Politechnika Śląska

PROJEKT INŻYNIERSKI

„Automatyczna konfiguracja środowiska dla aplikacji webowej”

Dominik Henzel
Nr albumu 300810

Kierunek: Teleinformatyka

PROWADZĄCY PRACĘ
dr inż. Marcin Kucharczyk
KATEDRA Telekomunikacji i Teleinformatyki
Wydział Automatyki, Elektroniki i Informatyki

GLIWICE Rok
2025

Tytuł pracy:

Automatyczna konfiguracja środowiska dla aplikacji webowej

Streszczenie:

Celem projektu jest opracowanie rozwiązania umożliwiającego automatyczną konfigurację środowiska dla aplikacji webowych, co przyspiesza i usprawnia proces przygotowywania infrastruktury IT. Realizacja projektu obejmuje stworzenie skryptów automatyzujących tworzenie maszyn wirtualnych, instalację niezbędnych pakietów oraz konfigurację kluczowych elementów infrastruktury, takich jak serwery baz danych, klucze dostępowe, firewalle czy load balancer.

Efektem końcowym projektu jest w pełni zautomatyzowane środowisko gotowe do wdrożenia aplikacji webowych, co minimalizuje nakład pracy administracyjnej, redukuje błędy konfiguracyjne oraz umożliwia szybkie skalowanie infrastruktury. Rozwiązanie to ma zastosowanie w różnych scenariuszach, takich jak budowanie środowisk testowych, stagingowych i produkcyjnych.

Słowa kluczowe:

Automatyzacja infrastruktury, Terraform, konfiguracja środowiska, maszyny wirtualne, maszyny chmurowe

Thesis title:

Automatic Environment Configuration for Web Applications

Abstract:

The aim of this project is to develop a solution for automatic environment configuration for web applications, which accelerates and improves the process of preparing IT infrastructure. The project involves creating scripts that automate the provisioning of virtual machines, installing necessary software packages, and configuring key infrastructure components such as database servers, access keys, firewalls, and load balancers.

The final outcome of the project is a fully automated environment ready for deploying web applications. This significantly reduces administrative workload, minimizes configuration errors, and enables rapid infrastructure scaling. The solution is applicable in various scenarios, including building test, staging, and production environments.

Keywords:

Infrastructure automation, Terraform, environment configuration, virtual machines, cloud machines

Spis treści

Rozdział 1

Wstęp

Współczesny rozwój technologii informacyjnych, połączony z rosnącym znaczeniem aplikacji webowych, stawia przed organizacjami wyzwania związane z zarządzaniem infrastrukturą IT. Zapewnienie stabilności, skalowalności i wydajności usług wymaga stosowania efektywnych metod konfiguracji i zarządzania zasobami. Tradycyjne, ręczne podejście do konfiguracji środowisk IT, mimo że wciąż stosowane, często okazuje się niewystarczające. Proces ten bywa czasochłonny, podatny na błędy i może prowadzić do problemów związanych z bezpieczeństwem oraz nieprzewidzianymi przestojami.

W odpowiedzi na te wyzwania coraz większą popularność zyskuje automatyzacja infrastruktury IT. Rozwiązania oparte na koncepcji Infrastructure as Code (IaC) pozwalają na definiowanie infrastruktury w sposób deklaratywny, za pomocą kodu. Narzędzia takie jak Terraform i Vagrant umożliwiają uproszczenie procesów wdrożeniowych, eliminację błędów konfiguracyjnych oraz lepsze wykorzystanie dostępnych zasobów. Dzięki nim zespoły deweloperskie i administratorzy IT mogą przyspieszyć proces tworzenia i utrzymywania środowisk, zapewniając jednocześnie powtarzalność oraz przejrzystość.

Celem niniejszej pracy jest zaprojektowanie oraz implementacja rozwiązania automatyzującego konfigurację środowiska IT dla aplikacji webowych. Przedstawione rozwiązanie składa się z zestawu skryptów, które wspierają użytkowników w szybkim tworzeniu gotowych do użycia środowisk. Automatyzacja ta umożliwia redukcję czasu i wysiłku wymaganego do konfiguracji oraz zapewnia elastyczność, pozwalając na dostosowanie środowisk do konkretnych potrzeb.

Praca skupia się na porównaniu dwóch podejść do automatyzacji infrastruktury. Pierwsze podejście dotyczy lokalnej konfiguracji przy pomocy narzędzia Vagrant, które znajduje zastosowanie w środowiskach deweloperskich i testowych. Drugie natomiast odnosi się do chmurowej automatyzacji zarządzanej za pomocą Terraformu, w kontekście

wdrażania w Amazon Web Services (AWS). Oba podejścia zostaną przeanalizowane i porównane w ramach praktycznego wdrożenia systemu Moodle – popularnego narzędzia do zarządzania nauczaniem, które wymaga odpowiedniej infrastruktury obejmującej serwery aplikacji, bazy danych oraz połączenia sieciowe.

Rozdział 2

Analiza tematu

Problematyka automatyzacji konfiguracji jest osadzona w szerszym kontekście rozwoju narzędzi wspierających DevOps. Zasadniczym celem tych narzędzi jest integracja procesów deweloperskich i operacyjnych, co prowadzi do skrócenia cyklu wdrożeniowego oraz redukcji liczby błędów.

Automatyzacja konfiguracji środowisk IT w środowisku lokalnym jest jednym z fundamentów efektywnego wdrażania aplikacji webowych, szczególnie w fazie rozwoju i testowania. W tym kontekście narzędzie Vagrant odgrywa istotną rolę, umożliwiając szybkie tworzenie replikowalnych środowisk. Jego działanie opiera się na integracji z wirtualizatorami, takimi jak VirtualBox, VMware czy Hyper-V, co pozwala na definiowanie środowiska w plikach konfiguracyjnych i automatyczne uruchamianie maszyn wirtualnych. Dzięki temu deweloperzy mogą uniknąć problemów wynikających z różnic w lokalnych konfiguracjach, co jest szczególnie istotne w zespołach pracujących na różnych systemach operacyjnych. Jednak Vagrant to nie jedyne rozwiązanie dostępne w tej dziedzinie. Innym popularnym narzędziem, które znajduje zastosowanie w lokalnej konfiguracji środowisk, jest Docker. W odróżnieniu od Vagranta, Docker opiera się na konteneryzacji zamiast wirtualizacji. Dzięki tej technologii aplikacje mogą być uruchamiane w izolowanych środowiskach zawierających wszystkie wymagane zależności. W kontekście tego projektu Vagrant okazuje się lepszym rozwiązaniem przede wszystkim ze względu na specyfikę środowiska i wymagania aplikacji Moodle, które wymaga pełnego systemu operacyjnego do uruchomienia wszystkich swoich komponentów, takich jak baza danych, serwer aplikacji czy inne zależności. Dzięki wykorzystaniu maszyn wirtualnych, Vagrant pozwala na precyzyjne odwzorowanie rzeczywistego środowiska produkcyjnego w warunkach lokalnych. Możliwość konfiguracji systemu operacyjnego i zasobów sprzętowych, takich jak procesor czy pamięć, sprawia, że środowisko lokalne jest bardziej zbliżone do faktycznych warunków, w których system będzie działał w produkcji. Dodatkowym argumentem przemawiającym za Vagrantem jest jego zdolność do zapewnienia pełnej izolacji środowiska od systemu hosta. To eliminuje ryzyko konfliktów między zależnościami aplikacji a konfiguracją lokalnego komputera dewelopera, co jest istotne w zespołach, gdzie różne osoby mogą korzystać z różnych systemów operacyjnych. Ponadto, z uwagi na to, że

Moodle nie jest domyślnie zaprojektowane z myślą o środowiskach kontenerowych, takich jak Docker, Vagrant staje się bardziej naturalnym wyborem. Umożliwia pełną kontrolę nad konfiguracją i instalacją wymaganych usług, co ułatwia dostosowanie środowiska do specyficznych wymagań tej aplikacji.

W przypadku konfiguracji środowisk chmurowych Terraform wyróżnia się jako narzędzie umożliwiające zarządzanie infrastrukturą w sposób deklaratywny. Jego wsparcie dla wielu dostawców chmurowych, takich jak AWS, Google Cloud Platform czy Microsoft Azure, pozwala na tworzenie kompleksowych środowisk w różnych chmurach. Terraform, wykorzystując podejście "Infrastructure as Code" (IaC), umożliwia opisanie infrastruktury w plikach konfiguracyjnych, a następnie jej automatyczne wdrożenie. Alternatywą w tej kategorii jest Ansible, które, choć działa w sposób proceduralny, również umożliwia automatyzację konfiguracji i zarządzanie zasobami w chmurze. W przeciwieństwie do Terraformu, Ansible nie przechowuje stanu infrastruktury, co oznacza, że każde uruchomienie playbooka wykonuje zdefiniowane kroki od początku, niezależnie od stanu bieżącego infrastruktury. W kontekście tego projektu Terraform przewyższa Ansible przede wszystkim dzięki deklaratywnemu podejściu do definiowania infrastruktury i możliwości automatycznego zarządzania całym cyklem życia zasobów w chmurze. Dzięki plikom konfiguracyjnym Terraform precyzyjnie określa pożądany stan infrastruktury, a jego mechanizm planowania pozwala na przewidywanie zmian przed wdrożeniem. Jest to kluczowe przy tworzeniu środowiska Moodle na AWS, które wymaga spójnego zarządzania zasobami, takimi jak instancje EC2, security groups, load balancer czy sieci VPC. W odróżnieniu od Terraformu, Ansible działa proceduralnie, co oznacza większą złożoność przy tworzeniu całych środowisk od podstaw. Choć Ansible sprawdza się w zarządzaniu konfiguracją istniejących systemów, w tym projekcie jego brak natywnego zarządzania cyklem życia zasobów i ograniczone wsparcie dla wieloplatformowej infrastruktury sprawia, że wymaga większego nakładu pracy i zwiększa ryzyko błędów. Terraform, z wbudowaną integracją z AWS i możliwością pełnego wykorzystania jego funkcji, takich jak autoskalowanie, jest bardziej efektywnym i spójnym narzędziem do automatyzacji w środowiskach chmurowych.

Literatura naukowa oraz dokumentacja narzędzi wskazują, że oba podejścia mają swoje zalety i ograniczenia, które zależą od skali, charakteru projektu oraz kosztów operacyjnych. Vagrant znajduje zastosowanie głównie w mniejszych projektach oraz środowiskach testowych, gdzie istotna jest spójność konfiguracji i szybki dostęp do zasobów. Koszty korzystania z Vagranta wynikają głównie z potrzeby posiadania odpowiedniego sprzętu lokalnego, który jest w stanie obsłużyć wymagania wirtualizacji. Chociaż jest to wydatek

jednorazowy, może stanowić barierę w przypadku konieczności skalowania środowiska lub pracy w zespołach z ograniczonymi zasobami sprzętowymi. Terraform, z drugiej strony, jest preferowanym narzędziem w dużych projektach produkcyjnych oraz w środowiskach korporacyjnych, gdzie zarządzanie infrastrukturą wymaga zaawansowanej automatyzacji i integracji z chmurami publicznymi, takimi jak AWS. W tym przypadku koszty wynikają z opłat za korzystanie z zasobów chmurowych, które są naliczane za każdą godzinę użytkowania. W krótkim okresie takie podejście może być bardziej elastyczne i ekonomiczne, ale w dłuższym czasie, szczególnie przy ciągłym wykorzystaniu zasobów, opłaty te mogą znacząco wzrosnąć.

Analiza istniejących rozwiązań pokazuje, że automatyzacja infrastruktury stanowi fundament nowoczesnego zarządzania IT. W ramach pracy szczególna uwaga zostanie poświęcona praktycznemu zastosowaniu Vagranta i Terraformu do wdrożenia systemu Moodle. Moodle, jako platforma edukacyjna, wymaga złożonej infrastruktury obejmującej serwery aplikacji, bazy danych oraz połączenia sieciowe. Analiza funkcjonalności obu narzędzi w kontekście tego konkretnego systemu pozwoli na ocenę ich przydatności oraz identyfikację potencjalnych obszarów usprawnień.

Rozdział 3

Specyfikacja zewnętrzna

Zaprojektowane rozwiązanie automatyzujące konfigurację środowiska IT zostało opracowane z myślą o jego praktycznym zastosowaniu i łatwości wdrożenia. Wymaga ono minimalnej interwencji użytkownika przy instalacji i aktywacji, jednocześnie umożliwiając szerokie możliwości dostosowania. Wdrożenie automatyzacji infrastruktury dla systemu Moodle wymaga precyzyjnego określenia specyfikacji zewnętrznej, obejmującej wymagania sprzętowe i programowe, proces instalacji, aktywacji oraz kwestie związane z obsługą i bezpieczeństwem systemu. Niniejszy rozdział szczegółowo opisuje te aspekty, stanowiąc wytyczne dla użytkowników oraz administratorów infrastruktury.

Wymagania sprzętowe i programowe

Aby uruchomić środowisko skonfigurowane przy pomocy Vagrant :

- Procesor z obsługą wirtualizacji sprzętowej
- Co najmniej 10 GB wolnego miejsca na dysku dla VirtualBox oraz dodatkowa przestrzeń dla obrazów maszyn wirtualnych
- Minimalnie 4 GB RAM. Większa ilość pamięci operacyjnej jest wymagana przy uruchamianiu większej liczby maszyn wirtualnych

Aby uruchomić środowisko skonfigurowane przy pomocy Terraform :

- Każdy współczesny procesor wielordzeniowy
- Przestrzeń dyskowa na poziomie 1 GB jest wystarczająca na instalację Terraform oraz przechowywanie plików skryptowych.
- Minimalne wymaganie to 512 MB RAM
- aktywne konto na jednej z obsługiwanych platform chmurowych, takich jak Amazon Web Services (AWS)

Proces instalacji narzędzi

Aby rozpocząć pracę z Vagrantem w środowisku lokalnym, konieczne jest wcześniejsze przygotowanie odpowiedniego zaplecza w postaci platformy wirtualizacyjnej, takiej jak

VirtualBox. Po jej zainstalowaniu należy pobrać i zainstalować samo narzędzie Vagrant według dokumentacji. W przypadku konfiguracji chmurowej, Terraform wymaga pobrania wersji narzędzia odpowiedniej dla danego systemu operacyjnego bezpośrednio z oficjalnej strony producenta, a następnie powiązanie terminala z odpowiednim dostawca usług chmurowych, w tym przypadku aby powiązać terminal z AWS należy użyć komendy *aws configure* a następnie wprowadzić unikalny identyfikator dostępu, który można uzyskać z konsoli AWS w sekcji „Użytkownicy IAM", Secret Access Key – hasło przypisane do Access Key ID, Region – domyślny region chmurowy, Output format – opcjonalny format wyjściowy najpopularniejszy wybór to json.

Konfiguracja Moodle oraz bazy danych

Konfiguracja bazy danych i systemu Moodle została zaprojektowana tak, aby była spójna w obu podstawowych podejściach – lokalnym i chmurowym. Cały proces opiera się na wykorzystaniu zmiennych zdefiniowanych w oddzielnym pliku, co umożliwia łatwe dostosowanie parametrów, takich jak nazwa użytkownika, hasło czy adres IP, do specyficznych wymagań projektu. Dzięki temu rozwiązanie jest elastyczne i pozwala na wielokrotne wykorzystanie tej samej konfiguracji w różnych środowiskach bez potrzeby modyfikowania kodu. Taki układ przyspiesza proces wdrażania i minimalizuje ryzyko błędów, czyniąc system bardziej uniwersalnym i łatwym w utrzymaniu.

Dla uporządkowania procesu stworzyłem trzy foldery: Konfiguracja_lokalna, Konfiguracja_chmurowa i Zaawansowana_konfiguracja_chmurowa. Podstawowe konfiguracje Moodle, lokalna i chmurowa, są identyczne, co pozwala na zachowanie spójności i prostoty wdrożeń. Folder Zaawansowana_konfiguracja_chmurowa zawiera dodatkowe ustawienia dla bardziej wymagających środowisk, takich jak integracja z Amazon Elastic File System (EFS) i Elastic Load Balancer (ELB). Te zaawansowane funkcje wspierają skalowalność i wysoką dostępność systemu, co jest istotne przy dużych wdrożeniach produkcyjnych. Taka organizacja struktury ułatwia zarządzanie konfiguracją w zależności od potrzeb i stopnia skomplikowania środowiska.

Baza danych:

```
1 sudo apt update
2 sudo apt upgrade -y
3 sudo apt install -y mysql-server
4 sudo mysql -e "CREATE DATABASE moodle DEFAULT CHARACTER SET utf8mb4
  COLLATE utf8mb4_unicode_ci;"
5 sudo mysql -e "CREATE USER '#{$db_user}'@'#{$web_ip}' IDENTIFIED BY
  '#{$db_password}';"
```

6	sudo mysql -e "GRANT ALL ON moodle.* TO '#{$db_user}'@'#{$web_ip}';"
7	sudo mysql -e "FLUSH PRIVILEGES;"
8	sudo sed -i 's/^bind-address.*/bind-address = 0.0.0.0/' /etc/mysql/mysql.conf.d/mysqld.cnf
9	sudo systemctl restart mysql

Flaga *-y* automatycznie potwierdza wszystkie pytania, które mogłyby pojawić się podczas procesów.

- **sudo apt update**

To polecenie aktualizuje listę pakietów dostępnych w repozytoriach systemowych.

- **sudo apt upgrade -y**

Wykonuje aktualizację wszystkich zainstalowanych pakietów do ich najnowszych wersji.

- **sudo apt install -y mysql-server**

Instaluje serwer bazy danych MySQL.

- **sudo mysql -e "CREATE DATABASE moodle DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;"**

Tworzy nową bazę danych o nazwie moodle z domyślnym zestawem znaków utf8mb4 i porównywaniem (collation) ustawionym na utf8mb4_unicode_ci zgodnie z dokumentacja z oficjalnej strony moodle

- **sudo mysql -e "CREATE USER '#{$db_user}'@'#{$web_ip}' IDENTIFIED BY '#{$db_password}';"**

Tworzy nowego użytkownika bazy danych o nazwie określonej przez zmienną db_user, który będzie mógł łączyć się z bazy z adresu IP określonego przez web_ip. Użytkownik ten będzie uwierzytelniany za pomocą hasła zdefiniowanego w zmiennej db_password.

- **sudo mysql -e "GRANT ALL ON moodle.* TO '#{$db_user}'@'#{$web_ip}';"**

Nadaje nowemu użytkownikowi pełne uprawnienia (GRANT ALL) do bazy danych moodle i jej tabel. Umożliwia to użytkownikowi pełne zarządzanie bazą, w tym wstawianie, aktualizowanie i usuwanie danych.

- **sudo mysql -e "FLUSH PRIVILEGES;"**

Odświeża pamięć podręczną uprawnień MySQL, aby zastosować zmiany wprowadzone w uprawnieniach użytkownika bez potrzeby restartowania serwera MySQL.

- **sudo sed -i 's/^bind-address.*/bind-address = 0.0.0.0/' /etc/mysql/mysql.conf.d/mysqld.cnf**

Edytuje plik konfiguracyjny serwera MySQL, zmieniając wartość bind-address na 0.0.0.0. To polecenie pozwala serwerowi MySQL nasłuchiwać połączeń na wszystkich interfejsach

sieciowych, umożliwiając zdalny dostęp do bazy danych. sed -i oznacza edycję pliku w miejscu (bez tworzenia kopii).

Moodle:

```
1  sudo apt update
2  sudo apt install -y mysql-client apache2 php php-mysql php-mbstring php-xml php-curl php-zip php-gd php-intl php-soap
3  sudo wget https://download.moodle.org/download.php/direct/stable405/moodle-latest-405.tgz
4  sudo mv moodle-latest-405.tgz /var/www/html/
5  cd /var/www/html/
6  sudo tar -xf moodle-latest-405.tgz
7  sudo chown www-data moodle
8  cd ..
9  sudo mkdir moodledata
10 sudo chown www-data moodledata
11 sudo sed -i 's/^;\?max_input_vars\s*=\s*[0-9]*/max_input_vars = 5000/' /etc/php/8.1/apache2/php.ini
12 sudo systemctl restart apache2.service
```

- **sudo apt update && sudo apt install -y mysql-client apache2 php php-mysql php-mbstring php-xml php-curl php-zip php-gd php-intl php-soap**

Aktualizuje listę pakietów, a następnie instaluje wymagane oprogramowanie i biblioteki, w tym klienta MySQL, serwer Apache, interpreter PHP oraz rozszerzenia PHP potrzebne do działania Moodle (np. obsługę baz danych, przetwarzanie XML, SOAP, czy manipulację obrazami.

- **sudo wget https://download.moodle.org/download.php/direct/stable405/moodle-latest-405.tgz**

Pobiera najnowszą wersję Moodle w wersji 4.0.5 z oficjalnej strony Moodle.

- **sudo mv moodle-latest-405.tgz /var/www/html/**

Przenosi pobrany plik archiwum Moodle do katalogu /var/www/html/, który jest domyślnym miejscem przechowywania plików serwera Apache.

- **cd /var/www/html/**

Przechodzi do katalogu /var/www/html/, gdzie został przeniesiony plik Moodle.

- **sudo tar -xf moodle-latest-405.tgz**

Rozpakowuje archiwum moodle-latest-405.tgz, tworząc katalog o nazwie moodle w katalogu /var/www/html/

- **sudo chown www-data moodle**

Zmienia właściciela katalogu moodle na użytkownika www-data, który jest użytkownikiem domyślnym używanym przez serwer Apache. Dzięki temu serwer ma odpowiednie uprawnienia do odczytu i zapisu w tym katalogu.

- **sudo mkdir moodledata**

Tworzy nowy katalog moodledata, który Moodle wykorzystuje do przechowywania danych takich jak pliki przesłane przez użytkowników, cache i dane sesji.

- **sudo chown www-data moodledata**

Zmienia właściciela katalogu moodledata na użytkownika www-data, aby serwer Apache miał dostęp do zapisu w tym katalogu.

- **sudo sed -i 's/^;\?max_input_vars\s*=\s*[0-9]*/max_input_vars = 5000/' /etc/php/8.1/apache2/php.ini**

Edytuje plik konfiguracyjny PHP (php.ini) odpowiedzialny za działanie PHP na serwerze Apache. Ustawia parametr max_input_vars na wartość 5000, co zwiększa maksymalną liczbę zmiennych, które mogą być przetwarzane przez PHP. Jest to konieczne dla poprawnego działania Moodle zgodnie z dokumentacją.

- **sudo systemctl restart apache2.service**

Ponownie restartuje usługę Apache, aby uwzględnić zmiany wprowadzone w konfiguracji PHP.

Moodle –zaawansowana konfiguracja chmurowa

1	efs_id="${aws_efs_file_system.efs.id}"
2	sudo apt update -y
3	sudo apt install -y mysql-client apache2 php php-mysql php-mbstring php-xml php-curl php-zip php-gd php-intl php-soap git binutils rustc cargo pkg-config libssl-dev gettext nfs-common
4	sudo wget https://download.moodle.org/download.php/direct/stable405/moodle-latest-405.tgz
5	sudo mv moodle-latest-405.tgz /var/www/html/
6	cd /var/www/html/
7	sudo tar -xf moodle-latest-405.tgz
8	sudo chown www-data moodle
9	cd ..
10	sudo mkdir moodledata
11	sudo mkdir /tmp/efs

12	cd /tmp/efs
13	sudo git clone https://github.com/aws/efs-utils
14	cd efs-utils
15	sudo ./build-deb.sh
16	sudo apt-get install -y ./build/amazon-efs-utils*deb
17	cd /var/www/
18	echo "$efs_id.efs.eu-west-2.amazonaws.com:/ /var/www/moodledata nfs4 _netdev,nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport 0 0" >> /etc/fstab
19	sudo mount -a
20	sudo rm -rf /tmp/efs
21	sudo chown www-data moodledata
22	sudo systemctl restart apache2.service
23	sudo sed -i 's/^;\?max_input_vars\s*=\s*[0-9]*/max_input_vars = 5000/' /etc/php/8.1/apache2/php.ini
24	sudo systemctl restart apache2.service

- **efs_id="${aws_efs_file_system.efs.id}"**

Przypisuje identyfikator systemu plików Amazon Elastic File System (EFS) do zmiennej efs_id. Identyfikator pochodzi z konfiguracji Terraformu.

- **sudo ./build-deb.sh**

Buduje pakiet instalacyjny **amazon-efs-utils** w formacie .deb.

- **echo "$efs_id.efs.eu-west-2.amazonaws.com:/ /var/www/moodledata nfs4 _netdev,nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport 0 0" >> /etc/fstab**

Dodaje wpis do pliku /etc/fstab, który określa montowanie systemu plików EFS w katalogu /var/www/moodledata. Montowanie odbywa się przy użyciu protokołu NFSv4, z parametrami optymalizującymi wydajność i stabilność połączenia.

- **sudo mount -a**

Montuje wszystkie systemy plików zdefiniowane w pliku /etc/fstab, w tym EFS.

Sposób aktywacji

Aby rozpocząć pracę, należy otworzyć dowolny terminal, który obsługuje komendy systemowe i narzędzia takie jak Git czy Vagrant. W systemie Windows można użyć Command Prompt (CMD), PowerShell lub Gitbash. W przypadku systemów macOS i Linux, można skorzystać z wbudowanego terminala dostępnego w systemie. Po otwarciu terminala należy przejść do katalogu, w którym zamierzasz umieścić pliki projektu, za

pomocą komendy cd. Następnie wykonaj polecenie git clone **!!!!!!!!!link** , aby pobrać pliki z repozytorium. W tym momencie użytkownik wybiera konfiguracje poprzez przejście do folderu z odpowiednia konfiguracją.

Folder **Konfiguracja_lokalna** – należy wpisac komendę *vagrant up*, aby uruchmoić maszyny, po ich uruchomieniu system Moodle będzie dostępny pod wskazanym adresem IP, zależnym od wybranej konfiguracji w pliku **variables**.

Folder **Konfiguracja_chmurowa** - należy wpisac komendę *terrafrom init*, to polecenie inicjalizuje katalog projektu. Terraform pobiera wszystkie wymagane moduły i wtyczki potrzebne do pracy z wybraną platformą chmurową (np. AWS, Azure). *terraform plan* – opcjonalnie do sprawdzenia jakie zmiany Terraform zamierza wprowadzić w infrastrukturze na podstawie plików konfiguracyjnych oraz *terraform apply*, aby uruchmoić skrypty. Należy odczekać aż maszyny zostaną automatycznie skonfigurowane, po ich uruchomieniu system Moodle będzie dostępny pod adresem IP wskazanym w terminalu.

Po uruchomieniu skryptów konfigurujących Moodle, infrastruktura systemu jest przygotowana, a serwer aplikacji oraz baza danych zostały odpowiednio skonfigurowane. Aby zakończyć instalację, użytkownik musi wejść na stronę Moodle, korzystając z adresu IP serwera. W przypadku konfiguracji lokalnej będzie to adres maszyny wirtualnej, natomiast w środowisku chmurowym publiczny adres przydzielony przez dostawcę, na przykład AWS.

Po otwarciu strony użytkownik zostanie przekierowany do instalatora Moodle, który poprowadzi przez proces konfiguracji. W trakcie tego etapu konieczne jest podanie szczegółów dotyczących bazy danych, takich jak jej adres, nazwa, użytkownik oraz hasło. Wprowadzone dane powinny być zgodne z ustawieniami użytymi w skryptach. Instalator przeprowadzi następnie automatyczne przygotowanie systemu i poprosi o określenie podstawowych ustawień witryny, w tym założenie konta administratora.

Po zakończeniu instalacji użytkownik uzyskuje dostęp do panelu administracyjnego Moodle, gdzie może dostosować system do swoich potrzeb. Moodle umożliwia rozpoczęcie pracy od utworzenia pierwszego kursu za pomocą wbudowanego kreatora. Proces ten pozwala na szybkie zorganizowanie struktury materiałów, dodanie treści dydaktycznych oraz skonfigurowanie uczestników. Po wykonaniu tych kroków system jest gotowy do użytkowania.

Folder **Zaawansowana_konfiguracja_chmurowa** wymaga realizacji procesu w dwóch etapach. W pierwszym kroku należy powtórzyć wszystkie czynności z konfiguracji opisanej w folderze **Part1**, obejmujące wdrożenie podstawowej infrastruktury przy użyciu polecenia *terraform apply*. Następnie użytkownik powinien uruchomić instalator

Moodle, skonfigurować bazę danych, dostosować ustawienia systemu oraz stworzyć pierwszy kurs zgodnie z własnymi potrzebami.

Po zakończeniu konfiguracji podstawowej należy przejść do folderu **Part2** i ponownie wykonać polecenie terraform apply. W tym kroku Terraform wdraża dodatkowe zasoby, takie jak Elastic Load Balancer (ELB), które zapewniają równoważenie obciążenia oraz skalowalność systemu. Po tej operacji Moodle będzie dostępny pod nowym adresem DNS przypisanym do Load Balancera. Dzięki tej konfiguracji ruch użytkowników zostaje efektywnie rozdzielony pomiędzy większą liczbę maszyn, co zwiększa wydajność i niezawodność systemu.

Bibliografia

[1] Imię Nazwisko, Imię Nazwisko. *Tytuł książki*. Wydawnictwo, Warszawa, 2017.

[2] Imię Nazwisko, Imię Nazwisko. Tytuł artykułu w czasopiśmie. *Tytuł czasopisma*, 157(8):1092–1113, 2016.

[3] Imię Nazwisko, Imię Nazwisko, Imię Nazwisko. Tytuł artykułu konferencyjnego. *Nazwa konferencji*, str. 5346–5349, 2006.

[4] Autor, jeśli znany. https: www.adres.strony (dostęp:dzień.miesiąc.rok)

Spis skrótów i symboli

DNA	kwas deoksyrybonukleinowy (ang. *deoxyribonucleic acid*)
MVC	model – widok – kontroler (ang. *model–view–controller*)
N	liczebność zbioru danych
μ	stopień przynależności do zbioru

Źródła

Jeżeli w pracy konieczne jest umieszczenie długich fragmentów kodu źródłowego, należy je przenieść do tego miejsca.

Lista dodatkowych plików, uzupełniających tekst pracy

W systemie, do pracy dołączono dodatkowe pliki zawierające:

- źródła programu,
- dane testowe
- film pokazujący działanie opracowanego oprogramowania lub zaprojektowanego i wykonanego urządzenia,
- itp.

Spis rysunków

Spis tablic