Agent AI: czym naprawdę jest cyfrowy pracownik i czym różnią się OpenClaw oraz Hermes

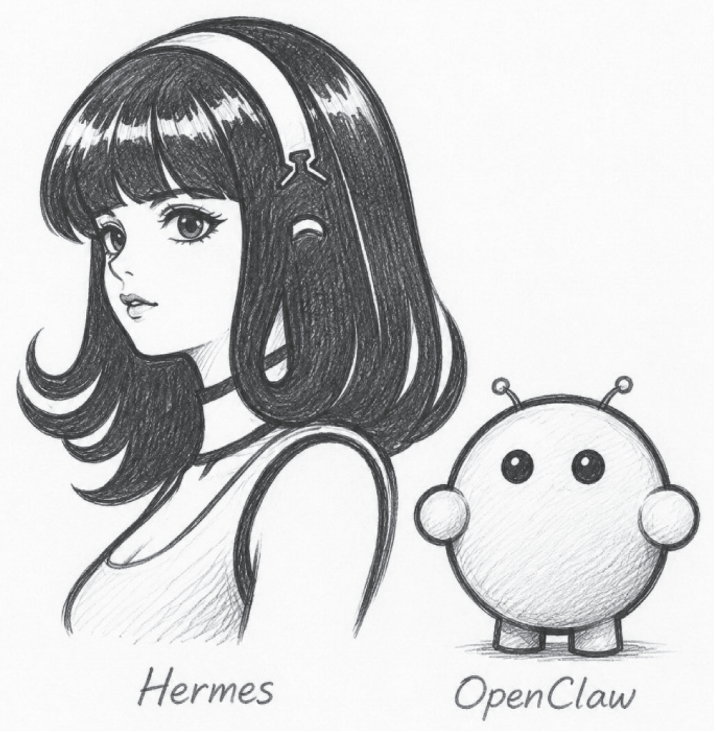
Większość osób zetknęła się już z ChatGPT, Gemini albo Claude. Wpisujemy pytanie, otrzymujemy odpowiedź i na tym sprawa się kończy. Taki system potrafi napisać e-mail, streścić dokument, wyjaśnić trudny temat albo przygotować plan podróży. Nadal jednak czeka na człowieka. Sam z siebie nie otworzy skrzynki pocztowej o ósmej rano, nie sprawdzi nowych faktur, nie zapisze załączników na dysku i nie wyśle informacji, że wykrył problem.

Chatbot odpowiada. Agent działa

Agent AI działa inaczej. Nie jest tylko rozmówcą. Jest programem, który potrafi otrzymać cel, zaplanować kolejne kroki, użyć narzędzi i wykonać zadanie. Może działać na serwerze przez całą dobę. Może korzystać z poczty, kalendarza, przeglądarki, plików, komunikatora, bazy danych albo firmowego systemu. Może też wrócić do użytkownika z wynikiem bez czekania na kolejne pytanie.

Najprostszy przykład wygląda tak. Zwykłemu chatbotowi trzeba napisać: „Przejrzyj te trzy wiadomości i przygotuj odpowiedź". Agentowi można wydać stałą instrukcję: „Codziennie o 8:00 sprawdź nowe wiadomości od klientów, oznacz pilne, przygotuj projekty odpowiedzi i wyślij mi podsumowanie na Telegramie". Chatbot wykonuje pojedynczą rozmowę. Agent realizuje proces.

Różnicę dobrze widać również na przykładzie kalendarza. Chatbot, np. ChatGPT, może podpowiedzieć, jak napisać zaproszenie na spotkanie. Agent może otworzyć kalendarz, sprawdzić wolne terminy, znaleźć godzinę pasującą uczestnikom, przygotować wydarzenie i poprosić człowieka tylko o zatwierdzenie. Nie oznacza to, że agent jest samodzielnym sztucznym człowiekiem zamkniętym w komputerze. To nadal oprogramowanie. Jego zdolność analizowania poleceń pochodzi z modelu językowego, a możliwość działania z podłączonych narzędzi. Gdy agent ma dostęp do Gmaila, może czytać i wysyłać wiadomości. Gdy ma dostęp do Kalendarza Google, może sprawdzać terminy.



Gdy ma dostęp do serwera, może uruchamiać skrypty. Bez takich połączeń pozostaje głównie chatbotem z lepszą pamięcią.

Z czego składa się agent AI

Pierwszym elementem agenta jest model językowy, czyli jego silnik decyzyjny. Może to być model OpenAI, Anthropic, Google, DeepSeek albo model sztucznej inteligencji uruchomiony lokalnie na własnym komputerze. Agent wysyła do niego opis zadania, aktualny kontekst i wyniki wcześniejszych działań. Model językowy ocenia, co należy zrobić dalej.

Załóżmy, że agent otrzymuje polecenie: „Znajdź najnowszą fakturę od firmy transportowej i zapisz jej dane w arkuszu". Model może ustalić, że najpierw trzeba otworzyć skrzynkę pocztową, następnie wyszukać wiadomości od dostawcy, sprawdzić daty, pobrać załącznik, odczytać numer faktury i na końcu wpisać dane do arkusza. Model wybiera kroki, ale każdy krok wykonuje odpowiednie narzędzie.

Drugim elementem jest pamięć. Zwykła rozmowa ze zwykłym chatbotem, czyli naszym przysłowiowym Chatem GPT z telefonu przypomina spotkanie z konsultantem, który po pewnym czasie zapomina szczegóły. Agent może przechowywać informacje w plikach albo bazie danych. Może pamiętać, coś co usłyszał dwa miesiące temu, że użytkownik nie chce spotkań przed dziewiątą, że faktury mają trafiać

do konkretnego folderu, a raport sprzedaży powinien zawierać trzy ustalone wskaźniki.

Trzecim elementem są narzędzia. Sam model językowy, albo prościej sam ChatGPT, nie wysyła e-maila i nie rezerwuje spotkania. Może jedynie wygenerować tekst. Dopiero agent wyposażony w odpowiednie połączenie z pocztą lub kalendarzem wykonuje realną operację. To różnica podobna do różnicy między doradcą a pracownikiem. Doradca podpowiada. Pracownik otwiera system i robi to, co trzeba.

Czwartym elementem jest mechanizm uruchamiania zadań. Agent może działać na żądanie, według harmonogramu albo po wykryciu zdarzenia. Harmonogram może uruchamiać raport codziennie o 7:30. Zdarzeniem może być pojawienie się nowej wiadomości, pliku albo zgłoszenia klienta. Dzięki temu agent nie musi czekać, aż człowiek otworzy okno rozmowy.

Piątym elementem są zasady bezpieczeństwa. Agent powinien wiedzieć, czego nie wolno mu robić. Może mieć prawo do przygotowania projektu wiadomości, ale nie do jej wysłania. Może przenosić pliki, ale nie usuwać ich bez potwierdzenia. Może rezerwować spotkania tylko w dni robocze i tylko między dziewiątą a siedemnastą. Bez takich granic nawet poprawnie działający system może popełnić kosztowny błąd.

Prosty dzień pracy z agentem

Wyobraźmy sobie właściciela małej firmy. Rano w jego skrzynce znajduje się trzydzieści wiadomości. Część to reklamy, część to pytania klientów, a kilka dotyczy zaległych płatności. Zwykły chatbot pomoże dopiero wtedy, gdy użytkownik skopiuje do niego treść wiadomości. Agent może sam sprawdzić skrzynkę, rozpoznać temat każdej wiadomości, przenieść spam, przygotować odpowiedzi i oznaczyć sprawy wymagające decyzji człowieka.

O godzinie dziesiątej agent może pobrać dane sprzedażowe z systemu księgowego i porównać je z poprzednim tygodniem. Gdy zauważy spadek liczby zamówień, może przygotować krótkie wyjaśnienie: „Sprzedaż spadła o 12%. Największy spadek dotyczy produktów ogrodowych. W tym tygodniu nie uruchomiono kampanii, która działała tydzień wcześniej".

Ciekawe dlaczego firmy nagle przestały szukać masowo analityków danych?

Po południu agent może sprawdzić kalendarz i znaleźć wolny termin na spotkanie z klientem. Może przygotować zaproszenie, ale wysłać je dopiero po zatwierdzeniu. Może je też wysłać bez zatwierdzenia. Wieczorem zapisze wykonane działania w pamięci i utworzy krótkie podsumowanie dnia. Może prowadzić masową wysyłkę bez pytania, może dawać nam swój plan działania do zatwierdzenia. Wszystko tak jak prawdziwy pracownik.

Inny przykład dotyczy osoby szukającej pracy. Agent może sprawdzać nowe wiadomości od rekruterów, wyciągać z nich nazwę stanowiska, stawkę, tryb pracy i wymagania. Następnie zapisuje informacje w arkuszu i przygotowuje odpowiedź. Użytkownik nie musi ręcznie przepisywać danych z kilkunastu wiadomości. Nadal jednak decyduje, czy odpowiedź zostanie wysłana.

To nie jest magia. Każdy krok wymaga konfiguracji. Trzeba podłączyć pocztę, kalendarz i dane. Trzeba ustalić reguły. Trzeba także zdecydować, które działania agent może wykonywać samodzielnie, a które wymagają zgody człowieka.

OpenClaw: agent oparty na gotowych umiejętnościach

OpenClaw jest pomyślany jako osobisty asystent AI, którego można rozbudować za pomocą gotowych umiejętności. Jego mocną stroną jest duża ilość darmowych dodatków. Według analizowanego opracowania katalog Claw Hub zawierał ponad 50 tys. umiejętności. Zamiast budować każdą integrację od początku, użytkownik może zainstalować gotowe połączenie z pocztą, kalendarzem, dyskiem, generatorem obrazów, systemem CRM albo inną usługą.

Dla początkującego jest to wygodne. Użytkownik nie musi od razu rozumieć kodu, protokołów i interfejsów API. Może uruchomić panel internetowy, wybrać funkcję i skonfigurować dostęp. OpenClaw przypomina pod tym względem smartfon. Sam telefon jest platformą, ale prawdziwe możliwości pojawiają się po zainstalowaniu aplikacji ze sklepu Play.

Zachowanie OpenClaw może być opisane w zwykłych plikach tekstowych. Plik „agents.md" zawiera reguły działania. Można w nim zapisać, że agent nie wysyła wiadomości po godzinie osiemnastej albo nie usuwa dokumentów bez potwierdzenia. Plik „soul.md" określa jego sposób komunikacji. Można tam ustalić, że odpowiedzi mają być krótkie, rzeczowe i pozbawione przesadnie uprzejmych formuł. Plik „user.md" przechowuje informacje o użytkowniku, jego zwyczajach i pracy. Agent może zapamiętać, że właściciel firmy nie organizuje spotkań w poniedziałki rano, a raporty finansowe chce otrzymywać w formacie PDF. Z kolei „memory.md" może zawierać informacje o wcześniejszych rozmowach i wykonanych zadaniach.

Takie rozwiązanie jest proste i konkretne. Gdy agent zachowuje się źle, można otworzyć odpowiedni plik i poprawić zapis. Jeżeli zaczyna przygotowywać zbyt długie wiadomości, w pliku dotyczącym stylu można wpisać: „Odpowiedź nie może przekraczać pięciu zdań". Nie trzeba przebudowywać całego systemu.

OpenClaw może działać aktywnie. Nie musi czekać na pytanie. Może sprawdzać nowe informacje według harmonogramu i pierwszy napisać do użytkownika. O godzinie 7:45 może wysłać na Telegramie komunikat: „Masz

dziś trzy spotkania. Pierwsze zaczyna się o 9:30. W skrzynce są dwie pilne wiadomości od klientów. Przygotowałem odpowiedzi do zatwierdzenia". Może też sprawdzić ceny wybranych produktów, status przesyłki, prognozę pogody albo stan firmowej strony internetowej. Jeżeli wykryje problem, wysyła ostrzeżenie. Człowiek nie musi pamiętać o regularnym uruchamianiu kontroli.

Wygoda ma jednak cenę. Duży katalog dodatków oznacza dużą liczbę elementów, którym trzeba zaufać. Każda wtyczka może zawierać błędy. Złośliwy dodatek może próbować wykraść dane albo wykonać niepożądaną operację. Początkujący użytkownik może zobaczyć popularną nazwę, kliknąć „instaluj" i nie sprawdzić, kto stworzył dany moduł. W przypadku agenta z dostępem do poczty i plików jest to poważne ryzyko.

Hermes: agent, który zapamiętuje sposób pracy

Hermes reprezentuje inną filozofię. Zamiast opierać się przede wszystkim na wielkim katalogu gotowych dodatków, stawia na uczenie się powtarzalnych sposobów pracy. Jego ważnym elementem jest mechanizm Curator. Gdy agent rozwiąże złożone zadanie, może przekształcić wykonany proces w prostszą i tańszą umiejętność do ponownego użycia.

Załóżmy, że użytkownik co tydzień pobiera plik CSV ze sprzedażą, usuwa puste wiersze, grupuje wyniki według regionów i przygotowuje raport PDF. Za pierwszym razem Hermes może wykonać wiele kroków z pomocą modelu językowego. Model analizuje polecenie, sprawdza dane i decyduje, co zrobić. To zużywa tokeny. Gdy proces zaczyna się powtarzać, Curator może zapisać go jako gotowy skrypt. Przy kolejnym uruchomieniu agent nie musi ponownie analizować każdego kroku. Uruchamia sprawdzoną procedurę. Model językowy zostaje wykorzystany tylko wtedy, gdy pojawi się nietypowa sytuacja, np. zmieni się format pliku albo zabraknie jednej z wymaganych kolumn.

To ważna różnica. Model językowy, z którym łączy się agent jest droższy niż zwykły skrypt z chatGPT. Polecenie „usuń puste wiersze z kolumny adres" nie wymaga za każdym razem zaawansowanego rozumowania. Po zapisaniu procedury komputer może wykonać operację szybko i tanio. Tanio, bo wymagać będzie mniejszej ilości tokenów.

Hermes może również uczyć się nawyków użytkownika. Nie chodzi tylko o pamiętanie faktów. Agent może zauważyć, że użytkownik zawsze chce raport w tym samym układzie, że daty zapisuje w formacie dzień-miesiąc-rok, a pliki archiwizuje w folderach nazwanych numerem tygodnia. Z czasem coraz mniej rzeczy trzeba tłumaczyć od początku.

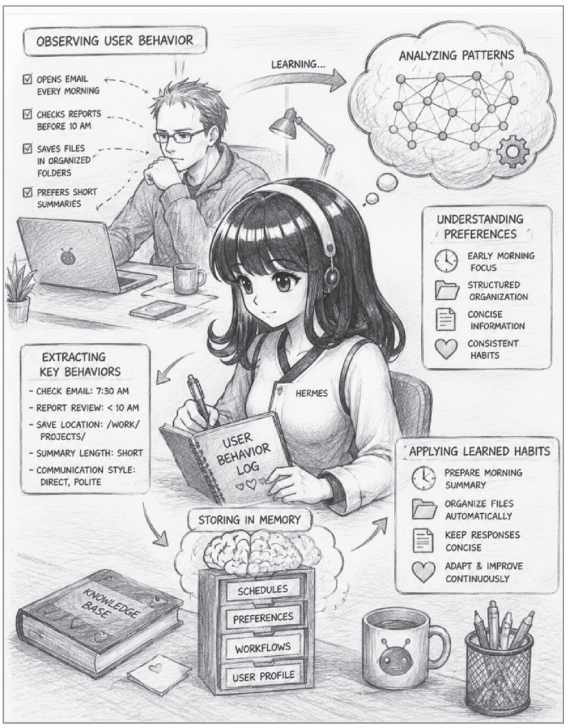
Przykładem może być analiza ofert pracy. Za pierwszym razem użytkownik dokładnie opisuje, że interesują go stanowiska AI Engineer, praca zdalna i stawka powyżej określonego poziomu. Hermes obserwuje sposób oceny ofert. Po pewnym czasie potrafi sam przygotować tabelę, odrzucić niedopasowane propozycje i oznaczyć te, które spełniają kryteria.

Hermes jest bardziej techniczny w obsłudze. Jego naturalnym środowiskiem jest wiersz poleceń, nazywany CLI. Dla osoby, która nigdy nie korzystała z terminala, czarne okno z tekstem może być mniej przyjazne niż panel internetowy OpenClaw. Nie oznacza to, że Hermes jest przeznaczony wyłącznie dla programistów. Początkujący może jednak potrzebować pomocy przy instalacji i pierwszej konfiguracji.

Dwie różne filozofie

OpenClaw mówi: „Wybierz gotową umiejętność z dużego katalogu". Hermes mówi: „Pokaż mi proces, a nauczę się go powtarzać". Oba podejścia mają sens.

OpenClaw będzie naturalnym wyborem dla osoby, która chce szybko podłączyć kilka popularnych usług. Gdy potrzebne jest połączenie z Gmailem, Kalendarzem Google, Dyskiem i komunikatorem, gotowe moduły skracają drogę. Użytkownik nie musi samodzielnie pisać każdego skryptu. Hermes będzie atrakcyjny tam, gdzie firma ma własne, powtarzalne procedury.



Przykładem może być codzienna kontrola plików z produkcji, cotygodniowe czyszczenie danych, generowanie raportu w ustalonym formacie albo sprawdzanie konkretnych stron internetowych. Agent uczy się przebiegu pracy i z czasem wykonuje ją coraz taniej.

Różnica przypomina wybór między dużym marketem budowlanym a warsztatem. W markecie można szybko kupić gotowy element. W warsztacie można stworzyć narzędzie dopasowane do własnego procesu. Market daje szybkość i wybór. Warsztat daje dopasowanie. OpenClaw daje większy wybór dostawców modeli AI i gotowych integracji. Hermes koncentruje się na mniejszej liczbie połączeń, ale mocniej rozwija mechanizm samodzielnego utrwalania procedur. OpenClaw częściej wymaga zatwierdzenia nowej umiejętności przez użytkownika. Hermes może optymalizować ją automatycznie. Nie ma tu jednego zwycięzcy. Użytkownik, który potrzebuje graficznego panelu i szybkich integracji, może lepiej czuć się z OpenClaw. Użytkownik, który akceptuje terminal i chce automatycznie utrwalać własne procedury, może wybrać Hermesa.

Pamięć, czyli dlaczego agent nie zaczyna ciągle od zera

Pamięć jest jednym z najważniejszych elementów agenta. Bez niej każda rozmowa zaczynałaby się od ponownego

tłumaczenia zasad. Agent nie wiedziałby, jak nazywają się klienci, gdzie trafiają raporty, jaki format ma faktura ani jakiego stylu odpowiedzi oczekuje użytkownik. Pamięć nie powinna jednak oznaczać zapisywania wszystkiego. Gdy agent przechowuje tysiące stron historii i wysyła je do modelu przy każdym zadaniu, rosną koszty. Model musi przetworzyć więcej tokenów, nawet gdy większość danych nie ma związku z bieżącą sprawą.

OpenClaw może porządkować pamięć w okresach bezczynności. Mechanizm ten bywa określany jako „dreaming", czyli śnienie. System przegląda logi, wyciąga ważne fakty i usuwa zbędne szczegóły. Można to porównać do pracownika, który pod koniec dnia przepisuje chaotyczne notatki do krótkiego dokumentu. Zamiast przechowywać całą rozmowę o wyborze terminu, agent zapisuje tylko wynik: „Spotkania z klientem X najlepiej planować we wtorki po 12:00". Przy kolejnym zadaniu nie musi analizować kilkudziesięciu wcześniejszych wiadomości.

Dobrze zaprojektowana pamięć jest krótka, konkretna i możliwa do poprawienia. Źle zaprojektowana jest zbiorem przypadkowych zdań. Agent może wtedy wyciągać błędne wnioski. Jeżeli raz użytkownik poprosi o formalny e-mail, agent nie powinien uznać, że wszystkie przyszłe wiadomości mają być formalne. Pamięć musi odróżniać stałą regułę od jednorazowej sytuacji.

Gdzie uruchomić agenta

Agent może działać na zwykłym komputerze, na osobnym urządzeniu albo na serwerze VPS. Każda opcja ma konkretne konsekwencje.

Instalacja na własnym laptopie jest najtańsza, ponieważ nie płaci się za hosting. Problem pojawia się po zamknięciu pokrywy. Agent przestaje działać. Nie sprawdzi poczty w nocy, nie wykona zadania rano i nie odpowie na wiadomość z Telegramu.

Drugi problem dotyczy bezpieczeństwa. Agent uruchomiony na głównym komputerze może otrzymać dostęp do prywatnych dokumentów, zdjęć, haseł zapisanych w przeglądarce albo urządzeń w domowej sieci. Błąd agenta lub luka w zainstalowanym dodatku może wtedy doprowadzić do poważnych szkód.

Osobny stary laptop, Raspberry Pi albo niewielki komputer rozwiązuje część problemu. Agent może działać bez dostępu do głównego komputera użytkownika. Nadal jednak zależy od prądu, domowego internetu i samodzielnej administracji. Po awarii routera albo aktualizacji systemu może przestać działać.

VPS jest wirtualnym serwerem wynajmowanym w centrum danych. Działa przez całą dobę i jest oddzielony od komputera użytkownika. Agent może być uruchomiony w kontenerze Docker, czyli w ograniczonym środowisku. Gdy coś pójdzie źle, skutki łatwiej zamknąć w obrębie jednego serwera. Dla początkującego VPS często jest rozsądnym kompromisem. Kosztuje niewiele, jest dostępny stale i oddziela eksperyment od prywatnego komputera. Nie zwalnia to jednak z aktualizacji, kopii zapasowych i ochrony kluczy API.

Ile naprawdę kosztuje agent AI

OpenClaw i Hermes są projektami open source. Samo pobranie programu nie wymaga opłaty licencyjnej. Nie oznacza to jednak, że agent jest darmowy. Koszt powstaje przede wszystkim w dwóch miejscach: na serwerze i przy korzystaniu z modelu językowego. Najprostszy VPS zwykle kosztuje kilka dolarów miesięcznie. Do lekkich zadań nie potrzeba potężnej maszyny. Agent, który czyta pocztę, uruchamia proste skrypty i korzysta z modelu przez API, może działać na małym serwerze. W praktycznym budżecie warto przyjąć ok. 5-10 dolarów miesięcznie za infrastrukturę. Można zejść niżej w promocji, ale nie należy budować kalkulacji na cenie obowiązującej tylko przez pierwszy miesiąc.

Druga część kosztu to tokeny. Token jest fragmentem tekstu przetwarzanym przez model. Im dłuższe polecenia, większa pamięć i więcej odpowiedzi, tym wyższy rachunek. Tani model może kosztować bardzo mało przy prostych zadaniach. Model klasy premium używany przez cały dzień może kosztować wielokrotnie więcej niż serwer.

Lekko używany agent może zamknąć się w kilku dolarach za modele miesięcznie. Przykładem jest codzienny raport z pięciu stron, krótkie podsumowanie poczty i kilka rozmów na Telegramie. Przy takim obciążeniu łączny koszt serwera i modelu może wynosić ok. 8-20 dolarów miesięcznie. Intensywnie używany agent kosztuje więcej. Gdy co kilka minut sprawdza skrzynkę, analizuje długie dokumenty, korzysta z drogiego modelu i prowadzi rozbudowaną pamięć, rachunek może szybko wzrosnąć do kilkudziesięciu albo kilkuset dolarów. Nie ma tu górnej granicy. Agent może wywoływać model setki razy dziennie.

OpenClaw nie pobiera opłaty licencyjnej, ale jego rozbudowany ekosystem zachęca do instalowania wielu funkcji. Każda z nich może wykonywać dodatkowe zapytania. Szczególnie kosztowne jest częste sprawdzanie stanu systemu, nazywane heartbeat. Jeżeli agent co pięć minut pyta model, czy wydarzyło się coś ważnego, wykonuje 288 kontroli na dobę. Nawet tanie pojedyncze zapytanie staje się kosztowne po przemnożeniu przez cały miesiąc.

Hermes również nie pobiera opłaty za sam program. Jego przewaga kosztowa pojawia się przy powtarzalnych zadaniach. Curator może zamienić proces oparty na wielu wywołaniach modelu w prosty skrypt. Pierwsze wykonania mogą kosztować podobnie jak w OpenClaw, ale później rutynowa praca może być tańsza.

Realny przykład wygląda tak. Mała firma uruchamia OpenClaw na VPS kosztującym 7 dolarów miesięcznie. Agent raz dziennie analizuje pocztę, raz w tygodniu tworzy raport i kilka razy dziennie odpowiada na polecenia. Tani model generuje rachunek na poziomie 5 dolarów. Całość kosztuje ok. 12 dolarów miesięcznie. Gdy firma przełączy wszystkie zadania na model premium i doda kontrolę skrzynki co pięć minut, ten sam OpenClaw może kosztować kilkadziesiąt dolarów miesięcznie. Przy bardzo dużej liczbie dokumentów koszt może przekroczyć 100 dolarów. W przypadku Hermesa początkowy koszt może wyglądać podobnie. Firma płaci średnio miesięcznie 7 dolarów za VPS i kilka dolarów za model. Różnica pojawia się później. Jeżeli Hermes zapisze cotygodniowe przetwarzanie pliku jako gotowy skrypt, model językowy nie musi za każdym razem analizować całej procedury. Rachunek za powtarzalne zadanie może więc spaść.

Nie oznacza to, że Hermes zawsze będzie tańszy. Jeżeli użytkownik codziennie zleca mu inne, skomplikowane analizy, Curator nie będzie miał wielu okazji do wykorzystania zapisanych, powtarzalnych procedur. W takim przypadku koszt zależy przede wszystkim od wybranego modelu i długości przetwarzanych materiałów.

W materiałach dotyczących OpenClaw pojawia się również zalecenie początkowego doładowania konta dostawcy modelu kwotą ok. 40 lub 50 dolarów. Nie jest to abonament za OpenClaw. Są to środki przeznaczone na przyszłe zużycie API i uzyskanie wyższych limitów.

Najważniejsza zasada kosztowa jest prosta. Drogi model powinien wykonywać tylko zadania wymagające trudnego rozumowania. Do sortowania wiadomości, zmiany nazw plików i prostych raportów wystarczy model tańszy albo zwykły skrypt. W przeciwnym razie firma używa kosztownego eksperta do pracy, którą może wykonać kalkulator.

Harmonogram czy ciągłe sprawdzanie?

Agent może działać według stałego harmonogramu albo stale kontrolować czy pojawiło się coś nowego. Stały harmonogram jest tańszy i bardziej przewidywalny. Polecenie: „sprawdź skrzynkę o 8:00, 12:00 i 16:00" uruchamia trzy zadania dziennie. Ciągłe sprawdzanie daje szybszą reakcję, ale łatwo je źle skonfigurować. Polecenie: „sprawdzaj co minutę" oznacza 1440 uruchomień na dobę. Jeżeli każde uruchomienie angażuje model językowy, rachunek rośnie nawet wtedy, gdy nic się nie wydarzyło.

Początkujący powinien zaczynać od harmonogramów. Raport poranny nie musi być sprawdzany co minutę. Faktury można kontrolować raz na godzinę. Stan strony internetowej może sprawdzać prosty skrypt, a model językowy powinien zostać uruchomiony dopiero wtedy, gdy skrypt wykryje błąd. Dobre rozwiązanie oddziela wykrywanie zdarzenia od jego analizy. Prosty i tani mechanizm sprawdza czy pojawił się nowy plik. Dopiero po znalezieniu pliku uruchamia model AI. Dzięki temu agent nie wydaje pieniędzy na tysiące odpowiedzi typu „nic nowego".

Bezpieczeństwo nie jest dodatkiem

Agent z dostępem do poczty, kalendarza i plików ma realną władzę. Może wysłać wiadomość do klienta, usunąć dokument albo ujawnić poufne dane. Dlatego bezpieczeństwo nie może być ostatnim etapem projektu. Najpierw trzeba ograniczyć uprawnienia. Agent do porządkowania faktur nie potrzebuje dostępu do całego Dysku Google. Wystarczy jeden folder. Agent przygotowujący odpowiedzi nie musi mieć prawa do ich automatycznego wysyłania. Może tworzyć szkice. Drugą zasadą jest izolacja. Agent uruchomiony na osobnym VPS nie powinien mieć bezpośredniego dostępu do prywatnego laptopa. Gdy zostanie zaatakowany albo wykona błędne polecenie, zakres szkody będzie mniejszy. Trzecia zasada dotyczy kluczy API. Klucz jest odpowiednikiem hasła do usługi. Nie powinien znajdować się w wiadomości, publicznym pliku ani kodzie przesłanym do repozytorium. Powinien być zapisany w chronionych zmiennych środowiskowych na serwerze. Czwarta zasada dotyczy dodatków. OpenClaw daje dostęp do ogromnego katalogu umiejętności, ale nie każda wtyczka jest godna zaufania. Im większy ekosystem, tym większa powierzchnia ataku. Instalowanie przypadkowego dodatku do agenta jest poważniejsze niż instalowanie prostej wtyczki do edytora tekstu, ponieważ agent może posiadać szerokie uprawnienia. Hermes ma mniejszy ekosystem zewnętrznych dodatków, co ogranicza część ryzyka. Z drugiej strony jego mechanizm automatycznego tworzenia umiejętności może być mniej przewidywalny dla początkującego. System może sam zoptymalizować procedurę. To wygodne, ale wymaga logów, kontroli zmian i możliwości cofnięcia błędnej wersji. Piąta zasada jest najważniejsza: na początku nie podłącza się konta bankowego, menedżera haseł ani systemu, którego błąd może spowodować dużą stratę. Pierwszy agent powinien pracować na danych testowych, kopiach dokumentów i ograniczonych kontach.

Warto również wymagać potwierdzenia działań destrukcyjnych. Agent może zaproponować usunięcie pliku, ale nie powinien wykonywać tej operacji samodzielnie. Może przygotować przelew do sprawdzenia, ale nie powinien go zatwierdzać. Może znaleźć lot, ale nie powinien kupować biletu bez przedstawienia ceny i uzyskania zgody.

Proste zadania na początek

Dobry pierwszy projekt nie powinien samodzielnie podejmować ważnych decyzji. Może porządkować pliki, przygotowywać projekty odpowiedzi, streszczać raporty albo pilnować terminów.

Przykładem jest agent do faktur. Raz dziennie sprawdza wskazany folder, odczytuje nazwy dostawców i daty, zmienia nazwy plików według ustalonego schematu i przygotowuje tabelę. Nie wykonuje płatności. Nie usuwa dokumentów. Gdy nie jest pewny wyniku, przenosi plik do folderu „do sprawdzenia". Inny przykład to agent rekrutacyjny. Czyta nowe wiadomości od rekruterów, wyciąga nazwę stanowiska, stawkę, tryb pracy i wymagania, a następnie zapisuje dane w arkuszu. Może przygotować odpowiedź, ale nie wysyła jej bez zgody.

Agent domowy może codziennie rano sprawdzić pogodę, kalendarz i czas dojazdu. Może napisać: „O 14:00 masz spotkanie w centrum. Przewidywany dojazd zajmie 45 minut. Po południu możliwy jest deszcz". To proste zadanie, ale pokazuje wszystkie cechy agenta: pamięć, harmonogram, narzędzia i aktywne powiadomienie.

Agent w sklepie internetowym może raz dziennie analizować pytania klientów. Jeżeli pytanie dotyczy statusu przesyłki, pobiera dane z systemu logistycznego i przygotowuje odpowiedź. Jeżeli klient składa reklamację, agent nie podejmuje decyzji. Przekazuje sprawę pracownikowi razem z krótkim podsumowaniem.

Agent dla handlowca może rano przygotowywać listę klientów, z którymi należy się skontaktować. Sprawdza datę ostatniej rozmowy, otwarte oferty i historię zakupów. Nie dzwoni samodzielnie. Dostarcza handlowcowi gotowy materiał do pracy.

Który system wybrać?

OpenClaw jest lepszym punktem startowym dla osoby, która chce szybko zobaczyć działający panel, podłączyć popularne usługi i korzystać z gotowych umiejętności. Trzeba jednak ostrożnie

wybierać dodatki i regularnie aktualizować system.

Hermes pasuje do użytkownika, który chce automatyzować własne, powtarzalne procesy i akceptuje bardziej techniczny sposób obsługi. Jego Curator może z czasem obniżyć koszty, ponieważ część pracy zamienia w zwykłe procedury.

Osoba prowadząca małą firmę i potrzebująca szybkiego połączenia z pocztą, kalendarzem i Dyskiem Google może zacząć od OpenClaw. Osoba analizująca co tydzień podobne pliki i generująca raporty według własnego schematu może więcej zyskać dzięki Hermesowi. Dla osoby całkowicie początkującej ważniejszy od wyboru nazwy jest wybór pierwszego zadania. Źle wybrany projekt będzie kosztowny i niebezpieczny niezależnie od platformy. Dobrze wybrany pozwoli zrozumieć działanie agenta bez dużego ryzyka.

Najrozsądniejszy start to osobny VPS, jedno konto testowe, jeden komunikator i jedno proste zadanie uruchamiane o stałej porze. Agent powinien przygotowywać wynik, a człowiek zatwierdzać działania. Dopiero po kilku tygodniach stabilnej pracy można zwiększać uprawnienia.

Agent nie zastępuje odpowiedzialności

Największym błędem jest traktowanie agenta jak nieomylnego pracownika. Model językowy potrafi błędnie zrozumieć polecenie, pomylić dane albo wymyślić brakującą informację. Agent może dodatkowo wykonać ten błąd w prawdziwym systemie. Jeżeli chatbot błędnie napisze odpowiedź, użytkownik może ją poprawić przed wysłaniem. Jeżeli agent ma prawo do samodzielnego wysyłania wiadomości, błąd natychmiast trafia do klienta. Jeżeli ma prawo do usuwania danych, błędna decyzja może doprowadzić do utraty dokumentów.

Dlatego dobry agent nie działa bez granic. Ma jasno opisany cel, ograniczony dostęp, rejestr działań i punkty zatwierdzania. Przy prostych operacjach może pracować samodzielnie. Przy działaniach finansowych, prawnych albo nieodwracalnych powinien zatrzymać się przed wykonaniem i poprosić człowieka o decyzję. To nie ogranicza wartości agentów. Przeciwnie. Dobrze ustawione granice sprawiają, że można im powierzyć więcej rutynowej pracy. Człowiek nie musi kopiować danych, przeglądać tych samych folderów i pisać podobnych odpowiedzi. Zachowuje jednak kontrolę nad tym, co ważne.

Agent na prywatnym serwerze VPS

Instalacja agenta na prywatnym serwerze VPS jest rekomendowanym rozwiązaniem dla większości użytkowników, szczególnie tych początkujących. Źródła wskazują, że jest to najstabilniejszy sposób na posiadanie osobistego asystenta AI.

Oto szczegółowe zestawienie celu takiej instalacji oraz jej zalet.

Głównym celem jest stworzenie autonomicznego asystenta, który pracuje niezależnie od osobistego sprzętu. Dzięki umieszczeniu agenta w chmurze, może on monitorować sieć, zarządzać e-mailami czy wykonywać zaplanowane zadania (cron) przez całą dobę, 7 dni w tygodniu, nawet gdy Twój komputer jest wyłączony. Pozwala to na pełną mobilność – możesz komunikować się z agentem za pomocą telefonu (np. przez Telegram) z dowolnego miejsca na świecie.

Bezpieczeństwo i izolacja (Sandbox): VPS tworzy odizolowane środowisko. Jeśli coś pójdzie nie tak lub agent zostanie zaatakowany, zagrożona jest tylko maszyna wirtualna, a nie Twoje prywatne pliki na domowym laptopie.

Łatwe zarządzanie: w przypadku błędów w konfiguracji możesz łatwo przywrócić system z kopii zapasowej lub wyczyścić serwer i wdrożyć go ponownie w kilka minut.

Jakie są wady tego rozwiązania?

W przeciwieństwie do instalacji na własnym starym laptopie, VPS wiąże się z miesięczną opłatą zazwyczaj od kilku do kilkunastu dolarów.

Złożoność wstępnej konfiguracji: mimo łatwej instalacji samego systemu, nadal musisz samodzielnie pozyskać klucze API od dostawców (np. Anthropic, OpenAI) oraz skonfigurować tokeny dla komunikatorów.

Zależność od dostawcy: korzystasz z infrastruktury firmy trzeciej, co wiąże się z koniecznością akceptacji ich regulaminów i polityki prywatności.

Koszty dodatkowe: poza opłatą za serwer, musisz płacić dostawcom modeli AI za każdy przetworzony token (słowo), co przy źle skonfigurowanych zadaniach automatycznych może generować nieprzewidziane wydatki.

Co naprawdę zmienia agent AI?

Agent AI zmienia sposób korzystania ze sztucznej inteligencji. Chatbot, np. chat GPT, jest narzędziem do rozmowy. Agent staje się częścią procesu. Pamięta zasady, korzysta z usług, uruchamia zadania i wraca z wynikiem.

OpenClaw pokazuje siłę dużego ekosystemu gotowych umiejętności. Hermes pokazuje siłę automatycznego uczenia się powtarzalnych procedur. Pierwszy daje szybszy dostęp do wielu integracji. Drugi może lepiej dopasować się do konkretnego sposobu pracy i obniżać koszt rutynowych zadań. Oba systemy można uruchomić bez opłaty licencyjnej, ale oba generują rachunki za serwer i modele AI. Przy rozsądnym użyciu początkujący może zmieścić się w kwocie poniżej 20 dolarów miesięcznie. Przy złej konfiguracji ten sam agent może zużyć wielokrotnie więcej.

Najważniejsze nie jest więc pytanie, czy agent potrafi działać sam. Potrafi. Najważniejsze pytanie brzmi: jakie dokładnie działania powinien wykonywać, z jakimi uprawnieniami i przy jakim limicie kosztów?

Dobry agent nie jest cyfrowym, wszechwiedzącym pomocnikiem. Jest precyzyjnie skonfigurowanym wykonawcą. Ma jasno określoną pracę, dostęp tylko do potrzebnych narzędzi i obowiązek zatrzymania się tam, gdzie zaczyna się ryzyko. W takiej formie agent AI przestaje być demonstracją technologiczną. Staje się użytecznym elementem codziennej pracy.

Wojciech Moszczyński

Wojciech Moszczyński – absolwent Katedry Ekonometrii i Statystyki Uniwersytetu Mikołaja Kopernika w Toruniu, specjalista z zakresu ekonometrii, finansów, data science oraz rachunkowości zarządczej. Specjalizuje się w optymalizacji procesów produkcyjnych i logistycznych. Prowadzi badania w obszarze rozwoju i zastosowania sztucznej inteligencji. Od lat zaangażowany w popularyzację machine learning oraz data science w środowiskach biznesowych.