Lab 08: Deploy Multiple Azure Web Apps with a Shared App Service Plan Using Terraform Modules

Objectives

- Create a reusable Terraform module for an Azure Web App.
- Use the module three times to deploy 3 web apps (.NET 6) on a single shared App Service Plan.
- Use variables and `terraform.tfvars` to parameterize the deployment.
- Output the URLs of all three web apps.

Prerequisites

- Terraform and Azure CLI installed.
- Visual Studio Code.
- An Azure subscription with contributor access.
- Completion of Lab 06 (remote backend setup).

Folder Structure

```
Lab-08-WebApps-With-Modules/
├── main.tf
├── variables.tf
├── terraform.tfvars
├── outputs.tf
└── modules/
    └── webapp/
        ├── main.tf
        ├── variables.tf
        └── outputs.tf
```

Lab Steps

✅ Step 1: Set Up Folder Structure

1. Open VS Code and in the integrated terminal:

```
mkdir Lab-08-WebApps-With-Modules
cd Lab-08-WebApps-With-Modules
mkdir -p modules/webapp
```

2. Create the following files:
   - In the root: `main.tf`, `variables.tf`, `terraform.tfvars`, `outputs.tf`
   - In `modules/webapp/`: `main.tf`, `variables.tf`, `outputs.tf`

✅ Step 2: Configure Remote Backend

- In the root `main.tf`, configure the backend to store state in your existing Azure Blob Storage (from Lab 6).
- Use a unique key like `terraform-lab08.tfstate` to avoid state conflicts.

✅ Step 3: Define Variables

In the root `variables.tf` and `terraform.tfvars`, define:

- Resource group name and location
- App Service Plan name
- App names (as a list of 3)
- App Service Plan SKU and runtime stack

✅ Step 4: Define Shared Resources

In `main.tf`, define:

- A resource group
- A shared App Service Plan (`B1`, Windows)
- A loop (`for_each`) to call the web app module for each app name

✅ Step 5: Create the Web App Module

In `modules/webapp/`:

- Define a Windows Web App with `.NET 6` stack.
- Accept inputs for:

    - App name
    - Resource group
    - Location
    - Service Plan ID
- Output the default hostname of the app

✅ Step 6: Outputs

In the root `outputs.tf`:

- Output the URLs of all three web apps.

✅ Step 7: Deploy the Infrastructure

In the root terminal:

```
terraform init
terraform plan
terraform apply
```

Confirm with `yes` when prompted.

✅ Step 8: Validate and Test

- After deployment, the output should display all 3 app URLs.
- Open each URL in a browser to confirm the apps are deployed.

✅ Optional Cleanup

```
terraform destroy
```

Type `yes` when prompted.

Lab Completion

You’ve successfully used a **Terraform module** to deploy **three .NET 6 Web Apps** on a shared **B1 App Service Plan**, following best practices with reusable modules, variables, and remote state.