n8n, czyli na nowy rok automatyzacja zamiast odchudzania

Nowy Rok zacznijmy od nowych wyzwań. Jakie mogą być wyzwania w rubryce, którą prowadzę? Oczywiście: wprowadzenie sztucznej inteligencji. Od co najmniej 10 lat piszę o tym na łamach „Przeglądu" i gorąco zachęcam, żeby rok 2026 potraktować jako rok sztucznej inteligencji.

Generalnie domyślam się, że taki apel może doprowadzić niejednego czytelnika do irytacji. Niby jak mamy „wprowadzić sztuczną inteligencję"? Jaką sztuczną inteligencję? I o co w tym wszystkim chodzi? Jeden wielki zgiełk, wśród znajomych i w świecie zawodowym, że „my musimy", że „mamy to", że to jest konieczność, że to jest wymóg bycia nowoczesnym, że to jest coś oczywistego w świecie zdominowanym przez AI.

Sztuczna inteligencja – naprawdę musisz?

Więc spieszę donieść: większość osób czytających ten tekst już dawno wprowadziła sztuczną inteligencję do swojego życia. Mamy ją w telefonach: usprawnienia związane z robieniem zdjęć, organizowaniem czasu, wyszukiwaniem, nawigacją itd. Natomiast, żeby wprowadzić sztuczną inteligencję do naszych piekarni, cukierni oraz do sieci sprzedaży, musimy się trochę bardziej postarać. Oczywiście bardzo wielu dostawców już ma do zaoferowania nowoczesne rozwiązania: organizery czasu pracy, planowanie zmian, planowanie produkcji, przewidywanie sprzedaży w sieciach dystrybucji pieczywa.

Tyle, że te aplikacje to często zwykłe, bardzo proste programy komputerowe, które mają połączenie ze sztuczną inteligencją. Pisząc „sztuczna inteligencja", mam na myśli supercentra obliczeniowe, w których „siedzi" jeden z kilku tysięcy modeli językowych, potocznie zwanych sztuczną inteligencją. To są modele typu ChatGPT, Gemini, DeepSeek czy Claude. Napisanie takiej aplikacji bywa dziecinnie proste: wystarczy napisać w Pythonie program, który ma coś wykonywać. Natomiast cały mózg tej aplikacji jest sprzężony z centrum obliczeniowym. A ono każe sobie płacić za tzw. tokeny, czyli za użycie myślenia trzeba płacić. Podobnie jak za użycie prądu płacimy abonament, tak płacimy też abonament za myślenie maszyn.

Nikt już nie produkuje oprogramowania, które ma w sobie własną „inteligencję", rozumianą jako coś naprawdę zaawansowanego. Najnowsze rozwiązania są podpięte pod superinteligentne modele językowe. Tak więc, jeżeli ktoś proponuje państwu „bardzo inteligentny organizer", który planuje kampanię reklamową

i organizuje logistykę transportu, to w praktyce często oznacza mały, łatwy do napisania programik podpięty pod model językowy. A miesięczna rata, którą trzeba płacić za użytkowanie tego rozwiązania, w dużej części jest dalej przekazywana jako abonament za korzystanie z jednego z tych potężnych modeli.

Nie umiemy programować i co nam Pan zrobi?

Idąc dalej tym tokiem rozumowania: dlaczego my – piekarze, cukiernicy – nie możemy sobie sami napisać

takiego programiku, podpiąć się pod sztuczną inteligencję i sami płacić abonament?

Odpowiedź jest uderzająca w swej prostocie: bo nie umiemy programować, nie umiemy organizować oprogramowania. Takie rzeczy robią specjaliści, którzy tworzą podobne rozwiązania od wielu lat i często są po prostu nie do prześcignięcia. Ich zdolności – przynajmniej z boku – wyglądają jak magia, a umiejętność pisania kodu bywa wręcz fascynująca. Niektórzy podejrzewają ich o kontakty z obcą cywilizacją.

I teraz uwaga: zdolności związane z programowaniem są faktycznie trudne do osiągnięcia, nauczenia i czynnego wykonywania. Pisanie kodu bywa smutne, trudne i łatwo się pomylić. Natomiast wszystko zmieniło się w ciągu ostatnich dwóch lat. Wprowadzenie tysięcy modeli językowych zmieniło cały obraz życia programistów. Obecnie modele językowe piszą kod za ludzi. To znaczy: wystarczy powiedzieć, co chcemy, aby program zrobił, a model językowy potrafi napisać całe oprogramowanie.

Niestety nie jest to jeszcze system idealny. Po pierwsze, trzeba umieć napisać projekt, czyli zapytanie – prompt – do modelu językowego. Po drugie, rzadko kiedy model dostarcza idealne rozwiązanie i trzeba dziesiątki razy korygować jego pracę kolejnymi promptami. Ale niewątpliwie nie jest to już sytuacja, w której człowiek samotnie na pustkowiach cyfrowej przestrzeni musi tworzyć coś z niczego, kodować i testować własne rozwiązania. Dziś sporą część tej pracy wykonuje za niego maszyna: model językowy, jeden z setek tysięcy (a czasem mówi się nawet o milionach) dostępnych wariantów.

Gdybym przeczytał to, co teraz napisałem, dwa lata temu, pomyślałbym, że ktoś przesadził z alkoholem albo popadł w jakiś rodzaj choroby psychicznej. Gdybym przeczytał to rok temu, uznałbym, że w społeczeństwie istnieją wizjonerzy, którzy mają wyobraźnię wystarczającą do przewidywania przyszłości. Dzisiaj ten tekst jest po prostu oczywistością w kręgach zawodowych ekspertów IT i data science.

Jeżeli więc założymy, że do tworzenia własnych rozwiązań związanych ze sztuczną inteligencją nie musimy już perfekcyjnie pisać kodu, a wystarczy jakiś zmysł organizacyjny oraz umiejętność efektywnej pracy z modelem, to zaczynamy zbliżać się do obszarów, w których możemy działać sami.

Tyle że… nadal musimy trochę znać kod, trochę poprawiać maszynę, trochę ją testować, wiedzieć o co chodzi i tak dalej. Czyli reasumując: nawet wykorzystując LLM-y, często nie jesteśmy w stanie „od zera" sami wprowadzić do swojej pracy – do własnej piekarni czy cukierni – rozwiązań z zakresu sztucznej inteligencji tak, żeby to było stabilne, bezpieczne i działało codziennie.

Czyli nie jesteśmy zawodową Cepelią

Warto w tym miejscu podkreślić: to nie jest tak, że my jesteśmy zacofani i powinniśmy się tego wstydzić. To nie jest tak, że „powinniśmy już mieć" system sztucznej inteligencji, który pilnuje temperatury na piecach, kontroluje korespondencję, jednocześnie sprawdza rozlokowanie samochodów oraz analizuje sprzedaż, robiąc wszystko to, co podobno „powinna robić sztuczna inteligencja".

Nie jesteśmy zacofani. Jesteśmy w realiach rynku. To, że nie znamy się na sztucznej inteligencji czy na kodowaniu, nie jest naszą słabością – to jest normalność. Gruntowna znajomość tych obszarów jest pracochłonna, wymaga dużych poświęceń i stałego aktualizowania wiedzy. Z drugiej strony wdrożenie takich rozwiązań przez firmę zewnętrzną wiąże się z dużymi kosztami i wcale nie gwarantuje powodzenia. Dlaczego? Bo firmy wdrożeniowe to też ludzie, a na rynku ludzi jest mało. Większość to początkujący, nieopierzeni „eksperci" od sztucznej inteligencji. Reasumując: nawet najbardziej renomowana firma składa się z ludzi, a ludzi brakuje – więc pojawia się wniosek, że sztuczną inteligencję powinniśmy wprowadzać możliwie samodzielnie, choćby częściowo.

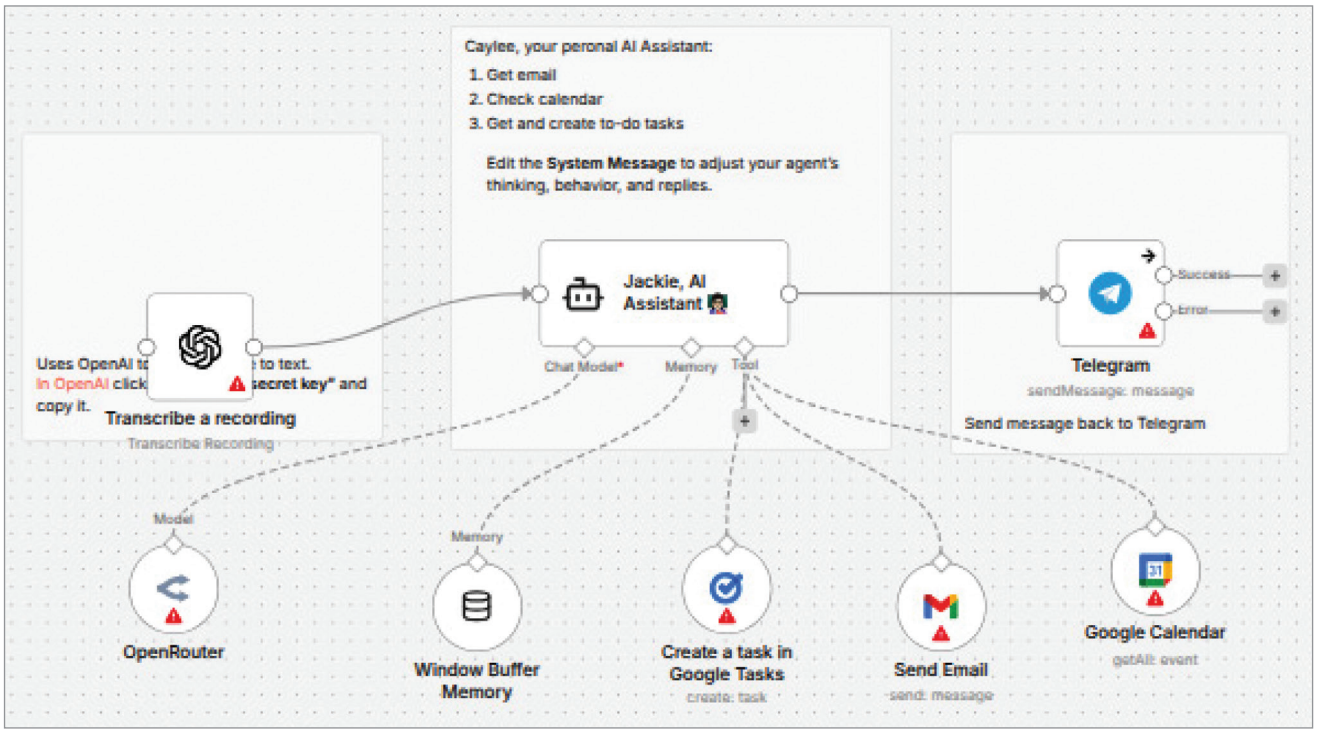
Bez spiny – przychodzi n8n

Zanim rzucisz ten artykuł – co w oczywisty sposób będzie objawem instynktu samozachowawczego – dowiedz się, że nie tylko ty dochodzisz do takich wniosków. Ludzie generalnie nie wdrażają w swoich firmach sztucznej inteligencji, bo widzą bariery związane z kodowaniem i ze „znajomością organizacji kodu". Wolą więc, w natłoku swojej pracy, zlecić to komuś. A ponieważ inni też mają problem ze znalezieniem odpowiednich fachowców, firmy ciągle tkwią ze swoimi rozwiązaniami w XX w. Właściciele często się wstydzą braków i raczej nie afiszują się z tym, że wciąż nie są w stanie wdrożyć niczego z nowoczesnych technologii.

I tutaj pojawia się ciekawy wątek. Pewna firma z Berlina doszła do podobnych wniosków i stworzyła narzędzie, które ma pozwalać wdrażać efektywną sztuczną inteligencję bez znajomości kodu, a nawet bez szczególnej znajomości środowiska IT. To narzędzie nazywa się n8n.

No i teraz zatrzymajmy się na chwilę, bo n8n to nie jest „kolejny program do planowania", tylko pewien sposób myślenia o pracy. n8n jest platformą automatyzacji przepływów pracy (*workflow automation*), w której logika procesu jest budowana jako graf połączonych „węzłów" (*nodes*), uruchamianych zdarzeniowo. Brzmi groźnie, ale chodzi o proste rzeczy: coś się wydarza (ktoś wysłał maila, pojawił się nowy dokument, wpadło zamówienie, ktoś wypełnił formularz), a system reaguje i odpala kolejne kroki.

To narzędzie łączy integracje z usługami zewnętrznymi, przetwarzanie danych oraz elementy programowalne, oferując kompromis pomiędzy podejściem „no-code" a klasycznym kodowaniem. Czyli: możesz klikać i układać,



ale jak chcesz, to możesz też wejść głębiej, dopisać własną logikę, coś poprawić, coś przeliczyć. I to jest ważne, bo w prawdziwym życiu firmy nie działają jak katalogowy przykład z prezentacji. Zawsze jest jakiś wyjątek, jakaś „nasza sytuacja", jakaś specyfika zakładu. n8n jest rozwijane przez spółkę n8n GmbH z siedzibą w Berlinie i jest dostępne zarówno jako usługa chmurowa, jak i do samodzielnego hostowania. Czyli możesz mieć to „u nich", albo możesz mieć to „u siebie w kompie", w swoim środowisku, jak ktoś lubi kontrolę i niezależność.

Dlaczego w ogóle powstała taka klasa narzędzi? Bo współczesne systemy informatyczne składają się z wielu usług naraz: aplikacji SaaS, baz danych, kolejek zdarzeń, repozytoriów plików, narzędzi analitycznych i modeli AI. Integracja tych komponentów bywa robiona ad hoc: ktoś napisze skrypt, ktoś ustawi cron, ktoś doda webhook i jakoś to działa... do momentu aż przestaje działać, a potem nikt nie wie, gdzie jest błąd, czemu to się zatrzymało i kto za to odpowiada. To prowadzi do rosnących kosztów utrzymania, słabej obserwowalności oraz trudności w wersjonowaniu i audycie procesów. I tu wchodzi workflow automation: modelowanie procesu jako jawnego przepływu sterowania i danych, uruchamianego w reakcji na zdarzenia. Coś jak rozpisanie pracy na kartce, tylko że ta kartka jest „żywa" i wykonuje robotę.

Rdzeniem n8n jest edytor przepływów, w którym układasz graf z węzłów pełniących role: wyzwalaczy (triggers), kroków integracyjnych (np. pobranie albo wysłanie danych do usługi), transformacji danych, warunkowania i sterowania przepływem oraz obsługi błędów i ponowień (retry). Przepływy są uruchamiane w środowisku serwerowym n8n, a integracje realizowane są przez gotowe konektory (jest ich dużo) oraz przez generyczne mechanizmy HTTP, czyli w skrócie: „jak nie ma gotowego klocka, to i tak da się dogadać z internetem".

I teraz ważna rzecz: w praktyce n8n pełni rolę warstwy orkiestracji. Zamiast implementować pełne procesy w jednym wielkim programie, opisujesz kolejność kroków, mapowanie pól i warunki przejść. Dostajesz czytelność, łatwiejszą diagnostykę, możliwość ponownego użycia fragmentów, szybsze prototypowanie. I w dodatku dokumentacja i społeczność mocno pchają temat połączenia automatyzacji z komponentami AI, co dziś ma znaczenie, bo ludzie nie chcą tylko „zautomatyzować", oni chcą „zautomatyzować mądrze" i za miesiąc wciąż wiedzieć co zrobili.

Czy n8n jest „dla osób, które nie umieją programować"? Tak i nie. To narzędzie low-code: podstawowe przepływy możesz budować bez pisania kodu, poprzez konfigurację węzłów i mapowanie danych w interfejsie. Ale „brak kodu" nie oznacza „braku myślenia technicznego". Efektywne użycie zwykle wymaga rozumienia takich rzeczy jak HTTP, OAuth (czyli logowanie do usług), modeli danych (JSON), semantyki błędów, limitów API oraz elementarnej inżynierii niezawodności, czyli żeby to działało nie tylko w poniedziałek rano, ale też w piątek wieczorem. W prostych scenariuszach jest to dostępne dla nietechnicznych, natomiast największa przewaga ujawnia się wtedy, kiedy ktoś potrafi świadomie projektować integracje i kontrolować jakość automatyzacji. I to jest uczciwe postawienie sprawy: n8n raczej nie robi cudów bez człowieka, ale usuwa 80% bólu i kosztów pieniężnych, który wcześniej był nie do przejścia.

Do tego dochodzi temat producenta i modelu dystrybucji, bo to też jest pytanie, które słyszę: „czy to jest open source", „czy to jest darmowe", „czy ktoś mnie nie

złapie za abonament". Producentem jest n8n GmbH. Projekt jest rozwijany publicznie (kod jest dostępny), a jednocześnie jest to model licencjonowania określany jako „fair-code" (*Sustainable Use License*). W praktyce oznacza to: można używać, można modyfikować, można budować na tym swoje procesy, ale są ograniczenia w typie „nie buduj konkurencyjnej usługi 1:1 i nie sprzedawaj tego jako własnego n8n". Czyli to nie jest „klasyczny open source w sensie OSI", ale też nie jest to czarna skrzynka, której nie wolno dotknąć. Dla normalnej firmy, która chce automatyzować własną pracę, to jest zwykle wystarczająco proste i zrozumiałe.

Ta firma postanowiła wprowadzić na rynek produkt, który praktycznie każdemu umożliwia tworzenie zaawansowanych systemów automatyzacji – takich „systemów aktywnie działającej sztucznej inteligencji". Każda działalność ma swoje potrzeby i nie będę tu udawał, że z miejsca wymyślę idealny katalog zastosowań dla piekarni. Mogę jedynie przytoczyć przykłady z internetu: ludzie mówią, że n8n pozwoliło im uporządkować spotkania, przesiać wydarzenia w sieci, wybrać najbardziej prawdopodobne tematy do omówienia, zautomatyzować analizę informacji i tak dalej. Rzeczy zaawansowane, a jednocześnie przedstawione w prosty sposób.

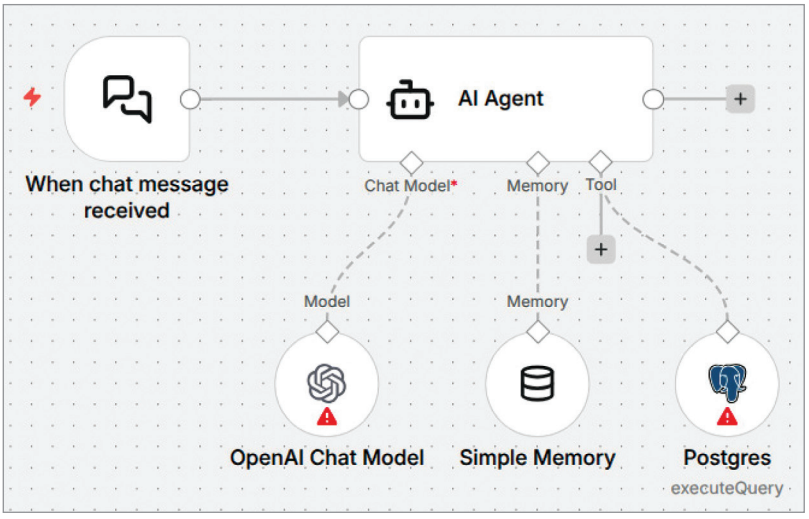
Za chwilę przejdę do meritum i pokażę kilka zrzutów, natomiast powiem od siebie tak: ja ciągle tkwię jeszcze w technologiach sprzed roku czy dwóch. Moje wdrożenia sztucznej inteligencji opierają się na Pythonie. Nowością w mojej pracy jest to, że korzystam z kilku modeli językowych naraz, bo modele różnią się między sobą: jedne są lepsze w rozróżnianiu określonych rzeczy, inne bardziej analityczne, kolejne są po prostu tańsze. W tym tygodniu napisałem rozwiązanie, które – lekko licząc – ma około 400 linijek kodu Pythona.

Poprosiłem więc sztuczną inteligencję, żeby to działające rozwiązanie przeniosła do narzędzia n8n. I okazało się, że moje „złożone" rozwiązanie to zaledwie pięć kwadracików ułożonych w formie schematu. Każdy z tych kwadracików może stworzyć praktycznie każdy: wystarczy kliknąć myszą i wybrać typ elementu. Przykładowo: kwadracik pierwszy to start procesu, drugi to nawiązanie kontaktu z modelem LLM, trzeci to zapis wyniku, czwarty to wysłanie wyniku na skrzynkę, piąty to kontrola błędu albo logowanie. Każde „pudełko" w schemacie trzeba uzupełnić, tam są takie formularze, ale to uzupełnianie bywa dziecinnie proste: w jednym miejscu piszemy modelowi, co ma zrobić, w innym – „przekaż wartość wejściową dalej".

I teraz najważniejsze: każdy, kto ma minimalną wiedzę techniczną (mam na myśli to, że wie, jak działa piec do wypieku bułek, albo rozumie, na czym polega użycie sprzęgła w samochodzie), jest w stanie w krótkim czasie nauczyć się „programować" te kwadraciki. Nazywają się node'ami, czyli węzłami. W środku są instrukcje: możemy napisać je sami, albo może za nas napisać je sztuczna inteligencja – nawet ta, którą mamy dziś w telefonie. A jeżeli ktoś już zna Pythona i ma doświadczenie z aplikacjami, to powiem wprost: to jest narzędzie komplementarne. Python zostaje do rzeczy ciężkich, obliczeń, ETL, modeli ML, usług API, a n8n robi za warstwę „kleju", która spina systemy, wysyła, odbiera, orkiestruje i pilnuje przepływów. To jest dokładnie ta część pracy, której nikt nie lubi robić ręcznie, a która potrafi zjadać tygodnie.

Wdrożenie jest proste: uruchamia się schemat i widać wyraźnie, jak informacja płynie, jak „myśli" model językowy i co się dzieje po kolei. To jest trochę jak budowanie rozwiązań technicznych z klocków LEGO. Oczywiście nikt z ulicy nie zbuduje skomplikowanej maszyny od razu – trzeba się trochę poduczyć, zrozumieć zasady. Ale tym razem wdrożenie sztucznej inteligencji do zakładu produkcyjnego przestaje być abstrakcją, a staje się realną perspektywą.

Co ważne, n8n w podstawowej wersji jest bezpłatne, a koszty pojawiają się zwykle wtedy, kiedy rozwiązanie działa intensywnie, w sposób ciągły, i jest realnie używane w firmie. Mam duże doświadczenie w tworzeniu takich rzeczy i powiem uczciwie: to nie jest „zupełnie proste". Trzeba nauczyć się myśleć trochę inaczej niż w klasycznym programowaniu. Budowanie pojazdów z LEGO też nie jest takie oczywiste, ale daje frajdę – i tutaj bywa podobnie.



Ci, którzy budują automatyzacje zawodowo i używają n8n, często twierdzą, że najlepszy start wygląda tak: sztuczna inteligencja buduje im wstępny schemat, oni pobierają z ekranu gotową definicję (to jest plik w formacie JSON – czasem ktoś to żartobliwie nazywa „dżejson"), wklejają do pliku tekstowego i importują do n8n. Te czynności są łatwiejsze niż instalacja aplikacji w telefonie. Potem uruchamiają rozwiązanie i zaczynają je poprawiać, ulepszać, dostrajać. Zaczynają zabawę z innego poziomu: nie od pustej kartki, tylko od prototypu, który już prawie działa.

Koniec z odchudzaniem od nowego roku?

Jakość pracy przy wdrażaniu sztucznej inteligencji w ostatnim roku zmieniła się radykalnie. Teraz wiele rzeczy stało się łatwiejszych. Poziom wejścia zszedł do czegoś, co przypomina budowanie maszyn z klocków: da się to zrobić bez wieloletniej praktyki w kodowaniu. I jeżeli gdzieś w tym wszystkim jest „nowy rok i nowe wyzwania", to właśnie tutaj: w tym, żeby nie bać się narzędzi, które pozwalają wejść w automatyzację i AI spokojnie, krok po kroku, po co się stresować i napinać. Wystarczy proste postanowienie: mam darmowy n8n, chcę wprowadzić automatyzację kontroli pieca, kontroli tylko temperatury. To jest mój cel – daję sobie na to 40 dni, codziennie po godzinie, zamiast oglądać rolki na telefonie, będę robił minimalne kroczki aby osiągnąć cel. Nasz 4-tygodniowy model może składać się z dwóch połączonych klocków. Ważne, że działa i robi robotę. To dokładnie tyle ile normalnie poświęciłbym czasu na zbudowanie pojazdu elektrycznego z klocków lego. Jeżeli to zadanie się uda dalsze rozszerzanie automatyzacji pójdzie już prosto. To tylko tyle i aż tyle.

Bo może największy paradoks jest taki: my przez lata myśleliśmy, że sztuczna inteligencja to będzie „system za miliony", wdrożenie na pół roku, konsultanci, slajdy, audyty i „zaraz wrócimy do tematu po wakacjach". A tymczasem realna sztuczna inteligencja w firmie, ta codzienna, sprowadza się do tego, czy umiemy połączyć kilka kroków w sensowny proces i czy umiemy go utrzymać w ruchu. I n8n jest właśnie o tym: o tym, żeby proces dało się zobaczyć, zrozumieć, poprawić i – co jest kluczowe – żeby nie był zależny od jednego człowieka, który ma „tajemną wiedzę", tylko od czegoś, co stoi na ekranie i działa jak schemat elektryczny. W mojej głowie to jest różnica między abstrakcją a realnym wdrożeniem. I dlatego uważam, że 2026 to dobry moment, żeby przestać się czaić „czy", a zacząć się coś z tym robić.

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.