**QUESTION 88**

A company is building a containerized application on-premises and decides to move the application to AWS.

The application will have thousands of users soon after li is deployed.

The company Is unsure how to manage the deployment of containers at scale.

The company needs to deploy the containerized application in a highly available architecture that minimizes operational overhead.

Which solution will meet these requirements?

1. Store container images In an Amazon Elastic Container Registry (Amazon ECR) repository.

Use an Amazon Elastic Container Service (Amazon ECS) cluster with the AWS Fargate launch type to run the containers.

Use target tracking to scale automatically based on demand.

1. Store container images in an Amazon Elastic Container Registry (Amazon ECR) repository.

Use an Amazon Elastic Container Service (Amazon ECS) cluster with the Amazon EC2 launch type to run the containers.

Use target tracking to scale automatically based on demand.

1. Store container images in a repository that runs on an Amazon EC2 instance.

Run the containers on EC2 instances that are spread across multiple Availability Zones.

Monitor the average CPU utilization in Amazon CloudWatch.

Launch new EC2 instances as needed.

1. Create an Amazon EC2 Amazon Machine Image (AMI) that contains the container image.

Launch EC2 Instances in an Auto Scaling group across multiple Availability Zones.

Use an Amazon CloudWatch alarm to scale out EC2 instances when the average CPU utilization threshold is breached.

**Answer:** A

**QUESTION 131**

A company wants to run its critical applications in containers to meet requirements tor scalability and availability.

The company prefers to focus on maintenance of the critical applications.

The company does not want to be responsible for provisioning and managing the underlying infrastructure that runs the containerized workload.

What should a solutions architect do to meet those requirements?

1. Use Amazon EC2 Instances, and Install Docker on the Instances.

1. Use Amazon Elastic Container Service (Amazon ECS) on Amazon EC2 worker nodes.
2. Use Amazon Elastic Container Service (Amazon ECS) on AWS Fargate.
3. Use Amazon EC2 instances from an Amazon Elastic Container Service (Amazon ECS)-optimized Amazon Machine Image (AMI).

**Answer:** C

**Explanation:**

using AWS ECS on AWS Fargate since they requirements are for scalability and availability without having to provision and manage the underlying infrastructure to run the containerized workload.

<https://docs.aws.amazon.com/AmazonECS/latest/userguide/what-is-fargate.html>

**QUESTION 187**

An ecommerce website is deploying its web application as Amazon Elastic Container Service (Amazon ECS) container instance behind an Application Load Balancer (ALB).

During periods of high activity, the website slows down and availability is reduced.

A solutions architect uses Amazon CloudWatch alarms to receive notifications whenever there is an availability issues so they can scale out resource Company management wants a solution that automatically responds to such events.

Which solution meets these requirements?

1. Set up AWS Auto Scaling to scale out the ECS service when there are timeouts on the ALB.

Set up AWS Auto Scaling to scale out the ECS cluster when the CPU or memory reservation is too high.

1. Set up AWS Auto Scaling to scale out the ECS service when the ALB CPU utilization is too high.

Set up AWS Auto Scaling to scale out the ECS cluster when the CPU or memory reservation is too high.

1. Set up AWS Auto Scaling to scale out the ECS service when the service's CPU utilization is too high.

Set up AWS Auto Scaling to scale out the ECS cluster when the CPU or memory reservation is too high.

1. Set up AWS Auto Scaling to scale out the ECS service when the ALB target group CPU utilization is too high.

Set up AWS Auto Scaling to scale out the ECS cluster when the CPU or memory reservation is too high.

**Answer:** C

**Explanation:**

Match deployed capacity to the incoming application load, using scaling policies for both the ECS service and the Auto Scaling group in which the ECS cluster runs. Scaling up cluster instances and service tasks when needed and safely scaling them down when demand subsides, keeps you out of the capacity guessing game. This provides you high availability with lowered costs in the long run.

<https://aws.amazon.com/blogs/compute/automatic-scaling-with-amazon-ecs/>

**QUESTION 248**

A company runs a containerised application on a Kubernetes cluster in an on-premises data center.

The company is using a MongoDB database for data storage.

The company wants to migrate some of these environments to AWS, but no code changes or deployment method changes ate possible at this time.

The company needs a solution that minimizes operational overhead.

Which solution meets these requirements?

1. Use Amazon Elastic Container Service (Amazon ECS) with Amazon EC2 worker nodes for compute and MongoDB on EC2 for data storage.

1. Use Amazon Elastic Container Service (Amazon ECS) with AWS Fargate for compute and Amazon DynamoDB tor data storage.
2. Use Amazon Elastic Kubernetes Service (Amazon EKS) with Amazon EC2 worker nodes for compute and Amazon DynamoDB for data storage.
3. Use Amazon Elastic Kubernetes Service (Amazon EKS) with AWS fargate for compute and Amazon DocumentDB (with MongoDB compatibility) for data storage.

**Answer:** D

**Explanation:**

Amazon DocumentDB (with MongoDB compatibility) is a fast, reliable, and fully managed database service. Amazon DocumentDB makes it easy to set up, operate, and scale MongoDBcompatible databases in the cloud. With Amazon DocumentDB, you can run the same application code and use the same drivers and tools that you use with MongoDB.

<https://docs.aws.amazon.com/documentdb/latest/developerguide/what-is.html>