**Terraform Infrastructure Documentation on VPS Platform**

10/09/2025

Haseef Ahamed

Contents

[1. Project Overview 3](#_Toc208413290)

[2. Architecture Diagram 3](#_Toc208413291)

[3. Terraform Modules 3](#_Toc208413292)

[4. Key Terraform Concepts 4](#_Toc208413293)

[5. File Structure 4](#_Toc208413294)

[6. Root Configuration (main.tf) 5](#_Toc208413295)

[7. Modules Explained 8](#_Toc208413296)

[8. Outputs 9](#_Toc208413297)

[9. Nginx Config 10](#_Toc208413298)

[10. Step-by-Step Workflow 10](#_Toc208413299)

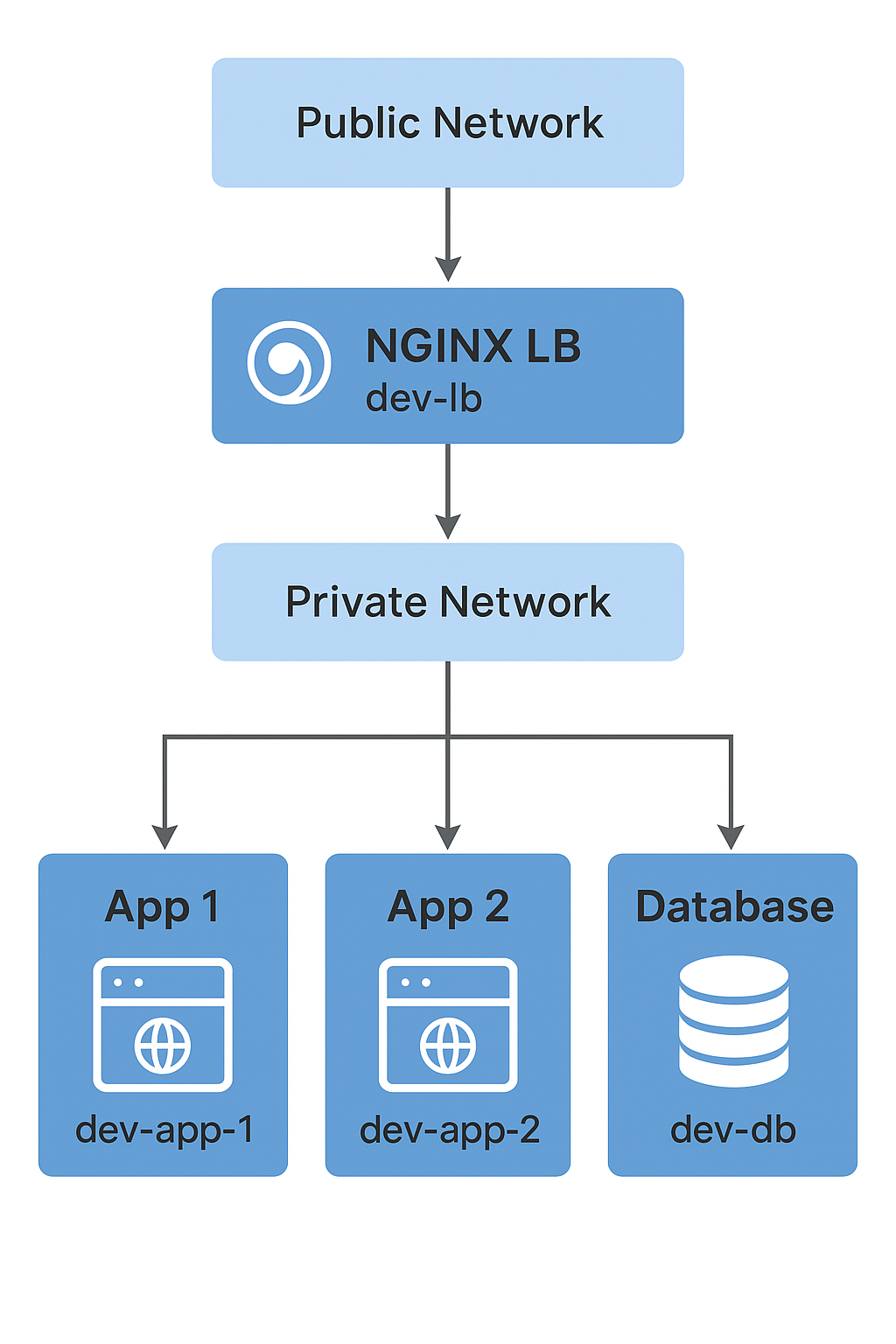
[11. Buggs and Solution 13](#_Toc208413300)

# Project Overview

This project deploys a multi-container environment using **Terraform** and **Docker** on a VPS. The architecture includes:

* **Load Balancer (LB)**: NGINX container to distribute traffic to backend app containers.
* **Application (App)**: Multiple HTTPD containers serving web apps.
* **Database (DB)**: MySQL container for persistent storage.
* **Network**: Public and private Docker networks for container isolation and communication.

# Architecture Diagram



# Terraform Modules

|  |  |  |
| --- | --- | --- |
| **Module** | **Description** | **Resources** |
| network | Manages Docker networks | docker\_network.public, docker\_network.private |
| compute | Creates application containers | docker\_image.app, docker\_container.app |
| database | Creates database container | docker\_image.mysql, docker\_container.db |
| main.tf | Manages load balancer and provider configuration | docker\_image.nginx, docker\_container.lb |

# Key Terraform Concepts

* **Providers**: kreuzwerker/docker is used for Docker container management.
* **Backend**: Local backend (terraform.tfstate) stores state file on the VPS.
* **Absolute Paths**: Required for Docker volume mounts.
* **Networks**: Containers communicate using Docker networks (private\_net\_id and public\_net\_id).
* **Outputs**:
  + lb\_access\_url: URL to access the NGINX load balancer.
  + app\_ips: IP addresses of app containers.
  + db\_endpoint: Database connection endpoint.

# File Structure

A screenshot of a computer

AI-generated content may be incorrect.

# Root Configuration (main.tf)

Providers and Backend

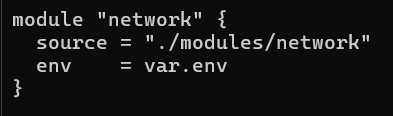
A computer screen with white text

AI-generated content may be incorrect.

* Uses the Docker provider to manage Docker resources on your VPS.
* Stores Terraform state locally (terraform.tfstate).

Modules

Network Module



* Creates public and private Docker networks.
* Output: public\_net\_id, private\_net\_id.

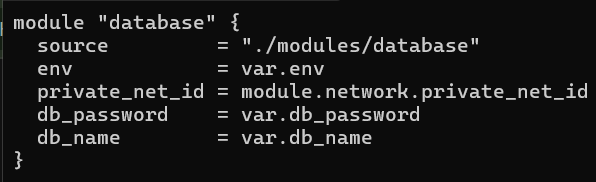
Compute Module

A black screen with white text

AI-generated content may be incorrect.

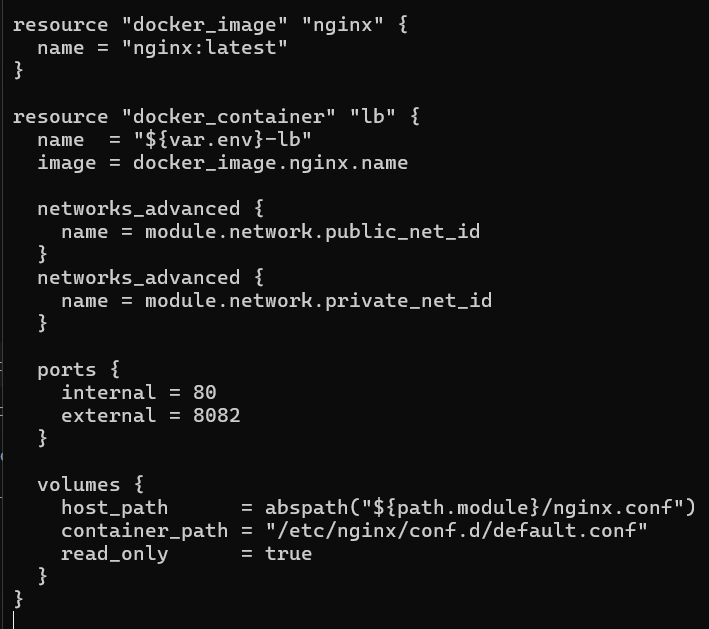
* Deploys multiple app containers (httpd:latest) on private network.
* Output: app\_ips.

Database Module



* Deploys MySQL container on private network.
* Output: db\_endpoint.

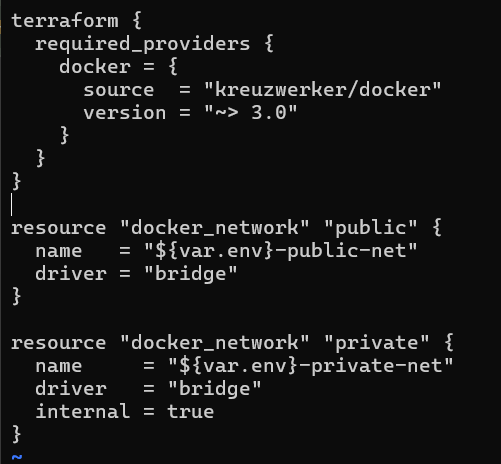
Load Balancer



* Nginx container bridges **public and private networks**.
* Uses nginx.conf to route traffic to app containers.
* Exposes port 8082 on VPS.

# Modules Explained

* + 1. Network Module



* public network allows external access.
* private network is internal only for apps and DB communication.
  + 1. Compute Module

A computer screen shot of a computer program

AI-generated content may be incorrect.

* Deploys var.app\_count HTTP app containers.
* Connected to the private network.
* count = var.app\_count -> number of app containers.
* Outputs IPs for LB to use.
  + 1. Database Module

A computer screen shot of a computer code

AI-generated content may be incorrect.

* MySQL container on private network.
* Environment variables configure DB credentials.
* Exposes external port 3307 for optional access.

# Outputs

* lb\_access\_url → Access load balancer via VPS public IP.
* db\_endpoint → DB IP:port for internal or external access.
* pp\_ips → List of all app container IPs in private network.

# Nginx Config

A computer screen with white text

AI-generated content may be incorrect.

* Routes traffic from public network to private app containers.

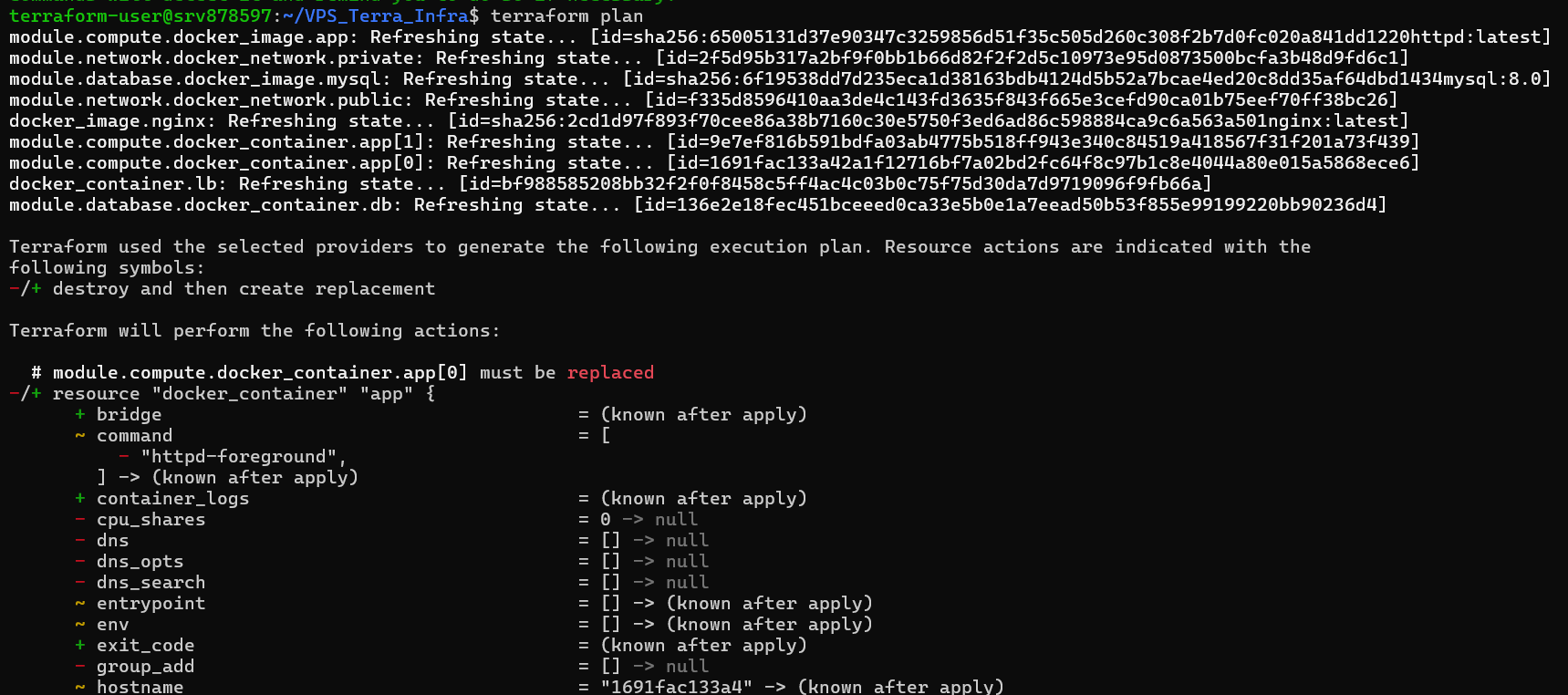
# Step-by-Step Workflow

* + 1. Initialize Terraform

A screenshot of a computer program

AI-generated content may be incorrect.

* Downloads Docker provider (kreuzwerker/docker).
* Prepares local backend (terraform.tfstate).
  + 1. Check What Will Happen



Terraform **simulates** what it will create:

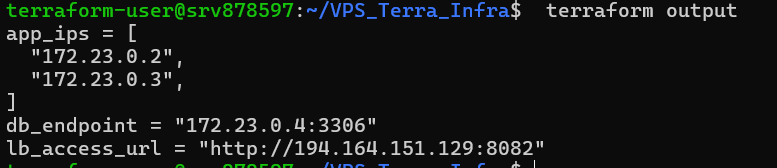
* Networks → public & private
* App containers → 2 by default
* Database container
* Nginx load balancer
* It’s like **drawing a blueprint** before building.
  + 1. Apply Terraform

# terraform apply

A screenshot of a computer screen

AI-generated content may be incorrect.

* Terraform actually **creates the resources**.
  1. Check Outputs



* Open browser → http://194.164.151.129:8082 → traffic is routed to one of the app containers.
* Use mysql -h 172.23.0.4 -P 3306 -u root -p → connect to DB (inside private network).

A screenshot of a computer program

AI-generated content may be incorrect.

Verify the Containers status:

A screen shot of a computer program

AI-generated content may be incorrect.

A computer screen with white text

AI-generated content may be incorrect.

# Buggs and Solution

* + - * 1. Terraform Provider Issue

Problem:

When running terraform init, you got:

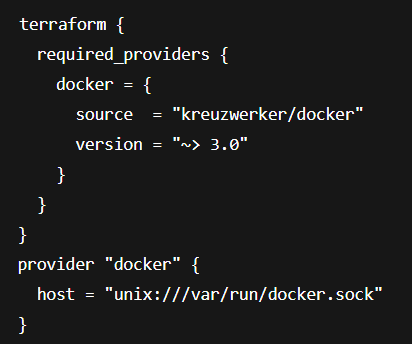


Cause:

* Terraform was trying to use the **wrong provider** (hashicorp/docker) which does not exist in the registry.
* Your modules were implicitly depending on hashicorp/docker.

Solution:

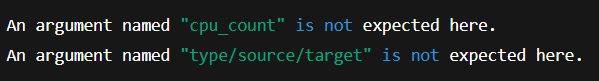
* Explicitly specify the correct provider in terraform block:



* + 1. Unsupported Arguments in docker\_container\

Problem:

Running terraform plan gave errors:

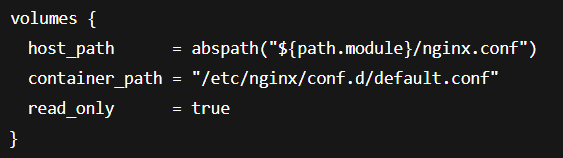


Cause:

* kreuzwerker/docker provider v3+ does **not use cpu\_count**.
* volumes { type, source, target } was the old syntax from hashicorp/docker.

Solution:

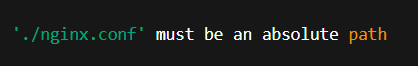
* Use the **current provider syntax**:



* Remove unsupported fields (cpu\_count, type, source, target).

1. Path Issue for Volume Mount

Problem:



Cause:

* Docker provider requires **absolute paths** for host volume mounts.

Solution:

* Use abspath():



1. LB Container Exited Immediately

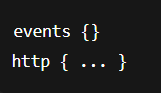
Problem:



* Even after terraform apply, LB container kept crashing.

Cause:

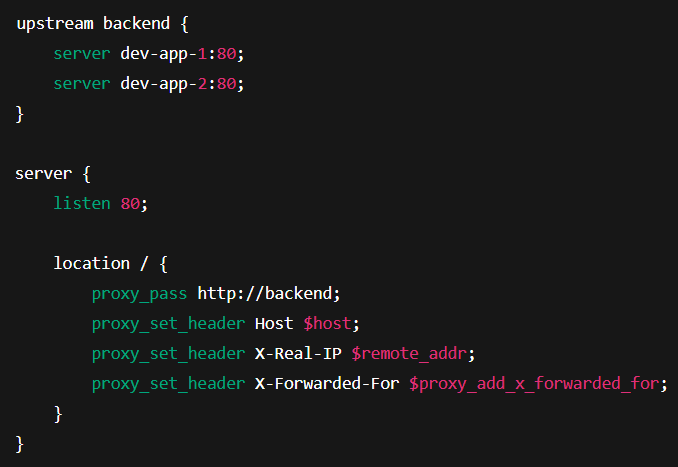
* Your nginx.conf included:



* When mounted to /etc/nginx/conf.d/default.conf, the extra http {} caused **nested http blocks**, which NGINX rejects.
* Also, LB might not have been on the same network as app containers, so dev-app-1/dev-app-2 could not be resolved.

Solution:

* Correct nginx.conf:



* Mount to /etc/nginx/conf.d/default.conf.
* Ensure LB is connected to the **same private network** as app containers.

1. App Not Accessible in Browser

Problem:

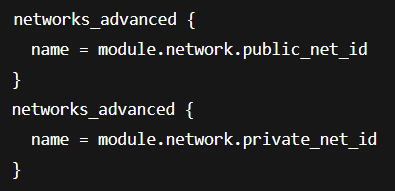
* Containers were running, but visiting http:// 194.164.151.129:8082 returned nothing.

Cause:

* NGINX LB was failing to start (due to nginx.conf errors).
* App containers were on private network, LB was not able to resolve them.

Solution:

* Fixed nginx.conf as above.
* LB connected to **both private and public networks**:



* Exposed LB port 8082 to VPS.