IaC - DevOps з використанням Terraform

Мета: Отримати практичний досвід у DevOps та підході "Інфраструктура як код" (Infrastructure-as-Code) через автоматизоване створення та розгортання хмарної інфраструктури в **Microsoft Azure** за допомогою фреймворку **Terraform**.

Завдання:

1. Описати віртуальну інфраструктуру (Група ресурсів, Мережа, Група безпеки, VM) у коді.
2. Розгорнути віртуальну інфраструктуру в Azure.
3. Перевірити роботу веб-сервера.
4. Видалити ресурси після завершення.

Лабораторне середовище:

- Visual Studio Code з встановленим плагіном *HashiCorp Terraform*.
- Встановлений **Terraform**.
- Встановлений **Azure CLI**.

Крок 1: Підготовка проекту

1. Створіть на своєму комп'ютері папку для лабораторної роботи, наприклад `Terraform-Azure-Lab`.
2. Всередині цієї папки створіть підпапку `files`.
3. Створіть 4 текстові файли з розширеннями `.tf` та `.tpl` і вставте в них наведений нижче код.

1.1. Файл `provider.tf`

Цей файл вказує Terraform, що ми будемо працювати з хмарою Azure.

```
terraform {
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "=3.0.0"
    }
  }
```

```
}

provider "azurerm" {
  features {}
}
```

1.2. Файл `vars-common.tf` (Змінні)

Тут ми визначаємо змінні, щоб легко змінювати налаштування в одному місці.

```
variable "resource_group_name" {
  default = "Terraform-Lab-RG"
}

variable "location" {
  default = "West Europe"
}

variable "vm_name" {
  default = "my-linux-vm"
}
```

1.3. Файл `files/template.tpl` (Скрипт запуску)

Створіть цей файл у папці `files`. Це bash-скрипт, який автоматично встановить веб-сервер Apache при першому запуску віртуальної машини (зверніть увагу: для Azure/Ubuntu ми використовуємо `apt`, а не `yum`).

```
#!/bin/bash
# Оновлюємо пакети
sudo apt-get update
# Встановлюємо Apache, PHP та MySQL клієнт
sudo apt-get install -y apache2 php libapache2-mod-php php-mysql

# Вмикаємо та запускаємо веб-сервер
sudo systemctl enable apache2
sudo systemctl start apache2

# Завантажуємо тестовий сайт (як в оригінальній лабораторній)
if [ ! -f /var/www/html/bootcamp-app.tar.gz ]; then
    cd /var/www/html
    sudo rm index.html
    sudo wget [http://fgcom.org.uk/wp-content/uploads/2020/08/bootcamp-
app.tar](http://fgcom.org.uk/wp-content/uploads/2020/08/bootcamp-app.tar)
```

```
    sudo tar xvf bootcamp-app.tar
    sudo chown www-data:www-data /var/www/html/rds.conf.php
fi
```

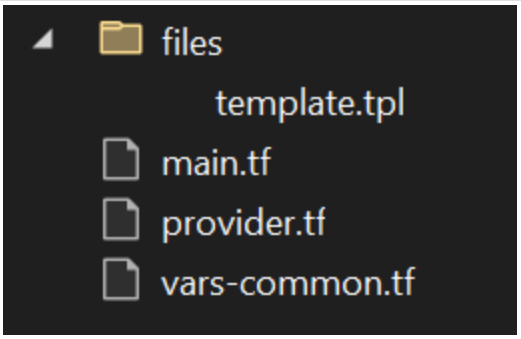
1.4. Файл `main.tf` (Головна конфігурація)

Це основний файл, де описується вся інфраструктура. В Azure вона складається з багатьох пов'язаних компонентів.

```
# 1. Створюємо групу ресурсів
resource "azurerm_resource_group" "rg" {
  name     = var.resource_group_name
  location = var.location
}

# 2. Створюємо віртуальну мережу (VNet)
resource "azurerm_virtual_network" "vnet" {
  name                = "my-vnet"
  address_space       = ["10.0.0.0/16"]
  location            = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name
}

# 3. Створюємо підмережу (Subnet)
resource "azurerm_subnet" "subnet" {
  name                 = "internal"
  resource_group_name  = azurerm_resource_group.rg.name
  virtual_network_name = azurerm_virtual_network.vnet.name
  address_prefixes     = ["10.0.1.0/24"]
}

# 4. Створюємо публічну IP-адресу (Standard SKU)
resource "azurerm_public_ip" "pip" {
  name                = "my-public-ip"
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
  allocation_method   = "Static"   # Standard SKU вимагає Static
  sku                 = "Standard" # Змінюємо Basic на Standard
}

# 5. Створюємо групу безпеки (Firewall)
resource "azurerm_network_security_group" "nsg" {
  name                = "my-nsg"
  location            = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name
```

```
  # Дозволяємо SSH (порт 22)
  security_rule {
    name                       = "SSH"
    priority                   = 1001
    direction                  = "Inbound"
    access                     = "Allow"
    protocol                   = "Tcp"
    source_port_range          = "*"
    destination_port_range     = "22"
    source_address_prefix      = "*"
    destination_address_prefix = "*"
  }

  # Дозволяємо HTTP (порт 80) для веб-сайту
  security_rule {
    name                       = "HTTP"
    priority                   = 1002
    direction                  = "Inbound"
    access                     = "Allow"
    protocol                   = "Tcp"
    source_port_range          = "*"
    destination_port_range     = "80"
    source_address_prefix      = "*"
    destination_address_prefix = "*"
  }
}

# 6. Створюємо мережевий інтерфейс (NIC)
resource "azurerm_network_interface" "nic" {
  name                = "my-nic"
  location            = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name

  ip_configuration {
    name                          = "internal"
    subnet_id                     = azurerm_subnet.subnet.id
    private_ip_address_allocation = "Dynamic"
    public_ip_address_id          = azurerm_public_ip.pip.id
  }
}

# Приєднуємо Firewall до інтерфейсу
resource "azurerm_network_interface_security_group_association" "example" {
  network_interface_id      = azurerm_network_interface.nic.id
  network_security_group_id = azurerm_network_security_group.nsg.id
```

```
}

# 7. Створюємо Віртуальну Машину (Ubuntu Linux)
resource "azurerm_linux_virtual_machine" "vm" {
  name                = var.vm_name
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
  size                = "Standard_B1s" # Економний розмір
  admin_username      = "azureuser"

  # Передаємо скрипт запуску (встановлення Apache)
  custom_data = filebase64("./files/template.tpl")

  network_interface_ids = [
    azurerm_network_interface.nic.id,
  ]

  # Вказуємо шлях до вашого SSH-ключа (створимо його на наступному кроці)
  admin_ssh_key {
    username   = "azureuser"
    public_key = file("./cloud_key.pub")
  }

  os_disk {
    caching              = "ReadWrite"
    storage_account_type = "Standard_LRS"
  }

  source_image_reference {
    publisher = "Canonical"
    offer     = "0001-com-ubuntu-server-jammy"
    sku       = "22_04-lts"
    version   = "latest"
  }
}

# Виводимо IP-адресу після створення
output "public_ip_address" {
  value = azurerm_public_ip.pip.ip_address
}
```



Місце для скріншота: Структура проєкта, після виконання цих кроків

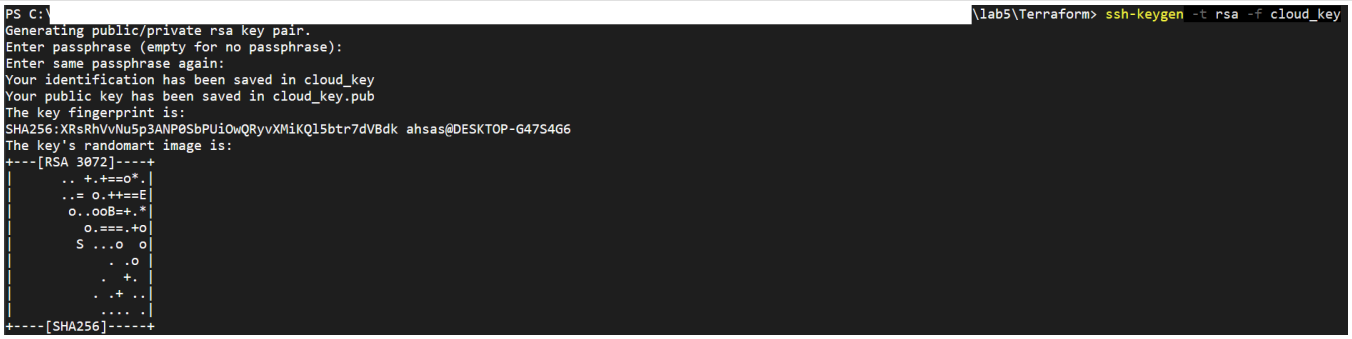
Крок 2: Генерація SSH-ключів

Azure Linux VM вимагають SSH-ключі для безпечного доступу.

1. Відкрийте термінал у VS Code (Ctrl + `).
2. Переконайтеся, що ви знаходитесь у папці проекту.
3. Виконайте команду для створення ключів (натискайте Enter на всі питання):

```
ssh-keygen -t rsa -f cloud_key
```

4. Це створить два файли у вашій папці: `cloud_key` (приватний ключ) та `cloud_key.pub` (публічний ключ, який Terraform завантажить в Azure).



Місце для скріншота: Виконання команди в терміналі

Крок 3: Встановлення та налаштування Azure CLI

Для того, щоб Terraform міг створювати ресурси у вашому акаунті Azure, на комп'ютері має бути встановлений та налаштований інструмент **Azure CLI**.

1. **Перевірка наявності:**
   Відкрийте термінал у VS Code і введіть команду:

   ```
   az --version
   ```

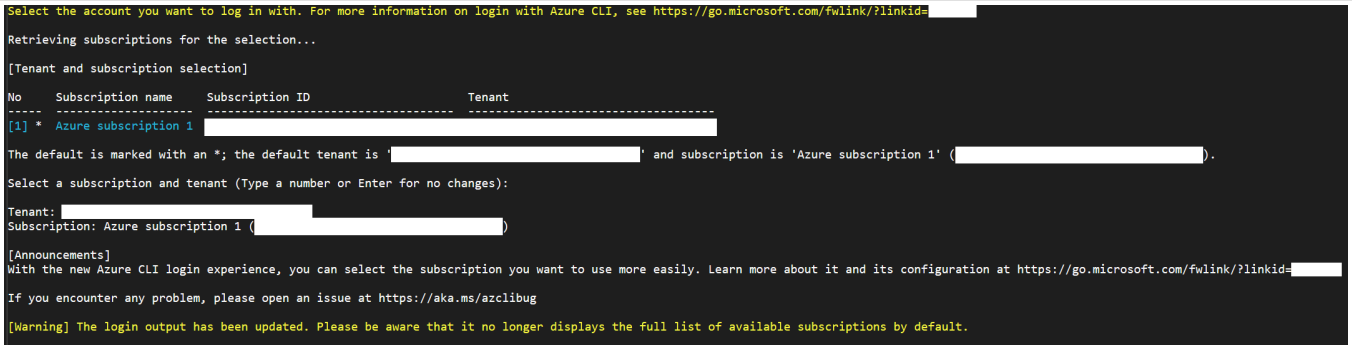
   Якщо ви бачите помилку (наприклад, *"The term 'az' is not recognized"*), перейдіть до пункту 2. Якщо виводиться номер версії — переходьте одразу до пункту 4.
2. **Встановлення:**
   Найшвидший спосіб встановити Azure CLI на Windows — виконати наступну команду в терміналі:

   ```
   winget install -e --id Microsoft.AzureCLI
   ```

   (Якщо система запитає підтвердження ліцензійної угоди, натисніть клавішу `Y` та `Enter`).

   Альтернативний варіант: Якщо `winget` не спрацював, завантажте та запустіть MSI-інсталятор з офіційного сайту Microsoft.
3. **Перезапуск середовища (Важливо!):**
   Після завершення встановлення **повністю закрийте Visual Studio Code** і відкрийте його знову. Це необхідно, щоб термінал оновив змінні середовища і "побачив" нову команду `az`.
4. **Авторизація:**
   Тепер авторизуйтесь у своєму акаунті, щоб надати доступ Terraform:

   ```
   az login
   ```
5. Відкриється браузер. Увійдіть у свій обліковий запис Microsoft Azure (використовуйте студентський акаунт, якщо є).
6. Після успішного входу закрийте браузер і поверніться до терміналу. Ви побачите список ваших підписок.



Місце для скріншота: Результат виконання команди az login у терміналі (список підписок)

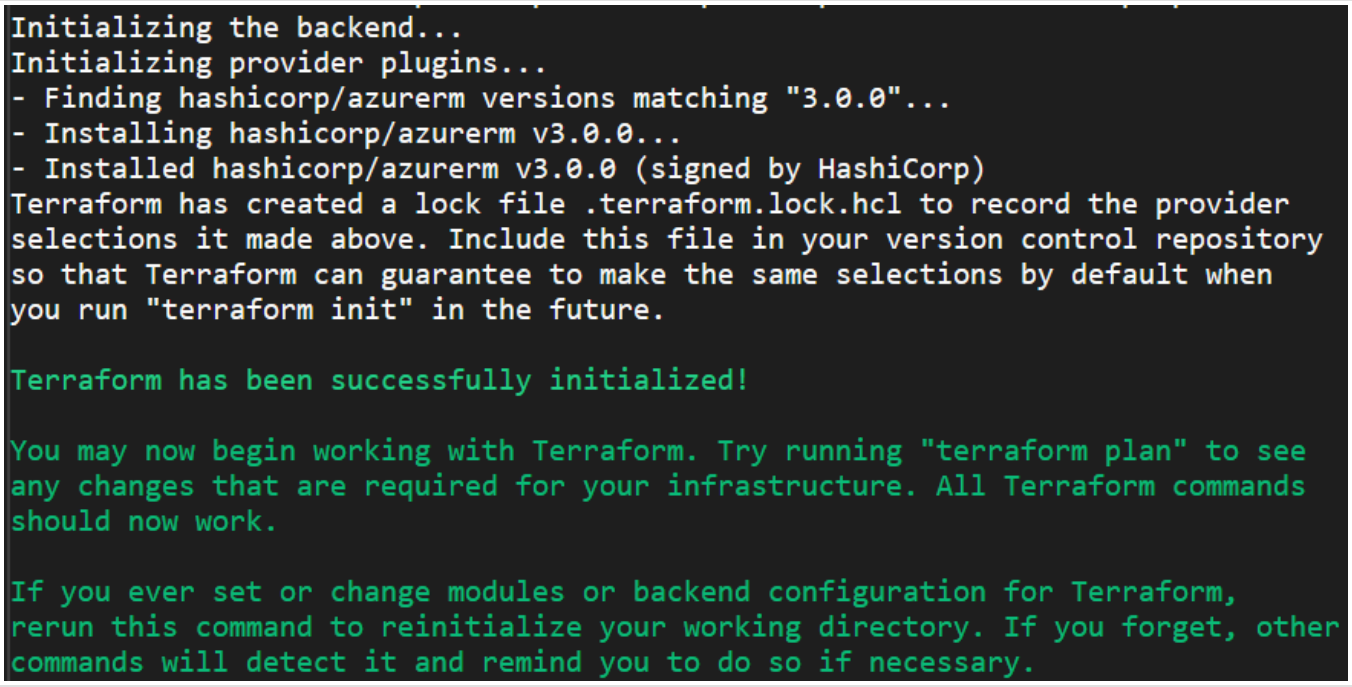
Крок 4: Розгортання інфраструктури

Встановлення Terraform (якщо не встановлено)

1. Відкрийте термінал і перевірте наявність Terraform:
   ```
   terraform -version
   ```
2. Якщо ви отримали помилку, встановіть його командою:
   ```
   winget install HashiCorp.Terraform
   ```
3. Після встановлення **перезапустіть термінал/VS Code**.

Тепер ми готові запустити Terraform.

4. **Ініціалізація:** Завантажте необхідні модулі для Azure.
   ```
   terraform init
   ```
   (Має з'явитися повідомлення "Terraform has been successfully initialized!")



Місце для скріншота: Результат виконання команди terraform init у терміналі

4. **Планування:** Перевірте, що саме Terraform збирається створити.

```
terraform plan
```

(Ви побачите список ресурсів з плюсиками `+`, що означає "буде створено").

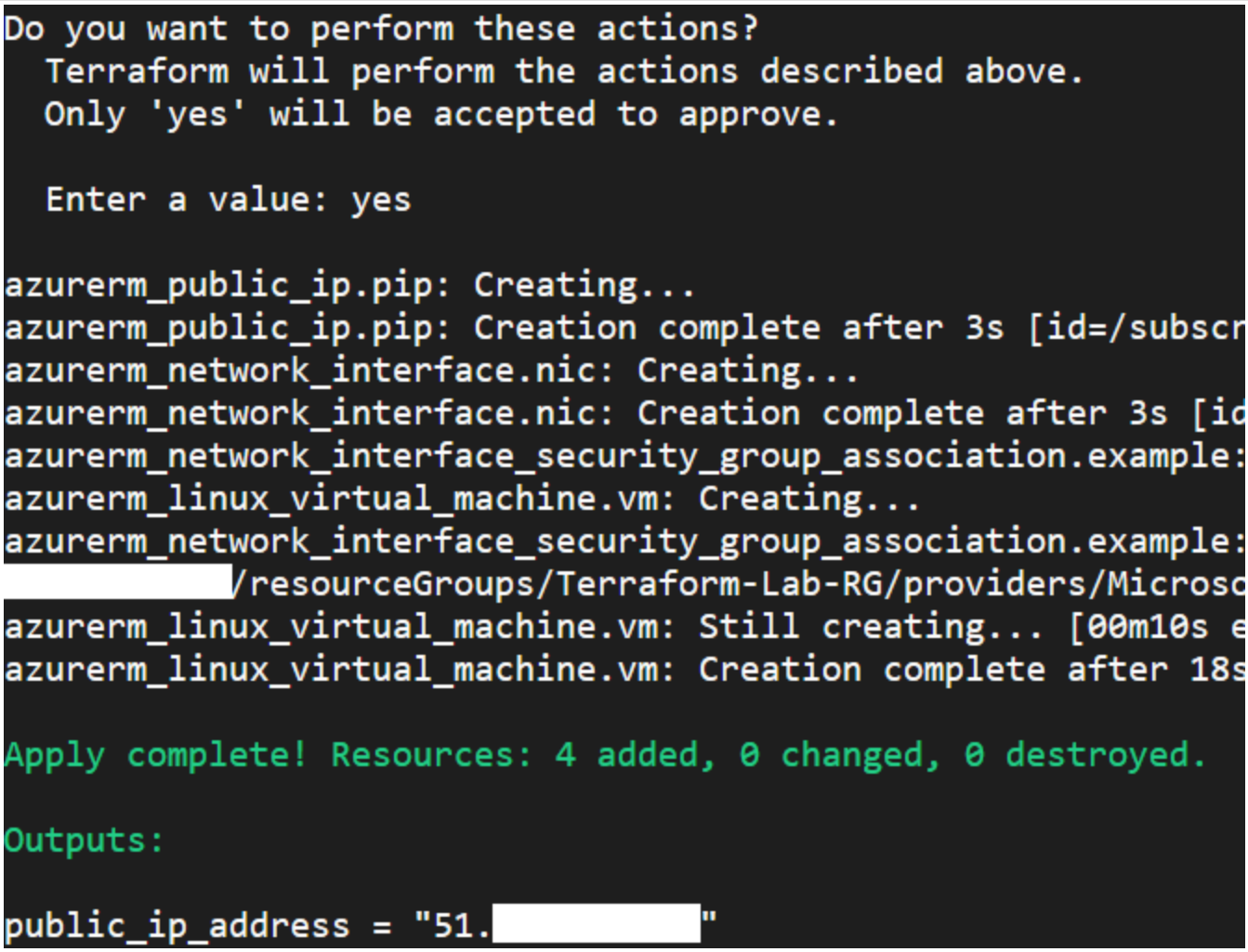
5. **Застосування:** Розгорніть ресурси в хмарі.

```
terraform apply
```

- Коли система запитає підтвердження `Enter a value:`, введіть `yes` і натисніть Enter.
- Чекайте завершення (це може зайняти 2-5 хвилин).

6. У кінці ви побачите повідомлення `Apply complete!` і вашу IP-адресу:

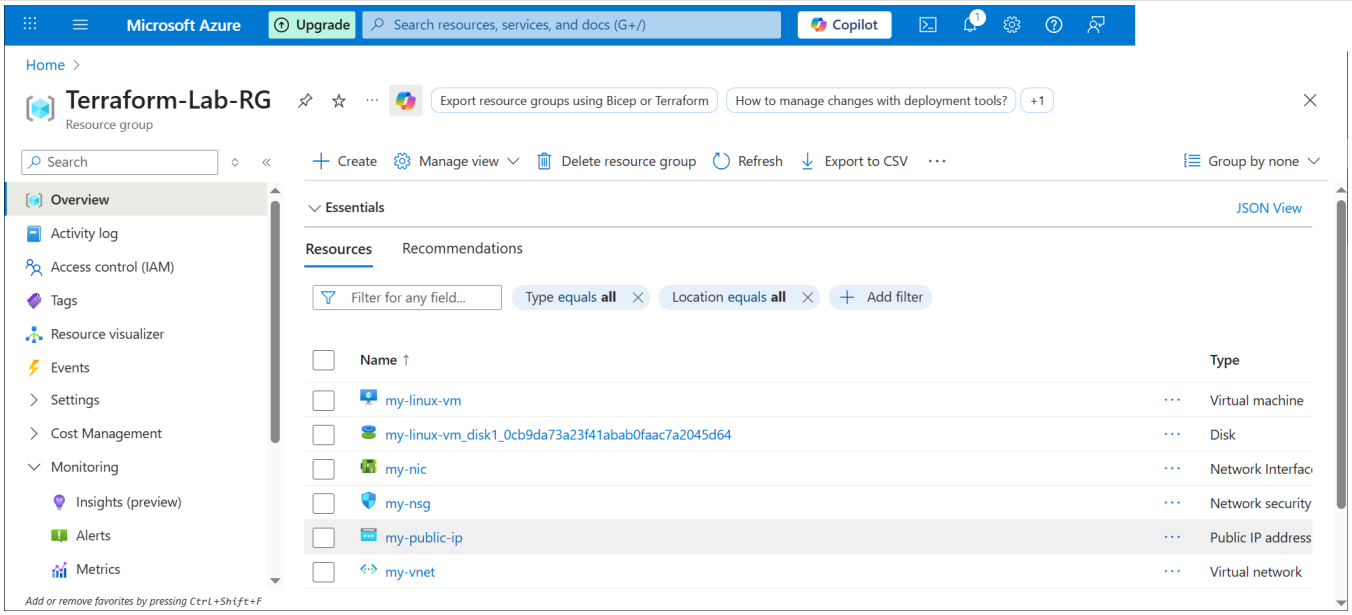
```
Outputs:
public_ip_address = "20.x.x.x"
```



Місце для скріншота: Результат виконання команди terraform apply у терміналі

Крок 5: Перевірка результатів

1. **Перевірка в порталі Azure:**
   - Зайдіть на portal.azure.com.
   - Перейдіть у "Resource groups".
   - Знайдіть групу `Terraform-Lab-RG`. В ній ви побачите всі створені ресурси (VM, Network, Disk тощо).

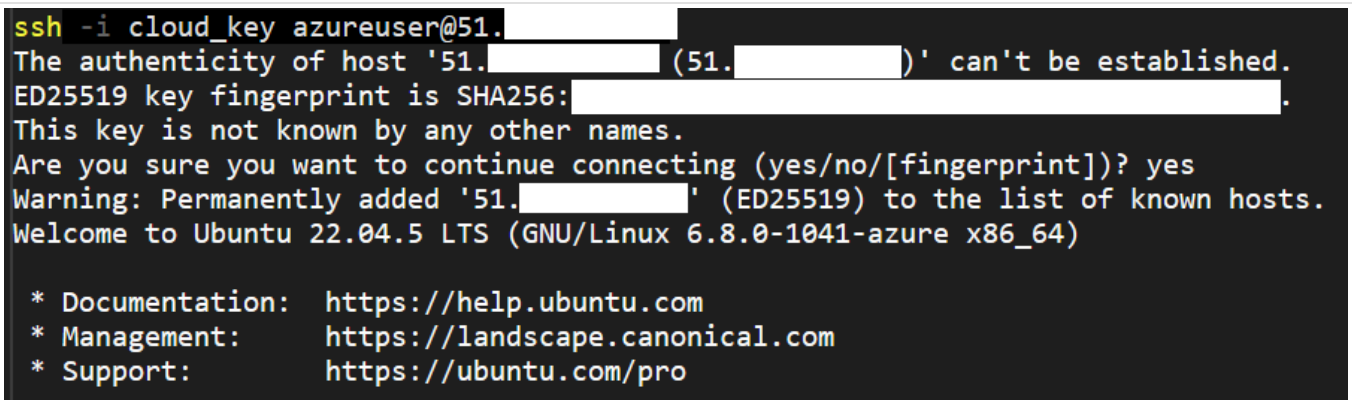


Місце для скріншота: Список ресурсів у створеній групі

2. **Підключення через SSH (опціонально):**
   - У терміналі VS Code введіть:

```
ssh -i cloud_key azureuser@<ВАША_IP_АДРЕСА>
```

   - Ви потрапите в консоль вашої віддаленої Linux машини.



Місце для скріншота: Виконання команди підключення

3. **Перевірка веб-сайту:**
   - Скопіюйте отриману `public_ip_address`.
   - Відкрийте браузер і вставте адресу: `http://<ВАША_IP_АДРЕСА>`.
   - Ви маєте побачити сторінку "Welcome to the Cloud Computing Development Module!".
     (Примітка: Якщо сайт не відкривається одразу, зачекайте 1-2 хвилини, поки скрипт `template.tpl` завершить встановлення Apache).

Налагодження веб-сервера (Troubleshooting)

Якщо після розгортання ви відкрили IP-адресу в браузері, але отримали помилку з'єднання (наприклад, `ERR_CONNECTION_REFUSED`), це означає, що скрипт автоматичного встановлення (`template.tpl`) не спрацював або ще не завершився. У цьому випадку налаштуйте сервер вручну.

1. **Підключіться до віртуальної машини через SSH:**
   У терміналі VS Code виконайте команду (замініть `<IP_АДРЕСА>` на вашу):

   ```
   ssh -i cloud_key azureuser@<IP_АДРЕСА>
   ```

   (Якщо запитає підтвердження `Are you sure...`, введіть `yes`).
2. **Встановіть та запустіть веб-сервер вручну:**
   Скопіюйте та вставте наступні команди у термінал SSH (по черзі або блоком):

   ```
   # Оновлення списків та встановлення Apache
   sudo apt-get update
   sudo apt-get install -y apache2 php libapache2-mod-php php-mysql

   # Запуск служби
   sudo systemctl enable apache2
   sudo systemctl start apache2
   ```
3. **Налаштуйте контент сайту:**
   Очистіть папку та створіть просту сторінку для перевірки:

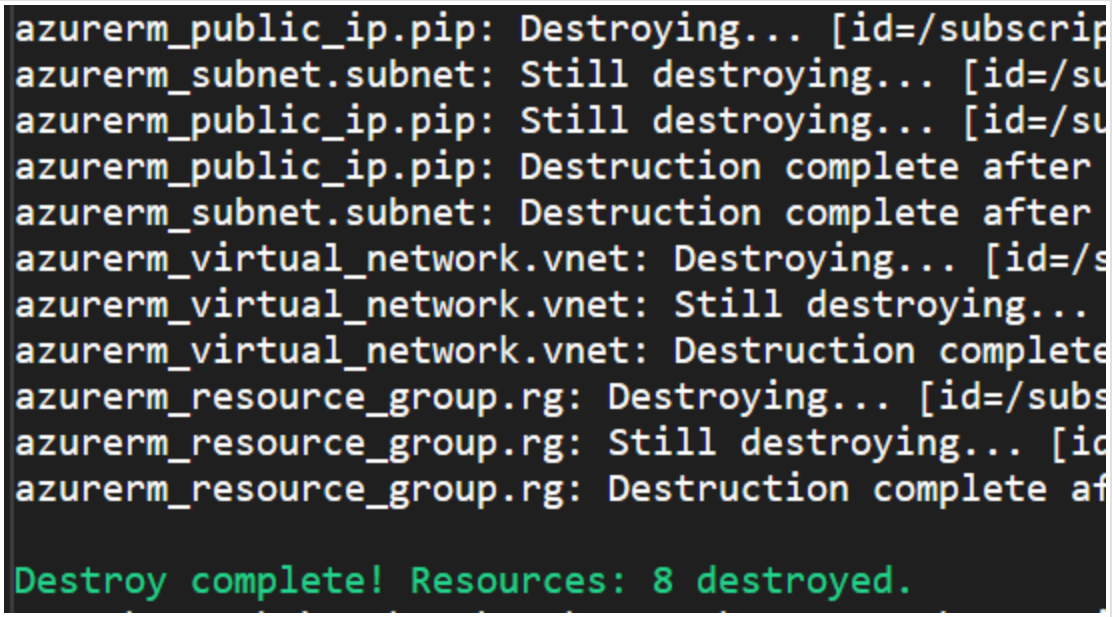
   ```
   # Очищення папки
   cd /var/www/html
   sudo rm -rf *

   # Створення тестової сторінки
   echo '<html><head><title>Apache Default</title></head><body><h1>Apache is
   running correctly!</h1><p>Manual setup successful.</p></body></html>' |
   sudo tee index.html

   # Перезапуск Apache
   sudo systemctl restart apache2
   ```
4. **Перевірка:**
   Знову відкрийте вашу IP-адресу у браузері. Ви маєте побачити сторінку з текстом **"Apache is running correctly!"**.

Місце для скріншота: Відкритий веб-сайт

Крок 6: Видалення ресурсів

ВАЖЛИВО: Після завершення роботи обов'язково видаліть ресурси, щоб не витрачати кошти. Terraform робить це однією командою.

0. Для виходу з `ssh` введіть у терміналі:
   ```
   exit
   ```
1. У терміналі введіть:
   ```
   terraform destroy
   ```
2. Підтвердіть дію, ввівши `yes`.
3. Terraform автоматично видалить усі ресурси, які він створив (VM, IP, мережі тощо).



Місце для скріншота: Підтвердження видалення групи ресурсів