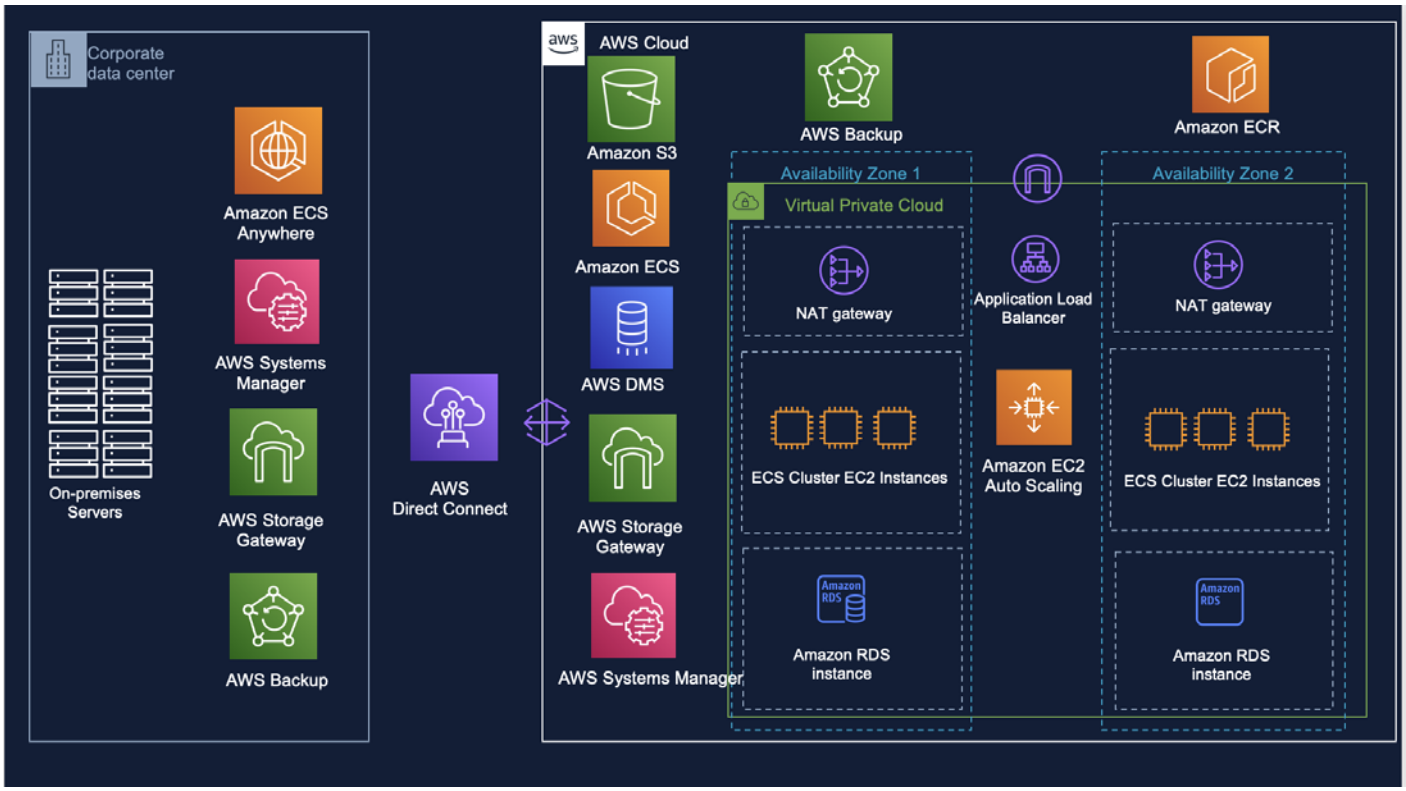
Architecture Optimizations for Week 3

In this reading, you will find further information about the topics that Morgan and Raf talked in the video where they both played solutions architects.

The following diagram is a more detailed view of the solution that was covered this week.

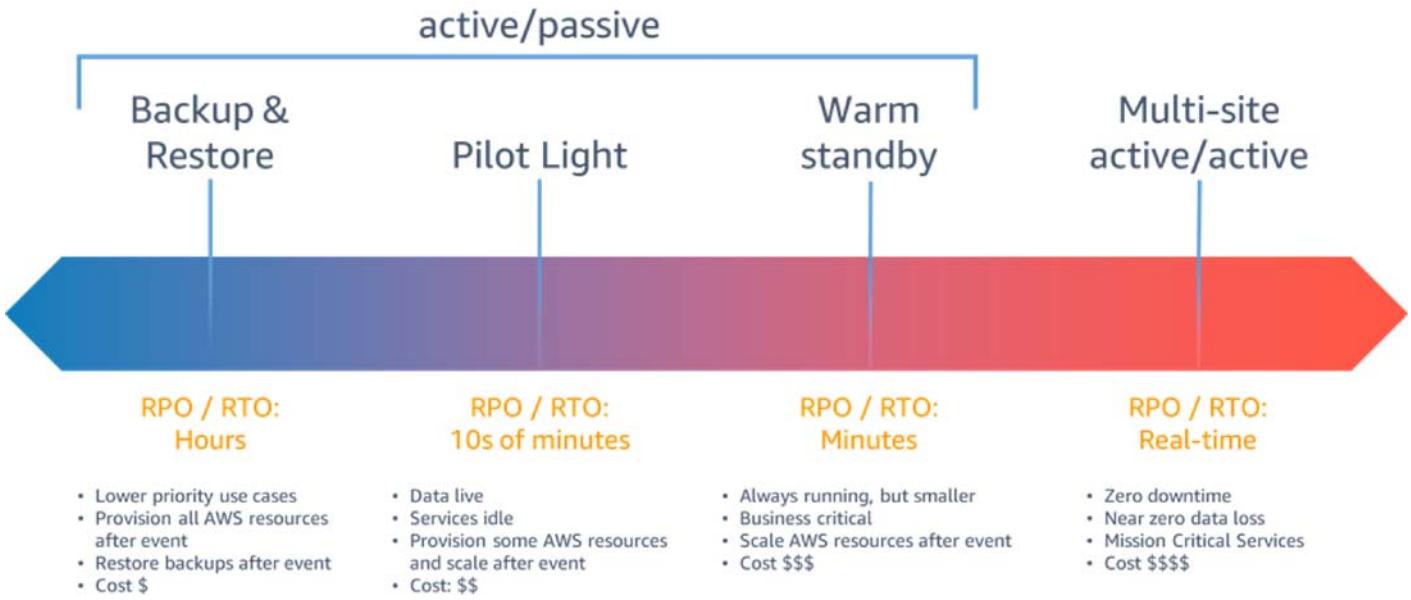


Disaster recovery

Morgan mentioned that because this week's customer is an enterprise company, they could consider designing a disaster recovery (DR) plan for this workload.

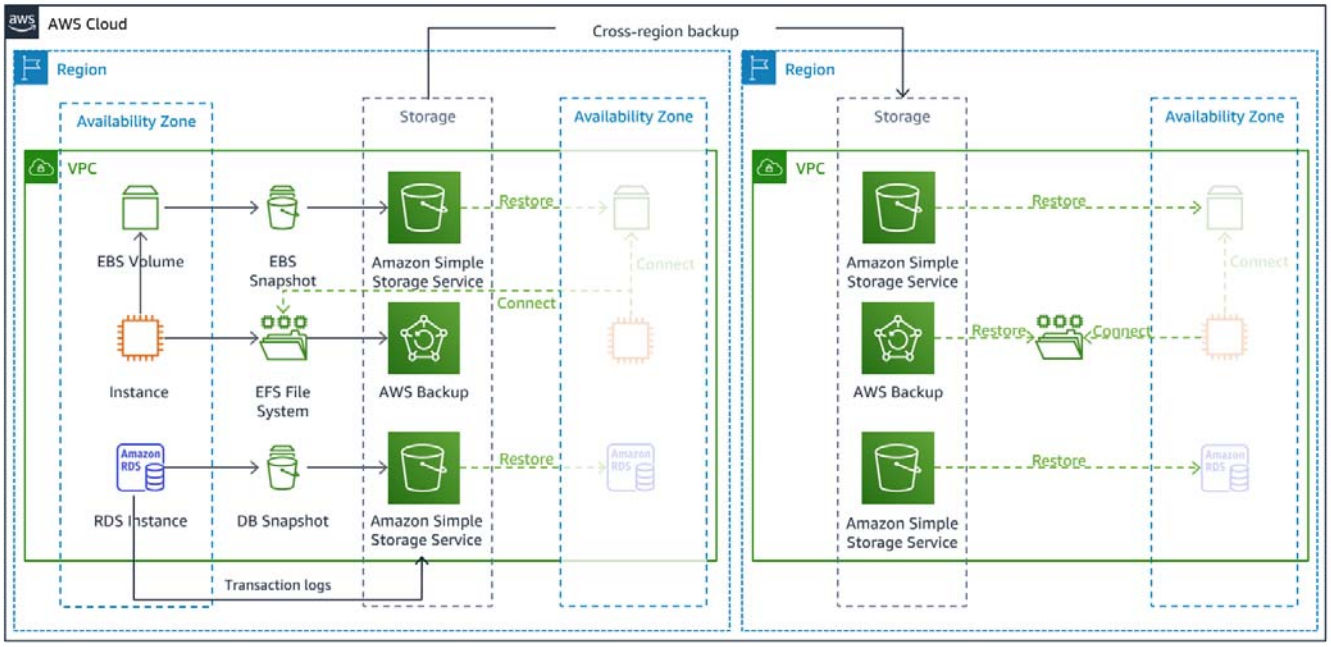
The DR strategies that are available to you within AWS can be broadly categorized into four approaches, which range from the low cost and low complexity of making backups to more complex strategies that use multiple active Regions. Active/passive strategies use an active site (such as an AWS Region) to host the workload and serve traffic. The passive site (such as a different AWS Region) is used for recovery. The passive site doesn't actively serve traffic until a failover event is triggered.

The following diagram outlines the different approaches to disaster recovery on AWS.



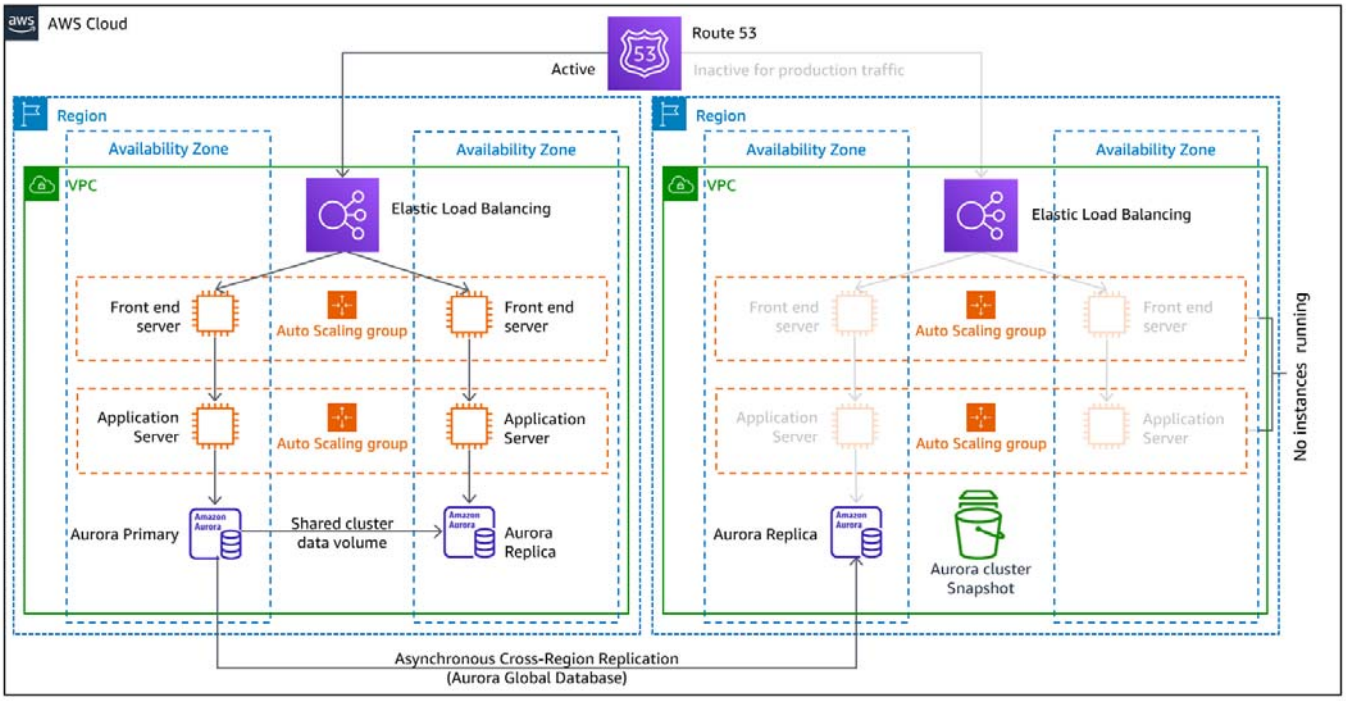
Backup and restore

Backup and restore is a suitable approach for mitigating against data loss or corruption. This approach can also be used to mitigate against a regional disaster by replicating data to other AWS Regions, or to mitigate a lack of redundancy for workloads that are deployed to a single Availability Zone. In addition to data, you must redeploy the infrastructure, configuration, and application code in the recovery Region. To enable infrastructure to be redeployed quickly without errors, you should always deploy by using infrastructure as code (IaC), with services such as AWS CloudFormation or the AWS Cloud Development Kit (AWS CDK). Without IaC, it might be complex to restore workloads in the recovery Region, which can lead to increased recovery times and possibly exceed your Recovery Time Objective (RTO). In addition to backing up your user data, back up your code and configurations—including Amazon Machine Images (AMIs) that you use to create Amazon Elastic Compute Cloud (Amazon EC2) instances. You can use AWS CodePipeline to automate the redeployment of your application code and configurations.



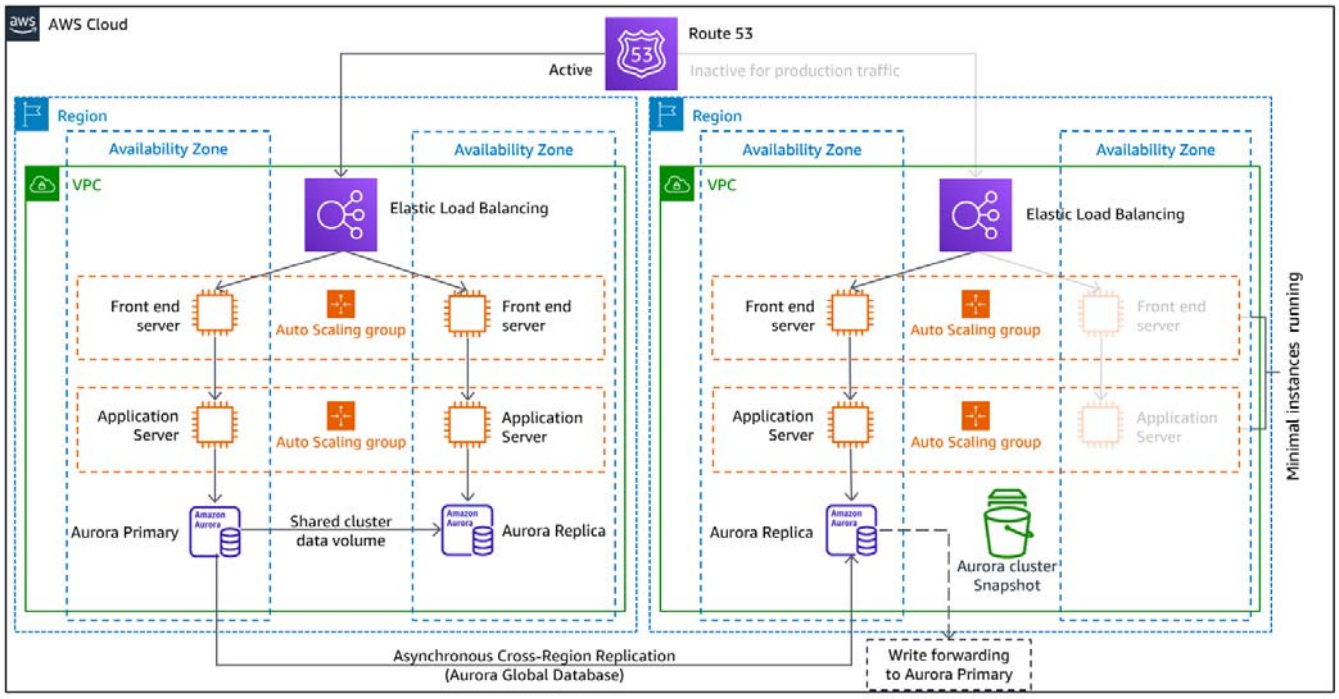
Pilot light

With the pilot light approach, you replicate your data from one Region to another, and you provision a copy of your core workload infrastructure. Resources that are needed to support data replication and backup—such as databases and object storage—are always on. Other elements (such as application servers) are loaded with application code and configurations, but are turned off. They are only used during testing or when DR failover is invoked. In the cloud, you have the flexibility to deprovision resources when you don't need them, and provision them when you do. A best practice for a resource that's turned off is to not deploy the resource, and then create the configuration and capabilities to deploy it (turn on) when needed. Unlike the backup and restore approach, your core infrastructure is always available. Thus, you always have the option to quickly provision a full scale production environment by turning on and scaling out your application servers.



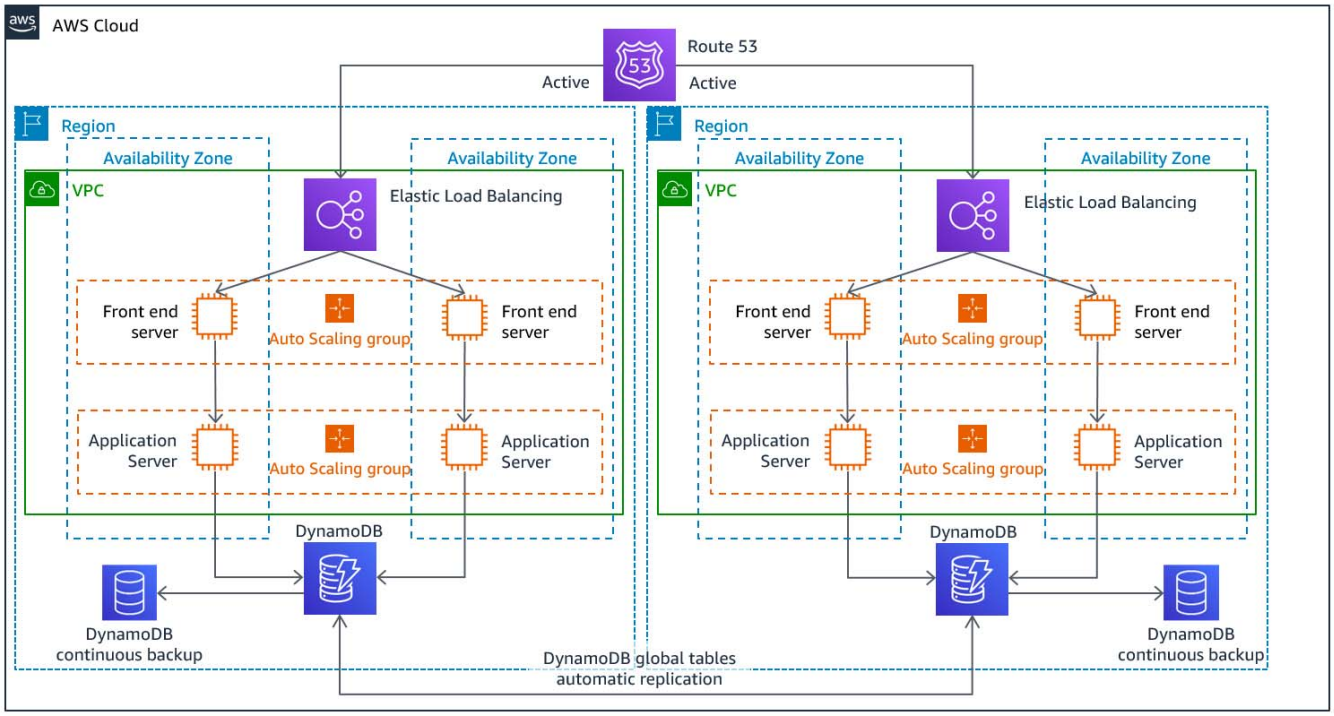
Warm standby

The warm standby approach involves provisioning a scaled down—but fully functional—copy of your production environment in another Region. This approach extends the pilot-light concept and decreases the time to recovery because your workload is always-on in another Region. With this approach, you can perform testing or implement continuous testing more easily, which can increase your confidence in your ability to recover from a disaster.



Multi-site active/active

You can run your workload simultaneously in multiple Regions as part of either a multi-site active/active strategy or a hot standby active/passive strategy. A multi-site active/active approach serves traffic from all Regions where it's deployed. In contrast, hot standby approach serves traffic from only a single Region, and the other Regions are used only for DR. With a multi-site active/active approach, users can access your workload in any Regions where it's deployed. This approach to DR is the most complex and most costly. However, it can reduce your recovery time to near zero for most disasters, with the correct technology choices and implementation. (Note that data corruption might need to rely on backups, which usually results in a non-zero recovery point.) Hot standby uses an active/passive configuration, where users are directed to only a single region and DR Regions don't take traffic. Most customers find that if they're going to stand up a full environment in the second Region, it makes sense to use it in an active/active approach. Alternatively, if you don't want to use both Regions to handle user traffic, then warm standby offers an approach that's more economical and operationally less complex.



For more information about DR, see the Disaster Recovery of Workloads on AWS: Recovery in the Cloud whitepaper.

AWS Direct Connect with AWS VPN for failover

Raf commented that to make the connection to AWS more resilient, the customer can consider using Amazon Site-to-Site VPN as a failover for AWS Direct Connect so that the connection is redundant. This setup would protect the customer if the AWS Direct Connect connection became unavailable. With this setup, the customer would be able to fail over to the virtual private network (VPN) connection and remain connected to their AWS resources.

For more information about how to set up Site-to-Site VPN as a failover for Direct Connect, see How do I configure Direct Connect and VPN failover with Transit Gateway?

Automatic scaling for containers

Morgan commented that this week's customer needs to look into how they will scale their containers. When the customer uses Amazon Elastic Container Service (Amazon ECS), they will need to think about scaling both their underlying EC2 cluster and the containers themselves.

Scaling the cluster

Amazon ECS can manage the scaling of EC2 instances that are registered to your cluster. This capability is referred to as *Amazon ECS cluster auto scaling*, and is performed by an Amazon ECS Auto Scaling group capacity provider that has managed scaling turned on.

When you use an Auto Scaling group capacity provider with managed scaling, Amazon ECS creates two custom Amazon CloudWatch metrics, and a target tracking scaling policy that attaches to your Auto Scaling group. Amazon ECS then manages the scale-in and scale-out actions of the Auto Scaling group based on the load that your tasks put on your cluster.

Scaling the containers

Automatic scaling is the ability to increase or decrease the desired count of tasks in your Amazon ECS service automatically. Amazon ECS uses the Application Auto Scaling service to provide this functionality.

Your Amazon ECS service can be optionally configured automatically scale its desired count of tasks in your Amazon ECS service, either up or down, in response to CloudWatch alarms.Amazon ECS Service Auto Scaling supports the following types of scaling policies:

- Target tracking scaling policies (Recommended): Increase or decrease the number of tasks that your service runs, based on a target value for a specific metric. This scaling approach is similar to the way that your thermostat maintains the temperature of your home. You select the temperature, and the thermostat manages the temperature.
- Step scaling policies: Increase or decrease the number of tasks that your service runs, based on a set of scaling adjustments, which are also known as step adjustments. These adjustments vary based on the size of the alarm breach.

For more resources about automatic scaling and Amazon ECS, see the following:

- For more information about how to scale an ECS cluster, see Amazon ECS cluster Auto Scaling.
- For a hands-on tutorial about how to scale an ECS cluster, see the Amazon ECS Workshop: Deploy ECS Cluster Auto Scaling.

- For more information about service auto scaling, see Service auto scaling in the *Amazon ECS Developer Guide*.

Automatic scaling for Amazon RDS

Morgan commented that this week's customer should look into RDS storage autoscaling.

When you enable storage autoscaling, Amazon RDS automatically scales up your storage when it detects that you are running out of free database space.

If your workload is unpredictable, you can enable storage autoscaling for an RDS DB instance. To do so, you can use the Amazon RDS console, the Amazon RDS API, or the AWS Command Line Interface (AWS CLI).

For more information about Amazon RDS storage autoscaling, see Managing capacity automatically with Amazon RDS storage autoscaling.

Amazon S3 Intelligent-Tiering

Raf commented that this customer could also optimize for cost by using S3 Intelligent-Tiering on the S3 bucket where their data is being stored.

S3 Intelligent-Tiering is the only cloud storage class that delivers automatic storage cost savings when data access patterns change, thus reducing performance impact or operational overhead. The Amazon S3 Intelligent-Tiering storage class is designed to optimize storage costs by automatically moving data to the most cost-effective access tier when access patterns change. For a small monthly object monitoring and automation charge, S3 Intelligent-Tiering monitors access patterns and automatically moves objects that haven't been accessed to lower-cost access tiers.

S3 Intelligent-Tiering is a good storage class for data with unknown, changing, or unpredictable access patterns—independent of object size or retention period. You can use S3 Intelligent-Tiering as the default storage class for virtually any workload, especially data lakes, data analytics, new applications, and user-generated content.

For more information about S3 Intelligent-Tiering, see Amazon S3 Intelligent-Tiering storage class.