Okay, so that took a few minutes but we now in a position where the load balancer is working and it's got three out of three instances in service. If I click on the instances tab, you'll be to see each of our three cat web servers. So the one that's in 1a, the one that's in 1b, and the one that's in 1c and each of them is marked as in service, which means that it's currently passing the health check.

**Now, for every load balancer it's created and is given a DNS name. So** in this case, we've got this CLB so it's the name of load balancer and then some randomness and in the region so U.S. East-1.elb.amazon.aws.com and if we go ahead and open that in a new tab, we'll see that we actually get a page load, and that's obviously connected to one of our back end instances because it's loading up the cat picture so we can see that is 125.29 and if I go back to our running instances and open in a new tab and then I scroll across looking for the public IP address, we've got the 125.29 and that matches that IP address. So we know that this is actually talking to the backend instance cat web three and then, if I refresh tha, a couple of times we'll see that the IP address changes but the image hasn't and I want to explain that, because that can sometimes be confusing when you're trying to demonstrate how load balancers work. Now this web page is actually displaying an IP address, and it's loading an image called cat.jpg. It's the same named image on each of the three backend web servers and so what's actually happening is this **load balancer is sending us to one of the three cat web servers it's loading the index.php file with this unique IP address, but it's referencing the same image file name**.

Now **Chrome has a habit of caching these images, So instead of loading this image every time it's actually caching it. So that's why the image didn't change**. If I move across to another browser and I'm going to demonstrate this with Firefox this time, if I refresh this in Firefox that each time we change between the servers, then we getting that brand new cat picture so we can see that we're moving between each of the back end servers we're getting a new IP address and a new cat picture. So every time we change server, we're loading up the individual cat image on that server and that's how load balancers are architected.

So **every time a new connection comes in using the DNS name of the load balancer, it's directed at one of the load balancer nodes and on that node picks one of the backend instances to direct that connection towards.** So you'll notice that with most load balancers, if you just keep hitting refresh then each time, it will make a connection to each of those individual servers. Now, I mentioned this earlier in this lesson, but if I do go to one of the service individually, so cat web one on a copy of the DNS name into my clipboard and open that in a new tab, I can still connect directly to those individual EC2 instances. Now, this might not be what you want to do. So one option is to go to the instances, find the security group that those instances use. So in this particular case, remember, I created a cat web security group that used by all of these instances so I can open that security group, click on the inbound tab, click on edit, and then instead of allowing HTTP connections from anywhere, I could delete this IP address, start typing SG for security group and then pick the security group that's associated with the load balancer. So I'm going to make that change, and I'm going to hit save. So at this point, the only thing that can make a port 80 connection to any of these web servers is the load balancer. So I go back to the tab that has the connection open directly an attempt to hit refresh. It won't be able to connect but if I go back to my Firefox window, which uses load balancer and hit refresh again, it'll continue to be able to connect to the web servers**. So by using load balancer and changing the security groups I have now prevented direct connections to the web servers but I've still allowed the load balancer to make connections on my behalf, and that is probably the best practice configuration when using load balancers, you want to prevent direct connections to the web servers, but allow connections from the low balancer.**

Now, one last thing I want to demonstrate before I finish this lesson is I want to show you what actually happens when an instance fails the health check. So I'm going to pick one of the instances and it's going to be cat web one going to right click and hit connect. Now, because I'm using a Linux compatible operating system, I'll need to change the permissions on this p key. If you are using Windows, you can go ahead and click on the connect using party and follow the instructions for connecting using the Pootie Windows Terminal program because I'm using a limits compatible operating system, though I'm going to move across to my terminal. I'm already in my Dallas holder, which is where I say this pen keys. I'm going to just the permissions that I'm going to get the connection string, move back to my terminal and connect to the instance. It's the first time I've done so solely to accept the authenticity and now that I'm connected, I'm going to go ahead and want to sudo service httpd for the web server and then stop and that's going to go ahead and stop the Apache web server that's running on this instance. Now the effect that this will have is it will stop the web server functionality. That will mean that the load balancer will no longer be able to connect as part of its health check. So if I move back across to my console, I go to load balancers I look at the instances. We can already see that because I set that health check interval to a really low value. It's already determined that this instance is out of service. If I move back to my Firefox window and keep hitting refresh after a couple of times, it switches over to the other web server. So now we've got two operational web servers we can see alternating between those two, but it won't attempt to make any connections to the server that's out of service. If I go back to the terminal and run the reverse of this so sudo service httpd but instead of stop, I go start, that will bring the web server back into an operational state. The load balancer will continue those health checks and after we've got two successful health checks, it'll mark that instance as being back in service and once it's back in service, which you can see now. It will begin sending connections back to that server again. So now if I go back to Firefox and it might take some time but if I keep refreshing it, we should see cat web one. So this is cat web one. This is the picture that we're expecting. This is the server that's been brought back into an operational state so it could be used by the load balancer. Now that's pretty much everything that I wanted to cover about classic load balancers. I wanted to cover the basic functionality that is common amongst all of the different types of load balancers, so we got the concept of listeners. So that's how the load balancer listens to incoming connections. We've got the protocol that used to connect to the backend instances. I've talked about how right now I'm manually associating instances with the load balancer, but we'll cover how to do automatically later in this topic. I've talked about how you can influence the availability zones that nodes for this load balancer to get placed in. So that's by editing the availability zones. I've talked about cross zone load balancing and how that works. You can, of course, click on the monitoring tab and gain access to a lot of metrics about the load balancer and this uses CloudWatch so you can see details on the unhealthy host, the healthy hosts, the average latency, the number of connections, the number of 500 status codes, 400 codes, and 200 status codes, which means everything's okay and much more. You've got a full set of monitoring that's available. Now, in the next lesson, I'm going to extend this and talk about application load balancers. Now these are much more fully featured and they're type of load balancer that AWS recommends that use for most situations. So at this point, go ahead, mark this video as complete and when you're ready to join me in the next lesson we'll talk about application load balancers.