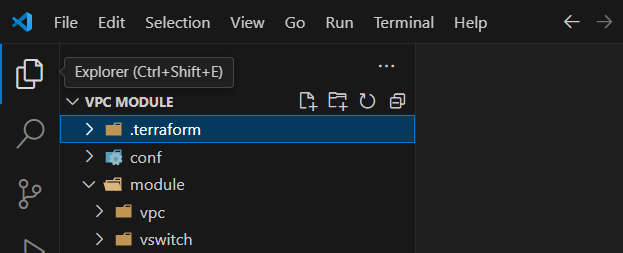
**MODULE**

A Terraform module is a collection of Terraform configuration files (.tf) that define a reusable chunk of infrastructure.

The .tf files in your working directory when you run the terraform plan or terraform apply together form the root module. That module may call other modules and connect them by passing output values from one to input values of another.

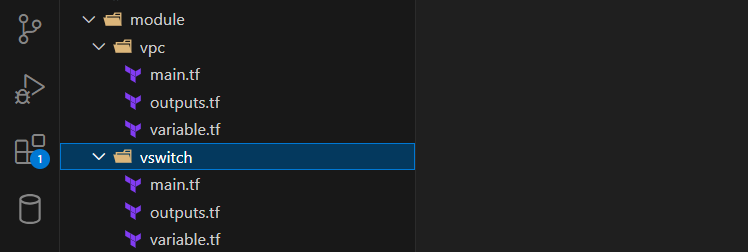
**Module Structure**

* Root Module: The main directory where you'll call the VPC and vSwitch modules.
* VPC Module (vpc/): A subdirectory containing the Terraform code for creating the VPC.
* vSwitch Module (vswitch/): A subdirectory containing the Terraform code for creating a vSwitch.

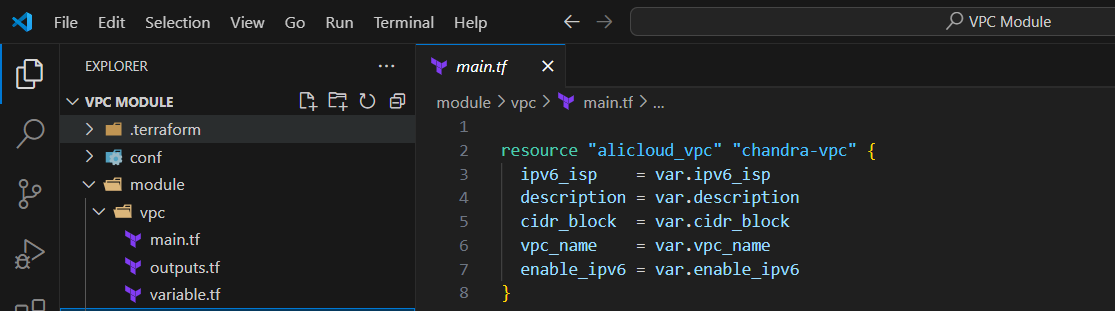


Inside each module directory (vpc/ and vswitch/)

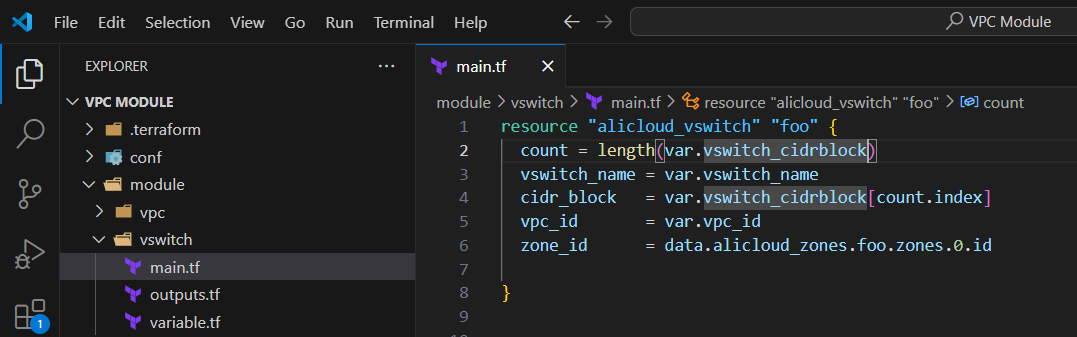
* main.tf: The primary Terraform configuration file.
* variables.tf: Defines input variables to make your module reusable.
* outputs.tf: Defines values that can be exported from the module for use in other parts of your configuration.



Example Code (vpc/main.tf)

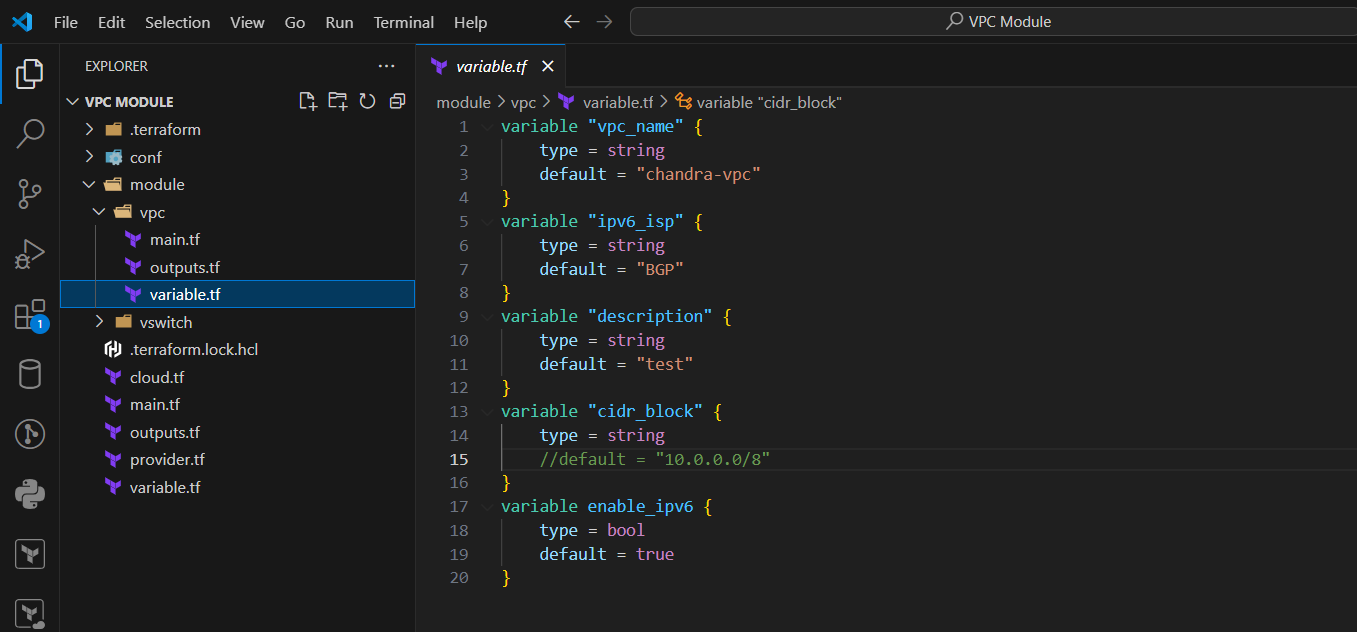


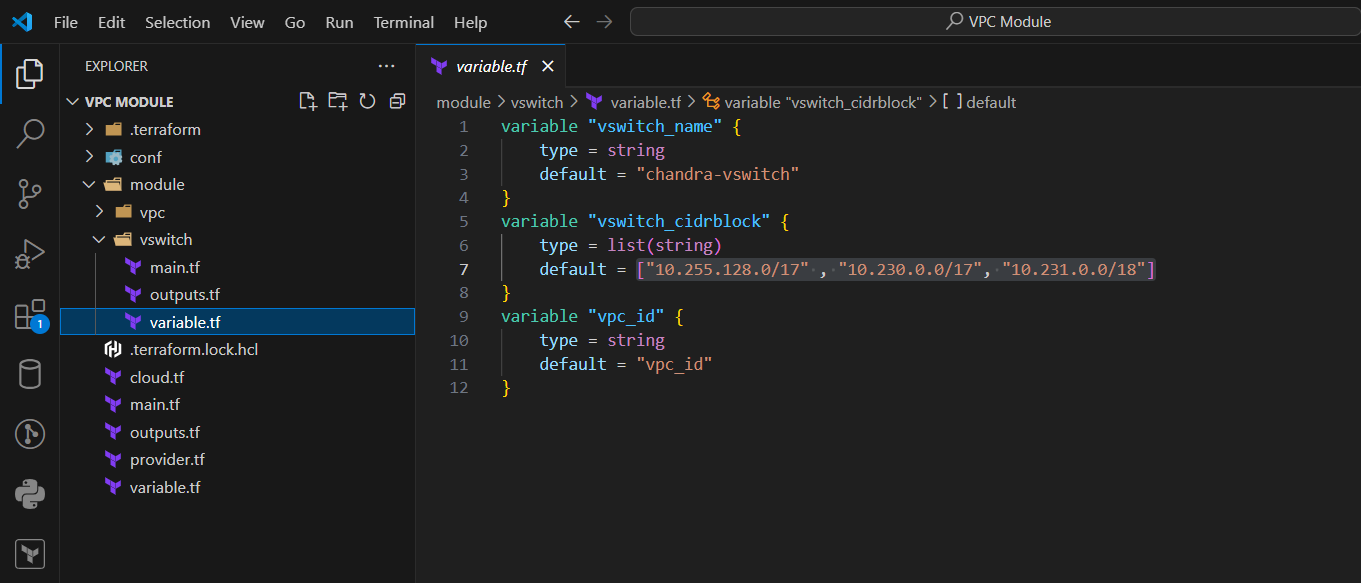
Example Code (vswitch/main.tf)



Variable Declarations (vpc/variables.tf, vswitch/variables.tf)

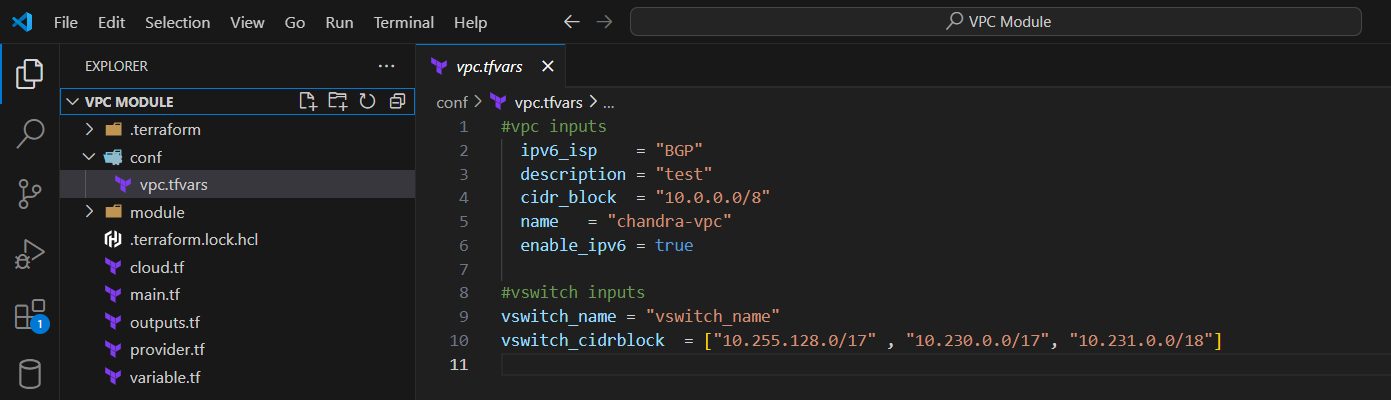
* Create variables.tf files in both module directories.
* Define variables for all configurable aspects.



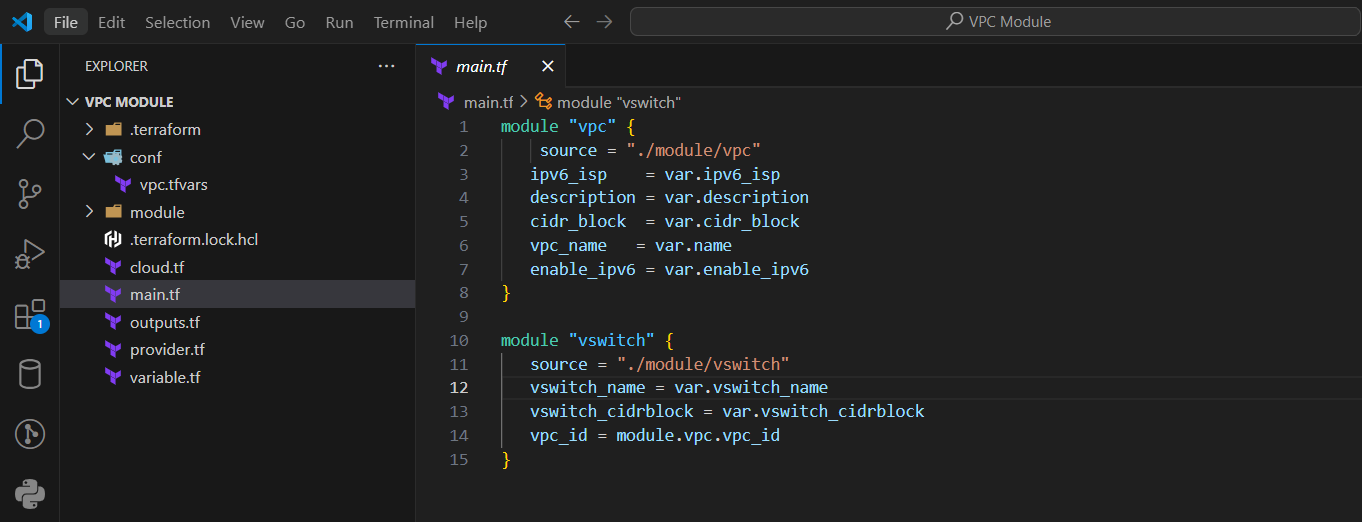


Configuration Plan:

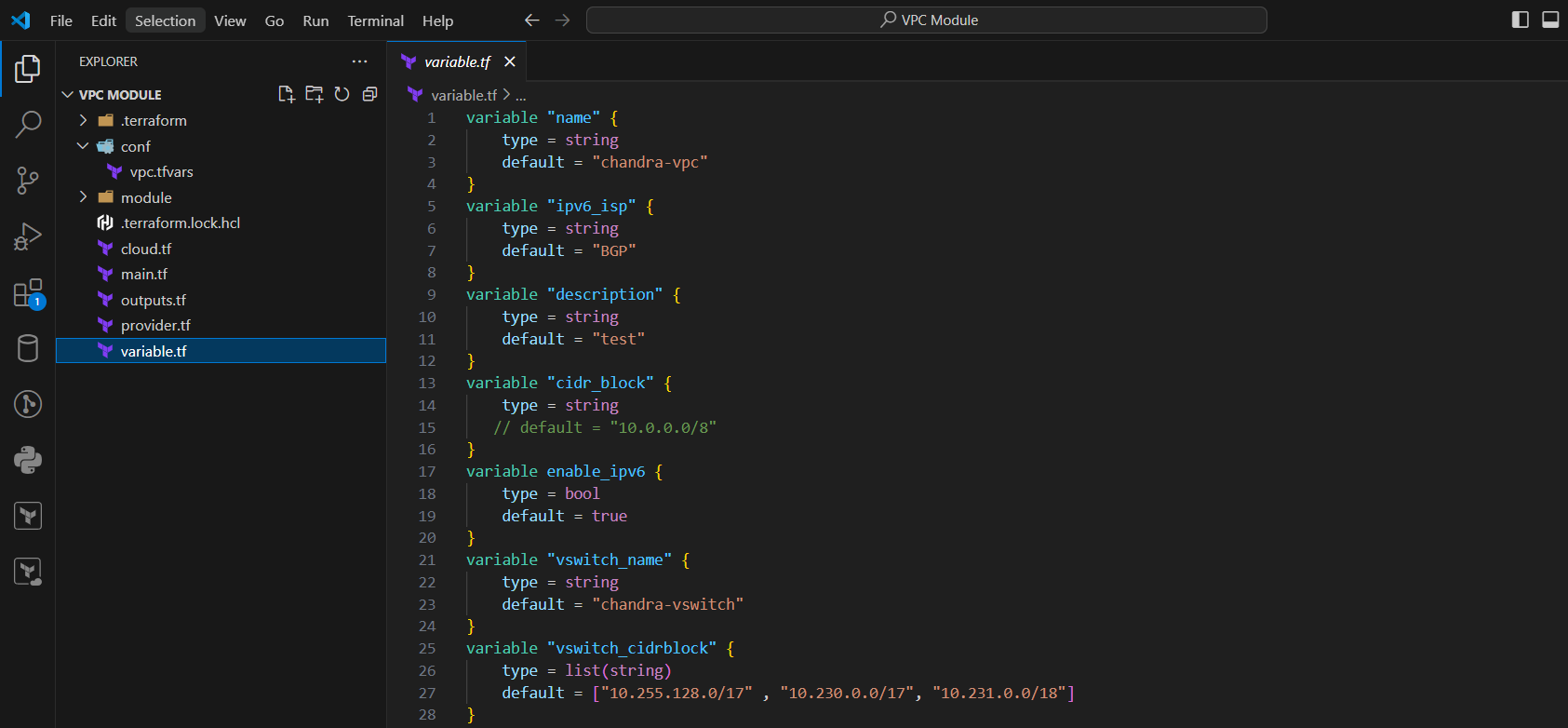
* tfvars: Decide the specific inputs and assign the values for your VPC and vSWITCH (name, size).



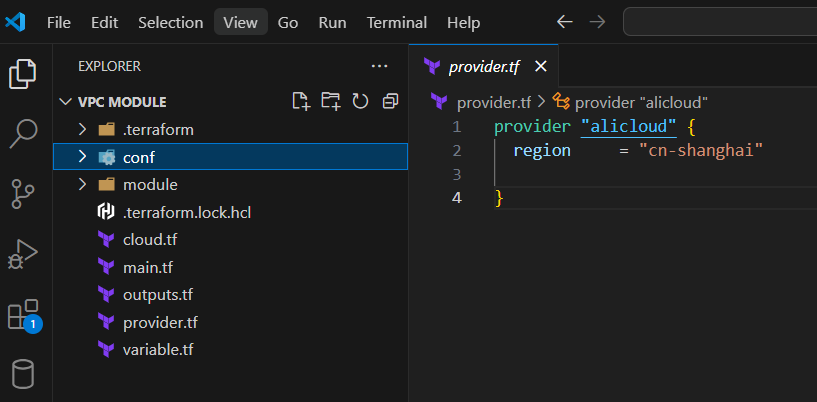
main.tf: Calls the vpc and vswitch modules, providing necessary values for variables and establishing the relationship between the two (the vSwitch is created within the VPC).



variables.tf : Declares additional variables used at the root module level.



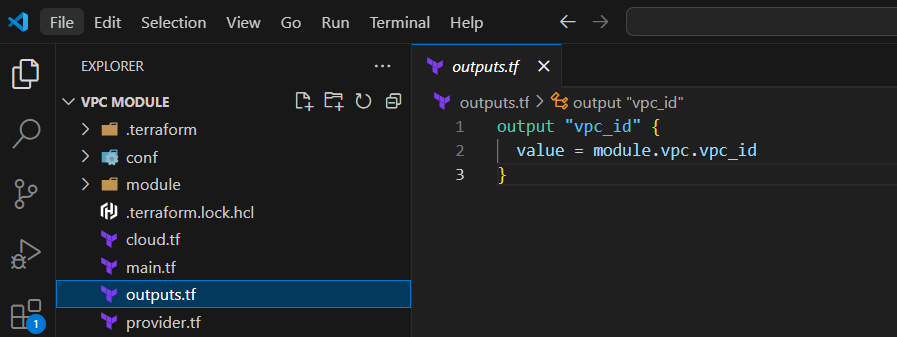
provider.tf: Configures the cloud provider (e.g., Alibaba Cloud) and any necessary authentication settings.



output.tf: The output.tf file defines values that you want Terraform to expose after it creates or modifies your infrastructure.

Why Outputs Matter

* When you run terraform apply, terraform will display the values defined in your outputs. You can then use these values: In Other Terraform Configs: Feed outputs from one module as inputs to another using remote state. In Scripts/Tools: Automate tasks based on the infrastructure created.
* Visibility: Outputs give you a concise view of the essential details of your deployed resources.



Basic Function of terraform init

The primary purposes of terraform init are:

* Plugin Downloads: Downloads the necessary provider plugins (e.g., AWS, Azure, Alibaba Cloud) required by the Terraform configuration in your current directory.
* Module Downloads: Fetches any external modules referenced in your Terraform code.
* State Initialization: Initializes the backend configuration (where your Terraform state file is stored).
* When to Use -var-file

Module-Specific Variables: Let's say you have modules (e.g., your VPC module) that use variables in variables.tf files. **terraform init -var-file="conf/vpc.tfvars"** achieves two things:

* It loads the variable values from your vpc.tfvars file.
* It checks if your module configurations are compatible with the provided variable values early in the workflow.

Multiple Environments: If you have separate .tfvars files for different environments (e.g., dev.tfvars, staging.tfvars, production.tfvars), the -var-file option lets you switch environments easily during initialization. For example:

terraform init -var-file="dev.tfvars"

Why Not Always Use -var-file?

* Simplicity: If your Terraform configuration doesn't rely heavily on modules with many variables, or you don't have separate variable files for environments, the plain terraform init is perfectly sufficient.
* Flexibility: You can choose to provide variables later using interactive prompts, command-line flags, or environment variables.

**WORKFLOW:**

Initialization (terraform init): Terraform downloads required plugins and initialize the working directory based on the modules present.

Planning (terraform plan): Terraform generates an execution plan, detailing the infrastructure changes that will be made based on the configuration and current state.

Deployment (terraform apply): Terraform instructs the cloud provider to create or modify infrastructure resources as defined in the plan.