



# **AWS Networking Infrastructure Using Powershell**

By: Salma Salah

Guidance & Support: Eng Saad El-Kennawy

# Task Scenario

Create a Networking Architecture containing:

- A VPC with CIDR entered by user and Tags entered by user
- Creating Internet Gateway
- With the IGW attached to the VPC
- Also Tags are Entered by the User
- Subnets With Number of Subnets and CIDR of each Subnet entered by the User
- Also Tags are Entered by the User
- Creating a route table with each subnet created subnet and attach it to the Subnet
- Creating a route with each subnet created subnet

# Task Scenario

Create a Networking Architecture containing:

- Creating NAT Gateway in a specified public subnet
- With Alerting the user that creation of a NAT GW will need an Elastic IP allocated first which will add additional cost
- Displaying NAT GW Details

# Part

# 01

**1. Adding Credentials and setting the session**

**2. Creating VPC with Tags**

- With VPC CIDR is Entered By the User
- Also Tags are Entered by the User

# First: Adding Credentials and setting the session



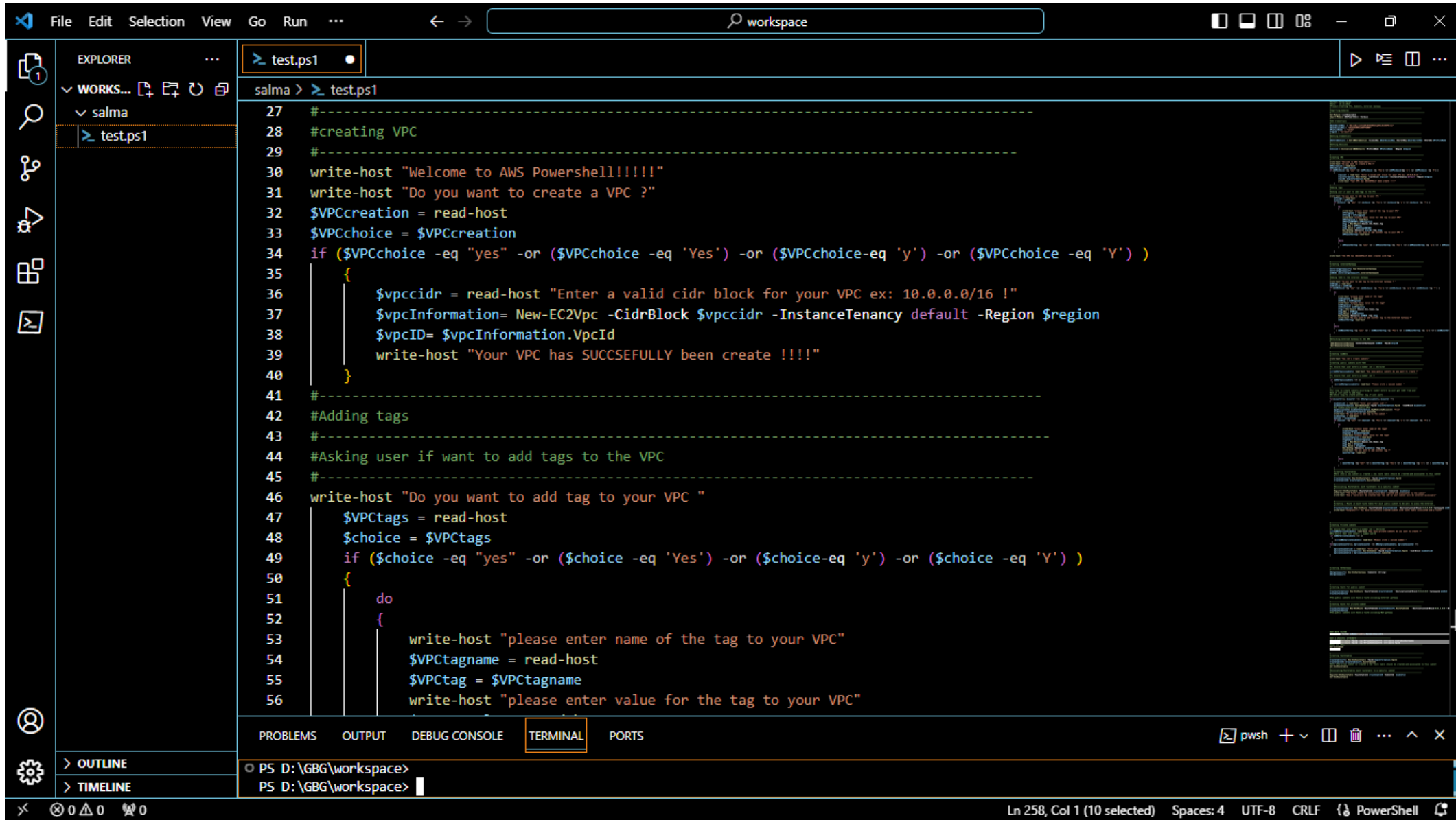
```
salma > test.ps1
1 #-----
2 #Author: Salma Salah
3 #Date: 18.12.2023
4 #Project:Creating VPC, Subnets, Internet Gateway
5 #-----
6 #Importing modules
7 #-----
8 Get-Module -ListAvailable
9 Import-Module AWSPowerShell -Verbose
10 #-----
11 #AWS Credentials
12 $UserSecretKey = "add your secret key"
13 $UserAccessKey = "add your access key"
14 $ProfileName = "salma"
15 $region = "us-east-1"
16 #-----
17 #Setting Credentials
18 #-----
19 $SetCredentials = Set-AWSCredential -AccessKey $UserAccessKey -SecretKey $UserSecretKey -StoreAs $ProfileName
20 #-----
21 #Setting Sessions
22 #-----
23 $session = Initialize-AWSDefaults -ProfileName $ProfileName -Region $region
24
25
26 #-----
27 #creating VPC
28 #-----
29 write-host "Welcome to AWS Powershell!!!!!"
30 write-host "Do you want to create a VPC ?"
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

- The VPC has SUCCSEFULLY been created with Tags
- PS D:\GBG\workspace> [int]\$Noofpublicsubnets= read-host "How many public subnets do you want to create ?"

Ln 13, Col 39 Spaces: 4 UTF-8 CRLF PowerShell

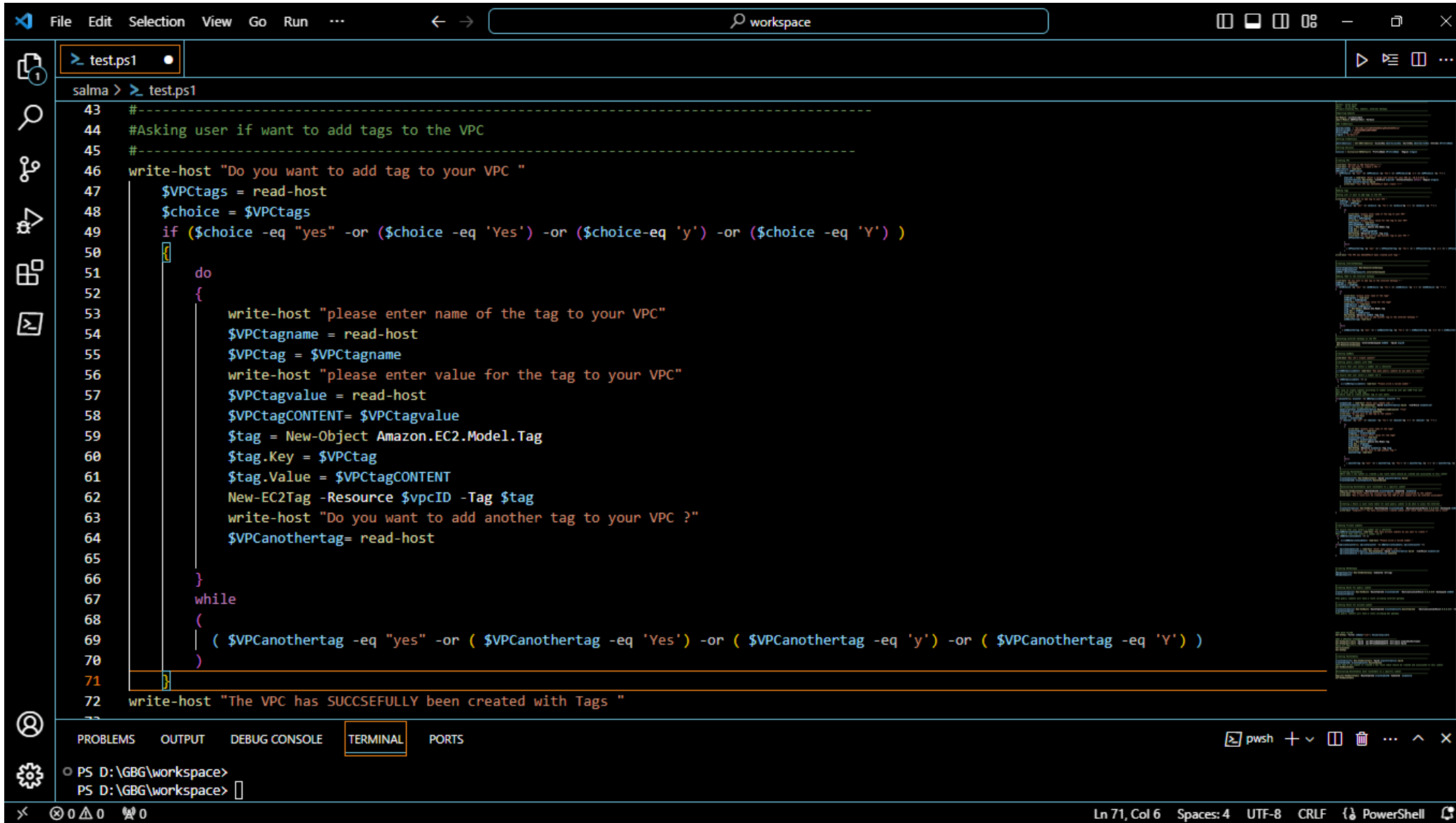
# Second: Creating VPC with Tags (1)



The screenshot shows the Visual Studio Code interface with a PowerShell script named `test.ps1` open in the editor. The script is designed to create a VPC and add tags to it. The Explorer sidebar on the left shows the file structure with `test.ps1` selected. The bottom status bar indicates the current line and column: `Ln 258, Col 1 (10 selected)`.

```
27 #-----
28 #creating VPC
29 #-----
30 write-host "Welcome to AWS Powershell!!!!!"
31 write-host "Do you want to create a VPC ?"
32 $VPCcreation = read-host
33 $VPCchoice = $VPCcreation
34 if ($VPCchoice -eq "yes" -or ($VPCchoice -eq 'Yes') -or ($VPCchoice -eq 'y') -or ($VPCchoice -eq 'Y'))
35 {
36     $vpccidr = read-host "Enter a valid cidr block for your VPC ex: 10.0.0.0/16 !"
37     $vpcInformation= New-EC2Vpc -CidrBlock $vpccidr -InstanceTenancy default -Region $region
38     $vpcID= $vpcInformation.VpcId
39     write-host "Your VPC has SUCCSEFULLY been create !!!!!"
40 }
41 #-----
42 #Adding tags
43 #-----
44 #Asking user if want to add tags to the VPC
45 #-----
46 write-host "Do you want to add tag to your VPC "
47 $VPCtags = read-host
48 $choice = $VPCtags
49 if ($choice -eq "yes" -or ($choice -eq 'Yes') -or ($choice -eq 'y') -or ($choice -eq 'Y'))
50 {
51     do
52     {
53         write-host "please enter name of the tag to your VPC"
54         $VPCtagname = read-host
55         $VPCtag = $VPCtagname
56         write-host "please enter value for the tag to your VPC"
```

## Second: Creating VPC with Tags (2)



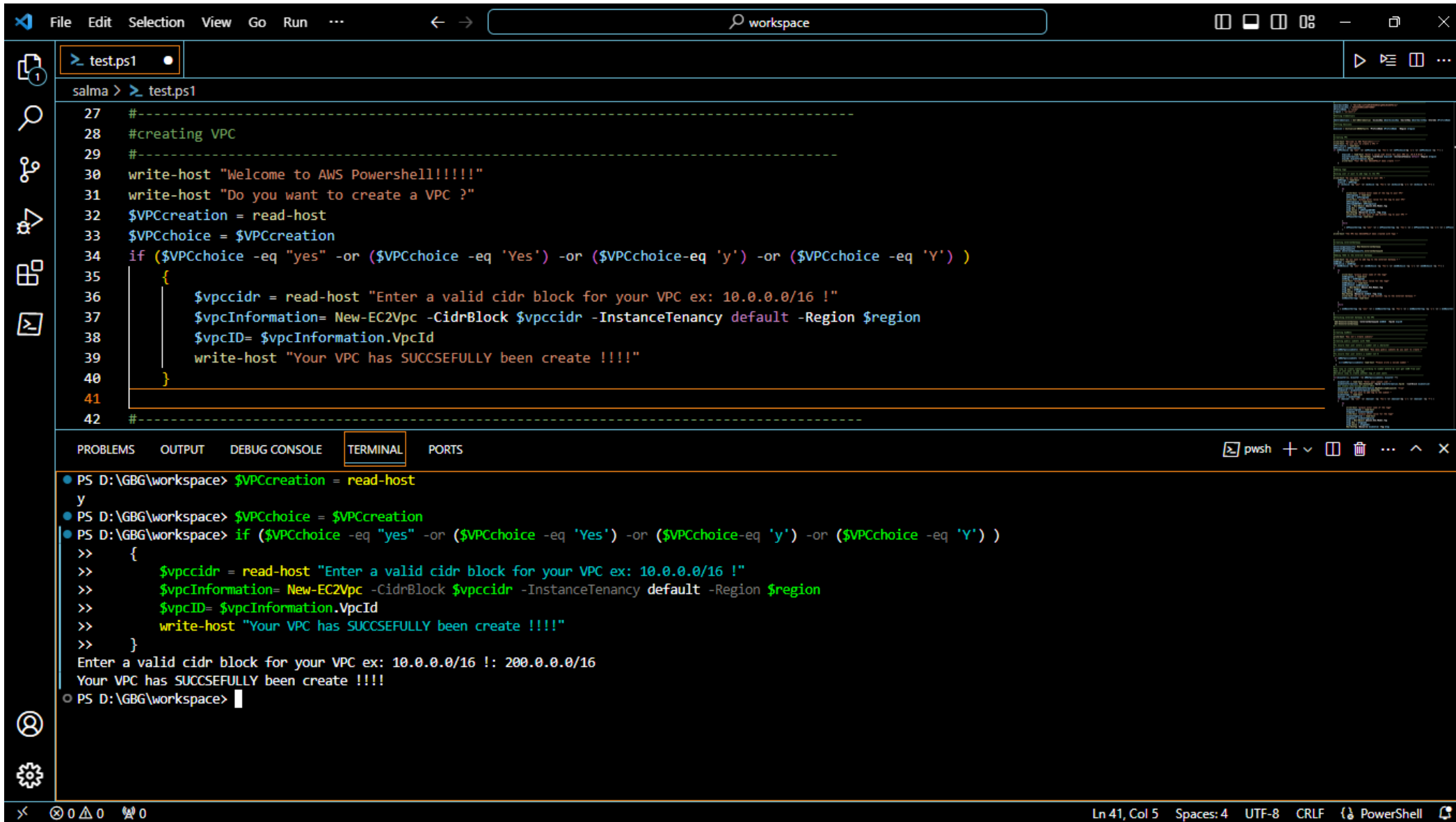
```
File Edit Selection View Go Run ... workspace
> test.ps1
salma > > test.ps1
43 #-----
44 #Asking user if want to add tags to the VPC
45 #-----
46 write-host "Do you want to add tag to your VPC "
47 $VPCTags = read-host
48 $choice = $VPCTags
49 if ($choice -eq "yes" -or ($choice -eq 'Yes') -or ($choice -eq 'y') -or ($choice -eq 'Y'))
50 {
51     do
52     {
53         write-host "please enter name of the tag to your VPC"
54         $VPCTagname = read-host
55         $VPCTag = $VPCTagname
56         write-host "please enter value for the tag to your VPC"
57         $VPCTagvalue = read-host
58         $VPCTagCONTENT= $VPCTagvalue
59         $tag = New-Object Amazon.EC2.Model.Tag
60         $tag.Key = $VPCTag
61         $tag.Value = $VPCTagCONTENT
62         New-EC2Tag -Resource $vpcID -Tag $tag
63         write-host "Do you want to add another tag to your VPC ?"
64         $VPCanotherTag= read-host
65     }
66     while
67     (
68         ( $VPCanotherTag -eq "yes" -or ( $VPCanotherTag -eq 'Yes') -or ( $VPCanotherTag -eq 'y') -or ( $VPCanotherTag -eq 'Y'))
69     )
70 }
71
72 write-host "The VPC has SUCCSEFULLY been created with Tags "
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

PS D:\GBG\workspace>  
PS D:\GBG\workspace>

Ln 71, Col 6 Spaces: 4 UTF-8 CRLF PowerShell

# Third: The user Enters VPC CIDR



The screenshot shows the Visual Studio Code interface with a PowerShell script named `test.ps1` open in the editor. The script is designed to create an AWS VPC. It prompts the user for confirmation to create a VPC and then for a valid CIDR block. The script uses `New-EC2Vpc` to create the VPC with a default instance tenancy and a specified region.

```
27 #-----
28 #creating VPC
29 #-----
30 write-host "Welcome to AWS Powershell!!!!!"
31 write-host "Do you want to create a VPC ?"
32 $VPCcreation = read-host
33 $VPCchoice = $VPCcreation
34 if ($VPCchoice -eq "yes" -or ($VPCchoice -eq 'Yes') -or ($VPCchoice -eq 'y') -or ($VPCchoice -eq 'Y') )
35 {
36     $vpccidr = read-host "Enter a valid cidr block for your VPC ex: 10.0.0.0/16 !"
37     $vpcInformation= New-EC2Vpc -CidrBlock $vpccidr -InstanceTenancy default -Region $region
38     $vpcID= $vpcInformation.VpcId
39     write-host "Your VPC has SUCCSEFULLY been create !!!!"
```

The terminal window shows the execution of the script. The user enters 'y' for confirmation and '200.0.0.0/16' for the CIDR block. The script successfully creates the VPC and displays the VPC ID.

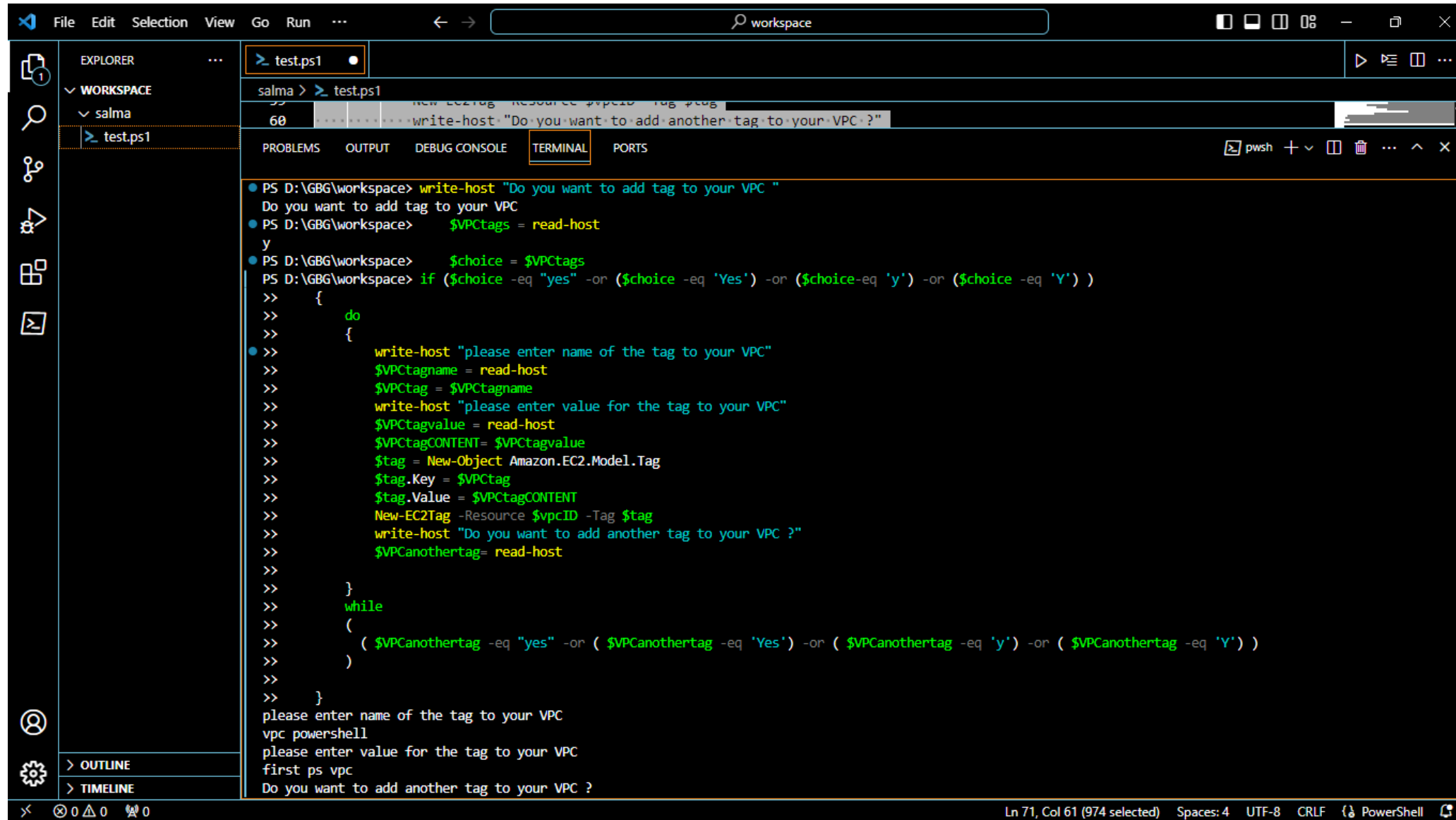
```
PS D:\GBG\workspace> $VPCcreation = read-host
y
PS D:\GBG\workspace> $VPCchoice = $VPCcreation
PS D:\GBG\workspace> if ($VPCchoice -eq "yes" -or ($VPCchoice -eq 'Yes') -or ($VPCchoice -eq 'y') -or ($VPCchoice -eq 'Y') )
>> {
>>     $vpccidr = read-host "Enter a valid cidr block for your VPC ex: 10.0.0.0/16 !"
>>     $vpcInformation= New-EC2Vpc -CidrBlock $vpccidr -InstanceTenancy default -Region $region
>>     $vpcID= $vpcInformation.VpcId
>>     write-host "Your VPC has SUCCSEFULLY been create !!!!"
```

Enter a valid cidr block for your VPC ex: 10.0.0.0/16 !: 200.0.0.0/16  
Your VPC has SUCCSEFULLY been create !!!!  
PS D:\GBG\workspace>

The status bar at the bottom indicates the current line is 41, column 5, with 4 spaces, UTF-8 encoding, CRLF line endings, and PowerShell as the shell.



# Third: The user Enters VPC Tags (1)



The screenshot shows the Visual Studio Code interface with a PowerShell script named `test.ps1` open in the editor. The script is designed to add tags to a VPC. It starts by asking the user if they want to add a tag. If the user responds with 'y', 'yes', 'Yes', 'Y', or 'y', it prompts for the tag name and value, then creates a new tag and attaches it to the VPC. It also asks if the user wants to add another tag.

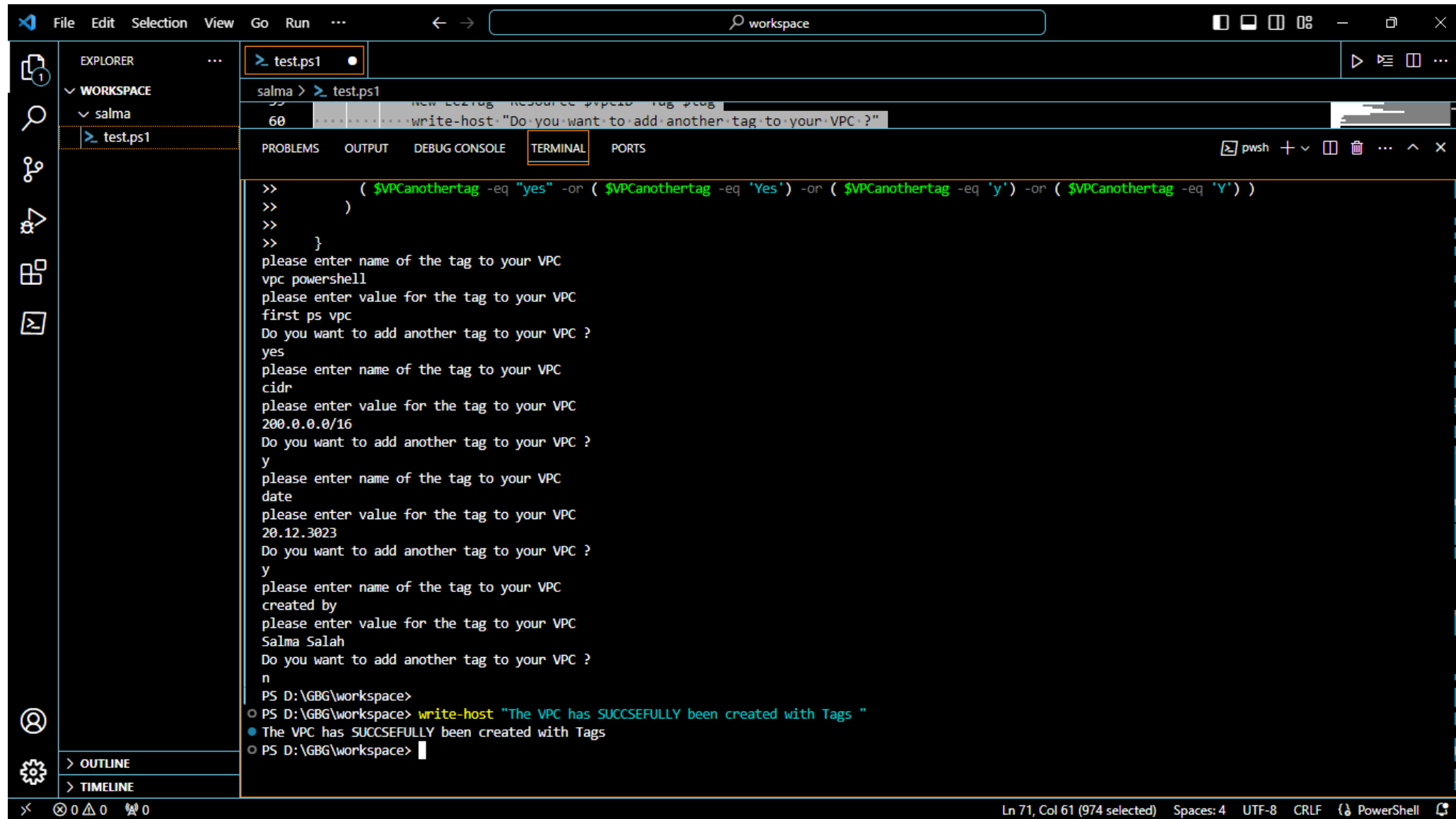
```
salma > test.ps1
60 ..... write-host "Do you want to add another tag to your VPC?"

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

• PS D:\GBG\workspace> write-host "Do you want to add tag to your VPC "
Do you want to add tag to your VPC
• PS D:\GBG\workspace> $VPCTags = read-host
y
• PS D:\GBG\workspace> $choice = $VPCTags
PS D:\GBG\workspace> if ($choice -eq "yes" -or ($choice -eq 'Yes') -or ($choice -eq 'y') -or ($choice -eq 'Y')) {
    do
    {
        write-host "please enter name of the tag to your VPC"
        $VPCTagname = read-host
        $VPCTag = $VPCTagname
        write-host "please enter value for the tag to your VPC"
        $VPCTagvalue = read-host
        $VPCTagCONTENT= $VPCTagvalue
        $tag = New-Object Amazon.EC2.Model.Tag
        $tag.Key = $VPCTag
        $tag.Value = $VPCTagCONTENT
        New-EC2Tag -Resource $vpcID -Tag $tag
        write-host "Do you want to add another tag to your VPC ?"
        $VPCanothertag= read-host
    }
    while
    (
        ( $VPCanothertag -eq "yes" -or ( $VPCanothertag -eq 'Yes') -or ( $VPCanothertag -eq 'y') -or ( $VPCanothertag -eq 'Y') )
    )
}
please enter name of the tag to your VPC
vpc powershell
please enter value for the tag to your VPC
first ps vpc
Do you want to add another tag to your VPC ?
```

Ln 71, Col 61 (974 selected) Spaces: 4 UTF-8 CRLF PowerShell

## Third: The user Enters VPC Tags (2)



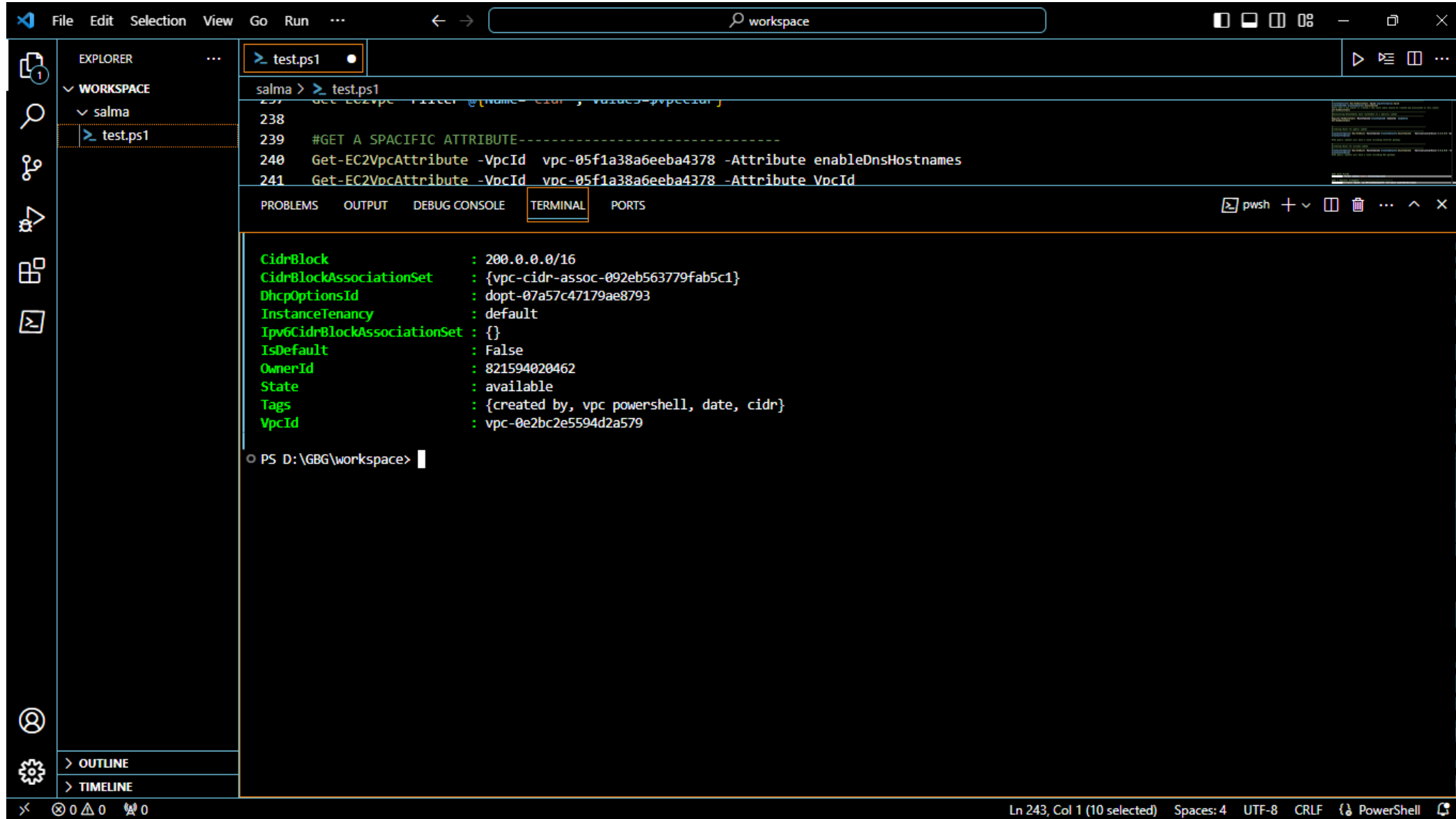
```
salma > test.ps1
60 .....write-host "Do you want to add another tag to your VPC ?"

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS pwsh + - ... ^ x

>> ( $VPCanothertag -eq "yes" -or ( $VPCanothertag -eq 'Yes') -or ( $VPCanothertag -eq 'y') -or ( $VPCanothertag -eq 'Y') )
>> )
>> }
please enter name of the tag to your VPC
vpc powershell
please enter value for the tag to your VPC
first ps vpc
Do you want to add another tag to your VPC ?
yes
please enter name of the tag to your VPC
cidr
please enter value for the tag to your VPC
200.0.0.0/16
Do you want to add another tag to your VPC ?
y
please enter name of the tag to your VPC
date
please enter value for the tag to your VPC
20.12.3023
Do you want to add another tag to your VPC ?
y
please enter name of the tag to your VPC
created by
please enter value for the tag to your VPC
Salma Salah
Do you want to add another tag to your VPC ?
n
PS D:\GBG\workspace>
o PS D:\GBG\workspace> write-host "The VPC has SUCCSEFULLY been created with Tags "
o The VPC has SUCCSEFULLY been created with Tags
o PS D:\GBG\workspace>
```

Ln 71, Col 61 (974 selected) Spaces: 4 UTF-8 CRLF PowerShell

## Fourth: VPC After Creation with Tags



The screenshot shows the Visual Studio Code interface with a PowerShell script in the editor and its output in the terminal. The Explorer sidebar on the left shows a workspace named 'salma' with a file 'test.ps1'. The terminal window is active, displaying the execution of the script. The script contains commands to get VPC attributes, and the terminal shows the resulting JSON output for a specific VPC.

```
salma > test.ps1
237 Get-EC2Vpc -Filter @{"Name"="cidr"; "Values"=$vpc.cidr}
238
239 #GET A SPACIFIC ATTRIBUTE-----
240 Get-EC2VpcAttribute -VpcId vpc-05f1a38a6eeba4378 -Attribute enableDnsHostnames
241 Get-EC2VpcAttribute -VpcId vpc-05f1a38a6eeba4378 -Attribute VpcId
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

```
CidrBlock           : 200.0.0.0/16
CidrBlockAssociationSet : {vpc-cidr-assoc-092eb563779fab5c1}
DhcpOptionsId       : dopt-07a57c47179ae8793
InstanceTenancy     : default
Ipv6CidrBlockAssociationSet : {}
IsDefault           : False
OwnerId             : 821594020462
State               : available
Tags                : {created by, vpc powershell, date, cidr}
VpcId               : vpc-0e2bc2e5594d2a579
```

PS D:\GBG\workspace>

Ln 243, Col 1 (10 selected) Spaces: 4 UTF-8 CRLF PowerShell

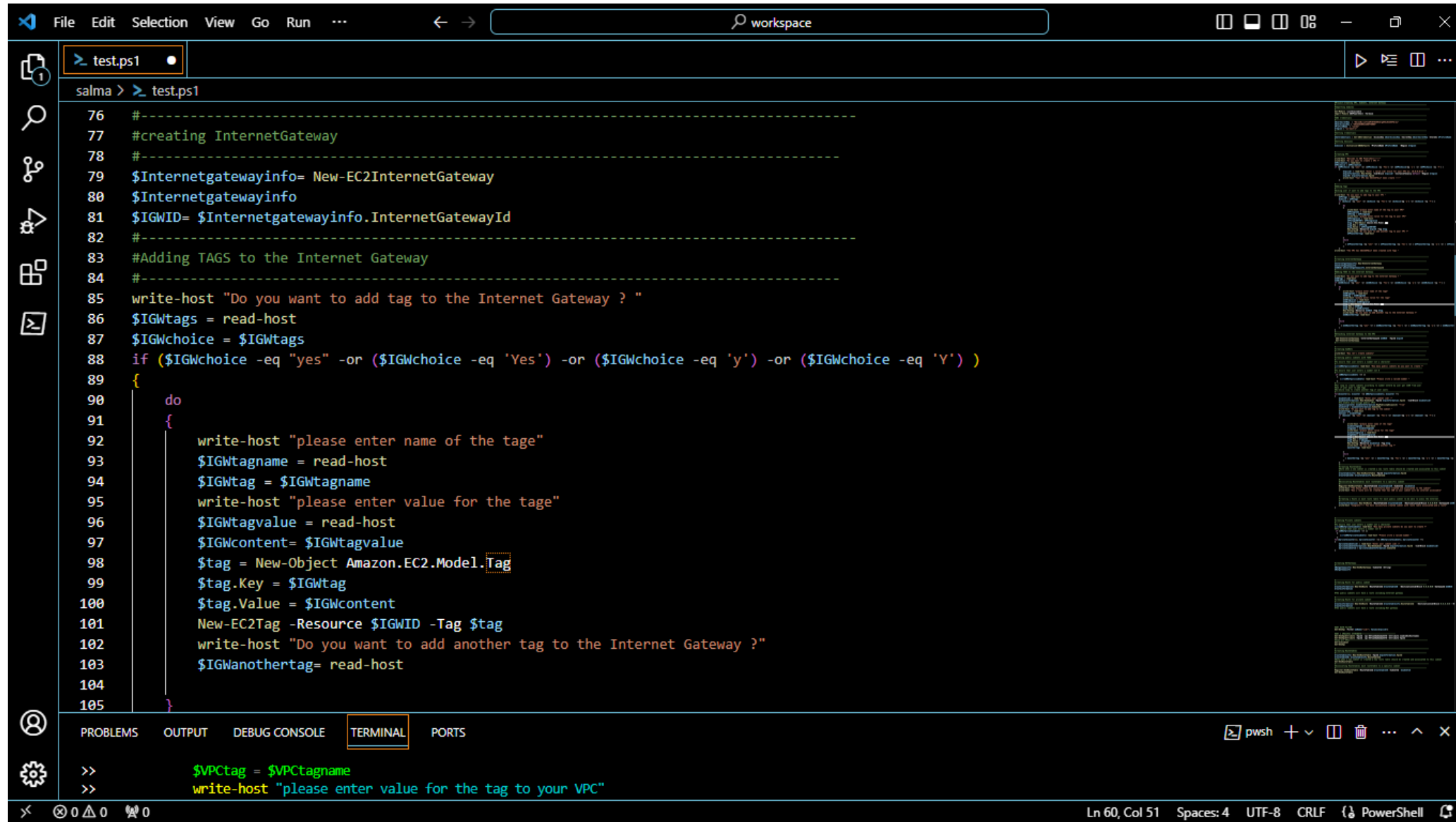
# Part

# 02

## Creating Internet Gateway

- With the IGW attached to the VPC
- Also Tags are Entered by the User

# First: Crating Internet Gateway With Tags



```
File Edit Selection View Go Run ... workspace
> test.ps1
salma > test.ps1

76 #-----
77 #creating InternetGateway
78 #-----
79 $Internetgatewayinfo= New-EC2InternetGateway
80 $Internetgatewayinfo
81 $IGWID= $Internetgatewayinfo.InternetGatewayId
82 #-----
83 #Adding TAGS to the Internet Gateway
84 #-----
85 write-host "Do you want to add tag to the Internet Gateway ? "
86 $IGWtags = read-host
87 $IGWchoice = $IGWtags
88 if ($IGWchoice -eq "yes" -or ($IGWchoice -eq 'Yes') -or ($IGWchoice -eq 'y') -or ($IGWchoice -eq 'Y') )
89 {
90     do
91     {
92         write-host "please enter name of the tage"
93         $IGWtagname = read-host
94         $IGWtag = $IGWtagname
95         write-host "please enter value for the tage"
96         $IGWtagvalue = read-host
97         $IGWcontent= $IGWtagvalue
98         $tag = New-Object Amazon.EC2.Model.Tag
99         $tag.Key = $IGWtag
100         $tag.Value = $IGWcontent
101         New-EC2Tag -Resource $IGWID -Tag $tag
102         write-host "Do you want to add another tag to the Internet Gateway ?"
103         $IGWanotherntag= read-host
104     }
105 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
>> $VPCTag = $VPCTagname
>> write-host "please enter value for the tag to your VPC"
```

Ln 60, Col 51 Spaces: 4 UTF-8 CRLF PowerShell

## Second: Attaching Internet Gateway to the VPC



The screenshot shows a Visual Studio Code editor with a dark theme. The top menu bar includes File, Edit, Selection, View, Go, Run, and a search bar containing 'workspace'. The left sidebar has icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The main editor area displays a PowerShell script in a file named 'test.ps1'. The script is written in a PowerShell console session with the prompt 'salma >'. The script includes a 'while' loop, a comment '#Attaching Internet Gateway to the VPC', and two AWS CLI commands: 'Add-EC2InternetGateway' and 'Get-EC2InternetGateway'. The script is formatted with syntax highlighting, and line numbers 104 through 117 are visible on the left margin.

```
salma > test.ps1

104 |
105 | }
106 | while
107 | (
108 | ( $IGWanothertag -eq "yes" -or ( $IGWanothertag -eq 'Yes') -or ( $IGWanothertag -eq 'y') -or ( $IGWanothertag -eq 'Y') )
109 | )
110 | }
111 | #-----
112 | #Attaching Internet Gateway to the VPC
113 | #-----
114 | Add-EC2InternetGateway -InternetGatewayId $IGWID -VpcId $vpcID
115 | Get-EC2InternetGateway
116 | #-----
117 |
```

# Third: Crating Internet Gateway With Tags (During Run(1))

The screenshot shows the Visual Studio Code interface with a PowerShell script in the editor and its execution output in the terminal.

**EXPLORER**

- WORKSPACE
  - salma
    - test.ps1

**test.ps1**

```
salma > test.ps1
135 {
136     $privatesubnetcidr = read-host "Enter your subnet cidr !"
137     $privatesubnetInformation= New-EC2Subnet -VpcId $vpcInformation.VpcId -CidrBlock $subnetcidr
138     $privatesubnetid = $privatesubnetInformation.SubnetId
139 }
140
141 #-----
142 #creating InternetGateway
143 #-----
144 $Internetgatewayinfo= New-EC2InternetGateway
145 $Internetgatewayinfo
146 $IGWID= $Internetgatewayinfo.InternetGatewayId
147 #-----
```

**TERMINAL**

```
PS D:\GBG\workspace> $Internetgatewayinfo= New-EC2InternetGateway
PS D:\GBG\workspace> $Internetgatewayinfo

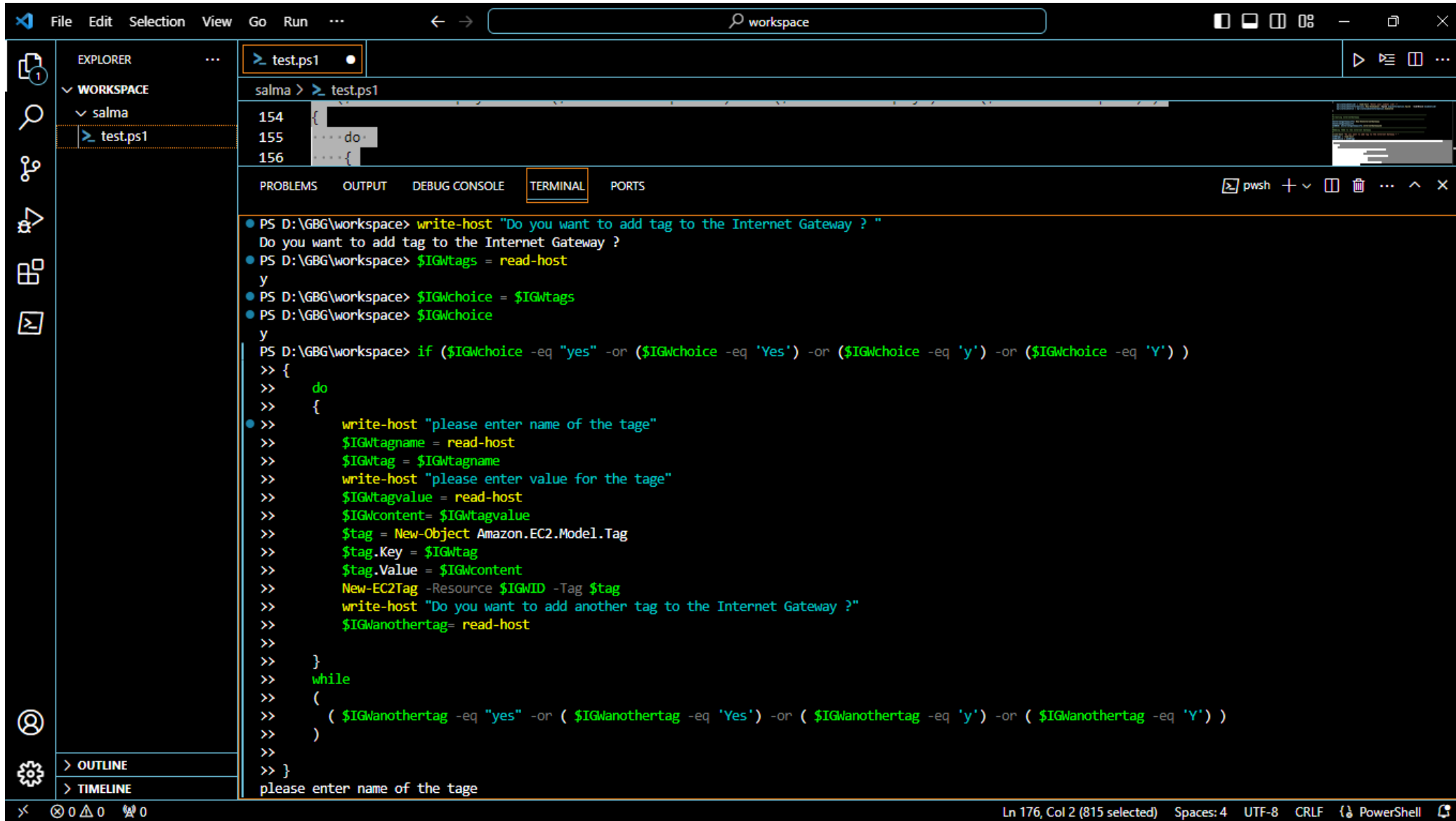
Attachments InternetGatewayId OwnerId Tags
-----
{} igw-0d1b681dffa91fe07a 821594020462 {}

PS D:\GBG\workspace> $IGWID= $Internetgatewayinfo.InternetGatewayId
PS D:\GBG\workspace> write-host "Do you want to add tag to the Internet Gateway ? "
Do you want to add tag to the Internet Gateway ?
PS D:\GBG\workspace> $IGWtags = read-host
GWtagname
PS D:\GBG\workspace> $IGWchoice = $IGWtags
PS D:\GBG\workspace> if ($IGWchoice -eq "yes" -or ($IGWchoice -eq 'Yes') -or ($IGWchoice -eq 'y') -or ($IGWchoice -eq 'Y') )
>> {
>>     do
>>     {
>>         write-host "please enter name of the tage"
>>         $IGWtagname = read-host
>>         $IGWtag = $I write-host "please enter value for the tage"
```

**STATUS BAR**

Ln 164, Col 27 Spaces: 4 UTF-8 CRLF PowerShell

## Third: Crating Internet Gateway With Tags (During Run (2))

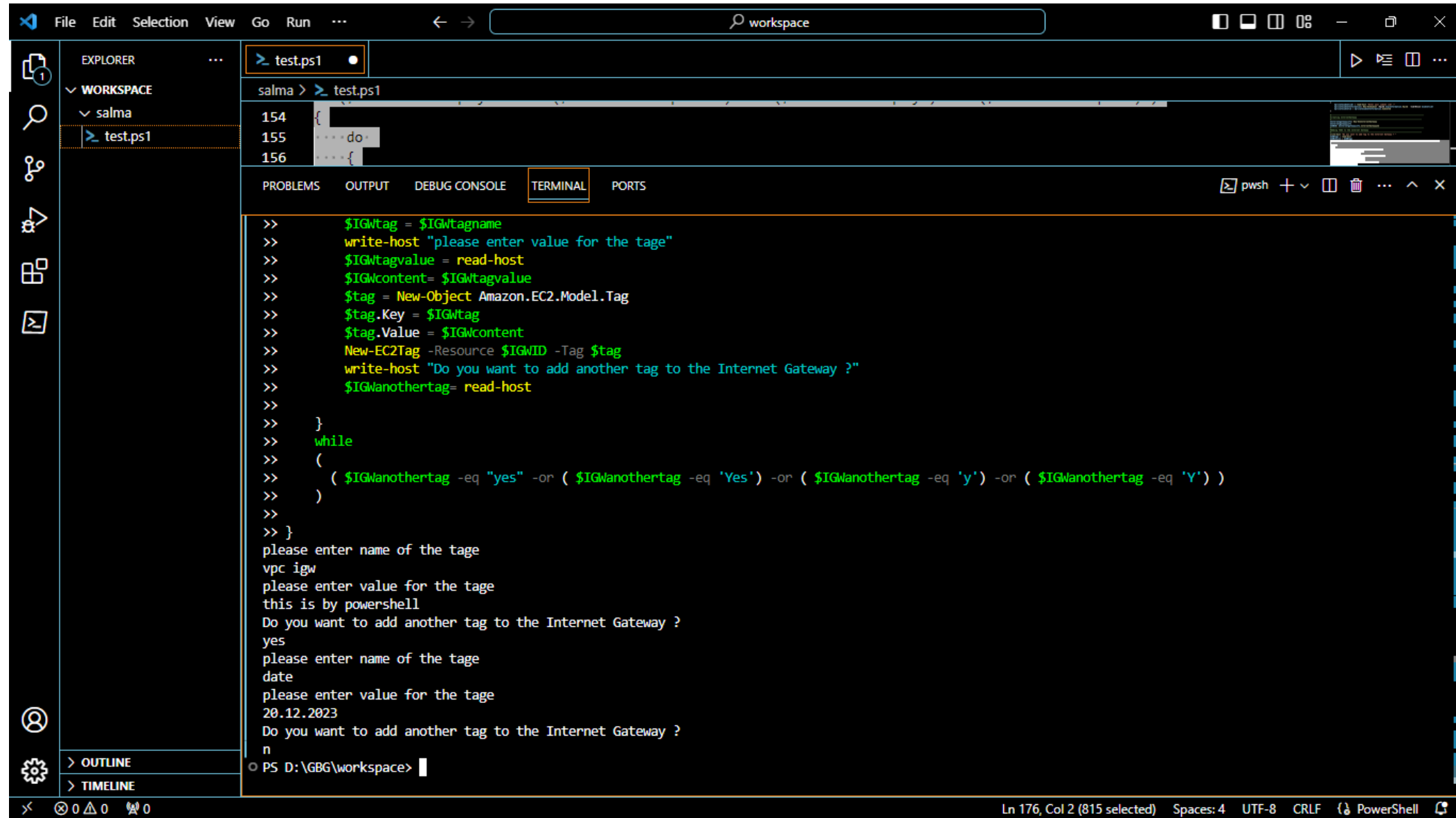


```
File Edit Selection View Go Run ... workspace
EXPLORER
WORKSPACE
salma
test.ps1
test.ps1
154 {
155     do
156     {
PS D:\GBG\workspace> write-host "Do you want to add tag to the Internet Gateway ? "
Do you want to add tag to the Internet Gateway ?
PS D:\GBG\workspace> $IGWtags = read-host
y
PS D:\GBG\workspace> $IGWchoice = $IGWtags
PS D:\GBG\workspace> $IGWchoice
y
PS D:\GBG\workspace> if ($IGWchoice -eq "yes" -or ($IGWchoice -eq 'Yes') -or ($IGWchoice -eq 'y') -or ($IGWchoice -eq 'Y') )
>> {
>>     do
>>     {
>>         write-host "please enter name of the tage"
>>         $IGWtagname = read-host
>>         $IGWtag = $IGWtagname
>>         write-host "please enter value for the tage"
>>         $IGWtagvalue = read-host
>>         $IGWcontent= $IGWtagvalue
>>         $tag = New-Object Amazon.EC2.Model.Tag
>>         $tag.Key = $IGWtag
>>         $tag.Value = $IGWcontent
>>         New-EC2Tag -Resource $IGWID -Tag $tag
>>         write-host "Do you want to add another tag to the Internet Gateway ?"
>>         $IGWanotherntag= read-host
>>     }
>>     while
>>     (
>>         ( $IGWanotherntag -eq "yes" -or ( $IGWanotherntag -eq 'Yes') -or ( $IGWanotherntag -eq 'y') -or ( $IGWanotherntag -eq 'Y') )
>>     )
>> }
>> }
please enter name of the tage
```

Ln 176, Col 2 (815 selected) Spaces: 4 UTF-8 CRLF PowerShell



## Fourth: User Enters the Tags

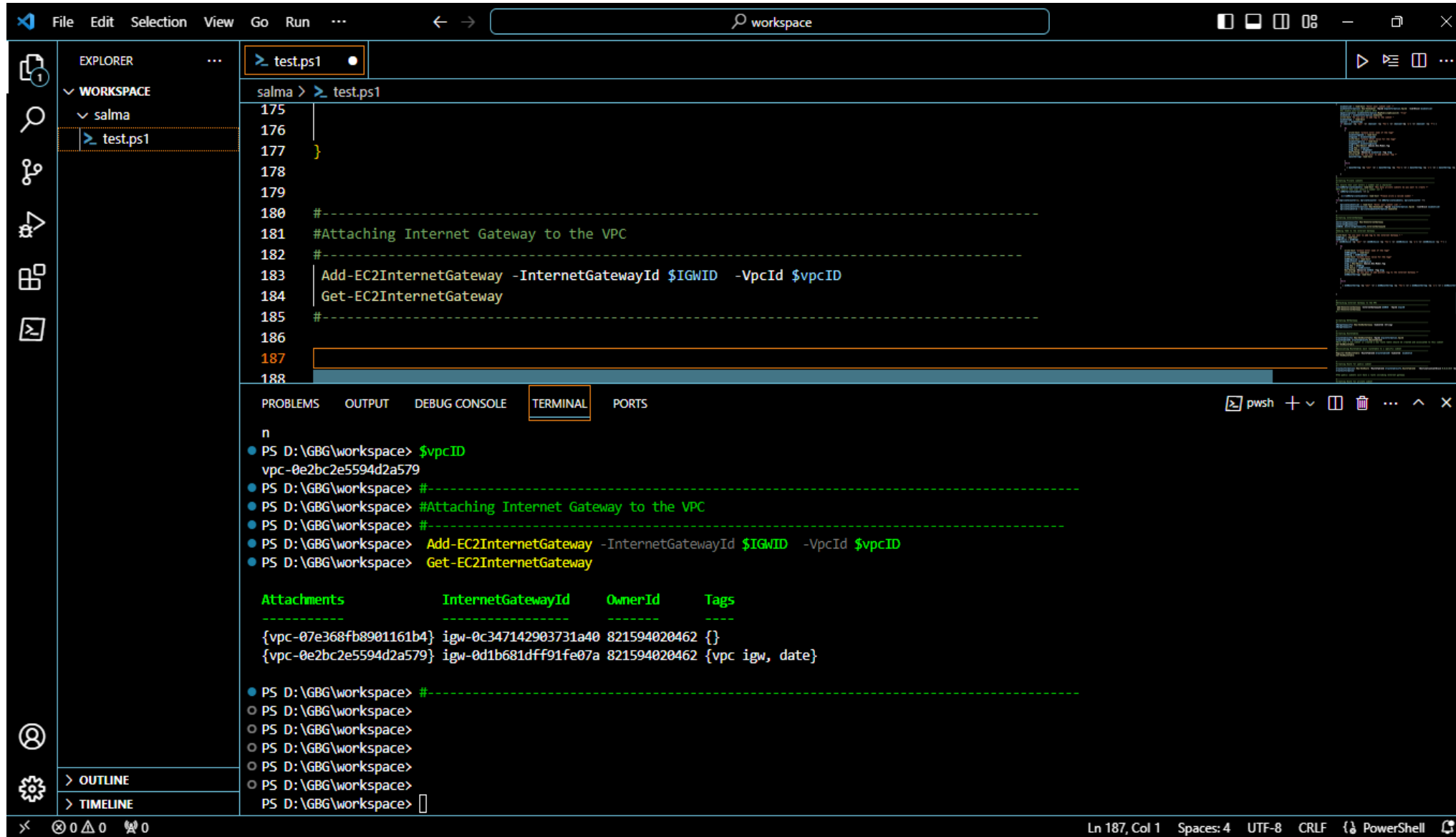


The screenshot shows the Visual Studio Code interface with a PowerShell script in `test.ps1` and its execution in the terminal. The Explorer sidebar shows the file `test.ps1` under the `salma` workspace. The terminal window shows the script's execution, including prompts for tag name, value, and whether to add another tag.

```
> test.ps1
salma > test.ps1
154 {
155     do
156     {
    >>     $IGWtag = $IGWtagname
    >>     write-host "please enter value for the tage"
    >>     $IGWtagvalue = read-host
    >>     $IGWcontent= $IGWtagvalue
    >>     $tag = New-Object Amazon.EC2.Model.Tag
    >>     $tag.Key = $IGWtag
    >>     $tag.Value = $IGWcontent
    >>     New-EC2Tag -Resource $IGWID -Tag $tag
    >>     write-host "Do you want to add another tag to the Internet Gateway ?"
    >>     $IGWanothertag= read-host
    >>
    >> }
    >> while
    >> (
    >> ( $IGWanothertag -eq "yes" -or ( $IGWanothertag -eq 'Yes') -or ( $IGWanothertag -eq 'y') -or ( $IGWanothertag -eq 'Y') )
    >> )
    >> }
please enter name of the tage
vpc igw
please enter value for the tage
this is by powershell
Do you want to add another tag to the Internet Gateway ?
yes
please enter name of the tage
date
please enter value for the tage
20.12.2023
Do you want to add another tag to the Internet Gateway ?
n
PS D:\GBG\workspace>
```

Ln 176, Col 2 (815 selected) Spaces: 4 UTF-8 CRLF PowerShell

# Fifth: Internet Gateway With Tags After Creation



The screenshot shows the Visual Studio Code interface with a PowerShell script in the editor and its execution output in the terminal.

**Explorer:** The file explorer on the left shows the workspace structure: `WORKSPACE` > `salma` > `test.ps1`.

**Editor:** The script `test.ps1` contains the following PowerShell code:

```
salma > test.ps1
175
176
177 }
178
179
180 #-----
181 #Attaching Internet Gateway to the VPC
182 #-----
183 Add-EC2InternetGateway -InternetGatewayId $IGWID -VpcId $vpcID
184 Get-EC2InternetGateway
185 #-----
186
187
188
```

**Terminal:** The terminal shows the execution of the script. The output includes the execution of the script, the output of the `Add-EC2InternetGateway` and `Get-EC2InternetGateway` commands, and a table of attachments.

```
n
PS D:\GBG\workspace> $vpcID
vpc-0e2bc2e5594d2a579
PS D:\GBG\workspace> #-----
PS D:\GBG\workspace> #Attaching Internet Gateway to the VPC
PS D:\GBG\workspace> #-----
PS D:\GBG\workspace> Add-EC2InternetGateway -InternetGatewayId $IGWID -VpcId $vpcID
PS D:\GBG\workspace> Get-EC2InternetGateway

Attachments      InternetGatewayId  OwnerId  Tags
-----
{vpc-07e368fb8901161b4} igw-0c347142903731a40 821594020462 {}
{vpc-0e2bc2e5594d2a579} igw-0d1b681dffb91fe07a 821594020462 {vpc igw, date}
```

**Terminal Output:**

```
PS D:\GBG\workspace> #-----
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
```

**Status Bar:** The status bar at the bottom indicates the current line and column: `Ln 187, Col 1`, `Spaces: 4`, `UTF-8`, `CRLF`, and `{ PowerShell }`.

# Part

# 03

## Creating Subnets with Tags

- With Number of Subnets and CIDR of each Subnet entered by the User
- Also Tags are Entered by the User

# First: Get Number of subnets from the user and Two validation

1. That user Enters a Number

2. That this number is not 0




The screenshot shows a Visual Studio Code editor window with a PowerShell script named `test.ps1` open. The script is being edited in a terminal window. The script's content is as follows:

```
salma > test.ps1
120
121 write-host "Now let's Create subnets"
122 #-----
123 #creating public subnets with TAGS
124 #-----
125 #To ensure that user enters a number not a character
126 #-----
127 [int]$NOofpublicsubnets= read-host "How many public subnets do you want to create ?"
128 #-----
129 #To ensure that user enters a number not 0
130 #-----
131 if ($NOofpublicsubnets -lt 1)
132 {
133     do
134     {
135         [int]$NOofpublicsubnets= read-host "Please write a valid number "
136     }
137     while
138     (
139         $NOofpublicsubnets -lt 1
140     )
141 }
```

The script includes comments for validation: ensuring the user enters a number (not a character) and ensuring the number is not less than 1 (which covers the requirement of not being 0). A `do-while` loop is used for the validation.

## Second: During Getting Number of Subnets From User and Testing the Validation

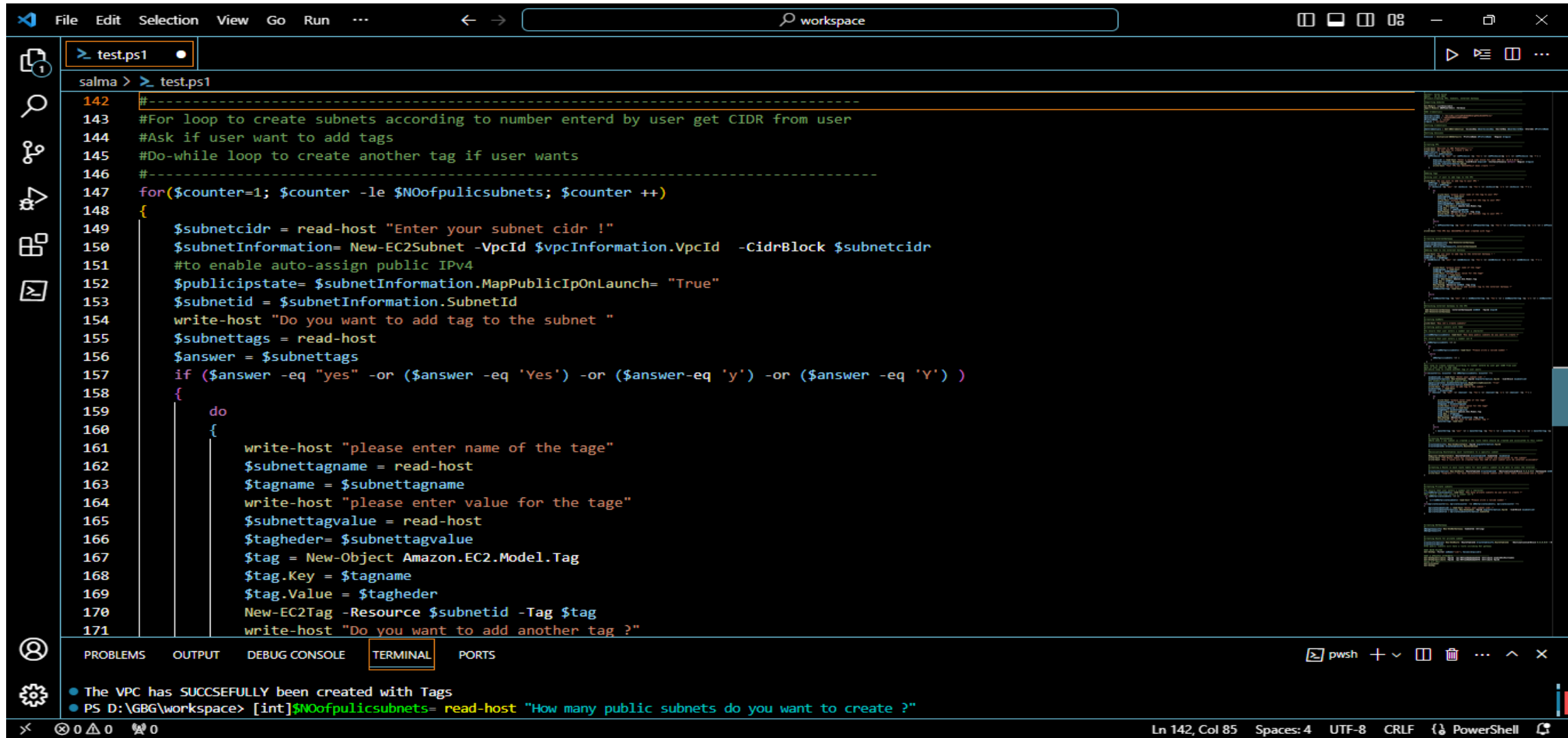


The screenshot shows a Visual Studio Code editor with a terminal window open. The terminal is running a PowerShell script named `test.ps1`. The script prompts the user to enter the number of public subnets they want to create. It then validates the input, ensuring it is a non-negative integer. If the input is invalid, it prompts the user to write a valid number. The script uses `read-host` for input and `if` and `while` loops for validation.

```
salma > test.ps1
120
121 write-host "Now let's Create subnets"
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
• The VPC has SUCCESSFULLY been created with Tags
• PS D:\GBG\workspace> [int]$NOofpublicsubnets= read-host "How many public subnets do you want to create ?"
How many public subnets do you want to create ?: 0
PS D:\GBG\workspace> if ($NOofpublicsubnets -lt 1)
>> {
>> do
>> {
>> [int]$NOofpublicsubnets= read-host "Please write a valied number "
>> }
>> while
>> (
>> $NOofpublicsubnets -lt 1
>> )
>> }
Please write a valied number : 0
Please write a valied number : 0
Please write a valied number : 00
Please write a valied number : 3
• PS D:\GBG\workspace>
```

The status bar at the bottom indicates the current line is 142, column 85, with 4 spaces, UTF-8 encoding, CRLF line endings, and PowerShell as the shell.

# Third: Creating Subnets according to the number Entered by user and creating tags Entered by user and asking user if want to add more tags



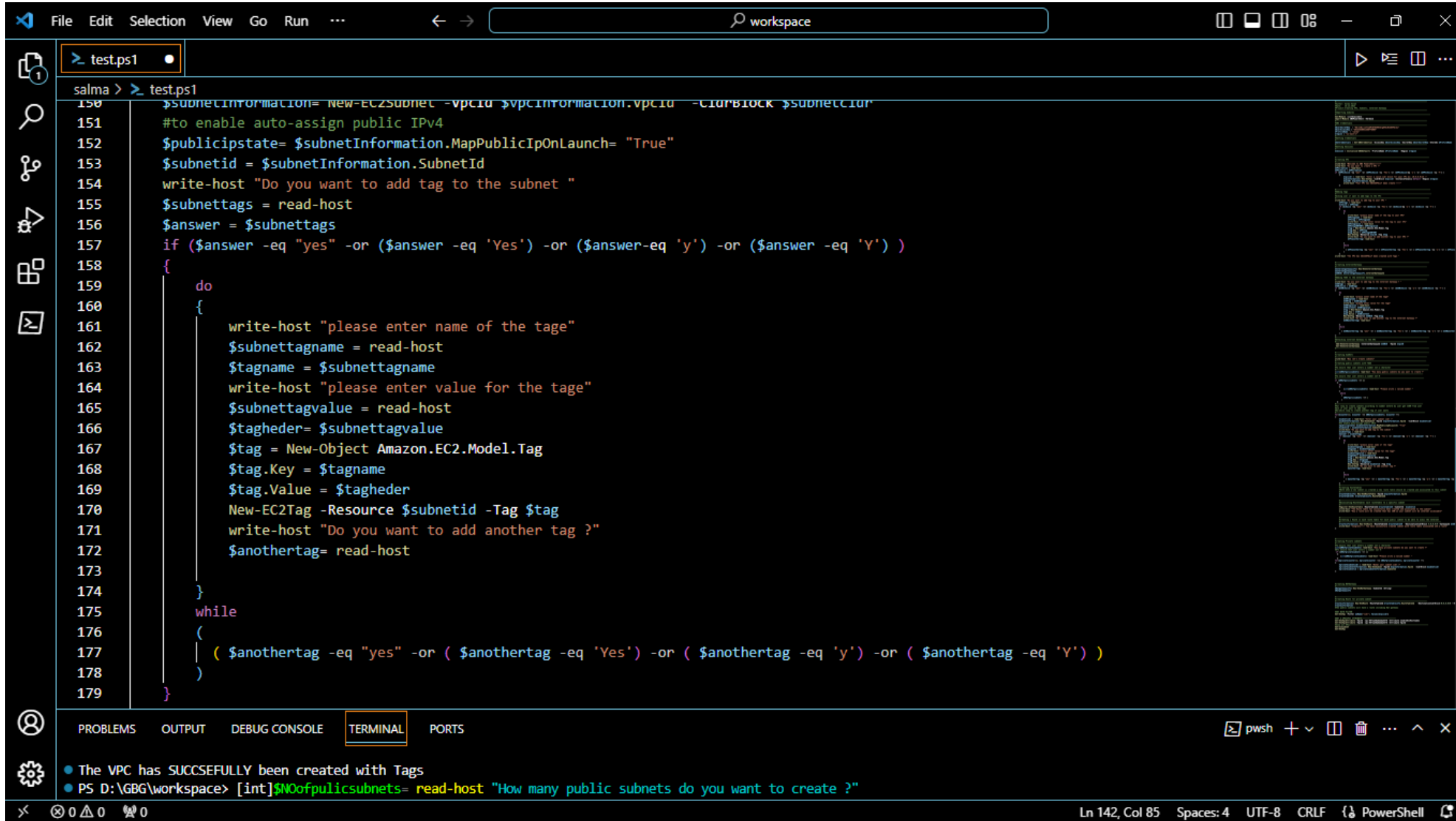
```
File Edit Selection View Go Run ... workspace
> test.ps1
salma > > test.ps1
142 #-----
143 #For loop to create subnets according to number entered by user get CIDR from user
144 #Ask if user want to add tags
145 #Do-while loop to create another tag if user wants
146 #-----
147 for($counter=1; $counter -le $NOofpulicsubnets; $counter ++){
148 {
149     $subnetcidr = read-host "Enter your subnet cidr !"
150     $subnetInformation= New-EC2Subnet -VpcId $vpcInformation.VpcId -CidrBlock $subnetcidr
151     #to enable auto-assign public IPv4
152     $publicipstate= $subnetInformation.MapPublicIpOnLaunch= "True"
153     $subnetid = $subnetInformation.SubnetId
154     write-host "Do you want to add tag to the subnet "
155     $subnettags = read-host
156     $answer = $subnettags
157     if ($answer -eq "yes" -or ($answer -eq 'Yes') -or ($answer -eq 'y') -or ($answer -eq 'Y')) {
158     {
159         do
160         {
161             write-host "please enter name of the tage"
162             $subnettagname = read-host
163             $tagname = $subnettagname
164             write-host "please enter value for the tage"
165             $subnettagvalue = read-host
166             $tagheader= $subnettagvalue
167             $tag = New-Object Amazon.EC2.Model.Tag
168             $tag.Key = $tagname
169             $tag.Value = $tagheader
170             New-EC2Tag -Resource $subnetid -Tag $tag
171             write-host "Do you want to add another tag ?"
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

- The VPC has SUCCEFULLY been created with Tags
- PS D:\GBG\workspace> [int]\$NOofpulicsubnets= read-host "How many public subnets do you want to create ?"

Ln 142, Col 85 Spaces: 4 UTF-8 CRLF PowerShell

## Third: Creating Subnets according to the number Entered by user and creating tags Entered by user and asking user if want to add more tags



```
salma > test.ps1
150 $subnetinformation= new-EC2Subnet -vpcid $vpcinformation.vpcid -cidrBlock $subnetcidr
151 #to enable auto-assign public IPv4
152 $publicipstate= $subnetInformation.MapPublicIpOnLaunch= "True"
153 $subnetid = $subnetInformation.SubnetId
154 write-host "Do you want to add tag to the subnet "
155 $subnettags = read-host
156 $answer = $subnettags
157 if ($answer -eq "yes" -or ($answer -eq 'Yes') -or ($answer -eq 'y') -or ($answer -eq 'Y') )
158 {
159     do
160     {
161         write-host "please enter name of the tage"
162         $subnettagname = read-host
163         $tagname = $subnettagname
164         write-host "please enter value for the tage"
165         $subnettagvalue = read-host
166         $tagheader= $subnettagvalue
167         $tag = New-Object Amazon.EC2.Model.Tag
168         $tag.Key = $tagname
169         $tag.Value = $tagheader
170         New-EC2Tag -Resource $subnetid -Tag $tag
171         write-host "Do you want to add another tag ?"
172         $anothertag= read-host
173     }
174     while
175     (
176         ( $anothertag -eq "yes" -or ( $anothertag -eq 'Yes') -or ( $anothertag -eq 'y') -or ( $anothertag -eq 'Y') )
177     )
178 }
179 }
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

• The VPC has SUCCSEFULLY been created with Tags  
• PS D:\GBG\workspace> [int]\$NoOfpublicsubnets= read-host "How many public subnets do you want to create ?"

Ln 142, Col 85 Spaces: 4 UTF-8 CRLF PowerShell

# Part

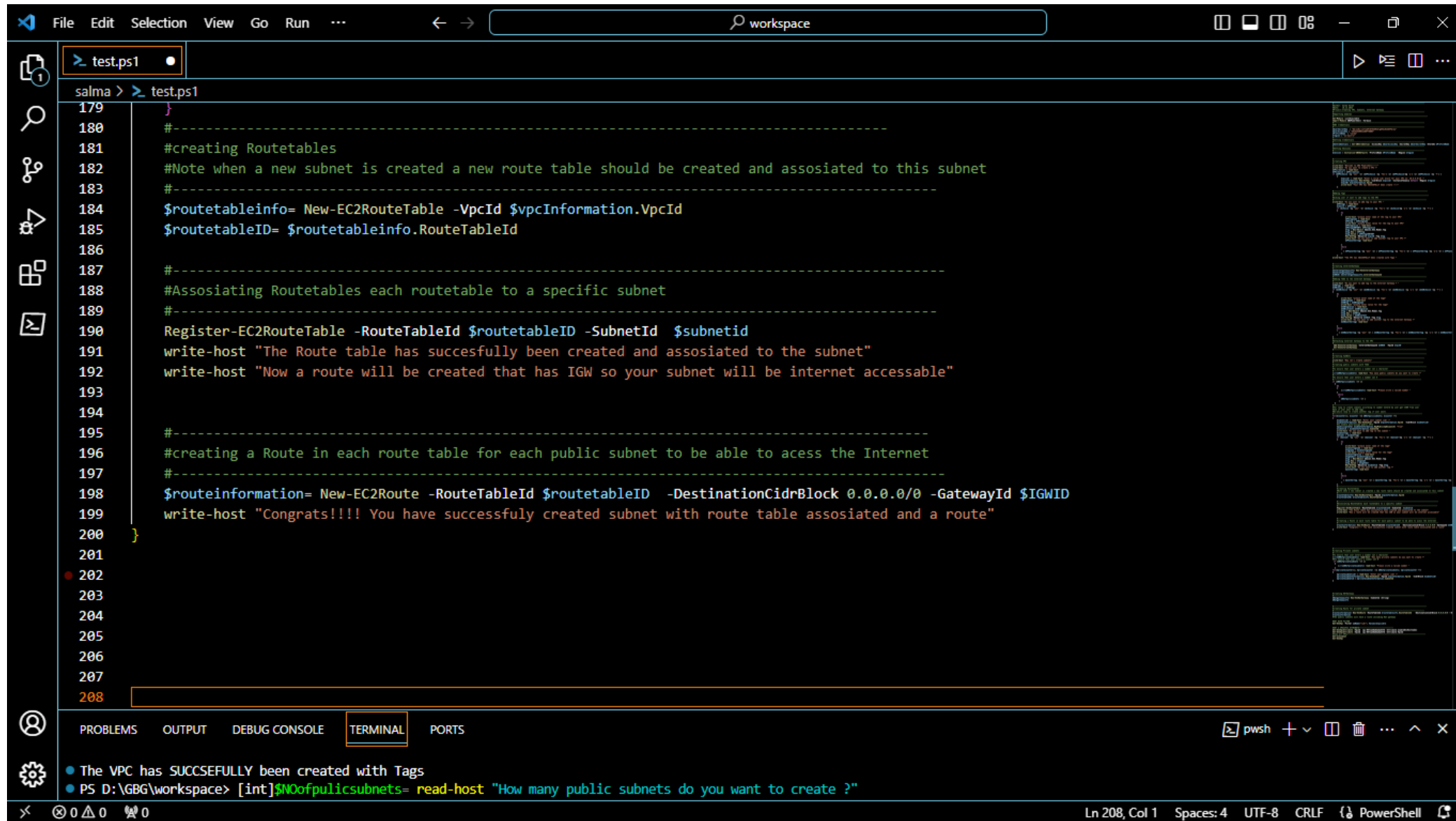
# 04

## Creating a route table and a route with each subnet created subnet

- In the same For loop that has loops equal number of subnets entered by user after the subnet is created a route table will be created and associated to this subnet
- Also a route will be created that has internet gateway to make this subnet internet accessible



# First: Creating route table associated for each subnet and a route



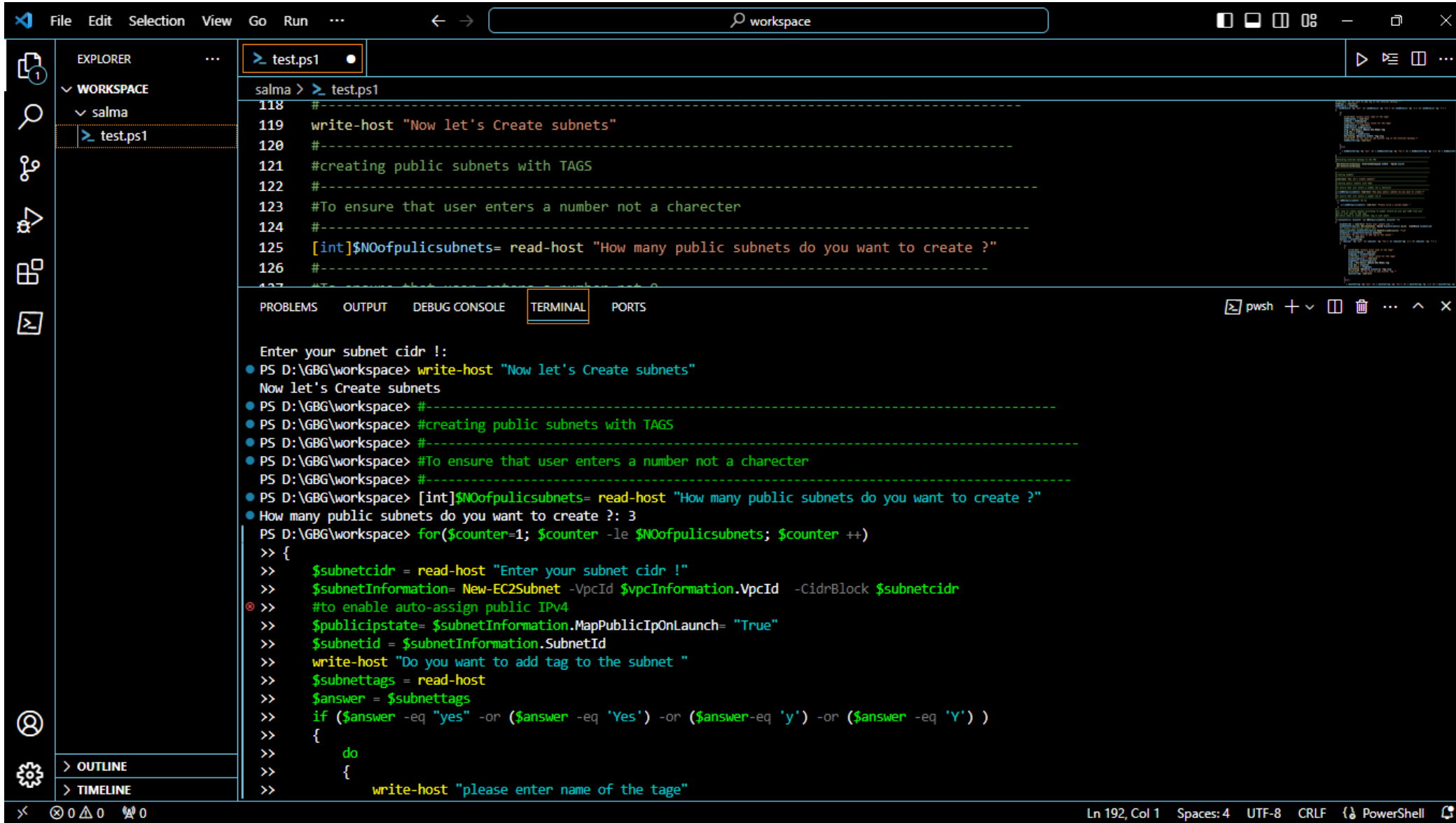
```
File Edit Selection View Go Run ... workspace
> test.ps1
salma > > test.ps1
179 }
180 #-----
181 #creating Routetables
182 #Note when a new subnet is created a new route table should be created and associated to this subnet
183 #-----
184 $routetableinfo= New-EC2RouteTable -VpcId $vpcInformation.VpcId
185 $routetableID= $routetableinfo.RouteTableId
186
187 #-----
188 #Associating Routetables each routetable to a specific subnet
189 #-----
190 Register-EC2RouteTable -RouteTableId $routetableID -SubnetId $subnetid
191 write-host "The Route table has succesfully been created and associated to the subnet"
192 write-host "Now a route will be created that has IGW so your subnet will be internet accessible"
193
194
195 #-----
196 #creating a Route in each route table for each public subnet to be able to access the Internet
197 #-----
198 $routeinformation= New-EC2Route -RouteTableId $routetableID -DestinationCidrBlock 0.0.0.0/0 -GatewayId $IGWID
199 write-host "Congrats!!!! You have successfully created subnet with route table associated and a route"
200 }
201
202
203
204
205
206
207
208
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

• The VPC has SUCCSEFULLY been created with Tags  
• PS D:\GBG\workspace> [int]\$NoOfPublicSubnets= read-host "How many public subnets do you want to create ?"

Ln 208, Col 1 Spaces: 4 UTF-8 CRLF { PowerShell

## Second: Creating Subnets route tables and routes (During Run)



The screenshot displays the Visual Studio Code interface with a PowerShell script in the editor and its execution in the terminal. The Explorer sidebar on the left shows the workspace structure with a file named `test.ps1`. The terminal window at the bottom shows the script's execution, including prompts for subnet creation and user input.

```
salma > test.ps1
118 #-----
119 write-host "Now let's Create subnets"
120 #-----
121 #creating public subnets with TAGS
122 #-----
123 #To ensure that user enters a number not a charecter
124 #-----
125 [int]$N0ofpulicsubnets= read-host "How many public subnets do you want to create ?"
126 #-----
127 #To ensure that user enters a number not a charecter
```

The terminal output shows the script's execution steps:

- Enter your subnet cidr !:
- PS D:\GBG\workspace> write-host "Now let's Create subnets"
- Now let's Create subnets
- PS D:\GBG\workspace> #-----
- PS D:\GBG\workspace> #creating public subnets with TAGS
- PS D:\GBG\workspace> #-----
- PS D:\GBG\workspace> #To ensure that user enters a number not a charecter
- PS D:\GBG\workspace> #-----
- PS D:\GBG\workspace> [int]\$N0ofpulicsubnets= read-host "How many public subnets do you want to create ?"
- How many public subnets do you want to create ?: 3
- PS D:\GBG\workspace> for(\$counter=1; \$counter -le \$N0ofpulicsubnets; \$counter ++)
- >> {
- >> \$subnetcidr = read-host "Enter your subnet cidr !"
- >> \$subnetInformation= New-EC2Subnet -VpcId \$vpcInformation.VpcId -CidrBlock \$subnetcidr
- >> #to enable auto-assign public IPv4
- >> \$publicipstate= \$subnetInformation.MapPublicIpOnLaunch= "True"
- >> \$subnetid = \$subnetInformation.SubnetId
- >> write-host "Do you want to add tag to the subnet "
- >> \$subnettags = read-host
- >> \$answer = \$subnettags
- >> if (\$answer -eq "yes" -or (\$answer -eq 'Yes') -or (\$answer -eq 'y') -or (\$answer -eq 'Y') )
- >> {
- >> do
- >> {
- >> write-host "please enter name of the tage"

## Second: Creating Subnets route tables and routes (During Run)

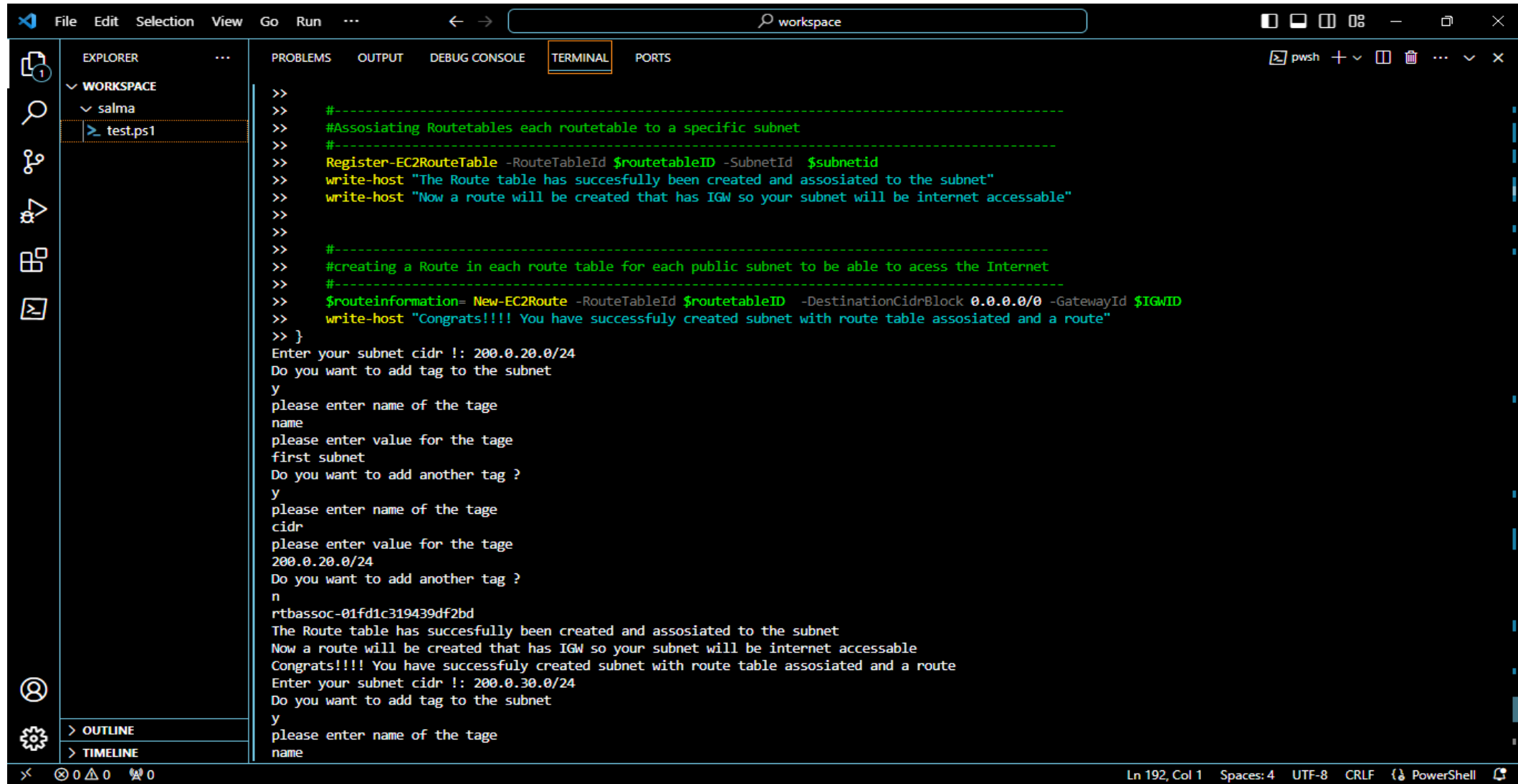


The screenshot shows a Visual Studio Code interface with a terminal window open. The terminal displays PowerShell code for creating subnets, route tables, and tags. The code is as follows:

```
PS D:\GBG\workspace> for($counter=1; $counter -le $NOofpublicsubnets; $counter ++)
>> {
>>     $subnetcidr = read-host "Enter your subnet cidr !"
>>     $subnetInformation= New-EC2Subnet -VpcId $vpcInformation.VpcId -CidrBlock $subnetcidr
>>     #to enable auto-assign public IPv4
>>     $publicipstate= $subnetInformation.MapPublicIpOnLaunch= "True"
>>     $subnetid = $subnetInformation.SubnetId
>>     write-host "Do you want to add tag to the subnet "
>>     $subnettags = read-host
>>     $answer = $subnettags
>>     if ($answer -eq "yes" -or ($answer -eq 'Yes') -or ($answer -eq 'y') -or ($answer -eq 'Y') )
>>     {
>>         do
>>         {
>>             write-host "please enter name of the tage"
>>             $subnettagname = read-host
>>             $tagname = $subnettagname
>>             write-host "please enter value for the tage"
>>             $subnettagvalue = read-host
>>             $tagheader= $subnettagvalue
>>             $tag = New-Object Amazon.EC2.Model.Tag
>>             $tag.Key = $tagname
>>             $tag.Value = $tagheader
>>             New-EC2Tag -Resource $subnetid -Tag $tag
>>             write-host "Do you want to add another tag ?"
>>             $anothertag= read-host
>>         }
>>         while
>>         (
>>             ( $anothertag -eq "yes" -or ( $anothertag -eq 'Yes') -or ( $anothertag -eq 'y') -or ( $anothertag -eq 'Y') )
>>         )
>>     }
>>     #-----
>>     #creating Routetables
>>     #Note when a new subnet is created a new route table should be created and associated to this subnet
>>     #-----
>>     $routetableinfo= New-EC2RouteTable -VpcId $vpcInformation.VpcId
>>     $routetableID= $routetableinfo.RouteTableId
```

The terminal window has a dark theme. The Explorer pane on the left shows the workspace structure with a file named `test.ps1`. The terminal output shows the execution of the PowerShell script, with prompts for subnet CIDR, tags, and route tables. The status bar at the bottom indicates the current line and column (Ln 192, Col 1), the number of spaces (Spaces: 4), the encoding (UTF-8), the line ending (CRLF), and the shell (PowerShell).

## Third: User Enters CIDR and Tags of each subnet and Route table and route created using the created subnet ID

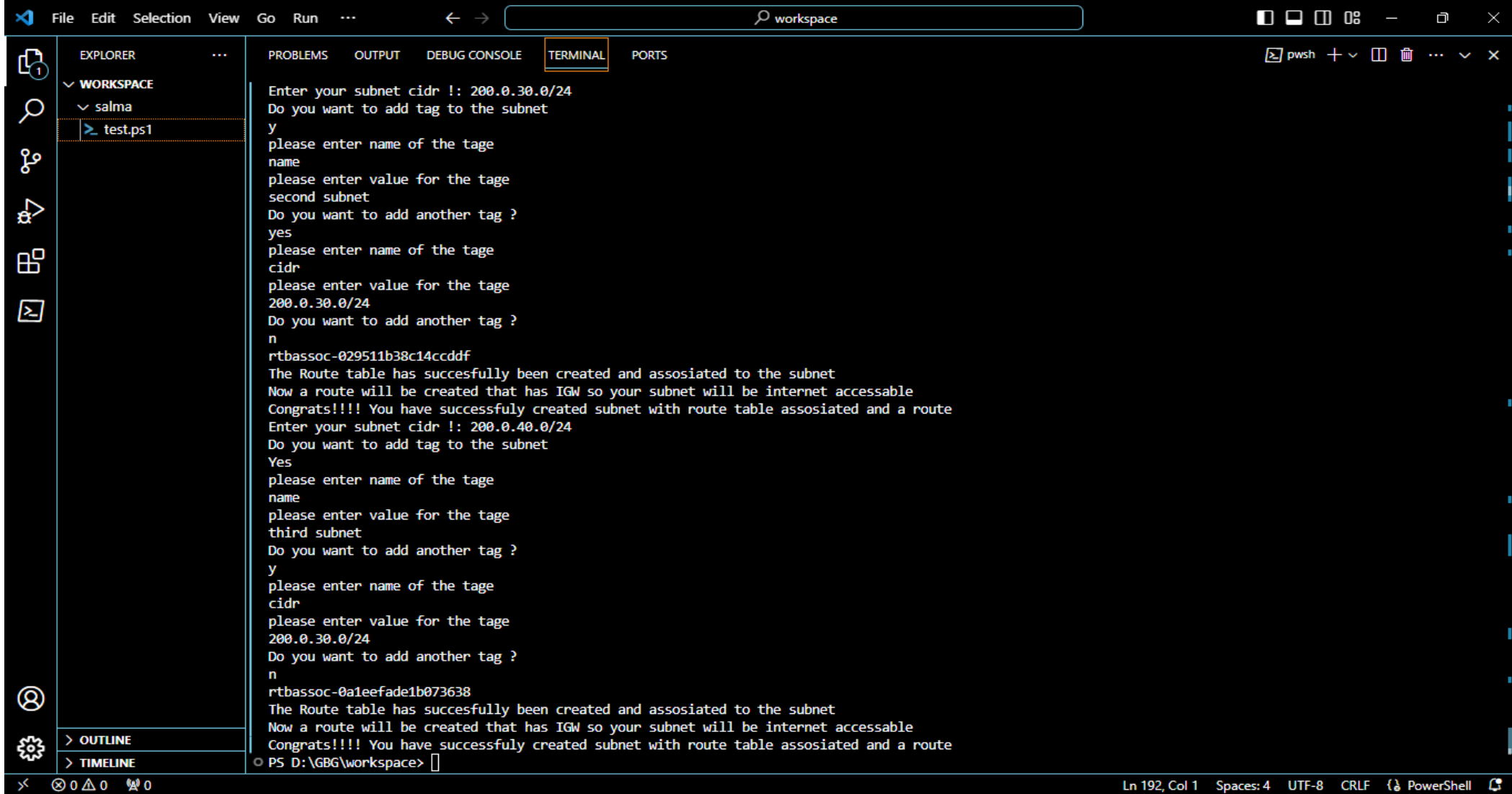


```
File Edit Selection View Go Run ... workspace
EXPLORER
WORKSPACE
  salma
    test.ps1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
pwsh + - - - x

>>
>> #-----
>> #Associating Routetables each routetable to a specific subnet
>> #-----
>> Register-EC2RouteTable -RouteTableId $routetableID -SubnetId $subnetid
>> write-host "The Route table has succesfully been created and associated to the subnet"
>> write-host "Now a route will be created that has IGW so your subnet will be internet accessable"
>>
>> #-----
>> #creating a Route in each route table for each public subnet to be able to access the Internet
>> #-----
>> $routeinformation= New-EC2Route -RouteTableId $routetableID -DestinationCidrBlock 0.0.0.0/0 -GatewayId $IGWID
>> write-host "Congrats!!!! You have successfully created subnet with route table assosiated and a route"
>> }
Enter your subnet cidr !: 200.0.20.0/24
Do you want to add tag to the subnet
y
please enter name of the tage
name
please enter value for the tage
first subnet
Do you want to add another tag ?
y
please enter name of the tage
cidr
please enter value for the tage
200.0.20.0/24
Do you want to add another tag ?
n
rtbassoc-01fd1c319439df2bd
The Route table has succesfully been created and associated to the subnet
Now a route will be created that has IGW so your subnet will be internet accessable
Congrats!!!! You have successfully created subnet with route table assosiated and a route
Enter your subnet cidr !: 200.0.30.0/24
Do you want to add tag to the subnet
y
please enter name of the tage
name
```

Ln 192, Col 1 Spaces: 4 UTF-8 CRLF PowerShell

## Third: User Enters CIDR and Tags of each subnet and Route table and route created using the created subnet ID



The screenshot displays the Visual Studio Code interface with a PowerShell terminal window open. The terminal shows a script that prompts the user to enter subnet CIDR, tags, and route table information. The script successfully creates two subnets and their associated route tables, associating them with an Internet Gateway (IGW).

```
File Edit Selection View Go Run ... workspace
EXPLORER
WORKSPACE
  salma
    test.ps1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Enter your subnet cidr !: 200.0.30.0/24
Do you want to add tag to the subnet
y
please enter name of the tage
name
please enter value for the tage
second subnet
Do you want to add another tag ?
yes
please enter name of the tage
cidr
please enter value for the tage
200.0.30.0/24
Do you want to add another tag ?
n
rtbassoc-029511b38c14ccddf
The Route table has succesfully been created and associated to the subnet
Now a route will be created that has IGW so your subnet will be internet accessable
Congrats!!!! You have successfully created subnet with route table assosiated and a route
Enter your subnet cidr !: 200.0.40.0/24
Do you want to add tag to the subnet
Yes
please enter name of the tage
name
please enter value for the tage
third subnet
Do you want to add another tag ?
y
please enter name of the tage
cidr
please enter value for the tage
200.0.30.0/24
Do you want to add another tag ?
n
rtbassoc-0a1eefade1b073638
The Route table has succesfully been created and associated to the subnet
Now a route will be created that has IGW so your subnet will be internet accessable
Congrats!!!! You have successfully created subnet with route table assosiated and a route
PS D:\GBG\workspace>
```

Ln 192, Col 1 Spaces: 4 UTF-8 CRLF PowerShell

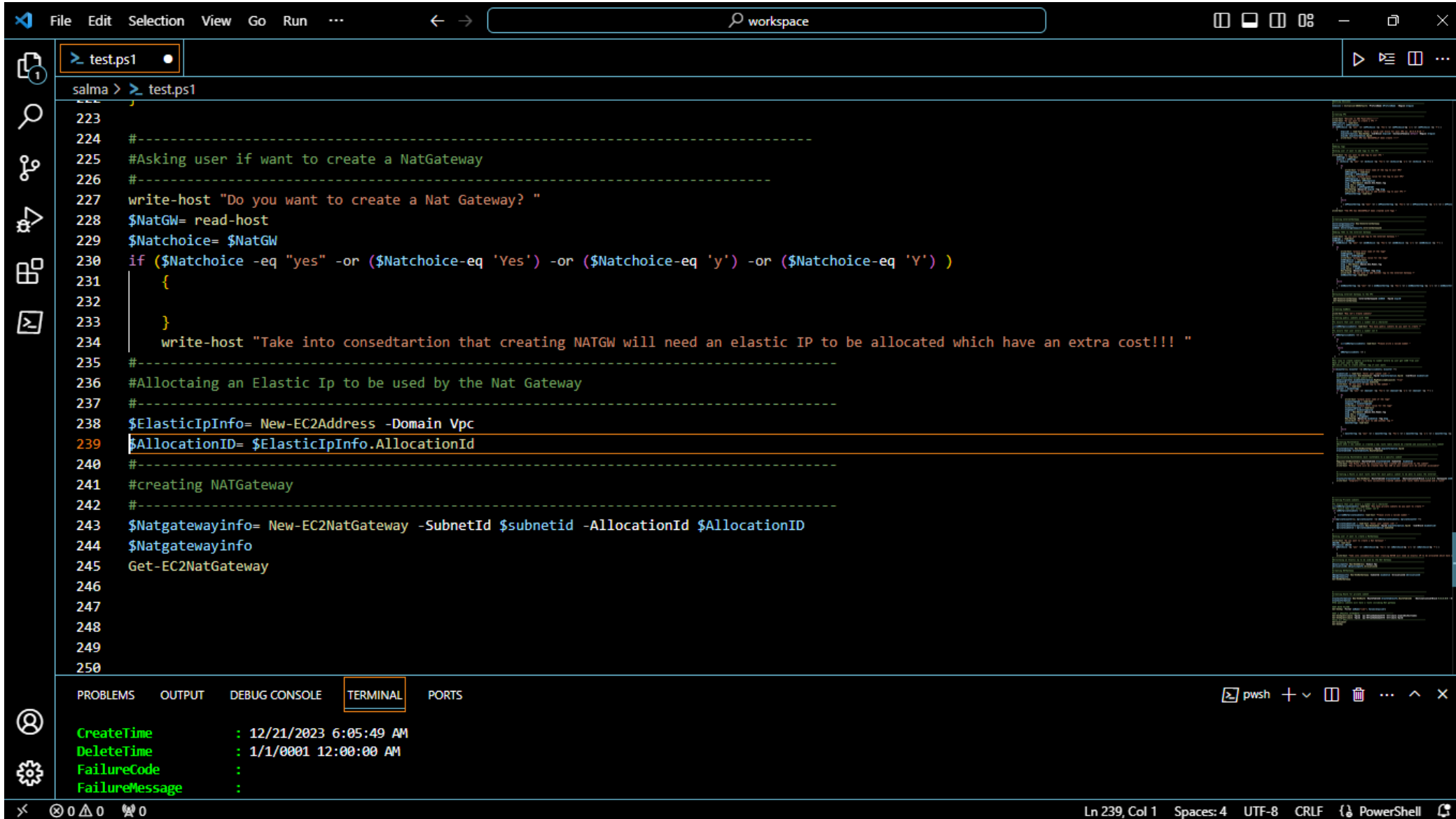
# Part

# 05

## Creating NAT Gateway in a specified public subnet

- With Alerting the user that creation of a NAT GW will need an Elastic IP allocated first which will add additional cost
- Displaying NAT GW Details

# First: alerting user then allocation of elastic IP & NAT GW creation



```
File Edit Selection View Go Run ... workspace
> test.ps1
salma > > test.ps1
223
224 #-----
225 #Asking user if want to create a NatGateway
226 #-----
227 write-host "Do you want to create a Nat Gateway? "
228 $NatGW= read-host
229 $Natchoice= $NatGW
230 if ($Natchoice -eq "yes" -or ($Natchoice -eq 'Yes') -or ($Natchoice -eq 'y') -or ($Natchoice -eq 'Y') )
231 {
232
233 }
234 write-host "Take into consedartion that creating NATGW will need an elastic IP to be allocated which have an extra cost!!! "
235 #-----
236 #Alloctaing an Elastic Ip to be used by the Nat Gateway
237 #-----
238 $ElasticIpInfo= New-EC2Address -Domain Vpc
239 $AllocationID= $ElasticIpInfo.AllocationId
240 #-----
241 #creating NATGateway
242 #-----
243 $Natgatewayinfo= New-EC2NatGateway -SubnetId $subnetid -AllocationId $AllocationID
244 $Natgatewayinfo
245 Get-EC2NatGateway
246
247
248
249
250
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

```
CreateTime : 12/21/2023 6:05:49 AM
DeleteTime : 1/1/0001 12:00:00 AM
FailureCode :
FailureMessage :
```

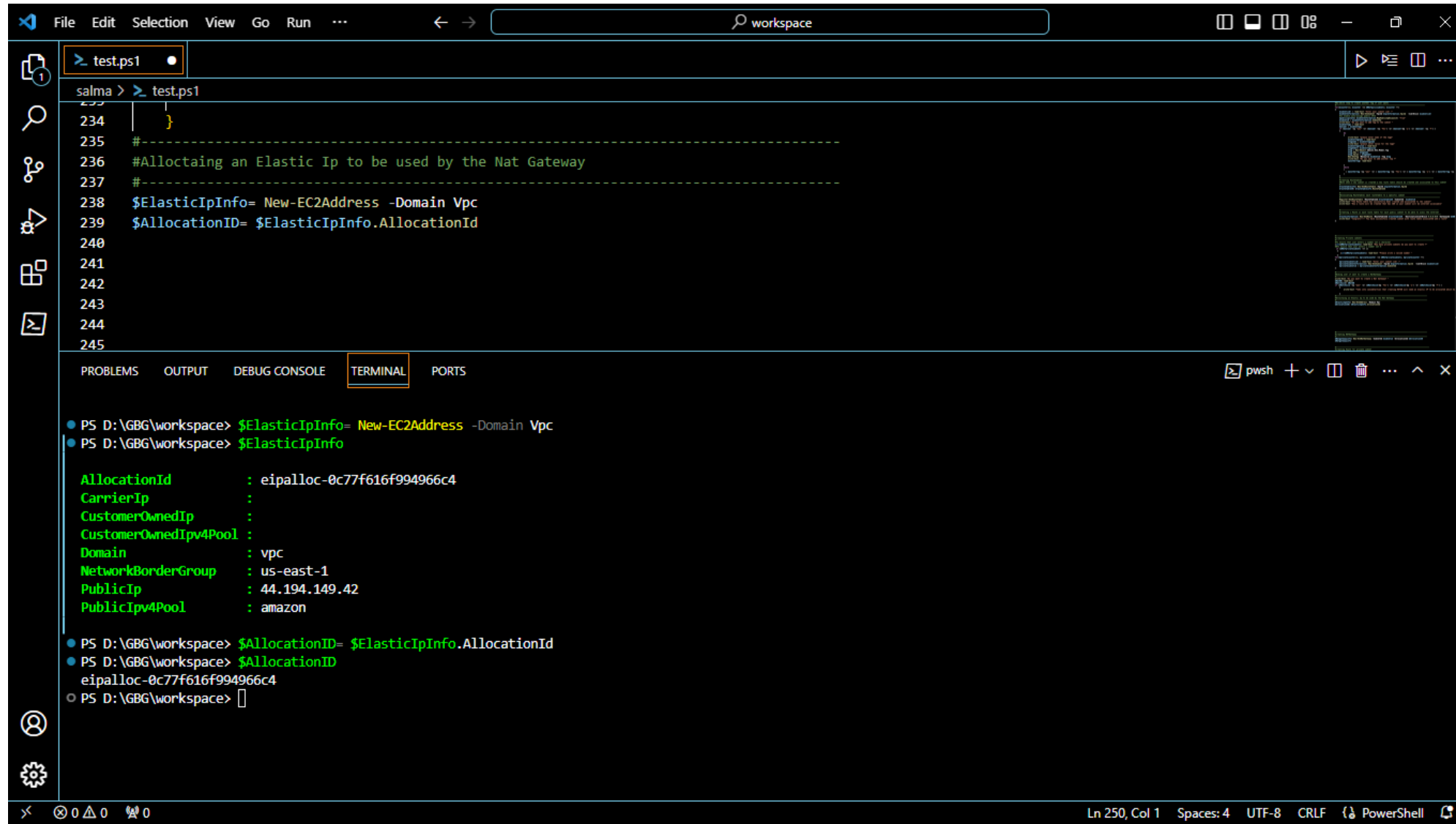
Ln 239, Col 1 Spaces: 4 UTF-8 CRLF PowerShell

## Second: Asking user if want to create a NAT GW

[illegible]



# Third: Elastic IP after allocation



The screenshot shows the Visual Studio Code interface with a PowerShell script named `test.ps1` open in the editor. The script is being executed in a terminal window, and the output shows the successful allocation of an Elastic IP.

```
salma > test.ps1
234 }
235 #-----
236 #Allocating an Elastic Ip to be used by the Nat Gateway
237 #-----
238 $ElasticIpInfo= New-EC2Address -Domain Vpc
239 $AllocationID= $ElasticIpInfo.AllocationId
240
241
242
243
244
245
```

The terminal output shows the execution of the script:

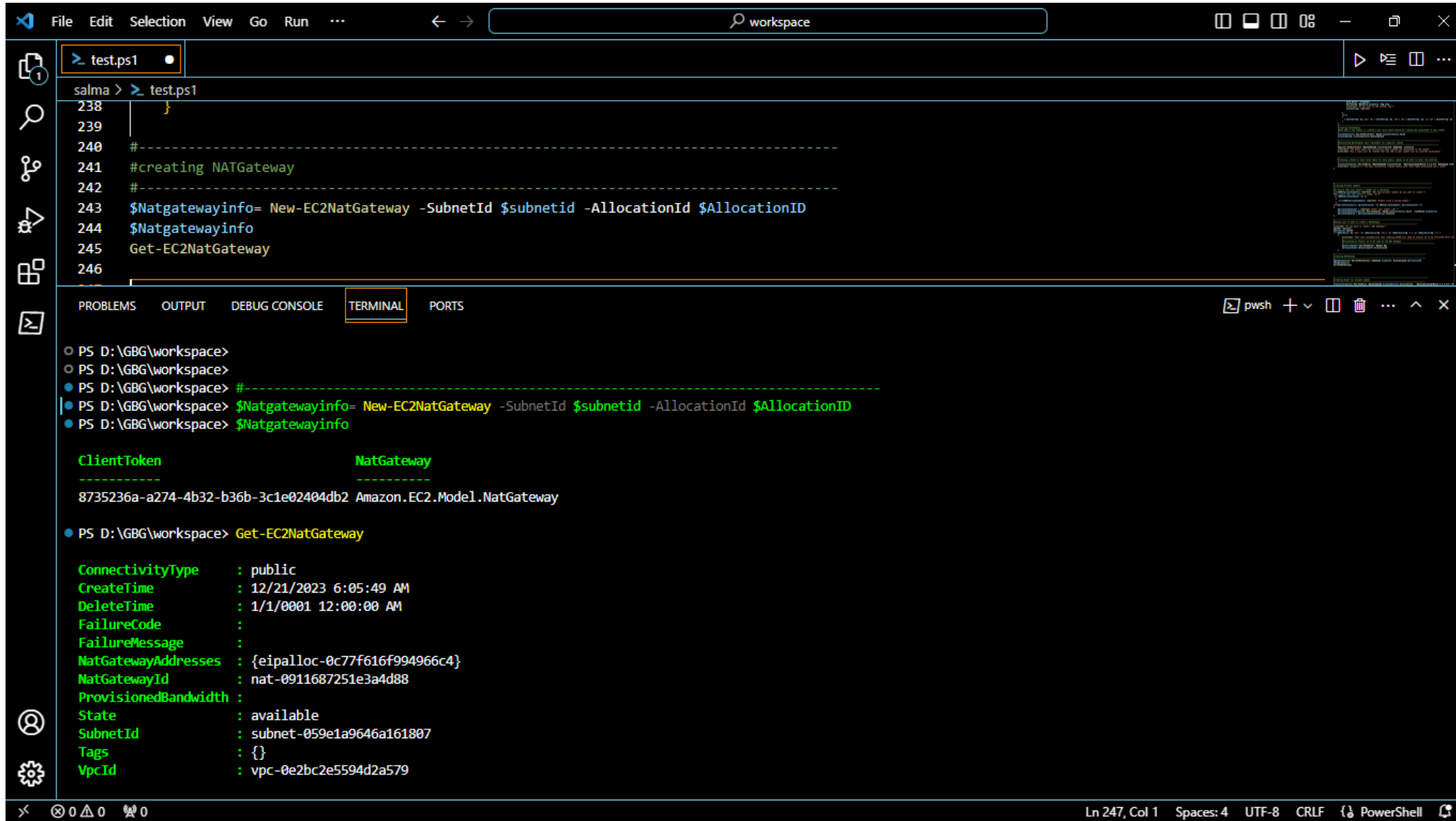
```
PS D:\GBG\workspace> $ElasticIpInfo= New-EC2Address -Domain Vpc
PS D:\GBG\workspace> $ElasticIpInfo

AllocationId      : eipalloc-0c77f616f994966c4
CarrierIp         :
CustomerOwnedIp   :
CustomerOwnedIpv4Pool :
Domain            : vpc
NetworkBorderGroup : us-east-1
PublicIp          : 44.194.149.42
PublicIpv4Pool    : amazon

PS D:\GBG\workspace> $AllocationID= $ElasticIpInfo.AllocationId
PS D:\GBG\workspace> $AllocationID
eipalloc-0c77f616f994966c4
PS D:\GBG\workspace>
```

The status bar at the bottom indicates the file is at line 250, column 1, with 4 spaces, using UTF-8 encoding, CRLF line endings, and the PowerShell shell.

## Fourth: NAT GW after creation



The screenshot shows the Visual Studio Code interface with a PowerShell script named `test.ps1` being executed. The script is run from a terminal window, and the output is displayed in the `TERMINAL` pane. The script creates a new NAT Gateway and then retrieves its details.

```
salma > test.ps1
238 }
239
240 #-----
241 #creating NATGateway
242 #-----
243 $Natgatewayinfo= New-EC2NatGateway -SubnetId $subnetid -AllocationId $AllocationID
244 $Natgatewayinfo
245 Get-EC2NatGateway
246
```

The terminal output shows the execution of the script and the resulting NAT Gateway information:

```
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace> #-----
PS D:\GBG\workspace> $Natgatewayinfo= New-EC2NatGateway -SubnetId $subnetid -AllocationId $AllocationID
PS D:\GBG\workspace> $Natgatewayinfo

ClientToken                               NatGateway
-----
8735236a-a274-4b32-b36b-3c1e02404db2 Amazon.EC2.Model.NatGateway

PS D:\GBG\workspace> Get-EC2NatGateway

ConnectivityType      : public
CreateTime             : 12/21/2023 6:05:49 AM
DeleteTime             : 1/1/0001 12:00:00 AM
FailureCode            :
FailureMessage         :
NatGatewayAddresses    : {eipalloc-0c77f616f994966c4}
NatGatewayId           : nat-0911687251e3a4d88
ProvisionedBandwidth   :
State                  : available
SubnetId               : subnet-059e1a9646a161807
Tags                   : {}
VpcId                  : vpc-0e2bc2e5594d2a579
```

The status bar at the bottom indicates the current position is Line 247, Column 1, with 4 spaces, UTF-8 encoding, CRLF line endings, and the PowerShell shell.

# Finally: The Completed architecture

aws

Services

Search [Alt+S]

N. Virginia

Salma\_Salah @ salma-salah

VPC dashboard

EC2 Global View

Filter by VPC:

Select a VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

Endpoints

Endpoint services

NAT gateways

No200.0.0.0/16-

Network Address Usage metrics Disabled

Route 53 Resolver DNS Firewall rule groups-

Owner ID821594020462

Resource map New

CIDRs

Flow logs

Tags

Integrations

Resource map Info

VPC Show details

Your AWS virtual network

vpc-0e2bc2e5594d2a579

Subnets (3)

Subnets within this VPC

us-east-1a

subnet-059e1a9646a161807

subnet-0d60566c2a5b0020e

subnet-09063ea8d640b9bd8

Route tables (4)

Route network traffic to resources

rtb-01eb90e8a266066b8

rtb-05308b436429e4e87

rtb-042c3466438bb1769

rtb-096dda5efcc7d2d42

CloudShell

Feedback

© 2023, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences

# Finally: The Completed architecture

aws

Services

Search

[Alt+S]

N. Virginia

Salma\_Salah @ salma-salah

VPC dashboard

EC2 Global View

Filter by VPC:

Select a VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

Endpoints

Endpoint services

NAT gateways

No

Network Address Usage metrics Disabled

200.0.0.0/16

Route 53 Resolver DNS Firewall rule groups

-

Owner ID 821594020462

group)

-

Resource map New

CIDRs

Flow logs

Tags

Integrations

Resource map Info

Subnets (3)

Subnets within this VPC

us-east-1a

subnet-059e1a9646a161807

subnet-0d60566c2a5b0020e

subnet-09063ea8d640b9bd8

Route tables (4)

Route network traffic to resources

rtb-01eb90e8a266066b8

rtb-05308b436429e4e87

rtb-042c3466438bb1769

rtb-096dda5efcc7d2d42

Network connections (2)

Connections to other networks

igw-0d1b681dff91fe07a

nat-0911687251e3a4d88

CloudShell

Feedback

© 2023, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences

# VPC with Tags

aws

Services

Search [Alt+S]

N. Virginia

Salma\_Salah @ salma-salah

VPC dashboard

EC2 Global View

Filter by VPC:  
Select a VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

Endpoints

Endpoint services

NAT gateways

vpc-0e2bc2e5594d2a579

Available

Disabled

Enabled

Tenancy  
Default

Default VPC  
No

Network Address Usage metrics  
Disabled

DHCP option set  
dopt-07a57c47179ae8793

IPv4 CIDR  
200.0.0.0/16

Route 53 Resolver DNS Firewall  
rule groups  
-

Main route table  
rtb-01eb90e8a266066b8

IPv6 pool  
-

Owner ID  
821594020462

Main network ACL  
acl-0b88e57abb177a285

IPv6 CIDR (Network border  
group)  
-

Resource map New

CIDRs

Flow logs

Tags

Integrations

Tags

Manage tags

Search tags

Key	Value
created by	Salma Salah
vpc power...	first ps vpc
date	20.12.3023
cidr	200.0.0.0/16

< 1 > ⚙

CloudShell

Feedback

© 2023, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences

# Internet Gateway With Tags After Creation

aws

Services

Q Search

[Alt+S]

N. Virginia ▼

Salma\_Salah @ salma-salah ▼

VPC dashboard

EC2 Global View

Filter by VPC:

Select a VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

Endpoints

Endpoint services

NAT gateways

VPC > Internet gateways > igw-0d1b681dff91fe07a

igw-0d1b681dff91fe07a

Actions

Details Info

Internet gateway ID	State	VPC ID	Owner
igw-0d1b681dff91fe07a	Attached	vpc-0e2bc2e5594d2a579	821594020462

Tags

Manage tags

Search tags

Key	Value
vpc igw	this is by powershell
date	20.12.2023

CloudShell

Feedback

© 2023, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences

# First subnet with Tags

aws

Services

Search [Alt+S]

N. Virginia

Salma\_Salah @ salma-salah

VPC dashboard

EC2 Global View

Filter by VPC:  
Select a VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

Endpoints

Endpoint services

NAT gateways

Default subnet  
No

Customer-owned IPv4 pool  
-

IPv6-only  
No

DNS64  
Disabled

vpc-0e2bc2e5594d2a579

Auto-assign public IPv4 address  
No

Outpost ID  
-

Hostname type  
IP name

Owner  
821594020462

Auto-assign IPv6 address  
No

IPv4 CIDR reservations  
-

Resource name DNS A record  
Disabled

Auto-assign customer-owned IPv4 address  
No

IPv6 CIDR reservations  
-

Resource name DNS AAAA record  
Disabled

Flow logs

Route table

Network ACL

CIDR reservations

Sharing

Tags

Tags

Manage tags

Search tags

Key	Value
cidr	200.0.20.0/24
name	first subnet

CloudShell

Feedback

© 2023, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences

# Second subnet with Tags

aws

Services

Search [Alt+S]

N. Virginia

Salma\_Salah @ salma-salah

VPC dashboard

EC2 Global View

Filter by VPC:

Select a VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

Endpoints

Endpoint services

NAT gateways

Default subnet

No

Customer-owned IPv4 pool

-

IPv6-only

No

DNS64

Disabled

vpc-0e2bc2e5594d2a579

Auto-assign public IPv4 address

No

Outpost ID

-

Hostname type

IP name

Owner

821594020462

Auto-assign IPv6 address

No

IPv4 CIDR reservations

-

Resource name DNS A record

Disabled

Auto-assign customer-owned IPv4 address

No

IPv6 CIDR reservations

-

Resource name DNS AAAA record

Disabled

Flow logs

Route table

Network ACL

CIDR reservations

Sharing

Tags

Tags

Manage tags

Search tags

Key	Value
name	second subnet
cidr	200.0.30.0/24

CloudShell

Feedback

© 2023, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences



# Third subnet with Tags

aws

Services

Search [Alt+S]

N. Virginia

Salma\_Salah @ salma-salah

VPC dashboard

EC2 Global View

Filter by VPC:  
Select a VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

Endpoints

Endpoint services

NAT gateways

Default subnet  
No

Customer-owned IPv4 pool  
-

IPv6-only  
No

DNS64  
Disabled

vpc-0e2bc2e5594d2a579

Auto-assign public IPv4 address  
No

Outpost ID  
-

Hostname type  
IP name

Owner  
821594020462

Auto-assign IPv6 address  
No

IPv4 CIDR reservations  
-

Resource name DNS A record  
Disabled

Auto-assign customer-owned IPv4 address  
No

IPv6 CIDR reservations  
-

Resource name DNS AAAA record  
Disabled

Flow logs

Route table

Network ACL

CIDR reservations

Sharing

Tags

Tags

Manage tags

Search tags

Key	Value
cidr	200.0.30.0/24
name	third subnet

CloudShell

Feedback

© 2023, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences

# The Three Route Tables Associated to three subnets

aws

Services

Search

[Alt+S]

N. Virginia

Salma\_Salah @ salma-salah

VPC dashboard

EC2 Global View

Filter by VPC:  
Select a VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

Endpoints

Endpoint services

NAT gateways

Route tables (6)

Find resources by attribute or tag

Create route table

	Name	Route table ID	Explicit subnet associations	Edge as...	Main	VPC	Own...
<input type="checkbox"/>	-	rtb-01eb90e8a266066b8	-	-	Yes	vpc-0e2bc2e5594d2...	821594
<input type="checkbox"/>	-	rtb-05308b436429e4e87	subnet-09063ea8d640b9bd8	-	No	vpc-0e2bc2e5594d2...	821594
<input type="checkbox"/>	-	rtb-042c3466438bb1769	subnet-0d60566c2a5b0020e	-	No	vpc-0e2bc2e5594d2...	821594
<input type="checkbox"/>	-	rtb-096dda5efcc7d2d42	subnet-059e1a9646a161807	-	No	vpc-0e2bc2e5594d2...	821594
<input type="checkbox"/>	-	rtb-043f9de6daad585c1	-	-	Yes	vpc-07e368fb89011...	821594
<input type="checkbox"/>	-	rtb-0b0e0d8c50b166bfa	-	-	Yes	vpc-0985d81d5b04c...	821594

CloudShell

Feedback

© 2023, Amazon Web Services, Inc. or its affiliates.

Privacy

Terms

Cookie preferences

# Elastic IP after allocation

The screenshot displays the AWS Management Console interface for Elastic IP addresses. The top navigation bar includes the AWS logo, a search bar, and user information for 'Salma\_Salah' in the 'N. Virginia' region. The left-hand navigation pane lists various services, with 'Virtual private cloud' expanded to show options like 'Your VPCs', 'Subnets', and 'Elastic IPs'. The main content area, titled 'Elastic IP addresses (1)', features a search filter and a table with one entry. The table columns are Name, Allocated IPv4 address, Type, Allocation ID, and Reverse DNS. The single entry shows an allocated IPv4 address of 44.194.149.42, which is a Public IP, with an Allocation ID of eipalloc-0c77f616f994966c4. An orange button labeled 'Allocate Elastic IP address' is visible in the top right of the main area. At the bottom, a notification bar suggests viewing IP address usage and recommendations with 'Public IP insights'.

aws Services Search [Alt+S]

VPC dashboard X

EC2 Global View

Filter by VPC:

Select a VPC

Virtual private cloud

- Your VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IPs**
- Managed prefix lists
- Endpoints
- Endpoint services
- NAT gateways

Elastic IP addresses (1)

Filter Elastic IP addresses

	Name	Allocated IPv4 add...	Type	Allocation ID	Reverse DN
<input type="checkbox"/>	-	44.194.149.42	Public IP	eipalloc-0c77f616f994966c4	-

Allocate Elastic IP address

View IP address usage and recommendations to release unused IPs with [Public IP insights](#).

CloudShell Feedback

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

# NAT GW after creation

The screenshot displays the AWS Management Console interface. At the top, the navigation bar includes the AWS logo, a 'Services' menu, a search bar, and utility icons. The left-hand navigation pane lists various VPC services, with 'NAT gateways' currently selected. The main content area shows the breadcrumb path 'VPC > NAT gateways > nat-0911687251e3a4d88' and the title 'nat-0911687251e3a4d88'. An 'Actions' dropdown menu is located in the top right of the content area. Below the title, the 'Details' tab is active, displaying a table of gateway information. The table includes fields for the NAT gateway ID, ARN, connectivity type, public and private IPv4 addresses, creation time, and state. The gateway is in an 'Available' state. At the bottom of the details section, there are tabs for 'Secondary IPv4 addresses', 'Monitoring', and 'Tags'. The 'Secondary IPv4 addresses' tab is selected, showing a table with one empty row and buttons to refresh or edit associations. The footer of the console contains links to CloudShell, Feedback, and legal notices.

aws Services Search [Alt+S]

VPC dashboard X

EC2 Global View

Filter by VPC:

Select a VPC

Virtual private cloud

- Your VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- Endpoints
- Endpoint services
- NAT gateways**

VPC > NAT gateways > nat-0911687251e3a4d88

## nat-0911687251e3a4d88

Actions

### Details Info

NAT gateway ID nat-0911687251e3a4d88	Connectivity type Public	State Available	State message Info
NAT gateway ARN arn:aws:ec2:us-east-1:821594020462:natgateway/nat-0911687251e3a4d88	Primary public IPv4 address 44.194.149.42	Primary private IPv4 address 200.0.40.194	Primary network interface ID eni-084972095396ebba7
VPC vpc-0e2bc2e5594d2a579	Subnet subnet-059e1a9646a161807	Created Thursday, December 21, 2023 at 06:05:49 GMT+2	Deleted -

Secondary IPv4 addresses Monitoring Tags

### Secondary IPv4 addresses

Edit secondary IPv4 address associations

CloudShell Feedback

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

**Attached below my github link with the source code**

<https://github.com/salmasalam024/Powershell/tree/code>