

Spis treści

O autorze	15
O recenzentach	16
Przedmowa	17
CZĘŚĆ 1. DevOps i infrastruktura jako kod	
Rozdział 1	
Kultura DevOps i praktyki kodowania infrastruktury	25
Pierwsze kroki z DevOps	25
Wdrażanie CI/CD i ciągłe wdrażanie	29
Ciągła integracja (CI)	29
Ciągłe dostarczanie (CD)	31
Ciągłe wdrażanie	33
Zrozumienie praktyk IaC	35
Korzyści IaC	36
Języki i narzędzia IaC	36
Topologia IaC	39
Ewolucja kultury DevOps	44
Podsumowanie	45
Pytania	45
Dalsza lektura	46
Rozdział 2	
Udostępnianie infrastruktury chmury za pomocą Terraform	47
Wymagania techniczne	48
Instalacja Terraform	48
Instalacja ręczna	49
Instalacja za pomocą skryptu	49
Integracja Terraform z Azure Cloud Shell	53
Konfigurowanie Terraform dla platformy Azure	55
Tworzenie jednostki usługi Azure SP	55
Konfiguracja dostawcy Terraform	57

Konfiguracja Terraform w celu rozwoju aplikacji i testowania	58
Tworzenie skryptu Terraform w celu wdrożenia infrastruktury Azure	60
Postępowanie zgodnie z dobrymi praktykami Terraform	63
Uruchamianie Terraform w celu wdrożenia	66
Inicjalizacja	67
Podgląd zmian	69
Stosowanie zmian	70
Zrozumienie cyklu życia Terraform z różnymi opcjami wiersza polecenia	72
Używanie polecenia destroy w celu przebudowy	72
Formatowanie i walidacja konfiguracji	74
Cykl życia Terraform w procesie CI/CD	75
Ochrona pliku stanu za pomocą zdalnego zaplecza	77
Podsumowanie	82
Pytania	82
Dalsza lektura	83
Rozdział 3	
Używanie Ansible do konfigurowania infrastruktury IaaS	84
Wymagania techniczne	85
Instalacja Ansible	86
Instalacja Ansible za pomocą skryptu	86
Integracja Ansible z Azure Cloud Shell	88
Artefakty Ansible	89
Konfiguracja Ansible	90
Tworzenie pliku inwentarza Ansible	92
Plik inwentarza	93
Konfigurowanie hostów w pliku inwentarza	94
Testowanie pliku inwentarza	95
Uruchomienie pierwszego playbooka	97
Tworzenie prostego playbooka	97
Opis modułów Ansible	98
Ulepszanie playbooków za pomocą ról	99

Uruchomienie Ansible	100
Korzystanie z podglądu lub z opcji testowej pracy (ang. dry run)	102
Zwiększanie poziomu logowania	104
Ochrona danych za pomocą Ansible Vault	104
Używanie zmiennych w Ansible w celu lepszej konfiguracji	105
Ochrona wrażliwych danych za pomocą Ansible Vault	108
Korzystanie z dynamicznego pliku inwentarza dla infrastruktury Azure	110
Podsumowanie	115
Pytania	115
Dalsza lektura	115
Rozdział 4	
Optymalizacja wdrażania infrastruktury za pomocą Packera	117
Wymagania techniczne	118
Opis Packera	119
Instalacja Packera	119
Tworzenie szablonów Packera dla maszyn wirtualnych Azure za pomocą skryptów	125
Struktura szablonu Packera	125
Tworzenie obrazu platformy Azure za pomocą szablonu Packera	130
Tworzenie szablonów Packera przy użyciu Ansible	133
Tworzenie playbooka Ansible	134
Integracja playbooka Ansible z szablonem Packera	135
Uruchamianie Packera	136
Konfigurowanie Packera do uwierzytelniania na platformie Azure	136
Sprawdzanie poprawności szablonu Packera	137
Uruchamianie Packera w celu wygenerowania naszego obrazu maszyny wirtualnej	138
Tworzenie szablonów Packera w formacie HCL	140
Korzystanie z obrazów utworzonych przez Packera za pomocą Terraform	143
Podsumowanie	145
Pytania	145

Dalsza lektura	146
Rozdział 5	
Tworzenie środowiska programistycznego z Vagrantem	147
Wymagania techniczne	148
Instalacja Vagranta	148
Instalacja ręczna (w systemie Windows)	148
Instalowanie Vagranta za pomocą skryptu w systemie Windows	150
Instalowanie Vagranta za pomocą skryptu w systemie Linux	151
Tworzenie pliku konfiguracyjnego Vagranta	152
Używanie Vagrant Cloud dla boksów Vagranta	152
Tworzenie pliku konfiguracyjnego Vagranta	154
Tworzenie lokalnej maszyny wirtualnej za pomocą interfejsu Vagrant CLI	156
Tworzenie maszyny wirtualnej	156
Łączenie z maszyną wirtualną	158
Podsumowanie	159
Pytania	160
Dalsza lektura	160
CZĘŚĆ 2. Potok CI/CD	
Rozdział 6	
Zarządzanie kodem źródłowym za pomocą Gita	163
Wymagania techniczne	164
Przegląd Gita i jego głównych poleceń	164
Instalacja Gita	166
Konfiguracja Gita	176
Terminologia Gita	176
Polecenia Gita	177
Zrozumienie procesu Gita i wzorca Gitflow	181
Zaczynamy od procesu Gita	181
Izolacja kodu za pomocą gałęzi	190
Strategia tworzenia gałęzi z Gitflow	195
Podsumowanie	198

Pytania	198
Dalsza lektura	199
Rozdział 7	
Ciągła integracja i ciągłe wdrażanie	200
Wymagania techniczne	201
Zasady CI/CD	201
CI	201
CD	202
Korzystanie z menedżera pakietów w procesie CI/CD	203
Prywatne repozytorium NuGet i npm	205
Repozytorium Nexusa OSS	205
Azure Artifacts	206
Używanie Jenkinsa do implementacji CI/CD	208
Instalowanie i konfigurowanie Jenkinsa	208
Konfiguracja webhooka GitHuba	210
Konfiguracja zadania CI w Jenkinsie	211
Wykonywanie zadania Jenkinsa	215
Korzystanie z Azure Pipelines dla CI/CD	216
Wersjonowanie kodu za pomocą Gita w Azure Repos	218
Tworzenie potoku CI	219
Tworzenie potoku CD — nowa wersja aplikacji	228
Tworzenie pełnej definicji potoku w pliku YAML	234
Korzystanie z GitLab CI	240
Uwierzytelnianie w GitLabie	241
Tworzenie nowego projektu i zarządzanie kodem źródłowym	242
Tworzenie potoku CI	245
Dostęp do szczegółów wykonania potoku CI	247
Podsumowanie	249
Pytania	249
Dalsza lektura	249
Rozdział 8	

Wdrażanie infrastruktury jako kodu za pomocą potoku CI/CD	251
Wymagania techniczne	252
Uruchamianie Packera w Azure Pipelines	252
Uruchamianie Terraform i Ansible w Azure Pipelines	255
Podsumowanie	260
Pytania	261
Dalsza lektura	261
CZĘŚĆ 3. Konteneryzowane mikrouслуги	
wykorzystujące platformę Docker i Kubernetes	
Rozdział 9	
Konteneryzacja aplikacji za pomocą Dockera	265
Wymagania techniczne	266
Instalowanie Dockera	267
Rejestracja w Docker Hubie	267
Instalacja Dockera	268
Przegląd elementów Dockera	272
Tworzenie pliku Dockerfile	273
Tworzenie pliku Dockerfile	273
Przegląd instrukcji Dockerfile	274
Budowanie i uruchamianie kontenera na komputerze lokalnym	276
Tworzenie obrazu Dockera	276
Tworzenie nowego kontenera obrazu	278
Lokalne testowanie kontenera	279
Wysyłanie obrazu do Docker Huba	279
Wysyłanie obrazu Dockera do rejestru prywatnego (ACR)	283
Wdrażanie kontenera do ACI za pomocą potoku CI/CD	285
Tworzenie kodu Terraform dla ACI	286
Tworzenie potoku CI/CD dla kontenera	287
Korzystanie z Dockera przy użyciu narzędzi wiersza poleceń	294
Pierwsze kroki z Docker Compose	296
Instalowanie Docker Compose	297

Tworzenie pliku konfiguracyjnego dla Docker Compose	298
Wykonywanie Docker Compose	299
Wdrażanie kontenerów Docker Compose w ACI	300
Podsumowanie	303
Pytania	303
Dalsza lektura	304
Rozdział 10	
Efektywne zarządzanie kontenerami za pomocą Kubernetesa	305
Wymagania techniczne	306
Instalacja Kubernetesa	306
Przegląd architektury Kubernetesa	307
Instalacja Kubernetesa na komputerze lokalnym	307
Instalacja pulpitu nawigacyjnego Kubernetesa	309
Pierwszy przykład wdrożenia aplikacji w Kubernetesie	312
Używanie Helma jako menedżera pakietów	316
Instalacja klienta Helma	316
Korzystanie z publicznego pakietu Helma, dostępnego w Artifact Hubie	317
Tworzenie niestandardowego charta Helma	321
Publikowanie charta Helma w rejestrze prywatnym (ACR)	323
Korzystanie z AKS	325
Tworzenie usługi AKS	326
Konfigurowanie pliku kubeconfig dla AKS	327
Zalety AKS	328
Tworzenie potoku CI/CD dla Kubernetesa za pomocą Azure Pipelines	329
Monitorowanie aplikacji i metryk w Kubernetesie	330
Korzystanie z wiersza poleceń kubectl	330
Korzystanie z interfejsu webowego	331
Korzystanie z narzędzi	332
Podsumowanie	334
Pytania	334
Dalsza lektura	335

CZĘŚĆ 4. Testowanie aplikacji

Rozdział 11

Testowanie interfejsów API za pomocą Postmana	339
Wymagania techniczne	340
Tworzenie kolekcji żądań Postmana	340
Instalacja Postmana	341
Tworzenie kolekcji	342
Tworzenie pierwszego żądania	343
Wykorzystywanie środowisk i zmiennych do dynamizowania żądań	346
Tworzenie testów Postmana	349
Wykonywanie lokalnych testów za pomocą żądań Postmana	351
Zrozumienie koncepcji Newmana	354
Przygotowywanie kolekcji Postmana dla Newmana	356
Eksportowanie kolekcji	356
Eksportowanie środowisk	357
Korzystanie z wiersza poleceń Newmana	359
Integracja Newmana z procesem potoku CI/CD	361
Budowa i udostępnianie konfiguracji	362
Wykonanie potoku	366
Podsumowanie	367
Pytania	368
Dalsza lektura	368

Rozdział 12

Statyczna analiza kodu za pomocą SonarQube	369
Wymagania techniczne	370
Odkrywanie SonarQube	370
Instalacja SonarQube	371
Przegląd architektury SonarQube	371
Instalacja SonarQube	372
Analiza w czasie rzeczywistym za pomocą SonarLint	378
Wykonywanie SonarQube w procesie CI	380

Konfigurowanie SonarQube	381
Tworzenie potoku CI dla SonarQube w Azure Pipelines	381
Podsumowanie	385
Pytania	385
Dalsza lektura	385
Rozdział 13	
Testy bezpieczeństwa i wydajności	386
Wymagania techniczne	386
Stosowanie zabezpieczeń internetowych i testów penetracyjnych za pomocą narzędzia ZAP	387
Korzystanie z ZAP-a w celu testowania bezpieczeństwa	388
Sposoby automatyzacji wykonywania ZAP-a	390
Uruchamianie testów wydajności za pomocą Postmana	392
Podsumowanie	395
Pytania	395
Dalsza lektura	395
CZĘŚĆ 5. Więcej informacji na temat DevOps	
Rozdział 14	
Bezpieczeństwo w procesie DevOps z wykorzystaniem DevSecOps	399
Wymagania techniczne	400
Testowanie infrastruktury Azure za pomocą InSpec	400
Omówienie InSpec	401
Instalacja InSpec	402
Konfigurowanie platformy Azure dla InSpec	403
Tworzenie testów InSpec	405
Wykonywanie InSpec	408
Ochrona poufnych danych dzięki Vault od HashiCorp	410
Lokalna instalacja programu Vault	411
Uruchamianie serwera Vault	413
Zapisywanie haseł w Vault	415
Odczytywanie sekretów z Vault	416

Korzystanie z interfejsu webowego (UI) programu Vault	418
Pobieranie sekretów Vault w Terraform	421
Podsumowanie	425
Pytania	425
Dalsza lektura	425
Rozdział 15	
Skrócenie czasu przestoju wdrażania	426
Wymagania techniczne	427
Skrócenie czasu przestojów we wdrażaniu dzięki Terraform	427
Zrozumienie zielono-niebieskich koncepcji i wzorców wdrażania	429
Korzystanie z wdrożenia zielono-niebieskiego w celu ulepszenia środowiska produkcyjnego	430
Opis wzorca Canary release	430
Badanie wzorca Dark launch	432
Stosowanie wdrożeń zielono-niebieskich na platformie Azure	432
Używanie App Service z gniazdami	433
Korzystanie z usługi Azure Traffic Manager	434
Wprowadzenie flag funkcjonalności	436
Używanie frameworka open source dla flag funkcjonalności	438
Korzystanie z narzędzia LaunchDarkly	443
Podsumowanie	447
Pytania	448
Dalsza lektura	448
Rozdział 16	
DevOps dla projektów open source	449
Wymagania techniczne	450
Przechowywanie kodu źródłowego w GitHubie	451
Tworzenie nowego repozytorium na GitHubie	451
Przyczynianie się do rozwoju projektu w GitHubie	453
Przyczynianie się do rozwoju projektów open source przy użyciu żądań pobierania	455

Zarządzanie plikiem dziennika zmian i informacjami o wydaniu	459
Udostępnianie plików binarnych w wydaniach GitHuba	461
Wprowadzenie do GitHub Actions	464
Analiza kodu za pomocą SonarCloud	468
Wykrywanie luk w zabezpieczeniach za pomocą narzędzia WhiteSource Bolt	473
Podsumowanie	476
Pytania	477
Dalsza lektura	478
Rozdział 17	
Najlepsze praktyki DevOps	479
Pełna automatyzacja	480
Wybór odpowiedniego narzędzia	480
Tworzenie całej konfiguracji za pomocą kodu	481
Projektowanie architektury systemu	482
Budowanie dobrego potoku CI/CD	484
Testy integracyjne	485
Przesunięcie bezpieczeństwa w lewo dzięki DevSecOps	486
Monitorowanie systemu	487
Ewoluuujące zarządzanie projektami	488
Podsumowanie	489
Pytania	489
Dalsza lektura	490
Odpowiedzi	491
Rozdział 1. Kultura DevOps i praktyki kodowania infrastruktury	491
Rozdział 2. Udostępnianie infrastruktury chmury za pomocą Terraform	491
Rozdział 3. Używanie Ansible do konfigurowania infrastruktury IaaS	492
Rozdział 4. Optymalizacja wdrażania infrastruktury za pomocą Packera	492
Rozdział 5. Tworzenie środowiska programistycznego z Vagrantem	493
Rozdział 6. Zarządzanie kodem źródłowym za pomocą Gita	493
Rozdział 7. Ciągła integracja i ciągłe wdrażanie	494
Rozdział 8. Wdrażanie infrastruktury jako kodu za pomocą potoku CI/CD	494

Rozdział 9. Konteneryzacja aplikacji za pomocą Dockera	494
Rozdział 10. Efektywne zarządzanie kontenerami za pomocą Kubernetesa	495
Rozdział 11. Testowanie interfejsów API za pomocą Postmana	495
Rozdział 12. Statyczna analiza kodu za pomocą SonarQube	495
Rozdział 13. Testy bezpieczeństwa i wydajności	496
Rozdział 14. Bezpieczeństwo w procesie DevOps z wykorzystaniem DevSecOps	496
Rozdział 15. Skrócenie czasu przestoju wdrażania	496
Rozdział 16. DevOps dla projektów open source	497
Rozdział 17. Najlepsze praktyki DevOps	497
Skorowidz	499