**Lession 1: AWS Cloud Architect program introduction**

**1. Welcome to Udacity**

<https://www.youtube.com/watch?v=9QadFJRKrEA>

**2. Welcome to the Nanodegree program experience**

<https://youtu.be/gd0Z0S58Y68>

**3. Welcome to AWS Cloud Architect Nanodegree Program**

Welcome to the AWS Cloud Architect Nanodegree Program! Hi, my name is Alison Rodal and I am a Product Lead here at Udacity. The Udacity content team partnered with a great team of industry experts to put this program together, and we are thrilled to have you join us.

Why a program for AWS Cloud Architcture? 80% of organizations will migrate to cloud solutions by 2025, according to Gartner, and today in North America nearly 60% of organizations utilize public cloud services, according to Forrester’s cloud computing predictions. As organizations continue to adopt cloud solutions, their demand for professionals with the expertise to design, build and migrate applications to the cloud continues to grow. Cloud architects that have experience in designing AWS infrastructure will meet a critical need, with AWS commanding over a third of today’s public-cloud market share.

In this Nanodegree program, you will acquire the necessary skills to advance your career and become an AWS cloud architect through hands-on practice building reliable, secure, and scalable AWS infrastructure. You’ll learn the critical role cloud architects play in the design and execution of an organization’s cloud computing infrastructure, and the important gap cloud architects bridge between business objectives and cloud solutions, by converting objectives into executable technical requirements that support the business. Cloud Architects work with cross-functional team members to design, plan, and deploy highly available, scalable, and secure cloud services, and we are excited to support you in this step in your career!

Throughout the program interactive lesson content, exercises, and hands-on projects will let you apply and practice your new skills. You’ll learn how to to build highly available and recoverable solutions that apply to common use cases, and design for cost, performance, and scale in AWS using industry standard and open source tools. You’ll also learn how to protect an organization’s data and other assets by implementing cloud security best practices and how to design a DevSecOps pipeline that ensures security practices are implemented early on. You will learn from AWS cloud architecture experts in the field who have designed infrastructure for organizations of various size and across industries with more than 25 years collective experience.

Every project you submit gets reviewed and our project reviewers will provide personalized feedback on how you can improve your work. These projects will serve as a portfolio you can use to demonstrate your skills. After you complete the program, you will be ready to become an AWS Cloud Architect or bring these crucial skills to your current role. We are excited and honored that you are taking this step in your journey with us.

**4. Prerequisites**

Prior experience as a cloud architect is not required. You should, however, be comfortable with an Object-Oriented-Programming language like JavaScript, Java or Python, web application development and knowledge of HTML and CSS, and Command Line Interface. One to two years of experience building applications or managing cloud services in AWS will also help you to succeed in this course.

**5. Meet Your Instructors**

Course 1: Design for Availability, Reliability, and Resiliency

Tom Verbiscer, Director of Engineering at Current Media

Course 2: Design for Performance and Scalability

Leslie Bell, Technical Trainer, AWS Training and Certification Group

Course 3: Design for Security

Mehdi Rezvi, Senior Cloud Architect, NuEra Automotive Solutions

**6. How to succeed?**

<https://youtu.be/hVwUijsLKzw>

**7. Sign in to AWS and Monitor Costs**

**Launch the AWS Web Console in your Udacity Classroom**

You are given a *federated user account*, a temporary AWS user account with limited permissions, that you can use in this program.

To log in to an AWS console, click the **button in the left navigation** labeled **"Open AWS Gateway"** and then **"Open AWS Console."** This will open the AWS console in a new browser tab. This may take a few moments to load the first time. See a brief video below for a walkthrough.

*If you have a pop-up blocker installed, it may prevent the new tab from launching. Please be sure to allow pop-ups from Udacity.*

**Important Points to Remember**

1. **Session limit**

Note that there is a **session time limit**. If reached, you will automatically be timed out. As long as you have not used your entire budget, your work will be saved. You can re-launch using the same "Open AWS Gateway" button in the left navigation menu to return to your session.

1. **Default AWS region**

The default AWS region for you will be US East (N. Virginia) (*us-east-1*). However, you can switch and practice in select few other regions as well.

1. **Budget allocated for you**

All AWS services are a pay-as-you-go service. Udacity has set a budget for each student to complete their course work. Please understand that these credits are limited and available for you to use judiciously. **The budget for this entire Nanodegree is $75 for you.**Although, we find about $20 sufficient for most to complete this Nanodegree.

1. **Shut down your resources | No extra credits**

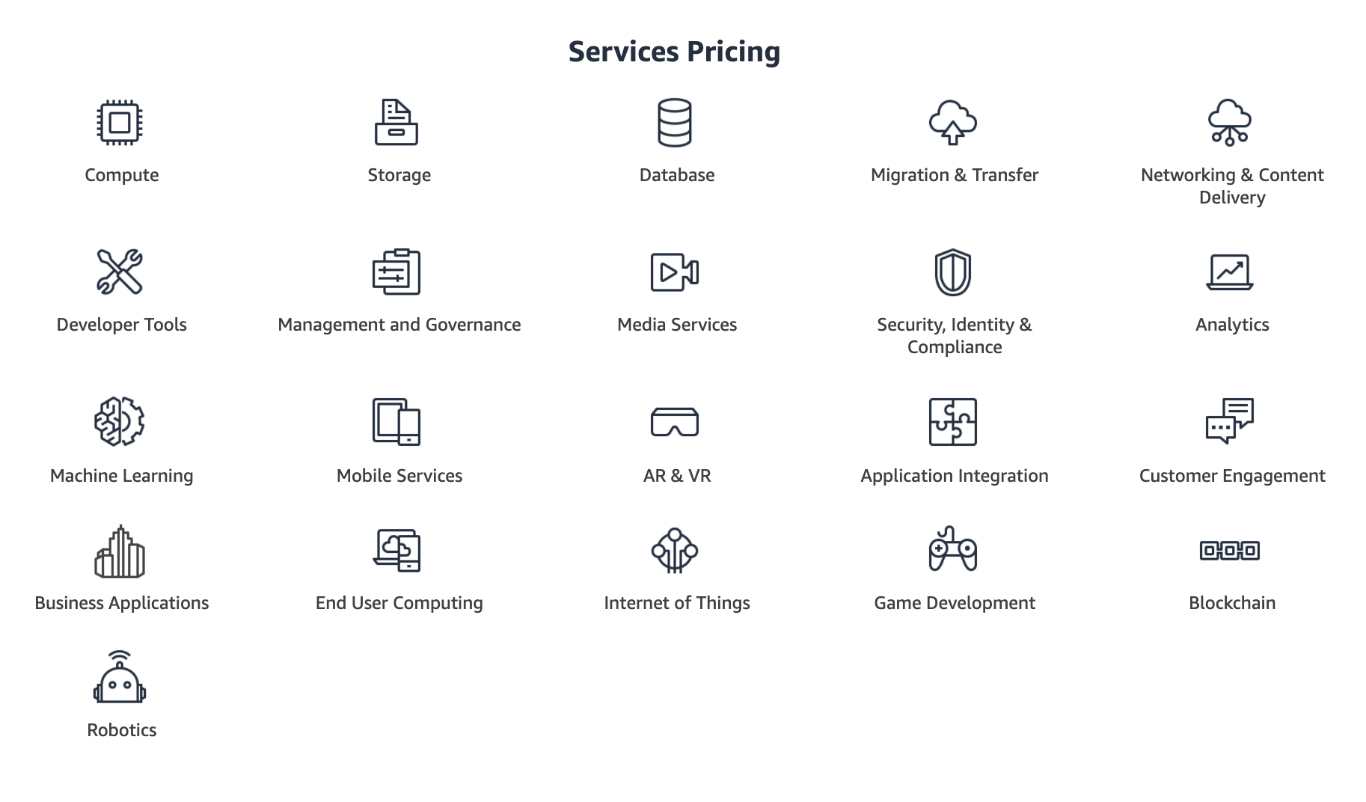
We recommend you **shut down/delete every AWS resource** (e.g., EC2, Sagemaker, Database, EMR, CloudFormation) immediately after the usage or if you are stepping away for a few hours. Otherwise, you will run out of your allocated budget of $75.

**Udacity will not provide additional credits.** In case you exhaust your credits:

* + **You will lose your progress on the AWS console.**
  + **You will have to use your personal AWS account to finish the remaining ND.**

Even if you are in the middle of the project/exercise and need to step away, you must shut down your resources. You can re-instantiate them later. To better understand pricing, see the [AWS Pricing](https://aws.amazon.com/pricing/) for all available services.

*For reference, any service available to you @$0.1/hour or higher should be monitored closely and shut down immediately after use or if you are stepping away.*

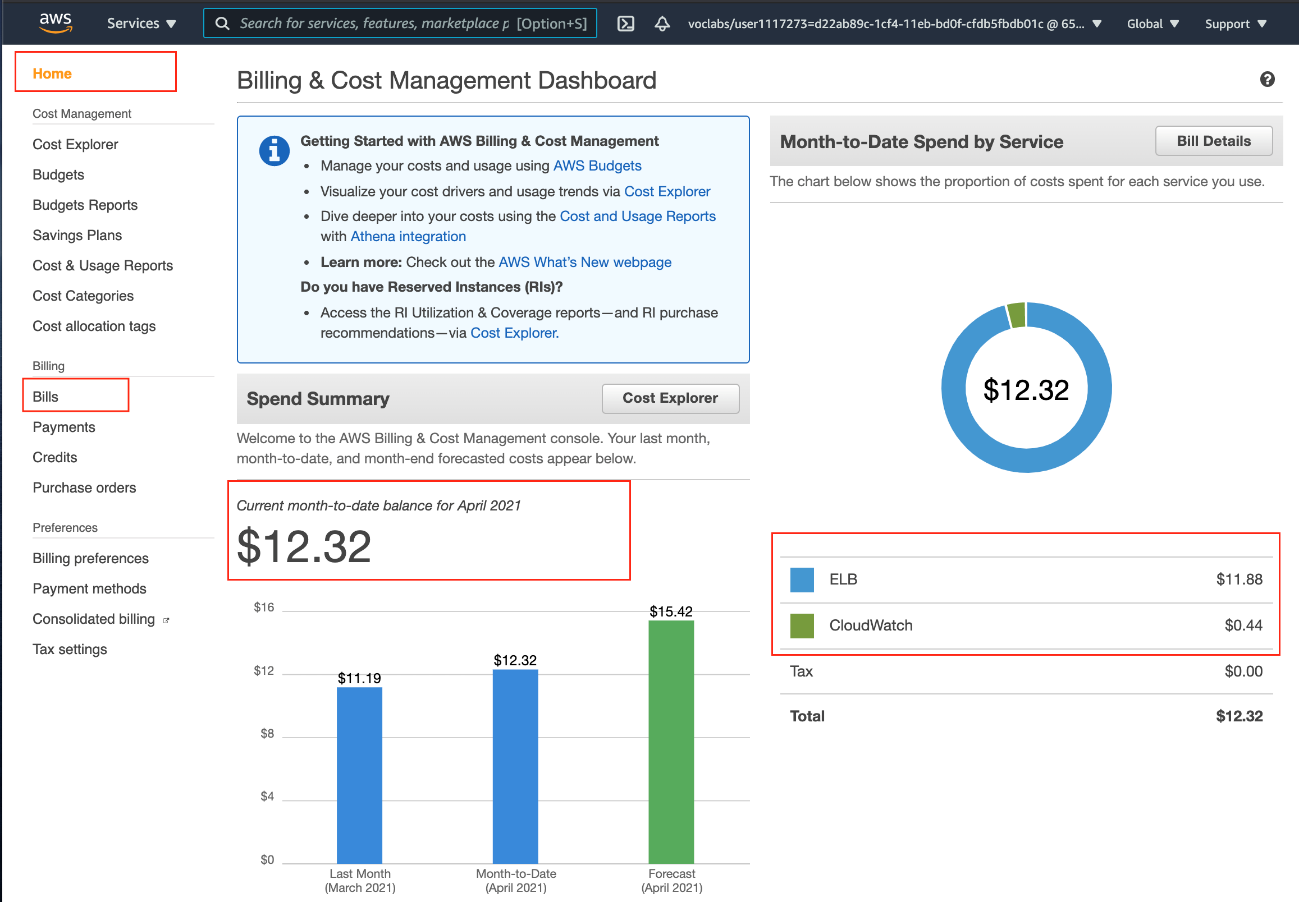


Check the pricing at <https://aws.amazon.com/pricing/>

1. **Tracking your usage**

You need to make sure that you have an adequate budget available to complete your project/task. **If you hit your budget, your session will time out and your work will be lost and unrecoverable.**Options for you to stay vigilant are:

* + Track your usage on the AWS web console. Go to [AWS Billing Dashboard](https://console.aws.amazon.com/billing/home#/), and view the monthly spending. It will list you the services constituting the spend.
  + Submit a [ticket](https://udacity.zendesk.com/hc/en-us/requests/new?ticket_form_id=110806) to Student Support Services to know your current balance.



AWS Billing dashboard

***Note****-  
As you are given a temporary AWS user account with****limited****permissions, you might not be able to avail****all****AWS services. We have allowed the necessary ones only. If you see a few warning messages related to insufficient permissions, you can ignore them and proceed with your practice.*

**QUESTION 1 OF 2**

Select all of the true statements about AWS usage.

* Any service at $0.1/hour or higher should be monitored closely and shut down immediately after use to avoid exhausting your budget
* 

All progress on the AWS console will be saved even if you exhaust your budget

* Most students have completed the program using about $20 from their budget
* You will have to work with your personal AWS account to finish the program if you exhaust your budget
* You can track your monthly usage of credits within the AWS billing dashboard

SUBMIT

**QUESTION 2 OF 2**

What is the default AWS region set to?

* 

US East (Ohio) (us-east-2)

* US East (N. Virginia) (us-east-1)
* 

US West (N. California) (us-west-1)

* 

US West (Oregon) (us-west-2)

SUBMIT

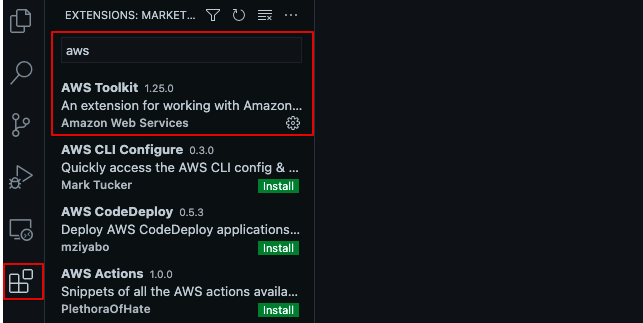
NEXT

**8. [Optional] Connect VS Code with AWS**

**Connect VSCode with AWS**

**Install AWS Toolkit Extension in VS Code**

* Open **Extensions** tab
* Search **AWS**
* Install **AWS Toolkit**



AWS Toolkit Extension Installation

**Step 1**: Create a new file called “credentials” with the following details:

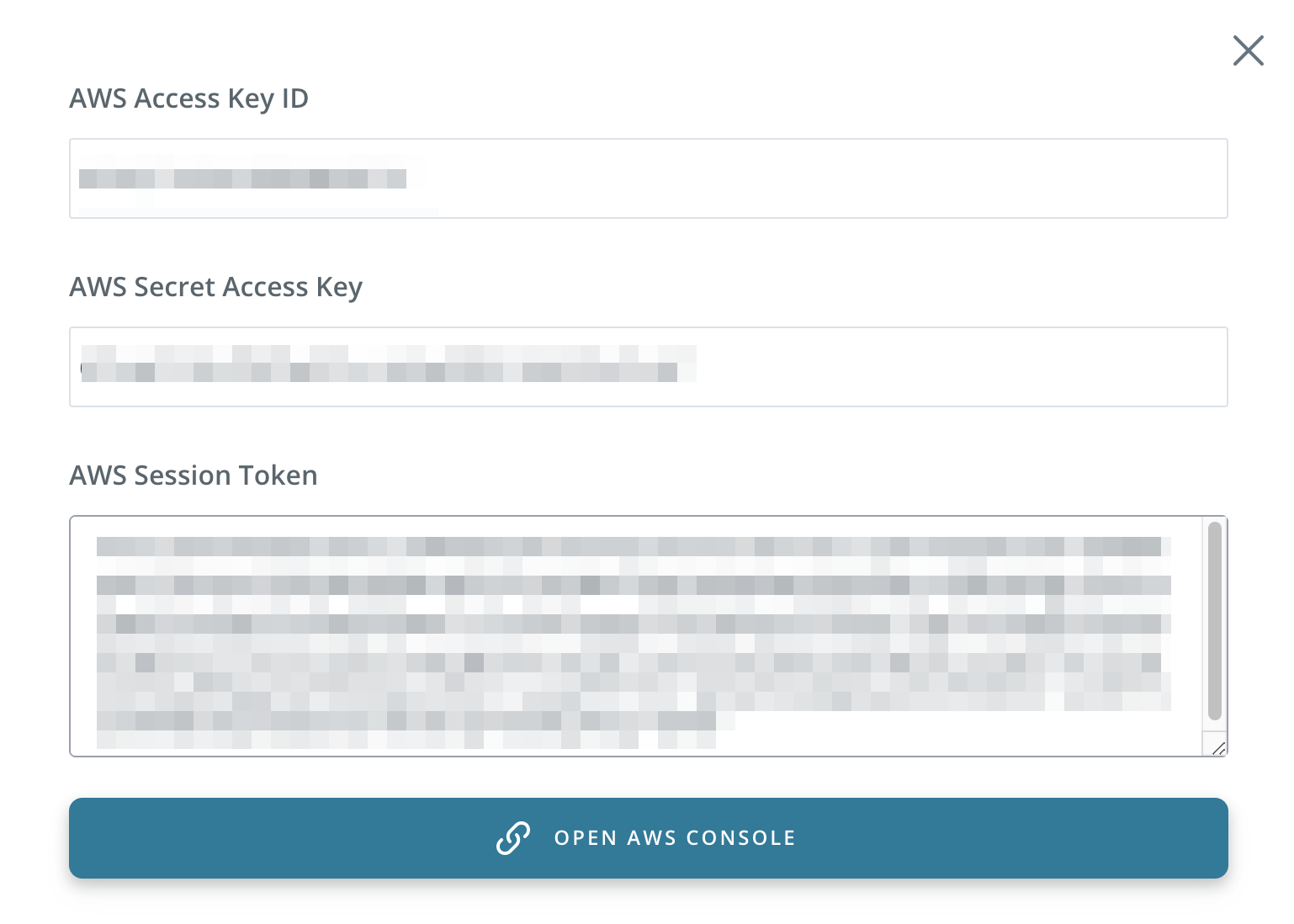
[default]

aws\_access\_key\_id=COPY\_FROM\_CLASSROOM

aws\_secret\_access\_key=COPY\_FROM\_CLASSROOM

aws\_session\_token=COPY\_FROM\_CLASSROOM

Replace COPY\_FROM\_CLASSROOM with the values you see when you click on the “Launch AWS Gateway” button in the classroom as shown below.



AWS Credentials

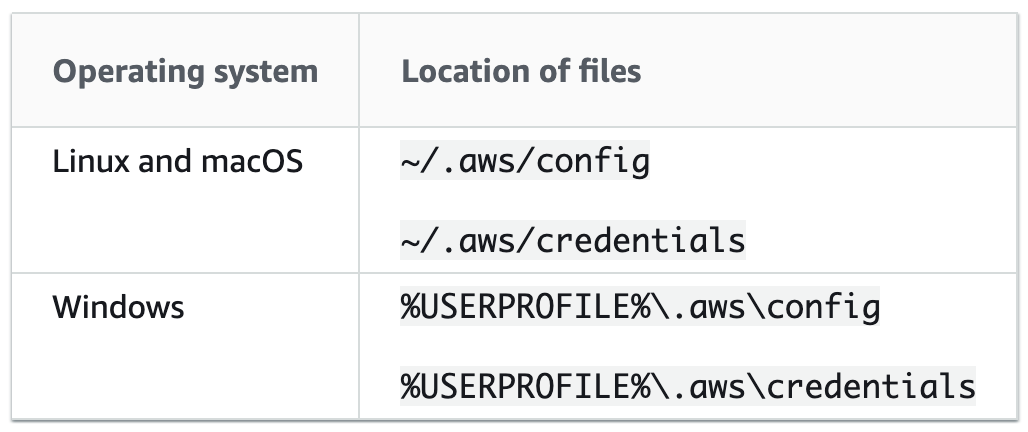
**Step 2**: Create a new file called “config” with the following details:

[**default**]

region=us-west-2

output=json

**Step 3**: Save the “credentials” and “config” file at the following location



Location of AWS credentials file

***Note****- If the .aws directory is not present, please create it first, and then save the files at the location specified in the table above.*

**Connect to AWS**

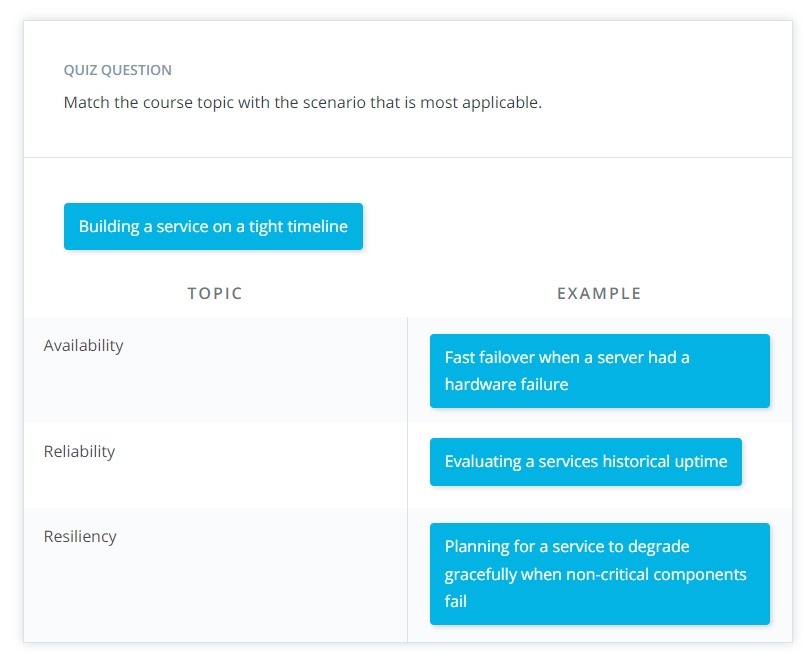
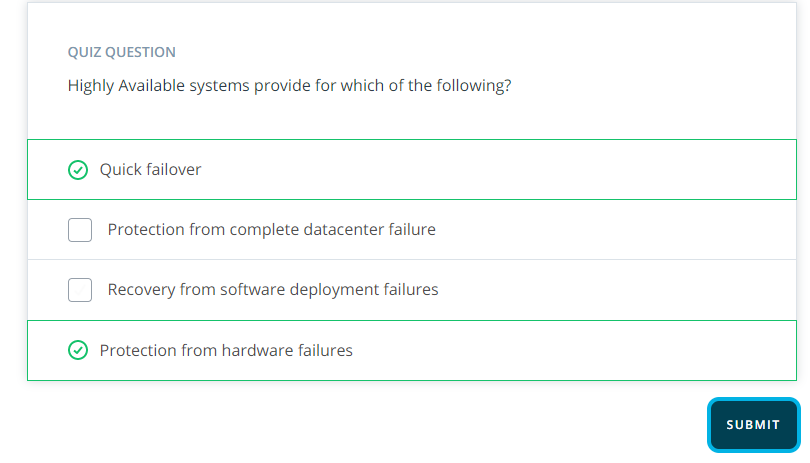
* Once the credentials are saved, open the VS Code’s command palette by pressing Ctrl+Shift+P for Windows/Linux OR Cmd+Shift+P for Mac, and select AWS: Connect to AWS
* On a successful connection, **AWS: profile:XXXXXX** will appear at the bottom left of VS Code window.

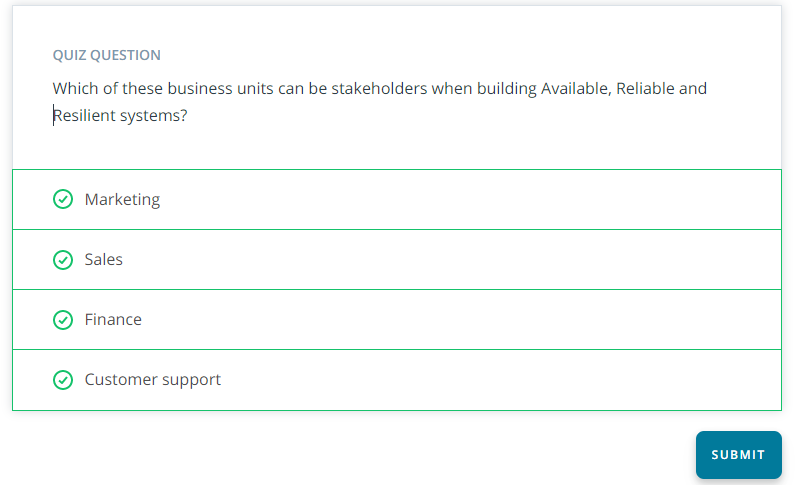


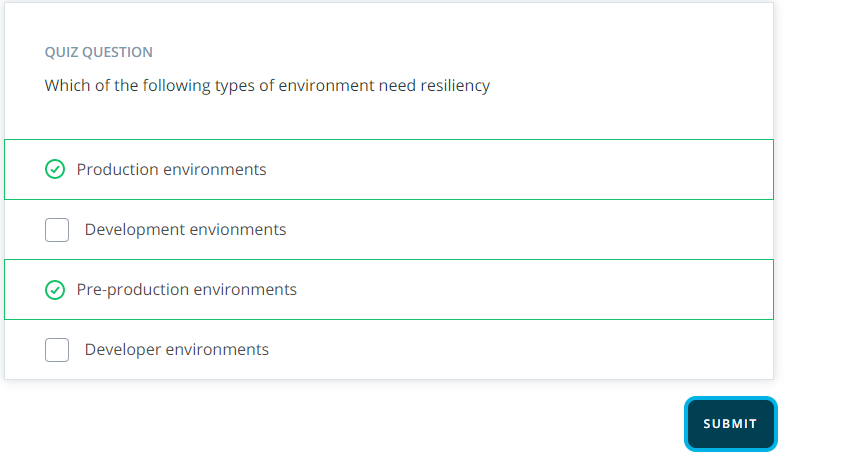
AWS: Connect to AWS

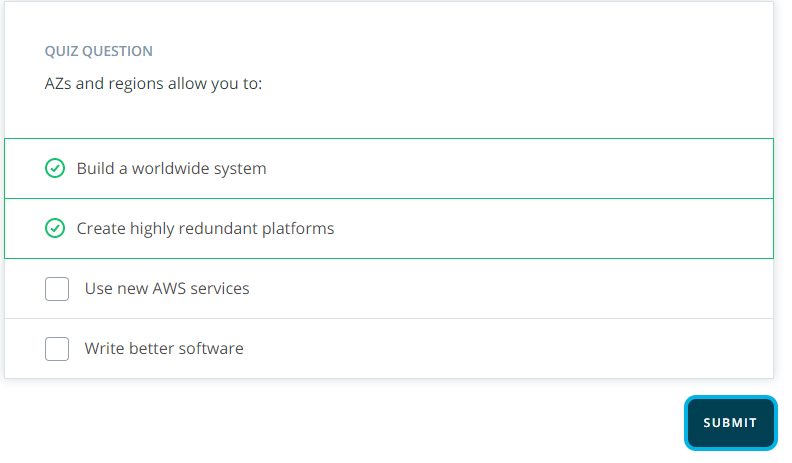


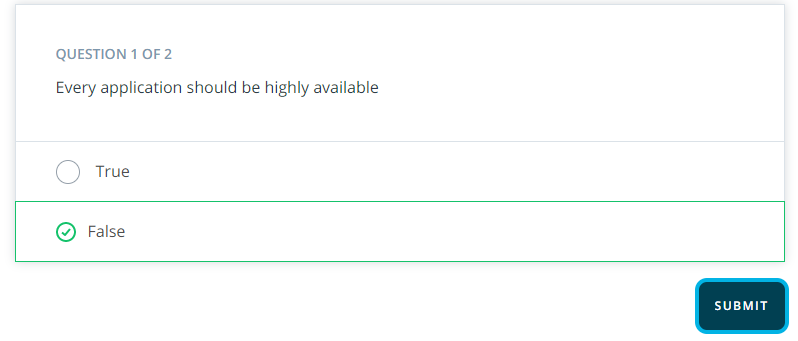
Successful AWS Connection Indicator

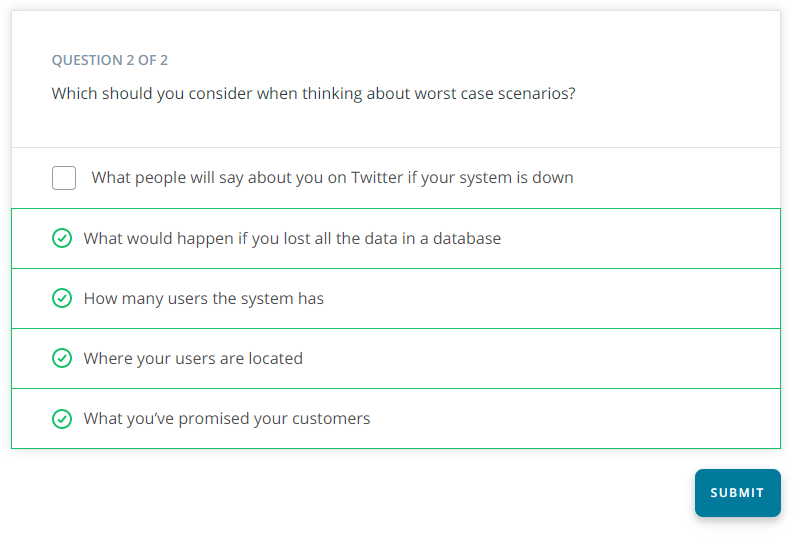
**  
**

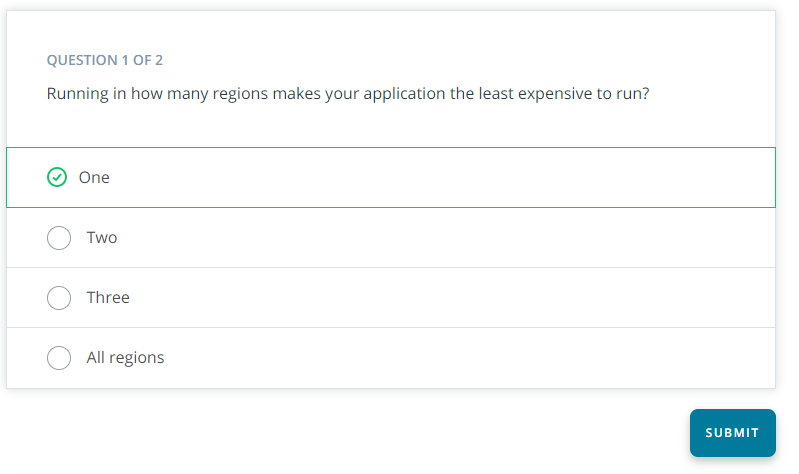
****

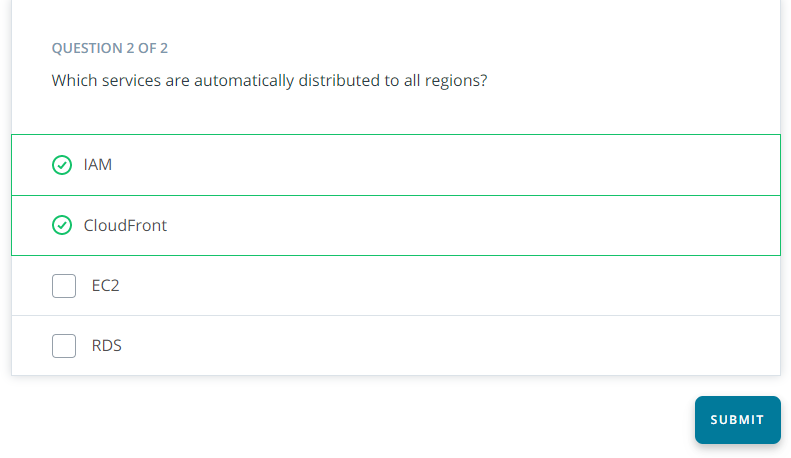
****

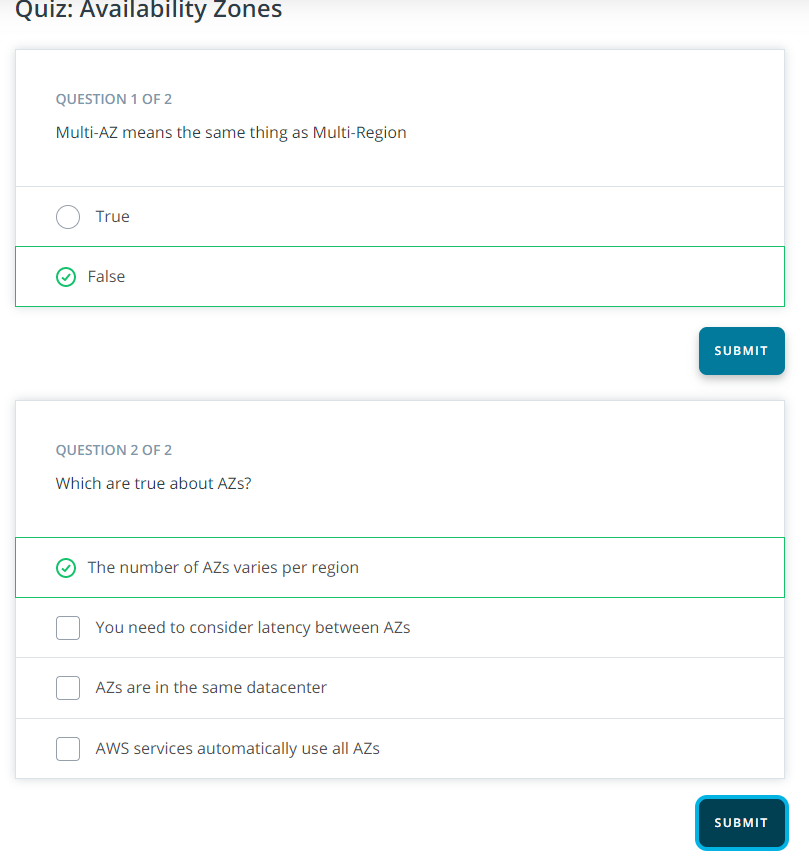
****

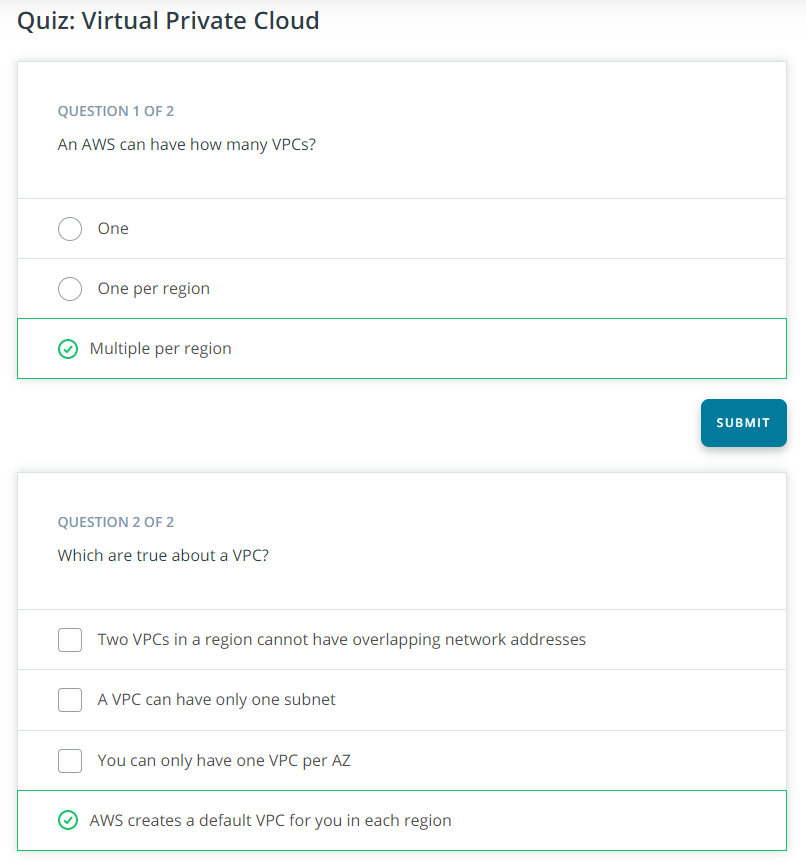
****

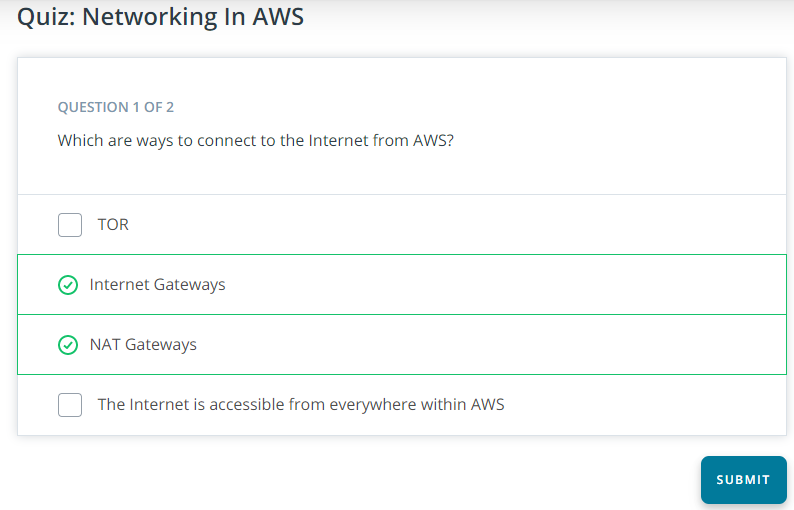
****

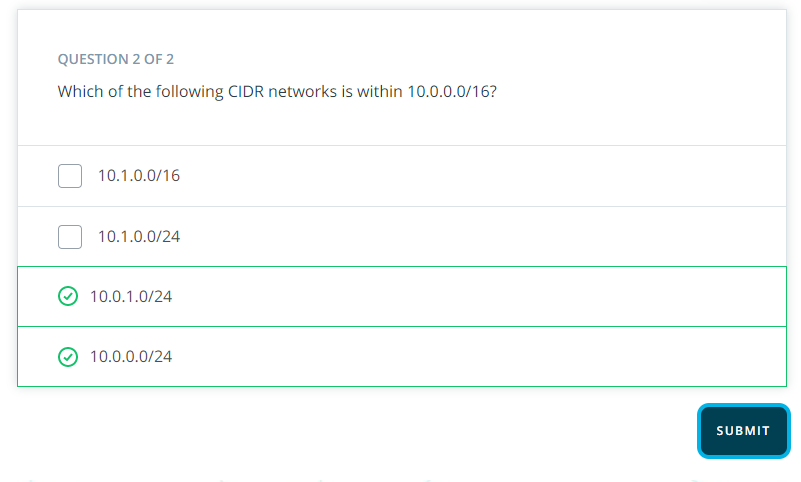
****

****

****

****

****

****

**Recovery Time Objective (RTO)**

When determining what type of RTO you can support, there are many factors to take into consideration. By not accounting for components that impact your RTO, you put your ability to meet the obligation at risk.

Let's consider the situation where a database runs out of disk space in the middle of the night. Some of the items to consider are:

Problem happens

An amount of time passes before an alert triggers

Alert triggers on-all staff

On-call staff may need to get out of bed, get to a computer, log in, log onto VPN

On-call staff starts diagnosing the issue

Root cause is discovered

Remediation started

Issue fixed

Estimate how much time each of the above is likely to take. Use that data to create a timeline of the incident. What is the RTO of this incident?

**Answer:** Recovery Time Objective (RTO)

00:00 - Problem happens (0 minutes)

00:05 - An amount of time passes before an alert triggers (5 minutes)

00:06 - Alert triggers on-all staff (1 minute)

00:16 - On-call staff may need to get out of bed, get to computer, log in, log onto VPN (10 minutes)

00:26 - On-call staff starts diagnosing issue (10 minutes)

00:41 - Root cause is discovered (15 minutes)

00:46 - Remediation started (5 minutes)

00:56 - Remediation completed (10 minutes)

Total time: 56 minutes

An RTO of one hour for this incident would be reasonable.

**Recovery Point Objective (RPO)** is the maximum amount of time that your system can lose data for. RPO is not tied to whether your system is available, it is a measure of the window of time that you may lose data in. If you take a database snapshot at midnight every day and you have a data corruption issue at 7:00am requiring you to restore from backup, you have a 7 hour RPO. It may take you until 8:00am to restore service, but as long as your data loss stops at 7:00am, that is when your RPO window closes.

Recovery Point Objective (RPO)

The RPO is a measure of how much data is lost during an incident. ***In a situation where you have an unintended change made to a database, the amount of time between the system being taken offline and the last backup is the RPO.***

In this exercise, you will create an RDS database and then demonstrate the ability to roll the database back to a specific point in time. This point-in-time recovery can greatly reduce the RPO of an incident.

Create a MySQL RDS database with backups enabled, let it run for a short period of time, and then "restore" it to a previous point-in-time.

**Disaster Recovery In AWS**

In AWS this typically means moving your service between regions. If you are not running active/active applications, you will need to prepare and plan in order to be able to bring up your platform in another AWS region.

Some services such as DynamoDB provide simple options for this. Services like CloudFront also have features such as Origin Groups that allow for automatic failover between AWS regions.

**Amazon CloudFront**

Amazon CloudFront là 1 dạng CDN

<https://viblo.asia/p/amazon-cloudfront-jvElaPLNZkw>

**CDN - content delivery network**

Một content delivery network (CDN) là một nhóm server đặt tại nhiều vị trái khác nhau để hỗ trợ nội dung được trải dài ở nhiều khu vực vị trí địa lý khác nhau.

<https://www.hostinger.vn/huong-dan/cdn-la-gi>

# Lesson Recap

* Uptime
* Downtime
* RTO
* RPO
* Disaster Recovery

# Lesson Objectives

You will be able to:

* Calculate availability in terms of up and down time
* Set reasonable business metrics for RTO and RPO
* Make determinations on what types of DR plans a company needs
* Implement a DR plan

Business objectives are a great conduit to have prioritization and cost discussions with the other functions of a business. Keeping communication open and setting reasonable, realistic goals is vital. AWS has all of the tools you will need to meet these objectives, but there will still be a cost in terms of time, money and resources.

## Glossary

* **Force Majeure:** Term describing an event or circumstance that is completely unavoidable.
* **IP whitelisting:** Allowing specific IP addresses only to access some resource.
* **Pilot light:** Old appliances such as furnaces or water heaters had a small flame that was always burning that was used to ignite a larger flame when needed.
* **Regulated industries:** Industries where the government sets strict rules and guidelines for data storage or operational practices.
* **Uptime:** The time where you application is available and correctly functioning.
* **Downtime:** The time where your application is not available service it's critical functions during normal business hours.
* **RTO:** Recovery Time Objective, or the time that your application can be down for a given incident.
* **RPO:** Recover Point Objective, or a measure of how long of a period that data can be lost.

## Further Reading

* [High availability](https://en.wikipedia.org/wiki/High_availability)
* [GitHub Post-incident analysis](https://github.blog/2018-10-30-oct21-post-incident-analysis/)
* [Atlassian Incident Handbook](https://www.atlassian.com/incident-management/handbook/postmortems)

# Lesson Outline

* Monitoring
* Alerting
* Recovering
* Automating

In this lesson, you will learn how to use AWS tools to monitor and alert on the systems that you build. You'll create alerts and think through how to find problems and recover from them.

# Lesson Recap

* Monitoring
* Alerting
* Recovering
* Automating

# Lesson Objectives

You will be able to:

* Monitor AWS applications
* Alert on problems in applications
* Recover failures in your platform
* Understand testing and tradeoffs in automating recovery from failure

In this lesson, you learned how to monitor and maintain systems in AWS. You also looked at what and how to recover systems that have failed. The larger your application grows, the more parts and services it will have. The more complex it grows, the more things that can go wrong. The more things that can go wrong, the more frequently they will go wrong. Expect failures, and plan to address and recover from them.

## Glossary

* **SSL certificate:** Cryptographic certificate for encrypting traffic between two computers.
* **Source of truth:** When data is stored in multiple places or ways, the "source of truth" is the one that is used when there is a discrepancy between the multiple sources.
* **Monitoring:** Systems to track and make visible metrics that are useful in identifying system performance.
* **Alerting:** Systems to attract attention when performance thresholds are crossed.
* **Chaos Engineering:** Intentionally causing issues in order to validate that a system can respond appropriately to problems.

## Further Reading

* [AWS Services That Publish CloudWatch Metrics](https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/aws-services-cloudwatch-metrics.html)
* [PagerDuty](https://www.pagerduty.com/)
* [VictorOps](https://victorops.com/)
* [Analyzing Log Data with CloudWatch Logs Insights](https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/AnalyzingLogData.html)
* [Everything Fails All the Time](https://cacm.acm.org/magazines/2020/2/242334-everything-fails-all-the-time/fulltext)
* [Principles of Chaos Engineering](http://principlesofchaos.org/)
* [Chaos Monkey](https://github.com/Netflix/chaosmonkey)

# Course Recap

### Regions and Availability Zones

Core to AWS resiliency. Regions and AZs are key to high availability in AWS.

### Building for Resiliency

Application service redundancy is different in different services. Here we learned how to identify these techniques.

### Business Objectives

Understanding how Engineering objectives map to Business objectives is important to keep your company speaking the same language.

### Monitor, React and Recover

Things will fail, and when they do, you must know before your customers and be able to react quickly to recover your platform.

[Design for Availability, Reliability, and](https://classroom.udacity.com/nanodegrees/nd063/parts/cd1908)

### What We Will Learn About in This Course

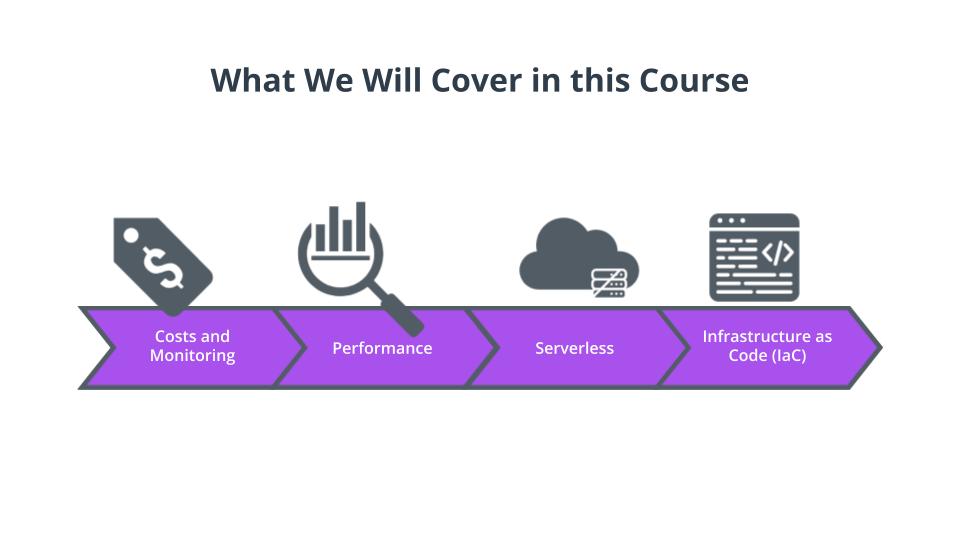
1. Cost Optimization and Monitoring
2. Performance Optimization
3. Serverless Technology
4. Infrastructure as Code

### Course Structure

The first half of the course covers broad topics that focus on how to make sure your environment is optimized for low cost and high performance.

After that, we'll get more technical, with a focus on AWS serverless technology, and how cost and performance optimization are built-in to the service platforms

We will finish by introducing you to Terraform by Hashicorp, the platform agnostic tool for documenting your infrastructure as code.



Performance Efficiency in the Cloud

### Four Cloud Models

* **Private Cloud**: Organization owns, operates, and governs their cloud computing resources.
* **Community Cloud**: Cloud resources provided for organizations and community groups.
* **Public Cloud**: Owned by the government, academic institutions, or a business (e.g. Amazon Web Services, Microsoft Azure and Google Cloud Platform).
* **Hybrid Cloud**: Combines public cloud and private cloud to allow data and resources to be shared between them.



### New Terms

| **Term** | **Definition** |
| --- | --- |
| Cloud | Servers and resources that are accessed over the internet |
| On-premises | Hardware and software that is contained within the physical confines of a business or their assigned section of a data center |

Getting to Faster and Cheaper

### Key Points

* **Moving to the cloud doesn’t guarantee that your application will be faster and cost less to operate**. In fact, without the proper baseline metrics, you won’t know where to start when planning your move.
* **Most performance issues can be traced to the application itself** and some applications will need to be redesigned to be cloud native in order to achieve truly optimized performance. Common causes of performance issues:
  + Poorly designed applications
  + Database design constraints
  + Inefficient network routes
* **Moving to the cloud should improve network response times.** Global edge cache locations can bring the server closer to the end user.

**High Performance In the Cloud vs. High Performance Computing**

### Key Points

* Supercomputing harnesses the computing power of multiple computers and aggregates them to tackle complex problems in business and academia.
* In supercomputing, computers are called nodes and nodes are organized into groups called clusters.
* Supercomputing is the ultimate in high performance computing in the cloud.

### New Terms

| **Term** | **Definition** |
| --- | --- |
| Baseline | Measurement of conditions at the start of your project used to make planning decisions |
| Cloud Computing | Storing and accessing data and applications over the Internet instead of a local or networked computer's hard drive |
| High Performance Computing (a.k.a. supercomputing) | Supercomputing harnesses the computing power of multiple computers and aggregates them to tackle complex problems in business and academia |
| Infrastructure as Code | The process of provisioning and managing your cloud resources by writing a template file that is both human-readable, and machine-consumable |
| Serverless | Compute model where the developer doesn’t need to be concerned with the server because the cloud provider takes care of it |

## Additional Reading

Please follow the links below to learn more about Amazon’s managed database services, their on premises hybrid cloud offering, and high performance computing in the AWS cloud.

* [AWS RDS](https://aws.amazon.com/rds/)
* [AWS Outposts Brings AWS Cloud Hardware on Premises](https://www.zdnet.com/article/aws-outpost-brings-its-cloud-hardware-on-premises/)
* [HPC on AWS](https://aws.amazon.com/hpc/)

Defining Goals

### Key Points

* Planning to move to the cloud is a process that requires pragmatism and neutrality to make the best decisions for your technology.
* The team is comprised of directors, managers, and engineers from Infrastructure, Operations, and Software Engineering teams
* It is important to define your migrations goals and base your migration plans on your actual metrics. Metrics provide the information you need to make data-driven decisions.

### Examples Of Cloud Migration Goals:

* We are migrating to the cloud to reduce our infrastructure costs by 25%
* We believe that by hosting our application in AWS, we will be able to deliver download speeds that are 60% faster than our baseline
* We are going to duplicate our infrastructure in the cloud and maintain our AWS account as a warm backup site for disaster recovery
* We are consolidating our data centers and moving to the cloud because our AWS account will provide one centralized view into our environment with more visibility into how our compute spend is being utilized.

## Additional Reading

The links below offer additional insight into the practice of cloud migration, and will offer you further details into the scope, procedures, and goals of a cloud migration project.

* [The 5 “W's of Cloud Migration](https://www.cloudindustryforum.org/content/getting-cloud-faster-5-ws-cloud-migration)
* [Cloud Migration](https://aws.amazon.com/cloud-migration/)

Historical Context

**Historical Context for Performance in the Cloud**

### Key Points

* AWS was started to sell excess compute capacity owned by Amazon's e-commerce business. The on-demand, no-contract service model has revolutionized IT operations.
* Managing cloud spend is a significant challenge for many organizations.
* Cloud architects who understand how to manage both performance and cost are in high demand.
* Our primary focus will be on technology provided by AWS, but the underlying concepts we discuss apply to all cloud providers.
* In this course, "Amazon" refers to AWS (Amazon Web Services), the cloud platform segment of Amazon Inc., which also owns the e-commerce giant Amazon.com.

#### AWS Changes Frequently!

What you see in the course may not match what you see when you access AWS yourself. This course reflects how the AWS management console looks and functions in March of 2020.

The [AWS documentation](https://docs.aws.amazon.com/) is a great place to find the latest information.

NEXT

Getting Started: AWS Console

### Key Points

AWS now has over 175 branded services available via the console. While it is great to familiarize yourself with their names and functionality, it isn’t realistic to expect to learn how to use all of them during this program.

New services are added and existing services are updated regularly. You'll want to follow the AWS blogs, watch their broadcasts on YouTube, and read their documentation and FAQs for details on new and updated services.

## Additional Reading

As you progress in your careers, you will become familiar with AWS notices and bulletins about new cloud services and you’ll hear the voices of experts in cloud technology. It is always advisable to stay updated on the latest updates by subscribing to AWS newsletters, listening to podcasts, and reading blog posts by cloud experts.

* [AWS News Blog](https://aws.amazon.com/blogs/aws/)
* [What’s New With AWS](https://aws.amazon.com/new/?whats-new-content-all.sort-by=item.additionalFields.postDateTime&whats-new-content-all.sort-order=desc&wn-featured-announcements.sort-by=item.additionalFields.numericSort&wn-featured-announcements.sort-order=asc)
* [AWS Release Notes](https://aws.amazon.com/releasenotes/)
* [AWS Insider](https://awsinsider.net/Home.aspx)
* [Reddit | Amazon Web Services](https://www.reddit.com/r/aws/)

Prerequisites and Tools Needed

## Tools Needed for This Course

## Are You Ready to Begin the Course?

### Prerequisites

Prior experience as a Cloud architect is not required, but you should have experience with:

* Using the Command Line Interface
* Basic scripting
* Familiarity with Web application architecture, hosting, infrastructure, and components

### Tools Needed

To successfully complete this course you will need:

* AWS CLI and Terraform installed on your local machine
* An integrated development environment (IDE) like VS Code or Atom
* Acces to diagramming application like Lucid Chart or Diagrams.net

### Installation Instructions

* [AWS CLI Installation Instructions](https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-install.html)
* [Terraform Installation Instructions](https://learn.hashicorp.com/terraform/getting-started/install.html)

### Suggested IDEs

* [VS Code](https://code.visualstudio.com/)
* [Atom](https://atom.io/)
* [Sublime Text](https://www.sublimetext.com/)

### Suggested Diagramming Applications

* [LucidChart](https://www.lucidchart.com/)
* [Creatly](https://creately.com/)
* [Diagrams.net](https://www.diagrams.net/)
* [Gliffy](https://www.gliffy.com/)

You'll also need access to the AWS Console. See the next page for instructions.

Course - AWS Sign In and Costs

## New Terms in This Lesson

| **Term** | **Definition** |
| --- | --- |
| Baseline | Measurement of conditions at the start of your project used to make planning decisions |
| Cloud | Servers and resources that are accessed over the internet |
| Cloud Computing | Storing and accessing data and applications over the Internet instead of a local or networked computer's hard drive |
| High Performance Computing (a.k.a. supercomputing) | Supercomputing harnesses the computing power of multiple computers and aggregates them to tackle complex problems in business and academia |
| Infrastructure as Code | The process of provisioning and managing your cloud resources by writing a template file that is both human readable, and machine consumable |
| On-premises | Hardware and software that is contained within the physical confines of a business or their assigned section of a data center |
| Serverless | Compute model where the developer doesn’t need to be concerned with the server because the cloud provider takes care of it |

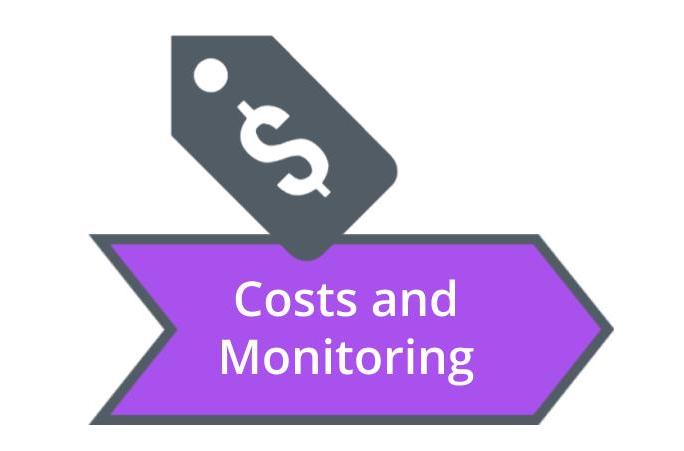
[Lesson 2:Cost and Monitoring](https://classroom.udacity.com/nanodegrees/nd063/parts/cd0345)

**Key Points**

Managing cloud spend is one of the major challenges facing experienced IT organizations today.

In this lesson, we will cover:

* Cloud Costs
* Service Pricing
* Cost Management
* Cost Optimization and Monitoring



Intro to Costs in the Cloud

**Three Ways to Calculate AWS Costs**

* **Simple Monthly Calculator** - allows you to explore AWS services, model solutions, and create estimates for the cost of your use cases on AWS
* **TCO Calculator** - used to compare the cost of running your applications in an on-premises or colocation environment to AWS
* **AWS Pricing Calculator** (NEW- replaces the Simple Monthly Calculator)

**Key Points**

An accurate cost estimation that meets and exceeds your organization’s budgetary goals requires you to ask important questions, interpret data, and implement AWS best practices

**Paid AWS Cloud Services include:**

* Running Compute Resources
* Storage
* Provisioned Databases
* Data Transfer

Remember, you only pay for services you use, and once you stop using them, AWS stops charging you immediately and doesn’t levy any termination fees.

**AWS does not charge for:**

* **AWS Elastic Beanstalk** - Rapid application deployment
* **AWS Cloud Formation** - AWS Branded Infrastructure as Code service
* **Auto-Scaling** - Scaling EC2 instances up/down or in/out based on your application requirements
* **AWS IAM** - User and access management

There is no cost for uploading data into the AWS cloud, although you will pay for storage and data transfer back out. Because of the massive scale of the AWS technology platform, there is no limit to how much data you can upload.



AWS Costs Include Maintenance

**Good Cost Hygiene Practices**

* Establish a naming convention for Servers and Databases
* Use Tags to track costs by:
  + Group
  + Lifecycle
  + Person
  + Application
* Create IT Governance rules
* Set Billing Alarms

**Costs In Regions**

**Key Points**

* AWS costs depend on the region
* Costs vary due to:
  + local regulations
  + cost of doing business
  + competition
* Least costly region is US East 1 in N. Virginia

**Tips for Reducing Costs**

* Use AWS CloudFront to cache data close to end users
* Avoid inter-region data transfer costs
* Peering via AWS Transit Gateway for VPCs reduces costs



US East 1 (N. Virginia)

**New Terms**

| **Term** | **Definition** |
| --- | --- |
| 11 9’s guarantee | 99.999999999% durability of objects over a given year. This durability level corresponds to an average annual expected loss of 0.000000001% of objects |
| Availability Zones | A logical data center in an AWS region with redundant and separate power, networking and connectivity reducing the likelihood of two zones failing simultaneously |
| AWS CloudFront | Fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency |
| AWS Local Zones | A type of AWS infrastructure deployment that places AWS compute, storage, database, and other select services closer to large population, industry, and IT centers where no AWS Region exists today |
| AWS Regions | A geographical location with a collection of availability zones physically isolated from and independent of every other region |
| AWS Support | Paid support plans offering AWS customers access to AWS technical experts and professional guidance |
| Durability | The probability that the object will remain intact and accessible after a period of one year |
| Edge Location | A physical site that CloudFront uses to cache copies of your content for faster delivery to users at any location |
| Fault Tolerance | The property that enables a system to continue operating properly in the event of the failure of (or one or more faults within) some of its components |
| High Availability | Refers to systems that are durable and likely to operate continuously without failure for a long time |
| Memory-Optimized | The R3 instance class recommended for applications that require high memory performance |
| Points of Presence | AWS Edge Locations and Regional Edge Caches used for both AWS CloudFront and Lambda@Edge to deliver content to end users at high speeds |
| Storage-Optimized | Instances are designed for workloads that require high, sequential read and write access to very large data sets on local storage. They are optimized to deliver tens of thousands of low-latency, random I/O operations per second (IOPS) to applications |
| VPC Peering | A networking connection between two AWS VPCs that allows you to route traffic between them using private IP addresses |
| VPC Sharing | allows you to share subnets with other AWS accounts in your organization |

Instance Pricing

**Key Points**

AWS EC2 instance pricing is straightforward, but it can quickly become complex when you take up the task of optimizing your environment to achieve the ideal cost/performance balance.

* Explore OS licensing pricing and options
* Limit the users and roles that can launch production instances
* Choose the best instance for your workload
* Save by moving to new generation instances when available

**Purchasing Options**

When optimizing your computing usage to reduce your monthly spend, one of the first places you want to count your costs is in your EC2 instances, and one of the best resources here is using reserved instances.

| **Option** | **How it Works** |
| --- | --- |
| Capacity Reservations | Reserve capacity for your EC2 instances in a specific Availability Zone for any duration. |
| Dedicated Hosts | Pay for a physical host that is fully dedicated to running your instances, and bring your existing per-socket, per-core, or per-VM software licenses to reduce costs. |
| Dedicated Instances | Pay, by the hour, for instances that run on single-tenant hardware. |
| On-Demand Instances | An AWS service or technology that can be acquired at any time for a predetermined standard cos |
| Reserved Instances | An AWS service or technology that can be reserved for a period of time at a discount in exchange for a payment commitment |
| Savings Plans | Reduces your Amazon EC2 costs by making a commitment to a consistent amount of usage, in USD per hour, for a term of 1 or 3 years. |
| Scheduled Instances | Purchase instances that are always available on the specified recurring schedule, for a one-year term. |
| Spot Instances | an EC2 instance that can be acquired by bidding for a low price in exchange for the understanding that AWS can reclaim it at any time |



AWS EC2 Primary Instance Types

Storage Pricing

**New Terms**

| **Term** | **Definition** |
| --- | --- |
| Edge Cache | A high-speed data storage layer which stores a subset of data, typically transient in nature, close to the end user so that future requests for that data are served up faster |
| EBS | Elastic Block Store - cloud-based and pre-allocated linux-based block storage system provided by Amazon Web Services (AWS) that is best used for storing persistent data |
| EFS | Elastic File System - a simple, scalable, elastic, and encrypted file storage system that can be used by concurrent compute resources in the AWS cloud and on-premises |
| HDD | Category of EBS hard disk drive. Can be throughput optimized (workhorse) or cold (infrequently scanned) |
| Lifecycle Policies | Automate the actions you want to take on an object in an S3 bucket over its lifetime, for example, move them to another storage class, archive them after a number of days or years, or delete them. |
| S3 | Amazon Simple Storage Service is general purpose object storage used for frequently accessed files |
| S3 Endpoints | A private connection between your VPC and S3 that doesn’t require internet access, potentially reducing NAT gateway costs |
| SSD | Category of EBS solid-state drive. Can be IOPS optimized for databases (fast) or general-purpose for boot volumes and dev/test systems |
| Storage Cache | A high-speed data storage layer that stores a subset of typically transient data |
| Tagging | Allows you to name and classify S3 buckets. AWS can provide a usage and cost report based on tags |

**Key Points**

* Serverless allows you to build and run applications without the burden of thinking about servers
* Serverless in AWS means built-in fault tolerance, maintenance, monitoring, and security,
* Serverless allows the enterprise to reclaim the time and money that would be spent on operational tasks, and reduces the number of infrastructure engineers necessary to maintain the production environment



Here is the code I used to create an RDS database from the command line:

aws rds create-db-instance \

--allocated-storage 20 --db-instance-class db.t2.micro \

--db-instance-identifier myinstancename \

--engine postgres \

--master-username myname \

--master-user-password secret99 --no-publicly-accessible

Don't forget to return to the CLI to delete your database!

aws rds delete-db-instance \

--db-instance-identifier myinstancename \

--skip-final-snapshot

Try it out with a simple terraform file like the example shown below. Name the file terraform.tf and run it from a working directory in your terraform root folder.

provider "aws" {

access\_key = "<Your Access Key>"

secret\_key = "<Your Secret Key>"

region = "us-east-1"

}

resource "aws\_instance" "example" {

ami = "ami-0742b4e673072066f"

instance\_type = "t2.micro"

}

Run terraform init, terraform plan, terraform apply, and terraform destroy and observe the results in the AWS EC2 Management console.

**My Exercise Solution**

Here is the code for the main.tf file I used to create AWS infrastructure with Terraform:

provider "aws" {

access\_key = "<Your Access Key>"

secret\_key = "<Your Secret Key>"

region = "us-east-1"

}

resource "aws\_instance" "Udacity" {

count = "2"

ami = "ami-0323c3dd2da7fb37d"

instance\_type = "t2.small"

tags = {

name = "Udacity Terraform"

}

}

**Demo Code**

Here is the code I used to create the backend.tf file. You'll need to update it with the name of your S3 bucket and the path to your terraform.tfstate file.

terraform {

backend "s3" {

bucket = "<Name of your S3 bucket>"

key = "<Path To Your terraform.tfstate file>"

region = "us-east-1"

}

}

Here is the code for terraform.tf used to create the S3 backend. You can model yours off my example, or be creative and create your own- just make sure you destroy any infrastructure you create!

provider "aws" {

access\_key = "<Your Access Key>"

secret\_key = "<Your Secret Key>"

region = "us-east-1"

}

resource "aws\_instance" "Backend" {

count = "2"

ami = "ami-0323c3dd2da7fb37d"

instance\_type = "t2.micro"

}

Save your backend.tf file with your terraform.tf (or main.tf file) in a working directory under your Terraform root directory. In the demo, the working directory is ~/terraform/udacity\_s3