Możliwość dofinansowania

Claude Code - praktyczny przewodnik programisty

Numer usługi 2026/09/28/196900/3840522

2 952,00 PLN brutto
2 400,00 PLN netto
184,50 PLN brutto/h
150,00 PLN netto/h
157,50 PLN cena rynkowa

CONTROLBYTE SPÓŁKA Z OGRANICZONĄ ODPOWIEDZIALNOŚCIĄ

4,4 / 5

6 ocen

Usługa szkoleniowa
zdalna w czasie rzeczywistym
Zajęcia grupowe
16:00 h
21.11.2026 do 22.11.2026

Informacje podstawowe

Kategoria	Informatyka i telekomunikacja / Programowanie
Grupa docelowa usługi	Szkolenie jest skierowane do osób, które zawodowo piszą lub utrzymują kod i chcą pracować z agentem AI Claude Code w terminalu. W szczególności do programistów aplikacji webowych, desktopowych, mobilnych i usług backendowych, testerów automatyzujących i inżynierów QA piszących testy, inżynierów DevOps utrzymujących skrypty i infrastrukturę jako kod, liderów technicznych i architektów wprowadzających narzędzia AI w zespole oraz studentów i absolwentów kierunków informatycznych, którzy już programują. Wymagana jest praktyczna umiejętność programowania w dowolnym języku oraz podstawy pracy z terminalem i systemem kontroli wersji git. Szkolenie nie uczy programowania od zera.
Minimalna liczba uczestników	5
Maksymalna liczba uczestników	10
Data zakończenia rekrutacji	19-11-2026
Forma prowadzenia usługi	zdalna w czasie rzeczywistym
Podstawa uzyskania wpisu do BUR	Znak Jakości Małopolskich Standardów Usług Edukacyjno-Szkoleniowych (MSUES) - wersja 2.0

Cel

Cel edukacyjny

Samodzielna konfiguracja agenta kodującego Claude Code w repozytorium i powierzanie mu zadań programistycznych z zachowaniem kontroli nad kodem i bezpieczeństwa projektu.

Efekty uczenia się oraz kryteria weryfikacji ich osiągnięcia i Metody walidacji

Efekty uczenia się	Kryteria weryfikacji	Metoda walidacji
Uczestnik zna budowę i zasadę działania sterowników PLC Siemens S7-1200/S7-1500 (cykl pracy, pamięć, typy zmiennych), języki LAD i FBD oraz strukturę bloków OB/FC/FB	Poprawnie opisuje cykl pracy sterownika, typy danych, przeznaczenie bloków OB/FC/FB oraz zasadę działania podstawowych instrukcji LAD/FBD.	Obserwacja w warunkach symulowanych
. Uczestnik potrafi skonfigurować sterownik i utworzyć projekt w TIA Portal oraz zaprogramować instrukcje bitowe, czasowe (timery), liczniki, arytmetyczne i porównania.	Samodzielnie tworzy projekt, konfiguruje CPU i wejścia/wyjścia oraz poprawnie stosuje timery, liczniki i operatory w programie sprawdzonym w symulatorze	Obserwacja w warunkach symulowanych
Uczestnik potrafi zaprojektować i uruchomić kompletną aplikację sterowania (PLC + panel HMI) oraz przetestować ją w środowisku Factory I/O.	Wykonuje program sterujący stanowiskiem (np. linia sortująca / manipulator 2D), tworzy wizualizację HMI i uruchamia całość w symulacji, uzyskując poprawne działanie.	Obserwacja w warunkach symulowanych
Uczestnik samodzielnie rozwiązuje problemy programistyczne, podejmuje decyzje projektowe i stosuje dobre praktyki inżynierskie	Samodzielnie diagnozuje i usuwa błędy w programie, uzasadnia przyjęte rozwiązania oraz stosuje czytelną, uporządkowaną strukturę kodu.	Obserwacja w warunkach symulowanych
Uczestnik charakteryzuje narzędzia sztucznej inteligencji wspierające pracę automatyka (asystenci kodu, agenci AI, modele LLM) oraz zasady ich bezpiecznego stosowania w środowisku przemysłowym	Poprawnie opisuje możliwości i ograniczenia narzędzi AI oraz dobiera właściwe narzędzie do zadania inżynierskiego.	Obserwacja w warunkach symulowanych
Uczestnik potrafi z pomocą asystenta AI tworzyć i uruchamiać programy PLC (LAD/SCL) oraz kompletne aplikacje sterowania w symulatorze Factory I/O (sekwencje, paletyzator, magazyn).	Samodzielnie tworzy z pomocą AI działający program sterowania i uruchamia go w symulatorze, uzyskując poprawne działanie.	Obserwacja w warunkach symulowanych
Uczestnik potrafi automatyzować tworzenie struktury projektu w TIA Portal Openness (UDT, tablice tagów, OB/FC/FB) oraz integrować sterowniki PLC z aplikacjami w Pythonie (VS Code).	Generuje automatycznie strukturę projektu TIA Portal i uruchamia skrypt Python komunikujący się ze sterownikiem.	Obserwacja w warunkach symulowanych
Uczestnik programuje urządzenia brzegowe (Finder Opta, Arduino) z wykorzystaniem agenta AI (Cline) - od analizy schematu połączeń po działający program.	Z pomocą agenta AI analizuje schemat połączeń i koduje działający program dla urządzenia brzegowego.	Obserwacja w warunkach symulowanych

Efekty uczenia się	Kryteria weryfikacji	Metoda walidacji
Uczestnik zna zasady działania protokołów komunikacji przemysłowej (Modbus RTU/TCP, PROFINET IO, OPC UA, MQTT) oraz formaty wymiany danych (CSV/JSON/XML) i metody archiwizacji danych (SQL).	Poprawnie opisuje różnice między protokołami, strukturę ramek/telegramów oraz zasady logowania danych do bazy.	Obserwacja w warunkach symulowanych
Uczestnik potrafi skonfigurować i uruchomić komunikację Modbus (TCP/RTU) oraz PROFINET IO między PLC a urządzeniami.	Samodzielnie konfiguruje master/serwer Modbus i sieć PROFINET, uzyskując poprawną wymianę danych sprawdzoną w symulatorze.	Obserwacja w warunkach symulowanych
. Uczestnik potrafi uruchomić komunikację OPC UA i MQTT, logować dane procesowe do bazy SQL oraz wizualizować je narzędziami OT/IT.	Konfiguruje serwer/klienta OPC UA lub MQTT, zapisuje dane do bazy i prezentuje je (Node-RED/Grafana), uzyskując działający przepływ danych.	Obserwacja w warunkach symulowanych
Uczestnik samodzielnie diagnozuje problemy komunikacyjne i stosuje dobre praktyki inżynierskie.	Samodzielnie lokalizuje i usuwa błędy komunikacji, uzasadnia dobór protokołu i konfiguracji.	Obserwacja w warunkach symulowanych

Kwalifikacje

Kompetencje

Usługa prowadzi do nabycia kompetencji.

Warunki uznania kompetencji

Pytanie 1. Czy dokument potwierdzający uzyskanie kompetencji lub wyraźnie z nim powiązane inne dokumenty związane ze wsparciem zawierają opis efektów uczenia się?

TAK

Pytanie 2. Czy dokument lub wyraźnie z nim powiązane inne dokumenty związane ze wsparciem potwierdzają, że walidacja została przeprowadzona w oparciu o zdefiniowane w efektach uczenia się kryteria ich weryfikacji i zgodnie z zaplanowanymi metodami walidacji?

TAK

Pytanie 3. Czy dokument lub wyraźnie z nim powiązane inne dokumenty związane ze wsparciem potwierdzają zastosowanie rozwiązań zapewniających rozdzielenie procesów kształcenia i szkolenia od walidacji?

TAK

Program

1. Start i sterowanie sesją (2 h)
2. Instalacja i pierwsze uruchomienie Claude Code w repozytorium.
3. Tryb -p: praca w potoku powłoki i w skryptach.

4. Wznawianie rozmowy, dobór modelu i jego zmiana w trakcie sesji.
5. Polecenia powłoki i wskazywanie plików w rozmowie.
6. Przerywanie pracy agenta i cofanie zmian w kodzie.
7. Ćwiczenia: pierwsze zadania w repozytorium ćwiczeniowym.
8. Pamięć projektu i okno kontekstu (2 h)
9. Plik CLAUDE.md, hierarchia plików pamięci i importy.
10. Notatki przechodzące między sesjami.
11. Czyszczenie i kompaktowanie kontekstu, kiedy stosować które.
12. Prompt caching i co go zeruje.
13. Poziomy namysłu modelu i ich koszt.
14. Ćwiczenia: CLAUDE.md dla własnego projektu.
15. Uprawnienia i bezpieczna praca (2 h)
16. Tryby zgód i tryb planowania.
17. Dlaczego pomijanie zgód nie jest skrótem do celu.
18. Reguły allow, ask i deny w pliku settings.json.
19. Dostęp do katalogów spoza repozytorium, zawężanie zestawu narzędzi.
20. Sandbox dla poleceń powłoki.
21. Ćwiczenia: konfiguracja uprawnień projektu.
22. Automatyzacja: hooki, własne komendy i skille (2 h)
23. Hook PostToolUse: automatyczne formatowanie pliku po edycji.
24. Hook PreToolUse: zatrzymanie niebezpiecznego polecenia przed wykonaniem.
25. Własna komenda i skill z jednego pliku SKILL.md.
26. Skille ładowane na żądanie, kto uruchamia skill: człowiek czy model.
27. Ćwiczenia: hook i skill dla repozytorium ćwiczeniowego.
28. Agenci, MCP i wtyczki (2 h)
29. Subagenci ze świeżym kontekstem i fork rozmowy.
30. Podłączenie serwera MCP, zakresy local, project i user oraz plik .mcp.json w repozytorium.
31. Marketplace wtyczek i koszt kontekstu.
32. Złożenie poznanych narzędzi we własny workflow.
33. Agent w codziennym cyklu pracy i projekt własny (3 h)
34. Praca od testu: agent implementuje, dopóki testy nie przejdą.
35. Równoległe sesje w osobnych git worktree.
36. Przegląd zmian przez agenta przed commitem, opis commita i pull requesta.
37. Warsztat na własnym projekcie: zadanie przeprowadzone od planu do zatwierdzonej zmiany.
38. Konsultacje indywidualne z prowadzącym w trakcie warsztatu.
39. Omówienie wybranych rozwiązań uczestników i sesja pytań.
40. Walidacja (1 h)
41. Test końcowy online, 20 pytań zamkniętych jednokrotnego wyboru, próg zaliczenia 60 procent.
42. Usługa realizowana w formule godzin zegarowych (60 min). Przerwy wliczają się w czas trwania usługi. Walidacja wlicza się w czas trwania usługi.

Harmonogram

Liczba pozycji harmonogramu: 16

Przedmiot / temat	Typ aktywności	Prowadzący	Data realizacji zajęć	Godzina rozpoczęcia	Godzina zakończenia	Liczba godzin
1 z 16 Blok 1. Start i sterowanie sesją: instalacja, tryb -p, wznawianie rozmowy, dobór modelu, cofanie zmian	Zajęcia	Paweł Halladin	21-11-2026	08:00	09:00	01:00

Przedmiot / temat	Typ aktywności	Prowadzący	Data realizacji zajęć	Godzina rozpoczęcia	Godzina zakończenia	Liczba godzin
2 z 16 Blok 1 cd. Sterowanie sesją: wznawianie rozmowy, dobór modelu, polecenia powłoki, cofanie zmian	Zajęcia	Paweł Halladin	21-11-2026	09:00	10:00	01:00
3 z 16 Blok 2. Pamięć projektu: CLAUDE.md, hierarchia plików pamięci, importy	Zajęcia	Paweł Halladin	21-11-2026	10:00	11:00	01:00
4 z 16 Blok 2 cd. Okno kontekstu: czyszczenie i kompaktowanie, prompt caching, poziomy namysłu	Zajęcia	Paweł Halladin	21-11-2026	11:00	12:00	01:00
5 z 16 -	Przerwa	-	21-11-2026	12:00	13:00	01:00
6 z 16 Blok 3. Uprawnienia: tryby zgód, tryb planowania, reguły allow, ask i deny	Zajęcia	Paweł Halladin	21-11-2026	13:00	14:00	01:00
7 z 16 Blok 3 cd. Bezpieczna praca: zawężanie narzędzi, katalogi spoza repozytorium, sandbox	Zajęcia	Paweł Halladin	21-11-2026	14:00	15:00	01:00

Przedmiot / temat	Typ aktywności	Prowadzący	Data realizacji zajęć	Godzina rozpoczęcia	Godzina zakończenia	Liczba godzin
8 z 16 Blok 4. Hooki: PostToolUse i PreToolUse	Zajęcia	Paweł Halladin	21-11-2026	15:00	16:00	01:00
9 z 16 Blok 4 cd. Własne komendy i skille (SKILL.md)	Zajęcia	Paweł Halladin	22-11-2026	08:00	09:00	01:00
10 z 16 Blok 5. Subagenci i fork rozmowy	Zajęcia	Paweł Halladin	22-11-2026	09:00	10:00	01:00
11 z 16 Blok 5 cd. Serwery MCP, wtyczki, własny workflow	Zajęcia	Paweł Halladin	22-11-2026	10:00	11:00	01:00
12 z 16 Blok 6. Agent w cyklu pracy: praca od testu, równoległe sesje, przegląd zmian przed commitem	Zajęcia	Paweł Halladin	22-11-2026	11:00	12:00	01:00
13 z 16 -	Przerwa	-	22-11-2026	12:00	13:00	01:00
14 z 16 Blok 6 cd. Warsztat na własnym projekcie, konsultacje indywidualne	Zajęcia	Paweł Halladin	22-11-2026	13:00	14:00	01:00
15 z 16 Blok 6 cd. Omówienie rozwiązań uczestników, sesja pytań i odpowiedzi	Zajęcia	Paweł Halladin	22-11-2026	14:00	15:00	01:00
16 z 16 -	Walidacja	-	22-11-2026	15:00	16:00	01:00

Podsumowanie

Rodzaj godzin	Liczba godzin
Suma godzin zegarowych usługi	16:00
w tym suma godzin zajęć	13:00
w tym suma godzin walidacji	01:00
w tym suma przerw	02:00
Suma godzin dydaktycznych bez przerw	18:30

Cennik

Jeżeli korzystasz z dofinansowania i usługa stanowi usługę kształcenia zawodowego lub przekwalifikowania zawodowego wraz z usługą lub dostawą towarów ściśle związaną z usługami kształcenia zawodowego lub przekwalifikowania zawodowego to możesz mieć możliwość skorzystania za zwolnienia z podatku VAT na podstawie art. 43 ust. 1 pkt 29 lit. c ustawy z dnia 11 marca 2004 r. o podatku od towarów i usług, jeśli usługa w całości jest finansowana ze środków publicznych lub § 3 ust. 1 pkt 14 rozporządzenia Ministra Finansów z dnia 20 grudnia 2013 r. w sprawie zwolnień od podatku od towarów i usług oraz warunków stosowania tych zwolnień w przypadku, gdy usługa jest finansowana w co najmniej 70% ze środków publicznych.

Cennik

Rodzaj ceny	Cena
Koszt przypadający na 1 uczestnika brutto	2 952,00 PLN
Koszt przypadający na 1 uczestnika netto	2 400,00 PLN
Koszt osobogodziny brutto	184,50 PLN
Koszt osobogodziny netto	150,00 PLN

Liczba godzin usługi

Rodzaj godzin	Liczba godzin
Liczba godzin zegarowych usługi	16:00

Prowadzący

Liczba prowadzących: 1

1 z 1

Paweł Halladin

Programista z ponad 20-letnim doświadczeniem, od 2004 roku projektuje i tworzy aplikacje desktopowe, webowe, mobilne i embedded. Od 5 lat trener IT: prowadzi szkolenia grupowe dla osób dorosłych, w tym usługi realizowane w ramach Bazy Usług Rozwojowych, oraz zajęcia indywidualne. Specjalizuje się w C#/.NET, Java i Spring, Python, C/C++, SQL oraz w wykorzystaniu AI w pracy programisty (Claude Code, MCP, lokalne modele językowe). W latach 2025-2026 przeprowadził 195 godzin szkoleń grupowych, między innymi dla ERNABO (Java i Spring, 55 h, w ramach BUR), ESKADA (C++ i .NET, 130 h) oraz PARTNER-BHP (10 h). Średnia ocena 4,5/5 z 60 opinii kursantów. Autor kanału YouTube @pawelhalladin z ponad 40 filmami edukacyjnymi, w tym serii o Claude Code liczącej ponad 30 odcinków.

Informacje dodatkowe

Informacje o materiałach dla uczestników usługi

Uczestnik otrzymuje bezterminowy dostęp do repozytorium ćwiczeniowego na GitHubie, w którym pracuje w trakcie zajęć: zadania do każdego bloku szkolenia wraz z rozwiązaniami oraz gotowe pliki konfiguracyjne do wykorzystania we własnych projektach (CLAUDE.md, settings.json, przykładowe hooki, skill oraz .mcp.json). Uzupełnieniem jest prowadzona przez trenera publiczna seria nagrań o Claude Code na YouTube, ponad 30 odcinków, do której Uczestnik dostaje odnośniki tematycznie przypisane do bloków szkolenia. Materiały są uzupełnieniem zajęć na żywo i nie wchodzą do rozliczenia z operatorem.

Warunki techniczne

Platforma / komunikator: usługa realizowana zdalnie w czasie rzeczywistym za pośrednictwem platformy Google Meet, z dwustronną komunikacją audio-wideo między trenerem a Uczestnikami oraz współdzieleniem ekranu. Link do spotkania przekazywany jest Uczestnikom przed rozpoczęciem zajęć wraz z instrukcją dołączenia.

Minimalne wymagania sprzętowe:

- komputer stacjonarny lub laptop z systemem Windows 10/11 (64-bit), macOS 13 lub nowszym albo Linux,
- procesor 4-rdzeniowy, min. 8 GB RAM (zalecane 16 GB), min. 10 GB wolnego miejsca na dysku,
- monitor o rozdzielczości min. 1920x1080 (zalecany drugi monitor: kod na jednym, prowadzący na drugim),
- kamera internetowa i mikrofon (wymagane do udziału w zajęciach i potwierdzenia obecności), zalecane słuchawki,
- uprawnienia administratora na komputerze, niezbędne do instalacji oprogramowania.

Minimalne wymagania sieciowe:

- stałe łącze internetowe o przepustowości min. 10 Mb/s pobierania i 5 Mb/s wysyłania (zalecane 20/10 Mb/s),
- stabilne połączenie umożliwiające ciągły przekaz audio-wideo przez cały czas trwania sesji,
- dostęp do internetu bez blokad firmowego firewalla na api.anthropic.com oraz github.com (Claude Code i repozytorium ćwiczeniowe wymagają połączenia z tymi usługami).

Niezbędne oprogramowanie (bezpłatne, instalowane wg przekazanej instrukcji):

- Claude Code (instalowany na początku szkolenia wg instrukcji prowadzącego),
- Node.js w wersji 18 lub wyższej,
- git, a na systemie Windows dodatkowo Git for Windows, z którego Claude Code korzysta do wykonywania poleceń powłoki,
- edytor kodu, zalecany Visual Studio Code,
- aktualna przeglądarka Google Chrome (zalecana do obsługi Google Meet i testu walidacyjnego),

- bezpłatne konto GitHub, potrzebne do pobrania repozytorium ćwiczeniowego.

Wymagany dostęp po stronie Uczestnika:

- własny dostęp do Claude Code: aktywna subskrypcja Claude Pro lub wyższa albo klucz API z doładowanymi środkami. Dostęp zapewnia Uczestnik i nie jest on wliczony w cenę usługi.

Wymagania wstępne wobec Uczestnika: praktyczna umiejętność programowania w dowolnym języku oraz podstawy pracy z terminalem i systemem kontroli wersji git. Szkolenie nie uczy programowania od zera.

Nie jest wymagany żaden dodatkowy sprzęt ani płatne licencje oprogramowania. Wszystkie ćwiczenia realizowane są na komputerze Uczestnika, w repozytorium ćwiczeniowym udostępnionym przez prowadzącego oraz opcjonalnie na własnym projekcie Uczestnika.

Kontakt

MACIEJ KURANTOWICZ

E-mail maciej.kurantowicz@controlbyte.pl

Telefon (+48) 509 735 835