Writeup:

set Up and Monitor a WordPress Instance

Objective: To set up and monitoring a WordPress instance for your organization is to establish a reliable and secure online presence that supports your business or organizational goals

1. Creating a CloudFormation Stack

The first task in the project was to create a CloudFormation stack to automate the setup of necessary resources on AWS. This included provisioning a Virtual Private Cloud (VPC), subnets, internet gateway, and security groups to allow secure communication between resources.

The CloudFormation stack uses YAML to define the AWS resources. Below is the basic structure:

```
AWSTemplateFormatVersion: '2010-09-09'Description: Deploy WordPress with Auto Scaling and Scheduled Shutdown in us-east-1
Parameters:
  KeyName:
    Type: AWS::EC2::KeyPair::KeyName
    Description: Name of an existing EC2 KeyPair to enable SSH access
Resources:
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.0.0.0/16
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: WordPressVPC
```

This template creates a VPC with internet access and tags to identify the resources. It also ensures that the WordPress environment will have the proper network configuration to communicate with other AWS services.

2. Creating an AMI of the WordPress Instance

Once the WordPress instance was successfully deployed, I created an Amazon Machine Image (AMI) of the instance. This allows us to replicate the configuration quickly without having to manually install WordPress and its dependencies every time.

The process to create the AMI:

aws ec2 create-image --instance-id i-xxxxxxx --name "WordPress-AMI" --no-reboot

This AMI includes the WordPress setup along with all necessary configurations and updates, so future instances launched from this image will be identical to the original one.

3. Configuring Auto Scaling to Launch a New WordPress Instance

To ensure the system is scalable and responsive to demand, I configured Auto Scaling for the WordPress instance. Auto Scaling helps maintain the desired number of instances running to handle incoming traffic. If the traffic increases, Auto Scaling automatically launches new instances. If an instance goes down, it will also be replaced automatically.

Here is the Auto Scaling configuration:

```
AutoScalingGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    VPCZoneIdentifier:
      - !Ref PublicSubnet
    LaunchTemplate:
      LaunchTemplateId: !Ref LaunchTemplate
      Version: !GetAtt LaunchTemplate.LatestVersionNumber
    MinSize: '1'
    MaxSize: '2'
    DesiredCapacity: '1'
    Tags:
      - Key: Name
        Value: WordPressInstance
        PropagateAtLaunch: true
```

The Auto Scaling configuration ensures that at least one instance is running at all times, and it adjusts based on demand, providing cost efficiency while ensuring high availability.

4. Configuring the WordPress Instance to Shut Down Automatically

To optimize resource usage and costs, I implemented an automatic shutdown mechanism for the WordPress instances. Using AWS Lambda, I created a function that stops the WordPress instances at a scheduled time. A CloudWatch Event Rule was configured to trigger the Lambda function every weekday at 6 PM UTC (1:30 AM IST).

Here is the configuration for the scheduled shutdown:

LambdaScheduleRule:

```
  Type: AWS::Events::Rule
  Properties:
    ScheduleExpression: cron(0 18 ? * MON-FRI *)
    State: ENABLED
    Targets:
      - Arn: !GetAtt StopInstanceLambda.Arn
        Id: StopInstanceLambdaTarget
```

This ensures that the WordPress instances will be stopped automatically after working hours, saving costs for the resources when they're not in use.

5. Monitoring the Instance Using Route 53 Health Checks

To monitor the health and availability of the WordPress instance, I used AWS Route 53's Health Checks. These health checks periodically check if the WordPress instance is available and reachable. If an instance fails the health check, Route 53 can take necessary actions, such as rerouting traffic to a healthy instance.

Here is the configuration for the health check:

```
HealthCheck:
  Type: AWS::Route53::HealthCheck
  Properties:
    HealthCheckConfig:
      IPAddress: "127.0.0.1"
      Port: 80
      Type: HTTP
      ResourcePath: "/"
      RequestInterval: 30
      FailureThreshold: 3
```

This ensures that the WordPress site is always up and running, and any issues with the instance can be quickly detected and resolved.

Conclusion

The project successfully automates the deployment of WordPress on AWS using CloudFormation. By leveraging Auto Scaling, automatic instance shutdown, and health checks, we ensured that the infrastructure is both cost-effective and reliable. Monitoring with Route 53 guarantees the availability of the WordPress site, and the use of an AMI streamlines future deployments.