**DevOps Engineer Technical Exercise**

**Objective:**

The goal of this exercise is to assess your ability to use Python in conjunction with AWS CDK and CloudFormation to create, manage, and deploy infrastructure as code. You will create a basic web application stack, ensuring it's scalable, secure, and follows best practices.

**Task:**

You are required to create a simple web application infrastructure using AWS CDK in Python. The infrastructure should include the following components:

1. VPC (Virtual Private Cloud):

- Create a VPC with public and private subnets across at least two availability zones.

- Include a NAT Gateway for allowing instances in private subnets to access the internet.

2. ECS Tasks:

- Deploy ECS tasks in the private subnets.

- The instances should be part of an Auto Scaling group with a minimum of 2 instances and a maximum of 4 instances.

- The instances should have an IAM role attached that allows them to interact with S3 and CloudWatch.

3. Load Balancer:

- Deploy an Application Load Balancer (ALB) in the public subnets.

- The ALB should be configured to distribute traffic to the ECS tasks in the Auto Scaling group.

- Ensure the ALB has security groups that only allow HTTP (port 80) and HTTPS (port 443) traffic.

4. S3 Bucket:

- Create an S3 bucket to store static assets for the web application.

- Ensure the bucket is private by default and configure it to serve content using a pre-signed URL mechanism.

5. CloudFormation Outputs:

- Make sure to output relevant details like the ALB DNS name, the S3 bucket name, and the RDS endpoint (if applicable) using CloudFormation outputs.

**Requirements:**

- Python: Write the CDK code using Python.

- CDK Constructs: Use AWS CDK constructs to define the resources.

- CloudFormation: The resulting CloudFormation templates should follow best practices.

- Version Control: Provide a GitHub repository link with your CDK code and a README file that includes instructions on how to deploy the stack.

**Evaluation Criteria:**

- Code Quality: Is the code well-organized, modular, and adheres to Python best practices?

- Infrastructure Design: Does the design of the infrastructure follow AWS best practices (e.g., security, scalability)?

- Use of CDK: Are the CDK constructs used effectively and correctly?

- Documentation: Is the README clear and does it accurately describe how to deploy the stack?

*- Bonus Points: For adding advanced features like RDS, implementing security best practices (e.g., encryption, least privilege), or setting up CI/CD pipelines.*

**Submission:**

- Upload your code to a GitHub repository.

- Include a `README.md` with detailed instructions on how to deploy the infrastructure.

- Ensure that your repository is public or provide access to the reviewers.

**Time Estimate:**

This task should take approximately 3-6 hours, depending on your familiarity with the tools and the additional features you choose to implement.