a | usw2-az2 | us-west-2 | rtb-049d20f4b1748112 |

Subnet: subnet-0eb6826d5bfeba2b

Description: Subnet ID: subnet-0eb6826d5bfeba2b, VPC: vpc-078b314202bcb0ff7, State: available, IPv4 CIDR: 10.0.101.0/24, IPv6 CIDR: -, Network Border Group: us-west-2, Network ACL: acl-09d4a7265ac86be84, Auto-assign public IPv4 address: Yes, Outpost ID: -

Services => VPC => Elastic IPs. Elastic IPs are public IP addresses assigned to the management and traffic interfaces of the F5 Virtual Editions deployed.

The screenshot shows the AWS Management Console's Elastic IP Management page. The 'Allocate new address' button is clicked, and the details for the newly allocated Elastic IP are shown. The Elastic IP is associated with the 'management' subnet in the us-west-2a Availability Zone.

| Name | Elastic IP | Allocation ID | Instance | Private IP address | Scope | Association ID | Network Border Group | Network interface ID | Network interface owner |
|------|---------------|----------------------|----------|--------------------|-------|---------------------|----------------------|-----------------------|-------------------------|
| | 34.215.219.51 | eipalloc-03a364f4... | - | 10.0.2.40 | vpc | eipassoc-0e1068... | us-west-2 | eni-0937b721d143bbf | 346022609392 |
| | 44.231.146.69 | eipalloc-050ad80... | - | 10.0.102.232 | vpc | eipassoc-045c185... | us-west-2 | eni-0311f0f632d5d4a23 | 346022609392 |
| | 44.231.46.113 | eipalloc-0e1fea11... | - | 10.0.1.33 | vpc | eipassoc-020021... | us-west-2 | eni-02182b1051ea226ee | 346022609392 |
| | 44.232.76.46 | eipalloc-01b5df06... | - | 10.0.101.169 | vpc | eipassoc-0d1066... | us-west-2 | eni-01125f0a020a1ef | 346022609392 |
| | 54.70.100.164 | eipalloc-0be518b... | - | 10.0.1.248 | vpc | eipassoc-08de2d... | us-west-2 | eni-02182b1051ea226ee | 346022609392 |

Address: 44.232.76.46

Description: Elastic IP: 44.232.76.46, Allocation ID: eipalloc-01b5df068b309e99, Address Pool: amazon, Private IP address: 10.0.101.169, Association ID: eipassoc-0d106675ab3e9f8a, Public DNS: -, Network interface owner: 346022609392

Services => VPC => Security Groups. Security Groups are stateful port filters applied to either instances or network interfaces.

The image displays two screenshots of the AWS Management Console, specifically the VPC Dashboard, showing the configuration of security groups for a VPC.

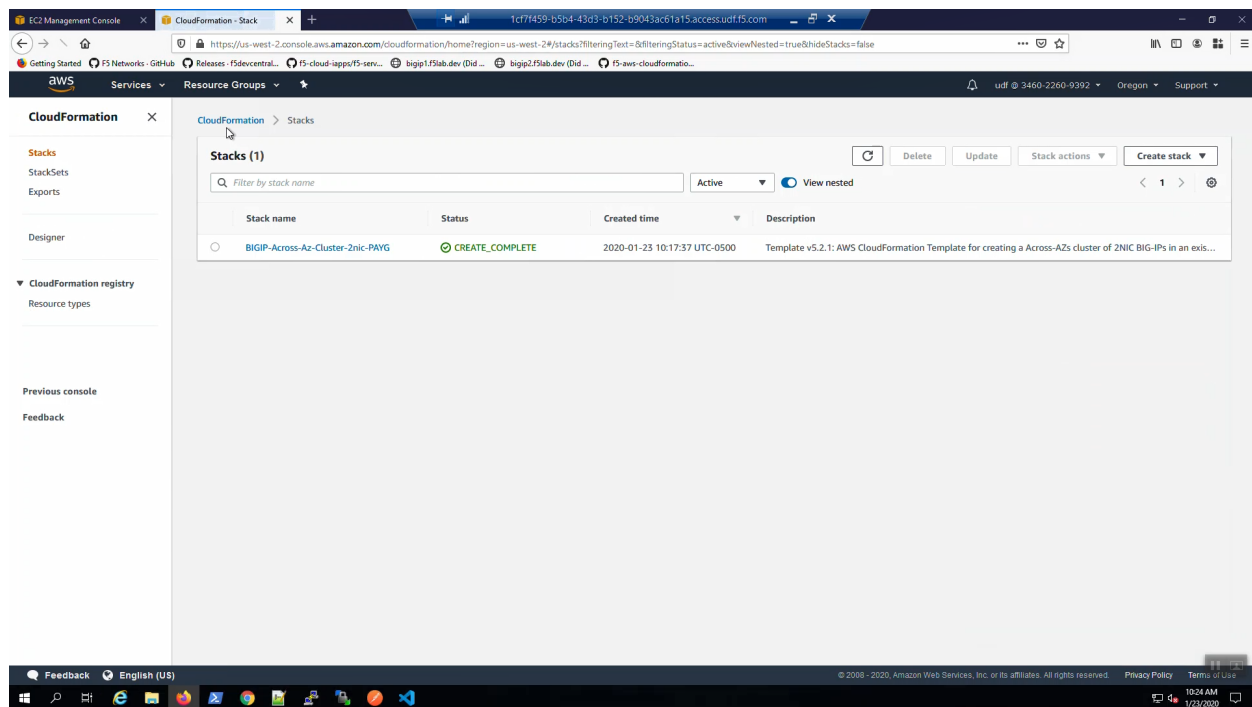
Top Screenshot: The 'Inbound Rules' tab is selected for the security group **sg-056de270316ea8bc1**. The rules table shows the following configuration:

| Type | Protocol | Port Range | Source | Description |
|-----------------|----------|------------|----------------------|-------------|
| HTTP | TCP | 80 | 0.0.0.0/0 | |
| Custom TCP Rule | TCP | 4353 | sg-056de270316ea8bc1 | |
| Custom UDP Rule | UDP | 1026 | sg-056de270316ea8bc1 | |
| HTTPS | TCP | 443 | 0.0.0.0/0 | |

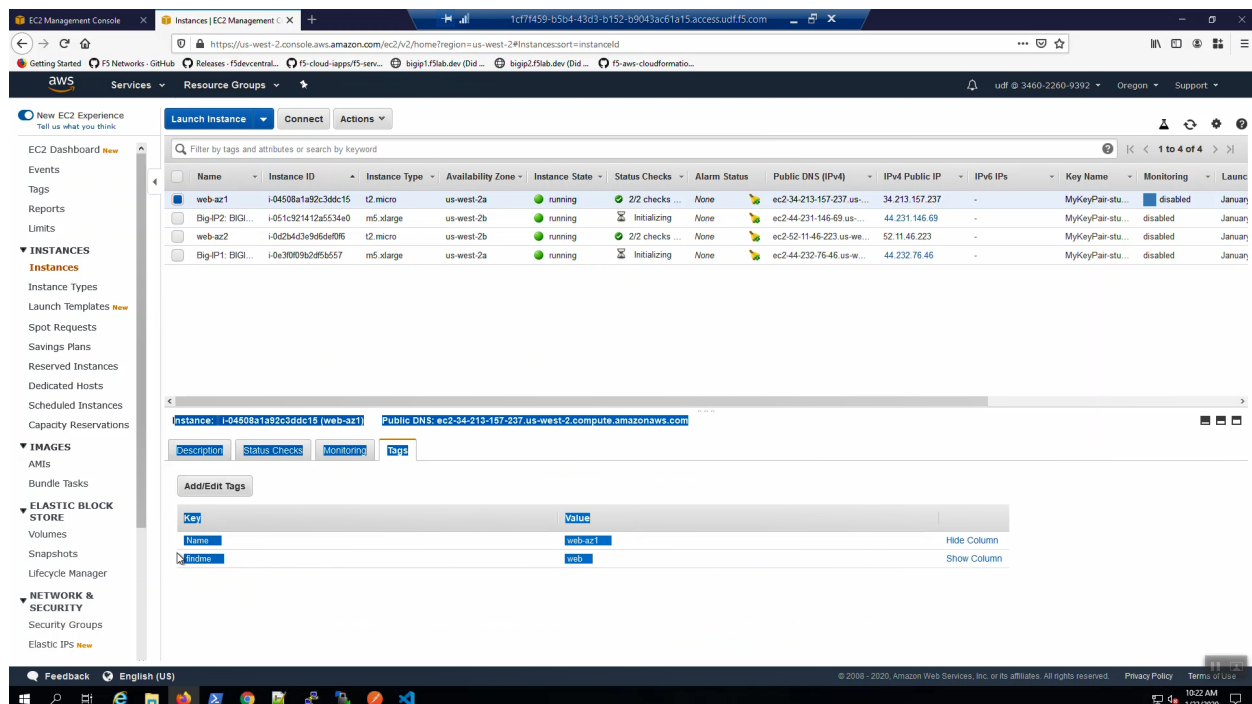
Bottom Screenshot: The 'Inbound Rules' tab is selected for the security group **sg-04ce53b880cd526c2**. The rules table shows the following configuration:

| Type | Protocol | Port Range | Source | Description |
|-------|----------|------------|----------------------|-------------|
| SSH | TCP | 22 | 0.0.0.0/0 | |
| HTTPS | TCP | 443 | 0.0.0.0/0 | |
| HTTPS | TCP | 443 | sg-056de270316ea8bc1 | |

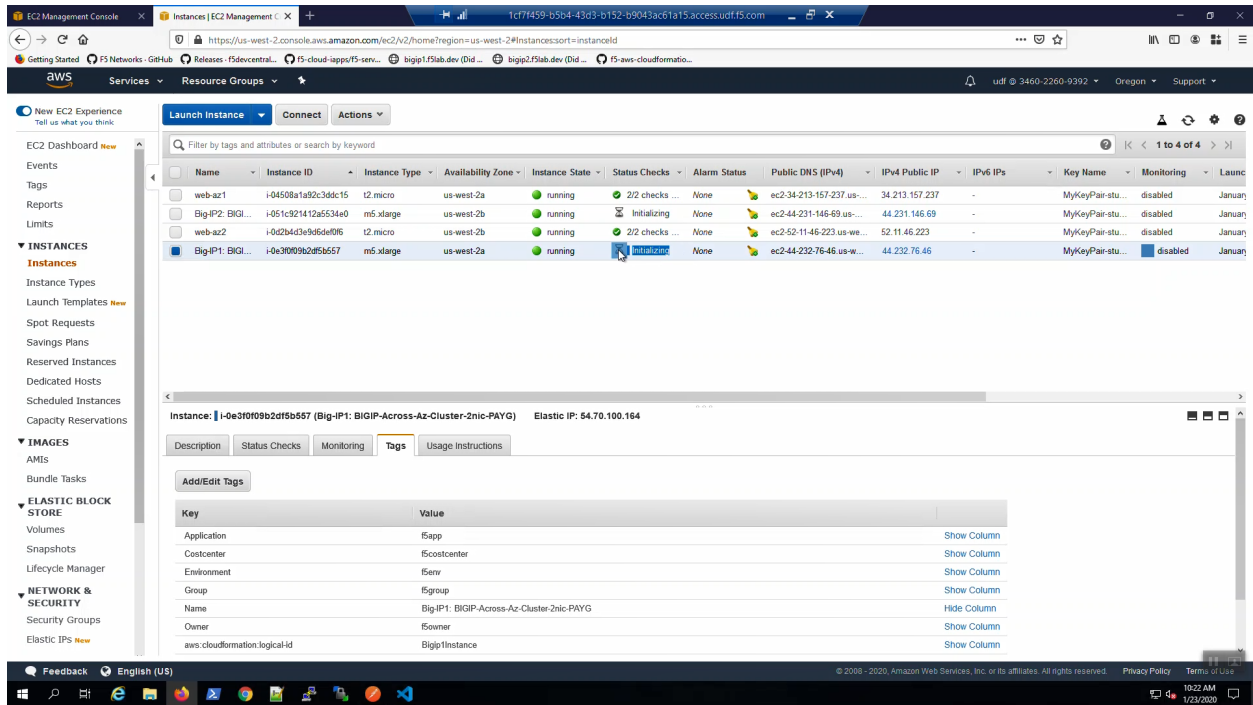
Services => CloudFormation. The CloudFormation Stack "BIGIP-Across-Az-Cluster-2nic-PAYG" will quickly show status of "CREATE_COMPLETE"



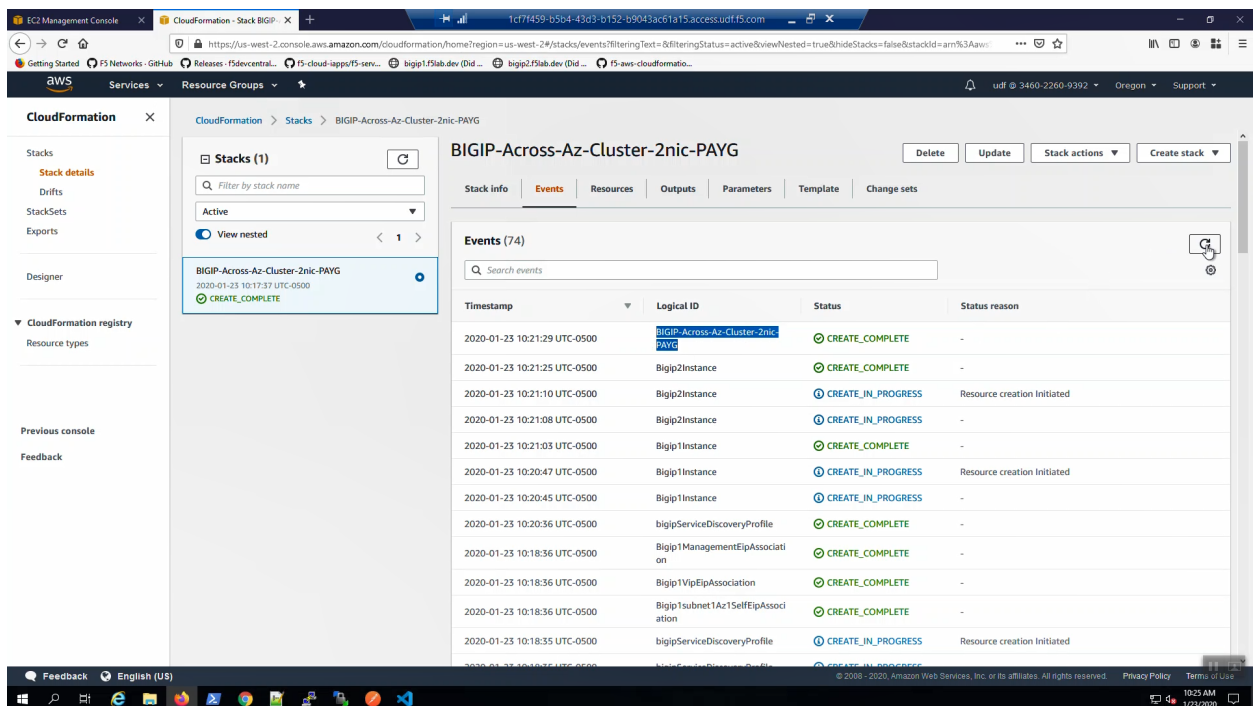
Services => EC2 => Instances. Two example NGINX web servers were deployed: web-az1 and web-az2. The “findme : web” tag will be used later in the lab so that our F5 Virtual Edition can auto-discover its pool members.



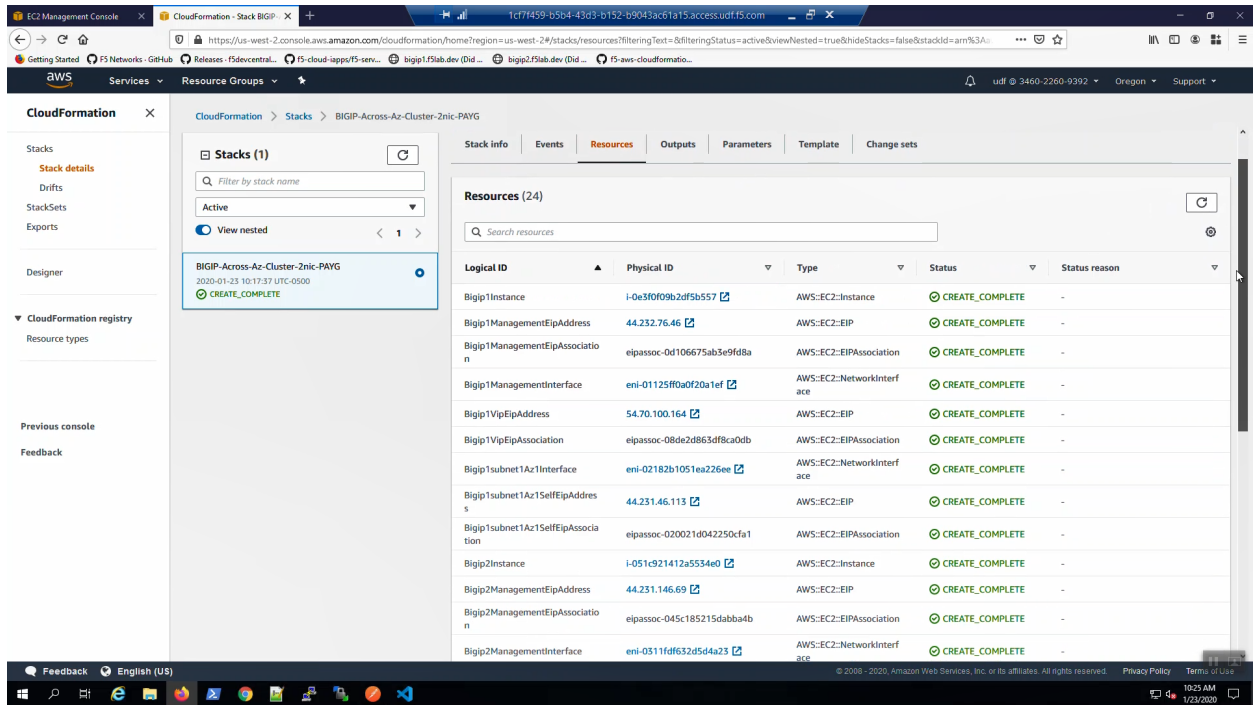
The “Big-IP1” and “Big-IP2” F5 instances are still “initializing” and will come online shortly.



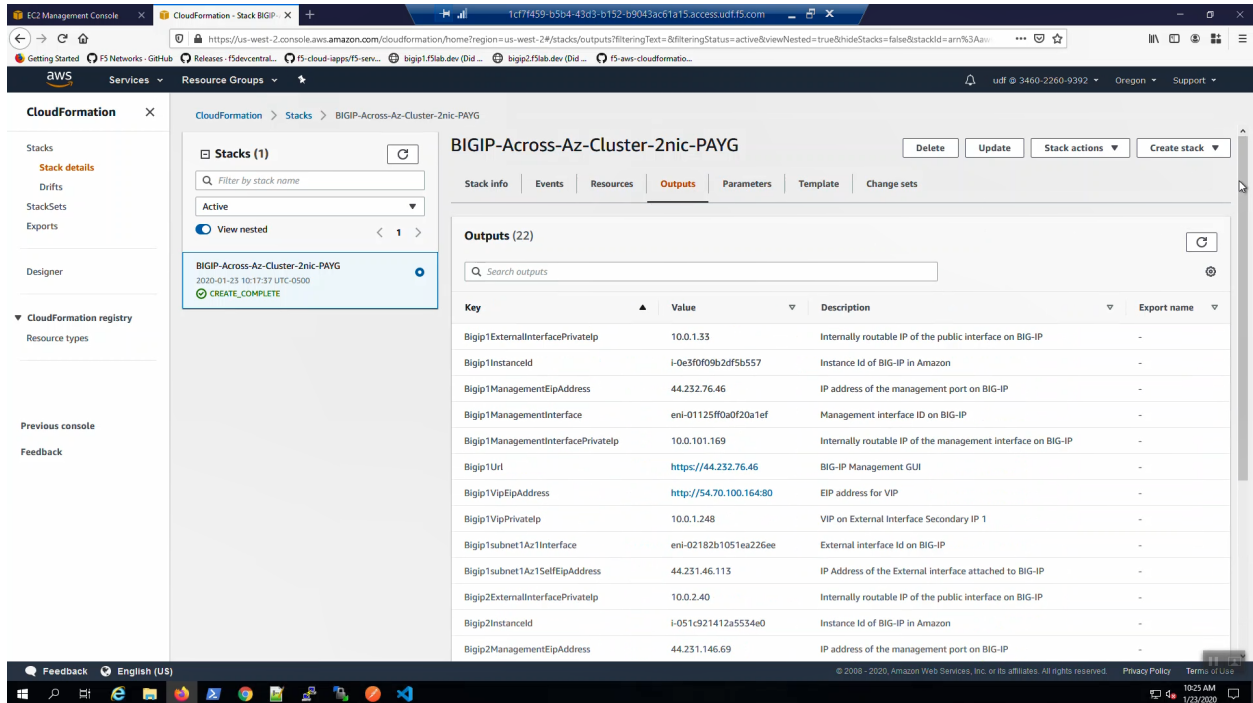
Services => CloudFormation => Stacks => "BIGIP-Across-Az-Cluster-2nic-PAYG" => Events. When you deploy via CloudFormation template, all of the steps to build the F5 in AWS environment are recorded as CloudFormation Events.



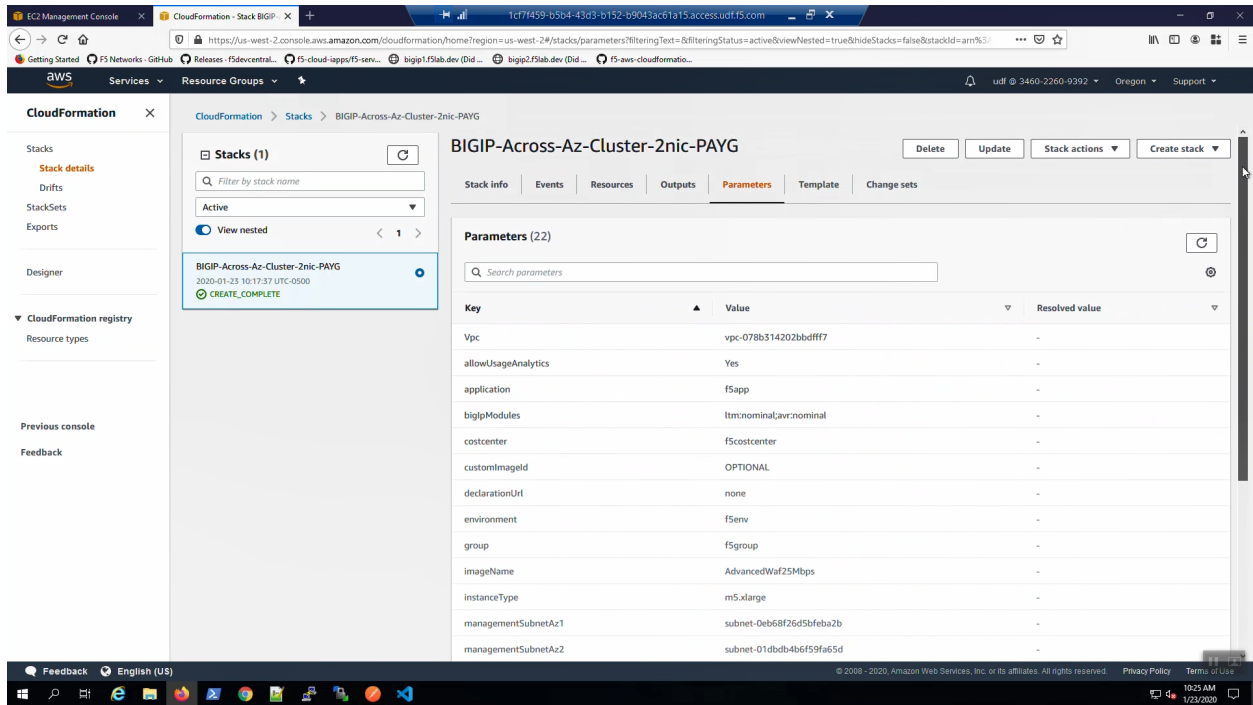
The CloudFormation template created a pair of F5 Virtual Edition instances and many other supporting resources. All of the resources created by CloudFormation are recorded here.



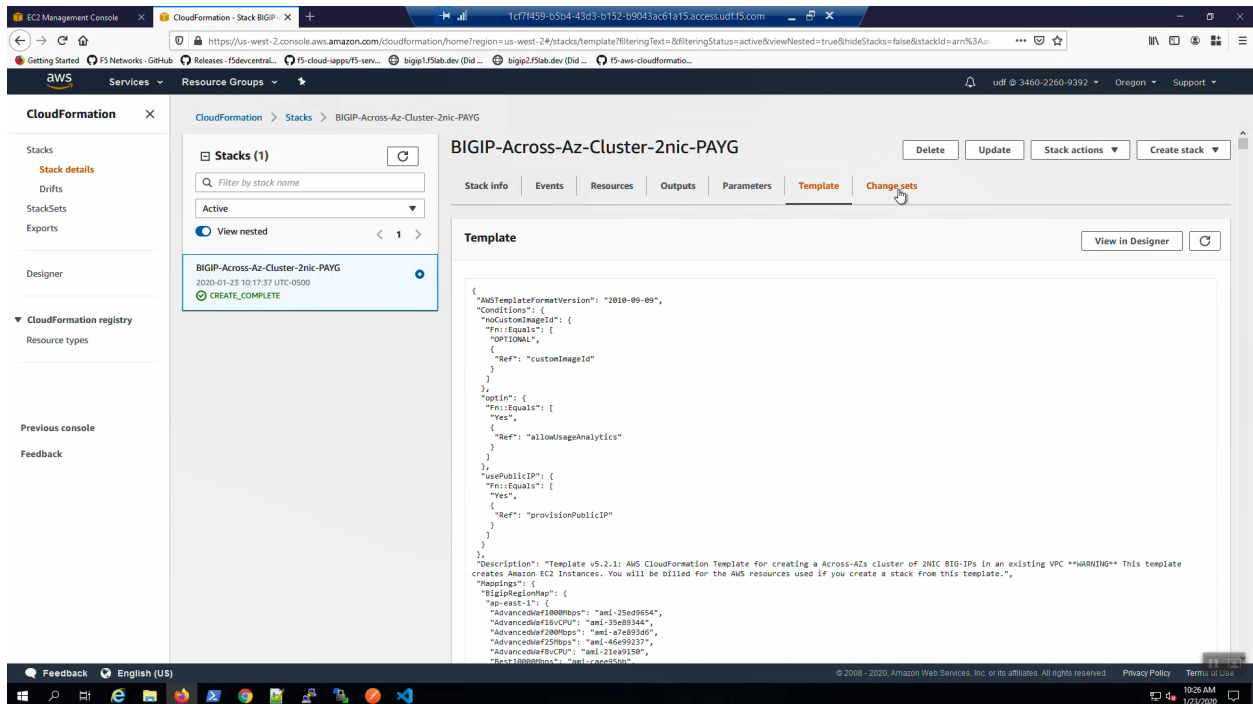
The CloudFormation stack will compute output attributes and report them here. What's the public management IP addresses URL for Big-IP1? What is the Virtual Server Elastic IP address? These sorts of questions can be answered by querying the CloudFormation stack outputs. We make use of this useful convenience in a future lab section.



CloudFormation parameters will report on the values fed into the CloudFormation template. Look here to confirm a value or when you suspect an error/typo and want to confirm.



The CloudFormation template can be customized when you need to deviate from the default behavior.



1.4 F5 Admin SSH and Configuration Utility (WebUI) Access

Configure admin SSH and Configuration Utility (WebUI) access to the F5 Virtual Editions. Retrieve information about our deployment using CloudFormation outputs.

1.4.1 Check status of F5 instances before proceeding.

From the AWS Web Console => Services => EC2 => INSTANCES => Instances. Select the Big-IP1... Select the “Status Check” tab below.

The screenshot shows the AWS Management Console interface. The left sidebar contains navigation links for EC2 Dashboard, Events, Tags, Reports, Limits, INSTANCES, IMAGES, ELASTIC BLOCK STORE, and NETWORK & SECURITY. The main content area displays a table of EC2 instances. The instance 'Big-IP1' is selected, and the 'Status Checks' tab is active. The instance is in a 'running' state with '2/2 checks' passed. The instance details show the name 'Big-IP1', instance ID 'i-0e3f0f09b2d5b657', instance type 'm5.xlarge', availability zone 'us-west-2a', and public IP address '44.232.76.46'.

| Name | Instance ID | Instance Type | Availability Zone | Instance State | Status Checks | Alarm Status | Public DNS (IPv4) | IPv4 Public IP | IPv6 IPs | Key Name | Monitoring | Launch |
|------------------|---------------------|---------------|-------------------|----------------|----------------|--------------|---------------------------|----------------|----------|------------------|------------|-----------|
| web-az1 | i-04508a1a50c3d6c15 | t2.micro | us-west-2a | running | 2/2 checks ... | None | ec2-34-213-157-237.us-... | 34.213.157.237 | - | MyKeyPair-stu... | disabled | Januar... |
| Big-IP2: BIGI... | i-051c921412a5534e0 | m5.xlarge | us-west-2b | running | 2/2 checks ... | None | ec2-44-231-146-69.us-... | 44.231.146.69 | - | MyKeyPair-stu... | disabled | Januar... |
| web-az2 | i-0d2b43e9b6de0f06 | t2.micro | us-west-2b | running | 2/2 checks ... | None | ec2-52-11-46-223.us-we... | 52.11.46.223 | - | MyKeyPair-stu... | disabled | Januar... |
| Big-IP1: BIGI... | i-0e3f0f09b2d5b657 | m5.xlarge | us-west-2a | running | 2/2 checks ... | None | ec2-44-232-76-46.us-w... | 44.232.76.46 | - | MyKeyPair-stu... | disabled | Januar... |

From the AWS Web Console => Services => EC2 => INSTANCES => Instances. Select the Big-IP2... Select the “Status Check” tab below.

The screenshot shows the AWS Management Console interface. The left sidebar contains navigation links for EC2 Dashboard, Events, Tags, Reports, Limits, INSTANCES, IMAGES, ELASTIC BLOCK STORE, and NETWORK & SECURITY. The main content area displays a table of EC2 instances. The instance 'Big-IP2' is selected, and the 'Status Checks' tab is active. The instance is in a 'running' state with '2/2 checks' passed. The instance details show the name 'Big-IP2', instance ID 'i-051c921412a5534e0', instance type 'm5.xlarge', availability zone 'us-west-2b', and public IP address '44.231.146.69'.

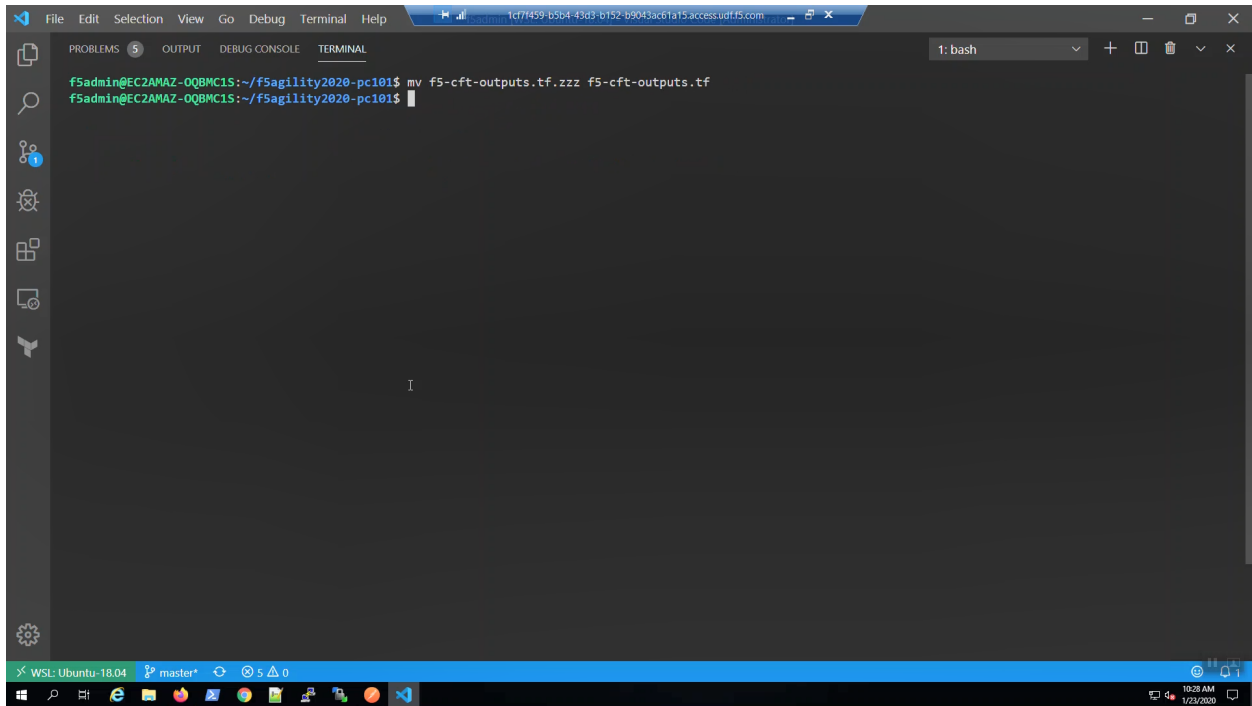
| Name | Instance ID | Instance Type | Availability Zone | Instance State | Status Checks | Alarm Status | Public DNS (IPv4) | IPv4 Public IP | IPv6 IPs | Key Name | Monitoring | Launch |
|------------------|---------------------|---------------|-------------------|----------------|----------------|--------------|---------------------------|----------------|----------|------------------|------------|-----------|
| web-az1 | i-04508a1a50c3d6c15 | t2.micro | us-west-2a | running | 2/2 checks ... | None | ec2-34-213-157-237.us-... | 34.213.157.237 | - | MyKeyPair-stu... | disabled | Januar... |
| Big-IP2: BIGI... | i-051c921412a5534e0 | m5.xlarge | us-west-2b | running | 2/2 checks ... | None | ec2-44-231-146-69.us-... | 44.231.146.69 | - | MyKeyPair-stu... | disabled | Januar... |
| web-az2 | i-0d2b43e9b6de0f06 | t2.micro | us-west-2b | running | 2/2 checks ... | None | ec2-52-11-46-223.us-we... | 52.11.46.223 | - | MyKeyPair-stu... | disabled | Januar... |
| Big-IP1: BIGI... | i-0e3f0f09b2d5b657 | m5.xlarge | us-west-2a | running | 2/2 checks ... | None | ec2-44-232-76-46.us-w... | 44.232.76.46 | - | MyKeyPair-stu... | disabled | Januar... |

Attention: Ensure both the System Status Checks and the Instance Status Checks have passed (green font) before proceeding.

1.4.2 Create terraform data resource to read CloudFormation outputs.

From the Visual Studio Code Terminal, enable the f5-cft-outputs.tf terraform file.

```
mv f5-cft-outputs.tf.zzz f5-cft-outputs.tf
```

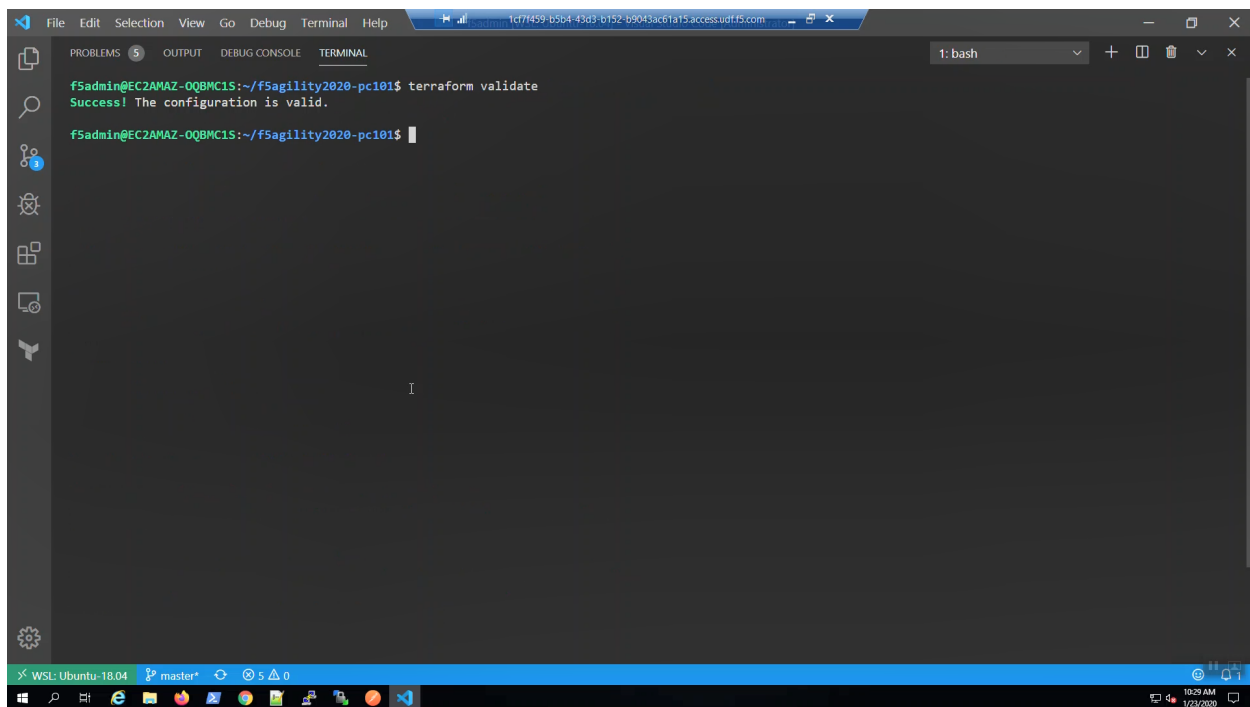


terraform init

```
terraform init
```

terraform validate

```
terraform validate
```

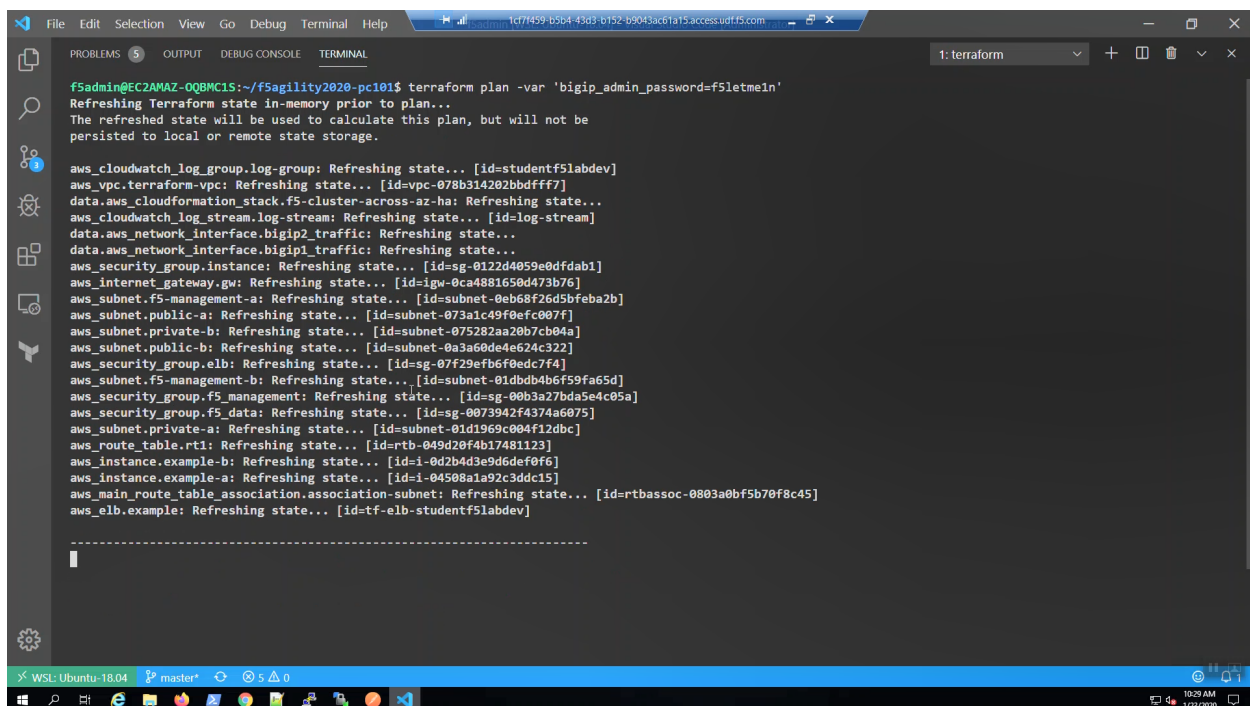



```
f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$ terraform validate
Success! The configuration is valid.

f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$
```

terraform plan

```
terraform plan -var 'bigip_admin_password=f5letmein'
```



```
f5admin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$ terraform plan -var 'bigip_admin_password=f5letmein'
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

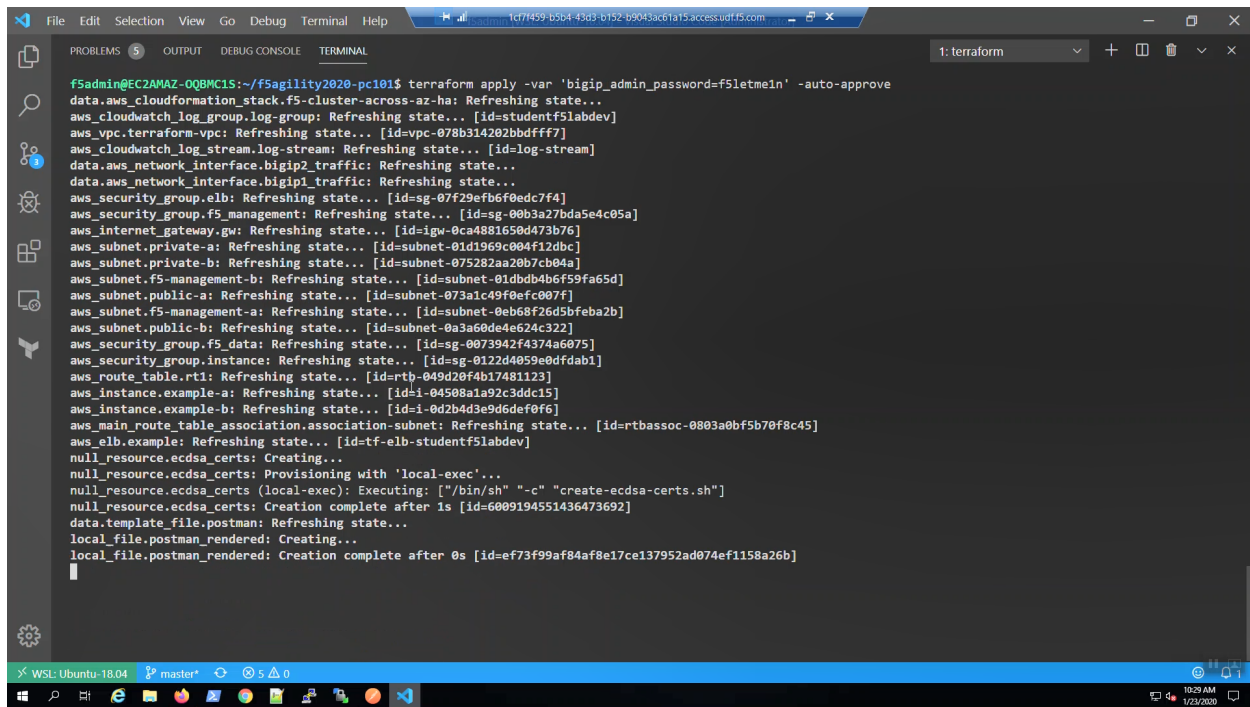
aws_cloudwatch_log_group.log-group: Refreshing state... [id=studentf5labdev]
aws_vpc.terraform-vpc: Refreshing state... [id=vpc-078b314202bdf7f7]
data.aws_cloudformation_stack.f5-cluster-across-az-ha: Refreshing state...
aws_cloudwatch_log_stream.log-stream: Refreshing state... [id=log-stream]
data.aws_network_interface.bigip2_traffic: Refreshing state...
data.aws_network_interface.bigip1_traffic: Refreshing state...
aws_security_group.instance: Refreshing state... [id=sg-0122d4059e0dfdb1]
aws_internet_gateway.gw: Refreshing state... [id=igw-0ca4881650d473b76]
aws_subnet.f5-management-a: Refreshing state... [id=subnet-0eb68f26d5bfeba2b]
aws_subnet.public-a: Refreshing state... [id=subnet-073a1c49f0efc007f]
aws_subnet.private-b: Refreshing state... [id=subnet-075282aa20b7cb04a]
aws_subnet.public-b: Refreshing state... [id=subnet-0a3a60de4e624c322]
aws_security_group.elb: Refreshing state... [id=sg-07f29efb6f0edc7f4]
aws_subnet.f5-management-b: Refreshing state... [id=subnet-01dbdb4b6f59fa65d]
aws_security_group.f5_management: Refreshing state... [id=sg-00b3a27bda5e4c05a]
aws_security_group.f5_data: Refreshing state... [id=sg-0073942f4374a6075]
aws_subnet.private-a: Refreshing state... [id=subnet-01d1969c004f12dbc]
aws_route_table.rtl: Refreshing state... [id=rtb-049d20f4b17481123]
aws_instance.example-b: Refreshing state... [id=i-0d2b4d3e9d6def0f6]
aws_instance.example-a: Refreshing state... [id=i-04508a1a92c3ddc15]
aws_main_route_table_association.association-subnet: Refreshing state... [id=rtbassoc-0803a0bf5b70f8c45]
aws_elb.example: Refreshing state... [id=tf-elb-studentf5labdev]

-----

```

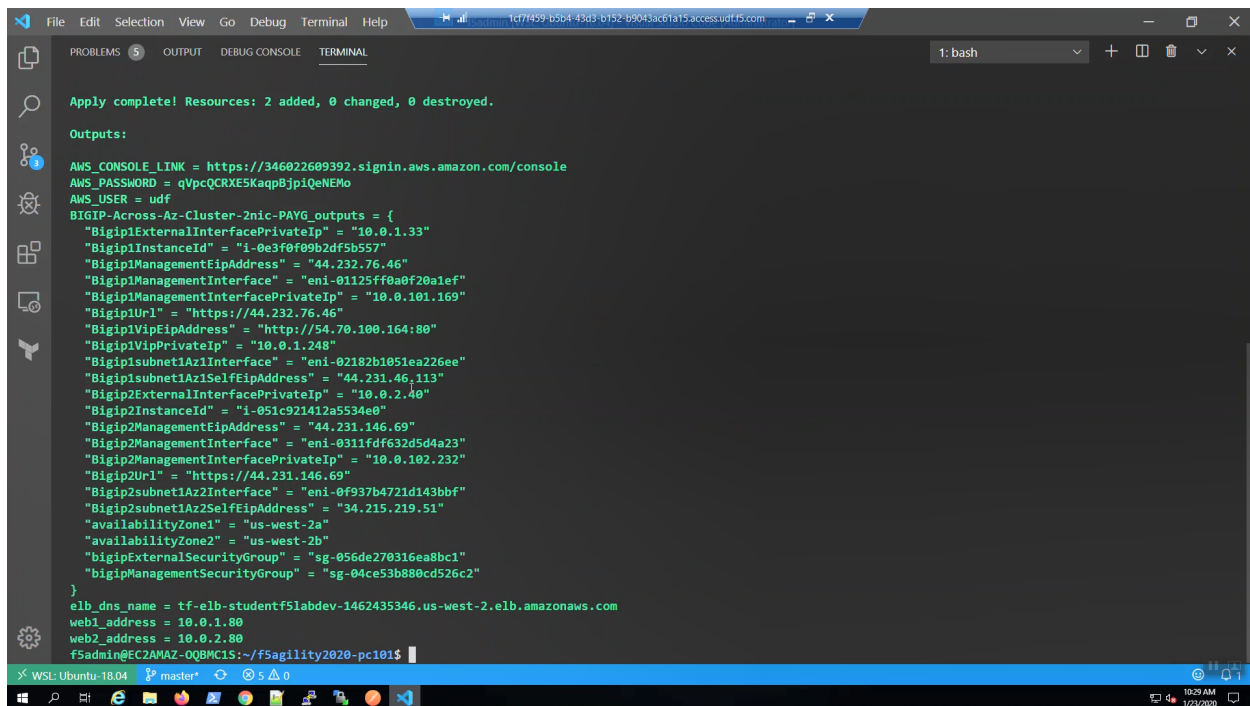
terraform apply

```
terraform apply -var 'bigip_admin_password=f5letmein' -auto-approve
```



```
f5admin@EC2AMAZ-Q08MC1S:~/f5agility2020-pc101$ terraform apply -var 'bigip_admin_password=f5letmein' -auto-approve
data.aws_cloudformation_stack.f5-cluster-across-az-ha: Refreshing state... [id=studentf5labdev]
aws_cloudwatch_log_group.log-group: Refreshing state... [id=studentf5labdev]
aws_vpc.terraform-vpc: Refreshing state... [id=vpc-078b314202bdfbf7]
aws_cloudwatch_log_stream.log-stream: Refreshing state... [id=log-stream]
data.aws_network_interface.bigip2_traffic: Refreshing state...
data.aws_network_interface.bigip1_traffic: Refreshing state...
aws_security_group.elb: Refreshing state... [id=sg-07f29efb6f8edc7f4]
aws_security_group.f5_management: Refreshing state... [id=sg-00b3a27bda5e4c05a]
aws_internet_gateway.gw: Refreshing state... [id=igw-0ca4881650d473b76]
aws_subnet.private-a: Refreshing state... [id=subnet-01d1969c004f12dbc]
aws_subnet.private-b: Refreshing state... [id=subnet-075282aa20b7cb04a]
aws_subnet.f5_management-b: Refreshing state... [id=subnet-01dbdb4b6f59fa65d]
aws_subnet.public-a: Refreshing state... [id=subnet-073a1c49f0efc007f]
aws_subnet.f5_management-a: Refreshing state... [id=subnet-0eb68f26d5bfeba2b]
aws_subnet.public-b: Refreshing state... [id=subnet-0a3a60de4e624c322]
aws_security_group.f5_data: Refreshing state... [id=sg-0073942f4374a6075]
aws_security_group.instance: Refreshing state... [id=sg-0122d4059e0d0fdab1]
aws_route_table.rtl: Refreshing state... [id=rtb-040d20f4b17481123]
aws_instance.example-a: Refreshing state... [id=i-04508a1a92c3ddc15]
aws_instance.example-b: Refreshing state... [id=i-0d2b4d3e9d6def0f6]
aws_main_route_table_association.association-subnet: Refreshing state... [id=rtbassoc-0803a0bf5b70f8c45]
aws_elb.example: Refreshing state... [id=tf-elb-studentf5labdev]
null_resource.ecdsa_certs: Creating...
null_resource.ecdsa_certs: Provisioning with 'local-exec'...
null_resource.ecdsa_certs (local-exec): Executing: ["/bin/sh" "-c" "create-ecdsa-certs.sh"]
null_resource.ecdsa_certs: Creation complete after 1s [id=6009194551436473692]
data.template_file.postman: Refreshing state...
local_file.postman_rendered: Creating...
local_file.postman_rendered: Creation complete after 0s [id=ef73f99af84af8e17ce137952ad074ef1158a26b]
```

terraform apply completed.



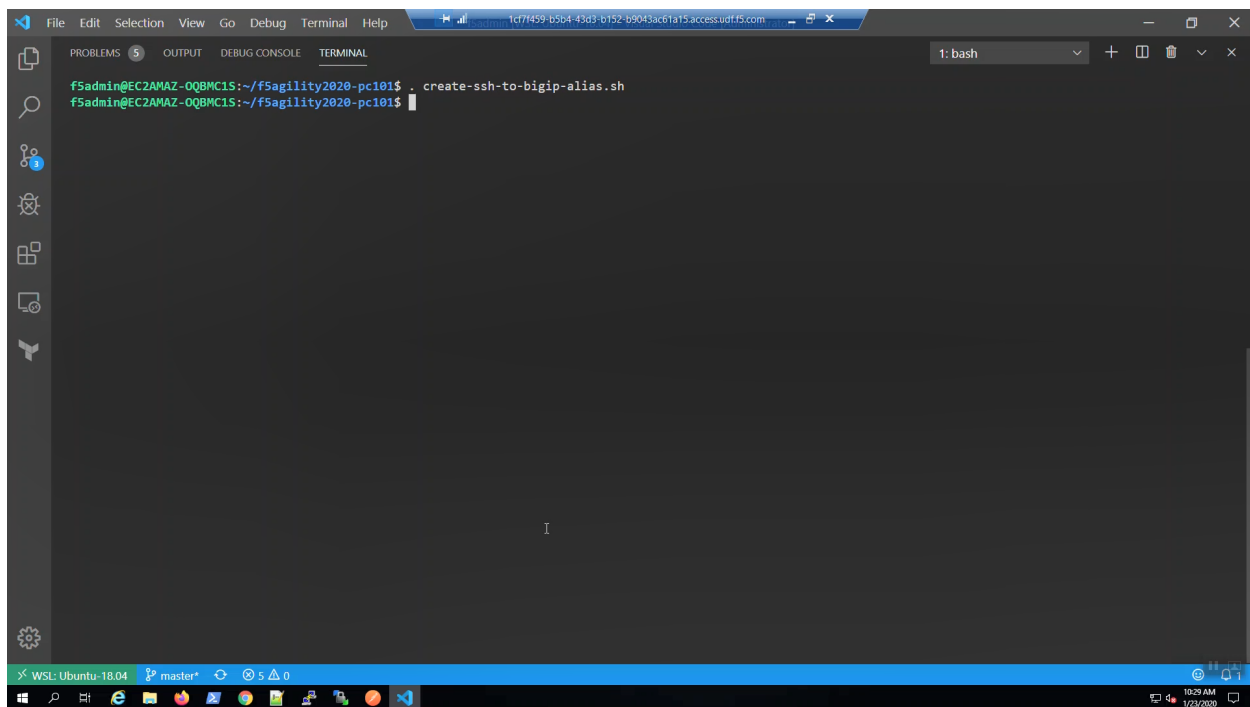
```
Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

Outputs:
AWS_CONSOLE_LINK = https://346022609392.signin.aws.amazon.com/console
AWS_PASSWORD = qVpcQCRXESkaqpBjpiQeNEMo
AWS_USER = udf
BIGIP-Across-Az-Cluster-2nic-PAYG_outputs = {
  "Bigip1ExternalInterfacePrivateIp" = "10.0.1.33"
  "Bigip1InstanceId" = "i-0e3f0f09b2df5b557"
  "Bigip1ManagementEipAddress" = "44.232.76.46"
  "Bigip1ManagementInterface" = "eni-01125ff0a0f20a1ef"
  "Bigip1ManagementInterfacePrivateIp" = "10.0.101.169"
  "Bigip1Url" = "https://44.232.76.46"
  "Bigip1VipEipAddress" = "http://54.70.100.164:80"
  "Bigip1VipPrivateIp" = "10.0.1.248"
  "Bigip1subnet1Az1Interface" = "eni-02182b1051ea226ee"
  "Bigip1subnet1Az1SelfEipAddress" = "44.231.46.113"
  "Bigip2ExternalInterfacePrivateIp" = "10.0.2.40"
  "Bigip2InstanceId" = "i-051c921412a5534e0"
  "Bigip2ManagementEipAddress" = "44.231.146.69"
  "Bigip2ManagementInterface" = "eni-0311fd6f632d5d4a23"
  "Bigip2ManagementInterfacePrivateIp" = "10.0.102.232"
  "Bigip2Url" = "https://44.231.146.69"
  "Bigip2subnet1Az2Interface" = "eni-0f937b4721d143bbf"
  "Bigip2subnet1Az2SelfEipAddress" = "34.215.219.51"
  "availabilityZone1" = "us-west-2a"
  "availabilityZone2" = "us-west-2b"
  "bigipExternalSecurityGroup" = "sg-056de270316ea8bc1"
  "bigipManagementSecurityGroup" = "sg-04ce53b880cd526c2"
}
elb_dns_name = tf-elb-studentf5labdev-1462435346.us-west-2.elb.amazonaws.com
web1_address = 10.0.1.80
web2_address = 10.0.2.80
f5admin@EC2AMAZ-Q08MC1S:~/f5agility2020-pc101$
```

1.4.3 SSH to Big-IP1 and Big-IP2

Run the *create-ssh-to-bigip-alias.sh* shell script to create convenient command aliases for *bigip1* and *bigip2*.

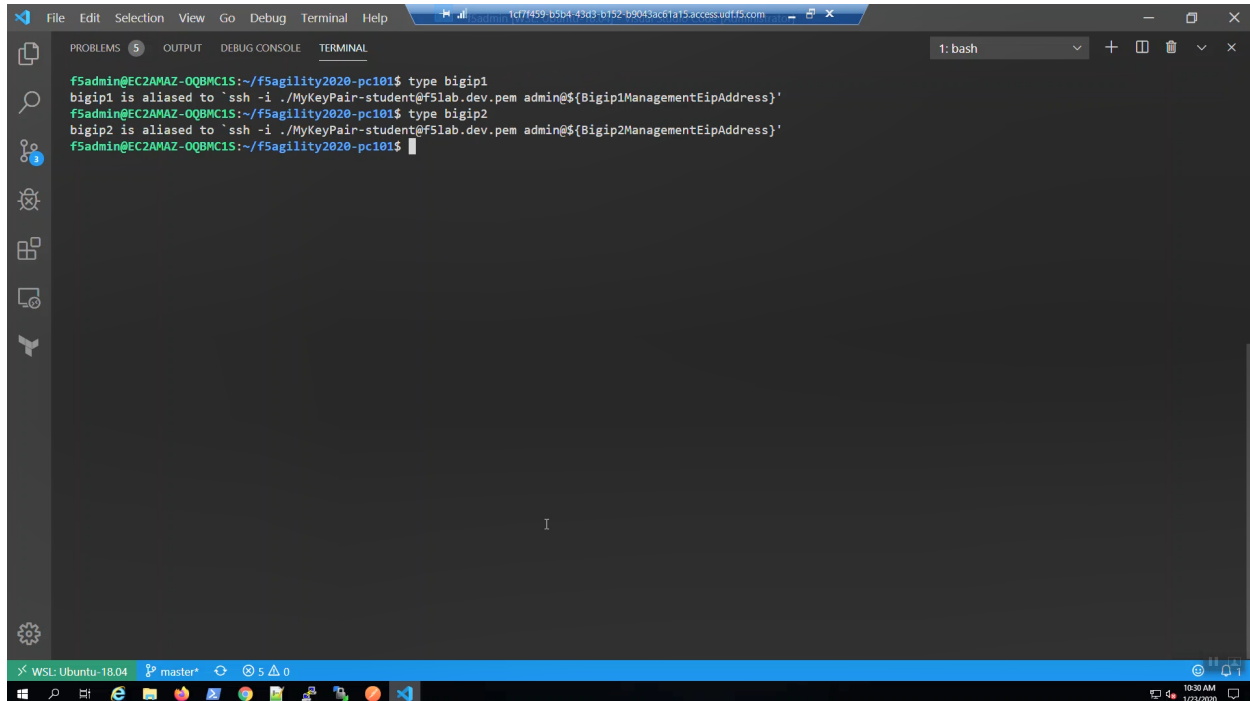
```
source create-ssh-to-bigip-alias.sh
```



A screenshot of a Visual Studio Code terminal window. The terminal shows the command `create-ssh-to-bigip-alias.sh` being executed. The prompt is `fsadmin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$`. The terminal output is empty, indicating the script executed successfully without any visible output. The window title bar shows the file path `1c77459-b5b4-43d3-b152-b9043ac61a15.access.udf.f5.com`. The status bar at the bottom indicates the terminal is running on `WSL: Ubuntu-18.04` with the `master*` branch.

Confirm the bigip1 and bigip2 aliases were created.

```
type bigip1
type bigip2
```



A screenshot of a Visual Studio Code terminal window showing the output of the `type` command for `bigip1` and `bigip2`. The terminal shows the following output:

```
fsadmin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$ type bigip1
bigip1 is aliased to `ssh -i ./MyKeyPair-student@f5lab.dev.pem admin@${Bigip1ManagementEipAddress}`
fsadmin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$ type bigip2
bigip2 is aliased to `ssh -i ./MyKeyPair-student@f5lab.dev.pem admin@${Bigip2ManagementEipAddress}`
fsadmin@EC2AMAZ-QQBMC1S:~/f5agility2020-pc101$
```

The window title bar shows the file path `1c77459-b5b4-43d3-b152-b9043ac61a15.access.udf.f5.com`. The status bar at the bottom indicates the terminal is running on `WSL: Ubuntu-18.04` with the `master*` branch.

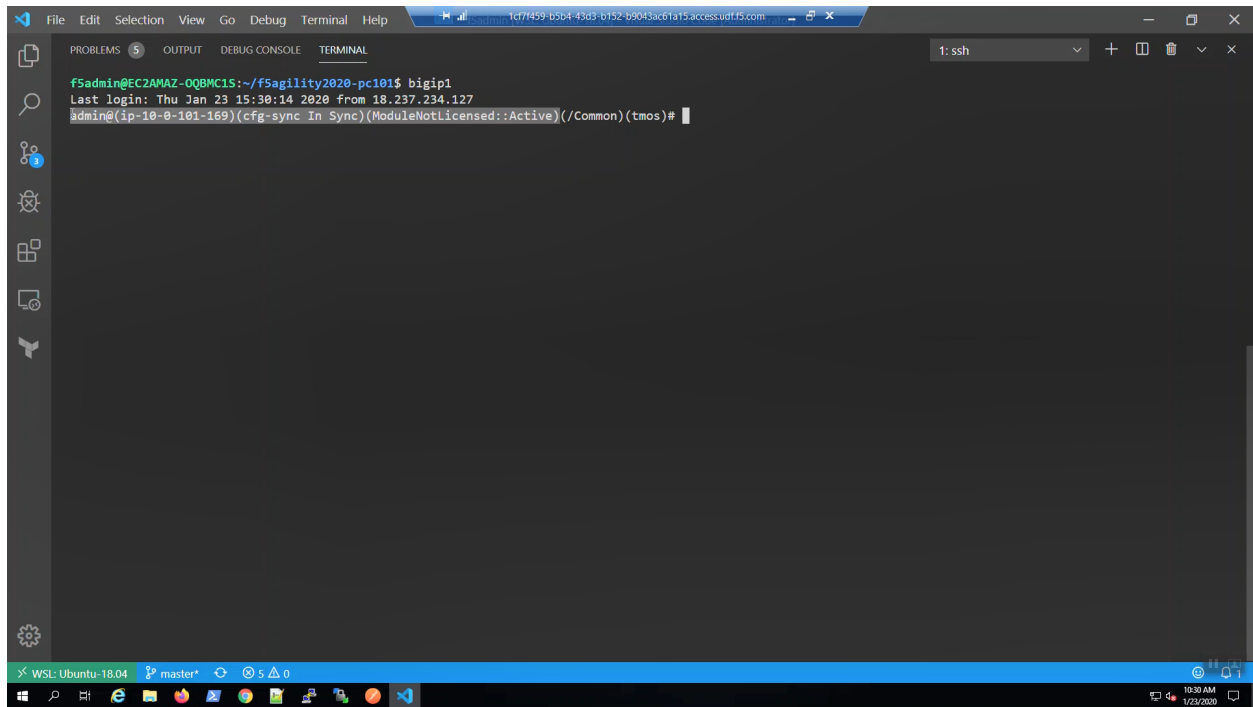
SSH to Big-IP1 to confirm state and change the admin password.

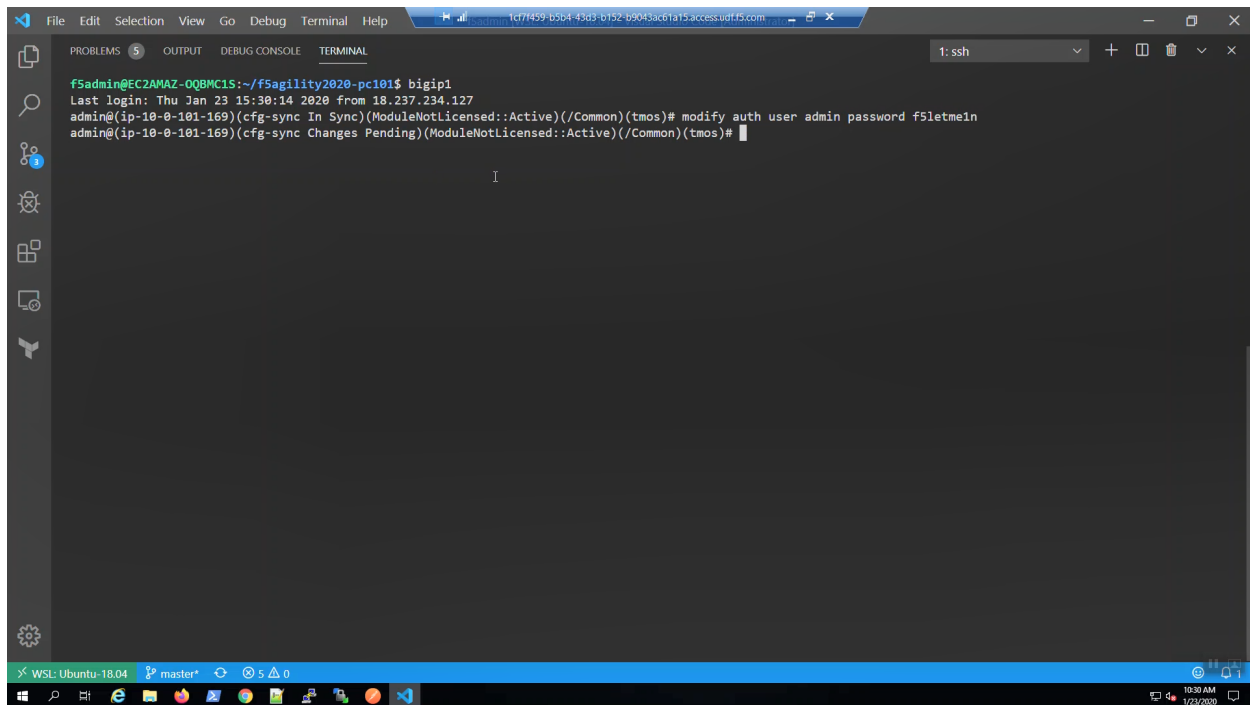
Attention: Ensure the system status prompt is either Active or Standby before proceeding. All other status prompts indicate the system is not ready. Use the screenshots as a guide.

```
bigip1
```

Acknowledge the ssh remote host when prompted “Are you sure you want to continue connecting (yes/no)?”

```
yes
modify auth user admin password f5letmein
modify /sys global-settings gui-setup disabled
save /sys config
quit
```





```
File Edit Selection View Go Debug Terminal Help 1c77459-b5d4-43d3-b152-b9043ac01a15.access.upf.is.com
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: ssh
f5admin@EC2AMAZ-QQ8MC1S:~/f5agility2020-pc101$ bigip1
Last login: Thu Jan 23 15:30:14 2020 from 18.237.234.127
admin(ip-10-0-101-169)(cfg-sync In Sync)(ModuleNotLicensed::Active)(/Common)(tmsh)# modify auth user admin password f5letmein
admin(ip-10-0-101-169)(cfg-sync Changes Pending)(ModuleNotLicensed::Active)(/Common)(tmsh)#
```

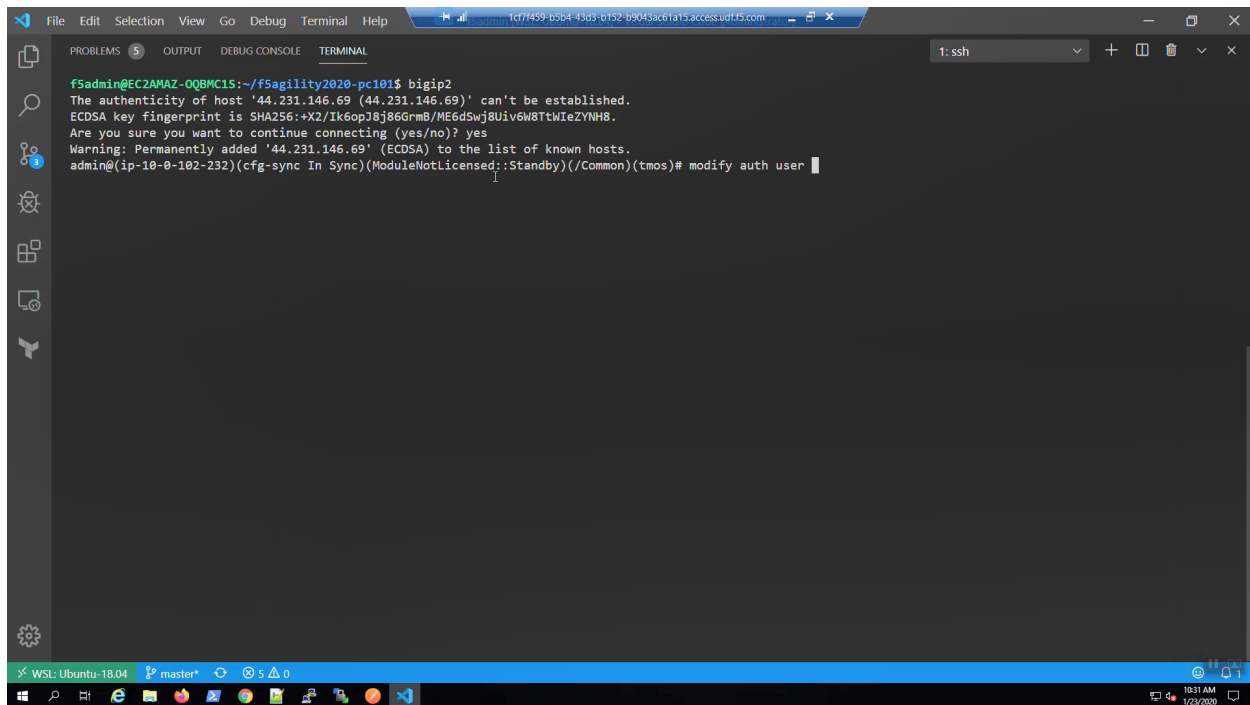
SSH to Big-IP2 to confirm state and change the admin password.

Attention: Ensure the system status prompt is either Active or Standby before proceeding. All other status prompts indicate the system is not ready. Use the screenshots as a guide.

```
bigip2
```

Acknowledge the ssh remote host when prompted “Are you sure you want to continue connecting (yes/no)?”

```
yes
modify auth user admin password f5letmein
modify /sys global-settings gui-setup disabled
save /sys config
quit
```



```
fsadmin@EC2AMAZ-QQ8MC1S:~/f5agility2020-pc101$ bigip2
The authenticity of host '44.231.146.69 (44.231.146.69)' can't be established.
ECDSA key fingerprint is SHA256:+X2/1k6opJ8j86GnmB/ME6dSwj8Uiv6W8tWleZYNH8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '44.231.146.69' (ECDSA) to the list of known hosts.
admin@ip-10-0-102-232(cfg-sync In Sync)(ModuleNotLicensed::Standby)(/Common)(tmsh)# modify auth user
```

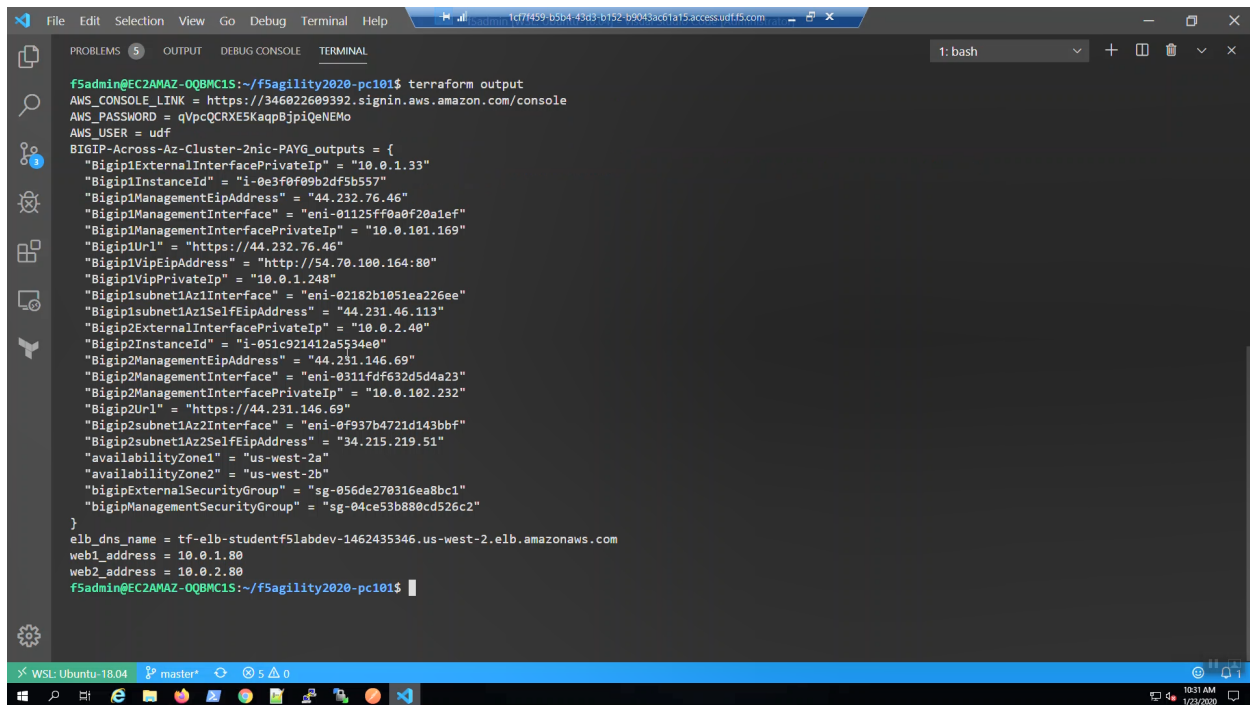
1.4.4 Configuration Utility (WebUI) HTTPS access to Big-IP1 and Big-IP2

We need the management address to connect to the Configuration utility (WebUI). Invoke *terraform output* and look for:

Bigip1Url

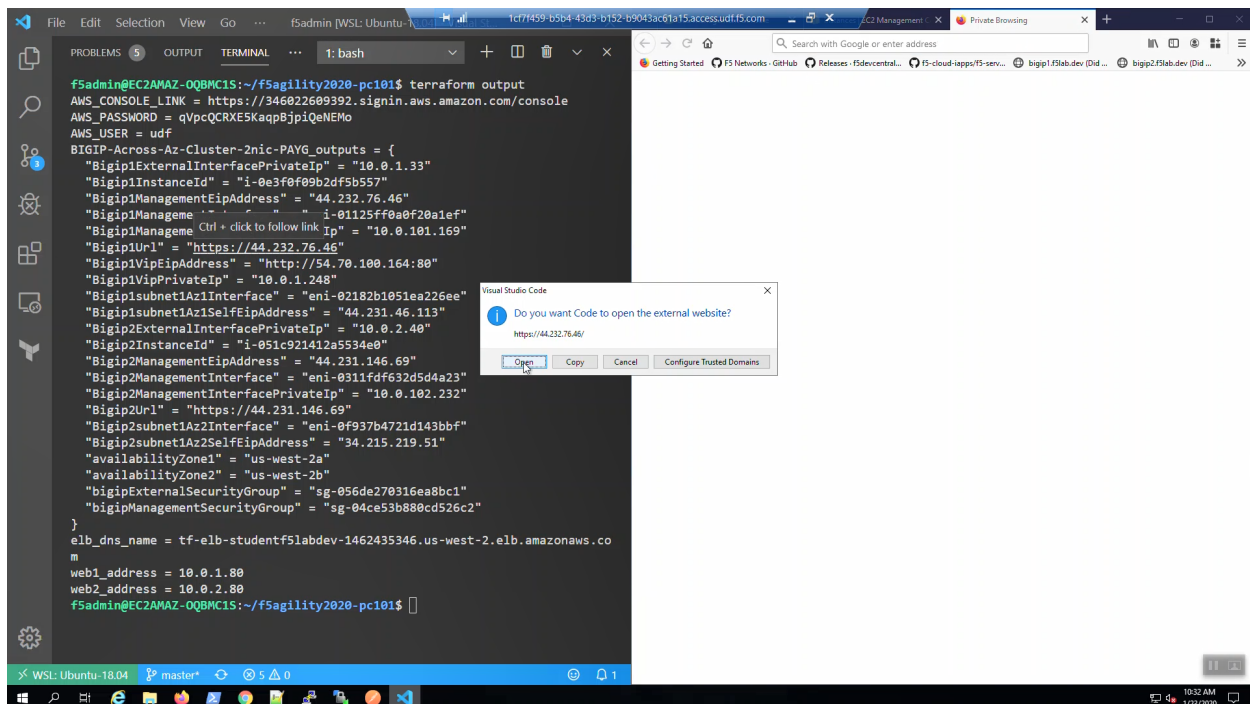
Bigip2Url

```
terraform output
```

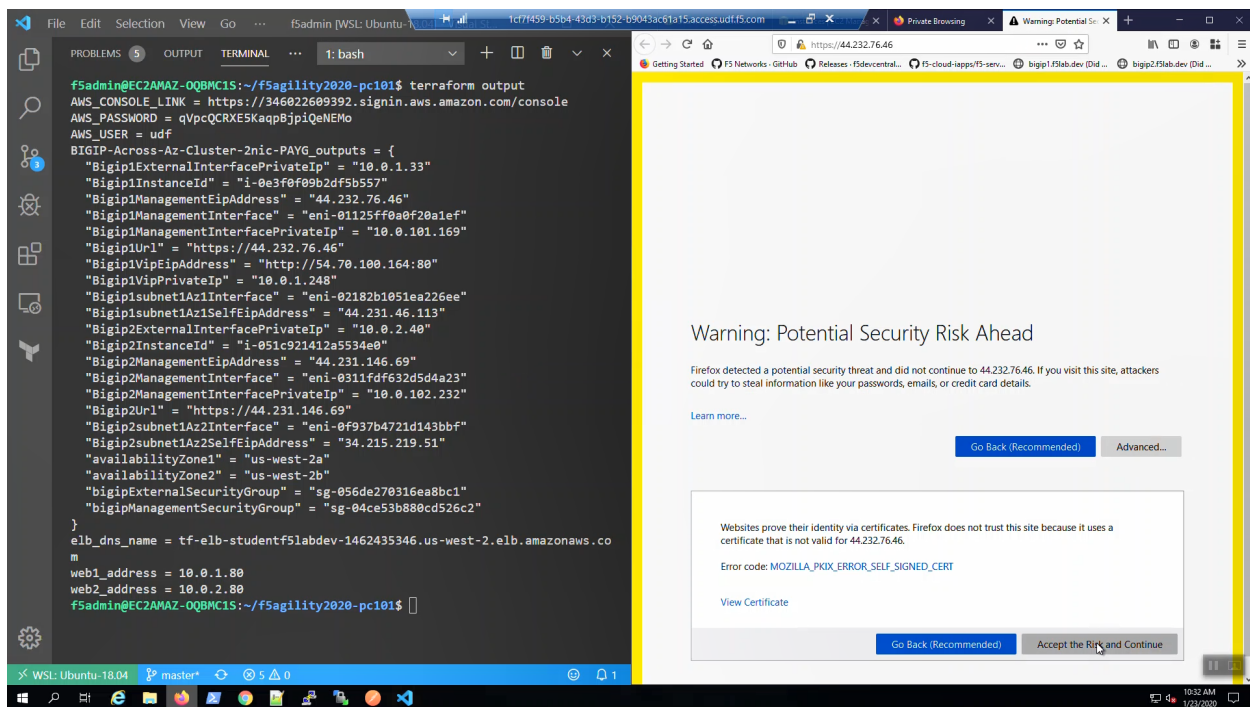


```
f5admin@EC2AMAZ-Q0BMC1S:~/f5agility2020-pc101$ terraform output
AWS_CONSOLE_LINK = https://346022609392.signin.aws.amazon.com/console
AWS_PASSWORD = qVpcQCRXE5KaqpBjpiqeNEHo
AWS_USER = udf
BIGIP-Across-Az-Cluster-2nic-PAYG_outputs = {
  "Bigip1ExternalInterfacePrivateIp" = "10.0.1.33"
  "Bigip1InstanceId" = "i-0e3f0f09b2df5b557"
  "Bigip1ManagementEipAddress" = "44.232.76.46"
  "Bigip1ManagementInterface" = "eni-01125ff0a0f20a1ef"
  "Bigip1ManagementInterfacePrivateIp" = "10.0.101.169"
  "Bigip1Url" = "https://44.232.76.46"
  "Bigip1VipEipAddress" = "http://54.70.100.164:80"
  "Bigip1VipPrivateIp" = "10.0.1.248"
  "Bigip1subnet1Az1Interface" = "eni-02182b1051ea226ee"
  "Bigip1subnet1Az1SelfEipAddress" = "44.231.46.113"
  "Bigip2ExternalInterfacePrivateIp" = "10.0.2.40"
  "Bigip2InstanceId" = "i-051c921412a5534e0"
  "Bigip2ManagementEipAddress" = "44.231.146.69"
  "Bigip2ManagementInterface" = "eni-0311fdf632d5d4a23"
  "Bigip2ManagementInterfacePrivateIp" = "10.0.102.232"
  "Bigip2Url" = "https://44.231.146.69"
  "Bigip2subnet1Az2Interface" = "eni-0f937b4721d143bbf"
  "Bigip2subnet1Az2SelfEipAddress" = "34.215.219.51"
  "availabilityZone1" = "us-west-2a"
  "availabilityZone2" = "us-west-2b"
  "bigipExternalSecurityGroup" = "sg-056de270316ea8bc1"
  "bigipManagementSecurityGroup" = "sg-04ce53b880cd526c2"
}
elb_dns_name = tf-elb-studentf5labdev-1462435346.us-west-2.elb.amazonaws.com
web1_address = 10.0.1.80
web2_address = 10.0.2.80
f5admin@EC2AMAZ-Q0BMC1S:~/f5agility2020-pc101$
```

From the Visual Studio Code Terminal, Ctrl + click to follow link for “BigIP1Url” =



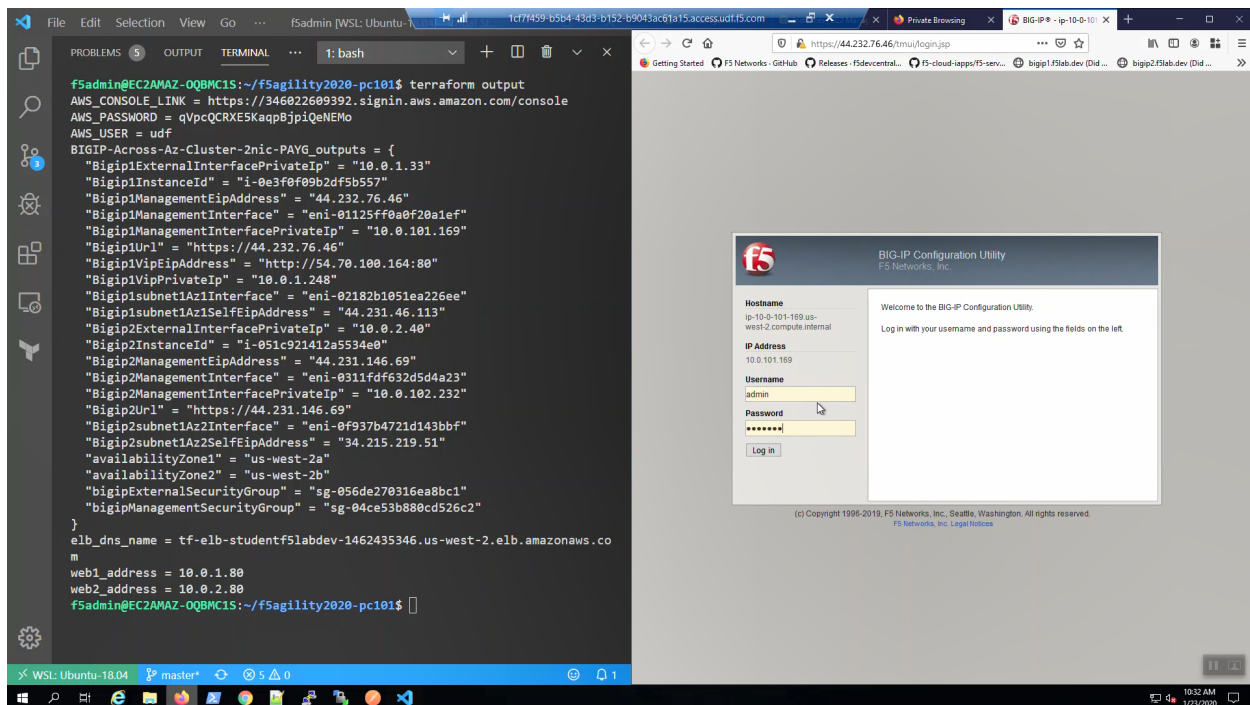
We are using self-signed certificates in the lab. Bypass the TLS warnings. “Accept the Risk and Continue”.



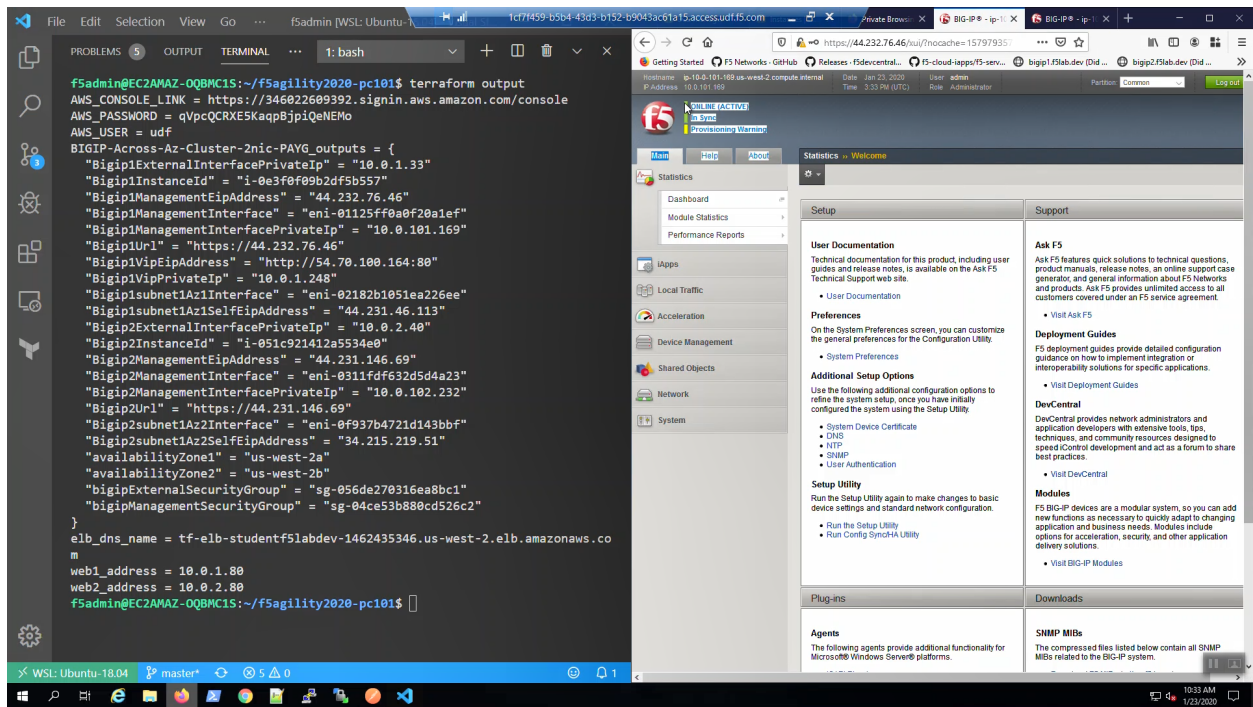
Login to the Big-IP1 Configuration Utility (WebUI):

Username: admin

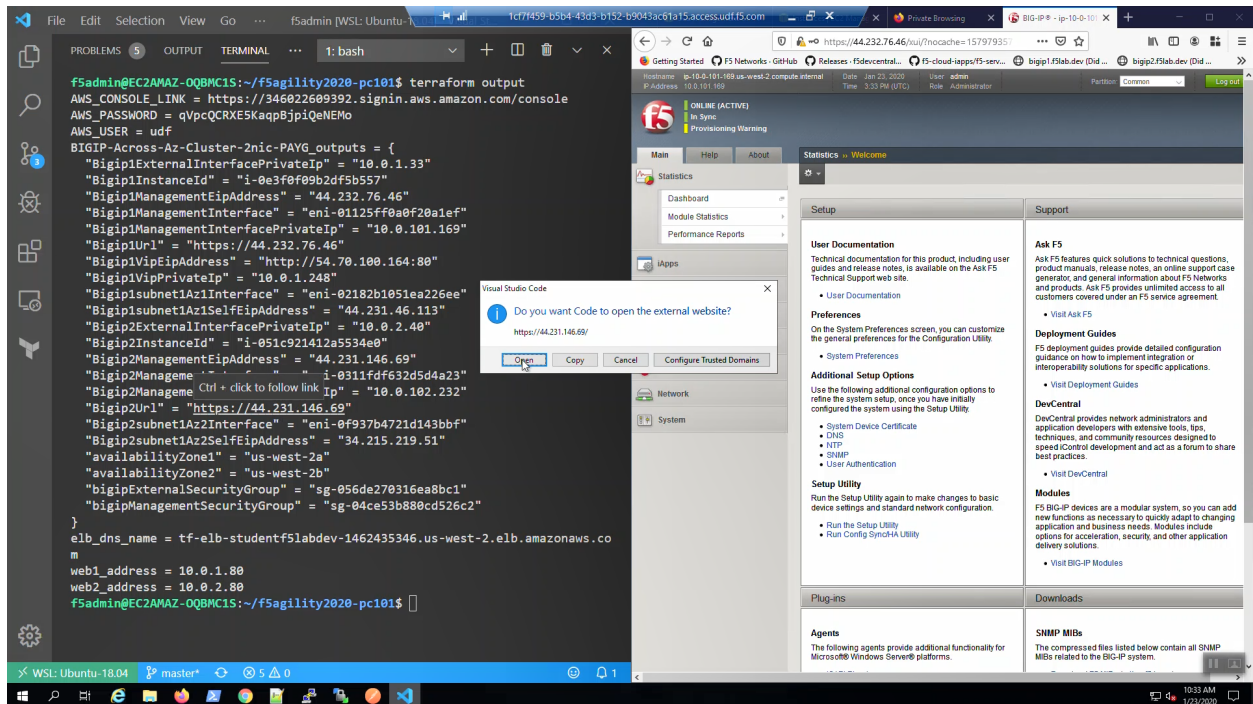
Password: f5letme1n



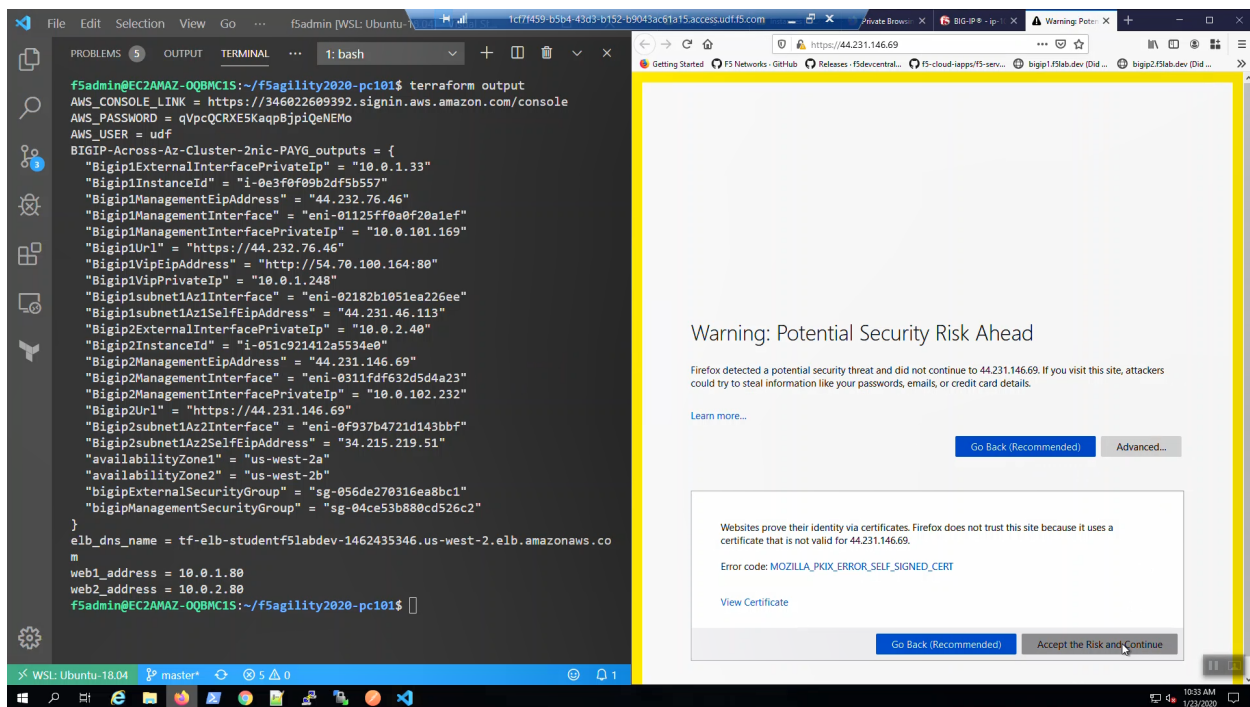
Look in the upper-left-hand corner. Note whether this is the active or standby unit.



From the Visual Studio Code Terminal, Ctrl + click to follow link for “BigIP2Url” =



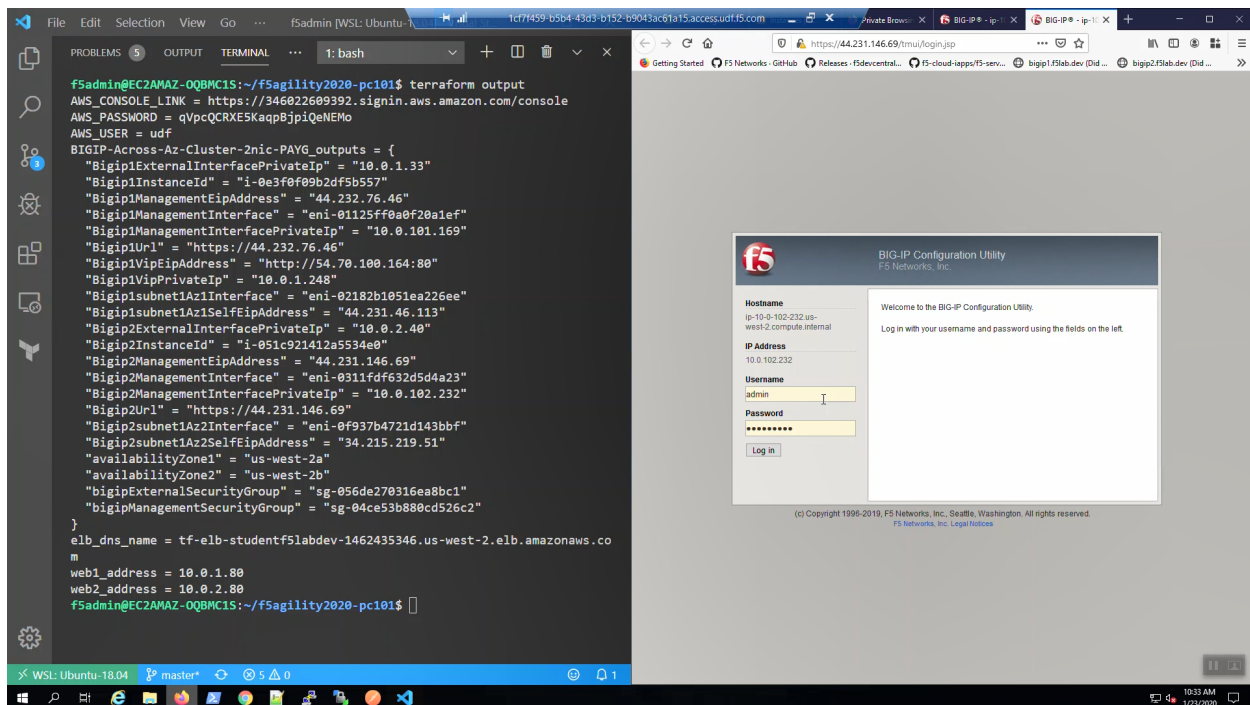
We are using self-signed certificates in the lab. Bypass the TLS warnings. “Accept the Risk and Continue”.



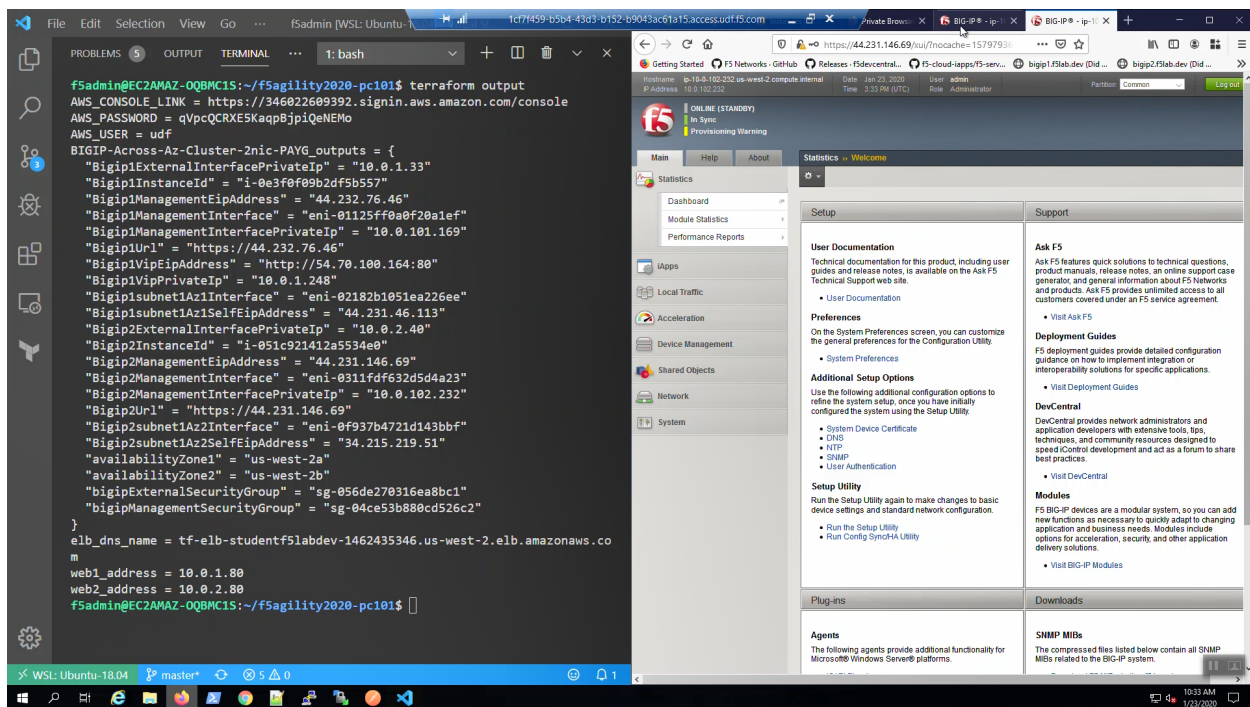
Login to the Big-IP2 Configuration Utility (WebUI):

Username: admin

Password: f5letme1n



Look in the upper-left-hand corner. Note whether this is the active or standby unit.



Attention: Make sure Big-IP1 is active and Big-IP2 is standby before proceeding. To force Big-IP2 to standby: "Device Management" => "Devices" => bigip2.f5lab.dev => [Force to Standby].

1.5 Creating High Availability Virtual Servers in AWS

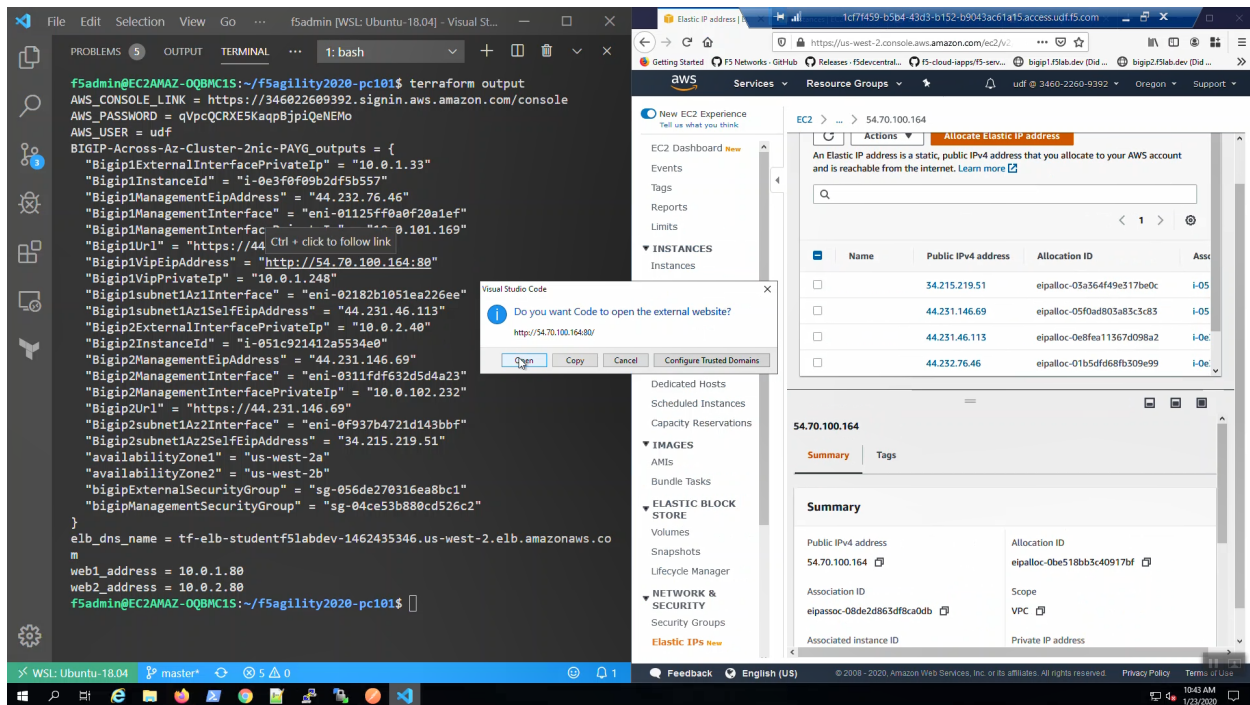
The F5 Virtual Edition can discover instances in AWS dynamically using tags.

1.5.1 Check example application is running

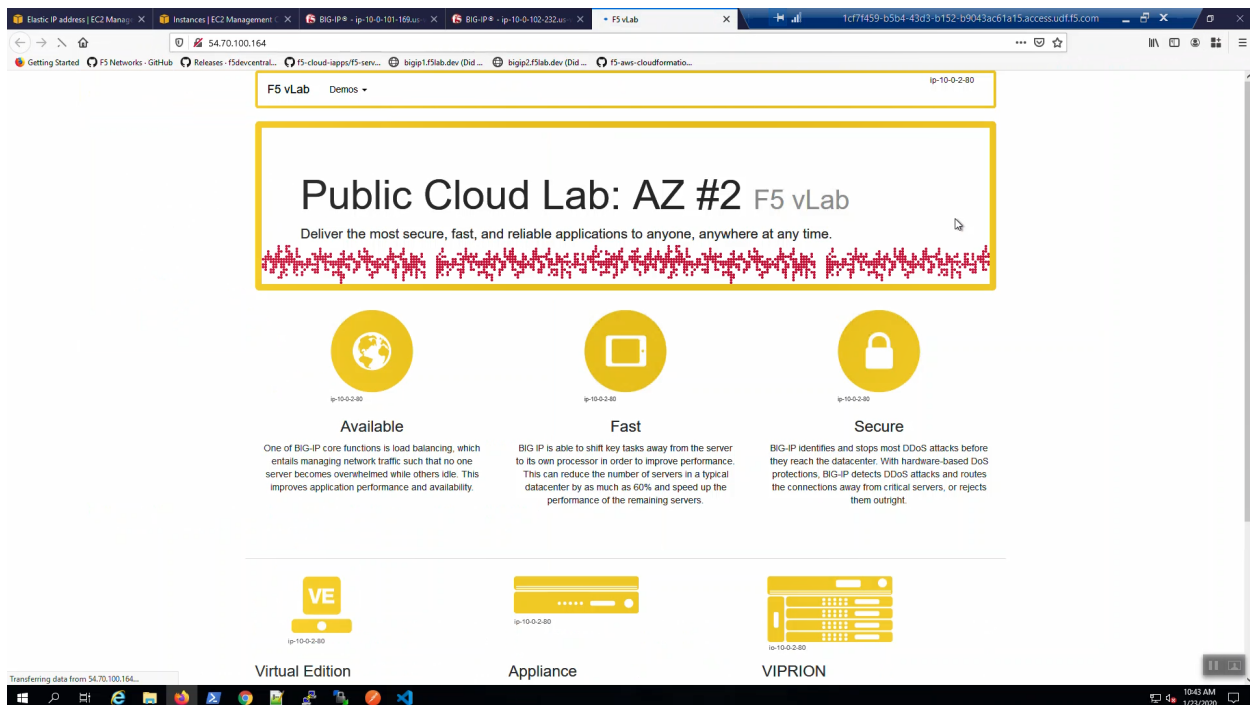
From the Visual Studio Code Terminal, invoke *terraform output*.

```
terraform output | grep Bigip1VipEipAddress
```

...Ctrl + click on the *Bigip1VipEipAddress*=. This is the same Elastic IP we just reviewed in the AWS Console.

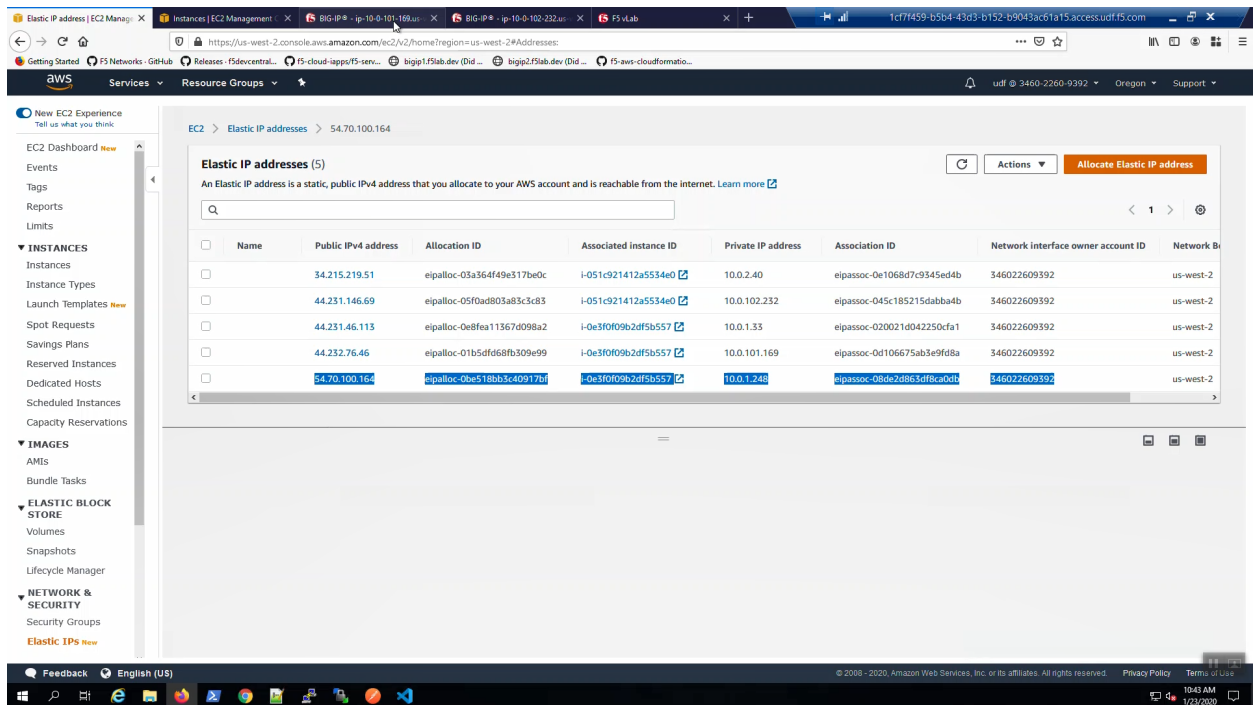


We are using self-signed certificates in the lab. Bypass the TLS warnings. “Accept the Risk and Continue”. You will see the example app.

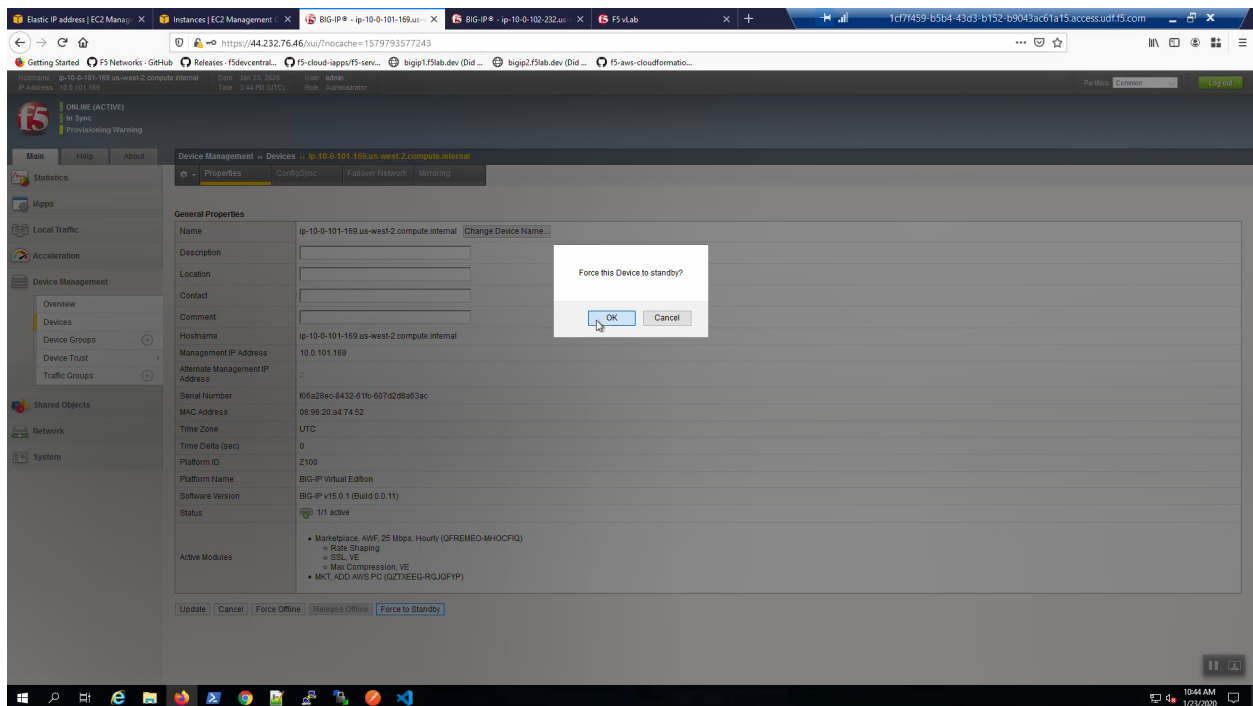


1.5.2 Survive a fail-over event across Availability Zones

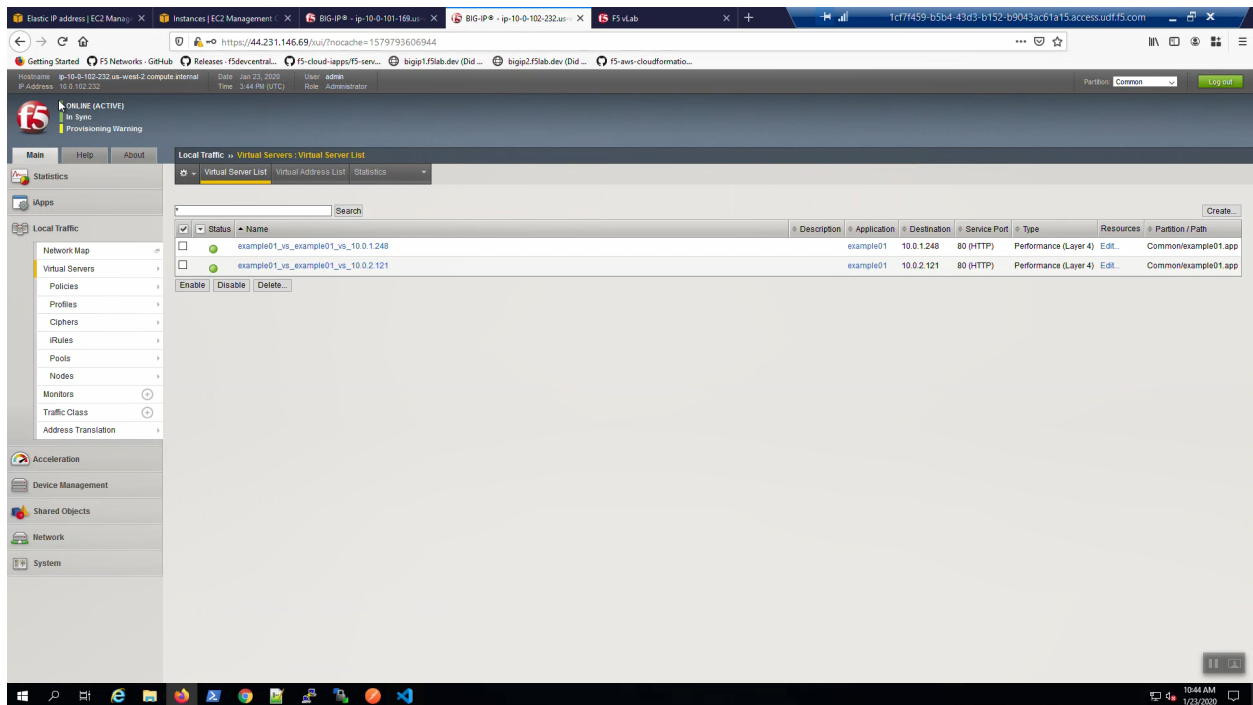
From the AWS Console, Services => EC2 => NETWORK & SECURITY => Elastic IPs. Note the Elastic IP address (public IPv4 address) mapping for the Secondary IP address of our **Active** Big-IP1 (10.0.1.x).



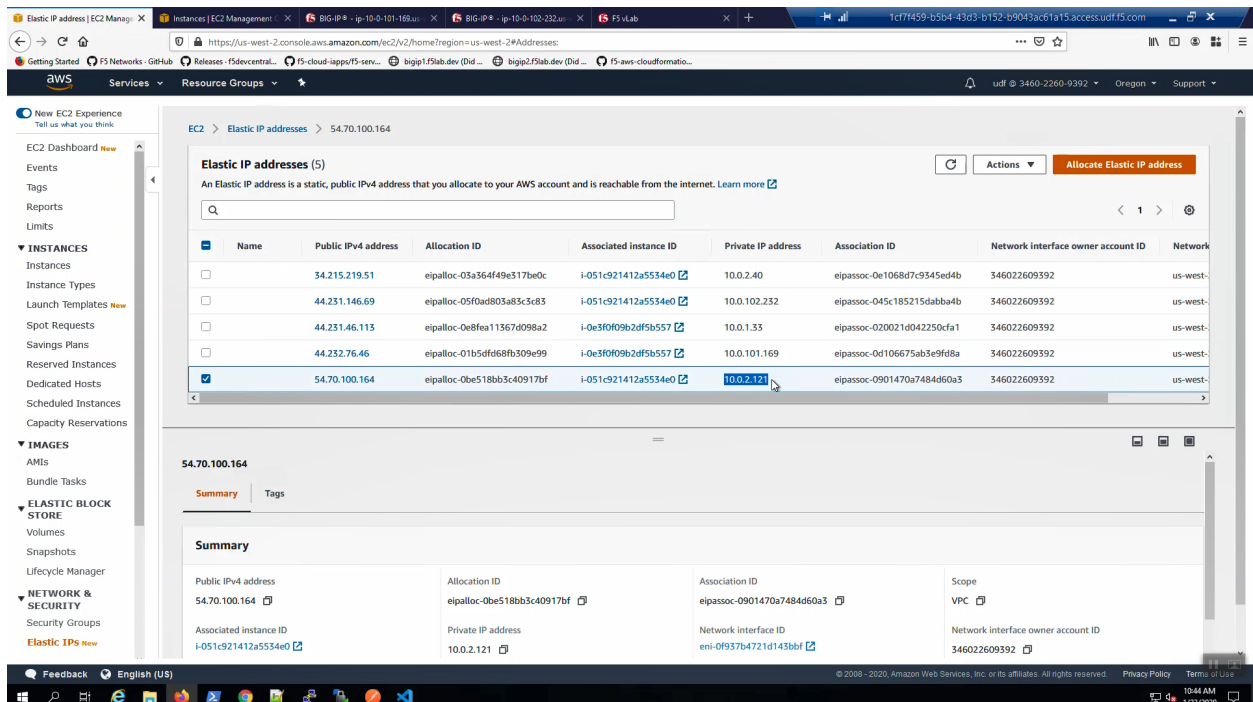
Big-IP1 => Device Management => Devices => Self => [Force to Standby]. Click [OK] to confirm.



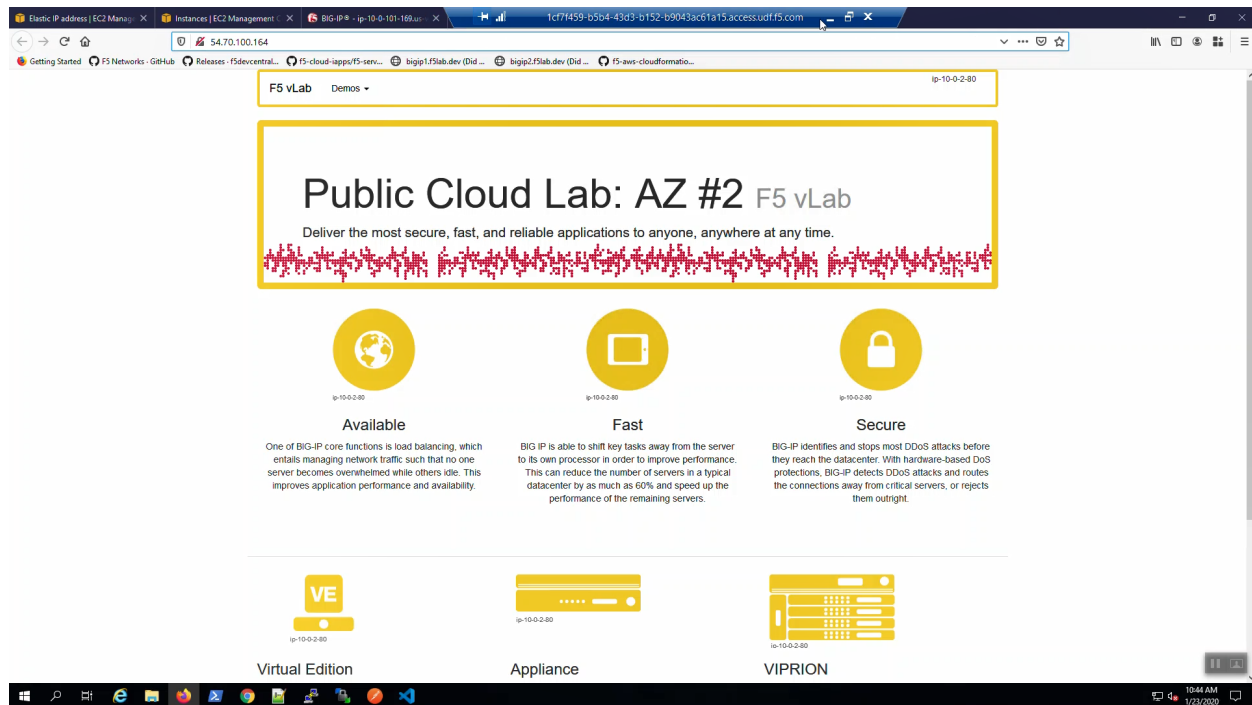
Big-IP2 is now active.



From the AWS Console, Services => EC2 => NETWORK & SECURITY => Elastic IPs. Note the Elastic IP address (public IPv4 address) mapping for the Secondary IP has changed to the new **Active** Big-IP2 (10.0.2.x). Hit the refresh icon in the upper-right-hand side a few times until you notice the change.



Back to the example app screen. We are using self-signed certificates in the lab. Bypass the TLS warnings. “Accept the Risk and Continue”. You will see the example app now behind the new active Big-IP2.



Attention: The example application reports which Availability Zone is serving up the content (pool member), *not* which Availability Zone is hosting the active Big-IP.

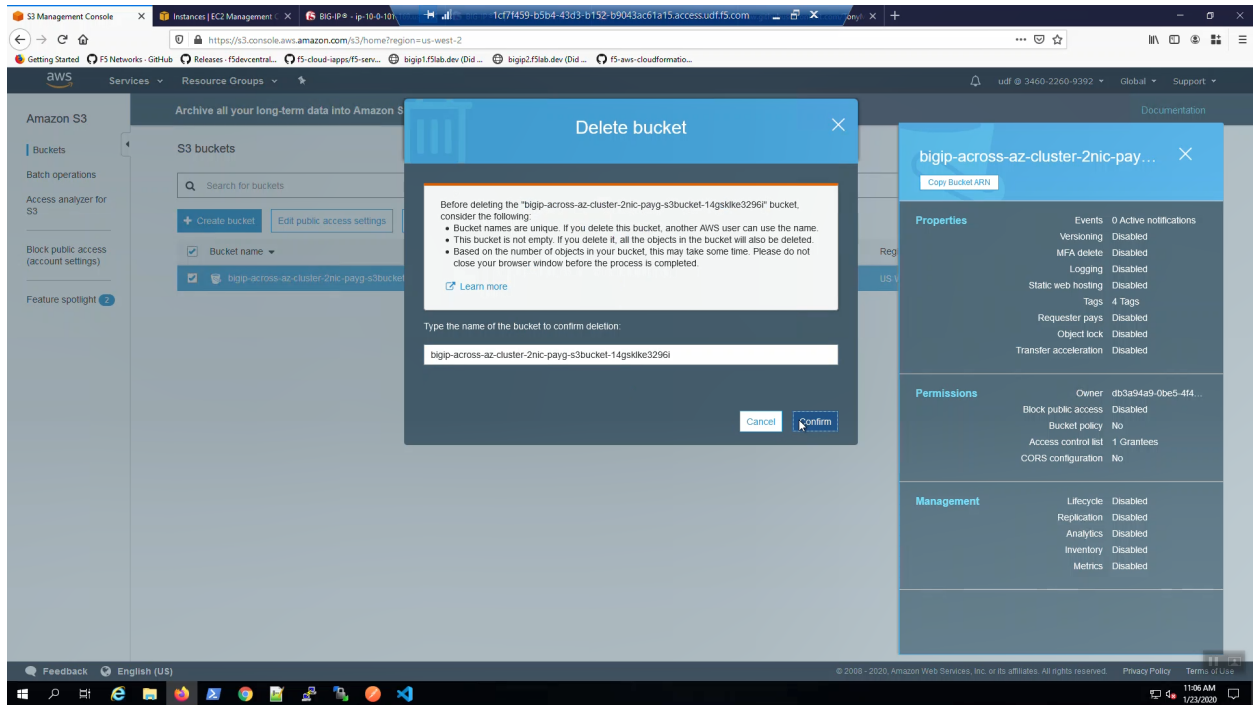
1.6 Delete Environment in Fell Swoops

The promise of declarative automation tools is that we can both quickly stand up *and* tear down environments.

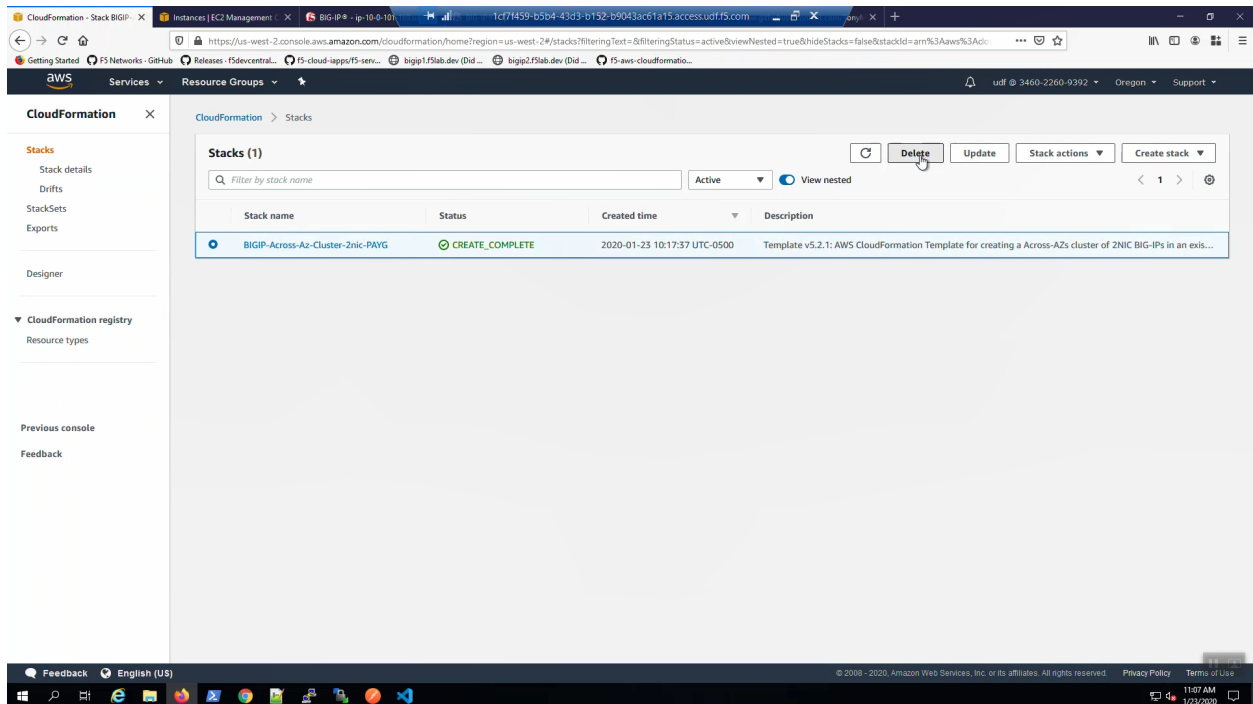
1.6.1 Cleanup

Delete the S3 bucket.

AWS => Services => S3 => delete bucket.



Delete the CloudFormation Stack.



1.6.2 Terraform destroy

From the Visual Studio Code Terminal, invoke *terraform destroy*.

```
terraform destroy --force -var 'bigip_admin_password=f5letmein'
```

```
File Edit Selection View Go Debug Terminal Help 1c7f459-b504-43d3-b152-b9043ac01a15.access.uff.fs.com
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: terraform
fsadmin@EC2AMAZ-QQ8MC1S:~/f5agility2020-pc101$ terraform destroy --force
var.bigip_admin_password
Enter a value: f5letmein

null_resource.ecdsa_certs: Refreshing state... [id=6009194551436473692]
aws_cloudwatch_log_group.log-group: Refreshing state... [id=studentf5labdev]
data.aws_cloudformation_stack.f5-cluster-across-az-ha: Refreshing state...
aws_vpc.terraform-vpc: Refreshing state... [id=vpc-078b314202bbdff7]
aws_cloudwatch_log_stream.log-stream: Refreshing state... [id=log-stream]
data.aws_network_interface.bigip2_traffic: Refreshing state...
data.aws_network_interface.bigip1_traffic: Refreshing state...
aws_subnet.private-b: Refreshing state... [id=subnet-075282aa20b7cb04a]
aws_subnet.private-a: Refreshing state... [id=subnet-01d1969c004f12dbc]
aws_subnet.f5-management-a: Refreshing state... [id=subnet-0eb68f26d5bfeba2b]
aws_security_group.instance: Refreshing state... [id=sg-0122d4059e0dfdb1]
aws_internet_gateway.gw: Refreshing state... [id=igw-0ca4881650d473b76]
aws_subnet.public-a: Refreshing state... [id=subnet-073a1c49f0efc007f]
aws_security_group.f5_data: Refreshing state... [id=sg-0073942f4374a6075]
aws_security_group.elb: Refreshing state... [id=sg-07f29efb6f0edc7f4]
aws_subnet.public-b: Refreshing state... [id=subnet-0a3a60de4e624c322]
aws_subnet.f5-management-b: Refreshing state... [id=subnet-01dbdb4b6cf59fa65d]
aws_security_group.f5_management: Refreshing state... [id=sg-00b3a27bda5e4c05a]
aws_route_table.rtl: Refreshing state... [id=rtb-049d20f4b17481123]
aws_instance.example-a: Refreshing state... [id=i-04508a1a92c3dc15]
aws_instance.example-b: Refreshing state... [id=i-0d2b4d3e9d6def0f6]
aws_main_route_table_association.association-subnet: Refreshing state... [id=rtbassoc-0803a0bf5b70f8c45]
```

```
File Edit Selection View Go Debug Terminal Help 1c7f459-b504-43d3-b152-b9043ac01a15.access.uff.fs.com
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: bash
aws_internet_gateway.gw: Still destroying... [id=igw-0ca4881650d473b76, 1m0s elapsed]
aws_subnet.public-a: Still destroying... [id=subnet-073a1c49f0efc007f, 30s elapsed]
aws_instance.example-b: Still destroying... [id=i-0d2b4d3e9d6def0f6, 1m10s elapsed]
aws_security_group.elb: Still destroying... [id=sg-07f29efb6f0edc7f4, 1m10s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0ca4881650d473b76, 1m10s elapsed]
aws_subnet.public-a: Still destroying... [id=subnet-073a1c49f0efc007f, 40s elapsed]
aws_instance.example-b: Still destroying... [id=i-0d2b4d3e9d6def0f6, 1m20s elapsed]
aws_security_group.elb: Still destroying... [id=sg-07f29efb6f0edc7f4, 1m20s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0ca4881650d473b76, 1m20s elapsed]
aws_security_group.elb: Destruction complete after 1m27s
aws_subnet.public-a: Still destroying... [id=subnet-073a1c49f0efc007f, 50s elapsed]
aws_instance.example-b: Still destroying... [id=i-0d2b4d3e9d6def0f6, 1m30s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0ca4881650d473b76, 1m30s elapsed]
aws_subnet.public-a: Destruction complete after 56s
aws_instance.example-b: Still destroying... [id=i-0d2b4d3e9d6def0f6, 1m40s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0ca4881650d473b76, 1m40s elapsed]
aws_instance.example-b: Still destroying... [id=i-0d2b4d3e9d6def0f6, 1m50s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0ca4881650d473b76, 1m50s elapsed]
aws_instance.example-b: Still destroying... [id=i-0d2b4d3e9d6def0f6, 2m0s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0ca4881650d473b76, 2m0s elapsed]
aws_instance.example-b: Still destroying... [id=i-0d2b4d3e9d6def0f6, 2m10s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0ca4881650d473b76, 2m10s elapsed]
aws_instance.example-b: Still destroying... [id=i-0d2b4d3e9d6def0f6, 2m20s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0ca4881650d473b76, 2m20s elapsed]
aws_internet_gateway.gw: Destruction complete after 2m26s
aws_instance.example-b: Still destroying... [id=i-0d2b4d3e9d6def0f6, 2m30s elapsed]
aws_instance.example-b: Destruction complete after 2m30s
aws_subnet.public-b: Destroying... [id=subnet-0a3a60de4e624c322]
aws_security_group.instance: Destroying... [id=sg-0122d4059e0dfdb1]
aws_subnet.public-b: Destruction complete after 0s
aws_vpc.terraform-vpc: Destroying... [id=vpc-078b314202bbdff7]
aws_vpc.terraform-vpc: Destruction complete after 1s

Destroy complete! Resources: 21 destroyed.
fsadmin@EC2AMAZ-QQ8MC1S:~/f5agility2020-pc101$
```

Confirm it's all gone. Review AWS => Services => ec2 => Instances.

Fin

WE MAKE APPS  FASTER.
SMARTER.
SAFER.

F5 Networks, Inc. | f5.com



US Headquarters: 401 Elliott Ave W, Seattle, WA 98119 | 888-882-4447 // Americas: info@f5.com // Asia-Pacific: apacinfo@f5.com // Europe/Middle East/Africa: emeainfo@f5.com // Japan: f5j-info@f5.com
©2017 F5 Networks, Inc. All rights reserved. F5, F5 Networks, and the F5 logo are trademarks of F5 Networks, Inc. in the U.S. and in certain other countries. Other F5 trademarks are identified at f5.com. Any other products, services, or company names referenced herein may be trademarks of their respective owners with no endorsement or affiliation, express or implied, claimed by F5. These training materials and documentation are F5 Confidential Information and are subject to the F5 Networks Reseller Agreement. You may not share these training materials and documentation with any third party without the express written permission of F5.