#_____Section 11_____#

Welcome to this section
on deploying and managing infrastructure at scale.
In this section, we'll see different ways
to deploy your workloads onto AWS.
And the first technology I wanna talk about
is CloudFormation.
So CloudFormation is such an important technology
in the AWS
because it is a declarative way
of outlining your AWS infrastructure, for any resources,
and most of them are supported.
So to give you a concrete example,
in CloudFormation, you would say,
"I want a security group.
I want two EC2 instances
that will be using that security group.
I also want an S3 bucket.
And I want a load balancer in front of all these machines."
Then CloudFormation
automatically creates all these things for you
in the right order
with the exact configuration that you specify.
So the benefits of using CloudFormation are multiple.
But the first one
is that all your infrastructure is as code.
That means that
you will never, ever create resources manually
like we've done in this course,
which is excellent for control.
And that means that anytime you do a changes
to how your AWS cloud is doing,
then it needs to be reviewed through code review,
which is a great way to operate in a cloud.
On top of things, there is a cost advantage,
because each resource within the stack
is going to get a tag that is going to be similar
to all the other resources created within the stack.
And you can also easily estimate the cost of your resources
using the CloudFormation templates.
And finally, thanks to CloudFormation,
you can have a saving strategy.
For example, you can say that in some environment,
you could automate the deletion of all the templates
at 5:00 PM,
which will delete all the associated resources
with that template,
and then recreate it at 9:00 AM or 8:00 AM safely.

And so therefore, you have cost savings
because you don't have any resources
between 5:00 PM and 8:00 AM.
With CloudFormation,
it's super easy to create and delete resources,
which is one of the biggest cloud principle.
Then for productivity.
So as I said, you're able to destroy
and recreate infrastructure on the fly.
It's also generating diagrams for you,
for your templates, as we'll see very quickly.
And there's declarative programming,
so you don't need to figure out
if you need to create a DynamoDB table first,
or an EC2 instance, or all these things together.
The CloudFormation template is smart enough
to figure out how to do things.
Finally, with CloudFormation, we don't reinvent the wheel.
So that means that we can leverage existing templates
on the web, we can leverage documentation,
and CloudFormation supports almost all AWS resources.
That means that everything we'll see in this course
is supported by CloudFormation,
and in case it isn't,
you can use something called a custom resource
for resources that are not supported.
So CloudFormation
really is the base of infrastructure as code on AWS.
So as I said, you could visualize a CloudFormation template,
and for this you would use the Application Composer service.
So I took the liberty
to use a WordPress CloudFormation stack,
and visualize it in Application Composer.
And as you can see, we can see all the resources
of our CloudFormation templates.
So we can see the ALBListener, the database security group,
the SQL database, different security groups,
launch configuration, application balancers, and so on.
But on top of it, we can see the relations
between all of these components
and how they're linked together,
which is very handy
when you want to understand your architecture diagrams.
So from an exam perspective,
CloudFormation is going to be used
when we have infrastructure as code,
when we need to repeat an architecture
in different environments, different regions,
or even different AWS accounts.
So that's it for me.

I will see you in the next lecture
for a short practice on CloudFormation.

Okay, so I'm going to give you
a quick introduction to CloudFormation
and give you an overview that really allows you
to understand how it works.
So let's create a stack
and let's make sure first of all that we are in the US East,
Northern Virginia, US East 1 region,
because the template I've created for you
only works in that region.
And I will tell you why when we see the template.
So please make sure to switch the region to this one.
Next we create a stack and we have to prepare templates.
So multiple options.
We can choose an existing one,
use a sample template and some samples are provided
or build from Application Composer.
But we're going to choose an existing template.
We're going to upload a template file
and let me show you what the file looks like.
So in your course code under CloudFormation,
you have 0-just-EC2.yaml.
And this is a very simple file right now
which has a resource blog and it creates
an instance called My instance.
The type is EC2 instance,
and then you have a few properties.
The first one is the availability zone,
which is US East 1a
and this is why you have to choose US East 1
as your region in CloudFormation service right now.
And the image ID is this AMI ID.
And as you should know,
AMI IDs are scoped within the region.
So for these two reasons, you must be in US East 1
for this hands-on.
The instance type is T2 micro.
And so as you can see we define how to launch
an EC2 instance through this YAML file.
So let's go ahead and actually upload this file.
So select it and then you can actually view it
in Application Composer.
So open this in a new tab.
So here is my template.
So as you can see, Application Composer gives us
a visual understanding of our templates.
So if we look within templates,
we get back the code that we just uploaded right here

and we can switch it to YAML and JSON if we wanted to.
But in the canvas we see that we have
one standard component,
which is an EC2 instance in my instance.
You could double click on it
and view it the fact that it's an EC2 instance as well.
So Application Composer is a nice way
of getting a visual feedback of your templates.
Anyway, back into our stack.
Let's click on next.
We have to provide a demo stack name.
So I'll call this one demo CloudFormation
and then some parameters.
But because we don't define any parameters
in our templates so far, we don't do anything here.
Let's click on next here we have tags.
So I'm just going to show you tag being CFDemo
just to show you how tags work in CloudFormation.
We scroll down, no permissions to set.
We don't touch these options and neither of these options.
So let's click on next and we review and create.
And when we're good to go, let's click on submit.
So as you can see now this template I uploaded
is going to generate some events and that was very quick.
So the events are done
and this actually created a resource right here,
which is an EC2 instance.
So the code got a resource out of it,
and this is why it's called Infrastructure as Code.
So you can click on this EC2 instance.
As you can see right now in the EC2 console,
my instance is running
and you can check the fact that indeed it is a T2 micro
type of instance.
And you can also check the fact
that the AMI ID we have selected is the one
that was entered in our template.
So this is all very convenient.
On top of it, we notice that if you go to our EC2 instance
and we look at the tags,
we can see that some tags were applied by CloudFormation,
which are the name of the CloudFormation,
the name of the logical ID and the stack ID,
but also the name we have specified.
So the tag we specified has name CFDemo
has also been applied to this EC2 instance,
and therefore it's named correctly.
Okay, so now that we have an instance,
let's go ahead and update this stack.
So we click on update

and we replace the existing template.
And this time we're going to choose this
1-ec2-with-sg-eip file.
And if you have a look at it, this file is right here.
So we have a more complete file now.
We have a parameter section
to set the security group description.
We have the EC2 instance section,
which is a bit more complete
because now it has security groups and it has two of those.
We have an Elastic IP here that is attached to my instance.
And we have one security group here defined for SSH rules.
So port 22 is going to be open
and we have the server security group,
which is going to be open on port 80 from everyone
and port 22 from a very specific IP.
So this is a more complete CloudFormation template.
Now let's go back to CloudFormation
and we're going to apply these template.
So here we are prompted with a parameter.
So what is the security group description?
So let's enter demo description.
And as you can see now we are entering some texts,
which is going to make everything vary.
We don't touch the tags, we don't touch anything.
Let's click on next.
And now because we are applying an update,
as you can see, we have a change set.
And so we can preview the change sets,
which is what is going to change
in our CloudFormation stack.
So as you can see, a few things are added.
There is an Elastic IP, there is an SSH security group
and a server security group.
And it's add, so it's new.
And my instance is going to change,
it's going to be modified, and there is a column here
says replacement true.
That means that this EC2 instance is going to be replaced,
that means that the previous one is going to be deleted
and a new one is going to be created.
So it's good to know in case you had some data
on your EC2 instance.
So now let's submit this update and see what happens.
So CloudFormation is a very smart service
because well, from just this code right here
it's able to figure out exactly what to do
thanks to the change sets,
so it knows exactly what to create first.
So as you can see, the server security group

and the SSH security group already are created.
And only then will it update the EC2 instance.
So now the my instance is an update in progress,
and as you can see, it says requested updates requires
the creation of a new physical resource, hence creating one.
So if you go into the EC2 console
and we remove this filter,
now we see that we have two EC2 instances.
This one was created before and this one is the new one
and it was pending and now it is running.
So a new instance got created
and once it's created, again, it's going to be updated here,
the update is complete.
And now we have MyEIP.
So now the Elastic IP is going to be created
and then attached to my EC2 instance.
So if we're quick and we go on the left hand side
to Elastic IP, even though we may not have seen what it is
here is Elastic IP that is properly created,
properly tagged, and it is attached already
to our EC2 instance.
So now if I click on this one
and then I click on networking at the bottom,
you see Elastic IP address here was attached.
So everything was done by CloudFormation and was done well.
And if you look at CloudFormation now in the events,
as you can see, the Elastic IP was created
and then there was a cleanup in progress due in progress.
So that means that the previous EC2 instance right here,
this one got terminated,
which is very handy because CloudFormation
takes care of everything.
So you can go into the resources tab
and see everything that was created through CloudFormation.
On top of it, if you go to the template
and you click on view in Application Composer,
you're able to see now your new architecture.
So we have an EC2 instance connected to an Elastic IP
and connected to two security groups.
So this is very handy
because it gets a visual representation
again of our templates.
So finally, of course, confirmation.
You could go ahead and delete things manually.
You could delete this Elastic IP and so on,
and the EC2 instances manually.
But this is actually not recommended.
You don't wanna touch anything manually
when using CloudFormation.
Instead, you want to either update the templates,

or if you wanted to delete everything, just click on delete.
And CloudFormation will delete all the stack resources.
And the stack resources are also going to be deleted
in the right orders.
So CloudFormation will figure out what to delete first
and so on to clean up everything.
So CloudFormation is a really powerful service
to do infrastructure as code because it's declarative.
You just say what you want and CloudFormation
figures it out, and all the code will control
your infrastructure, which is very, very handy.
So learning how to use it
and write it can be a very good skill in AWS.
So I hope you liked it
and I will see you in the next lecture.

Okay, so now let's talk about the CDK
or AWS Cloud Development Kit.
So this is a way for you to define your cloud infrastructure
using a familiar programming language.
For example, you do not want to use CloudFormation directly
because this is a YAML format.
You like JavaScript or TypeScript, you like Python,
you like Java, you like .NET,
and you would like to just write your cloud infrastructure
using these languages.
So thanks to the CDK, you can do that,
and then once you do so by using your programming languages,
the code will be compiled by the CDK
into a usable CloudFormation template
in JSON or YAML formats.
Therefore, you can deploy your infrastructure
and your application runtime code together
because they can possibly share the same languages.
Which is great for Lambda function,
great for Docker containers in ECS and EKS.
So let's take an example.
So let's choose Python, for example,
as a programming language.
So we're going to write our own CDK application in Python,
and we're going to do define our Lambda function
of that MDB table and maybe for other services in AWS.
Then this CDK application, using the CDK CLI,
is going to be transformed into CloudFormation templates.
And that CloudFormation template,
which is a generated CloudFormation template,
can then be applied into CloudFormation
to deploy our infrastructure.
But the idea is that we want to use

a cloud infrastructure using a programming language,
because it allows us to get, for example, type safety.
Or to have more familiar constructs and so on,
or to go quicker, or to reuse some code,
or to have for loops, these kind of things.
So here's an example of what a CDK code would look like.
So in this example, this is using,
I believe, JavaScript or TypeScript.
We have a VPC that is defined, we have an ECS cluster,
and we have an Application Load Balancer
with a Fargate service.
And so these three things will be compiled by the CDK, CLI
into CloudFormation template that will be usable,
and that you can upload and deploy.
So that's it for the CDK.
I hope you liked it.
And I will see you in the next lecture.

So now let's talk about Beanstalk.
So when we have deployed a web application in AWS,
we typically follow a architecture
called a 3-tier architecture.
So our users talk to a load balancer
that could be in multiple available zones.
Then the load balancer will forward traffic
to multiple EC2 instances managed by
an auto scaling scaling group.
And then these EC2 instances need to store data somewhere
so they will use a database such as Amazon RDS
for a relational database to read and write data.
And if they need to have an in-memory database
for an in-memory cache, then they can also use ElastiCache
to store a retrieve the session data or the cached data.
Now this architecture is something
we can easily reproduce manually.
We can also reproduce it on AWS through CloudFormation,
but there is a better way.
So when you're a developer on AWS,
you don't want to be managing infrastructure,
you just want to be deploying code, okay?
You don't want to be able to configure all the databases,
the load balancers, et cetera,
and you wanna make sure that whatever you're doing scales.
And as we saw, most web applications will have the same
or similar architecture with a load balancer
and an auto scaling group.
So as a developer on AWS, all that you want to do
is to run your code.
Possibly you want to run your code

for different applications and environments the same way.
So there comes Elastic Beanstalk.
So Elastic Beanstalk is a developer centric view
of deploying an application on AWS.
And behind Beanstalk
we have the same components we've seen before.
So your EC2 instances, your auto scaling group,
your Elastic Load Balancer, your RDS database, et cetera.
But in Beanstalk, it's a developer centric view.
So it's one view that's easy to make sense of
with everything in it.
And we still have control over the configuration
of all the components, but it is all within Beanstalk.
So Beanstalk from a cloud perspective
is a platform as a service or PaaS
because we just worry about the code.
So to summarize, we saw infrastructure as a service,
so IaaS.
We saw platform as a service for Beanstalk, PaaS,
and then we're going to see software as a service
for other services on AWS.
So using Beanstalk is free,
but you're going to pay for the underlying instances.
So Elastic Beanstalk is a managed service.
That means that all the EC2 instance configuration
and operating system will be handled by Beanstalk itself.
The deployment strategy can be configured,
but again, the deployment itself
is going to be performed by Elastic Beanstalk.
All the capacity provisioning through an auto scaling group
and load balancing are done by Beanstalk.
And the application health monitoring
and responsiveness is also included
in the Beanstalk dashboard.
So what is your responsibility as a developer?
Well, just the application code,
and that makes Elastic Beanstalk
a very developer friendly service.
There are three architecture models with Elastic Beanstalk.
The first one is single instant deployment,
which is good for development environment,
but if you wanna scale up, you can have a load balancer
and an ASG, which is going to be great for production
or pre-production web applications.
And finally, if you want to have non-web apps in production,
for example, workers, you have an option to only have
an auto scaling group on as a standalone.
Beanstalk can be used to support many platforms.
For example, Go, Java, as well .NET, Node.js, PHP,
Python, Ruby, Packer, Docker, Multi Docker,

and Preconfigured Docker.
You don't need to remember this for the exam,
but as you can see, Beanstalk supports a lot of ways
to deploy your application, including Docker
and many programming languages.
Finally, a question that may come up in the exam
is around health monitoring.
So Beanstalk does have a full monitoring suite available
within the service itself.
And so there's going to be a health agent
on each EC2 instance within Beanstalk
that is going to push metrics to CloudWatch.
And then within Beanstalk you can view these metrics,
do some monitoring and so on.
But it will also check for application health
and will publish health events.
So these are just a bunch of screenshots that I've gotten
to really show you that Beanstalk
is a way to do health monitoring for your applications.
So now I really want to give you an idea
of how Elastic Beanstalk works.
So we'll see you in the next lecture for the hands-on
to give you a concrete demo.

So we have seen CloudFormation and Beanstalk,
but now let's talk about CodeDeploys.
So CodeDeploys is also a way
for us to deploy our application automatically.
Now the difference that CodeDeploy is a bit more permissive.
It doesn't need to be using Beanstalk or CloudFormation.
This is completely independent.
So with CodeDeploy,
We have our application,
it's in version one,
and we want you to upgrade it into version two.
So CodeDeploy will find a way for us to do this.
So CodeDeploy works with two things.
It works with EC2 instances.
And so you can have many EC2 instances being upgraded
from V1 to V2,
but it also works with On-Premises Servers.
So if you have Servers On-Premises,
and you want to help them upgrade from version one
to version two for your applications,
it is possible to do it with CodeDeploy.
So CodeDeploy, If you look at it,
it is what's called a hybrid service
because it works both for On-Premises,

and for EC2 instances.
So CodeDeploy really allows you to work
with any kind of servers,
but you must provision the servers ahead of time.
Okay, It's up to you.
And you must configure them
to install the CodeDeploy agent
that will be assisting you to do these upgrades.
So CodeDeploy is quite a handy service on AWS.
It really allows people to do the transition
from On-Premises to AWS by using the same way
to deploy the application
either with your On-Premise Servers,
or your EC2 instances.
And it is an advanced service,
so I can't really demo it for you,
but what you need to remember is that it allows you to
upgrade both your EC2 instances, applications,
and your On-Premises Servers applications
from version one to version two,
automatically from a single interface.
So hope that was helpful.
And I will see you in the next lecture.

So now let's get
into the code-related tools in AWS.
So before pushing the application code to servers,
you need to store it somewhere.
And developers usually choose a code repository.
Usually that was going to be backed
using the Git technology, G-I-T.
And so a very famous public offering is called GitHub.
But AWS has a competing product.
That is going to be CodeCommit.
And so CodeCommit is a way for you
to store your code
within AWS in a version control repository.
So it's going to be for Git-based repository.
And why would you have that?
Well, once you have a Git-based repository,
it becomes incredibly easy for developers
to collaborate with each other on the code.
The code changes are going
to be automatically versioned and can be rolled back.
And so the benefit is that with CodeCommit,
you have a fully-managed code repository.
It's scalable and highly available,

but also it lives within your AWS accounts.
So it's private, secure,
and it's going to be integrated with all AWS services.
So I will see you in the next lecture.

Now let's talk about
another service called CodeBuild.
So the name is explicit,
it allows you to build your code in the cloud.
So what does that mean?
That means that the source course is going to be compiled,
the tests are going to be run,
and then the output of which is going to produce packages,
and these packages are going to be ready to be deployed
for example, by CodeDeploy
onto servers so that your application can run.
So, as a diagram, what does it look like?
Well, say your code is in CodeCommit,
CodeBuild is going to retrieve this code from CodeCommit,
run some script that you have to define, build your code,
and then you will have a ready-to-deploy artifacts.
So why would you use CodeBuild?
Well it's fully managed and serverless.
It's continuously scalable and highly available
and secure, and with pay-as-you-go pricing,
that means that you only pay
for the time your code is being built.
There are no servers to manage.
And that means that you can really worry about just coding,
and making sure that a service within AWS,
will take its time to build your code
every single time you push a code updates
into your CodeCommit reupholstery.
So that's it, very simple, I hope you liked it,
and I will see you in the next lecture.

So next we have CodePipeline.
So how do we know that
CodeCommit and CodeBuild are connected?
Well, we can connect them using CodePipeline.
So CodePipeline is going to be a way
for us to orchestrate the different steps
to have the code automatically pushed to production?
What does that mean?
Well, maybe we want to define a pipeline that takes the code
builds it, tests it, provision some servers
and then deploys the application on those servers
but it could be more complicated.
And so to orchestrate all these steps

you need a pipeline tool
and this is going to be CodePipeline.
And so you may or may not
have heard of the term CICG, which may stand
for Continuous Integration and Continuous Delivery.
It is the whole concept
that every time a developer pushes the code
into a repository, it is being built, tested
and deployed on to some servers.
So if we look at CodePipeline, say we have code
as an orchestration layer,
it will take its code from CodeCommit
build it with CodeBuild, then decide to deploy it
with CodeDeploy, and may be deployed
into an Elastic Beanstalk environment as one example.
But it's just one way of building a pipeline,
okay, there's so many different ways.
So why would you use CodePipeline?
Well, the benefits is that it's fully managed.
It's compatible with so many services such
as CodeCommit, CodeBuild
CodeDeploy, Elastic Beanstalk, CloudFormation, GitHub
and other third party services and custom plugins.
And it gives you fast delivery and rapid updates.
So it is at the core of the CICD services within AWS.
And so anytime you see orchestration of pipeline
in your exam, you have to think AWS CodePipeline.
So that's it, I hope you liked it.
And I will see you in the next lecture.

So now let's talk about AWS CodeArtifact.
So software packages the developer creates depends usually
on each other to be built,
and so there's like an architecture of software packages.
And so it's called also code dependencies.
And so to store and retrieve these dependencies,
it's called artifact management.
And so traditionally,
you need to set up your own artifact management system
maybe on Amazon S3
or using some custom software on EC2 instances,
and that may be complicated.
So AWS came up with CodeArtifact, which is a secure,
scalable and cost-effective artifact management software,
service for software development, sorry.
And so what that means is that now,
instead of setting up your own infrastructure,
you can just use CodeArtifact,
and all the common dependency management tools
that developers use, such as Maven, Gradle, npm,

yarn, twine, pip, and NuGet,
can just talk to CodeArtifact
to store and retrieve these code dependencies.
So your developers now have a place by default
that's secure to store and retrieve these dependencies,
and that means that once you push your code to CodeCommit
and CodeBuild we'll build it, then CodeBuild
can also retrieve the dependencies directly
from CodeArtifact.
So from an exam perspective,
CodeArtifact is going to be very helpful
if your team needs a artifact management system
or a place to store their code dependencies.
So that's it.
I hope you liked it,
and I will see you in the next lecture.

Next we have AWS CodeStar.
So CodeStar is going to be a unified UI
to easily manage software development activities
in one place.
So you may ask me, "How do I set up CodeCommit
"and then CodeBuild and then CodeDeploy,
"and then I have to unify everything with CodePipeline?"
It sounds like a lot of work, right?
So this is what CodeStar solves as a problem.
So it gives you a one-stop shop
for you to just start a project, a development project,
and automatically, CodeStar, for you,
will give you this neat dashboard.
And in this dashboard and behind the scenes,
it will have created a CodeCommit repository,
a CodeBuild build process a CodeDeploy, a CodePipeline.
There will be some monitoring and so on.
So this is a quick way to get started,
and even a Beanstalk environment, EC2 instances,
et cetera, et cetera.
And as a cool thing and we'll see in the next lecture,
we can edit the code in the cloud directly using AWS Cloud9.
So all of this makes CodeStar a central service
that allows you developers to quickly start
with development while using the best CI/CD practices,
and that's all you need to remember going into the exam.
So hope that it's for you.
I hope you understand it,
and I we'll see you in the next lecture.

Now let's talk about AWS Cloud9.
So Cloud9 is a cloud IDE, which stands
for integrated development environment,

that is used for writing, running,
and debugging code directly in the cloud.
So it looks like a code editor, but it is run in the cloud,
so it is run in a web browser.
So a classic IDE, such as famous ones such has IntelliJ J
or Visual Studio Code, are downloaded
on the computer and installed before being used.
But with a cloud IDE, you can use it
within your web browser, so Chrome, Firefox,
Internet Explorer, whatever you're using.
And that means that you can work on your projects
from your office, from your home or from anywhere
with internet access without any setup necessary,
and you can get back to your work very quickly.
It also allows you to have code collaboration
in realtime and do pair programming.
So as you can see on the right-hand side,
three people are collaborating on the same code
at the very same time from within Cloud9.
So anytime you see cloud IDE, think Cloud9.

Now let's talk about AWS Systems Manager,
also called SSM.
So, SSM helps you man manage your fleet of EC2 instances,
and On-Premises systems at scale,
and yet again,
it is a way to manage your both On-Premises
and AWS.
So, therefore it is called a Hybrid AWS service.
SSM allows you to do a ton of things
that is quite complicated,
but the idea is that
its a Systems Manager,
so you can get operational insights
about the state of your infrastructure,
and you also get access to a
suite of 10 plus products.
You don't need to know all the products
going into the exam,
but the most important products and features are
that you can do automotive patching
of all your servers and instances
for enhanced compliance.
You can also run a command across your entire fleet
of server directly from SSM,
and you can store primary configuration
with the SSM Parameter Store.
Finally, SSM is nice because it works
for Linux, Windows,
Mac OS and Raspberry Pi

So, from an exam perspective,
anytime you see a way to patch your fleet
of EC2 instances or On-Premises servers,
you have to think about SSM,
or if you wanted to run a command consistently
across all your servers,
again, SSM would be the right way.
Now how does SSM work,
just to give you a better idea.
So, SSM service works on its own,
but you need to first install the SSM agent
onto the systems we control,
and that is a small program
that will be running in the background.
By default, if you use Amazon Linux AMI
or some Ubuntu AMI on the AWS,
it will be installed by default.
So, if we look at our EC2 instances
and on-premise virtual machines,
we first have to install the SSM agent
on all of these,
and the SSM agent will be reporting back
to the SSM service in AWS.
As you can see,
it is linked to both EC2 instances
and On-Premises VM,
so this makes it a hybrid service.
Now if an instance cannot be controlled by SSM,
it's only probably an issue with the agent,
and now that the agent is installed
on both our servers
and our EC2 instances,
then we can use the SSM service
to run commands across all these servers,
or we can patch them all at once,
or we can configure them consistently
using the SSM service.
So, if you remember this,
you're good to go for the exam.
Again, SSM is an advanced service,
so I won't do a hands-on on it,
but from an exam perspective,
you should be good to go.
So, I hope you liked this lecture,
and I will see you in the next lecture.

Okay. So now, let's discuss
the SSM Session Manager feature of Systems Manager.
So this allows you to start a secure shell
on your EC2 instances and on-premises servers

without having SSH access or the need for bastion host
or any SSH keys.
That means that the port 22 on your EC2 instances
is going to be closed because there is going to be
no need to do SSH to establish a secure shell
onto your EC2 instance.
That means better security.
So how does that work?
Well, the EC2 instance that we're going to show you
in a second has an SSM Agent
and that agent is connected to the Session Manager service.
So that means that our users can access
through the Session Manager service
these two instance and execute some commands on it.
This has support for Linux, macOS and Windows
and we can send log data to Amazon S3 or CloudWatch Logs
to make it super secure.
I think it'll make a lot more sense
when we do the hands-on so let's do it.
So let's have a look at the SSM Session Manager service,
but first, we need to launch an EC2 instance.
So let's launch an instance right now
and I will scroll down.
We will choose Amazon Linux 2 AMI, t2.micro.
We will not use any key pair.
And then I will disable SSH traffic.
So as you can see, my EC2 instance will have
a security group that allows nothing.
No HTTP, no HTTPS, no SSH.
Yet, we will be able to use an SSM Session Manager shell.
So what I need to do actually is
to attach an IAM instance profile to my instance
to allow it to talk to the SSM service.
So I have a few right here,
but I'm going to create a new one just to show the process.
So click on create a new IAM role profile.
Then let's create a role.
For a service on AWS, it will be Amazon EC2.
Click on next. Then, for permissions,
we're going to filter for SSM
and we will choose the Amazon SSM managed instance core.
So let's select this and click on next.
And then I will call it demo EC2 role for SSM,
which allows the EC2 instance to use this policy
to talk to the SSM service.
So this is necessary for my instance
to be managed by the SSM service
and so we can use the SSM Session Manager feature.

So now that this role is created,
we're going to refresh here and we're going to look
for this demo EC2 role for SSM.
Perfect. So let's go ahead
and now, create and launch our instance,
which is now launched
and it's going to boot.
And so the thing I have to check now is the SSM service.
So let's go into Systems Manager.
And on the left hand side,
you're going to look for Fleet Manager
and Fleet Manager is a service where all the EC2 instances
that are registered with SSM will appear here.
So they're called managed nodes.
And as you can see right now,
we don't have any managed nodes,
but we need to wait for the EC2 instance to boot up
and then it will appear right here.
Okay. So I just refreshed and as you can see,
there's one instance. It's running.
We can see the SSM Agent is online.
We can see the platform, the operating system,
Amazon Linux 2, the SSM Agent version
and then links to the EC2 instance if we wanted to.
Okay. So as soon as our instance is under Fleet Manager,
that means that we're ready
to run a secure shell against it.
So to do so, I will scroll down
and I will look for Session Manager.
Now, Session Manager is a way for us
to access our Linux instances and Windows instances.
So let's start a session.
And as you can see here, okay,
my EC2 instance under the security group
does not have any inbound rule.
Okay. So zero inbound rules in here.
So let's start a session on this EC2 instance and start it.
And the idea is that we're going to get a secure shell,
as you'll see in a second.
Here we go, we have a secure shell
and I didn't need to have SSH access
so I can do ping google.com
and this command is going to work, obviously.
And if I do host to get the host name, so host name,
we can see that this is IP 172-31-1-148,
which is exactly corresponding to,
and if you go into networking,
the private IP address of my instance.

So that means that using the SSM secure shell,
we're able to have indeed a secure shell directly
from AWS without having SSH security keys and SSH access.
So to summarize, we have three ways
of accessing our EC2 instance.
Number one is to open the port 22 and then use SSH keys
and with a terminal to do the SSH command.
Number two is to use EC2 Instance Connect
and that didn't require to get SSH keys
because they will be temporarily uploaded
onto the Amazon EC2 instance if we need to.
So that was number two, but this required still
the port 22 to be opened on our EC2 instance
to have access with using EC2 Instance Connect.
And then we've explored the third option
which is Session Manager.
So we need to make sure that we had an EC2 instance
with Amazon Linux 2 and make sure
that this EC2 instance had an IAM role
and this IAM role, okay, as you can see under security,
had an IAM role and this IAM role allowed access
from the EC2 instance to Systems Manager
and this is what allowed us to get the secure shell running.
So that's it for this lecture.
We can terminate this session and the cool thing is
that the session history will be saved in terms of logs.
So we can see the session history right here.
So this is awesome.
And to finish, just take your instance and terminate it.
That's it.
I hope you like this lecture
and I will see you in the next lecture.

So let's talk about the systems manager
parameter store.
So it's a way for you to store configuration
and secrets securely on AWS.
You can store anything you want, such as API keys,
passwords, configurations, and it's serverless.
That means you have nothing to provide.
It's scalable.
It can respond to many API calls all the time.
It's durable and it's very easy to use.
On top of it, it's secure
because you control access to each parameter
in the parameter store using IAM.
On top of it, your configurations can evolve,
your parameters can evolve

and so therefore you have version tracking
and optional encryption.
So we'll see in a second.
But in the parameter store, our applications
or our users can enter plain text configurations
or encrypted configurations
in which case it is encrypted with KMS.
And so therefore you can manage
and centrally store the configurations
of many of your applications in one place.
So I am in systems manager, and on the left hand side
there is the parameter store.
And as you'll see, it's extremely easy to use.
So going to create a parameter
and the name is going to be demo parameter.
Then you can have a different tier, so standard or advanced.
So the free one is standard.
We'll use standard and then the type of parameter.
So is it a string
meaning it's a value that is, for example, a configuration
or a list of values if you wanted to, or a secure string.
And secure string is when you want to encrypt something
in the parameter store because you're, for example
going to add in API keys or passwords and so on.
So in this case, using a secure string is much better.
But for simplicity's sake, let's use a string.
And so you can choose the data type, so it could be a text
or an image of type EC2, but we'll use text and say, hey
the string is my configuration parameter.
And then that's it.
You just create this parameter and there you go.
In your parameter store
you have this parameter named demo parameter
that you can just click on
and then retrieve the value of value.
My configuration parameter on top of it.
If you were to edit it, there would be different versions
which is very helpful for tracking them over time.
And that's it.
The parameter store is a very simple service
but a very helpful one in AWS.
When you're done, you can just delete this parameter
and you're back where you started.
Okay, so that's it for this lecture.
I hope you liked it, and I will see you in the next lecture.

Okay, so let's summarize
everything we've learned about deployment on AWS.

So first, CloudFormation, which is an AWS-only tool,
which is allowing you to do infrastructure as code
and works with almost all types of a AWS resources.
This allows you to create templates,
and these templates can be used
to deploy infrastructure on AWS
and to use these templates across different regions
and accounts to make your infrastructure truly repeatable.
Then we have Beanstalk,
which Is again a AWS only, so AWS Beanstalk,
which is a platform as a service, or PaaS,
and it's limited to certain programming languages or Docker.
You can deploy your code consistently
with a known architecture,
for example, using the combination of a load balancer,
EC2 instances, and an RDS database.
CodeDeploy is to deploy and upgrade
any applications onto servers.
And this can be done on AWS,
for example, your EC2 instances,
but also can be done on your on-premises infrastructure.
And this is why it's called a hybrid type of service.
Systems Manager, yet again, is a hybrid type of service,
which allows you to patch, configure,
and run commands at scale across all your servers.
So that's for all the deployment services,
but now let's talk about the developer services.
We've learned about CodeCommit,
which does come up in the exam,
which is to store code in a private git repository
giving you a version controlled code repo.
CodeBuild, which is allowing you to build
and test your code in AWS in a serverless fashion.
CodeDeploy, which is allowing you
to deploy code onto servers.
So, I bundle it both into the deployment
and the developer services, because they go along.
CodePipeline to do the orchestration
of a pipeline within AWS.
So, from your code, to your build to your test,
your deployment, and your provisioning, and so on.
CodeArtifact to store software packages
and dependencies on AWS.
CodeStar, which is giving you
this unified view one-stop shop
to allow developer to do CICD and code directly.
And finally, Cloud9, which is a cloud IDE,
so integrated development environments,

which allows you to edit the code
directly from within a web browser
and has some collaboration features
to do pair programming for example.
And then we have the CDK to define your cloud infrastructure
using a programming language, for example,
JavaScript, TypeScript, Java, Python, et cetera.
And this gets compiled into a CloudFormation templates.
So that's it I hope you liked it,
and I will see you in the next section.

#_____Section 13_____

Okay, so now let's talk about
Cloud Integration.
So when we have multiple applications,
at some point they will have to communicate
with one another.
And there are two types of patterns
to make applications communicate.
The first one is easy to understand,
it's called a synchronous communication
in which an application talks to another application.
For example, you have created a service to buy something
and then you need to talk to a service
that ships what has been bought.
And so therefore you want to integrate the buying service
and the shipping service synchronously
because it talk directly to one another.
That's the first way, but there's a second way,
which is called asynchronous or event based,
for example when we have a queue to talk to.
So let's have an example,
our buying service this time, anytime something is bought,
will put an order in a queue.
And the shipping service will be reading
from the queue to get the orders.
As you can see in this example,
your buying service and your shipping service
are not directly integrated with one another.
There are something called decoupled
because there is a queue in between to talk to.
And this allows us to get some nice integration patterns.
So if we get synchronous communication
between an application and another one,
it could be a problem.
For example, what if you have a sudden spike of traffic?
What if you need to encode 1000 videos, but usually it's 10?
In this case, the service you are talking to
may get overwhelmed and you may get an issue

to encode these 1000 videos and things may fail.
In that case.
It's a lot better to decouple your applications
and to use something like SQS, which is a queue model,
or SNS, which is a pub/sub model,
or Kinesis, which is used for real time data streaming.
Now these services, once they're decoupled,
they can scale independently for our applications.
And so this is great.
So in this section,
we're going to have a deeper look at SQS and SNS.
So I hope you like this
and I will see you in the next lecture.

So let's first talk about their first service
which would allow us to decouple our applications
which is Amazon SQS,
which stands for Simple Queue Service.
So what's a queue?
Well, say we are creating an SQS queue right here.
What we enable us to do is to have producers
send messages into that queue,
and then it could be one producer
but it could be as well, multiple producers.
And then once the messages are stored in the queue,
then they could be read by consumers
who will be polling the queue.
That means requesting messages from the queue.
They will be polling the queue
and it could be one consumer.
They could also be multiple consumers.
And in this example, once the consumer poll messages
they will share the work.
So each consumer will get different messages
and when they're done processing a message
maybe for example, to process a video
then they will delete the message
from the queue and it will be gone.
So in this mechanism, we have the producers sending messages
into the queue and they're decoupled from the consumer
reading the messages from the queue
and processing them at different speed.
So when you have an SQS
turns out that it is the AWS oldest offering.
It's over 10 years old.
It was one of the first services to appear
as part of the AWS cloud.
It is fully managed, so it's a serverless service.
You don't provision servers

and it's used to decouple applications.
So this is the exam tip.
If you see decouple, then think of SQS.
It will scale seamlessly from example, from one message
per second to tens of thousands of messages per second.
And the default retention
of the messages is four days, maximum of 14 days.
So you need to process them within that default retention.
There's no limit to how many messages can be in a queue.
And then once the consumers read the messages
then they have to be deleted.
So they're gone.
There is low latency, we're talking about less
than 10 milliseconds, unpublish and subscribe.
And then consumers share the work to read messages
and scale horizontally.
So SQS can be used to decouple
between your application tiers.
So here's a classic solution architecture.
We have our web servers and they're taking request
maybe through an application balancer.
They're served through EC2 instances
in an auto scaling group.
And then, for example, say that our users
want us to process some videos.
Then instead of sending it directly
to the video application, we can instead
insert messages into an SQS queue,
and then we will have a video processing layer
made of an auto scaling group with EC2 instances.
And these EC2 instances, we'll be reading from the SQS queue
and processing our videos.
The cool thing about it is that we can scale the second auto
scaling group independently from the first one
and this is why it's called decoupling.
And on top of it, the scaling can happen based
on how many messages, for example,
there are in the SQS queue.
And this would really allow us to have two layers,
the web servers and the video processing fully decoupled
from the SQS queue and scaling independently.
This will give us the best user experience
and also the best cost efficiency and scaling concerns.
Another feature for Amazon SQS is to have FIFO queues.
So FIFO means First In First Out, and that corresponds
to the ordering of the messages within the queue.
So if you have a producer sending messages
in a specific order such as 1, 2, 3, and 4
then the consumer will also read these messages in order.
So example 1, 2, 3, 4.

So when you have a normal SQS queue
consumers can read messages altogether
and they could be in different orders.
But with Amazon SQS FIFO queues
the message are going to be in order
and it's just a feature you need to remember for the exam.
Okay, that's it for this lecture.
I hope you liked it and I will see you in the next lecture.

So let's go and practice the SQS service.
So in SQS we see it's a message queuing service
and the idea is that we have a high-throughput
system-to-system messaging service.
So we're going to go ahead and create a queue,
which is going to be free.
So as you can see, we have two types of queue
we can create, a standard queue or a FIFO queue.
Now this is way beyond the scope of the exam
for the Cloud Practitioner.
So I'm just going to do standard
and I will call my queue demo-sqs.
In terms of the configuration, you can safely don't look
at all these things, again, it's more advanced
than we need to know as well as the access policy,
we can leave everything as default.
So what I'll just do is create the queue.
Now my queue is created, and we can start sending
and receiving messages from it.
So let me close these panels right here.
So we have all the configuration panels here, but again,
not in this scope, so let's not do,
deal with them right now,
but here we have our queue details where we can find
the type of queue it is, the name,
whether or not we enable encryption and so on.
And we if we click on more,
we get some more information and we get some information
around how many messages are available,
how many messages are in flight,
how many messages are delayed?
So what we're going do is to produce and consume permits.
So for it we'll go on the top right hand side
and there is send and receive messages.
So once I click on send and receive messages,
you can send a message
so we'll send the message, hello world.

And then we can just click on send message.
Now as you can see, the message has been sent
and it's ready to be delivered,
received and if you click on view details,
you can get some information around that message,
so the message ID as well as the MD5 attributes,
but more importantly, if we scroll down,
we can see that now in the receive message section,
one message is available.
So if I just do another one, so I say another hello
and press send message, now you see that
the messages available is two.
So now we need to poll the queue to receive these messages.
So click on poll for messages,
and this will go ahead and retrieve these messages for me.
As we can see, the two messages here are showing up
and on application, we click on them,
we'd actually read them, so look at the body and say,
okay, this looks like hello world,
which is the exact message I sent.
We could look at some attributes and so on.
If there were say, if there were some sets,
but what the application will do is that it will read
the messages and this is the application you would have
to code obviously and then when the message processing
is done, you would click on the messages
and then delete them to remove them from the queue
and as you can see, messages available is zero.
Very, very simple, but that's how SQS works at a high level.
Now, if you go back to your queues,
you will get a list of all your queues
and to finish this hands on, you can go ahead
and delete this queue.
This won't cost you any money, but it's not bad
to clean up after yourself.
So that's it.
We should enter the word delete, sorry.
So that's it for this lecture, I hope you liked it,
and I will see you in the next lecture.

Now let's talk about Amazon Kinesis.
So, for the exam,
Kinesis is equal to real-time big data streaming
and it's all you should know.
And I'm going to give you a little bit more information
because there will be not enough, I guess,
for your own understanding.
So Kinesis is as a managed service used to collect, process

Cloudwatch
and analyze real-time streaming data at any scale.
It's too detailed to know all these things
for the Cloud Practitioner Exam
but still I'm going to give them to you.
It's good to know.
So there's, Kinesis data streams
which is a low latency streaming service to ingest it
at scale from hundreds of thousands of sources
and the source could be whatever can produce data
for example, a truck, a boat, an IOT device
whatever you can think of.
Then you have Kinesis Data Firehose
which is to load these streams
into places that we know already, such as Amazon S3,
Redshift, ElasticSearch, et cetera, et cetera.
We have Kinesis Data Analytics
which is to perform real-time analytics
on the stream using the SQL language.
And finally Kinesis Video Streams
to monitor real-time video streams
for analytics or machine learning.
So at the CCP level, all you need to know
is that Kinesis is used
for real-time big data streaming,
but the sub-services of Kinesis
really line up this way.
So we have Amazon Kinesis streams
that are going to be getting data
from click streams, IOT devices, metrics,
log servers, all these kind of things.
Then we can use Kinesis Data Analytics
if we wanted to analyze this data
and produce output in real-time.
And then we could use Kinesis Firehose
to send these outputs directly into destinations
such as an Amazon S3 buckets
or an Amazon Redshift database
where we can analyze this data
and perform more analytics down the road if we wanted to.
So that's it for Kinesis, I hope you liked it.
And I will see you in the next lecture.

Okay, so now let's talk about
the second way we can decouple our application using SNS.
So what if you want to send one message to many receivers?
We can go the route of direct integration.
For example, having a bank service
sending an email notification
then talking to a fraud service,

talking to a shipping service
and talking to the SQS queue.
But that would be quite complicated
because we need to write four direct integrations.
Instead, we can use something
called a pub/sub type of integration
in which we have an SNS topic
and the buying service will be sending a message
into our SNS topic.
And the topic automatically will be smart enough
to send a notification via email to the fraud service,
to the shipping service and even to an SQS queue.
So this is the premise of SNS.
SNS stands for Simple Notification Service
and the event publishers will only send messages
to one SNS topic.
And you can have as many event subscribers as you want
to listen to the SNS topic notifications.
Each subscriber to the topic will get all the messages.
So this is different from SQS
where the consumers were sharing the messages.
In this example, each subscriber to the topic
will get all the messages sent to the SNS topic.
Each SNS topic can have more than 12 million
subscriptions per topic
and also we have a soft limit
of 100,000 topic limits for each account.
So SNS has many destinations,
it can publish to many subscribers
and from the AWS services side
we have SQS, Lambda and Kinesis Data Firehouse
that can be all targets of our SNS publish action.
But also we can send emails directly from SNS.
We can send SMS and mobile notifications
and finally, we can send data directly
into an HTTP or HTTPS endpoint.
So bottom line is, any time in the exam
you see notification, publish subscribe,
subscribers, et cetera, et cetera.
Then think about Amazon SNS.
Okay, this is the first lecture,
I hope you liked it.
And I will see you in the next lecture.

Okay, so let's practice using SNS.
So I'm going to go into
the simple notification service console,
and I'm going to create a topic called demo SNS.
Now this topic will leave it
as default for all the options,

and we'll just click simply on create topics.
So the idea is that we want to send messages
to this SNS topic,
and we want subscribers to this topic
to receive the messages.
So we can define subscriptions here currently we have zero,
but it will create a subscription,
and as you can see
we have different protocols for your subscription.
It could be HTTP, HTTPS, Email, Email-JSON, SQS and Lambda.
So lots of different targets for your SNS,
and as you remember, if you send a message to one SNS topic,
then all the subscribers whatever they might,
may be in terms of protocol will receive that message.
For this hands on
we'll just go with email because it's easy.
And for the endpoints,
I'm going to go for stephaneccpdemo@mailinator.com.
Now if you don't know this,
this is a service to get a temporary email address.
So if I click and enter,
if I enter this defense CCP demo mailbox,
this is a temporary mailbox for me.
So I'm going to use this mailbox,
and I'm going to create a subscription.
And now if we go back to the demo SNS topic,
we can see that the subscription is pending confirmation.
Very easy for this to fix it,
I'm gonna go into my Mailinator mailbox and as you can see,
I have an email from AWS,
asking me to confirm my subscription
which I will do right now.
So my subscription ID is confirmed,
and what you could do,
if you are savvy would be to create other subscriptions.
They're more complicated to create besides email,
so we'll just keep it as one for now.
But you could have many subscribers for your SNS topic.
So now what I'm going to do is
to practice sending a message to my SNS topic.
For this I'm going to publish message,
say demo subject line,
and then I will say hello world in terms of payload.
And then finally I'm going to publish this message,
and the message has not been published successfully.
And so all the subscribers in my subscription
should receive that message.
So if we go into Mailinator, go back to my inbox,
I can see that moments ago I received a new email
called demo subject line,

and the content says hello world, so this has worked.
So this is a very simple demo of how SNS works.
And when you're done, you can go ahead and delete the topic.
If you want it too, it won't cost you any money.
So that's it I hope that was helpful,
and I will see you in the next lecture.

So now let's talk about Amazon MQ.
So we know about SQS and SNS
and they're cloud-native services
because they are proprietary protocols from AWS.
They use their own sets of APIs.
But if you are running traditional application on-premises,
you may use open protocols
such as MQTT, AMQP, STOMP, Openwire, WSS.
And when you're migrating your application to the cloud,
you may not want to re-engineer your application
to use the SQS and the SNS protocols or APIs.
So instead, you wanna use the traditional protocols
you used to, such as MQTT, AMQP, and so on.
So for this, we can use Amazon MQ.
So Amazon MQ is very simple.
It's a managed message broker service for two technologies,
for RabbitMQ and for ActiveMQ.
So RabbitMQ and ActiveMQ are, for example,
on-premises technologies that provide you access
to the open protocols I just mentioned.
So then we can get a managed version of these brokers
onto the cloud thanks to Amazon MQ.
So as such, you can understand the implications.
First of all, Amazon MQ doesn't scale as much as SQS or SNS,
which have sort of like infinite scaling.
And because Amazon MQ runs on servers,
you may have server issues.
And so you can run multi-AZ setup
with a failover if you want to be highly available.
Also, Amazon MQ comes with both a queue feature,
so it looks like SQS,
and topic features, it looks like SNS
as part of a single broker.
So Amazon MQ is going to be used only and only if
a company is migrating to the cloud
and needs to use one of these open protocols,
such as MQTT, AMQP, STOMP, et cetera, et cetera.
Otherwise, it should be using SQS and SNS
because they scale a lot better
and they're way more integrated
with Amazon Web Services than Amazon MQ.
So I hope that helps and I will see you in the next lecture.
Okay, so let's summarize everything

we've learned in this section.
So SQS is a queuing service in AWS
and we can have multiple producers into an SQS queue.
The messages are going to be kept
up to 14 days in the queue, then deleted.
Consumers can read these messages
and they will share the reads.
So they will split the reads
and then once a message is read and processed
then it's going to be deleted.
So when a consumer is done, the message is gone.
It's used to decouple applications within AWS.
So anytime you see queuing and decoupling, think SQS.
SNS is a notification service in AWS.
And so we have producers and subscribers.
So it could be email, Lambda, SQS queues, HTTP, mobile
and if you have multiple subscribers onto one SNS topic
then SNS will send a message to all of them.
SNS doesn't retain any message
so it's not a durable store of messages
and it's going to be used again for pub/sub,
for subscribers, for topics and notifications within AWS.
Finally, we've seen Kinesis
which is a real time data streaming service
which is going to have some data persistence
and also you can run analytics
on top of Kinesis in real time.
And finally, we have Amazon MQ
which is a managed message broker
for Active MQ and RabbitMQ in the cloud.
So if you want to migrate from on-premises
to the cloud and use the MQTT, AMQP,
or other kind of protocols, this is a service for you.
So that's it for this section.
I hope you liked it and I will see you in the next section.

#_____Section 14_____
Okay, so let's go into CloudWatch.
And we're going to look at first CloudWatch metrics.
So let's go onto the right hand side, onto metrics,
All metrics, and here we have some information
and we can try the new interface. Okay.
So, here we have some information around all the metrics
that we have created in this course, and you may have
a different screen, but as you can see, lots of services.
We're already publishing some metrics into CloudWatch,
which is pretty cool.
So for example, I can click on the SQS, Queue Metrics,
and look at this demo SQS queue.

And, we can see that the number of messages received
has like, one data point right here.
If I look for example, at number of messages deleted,
it doesn't show anything.
Number of sessions sent, here's one, and then we can have,
for example, Number of Empty Received and so on.
So you can get a lot of information for SQS.
But, let's have a look for example for EC2 instances.
So let's go into EC2, Per-instance Metric,
and then we can look at the CPU utilization of my instance,
and I've just launched an instance before, so again,
it will show you some metrics.
So you need to, you know, have it (stutters)
running for a longer period of time
to get some information out of it.
But at least it shows you that within the dashboard
of culture metrics, you can get a lot of information
for a lot of different services, okay?
Next, what we want to do is to create an alarm.
So let's go through the process of creating an alarm.
So, let's go to All alarms and create an alarm.
We need to first select a metric, so let's find
that CPU utilization metric for our EC2 instance,
we'll select it.
And, then, we can say whether we want, we want to consider
the average, the sum, the maximum zone,
so we have different settings for this alarm.
We'll keep it very simple, exhibit for the average
over the five minute period,
and while saying, hey, if the CPU utilization
is greater than 80%, okay, then please
treat this as a alarm.
But you could say Alarm, OK, or Insufficient data,
but we'll say it's an alarm.
In which case, well, you need to send a notification
in to a new topic of SNS called
the Default_CloudWatch_Alarms_Topic,
and then the recipient will be stephan@example.com.
Let's create this topic.
And now, any time the Cloudwatch alarm is going to go
into the alarm state, it's going to send an email
into this SNS- sorry,
a notification to this SNS topic,
which will in turn, send an email to stephanie@example.com.
So this is pretty cool, cause we have notifications,
but also we have other options to set up
Auto Scaling actions, EC2 actions,
or Systems Manager actions.

But again, to keep things very simple,
let's just consider only the notification.
Click on Next.
Then, this one is my DemoAlarm.
Click on Next.
And then as you can see,
this is the alarm threshold, the red line,
and the blue dot is the actual value,
so as we can see we're, far below the actual threshold.
So let's click on Create alarm and here we go,
I have created my first alarm.
There's another way you can create an alarm,
it's directly from within the EC2 console for EC2 instances.
So I am in my EC2 console,
and this is the instance I created from before,
so just a very simple instance with no configuration.
Okay. And I just launched it.
And if you go first under the Monitoring tab, here.
As we can see, we have some information around
all the monitoring of our EC2 instance,
So we get CPU utilization, status checks, and so on.
So lots of information right here.
And, what we can do here, is that we can add an alarm
to our EC2 instance.
So to do so, you scroll to the right hand side
and there is an alarm status,
and you can click on this plus button right here
to create an alarm.
So here we manage CloudWatch alarms for our EC2 instance.
We're going to create an alarm for it.
And then we're going to say, hey,
the alarm notification should go to this topic
that we've created from before.
And, then there should be an alarm action.
And, the alarm action is going to be
recover my EC2 instance.
So we're going to say, hey,
I want to recover my EC2 instance, when,
and then you do say the data to sample.
So we'll choose Status check failed: either.
So when either status check is failing,
please recover my EC2 instance.
So this is a very handy way
to create an alarm for CloudWatch.
And then, click on Create.
And this is not working,
because we need to choose (stutters)
Systems check failed for the metrics.

So, always good to be wrong sometimes.
So, Status check failed: system, and Create, and here we go.
Now I have created a CloudWatch alarm on my instance.
So now we have two alarms for my CloudWatch instance,
uh, for my EC2 instance.
Great. And we can create a third alarm if you wanted to,
For example, we're saying, hey, if, then,
so we can notify to the same topic
and we're going to add an alarm action to reboot this time.
And we'll say, hey, if the CPU utilization is greater
than 95% for three consecutive periods of five minutes,
then we're assuming that's, hey, I think
our EC2 two instance is stuck on a CPU solution loop.
Therefore, please reboot it.
So this is just a use case.
So as we can see we are far below,
and we can create this alarm.
And again, now we have three alarms
attached to our EC2 instance,
which is quite good.
Okay, and if you go back into the Cloudwatch console,
as you can see, and refresh this page,
now we have three alarms.
So there's another kind of alarm
that we can not create in all the regions.
You can actually create it in only one region.
And this region is us-east-1
because it is a billing alarm.
So as you can see on the left hand side right now,
there is no billing alarm.
To make billing alarm appear here,
you click on the region selection
and go into us-east-1.
So going into us-east-1,
as you can see on the left hand side now, there is billing.
And, now we can create a billing alarm,
which is to alarm us based on the billing
for specific services.
So, we're going to create an alarm.
And then when you specify a metric,
so there's going to be estimated charges of USD.
And in here you can say, okay, if my charters
are greater than eight USD, then go into an alarm state
and then, send a notification into a new topic,
so we'll call this one-
this one is the default topic for billing,
and then against it's stephan@example.com.
So we need to recreate an SNS topic

because we are in another region.
And obviously, every resources in the database
is scoped by region. Okay.
Almost every resources.
So we need to create this topic, here we go,
and click on Next and then DemoBillingAlarm.
Click on Next.
And we're good to go.
So again, just to show you that billing data
is only available in us-east-1,
and this could come up in the exam, okay.
So as we can see with alarms and metrics,
we can do a lot of automation in AWS, which is quite handy.
So just to finish with this hands-on,
we just need to clean up what we have.
So let's clean up this alarm right here, the billing alarm.
And then please make sure to go back
to the region where you were working from before.
So for me, this is Ireland.
And what I'm going to do is delete all my alarms,
so I'm going to delete my three alarms right here.
So Actions, Delete.
And then I'm going to also go into my EC2 instance
and terminate it.
So that's it for this hands-on.
I hope you liked it.
And I will see you in the next lecture.

Now let's talk about Amazon CloudWatch Logs,
so CloudWatch Logs, as the name indicates,
is to collect log files.
So what is a log file?
Well, when you have an application running
on any server, usually you want the application
to write some text about how it is doing.
For example, when it is doing some actions for a user
when it is performing some clean ups, et cetera, et cetera,
and so all these logs can be collected
and when a user needs to troubleshoot something,
then they will go through the log file
and see what the application did or said,
and so these logs can exist in different forms,
but you can collect the logs from Elastic Beanstalk.
You can collect the logs from ECS, Lambda, CloudTrail,
CloudWatch logs agents,
which is when you install a log agent
on an EC2 machine or an on-premise server

to get your log directly from that server onto AWS,
or Route53 for logging DNS queries,
and CloudWatch logs overall,
when they collect all these logs
it allows for real-time monitoring of your logs
and then with it you can react to whatever is happening
within your logs.
Also, your logs can be readjustable for the retention.
That means that you can have your logs,
for example, for just one week
or 30 days or a year or infinitely.
So how does CloudWatch Logs work for EC2 instances?
By default, your EC2 instances will not send any log files
to CloudWatch Logs.
For this you need to create a CloudWatch Log agent
on your EC2 instances,
and they will push the log files that you want
onto the CloudWatch Log service,
so to summarize, the CloudWatch Logs service is running.
Here's your EC2 instance.
We will install the CloudWatch Logs agent
and this agent will send the log files
directly into CloudWatch Logs.
For this to work we need to make sure that our EC2 instance
has a proper instance role with the correct IAM permissions
to send the log data into CloudWatch Logs.
Then, the log agent can also be setup on on-premises servers
as well, so this is a hybrid agent.
It works both on-premises or on AWS
and it allows you to collect logs
from both your EC2 instances and your on-premises servers
directly into the CloudWatch Log service,
so that's it.
Let's go in the next lecture
to see how CloudWatch Logs works

So let's have a look at CloudWatch Logs.
And for this, on the left hand side
let's go into Log groups.
So as you can see
we already have one log group available to us, which is,
AWS lambda, demo-lambda.
Well that's because before we created a lambda function
that we ran and that automatically
created some log into CloudWatch logs.
So if we click on this function,
and this is what it was doing, he was running this code.

And so if we go into CloudWatch logs
and click on this log group,
we can see we have one or more log streams.
So here's my log stream.
And within log stream, as you can see,
this represents all the log lines that were logged by
my lambda function. So we have the request ID,
the loading function. Then we get some values,
so value one... and so on,
then end of the request ID and end of the reports and we can
play with it. So that means that
any log line logged by my lambda is going to appear here.
So for example, I can change my lambda function,
and say, "print an extra log line",
and then we're going to deploy this and then I'm going to
test my changes.
So we're going to test it with this test events and then
create, and then demo events perfect
and create now tests.
So now my function is running.
And so what should happen is that
if we go back into CloudWatch logs,
go to my log groups and then within log group,
we have a new log stream. This log stream right here,
contains the log line and extra log line. Okay.
So this is excellent. And this is good for just monitoring,
obviously information about your lambda functions,
but also what happens if there's an exception.
So in this case, what I'm going to do is
I'm going to put a hash sign
in front of this line to comment it.
And now we're going to remove the hash sign
from the line 13,
to have this raise exception, something went wrong.
We're going to deploy the changes again
and test our function.
So now something went wrong and what's going to happen is
that if I go into my Lambda,
there's a new log stream, click on this log stream.
And in here, I'm able to see that exception right here,
and we can look at why our function went wrong.
So this is very handy.
That means that every time a Lambda function is going to run
the logs are going to appear in CloudWatch logs.
But this is not just true for lambda functions.
Obviously this is true for any kind of logs that you want to
appear in CloudWatch logs.
So CloudWatch logs is very handy because you can

troubleshoot maybe errors in your programs
or applications and so on by just looking at the log lines,
within your log groups,
for your services or for your applications directly within
CloudWatch Logs.
And then you can do some login analytics, similar things,
some monitoring and so on. Okay.
And this is super cool again,
and something to definitely go deeper into when you are a
DevOps in a company. So that's it for this lecture.
I hope you liked it. And I will see you in the next lecture.

So now let's talk about Amazon EventBridge,
and it used to be called CloudWatch Events.
So if online, you see CloudWatch Events,
think about Amazon EventBridge and vice versa,
but EventBridge is the new name.
So with EventBridge, you can react to events
happening within your AWS accounts.
And one use case for them is to schedule cron jobs.
So you want to have a script scheduled on a regular basis.
For example, in EventBridge, you can create a rule that says
that every one hour you should have an event created
and that event will trigger a script
running on a Lambda function.
Effectively, you've done a serverless cron job.
But you can also not just react
to events happening every hour,
we can react to a service doing something.
For example, say you wanted to give alerts
to your security team whenever someone
is going to log in using the root user
because, well, the rule is that you should not reuse
the root user or only very rarely.
So maybe we want to react
to the IAM root user sign-in events
and then send this into an SNS topic
that is combined with email notifications.
In that case, whenever someone signs in,
we will receive an email.
So you can create a lot of these different integrations
all using Amazon EventBridge.
And you can, as the destination, trigger Lambda functions,
send SNS and SQS messages, and so on.
Actually, you can do all these things.
I just wanted to give you a quick example.
You don't need to linger on this slide, okay.
But the sources can be anything you want.
Could be EC2 Instances, CodeBuild, S3 Event,

Trusted Advisor, and so on.
There's a lot and, of course, a schedule.
It goes into EventBridge, and from EventBridge, you can send
and trigger many different kinds of destinations.
Would it be for compute, for integration,
orchestration, maintenance, and so on?
And what I've shown you here is just a little sample.
Finally, EventBridge has more capability.
So what I've shown you is called the default event bus.
And these are events happening from within AWS Services,
or, for example, your schedules.
But it is possible for you to receive events
from partners of AWS.
For example, if you're using Zendesk, or Datadog,
or others that are partnered with AWS,
then they can send their own events
into your account through a partner event bus,
and, therefore, you can react to events
happening outside of AWS as well.
And finally, you could plug in your own custom applications
that would send their own event to your own custom event bus
to write any kind of integration you want
and be really able to customize everything.
EventBridge also has more capability.
There is the Schema Registry to model the events schema
to see what it looks like, the data types and so on.
You can also archive all the events sent to an event bus
indefinitely or for a set period,
and then you can replay these archived events.
But I think that for the cloud practitioner exam
you know more than enough.
Again, you need to remember,
conceptually, what is Amazon EventBridge?
And now, I think you do.
Okay, that's it for this lecture.
I hope you liked it, and I will see you in the next lecture.

So let's practice using Amazon EventBridge.
And for this we're going to go into the EventBridge console,
and then you have lots of options.
But for now, we're just going to create an EventBridge rule.
So let's create this rule, and the rule name is
going to be called InvokeLambdaEveryHour.
Now, if you click on schedule
you will see that there is a new capability
of EventBridge called EventBridge Scheduler.
And this is what we're going to use to
create this Lambda function scheduler.
So we're going to go into another console
and this is a new thing.

So it's called InvokeLambdaEveryHour.
And then it's a one time schedule
it's a recurring schedule, excuse me,
and then it's going to be a rate based
and we can say that we want every one hour to
have our Lambda function being invoked.
Okay?
Then we can specify more options about
the timing that we want to.
Flexible time we say of,
so that's we, happen to execute it exactly
every one hour.
Target API could be different targets.
In this case we're going to use AWS Lambda,
and then we need to select Lambda function.
So I'll just choose my demo Lambda, that's it
and then click on next.
And then whether or not we want to enable the schedule
and retry in case of errors,
we'll just leave everything as the default.
Then a role is created for me automatically.
I click on next, and then I create this schedule.
So by creating this schedule, we're having actually
something called EventBridge scheduler to run a schedule
and invoke our Lamda function.
This used to be something that was part
of EventBridge rules, but now has been separated.
Even though if you wanted to,
you could create a rule,
choose a schedule, and then instead
of clicking here continue in EventBridge schedule
you could just click on continue to create rule,
And there you go.
But it's not the recommended way.
So I'll just show you the better way,
with EventBridge scheduler.
So here with EventBridge scheduler
my schedule is now created,
and this is going to invoke my Lamda function
every one hour.
So it's quite nice.
But what if you wanted it to react
to events happening within your accounts in EventBridge?
So again, back into rules, we're going to create a rule.
So let's call this rule, send notification for login,
and we'll make sure this rule is triggered whenever
someone does a login into AWS.
So we'll use a rule with an event pattern and click on next
and then we'll use AWS events in it.
And then amongst the events of AWS,

we need to filter them.
So let's have a look at the event source.
So we scroll down, sorry,
and then we'll need to find a service.
And then we'll just type in console
to find the AWS Console Sign-in.
And then the event type will be Sign-in events.
So any user that signs in
through the console will trigger this events
pattern, which is amazing.
And then for the targets,
we can, for example send this into an SNS topic.
For example, my demo-ccp topic
And this will ensure that events are sent,
as SNS notifications whenever someone logs in.
So next, next, and then create rule and we're good to go.
So now anyone, someone logs into your account
you'll receive an SNS notification and email.
One last I can do,
for example is EC2InstanceTerminateNotifications.
So say you wanted to be notified anytime,
and this two instance was terminated for whatever reason.
So again, we'll use a rule
with an event pattern,
and then we'll use every services
and then we'll use the EC2 service,
And then we'll have a look
at the EC2 instance State-change notification.
And then what state do we want to change to?
So here we're can apply a filter and say,
Hey, I want to apply to specific states,
the state being terminated.
So whenever an instance goes to a terminated states
then this rule will apply
and here will apply to any instance.
So you can see EventBridge is very powerful,
and the event patterns can be quite complicated.
So next, what do we want to invoke?
For example, we can invoke, again, SNS topic
and send a message into the demo CCP topic.
So we have one more rule, and now you start to get an idea
around the power of EventBridge.
So we created three rules.
Now before we heads up MDs, hands on
you need to disable them, for example
or you need to just delete them, whatever you prefer.
And lastly, as you can see
there can be different event bus,
as I told you, I'm not going to go
into it because it's two events for this CCP level.

But you can send your own events into EventBridge
or you can have partners
of AWS send events into EventBridge as well
for you to react to them in real time.
So that's it for this lecture,
I hope you liked it and I will see you in the next lecture.

So now let's talk about AWS CloudTrail.
So CloudTrail is a service that provides governance,
compliance and audit for your AWS accounts.
And whenever you use an account
it's going to be enabled by default
because CloudTrail will get an history of all the API calls
or events that happen within your accounts.
And this is very important because you've
if someone, for example, logs in the console
then whatever they do will be logged in CloudTrail.
If someone uses the SDK, it will be logged in CloudTrail.
If someone does a command with the commanded line interface
it will again be logged
with CloudTrail
as well as any service activity
as well will be logged in CloudTrail.
So that means
that anything that happens will be put in CloudTrail.
And then for you,
for audit and security purposes
you can take the logs of all the history
of events and API calls made within CloudTrail
and send them to two locations,
either CloudWatch Logs or Amazon S3.
Now, when you create a trail in CloudTrail,
you can actually apply it to all the regions
to monitor what's happening in all regions.
And then the trail can go into CloudWatch Logs or Amazon S3
or just trail it down to a single region.
So the example that's queued, hey for example,
a user has deleted something.
How would we know what has been deleted
and who deleted it and when?
Then the answer is going to be CloudTrail.
So anytime there is an API call that needs to be looked up
CloudTrail is going to be the right answer.
So to summarize.
From within the CloudTrail console
we can have information about usage of the SDK,
CLI and console,
as well as any IAM users and IAM roles
and all the API calls they make,
then the CloudTrail consult will display it.

But if you want long term retention of data
what you can do is that you can send them to CloudWatch Logs
or to your S3 bucket for longer term retention.
And from within CloudTrail
you can do any type of inspection and audit.
So that's it for this lecture.
I hope you liked it.
And I will see you in the next lecture.

So let's have a look at CloudTrail.
And CloudTrail is a service to intercept any API calls
or user activity within your accounts.
And so here on here on the left hand side panel,
we can have a look at the event history
and this is the event history
for the last 90 days of management events.
So you can see all the API calls that are being made
over time in this account.
So it doesn't have to be very interesting, okay,
but all of them will be here.
So what I wanna do for example, is that I want
to look in my EC2 console, and I created a demo instance.
And what I'm going to do is that I'm going
to terminate this instance.
So I do right click, terminate,
and now the instance is being terminated.
And what I'm going to do is I'm going to check whether
or not this event happens and appears within CloudTrail.
So I'm going to wait about five minutes and get back to you.
And so I just refreshed my pages, and as you can see,
I ran the terminate instances, API call.
And we can see what's the event source.
So it's EC2 from where it was done,
the access key that was used,
the region that was used, and so on.
And we can get the whole event right here.
So that's the full power of CloudTrail is
that we can see all the events really happening
from within CloudTrail directly in this UI.
And this is a short introduction at the practitioner level,
but this is enough for you to get started
and to answer questions at the exam.
So that's it.
I hope you liked it.
And I will see you in the next lecture.

Now, let's talk about another service
that's called AWS X-Ray.
So, when by default some people do debugging in productions,
so, when your application is actually deployed,

the good old way, you would test locally,
then you would add log statements anywhere.
Maybe look into how it logs,
then you would re-deploy in production,
and the see if you can find the problem this way.
And the problem is that we have logs
from different services and different applications.
Doing log analysis it is very hard
because we have to combine everything.
So, if you have one application
that's called a big monolith, so, one giant application,
it's sort of easy to do debugging.
But if you have distributed services
they're connected through SQS queues,
SNS topics, they're decoupled and so on,
it becomes really, really hard to trace
and see what is happening within your system.
So, you have no common view of your entire architecture.
But to solve that problem,
you can use AWS X-Ray.
So, with X-Ray, you're going to be able
to do a tracing and get visual analysis of your application.
So, X-Ray, once you enable it on your services,
then you'll get a full picture
of what is happening for each service.
And see where they're failing, their performance,
and in case one request goes wrong,
you will be able to visualize it
directly into the X-Ray console.
So, the X-Ray advantages,
is to do troubleshooting of the performance
through the bottlenecks,
or to understand the dependencies
in a microservice architecture,
because they're all connected
as you saw in the previous graph.
We can pinpoint a service issue with tracing.
We can review a specific request behavior,
and find the errors and exceptions for that request.
We can know if we're meeting or not
our service-level agreement, or SLA,
meaning are we replying on time for all the requests.
And if we're being throttled,
if we're being slowed down,
where is it happening, in which service?
And finally, what or which users
are going to be impacted by these outages.
So, X-Ray really is great when you see,
distributed tracing, troubleshooting,
and you want to have a service graph.

That's it, it's a more complicated service to use,
so, I will not do any hands-on.
But you get a high level of review
that will be enough for the exam.
So, hope you liked this lecture,
and I will see you in the next lecture.

Now let's talk about Amazon CodeGuru.
And this is a machine learning-powered service
that will do two things.
Number one is automated code reviews,
and number two is application performance recommendations.
So when developers push our code,
there is usually another developer that does a code review.
And then when the code is deployed into production,
you need to be able to monitor the performance of your code,
and maybe you'll detect bugs by looking at the performance.
So CodeGuru does that in an automated fashion.
So CodeGuru Reviewer is here to do automated code reviews
with static code analysis.
So that means when you deploy your code onto a repository,
for example, CodeCommit or GitHub,
then the CodeGuru can have a look at all the lines of code
and then can give you actionable recommendations
in case it detects a bug or a memory leak
or something that it has seen before.
So, because of this machine learning capability,
it can detect bugs before even other reviewers detect them,
which is very helpful.
And then CodeGuru Profiler is going to be here
to give you visibility or recommendations
about your application performance
during runtimes or in production.
So when you build and test your application,
CodeGuru Profiler is already going to detect
and optimize the expensive lines of code pre-production.
And then when you deploy your application,
you're going to measure your application in real time,
and CodeGuru Profiler, yet again,
is going to identify performance
and cost improvements in production
and give you these recommendations directly in your code.
So this is the whole power CodeGuru.
So if we do a deep dive,
CodeGuru Reviewer really looks at your commits,

so whenever you push your code,
and tells you the lines of code that are probably wrong,
so it could be very, very handy,
so you can identify critical issues,

security vulnerabilities, and hard-to-find bugs.
So, for example, you can implement coding best practices,
you can find resource leaks, do security detections
in case you're creating a security hole,
or input validation.
The way it does it is using machine learning
and automated reasoning.
And how does it do it?
Well, there are code reviews that were analyzed by CodeGuru
across thousands of open source repositories out there,
and also on all the amazon.com repositories,
so this is how it learned to be a code reviewer
through machine learning.
It supports currently Java and Python,
and it has integration with GitHub,
Bitbucket, and CodeCommit.
Now, don't quote me on this.
Maybe things will evolve over time.
If it does evolve, don't worry:
you don't need to know that level of detail.
Just know about CodeGuru, CodeGuru Reviewer,
and then, lastly, CodeGuru Profiler.
So Profiler is when your application is in production
or in pre-prod,
and it helps understand the runtime behavior
of your application,
and to look at what consumes excessive CPU capacity,
for example, on the logging routine.
So it will allow you to identify
and remove code inefficiencies,
improve the application performance,
for example, reduce the CPU utilization,
decrease compute costs, provide heap summaries
so to identify which objects
are taking a lot of space in memory,
anomaly detection, as well,
in case your application behaves weirdly.
It also supports applications
that will be running on AWS Cloud or even on-premises.
And there's going to be
a minimal overhead on the application
to be monitored using CodeGuru Profiler.
So that's it for this lecture.
Just, again, remember the high level of CodeGuru,
CodeGuru Reviewer, and Profiler,
and you should be good to go.
And I will see you in the next lecture.

So now, let's talk about
the AWS Health Dashboard.

So there are two parts.
There is a Service History and then your accounts.
So the Service History will show you all the regions
and all the services' health.
It looks like this.
So you can follow, based on the region you're in,
how a service behaved, if there was any issue with it.
And you can look at history.
So this is for each day
and there is an RSS feed you can subscribe to.
It used to be previously called
the AWS Service Health Dashboard.
So this is just general information.
The next one we have is the AWS Health Dashboard,
which is for your accounts.
So it used to be called
the AWS Personal Health Dashboard, PHD,
but now just Health Dashboard for Your Accounts,
and it'll provide you alerts and remediation guidance
when AWS is going to be experiencing events
that impact you directly.
So while the Service Health Dashboard
will display a general status of all your services,
the Account Health Dashboard will give you a view
for the performance and the availability
for the services that you're actually using
in your accounts and your resources.
It gives you relevant and timely information
and also gives you notifications
to proactively look out
for scheduled maintenance activities.
On top of it, within this Health Dashboard for Your Accounts
you can aggregate data for your entire AWS organization.
So to get access to your health dashboard,
just click on the top right corner next to the bell
and you'll have access to it.
So it's a global service and it will show directly
the outages that directly impact you.
You will get the event log to see past events.
And you can see, for example, on this one,
that there was an EC2 issue in US East 2
that could have impacted me.
So you'll get alerts, remediation information,
proactive notification
in case there is a scheduled change,
as well as scheduled activities.
Alright, that's it.
I hope you liked it and I will see you in the next lecture.

So here I am in my console.
And to get to the Health dashboard,
you can just click on this bell right here
and click on Event Log.
This will take you straight into your Health dashboards.
So first things first, let's look at Service Health.
So if you click on Service History,
this is going to give you the Service Health
of all the services in AWS by region, by day.
So you can see, for example, for North America,
we can look at Amazon EventBridge Scheduler
and see that today it was fine,
and yesterday was fine, and so on.
And you can scroll down and look at all the services,
and you can see that AWS is quite stable.
So you can find a specific service
or a specific region you want to drill into.
On top of it, in case you have open issues,
you will see them right here,
confirming the fact that maybe you are experiencing an issue
with a service and so are other customers.
But this is just general information.
For your Account Health,
for the issues that are relevant to you,
you will find them in here in the Open and Recent Issues.
So when you have one,
you'll see the bell will have a little icon.
Right now there's none,
because there is no open issues,
no schedule changes, and no other notifications.
But in case you have one, you'll see a bell here.
And this is going to give you the issues
that are impacting you right now.
For this tab, and for this tab, the Schedule Changes,
for example, they're gonna do
some maintenance on EBS volumes,
and that will impact you or other notifications.
And if you go into the Event Log,
you will find actually issues
that were opened and closed, and where,
the start time, and the last update time.
And these issues are received only applied to me,
because I was using underlying services.
So if I look at this Operational Issues for EC2,
you can click on it, you can view when it got started,
the fact it got resolved,
the description, and how it impacted me.
And on top of it, you could click on Affected Resources
to see the resources that were affected by this issue.

So, hopefully, you see the difference
between the Service Health and the Account Health.
And, finally, we have the Organization Health,
where you can configure it
to get organization-wide visibility
into the health of all your accounts.
And if you wanted to have automations,
you can integrate health
with Amazon EventBridge, for example.
Okay, so that's it for this lecture, I hope you liked it.
And I will see you in the next lecture.

So let's summarize everything we've learned
in the monitoring section.
The first ones is around CloudWatch
and CloudWatch has multiple flavors.
There is CloudWatch metrics to monitor the performance
of your AWS services and billing metrics.
CloudWatch alarms
if you want to automate notifications when
a metric goes outside of a specific range.
And then you can automate it to perform EC2 actions
such as reboots, et cetera.
You can also send notifications that are clean
to the SNS service based on a metric
going over certain limits.
CloudWatch Logs are used to collect log files
from EC2 instances, servers and lender functions
and they're centralized within one service.
And CloudWatch events also called Event Bridge is a way
for you to react to events in AWS
or to trigger a rule based on the specific schedule.
Now, if you want to audit API calls
made within your account,
you should use the CloudTrail service,
and on top of it there is CloudTrail Insights
which is for you to get an automated analysis
of your CloudTrail events.
Amazon x-Ray is used to trace requests made
through your distributed applications
and this is very helpful when you want to do
performance analysis or root cause analysis
especially when you have errors
and all your applications are talking with one another.
The AWS Health Dashboard gives you the status
of all the AWS services across all regions
whereas the AWS Account Health Dashboard is talking
about the AWS events that only impact
your specific infrastructure.
Finally, we have CodeGuru,

CodeGuru is a way for you to perform automated
code reviews using machine learning
and also application performance recommendations, again,
by monitoring the performance
of your application in production
and applying yet against some machine learning.
So that's it,
I hope you liked it and I will see you in the next lecture.

#_____section 15 _____

Welcome to the section on VPC.
So VPC stands for virtual private cloud and this is going
to be just a small crash course.
VPC is quite complicated to be honest and this is something
you should know in-depth if you are going for the AWS
Certified Solutions Architect Associate certification
or the AWS Certified SysOps Administrator
Associate certification.
At the AWS Certified Cloud Practitioner level
there need to know about a few about a few concepts
but at a high level and what they're used for.
So this includes VPC, Subnets, Internet Gateways,
and NAT Gateways, Security Groups, Network ALC or NACL,
and VPC Flow Logs, VPC Peering and VPC Endpoints,
Site to Site VPN and Direct Connect,
and finally Transit Gateway.
I'm going to give you an overview of all these things.
This represents less than one or two questions
at your CCP exam.
And then we'll have a look at the default VPC
created by default by AWS for you.
This is going to be our hands-on.
We're not going to create our VPC from scratch.
At the end of this section there is a summary lecture
to explain what each of these services lined up above do
and what they are.
And it's okay if you don't understand it all.
Hopefully that summary lecture should do it for you.
And don't worry, the exam is quite high-level on VPC
so if you pay attention to this lecture
and this section closely, you should be good to go.
Okay, that's it.
I hope you're excited and I will see you
in the next lecture.

Before we learn about VPCs in AWS,
let's learn about IP addresses in AWS.

So you have the IPv4 protocol.
That's the one you're most used to, I guess,
which is 4.3 billion addresses.
And the public IPv4s are IP addresses
that can be used on the internet
and that allows whatever is the IP
to be publicly reachable from anywhere.
So if you create an EC2 instance,
we've seen that it gets a public IPv4
And what's going to happen is that if you stop the instance,
the IPv4 is going to be released.
And if you start it again,
then it's going to get a new public IP address at launch.
So this is a behavior we've seen in this course.
Now we've also seen the private IPv4.
So this time this is an IPv4 in the form,
for example of 192.168.1.1.
And this is an IP
that can only be used on a private network
such as your internal AWS VPC.
And so these are not publicly accessible.
So if you try this on your web browser,
you can only access IPs within your own network.
So the private IPv4 is going to be the same
for your entire EC2 instance lifetime,
even if you stop and restart the instance.
Now, there is an elastic IP in AWS.
It's a way for you to get a fixed public IPv4 address
into an EC2 instance.
That means that if you stop the instance
and start the instance again,
it will have the same public IPv4,
which can be desirable.
But of course, if you leave
the instance stopped for a long time,
then this elastic IP is used for nothing.
Now in terms of pricing,
every single public IPv4 on AWS
is going to be charged at $0.005 per hour.
This is including the elastic IP
and the normal public IPv4,
and to allow you to start using AWS at no cost,
in the free tier,
there is 750 hours of usage per month
for public IPs.
So that means that AWS is trying to get you
to use something else than public IPv4,
which is IPv6.

And so IPv6 is the newer internet protocol version,
and you have a lot more addresses.
So 3.4 times 10 to the power 38 addresses.
So that means you have 38 zeros in there.
So it's a lot more addresses.
And every IP address in this range in IPv6
in AWS is going to be public.
There is no private range for IPv6.
And so this is what an IPv6 IP address looks like.
And they are free in AWS.
So that means that if you want to
expose some services for free
from an IP standpoint on the internet,
you must use IPv6.
So that's it for IP addresses in AWS.
I hope you liked it.
And I will see you in the next lecture.

(Instructor) So let's start by understanding
what a VPC is.
So VPC is a Virtual Private Cloud.
And is a private network for you
to deploy your resources in, for example,
your EC2 instances,
a VPC is linked to a specific region.
So if you have multiple regions in AWS
and you have multiple VPC.
So here's an example.
Here's my VPC that I've created.
And within the VPC,
we can have subnets.
And a subnet is going to be a part of the VPC.
So partition of your network.
And that is going to be associated
with an availability zone.
So here's an example we have an AZ A.
And within an AZ we're going to have First of all,
a public subnet.
So this public subnet is special because it is a subnet
that is accessible from the internet.
As you can see on the diagram on the right hand side,
the public subnet has direct connectivity to the Internet,
and the internet can directly reach our public subnets.
Also, we can have a private subnet
and a private subnet is a subnet that is not accessible
from the internet.
We'll see how we can define a public subnet
and a private subnet later on.

So to define access to the internet
and access between the subnets so
that the resources can communicate,
we need to use Route Tables.
So the question is,
what do you put in a public subnet?
Well, when we did create our EC2 instances,
we created them in a public subnet.
But in a public subnet,
you can also put,
for example, a load balancer.
And in a private subnet,
which we don't have,
when we have a default VPC,
you could place your databases,
for example,
because they don't need access to the internet
and therefore they're going to be more secure.
So if we look at a VPC,
a more complete one,
we have the cloud of AWS, we have regions
within the region will have a VPC.
And the VPC will have what's called a CIDR Range,
which is a range of IP addresses
that is allowed within your VPC.
And then the VPC can go across two
or three availability zones.
So we have AZ1 that contains a public subnet,
and the private subnet and AZ2
which contains also a public subnet and a private subnet.
So in this example we have two AZ, one VPC, four subnets.
Two of them are public and two of them are private.
And we can launch EC2 instances.
In each of these subnets.
Finally, how do we define access to
the internet for these subnets?
Well, if we look at the same example as before,
say we had an EC2 instance in a public subnet,
for it to be able to access the internet,
we need to create what's called an Internet Gateway.
And this Internet Gateway will help our VPC instances
to connect directly to the internet.
So the VPC will be having an Internet Gateway.
And then the public subnet will have a route
to the Internet Gateway to be able to access the internet.
So fairly simple,
as soon as we have a Internet Gateway
and a route to the Internet Gateway that makes
the subnet a public subnet.
Now if you have an instance in a private subnet,

it is not going to be accessible from the internet.
But you may want to give it access to the internet
For example, to get updates for your operating system
or to download files.
So for this,
we can create what's called a NAT Gateway,
which is managed by AWS,
or in that instance,
which is self managed.
And that will allow your instances in your private subnets
to access the internet while remaining private.
So concretely, we create a NAT Gateway
or Nat instance in our public subnet.
And we create a route
from the private subnets to the NAT Gateway,
and from the NAT Gateway to the Internet Gateway.
And this will allow your private subnets
to get internet connectivity.
So that's it for the theory,
now I will just show you in the console what's
the default VPC and what is created for us.
So let's go in the Console of the VPC.
So I'll type in VPC.
And as was mentioned,
when we did create our account,
there was a default VPC created for us.
So this is what we observe
from here we have one VPC, three subnets,
one Route Table and one Internet Gateway.
So first, let's have a look at the VPC.
So if we look at the VPC,
we can see that this VPC is available
and we get the CIDR which corresponds to the IP
range of that VPC.
So if you want to know what this means,
you go to CIDR.XYZ,
which is a website.
And on this website you're going to enter
the CIDR you see here so 172
dot and then we have 31 00 16
so 31, zero, zero
and then 16
and this shows you the kind
of addresses range we're going to get.
So the first IP
is 172 point 31.01
and the last IP is 172 point 31
point 255 point 254.
And we get about 65,000 IP in this entire range.
So this is why when we've been creating EC2 instances

in our default VPC,
the private IP where within the range shown below,
okay, great. So we have our VPC.
And so far, it's good.
We have one CIDR block,
we could add more if you wanted to.
Next we want to look at the subnets.
So for this,
we go to subnets.
And we open this as a new tab.
And we can see that three subnets are created
for us in our VPC.
So each subnet correspond to a specific availability zone.
So for example,
this subnet right here is for eu-West-1a,
this subnet right here is for eu-West-1b.
And this right here for eu-West-1c.
And within each subnet,
we get a different IPv4 CIDR.
So if we look at this one
for example, for eu-West-1a, it is the same as before,
but slash 20.
So if we go to this website and do slash 20,
we can see what is the first IP and what is the last IP.
And then again,
this is a subset of everything we had from before.
So if we go back into our VPC,
our subnets excuse me,
and then look at this CIDR right here.
So this is a new CIDR's, point 16.0.
So copy this and paste it and shows you again,
the first IP and the last IP.
So we can have multiple subnets their partitions
within our VPC and each subnet in this example
Has 4091 available IPv4.
So if we launch an EC2 instance,
within a subnet,
an ipv4 will be used.
Okay. So we have three subnets,
corresponding to three AZ.
And so when we went into the EC2 Console,
I will show you this right now,
when we went to the EC2 Console,
and we launched an instance,
we're going to do it again,
launch an instance,
we had to choose the AMI type,
we had to choose the instance size
and then we had to choose the instance details.
The network was associated to the default VPC.

And for the subnets,
you have to choose the CIDR you want it to be in.
And these subnets correspond to AZ
and they correspond to the subnets
that we created right here.
So as an example,
if I create an instance in eu-West-1a,
so I'm going to create that instance,
do review and then launch
and then acknowledges our instance
is getting created automatically.
And if we look at
the private IP 172 point 31.6 point 131
this IP, is within the CIDR Range.
That has been defined, right here.
Okay, this is good.
So we understand better how it works.
And because this is a public subnet,
then this is why we were able to connect
to our EC2 instance,
and this is why our EC2 instance was able
to be used as a web server
and also, install different packages.
Okay, so this is good.
We understand better now
what goes on behind the scenes.
So next, what I told you about was
the Internet Gateway.
So the Internet Gateway here has been created already.
And it's attached to one VPC
so VPC can only have one Internet Gateway.
And this is what allows my EC2 instances in here
to get internet access.
If I do remove the Internet Gateway,
if I detach it,
I won't do it,
then I'm not going to be able
to access my EC2 instance anymore.
And for the Internet Gateway to be used,
it needs to be used by a Route Table.
So if I go back to my subnetS
and click on any of these subnets and go to the Route Table,
I can see there is a Route Table associated with it.
If I click on this Route Table,
this road table contains a Route in here
That says that if you're going within this CIDR,
you stay within your local VPC.
But if you're going anywhere else,
then please use the Internet Gateway.
This is a link to the Internet Gateway.

Well, this means that our instance has a Route
and within our subnets to the Internet Gateway,
and therefore it is a public subnet
and the EC2 instance can access the internet.
As I mentioned,
we don't have any private subnet in this example,
because it is a default VPC.
But if you wanted to as an exercise,
you could create a subnets.
It's more complicated,
I do this in my other courses.
And then you will need to also define
a specific Route Table.
Okay, finally, we are not using them
because we don't have any private subnets.
But you could create a NAT Gateway,
and the NAT Gateway would be associated with a subnet
and would allow your instances in your private subnets
to access the internet.
And this is more advanced.
You just need to know what it is at a high level.
Finally to clean up but do not forget
to take your instance right click
and terminate it to leave it clean after yourself.
That's it, I will see you in the next lecture.

Now let's talk about network security
within our VPC.
So we have Network ACL and security groups.
So if we look at our VPC and look at a public subnet,
we have an EC2 instance.
And the first line of defense for our EC2 instance
is a NACL or a Network ACL, which is a firewall
that is controlling traffic from and to the subnet.
So remember it is at the subnet level.
In which we can define allow or deny rules,
and then attach at the subnet level.
And the rules can only include IP addresses.
So if we look at this diagram, as we can see,
the NACL is going to be filtering traffic in and out
of the subnet before it reaches our EC2 instance.
The second line of defense we have,
and we've been already using it so far in this course,
are security groups.
And this is a firewall that controls traffic
to and from an EC2 instance.
And the security groups can only have allow rules.
And finally they can reference either IP addresses
and other security groups.
So in this example, we have the security group

around our EC2 instance, and that controls the traffic
going in and out of our EC2 instance.
So the security group and the NACL are quite different.
The NACL is at the subnet level,
and the security group is at the EC2 instance level.
And from an exam perspective,
that should be all you need to know.
Now there is a more complete table around the differences
of the security group and the Network ACL.
Let's look at the most important ones,
and these are the first three.
While a security group operates at the instance level,
whereas a Network ACL operates at the subnet level,
a security group supports only allow rules,
whereas a Network ACL supports allow rules and deny rules.
And finally in the security group,
the return traffic is automatically
allowed regardless of any rules.
So that's called statefulness.
And in Network ACL, the return traffic must be explicitly
allowed by rules, which is called stateless.
So that's it, again, from an exam perspective,
I really think the first two
or the first three rules are enough.
So let's go into our default VPC to see what we have.
So in our default VPC, on the left hand side,
we can see for security, either security groups
or Network ACLs.
So if I click on Security groups,
we find the exact same security groups we have from
before when we were using the EC2 console.
So this menu right here and the menu in the EC2 console
are the exact same.
It depends on if you wanna see it from the perspective
of a VPC or if you want to see it from the perspective
of an EC2 instance.
So we have five security groups in here
and we can look obviously at the security group rules.
For example, for Launch-wizard-1,
which was created from before.
As we can see in this security group,
the inbound rules allowed HTP on port 80
and SSH on port 22 from anywhere.
For the outbound rules, it did allow all traffic
on all ports, all protocols, and anywhere effectively giving
our EC2 instance the right to talk to any websites.
Next, for Network ACLs that we haven't seen yet,
as we can see, one Network ACL has been created
and it is associated at the subnet level,
so it's associated with three subnets.

And then when we can look at it,
we can see that there are some inbound rules.
So this is saying that all traffic on all ports,
on all from everywhere is allowed effectively saying
that anything can go through it.
And if you look at the outbound rules, yet again,
it's allowing all traffic on all ports anywhere.
So this is called the default Network ACL
because it allows everything in and everything out.
But if we wanted to, we could create some rules,
create a rule number, for example, rule number 200,
and then you can define the type of traffic you want it
to allow or deny in and out of your subnets.
Again, remember, Network ACL is at the subnet level,
and we can see that the default Network ACL has been
associated with my three subnets,
which is what we would expect from a Network ACL.
So that's it.
I hope you liked it, and I will see you in the next lecture.

Now let's talk about VPC flow logs.
VPC flow logs are a log of all the IP traffic going
through your interfaces.
So you can get a VPC flow log, a subnet flow log
or even an elastic tech network interface flow log
to see the traffic going in
and out of your EC2 instances for example.
By enabling the flow log, we can monitor
and troubleshoots for connectivity issues.
For example, if a subnet cannot connect to the internet,
or a subnet cannot connect to another subnet,
or the internet cannot access a subnet,
this will be captured by the VPC flow log
and we can look at it to get to the root cause of it.
So this is super important because
on top of getting information out
of our EC2 instances,
we also get information
for elastic load balancers, ElastiCache,
RDS, Aurora, et cetera, et cetera.
And the VPC flow logs can go to S3, CloudWatch Logs,
and Kinesis Data Firehose.
Next we have VPC peering and VPC peering is to connect
to VPC privately using the network from AWS
and to make them behave
as if they were part from the same network.
So this is an example.
We have VPC A and VPC B
and we can peer them together and as soon as that's done
then they will have the same network

or behave as if they were in the same network.
So for this, you need to make sure
that the IP address's range do not overlap.
If they do overlap
then you cannot establish a VPC peering connection.
The other thing is that a VPC peering connection
is not transitive.
That means that if you add a new VPC, for example, VPC C,
and you create a peering connection between VPC A and VPC C
then that means that VPC B and C cannot talk
to each other yet.
If you want to have VPC B and C talk to one another
then you would need to create another peering connection
between your VPC B and C.
So let's go in the console to see how this works.
So let's explore VPC flow logs.
For this, you click
on your VPC and then click on flow logs.
And here you can create a flow log.
So we'll just have a look at the options for flow logs.
So you can name it,
you can say filters if you want all traffic,
just accepted traffic, or rejected traffic.
What is the maximum aggregation interval.
Do you want it to be
in 10 minutes interval or one minute interval?
And then what is the destination.
For example, you can send to CloudWatch Logs.
You can send to an Amazon S3 bucket, or you can send
to Kinesis Data Firehose in the same or different accounts.
And based on the option you choose
you have to specify different parameters.
For example, for CloudWatch Logs, the log group
as well as an IAM role.
And finally, here's the log record format
which is going to create some information
on the VPC flow logs such as version, account id,
interface id, source address, destination address,
source port, destination ports, protocol, packets,
bytes, start, end, action, log-status.
So that's it for the VC flow logs.
And regarding VPC peering connection, you will go
on the left hand side and you find peering connections.
So here we can create a peering connection.
So we can name it.
We have to select a local VPC to peer with
so that's called the requester VPC.
And then you have to select another VPC to peer with.
It could be in my accounts or it could be
in another accounts, and then it could be in this region

or it could be in another region.
For example, let's choose Cape Town Africa.
And then we need to enter the VPC id.
Once you've done that
you can just create a peering connection.
And then if accepted
then the two networks will behave as one
which is the whole point of using VPC peering connections.
So I don't have an extra VPC to show you right now
but you get the idea.
So that's it.
In this lecture
we've seen VPC flow logs to capture traffic information,
or VPC, as well as VPC peering connections.
I hope you liked it and I will see you in the next lecture.

So all the AWS services
we've been using so far, are public.
That means that when we connect to them,
we are connecting to them publicly.
But if we use a VPC endpoint,
we can connect to these services,
using a private AWS network
instead of using the public internet network.
Why?
Well, this will give you a better security
because you're not going over the public internet
and also, less latency
because you don't access the services through network hops.
So let's take an example,
we have a VPC, a private subnet
and an EC2 instance in that private subnets
and say, we want to connect to Amazon S3 or DynamoDB.
For this we create a VPC endpoint, of type gateway.
So you have to remember this.
The gateway endpoint is for Amazon S3 and DynamoDB only.
And using this EC2 instance,
you can connect through the gateway, into Amazon S3
and DynamoDB, but privately.
The other type of endpoints you have,
is a VPC endpoint interface
which is to connect to any other services on AWS.
For example, CloudWatch,
if you want it to push a custom metric
from your EC2 instance into the CloudWatch service.
So for this, we will have a VPC endpoint interface
and then the EC2 instance will connect to it,
to connect to CloudWatch.
So on the left hand side, I have to click on endpoints,
not endpoint services.

You have to click on the endpoints on the top
to create an endpoint and see what I see.
So we can create an endpoint for an AWS service,
and you have to choose the service.
For example, if I have any of these services,
is going to be of type interface.
So it says interface everywhere.
But if it's for DynamoDB, or if it is for Amazon S3,
which is at the bottom,
it's going to be and I need to go all the way to the bottom.
To see this, so I need you to go to the next page
and then find Amazon S3, here it is.
It is going to be a gateway.
So this is the only thing that you remember, okay?
VPC endpoints are for accessing your services privately.
For Amazon S3 and DynamoDB, it is a gateway.
And for all the other services,
it is going to be an interface.
I won't go and I won't set it up for you right now,
because this is more complicated
and out of scope for the exam.
You just need to remember
about the concept of a VPC endpoint.
Okay, so that's it
I hope you liked it
and I will see you in the next lecture.

So now let's talk about AWS PrivateLink,
also from the VPC Endpoint Services family.
So how does that work?
Well, say you have a service that you run within AWS,
or say there is a vendor on the Marketplace,
and they run a service on their own account
within their own VPC.
And they want to expose a service to customers of AWS.
So to thousands of VPC, they need to have private access
to that service, to establish a connectivity.
You could, for example, use VPC peering,
but that doesn't scale, and it's not very secure.
What you want is use something else,
and that's something else is a Private Link.
So a Private Link allows you
to connect a service running within your VPC
to other VPCs directly and privately.
So it does not require VPC peering or internet gateway,
because it's from the private network,
or NAT or route tables or anything like this.
So let's go through a diagram.
Say, for example, you are talking to a vendor
on the AWS Marketplace, and they run application service.

So they have their own service that you wanna use
in their own VPC.
And you wanna have access to it from your own VPC
in your own accounts with your own consumer applications.
In that case, you're going to ask your vendor
to do a Private Link.
On their end,
they will have to create a Network Load Balancer
to expose that service.
And on your end,
you will create an Elastic Network Interface,
and then you will establish a Private Link between the two
so that you have private access
to their Network Load Balancer
and therefore to their service.
And all the internet traffic is actually not gonna go
through the public internet,
but it's actually gonna go through your private network.
And so therefore all communications will remain private.
And so for every new customer
that that third party will need,
all they will have to do is to create a new Private Link
for their customers, which is very easy to manage
and way more scalable.
So that's it, I hope you like this lecture,
and I will see you in the next lecture.

Okay,
So now we are getting into the hybrid cloud.
So say you have an on-premises data center.
This is your own data center,
and you want to connect it to the cloud, to your VPC.
For this, you have two options.
Number one is to use a site-to-site VPN.
This is to connect your on-premises VPN to AWS.
And what is a VPN?
Well, it is a connection
between your on-premises data center
and VPC that is going to be encrypted,
and that goes over the public internet.
So this looks like this.
Your on-premises data center
will connect to your VPC through the public internet,
and then it will be encrypted.
So no one else can access to communication.
So this is very good because it can be set up very quickly.
In about five minutes,
you can have a connection between your data center in AWS.
But it goes over the public internet,
so you may have some limited bandwidth

and you may have some security concerns,
even though it is obviously encrypted.
The other option is to use direct connect or DX.
Direct connect, is to establish
a natural physical connection
between your on-premises data center and AWS.
And then the connection is going to be
private, secure and fast.
And it will go over the private network.
And that's going to be a lot more expensive
because you have to do a physical connection
between yourself and a direct connect partner into AWS.
And they will take at least a month to establish this.
But it's going to be more private
and obviously faster and more reliable.
So from an exam perspective,
they will ask you if it's a site-to-site VPN,
you should choose or a direct connect
to connect between your on-premises data center and AWS.
And it really depends on two factors.
Number one, is it going to be private or not?
And number two is,
does it need to be established fast or not.
And from these information's,
you should be able to select either the site-to-site VPN
or direct connect.
Now, just a little bit more details on site-to-site VPN,
so, it is to connect your corporate data center to your VPC.
For example, you see two instances running
in your private sub-net.
So for this, to establish a site-to-site VPN,
we need on-premises a customer gateway or CGW,
and that's something you have to remember at the exam.
So it's a customer gateway, or CGW and then
on the AWS site you will need virtual private gateway,
or VGW and once the two things are provisioned and created,
then you can connect them together using a site-to-site VPN.
And this is how site-to-site VPN is implemented
over the public internet.
So remember going into the exam,
customer gateway and virtual private gateway
are needed to establish a site-to-site VPN.
That's it for this lecture.
I hope you liked it.
and I will see you in the next lecture.

Now let's talk about the AWS client VPN.
So the idea is that you have your computer
and you want to privately connect into your AWS VPC.
Therefore you will leverage the client VPN

to establish a connection using the open VPN
to your private network in AWS or on premises.
I will show you how in the second.
Why would you want to do so?
Well, for example say you have deployed EC2 instances
in a private VPC and you want to access them
using a private IP.
Well, that's difficult if you don't have a VPN
but if you have a VPN, then it's super easy.
Once the VPN connection is established
you will be able to access your EC2 instances
using their private IP
just as if you were in the VPC network yourself.
So your VPC is right here
and then your client's VPN is installed on your computer.
You will establish the VPN connection over the internet.
So this goes over the public internet, of course.
And then you will be as if you are connected
privately into your VPC.
And if your VPC has established a site to site
VPN connection to your on-premises data center
then your computer will also be able to access,
privately, your servers on your, on premises data center.
And that's pretty magical, but it works.
I hope you liked it.
And I will see you in the next lecture.

Okay, so now we see
how we can connect our VPC together
through peering connections.
And we also saw about the site to site VPN
and Direct Connect.
And as you can see now,
if we have a serious infrastructure on AWS,
the network topology can become quite complicated.
So to solve this mess, there has been a new service created,
which is called the Transit Gateway.
So the Transit Gateway is to have a peering connection
between thousands of VPC and your on-premises system
with a hub-and-spoke star connection.
So this looks just like this.
You have the VPC Transit Gateway in the middle,
and all your Amazon VPC, your Direct Connect gateway,
as well as your VPN connections
are connected through the Transit Gateway.
So you don't need to peer the VPC with one another,
you don't need to have different connections and routes
between each of them and Direct Connect
and side to side VPN, all of this is done
within a single gateway,

that will provide this functionality.
So it works with Amazon VPCs,
it works with the VPN connections
and the Direct Connect gateways.
And that's it.
So when in the exam, you see a way to connect hundreds
or thousands of VPC together,
with as well your on-premise infrastructure,
think no more than the Transit Gateway.
I hope that was helpful,
and I will see you in the next lecture.

So we've learned a lot about the VPC
so let's just try to remember all of it.
So VPC stands for virtual private cloud,
and within a VPC you have subnets.
A subnet is tied to a specific availability zone
and it represents a network partition from within your VPC.
If you wanted to have internet access at the VPC level,
you will need to create an Internet Gateway.
But if your instances are private instances
and private subnets, then instead, you will need
to use a NAT gateway, which is managed
or NAT instances that you manage to give internet access
to your private subnets.
To have a firewall that's going to be stateless,
you need to have a NACL or network ACL,
which is going to represent subnet rules
for inbound and outbound traffic.
If you wanted to have stateful rules,
you can use security groups and they operate
at the EC2 instance level or the ENI level.
If you wanted to connect two VPC together
and they have non-overlapping IP ranges,
you can use VPC peering and it's a non-transitive thing.
So if VPC A is connected to B and then B to C,
it's not necessary and it won't happen.
A cannot communicate with C.
We have also learned about elastic IPs,
which are fixed public IPv4
that will have an ongoing cost if you're not using them.
Okay, next with VPC endpoints.
And that's to provide private access to any AWS service
within your VPC.
And if you wanted to establish a connection,
a private connection to a third-party service
in a third-party VPC, you could use PrivateLink.
If you wanted to have access to your network traffic logs,
you could use the VPC Flow Logs.
Now, in terms of private connectivity,

if you wanted to connect your on-premises data center
to AWS, you would have to set up a site-to-site VPN
over the public internet.
And then if you wanted to connect
your own personal computer directly
into your VPC, you could use client VPN
to establish an OpenVPN connection directly into your VPC.
Now, if you wanted to establish a very private connection
and direct within AWS and your data center,
you would use Direct Connect.
And if you want to connect everything together,
including thousands of VPC on-premises network,
site-to-site VPN, Direct Connect, and so on,
then you would use a Transit Gateway.
So hopefully all of this makes sense.
I know VPC has a lot of information
but the exam will ask you to choose the correct sub-service
within your VPC.
So I hope you liked it
and I will see you in the next lecture.

#_____section 16 _____#

Welcome to this section
on security and compliance.
We're going to start right away with the shared
responsibility model.
So this is something we've seen all along this course,
but now it is time for us to formally introduce it.
So, AWS responsibility is the security of the cloud.
That means that all the infrastructure they provide to you,
that includes the hardware, the software facilities,
networking.
They have to protect it
because this infrastructure will run
all the services that you're using on AWS.
On top of it, any service that is managed by AWS,
such as S3 DynamoDB, RDS, is the responsibility of AWS.
But once they provide a service to you,
then how you use that service is your responsibility.
So for example, as a customer,
you are responsible for the security in the cloud.
So in the instance of EC2 instance, your customer,
so you, is responsible for the management of all the
operating system that includes patching the operating system
and making updates to it.
You must configure the firewall,
so that means that's you Mister Green figure, for example,
the network ACL and the security group,

and also you need to make sure that your EC2 instance has
the correct IAM information through the use of,
IAM instance role.
Then, we also need to ensure that we
encrypt the application data,
according to our compliance requirements.
Then, there are some controls that are shared.
For example, patch management, configuration management,
awareness and training are both shared between you and AWS.
For example, for batch managements,
if using something like RDS,
then AWS will do the patch management for us.
And if we are using something like EC2 then we have to
patch our operating system,
so the shared control is here.
For example, for awareness and training,
AWS has to train their employees
to use their facilities correctly,
and to make sure they adhere to their security guidelines.
And you have to make sure to train your employees correctly,
to use the cloud,
and doing this training is one of these ways, obviously.
Next, let's look at a detailed technology, for example.
So for RDS,
the responsibility of AWS is to manage the underlying
EC2 instance and to the civil SSH access,
to automate the database patching,
to automate the operating system patching,
and to edit the underlying instance and disk
to guarantee that it functions over time.
Your responsibility as a user of RDS
is to check that the ports,
IP security groups inbound rules
in your database security group are set up correctly.
It's also to make sure that the in-database user creation
and the permission of these users is done the way you want,
and also you need to make sure that if you want to create a
database with or without public access.
And if you wanted to configure a database you could use
parameter groups, for example,
to force only encrypted connections.
Finally, if you wanted to encrypt the data
within the database, it is again,
your responsibility to enable.
Next, for Amazon S3,
The responsibility of AWS is to guarantee you
to get unlimited storage,
to guarantee you to get encryption when you enable it,

and to ensure the separation of data between all your,
all the different customers of AWS.
As well, they need to make sure
that all the employees of AWS can not access your data.
Your responsibility is to configure your bucket
the way you want to make sure
the bucket policy adheres to your standards,
and also to use IAM users and roles accordingly.
And finally, if you want to encrypt your data,
it is your responsibility to enable it and to use the
encryption scheme that works for you.
So, hopefully that makes sense.
Here is a diagram from the website of AWS,
which shows the responsibility in the cloud is a customer
and responsibility of the cloud is for AWS.
So as a customer, your data,
the applications, the platform,
identity, and access management is up to you,
operating system, network and firewall configuration,
as well.
Then your client's side data encryption,
your server-side encryption,
and your network traffic protection is all yours.
AWS instead, is responsible for their software,
so their services,
also making sure that the compute, storage, database,
and networking are working correctly when they
provide it to you,
and they're responsible for their hardware
and the global infrastructure.
So the regions, the AZ, and the edge locations.
Hopefully that makes sense because their shared
responsibility comes up at least two to three questions
in your exam.
So understanding what is your responsibility
and what is the responsibility of AWS is very important.
I hope that was helpful
and I will see you in the next lecture.

Now let's talk about
how we can protect ourselves from a DDoS attack.
A DDoS attack is a Distributed Denial-of-Service attack
on our infrastructure
and we'll see how this works.
So say there's an attacker,
that's a hacker
and they want to do a DDoS attack

against our application server.
In this case they're going to launch multiple master servers
and these servers are going to launch bots,
a lot of bots
and all these bots
are going to send a request to our application server.
Now our server is not meant
to handle this many requests
so it will be overwhelmed
and it will not be working anymore,
he will be denied a service
and therefore any normal user trying to connect
to our application server,
will see that our server is not accessible
or not responsive,
effectively making our application down.
So a DDoS attack is quite scary
when you think about it
but on AWS
you can protect yourself from it.
The first way is to use AWS Shield Standard
and that's enabled for all customers at no additional cost
and it will protect you against a DDoS attack
for your websites and application.
If you want a premium DDoS protection,
you have to use AWS Shield Advanced
which is going to give you 24/7
so 24 hours a day,
seven days a week protection on DDoS.
Then you have WAF
to filter specific requests based on rules,
this is the web application firewall.
CloudFront and Route 53 that we've already seen
to give us protection by using the global edge network
and so when it's combined with Shield
it will provide attack mitigation at the edge locations
and finally you need to be ready to scale
if you're under attack
maybe by leveraging auto scaling on AWS.
So here is what the sample reference architecture
looks like for DDoS protection.
So we have our users
and they will be routed through the DNS on Route 53
which is protected by shield
so your DNS is safe from DDoS attack,
then you should use a CloudFront distribution
to make sure your content is cached at the edge
and then it is also protected by shield
and in case you need to filter
and protect from an attack,

you can use the web application firewall,
then to serve that application
you can use a load balancer in the public subnet
that will scale for you
and finally behind the load balancer
you should use EC2 instances
in an auto scaling group
to be able to scale to the higher demand.
So all of this will give you a really good DDoS protection
against these type of attacks.
Now let's do a deep dive into the services
that I just mentioned.
So Shield is made up of two components,
we have Shield Standard
which is a free service
that is activated for every AWS customer
and this will provide you
protection against the common attacks for DDoS,
they're called SYN/UDP Reflection Floods,
Reflection attacks
and other layer three or layer four attacks.
Then you have Shield Advanced
which is an optional service,
it cost you about $3000 per month per organization
and they will give you protection
against more sophisticated attacks
on your EC2, ELB, CloudFront, Global Accelerator
and Route 53.
You also get access to your response team
when you need it
to help you protect yourself
during these DDoS attacks
and in case you are incurring some costs
on these attacks,
then any fees that is incurred during this attack
is on AWS.
So Shield from an exam perspective
remember that the free version
is activated by default for every customer
and if you need that response team
if you need to be having a higher level of defense
then Shield Advanced
is something that you enable yourself
and it cost about $3000 per month.
Next we have the Web Application Firewall, so WAF
and this is to protect your web applications
from common web exploits,
for example on layer seven.
Layer seven as you remember
maybe is HTTP

whereas layer four was for TCP.
So because it is layer seven
it can be deployed only on HTTP friendly devices
so it can be deployed on your application load balancer
your API gateway we haven't seen it,
it's out of scope for the exam
and CloudFronts.
On your web application firewall
you can define Web ACL
so Web Access Control Lists
and these rules on this ACL
can include filtering for example
based on the IP addresses,
the headers of HTTP,
the body, some strings
it can protect you against common attacks
such as a SQL injection
or Cross-Site Scripting.
You can have size constraints
to make sure the requests are not too big
and also block certain countries using a geo-match.
Finally for DDoS protection
you can use Rate-based rules
to count the occurrences of events,
therefore saying that
you know a user cannot do
more than five requests per second
and that would help to be protected against a DDoS attack.
So that's it's,
just at a high level remember
that it is a combination of
Shield, WAF, CloudFront, Route 53
they will give you an entire DDoS protection
and again all these service you need to them
at a high level.
So hope that was helpful
and I will see you in the next lecture.

So if you come across a question at the exam
that is asking you how to protect your VPC overall
then you can use the AWS Network firewall service.
So the idea is that the network firewall is going to
protect your entire VPC all at once
from layer three to layer seven protection in any direction.
So you can inspect any traffic
from any your VPC and out of your VPC
from outbound to the internet, from inbound to the internet
to and from your direct connect and site to site VPN.
So pretty much all traffic represented here such as
Peered VPC or Direct Connect

or internet can be protected from network firewall.
And this is a much better protection mechanism
than using, for example, network SCLs
which operate at the subject level.
This operates at the VPC level.
Okay, so that's it for this lecture.
I hope you liked it and I will see you in the next lecture.

So now let's talk about AWS Firewall Manager.
So it's a service that's very simple.
It allows you to manage all the security rules
in all the accounts of your AWS Organization
in a central place.
So from an exam perspective, you're going
to usually get the question on VPC Security Groups.
So if you see a question
around managing your VPC Security Groups
across multiple accounts in an organization,
think no more than the AWS Firewall Manager.
Now, I want to tell you a bit more about it.
So on top of managing VPC Security Groups,
you can also manage WAF rules, AWS Shield Advanced rules,
as well as AWS Network Firewall and their other services.
But from an exam perspective,
I think the most important one
is going to be the Security Groups.
Now, whenever you apply these rules in Firewall Manager,
they're going to be applied
to all the current and future accounts
and for all the resources
as they are being created in these accounts,
really allowing you to have the certainty
that the rules are managed
across all accounts at any point of time.
All right, so that's it.
I hope you liked it, and I will see you in the next lecture.
Okay, so now let's talk about
penetration testing on the Cloud.
So, penetration testing is when you're trying to attack
your own infrastructure to test your security.
A customer of AWS is welcome
to carry out these security assessments
and penetration testing against your own infrastructure
without prior approval for eight services.
So, our Amazon EC2 instances,
NAT Gateways and Elastic Load Balancers, Amazon RDS,
CloudFront, Aurora, the API Gateways, Lambda,
and Lambda Edge functions,
Lightsail resources and Elastic Beanstalk environments.

The list can increase over time,
but this is not something that you will be tested
on at the exam.
Just remember that you don't need an authorization
for these eight services,
but if you wanted to do other type of activities
that could be prohibited.
For example, you cannot do a DNS zone walking
via Amazon Route 53 Hosted Zone.
You can not perform a distributed attack on your system,
so you cannot perform a DoS or DDoS or a Simulated DoS
or Simulated DDoS.
You cannot just attack your own infrastructure
with a denial of service.
You can not do port flooding.
You can undo protocol flooding, request flooding,
which are, you know, variants of an attack.
And for any other events,
you need to contact the security team at AWS
to ensure that they can approve it.
If you wanted to read more, you can read more here.
So, from an exam perspective, yes,
you can do pen testing on your Cloud.
Remember that some are authorized,
but anything that looks like an attack such as DDoS attack
or a DNS zone walking or a port flooding is not authorized
because for AWS,
it would seem like you're trying to attack
their infrastructure and they wouldn't like it.
So, I hope that was helpful
and I will see you in the next lecture.

Okay, so now, let's talk about encryption.
And we've seen encryption a little bit in this course,
but now let's try to formalize what it means.
So there are two types of encryptions happening within AWS.
You have encryption at rest and encryption in transits.
So data at rest means that the data is stored
or archive on a physical device.
So that could be a hard drive or like a hard disk.
It could be an RDS instance because it's a database.
It could be an S3 Glacier deep archive,
or any kind of S3 buckets actually.
These kind of things, right? It's at rest
because it's not moving, it's written somewhere.
So for example, we can have encrypted data at rest
on an EFS network drive,
or we can have encrypted data at rest on Amazon S3.
But there's a second kind of encryption,
which is in transit.

In transit means that encryption while the data
is being moved from one place to another.
And so that means in motion.
And so for example, it says
whenever you transfer data from your on-premises
data centers to AWS
or from you, when you move from between an EC2 instance
and a DynamoDB table, for example, because it's writing data
or retrieving data, or in this example up there from EFS
to Amazon S3.
So in transit means the data is transferred on the network.
And so there are two kinds of encryption.
There's encryption at rest and encryption in transit.
They use different techniques, okay?
But ideally, because we want our data to be encrypted
and protected all the time, we want
to encrypt data in both states to protect it.
For this, we leverage something called encryption keys.
And I'm going to stop right there
because you don't need to know
how the encryption key works at the CCP level.
But just so you know that using these encryption keys,
we can encrypt the data.
That means that someone who does not have access
to these encryption keys, even if they had access
to our data, they could not decrypt it
and they could not read it, and therefore, it's protected.
So the encryption service at the center
of AWS is called KMS, Key Management Service.
So anytime you hear encryption for a service,
it's most likely going to be KMS.
And so with KMS, we don't have access to the keys.
AWS will manage the keys for us,
and we just define who can access these keys.
So there is an opt-in for encryption.
So for example, for your EBS volumes, you can choose
to encrypt them with KMS.
For S3 buckets, you also have the option
to do server side encryption of objects.
For redshift databases,
you can do the encryption of your database.
For RDS, the same thing.
For EFS, you can encrypt your files.
And there's also some services
that have encryption automatically happen no matter what.
For example, CloudTrail Logs, S3 Glacier,
and Storage Gateway.
The second service used
to perform encryption is called CloudHSM.
So with KMS, it is AWS

who manages the software for encryption.
But with CloudHSM, AWS will just provision
to us the encryption hardware,
but we are managing the keys ourselves.
So a dedicated hardware for us is called an HSM module,
so hardware security module,
and it looks like this.
And so with this aspect,
AWS will give us the hardware in their infrastructure,
but we manage our own encryption keys entirely, not AWS.
And to make sure that no one can to have our keys
by sneaking into the data center of AWS,
the HSM device is temper resistant.
I mean, if someone tries to temper it, it will fail.
And there is FIPS 140-2 level three compliance,
which is a security standard.
So CloudHSM, how does that work?
Well, AWS will manage the hardware.
So they will manage the device I just showed you,
but we use the CloudHSM service
with the CloudHSM clients to manage the keys.
And obviously, the connection between our clients
and CloudHSM is encrypted
so that we can manage the keys securely.
Okay, so what type of KMS keys do we have?
We have the customer managed key.
This is a key that is created, managed,
and used by the customer.
So you, and you can enable or disable a specific key.
You can also define a key rotation policy.
For example, you want a new key to be generated every year.
While of course, the old key is preserved,
you can also have the possibility to bring your own key.
Then you have the AWS managed keys.
So this time, it's created, managed,
and used on the customer's behalf by AWS.
So you're going to find it
whenever an AWS service has some encryption managed by AWS,
and the key name is AWS/s3,
or AWS/ebs and so on.
So whenever you see AWS/, it's an AWS managed key.
Then we have AWS owned keys.
So it's a collection of keys that a service owns
and manages to use in multiple accounts.
So AWS can use these keys
to protect some resources in your accounts,
but you actually don't have any power to view the keys.
And finally, you have the CloudHSM Keys
or custom keystore where the keys are generated
from your own CloudHSM hardware device,

and all the cryptographic operations
will be performed within the CloudHSM cluster.
So that's it for encryption.
I hope you liked it, and I will see you in the next lecture.

Now let's talk about
AWS certificate manager or ACM,
which is a service to easily provision, manage
and deploy SSL or TLS certificates.
What do you use the certificates for?
Well it's to provide in-fight encryptions
for your websites by providing an HTTPS endpoints.
So let's take an example.
We have our application load balancer
and it is connected in the backend through HTTP
to an auto scaling group with our EC2 instances.
But we want our end users to have HTTPS exposed
on our application of answer.
So to do so we're going to use ACM.
And ACM once it's connected to our domain
is going to be allowing us to provision
and maintain these TLS certificates.
They will be loaded onto our application load balancer
and then automatically the load balancer will be able to
offer HTTPS as an end point for our clients,
which allows us to get in-flight encryption
over the public web.
So ACM supports both public and private TLS certificates
and it is free of charge for public TLS certificates.
There's also a very nice feature
which is automatic TLS certificate renewal
which is very helpful, I use it all the time.
And it has integration, that means
that it loads the TLS certificates on different services.
It could be your elastic load balancer,
your CloudFront distributions
or your API is on API gateway, for example, okay?
So anytime you see what service can help us do
in-flight encryption and generates these certificates
then think ACM, that's it.
I will see you in the next lecture.

Okay, so now let's
talk about Secrets Manager.
So this is a newer kind of service and it is meant
for storing secrets, as the name indicates.
So Secrets Manager is a great way to store secrets,
but on top of it, you can force the rotation
of these secrets, I mean, they have to change

every X number of days.
For example, you can say, "Okay, every 90 days,
I want to change my passwords."
And so, you can do this in Secrets Manager.
You can automate the generation
of these secrets using Lambda.
And it has an integration with Amazon RDS.
So using the Secrets Manager, we can create
the passwords for Amazon RDS automatically.
The secrets are going to be encrypted using KMS,
that we just saw from before, automatically.
And so from an exam perspective,
anytime you see a secret to be managing in RDS
and to be rotated, you have to think about Secrets Manager.
It is a paid service,
but I'll just show you how it works very briefly.
So let's find the Secrets Manager UI.
So I'll go into Secrets Manager and from there,
I will be able to create and manage secrets.
So as you can see, they can be rotated, they can be managed,
and retrieved throughout their lifecycle.
I'm going to store a new secret
and if you see, before the pricing,
it says it's 40 cents per secret per month
and then you're going to pay for API call,
but you get a 30-day free trial available.
So we can store a new secret
and the secret can be a credential for a database,
for example, for RDS or for Redshift
or for another database or for another type of secret.
So this is where you can enter
whatever you want as a secret.
But, for example, as with for RDS database,
you need to specify the user name and the password,
as well as how the secret will be encrypted
and which RDS database will be linked to that secret.
So we can't do this right now
because we don't have an RDS database,
but if we wanted to store just a random kind of secret,
we could say password and then here
you could say MYSECRETPASSWORD.
And this will be the value of our secret.
And then you just choose, again,
how you would encrypt that secret.
Then you would click on next
and then you would give the secret a name,
for example, myproductionapplicationpassword,
which is a terrible name and then you click on next
and then when you're ready, you can configure the secret.
So, do you want it to have automatic rotation or not?

And yes, I want it.
I want it to be rotating every 30 days, for example,
and then choose the Lambda function
that will be rotating the password for me.
So you would need to create it in advance.
When you're done, you can review everything
and see that yes, we have created a secret,
it will have some secret value,
automatic rotation is enabled using a Lambda function
and then you can see some sample code
to retrieve the secrets in our applications.
I won't go with creating a secret, but you get the idea
and this is all you need to know about the Secrets Manager.
So that's it.
I will cancel this and I will see you in the next lecture.

So now let's talk about AWS Artifacts,
which is not really a service
but it is presented as one in the consult.
So what is it?
It is a portal that will give you the customer access
on demand to the compliance reports
and AWS agreements.
So these artifact reports can be downloaded
and they represent a security
and compliance documents from third party auditors,
such as the alias ISO certifications,
the payment card industry.
So PCI reports and the SOC reports.
There's also artifacts agreements
which is allowing you to review,
accept and track the status of alias arguments
such as the BAA agreements
or the Health Insurance Portability and Accountability Act,
which is HIPAA, we'll see this maybe in exempt,
for an individual accounts or in your organization.
And so these reports can be used
to support internal audits capabilities within your company
or complaints needs to show
that your compliance by using the AWS cloud.
So if you go into artifacts for AWS,
as we can see this is a global service
and we can get some of the artifacts.
We can view the reports or view the agreements.
So if I click on the reports,
I can see 61 reports right now that I can download.
For example, I can say,
I really want you to get this report right now.
I'm going to download it,
and I say, okay, I accept the NDA

and then I download this report and here I am,
I have a report that I can use for compliance internally
or also I can go into agreements
and find different agreements.
For example, three account agreements
and we don't have any organization agreements.
So I can take one of these agreements.
For example, this one, the BAA agreements
accept the agreements and then download it,
I scroll down, I accept it and then I download it.
So it's very, very simple.
It's not really a service.
It's a way for you to download compliance documents,
I just want to show you once
and then you have to remember it
and that will be it for this lecture.
I will see you in the next lecture.

So now let's talk about Amazon GuardDuty.
GuardDuty helps you do intelligent threat discovery
to protect your AWS accounts.
How does it do it?
Well, it has machine learning algorithm, performs anomaly
detection and uses third party data to find these threats.
So to enable it, it's just one click.
Then you have a 30 days trial.
You don't need to install any software.
So GuardDuty is going to look at a lot of input data,
such as your CloudTrail event logs
to look for unusual API calls
or unauthorized deployments.
It's going to look at your management events
and your data events.
So for example, on the management side,
the create VPC Subnet event and so on,
whereas on the S3 data events
for example, get object, list objects, delete objects
and so on.
And then for VPC flow logs, it's going to look
at unusual internet traffic.
It's going to look at unusual IP addresses.
DNS logs to look at EC2 instances, sending encoded data
within DNS queries, which would mean they're compromised
and optional features to allow you to analyze, for example,
other input data sources such as your EKS audit logs,
your RDS and Aurora login events, your EBS, your Lambda
and your S3 data events.
So we can also set up EventBridge rules to
be notified automatically in case you have findings.
And then these rules can target whatever EventBridge

can target, such as AWS, Lambda, or SNS topics.
Also, this can come up in the exam.
GuardDuty is a very good tool to protect you
against cryptocurrency attacks
because there is a dedicated finding for it.
So it knows how to analyze all these input data
and find that there is a cryptocurrency attack.
So to summarize, within GuardDuty,
we have several input data.
We have the VPC flow logs,
the CloudTrail logs, and the DNS logs
that will be, no matter what, into GuardDuty
as well as some optional features you can enable,
such as your S3 logs, your EBS volumes,
your Lambda network activity, RDS and Aurora login activity
and your EKS logs and runtime monitoring as well
as most likely, more features over time
that I will not put here
because you get the idea of optional features.
And so from all these things,
then GuardDuty can generate findings
and if these findings are detected,
an event is created in Amazon EventBridge.
And therefore from EventBridge, thanks to rules,
we can trigger automations, for example,
using Lambda functions
or send notifications, for example, using SNS.
So that's it for this lecture.
I hope you liked it and I will see you in the next lecture.

So now let's talk about Amazon Inspector.
So Amazon Inspector is a service that allows you
to run automated security assessments on a couple of things.
First of all, on the E2 instances.
So you are going to be leveraging the Systems Manager agent
on your EC2 instances and Amazon Inspector
is going to start to assess the security
of that E2 instance.
It's going to analyze against unintended network
accessibility and also analyze the running operating system
for known vulnerabilities.
This is done continuously.
Then we have also Amazon Inspector
for your Container Images push to Amazon ECR.
For example, your Docker images.
So as your Container Images are being pushed
to Amazon ECR, they will be analyzed
by Amazon Inspector against known vulnerabilities.
And we also have Amazon Inspector for Lambda functions.
So Lambda functions, when they're deployed,

will be analyzed again
by Inspector for software vulnerabilities
in the function code and the package dependencies.
And this assessment happens
as the functions are being deployed.
So once Amazon Inspector is done doing its job,
it can report its findings
into the AWS Security Hub and also send findings
and events of these findings into Amazon EventBridge.
This gives you one way to centrally
see the vulnerabilities running on your infrastructure
and with EventBridge you can run some kind of automations.
So what does Amazon Inspector evaluate?
You have to remember, the Inspector is only
for your running EC2 instances, your Container Images
on Amazon ECR and your Lambda functions.
And it's going to do a continuous scanning
of the infrastructure only when needed.
So it's going to look
at a database of vulnerabilities, so CVE,
for package vulnerability for EC2, ECR and Lambda.
And it's going to look
at network reachability on Amazon EC2
and in case the database of CVE gets updated,
then Amazon Inspector is going to automatically
run again to make sure
that all your infrastructure is tested one more time.
Every time it will run,
a risk score is going to be associated
with all the vulnerabilities for prioritization.
So that's it for Amazon Inspector.
I hope you liked it and I will see you in the next lecture.

Now let's talk about AWS Config.
So Config helps with auditing and recording the compliance
of your resources by recording the configuration
and their changes over time.
So any time we've been doing some manual changes
of the configuration in AWS, we did not have a list
of all the changes that happened,
but we can have this using Config.
Then this configuration data can be stored into Amazon S3,
to be later analyzed by Athena, or to be recovered.
The question that can be solved by using Config,
can be is there unrestricted SSH access
to my security groups?
Or do buckets have any public access?
Or has my ALB configuration changed over time?
All these things can be resolved by Config and Config rules.
Then you can receive alerts through SNS notifications

for any changes done onto your infrastructure,
and Config is a per-region service,
but you can create multiple Config configurations
and then aggregate all the results across all the accounts
and all the regions.
So let's have an example of how Config work.
So if you want to see the compliance of a resource
over time, this is for a security group,
you can see that it was noncompliant,
and then after making some changes,
it became compliant and green.
You can also view the configuration of a resource over time
to see how their configuration
of that security group has changed.
And finally, you can view who made these changes
to the resources based on CloudTrail
if you have enabled CloudTrail in your accounts.
So I am in the Config console, I'm going to get started,
and I need to choose the type of resources
that I want to record.
Just know that Config is not a free service,
so if you enable it, then you will have to pay.
For this, I will enable to record all the resources
in this region, including the global resources,
and it will create a bucket
in which all the configuration will be stored.
Then I can select a topic to send notifications
to for all the changes of the configurations.
I will disable this for now,
and it will create a Config service-linked role.
I'll click on next, then I can choose Config rules,
so these are rules to apply on my account.
For example, to check if SSH is open.
So the restricted-ssh is one rule,
you can have also public entry buckets.
So if you go to rds-instance-public-access-check,
for example this is an example, or if I type S3,
we can see s3-bucket-logging-enabled,
or s3-account-level-public-access-blocks.
So these are a bunch of rules in which they can be enabled
to check the compliance of our resources within our account.
For this example, and remember this is going to be paid,
I'm going to enable the restricted-ssh rule to show you
the compliance of my security groups.
I'll click on Next, and then as I can see,
I'm going to record the configuration of all the resources,
it's going to be put in the S3 bucket,
and it is going to apply this Config rule, restricted-ssh.
I'll click on Confirm,
and setting up Config can take a time,

so I will wait for Config to be done,
and to record everything in my account.
Okay, so I'm in Config, and I'm going to use the new console
to match your experience, and so in Config,
what I'm going to be able to see is the inventory
of all my resources in AWS.
As you can see, I have five security groups,
three subnet, one InternetGateway,
and so on you can read with me.
And it turns out, that as part of all my resources,
I have some resources that are not compliant.
So there's one rule that is not compliant,
and that is the restricted-ssh rule.
If I click on this, I can look at the rule detail,
and I can see the resources in scope,
and I can see that three out of five security groups
are not compliant.
So if I click on one of these security group,
I can look at the details of the security group,
and I can see the rule is applied,
and I can see that this rule is not compliant.
So this rule, what did it describe?
Well, it described the fact that there is unrestricted
SSH allowed onto my security group.
So what I can do, is that I can fix this rule.
So let's go into Resources and then I will find
my security group that is not compliant, so this one,
and we'll click on it,
and then I'm going to click on Resource Timeline.
So I can see my configuration timeline,
and I can see that my configuration was done
on the 2nd of June, and I can also look at
my compliance timeline, and I can see
that my security group right now, is not compliant.
So if I scroll down, I look at the rules,
and say, yes, this is not compliant,
and this is because we have a port opened
on SSH for everyone to use.
So what I can do is that I can fix this resource.
So for this I'm gonna go
into our security group configurations,
so this is this one I'm looking at,
and I'm gonna go into the EC2 consoles,
so I'll go into EC2, I will find the security group.
On the left-hand side I will go under security groups,
I will filter for the security group I'm looking for,
press Enter, here it is, and for the inbound rule
I can see that yes, SSH on port 22 is opened for everyone.
So what I'm going to do is that I will for example,
delete this rule temporarily, save the rule,

and then what I'll be waiting for is for my resource
to become compliant again.
So for this, I can either wait a little bit of time,
and this would show me the compliance,
or I can go a bit faster,
and then I can launch a rule and make sure it runs again.
So this rule that I created right here, I can do Action,
and Re-evaluate, and this is going to re-evaluate
all my noncompliant resources.
So let me pause a bit to wait for it to be done.
Now I'm going to refresh this page,
and what we are seeing now is that only two noncompliant
resources are done, so what I did, did fix one compliance.
If you go back to the Resource Timeline,
and I'm going to refresh this again,
let's see what happened.
We can see that now my resource is green,
and it's compliant, and if I click on Changes,
I can see that the fact that my Config rule
went from noncompliant to compliant.
If we want to look at the configuration changes
that are associated with this,
if I go to Configuration Timeline, I can see
that the resource has changed configuration on this.
So if I go to one change, I can click on it,
and I can see that this rule right here
that is described as-is, went to nothing,
which means it was deleted.
And if you look at CloudTrail,
we have the CloudTrail integration that said
that my root user did remove that security ingress rule.
And we can view that event in Details in CloudTrail.
So I hope that's helpful, I hope you now understand
the whole aspect of using Config to track
the resource configuration and their compliance over time,
and this is very helpful to ensure that all the resources
created by your employees within your company are compliant
with whatever you have decided for security rules.
So that's it, I hope that was helpful,
and I will see you in the next lecture.

Now let's talk about AWS Config.
So Config helps with auditing and recording the compliance
of your resources by recording the configuration
and their changes over time.
So any time we've been doing some manual changes
of the configuration in AWS, we did not have a list
of all the changes that happened,
but we can have this using Config.
Then this configuration data can be stored into Amazon S3,

to be later analyzed by Athena, or to be recovered.
The question that can be solved by using Config,
can be is there unrestricted SSH access
to my security groups?
Or do buckets have any public access?
Or has my ALB configuration changed over time?
All these things can be resolved by Config and Config rules.
Then you can receive alerts through SNS notifications
for any changes done onto your infrastructure,
and Config is a per-region service,
but you can create multiple Config configurations
and then aggregate all the results across all the accounts
and all the regions.
So let's have an example of how Config work.
So if you want to see the compliance of a resourc

Now let's talk about Macie.
Macie is a fully managed data security
and data privacy service that will use machine learning
and pattern matching to discover
and protect your sensitive data in AWS.
More specifically, it will alert you around sensitive data
such as personally identifiable information,
which is named PII.
So very simply, your PII data will be in your S3 buckets
and it will be analyzed by Macie
which will discover what data can be classified as PII.
And then will notify you
through EventBridge of the discoveries.
Then you can have integrations into an SNS topic,
Lambda functions and so on.
So Macie in this instance will be used
to find the sensitive data in your S3 buckets
and that's the only thing it will do.
It's just one click to enable it.
You just specify the S3 buckets you want to have
and that will be it.
So that's it for this lecture, very, very short,
but that's enough on Macie.
I hope you liked it, and I will see you in the next lecture.

So now let's talk about the AWS Security Hub.
So it's a central security tool
that is used to manage the security
across several AWS accounts and automate security checks.
So this has an integrated dashboard
that's going to show you the current security

and compliance status
that allows you to quickly take actions.
So it's going to aggregate alerts across different services
and different partner tools.
So you're going to get services such as Config,
GuardDuty, Inspector, Macie, IAM Access Analyzer,
AWS Systems Manager, AWS Firewall Manager,
AWS Health, and AWS Partner Solutions,
and maybe even more,
I may not change this lecture over time,
but to give you an idea,
it just aggregates all these partners tools
and all these services into one central dashboard,
one central hub.
And so for Security Hub to work, first of all,
you must enable the AWS Config service.
So if we look at a diagram, just to summarize,
Security Hub can cover multiple accounts at a time
and gather findings from Macie, GuardDuty, Inspector,
Firewall Manager, IAM Access Analyzer,
Systems Manager, Config, Health,
and is going to find all the potential issues and findings.
And thanks to automatic checks,
you're going to see in your dashboard,
your Security Hub findings.
But also anytime there's a security issue
this is going to generate an event in EventBridge.
And finally, to investigate the source of these issues
you can use Amazon Detective
which is going to allow you to very quickly know
where the security issues are coming from.
And so in AWS we can see the Security Hub.
So as we can see, there's a pricing per check.
So the first 1000 checks cost you this much
and then on so on.
So the more checks you look
the more pricing you're going to have.
Then there is ingestion events.
So the first 10,000 events are free
but then you have to pay per finding, okay?
Now you get 30 day trial for Security Hub.
So I'm not going to enable it
but I just wanna show you the options.
And it really shows you here
that we have these automated security checks
and the fact that it looks at different services
such as GuardDuty, Inspector, Macie and others
to find these security issues

and then allow you to take actions, okay?
So if we go to Security Hub and enable the trial
we'll see that first we need to enable Config
to make sure that Security Hub can work.
And then we need to choose the security standards
we want to adhere to so we can have three different options.
And finally, the integrations
based on the services that we have enabled.
And then you click on Enable Security Hub
and you'll be good to go.
But I'm not going to show you anything.
I'm not gonna have security issues in my accounts
because I don't really use it,
but at least I showed you the option
and you can see the kind of service that it is.
Okay, I hope you liked it
and I will see you in the next lecture.

Okay, so very short lecture
on Amazon Detective.
So, when you have all these services
such as GuardDuty, Macie, and Security Hub
that are used to identify potential security issues
or findings, you need to find how these happen
and get to the root cause.
And so, sometimes the deeper analysis to isolate
the root cause can be long and complicated.
It's a complex process to analyze data
from different places and link it together.
It could be quite long
and when you're dealing with security,
you want to get to the root cause as quickly as possible
because there may be a security hole in your architecture.
And so, this is the purpose of Amazon Detective.
The name is quite explicit.
Detective is going to analyze, investigate,
and quickly identify the root cause of security issues
or suspicious activities using machine learning
and graphs in the backend to really allow you
to quickly get down to where the issue is coming from.
And to do so, it's going to automatically collect
and process events from your VPC Flow Logs,
your CloudTrail trails, and GuardDuty
to create this unified view.
And it will, in turn, give you visualizations
with details and context, so you can get to the root cause.
So that's it.
I hope you liked it and I will see you in the next lecture.

Okay, so just a quick note on AWS abuse
which is something that can come up in the exam.
So in case you are suspecting some resources
in AWS, for example, some IPs
or some institute instances that are used
for abusive or illegal purposes,
then you can report it to AWS.
And so these kind of abusive and prohibited behaviors
are going to be around spam, so for example,
you receive undesired emails from IPs owned by AWS,
websites and forums spammed by AWS resources,
port scanning, for example, someone is trying a port scan
from AWS which is not accepted,
a DDoS attack, intrusion attempts,
so someone is trying to logging onto your resources,
or for example, someone that hosts questionable
or copyrighted content
and so therefore it's illegal content hosted on AWS,
you can also contact the abuse team.
And finally distributing malware is also prohibited.
So you cannot host malware on AWS obviously, right?
So in case you suspect any of these activities,
what you should be doing is to contact the abuse team.
So there is either an online form
or you can just send them an email at abuse@amazonaws.com
and that's all you need to know for this in the exam.
Okay, that's it. I will see you in the next lecture.

Okay, so another question that has come up
in the exam is around the root user privileges.
So the root user is the account owner.
It's the first user that is going to be used
when the account is created and the root user
has complete access to all AWS services and resources.
And the idea is that the root user can do some actions
that even the most privileged created user
in your account cannot do.
And so, this is what we have to look at in this lecture.
So, by the way, if you're using the root user,
please lock away the account
'cause you shouldn't be using it
and as well as your excess keys,
secret access keys, this kind of stuff.
So do not use the root account for everyday tasks,
even administrative tasks.
For this, create a specific admin user in your accounts,
but some actions can only be performed
by the root user and you have to remember them,
especially the one that I put in bold.

So, the first one is that only the root user
can change the account settings,
such as the account name, the email address,
the root user password and root user access keys,
view certain tax invoices, close your account,
so it can only be done by the root user
and that's pretty helpful, I would say,
restore IAM user permissions,
change or cancel your AWS Support plan,
register as a seller in the Reserved Instance Marketplace.
And this one is important,
so I'll give you a use case for this.
For example, say you are buying
a Reserved Instance for three years,
but after two years, you realize
you don't need to have it anymore,
so what you can do is that there is a marketplace
in which you can sell back your Reserved Instance
and so to do so, you need to register as a seller first
and only the root user can do it.
Finally, two more things or three more things,
to configure an Amazon S3 bucket to enable MFA,
to edit or delete an S3 bucket policy
that is getting an invalid VPC ID
or VPC endpoint ID, to sign up for GovCloud as well.
So you do have to remember those,
especially the one that I put in bold
because the exam may ask you which types of actions
are only possible to be done by the root user
and you have to know.
Okay, that's it.
I hope you liked it and I will see you in the next lecture.

So now let's discuss IAM Access Analyzer,
and this is a service within the IAM console that is used
to find out which resources
are going to be shared externally.
So S3 buckets, IAM Roles, KMS Keys,
Lambda Functions and Layers, SQS Queues
and Secrets Manager Secrets.
So, the idea is that some of these obviously
can have resource policies attached with them
or they can be shared with other accounts.
But sometimes you forget about sharing these items
and it can be a security risk for your company
because some of the app may be accessible
by external sources.
And so you define a zone of trust,
which is going to correspond to your alias accounts
or your entire alias organization,

and then anything outside your zone of trust that has access
to the resources said above
are going to be reported as findings.
So for example, we have an S3 buckets,
we can share it with a specific role, a user, an account,
an external client by IP or VPC endpoint.
But if we define the zone of trust to be our accounts
and only the role of the user and the VPC endpoints
are within our accounts,
then the accounts and the external clients
are going to be flagged as a finding.
And you can look at it within the consultant to take action
if you think this is a security risk.
So in the IAM console, on the left hand side,
you have access analyzer and here you can create
your first analyzer.
So you name it, console analyzer.
This is free by the way to enable.
The zone of trust
is going to be the current account right now.
And the findings are going to be reported
outside of your zone of trust.
You can add tags to the analyzer but we don't need to.
And then I'm going to create the analyzer.
This is going to create a service linked role
which is going to allow the analyzer
to interact with resources on our behalf.
So my scan has not completed and as you can see,
I have three active findings right now in my accounts.
So one SQS queue and two S3 buckets are shared
with all principals.
So this one is shared through a bucket policy
and this one doesn't say how, and this is write access
and this is read access.
So it's very interesting.
For example, let's consider this SQS queue
called a demo S3 notification.
So we can click here to have a link
directly into the resource and then go here
and then go to access policy.
And as you can see, I allow anyone to send a message
and anyone could be anyone from an external accounts.
So this is not good, right?
So what I should do then is have a look at this
and make sure that this is either the intended access
of my queue so I can archive it or a not intended access.
And then I go to the SQS console, I fixed this policy,
and then I rescan. So I'm going to edit this.
And I don't like this policy,
so I'm going to just remove this policy overall,

because I don't want to allow anyone
to send a message into my SQS queue.
I will click on save and now I will do a rescan.
And now the status is resolved
because the access is no longer allowed.
So if I go back to my findings now only have two findings.
And here as well, I have two findings
related to my S3 buckets being public.
And so maybe this is an intended access
in which case I will archive this buckets.
And then If I go back to my findings,
it will be into the archived column.
We see we have the other one from the resolve column
and we can still look at the active ones.
And then finally, if you wanted to always archive
these kind of S3 public buckets,
you can go into archive rules
and you can create your own rule
to automatically say which criteria
should make a finding irrelevance. Okay.
If you want it to.
So that's it for this lecture. I hope you liked it.
And I will see you in the next lecture.

So let's summarize this section
on security and compliance and I have to agree,
yes, this was a very, very long section
and I'm sorry about that.
You have to just remember a lot of things.
So the shared responsibility model on AWS,
we've seen it at length in this course
so that you know about it.
Shield is a way for you to get automatic DDoS protection
and then if you use Shield Advanced, you get 24/7 support.
You get WAF web application firewall, which is a firewall
to filter incoming requests, based on specific rules.
KMS to manage your encryption keys on AWS,
CloudHSM to have hardware encryption.
And this time, it's not AWS that manages the keys,
it is ourselves that manages the encryption keys.
AWS will only manage the hardware behind it.
ACM, so AWS Certificate Manager is a way for you
to provision, manage, and deploy SSL and TLS certificates
and to get in-flight encryption.
Artifact is going to give you access to complaints reports,
such as PCI, ISQ, et cetera, et cetera.
GuardDuty is a way for you
to find malicious behavior automatically
by analyzing VPC logs, DNS logs,
and CloudTrail logs and Inspector,

you'll find software vulnerabilities in EC2, ECR container
images and Lambda functions
and Network Firewall, which allows you
to protect your VPC against network attacks.
Next, we have Config, which is allowing us for compliance
to track configuration changes
and also create rules to check compliance of these
resources configuration over time.
Macie is a way for us to find sensitive data.
For example, PII data.
So personal information in Amazon S3 buckets.
CloudTrail is a way for us to track API calls,
made by users within the accounts.
Security Hub is a way for us
to gather all the security findings from
so many different services, from multiple AWS accounts
into one place and really act on these
security findings directly from there.
Detective is in case we have a security finding,
how do we get to the root cause very quickly,
and this is with Detective, which is going
to link up all these services together
and help you with that.
The Abuse team is a team that you report
to when you see abusive behavior by using AWS resources
for abusive or illegal purposes.
And you either have a form or you send them an email.
And then you have to remember,
I think the four most important things
that a root user can do in your accounts.
It can change the account settings,
it can close your AWS accounts, it can change
or cancel your support plan, or it can register as a seller
in the Reserved Instances marketplace.
We have IAM Access Analyzer, which is used
to identify which resources are going
to be shared externally, so outside of your zone of trust.
And finally, we have the Firewall Manager that is used
to manage the security rules across an organization,
that is for your security groups, but also for WAF
and Shield and so on.
Alright, so that's it for this lecture.
I hope you liked it and I will see you in the next lecture.
#_____Section 17 _____

Welcome to this section on machine learning.
So machine learning is not something you need to know
in-depth for the exam but there are a few
certification services that you need to know.
So the first one is Amazon Rekognition,

and as the name indicates, it's used to recognize objects,
people, text and scene in images and videos
using machine learning.
So you can do facial analysis and facial search
to do user verification and count people.
And with this, we can create a database of familiar faces
or compare against celebrities.
The use cases for Rekognition would be labeling,
content moderation, text detection,
face detection and analysis,
for example the gender, the age range and the emotions,
face search and verification, celebrity recognition,
and finally, pathing.
So if you wanna learn more, you can check out their website
which I think is very helpful.
So on the Rekognition websites,
we can we can automate our image and video analysis
with machine learning.
And I really like this website
because it shows you how it works.
So for example, for this image,
we can identify the elements of this image,
for example, a person, a rock and mountain bike,
a crest and outdoors.
We can label, for example, what we see in the images,
for example, Golden Retrievers or dogs.
Then I can look at content moderation
to make sure that it's appropriate for all ages.
I can detect text, for example for a run,
we want to see the numbers of each runner in the run.
We can do face detection and analysis.
For example, this person looks happy, she is smiling,
her eyes are open and she's a female.
Face search and verification,
maybe if you have a security application.
Finally, if you want to recognize a celebrity,
you can take a picture of them,
and this is the CTO of AWS.
And then finally, pathing.
For example, if you're monitoring a soccer game,
you could see where everyone is going to do
maybe some real time analytics.
So Rekognition is just a service you need to know
at a high level for the exam
but I really like this page
because it demonstrate the use cases.
I will see you in the next lecture.

So now let's talk about Amazon Transcribe.
So as the name indicates,
it allows you to automatically convert speech into text.
So you pass in some audio
and automatically is going to be transcribed into text,
so you could say, hey, hello,
my name is Stephane and I hope you're enjoying the course.
Now, how does it do it?
Well, it uses a deep learning process called the ASR,
the Automatic Speech Recognition,
to convert speech to text very quickly and accurately
and there are a few features you need to know about.
The first one is that you can automatically remove any PII,
so personally identifiable information, using reduction.
So that means, for example,
if you have someone's age or name or Social Security Number,
this can be automatically removed
and also you can have access
to automatic language identification for multilingual audio.
So if you have some French and some English,
some Spanish, Transcribe is smart enough
to recognize all of those.
So the use cases for Amazon Transcribe
is to transcribe customer service calls,
to automate closed captioning and subtitling,
and also to generate metadata
for media assets to create a fully searchable archive.
So let's have a look at Amazon Transcribe.
So I can create a transcript and I can see
that the specific language right now is set to English US,
so now if I click on start streaming,
hello, I really like this course.
As you can see, the outcome of the audio
get directly transcribed into some text,
so that's pretty cool, right?
Also, you can have the setting to remove some content
and you can remove PII identification.
So I could say, hey,
you can identify and redact it to remove it
and these are the kind of things that can be removed.
So for example, let's have an example
and try it out, so I won't show this all again.
Hello, my name is Stephane, I am 31 years old
and my phone number is 910-747-280
and as you can see now, the things got hidden.
So my name is hidden and my phone is hidden.
Obviously this was not my real phone number.

You cannot try it.
Okay and last thing I want to show you
is that you can actually stream into multiple languages.
So you have automatic language identification.
So I would choose English and French
and then let's start again.
Hello, this is some recognition happening in English.
(speaking in French)
Pretty awesome, right?
Well, that's it for Amazon Transcribe.
If you wanted to play some more,
you can have a look at all the options in the bottom,
but that's it for this lecture.
I hope you liked it and I will see you in the next lecture.

Polly is the opposite of transcribe.
You turn text into speech using deep learning.
This allows you to create applications that will talk.
For example, it says, "Hello, my name is Stephane
and this is a demo of Amazon Polly."
And then, with Polly, it would generate an audio
which I can play right here.
Hi, my name is Stephane
and this is a demo of Amazon Polly.
I think I'm better at speaking than that,
but (chuckles) this gives you a good demo, right?
And you can play with it on the console.
So let's go into the Amazon Polly console and try Polly.
And we can say, "Hi, my name is Stephane.
I love AWS courses,
and I love machine learning."
Okay. And then, you would just press listen.
Hi, my name is Stephane.
I love at AWS courses, and I love machine learning.
Pretty cool, right?
And if you wanted to have a more robot sounding voice,
you would use a standard engine.
Hi, my name is Stephane.
I love AWS courses, and I love machine learning.
And that's it. That's Polly.

Now let's talk about Amazon Translate.
So Translate, as the name indicate,
is a natural and accurate language translation.
Translate allows you to localize content,
for example, your websites and your application,
for your international users
and easily translates large volume of text efficiently.

So for example, say in English, I say,
"Hi, my name is Stephen,"
In French, it would be. (speaking in foreign language)
In Portuguese, it would be. (speaking in foreign language)
And in Hindi, it would be. (speaking in foreign language)
Okay, I just showed off.
So that was it, super easy service.
I hope you liked it.
And I will see you in the next lecture.

So now let's discuss Amazon Lex and Connect.
So Amazon Lex is the same technology that powers
the Alexa devices by Amazon.
And if you don't know Alexa, Alexa is like a little device
that you have in your home and you say,
"Alexa, what's the weather like tomorrow?"
And it will reply, "The weather is good,
it'll be 24 degrees and it'll be sunny."
So, the idea is that with Amazon Lex,
you get automatic speech recognition, so ASR,
that allows you to convert speech,
so spoken words, into text.
There's also a very good thing
because Lex understands the intent of text, callers
by doing natural language understanding,
so it understands sentences.
And Amazon Lex is a technology
that will help you build chatbots or call center bots.
And talking about call centers,
how do we build a call center?
We have Amazon Connect.
So Amazon Connect is a visual contact center
that allows you to receive calls, create contact flows,
and it's all cloud-based.
And it can integrate with other customer
relationship system, management systems, so CRMs
or AWS services.
And the cool thing about Amazon Connect versus traditional
offering is that there is no upfront payment.
It's about 80% cheaper
than traditional contact center solutions.
So the whole flow of building a smart contact center is
that you will have, for example,
a phone call to schedule an appointment that is made
into a number that is defined by Amazon Connect.
So they call Amazon Connect.
Lex is streaming all the information from this call
and understand the intent of the phone call,

and therefore, it will invoke the right lender function.
And that lender function, for example,
can be very smart and say, "Hey, someone has said
to schedule a meeting tomorrow with Tom at 3:00 PM."
Okay, I will go into my CRM and schedule that meeting
by writing some code.
So this is the whole idea behind Lex and Connect.
So remember Lex is for ASR
and Connect is for contact centers.
So that's it.
I hope you liked it and I will see you in the next lecture.

Now, Amazon Comprehend
is a very simple service from an exam perspective.
All you need to know is that Comprehend is
to comprehend stuff,
so it is for Natural Language Processing, or NLP.
So anytime you will see NLP on the exam,
you would think Amazon Comprehend.
Now what it is, it is a fully managed and serverless service
and it will use machine learning to find insights
and relationships in your text.
For example, it can understand what is the language
of the text, it can extract key phrases,
places, people, brands, or events from that text.
It can understand, do a sentiment analysis to understand
how positive or negative the text you're analyzing is.
It can also analyze a text using tokenization
and parts of speech.
It also has audio.
And finally, it can organize a collection
of text files by topic, and find topics itself.
So, Comprehend is really about getting a lot of data in
and then Comprehend will do the rest
to try to understand the meaning of that data.
So it's about taking text or unstructured data
and structuring it around these features.
So some sample use cases of Natural Language Processing,
or NLP, are for example, to analyze customer interaction.
So say you have a bunch of customers sending you emails
and you want to understand overall based
on your support service what leads to a positive
or negative experience from your customers.
So we use Comprehend to extract these features
and then you will have the business insight
and then you can improve your business,
thanks to that analysis.
You can also, for example, create
and group articles by topics that Comprehend will uncover.
So imagine you have a lot of articles

and you want to group them together instead
of going to them one by one, you would feed them
into Comprehend and then it would output you
the topics you need to group them by.
So that's it.
So again, a bit more information than what you need,
but Comprehend from an exam perspective is
for Natural Language Processing.
So hope you liked it,
and I will see you in the next lecture.

Okay, so now let's talk about Amazon SageMaker.
So SageMaker is a fully managed service
for developers and data scientists
to build machine learning model.
So all of the services so far we've seen
in this section that are all
managed machine learning service with
a very specific purpose,
for example translates some text,
transcribes some audio,
or convert text into audio
or analyze parts of a text,
but SageMaker is a higher level
machine learning service where you have
your actual developers or your data scientists
within your organization create
and build machine learning model.
So it is a lot more involved
and a lot more difficult to use.
Now, when you want to do this kind of processes
to build a machine learning model,
you have to do a bunch of steps that I will show
you in a second,
and all these are quite difficult to do
in one place,
plus you need to provision some servers
to perform these competitions to create these models,
and that can be cumbersome as well.
So this is where SageMaker comes in.
So SageMaker will try to help you all along the way
for the process.
So I will show you what machine learning actually is,
and this is a simplified version,
if you are a machine learning expert
don't get mad at me please,
but let's say you wanted to build a model,
or let's say I wanted to build a model
to predict what score you're going to get
at your certified CLAP practitioner exam.

So how will I do it?
Say for example that I am a developer
or a data scientist,
so I'm going to gather all your data from
the actual performance of my students.
So I will ask maybe 10,000 students
to give me information about how many years of experience
in IT they had,
how many years of experience with AWS they had,
how much time they spent on the course,
how many practice exams they did, etc. etc,
so I'll try to gather as much data as possible,
and then I'm going to label the data.
So that means you need to say which columns
corresponds to what,
and also you need to give some kind of score,
and the score is the actual score for me
at the exam that someone obtains.
For example someone did not pass,
670, maybe he didn't do the course completely,
that would be the reason.
Maybe someone passed with high grade,
so 990 or maybe someone with an even higher grade 934,
so, each student will get a specific score,
and my guess is that based on the data that I've collected
I can predict what the score will be.
So I've first done the labeling,
and that labeling process can be quite complicated to do
in practice.
Then I need to build a machine learning model,
which is how can I predict these scores
from the historical data.
So, then you build that machine learning model
and then you have to train it
and tune it.
So this is another part that's actually quite difficult
to do, which is how to refine my model over time
to better fit my data and my outputs.
Okay, so now SageMaker and all of this will help
you with the labeling,
the building, the training
and tuning, but not only.
Now we have a machine learning model
and it is created,
it's fully working.
So now I need to use it.
So this is called deploying machine learning models.
So, we're going to get new data coming in.
For example you are the new student,
and I'm going to survey you,

and I'm going to say okay,
how many years of experience do you have in IT,
with AWS, how much time have you spent on this course,
and then I will apply based on this data
you just given me,
I will apply the machine learning model that I have created
from before.
And then the machine learning model will say,
for example, hey it sends based on the data you have
I'm going to predict that this student
will pass with a score of 906.
And this whole process,
of labeling, building the model,
training it, tuning it,
applying it can be all done within SageMaker.
So, that's it for a quick introduction,
but I hope you liked it,
and I will see you in the next lecture.

Now, let's talk about Amazon Forecast.
And this is a very easy one
because it allows you to do forecasts.
So this is a fully managed services
that will use machine learning
to deliver a highly accurate forecast.
For example, you want to predict the future sales
of a raincoat.
The idea is that it's going to be 50% more accurate,
than looking at the data itself.
And you reduce forecasting time from months to hours
by using a managed service.
So the use cases can be multiple,
whenever you need a forecast.
For example, product demand planning, financial planning,
resource planning, and so on.
So how does it work?
Well, you take your historical time-series data,
for example, and you also add your product features,
prices, discounts, website traffic, store locations,
basically, any kind of data you can
to then enhance your model.
Then you upload this into Amazon S3.
You then start the Amazon Forecast service
which will create a forecasting model.
And you can use that forecasting model, for example,
to say that your future sales
of raincoats are going to be $500,000 next year.
And that's it, super simple.
Whenever you see forecasts at the exam,
think Amazon Forecast.

Another machine learning service on AWS
is called Amazon Kendra.
So this one is a fully-managed document search service
that is powered by machine learning
and it allows you to extract answers from within a document.
That document could be text, PDF, HTML
PowerPoints, Microsoft Word, FAQs, et cetera, et cetera.
So you have a lot of data sources
where these documents may be
and you see some of them right now in the screen
and they're going to be indexed by Amazon Kendra
which is going to build internally
a knowledge index powered by machine learning.
And how does it help from an end-user perspective?
Well, we get natural language search capabilities
just like you go on Google.
So for example, if a user says,
Hey, where is the IT support desk into Amazon Kendra?
Kendra can reply, 1st floor.
And this could be due to the fact that Kendra knows
from all the resources that it took
that the IT support desk was on the 1st floor,
which is quite awesome.
And also, you can just do a normal search
and it will learn from the user interaction and feedback
to promote preferred search results
which is called incremental learning.
Finally, you can fine tune the search results, for example,
based on the important data, importance of data,
the freshness, or whatever custom filters you have, okay?
So from an exam perspective,
whenever you see a document search service,
think Amazon Kendra.
That's it, I will see you in the next lecture.
Next, we have Amazon Personalize,
which is a fully machine learning service to build apps
with real-time personalized recommendations.
So what could be a recommendation?
For example, a personalized product recommendation,
or re-ranking, or customized direct marketing.
For example, a user has bought a lot of gardening tools,
and you want to provide recommendations
on the next one to buy based on a personalized service.
So this is the same technology used by Amazon.com.
So when you go and shop on Amazon.com
and after buying a few products, what you will see
is that Amazon.com start recommending products
in the same category or in completely different categories
based on how you've been searching

and how you've been buying
and user interest and that kind of things.
So Personalize is how you access this from within AWS.
So, you read your input data from Amazon S3.
For example, it could be user interactions,
those kind of things.
Also, you can use the Amazon Personalize API
to have real-time data integration
into the Amazon Personalize service.
And then this will expose a customized personalized API
for your websites and applications,
your mobile applications.
Also, you can send SMS or emails
for personalization as well.
So you have all these integrations.
It takes days, not months, to build this model.
So you don't need to build, train, and deploy ML solutions.
You can just use this bundled as is.
And so the use cases is going to be retail stores,
and media, and entertainment.
So from an exam perspective,
anytime you see a machine learning service
to build recommendations and personalized recommendations,
think Amazon Personalize, that's it.
I will see you in the next lecture.

So now let's talk about Amazon Textracts.
And Amazon Textract is used to extract texts,
so, hence the name.
So you extract text, handwriting,
or data from any scanned document
and behind the scenes, of course,
uses AI or machine learning.
So we have, for example, a driver license,
and then we upload it into Amazon Textract,
and then, automatically, will be analyzed,
and the results will be given to you as a data file,
and so you'll be able to, for example,
extract the date of birth, document ID, and so on.
So you can extract any data, even from forms and tables,
and you can read PDFs, images, and so on.
So the use cases for extracting texts are multiple,
but you could be for financial services
to process invoices or financial reports,
could be for healthcare,
for medical records, and insurance claims,
or for the public sector, for example, for tax forms,
ID documents, and passports.
So that's it for this lecture,
I hope you liked it,

and I will see you in the next lecture.

Now let's talk about
the machine learning services on AWS,
and you have to remember them all going into the exam.
So, recognition is a way for you to do face detections,
to do labeling, and celebrity recognition.
Transcribe is for you to get,
a way to get subtitles.
For example, to convert your audio into text.
Whereas Polly is the opposite.
It allows you to get,
to use your text and create audio out of it.
Translate is for you to get translations.
Lex is to build conversational robots or chatbots.
And if you bundle it with Connect,
then you can create a cloud contact center.
Comprehend is a way for you
to do natural language processing.
SageMaker is a fully featured machine learning service
that is accessible to developer and data scientist.
Forecast allows you to build highly accurate forecast.
Kendra is going to be an ML-powered document search engine.
Personalize is used for real-time
personalized recommendations for your customers.
And Textract is used to detect text and data,
and extract them from various documents.
So, hopefully you can remember this whole list.
This will get you a few points on the exam.
I hope you like this lecture,
and I will see you in the next lecture.
Autoscroll

#_____Section 18 _____

Now let's tackle a long section
on Account Management, Billing and Support.
And we will be starting with AWS Organizations.
So it's a very simple service, it's a global service.

And the idea is that by creating an organization
you're able to manage multiple AWS accounts.
The main account is going to be called the master accounts
and all the other ones will be called child accounts.
The cost benefits you get from using an organization
is that you get consolidated billing.
That means that all the accounts will be paid
by just the master accounts.
And so you will have one longer bill at the end.
So you don't just set up as payment method
for all the other accounts.
The other thing is that you get pricing benefits
from aggregated usage.
So when you use a lot EC2, when you use a lot S3
you get a discounts because you've used that at lots.
And so if you have multiple accounts,
you could lose that volume, but with an organization
because the billing is consolidated
the aggregated usage is as well consolidated.
And that means that you get more discounts.
Also, if you're using reserved instances, they're shared
across all the accounts to make sure
that if one account does not use a reserved instance
another one can and again, maximize the cost savings.
There's an API that is available to automate AWS account
creation to do so automatically, which is very helpful.
For example if she had some processes to create an accounts
programmatically for someone, for example
is a sandbox accounts.
And then finally you can restrict account
privileges using a Service Control Policy or SCP.
And that is a common exam question.
So all the things highlighted in bold
in this slides are going to be common exam questions.
And I'm gonna go a bit deeper on the organization.
So you can have a multi account strategy in AWS.
That means that you wanna create the accounts
for example, per department per cost center, per
environment, for example dev test and prod based
on regular share restrictions.
For example, if you don't want a service to be using
an account you can use an STP or if you want to isolate
the resources better you could have different VPC
in different accounts and is also very good to have separate
per account service limits
and also isolated accounts for logging.
So all of these could be multi account strategies is really
up to each organization to choose what type

of accounts they want.
So the idea is that you have two options
you can use Multi Accounts or One Account and Multiple VPC.
That is a trade-off, I personally liked
the multi account better.
You can use tagging standards across all the accounts
for billing purposes and we'll see billing
in depth in the section.
And you should enable CloudTrail on all the accounts
send that the logs to a central S3 accounts.
And also as well for the CloudWatch Logs
you should send them all to a central logging accounts.
So how can you organize your accounts?
Well, you can organize them by business units, for example
with a master account.
And then we have the sales OU, we have the retail OU
and the finance OU.
And within each OU, so each organizational units
you will have multiple accounts.
Or you can organize them by environments, production,
developments and tests.
Or we can have them project-based, for example
project one, project two, project three
or a mix of all these things.
Okay so an organization it looks like this.
The Roots OU contains everything.
It contains the master accounts
and then you can create a different OU.
So the Dev OU maybe with the two accounts in it.
The Prod OU maybe with two accounts in it.
And within the Prod OU you can also have different OU.
So a Finance OU and an HR OU with their respective accounts.
Oh, that makes sense.
There's something called a Service Controlled Policy or SCP.
It allows you to whitelist or blacklist IAM actions applied
at the OU or account level
but it doesn't apply to the master accounts.
The SCPs have no effects on the master accounts.
So the SCPs, we'll see an example of very shortly.
They can be applied to only the users and the roles
of the accounts, including the roots.
So if you apply an SCP onto your account
with an OU and you say you cannot use EC2
even an admin within an account can not use EC2.
But the SCP does not apply to service-linked role.
So this is a service roles that other services
use to integrate with organizations.
Okay, SCP must have an explicit Allow to allow things.

So by default, it does not allow anything.
And so use cases for SCP and this is what the exam
will test you on will be to restrict access
to certain services.
For example, you were saying, okay in my production accounts
you cannot use EMR or to enforce PCI compliance
by explicitly disabling services that are not compliant
with PCI yet in AWS.
So I'll try and make this a little bit clearer.
Let's have a look at our OU.
So we have the Root OU with a root accounts, a production OU
with an account A in it and with an HR OU
with account B and a Finance OU with account C.
So let's assume that you usually do this on the Root
OU you add an SCP called FullAWSAccess
which tells that every services in every action can be done
basically allowing you to use your accounts.
But let's apply a DenyAccessAthena SCP
onto Veet master accounts.
Now what can the master it can do or cannot do?
Well, the master accounts can do anything because
it inherited the full alias access SEP from the root OU.
And even though we have attached a DenyAccessAthena SCP
to the master accounts because it is the master account
of your root OU no SCP apply.
And therefore this SCP replied to the master account
is completely not taken into account.
So to summarize, we've inherited stuff from the Root OU
and the SCP applied to the master account
to deny anything does not apply.
Now, let's go on.
We have a DenyRedshift SCP that is applied to the Prod OU.
And an AuthorizeRedshift SCP applied
it to the accounts A.
So now about account A, it can do anything because
you have full access SCP.
But it cannot access Redshift because
there's a DenyRedshift SCP applied to the Prod OU.
And even though I would attach an AuthorizedRedshift SCP
to my account A, because we have an explicit deny
on Redshift at the OU level the deny is going to take
precedence over the authorized.
So even though I have this authorized Redshift SCP
on the account A, actually that authorized is useless
because we already have a deny at the OU level.
So it's interesting
for you to know that this is a full tree.
And so account A is going to inherit the SCP at its level,

at the OU level and even the roots of the OU.
So it goes like a tree and so if one of these says deny,
then it's going to be a deny.
Now let's look at HR OU, it has a DenyAWSLambda SCP.
And so what about account B?
Well, it can do anything because of the full access
but it cannot access Redshift because it's within
the Prod OU is the bigger OU.
And also it can not access AWS Lambda
because it's within the HR OU.
So accounts C though in finance OU is not affected
by this DenyAWSLambda the SCP because only apply
to the HR OU but not the finance OU.
And therefore account C has the exact same information
as account A which is to do anything but access Redshift.
Hopefully that makes sense.
If you understand this, you've basically understood SCP
and their power.
So let's take an example of what it looks like
an SCP looks just like a IAM policy.
So this is AllowALLActions, so we allow star on star.
So this star you can do anything.
But DenyDnynamoDB and we're saying the effect is Deny
on dynamodb star for any resource.
Another strategy would be to whitelist
only a certain type of services.
So we're saying allow ec2 star in CloudWatch star
on resource star.
But any other services but ec2 in CloudWatch
cannot be usable.
If you don't know exactly what this means
or you want more examples, there's a link right
here that takes you to the documentation
and shows you different OUs, SCPs you can have.
So that's it for organizations, I hope you liked it.
And in the next lecture I'm going to do an optional
walk-through organization, it's a bit more complicated.
So I would just recommend if you just want to, to just watch
me do and see how I use organization to demonstrate
the creation of accounts and to demonstrate
as well the use of SCP.
So hope you like this and I will see
you in the next lecture.

Okay. So we're going to practice
using the organizations.
For this I'm just going to go into the organization service
and get started.

So as we can see in this example
Organizations is a global service
because it has to do with accounts
and regrouping them together, okay?
The other thing I did is that I created
my own new account for this.
So I created AWS course master account
and on the other window I have AWS course child account
because I don't wanna use my main accounts for this
and I wanted to do a demo with two separate accounts.
So if you wanna follow along
I would suggest creating two new accounts.
Call them as you want so that you can have one master
and one child account within your organization.
So from the master account
I'm going to go ahead and create an organization.
Now within the organization
we have to define the accounts within it.
So as we can see right now,
this is very quick, the organization is created
and we have the root organizational units.
And within it, we have the AWS course master account
which is the master account
or also called the management account, okay?
So we're going to do that.
And the organization is created.
Now we want to add a second
AWS as account into this organization.
And to do so I'm going to add an account.
And we have two options,
either we want to create an account
and you specify the account name,
the email address of the account owner
as well as an IAM role that will be created
in the target account
to be allowed to be managed by the organization.
Or you can invite an existing AWS account,
in which case you need to provide the email address
associated with that account
or the account ID of the account to invite.
And for this, I will just do the name of my account.
So I would just add the email which is
aws-child-account@stephanemaarek.com.
And this is good to go.
We can include the message if you wanted to
and add some tags but I will just go ahead
and send my invitation.
So now my invitation has been sent to my other account
and we can view all pending invitations through this UI
and it hasn't expired yet,

so if in two weeks
it doesn't get accepted, then this will expire.
So what I can do next is go
to my organization on my child account.
And on the left hand side, there is Invitations.
So I click on Invitations.
I'm going to refresh this page.
And now we see my invitation from the master account.
So as we can see in this organization right now
we'll get full control as this organization
has full features enabled
and can assume full control of your account.
So as soon as you're part of an organization,
you accept to be controlled
by whoever is the master of that organization.
So we'll accept the invitation.
And here we go.
Now my account, the child account is enrolled into
my AWS organization and we can only see the organization ID
as well as the feature set.
And an account may have
the ability to leave the organization.
So back into my AWS organization.
Now, if I go to my accounts, I click on AWS accounts.
As we can see now within my organization
we have roots and within roots, we have two accounts now,
the master and the child accounts.
So we can do is now organize our accounts
using organizational units or OUs.
So for this, we'll just do action
and we can create a new OU.
So to do so we'll go on the roots, okay?
And action creates new OU and I can have one,
for example, for my Dev accounts.
And I create the OU.
I can also go again in here and create the OU,
And this time I will say tests and maybe less time
we'll have a product, so I'll just do a prod OU.
And maybe within the prod OU we have different departments.
So I can again create OUs within OUs.
So I can have HR, if we have an HR department
that has production applications,
or maybe we have a finance department
that has analytics applications within it.
So as you can see here
you can create as many nested OUs as you want.
And if you go all the way to your organization
and then you look at the OU,
now we can see we have roots, dev,
and right now, no accounts within dev,

prod and we have finance and HR within prod
and then we have test.
So as we can see, we can start organizing the accounts
and we have many accounts in organization
within specific OUs.
And the reason we do so is to have service control policies.
So what we're going to do is first take our child account
and we want to move it in to,
for example, the finance department within prod.
So I take this account and I can say move
and then I can have it into my finance department
within my prod OU.
So I move the account there.
And now if we have a look we can see
that the finance department contains the course child.
It's best practice as well to leave the management account
under the root OU but you could move it if you wanted to.
Okay. So now we want to enable service control policies
to restrict what my course child account can do.
So to do so we go into Policies and as we can see
we have four different kinds of policies available
to us right now, and they're currently disabled.
So what we can do is take the important policy types
that we want and enable them.
So one we definitely want to enable is the
service control policy, because this will allow you to
restrict what our children account can do.
So this is enabled and I go back to Policies.
We have other ones that could be of interest,
for example, backup policy allows you to
deploy organization-wide backup plans, to ensure
that all your accounts are compliant
and have backups enabled
or tag policies also to help standardize how you use tags
within all the different accounts in your organization.
But for the sake of this hands-on
and from an exam perspective
I believe only service control policies will be used,
but still good to know
that you can apply a backup policy across all the accounts
and a tag policy across all the accounts as well.
Okay. So service control policies are enabled.
And so now what we'd like to do
is to have service control policy defined.
So I'm going to click on service control policy
and this is the documentation, excuse me.
And here we have one service control policy
that has been created so far, which is the full AWS access.
Okay? And the full AWS access allows all the
accounts to access all the services.

But we can create a new policy and attach it.
So we can created a policy called oops-
We can create a policy called DenyAccess to S3
and this will deny access to the S3 service
to whichever OU or account this is attached to.
So in terms of the policy, we could find a statement.
For example, we can find the S3 service in here
and within S3, we can say all actions
and the resource is going to be star as well.
So I'm going to have a star in here.
So we're denied anything on this (murmurs),
a very simple policy
and I'll call it deny S3 as an Sid.
And then I will click on Create policy.
So this, when attached to my accounts,
should deny access to S3.
So we can have a look.
So let's go into our accounts.
Okay. So if we look at the root to you and click on root,
as we can see, there is enabled policy types
which is service control policies.
And if I click on Policies
there is one applied policies that is attached directly
to the root OU, which is the full access to AWS,
which allows everything on root
and all its children to access all the services within AWS.
So if you look at the children
of the root OU, we have, for example, the prod OU.
And if we look at the prod OU, in terms of policies
there are two policies, one that is attached directly
which is the full AWS access,
but also one that is inherited from root,
which is the full AWS access.
So it has duplicated this one for some reason.
And then if I go into children in a go into finance
and click on policies, we have three attached policies.
So one inherited from prod, one inherited from root
and one attached directly.
And this is probably because I've enabled
service control policies after creating the OUs.
So this full AWS access was attached
to every single element within my account.
And if we look at the children of the course
of the finance OU within the prod OU,
and you click on the course itself, the account itself
and go to policies, now we have four.
So we have full AWS access four times.
So you understand at least the concept of inheritance,
which makes sense.
And you can just inherit things from Root that are clear.

You inherit things from the topmost layer,
but what we can do is if we go back one up.
So if we go to my prod and finance OU, for example,
we can attach a new policy.
So I'm going to attach a new policy
and this one will be the DenyAccessS3.
I will attach it
and now that means that anything within my finance OU
should also have this inherited.
So if I click on my course child and then policies,
as we can see the DenyAccessS3 has been inherited
from finance.
So how do we make sure that this is working?
Well, if I go to my account
now my child account, and open the S3 console in a new tab.
We are in S3 and the buckets are being loaded
but as we can see, we don't have permission
to list buckets and therefore we can not use Amazon S3.
And this was due to the policy we have attached to the OU.
So it's quite powerful because we are able to
restrict what an account can do overall,
even though I am logged in right now
with my root user, okay, with my root user of my account,
I still don't have the access to Amazon S3.
So this is very powerful and this is how STPs work.
And hopefully that makes sense for you.
So that's it for this hands-on.
I hope you liked it.
And I will see you in the next lecture.

So, let's talk about Consolidated Billing
for AWS Organization.
So, when you enable this setting,
this provides you two things.
Number one, and I'll describe it in a second.
It gives you the combined usage,
that means that you can combine all the usage
of all accounts together to do two things.
Number one, you're going to share the volume pricing.
For example, in Amazon history,
if you go over maybe five terabytes of data,
then the next service is going to be cheaper.
Well, obviously if you combine
all your accounts together,
if every account is using one terabyte of data
and you have six accounts,
then altogether they will be benefiting
from the discount pricing.
So, the combined usage is very important,

and also,
there are some Reserved Instances
or Savings Plan discounts on one account.
They can be shared across all accounts
again to maximize the discounts
and the savings,
and I will show you how in a second.
The other advantage that it gives you,
One Bill.
So, get one bill for all the accounts
in the organization,
which really can help
your accounting departments
and allow you not to be restricted
in how many accounts you can create for AWS.
So, let's take an example of
Reserved Instance Sharing.
So we have an organization with two accounts
and in the first one, account A,
there is no Reserved Instances,
and account B,
there are five reserved EC2 instances.
Now, let's assume that we are within one AZ
because Reserve Instances are at the AZ level
and we have nine EC2 instances.
As you can see,
three are launched in account B,
and six are launched in account A.
Now, what is going to happen?
Well,
if we have five EC2 Reserved Instances on account B,
then the three EC2 instances
obviously are going to be reserved,
but because we have enabled Reserved Instances sharing,
then two instances in account A
will also benefit from Reserved Instances pricing
and therefore rebuking cost savings.
So, at the end of the day,
we have five Reserved Instances
and four not Reserved Instances in this use case
even though Susan in account B
only launched three EC2 instances
out of the five that were reserved.
Okay?
So, you really need to understand this
because the exam will test you on the shared volume pricing,
so shared volume discounts
and how to share and what it means
to share Reserved Instances,
and finally, the Reserved Instances discount sharing

can be turned off for any accounts in the organization,
including the management accounts,
the main accounts of your organization.
Okay.
That's it for this lecture.
I hope you liked it,
and I will see you in the next lecture.

Now let's talk about AWS Control Tower.
So it is for you an easy way to set up and govern
a secure and compliant, multi account, AWS environment
based on best practices.
So instead of doing everything manually
creating an organization and so on
and then applying security practices,
we have Control Tower where you can, with a few clicks,
creates a multi account, AWS environment.
The benefits is that you can automate the setup
of your environments in a few clicks.
You can automate ongoing policy management using guardrails.
You can detect the policy violations and remediate them.
You can monitor your compliance
through an interactive dashboard
and Control Tower is running on top of organizations.
That means that it will automatically set up organizations
for you to organize your accounts.
And it will implement STP service control policies
to make sure that the guardrails are operating effectively.
Okay, so I'll go see you in the next lecture
for a demo of Control Tower.

Let's go ahead and set up Control Tower.
Now, this is not something
that I would recommend for you to do.
It is quite complicated.
It creates a lot of accounts
and there will be cost associated with creating
and using new services, which can be very, very costly.
So, I would really recommend for you
not to do this hands-on.
But for you to understand what Control Tower is,
I'm going to perform the steps in front of you
and you get a better idea of how it works.
So, we're going to create a Landing Zone,
and so as such, we have a Home Region.
This is where we are going to have
the home of Control Tower.
Then, you can deny access to a few regions
by enabling the region deny setting, so you can enable it
and say, "Okay, some regions should be denied."

Next, you have additional regions for governance.
So which regions do you want to be monitored
for governance purposes.
Again, I'm going to use all the defaults
to keep things simple.
Next, we have the OUs.
So, the OUs are gonna be part of your organization
and there is a Security OU that is being created
for the log archive accounts
and the security audit accounts.
Then, we're going to get an additional OU
called the Sandbox.
This is where you can have your other accounts,
for example, and you can create more OUs
after setting up your landing zone.
Okay, so next we have to create
those specific accounts for log and so on.
So, as you can see for the log archive accounts,
you can create a new account and you have to enter an email.
So for example, stephane+archive@example.com, right?
And this is the log archive accounts.
Next, we have the audit accounts.
And so, we can have stephane+audit@example.com,
and then we click on Next.
So in terms of additional configurations,
there is the AWS account access configuration.
So we can use IAM Identity Center to access
all the accounts within your control tower.
But if you wanted to,
you can also do self-manage account access,
but it is way more complicated.
So again, I would suggest you use the default
of using IAM Identity Center.
Next, we want to enable CloudTrail for everything,
and of course it is better to enable CloudTrail.
So, we'll have a look at CloudTrail
across our entire Landing Zone.
Then, do we want to send logs to Amazon S3?
This is optional.
I'm not going to touch any of these settings.
And finally, do we want to have KMS encryption?
This is optional, but we could have a KMS key
to encrypt everything.
Overall, I'm not going to do this.
I just want to keep it simple for my Landing Zone.
So as we can see now, we have two OUs
created as part of my Landing Zone.
We have the Security and the Sandbox OU.
We have a management account.
We have log archive and audit accounts.

And then, we're good to go,
account access configuration, and perfect.
So, I understand that this will create a lot of resources
on my behalf and this can take a lot of time.
And so as such, I will be paying a lot of money if I do so.
So, I will click on Setup Landing Zone
but you don't have to do it really.
It's just for you to watch this lecture.
So as you can see, the setup is going to take
about 60 minutes, and it's going to take a long time.
It's going to set up two OU, three shared accounts,
a native cloud directory with preconfigured group
and single sign-on access,
and 20 preventive guardrails to enforce policies,
and two detective guardrails
to detect configuration violations.
So, a lot of things are being set up.
I'm just going to wait a little bit until this is done.
Okay, so my Landing Zone is now available.
And it has set up two things.
Two organizational units,
three shared accounts with master accounts
and isolated accounts for log archive and security audit.
There's a native cloud directory with single sign-on access
and I'll show you this in a second.
And then, 20 preventive guardrails to enforce policies
and two detective guardrails
to detect configuration violations.
So, a lot of things was created using Control Tower.
And if I go to Organizations right now,
I can show you right away what was created.
So as we can see here,
we have the three accounts already in my organization.
And if I look to organize accounts,
we see there's custom and core organizational units.
So, in core we have the audit and archive.
And in the custom, we currently have no accounts, okay?
So, we shouldn't manage the accounts
through organizations though.
We should every time manage the accounts
through Control Tower.
And so, here are some recommendations.
So add or register OUs, configure your account factory,
more guardrails and review users and access,
and then review shared accounts.
So, a lot of things are happening here in this dashboard.
We get also access to non-compliant resources
based on the rules that we have defined.
We get some information
around the registered OU that have been created

and whether or not they're in compliance.
That's perfect.
As well as all the enrolled accounts into my accounts.
And for the guardrails,
we can view all the guardrails right here.
We can review guardrails, and so here we get the information
around all the rules that allow,
that are enforced on our OUs, for example.
Disallow the deletion of log archive.
Obviously that makes a lot of sense.
Disallow public read access to log archive and so on.
Disallow configuration changes to CloudTrail.
All these kind of things are excellent to have
and are set up by Control Tower
according to best practices, okay?
You could always create accounts right here.
So, you can view all the accounts here.
And then in the OU you can, then show your OU,
and add an OU if you wanted to.
And here is the guardrails.
The account factory is how you enroll an account
into your Control Tower, which is great.
Users and access.
So, this is how you manage the user access
to your whole account sets.
So, we have a single sign-on right here
and there is a user portal URL right here.
And the way to handle user entity management right now
is with single sign-on, which is a service by AWS.
Shared accounts are here, landing zone settings, and so on.
So as we can see, this is a full management suite
for multiple accounts.
And so if we go into the SSO portal,
as we can see here there's a sign-in button.
And then, there's a password that I have to share.
So, I'll just use the password that I have right here
that I've created from before.
And I sign in.
And now I am into my SSO.
And I'm able to log into any of my three AWS accounts.
So, we have the audit accounts, the log archive accounts,
and the Stephane CCP accounts
directly accessible from this UI.
So for example, if I want to go into the audits,
I can click here to go into the management console
of this audit account or click here for,
get the command line or programmatic access.
So it is really, really neat.
And here I go, I am into my audit account right now.
So it really shows you the power of Control Tower.

Now, it's not something
that you would do on your own, obviously.
And now that I've moved away from my accounts
and then to reload my screen,
but this is to show you that,
yeah it's quite handy I would say.
So, I'm just going to login back into my CCP accounts.
It's quite handy to use Control Tower in your accounts
to set up multiple accounts going to best practices,
and manage it from there.
So, if you are an organization
that wants to have a multiple best practice AWS set up,
then please use Control Tower.
Okay, that's it.
I hope you liked this lecture
and I will see you in the next lecture.

So now let's talk
about the AWS Resource Access Manager Service or AWS RAM.
So, the idea is that you can share with other AWS accounts,
resources that are owned by your accounts,
and this can be any other accounts
or accounts within your organization.
This allows you to avoid resource duplication
and the supported resources include Aurora databases,
VPC Subnets, Transit Gateway, and a lot of other resources.
Now, you don't remember these resources in particular,
but just the idea behind RAM service.
So, let's say we have a cloud accounts
and we have a VPC within the cloud accounts
and within the VPC, there are private subnets.
Now, we can actually share this VPC with other accounts,
such as Account one and Account two
which are different accounts, still have access
to the same VPC and the same subnets.
The result of this is that, for example,
if an account decides to create EC2 instances
and the load balancer within your VPC Account two,
with their own EC2 instances,
can access directly from the network,
for example an Amazon RDS database,
or your application load balancer.
So, the idea here is that because we've shared our VPC,
all the resources within the VPC
are going to be able to connect with each other
from a network perspective,
which is going to simplify our deployments.
So, that's it for this lecture.
I hope you liked it, and I will see you in the next lecture.

So now let's talk about
the AWS Service Catalog.
So the users that are new to AWS, they have too many options
and they don't follow this course, for example
when they get started.
And so if you leave them the option
to do anything they want in AWS,
whatever they create may not be in line
with what the rest of your organization is doing.
And some users that just want to have access
to quick self-service portal
that allows them to launch a set of authorized products,
and these have to be predefined by admins.
So as you can see,
these could include virtual machines, databases,
storage options, and so on.
So to do so, to have this self-service portal,
you use the AWS Service Catalog.
So very simply, as an admin in AWS and Service Catalog,
you're going to create products.
And products are just CloudFormation templates
with the proper parameters.
And you can include them in a portfolio,
and a portfolio is a collection of products.
And then you define who has access
to launching what products within my portfolio.
And then as a user,
you log in to your portal on Service Catalog
and you have a quick list of all the products
that you can use because of your permissions.
And then the users can launch these products
and automatically they get provisioned by CloudFormation.
But we know
that they've been properly configured, properly tagged,
and that they respect the way
our organization is supposed to work.
So say for example,
that a user wants to have access to a quick RDS database
but doesn't know how to create one properly.
Then you could offer this as a service
from within the Service Catalog.
So hopefully that makes sense. I hope you liked it.
And I will see you in the next lecture.
So now let's talk pricing models in AWS.
AWS has four different pricing models.
There is pay as you go,
where you pay for what you use.
You remain agile,
because you can start and stop
and delete resources whenever you want,

responsive,
and you can scale to meet demands as demand happens.
There's also a model where you save when you reserve.
So this is to minimize risk
and to get a predictable budget
and to comply with long-term requirements.
For example, we can do instance reservations
for EC2, DynamoDB, ElastiCache, RDS and Redshift.
Also, you get to pay less by using more.
You get volume-based discounts,
for example, on Amazon S3.
And as AWS grows their infrastructure,
they will have some cost savings,
and they will pass them onto you.
So AWS is very famous for doing cost reductions
every month or every year,
which is really good,
because as their infrastructure grow,
and more people use AWS,
they have volume and scale,
and therefore, they will pass on that economy of scale
onto you to get discounts.
On top of it, in AWS,
some services are going to be free,
and other will be part of the free tier.
So I AM is free,
VPC is free,
and consolidated billing is free as well.
If you use the Elastic Beanstalk, CloudFormation and ASG,
they are free services.
But whatever these services creates,
obviously, you need to pay.
So Elastic Beanstalk,
if you create an application load balancer
and EC2 instance,
it has to be paid by you.
And your Auto Scaling group obviously,
when it adds more EC2 instances,
you're going to pay for those.
And obviously, what is created through CloudFormation,
you will have to pay as well.
Those are free tier that you can find at this address.
And for example, as part of the free tier,
you get a t2.micro instance for a year,
or you get S3, EBS, ELB and data transfer for free
at up to a certain amounts.
Okay. So we have the free tier here,
and we can see that there are kinds of free tiers.
There is featured.
There is 12 months free

which is when you start a new account.
There is always free
which is going to be free even after 12 months.
And trials, for example, the 30-day trials
for GuardDuty, for SageMaker, Inspector and so on.
So here, I think it's really helpful for you
to have a look at it and to see what is free.
I think the most important ones are going to be Amazon EC2,
Amazon S3, Amazon RDS and so on.
So now let's do a deep dive into the pricing
for different services.
For example, for EC2,
with the On-Demand,
you are only charged for what you use,
which is the number of instances
as well as the instance configuration,
what's the capacity number of CPU and RAM,
the region you in,
the operating system and the software you install
on your EC2 instance,
the instance type and instance size.
And if you're using a load balancer, for example,
how long the load balancer will run
and the amount of data processed by your load balancer.
Also, you can enable detailed monitoring
on your EC2 instance to get CloudWatch metrics every minute
instead of every five minutes,
and you will have to pay for that as well.
So now let's talk about the different models
for pricing on EC2.
The first one is with On-Demand instances.
And you're going to have a minimum billing time
of 60 seconds.
And then you're going to pay per second
for Linux and Windows
or per hour for your other EC2 instances.
But if you know you're going to use your instances
for a long time,
then you should use Reserved instances,
and it's going to give you up to 75% discount
compared to the On-Demand price for the hourly rate.
But you need to commit for a long period of time,
so either one or three-year commitment.
And you get more discounts if you pay early.
So you can pay everything upfront,
and you get more discounts.
Or partial upfront,
so just a part of the total amounts.
Or no upfront.
But this is going to give you a bit less discounts.

You also have Spot instances,
and you get up to 90% discount
compared to your On-Demand hourly rate.
And what this means is that
you're going to bid for unused capacity
in the EC2 instances.
And so because you use these unused capacity,
you can get it at a very, very aggressive discount.
But with Spot instances,
you run the risk of losing them
in case someone is willing to pay a bit more
for your Spot instances.
You can also have Dedicated Host
which can be On-Demand and can be Reserved as well.
And this is a tendency way of running EC2.
This means that you are going to run a loan on these hosts,
because they're dedicated to you.
And finally, there's another component called Savings Plan
which is an alternative to all these things above.
And I'm going to describe Savings Plans to you
in a future lecture.
So don't worry about it too much.
Next, we have Lambda and ECS,
again under compute category.
So Lambda, you pay per API call,
and you pay per duration of your Lambda functions
times the number of the amount of RAM
that you have assigned for your Lambda function.
For ECS, you get an EC2 launch type model.
That means that you don't get any fees for using ECS,
but any time you start an EC2 instance
underlying your ECS cluster,
then you will have to pay for that EC2 instance.
For Fargate, it sort of different,
because we don't manage EC2 instances,
and therefore, we're going to pay for each container
for the amount of CPU and memory
that gets assigned for your container.
Now let's talk about the pricing
of the storage on Amazon S3.
So as we saw, there are different storage classes
that you need to know for the exam.
We have S3 Standard, S3 Infrequent Access, S3 One-Zone IA,
S3 Intelligent Tiering, Glacier and Glacier Deep Archive.
As well on S3, you're going to pay
for the number and the size of the objects.
And the price is going to be tiered,
so the more volume you have,
the more objects and the more,
the bigger the objects are,

the more discounts you're going to get.
Then if you do request in and out of Amazon S3,
you're going to pay for these requests.
You're also going to pay for any data transfer
out of the S3 region.
So sending data into S3 is free,
but retrieving data from S3,
you will have to pay something.
If you use S3 Transfer Acceleration,
it is also a cost that has to be incurred for it.
And any time you do a Lifecycle transition
between the storage classes defined above,
then you're going to have to pay for it.
A similar service is going to be EFS.
Because EFS, you're going pay per use.
It will also have an infrequent access tier
and lifecycle rules.
A different way to do storage pricing is with EBS.
So with EBS,
we're going to have pricing based on the volume type
that we've provisioned,
so the performance of that volume.
Then the size of the volume in gigabytes
that you provision in advance.
Okay, you don't pay per use with EBS.
You say, yes, I want one hundred gigabyte EBS volume,
and you're going to pay for it
regardless whether your use it or not.
Then the IOPS, so the performance of your volumes.
So if it's a General Purpose SSD, it's included.
But if it's Provisioned IOPS,
you're going to pay for the IOPS provision.
And for Magnetic, the number of requests.
Then you're going to pay for your EBS snapshots.
So the more snapshots you get, the more you pay.
So you're gonna get a cost
per gigabyte of snapshot per month.
And then data transfer.
Any data transfer out of EBS is going to be paid
and is going to be tiered for volume discounts.
But anything you write into EBS inbound is going to be free.
Now let's talk about database pricing,
for example, for RDS.
So it's a per hour billing.
And based on the database you use,
you have different pricing.
So the engine type, the size, the memory class.
Then for an RDS database,
you can either have On-Demand as a purchase type,
or you can reserve an RDS database

which is a very smart thing to do,
for example, for one or three years,
and you can pay upfront for that RDS database.
Then if you do RDS backups,
you're gonna have to pay for those.
But usually, there is no charge for backup storage
up to 100% of your total database storage for a region.
And in the experience of AWS,
that means that all in all,
the backup storage should be free
if your database is not all the way up to full.
Then for RDS,
you're going to pay obviously for the underlying storage
in gigabytes per month
based on the EBS volume that you provision
for your RDS database.
You're going to pay for the number of input
and output requests per month.
The deployment type.
Is it a Single AZ or is it Multi AZ,
in which case you have to pay for two RDS databases.
And as well as the data transfer.
So any data transfer out of your database
is going to be paid and tiered based on volume.
And any transfer into your database is free.
Now let's talk about CloudFront.
So CloudFront is a CDN,
and it is global,
so therefore, the pricing is going to be different
based on where the content is served from.
So you have different pricing based on usually continents.
So for example, America, Europe, South America,
Japan, Australia, India and so on.
And this is in the table
that you can see in this slide down.
And then the more you're going to use CloudFront
in a specific edge location,
the more you're going to get discounts,
and you're going to get to an aggregated bill
for all the edge locations.
You're going to pay for any data transfer
out of CloudFront,
but not for in.
In is always free.
Finally, you're going to get a paid bill
based on the number of requests
that are made into CloudFront.
They are counted.
Okay. So in AWS finally,
you need to understand the networking costs

which are, to be honest, very, very complicated.
But I try to simplify it in this slide.
So you have a region,
and we have two AZ.
And say we have an EC2 instance in one AZ.
Any traffic going inbound of your EC2 instance
is going to be free.
And then say you have another EC2 instance
in the same AZ.
If these two EC2 instance talk together,
then the traffic is free using the private IP.
Now say you have another easy three instance in another AZ.
Then if you use the public IP of the EC2 instances
to talk to one another,
then the traffic will have to go through the internet,
and you're going to pay 2 cents per gigabytes.
But if you're using the private IP instead
to make these EC2 instance talk to one another,
then it's going to be 1 cents per gigabyte.
So overall, I would recommend you always use the private IP
if you can to have communication between your EC2 instances.
And if you happen to have EC2 instance in another region,
then you're going to get a fee of 2 cents per gigabyte
as an inter-region cost.
Okay.
So summary is that you should use a private IP
instead of a public IP
for good savings and better network performance.
And then if possible,
if you want to have the maximum amount of cost saving,
use one Availability Zone,
but you sacrifice obviously the high availability.
So there's a trade-off here
between cost and high availability.
So that's it for an overview of pricing on AWS.
You won't be testing on the actual details of each service,
but it's good for you to understand
the reasoning behind cost in AWS,
because this is what the exam will test you on
to understand how at a high level it cost
to use some AWS services.
So hope you like this.
And I will see you in the next lecture.

Okay, so now let's talk about a new way
of saving on your cost on AWS, which is called Savings Plan.
So Savings Plan are a much simpler way
to look at your infrastructure on AWS.
Instead of reserving instances, you just commit
to spending a certain dollar amount per hour

for the next one or three years.
So it's easier than the rest because you just think
in terms of dollars and you don't think
in terms of specific resource, and types, and so on.
So let me show you the two types
of Savings Plans you have on AWS.
The first one is an EC2 Savings Plan,
and this is going to give you up to 72% discount
compared to On-Demand prices for EC2.
And you're going to commit to the usage
of the individual instance families in a region.
For example, you say, "Hey, I'm willing to spend
$10 per hour for the next three years
on the C5 type of instances."
But regardless of the AZ they're launched in,
regardless of the size, so you can choose a C5.xl
or a C5.4xl, regardless of the operating system,
so Linux or Windows, and regardless of the tenancy
if you want to choose shared or dedicated host.
You can choose, just like reserve instances,
to pay everything upfront, partial upfront, or no upfront.
And, obviously, if you pay everything upfront,
you're going to get a bigger discounts.
And the other types of Savings Plan is even more flexible.
It's called the Compute Savings Plan.
And this one is going to give you up to 66% discounts
compared to On-Demand.
And this is regardless of the family, the region,
the size, the OS, the tenancy, and the compute option.
And what do I mean by compute option?
I mean that it's applicable to EC2 instances,
but also Fargate containers and Lambda functions.
So the Compute Savings Plan is really the most flexible
where you don't really think much
about what this applies to.
You just commit a certain dollar amount per hour,
and you get discounts based on that.
And the last option you have for Savings Plan
is the Machine Learning Savings Plan
for services such as SageMaker.
And how to set up a Savings Plan.
Well, you go into the AWS Cost Explorer console,
and it will suggest to you the type of right Savings Plan
that will work for your current infrastructure.
And you can estimate the pricing
of Savings Plan at this URL.
And so if we go to the Savings Plan website,
we have two kinds of Savings Plans,
this Compute and the Machine Learning Savings Plan.
So for the Compute Savings Plan, we have different ones.

So we have Compute or EC2 Instance Savings Plan.
And here on this UI, you can basically navigate it
so you have for EC2, for Fargate,
for Lambda, or for EC2 Instances.
And then you can modify the region.
So the region, the relocation type you want,
the region you want, the term length, so one or three years,
payment upfront or not, or partial, and so on.
And then it gives you a list of available instances.
For example, if we take an a1.large,
then using a Savings Plan,
we can expect 31% discounts over On-Demand.
So this UI is great to estimate
all your Compute Savings Plan.
And then for Machine Learning, we have something similar.
So they will be added over time,
but I won't probably add it this lecture
because it's not needed.
So, right now, we have the Amazon SageMaker Savings Plan,
which is enough for us to know from an exam perspective.
And we have the region again, the type, and so on.
And then besides here, we can take a look
at the Notebook instances for SageMaker, for example.
And if we do decide to run
ml.t3.large-Notebook for a long time,
then we can expect savings of 28% over On-Demand, okay.
So if you go and type savings plan in your services search,
you go under Cost Explorer,
there is the top feature, which is Savings Plan,
and it takes you to this URL.
And you can have a look at the fact you can purchase
Compute Savings Plan, which is for Fargate,
Lambda, and EC2 instances.
You have EC2 Instances Savings Plan,
just only for EC2 instances.
And then SageMaker Savings Plan,
which is the Machine Learning Savings Plan.
So if you decide to purchase a Savings Plan,
you can choose whatever Savings Plan type we want.
The term commitment, for example, one to three year.
The purchase commitment, so, for example,
I decide to have an hourly cost of $500.
And the payment options, so Partial Upfront,
and this would be a lot of money to front,
so $6 million to say upfront.
Anyways, you would see the Savings Plan upfront cost,
the monthly payment, and the total cost.
And then if you're ready, you just add it to cart
and you would have a Savings Plan for your infrastructure.
So that's it for this lecture.

I hope you liked it,
and I will see you in the next lecture.
Now, let's talk about AWS Compute Optimizer
which is used to reduce costs and improve performance
by recommending optimal AWS resources for your workloads.
So it's going to do an analysis
of your EC2 instances, your auto scaling groups,
and tell you, for example,
which one are over-provisioned or under provisioned
and then you can build your optimizations
and then you can have a better cost perspective
and also a better performance.
This way it does it is that it will use machine learning
under the hood to analyze your resource configuration
as well as track their CloudWatch metrics
to understand their utilization.
So supported resources by Compute Optimizer
are EC2 instances, auto scaling groups,
EBS volumes, Lambda functions.
And this allows you to lower your costs by up to 25%
without doing much
and the recommendations themselves
can be exported into Amazon S3.
So that's it, I hope you liked it
and I will see you in the next lecture.

So, now let's talk about billing and costing tools
and you have to know them all for the exam
although they can be a little boring.
So, first one is to estimate cost in the cloud.
For this, we have the pricing calculator
and it will have its own lecture, of course.
The second category is to track cost in the cloud.
For this, we have the billing dashboard,
the cost allocation tags, the cost and usage report,
as well as Cost Explorer.
And, finally, you can monitor, again, costs plan.
For this, we have billing alarms and budgets.
So, as you can see, we have a lot of services to know.
But, by the end, you will know them all
and you'll be confident to choose the right one at the exam.
I hope you're excited
and I will see you in the next lecture.

So let's talk about the
a AWS pricing calculator to estimate cost in the cloud.
It is available at this URL and we'll have a visit
in a second, and it allows you to estimate the cost
for your designated solution architecture.

The UI looks like this.
You will know exactly how much this will you for example
on a year.
So here I can create an estimate
from the pricing calculator.
So you need to select some services you're going to use.
For example, let's assume
that we are using EC2 instances.
So will look for EC2 and I will configure it.
And then we can do select a region.
We can say, okay, so this is my EC2 workload.
We can do an advance or a quick estimate.
We'll do quick for now.
So we have Linux and we say, okay, we have each instance
that's for VPU, 16 gigs of Ram.
So we're going to choose a T4gXlarge
and then maybe we want four of them.
Okay. And we expect to have 80% of utilization per month.
Okay. We have a pricing strategy.
So can say, hey, maybe we want to reserve them with
EC2 instance saving plans
for one year and no upfront payments.
Then we can specify Amazon elastic block storage.
We say, hey, we want 200 gigabytes of storage per instance.
And then we have an estimate.
We can click on add estimates
to our cost pressing calculator.
And so, as you can see
this was my estimate for Amazon EC2.
But for example, say, I also want to add a load balancer.
So I will type load balancer
and I will choose elastic load balancing and say, yes
I want an application balancer in the same region.
And then we want one of them.
And we need to say, how many bites per hour going
to per so let's imagine that we have EC2 instances
and IP targets who say, hey
five gigabytes per hour processed of data.
And we get about five connections per second and so on.
So we have an estimator again
and we can add it to our estimate.
And now we have the total 12 months cost
of this architecture.
It's four and a half thousand us dollars.
So this is a way
for you to estimate your cost and it can get very granular
because you can have different services.
You can create groups, you can add support and so on.
So it's quite a very comprehensive service on AWS.
Okay. That's it for this lecture.

I hope you liked it.
And I will see you in the next lecture.

So now let's talk about the ways you can track
your cost in the cloud.
The first way is to use the billing dashboard,
which will show you all the costs actually for the month,
the forecast, and the month-to-date.
This is a very high level tool,
but it will be helpful to just get
an overview of what's happening.
And on the same page, you will also get access
to the AWS free tier dashboard,
which will show you the usage for each free tier based on
what you've been doing so far for the month.
So let's have a look at the billing dashboard.
And for this, you can click on the top right,
and then click on billing and cost management,
or use a search bar and type billing.
So this is one of my main accounts,
because I want to show you some cost.
So as you can see, this is what my bill looks like
on AWS for this account.
So we have a cost breakdown by month.
Here, we have some information around
the month-to-date cost,
and then the last month cost and the total forecasted.
So a lot of good information.
And here, you get a cost breakdown per service per month,
which is gonna be very helpful.
So it turns out that in November and then December,
I had some costs regarding, for example,
the Amazon RDS service.
So this is very important for me to see,
and you can group them by service, by accounts, by region,
and so on to start to drill down into them.
So this is very indeed and this is a nice home dashboard.
One other thing I want to show you as part
of this dashboard is the free tier.
So if you click on free tier, as you can see,
you will see the service tier you're using,
the current usage and the forecasted usage,
as well as what is the free tier limits.
So it turns out that I'm maxing out on many free tier
for Lambda, for SQS and so on.
But then for other service such as the Amazon SNS service,
I have a current usage of about 600,000.
And the forecasted is this,
but I got 1 million free requests a month,
so I'm going to be fine for this month though.

This is very handy to look at the free tier,
and to look at your cost per month from the home dashboard.
Okay, so next we want to explore our bill a bit deeper.
So for this, we're going to use cost allocation tags.
They will allow us to track our cost on a detail level,
and group them together.
For example, I have defined a cost allocation tag.
And then you export your report,
and then you would get the cost
by category in an Excel format.
So for this, you can use different tags,
you can use the AWS generated tags, for example.
They will be automatically applied to the resource
that you create, and they will start with the AWS prefix,
for example, aws createdBy.
And you can also define your own user tags
that will be defined by the user
that will start with the prefix user column.
And this will be helpful, for example,
to tag your resources and then have them
grouped by call center, by tag, by owner,
by stack, by application, and so on.
You can use tagging as well to create groups.
So tags are used for organizing resources.
For example, your EC2 instances, your images,
your load balancers, your security groups,
your RDS databases, your VPC resources, route 53,
your IAM users, et cetera, et cetera.
And if we create resources through CloudFormation,
they will all be tagged the same way
if you remember it from the CloudFormation hands-on.
There's free naming for your tag,
you can do whatever you want,
but common tag names are going to be name,
environment, team, cost center and so on.
And these tags can be used for cost reason obviously,
but also to create resource groups,
to create, maintain a view, a collection of resources
that will share the common tags,
and you can edit those using the Tag Editor.
So to show you where that would be,
you would go in the search and you type resource groups,
and you arrive onto the resource groups,
and tag editor console.
And you open one tab here and one tab for tag editor.
And this would allow you to edit
your tags for your resources.
So for example, for the tag editor in here,
I can look at resource types of, for example,
security groups for EC2, and I will search my resources,

I will find all the security groups,
and I can for example,
manage the tag of the selected resources,
and say that, yes, I'm going to add a tag of department,
and then say IT, review and apply the tag changes,
and it will apply the tag to all these resources.
And then I can create resource groups,
and I say, okay, it's going to be tag-based.
I can select the resource type or so do all resource types.
So this is great.
And I will say, for example,
that I want my departments to be IT, add it.
And here I will have to see
that if I preview my resource groups,
I can find the five security groups
that I've tagged from before.
Then we can call it IT Resources,
and then create group.
And we don't need a space, I'm going to have IT, underscore,
or now, IT, dash, Resources.
Here we go.
And I have created my first resource group
based on this tag.
And this is very helpful to see
what belongs to different departments.
And you can use this tags for cost purposes.
So then I can click on the left hand side
to cost allocation tags,
and actually activate the one I want.
And then I will be able to generate a report
for this specific user-defined cost allegation tags.
So now, let's talk about the reports we can generate in AWS.
We can generate a cost,
and user report to dive deeper into your cost and usage.
And they will give you an information,
and it's going to be the most comprehensive set of cost,
and usage data available on AWS.
It will include all the additional metadata about services,
the pricing, and the reservations.
So you can get some information, for example,
around your Amazon EC2 reserved instances usage.
This cost report will give you all the AWS usage
for each service category used by an account,
and it's IAM users in hourly or daily line items,
as well as any tags that you have activated
for cost allocation purposes.
And this report can be integrated,
and analyzed using Athena, Redshift or QuickSight,
which is a dashboarding tool.
So this report looks like this.

It is going to be the most granular report you can get
with the most comprehensive data,
and it will describe everything for every cost,
when it was incurred, why it was incurred,
and the description of the associated cost.
So this is definitely a report you may want to use
to analyze your bill in details,
and really understand where a charge is coming from.
Next, we have Cost Explorer,
which is a more visual tool in which we visualize,
understand and manage your cost and usage over time.
You can create custom reports that will analyze the cost,
and usage data, but it is for a high level, okay?
You can get your total cost and usage data
across all accounts or monthly or hourly,
or at the resource level.
And with this Cost Explorer tool,
you will be able to access your optimal savings plan
to lower the prices on your bill.
And most importantly, you can forecast usage up
to 12 months based on the previous usage,
which is an exam question.
So if you're looking for a tool that will allow you
to forecast your bill for 12 months ahead,
then it's going to be Cost Explorer.
So here is an example, with Cost Explorer,
you can get the monthly cost by AWS service,
and as you can see here, you will get these graphs based
on each different month.
And we'll give you, in this example, for EC2 instances,
which one are costing you money, as well as the types.
You can get hourly and resource level information.
So this is your cost every hour with, again,
the cost defined by EC2 instance,
so you can understand which EC2 instance
is costing you more every single hour.
And we can define savings plan,
which is again an alternative to using reserved instances.
And so you would get recommendations on savings plans
right away directly from Cost Explorer.
Finally, as I said, you can forecast usage on Cost Explorer.
So for example, you can have a look at your previous cost,
and then you can go up to 12 months ahead of time
to forecast your usage,
and see how much it's going to cost you
to use AWS in the future based on the growth
you have been experiencing in the past.
So under cost analysis,
we have two things you need to look at.
The first one is data exports.

So if you click on it,
that allows you to export your cost and usage reports.
So for this, you would just create an exporting dashboard.
You would say it's a standard data export,
then you specify an export name, you say whatever you want
to have included in your export, the time granularity,
the columns you want in that export.
So as you can see, it's a lot of columns,
and a lot of information you can get in the report.
And then if you want the table
to be delivered in specific formats, for example CSV,
or Parquet, if you want to export over the new file
over an existing file or not,
and then an S3 bucket to export to and you'd be good to go.
This would allow you to use your tool of choice
to analyze this data or even use Athena if you wanted to.
But if you wanted to just use the standard AWS tools
to analyze this data,
then you would look into Cost Explorer.
So here you can have a look at all your costs over time.
So here you have it based by month of January, February,
and then and so on, all until August,
and then for the other months had no cost,
so they're not showing here.
And then for each month, you can see the services
that were billed the most.
So for example, on this one,
Elastic Load Balancing was billed a Lot.
You can also have a table that you can download as a CSV
to get that breakdown,
and you can change the parameter of your report in here,
and you have a lot of parameters
to choose from as you can see.
And if you're happy with it, then you can save this
to the report library to have a quick access to it,
and access it on the left hand side.
So that's it for this lecture.
Now, we've seen data exports and Cost Explorer.
I hope you liked it and I will see you in the next lecture.

Now we are getting into
the monitoring aspect of our billing.
So we've seen about billing alarm and billing metric.
So we know that the billing metric is only stored
in the region us-east-1 in CloudWatch,
and this will be a nice graph that looks like this.
And then the data in the us-east-1
it is aggregated for all your regions in AWS.
It corresponds to the actual cost,
not for the projected cost.

And then you can create the billing alarm
on top of the billing metric.
It's a simple alarm.
It's not as powerful as budgets
as we'll see in the next slide,
but this will be helpful to send you an email notification,
for example, in case you go over a certain threshold.
And on the graph right here I set my threshold to $70.
And so as soon as I crossed the $70,
I received an email notification.
So we've seen billing alarms, so I won't go over it again.
Next we have budgets.
We haven't seen budgets yet.
So budgets are very helpful to send alarms
when costs exceed the budget,
or when a forecast also exceeds the budget.
So we have four types of budgets we can create,
usage, cost, reservation, and savings plan.
And for Reserved Instances
it can track the utilization of them.
It supports EC2 Reserved Instances,
ElastiCache Reserved Instances, RDS and Redshift.
And it supports up to five SNS notifications per budget.
So we can send emails, trigger another function, and so on.
And we can filter by service, linked account, tag,
purchase option, instance type, region, AZ,
API Operation, et cetera, et cetera.
You actually get the same operations
as in the cost explorer,
and their first two budgets are going to be free,
then you're going to pay $0.02 per day per budget.
So for the budget we can just in the console enter Budgets
and we are taken into the budget console.
So we can see we have one budget we have created
from before called Don't go over $10.
And we see we exceeded it.
This was the budget,
this is the amount used and the forecasted amount,
and the current versus budgeted amount.
So it's just because I've been using my account a lot,
so I went over it.
But we could go and create our own budget if we wanted to.
And we have two options.
Either we use a template and we have one of these options,
or we do and want to customize the budget,
and we have the four options
that I discussed with you before.
So cost budget, usage, savings plan,
and reservation budget.
So just to go again over the cost budgets,

if I click on it
I'm able to have a look at my cost data over time.
Then I can enter my budget name.
So, DemoBudget.
Then we can set the period.
Is it a monthly budget, a daily, quarterly, or annually?
Is it recurring or expiring?
And when does it start and when does it end?
So it's, say it's a fixed amount of $10 again.
And then we can filter the budget scope.
So we can have all AWS services if we wanted to,
or we can filter for specific AWS cost dimensions.
For example, we just want to have a look at a filter.
And the filter being that we want the service
to be only EC2, so elastic.
EC2-Other, for example.
Or if we go here, we can have a look at other things
such as, for example, the Key Management Service.
So you can apply this filter
and then see what is your actual budget
regarding these two services.
So we can definitely drill down into our budget
and make it the way we want to, which is super nice.
Then we can click on Next.
And then we have budget alerts.
So we can add an alert, a threshold, and, for example, say,
"Hey, at 80% of the actual budgeted amount,
then send me an email."
And we can set up emails right here.
Or we can say, for example, and I can say,
"stephane@example.com."
And also we can add another one.
For example, say, "At 80% of the forecasted amounts
right here on the right hand side,
then again, send me an email at stephane@example.com."
And the cool thing is that here on this graph
you can see where the thresholds lie
and then where we are at, and so on.
We can also, if you wanted to, drill down all these costs
in Cost Explorer because the two things are linked.
So that's it.
I won't go over creating that budget,
but we've seen all the options
and now we understand how AWS Budgets can be used.
So I hope you liked it,
and I will see you in the next lecture.

So here is a quick lecture
on the service named AWS Cost Anomaly Detection.
And the name is quite explicit this time.

So this continuously monitors your cost and usage data,
and it will use machine learning to detect unusual spends.
So it's going to learn from you.
It's going to learn your unique historical patterns,
and then it will detect one-time cost spikes
and/or continuous cost increases.
And you don't have to define anything,
you don't have to define thresholds.
It just knows on its own what looks a little bit weird.
So it's going to monitor for your AWS services,
your member accounts, your cost allocation tags,
and your cost categories.
And it will send you an anomaly detection report
with the root cause analysis
of what is happening in your account.
You can get notified by either using individual alerts
or a daily or weekly summary that leverages SNS.
So to summarize, using machine learning,
you can have a look at your cost,
you can get alerted and quickly analyze the root cause,
all of using the AWS cost anomaly detection service.
Okay, that's it.
I hope you liked it,
and I will see you in the next lecture.

So now let's talk
about the AWS Service Quotas.
So the idea is that with AWS, you have limits
and these limits are called quotas.
And sometimes you may hit them.
For example, you may get close
to how many Lambda functions you can run at the same time
and so on.
And you wanna be notified whenever you hit these limits
because they're important limits in your accounts.
For this, you can use the service quota service.
So you can create CloudWatch Alarms directly
on the Service Quotas console and get alerted.
For example, whenever your Lambda concurrent executions
are reaching your quota.
So on top of being alerted
of your limits and whenever you reach them,
you can also request a quota increase directly
from the AWS Service Quotas console
and then you would be able to extend your quotas
or you can decide if you wanted to to shut down resources
because you figure out maybe there's too many things running
and you're not supposed to have these things running.
So to summarize,
the Service Quotas service actually monitors all your quotas

across AWS, and that's a lot of quotas.
You can get alerting out of it from CloudWatch Alarms
and you can request quota increase directly
from the consult.
So that's it.
I hope you liked it, and I will see you in the next lecture.

So now, let's talk about AWS Trusted Advisor.
So you don't need to install anything.
It's a service that gives you a high level
account assessment on your account.
It's going to check for a few things
and advise you on them.
So the checks can be, for example,
do you have EBS Public Snapshots?
Or do you have RDS Public Snapshots?
Or are you using the root accounts for your accounts?
So all these things are checked by Trusted Advisor
and they are grouped in six categories.
We have cost optimization, performance,
security, fault tolerance,
service limits, and operational excellence.
So you have what's called the three sets of checks,
the core sets of checks,
and then you have the full set of checks.
And to have access to the full set of checks,
you need to have a business
or an enterprise support plan.
On top of it, if you do switch
on the business and enterprise support plan,
then you get programmatic access
to Trusted Advisor through the AWS Support API.
So I think it's best for you to see what Trusted
Advisor is made of to really understand it.
So here, I am in Trusted Advisor.
And as you can see, you have recommendations.
So zero actions are recommended,
but two investigations are recommended for me,
and then there are some checks on excluded items or not.
But as we can see, we have two
on security that must be looked at.
So it turns out that one of my bucket
is actually allowing a global access.
So I need to verify it and make sure it's correct.
And as you can see, 29 of my 60 security group rules
allow unrestricted access to a specific port.
So again, this is something I should look at.
Maybe that's my intention, maybe that's a problem.
But you can see right away,
I get prompted to upgrade my support plan

to get all Trusted Advisor checks.
So let me show you what I mean.
On the left-hand side,
we have the Recommendation categories.
And if I click on Cost optimization,
as you can see, I get none of the checks available
because I need to update my support plan.
So all these things are actually not available for me
and I need to pay for the service
to actually make some cost optimizations.
Same for performance, I get access to nothing.
If I go to Fault tolerance,
again, I get access to nothing.
Operational excellence, again, access to nothing.
The only two things I have access to is security.
So here, we get some checks, the core checks.
And in here, I have my Bucket Permissions,
my Security Group ports, my EBS Public Snapshot,
RDS Public Snapshot, and so on.
But as soon as I go in here
to the more advanced security checks,
then, again, I need to update my support plan.
Finally, you can have a look
at Service limits directly in Trusted Advisor.
That's one way of doing it.
So you can have a look at your Auto Scaling Groups,
your CloudFormation Stacks, your DynamoDB Read
and Write Capacity and so on.
So Trusted Advisor is not a very interesting service
to look at when you don't pay for the support plan,
but at least this should give you an idea
of how Trusted Advisor is used in AWS,
and therefore answer your exam questions on it.
All right, so that's it.
I hope you liked it
and I will see you in the next lecture.

Now let's learn
about the AWS Support Plan Pricing.
So when we started our accounts,
we were under the basic supports, which is a free.
But we have four different plans we can choose from,
and based on how much we spend
we qualify for different plans.
And so at the exam you're expected
to know the differences between this plan
and choose the right one
based on the use case that is proposed to you.
The basic support plan is free
and you get customer service and communities,

so you have 24 access to the customer service,
the documentation, the white papers, and the support forums.
For trusted advisor we only get access
to the seven core checks and this was not gonna be a lot,
but this is what we get from the basic plan.
As well as the personal health dashboard,
which we saw before which gives us a personalized view
of the health of our services and give us alerts
when our resources are going to be impacted.
So this is just basic, but say you wanted to upgrade,
you could upgrade to the next plan
which is called the developer plan.
So you get everything from the basic plan obviously,
and then you get this hours email access
to Cloud Support Associates,
so now you can start emailing AWS
through opening support tickets in the console.
You can get an unlimited amount of cases
and you can have one primary contact on your accounts.
Based on the case severity,
you get different response times,
so if you're looking for general guidance,
hopefully it will give you a response
within 24 business hours.
If the system is impaired,
something is not working in your accounts,
they hope to give you a response within 12 business hours.
So this is good when you're developing on AWS.
But as soon as you plan to have production workloads,
so workloads that are going to be supporting
your main business,
then you need to sure to have a business support plan.
So the business support plan
will give you the full sets of checks for trusted advisor,
as well as API access.
You will have twenty four seven phone,
email, and chat access to Cloud support engineers
so that you're definitely upgraded here.
And you have unlimited cases and unlimited contacts.
You will also get access to infrastructure event management
for an additional fee.
And now for the case severities
you have the general guidance, the system impaired,
but you also get the production system impaired
which is going to give you a response time
of less than four hours or a production system down
so when this is not working at all
with a support time of less than one hour.
So definitely if you plan on getting some phone support
or you need to have some production support

when it is impaired or down,
you will have to use the business support plan.
So next we have the AWS Enterprise On-Ramp Support Plan.
So it's 24/7 and it's intended whenever we have production
or business critical workloads.
So you get everything from the business plan,
plus access to a pool of technical account managers,
so you have multiple ones from a pool.
A concierge support team,
which is going to give you billing
and account best practices
as well as some operations reviews.
So, you're going to get infrastructure event management
well-architected and operations review given to you.
In terms of severity and response times,
you're going to get the same as before
but for a production system that is impaired
it's less than four hours.
And for a production system done is less than one hour,
and for a business critical system down
is less than 30 minutes.
Which is different from the enterprise support plan,
which is not on ramp, okay?
So this one is for mission critical workloads.
You get everything from before
plus a designated technical account manager.
So this time, the time is designated for your accounts.
You still get at the support team,
you still get the reviews.
And for the response times
it's four hours for impaired, one hour for down,
but most importantly if it's a business critical system down
it is not 30 minutes anymore, it is less than 15 minutes,
which is why you would pay extra
for the enterprise support plan.
Okay, so that's it for this lecture, I hope you liked it,
and I will see you in the next lecture.

Now let's do a summary
of the accounts best practices.
So first, if you want to operate multiple accounts,
it is recommended you use AWS organizations.
And if you want to restrict
which account can do what, their power,
then you should use SCP for service control policies.
It's very easy
for you to set up multiple accounts
with best security practices with AWS Control Tower,
which is sitting on top of organizations.
You should use tags and cost allocation tags

for easy management and billing in your accounts.
And you need to remember the IAM guidelines,
such as enabling multifactor authentication, MFA,
the lease privilege, creating a password policy
and enabling password rotation.
Use AWS Config to record all resource configurations
and compliance over time
in case something goes wrong.
CloudFormation is extremely helpful to deploy stacks
across multiple accounts and regions,
and for you to really manage all these accounts all at once.
Trusted Advisor is great to get insights
and to find the right support plan adapted to your needs.
Send your service logs and access logs
through Amazon S3 or CloudWatch logs,
maybe even in a separate account for logging
just as you adhere to the best security practices.
Use CloudTrail to record API calls made within your accounts
or in different regions and so on.
And then if your account happens to be compromised,
then please change the root password,
delete and all passwords and keys,
and contact the AWS Support.
Finally, you can use AWS Service Catalog
to allow users to create predefined stacks
that are defined by administrators in the first place.
So that's it for this lecture.
I hope you liked it,
and I will see you in the next lecture.

So here is a quick summary
on billing and custom tools in AWS.
So the first one is Compute Optimizer,
which allows you to get recommendation
on resource configurations in order to reduce cost.
The second one is the Pricing Calculator,
which is allowing you to estimate the cost
of services on AWS.
The Billing Dashboard gives you a high-level overview,
as well as a free tier dashboard to know where you stand
on your free tier usage.
Your Cost Allocation Tags allow you to tag resources
to create detailed reports with your own tags as a filter.
Cost and Usage Reports give you
the most comprehensive billing datasets.
Cost Explorer allows you to get your current usage
in a detailed manner as well as forecast usage,
months in advance.
And Billing Alarms in us-east-1, they live,
they allow you to track your overall

and per-service billing.
But if you want to get more advanced, you can use budgets,
which allows you to track usage, costs, reserve instances
and get alerts in real time.
We get savings plan and they're an easy way
for you to save on your bill based on long-term usage
of AWS by committing a specific dollar amount.
Cost Anomaly Detection helps you detect unusual spends
using machine learning
and Service Quotas helps you get notified
whenever you're reaching a service limit threshold
and increase these limits directly from this console.
Okay, this is it for this lecture.
I hope you liked it and I will see you in the next lecture.

#_____Section 19_____----
So there is a service we've been using
behind the scenes without really knowing it.
And it's at the center of AWS.
It's called AWS STS for security token service.
This enables you
to create temporary, limited privileges credentials
to access your AWS resources.
That means that you get short-term credentials,
just like your access key and your secret access key,
but you can configure your expiration period.
So the user, for example, has access to a role
and then wants to leverage that role.
So what it's going to do is that it's going
to assume the role using an STS API call.
It will talk to the STS service,
and the result of this API call will be STS sending us back
some temporary security credentials,
and they look just like an access key, security key
as well as a session key,
which is going to be limited in time.
And using these three credentials,
we're going to access our AWS resources
and using the (mumbles) role we just assumed.
So the idea is that the use cases can be multiple.
Number one they would be identity federation.
For example, to manage their identities in external systems
and provide them with STS tokens to access AWS resources
or for IAM roles, access for cross or same account access,
which is the exact example I showed you

on the right hand side.
Or finally, when we used it behind the scenes
in the course is that when we provide an IAM role
to our EC2 instances.
Behind the scene there's a script
that actually refreshes the EC2 credentials,
using the security token service in the backhand.
So that's it.
Anytime from an example perspective that you see,
you need to create temporary, limited privileges credentials
to AWS (mumbles) STS.
That's it.
I will see you in the next lecture.

Now let's learn about Amazon Cognito.
So Cognito is a way for you to provide identity
for your web and mobile applications users.
So potentially you have millions of these users
that are using, for example, your newest web application
or your newest mobile application
but you shouldn't create IAM users for them, okay?
IAM users are only for the people who belong
to your company and need to use AWS directly.
Instead, what you can do is to create users
for your mobile and web apps in Cognito.
So this is a simplified version,
but you would have Amazon Cognito
which will have its own internal database of users,
potentially millions of users,
and then your mobile and web applications
will have an integrated login into Amazon Cognito.
Now Cognito is what you would maybe see behind some
of these websites where you can also login
with Facebook, Google, and Twitter
by saying by click on the button
and then you're redirected to Google or Facebook
so it does that as well.
So if you're thinking of building a web
or mobile application and you want to have
a way to mange users on AWS
then Cognito would be the way to go.
So that's it, I hope you liked it
and I will see in the next lecture.

So now let's talk
about Microsoft Active Directory
and this is not AWS just yet, but I will get to it.
So this is a simplified version

but Microsoft Active Directory or AD is found
on any windows server that has AD domain services.
And it's going to be a database of objects,
and these objects can be user accounts, computers, printers
file shares, security groups,
and with Microsoft Directory,
you have Centralized Security Management,
you can create your accounts,
you can assign permissions all within it.
So when would you have used Active Directory before?
Well, say for example, that you had a Windows laptop,
and you were connected to your corporate internet,
then you would have a Domain Controller
which defined that you, John, have a password,
maybe the password is password.
And so that means that if you log in
onto any of the machines, belonging to your company,
because they are connected to the Domain Controller,
you can use this combination of username
and password on password of any of these machines
allowing you to seamlessly use your log in
on too many different services or laptops and so on.
So this is a very, very high level
of what Active Directory is.
It's a way for you to manage users,
computers, printers, and so on,
usually within on-premises system.
Now, AWS is different
because it doesn't have active directory on it,
but you can extend Active Directory
using AWS directory services.
So you have three files on it
and to be honest, you don't really need know them,
but I just wanna give you an overview again
that what is possible on active on directory services.
So the first one is, AWS Manage Microsoft AD,
which is to create your own Active Directory in AWS,
where you can manage users locally,
and you have support for multifactor authentication.
If you have a on-premises Active Directory already
you can establish a trust between the two
and they will be joined together,
and they will be basically trusting each other.
So this is the first way to do Active Directory on AWS.
Then you have the AD connector, and this is a proxy,
that is going to redirect request originating
from AWS, to your on-premises AD,
and it supports multifactor authentication

but the users themself, they live on the on-premises AD.
So that means that you authorize into the AD connector
which is going to proxy request into your on-prem AD.
And finally, you have simple AD,
which is not Microsoft Active Directory,
but it's an Active Directory
compatible managed directory on AWS,
and it cannot be joined with an on-premise AD.
It would just be a simple standalone
Active Directory in the Cloud.
Now you don't need to know all these staff
going into the exam, this is more something
for the Certified Solutions Architect Associates,
or professional, but from a exams stand point
at the Cloud Practitioner Exam,
all you need to is know that Directory Services
is used whenever you hear about Active Directory
or Microsoft Active Directory.
So that's it, I hope you liked it.
And I will see you in the next lecture.

So now let's talk
about the AWS IAM Identity Center
which is a successor
to the surveys called the AWS Single Sign-On.
So if you see one or the other at the exam,
just know that the feature
is to give you a single sign-on.
So one login for all your AWS accounts
in your organization which is mostly
where the exam is testing you on.
But also, you're going to have one login across
Business cloud applications,
SAML2.0 enabled applications,
and your EC2 Windows Instances.
So the user logs in and has access
to everything you define for that user have access to.
And in terms of the identity providers,
So where the user data is stored,
where you can have a built-in identity store
in the IAM Identity Center,
or you can connect to a third party identity store
such as Microsoft Active Directory,
or OneLogin or Okta.
The idea here is that from an exam perspective,
anytime you see one access to multiple AWS accounts,
you have to think about the IAM Identity Center.
So very simply, what does it look like?

Well, you log in through this one URL,
and then you provide your username and password.
Then, you have access
to the AWS IAM Identity Center portal,
which is, I'm showing you mine right now,
have access to four accounts under my organization.
And then, I can click on one of them
and click on management console
and I'm going to have direct access
to the management console of a specific account.
All of this just by remembering one login
versus remembering four logins
for all of my accounts,
and managing my users across my accounts
in a central manner instead
of managing it on a per account basis.
So it's very powerful,
and that's all you need to know for the exam.
I hope you liked it,
and I will see you in the next lecture.

So now let's summarize
Advanced Identity in AWS.
So first of all, the basics.
So we've seen IAM.
IAM is for doing identity and access management
inside of your AWS accounts,
and this is where you can create users that you trust
and belong to your company.
For AWS Organization,
even though we saw it in the previous section,
it can also be used in this section
because thanks to Organization
you can manage multiple accounts.
Then we have the STS service, the Security Token Service,
which is a way to issue temporary
and limited-privileged credentials to access AWS resources.
We have seen Amazon Cognito to create a database of users
for your mobile and web applications.
We have seen Directory Services
to integrate Microsoft Active Directory in AWS.
And we have seen IAM Identity Center,
which is one login for multiple accounts and applications
so that you can seamlessly navigate
between your different accounts.
Okay, so that's it.
I hope you liked it and I will see you in the next lecture.

_____Section 20 ____

So welcome to this section,
which is other AWS list services.
So the idea in this section is that I'm going to give you
an overview of other services that I couldn't group
with the other ones, and they're services
that students did report to me as sometimes
but rarely appearing on your AWS exam.
So I wanna keep this section short.
So the lectures are going to be very short
and brief and most likely without any hands-on, okay?
And there's going to be no summary lecture
at the end of this section, so I can keep things flexible
and keep on adding services to the section
in case I hear feedback from students.
It's going to be very easy.
It's one service and one definition, so don't stress
if you don't understand the service directly.
Just remember the definition,
and you should be all set for your exam.
So that it.
Hope you like this lecture,
and I will see you in the next lecture.

So let's talk about Amazon WorkSpaces.
So it's a managed desktop as a service,
so DaaS solution to easily provision
Windows or Linux desktop.
So the idea is that, it's just going to be a solution
that is going to allow you
to eliminate what's called on-premises VDI,
so, Virtual Desktop Infrastructure.
So the idea is that if someone,
someone wants you to have a windows laptop in the cloud,
they can do so using WorkSpaces.
It's fast and it quickly scales to thousands of users.
It's secured because it's integrated with KMS,
and it's pay as you go service,
so you only pay for the usage of these actual desktops.
So you're a user, you're maybe at home,
and you want to access a secure windows desktop
from within AWS,
you would provision one using workspaces,
and then you would get access to your cloud
or your corporate data center
directly from this virtual desktop
that is going to be secure and maybe within your VPC.

So that's the IDB one workspaces,
if you see anything related to virtual desktops
or manage desktop as a service, think workspaces.
Finally, there's been some questions in the exam about,
minimizing latencies for workspaces.
So if you have, for example,
two corporate offices in California and Paris
and uses there, the best practice is to deploy workspaces
as close as possible to your users,
so you would deploy maybe one workspace,
in the U.S.
And then your California user would access
that workspace directly for them there
and the other one would be obviously to place
another workspace closer to your european users in Europe.
And so the idea is here,
you'd have as many workspace regions as
you know, center locations you have for your company
to minimize latency.
And overall, if you wanna minimize latency,
always think about deploying close to users.
Okay so, not just workspaces, but any other app,
if you wanna minimize latency,
deploy it as maintained as possible across the world,
close to users.
So that's it for this lecture. I hope you liked it.
And I will see you in the next lecture.

Now let's talk about Amazon AppStream 2.0.
So it is a desktop application streaming service.
So it's very different from Workspaces,
and I'll give you the differences right after.
But so the idea is that you want
to stream an application to any computer
without acquiring and provisioning infrastructure.
And the application itself is going to be accessible
and delivered from within a web browser.
So here is a very concrete example.
Say you want to have the Blender application
to create the 3D models directly from within a web browser,
you can with Amazon AppStream.
So it is application-focused,
and they're delivered through your web browser.
But as you can see in the screen,
you could also use Eclipse, or Firefox,
or whatever, OpenOffice application directly
from within your web browser.
So how does that work versus WorkSpaces?
Well, WorkSpaces is to give you a full VDI
and a full Windows desktop or Linux desktop.

So you can launch as many applications as you want
within your desktop,
and you have to connect directly
using your remote desktop application, okay?
The WorkSpaces are on demand, or they're always on.
Whereas with AppStreams 2.0,
you're going to stream a very specific desktop application
directly into your web browser,
so there is no need to connect
to a virtual desktop in the cloud.
And it works for any device that will have a web browser,
so it has more compatibility.
And you're allowed to configure an instance type
per application. So for example, you were saying,
"Hey, for this application, like Photoshop,
I need more CPU, more RAM and more GPU."
So, hopefully, they're different to you in your mind.
So that's it, I will see you at the next lecture.

Next we have AWS IoT Core
and IoT stands for "Internet of Things."
It's a network of internet-connected devices
that are able to collect and transfer data.
And so IoT Core allows you to easily connect IoT devices
into the AWS Cloud.
This is as much as new to remember.
So what is an IoT device?
Well, it could be for example, a connected car.
It could be a connected light.
It could be a connected fridge,
a connected whatever you want, really.
And with IoT core, you're able to connect these
for example connected cars into the cloud.
So with IoT core, these devices can exchange securely
and scalably billions
within billions of devices, trillions of messages.
So IoT Core can act as a pub-sub to allow these devices
to exchange messages and to communicate.
And your applications can even communicate
with your devices even when they're not connected.
There's also integration between IoT Core and AWS
as you would expect.
So you can use Lambda, Amazon is free,
SageMaker behind the scenes to really add value
to your IoT devices. And the whole suite of projects
within the IoT Core allows you to gather, process,
analyze and act on data that are created
by your IoT devices and therefore build IoT applications.
So that's it. I hope you liked it
and I will see you in the next lecture.

Now let's talk about AWS AppSync.
So the idea with AppSync is to build a backend
for your mobile and web application.
So to store and synchronize data in real time.
For this AppSync actually leverages a technology
from Facebook called GraphQL,
and usually the exam, if you see GraphQL,
it will hint heavily for you to use AWS AppSync.
So thanks to GraphQL,
the client code for your APIs
can be generated automatically.
And on top of it, to build this GraphQL backend,
thanks to AppSync,
you will have integration with DynamoDB and Lambda.
It also gives you, thanks to GraphQL again,
real time subscription
so you can get updates of data for your web
and mobile applications in real time
and also offline data synchronization if you ever need.
There's also built in security.
And then we'll see in the next lecture
that there is a framework called AWS Amplify
that can leverage AppSync in the background
if you wanted to build a GraphQL backend
with Amplify as well.
So that's it for this lecture just remember,
AppSync (indistinct) storing data
for your mobile and web applications using GraphQL.
I hope you liked it, and I will see you in the next lecture.

So now let's talk about AWS Amplify.
It's a set of tools and services that helps you develop
and deploy scalable full stack web and mobile applications.
So the idea is that thanks to Amplify,
you're going to get a comprehensive suite
to manage all you need to do
to have your web and mobile applications.
So this means that through Amplify,
you can manage authentication, storage,
API, whether it be a REST API or a GraphQL API,
CI/CD, PubSub, analytics,
even machine learning, monitoring,
and you can get your source code from AWS,
from GitHub, or other places.
So see it as like the Elastic Beanstalk
for web and mobile applications.
So the idea is that when you go to Amplify,
you can go into the Amplify Studio,
and within the Studio you can set up everything you need,
for example, your data, your authentication, your storage,

your functions, your graphical API, and so on.
And then Amplify behind the scenes
is going to configure an Amplify backend.
And that backend will be leveraging existing services of AWS
that you know, for example, Amazon S3, Amazon Cognito,
AWS AppSync for your GraphQL backend, for example,
API Gateway for your REST backend,
SageMaker, Lex, Lambda, DynamoDB,
and of course more services.
So Amplify is a really good wrapper
around all these services, again, the target of which
is to help you build mobile applications backend.
I hope you liked it, and I will see you in the next lecture.

So now let's talk
about AWS Application Composer.
So it's a way for you to visually design
and build serverless applications quickly on AWS.
So we'll have a look at the console in a second.
As you can see, you have an interface
that allows you to drag and drop
and to build your template visually.
So the idea is that without being an expert in AWS,
you can quickly create infrastructure as code.
Now, you also configure how your resources
interact with each other by connecting them,
and eventually the output of this is an IaC.
So Infrastructure as Code template
that you get from CloudFormation.
You also have the ability to instead import a CloudFormation
or a SAM template in order to visualize them.
So the best way to show you
what is Application Composer is actually to go
into the console and just demo that to you right now.
So here I am in the Application Composer console.
And what I'm going to do is just to open a demo
to show you what a canvas may look like.
So you can start a tour of the composer if you wanted to.
But let's go with manually.
So as you can see, this is my composer
in which we have an API gateway right here
with several routes.
We have some Lambda functions that are linked
to each route of this API gateway.
And finally, at DynamoDB table to store items
and all these things have been connected
through Application Composer.
So very easily what you can do is that you can take,
for example here, a Kinesis stream,
you click and drag and drop,

and then a Lambda function.
And then you connect these two things together.
And there you go.
The Lambda function is now reading
from a Kinesis data stream.
So whenever you click on a little card here,
you get the configuration.
So here we have this stream logical id,
and here we have some Lambda function configuration
that we can do more specifically.
So you can go very detailed
and of course edit any single one of them.
For example, for DynamoDB as well,
you can just click on details.
And there you go.
You can define whatever you want.
So this is very handy.
But on top of it, on the left hand side
we have the enhanced components, which are very specific
to serverless environments,
but also you can set up any kind
of IaC resources from CloudFormation.
So for example,
if you wanted to define a AppMesh VirtualNode, here we go.
You just drag and drop it and then click on details
and actually update the configuration right here.
So once we have this, once we have all these things,
we can delete the ones we have.
Once you have all these things,
you can actually go into the templates.
And here you have the equivalent CloudFormation templates
from the canvas where you have designed everything.
So it's very handy because this is pure CloudFormation code
that you can use in your code and just get started.
So it's a very nice way
to visually build a CloudFormation template
and serverless applications.
All right, so that's it for this lecture.
I hope you liked it
and I will see you in the next lecture.

Now let's talk about AWS Device Farm,
which is a fully managed service that will test your web
and mobile applications against actual real
desktop browsers, real mobile devices and tablets.
And you can run these tests concurrently
to on multiple device which will speed up
the execution of all your tests.
And on top of it you can configure these devices
just like you want for example, the GPS

the language setting, the WiFi, the Bluetooth, et cetera.
So Device Farm is a real farm of devices
in the aid of its clouds.
So we have desktop browsers,
we have mobile devices, we have tablets.
All of these kinds of things are actual devices.
And us as a user of the Device Farm service
we can for example, test our application.
So think of it as like you're an Android programmer
or an iOS programmer
and you want you to make sure your app is working across
maybe tons of device, thousands of devices
and they're all different.
They'll have different screen size, et cetera, et cetera.
You can test them on Device Farm
and then you can catch these bugs very early.
You can even interact with these devices
and the Device Farm service will send you reports,
lugs and screenshots so you can deal with bugs in advance
based on the different devices that you run
your application onto
which can be extremely helpful.
So that's it for this service.
I hope you liked it.
And I will see you in the next lecture.

Now, let's talk about AWS Backup,
which as the name indicates,
a fully-managed service to centrally manage
and automate backups across AWS services.
It gets on-demand and schedule backups,
and it supports point-in-time recovery, so PITR.
You can define the retention periods of your backup,
the lifecycle management, the backup policies,
et cetera, et cetera.
You can do cross-region backups
or even cross-account backup
that is backed by either of AWS Organizations.
So how does that work?
Well, when you have the backup service,
you're going to create a backup plan,
for example, the frequency,
as well as the retention policy of your backups.
You assign the resources to be backed up
by the backup service or it can be Amazon EC2, EBS,
DynamoDB, RDS, EFS, Aurora, FSx and Storage Gateway.
And then all these services, thanks to this backup plan,
will be automatically backed up into Amazon S3.
So fairly simple.
Anytime you see backups at the exam, think AWS Backup.

This is a nicely-named service.
I will see you in the next lecture.

[Speaker Not Visually Identified]Okay.
So, now let us discuss, Disaster Recovery Strategies.
In the exam, we'll just ask you.
Hey, which one is the cheapest.
So the cheapest is backup and restore,
but I want to give you more information, obviously.
So backup and restore is when your data is backed up into
the cloud. And then in case of a disaster,
you're going to restore it somewhere else.
And you can get back your application this way.
So, because the data is just being backed up
and your application is not running.
It will be only running when you restore it into
whatever place you want.
Then the cost is very, very minimum.
Because you're just backing up data over time.
This is it why the backup and restore
disaster recovery strategy is the cheapest.
Next we have pilot lights.
Pilot Light is saying that, hey,
we have the cloud and we're going to run the core functions
of the app. For example, just the database.
We have just a database in the cloud and it's ready to
scale, but it has a minimal setup.
So, the database is here, but it's not fully scaled.
And we don't have our application servers.
We just have a pilot lights.
And this is just, you say
with the minimal critical functions of the app are
up in the cloud, at a minimal setup.
So it's a little bit more expensive than backup and restore.
And if you needed to do a disaster recovery strategy,
then you need to obviously upgraded database type,
and start your application servers.
But at least you core functions of the app
is ready to be here.
Next we have warm standby.
And warm standby is going to be more expensive.
So here we have the full version of the app ready in the
cloud, but at minimum size.
And so if we need to do a disaster recovery strategy,
we just need to increase the size of the app
and we're good to go.
As we see, the cost is a bit higher for this.
And then finally we have multi-site hot sites,
and this is where we have the full version of the app
at full size, ready to be used no matter what.

And so this has the maximum amount of costs because well,
you have a department that's ready to use in case of
disaster is right away.
So this is what you need to know.
And again, just remember the which one's the cheapest,
which one is the most expensive and why,
but again,
at a high level, the examiner will just say,
hey, what's the cheapest one.
And it's going to be backup and restore obviously.
Okay.
So in the cloud, we can also do this to recovery,
It is multi-region.
So say we have our instance in us-east-one,
and there's a disaster that strikes us-east-one.
Then we can fill over all our traffic into another region.
For example, eu-west-two using route 53.
This could be a very possible setup.
So let's say for this to recovery,
definitely more in depth to know for
the disaster recovery strategies course,
but for the CCP level,
you have all you need.
That's it.
I will see you in the next lecture.

Hi, and welcome to AWS
Elastic Disaster Recovery, or DRS.
This service used to be named Cloud Endure Disaster Recovery
because it was acquired by AWS
but then it's been renamed as a proper AWS service.
So elastic disaster recovery, DRS, allows you to quickly
and easily recover your physical, virtual,
and cloud based servers
into AWS, to do, well, disaster recovery.
So for example,
say you wanted to protect your most critical databases.
That includes Oracle, MySQL, SQL server,
your enterprise apps, such as SAP
or you want to protect your data
in case someone's performs an attack
and tries to get a ransom out of it.
Well, for this
you're going to do continuous block level replication
of your servers from your corporate data center
into the cloud, using the elastic disaster recovery service.
So your operating systems, your apps, your database,
all of them write to disk and what's going to happen is
that thanks to an AWS replication agent that is present
on your corporate data center

then you're going to get continuous replication
of these disks into a staging environment in AWS
with low cost in EC2 instances and EBS volumes.
And then in case a disaster strikes your data center
and you need to perform digital recovery,
then you can fell over within minutes from
staging to production by creating bigger
EC2 instances, better EBS volumes,
and you have performed effectively your disaster recovery.
And then when your incorporated sensor
or anywhere else really is back online
you can perform what's called a failback
which is that the system falls back
into your incorporated sensor and you're operating normally.
Okay?
So that's it, elastic disaster recovery is a service
that allows you to do better disaster recovery.
I hope you liked it.
And I will see you in the next lecture.

Okay.
So now let's discuss that Data sync very quickly, so data
sync allows you to move large amount of data's from
on-premises to AWS.
And you can synchronize the data into Amazon S3, Amazon EFS,
or Amazon FSX for windows.
Now there's some replication tasks and that can be scheduled
regularly, for example,
every hour or every day or every week.
And what's going to happen is that after the first full load
of data from on-premises to AWS,
all the other tasks are going to be incremental.
So look for that word incremental at the exam.
This is going to be a hint.
That data sync is going to be the right insert.
So if we have a look at a diagram,
we have data sync running in AWS,
and we want to migrate or replicate some data from an on
premises server.
So we're going to run the data sync agent on premises,
and then the data sync agent is going to connect to
the data sync servers and send the data directly
into Amazon is free on any storage class
or EFS or FSX for windows file server.
That's it? This is as simple as it is,
and then made sure to keep it very, very simple for you.
So hope you liked it.
And I will see you in the next lecture.

Okay, so now let's discuss
cloud migration strategies, and there is a blog on the AWS
that describe the seven Rs of cloud migration.
Now there's a very complex diagram.
We're going to go over it in time, not the diagram itself,
but the concepts behind the diagram.
What I would recommend you to do is
to actually read the blog if you can,
after this lecture, just to get a better understanding
of cloud migration,
if that's something you're interested into.
But let's go over the seven Rs
because the exam may ask you which strategy you use.
So you want to migrate to the cloud,
and the first strategy is to actually retire.
What does that mean?
That means you turn off things you don't need.
So for example, you have some services
and you actually don't need to migrate them,
so you turn them off.
It helps to also reduce the service areas for attacks
because you have more security, because you have
less services active.
You save costs because you retire services, maybe 10 to 20%,
and you focus your attention on resources
that must be maintained.
The second option is to retain.
So that means that you actually don't do a cloud migration,
which is still a decision in your cloud migration strategy.
And you may want to just retain the resource on premises,
for example, because of security, data compliance,
performance, and unresolved dependencies,
or because maybe there is no business value to migrate
or it's too complicated.
Next, we have relocate.
So relocate is actually to move your app from on-premises
to its cloud version,
or move EC2 instances to a different VPC,
a different account, or a different AWS region.
So for example, you have servers
that you manage on-premises using VMware software-defined
data centers, so SSDC, and you just want to keep the same
but use now VMware cloud on AWS,
you just relocate, but nothing has changed.
Then there is rehost for lift and shift.
So these are very simple migrations,
and what you do is that you rehost your application on AWS.
So that can be your applications,

your databases, and your data.
You're going to migrate your machines.
That could be physical, virtual, or servers
or another cloud into AWS cloud.
So here you don't do any kind of cloud optimization.
The application stays as is,
but you leverage the cloud resources.
And so possibly, by using the cloud resources,
you could save as much as 30% on cost.
And some services, for example, the
AWS application migration service,
allow you to just do that, the lift and shift.
That means you take your application
and you just put it on the cloud.
Next, we have replatform for lift and reshape.
So as an example, you want to migrate your database
into RDS, or you want to migrate your application
to Elastic Beanstalk.
So here you don't change the core architecture
of your application, but you're going
to leverage some cloud optimizations,
and you're going to save time and money
by moving to fully-managed service or a serverless service.
For example, for a database,
instead of just moving it from a server on premises to EC2,
you move it to RDS,
and RDS is going to give you a lot of benefits
around backups, around resiliency,
around high availability, and so on.
So your application hasn't changed,
but now you're using managed services.
Then we have repurchase for drop and shop.
So you move to a different product
while moving to the cloud.
For example, you want to move to a software
as a service platform, SaaS.
It's expensive in the short term, but very quick to deploy.
For example, you move your CRM
to the managed version on salesforce.com
or your HR to Workday or your CMS to Drupal.
So instead of having everything on-premises,
now you leverage SaaS platforms.
And finally, we have refactor and re-architect.
So here you want to move your application to the cloud,
but you are re-imagining how the application is going
to be architected using Cloud Native features.
So the reason you may want to do this is
because you actually want to use the cloud

to improve the scalability, the performance, security,
and agility of your application.
You may also want to break up your monolithic application
into microservices.
So this is where you're going to have the most amount
of effort, but usually the most amount of payoff
from a leveraging the cloud capabilities.
So for example, you move your application
to a serverless architecture,
or you're going to use Amazon S3
to store some data and so on.
So that's it.
I hope now the seven Rs in the cloud migration
strategy makes sense.
I hope you liked it, and I will see you in the next lecture.

So there are two use cases
when you move to the cloud.
For example, you wanna start fresh
and you wanna leverage the cloud directly.
In this case, you don't need to do a migration.
But if you have on-premises servers and data centers
and you want to migrate to the cloud,
then you need to plan your migration.
And a way to do this is to plan your migration
using the AWS Application Discovery Service.
So the idea is that you're going to scan your servers,
and going to gather information
about the server utilization data and dependency mapping
which are going to be important for your migrations
so we can understand how to migrate
and what to migrate first.
So there are two types of migration you can do.
One is called the Agentless Discovery using a Connector.
And this gives you information around your virtual machines,
your configuration, your performance history
such as a CPU, memory, and disk usage.
Or you can run an agent
to do an Application Discovery Agent,
and this gives you more updates
and more information from within your virtual machines.
For example, system configuration, performance,
processes are running, and the details
of all the network connections between your systems,
which is good to get your dependency mapping.
Now, all this results data can be viewed
within another service called the AWS Migration Hub.
So this Application Discovery Service
really helps you to map out what you need to move

and how they are interconnected.
But then you actually need to move.
And the simplest way to move from on-premises
to AWS is using the AWS Application Migration Service,
also called MGN.
So this used to be called CloudEndure Migration,
but now it's been replaced.
And the idea is that using
the AWS Application Migration Service, so MGN,
you can do rehosting, also called lift-and-shift solution,
in which you convert your physical, virtual,
or other service on the other clouds to run natively on AWS.
How does that work?
Well, say you have a corporate data center
with OS apps and databases, and they run on disks.
What's going to happen is that you're going to run
the Application Migration Service.
And then the replication agent
that you have to install on your data center
is going to perform a continuous replication of your disks
so that you have, for example, low-cost EC2 instances
and EBS volumes that get this replication of data.
Now, the day you're ready to perform a cut over,
you can actually move from staging to production,
and have a bigger EC2 instance of the size you want,
as well as EBS volumes that match the performance you need.
So the idea is that you replicate data,
and then at some point, you do a cut over.
And that is by far the simplest way of doing it.
So this supports a wide range of platforms,
operating systems, and databases.
And this gives you minimal downtime,
as well as reduced costs
because, well, you don't need to hire complex engineers
to do this.
This is done automatically by this service.
So that's it.
I hope you liked it.
And I will see you in the next lecture.

So now let's talk about
AWS Migration Evaluator,
which is a service to help you build a data-driven
business case for migrations to AWS.
So the idea is that you wanna migrate,
and first you need to get the clear
baseline of what your organization is running today,
so you know your workloads.
And so therefore, you install the Agentless Collector
to conduct a broad-based discovery

of all your infrastructure.
And then you're going to take a snapshot
of your on-premises footprint,
your server dependencies, and so on.
And you will analyze your current state,
you will define the target state in AWS,
and then develop a migration plan.
So to summarize, you can install a Collector,
and that will gather data for you.
Or you can use the data import feature,
and there is a tool, and there is a template.
And what you can do is that you can
shape the data you have into this template,
and these sources of data will allow you
to run the Migration Evaluator Service,
which is going to give you quick insights
to understand your customized cost insights,
and to make sure that your migration can be cost efficient
and also good for your business.
And therefore, you will have a good business case.
And if you need to, you can even get expert guidance
on your business case from AWS.
So that's it for this lecture, I hope you liked it.
And I will see you in the next lecture.

So now let's talk about AWS Migration Hub.
So this is a hub
so it's going to be a central location
where you can collect server
and applications inventory data for the assessment, planning
and tracking of migrations to AWS.
The idea is that everything is centralized
and this is going to help you accelerate your migration
into AWS and to automate the process of lift-and-shift.
There's also a sub feature of Migration Hub called
the AWS Migration Hub Orchestrator
where you can use pre-built templates to save time
and efforts migrating enterprise apps.
Such as SAP or Microsoft SQL Server and so on.
On top of it, Migration Hub is integrated
with services we've seen before such
as the Application Migration Service, MGN
or the Database Migration Service, DMS.
So, whenever the exam is asking you for a central location
to discover access, plan and track your migration
and modernization, think no more
than the Migration Hub where you can have a look
at all your servers and apps and centralize them.
Right size your workload and strategy recommendation.
Where you can orchestrate migrations between multiple tools.

And finally, incrementally refactor applications
to move them to AWS.
So that's it for this lecture.
I hope you liked it and I will see you in the next lecture.

Now let's talk
about AWS Fault Injection Simulator, FIS.
So this is a way for you to run fault injection experiments
on AWS workloads.
And this is based on something called Chaos Engineering,
which is that you want to stress an application
by creating really, really disruptive events.
For example, the CPU starts skyrocketing
or the memories go out of memory,
or the database is having a failure,
or all these kinds of things.
And you want to see how your entire application stack
reacts to these disasters.
And this is why it's called Chaos Engineering,
because you're actually creating chaos
within your infrastructure.
So why do we do this?
Well to make sure our application is really solid,
and this allows us to uncover hidden bugs
and performance bottlenecks.
And currently, FIS supports some services,
you don't need to know them all,
so here's a list and maybe there'll be others over time,
but EC2 for example, by terminating EC2 instances,
ECS by stopping ECS tasks, EKS, and so on
to stop a Kubernetes task and RDS to make failures
go alongside our database.
So how does that work?
Well, we have FIS and we're going to create experiments.
And so we can use pre-built templates
to generate these disruptions,
and then they're going to have disruptions on resources.
So really you have to choose what happens
to your EC2 instances, what happens to your ECS clusters,
your RDS database and so on.
And then these experiments will start.
So the resources will get disrupted,
and then you have to see how your application behaves.
So we can monitor it using CloudWatch
or EventBridge or X-Ray or whatever you want.
And then once you're done,
then you stop the experiment
and you have a look at your results.
Was there any performance issue?
Was there observability issues or resiliency issues?

And you can improve your application
to see where the bottlenecks are okay?
So FIS is definitely an advanced kind of monitoring
and advanced kind of debugging, but it's definitely helpful.
And I'm so glad that AWS is now having this
as a native feature from within AWS.
So that's it for FIS.
I hope you liked it.
And I will see you in the next lecture.

Now let's learn
about the AWS Step Functions.
So, Step Functions is a way for you
to build a serverless visual workflow
to perform orchestration,
and it's usually of your Lambda functions.
So you would design a graph,
and you would say at each step of the graph,
in case of success or failure, what goes on next?
And this is really cool,
because you can start really having some complex workflows
within AWS.
So Step Functions have some internal features,
such as sequencing, parallel functions, conditions,
timeouts, error handling, and so much.
And it doesn't just do Lambda functions.
It is actually integrate with EC2 instances, ECS tasks,
On-premises servers, the API Gateway,
the SQS queues, [00:00:47 NMADB],
and actually a lot of AWS services.
Also, it is possible for you
within the Step Function workflow
to implement a human approval feature.
For example, a workflow goes up to some points,
and you say, okay, a human will have a look at the results.
And if the human says "Yes," then please go on.
If "No," then fail.
This is something that's possible
thanks to the Step Functions.
So the use cases of Step Functions are manifold,
but, like, we can have, for example, the order fulfillment.
We can have data processing, web applications,
or any kind of workflow that is complex
to describe and that would require some sort
of graph for you to visualize it, okay?
So that's it for this little intro.
I hope you liked it.
And I will see you in the next lecture.

So here is an service

that I'm sure you won't forget once I tell you about it.
It's called the AWS Ground Station.
So the idea is that this is fully managed service
that lets you control satellites communication,
process data, and scale your satellites operations.
So this is not for everyone,
but you have satellites running around the Earth,
a lot of them, and you may want to get access
to their data for whatever reason.
So Ground Station provides you a global network
of satellite ground stations near AWS regions.
So that it's easier for you to get data
from your satellites onto your AWS cloud.
So how does that work?
Well, thanks to the Ground Station service,
you're going to be allowed to download the satellite data
to your AWS VPC within seconds.
So the Ground Station runs within the cloud.
It's connected to satellites,
and then you download data from satellites
to Amazon S3 buckets, or your EC2 instance,
and from that, you can process it the way you want.
So this is something, believe me,
that was hard to do before the AWS Ground Station service,
but now, it is very easy to do.
The use cases for this,
if you have access to satellites of course,
is to do weather forecasting, surface imaging,
communications, or video broadcast.
I hope you liked this lecture,
and I will see you in the stars in the next lecture.

So now, let's talk about Amazon Pinpoint.
And Amazon Pinpoint is scalable two ways,
so inbound and outbound marketing communication service.
The idea is that you want to send email,
SMS, push notifications, voice,
and in-app messaging through Pinpoint.
And one of the main use cases is SMS.
So the customers will receive an SMS
that you send from Amazon Pinpoint.
And you have the ability to segment
and personalize the messages
with the right content to customers.
So you can create groups and segments and so on.
You also have the possibility to,
of course, receive replies.
And this scales to billions of messages per day.

So the use cases for Pinpoint is to run campaigns
by sending marketing email in bulk,
or sending transactional SMS messages.
And should someone reply or should it be successful,
then all the events such as text success,
text delivered replies, and so on,
will be delivered to Amazon SNS,
Kinesis Data Firehose, and CloudWatch Logs.
So you can build any kind of automation
easily on top of Amazon Pinpoint.
And so you may be wondering,
well, what is the difference between Pinpoint
and Amazon SNS or Amazon SES?
Because there is obviously some overlap.
So the idea is that in SNS or SES,
you have to manage each message's audience,
content, and delivery schedule.
And this is the responsibility of your application.
And that may be a lot of work,
and that may be not very scalable.
So in Amazon Pinpoint, instead, you're going to create
message templates, delivery schedules,
highly targeted segments, and full campaigns.
And all of this is managed by the Pinpoint service.
So see Pinpoint as the next evolution of SNS and SES,
in case you wanna do full-blown
marketing communications service.
So that's it for this lecture.
I hope you liked it, and I will see you in the next lecture.

#_____Section 21_____
Into this section on architecting on AWS
and the AWS ecosystem.
So first we're going to talk about
the well architected framework
and the general guiding principles.
So in the cloud, when you want to have a good architecture
you need to stop guessing your capacity needs.
Instead, you should use auto scaling
and scale the based on what the actual demand
on your system is going to be.
You should also test your system at production scale.
In the cloud
you can create as many resources as you want
very, very quickly.

And as such there is no reason why,
for example, for one hour
you couldn't test a system at production scale.
This allows you in the future to make sure that your system
is ready when you actually publish it to your customers.
Now, it's also very important
to automate to make architectural experimentation easier.
So for this CloudFormation is very important
because if you get infrastructure as code
then you can easily create an architecture
on different accounts and different regions.
Also using platform as a service,
such as Beanstalk could be very helpful
to experiment quickly.
You should also allow to make your architecture evolve.
So you need to design based on changing requirements.
That means that when you migrate a workload
from on premises to the cloud,
maybe at first you make it match one to one,
but then you try to rethink
how can I leverage the cloud better,
making it serverless for example.
Then drive architectures using data.
It's not good to guess.
It's good to look at what the actual usage is.
What are the patterns?
What are the queries?
And then drive your architecture
and use the right services based on what you actually need
based on this data.
Also improve through game days.
So that means that you simulate applications
for flash sale days, and this will stress your system.
And by stressing your system
you will know if you are doing well.
So for example, I'll give you an example,
Netflix they have something called the chaos monkey
and it's a program that's in their EC2 environment
goes ahead and terminates EC2 instance at random
in production.
And they see thanks to this chaos monkey
if they are ready for failures
if they're ready for big spikes,
if they're ready for errors.
So it's super important
to think that now you are in the cloud,
you have new capabilities.
So you need to start you need to start thinking

like you are using the cloud.
So the design principles
when using the cloud and the best practices.
The first one is to be scalable.
So we've seen the vertical scalability
and the horizontal scalability
also because we can easily create and remove servers.
We have disposable resources.
So your servers should be easily disposable
and easily configured.
If you put too much configuration onto a server
and somehow that server, if lost
will be losing its resource as well
you will need to have maybe three days of work again
to reconfigure that server
if you haven't done things properly.
In the cloud everything is supposed to be disposable
including your infrastructure.
And so you need to make sure to back up your data
you need to make sure that you back up your configuration
and that you have ways to very, very quickly
reconfigure your entire architecture.
Automation.
So these are guided by the principles of serverless,
infrastructure as a service, auto scaling and so on.
Loose coupling, which is very important.
So when you have an application at first,
(mumbles) it's called a monolith.
That means it's an application that can do everything.
And over time, the more you add onto the application
the bigger it becomes.
And because it's a monolith
it could be difficult to maintain and difficult to scale.
Instead, you should break it down into smaller
loosely coupled components so that maybe they will be linked
through SNS or SQS or by other means.
And the idea is that a change or a failure in one component
should not cascade to the other components
because in the cloud, again
things can fail and you need to be prepared for failure.
Finally, think in services, not servers.
So you could just use EC2 but it's not recommended.
EC2 would be to translate whatever you have on premises
in the cloud but think in terms of services
what managed services can you use databases, serverless
et cetera, that can make your life a lot easier.
So this is where the well architect framework
which is made of six pillars comes in.

The first one is operational excellence.
The second one is security.
The third one, reliability,
the fourth one performance efficiency,
fifth one cost optimization and six one sustainability.
And the idea is that by acting with these six pillars
you are having good architecture on AWS.
So they're not something you want to balance
and compromise between them or trade offs,
they're actually a synergy.
So for example, when you prove your operational excellence
you're probably also improving your cost optimization.
So the next six lectures are all about
doing a deep dive into all of these
to understand what they mean in AWS.
I hope you're excited
and I will see you in the next lecture.

The first pillar
in the well architected framework
is operational excellence.
So this is the ability to run and monitor systems
to deliver business value and to continuously
improve supporting processes and procedures.
So what are the design principles of operational excellence?
The first one is to perform operations as code.
That means using infrastructure as code.
So in AWS, that is going to be,
for example, AWS CloudFormation.
Also make sure to make frequent small
and reversible changes so that in case
of any failure, you can reverse it.
If you start making huge changes every three months,
things are not going to go well.
Also, refine operations procedures frequently
and ensure that all your team members
are familiar with these new operations procedures.
Anticipate failure but in case you have failure,
you need to learn from all these failures.
So failing is learning and of course,
if you don't take the feedback from your failures,
then you're not getting any better.
Finally, use managed services to reduce operational burden
and implement observability for actionable insights.
That includes performance, reliability, and cost.
So now, I'd like to just transcribe it
in terms of AWS Services.
And the first one is going to be prepare so prepare
is going to be how do we basically prepare everything

to have operational excellence?
So you should use runbooks,
you should have good infrastructure standards,
you should do run deployments and like mock deployments.
And for this we could use AWS CloudFormation basically,
to prepare everything as a nice infrastructure as code.
And then Config would be really nice as well to be using.
We haven't seen Config in depth in this course,
but Config could be used to evaluate
the compliance of your CloudFormation templates.
Then operate, you need to operate
and basically, automate as much as you can.
You should release fast, you should avoid manual processes.
So again, CloudFormation would be
a great tool for this, Config as well.
CloudTrail to make sure to track all the API codes
that are done and make sure nothing is done on purpose
or nothing is changed manually.
CloudWatch to basically monitor the performance over time
of your stack and make sure that if you need to operate it,
you know what operations to do.
And finally, X-Ray, it's not something we've seen in depth
in this course, but X-Ray will be able
to trace HTTP requests,
and to make sure that they're working correctly.
And if they're not working correctly, it will point us
to where the incorrect stuff happened.
Finally, you need to evolve your infrastructure
over time, so how do we do this?
Well, again, CloudFormation, as you understand,
CloudFormation is a centerpiece
of this whole operational excellence pillar,
but also you need to be able to evolve over time.
So all the CI/CD tools basically CodeBuild,
CodeCommit, CodeDeploy, CodePipeline,
allow you to basically iterate quickly,
deploy quickly, deploy often, deploy small changes,
and all these things will contribute
to operational excellence.
Now remember just going into the exam, you don't need
to understand and remember any of that.
I'm just trying to educate you to all
the AWS Services and how they can help
with this operational excellence pillar.
This is just educative, okay?
Have a read through the the pillar, by the way,
on the white paper if you want to,
but here I'm just trying to give you a small run down.
You don't need to remember all these services
and there's not this service correspond exactly

to that operational pillar or whatever,
but it's just to give you ideas of how
all these service can be used as a synergy
to basically drive operational excellence.
All right, so that's it for this lecture.
I will see you in the next one.
Autoscroll

Security is the second pillar
of the Well-Architected Framework and I really like it.
So security we all know what it is,
but it includes the ability to protect information,
systems and assets while delivering business value
through risk assessment and mitigation strategies.
So usually people see security as like oh we have to do it
but in the end when your application is secure,
you're really minimizing a risk over time and you save cost
from disasters and you really don't want to have a risk,
or a security issue in your company.
Now how do we design strong security?
Well we need to have a strong identity foundation.
So we want to centralize how we manage user accounts.
We want to rely on least privilege and maybe IAM
is going to be one of these services to help us do that.
We want to enable traceability, that means we need to look
at all the logs, all the metrics and store them
and automatically respond and take action,
every time something looks really weird.
We need to apply security at all layers, okay?
You need to secure every single layer,
such as if one fail maybe the next one will take over.
So edge network, VPC, subnet, load balancer,
every institute instance you have, the OS, patching it,
the application, making sure it's up to date,
all these things.
You need to automate security best practices, okay?
Security is not something you do manually,
it's mostly done well, when it's automated.
You need to protect data in transit and at rest.

That means always enable encryption, always do SSL,
always use tokenization and do access control.
And you need to keep people away from data.
So why is someone requesting data?
Isn't that a risk when you allow someone to access data,
do they really need it or is there a way to automate
the need for that direct access
in that manual processing of data?
And then you need to prepare for security events.
So security events must happen some day
in every company, I think, and so run response simulations,
use tools to automate the speed of detection,
investigation and recovery, okay?
So in terms of AWS Services, what does that mean?
Well the first one is going to be identity
and access management.
So we all know what that means, that means IAM,
STS for generating temporary credentials,
maybe Multi-Factor Authentication token
and AWS Organization to manage
multiple address accounts centrally.
Then detective controls.
So how do we detect stuff goes wrong?
AWS Config for compliance, CloudTrail to look
at API calls that look suspicious.
CloudWatch to look at metrics and things
that may go completely out of norm.
And then we get infrastructure protection.
So how do we protect our own Cloud?
Well CloudFront is going to be a really great
first line of defense against DDoS attacks.
Amazon VPC to secure your network
and making sure you set the right ACLs.
And then Shield, we haven't seen this, but it's like
basically a way to protect your AWs account from DDoS.
WAF which is a Web Application Firewall
and Inspector to basically look at the security
of our institute instances.
Now we haven't see all these services, they are more
for the SysOps exam, but again I just wanna give you here
an overview of all the AWS Services
that can help you achieve full security.
For data protection, well we know there is KMS
to encrypt all the data at rest.
Then there is S3, which has a tons of encryption mechanism,
we have SSES3, we have SSEKMS,
SSEC or client setting encryption.
On top of it we get bucket policies and all that stuff.
And then every, you know, every managed service
has some kind of data protection.

So Load Balancer can enable to expose a HTPS end point.
EBS volumes can be encrypted at rest, RDS instances
also can be encrypted at rest and they have SSL capability.
So all these things are here to protect your data.
Incident response, what happens when there's a problem?
Well IAM is going to be your first good line of defense
if there is an account being compromised
and just delete that account or give it zero privilege.
CouldFormation will be great, for example
if someone deletes your entire infrastructure,
how do you get back into a running state?
Well CloudFormation is the answer.
Then, for example, how do we automate
all these incident response?
How do we automate the fact that if someone deletes
a resource, maybe we should alert, CloudWatch Events
could be a great way of doing that.
So that's it, again just to show you the synergy.
Here there is a lot of AWS Services
and this is just an example of how you could achieve
all these principles with the Services.
I hoped you liked it and I will see you in the next lecture.

The third pilar
of the Well-Architected Framework is reliability,
and so reliability is the ability of a system
to recover from infrastructure or service disruptions,
dynamically acquire computing resources to meet demand,
and mitigate disruptions such as misconfigurations
or transient network issues.
So it's about making sure your application
runs no matter what.
The design principles are simple.
We need to test recovery procedures
so you need to use automation to simulate different failures
or to recreate scenarios that led to failures before.
We need to automatically recover from failure.
That means that you need to anticipate
and remediate failures before they occur.
Then scale horizontally in case you need
to have increased system availability, or increased load.
And then stop guessing capacity.
So basically that means that if you think,
oh, I need four streams in this for my application,
that probably isn't going to work in the long term.
Use auto scaling wherever you can
to make sure you have the right capacity at any time.
And then in terms of automation,
you need to basically change everything through automation,
and this is to ensure that your application will be reliable

or you can roll back, or whatever.
In terms of AWS Services, what do we have?
Well the foundations of reliability is going to be IAM,
again making sure that no one has too many rights
to basically wreak havoc on your account.
Amazon VPC, this is a really strong
foundation for networking.
And Service Limits, making sure that you
do set appropriate service limits.
Not too high, and not too low,
just the right amount of service limits,
and you monitor them over time.
Such as if your application has been growing,
and growing, and growing,
and you're about to reach that service limit.
You don't want to get any service disruptions,
so you would contact AWS,
and increase that service limit over time.
Trusted Advisor is also great,
we'll see this in this section
about how we can basically look at these service limits,
or look at other things,
and get strong foundations over time.
Change management, so how do we manage change overall?
Well, Auto Scaling is a great way.
Basically if my application gets more popular over time
and I have set up auto scaling
then I don't need to change anything, which is great.
CloudWatch is a great way also of looking at your metrics.
For your databases for your application,
making sure everything looks reliable over time,
and if the CP utilization starts to ramp up
maybe do something about it.
CloudTrail in terms of are we secure enough
to track our API calls?
And Config, again.
Failure Management, so how do we manage failures?
Well, we'll see this for disaster recovery explanation
in this section, but you can use backups,
all along the way to basically make sure
that your application can be recovered
if something really really bad happens.
CloudFormation to recreate your whole infrastructure
at once, S3, for example, to backup all your data
or, you know, S3 Glacier if we're talking about
archives that you don't need to touch once in a while.
Finally maybe you want to use a reliable,
highly available global DNS system,
so Route 53 could be one of them.
And in case of any failures, maybe you want

to change Route 53 to just point to a new application stack
somewhere else and really make your your application
has some kind of disaster recovery mechanism.
Don't worry, we'll see disaster recovery
in this section as well,
and I'll try to make it as simple as possible.
So that's it, for this pillar, I hope you liked it,
and I will see you in the next lecture.

The fourth pillar in the Well-Architecture
Framework is the performance efficiency.
So what is it?
It includes the ability to use computing resources
efficiently to meet system requirements, and to maintain
that efficiency as demand changes and technologies evolve.
So it's all about adapting
and providing the best performance.
The design principles are simple.
First, you need to use advanced technologies, okay?
You need to democratize them, and basically,
as the services become available, maybe they can be helpful
for your product developments so track them.
You need to be able to go global in minutes
so if you need to deploy in multiple regions
it shouldn't last days, it should last minutes.
Maybe using cloud formation.
Use serverless infrastructure.
So that's the golden state.
So that means you don't manage any servers,
and everything scales for you, which is really awesome.
Experiment more often.
Maybe you have something working really well today,
but you think it won't scale to 10 times the load.
Experiment maybe try serverless architectures.
See if that works for you.
Basically, give it a go.
And mechanical sympathy.
So be aware of all the AWS services,
and that's really, really hard.
I mean, doing this course is a right way of doing it.
Reading some blogs is also the right way of doing it.
Even for me, it's really hard to track
of everything of AWS services,
but you still need to be on top of the game.
Because when new changes have to happen,
they can really change dramatically,
your solution architecture.
So for AWS services for performance efficiency,
well, selections, so Auto Scaling, Lamda, EBS, S3, RDS
so you have so many choices

of technology that scale a little different patterns.
So choose the right one for you.
Definitely, for example, Lamda needs to serverless,
Auto Scaling is going to be for more EC2,
EBS is when you know you need to have a disc,
but you can sort of manage performance
over time using GB2 or I01.
S3 if you want to scale globally,
RDS, maybe you wanna use it to provision a database
and maybe you want to migrate to Aurora.
So you have a lot of things to select from.
Review, so how do we review our performance?
Well, the cloud formation, making sure we get
exactly what we need before we create it.
And how do we keep updated
on all this performance improvements?
Then the AWS News Blog is also a really good one,
and that's a tricky one, yes, but do read the AWS News.
I do read them once in a while, once a week,
and really keep myself updated on what's coming up.
Monitoring.
So how do we know we are performing
really well and as expected?
Then CloudWatch with CloudWatch Alarms, CloudWatch metrics.
All these things CloudWatch dashboards can help you
understand better how things work.
AWS Lamda, as well.
Making sure that you don't throttle,
that your application Lambda function runs
in a minimal time, all that kind of stuff,
And tradeoffs.
So how do we make sure that we are doing
the right performance decision?
So RDS maybe versus Aurora.
ElastiCache if you want to improve real performance,
maybe using Snowball.
So Snowball, for example, will give us a lot of data
moving very fast, but it will take maybe a week
for the data to arrive.
So the tradeoff is, do we want the data right away
in the cloud and use all our network capacity
or do we wanna move that data through a track
and get this in a week from now.
So always Tradeoffs, right?
With ElastiCache, always a Tradeoff as well.
Do I want to have possibly outdated,
stale data in a cache but really improve performance?
Or do I wanna get the latest and not use ElastiCache?
CloudFront same thing.
It does cache stuff around the edges.

So if you use CloudFront, yes, you go global in minutes,
but you have the possibility of everything being cached
for one day on people's laptops.
So when you release an update to your websites,
maybe it will take time for people to get the new stuff.
So think about all these things.
Really, it's important, but overall performance
should be really in the middle of your thought process
as you do Solution Architecture.
Okay, that's it.
I will see you in the next lecture.

All right,
let's talk about cost optimization.
So what it is, we exactly know what it is,
is stability to run systems, to deliver business value
but at the lowest price point possible.
That makes a lot of sense.
Now, design principles will adapt a consumption mode.
So pay for only what you use.
So for example, AWS Lambda is one of these services.
If you don't use AWS Lambda, you don't pay for it,
whereas RDS, if you don't use your database,
you still pay for it
because you've provisioned your database.
So it's a really interesting trade off here.
Measure overall efficiency, use CloudWatch.
Are you using your resources effectively?
Then we have a small ad for AWS,
but the idea is that if you move to the cloud
then you stop spending money on data center operations
because AWS does the infrastructure for you,
and they just allow you to focus
on your applications, your systems.
And you need to analyze and attribute expenditure.
So that means if you don't use tags on your AWS resources,
you're gonna have to have a lot of trouble figuring out
which application is costing you a lot of money.
So using tags ensures that you're able
to track the cost of each application
and optimize them over time,
and get a ROI based on how much money
it generates for your business.
Finally, use managed application level services
to reduce the cost of ownership.
So that means that because managed services
operate at cloud scale,
they can offer such a lower cost per transaction or service,
and that's really something
you have to remember about the cloud.

They operate at cloud scale.
I've seen, I've read news about people
who are like three engineers,
only three of them, AWS engineers,
and they manage an application that serves 5 million people.
I mean, imagine that three people
manage a global application on AWS
for 5 million people just because they're able
to leverage the cloud and operate at cloud scale.
So in terms of cost optimization, what do we get?
Making sure we know what cost does something.
So budgets, custom research reports, cost explorer, and,
for example, Reserved Instance reporting,
making sure that if we do reserve an instance,
we're actually using them
and not just paying for unused Reserved Instances.
Cost-effective resources.
So are we using the right stuff?
For example, can we uses spot instances?
They're considerably cheaper.
Yes, they do have some trade-offs, but can we use them?
Can we use cost-effective resources?
Or if we know we're using an EC2 Instance for over a year,
maybe three years, because we provision a database on it,
can we use Reserved Instances?
That would be a great way of saving money.
AWS Glacier.
So are we basically putting our archives
in the lowest price point possible?
And Glacier is the lowest price point possible.
Are we matching supply and demand?
So are we not over provisioning?
So again, Auto Scaling, or maybe AWS Lambda
if you're using serverless infrastructure.
And are we optimizing over time?
So getting information from Trusted Advisor or again,
looking at our Cost and Usage Report,
or even reading the News Blogs.
Let me just share with you a small story.
There was this ELB feature,
and it allowed to use HTTP and HTTPS traffic going in,
but you couldn't do redirect of HTTP to HTTPS before.
And so you had to spin up an application
that was doing the redirect behind the scenes,
and that application was costing me a little bit of money.
But then reading the News Blog,
they said "now you can, straight from the ELB,
configure redirect of HTTP to HTTPS."
And that was great.
It saved me, you know, a bit of money every month

just with that one feature.
So reading the news does allow you to optimize your cost
and make sure you have the right price point.
Last story is, for example,
if you run an application on DynamoDB
but it's really inactive, it's a really slow application,
or you don't need to use a lot of operations,
maybe you're way better off using
the on-demand feature of DynamoDB
instead of using the reserve capacity, WCU, RCU, and so on.
Okay, so that's it just for small anecdotes,
but I hope that helps.
We've seen all the pillars now,
and I hope you understand better
the AWS Well-Architected Framework.
Again, the exam will not ask you deep questions on it,
but it's good as a solution architect
to understand it overall.
And if you're really curious about it,
I encourage you to read the Whitepaper.
Okay, that's it.
I will see you in the next lecture.

So, let's talk about the actual last pillar
in the well architecture framework, which is sustainability.
It's a new pillar.
So, this pillar focuses on minimizing
the environmental impact of running cloud workloads.
So the idea is that you want to understand
your impact, establish performance indicators,
evaluate improvements,
make sure you reach sustainability goals.
So, find these long terms and find your ROI.
Maximize the utilization of your services
because you wanna be energy efficient
and obviously be environmental conscious.
You want to anticipate
and adopt new more efficient hardware over time
because AWS does some optimizations to their infrastructure
and if you use their newer stuff,
then you are being more efficient.
Use manage services where possible
because you share the infrastructure with many people
and therefore you are in a better space for sustainability.
And reduce the downstream impact of your cloud workloads.
So, reduce the amount of energy
or resources required for your services
and your need for your customers
to constantly upgrade their devices.
So, some services that help with sustainability in AWS,

for example, EC2 Auto Scaling
or Serverless Offerings such as Lambda or Fargate.
And you want to get basically use
the right amount of compute for your task.
Cost Explorer; Graviton 2,
EC2T type of instances and Spot Instances.
All of them allow you to make sure you're energy efficient
when using the capacity compute of AWS,
and spot instances will allow you to use spare capacity,
which will be wasted otherwise.
Then some tiering storage,
so EFS-IA, Amazon S3 Glacier, Cold HDD for EBS volumes
all allow you to optimize your cost regarding storage.
Like is all your data needing to be hot,
these kind of questions.
Then S3 Lifecycle Configurations
S3 Intelligent Tiering
to make sure your data is in the right tier
as well as Amazon Data Lifecycle Manager.
And finally, databases can help you,
so Read Local, Write Global.
So think about RDS Read Replicas,
Aurora Global Databases,
DynamoDB Global Tables or using CloudFronts.
Okay. That's it for this lecture, I hope you liked it.
And I will see you in the next lecture.

So now let's talk about
the AWS Well-Architected Tool.
So the idea is that you want to review your architectures
against the six pillars we've just learned about
in the Well-Architected Framework
and then you want to adapt architectural best practices.
And so this is what it will look,
I will show you the hands on in a second.
And so how does it work?
Well you select your workload,
then you answer a few questions,
then you review your answers against the six pillars
in the Well-Architected Tool, Framework, sorry.
And then you obtain advice.
You get videos, documentation, you get reports and so on.
So let's have a look at how it works in the console.
So here I am in the console of the
AWS Well-Architected Tool,
and I can go ahead and define a workload.
So I'll call this Demo Workload
and you can add in some description, a review owner,
so, the description, then the review owner would be
john@example.com, and then what environment you're in.

So production, pre-production,
the regions you're operating in, for example, one,
and then if you're operating Non-AWS regions and so on.
The account IDs you're in, and so on and so on.
So you answer these things
and then you apply lenses to your workload.
So you can have these lenses such as
the Well-Architected Framework itself,
or the FTR Lens.
Or other lenses, the Serverless Lens,
the SaaS Lens, and so on.
And then you could also define your own custom lens
if you wanted to.
But, we'll just use is right now,
the AWS Well-Architected Framework lens.
So we click on Define Workload.
And then we can start reviewing
and start answering questions.
So the idea is that we'll start reviewing, and then this,
and the idea is that you're going to get questions
regarding all the six pillars in AWS.
So you get questions on operational excellence, security,
reliability, performance efficiency,
cost optimization, and sustainability.
And so you answer every question.
So you say, okay, these are my answers for this question.
And you say next.
And then you go into this one, you say,
this doesn't apply to my workload and here is the reason.
And then I click on next, and then so on and so.
And so you can see you have lots of questions
in lots of different areas,
so you can go to security, and I can say, okay,
I implement all these things for data in transit,
click on next.
And then when you're done,
and you can take a lot of time of course,
but this is for serious production application.
You click on Save and Exit,
and then you're going to get some risks.
So high risk, medium risk, and so on.
And so you can save this milestone
and say, okay, this is my first draft.
And click on save.
And in here, we can have a look at the lenses itself, so.
So I click on this lens.
And now because of my answered questions,
I have two risks being evaluated.
So I can go to improvement plan
and see that there is a high risk here

and a medium risk here.
And it gives me improvement items that link
to the documentation based on my answer.
So it's quite helpful for your teams to
go through this framework to make sure
that your applications are Well-Architected.
And in case they're not Well,
you get improvement areas, as well as
documentation links to make sure you are
running at your best.
Okay, that's it for this lecture.
I hope you liked it.
And I will see you in the next lecture.

So now let's talk about
the AWS customer carbon footprint tool,
which is a tool that's used to track, measure, review,
and forecast your carbon emissions
generated from your AWS usage.
So this is very helpful
if you need to meet some sustainability goals.
So this tool is going to give you access
to your carbon emissions, your savings,
and understanding which services
is emitting a lot of carbon,
as well as tracking your emissions over time
and seeing the path to 100% renewable energy
used for your AWS accounts.
So if you go under your billing
and cost management platform,
I prefer to type in the search "carbon".
And then under features,
you will find the customer carbon footprint tool,
which is going to take you directly where this tool is,
and it may move, so this is why I use the search bar here.
So here you get access to a start month and end month.
You get access to your emission summary,
your savings by geography and by services,
and in the bottom you see some statistics.
I'm not sure why I don't have data here
but I had data beforehand in 2021,
and we can see your emissions over time
as well as your path to 100% renewable energy.
Alright, so that's it, I hope you like this lecture,
and I will see you in the next lecture.

So now let's discuss
the AWS Cloud Adoption Framework also called CAF.
And this is something that is coming up at the exam
and you will have to remember a few things by heart

but hopefully by the end
of this lecture it should start to make sense.
So CAF is basically an ebook.
It's a white paper, and so at such, it's not a service
but the cloud adoption framework will help you
build and then execute a comprehensive plan to
do your digital transformation
through the innovative use of AWS.
So how to leverage a cloud to be transformed.
So it was created by AWS
by the professionals and its best practices
grouping thousands of customers all in one framework.
So the CAF has two components, I would say.
The first one is called organizational capabilities that
could be underpinning successful platform transformations.
And the second one is
that these capabilities are regrouped into six perspectives
and these six perspectives you have to remember
for the exam.
So we have business, people, governance, platform security
and operations, and then we'll go over them a little bit
for you to understand what they're about.
So we can divide them into, two categories.
The first one is the business capability.
And so we have the three pillars.
So we have the business perspective first.
And this business perspective helps ensure
that your cloud investments accelerate your
digital transformation, ambitions and business outcomes.
The people perspective is a second perspective
within the business umbrella, and it serves
as a bridge between technology and business.
Very important to remember for the exam.
This is a question that can be asked of you
and it helps you accelerate the cloud journey to
help organizations more rapidly evolve
to culture of continuous growth, learning
and where change becomes business as normal.
And so through the people you have a focus on culture
organizational structure, leadership and workforce.
And finally we have the governance perspective
which helps you orchestrate your cloud initiatives
while maximizing the organizational benefits
and minimizing transformation related risks.
So business people and governance
with people being the bridge
between technology and business.
And so within each of these perspectives
we have the capabilities.
And so for business

we have capabilities that are related to business.
You don't have to remember them
but it's good to look at them.
So strategy management, portfolio management
innovation management, product management
strategic partnership, data monetization, business insight
and data science.
For people, we have anything with two people.
So culture, evolution, transformational leadership,
cloud fluency, workforce transformation,
change acceleration
organization design and organizational alignment.
And for governance
we have program and project management, benefits management
risk management, cloud financial management
application portfolio management
data governance and data curation.
So that's for the business capabilities.
Regrouped under the three business perspectives
of business, people and governance.
And then we have what's called the technical capabilities.
So the technical capabilities are platform, security
and operations.
And platform helps you build an enterprise grade scalable
hybrid cloud platform to modernize your existing workloads
and to implement new cloud native solutions.
From the security perspective
it helps you achieve the confidentiality
integrity and availability of your data and cloud workloads.
And finally, from an operations perspective
it helps you ensure that your cloud services are delivered
at a level that meets the needs of your business.
So now if you look
at individual capabilities within these perspectives
we have platform security and operations.
So for platform, we have platform architecture
data architecture, platform engineering, data engineering
provisioning and orchestration
modern application development
and continuous integration and continuous delivery.
So for security, we have anything related to security
of course, such as security governance, security assurance
identity and access management, threat detection
vulnerability management, infrastructure position
data protection, application security
and incident response.
And finally, operations is all
around the operations review in the cloud.
So we're talking about observability, event management
incidents and problem management

change and release management, performance
and capacity management, configuration management
patch management, availability and continuity management.
And finally, application management.
So the goal is really when you go
at the exam to really understand what are
the perspectives that regroup these capabilities.
So you definitely have to remember business, people
governance, platform security and operations
because there will be something asked
of you and maybe sometime you will have to
identify a specific capability
and associate it with the correct perspective.
But if you look at the list we just went over
usually it makes sense, what is related to security?
Sounds like security, what is related to business
sounds like business and so on.
Now, there's one last part in this little diagram right here
which is the transformation domain.
So we have technology, process, organization, and products
and that helps you drive business outcomes.
And some questions at the exam can be geared
towards these transformation domains.
So I'm also gonna go over this.
So these four transformation domains.
The first one is technology.
So we use the cloud to migrate
and modernize legacy infrastructure, applications
data and analytics platforms.
For process, we want to be digitizing everything
automating and optimizing your business operations.
That means you want to two things, leverage new data
and analytics platform to create actionable insights
thanks to the cloud.
And using machine learning to improve your customer
service experience.
This is all thanks to the cloud, for organization.
How do you reimagine your operating model?
So for this, we want to reorganize the teams
around the products and the value streams
and you want to leverage agile methods
to rapidly iterate and evolve.
This is something that can be asked
for at the exam as per my experience.
And finally, for products, re-imagining your business model
by creating new value propositions such
as product and services and revenue models.
So all this thanks to the cloud.
Finally, CAF has transformation phases
that you need to know about.

There are four of them.
The first one is envision.
This is when you want to demonstrate how the cloud is
going to accelerate your business outcomes
by identifying transformation opportunities
and create a foundation for your digital transformation.
Then you have the align phase.
This is where you look at the six CAF perspectives
we've seen, and we identify capability gaps
and this will result in an action plan.
Then we have a launch
where we actually build and deliver
pilot initiatives in production and demonstrate
incremental business value, and finally
scale to expand these pilot initiatives with
desired scale while realizing the desired business benefits.
So the exam also is going to ask you
about these four transformation phases.
So be ready to know about them.
It's pretty easy.
Envision is about making sure you have
identified business opportunities.
Then align is to align those to the CAF perspectives
to find gaps and have an action plan.
And then launch and scale are pretty explicit.
So CAF may seem like a very obscure framework
and to be honest, it's something you kinda have to just look
at and follow when you're doing your cloud adoption.
Now, from an exam perspective, again, I want to reassure you
you're going to be asked about the perspectives
and identifying which one is part of the framework or not.
You may be asked to identify
if a capability belongs to a specific perspective.
And finally, you may be asked
about these transformation domains and their phases
but all these things kind of make sense
by elimination when you go through at the exam.
So even if you don't remember by heart what these are
just remember that by using a little bit
of brain power and elimination, you will get there.
Okay, so that's it for this lecture.
I hope you liked it.
I did my best to explain this framework to you.
And I will see you in the next lecture.

So this is a little bit of a weird lecture,
but it's about right sizing in AWS
and the exam may ask you one question about it.
So the idea is that because you have so many
instance types to choose in the cloud,

choosing the most powerful one is not the best choice,
because the cloud is elastic
and you can change the instance type whenever you want.
So right sizing is a process of matching
the instance type and size to your workload performance
and capacity requirements at the lowest possible cost.
Making sure the size is right.
Because why?
In the cloud well, you can scale up easily.
So always start small to find the right size.
It's also the process of continuously looking at
all your deployed instances.
And for example, looking at their metrics to identify
which are the opportunities to eliminate
or downsize without compromising on quality,
on capacity, performance and so on,
which will in turn, results in lower cost.
So it's super important to right size
in two moments in your cloud journey.
Number one, before you do a cloud migration,
because it is very common for a company
to just migrate to the cloud,
puts everything with biggest instance size and forget it.
This is not the best way.
Please right size before you do cloud migration.
And then once you've done the climate migration,
you need to do it continuously.
Maybe once a month,
because requirements change over time
and it's important to see if you need to
right size up or right-size down.
Okay?
So there's some tools that can help you right size.
For example,
CloudWatch, Cost Explorer, Trusted Advisor,
and other third-party tools that can help.
Okay?
So right sizing is very important.
The thing is you need to always scale up over time,
so start small.
And number two, you right size just before a cloud migration
and also continuously in the cloud.
That's it.
That's all you need to know for the exam.
I hope you liked it.
And I will see you in the next lecture.

We have seen a lot of different AWS services
in this course, but now it's time to talk
about the ecosystem around AWS.

So, there are a lot of free resources.
For example, there is the AWS blog,
there are the forums, which is community based,
so people will help each other.
You have the whitepapers and guides,
for example, the Well-Architected Framework
is one of the whitepaper.
They're usually very long guides that AWS did approve,
or did write, and they help you understand
how to do something correctly,
for example, around security, or around architecture,
or around networking, and so on.
And then you have the AWS Partner Solutions
also called formerly Quick Starts.
And the idea is that with this
you get automated, gold-standard deployments
in the AWS Cloud.
And the idea is that you can build
a production environment very quickly with templates.
So, this is an example of a template.
For example, how to deploy WordPress on AWS,
we'll click on it in a second.
And the idea is that, once you click on it,
you have a CloudFormation template ready to go
that you can input some parameters in,
and deploy the partner solution on AWS.
So, for example, this one by WordPress,
the WordPress High Availability by Bitnami on AWS
allows you to use WordPress.
And so, if we will look at the actual infrastructure,
we can look at what you'll build.
So, there will be a highly-available infrastructure,
there will be Amazon Aurora, there will be Elasticache,
there'll be Amazon EFS, some EC2 instances,
some load balancing, and so on.
So, quite a lot of things that come into play.
And this is all part of the solution.
So, we can have a look at how to deploy,
and we have two options.
We can either deploy into a new VPC,
or deploy into an existing VPC.
So, for example, let's deploy this into an existing VPC
to remove all the VPC-related infrastructure.
So, the template is ready,
the template URL is referenced right here.
And instead of deploying it,
we can just view it in CloudFormation Designer,
and have a look at what is going to be deployed.
So, if we have a look at it, very simple,
we here have the WebserverStack, the RDSAuroraStack,

the SecurityGroupStack, and the ElastiCacheStack.
So, this is what's going to be deployed,
and each of them is a CloudFormation stack
with more resources within.
Then you have AWS Solutions, which is a way for you
to deploy vetted technology solutions in the cloud.
And so, one of them was called AWS Landing Zone,
which was a way for you to deploy a secure,
multi-account environment in AWS.
And this was the link.
But now, if you remember this from this course,
this has been replaced by this new service
named AWS Control Tower.
But there are, obviously, a lot of different solutions
available out there on the solutions link
that you can see in the slide.
So, all of these free resources are available
to you at no cost, but then there is support by AWS.
So, we've seen different level of supports.
So, on Developer, we get business hour email access
to cloud Support Associates.
We get general guidance of less than 24 hours,
and system impaired of less than 12 hours.
If we are in Business subscription,
then we're going to get 24/7 phone supports,
email and chat access to Cloud Support Engineers.
So, one level up.
In case your production system is down, you get,
sorry for impaired, you get less than four hours,
and for production system down,
you get less than one hour of reply time.
And for Enterprise, which is the most expensive tier,
you get access to a dedicated technical account manager,
or TAM, you get Concierge Support Team to help you
with billing and account best practices.
And in case your business system is down,
then it's going to be less than 15 minutes
before you get a reply from the support of AWS,
which is very handy.
So, again, remember the different levels of support
as part of the ecosystem.
Okay, next we have the Marketplace.
So, I just talked about it briefly in the EC2 section,
but it is going to be a digital catalog
with thousands of software listing
from independent software vendors, which are third party.
For example, on the Marketplace you can buy a custom AMI
for someone that was customized with some OS,
some firewalls, or some technical solutions.
You can also buy CloudFormation templates

that are going to be production-ready.
You could buy Software As A Service
directly from the Marketplace.
You can buy containers.
And if you buy through the Marketplace,
the advantage that it goes directly into your bill of AWS.
So, you don't need to source a new vendor.
Then you can sell your own solutions on the Marketplace,
if you want to become a Marketplace seller.
So, as we can see, a Marketplace can be very easy,
and we've seen it in the beginning of the course
regarding the AMIs.
Next, we have training from AWS.
So, we have two sources of training given directly by AWS
that can be Digital, so online,
or Classroom training with in-person, or virtual.
There is also Private Training, if you want to organize
a training session for your organization.
And then there is dedicated training
and certification program for the US Government,
and a training and certification program for the Enterprise.
So, that's for the training that AWS can give you.
Then we have AWS Academy to help universities teach AWS,
so that people, when they get their university degree,
they also know how to use the cloud.
And then, obviously, not sponsored training by AWS,
but you get your, obviously, favorite teacher online,
that is me, teaching you about certifications
and more on AWS.
Okay. So, this is for the options of training
that you can get.
Then you're going to get Professional Services,
and the Partner Network.
So, there is a team called
the AWS Professional Services Organization.
It's a global team of expert that can help you with AWS.
They will work alongside your team,
so your people handling AWS, and a chosen member of the APN.
APN stands for AWS Partner Network.
It's a network of people that AWS knows are good
with the cloud.
So, we have the Technology Partners,
so the APN Technology Partners,
they will be providing hardware, connectivity, and software.
So, they will be more helping you
on the infrastructure side,
whereas the Consulting Partners on APN
are going to be professional services firm
that will help you build on AWS.
Next, we have the APN Training Partners

who can help you learn AWS, they will be training partners
that will be able to deliver the AWS training
that I've told you just from the slide before.
And then we have the Competency Program,
which are given to the APN partners
who have demonstrated technical proficiency,
and proven customer success in specialized solution areas.
And then, finally, one last thing, the Navigate Program
to help partners become better partners,
to train the partners.
So, I know this is a very long lecture on the AWS ecosystem,
but at the exam you'll get one, maybe, or two questions
on finding the right service, or the right tool,
for the ecosystem.
And so, for you, it's really, really important
to remember all the things that I just said.
Thankfully, it makes sense.
The names are most likely going to be obvious,
once you know, and once you've seen them once.
So, I would recommend just trying to memorize those.
We're going to help you with a quiz in these sections.
But again, feel free to revisit this lecture
before your exam to give yourself a small refresher.
So, that's it.
I hope you liked it, and I will see you in the next lecture.

So now let's talk about AWS IQ.
So IQ is used to quickly find a professional
to help you with your AWS projects.
So the idea is that you're going to have
a lot of third-party experts that are AWS Certified
and they're available for on-demand project work.
So you'll use IQ to engage and pay them.
On top of it, IQ provides you with video conferencing,
contract management features,
secure collaboration and integrated billing.
So if you're a customer of AWS IQ, then you're
going to submit a request to describe your project,
and then it's like a freelancer platform, right?
So you're going to review your responses
and you're going to connect to experts
based on requirements and timelines and so on.
Then when you've shortlisted everyone
you can select an expert,
for example, based on rates and experience and so on.
And then you can work securely with that expert
by giving him access, or her access
securely to your AWS accounts.
And finally, once you're happy with the work
being delivered, you can unlock milestones

and then the milestones will be charged directly
into your AWS bill.
So instead of going to a random freelancer platform,
you can just use AWS IQ to find these experts directly
on this platform and pay from your AWS accounts.
And if you're an expert, then you would create your profile
such as who you are, your photo, your certifications
and then you connect with customers,
then you start a proposal,
and then of course if it's accepted,
then you work securely and you get paid
after the milestones are unlocked.
The other option to get help is to use AWS re:Post.
So this is actually a forum, okay,
so this is a community forum in which you find answers,
you answer questions, you'll find best practices
or you can join groups.
So it's an AWS-managed Q&A service
that's offering crowd-sourced expert reviewed
answers to your technical questions about AWS
and that replaces what used to be the original AWS forums.
So the idea that it looks just like, for example
if you know Stack Overflow,
someone asks a question and then you get answers,
some of them can be up-voted, some of them can be accepted
and they're reviewed by experts all the time.
So it's part of the AWS Free Tier,
so it doesn't cost you anything to access it.
And then if you're a community member,
you can go on this forum and start to earn reputations
to build your expert status and review other answers.
So you can get some good stuff
and some good reputation from participating on re:Post.
Then in case you are a premium customer on AWS,
and you do not receive a response from the community,
then automatically your question is passed
to AWS support engineers and they will answer your question.
So that's pretty cool.
And very important at the exam, you should know this,
is that the re:Post is not used to get
to questions that are time sensitive,
or to involve any preparatory action, okay?
If you need a help right now,
then re:Post is not your service.
So that's it for this lecture.
I hope you liked it.
And I will see you in the next lecture.

So, now let's talk
about AWS Knowledge Center,

which is part of the re:Post portal.
So this is a place where you can find
the most frequent and common questions and requests on AWS.
So, they have a lot of different categories
such as popular services, analytics,
application integration, and so on.
And what I'm going to do is just show you
the online portal right now.
So, I an now on the Knowledge Center
and it's part of AWS re:Post.
And so here we have the featured questions.
For example, I can look at this one.
And you have troubleshooting of objects,
replication for Amazon S3 buckets, for example.
So, you have the question and then you have the resolution,
and a long article that helps you with these things.
So, this is quite helpful.
And if you go into Knowledge Center,
you can go for newly created or popular ones,
and then you can browse all the Knowledge Center content.
And as you can see here, you can filter by tags.
So, you can filter by AWS service.
You can do a search and so on.
You can sort them, you can view them as a list.
I mean, there's a lot of things you can do with this portal.
From an exam perspective, it's going to be the place
to find the answers to very common questions,
so to frequently ask questions and best practices on AWS.
Okay, so that's it for this lecture.
I hope you liked it.
And I will see you in the next lecture.

So now let's talk about AWS Managed Services or AMS
which is a bit weird because well many services
on AWS that shares RDS are managed services.
But AWS Managed Services is actually a team of people
that is going to provide you infrastructure
and application support on AWS.
So AWS has a team of AWS experts
that provide services and they are called the AMS team,
the AWS Managed Services.
And what they will do is
that they will manage and operate your infrastructure
for security, reliability, and availability.
And they will help organizations upload routine
maintenance tasks and focus on their business objectives.
So it's a fully managed service and AWS is going to handle
many common activities such
as exchange requests, monitoring, patch management,
security, and backup services.

And they will implement best practices
and they will maintain your infrastructure
to reduce your operational overhead and risk.
They run 24/7 and 365 days a year
and they really help you just get started in the cloud
and worry about other things
and how to run the cloud itself.
So to summarize, the managed services of AWS
enable you to create a baseline
and then you can sustain, build or migrate
and they will help you with that as well.
And then they will help you operate your infrastructure
in the cloud.
So it's not a service, it's not like something you can do
from the console directly.
You have to contact the sales of AWS.
They will get in touch and then they will help you.
Finally, with thanks to this
(indistinct) the benefits while you get improved security,
you have focus on automation,
because most of what they do is actually automated,
stronger compliance, reduced operating costs,
because you are dealing with experts,
simplified management and frictionless innovation.
Alright, that's it for this lecture.
I hope you liked it and I will see you in the next lecture.