LOMBA KOMPETENSI SISWA (LKS)
SEKOLAH MENENGAH KEJURUAN
TINGKAT PROVINSI JAWA BARAT
TAHUN 2024

NASKAH SOAL

Bidang Lomba

Cloud Computing

Modul 1 – Monitoring Scalable Cloud Architecture

PEMERINTAH DAERAH PROVINSI JAWA BARAT

DINAS PENDIDIKAN

Jalan Dr. Radjiman No. 6 Telp. (022) 4264813 Fax. (022) 4264881
Website : disdik.jabarprov.go.id
e-mail: disdik@jabarprov.go.id/sekretariatdisdikjabar@gmail.com
BANDUNG - 40171

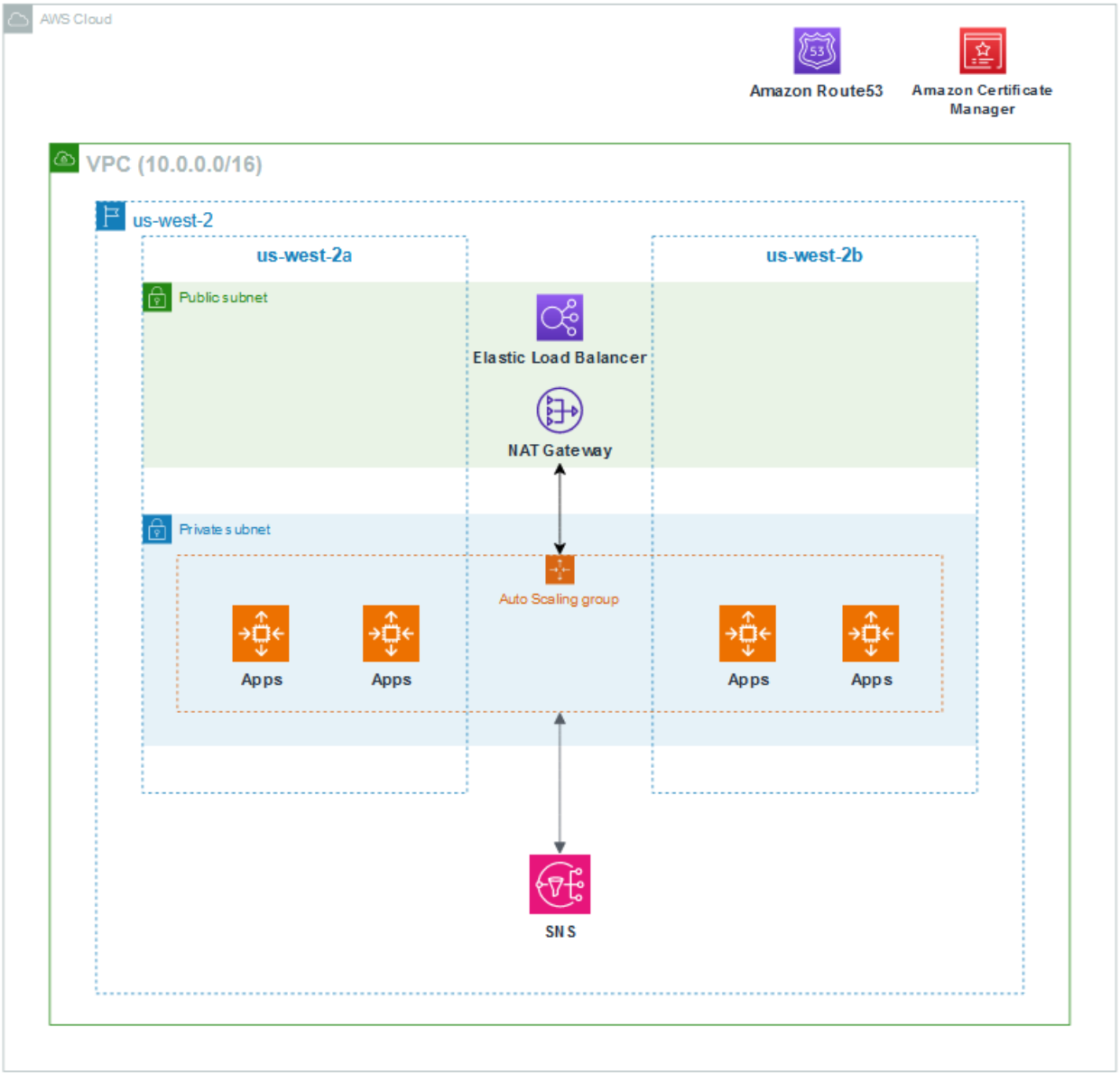
1. Overview

This project aims to deploy and scale a web application with PHP as our technology stack. Sometimes, our resources are high either CPU or memory, thus the servers we use are stuck at the peak time. So, we need a scalable cloud infrastructure that will handle these issues and not worry about incoming requests or traffic any more. You must build and deploy our web application into AWS infrastructure with auto scaling technology and auto setup the application servers with user data script. We need to set up a notification service to notify our team if any events on auto scaling occur whenever instances launch, terminate, fail to launch, and fail to terminate. Before doing tasks in this project, please read carefully the technical details below.

2. General Rules

2.1 Failure to comply with the rules will result in immediate disqualification.
2.2 You have 4 hours to finish the tasks.
2.3 Please share your AWS access key and secret key on this link https://credential.lkscc.my.id/.
2.4 You may use AWS Console and AWS CLI to deploy the solutions. You may not use SAM, CloudFormation, or CDK.
2.5 Between and after the event, you may not access your account. Any activity on AWS during this period is not allowed.
2.6 During the event, multiple logins are not permitted.
2.7 If you have any questions, do not hesitate to ask.

3. Architecture



3.1 The application will be deployed to the AWS cloud infrastructure with VPC as the network stack.

3.2 You need to set up NAT Gateway for accessing the internet connection from private instances (in a private subnet).

3.3 Amazon Route53 and Amazon Certificate Manager as hosted zone layer. You can set up your domain name here.

3.4 Amazon EC2 is for deploying web servers for PHP applications and the source code will be stored inside each instance.

3.5 Amazon Elastic Load Balancer automatically distributes network traffic to improve application scalability.

3.6 Amazon EC2 Auto Scaling instance that will help you maintain application availability and lets you automatically add or remove EC2 instances using scaling policies that you define. Dynamic or predictive scaling policies let you add or remove EC2 instance capacity to service established or real-time demand patterns.
3.7 As mentioned before in the overview section, the application is written in PHP. So you need to install the required dependencies using the *user data* script that will be explained in the task section.
3.8 Amazon SNS for the notification service that will notify our team. So, you need to subscribe to the Amazon SNS topic using your active email address.

4. Information

4.1 The application is written in PHP.
4.2 This solution must be deployed in the **us-west-2** (**Oregon)** region. Deploying in another region will result in a major point reduction.
4.3 An automated scoring system will check your AWS account to calculate the score of your task.
4.4 In order check your progress, the scoring system needs an AWS credential with administrator privilege. You will be asked to provide the AWS credential.
4.5 AWS tag is used by the scoring system to find your solution. Forgetting to add the required tag or adding the same tag value to multiple instance may result in an unexpected behavior and thus lower your score.
4.6 The system will calculate 90% of the final score. The remaining 10% will be evaluated by checking manually in front of the participants.

5. Task

5.1 First thing first, you must create an Access Key on your account IAM security credentials. Download and save the Access Key ID and Secret Access Key in the safest place. The Access Key ID and Secret Access Key will be used for scoring your result.
5.2 Create a VPC on your AWS account:
- Set IPv4 CIDR: 10.0.0.0/16
- Add a tag to the VPC: KEY=LKS-CC-2024, VALUE=VPC

5.3 Create public and private subnets with the following requirements:
- Public subnet on availability zone A with network 10.0.0.0/28, add a tag KEY=LKS-CC-2024, VALUE=PUBLIC-SUBNET-A
- Public subnet on availability zone B with network 10.0.1.0/24, add a tag KEY=LKS-CC-2024, VALUE=PUBLIC SUBNET-B
- Private subnet on availability zone A with network 10.0.2.0/24, add a tag KEY=LKS-CC-2024, VALUE=PRIVATE-SUBNET-A
- Private subnet on availability zone B with network 10.0.3.0/24, add a tag KEY=LKS-CC-2024, VALUE=PRIVATE-SUBNET-B

5.4 Create an Internet Gateway and attach it to the VPC that you have created before:
- Add a tag to the internet gateway: KEY=LKS-CC-2024, VALUE=INTERNET-GW

5.5 Create a Public NAT gateway on this VPC, thus EC2 instances in the private subnet could access the internet to install required dependencies:
- Add a tag to the NAT gateway: KEY=LKS-CC-2024, VALUE=NAT-GW

5.6 Create 2 route tables for Public and Private subnets:
- Add a tag to the public route: KEY=LKS-CC-2024, VALUE=PUBLIC-ROUTE
- Add a tag to the private route: KEY=LKS-CC-2024, VALUE=PRIVATE-ROUTE

5.7 Associate public subnets with the public route table and add a route to 0.0.0.0/0 through Internet Gateway

5.8 Associate private subnets with the private route table and add a route on 0.0.0.0/0 through NAT Gateway

5.9 Create a security group that will allow incoming requests (inbound) to EC2 instances in private subnet:
- Allow HTTP and HTTPS
- Add a tag to the security group: KEY=LKS-CC-2024, VALUE=SECURITY-GROUP

5.10 Create EC2 launch templates with the following specifications:
- Operating Systems could be Ubuntu 24.04. Using a custom AMI is not allowed
- EC2 instance type: t3a.micro
- Create or use existing key pair
- Use the security group that you have created web (SECURITY-GROUP)
- Add a tag on the launch template: KEY=LKS-CC-2024, VALUE=ASG-TEMPLATE
- On Advanced Details, add or upload the *user data* script from the repository https://github.com/itsgitz/lksccjabar2024modul1_aplikasi inside **scripts** directory named **user_data.sh**.

5.11 Create an application load balancer for this Auto Scaling Group and attach it to Auto Scaling Group (later)
- Specify load balancer name: **LKS-CC-2024-ELB**
- Select the application load balancer
- Select the internet-facing load balancer scheme
- Create a new target group with name **LKS-CC-2024-ELB-TARGET** and the tag: KEY=LKS-CC-2024, VALUE=ELB-TARGET-GROUP
- Add a tag to this load balancer: KEY=LKS-CC-2024, VALUE=ELB
- Enable health check

5.12 Create an Auto Scaling group with the following specifications:
- Select your launch template
- Select your VPC for this project
- Select Availability and Subnets
  - PRIVATE-SUBNET-A
  - PRIVATE-SUBNET-B
- Attach this Auto Scaling Group to the existing load balancer (the ELB that you have created before)
- On group size, you must specify the following requirements:
  - Desired capacity: 2
  - Minimum capacity: 2
  - Maximum capacity: 4
- For scaling policies, you must set up:
  - Metric type: Average CPU utilization
  - Target Value: 85
- Add a tag to this Auto Scaling Group: KEY=LKS-CC-2024, VALUE=ASG

5.13 Set Amazon Route53, and add your hosted zone with a domain name that you ordered from Amazon Route53 itself, or from your favourite ISP:

- Add a tag to the zone: KEY=LKS-CC-2024, VALUE=DNS

5.14 We need HTTPS protocol for securely accessing our website. So, you must create a certificate on Amazon Certificate Manager for *.[YOUR_DOMAIN] and use it on Elastic Load Balancer:

- Add a tag to ACM: KEY=LKS-CC-2024, VALUE=ACM

5.15 Go to the ELB that you have created before while and set up the following requirements:

- Listener HTTP (80) must redirect to HTTPS (443)
- Use your certificate from ACM on this load balancer listener

5.16 Back to Route53, create a zone for ELB with domain name modul1.[YOUR_DOMAIN]. Thus, our users can easily access https://modul1.[YOUR_DOMAIN].

5.17 You need to set up the Amazon SNS for our notification on our infrastructure.

- Add a SNS topic with the name **LKS-CC-2024-TOPIC** and specify the tag: KEY=LKS-CC-2024, VALUE=SNS
- Create a subscription and subscribe to the topic you have created before using your active email address
- Check your email address and confirm the Amazon SNS subscription to getting notification from the auto scaling group

5.18 After you run the Auto Scaling Group, then the instances will be created. In that process, the user data script that you entered on the EC2 Launch Template will be executed. Thus, you don't need to install any applications and dependencies anymore (it's automated by user data script).

6. How to test your deployment

6.1 Access our web application https://modul1.[YOUR_DOMAIN] from the browser.

6.2 You see your single-page web application on the browser.

6.3 For testing the Auto Scaling Group, destroy or terminate one instance in the private subnet. If the application still works and the user can access our web application, then the scalable cloud infrastructure in this project is perfect!

6.4 Also, you will receive an email notification whenever the instances launch, terminate, fail to launch, or fail to terminate.