# Complete Documentation for Azure App Service Deployment with Terraform

This document provides a comprehensive explanation of the Terraform code for deploying an Azure App Service with a Virtual Network, staging slot, and delegation for secure access.

**Components:**

The deployment involves several Terraform files working together:

- **main.tf:** (The code we provide) Defines the resources for the Azure App Service infrastructure.
- **variables.tf:** (Separate file, assumed to be referenced) Holds variable definitions for customization.
- **output.tf** (Optional): Captures information about deployed resources after successful deployment.

**Functionality:**

1. **Azure Provider:** The provider block configures Terraform to interact with Azure resources.
2. **Variables:** The code utilizes variables defined in variables.tf to allow customization during deployment. These variables include:
    - location: Azure region for deployment (e.g., "West US")
    - resource_group_name: Name of the resource group (e.g., "my-app-service-rg")
    - app_service_name: Name of the App Service (e.g., "my-app-service")
    - app_service_plan_name: Name of the App Service plan (e.g., "my-app-service-plan")
    - app_service_plan_sku: SKU tier for the App Service plan (e.g., "B1", "S1")
    - app_service_plan_size (Optional): Size for the App Service plan (defaults to "F1")
    - app_settings (Optional): Map of key-value pairs for app settings
    - connection_strings (Optional): Map of key-value pairs for connection strings
3. **Resource Group:** An Azure resource group is created using azurerm_resource_group to hold all the deployed resources.
4. **Virtual Network:** The azurerm_virtual_network resource defines the virtual network with a subnet for the App Service. The subnet includes a dependency on the azurerm_app_service_virtual_network_swift_connection resource to ensure delegation is configured before the App Service is deployed.
5. **App Service Plan:** The azurerm_app_service_plan resource creates the App Service plan with the desired SKU and size.
6. **App Service:** The azurerm_app_service resource configures the App Service with:
    - Name, location, and resource group referencing defined variables.
    - app_service_plan_id referencing the previously created App Service plan.
    - Conditional logic to include app settings and connection strings only if provided during deployment.
    - depends_on referencing the virtual network to ensure the network exists before deploying the App Service.
    - subnet_id referencing the virtual network subnet for integration.

7. **Staging Slot:** The azurerm_app_service_plan_staging_environment resource enables the staging functionality for the App Service plan.
8. **Delegation:** The azurerm_app_service_virtual_network_swift_connection resource configures delegation, allowing the App Service to access resources within the subnet. It references the actual App Service name (azurerm_app_service.app.name).

**Benefits:**

- **Infrastructure as Code (IaC):** This Terraform configuration defines the infrastructure in code, promoting consistency, repeatability, and version control.
- **Customization:** Variables allow you to easily customize the deployment based on your specific needs.
- **Virtual Network Integration:** The App Service is integrated into a virtual network, providing additional security and control over network access.
- **Staging Slot:** The configuration enables a staging slot for testing and deploying new versions of your application without impacting the production environment.
- **Secure Access:** Delegation allows the App Service to access resources within the subnet securely.

Using the Code:

1. Prerequisites:
    - Install Terraform: https://developer.hashicorp.com/terraform/install
    - Configure your Azure credentials for Terraform: https://learn.microsoft.com/en-us/azure/developer/terraform/authenticate-to-azure
2. Place the code files:
    - Create a directory for your Terraform configuration.
    - Place the main.tf file in this directory.
    - Ensure variables.tf exists in the same directory or a parent directory accessible by Terraform.
    - The output.tf file is optional but recommended.
3. Edit variables.tf:
    - Update the variable values according to your requirements (names, location, SKU, etc.).
4. Run Terraform commands:
    - Initialize Terraform: terraform init
    - Review the execution plan: terraform plan
    - Deploy the resources: terraform apply
    - Destroy the resources (optional): terraform destroy

**Outputs (Optional):**

The output.tf file defines outputs that capture information about the deployed resources, such as IDs for the App Service, App Service plan, virtual network, subnet, and staging environment. These outputs can be helpful for managing resources or integrating them with other tools.