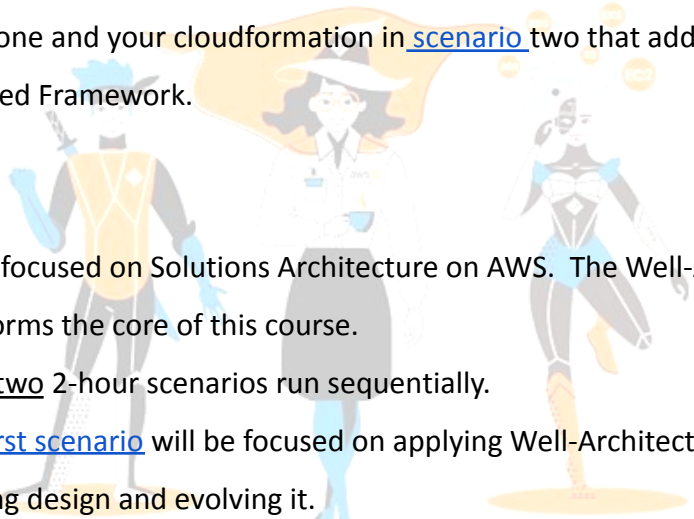
AWS Sims for
Solutions Architects
Lab Guide

Welcome!

Greetings and Welcome! You are participating in an AWS Simulation known as AWSims focused on AWS Solutions Architecture. In this scenario challenge event, your job will be to read through two scenarios and achieve the objective of those scenarios using Core AWS Services and Design Concepts. Since this is Architecture focused, there will be two scenarios. One will be focused on design in the cloud and enhancing designs in the cloud. The other will be about implementing corrections to those designs in mock architectures up and running in AWS. Both scenarios will use the AWS Well-Architected Framework principles as listed in detail after this page. The specific scenarios will only be shown to you in a separate document as part of taking the course and will vary slightly from course to course.

Setup and Foundations for both scenarios

This class intends to take a different approach when it comes to learning by challenging intermediate and advanced architects to put the AWS Well-Architected Framework into action by both correcting and creating architectures that adhere to the Cloud Native principles of the AWS Well-Architected framework. You will be put into 4-6 person teams and work together to achieve the desired outcome, which in all cases are architectures that are Secure, Reliable, Operationally Excellent, Cost Optimized, and Performance Efficient. You will have an hour to complete each scenario with your team. Teams are competing for 250 points per scenario for a total of 500 points for the day. Grading is based on adding specific features and services to your designs for scenario one and your cloudformation in scenario two that add one or more aspects of the Well-Architected Framework.



In summary:

1. The course is focused on Solutions Architecture on AWS. The Well-Architected Framework forms the core of this course.
2. It consists of two 2-hour scenarios run sequentially.
    a. The first scenario will be focused on applying Well-Architected principles to an existing design and evolving it.
    b. The second scenario will focus on implementing corrections to a running architecture via CloudFormation
3. In each 2-hour scenario, you will have 30 minutes to prep, an hour to execute, and 30 minutes of post-mortem activity and review.
4. Each scenario will have outputs that must be checked into your git repository. Outputs will either be design documents or code artifacts via CloudFormation.
5. You will be put into 4-6 person teams and work with those same teams on both scenarios.

6. Once your hour to execute is over, your team’s git repository will be graded and evaluated against the scenario's objectives. Teams will be ranked and placed on the leaderboard.

AWS Well-Architected Framework - General Design Principles

The Well-Architected Framework identifies a set of general design principles to facilitate good design in the cloud:

- **Stop guessing your capacity needs**
- **Test systems at production scale**:
- **Automate to make architectural experimentation easier**
- **Allow for evolutionary architectures**:
- **Drive architectures using data**:
- **Improve through game days**:

It consists of Pillars which are the primary items that we will be reviewing and applying to our first scenario and implementing in our second scenario.

Next will be the general principles of the well-architected framework.

The Pillars of the Well-Architected Framework

Creating a software system is a lot like constructing a building. If the foundation is not solid, structural problems can undermine the integrity and function of the building. When architecting technology solutions, if you neglect the six pillars of operational excellence, security, reliability, performance efficiency, and cost optimization, you will create a non-optimized cloud architecture. Note that we are not covering the relatively new pillar of sustainability. It can become challenging to build a system that delivers your expectations and requirements. Incorporating these pillars into your architecture will help you produce stable and efficient systems. This will allow you to focus on the other design aspects, such as functional requirements.

Pillars

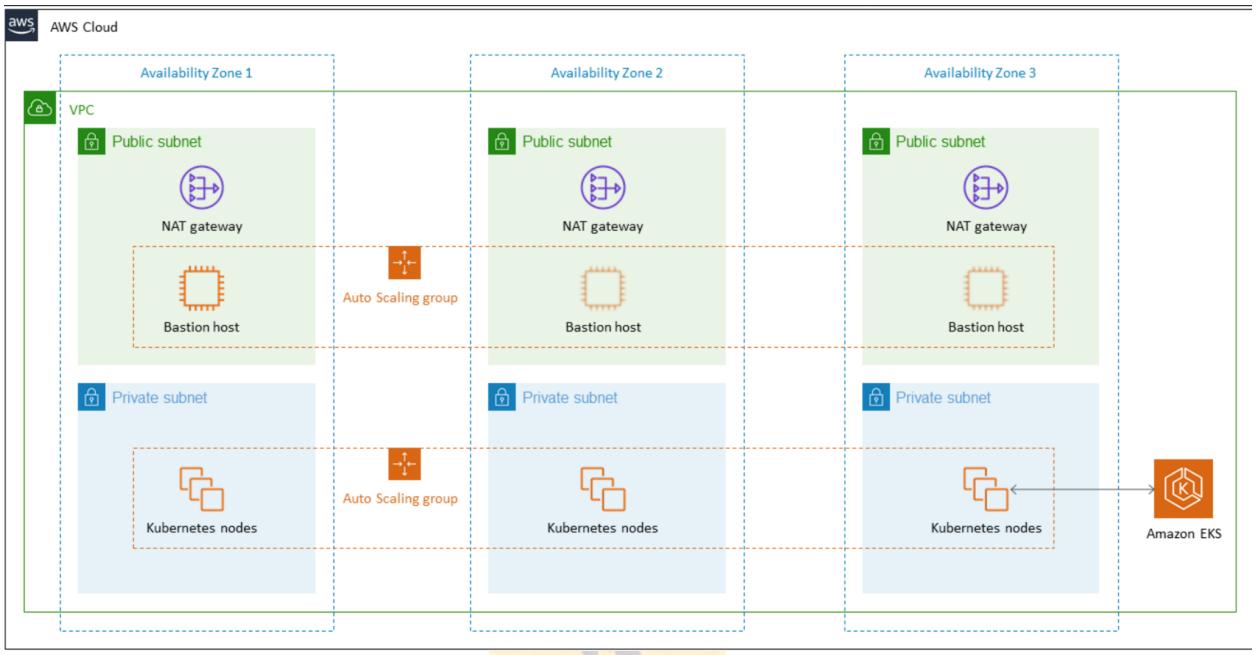
- Operational Excellence
- Security
- Reliability
- Performance Efficiency
- Cost Optimization

We will lay out the basic questions below that we want you to consider in the specific scenario for each pillar.

Scenario 1 - Design for the AWS Well-Architected Framework

In this scenario, you will be provided with a single design and questions covering the first five pillars of the well-architected framework. You will modify the design at each Phase, ensuring that your design evolves as you enhance security, reliability, etc. EACH time your team completes a phase in a scenario, you will check your design document into your respective Git repositories

An example design is rendered here.



- ## Phase 1 - Security

Security Design Principles

There are seven design principles for security in the cloud:

- **Implement a strong identity foundation**: Implement the principle of least privilege and enforce separation of duties with the appropriate authorization for each interaction with your AWS resources. Centralize identity management, and aim to eliminate reliance on long-term static credentials.

- **Enable traceability**: Monitor, alert, and audit actions and changes to your environment in real-time. Integrate log and metric collection with systems to investigate and take action automatically.

- **Apply security at all layers**: Apply a defense in depth approach with multiple security controls. Apply to all layers (for example, edge of the network, VPC, load balancing, every instance and compute service, operating system, application, and code).

- **Automate security best practices**: Automated software-based security mechanisms improve your ability to scale more rapidly and cost-effectively securely. Create secure architectures, including implementing controls that are defined and managed as code in version-controlled templates.

- **Protect data in transit and at rest**: Classify your data into sensitivity levels and use mechanisms such as encryption, tokenization, and access control where appropriate.

- **Keep people away from data**: Use mechanisms and tools to reduce or eliminate the need for direct access or manual processing of data. This reduces the risk of mishandling, modification, and human error when handling sensitive data.

- **Prepare for security events**: Prepare for an incident by having incident management and investigation policy and processes that align with your organizational requirements. Run incident response simulations and use tools with automation to increase your speed for detection, investigation, and recovery.

Security Evaluation Questions:

1. How do you securely operate your workloads on AWS?
2. How do you manage identities for people and machines?
3. How do you manage permissions for people and machines?
4. What Services are you using to detect and remediate security events?
    a. Application Hosting?
    b. Storage?

    c. Networking?
    d. Datastores?
    e. Infrastructure?
5. How do you protect your network resources?
6. How do you protect your compute resources?
7. How do you classify your data?
8. How do you protect data at rest?
9. How do you protect data in transit?

Apply these to your design. Update the design. Commit the design to the repository.

- ## Phase 2 - Reliability

Reliability Design Principles

There are five design principles for reliability in the cloud:

- **Automatically recover from failure**: You can trigger automation when a threshold is breached by monitoring a workload for key performance indicators (KPIs). These KPIs should be a measure of business value, not the technical aspects of the operation of the service. This allows automatic notification and tracking of failures and automated recovery processes that work around the failure.

- **Test recovery procedures**: In an on-premises environment, testing is often conducted to prove that the workload works in a particular scenario. Testing is not typically used to validate recovery strategies. You can test how your workload fails in the cloud and validate your recovery procedures. You can use automation to simulate different failures or recreate scenarios that led to failures. This approach exposes failure pathways that you can test and fix before an actual failure scenario occurs.

- **Scale horizontally to increase aggregate workload availability**: Replace one significant resource with multiple small resources to reduce the impact of a single failure on the overall workload. Distribute requests across multiple, smaller resources to ensure that they don't share a common point of failure.

- **Stop guessing capacity**: A common cause of failure in on-premises workloads is resource saturation when the demands placed on a workload exceed the capacity of that workload (this is often the objective of denial of service attacks). In the cloud, you can monitor demand and workload utilization, and automate the addition or removal of resources to maintain the optimal level to satisfy demand without over-or under-provisioning. There are still limits, but some quotas can be controlled, and others can be managed (see Manage Service Quotas and Constraints).

- **Manage change in automation**: Changes to your infrastructure should be made using automation. The changes that need to be managed include changes to the automation, which then can be tracked and reviewed.

Reliability Evaluation Questions

1. How do you manage service quotas or constraints?
2. How do you plan your network topology?
3. How do you design your workload service architecture?
4. How do you design interactions in a distributed system to prevent failure?
5. How do you design interactions in a distributed system to mitigate or withstand failures?
6. How do you monitor workload progress?

7. How do you design your workload to adapt to changes in demand?
8. How do you implement change?
9. How do you backup data?
10. How do you use fault isolation to protect your workload?
11. How do you test reliability?
12. How do you plan for disaster recovery?

Apply these to your design. Update the design. Commit the design to the repository.

- ## Phase 3 - Performance Efficiency

Performance Efficiency Design Principles

There are five design principles for performance efficiency in the cloud:

- **Democratize advanced technologies**: Make advanced technology implementation easier for your team by delegating complex tasks to your cloud vendor. Consider consuming the technology as a service rather than asking your IT team to learn about hosting and running a new technology. For example, NoSQL databases, media transcoding, and machine learning are all technologies that require specialized expertise. These technologies become services that your team can consume in the cloud, allowing your team to focus on product development rather than resource provisioning and management.

- **Go global in minutes**: Deploying your workload in multiple AWS Regions worldwide allows you to provide lower latency and a better experience for your customers at a minimal cost.

- **Use serverless architectures**: Serverless architectures remove the need for you to run and maintain physical servers for traditional compute activities. For example, serverless storage services can act as static websites (removing the need for web servers), and event services can host code. This removes the operational burden of managing physical servers and can lower transactional costs because managed services operate at a cloud-scale.

- **Experiment more often**: With virtual and automatable resources, you can quickly carry out comparative testing using different types of instances, storage, or configurations.

- **Consider mechanical sympathy**: Understand how cloud services are consumed and always use the technology approach that aligns best with your workload goals. For example, consider data access patterns when selecting database or storage approaches.

Performance Efficiency Evaluation Questions

1. How do you select the best-performing architecture?
2. How do you select the best compute solution?
3. How do you select your storage solution?
4. How do you select your database solution?
5. How do you configure your networking solution?
6. How do you evolve your workload to take care of new releases?
7. How do you monitor your resources to ensure they are performing?
8. How do you use tradeoffs to improve performance?

Apply these to your design. Update the design. Commit the design to the repository.

- ## Phase 4 - Operational Excellence

Operational Excellence Design Principles

There are five design principles for operational excellence in the cloud:

- **Perform operations as code**: In the cloud, you can apply the same engineering discipline you use for application code to your entire environment. You can define your total workload (applications, infrastructure) as code and update it with code. You can implement your operations procedures as code and automate their execution by triggering them in response to events. By performing operations as code, you limit human error and enable consistent responses to events.

- **Make frequent, small, reversible changes**: Design workloads to allow components to be updated regularly. Make changes in small increments that can be reversed if they fail (without affecting customers).

- **Refine operations procedures frequently**: As you use operations procedures, look for opportunities to improve them. As you evolve your workload, evolve your procedures appropriately. Set up regular game days to review and validate that all procedures are effective and that teams are familiar with them.

- **Anticipate failure**: Perform "premortem" exercises to identify potential sources of failure so that they can be removed or mitigated. Test your failure scenarios and validate your understanding of their impact. Test your response procedures to ensure that they are effective and that teams are familiar with their execution. Set up regular game days to test workloads and team responses to simulated events.

- **Learn from all operational failures**: Drive improvement through lessons learned from all operational events and failures. Share what is learned across teams and through the entire organization.

Operational Excellence Evaluation Questions

1. How do you determine what your priorities are?
2. How do you structure your organization to support your business outcomes?
3. How does your organizational culture support your business outcomes?
4. How do you design your workload so you can understand its state?
5. How do you reduce defects, ease remediation, and improve flow into production?
6. How do you mitigate deployment risks?
7. How do you know that you are ready to support a workload?
8. How do you understand the health of your workload?
9. How do you understand the health of your operations?
10. How do you manage workload and Operations Events?

11. How do you evolve operations?

Apply these to your design. Update the design. Commit the design to the repository.

- ## Phase 5 - Cost Optimization

Cost Optimization Design Principles

There are five design principles for cost optimization in the cloud:

- **Implement Cloud Financial Management**: To achieve financial success and accelerate business value realization in the cloud, you need to invest in Cloud Financial Management /Cost Optimization. Your organization needs to dedicate time and resources to build capability in this new domain of technology and usage management. Similar to your Security or Operational Excellence capability, you need to build capability through knowledge building, programs, resources, and processes to become a cost-efficient organization.

- **Adopt a consumption model**: Pay only for the computing resources that you require and increase or decrease usage depending on business requirements, not by using elaborate forecasting. For example, development and test environments are typically only used for eight hours a day during the workweek. You can stop these resources when they are not in use for a potential cost savings of 75% (40 hours versus 168 hours).

- **Measure overall efficiency**: Measure the business output of the workload and the costs associated with delivering it. Use this measure to know the gains you make from increasing output and reducing costs.

- **Stop spending money on undifferentiated heavy lifting**: AWS does the heavy lifting of data center operations like racking, stacking, and powering servers. It also removes the operational burden of managing operating systems and applications with managed services. This allows you to focus on your customers and business projects rather than on IT infrastructure.

- **Analyze and attribute expenditure**: The cloud makes it easier to accurately identify the usage and cost of systems, which then allows transparent attribution of IT costs to individual workload owners. This helps measure return on investment (ROI) and gives workload owners an opportunity to optimize their resources and reduce costs.

Cost Optimization Evaluation Questions

1. How do you implement cloud financial management?
2. How do you govern usage?
3. How do you monitor usage and costs?
4. How do you evaluate costs when you select services?
5. How do you meet cost targets when you select resource type, size, and number?
6. How do you use pricing models to reduce costs?
7. How do you plan for data transfer charges?
8. How do you manage demand and supply resources

9. How do you evaluate new services?

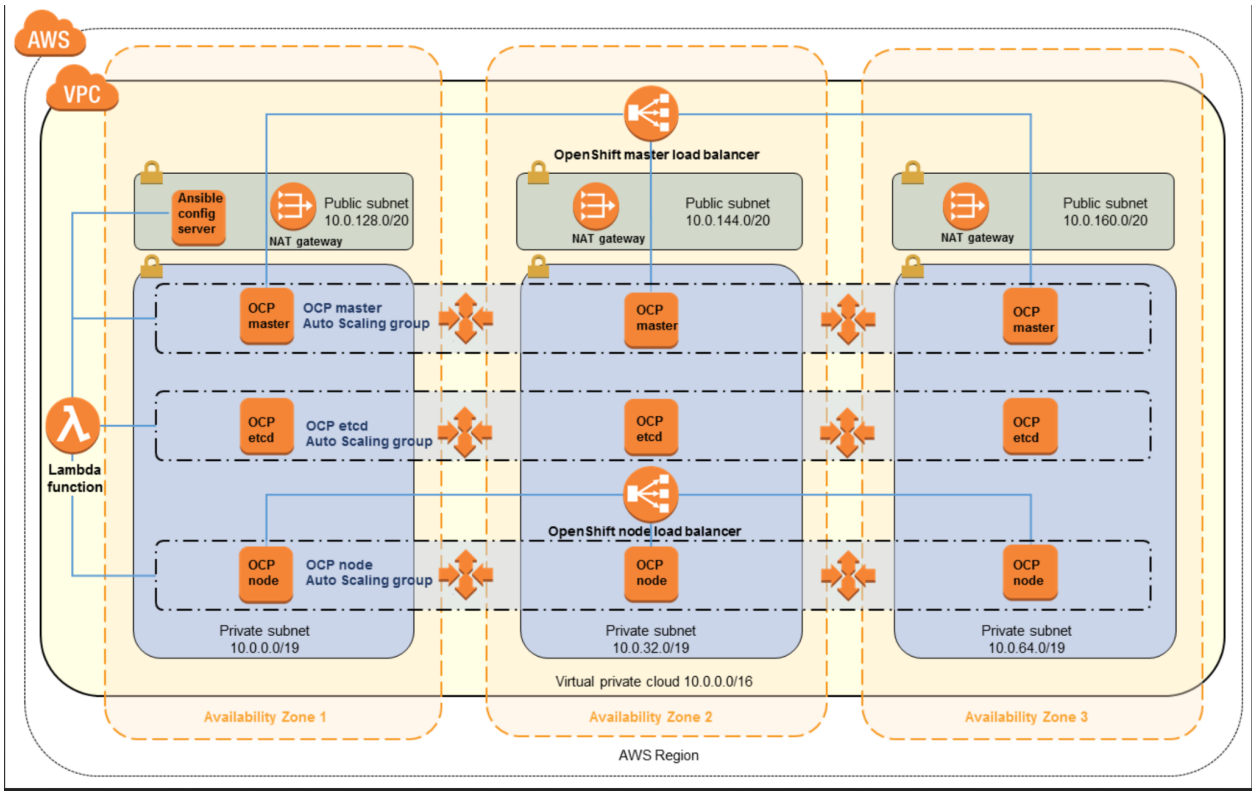
Apply these to your design. Update the design. Commit the design to the repository.

Scenario 2 - Build for the AWS Well-Architected Framework

In this scenario, you will implement at least one change from each of your suggested design changes in Phase 1 into your actual running system.

EACH time your team completes a phase you will check your design into your respective Git repositories

An example design is rendered below. Example Cloudformation can be found here.

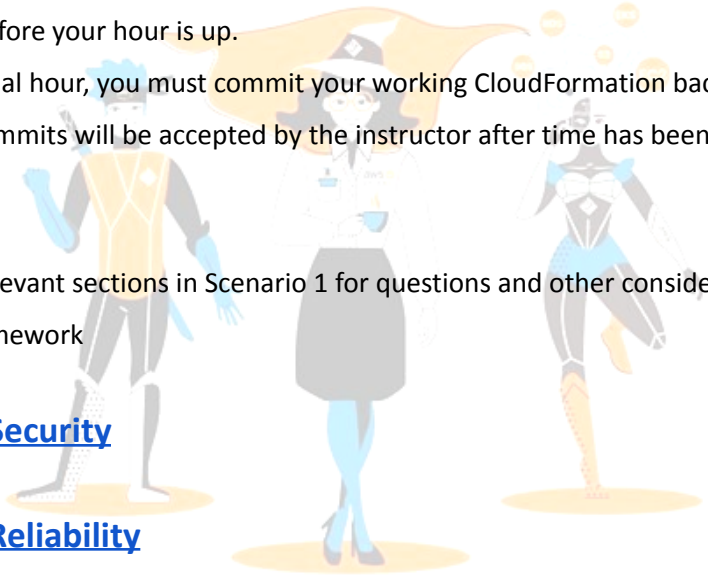


You will commit your cloudformation each time your team adds a WORKING feature to your cloudformation code. For example, if you have added Autoscaling to your EC2 virtual machines and it is tested and working in cloudformation then you will receive points. If your feature is broken, then no points will be awarded. Your goal is to add as many working features to your design via Cloudformation based on the Five Pillars that we have covered in the Well-Architected Framework.

Note that your goal here is to have a working cloudformation template/section for each feature that you add to the system. This can be one file. If the system does not work (meaning it breaks) or the CloudFormation doesn't load into another account, then that feature does not count for points. Remember quality over quantity. Also simplicity over complexity. You will not get extra points for a complex change versus a simple one.

In summary:

- You will all be presented with an AWS account for your team along with an architecture that has been implemented into that AWS account via CloudFormation.
- Your team will be presented with a GitHub repository that contains your respective cloudformation template. You will clone and modify that template, committing it back to the repository before your hour is up.
- Before the final hour, you must commit your working CloudFormation back to your git repo. No additional commits will be accepted by the instructor after time has been called.



Please refer to the relevant sections in Scenario 1 for questions and other considerations for the Well-Architected Framework

- **Phase 1 - Security**
- **Phase 2 - Reliability**
- **Phase 3 - Performance Efficiency**
- **Phase 4 - Operational Excellence**
- **Phase 5 - Cost Optimization**