Create a VPC using CloudFront

Objective:

- Create a VPC with public and private subnets

General Instructions:

- Each lab is to be done individually. However you can discuss with others, but effort should finally be yours.

References:

- https://faun.pub/aws-cloudformation-essentials-notes-from-the-field-8ed7162e0c5 (blog on Cloudformation which helps create a VPC from template)
- https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/Welcome.html (official notes from AWS)
- https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/gettingstarted.templatebasics.html (template details)
- https://faun.pub/aws-cloudformation-essentials-notes-from-the-field-8ed7162e0c5

Lab Instructions:

We could use the VPC wizard in the management console (https://console.aws.amazon.com/vpc/) for this. However, we will use an AWS service called CloudFormation to build the VPC. Why? This will demonstrate how to use code to launch an infrastructure (as against doing it manually). Further it is easy to update and reuse.

CloudFormation allows us to create a "stack" of "resources" in one go. Resources are the things we create (EC2 Instances, VPCs, subnets, route-table, gateway etc.). A stack, for instance, can include all the resources required to run a web application, such as a web server, a database, and networking rules. We can write a template (in json or yaml) that can easily setup a network stack and then once setup, we can invoke them where needed. All this will become clear as we proceed. We will be using the json format for the template. The first four references above give more details.

1. Goal

You are supposed to create a network with components such as:

0. **Region**
- us-west-2
1. **VPC (Virtual Private Cloud)**
    - A logically isolated section of the AWS cloud where you can launch AWS resources in a virtual network that you define.
    - **CIDR Block**: `10.1.0.0/16`
2. **Subnets**
    - Divisions within the VPC to organize and secure resources:
        - **Public Subnet 1**
            - CIDR Block: `10.1.1.0/24`
        - **Public Subnet 2**
            - CIDR Block: `10.1.2.0/24`
        - **Private Subnet 1**
            - CIDR Block: `10.1.3.0/24`
        - **Private Subnet 2**
            - CIDR Block: `10.1.4.0/24`
3. **Internet Gateway**
    - Provides internet access for resources within the public subnets.
4. **Route Tables**
    - Direct network traffic within the VPC:
        - **Public Route Table**
            - Associated with Public Subnet 1 and Public Subnet 2.
            - Default route to the Internet Gateway.
5. **VPC Gateway Attachment**
    - Attach the Internet Gateway to the VPC.

Communication Paths

1. **Public Subnets**
    - **Public Subnet 1 and Public Subnet 2** have internet access via the Internet Gateway.
    - Instances launched in these subnets can communicate with the internet and other instances within the VPC.
    - **Route Table**: Public Route Table with a route directing traffic to the Internet Gateway (`0.0.0.0/0`).
2. **Private Subnets**
    - **Private Subnet 1 and Private Subnet 2** do not have direct internet access.

    - Instances in these subnets cannot directly communicate with the internet unless they go through a NAT Gateway or a VPN connection (which is not defined in your template).
    - These subnets can communicate with other instances within the VPC, including those in the public subnets.

This setup is typical for applications where front-end servers (like web servers) need to be public-facing and access the internet, while back-end servers (like databases) are kept private and secure without direct internet access.

2. Use a CloudFormation template to create VPC and relevant resources

- In the AWS Console, click on **All Services**, under **Management and Governance**, click on **CloudFormation** to open the **CloudFormation dashboard**.
- Click **Create Stack** (top right). Select "**With new resources**".
- A CloudFormation template called **"vpc_template.json"** is already present in the **labDirectory**. Complete the template by filling in all the **<Fill here>** sections present in the template. You should fill in the template following the requirements(goals) of the activity.
  Note: Do not modify any other part of the template apart from <Fill here>
- Select **Template is ready** and then **Upload a template file.** Then choose the file and upload the **"vpc_template.json"** file. Click **Next.**
- In the **Stack name** textbox, type **vpc-stack**. Click **Next**. Skip the Options page and click **Next**.
- Click **Create**. You will notice that the status of the template is **CREATE_IN_PROGRESS.** The template should finish creating in some time.
- How to check if the resources are created?
    - Go to the AWS Management Console home page again.
    - Under All Services, under "Networking and Content Delivery", select VPC.
    - In the VPC dashboard, on the left navigation menu, click **Your VPCs**. You will see a VPC named **aws-vpc** in the list. If you look at the template file, you will notice that we have named our VPC, aws-vpc, which got successfully created.

- In the VPC dashboard, on the left navigation menu, click **Subnets**. You will see four subnets starting with subnet-xxxx. If you look at the template file, you will notice that we have named our subnets, subnet-xxxx, which also got successfully created. You can also check that the IPv4 address blocks match as well.

3. Create an IAM user for the instructor for evaluation

For the purpose of checking correctness of this lab you need to create an AWS IAM user(same as earlier) for instructor and attach the policy **instructor_vpc_policy.json** which is present inside **labDirectory.** You will then generate the access keys for the user. Those access keys will be used by the instructor to make programmatic calls to AWS services via AWS CLI or APIs.

4. Submission Instructions:

```
{
   "ACCESS_KEY_ID": "INSTRUCTOR Access key ID",
   "SECRET_ACCESS_KEY": "INSTRUCTOR Secret Access Key",
   "region": "us-west-2"
}
```

Replace **INSTRUCTOR Access key ID** and **INSTRUCTOR Secret Access Key** with Instructor IAM user credentials.

5. Delete the IAM user for instructor

- Delete the IAM user that you created for the instructor after evaluation is done.