DevOps Task

Task

Provisioning Infrastructure as a code with terraform for Web Applications and Databases (MongoDB, SQL Server) and Redis for caching, with Domain Name, Certificate, and Load Balancer, Using docker and Kubernetes.

Description

As a DevOps engineer, your task is to write a terraform template to provision the infrastructure for a web application and its associated databases MongoDB, SQL Server, and Redis for caching. Additionally, you will need to set up a domain name, SSL certificate, and load balancer to ensure scalability and security. This task will test your skills in infrastructure provisioning, database management, caching, networking, and security.

Application, databases and Redis will be each one as a dockers and use Kubernetes to let up a replication controller to run pods that are accessed as services on Docker.

Application docker to use in the task:

https://github.com/docker/awesome-compose/tree/master/aspnet-mssql

Requirements

Infrastructure Provisioning:

- Configure and provision virtual machines or containers to run the web application, MongoDB, SQL Server, and Redis.
- Ensure the infrastructure is scalable and fault-tolerant by utilizing appropriate cloud services (e.g., auto-scaling groups, availability zones).

Domain Name and SSL Certificate

- Register or configure a domain name for the web application.
- Obtain and install an SSL certificate for secure communication.
- Configure DNS settings to associate the domain name with the infrastructure.

Load Balancer

- Set up a load balancer to distribute traffic across multiple instances of the web application.
- Configure the load balancer to perform health checks on the instances and automatically route traffic to healthy instances.
- Implement SSL termination on the load balancer for end-to-end encryption.

Databases

- Provision MongoDB, SQL Server, and Redis instances are suitable for production use.
- Configure and secure the databases, including setting up authentication and access controls.
- Implement appropriate backup and recovery mechanisms for the databases.

Caching

- Set up a Redis caching layer to improve the performance of the web application.
- Configure the web application to utilize Redis for caching data.

Security and Access Management

- Ensure proper security measures are in place, including network security groups, firewall rules, and access controls for all components.
- Configure appropriate access credentials for the web application and databases, following the principle of least privilege.

Documentation:

- Provide clear and concise documentation on the infrastructure setup and configuration steps.
- Include diagrams, instructions, and any necessary code snippets to reproduce the environment.

Deliverables

1. Terraform template for:
    - Web application, MongoDB, SQL Server, and Redis provisioned and running over Kubernetes.
    - Domain name configured and accessible with a valid SSL certificate.
    - Load balancer set up and correctly distributed traffic across instances of the web application.
    - MongoDB, SQL Server, and Redis instances are properly configured and secured.
    - Redis caching layer integrated into the web application for improved performance.
    - Security measures were implemented, including network security groups and access controls.
    - Documentation detailing the setup, configuration, and any additional instructions.
    - A diagram shows the entire system architecture.