

Spis treści |

O autorze	11
O korektorach merytorycznych	12
Przedmowa	13
Wprowadzenie	14

CZĘŚĆ 1. Podstawy Gita, GitHuba i DevOps

ROZDZIAŁ 1

DevOps i wrażenia programisty	21
DevOps — przyspieszenie cyklu tworzenia oprogramowania poprzez zmniejszenie tarć	21
Kontekst dla podejścia DevOps	22
Czym jest DevOps?	23
Czym NIE jest DevOps?	25
DevOps to kultura pracy	27
Osiąganie doskonałości w stosowaniu praktyk DevOps	31
Następne wyzwanie	35
Wrażenia programisty — strategia sprzyjająca osiągnięciu doskonałości	36
Wrażenia programisty to strategia	36
Elementy wzmacniające podejście DevOps i wrażenia programisty	41
Git — system, od którego rozpoczyna się współpraca nad kodem źródłowym	43
Świat bez systemu kontroli wersji	44
Historia systemu Git	44
Czym jest VCS?	45
GitHub — platforma programistyczna wspierana przez sztuczną inteligencję	47
Wsparcie przez sztuczną inteligencję	47
Współpraca	48
Produktywność	49

Bezpieczeństwo	49
Skala	50
Podsumowanie	51
Dalsza lektura	51

ROZDZIAŁ 2

Rozpoczęcie pracy z systemem kontroli wersji Git	52
Wymagania techniczne	52
Rozpoczęcie pracy z systemem kontroli wersji Git	53
Podstawy systemu Git — praktyczne wprowadzenie	53
Praca z gałęziami — kamień węgielny współpracy	58
Anatomia systemu Git — zrozumiałe wyjaśnienie sposobu działania Gita	62
Cykl życiowy pliku w systemie Git	62
Pod maską — architektura systemu Git	64
Struktura drzewa w systemie Git	67
Jak stać się guru w zakresie komunikacji za pomocą systemu Git?	70
git commit — powtórzenie najważniejszego polecenia	71
Kontrola jakości i ilości jako wyznacznik dobrej komunikacji	72
Podsumowanie	79

ROZDZIAŁ 3

Zaawansowane funkcjonalności Gita do współpracy w zespole	81
Wymagania techniczne	81
Strategie korzystania z gałęzi systemu Git podczas współpracy w zespole	82
Dlaczego strategia stosowania gałęzi jest istotna?	82
Strategia i polityka stosowania gałęzi	83
Mniej i częściej kontra więcej i rzadziej	84
Typy polityk stosowania gałęzi	85
Konwencje nazewnictwa gałęzi — najlepsze praktyki w zakresie nadawania nazw gałęziom	90
Sposoby integrowania zmian w gałęzi	91
Scalenie kontra operacja rebase	92
Różne sposoby przeprowadzania operacji scalenia w systemie Git	94
Rozwiązywanie konfliktów	111
Dlaczego pojawia się konflikt?	111
Jak radzić sobie z konfliktem podczas scalania w systemie Git?	111
Jak rozwiązać konflikt powstały podczas scalania?	112
Polecenia przydatne podczas rozwiązywania konfliktów	113
Poprawa współpracy w zespole	114
Przywracanie do stanu z określonego momentu	115
Organizacja środowiska roboczego	118

Kto co zrobił, czyli doskonała pomoc podczas debugowania	120
Doskonałe wersjonowanie	121
Podsumowanie	122

CZĘŚĆ 2. Zaawansowane funkcje GitHuba oraz podstawy potoku ciągłej integracji i ciągłego wdrażania

ROZDZIAŁ 4

GitHub i wyższy poziom współpracy w zespole	125
Wymagania techniczne	125
Rozpoczęcie pracy z platformą GitHub	126
Tworzenie konta na platformie GitHub	126
Tworzenie pierwszego repozytorium na GitHubie	127
Rejestrowanie klucza SSH	129
git remote — połączenie repozytoriów lokalnego i zdalnego	133
git push — Twój kod ma znaczenie	134
Analiza kodu na platformie GitHub	135
git pull — połączenie środowisk pracy lokalnego i zdalnego	141
git fetch — synchronizacja bez zakłóceń	142
git fetch kontra git pull	142
git clone — skopiowanie repozytorium z GitHuba do przestrzeni roboczej	143
Tworzenie kopii repozytorium — więcej niż kopiowanie kodu źródłowego	144
GitHub Issues — sprawna współpraca na platformie GitHub	146
Z czego wynika unikatowość GitHub Issues?	147
Podstawy przygotowywania zgłoszeń problemu	148
Efektywna komunikacja	151
Prośba o scalenie kodu	155
Z czego wynika unikatowość prośby o scalenie kodu?	156
Tworzenie prośby o scalenie kodu	157
Prośby o scalenie kodu w szczegółach	166
Jeszcze bardziej zaawansowane funkcjonalności platformy GitHub	172
GitHub Projects — jedno miejsce, w którym można zarządzać zgłoszeniami problemów i prośbami o scalenie kodu	172
GitHub Codespaces — przepływ pracy programistycznej w środowisku opartym na chmurze	174
GitHub Discussions — wsparcie współpracy i społeczności	177

Jeszcze sprawniejsza praca z repozytorium GitHub	179
Reguły repozytorium — usprawnienie przepływu pracy i zapewnienie jakości kodu	179
CODEOWNERS — usprawniony przegląd i własność	182
Szablony zgłoszenia problemu i prośby o scalenie kodu	183
Podsumowanie	184

ROZDZIAŁ 5

Potok CI/CD utworzony za pomocą GitHuba	185
GitHub Actions — automatyzacja przepływu pracy	185
Zalety usługi GitHub Actions	186
Struktura przepływu pracy na platformie GitHub	188
Najlepsze praktyki w zakresie korzystania z GitHub Actions	194
Strategie wdrażania	201
Wdrożenie typu niebieski – zielony	202
Wdrożenia ciągłe	205
Wdrażanie kanarkowe	208
Strategie wydań funkcjonalności	211
Opcja włączająca funkcjonalność	211
Pociąg wydania	214
Podsumowanie	215
Dalsza lektura	216

CZĘŚĆ 3. Nie tylko DevOps

ROZDZIAŁ 6

Rozbudowanie implementacji DevOps	219
Wykorzystanie wskaźników w podejściu DevOps	219
Cztery klucze — wskaźniki DORA	220
Framework SPACE	221
Wskaźniki na platformie GitHub	222
DevSecOps — bezpieczeństwo jako nieustannie analizowany aspekt	230
Przesunięcie w lewo	231
Funkcje bezpieczeństwa na platformie GitHub	232
Skalowanie i współpraca	243
Dlaczego skalowanie współpracy jest ważne?	243
InnerSource — rozproszony model współpracy	244
Konfiguracja platformy GitHub na potrzeby skalowania współpracy	250
Podsumowanie	252
Dalsza lektura	252

ROZDZIAŁ 7

Zwiększenie produktywności dzięki sztucznej inteligencji	254
Pojawienie się sztucznej inteligencji w programowaniu	255
Wpływ dużych modeli językowych na programowanie	255
Duże modele językowe — krótkie wprowadzenie	256
Zastosowanie dużych modeli językowych w programowaniu	258
Zapytania dla modeli i kontekst	261
Możliwości i wykorzystanie sztucznej inteligencji w programowaniu	264
Uzupełnianie kodu — podstawa programowania wspomaganego przez sztuczną inteligencję	265
Wyjaśnianie kodu źródłowego	267
Strategie maksymalizujące efektywność sztucznej inteligencji	269
Dokładność	270
Kontekst	270
Spójność	272
Podsumowanie	274
Dalsza lektura	274

ROZDZIAŁ 8

Refleksja i podsumowanie	275
Refleksja nad technologiami Git, GitHub i DevOps — poprawa wrażeń programisty	275
Wykorzystanie sztucznej inteligencji w programowaniu — następny krok w ewolucji inżynierii oprogramowania	277
Ostatnie uwagi	278