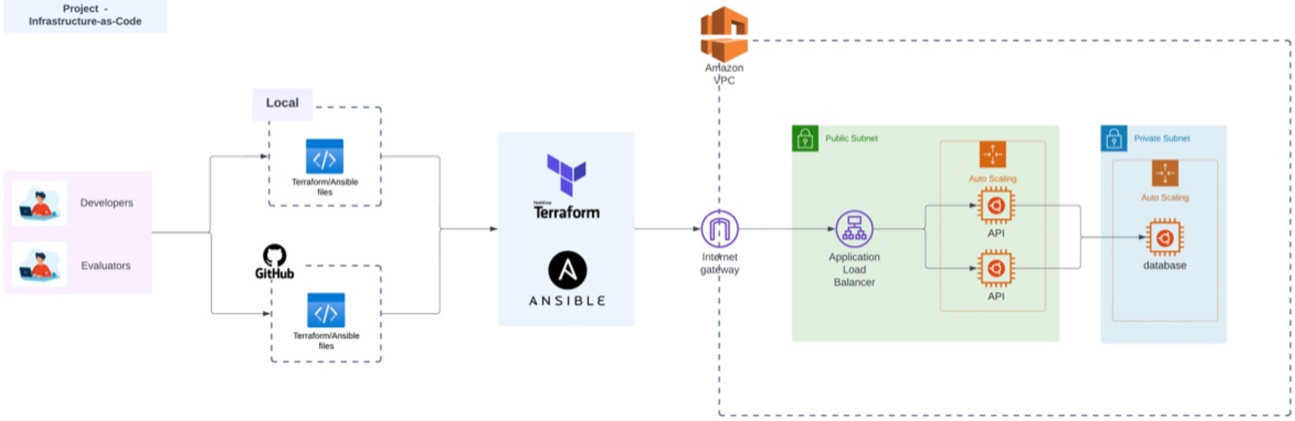
Project 2 - Infrastructure Provisioning Automation

Due date: October 22, 2023

Architecture



In this project, students will work in a team to deploy AWS infrastructure using infrastructure provisioning automation tools. The infrastructures that are deployed can be used to host typical multi-tiered web apps and microservices.

Requirements

Deploy and configure web app infrastructure with Terraform and/or Ansible

Use Terraform and/or Ansible to set up a multi-tiered application infrastructure architecture. You can leverage both Terraform and Ansible to automate the infrastructure provisioning and configuration or you can use just one tool of choice. For example you can use cloud init scripts within Terraform, or you can combine Terraform and Ansible roles to set up the infrastructure and configuration.

1. AWS VPC network.
    1. Create an AWS VPC.
    2. Create an internet gateway and associate it to the VPC.
    3. Create one private subnet and one public subnet in the VPC.
    4. Create a public route table routing 0.0.0.0/0 to the internet gateway.
    5. Associate the public subnet to the public route table.

   6. Create security groups for the subnets.
      1. Private subnet security group allows all incoming traffic from the public subnet security group.
      2. Public subnet security group allows incoming traffic on port 80 and 443 from 0.0.0.0/0 (public internet).
2. Database server.
   1. Create and launch an EC2 instance - e.g. Ubuntu 22.04 - into the private subnet created earlier.
   2. Download and install a database server in the EC2 instance - e.g. MySQL, PostgreSQL or MongoDB.
   3. Start and enable the database server service and ensure the database ports - e.g. 3306, 5432, etc. - are open and able to receive connections.
3. API server.
   1. Create and launch an EC2 instance - e.g. Ubuntu 22.04 - into the public subnet created earlier.
   2. Download and install packages for an API server.
      1. For example for a Python FastAPI API server:
         1. Python 3.10.
         2. fastapi python library.
      2. For MEAN stack NodeJS Express API server:
         1. Node 18
         2. Express library
4. Bonus - you can add the following components to your infrastructure but it is not mandatory.
   1. Autoscaling group.
   2. Application load balancer.

Documentation

1. The code is stored in a public GitHub repo.
2. Create a README.md in your GitHub repo.
3. In the README.md
   1. URL to public GitHub repo
   2. Architectural diagrams depicting the AWS cloud infrastructure setup and any other pertinent visualizations.
   3. Include instructions on:
      1. How users can leverage the Terraform and or Ansible scripts to provision and configure the infrastructure in your project.
      2. How users can shut down and clean up the infrastructure launched from your project.

Submission Instructions

1. In the README.md of your GitHub repo include the names of members in your group.
2. Download a Zip file of your completed GitHub repo.
3. Click on `Hand In` tab in the learning portal project page.
3. Click on `Upload Assignment` and upload the zip file.