

1-separate network resources into network module

```
File Edit View Terminal Tabs Help
outputs.tf ✕
% output "iti-vpc" > π value
8   output "iti-subnets" {
7   |   value = aws_subnet.iti_subnet
6   |
5   |
4   output "iti-sg" {
3   |   value = aws_security_group
2   |
1   output "iti-vpc" {
9   |   value = aws_vpc.iti-vpc
1   |
}
```

```
File Edit View Terminal Tabs Help
mod.tf ✕
% module "network"
9   source = "./mymodule/"
8
7   vpc_cidr_block = var.vpc_cidr_block
6
5   region = var.region
4
3   subnet_list = var.subnet_list
2
1   security_groups = var.security_groups
11
```

```
[mahmoud@hestia mymodule]$ ls
iti-igw.tf  iti-private-rt.tf  iti-sg.tf      iti-vpc.tf  vars.tf
iti-nat.tf  iti-public-rt.tf   iti-subnets.tf outputs.tf
[mahmoud@hestia mymodule]$
```

2-verify your email in ses service

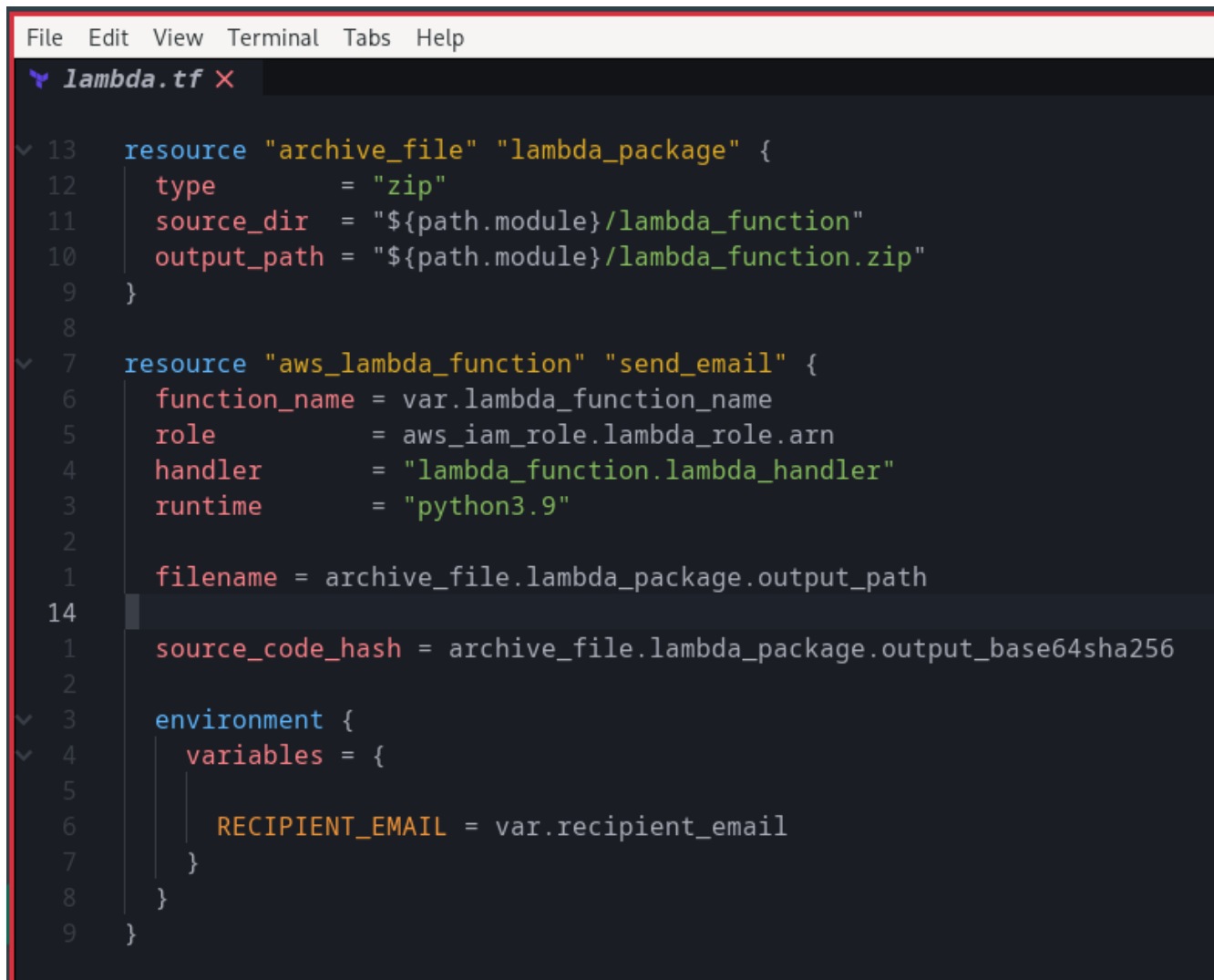
```
File Edit View Terminal Tabs Help
ses.tf x
resource "aws_ses_domain_identity" "ses_domain" > π domain
✓ 1 resource "aws_ses_domain_identity" "ses_domain" {
  2   domain = var.domain
  1   }
  2
```

### 3-create lambda function to send email

```
File Edit View Terminal Tabs Help
lambda-func.py X

9 import boto3
8 import os
7 from datetime import datetime
6
5 ses = boto3.client('ses')
4
3 def lambda_handler(event, context):
2     timestamp = datetime.utcnow().strftime('%Y-%m-%d %H:%M:%S UTC')
1     environment = os.environ['ENVIRONMENT']
10    sender = os.environ['SENDER_EMAIL']
1    receiver = os.environ['RECEIVER_EMAIL']
2
3    subject = f"State File - {environment} Environment"
4    body = f"Terraform state file has been changed.\n\nTime: {timestamp}\nEnvironment: {environment}"
5
6    response = ses.send_email(
7        Source=sender,
8        Destination={'ToAddresses': [receiver]},
9        Message={
10            'Subject': {'Data': subject},
11            'Body': {'Text': {'Data': body}}
12        }
13    )
14
15    return {"statusCode": 200, "body": "Email sent successfully!"}
```

4-create trigger to detect changes in state file and send the email  
in email mention time of change and environment (dev or prod)

A screenshot of a code editor window titled 'lambda.tf'. The editor shows Terraform configuration code for an AWS Lambda function. The code is color-coded and includes line numbers on the left. The configuration defines an 'archive\_file' resource to create a zip package from a directory, and an 'aws\_lambda\_function' resource that uses this package to create a Lambda function named 'send\_email'. The function is configured with a specific role, handler, and runtime. An environment variable 'RECIPIENT\_EMAIL' is also defined for the function.

```
File Edit View Terminal Tabs Help
lambda.tf ✕
13 resource "archive_file" "lambda_package" {
12     type      = "zip"
11     source_dir = "${path.module}/lambda_function"
10     output_path = "${path.module}/lambda_function.zip"
9     }
8
7 resource "aws_lambda_function" "send_email" {
6     function_name = var.lambda_function_name
5     role          = aws_iam_role.lambda_role.arn
4     handler       = "lambda_function.lambda_handler"
3     runtime       = "python3.9"
2
1     filename = archive_file.lambda_package.output_path
14
1     source_code_hash = archive_file.lambda_package.output_base64sha256
2
3     environment {
4         variables = {
5
6             RECIPIENT_EMAIL = var.recipient_email
7         }
8     }
9 }
```