Warszawa, 2023-06-03

*Prowadzący: Marcin Bodzan*
**Automatyzacja zadań w chmurze 2**
Zadanie końcowe

# Zadanie końcowe:
# wdrożenie środowiska dla nowej aplikacji.


Autor: **Mieszko Urbanowski**
Numer albumu: 3421
Grupa: IZ08TC1

# Spis treści

# Wykaz rysunków

# Cel laboratorium

Przygotowanie skryptu, który wykona wdrożenie środowiska dla nowej aplikacji.

## Wymagania

Struktura katalogowa skryptu:

- `\numerindexu\nazwaskryptu.ps1`
- `\numerindexu\log\log.txt`
- `\numerindexu\raport.txt`
- `\nuemrindexu\dane.txt`

**Podstawowe informację, które należy pobrać od użytkownika:**

- numer indeksu
- Region: (Default może być ustawiony na North Europe)

**Dane, które muszą być zapisane w zmiennych:**

- Nazwa grupy zasobów: **RG_numerIndexu**
- Nazwa VM: **VM-x-numerindexu**
- Wielkość VM: jakaś mała VM-ka (dopracować)
- Nazwa storage account: **mystorageacctplwit<NUMERINDEXU>**
- TAG name: **student** = TAG wartość: **<NUMERINDEXU>**

**Skrypt ma:**

- pobierać wszystkie potrzebne informację od użytkownika lub ze zmiennych(nie hard kodujemy żadnych informacji w kodzie)
- obsługiwać błędy (np.
- skrypt ma zawierać funkcję logowania zdarzeń do pliku (plik logu skryptu)
- skrypt ma zawierać komentarze, które opisują działanie poszczególnych sekcji kodu

- w skrypcie ma znajdować się
    - - blok zmiennych,
    - - blok z funkcjami i
    - - blok zwłaściwym skryptem

**Działanie skryptu:**

- utworzenie grupy zasobowej
- utworzenie VM z listy podanej w pliku dane.txt
- w pliku dane.txt znajduję się nazwy co najmniej 2 VM-ek
- Tworzy usługę LB do której podłączone są tworzone VM-ki
- utworzenie storage account
- utworzy NSG podłączy do VM-eki zezwoli na ruch tylko z portu 80 i 443
- utworzenie grupy security dla zespołu, który będzie zarządzał projektem
- przypisanie uprawnień grupie do grupy zasobów
- utworzenie raportu z listą wszystkich obiektów utworzonych dla projektu w wskazanej grupie zasobów z informacjami o TAGACH
- ostatni krok: usunie wszystkie obiekty, które zostały utworzone

**Wymagania dotyczące sprawozdania:**

- stronatytułowa
- spis treści
- spis ilustracji
- numerację stron
- cel laboratorium (treść zadania) i rozwiązanie podzielone na rozdziały
- Sprawozdanie ma być napisane jako dokumentacja rozwiązania, tak aby inna osoba po jego lekturze była wstanie odtworzyć zadanie

# Sprawozdanie

## Opis skryptu (ogólny):

Struktura plików:

**.\3421\nazwaskryptu.ps1** – plik ze skryptem który będziemy uruchamiać

**.\3421\dane.txt** – plik csv z danymi maszyn wirtualnych (nazwa,login,hasło)

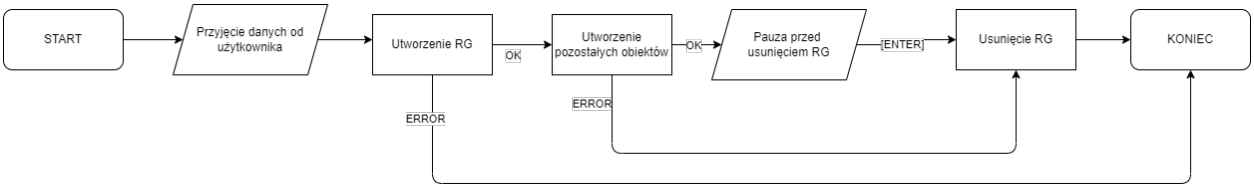
**.\3421\log\log.txt** – plik logu do którego będą zapisywane przez program działania

**.\3421\raport.txt** – plik raportu który zostanie wygenerowany na końcu

Po uruchomieniu pliku **nazwaskryptu.ps1** skrypt pobiera od użytkownika numer indeksu, region w którym mają być utworzone zasoby AZ oraz rozmiar maszyn wirtualnych.

Następnie tworzy kolejne obiekty zaczynając od grupy zasobów. Jeśli natrafi na błąd, przerywa działanie i usuwa utworzone zasoby.

Jeśli wykona się bez błędów wygeneruje plik **raport.txt** w którym będzie lista utworzonych obiektów oraz dane wejściowe z pliku **dane.txt.**




*Ilustracja 1: Ogólny diagram działania aplikacji*

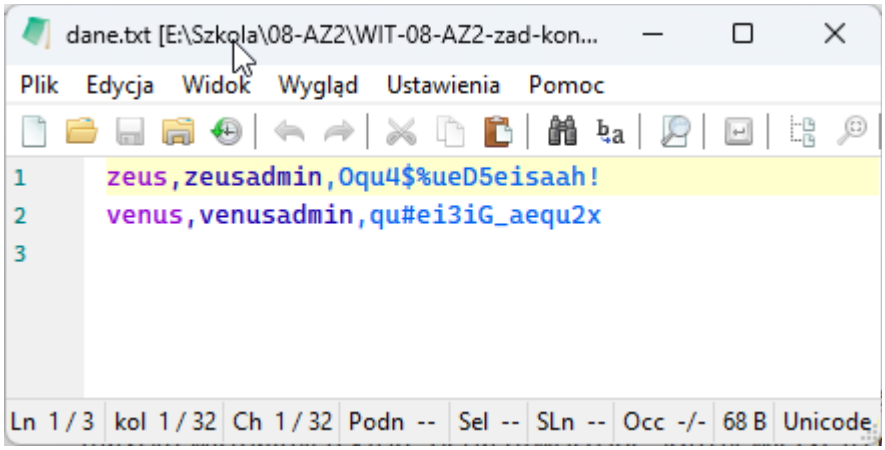
Przed zakończeniem pracy i usunięciem utworzonych obiektów skrypt wstrzyma działanie i pozwoli użytkownikowi na sprawdzenie efektów pracy.

Działanie skryptu można zobaczyć pod adresem: https://youtu.be/HndwAB7fnw4

# Sposób użycia

## Przygotowanie danych

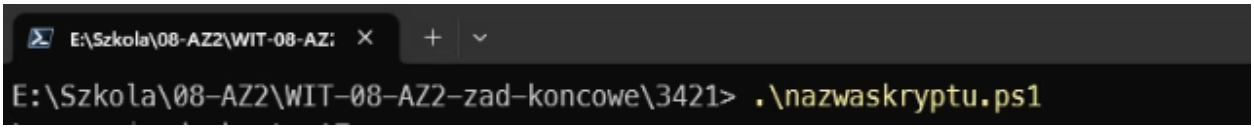
Przed uruchomieniem skryptu należy przygotować plik **dane.txt**. Znajdują się w nim dane maszyn wirtualnych które będą utworzone. Każdy wiersz to osobna maszyna. Format danych: nazwa_vm,login,hasło




*Ilustracja 2: Przykładowa treść pliku dane.txt*

## Uruchomienie

Skrypt należy uruchomić z poziomu PowerShell, będąc w folderze ze skryptem, wydając polecenie **.\nazwaskryptu.ps1**




*Ilustracja 3: Uruchamianie skryptu*

lub wykonać w PowerShell ISE

Jeśli ExecutionPolicy nie pozwala na uruchomienie skryptu można zmienić to (dla obecnej sesji powershell) poleceniem:

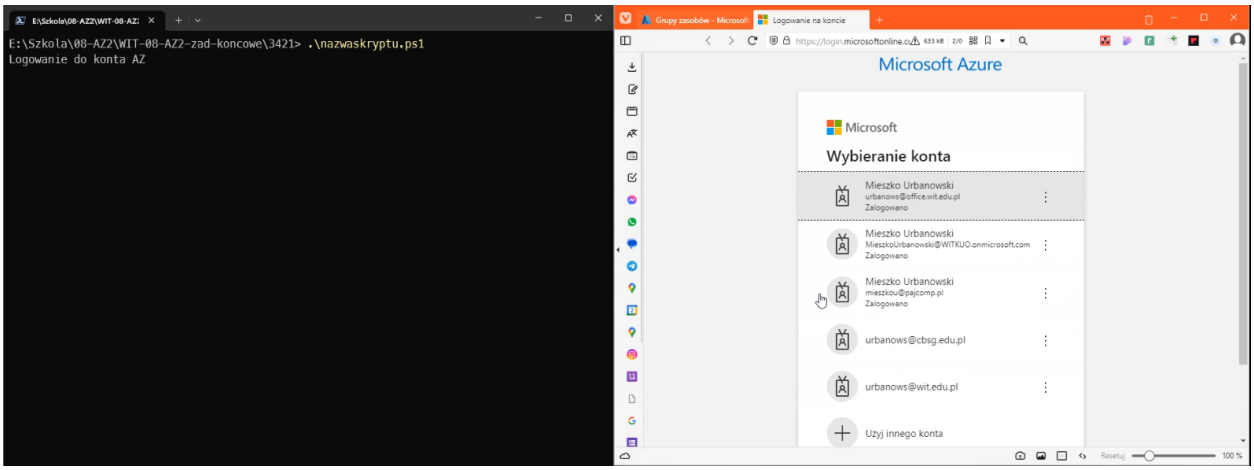
Set-ExecutionPolicy -ExecutionPolicy Unrestricted -Scope Process

Instalacja brakujących modułów:

```
Install-Module -Name Az.Accounts -Scope CurrentUser
Install-Module -Name Az.Compute -Scope CurrentUser
Install-Module -Name Az.Storage -Scope CurrentUser
Install-Module -Name Az.Resources -Scope CurrentUser
Install-Module -Name Az.Network -Scope CurrentUser
Install-Module -Name AzureAD -Scope CurrentUser
```

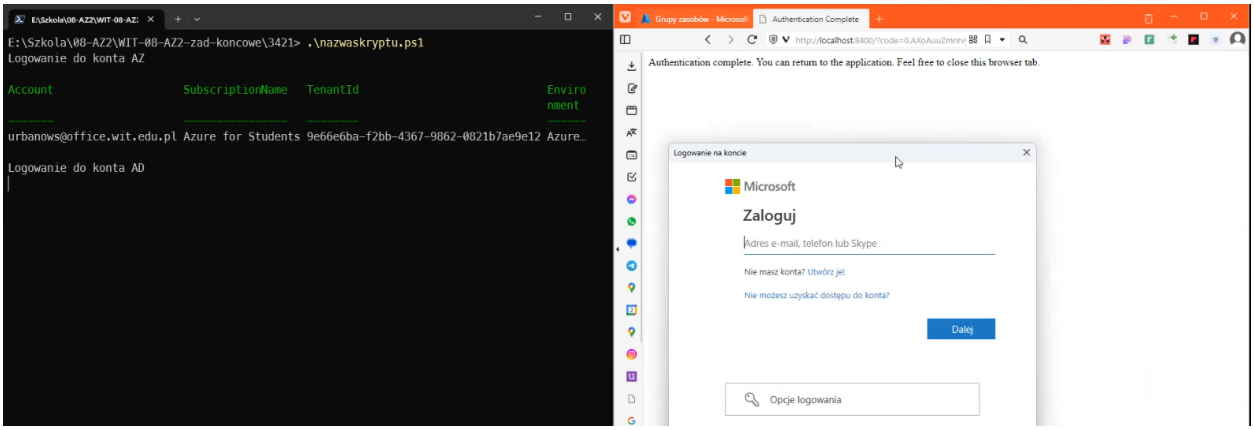
## Logowanie do AZ i AD

Następnie użytkownik zostanie poproszony uwierzytelnienie się w Microsoft Azure.




*Ilustracja 4: Uwierzytelnienie Microsoft Azure*

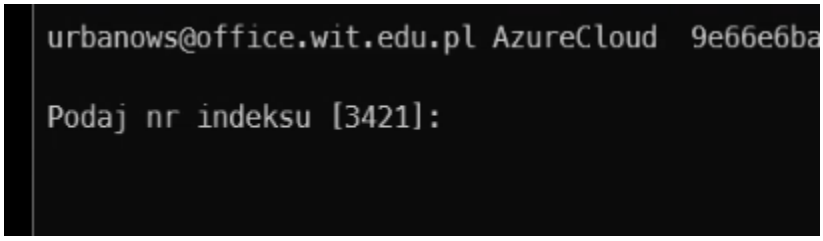
oraz o zalogowanie do konta AD (w celu utworzenia grupy zabezpieczeń)




*Ilustracja 5: Uwierzytelnianie AD*

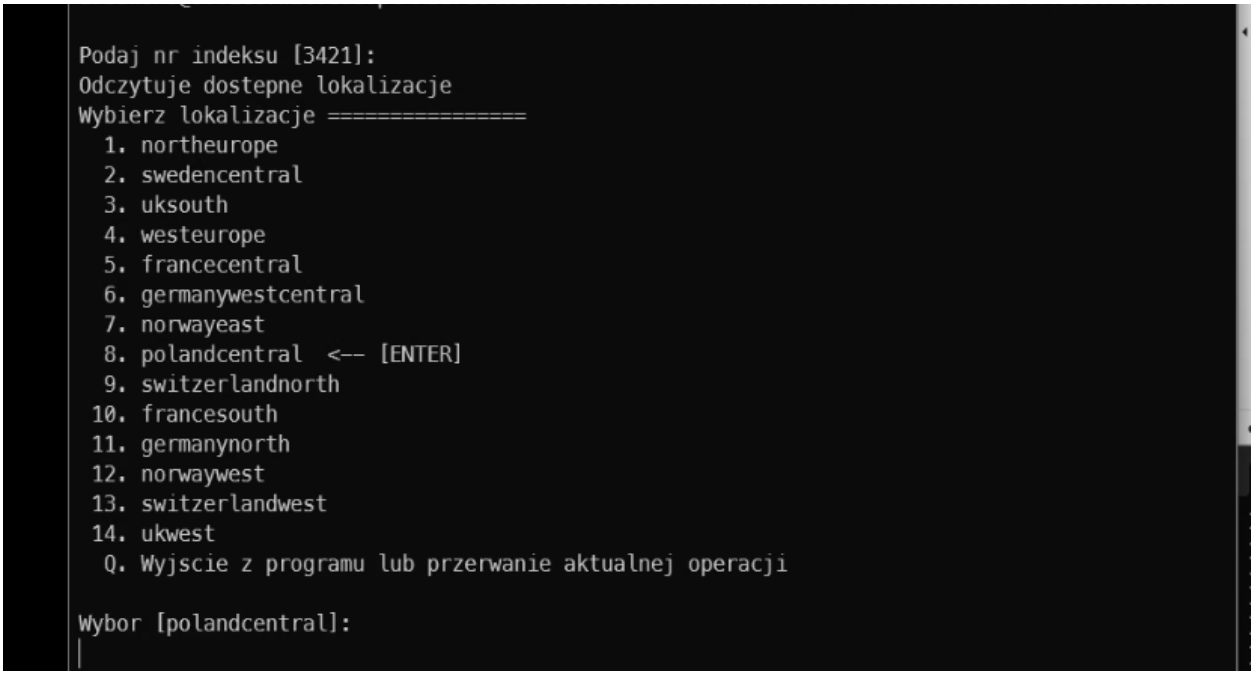
## Wprowadzenie danych wejściowych

Po zalogowaniu się, użytkownik zostaje zapytany o numer indeksu. Skrypt proponuje domyślną wartość którą jest nazwa folderu w którym znajduje się plik nazwaskryptu.ps1, jeśli jest poprawna wystarczy nacisnąć Enter, jeśli nie możemy wpisać inną.



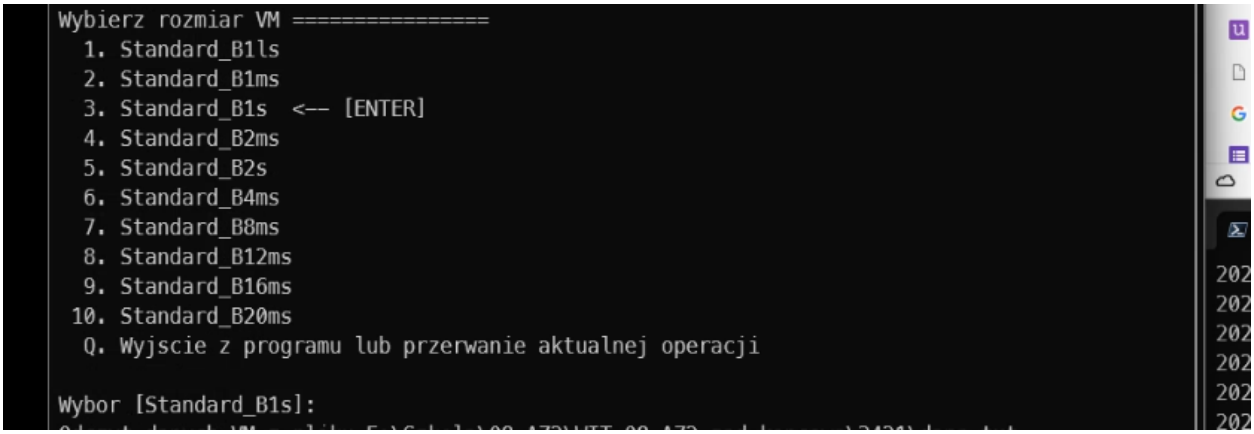

*Ilustracja 6: Pobieranie danych od użytkownika - numer indeksu*

Następnie skrypt odczytuje dostępne regiony, domyślnie lista jest ograniczona do Europy (można to zmienić w bloku zmiennych skryptu). Tu także skrypt proponuje domyślną wartość- polandcentral, jeśli jest poprawna wystarczy nacisnąć Enter, jeśli nie możemy wybrać inną podając numer.




*Ilustracja 7: Wybór regionu*

Ostatnim krokiem jest wybór rozmiaru maszyny wirtualnej. Skrypt proponuje domyślną wartość- Standard_B1s, jeśli jest poprawna wystarczy nacisnąć Enter, jeśli nie możemy wybrać inną podając numer.



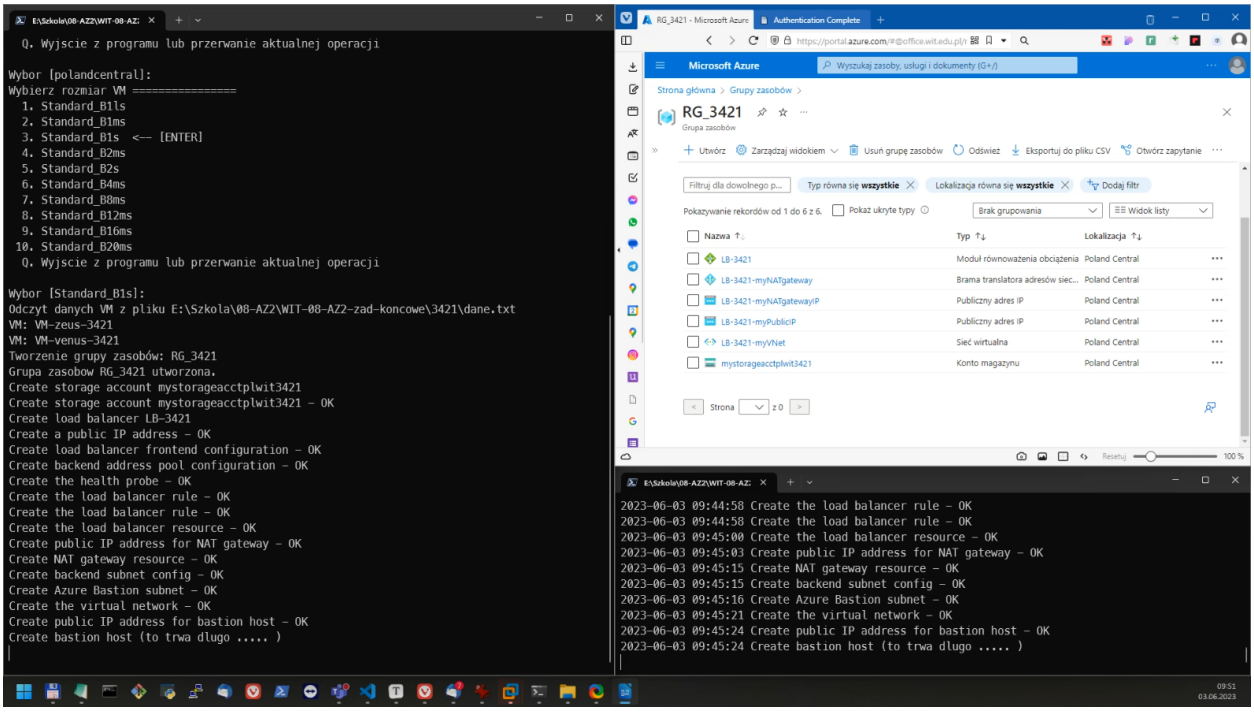

*Ilustracja 8: Wybór rozmiaru VM*

## Działanie skryptu

Skrypt rozpoczyna pracę i tworzy kolejno:

- Grupę zasobów
- Storage account
- Load Balancer (tu powstają niezbędne do jego działania elementy takie jak publiczne adresu IP, podsieci, reguły etc.
- Maszyny wirtualnej
- Grupa zabezpieczeń
- Do wszystkich obiektów przypisywany jest tag „student”

W końcowej fazie działania generowany jest plik raport.txt z listą utworzonych elementów.

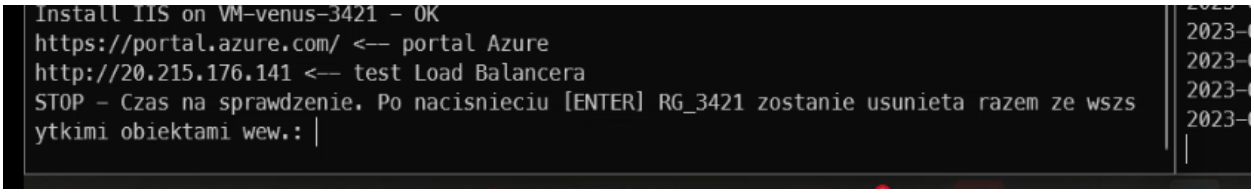



*Ilustracja 9: Działanie skryptu*

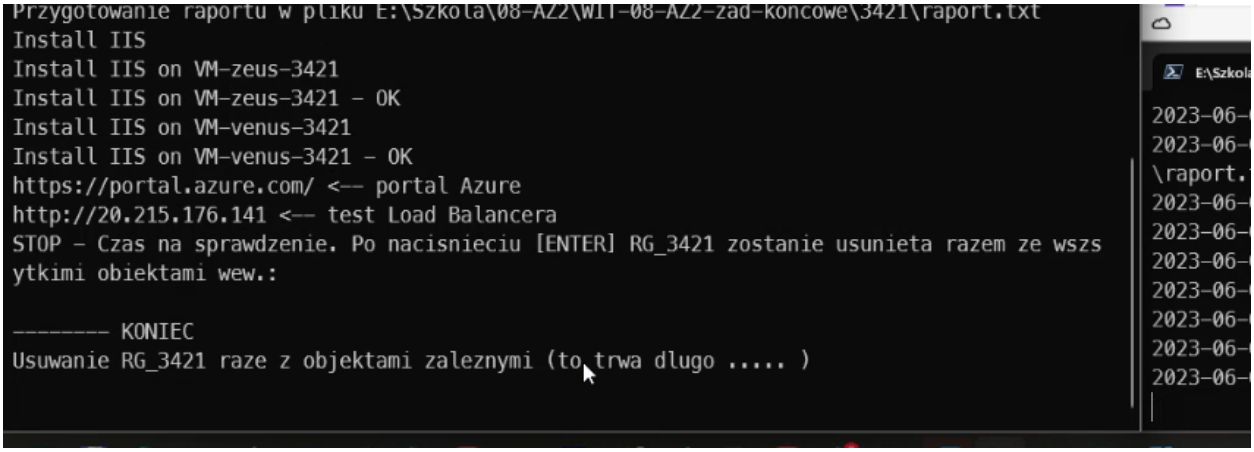
Aby umożliwić sprawdzenie load balancera na maszynach wirtualnych skrypt instaluje IIS (każda maszyna zwraca stronę www ze swoją nazwą).

## Koniec pracy

Użytkownikowi prezentowany jest adres IP pod którym można sprawdzić działanie wdrożenia, a program oczekuje na Enter aby przystąpić do usuwania utworzonych elementów.




*Ilustracja 10: Prezentacja wyniku działania skryptu*




*Ilustracja 11: Koniec pracy aplikacji*

Czas wykonania skryptu, przy 2 VM, wyniósł ok 20 minut.

# Opis skryptu (szczegóły):

Skrypt podzielony jest na 3 części:

- Blok zmiennych – tu zawarte są zmienne użyte w skrypcie. Tu możemy też dostosować działanie skryptu do własnych wymagań.
- Blok funkcji – zawiera kilka pomocniczych funkcji użytych w skrypcie
- Blok skryptu – główna część skryptu.

## Blok zmiennych

Modyfikując poniższe wartości, możemy zmienić działanie skryptu.

**$defaultRegion** - domyślnie proponowany region w którym będą tworzone obiekty

**$onlyEurope = $true** – ustawienie na $true powoduje że użytkownikowi są proponowane tylko europejskie lokalizacje, ustawienie na $false spowoduje że menu będzie zawierać wszystkie dostępne regiony

**$defaultVmSize = "Standard_B1s"** – domyślna wartośc rozmiaru maszyny

**$defaultVmSizesToShow = "Standard_B*"** - filtr wg którego są wyświetlane możliwe do wyboru rozmiary maszyn wirtualnych

**$scriptFolder = Split-Path $MyInvocation.MyCommand.Path -Parent** – folder w którym znajduje się skrypt i pliki (poniżej) – domyślnie jest ustawiony tu folder w którym jest plik nazwaskryptu.ps1

**$logFile** = "$scriptFolder\log\log.txt" – plik logu

**$dataFile** = "$scriptFolder\dane.txt" – plik z danymi wejściowymi

**$reportFile** = "$scriptFolder\raport.txt" – plik raportu

**$tagName = "student"** – nazwa tagu przypisanego do obiektów (wartością będzie podany przez użytkownika numer indeksu)

Nazwy zasobów tworzone są na podstawie poniższych szablonów i numeru indeksu. Można je dowolnie modyfikować. Nazwy obiektów związanych z load balancerem będą utworzone w oparciu o szablon **$nameTemplateLB**

**$nameTemplateRG = "RG_{0}"** # $rgName = $nameTemplateRG -f

**$nameTemplateVM = "VM-{0}-{1}"** # $vmName = $nameTemplateVM -f $vmName-FromFile, $idNumber

**$nameTemplateLB = "LB-{0}"** # $lbName = $nameTemplateLB -f $idNumber

**$nameTemplateSG = "SG-{0}"** # $sgName = $nameTemplateSG -f $idNumber

**$nameTemplateSA = "mystorageacctplwit{0}"** # $saName = $nameTemplateSA -f $idNumber

Domyślny numer indeksu jest odczytywany w tym miejscu z nazwy katalogu w którym znajduje się skrypt.

**$defaultIdNumber = Split-Path $MyInvocation.MyCommand.Path -Parent | Split-Path -Leaf**

Można tu ustawić inną domyślną wartość np.:
**$defaultIdNumber = '3421'**

---

## Blok funkcji

### Funkcja Write-Log

Param ([string]$LogString)

Funkcja pobiera $LogString który dopisuje do pliku $logFile dodając znacznik czasu.

### Funkcja Write-HostAndLog

param ([string]$LogString)

Funkcja pobiera $LogString który wyświetla w konsoli użytkownikowi i zapisuje do pliku logu przy użyciu funkcji Write-Log.

### Funkcja Show-Except

param ([System.Collections.Generic.List[PSObject]] $obj)

Funkcja do obsługi wyjątków. Wyświetla użytkownikowi opis problemu i zapisuje go w pliku logu przy użyciu funkcji Write-Log.

### Funkcja Show-Menu

param ([string]$MenuName, [string[]]$Options, [string]$defaultValue)

Funkcja wyświetla użytkownikowi menu $MenuName na podstawie tablicy $Options. Można wskazać domyślną wartość $defaultValue. Używana w funkcji Get-User-Choice

### Funkcja Get-User-Choice

param ([string]$MenuName, [string[]]$Options, [string]$defaultValue)

Funkcja wywołuje funkcję Show-Menu i zwraca wybrany przez użytkownika ciąg znaków.

### Funkcja Get-User-Input

param ([string]$UserPrompt, [string]$defaultValue)

Funkcja pobierająca tekst od użytkownika, z możliwością ustawienia wartości domyślnej

## Funkcja Test-VMName

param ([string]$ComputerName)

Funkcja sprawdzająca czy nazwa vm $ComputerName spełnia poniższy warunek

*„Windows computer name cannot be more than 15 characters long, be entirely numeric, or contain the following characters: ` ~ ! @ # $ % ^ & * ( ) = + _ [ ] { } \ | ; : . ' " , < > / ?.”*

Funkcja wygenerowana przez https://chat.openai.com

# Blok skryptu

## Sprawdzenie wymagań

Skrypt sprawdza czy są zainstalowane wymagane do działania modułu, jeśli nie wyświetla użytkownikowi stosowną informację z podpowiedzią co zrobić żeby zainstalować brakujące moduły.

Jeśli są, importuje je i przechodzi do logowania użytkownika.

```
Import-Module Az.Accounts
Import-Module Az.Compute
Import-Module Az.Storage
Import-Module Az.Resources
Import-Module Az.Network
Import-Module AzureAD
```

## Logowanie do AZ i AD

Wywoływane są **Connect-AzAccount** a następnie **Connect-AzureAD** (potrzebne do utworzenia grupy zabezpieczeń) w celu połączenia się z kontami Microsoft.

*W tym kroku, kilkukrotnie i na różnych maszynach PowerShell zwracał mi błąd:*
*(…)*
*... One or more errors occurred. (Could not load type 'System.Security.Cryptography.SHA256Cng' from assembly 'System.Core, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089'.)*
*Connect-AzureAD: .…*
*(…)*

*Pomagało wywołanie polecenia:*

***Import-Module AzureAD -UseWindowsPowerShell***

*Nie udało mi się zidentyfikować przyczyny problemu.*

W przypadku błędu działanie programu jest przerywane.

## Pobranie danych od użytkownika

Pobieramy od użytkownika nr indeksu z propozycja domyślnej wartości.

Na tej podstawie będą wygenerowane nazwy tworzonych obiektów.

Następnie skrypt odczytuje dostępne regiony, skrypt proponuje domyślną wartość- polandcentral, jeśli jest poprawna wystarczy nacisnąć Enter, jeśli nie możemy wybrać inną podając numer.

W kolejnym kroku skrypt odczytuje do tablicy $vms dane tworzonych maszyn wirtualnych, ustawia nazwę vm na podstawie numeru indeksu i nazwy vm odczytanej z pliku dane.txt zgodnie z szablonem zdefiniowanym w bloku zmiennych.

## Działania skryptu

Skrypt na podstawie pobranych od użytkownika danych oraz szablonów z bloku zmiennych tworzy nazwy obiektów, ustala wartość tag-u etc.

Pierwszym tworzonym obiektem jest grupa zasobów.

Jeśli wystąpi błąd skrypt przerywa działanie.

Jeśli udało się utworzyć grupę tworzone są kolejne obiekty. Błąd od tego momentu będzie oznaczał że przed przerwaniem pracy, skrypt usunie wcześniej utworzone obiekty.

## Tworzenie obiektów

*Obiekty niezbędne do pracy load balancera utworzyłem w oparciu o [https://learn.microsoft.com/en-us/azure/load-balancer/quickstart-load-balancer-standard-public-powershell](https://learn.microsoft.com/en-us/azure/load-balancer/quickstart-load-balancer-standard-public-powershell)*

Kolejno są tworzone:

- Storage account
- Public IP address
- Load balancer frontend configuration
- Backend address pool

- Health probe
- Load balancer rules and rules config
- Load balancer
- Public IP address for NAT gateway
- NAT gateway
- Backend subnet config
- Azure Bastion subnet
- Virtual network
- Public IP address for bastion host
- Bastion host
- Rules for network security group
- Create network security group
- Virtual machines
- Security Group

## Przypisanie tagów

Do wszystkich obiektów przypisany zostaje tag $tagName (domyślnie: "student") z wartością $tagValue (domyślnie: $idNumber)

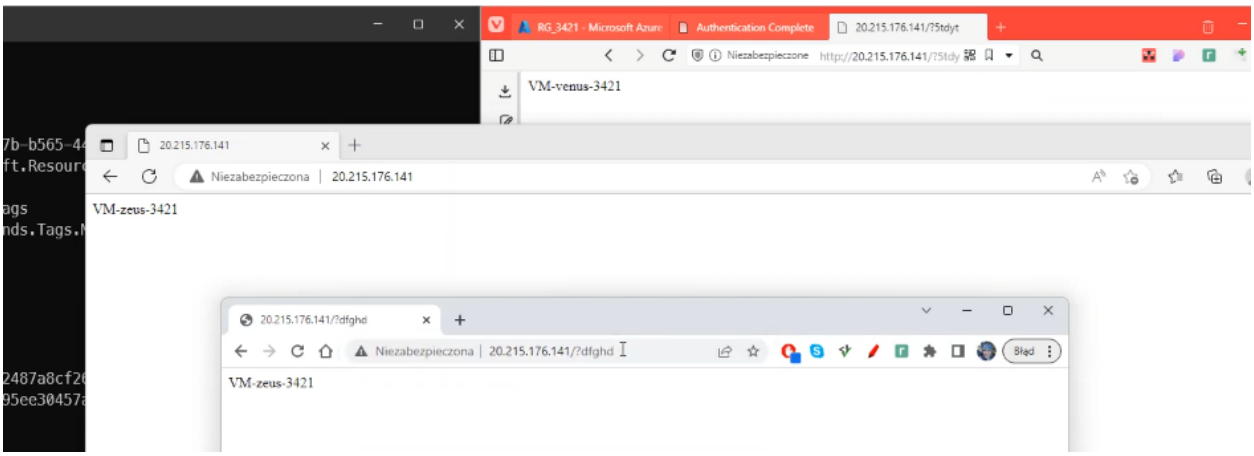
## Raport

Po przypisaniu tagów wygenerowany zostaje plik z raportem zawierającym wszystkie obiekty które utworzyliśmy.

Na końcu pliku wstawiona zostaje też zawartość pliku dane.txt z danymi na podstawie których tworzyliśmy VM-ki.

## Instalacja IIS

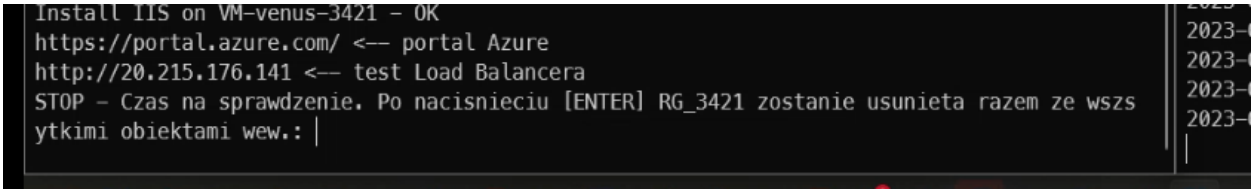
Aby ułatwić testowanie działania load balancera na wszystkich maszynach wirtualnych zostaje zainstalowany IIS, oraz stworzona strona www która prezentuje nazwę maszyny wirtualnej. Pozwoli to zilustrować działanie LB.



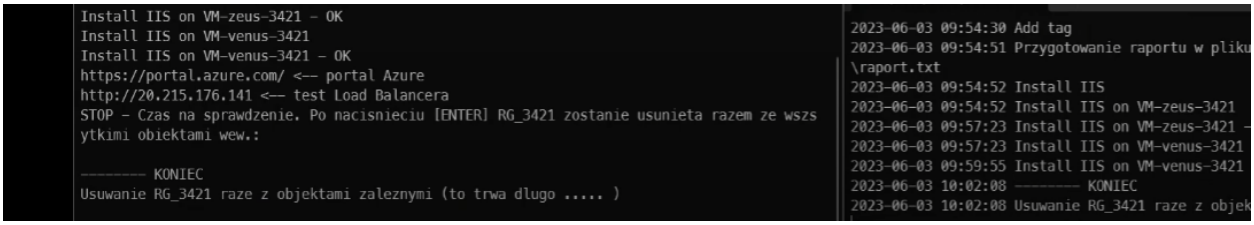

*Ilustracja 12: Efekt działania load balancera*

## Koniec pracy

Użytkownikowi prezentowany jest adres IP pod którym można sprawdzić działanie wdrożenia, a program oczekuje na Enter aby przystąpić do usuwania utworzonych elementów.




*Ilustracja 13: Prezentacja wyniku działania skryptu*




*Ilustracja 14: Koniec pracy*