

Politechnika Wrocławska

Wydział Informatyki i Telekomunikacji

**Studia podyplomowe
„Administrowanie Sieciami Komputerowymi” edycja 21**

PRACA KOŃCOWA

**Projekt i wdrożenie
skalowalnej infrastruktury w chmurze AWS
dla wojskowej uczelni wyższej**

Przemysław Stanisław

Opiekun pracy

dr inż. Wojciech Thomas

Słowa kluczowe: AWS, cloud, terraform, VPC, S3 Bucket

WROCŁAW 2025

Streszczenie

Tytuł Pracy: Projekt i wdrożenie skalowalnej infrastruktury w chmurze AWS dla wojskowej uczelni wyższej.

Cel Pracy:

Głównym celem pracy było zaprojektowanie i wdrożenie bezpiecznej, wydajnej infrastruktury IT w chmurze AWS dla jednej z komórek organizacyjnych AWL tj. Oddziału Koordnacji Kształcenia i Szkolenia.

Zakres Pracy:

- Projekt sieci VPC: Stworzenie wirtualnej sieci prywatnej z podziałem na podsieci publiczne i prywatne oraz zaimplementowanie mechanizmów routingu (Internet Gateway, NAT Gateway).
- Hosting danych w S3: Utworzenie izolowanych bucketów dla trzech grup użytkowników, konfiguracja hostingu statycznych stron WWW oraz polityk dostępu.
- Dostęp zdalny: Implementacja kontroli dostępu opartego o adresację IP.
- Bezpieczeństwo: Wielowarstwowa ochrona danych.

Metodologia:

Wykorzystano podejście Infrastructure as Code (IaC) z narzędziem Terraform. Konfiguracja obejmowała:

- Automatyzację wdrożenia sieci VPC i podsieci,
- Dynamiczne generowanie bucketów S3 z unikalnymi nazwami,
- Implementację polityk bezpieczeństwa.

Wnioski:

Wdrożona infrastruktura udowodniła, że chmura AWS może spełniać rygorystyczne wymagania wojskowe przy zachowaniu elastyczności i skalowalności. Powodzenie uzyskano poprzez zastosowanie automatyzacji (Terraform) oraz wielowarstwowego modelu bezpieczeństwa.

Abstract

Thesis Title: Design and Implementation of Scalable AWS Cloud Infrastructure for a Military University.

Objective:

The primary objective of this work was to design and implement a secure, efficient IT infrastructure in the AWS cloud for an organizational unit of the Military University of Land Forces (Akademia Wojsk Lądowych) - specifically Oddział Koordynacji Kształcenia i Szkolenia).

Scope of Work:

- VPC Network Design: Creation of a virtual private cloud (VPC) with segmentation into public and private subnets, and implementation of routing mechanisms (Internet Gateway, NAT Gateway),
- S3 Data Hosting: Deployment of isolated storage buckets for three user groups, configuration of static website hosting, and implementation of access policies,
- Remote Access: Implementation of IP address-based access control,
- Security: Multi-layered data protection architecture.

Methodology:

The Infrastructure as Code (IaC) approach was implemented using Terraform, featuring:

- Automated deployment of VPC networks and subnets,
- Dynamic generation of uniquely named S3 buckets,
- Implementation of granular security policies.

Conclusions:

The deployed infrastructure demonstrates that AWS cloud solutions can meet stringent military requirements while maintaining flexibility and scalability. The key success factors were the implementation of automation (Terraform) and a multi-layered security model.

Spis treści

Wstęp	7
1. Podstawy Technologiczne	8
1.1. Architektura chmurowa AWS	8
1.2. Narzędzie automatyzacji – Terraform	8
1.3. Hierarchia wdrożeniowa	8
2. Definicja wymagań	9
2.1. Uwagi ogólne	9
2.2. Wydział Planowania Kształcenia	9
2.3. Wydział Programowania Kształcenia	9
2.4. Materiały dla wykładowców	9
3. Projekt infrastruktury IT w AWS dla OKKiSz	11
4. Implementacja projektu infrastruktury IT w AWS dla OKKiSz	14
4.1. Architektura sieciowa	14
4.1.1. Projekt VPC	14
4.1.2. Podział na podsieci	14
4.1.3. Bramy sieciowe	15
4.1.4. Mechanizmy routingu	16
4.2. Wdrożenie Bucketów S3	17
4.2.1. Projekt i konfiguracja bucketów	17
4.2.2. Hosting statycznych stron internetowych	18
4.2.3. Automatyczne przysyłanie plików	18
4.3. Bezpieczeństwo	20
4.3.1. Grupy bezpieczeństwa - Security Groups	20
4.3.2. Polityki dostępu - Bucket Policies	21
4.4. Umieszczenie projektu infrastruktury w chmurze AWS	25
4.4.1. Sprawdzenie kodu	25
4.4.2. Uruchomienie kodu	26
5. Test infrastruktury sieciowej oraz zasad dostępu	28
5.1. Test dostępu publicznego zewnętrznego	28
5.1.1. Dostęp do hostingu publicznego z dowolnego adresu IP	28
5.1.2. Dostęp do hostingu prywatnego z niedozwolonego adresu IP	29
5.1.3. Dostęp do hostingu prywatnego z dozwolonego adresu IP	30
5.2. Test dostępu wewnętrznego	31
5.2.1. Dostęp do hostingu prywatnego	31
Podsumowanie	32
Bibliografia	33

Wstęp

Kontekst organizacyjny Akademii Wojsk Lądowych

Akademia Wojsk Lądowych imienia generała Tadeusza Kościuszki jest wojskową uczelnią wyższą kształcącą profesjonalne kadry dowódcze na potrzeby Sił Zbrojnych RP, a także kadry menedżerską na potrzeby sektora obronności i rynku cywilnego. AWL dąży do tego, aby jej absolwenci byli dobrze przygotowani do służby na pierwszych stanowiskach oficerskich, posiadali cechy przywódcze i umiejętności dowódcze, a także potrafili podejmować decyzje w trudnych sytuacjach.

Odział Koordynacji Kształcenia i Szkolenia (OKKiSz) jest komórką organizacyjną AWL odpowiedzialną za projektowanie i realizację procesu dydaktycznego w Akademii.

Cel pracy

Celem pracy jest analiza, zaprojektowanie i wdrożenie infrastruktury IT w AWS dla wybranej komórki organizacyjnej (OKKiSz) Akademii Wojsk Lądowych imienia generała Tadeusza Kościuszki.

Zakres pracy

Zakres pracy obejmuje:

1. Projekt infrastruktury sieciowej w chmurze.
2. Konfiguracja wybranych zasobów (np. maszyny wirtualne, S3 Bucket itp.).
3. Konfiguracja zdalnego dostępu do zasobów.
4. Zabezpieczenie dostępu do utworzonych zasobów.

1. Podstawy Technologiczne

1.1. Architektura chmurowa AWS

Architektura chmurowa Amazon Web Services stanowi fundament projektu, oferując elastyczne i bezpieczne środowisko dla infrastruktury wojskowej uczelni. W modelu AWS przyjęto koncepcję współdzielonej odpowiedzialności, gdzie dostawca odpowiada za fizyczne zabezpieczenia centrów danych, a klient - za konfigurację logiczną i dane.

Kluczowe elementy to:

- Regiony i Strefy Dostępności: AWS zapewnia 84 strefy dostępności w 26 regionach, co umożliwia budowę odpornych systemów. Na potrzeby projektu przyjęto rejon us-east-1,
- Sieć Wirtualna (VPC): Pozwala stworzyć izolowaną sieć prywatną z pełną kontrolą nad adresacją IP, tabelami routingu i bramami sieciowymi. W projekcie zastosowano hierarchiczny podział na podsieci publiczne i prywatne,
- Model Bezpieczeństwa: Obejmuje izolację sieciową (sieci prywatne i publiczne) Security Groups (wirtualne zapory) oraz Bucket Policies (polityki dostępu do przestrzeni przechowywania danych).

1.2. Narzędzie automatyzacji - Terraform

Terraform firmy HashiCorp to wiodące narzędzie Infrastructure as Code (IaC), które umożliwia programistyczne definiowanie i wdrażanie infrastruktury chmurowej. Jego główne zalety dla projektu to:

- Deklaratywny język HCL: prosta konfiguracja zasobów AWS w plikach *.tf z czytelną składnią,
- Stan Infrastruktury: Terraform przechowuje aktualny stan infrastruktury w plikach .tfstate, co pozwala śledzić zmiany,
- Modułowość i ponowne użycie: konfigurację można podzielić na moduły (np. sieć, bezpieczeństwo), co przyspiesza wdrażanie w nowych środowiskach,
- Cykl życia zarządzania: pełna automatyzacja: planowanie (terraform plan), wdrażanie (terraform apply) i niszczenie zasobów (terraform destroy).

1.3. Hierarchia wdrożeniowa

W projekcie zastosowałem następującą hierarchię wdrożeniową:

1. Infrastruktura sieciowa (VPC, podsieci, routing)
2. Zasoby magazynowe (S3 bucket)
3. Bezpieczeństwo (Security Groups, Bucket policies)

2. Definicja wymagań

2.1. Uwagi ogólne

W projekcie uwzględniłem rzeczywistą strukturę Oddziału Koordynacji Kształcenia i Szkolenia. Oddział składa się z dwóch wydziałów: Wydział Planowania Kształcenia oraz Wydział Programowania Kształcenia. Każdy z wydziałów realizuje część zadań na korzyść kadry dydaktycznej AWL stąd potrzeba utworzenia izolowanych przestrzeni roboczych dla każdego z wydziałów oraz na potrzeby wykładowców akademickich.

2.2. Wydział Planowania Kształcenia

W przypadku Wydziału Planowania Kształcenia mielibyśmy do czynienia z następującą sytuacją: pracownicy tego wydziału mogą korzystać ze swojej prywatnej przestrzeni w chmurze – czyli specjalnie dla nich utworzonej przestrzeni dyskowej, a także przeglądać przygotowane dla nich strony internetowe (główna strona index.html i strona błędu error.html) służące do nawigacji po zasobach. Dodatkowo mają pełny dostęp do ogólnodostępnej przestrzeni przeznaczonej dla wykładowców, gdzie znajdują się materiały udostępnione wszystkim. Natomiast nie mają dostępu do zasobów Wydziału Programowania Kształcenia (czyli prywatnej przestrzeni dyskowej Wydziału Programowania Kształcenia). Dopuszczalny zasięg dostępu ograniczony jest do sieci wewnętrznej uczelni (adresy w zakresie 192.168.1.0–192.168.1.255) lub połączenia przez wirtualną sieć prywatną Amazon (VPC), co gwarantuje, że nikt spoza tej bezpiecznej sieci nie uzyska dostępu do plików wydziałowych.

2.3. Wydział Programowania Kształcenia

Dla Wydziału Programowania Kształcenia dostęp do zasobów został zaplanowany w podobny sposób jak dla pierwszego wydziału. Pracownicy tego wydziału mają możliwość korzystania ze swojej własnej, prywatnej przestrzeni dyskowej w chmurze. Mogą także przeglądać przygotowane dla nich strony internetowe (główna strona index.html i strona błędu error.html) służące do nawigacji po zasobach. Tak samo jak w przypadku pierwszego wydziału, mają oni dostęp do ogólnodostępnej przestrzeni przeznaczonej dla wykładowców, gdzie znajdują się materiały wspólne. Natomiast nie mają możliwości przeglądania zasobów przypisanych do Wydziału Planowania Kształcenia (czyli prywatnej przestrzeni dyskowej Wydziału Planowania Kształcenia). Aby móc korzystać z zasobów, muszą znajdować się w sieci wewnętrznej uczelni (adresy w zakresie 192.168.2.0–192.168.2.255) lub korzystać z bezpiecznego połączenia sieciowego Amazon (VPC).

2.4. Materiały dla wykładowców

W przypadku wykładowców, ich dostęp został maksymalnie uproszczony. Mają oni dostęp wyłącznie do ogólnodostępnej przestrzeni, w której znajdują się materiały publiczne. Nie mają natomiast możliwości przeglądania prywatnych zasobów żadnego z wydziałów – nie mogą więc dostać się ani do plików Wydziału Planowania Kształcenia, ani Wydziału Programowania Kształcenia. Ich dostęp nie jest ograniczony lokalizacją – oznacza to, że mogą korzystać z zasobów publicznych z dowolnego miejsca na świecie, o ile mają dostęp do Internetu.

Podsumowując:

1. Wydział Planowania Kształcenia

Dostęp:

- Prywatna przestrzeń dyskowa Wydziału Planowania Kształcenia
- Statyczne strony WWW Wydziału Planowania Kształcenia (index.html, error.html)
- ogólnodostępna przestrzeń przeznaczona dla wykładowców

Brak dostępu:

- Prywatna przestrzeń Wydziału Programowania Kształcenia

Lokalizacja dostępu poprzez:

- Sieć akademicka - zakres IP: 192.168.1.0/24
- Prywatne połączenie AWS VPC

2. Wydział Programowania Kształcenia

Dostęp:

- Prywatna przestrzeń dyskowa Wydziału Programowania Kształcenia
- Statyczne strony WWW Wydziału Programowania Kształcenia (index.html, error.html)
- ogólnodostępna przestrzeń przeznaczona dla wykładowców

Brak dostępu:

- Prywatna przestrzeń Wydziału Planowania Kształcenia

Lokalizacja dostępu:

- Sieć akademicka - zakres IP: 192.168.2.0/24
- Prywatne połączenie AWS VPC

3. Wykładowcy

Dostęp:

- Publiczny

Brak dostępu:

- Prywatna przestrzeń dyskowa Wydziału Planowania Kształcenia
- Prywatna przestrzeń dyskowa Wydziału Programowania Kształcenia

Lokalizacja dostępu:

- Dowolne miejsce na świecie - publiczny dostęp internetowy

3. Projekt infrastruktury IT w AWS dla OKKiSz

Celem praktycznej realizacji projektu wykonam poniższe czynności:

Przygotowanie i definicja środowiska VPC

W pierwszej kolejności tworzę odizolowaną chmurową sieć prywatną (VPC) o przestrzeni adresowej 10.0.0.0/16 i nazwie „OKKiSz”. Dzięki temu uzyskuje jednolite środowisko bazowe, w którym zakotwiczę wszystkie dalsze elementy infrastruktury.

Podział przestrzeni na podsieci

W ramach głównej sieci VPC tworzę trzy podsieci w jednej strefie dostępności (us-east-1a), aby zachować spójność adresacji i uprościć zarządzanie:

- Podsieć prywatna dla Wydziału Planowania (10.0.1.0/24), do której będą trafiać zasoby służące temu wydziałowi.
- Podsieć prywatna dla Wydziału Programowania (10.0.2.0/24), wydzielona analogicznie dla drugiego wydziału.
- Podsieć publiczna dla Wykładowców (10.0.3.0/24), skonfigurowana z automatycznym przypisywaniem publicznych adresów IP, umożliwiającą dostęp z Internetu.

Dzięki powyższemu zyskuje izolację sieciową dla grup użytkowników z poszczególnych wydziałów oraz wykładowców.

Konfiguracja bram i translacji adresów

Celem umożliwienia podsieci publicznej komunikacji z Internetem tworzę zasób Internet Gateway a następnie przypisuje go do VPC „OKKiSz”. Natomiast dla podsieci prywatnych tworzę NAT Gateway w strefie publicznej, przydzielając mu trwały adres Elastic IP. NAT Gateway pozwala instancjom w prywatnych podsieciach na inicjowanie połączeń wychodzących do Internetu, bez narażania ich na dostęp z sieci zewnętrznej.

Zdefiniowanie tablic routingu

Kolejnym krokiem jest utworzenie dwóch odrębnych Route Tables:

- Publiczna tablica routingu: kieruje cały ruch wychodzący (0.0.0.0/0) przez Internet Gateway i jest skojarzona tylko z podsiecią wykładowców.
- Prywatna tablica routingu: przekierowuje cały ruch wychodzący z podsieci prywatnych przez NAT Gateway.

Następnie mapuje każdą podsieć do właściwej tablicy, co gwarantuje, że ruch sieciowy wydziałów będzie ograniczony do VPC, a wykładowcy zyskają bezpośrednie połączenie z Internetem.

Tworzenie bucketów S3 i konfiguracja hostingu WWW

Dla każdego obszaru użytkowników tworzę osobny Bucket S3, używając unikalnych nazw generowanych dynamicznie. Celem realizacji wymagań dostępowych założonych w rozdziale 2 tworzę buckety o unikalnych nazwach dla każdej grupy użytkowników:

- Wydział Planowania Kształcenia – group_1
- Wydział Programowania Kształcenia – group_2
- Wykładowcy – group_3

W konfiguracji bucketów aktywuję statyczny hosting stron, definiując dokument startowy (index.html) oraz dokument błędu (error.html). Każdy z trzech zasobów hostingowych aws_s3_bucket_website_configuration przypisany jest indywidualnie dla poszczególnych

grup użytkowników (group_1, group_2 i group_3). Zasoby hostingowe są niezależne od siebie tj. będą występować pod trzema osobnymi adresami. Dzięki temu zyskuje prosty mechanizm publikacji materiałów WWW dla wydziałów oraz wykładowców.

Automatyzacja przesyłania plików do S3

Lokalne pliki HTML (index.html i error.html) synchronizuję z odpowiednimi bucketami, stosując mechanizm wykrywania zmian za pomocą sum kontrolnych (etag/MD5). Gdy zawartość pliku lokalnego ulegnie zmianie, Terraform zaktualizuje obiekt w S3 automatycznie, zapewniając spójną i aktualną zawartość stron.

Definicja Security Groups

Celem kontroli ruchu sieciowego dla każdej podsieci tworzę dedykowaną Security Group:

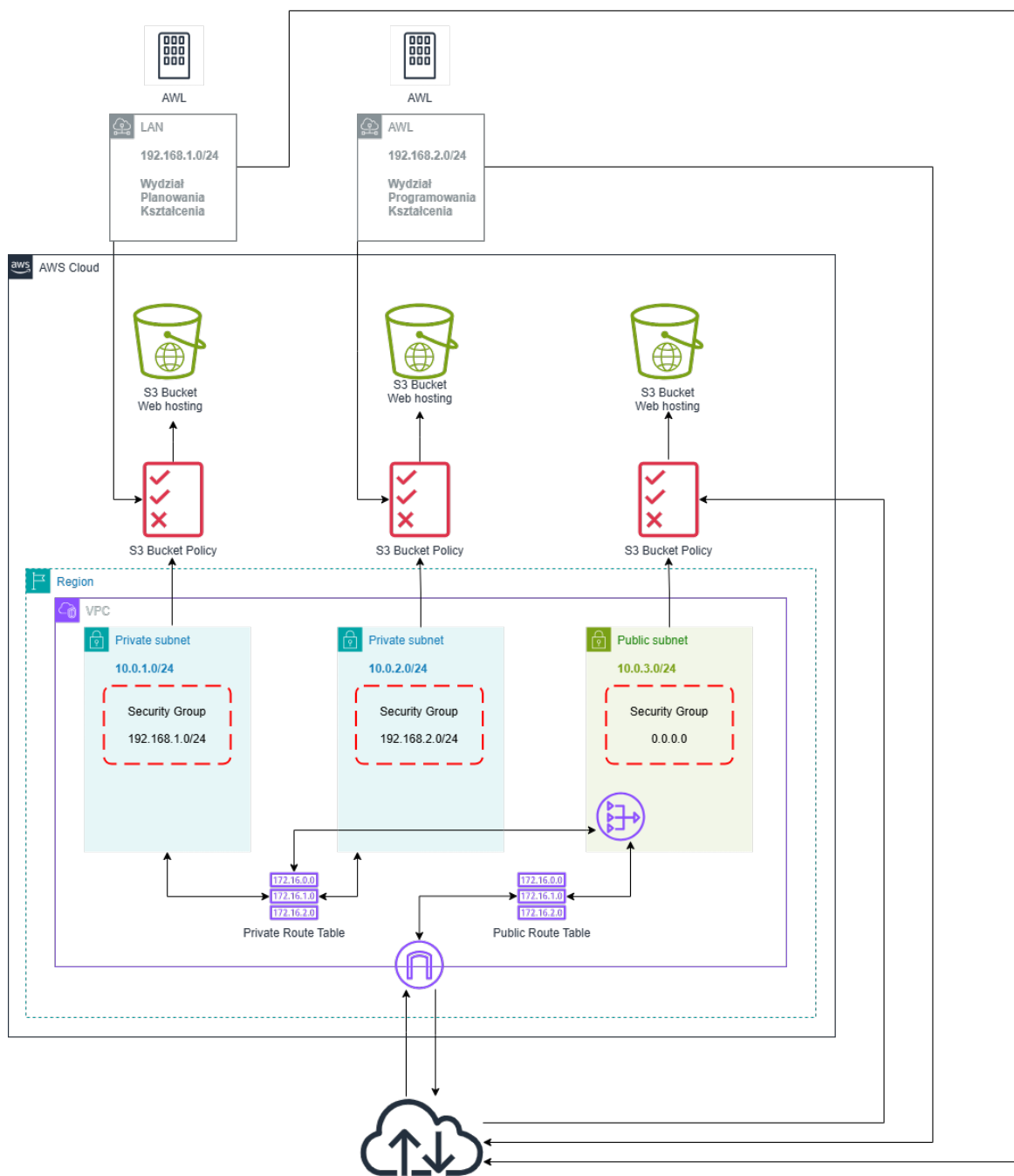
- SG dla Wydziału Planowania przyjmuje ruch tylko z zakresu IP 192.168.1.0/24.
- SG dla Wydziału Programowania akceptuje połączenia wyłącznie z adresów 192.168.2.0/24.
- SG publiczny otwiera ruch z dowolnego źródła.

Konfiguracja Bucket Policies

Dla każdego bucketa definiuję polityki, które precyzyjnie określają, kto i skąd może pobierać lub przeglądać obiekty:

- Buckety wydziałowe akceptują żądania GetObject i ListBucket tylko wtedy, gdy pochodzą z przypisanej puli adresów IP lub z zasobów wewnątrz VPC,
- Publiczny bucket zezwala na odczyt wszystkim (GetObject/ListBucket) i ma wyłączoną blokadę publicznego dostępu.

Uzyskaną za pośrednictwem powyższych kroków konfigurację infrastruktury IT OKKiSz w AWS przedstawia rys. 1.



Rys. 1 Diagram konfiguracji infrastruktury sieciowej OKKiSz w AWS

4. Implementacja projektu infrastruktury IT w AWS dla OKKiSz

Zgodnie z założeniem przyjętym w podrozdziale 1.2, do praktycznej realizacji projektu używam narzędzia Terraform firmy HashiCorp w oparciu o oprogramowanie firmy Microsoft Visual Studio Code. Wszystkie zmiany zapisuję w pliku main.tf. Wszystkie czynności wykonuję zgodnie z hierarchią wdrożeniową opisaną w podrozdziale 1.3 oraz założeniami przyjętymi w rozdziale 3.

4.1. Architektura sieciowa

4.1.1. Projekt VPC

Tworzę fundament sieciowy - Virtual Private Cloud. Przyjmuję przestrzeń adresową 10.0.0.0/16, która pomieści wszystkie komórki organizacyjne. Nadaję nazwę "OKKiSz" dla łatwej identyfikacji. To izolowane środowisko stanowi bezpieczną strefę dla wrażliwych danych wojskowych, chronioną przed nieautoryzowanym dostępem z zewnątrz.

```
resource "aws_vpc" "vpc_main" {
  cidr_block = "10.0.0.0/16"
  tags = {
    Name = "OKKiSz"
  }
}
```

4.1.2. Podział na podsieci

Dzielę główną sieć na trzy podsieci. Dla Wydziału Planowania Kształcenia i Wydziału Programowania Kształcenia tworzę podsieci prywatne (10.0.1.0/24 i 10.0.2.0/24), które nie mają bezpośredniego dostępu do Internetu. Dla Wykładowców projektuję podsieć publiczną (10.0.3.0/24) z automatycznym przypisywaniem publicznego IP. Każda w strefie dostępności us-east-1a dla zachowania spójności.

```
resource "aws_subnet" "private_1" {
  vpc_id      = aws_vpc.vpc_main.id
  cidr_block  = "10.0.1.0/24"
  availability_zone = "us-east-1a"

  tags = {
    Name = "Wydzial Planowania"
  }
}
```

```
resource "aws_subnet" "private_2" {
  vpc_id      = aws_vpc.vpc_main.id
  cidr_block  = "10.0.2.0/24"
  availability_zone = "us-east-1a"
```

```

tags = {
    Name = "Wydział Programowania"
}

resource "aws_subnet" "public_3" {
    vpc_id      = aws_vpc.vpc_main.id
    cidr_block = "10.0.3.0/24"
    availability_zone = "us-east-1a"
    map_public_ip_on_launch = true

    tags = {
        Name = "Wykładowcy"
    }
}

```

4.1.3. Bramy sieciowe

Zapewniam bezpieczny i kontrolowany dostęp do internetu dla różnych komórek organizacyjnych. Kluczowym elementem tego systemu jest Internet Gateway (IGW) - brama internetowa połączona bezpośrednio do sieci VPC. Dzięki temu komponentowi, podsieć publiczna (przeznaczona dla wykładowców) może inicjować połączenia z internetem i być dostępna z zewnątrz. Oznaczyłem ją znacznikiem "OKKiSz gateway" dla łatwej identyfikacji w konsoli AWS.

```

resource "aws_internet_gateway" "vpc_main_gateway" {
    vpc_id = aws_vpc.vpc_main.id

    tags = {
        Name = "OKKiSz gateway"
    }
}

```

Dla podsieci prywatnych (Wydział Planowania i Programowania) wdrożyłem NAT Gateway z Elastic IP - to rozwiązanie jest niezbędne dla bezpieczeństwa. Działa jak "pośrednik" między zasobami prywatnymi a internetem:

- Instancje w podsieciach prywatnych mogą inicjować połączenia wychodzące,
- Brak możliwości bezpośredniego dostępu do nich z internetu,
- Elastic IP zapewnia stały publiczny adres IP dla NAT.

NAT Gateway umieściłem w podsieci publicznej (public_3), ponieważ wymaga on bezpośredniego dostępu do internetu przez IGW.

```

resource "aws_eip" "nat_eip" {
    domain = "vpc"
}

resource "aws_nat_gateway" "vpc_main_nat" {

```

```

allocation_id = aws_eip.nat_eip.id
subnet_id     = aws_subnet.public_3.id
}

```

4.1.4. Mechanizmy routingu

Stworzyłem dwie główne tablice routingu dla ruchu sieciowego:

- Public Route Table:
 - Kieruje cały ruch (0.0.0.0/0) przez Internet Gateway,
 - Powiązana wyłącznie z podsiecią publiczną dla wykładowców.
- Private Route Table:
 - Kieruje ruch przez NAT Gateway,
 - Stosowana dla wszystkich podsieci prywatnych.

```

resource "aws_route_table" "public_route_table" {
  vpc_id = aws_vpc.vpc_main.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.vpc_main_gateway.id #
    powiązanie z IGW
  }

  tags = {
    Name = "publiczna tablica routingu"
  }
}

resource "aws_route_table" "private_route_table" {
  vpc_id = aws_vpc.vpc_main.id

  route {
    cidr_block = "0.0.0.0/0"
    nat_gateway_id = aws_nat_gateway.vpc_main_nat.id # powiązanie
    z NAT
  }

  tags = {

```



```

    Name = "prywatna tablica routingu"
  }
}

```

Mechanizm Route Table Associations zapewnia prawidłowe mapowanie:

- Podsieć Wydziału Planowania (private_1) → Private Route Table,
- Podsieć Wydziału Programowania (private_2) → Private Route Table,
- Podsieć Wykładowców (public_3) → Public Route Table.

```

resource "aws_route_table_association" "private_1" {
  subnet_id      = aws_subnet.private_1.id
  route_table_id = aws_route_table.private_route_table.id
}

```

```

resource "aws_route_table_association" "private_2" {
  subnet_id      = aws_subnet.private_2.id
  route_table_id = aws_route_table.private_route_table.id
}

```

```

resource "aws_route_table_association" "public_3" {
  subnet_id      = aws_subnet.public_3.id
  route_table_id = aws_route_table.public_route_table.id
}

```

4.2. Wdrożenie Bucketów S3

4.2.1. Projekt i konfiguracja Bucketów

W celu zapewnienia izolowanych przestrzeni przechowywania danych dla trzech grup użytkowników definiuje dedykowane buckety S3 dla Wydziału Programowania Kształcenia, Wydziału Planowania Kształcenia i Wykładowców. Dostęp do bucketów ustalę z wykorzystaniem polityk w następnej części projektu:

```

resource "aws_s3_bucket" "group_1" {
  bucket = "group-1-files-${random_id.suffix.hex}"
}

resource "aws_s3_bucket" "group_2" {
  bucket = "group-2-files-${random_id.suffix.hex}"
}

```

```
resource "aws_s3_bucket" "group_3" {
  bucket = "group-3-files-${random_id.suffix.hex}"
}
```

Unikalność nazw bucketów zapewniam użyciem `random_id`:

```
resource "random_id" "suffix" {
  byte_length = 4
}
```

4.2.2. Hosting statycznych stron internetowych

Konfiguruje hosting statycznych stron internetowych w oparciu o buckety celem udostępniania materiałów dla grup użytkowników poszczególnych podsieci. Każdy z elementów powiązuje z odpowiednim bucketem. Wskazuję plik główny otwierany przy wejściu na URL (`index_document`) oraz stronę błędu (`error_document`):

```
resource "aws_s3_bucket_website_configuration" "group_1_website" {
  bucket = aws_s3_bucket.group_1.bucket
  index_document { suffix = "index.html" }
  error_document { key = "error.html" }
}
```

```
resource "aws_s3_bucket_website_configuration" "group_2_website" {
  bucket = aws_s3_bucket.group_2.bucket
  index_document { suffix = "index.html" }
  error_document { key = "error.html" }
}
```

```
resource "aws_s3_bucket_website_configuration" "group_3_website" {
  bucket = aws_s3_bucket.group_3.bucket
  index_document { suffix = "index.html" }
  error_document { key = "error.html" }
}
```

4.2.3. Automatyczne przesyłanie plików

Zapewniam synchronizację lokalnych plików (`index.html` i `error.html`) w bucket z wykorzystaniem zasobu `aws_s3_object`. Każdy z zasobów powiązuje z odpowiednim bucketem. Za pomocą etag i sumy kontrolnej MD5 zapewniam autoamtyczną aktualizację zawartości bucketu:

```
resource "aws_s3_object" "group_1_index" {
  bucket = aws_s3_bucket.group_1.bucket
```

```

    key          = "index.html"
    source       = "./group_1/index.html"
    content_type = "text/html"
    etag         = filemd5("./group_1/index.html")
}

resource "aws_s3_object" "group_1_error" {
    bucket      = aws_s3_bucket.group_1.bucket
    key         = "error.html"
    source      = "./group_1/error.html"
    content_type = "text/html"
    etag        = filemd5("./group_1/error.html")
}

resource "aws_s3_object" "group_2_index" {
    bucket      = aws_s3_bucket.group_2.bucket
    key         = "index.html"
    source      = "./group_2/index.html"
    content_type = "text/html"
    etag        = filemd5("./group_2/index.html")
}

resource "aws_s3_object" "group_2_error" {
    bucket      = aws_s3_bucket.group_2.bucket
    key         = "error.html"
    source      = "./group_2/error.html"
    content_type = "text/html"
    etag        = filemd5("./group_2/error.html")
}

resource "aws_s3_object" "group_3_index" {
    bucket      = aws_s3_bucket.group_3.bucket
    key         = "index.html"
    source      = "./group_3/index.html"
    content_type = "text/html"
    etag        = filemd5("./group_3/index.html")
    depends_on = [aws_s3_bucket_policy.group_3_policy]
}

resource "aws_s3_object" "group_3_error" {
    bucket      = aws_s3_bucket.group_3.bucket
    key         = "error.html"
    source      = "./group_3/error.html"

```

```

    content_type = "text/html"
    etag         = filemd5("./group_3/error.html")
    depends_on   = [aws_s3_bucket_policy.group_3_policy]
}

```

4.3. Bezpieczeństwo

4.3.1. Grupy bezpieczeństwa - Security Groups

Dla każdej grupy użytkowników definiuję Security Group - ścisłe reguły dostępu oparte o adresację IP (192.168.1.0/24 dla Planowania, 192.168.2.0/24 dla Programowania). Dla Wykładowców otwieram dostęp publiczny:

Dla podsieci prywatnej 1 (tylko użytkownicy Wydziału Planowania Kształcenia - grupa 1):

```

resource "aws_security_group" "group_1" {
    name           = "group-1-sg"
    description    = "Dostęp tylko dla grupy 1"
    vpc_id         = aws_vpc.vpc_main.id

    ingress {
        from_port   = 0
        to_port     = 0
        protocol    = "-1"
        cidr_blocks = ["192.168.1.0/24"]
        #cidr_blocks = ["37.31.140.0/24"] #używany do testów
    }
}

```

Dla podsieci prywatnej 2 (tylko użytkownicy Wydziału Programowania Kształcenia - grupa 2):

```

resource "aws_security_group" "group_2" {
    name           = "group-2-sg"
    description    = "Dostęp tylko dla grupy 2"
    vpc_id         = aws_vpc.vpc_main.id

    ingress {
        from_port   = 0
        to_port     = 0
        protocol    = "-1"
        cidr_blocks = ["192.168.2.0/24"] # adresacja IP grupy 2 - zakres
        adresów fizycznych
    }
}

```

```
}
```

Dla podsieci publicznej:

```
resource "aws_security_group" "public" {
  name          = "public-sg"
  description    = "Dostęp dla wszystkich"
  vpc_id        = aws_vpc.vpc_main.id

  ingress {
    from_port     = 0
    to_port       = 0
    protocol      = "-1"
    cidr_blocks   = ["0.0.0.0/0"]
  }
}
```

4.3.2. Bucket Policies

Dla każdego bucketa definiuję precyzyjne reguły dostępu. Dla zasobów prywatnych łączę ograniczenia IP (sieć firmowa) i dostęp z VPC:

```
resource "aws_s3_bucket_policy" "group_1_policy" {
  bucket = aws_s3_bucket.group_1.id
  policy = jsonencode({
    Version = "2012-10-17",
    Statement = [
      # 1. Dostęp z sieci firmowej grupy 1
      {
        Effect      = "Allow",
        Principal   = "*",
        Action       = "s3:GetObject",
        Resource     = "${aws_s3_bucket.group_1.arn}/*",
        Condition = {
          IpAddress = {
            "aws:SourceIp" = ["192.168.1.0/24"]
            #"aws:SourceIp" = ["37.31.140.0/24"] #używany do
testów
          }
        }
      },
      {
        Effect      = "Allow",
        Principal   = "*",
```

```

        Action      = "s3:ListBucket",
        Resource     = aws_s3_bucket.group_1.arn,
        Condition = {
            IPAddress = {
                "aws:SourceIp" = ["192.168.1.0/24"]
                #"aws:SourceIp" = ["37.31.140.0/24"] #używany do
testów
            }
        }
    },
    {
        Sid = "Group1VPCEnter"
        Effect      = "Allow"
        Principal   = "*"
        Action      = [
            "s3:GetObject",
            "s3:ListBucket"
        ]
        Resource    = [
            aws_s3_bucket.group_1.arn,
            "${aws_s3_bucket.group_1.arn}/*"
        ]
        Condition = {
            StringEquals = {
                "aws:SourceVpc" = aws_vpc.vpc_main.id
            }
        }
    }
]
}))
}

```

```

resource "aws_s3_bucket_policy" "group_2_policy" {
    bucket = aws_s3_bucket.group_2.id
    policy = jsonencode({
        Version = "2012-10-17",
        Statement = [
            {
                Effect      = "Allow",
                Principal   = "*",
                Action      = "s3:GetObject",
                Resource    = "${aws_s3_bucket.group_2.arn}/*",
                Condition = {

```

```

        IPAddress = {
            "aws:SourceIp" = ["192.168.2.0/24"]
        }
    },
    {
        Effect      = "Allow",
        Principal   = "*",
        Action      = "s3:ListBucket",
        Resource    = aws_s3_bucket.group_2.arn,
        Condition = {
            IPAddress = {
                "aws:SourceIp" = ["192.168.2.0/24"]
            }
        }
    },
    {
        Sid = "Group2VPCEntry"
        Effect      = "Allow"
        Principal   = "*"
        Action      = [
            "s3:GetObject",
            "s3:ListBucket"
        ]
        Resource    = [
            aws_s3_bucket.group_2.arn,
            "${aws_s3_bucket.group_2.arn}/*"
        ]
        Condition = {
            StringEquals = {
                "aws:SourceVpc" = aws_vpc.vpc_main.id
            }
        }
    }
]
}))
}

```

Dla zasobu publicznego wyłączam blokadę publicznego dostępu.

```

resource "aws_s3_bucket_policy" "group_3_policy" {
    bucket = aws_s3_bucket.group_3.id
    policy = jsonencode({
        Version = "2012-10-17",

```

```

Statement = [
  {
    Effect      = "Allow",
    Principal   = "*",
    Action      = [
      "s3:GetObject",
      "s3:ListBucket"
    ],
    Resource    = [
      aws_s3_bucket.group_3.arn,
      "${aws_s3_bucket.group_3.arn}/*"
    ]
  }
]
})
depends_on = [aws_s3_bucket_public_access_block.group_3_access]
}

```

Wyłączenie blokady publicznego dostępu dla group_3

```

resource "aws_s3_bucket_public_access_block" "group_3_access" {
  bucket = aws_s3_bucket.group_3.id

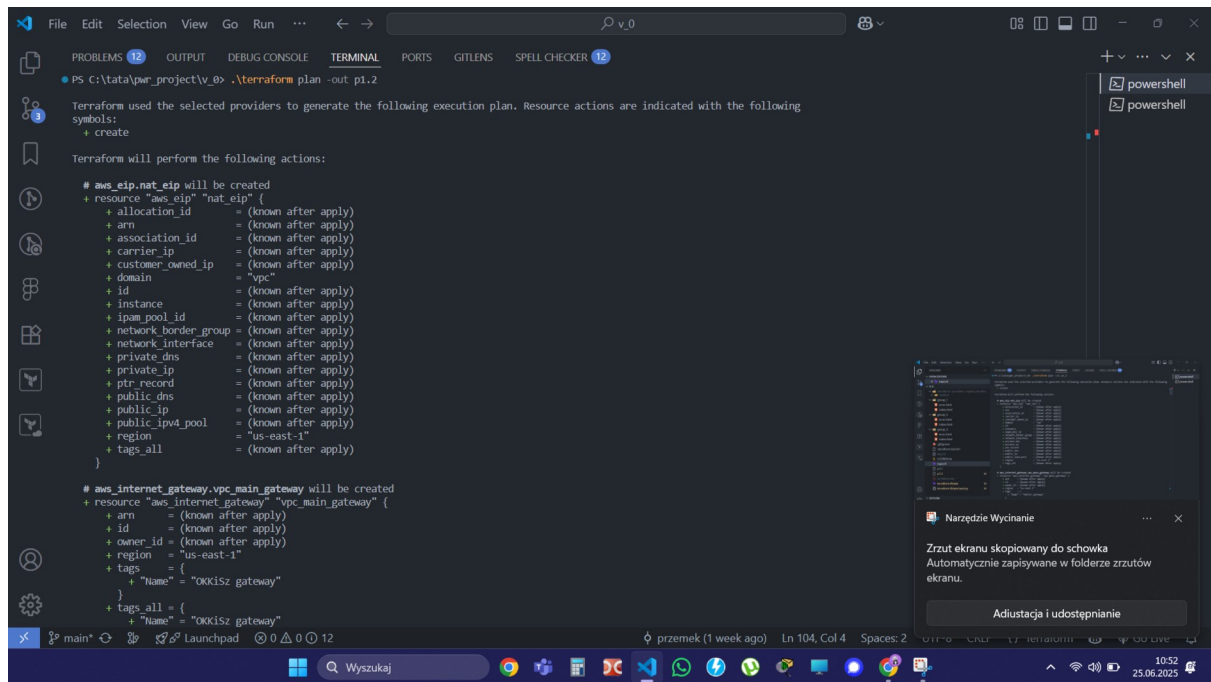
  block_public_acls       = false
  block_public_policy     = false
  ignore_public_acls     = false
  restrict_public_buckets = false
}

```

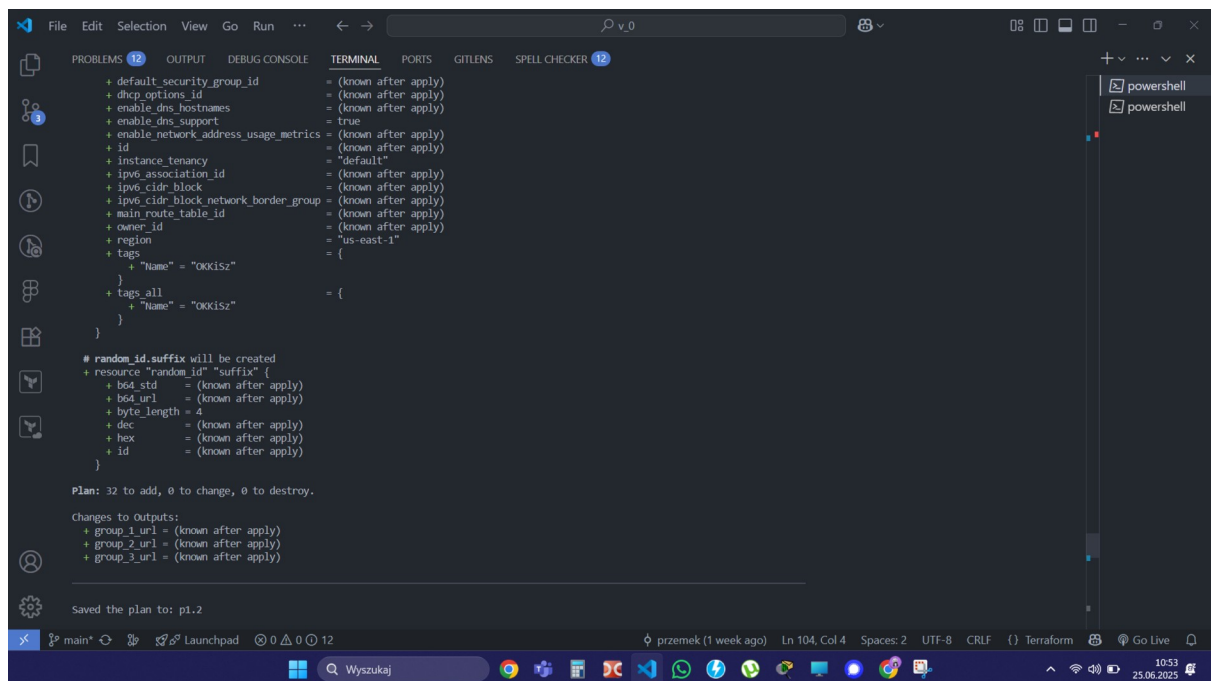

4.4. Umieszczenie projektu infrastruktury w chmurze AWS

4.4.1. Sprawdzenie kodu

Sprawdzam kod poleceniem `terraform plan -out p1.2` które pokazuje, jakie zmiany zostaną wykonane: co zostanie utworzone, zmienione lub usunięte (rys. 2.1). Polecenie to nie wprowadza jednak żadnych zmian. Dodatkowo opcja `-out p1.2` spowoduje zapisanie zmian do pliku binarnego `p1.2` (rys. 2.2). Plik ten może być później wykorzystany do zastosowania zmian, bez konieczności ponownego używania `terraform plan`.



Rys. 2.1 Zrzut ekranu – zastosowanie polecenia `terraform plan -out p1.2`

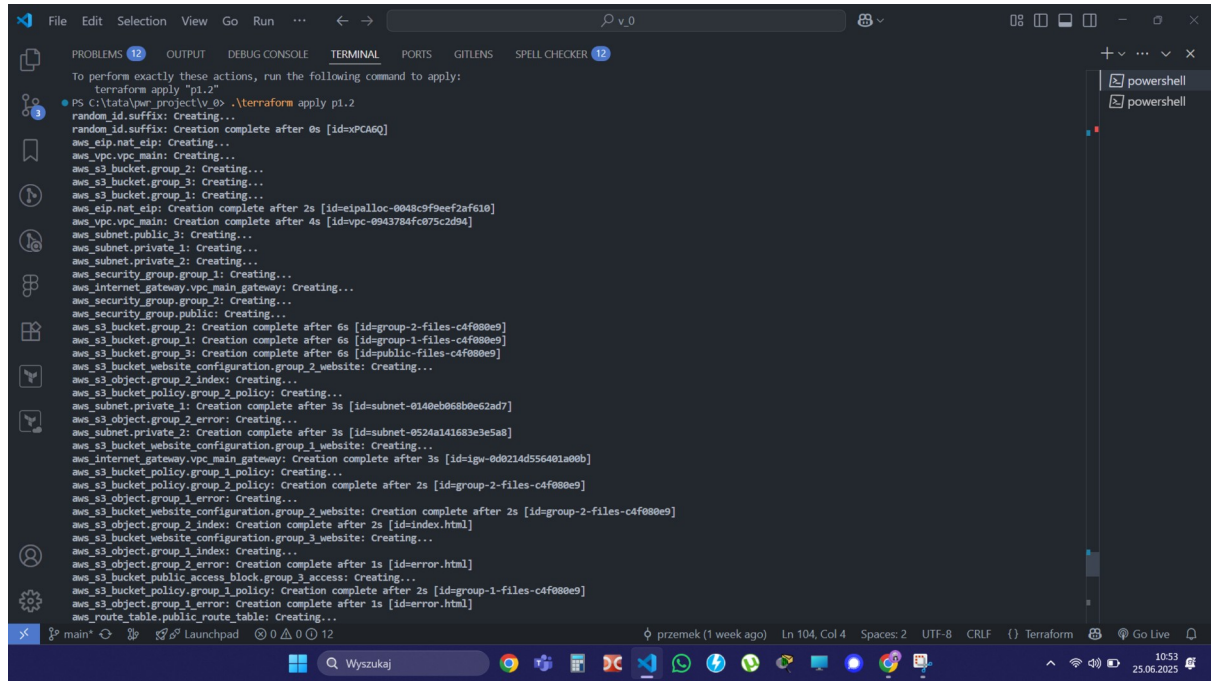


Rys. 2.2 Zrzut ekranu – zapisanie konfiguracji do pliku `p1.2`

Z powyższych zrzutów ekranu wynika, że do chmury AWS zostaną dodane 32 elementy infrastruktury. Zmiany zostały zapisane w pliku p1.2.

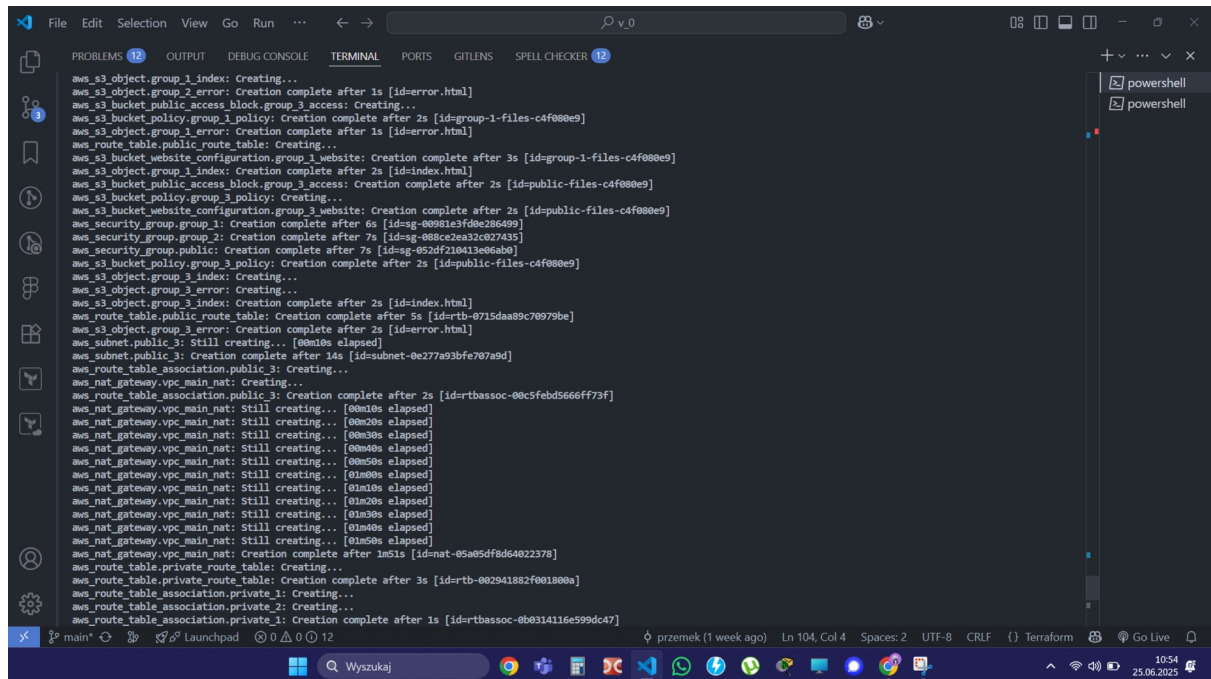
4.4.2. Uruchomienie kodu

Uruchamiam kod poleceniem `terraform apply p1.2` (rys. 3.1). Spowoduje przesłanie to projektu infrastruktury zapisanej w pliku p1.2 do chmury AWS.



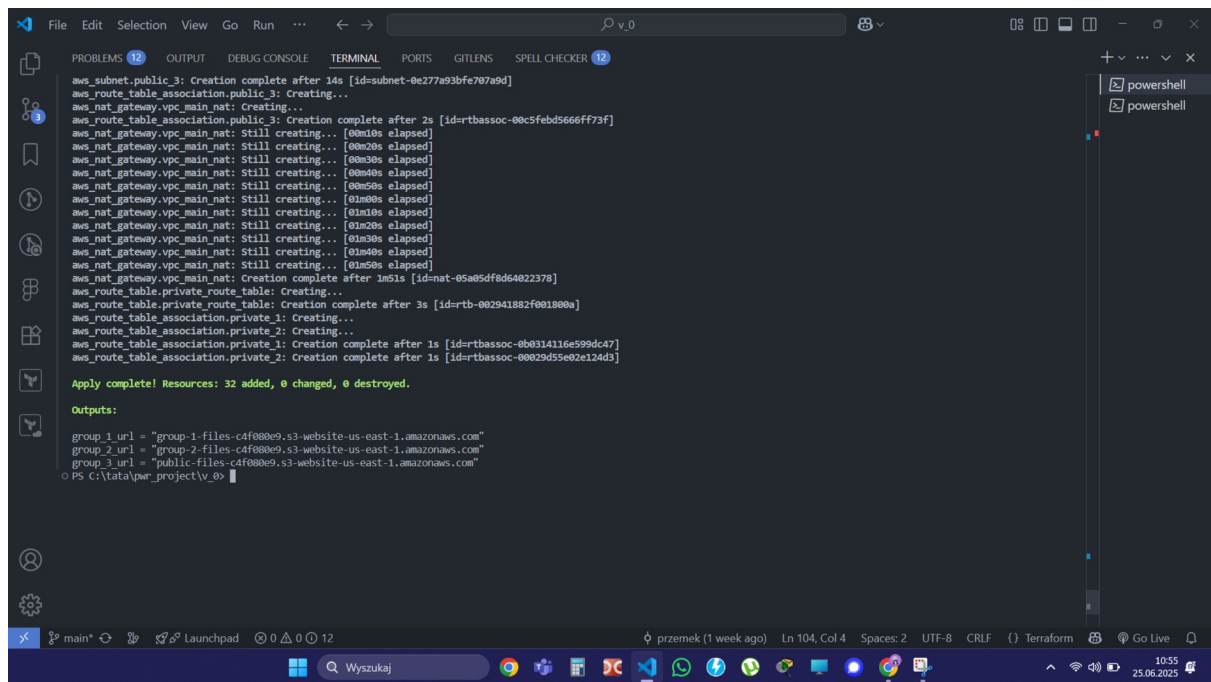
```
To perform exactly these actions, run the following command to apply:
terraform apply "p1.2"
PS C:\ata\aws-project\4.0> .\terraform apply p1.2
random_id.suffix: Creating...
random_id.suffix: Creation complete after 0s [id=xPCA6Q]
aws_eip.nat_eip: Creating...
aws_vpc.vpc_main: Creating...
aws_s3_bucket.group_2: Creating...
aws_s3_bucket.group_3: Creating...
aws_s3_bucket.group_1: Creating...
aws_eip.nat_eip: Creation complete after 2s [id=eipalloc-0048c9f9ee2af610]
aws_vpc.vpc_main: Creation complete after 4s [id=vpc-0943784fc075c2d94]
aws_subnet.public_3: Creating...
aws_subnet.private_1: Creating...
aws_subnet.private_2: Creating...
aws_security_group.group_1: Creating...
aws_internet_gateway.vpc_main_gateway: Creating...
aws_security_group.group_2: Creating...
aws_security_group.public: Creating...
aws_s3_bucket.group_2: Creation complete after 6s [id=group-2-files-c4f080e9]
aws_s3_bucket.group_1: Creation complete after 6s [id=group-1-files-c4f080e9]
aws_s3_bucket.group_3: Creation complete after 6s [id=public-files-c4f080e9]
aws_s3_bucket_website_configuration.group_2_website: Creating...
aws_s3_object.group_2_index: Creating...
aws_s3_bucket_policy.group_2_policy: Creating...
aws_subnet.private_1: Creation complete after 3s [id=subnet-0140eb068b0e62ad7]
aws_s3_object.group_2_error: Creating...
aws_subnet.private_2: Creation complete after 3s [id=subnet-0524a141683e3e5a8]
aws_s3_bucket_website_configuration.group_1_website: Creating...
aws_internet_gateway.vpc_main_gateway: Creation complete after 3s [id=igw-0d0214d556401a00b]
aws_s3_bucket_policy.group_1_policy: Creation complete after 2s [id=group-2-files-c4f080e9]
aws_s3_object.group_1_error: Creating...
aws_s3_bucket_website_configuration.group_2_website: Creation complete after 2s [id=group-2-files-c4f080e9]
aws_s3_object.group_2_index: Creation complete after 2s [id=index.html]
aws_s3_bucket_website_configuration.group_3_website: Creating...
aws_s3_object.group_1_index: Creating...
aws_s3_object.group_2_error: Creation complete after 1s [id=error.html]
aws_s3_bucket_public_access_block.group_3_access: Creating...
aws_s3_bucket_policy.group_1_policy: Creation complete after 2s [id=group-1-files-c4f080e9]
aws_s3_object.group_1_error: Creation complete after 2s [id=error.html]
aws_route_table.public_route_table: Creating...
```

Rys. 3.1 Zrzut ekranu – uruchomienie kodu poleceniem `terraform apply p1.2`



```
aws_s3_object.group_1_index: Creating...
aws_s3_object.group_2_error: Creation complete after 1s [id=error.html]
aws_s3_bucket_public_access_block.group_3_access: Creating...
aws_s3_bucket_policy.group_1_policy: Creation complete after 2s [id=group-1-files-c4f080e9]
aws_s3_object.group_1_error: Creation complete after 1s [id=error.html]
aws_route_table.public_route_table: Creating...
aws_s3_bucket_website_configuration.group_1_website: Creation complete after 3s [id=group-1-files-c4f080e9]
aws_s3_object.group_1_index: Creation complete after 2s [id=index.html]
aws_s3_bucket_public_access_block.group_3_access: Creation complete after 2s [id=public-files-c4f080e9]
aws_s3_bucket_policy.group_3_policy: Creating...
aws_s3_bucket_website_configuration.group_3_website: Creation complete after 2s [id=public-files-c4f080e9]
aws_security_group.group_1: Creation complete after 6s [id=sg-00881e3f0be286490]
aws_security_group.group_2: Creation complete after 7s [id=sg-008ce2ea32c027435]
aws_security_group.public: Creation complete after 7s [id=sg-052df210413e06ab0]
aws_s3_bucket_policy.group_3_policy: Creation complete after 2s [id=public-files-c4f080e9]
aws_s3_object.group_3_index: Creating...
aws_s3_object.group_3_error: Creating...
aws_s3_object.group_3_index: Creation complete after 2s [id=index.html]
aws_route_table.public_route_table: Creation complete after 5s [id=rtb-0715daa89c70979be]
aws_s3_object.group_3_error: Creation complete after 2s [id=error.html]
aws_subnet.public_3: Still creating... [00m10s elapsed]
aws_subnet.public_3: Creation complete after 14s [id=subnet-0e277a93bfe707a9d]
aws_route_table_association.public_3: Creating...
aws_nat_gateway.vpc_main_nat: Creating...
aws_route_table_association.public_3: Creation complete after 2s [id=rtbassoc-00c5febd566ff73f]
aws_nat_gateway.vpc_main_nat: Still creating... [00m10s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [00m20s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [00m30s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [00m40s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [00m50s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [01m00s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [01m10s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [01m20s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [01m30s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [01m40s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [01m50s elapsed]
aws_nat_gateway.vpc_main_nat: Creation complete after 1m51s [id=nat-05a05df8d64022378]
aws_route_table.private_route_table: Creating...
aws_route_table.private_route_table: Creation complete after 3s [id=rtb-002941882f001800a]
aws_route_table_association.private_1: Creating...
aws_route_table_association.private_2: Creating...
aws_route_table_association.private_1: Creation complete after 1s [id=rtbassoc-0b931411e6599dc47]
```

Rys. 3.2 Zrzut ekranu – przebieg polecenia `terraform apply p1.2`



```
aws_subnet.public_3: Creation complete after 14s [id=subnet-0e277a93bfe707a9d]
aws_route_table.association.public_3: Creating...
aws_nat_gateway.vpc_main_nat: Creating...
aws_route_table.association.public_3: Creation complete after 2s [id=rtbassoc-00c5febd566ff73f]
aws_nat_gateway.vpc_main_nat: Still creating... [60m10s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [60m20s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [60m30s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [60m40s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [60m50s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [61m00s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [61m10s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [61m20s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [61m30s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [61m40s elapsed]
aws_nat_gateway.vpc_main_nat: Still creating... [61m50s elapsed]
aws_nat_gateway.vpc_main_nat: Creation complete after 1m51s [id=nat-05a05df8d64022378]
aws_route_table.private_route_table: Creating...
aws_route_table.private_route_table: Creation complete after 3s [id=rtb-002941882f001800a]
aws_route_table.association.private_1: Creating...
aws_route_table.association.private_2: Creating...
aws_route_table.association.private_1: Creation complete after 1s [id=rtbassoc-0b0314116e599dc47]
aws_route_table.association.private_2: Creation complete after 1s [id=rtbassoc-00029d55e02e124d3]

Apply complete! Resources: 32 added, 0 changed, 0 destroyed.

Outputs:
group_1_url = "group-1-files-c4f080e9.s3-website-us-east-1.amazonaws.com"
group_2_url = "group-2-files-c4f080e9.s3-website-us-east-1.amazonaws.com"
group_3_url = "public-files-c4f080e9.s3-website-us-east-1.amazonaws.com"
PS C:\data\pwr_project\p0>
```

Rys. 3.3 Zrzut ekranu – zakończenie polecenia terraform apply p1.2

Z powyższych zrzutów ekranu (rys. 3.2 i 3.3) wynika, że do chmury AWS zostały dodane 32 elementy infrastruktury. Dodatkowo wygenerowane zostały adresy internetowe:

- **group-1-files-c4f080e9.s3-website-us-east-1.amazonaws.com** – prywatny hosting plików Wydziału Planowania Kształcenia
- **group-2-files-c4f080e9.s3-website-us-east-1.amazonaws.com** – prywatny hosting plików Wydziału Programowania Kształcenia
- **public-files-c4f080e9.s3-website-us-east-1.amazonaws.com** – publiczny hosting plików wykładowców

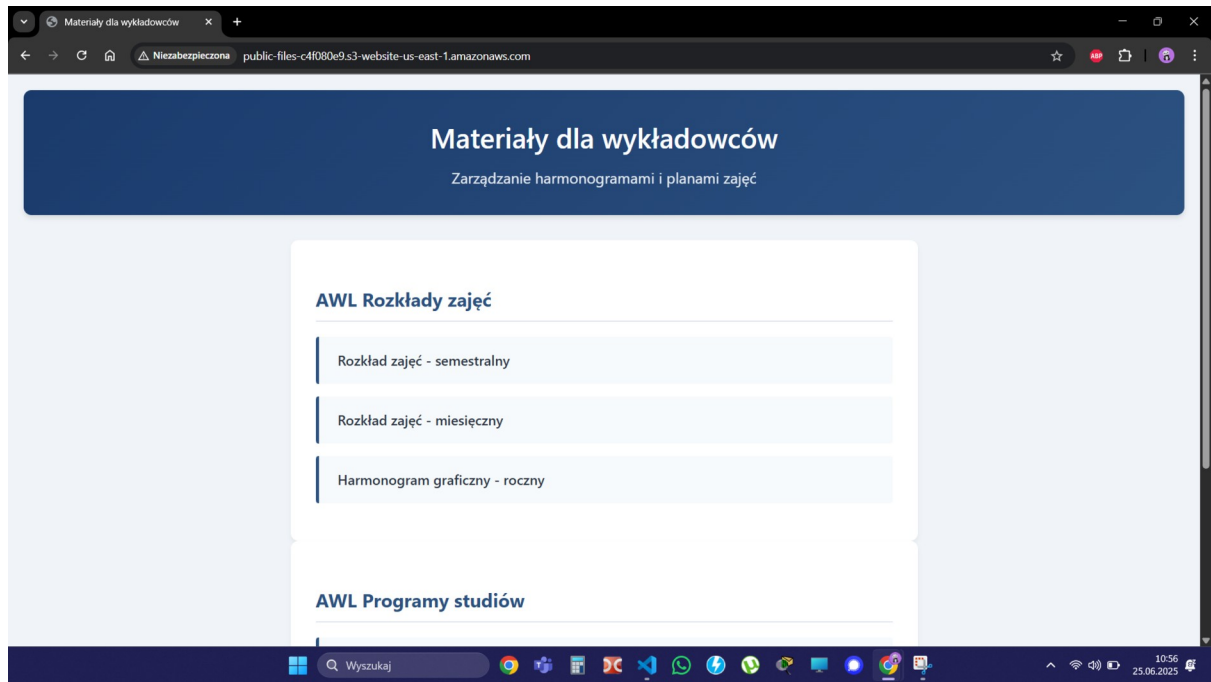
5. Test infrastruktury sieciowej oraz zasad dostępu

5.1. Test dostępu publicznego zewnętrznego

5.1.1. Dostęp do hostingu publicznego z dowolnego adresu IP

W pierwszej kolejności sprawdzam możliwość dostępu do zasobu publicznego z sieci zewnętrznej wpisując w pasku adresu przeglądarki adres **public-files-c4f080e9.s3-website-us-east-1.amazonaws.com**.

Zgodnie z oczekiwaniem zwrócona została strona internetowa w oparciu o protokół http (rys. 4).

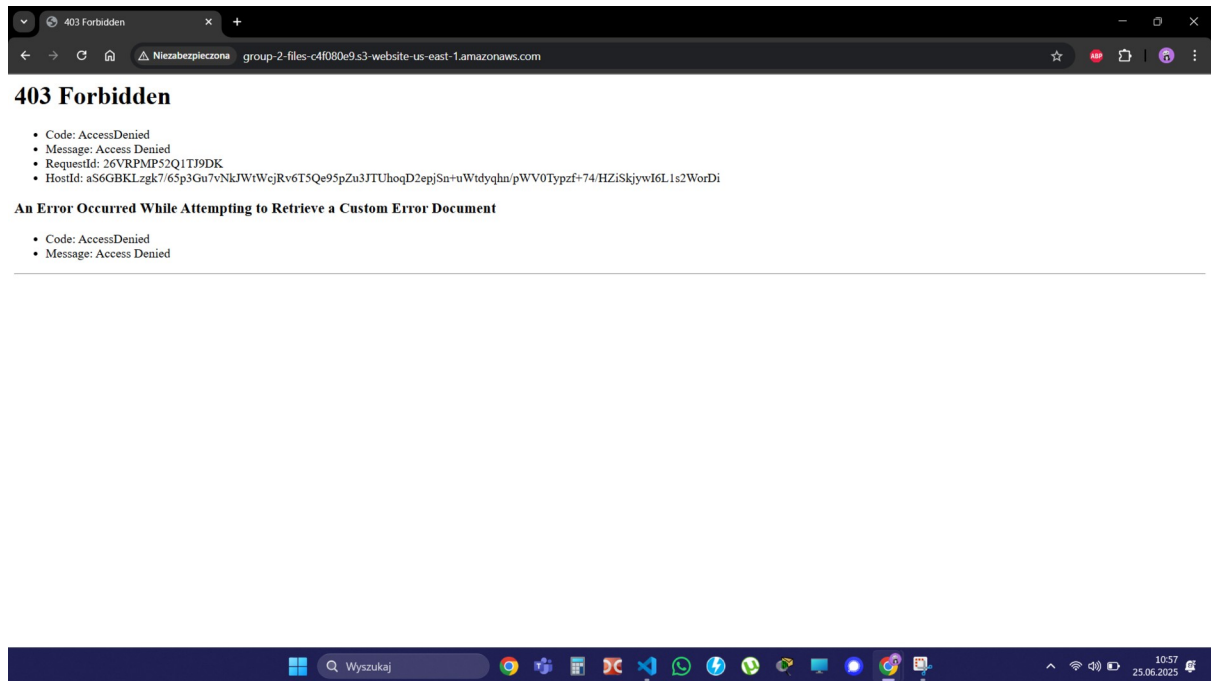


Rys. 4 Zrzut ekranu – widok witryny internetowej zasobu publicznego group_3

5.1.2. Dostęp do hostingu prywatnego z niedozwolonego adresu IP

Sprawdzam możliwość dostępu do zasobu prywatnego Wydziału Programowania Kształcenia z sieci zewnętrznej wpisując w pasku adresu przeglądarki adres **group-2-files-c4f080e9.s3-website-us-east-1.amazonaws.com**.

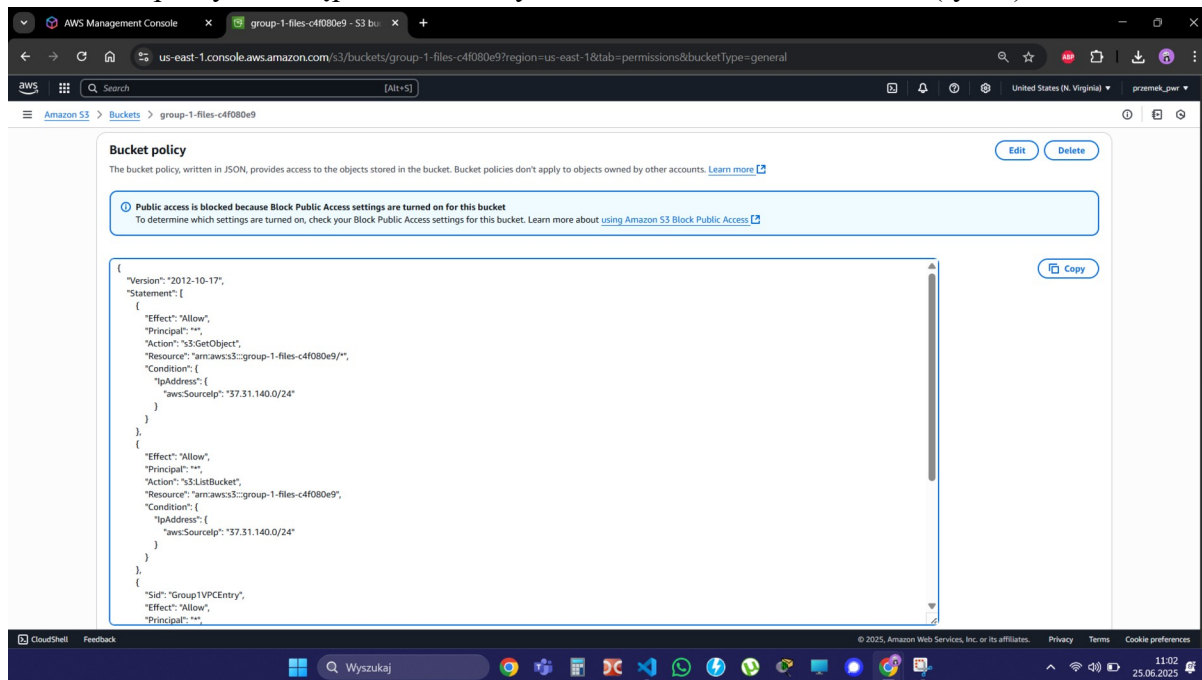
Zwrócony zostaje komunikat z kodem 403 informujący o braku dostępu do żadanego zasobu (rys. 5). Jest to pożądaný rezultat.



Rys. 5 Zrzut ekranu – widok komunikatu 403

5.1.3. Dostęp do hostingu prywatnego z dozwolonego adresu IP

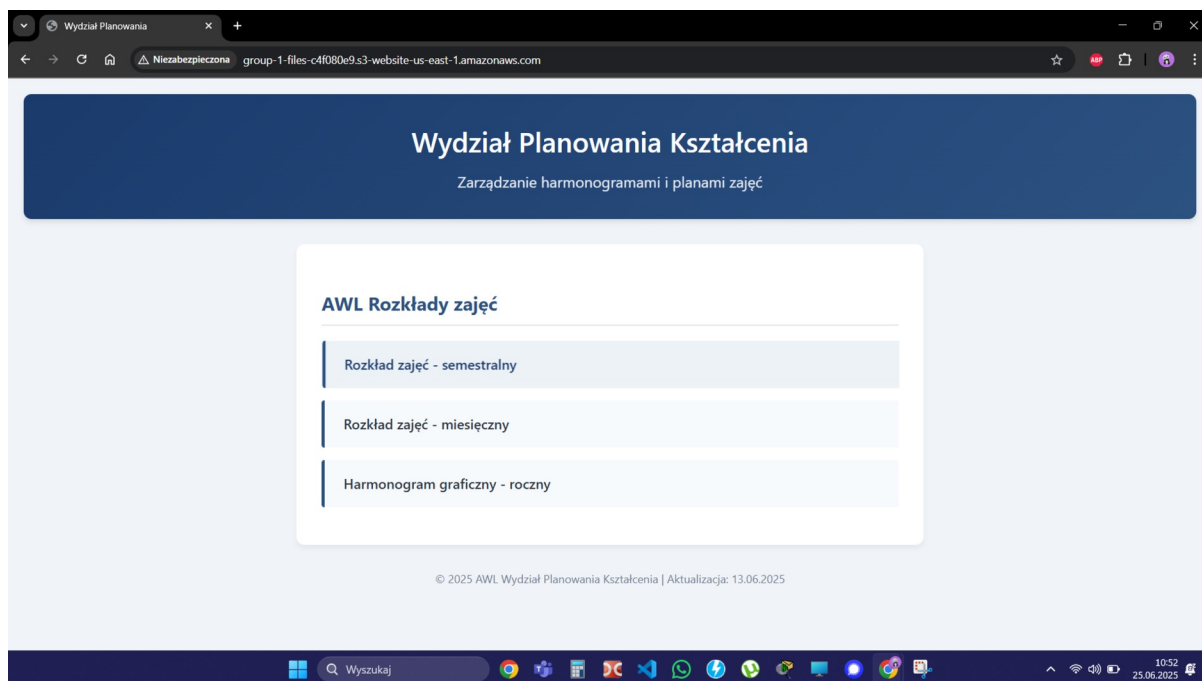
Sprawdzam możliwość dostępu do zasobu prywatnego Wydziału Planowania Kształcenia z dozwolonego adresu IP. W celu wykonania testu dostępu zdalnego użyłem publicznego adresu IP 37.31.140.7. Na potrzeby testu umieściłem odpowiedni zakres adresów w polityce dostępu Bucketa Wydziału Planowania Kształcenia (rys. 6).



Rys. 6 Zrzut ekranu – widok Bucket policy w konsoli administratora AWS

Następnie w pasku adresu przeglądarki wpisałem adres **group-1-files-c4f080e9.s3-website-us-east-1.amazonaws.com**.

Zgodnie z oczekiwaniem zwrócona została strona Wydziału Planowania Kształcenia w oparciu o protokół http (rys. 7).

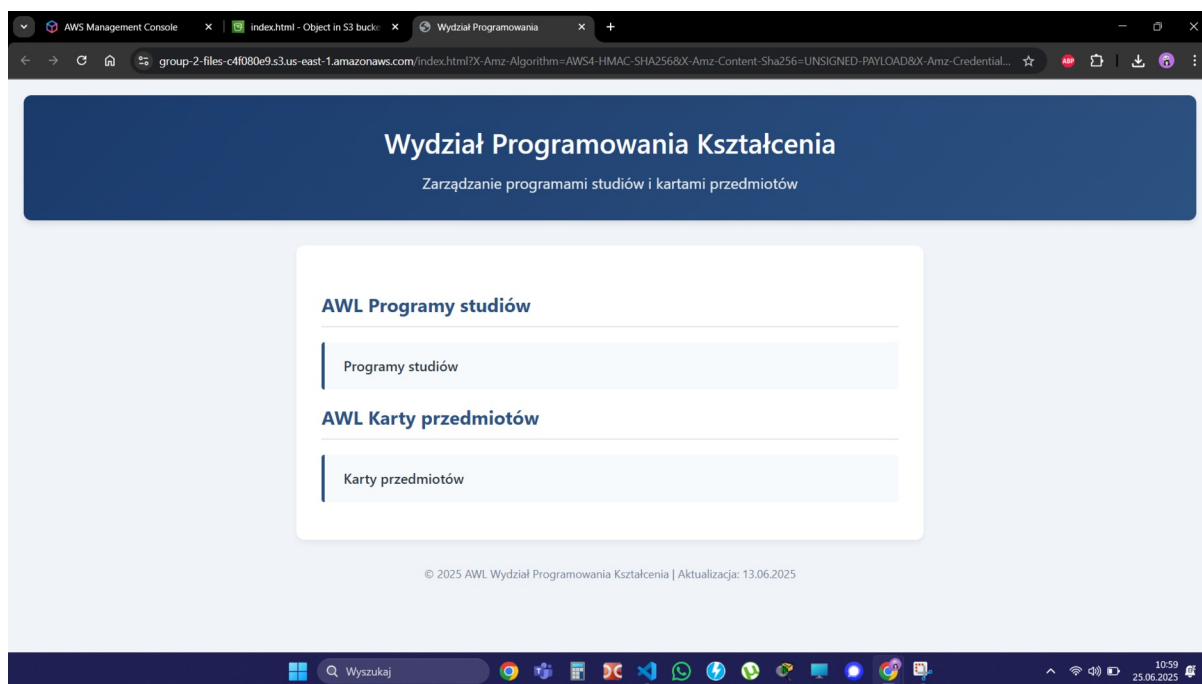


Rys. 7 Zrzut ekranu – widok witryny internetowej zasobu prywatnego group_1

5.2. Test dostępu wewnętrznego

5.2.1. Dostęp do hostingu prywatnego

Sprawdzam możliwość dostępu do zasobu prywatnego Wydziału Programowania Kształcenia z konsoli administratora AWS. W tym celu loguję się na swoje indywidualne konto AWS, nawiguję do zasobu S3 bucket, otwieram zasób group-2-files- c4f080e9 i otwieram plik index.html. W rezultacie przeglądarka zwraca stronę Wydziału Programowania Kształcenia w oparciu o protokół https (rys. 8).



Rys. 8 Zrzut ekranu – widok witryny internetowej zasobu prywatnego group_2

Podsumowanie

Projekt zrealizował wszystkie założenia celowe i zakresowe, tworząc specjalistyczną infrastrukturę AWS dla potrzeb uczelni wojskowej.

W odniesieniu do celów:

Stworzono trzy warstwy sieciowe z pełną izolacją ruchu. Wdrożono wielowarstwowy model ochrony danych łączący izolację sieciową (VPC), kontrolę dostępu opartą na adresacji IP (Security Groups) oraz polityki przestrzeni przechowywania danych (Bucket Policies). Architektura pozwala na dodawanie kolejnych komórek organizacyjnych bez modyfikacji podstawowej infrastruktury. Wdrożono automatyczny hosting dokumentów z kontrolą wersji.

Bibliografia

1. <https://registry.terraform.io/providers/hashicorp/aws/latest/docs>