Szukasz prostego przepisu na skuteczne prompty dla agentów AI w OpenClaw lub podobnym środowisku? Krótko i bez ogródek: dobry prompt w agentach to precyzyjny kontrakt pracy, nie ozdobny wstęp. Zdefiniuj rolę, cel, ograniczenia, narzędzia i kryteria sukcesu, a agent przestanie błądzić. W tym tekście pokazuję, jak to robić po polsku, na przykładach, z pułapkami, które kosztują najwięcej nerwów i tokenów.

OpenClaw traktuję tutaj jako parasolową nazwę dla podejścia agentowego, w którym model językowy ma rolę koordynatora, potrafi wzywać narzędzia, zapisuje pamięć i dąży do celu w kilku krokach. Jeśli używasz innego frameworka, zasady zostają te same. Różnica jest głównie w składni i konfiguracji, nie w logice projektowania promptu.

Co właściwie różni prompty agentowe od zwykłych

W czacie z modelem często wystarczy jeden opis zadania. Agenty ai są wybredniejsze, bo zamiast jednego strzału wykonują serię działań. Na starcie potrzebują więc mapy, a nie jednego znaku drogowego. Ta mapa składa się z kilku części: roli, narzędzi, pamięci, planu i kryteriów zakończenia. I to właśnie te elementy powinny znaleźć się w twoim promptcie.

Wyobraź sobie agenta jako seniora, który potrafi dużo, ale nie czyta w myślach. Jeśli mu nie opiszesz zakresu obowiązków, wydatków i definicji gotowego wyniku, zrealizuje swoją wersję. A jego wersja rzadko pokrywa się z twoją.

Krótki szkielet dobrego promptu w OpenClaw

Użyj tego jako punktu startowego. Nie jest to gotowiec do bezmyślnego wklejania, raczej szyna, po której łatwiej pojedziesz.

[Rola] Jesteś agentem [nazwa roli], który [misja w jednym zdaniu]. [Cel] Twoim celem jest [konkretny rezultat], mierzalny przez [kryteria]. [Kontekst] Oto aktualne informacje i ograniczenia: - użytkownik: [intencja lub dane wejściowe] - zasoby: [pliki, API, baza] - ograniczenia: [czas, budżet, styl, język] - definicja DONE: [kiedy uznajemy, że koniec] [Narzędzia] Masz do dyspozycji narzędzia: - [tool_1]: [kiedy używać, wejście, wynik] - [tool_2]: [jw.] Jeśli żadne narzędzie nie pasuje, wyjaśnij, czego brakuje. [Plan pracy] Zanim zaczniesz, zaproponuj krótki plan kroków, np. Analiza, wybór narzędzia, wykonanie, weryfikacja, raport. [Zasady] - pracuj po polsku - nie wymyślaj faktów; jeśli brak danych, zapytaj - po każdej akcji oceń: czy osiągnąłeś definicję DONE - jeśli wynik jest dłuższy niż [limit], zrób zwięzłe streszczenie [Format odpowiedzi] Zwróć: [oczekiwany format, np. JSON, markdown, tabela].

Ten format pasuje do OpenClaw, bo porządkuje rolę i zasoby, ale nie zamyka agenta w kajdanach. Zostawia mu swobodę wyboru narzędzi i taktyki, a to zwykle najlepszy kompromis między kreatywnością a kontrolą.

Dobre prompty są mało poetyckie i bardzo konkretne

Nie ma nagrody za najładniejsze zdania. Jest za poprawny wynik. Dwa zdania, które często robią różnicę: nazwanie definicji DONE oraz wyraźne wskazanie kryteriów oceny. Kiedy agent wie, że wynik ma przejść walidację X i Y, redukuje zbędne ruchy.

Przykład z prostego agenta analizy tekstu:

Rola: Analityk jakości Cel: wykryj niespójności w tekście instrukcji i zaproponuj poprawki Kontekst: plik docs/instrukcja.md; język polski; terminologia firmowa z glosariusza docs/glossary.csv Narzędzia: read_file(path),

write_file(path, content), validate_terms(text) Zasady: nie zmieniaj formatowania nagłówków; nie skracaj kroków proceduralnych; jeśli termin nie pasuje do glosariusza, zaproponuj 1 alternatywę Definicja DONE: istnieje wersja pliku z poprawkami oraz lista niespójności z cytatem linii Format odpowiedzi: markdown z dwiema sekcjami: Niespójności, Nowa wersja pliku

Takie polecenie precyzuje, co agent ma zrobić, jak ma ocenić sukces i w jakiej formie zwrócić wynik. Użytkownik dostaje to, co zamówił, a nie ogólną notatkę.

Jak opisywać narzędzia, żeby agent naprawdę ich użył

Modele nie czytają dokumentacji tak jak my. Krótkie, atomowe opisy narzędzi działają lepiej niż eseje. Jeśli możesz, wypisz dla każdego z nich trzy rzeczy: kiedy ma sens, jakie ma wejście i co zwraca. I koniecznie uwzględnij ograniczenia. Wtedy agent nie próbuje pchać URL do funkcji, która czeka na ścieżkę pliku.

Dobry opis narzędzia:

Tool: search_web(query: string, top_k: int=5) -> list[Result] Używaj: gdy brakuje ci aktualnych informacji lub musisz potwierdzić fakt Zwraca: lista wyników z tytułem, URL i krótkim opisem Ograniczenia: bez logowania; nie przetwarza PDF bezpośrednio

Świetnym trikiem jest dodanie krótkiego przykładu wejścia i wyjścia. Agent widzi, jak wygląda poprawna interakcja i chętniej z niej korzysta.

Rola i ton: naucz agenta mówić jak twój zespół

Jeśli tworzysz agenta do polskiej obsługi klienta, wpisz to wprost i podaj styl. Krótkie, uprzejme, bez żargonu, z jednym pytaniem kontrolnym na koniec. Dla zespołów technicznych dodaj preferencje: czy wolimy YAML, JSON, czy krótkie bloki kodu. Agent, który trafia stylem, od razu wygląda mądrzej.

Przykład regulacji tonu:

Styl: zwięzły, rzeczowy, bez żargonu marketingowego Język: polski Format: jeśli podajesz konfigurację, użyj YAML; komentarze minimalne

To naprawdę pomaga, szczególnie gdy pracują nad tym różne osoby i chcesz mieć spójny głos.

Jak zmusić agenta do planowania, nie do biegania w kółko

Agenty są szybkie, ale potrafią być impulsywne. Jeśli chcesz ograniczyć chaos, dodaj sekcję Plan pracy i poproś o zwięzły plan przed pierwszą akcją. Nie musisz oglądać całego łańcucha myślenia, wystarczy nagłówek kroków. To działa jak krótka rozgrzewka dla mózgu modelu.

Przykład:

Plan pracy: 1) analiza braków danych 2) wybór narzędzia 3) wykonanie 4) kontrola jakości według [kryteria] 5) raport końcowy

W niektórych środowiskach możesz wymusić tryb Plan-Act-Observe. W OpenClaw lub analogach zwykle da się to osiągnąć prostą konwencją: po każdym użyciu narzędzia agent zapisuje obserwację i przyrównuje ją do definicji DONE.

Definicja DONE: prosty miernik, który oszczędza budżet

Jeśli nie wskażesz końca, agent będzie kręcił pętlę. Idealna definicja DONE to krótka lista warunków, które można sprawdzić automatycznie. Plik istnieje, JSON waliduje się w schemacie, testy przeszły, w raporcie są sekcje A i B. Im mniej subiektywna ocena, tym lepiej.

Przykład twardych kryteriów:

- istnieje plik reports/audit_YYYYMMDD.md
- w pliku jest nagłówek H2 pod tytułem Ryzyka
- test unitowy risk *score*test.py zwraca OK

To jest nasza pierwsza lista w artykule. Trzy punkty, zero wątpliwości. I od razu łatwiej zautomatyzować weryfikację.

Jak dobrać budżet tokenów i temperaturę

Agent toczy dialog z narzędziami i kontekstem. W praktyce szybko topnieje limit tokenów. Dobrze jest przyjąć konkretne widełki:

- krótka sprawa, jedno narzędzie, bez długich plików: 1,5 do 3 tys. Tokenów
- przetwarzanie dokumentów i walidacje: 4 do 8 tys. Tokenów
- wieloetapowe badanie z trzema narzędziami: 8 do 16 tys. Tokenów

Nie chodzi o to, żeby zawsze brać maksimum. Wysoki kontekst zwiększa koszty i ryzyko halucynacji przez rozmycie celu. Lepiej podać agentowi tylko to, co naprawdę jest mu potrzebne. Temperatura ustaw blisko 0,2 dla zadań deterministycznych, 0,5 dla szkiców i 0,7, gdy prosisz o warianty kreatywne. Dla agentów technicznych trzymaj się nisko, inaczej potrafią wymyślać parametry API.

Pamięć i kontekst: co warto podawać, a czego nie

Pokusą jest dodać cały projekt do kontekstu. Agent tego nie przeczyta, a zapłacisz jak za zboże. Lepszy sposób to:

- podaj wycinki, które na pewno będą użyte
- trzymaj glosariusz terminów w osobnym, lekkim pliku
- ucinaj stare, nieistotne obserwacje z bieżącej sesji
- jeśli agent długo szuka danych, pozwól mu je dociągnąć narzędziem

W promptach po polsku precyzyjnie nazywaj pojęcia. Zamiast ogólnego dokument dodaj instrukcję: sekcja 3.2 pliku docs/procesy.md. Agent mniej błądzi po interpretacjach.

Jak zadawać ograniczenia, które agent naprawdę respektuje

Zasady działają, gdy są sprawdzalne. Zamiast prosić o krótko, określ limit znaków. Zamiast proszę nie kłam, napisz: jeśli nie masz źródła, zwróć brak danych i poproś o zgodę na wyszukiwanie. Zamiast bądź uprzejmy, wskaż frazy grzecznościowe i zakazane.

Przykłady twardych ograniczeń:

- maksymalnie 500 słów w sekcji Wnioski - każdy adres URL w raporcie musi przejść HEAD 200 - cytuj wersję pakietu w formacie nazwa==x.y.z - jeśli brakuje danych, zwróć "status":"needs_input","field":"X"

Takie reguły nadają się do automatycznego sprawdzenia, więc agent nie dyskutuje.

Polski w promptach a jakość odpowiedzi

Jeśli twoją publicznością są Polacy, pisz prompty po polsku i proś o odpowiedzi po polsku. To poprawia styl, ton i zgodność terminologii. Zdarza się jednak, że narzędzia preferują angielski, na przykład nazwy funkcji API. Wtedy wprowadź prostą zasadę: opis i komentarze po polsku, komendy i parametry po angielsku. Ta mieszanka jest zaskakująco praktyczna i minimalizuje zgrzyty.

Warto też doprecyzować odmianę i diakrytykę. Lepiej poprosić: używaj polskich znaków i odmiany przez przypadki niż potem poprawiać ręcznie.

Przykładowy prompt dla agenta badawczego w OpenClaw

Złóżmy to w całość na przykładzie prostego pipeline'u badawczego. Agent ma sprawdzić, czy nowa funkcja w produkcie X ma realny potencjał wśród małych firm.

Rola: Analityk rynku MŚP Misja: ocenić zainteresowanie funkcją [Feature] wśród małych firm w PL Cel: wypracuj krótki brief (do 400 słów) oraz tabelę argumentów za/przeciw Kryteria sukcesu: - uwzględnij min. 3 źródła z ostatnich 12 miesięcy - podaj liczby porządkowe lub widełki, jeśli brak twardych danych - zaznacz poziom pewności: niski, średni, wysoki Kontekst: - branża: SaaS B2B - grupa: firmy 5-50 osób - język: polski - dostępne dane: docs/personas_msp.md, docs/pricing.csv Narzędzia: - search_web(query, top_k=5) -> list[Result]; używaj do świeżych danych - fetch_url(url) -> text; używaj do weryfikacji źródła - read_file(path) -> text; lokalne pliki kontekstu - save_report(path, content) -> void; zapis końcowy Zasady: - cytuj źródła w nawiasach z URL - jeśli źródło nie zawiera daty, odrzuć je - nie używaj opinii bez danych; oznacz je jako hipotezy Plan pracy: 1) sprawdź pliki lokalne 2) wyszukaj 3-5 aktualnych źródeł 3) zweryfikuj źródła fetch_url 4) przygotuj brief i tabelę argumentów 5) zapisz raport do reports/msp_feature_brief.md Format odpowiedzi: - sekcja Brief - sekcja Argumenty w tabeli markdown - sekcja Źródła

Taki prompt jest krótki, ale zawiera wszystkie ważne elementy. Agent ma jasny cel, wie czym mierzymy sukces i jak ma wyglądać wynik. Ma też zapisany sposób pracy, więc nie pędzi na ślepo.

Walidacja wyników i autokorekta agenta

Nie zawsze trafimy od razu. Zamiast ręcznie przeglądać każde słowo, warto dodać do promptu lub do pipeline'u proste walidatory. Schema JSON na format raportu, kontrola obecności sekcji, sprawdzenie dat źródeł. Jeśli walidacja nie przechodzi, agent dostaje automatyczną informację zwrotną z krótkim komunikatem, na przykład brakuje sekcji Źródła lub limit 400 słów przekroczony o 120 słów, skróć.

Dobrze też działa mechanizm auto-review: drugi, lekki agent ocenia, czy odpowiedź pierwszego spełnia kryteria. Nie potrzebujesz do tego rozbudowanej infrastruktury. Wystarczy mini prompt:

Rola: Reviewer Zadanie: oceń odpowiedź względem kryteriów Wejście: [odpowiedź], [kryteria] Wyjście: JSON "score":0-1,"missing":[...],"notes":"..."

To zwiększa spójność i wymusza na głównym agencie trzymanie formatu.

Jak unikać halucynacji i złudnych pewników

Agenty z natury chcą być pomocne. Czasem aż za bardzo. Przy zadaniach faktograficznych pilnuj trzech rzeczy: rozdziału faktów od opinii, jawnego statusu źródeł i zgody na sięgnięcie do sieci. Jeśli model nie dostanie pozwolenia na web search, ale i tak będzie musiał zweryfikować dane, zacznie zgadywać. Daj mu więc prostą ścieżkę: jeśli brak danych, poproś o web search lub o wejście użytkownika.

Kolejna pułapka to niejednoznaczne nazwy. Jeżeli masz w organizacji produkt o nazwie podobnej do popularnej biblioteki lub firmy, nazwij go w promptcie pełną ścieżką i wersją, np. Produkt X, moduł fakturowania v2, a nie po prostu moduł. To mocno ogranicza błędne rozpoznanie kontekstu.

Debugowanie promptów bez marnowania dnia

Zamiast wróżyć, co poszło nie tak, przeprowadź mały audyt. Poniższa, krótka lista kroków zwykle wystarcza, by wyłapać 90 procent problemów.

- czy rola, cel, narzędzia i definicja DONE są zdefiniowane i niesprzeczne
- czy format odpowiedzi można automatycznie zweryfikować
- czy agent ma prawo i drogę do dociągnięcia brakujących danych
- czy temperatura i limit tokenów pasują do złożoności zadania
- czy kontekst nie jest zbyt długi w stosunku do celu

To nasza druga i zarazem ostatnia lista. Resztę pokażemy w praktyce, bez wypunktowań.

Przykłady krótkich klauzul, które ratują dzień

Dobrze napisany prompt ma kilka zwrotów, które wydają się drobiazgami, a porządkują całą rozmowę. Pierwszy to lakoniczne jeśli - to. Na przykład jeśli nie masz pewności, wróć do mnie z listą pytań. Drugi to rozdział informacji twardych od gładkich: cytuj liczby z jednostkami, a supozycje tak oznacz. Trzeci to jasny format odpowiedzi. Kiedy agent ma zwrócić JSON, daj mu prosty schema lub przykład zgodny z oczekiwaniami. Nie licz na to, że rozwinie skróty, jeśli sam ich nie rozwiniesz.

Przykład zwrotów, które sprawdzają się w polskich promptach:

- podawaj jednostki miary, jeśli pojawiają się liczby
- jeśli nie rozumiesz żądania, wypisz konkretne wątpliwości w punktach
- po każdym użyciu narzędzia zapisz pokrótce, co wynik oznacza dla celu

Niby drobiazgi, ale usuwają łańcuch mikro-niedomówień.

Odpowiedzialność i bezpieczeństwo: ograniczenia nie są opcjonalne

Jeśli agent ma dostęp do narzędzi, które coś zmieniają w środowisku, zamknij mu zakres działania. Określ katalogi, do których może pisać, hosty, do których może się łączyć, i maksymalny limit skutków, na przykład nie usuwaj plików, tylko przenoś do katalogu quarantine. W promptcie wpisz twarde zakazy. Niech agent nie ma wątpliwości.

Dobrym nawykiem jest też tryb dry run. Zanim agent wywoła operację ryzykowną, musi wygenerować plan i symulację efektu. Wtedy możesz go zatwierdzić lub odrzucić. Tę regułę można utrzymać jednym zdaniem w sekcji Zasady: operacje destrukcyjne tylko po akceptacji suchych kroków.

Jak pisać prompty wielojęzyczne i nie zgubić polskiej jakości

Czasem agent ma rozmawiać po polsku, a pracować na danych po angielsku. Nie komplikuj. Jasno przestaw dwie warstwy: język interfejsu to polski, język danych to angielski. Jeśli powstaną cytaty, zachowaj oryginał i dodaj krótkie tłumaczenie. To wystarczy, by zachować sens i nie wpaść w żmudne tłumaczenie openclaw na polski przepisywanie.

Możesz też dodać prostą glossę z polskimi odpowiednikami najczęściej używanych pojęć technicznych. Dzięki temu agent nie wymyśli alternatywnych tłumaczeń, które tylko mieszają, na przykład widok zamiast ekran lub szkielet zamiast scaffold.

OpenClaw po polsku w praktyce: mini-playbook

Załóżmy, że budujesz agenta, który kataloguje błędy zgłoszone przez użytkowników i tworzy szkice zgłoszeń Jira. Potrzebujesz stylu po polsku, ale technicznie precyzyjnego.

Szkic promptu:

Rola: Asystent wsparcia technicznego Cel: przekształć surowe zgłoszenia użytkowników w szkice ticketów Jira Kontekst: plik CSV z kolumnami [timestamp, user_id, message]; słownik komponentów apps/components.yml Narzędzia: read_file, write_file, classify_component(text), generate_summary(text, max_chars=280) Zasady: - pisz po polsku - tytuł do 80 znaków - opis w 3 akapitach: objawy, środowisko, kroki odtworzenia - przypisz komponent z classify_component; jeśli niepewne, component=Unclassified - nie zmieniaj oryginalnych cytatów użytkownika Definicja DONE: wygenerowane pliki tickets/*.md, każdy z polami Title, Component, Priority, Description Format: markdown z nagłówkami H2

Gdy taki agent zadziała, zwykle odbije się od trzech progów: błędy klasyfikacji komponentu, za długie tytuły i niekompletne kroki odtworzenia. Każda z tych przeszkód da się ogarnąć doprecyzowaniem zasad i dodaniem walidatora formatu. Na przykład wprowadź twardy limit znaków i schemat checklist dla kroków.

Różnica między instrukcją a coachingiem agenta

Czasem kusi, żeby uczyć agenta dobrych praktyk w długich esejach. Zwykle szkoda na to budżetu. Lepiej sprawdzają się krótkie, operacyjne reguły. Zamiast wykładu o pisaniu dobrych commit message, podaj wzór: Krótki tytuł w trybie rozkazującym, pusta linia, opis motywacji, pole Zahacza o issue #ID. Agent wejdzie w ten format jak w kapcie.

Podobnie z planowaniem. Jedno zdanie wystarczy: zanim użyjesz narzędzia, wyjaśnij, co chcesz osiągnąć i jak ocenisz wynik. Nie trzeba recenzji literatury.

Kiedy prompt jest już dobry, a problem leży gdzie indziej

Zdarza się, że nawet najlepszy prompt nie pomoże, bo przeszkoda jest systemowa. Na przykład:

- model ma zbyt małe okno kontekstowe i gubi kluczowe informacje
- brak narzędzia, które sięga po wymagane dane
- dane są niejednoznaczne, a ty nie przewidziałeś etapu doprecyzowania z użytkownikiem
- format wyjściowy jest niemożliwy do utrzymania bez walidatora po stronie systemu

Dobrze jest mieć zdrowy odruch: jeśli agent powtarza ten sam zły ruch, podejrzewaj brak zdolności, nie złe chęci. Dodaj narzędzie, zmniejsz zadanie, wprowadź walidator, a prompt nagle zacznie świecić.

Słowo o testach regresji promptów

Prompt to kod. Zmienisz jeden akapit i coś pęknie dalej. Warto trzymać małą paczkę testów regresji. Mogą to być trzy przykładowe wejścia z oczekiwanym szkieletem wyjścia i podstawową walidacją. Uruchamiaj je przy każdej zmianie promptu. W OpenClaw lub pokrewnych platformach najłatwiej dodać to jako lekki scenariusz, który weryfikuje format i kluczowe warunki DONE. Nie musisz testować jakości treści co do przecinka, wystarczy, że forma i najważniejsze elementy są na miejscu.

Najczęstsze błędy w promptach po polsku

Pierwszy to zbyt ogólne cele. Zrób plan, oceniaj wynik i przedstaw raport to nie jest plan. Drugi to sprzeczne zasady, na przykład bądź zwięzły i opisz każdy krok szczegółowo. Trzeci to brak wskazania narzędzi albo, przeciwnie, zalanie ich opisami. Złoty środek to krótkie, operacyjne charakterystyki i jednozdaniowe uzasadnienie wyboru.

Czwarty błąd to brak formatów. Jeśli nie powiesz agentowi, że chcesz JSON zgodny z tym schematem, dostaniesz coś między listą a esejem. Piąty to nieprzetestowane limity tokenów. Agent w połowie wypowiedzi nagle traci wątek, a ty patrzysz na niespójny raport.

Przydatne mini-wzorce, które można wkleić do swoich promptów

Wzorzec zapytania o brakujące dane:

Jeśli brakuje mi danych krytycznych do zrealizowania celu, zwrócę: "status":"needs_input","questions":[...] I wstrzymam pracę do czasu otrzymania odpowiedzi.

Wzorzec kontroli jakości:

Po wygenerowaniu wyniku porównaj go z kryteriami sukcesu. Jeśli warunek nie jest spełniony, popraw wynik i zaznacz, co zostało zmienione.

Wzorzec streszczenia:

Jeśli odpowiedź przekroczy 800 słów, dodaj na początku streszczenie do 120 słów o tytule "Szybki skrót".

Te krótkie klauzule wprowadzają porządek bez rozdymania promptu.

Gdzie wpleść specyfikę OpenClaw

OpenClaw, jak wiele platform agentowych, zwykle ma miejsca na: system prompt, opis narzędzi, pamięć i zasady planowania. Wykorzystaj to rozdzielenie. Trzymaj rolę i zasady w system prompt, narzędzia opisuj w sekcji narzędziowej z krótkimi przykładami, a definicję DONE włóż tam, gdzie platforma przewiduje metryki sukcesu. Jeśli zachowasz separację, łatwiej testować zmiany i szybciej dojrzysz, co psuje wynik.

Dopytaj też, jak OpenClaw obsługuje walidację formatu odpowiedzi. Jeśli możesz, wyegzekwuj JSON lub YAML przez schemat, a nie tylko prośbę w tekście. Im więcej mechanicznych bezpieczników, tym mniej błędów.

Szybkie odpowiedzi na najczęstsze pytania

Czy pisać prompty po polsku czy po angielsku? Jeśli docelowy język to polski, pisz po polsku, ale nazwy narzędzi i parametry zostaw po angielsku. Daje to najlepszą mieszankę precyzji i naturalności.

Czy lepiej dać długi kontekst, czy krótkie wycinki? Krótkie wycinki. Duży kontekst kosztuje i rozmywa cel. Trafne fragmenty robią lepszą robotę.

Jak często zmieniać prompt? Gdy pojawi się nowy błąd lub nowa potrzeba. Z każdą zmianą odpal testy regresji. I trzymaj historię zmian jak w kodzie.

Temperatura na zero rozwiązuje problemy? Nie. Zbyt niska temperatura czasem usztywnia model i pogarsza radzenie sobie z niejednoznacznościami. Lepiej dobrać ją do zadania.

Czy da się mieć jeden uniwersalny prompt do wszystkiego? Nie. Lepiej mieć dwa lub trzy wyspecjalizowane, niż jeden przeładowany regułami.

Esencja, która robi różnicę

Prompty agentowe to kontrakty, nie wiersze. W OpenClaw po polsku stawiasz na cztery filary: zdefiniowaną rolę i cel, opis narzędzi z ograniczeniami, twardą definicję DONE oraz walidowalny format wyjścia. Dodaj czułe ustawienie budżetu i temperatury, używaj pamięci oszczędnie, i dopisz kilka drobnych klauzul operacyjnych. Efekt jest natychmiastowy: mniej pętli, mniej halucynacji, więcej wyników, które można bez wstydu zintegrować z resztą systemu. Jeśli masz wrażenie, że agent dalej tańczy własne tango, wróć do krótkiego audytu, popraw narzędzia, podaj twardsze kryteria i włącz auto-review. I pamiętaj, że openclaw po polsku to nie tylko kwestia języka. To kwestia nawyku pisania jasnych instrukcji, które nawet bardzo bystry agent nie może źle zrozumieć.

Na tym właśnie polega praktyka: mniej magii, więcej klarownych decyzji. A kiedy już wejdziesz w ten rytm, prompty w openclaw stają się nudne w najlepszym sensie tego słowa. Działają. I o to w gruncie rzeczy chodzi.