Multi-disciplinary project

Mikaeil’s project

Group: RDEnvi

Robertas Asvicas

Dmitry Lvov

Client: Mikaeil Shaghelani

Table of Contents

[Project Scope 3](#_Toc62338247)

[Deliverables 3](#_Toc62338248)

[Infrastructure design 4](#_Toc62338249)

[Frontend 4](#_Toc62338250)

[Backend 4](#_Toc62338251)

[Database 4](#_Toc62338252)

[Terraform script 5](#_Toc62338253)

[Ansible playbook 5](#_Toc62338254)

[IAM Users 5](#_Toc62338255)

# Project Scope

Our client came to us with need to host his website. Client needed for it to be scalable, easily accessible, and easily configurable. His website consisted of 3 things:

* Frontend – react.js
* Backend – Flask
* Database – PostgreSQL

# Deliverables

* Infrastructure on Amazon Web Services
* IAM accounts with restricted permissions
* Terraform script for automated deployment of the infrastructure
* Ansible script for the deployment and configuration of the Flask/Apache application
* PostgreSQL database
* Auto-scaling groups for the instances
* User Manuals

# Infrastructure design

Starting from the top:

Internet Gateway is connected to Mikaeil’s VPC.

Mikaeil’s VPC (Virtual Private Cloud) has a CIDR Block of 10.102.0.0/16. It’s connection to the Internet gateway is made through the NAT Instance. NAT Instance works as a firewall and as a router. It has port forwarding enabled on it:

* :80 to 10.102.3.0:3000 (Frontend)
* :8000 to 10.102.5:3000 (Backend)

Nat is connected to Load Balancer which balances the load between to AZs.

## Frontend

Has Docker installed on it, which is setup in Continuous Deployment. The Continuous Deployment is setup with Docker Hub and Watchtower (container which monitors was another containers image changed, and if yes, then it updates the running container). The CIDR Blocks are 10.102.3.0/24 and 10.102.4.0/24 (depends on the AZ). How to setup CD is written in the “User Manual – Frontend”.

## Backend

Backend subnet has 2 different EC2 instances – Ansible Node and backend itself. Backend has Flask and Apache installed and configured on it; however it’s done from Ansible instances. IP CIDR Blocks: 10.102.5.0/24 and 10.102.6.0/24. Ansible node has Ansible playbooks, one of them will setup Backend Instance. How it works will be explained below. Backend wasn’t connected to the frontend

## Database

Wasn’t setup, due to the misconfiguration between the frontend and the backend.

# Terraform script

# Ansible playbook

Ansible playbook is used to deploy Flask/Apache application on the backend. IPs of the backend machines are in directory *‘/home/devops/ansibleonelove/inventories/dev*’ in the file ‘*hosts*’. Variables are also there that should be edited:

* Ansible\_ssh\_user – what user Ansible should ssh to
* Github\_user – which user in Github has repo
* App\_name – what repo should be pulled, also it will create folders with the same names

Ansible.cfg file is responsible for showing where other files (like previously mentioned file “hosts”) are.

In the directory ‘*templates*’ there are 2 files: FlaskAppBig.j2 – config file for the apache, flaskapp.j2 – wsgi file.

Ansible-playbook.yml is the playbook that you run. It installs apache, python tools, pip3, activates virtualenv, pull Github repo and configures flask and apache using FlaskAppBig.j2 and flaskapp.j2.

# IAM Users

Specific policy made that for user MikaeilTestUser instances with tag Group: Mikaeil are accessible.