PUBLIKACJE / RAPORT Z BADANIA

SZYBKI MODEL JĘZYKOWY W PRAKTYCE: TEST CELERIS-1

Teksty, kod strony internetowej i krótkie zadania

AUTOR

Tomasz Załucki

DATA PUBLIKACJI

17 listopada 2026 r.

CZAS CZYTANIA

13 min

STRESZCZENIE

Raport przedstawia wyniki testu modelu językowego Celeris-1 w pięciu rodzajach zadań: pisaniu długiego tekstu po polsku i po angielsku, tłumaczeniu, generowaniu strony internetowej, poprawianiu błędów w kodzie oraz krótkich zadaniach klasyfikacji i wyciągania danych. Opisuje szybkość i koszt, poprawność językową, zgodność wyników z poleceniem oraz zastosowania, w których model działał bez błędów.

Słowa kluczowe: modele językowe, sztuczna inteligencja, Celeris-1, generowanie tekstu, generowanie kodu, klasyfikacja, ekstrakcja danych

1. NAJWAŻNIEJSZE USTALENIA

Model Celeris-1 odpowiada bardzo szybko: artykuł liczący około 17 tysięcy znaków powstaje w 3 do 5 sekund, kompletna strona internetowa w niecałe 6 sekund, a klasyfikacja jednej wiadomości trwa około 0,4 sekundy. Cały test kosztował około 9 centów.

W krótkich zadaniach o ustalonej formie model był niemal bezbłędny: poprawnie sklasyfikował 72 z 72 wiadomości klientów i poprawnie wyciągnął 240 z 240 pól danych, za każdym razem zwracając poprawny JSON.

Długie teksty pisane po polsku zawierają średnio około 11 nieistniejących słów na 1000. Te same teksty pisane po angielsku nie zawierały ani jednego takiego błędu.

Tłumaczenie poprawnego angielskiego tekstu na polski tym samym modelem nie pomaga: wynik ma około 17 nieistniejących słów na 1000.

Wygenerowana strona internetowa wygląda poprawnie i w większości działa, ale nie spełnia wymagań, które model miał spełnić, w tym kontrastu tekstu. Z 15 zgłoszonych błędów model w pełni naprawił 5, a przy okazji wprowadził 3 nowe.

72/72
poprawnie sklasyfikowanych wiadomości klientów

100%
poprawnie wyciągniętych pól danych (240 z 240)

11
nieistniejących słów na 1000 w długim tekście po polsku

5 z 15
zgłoszonych błędów strony naprawionych w pełni

2. CEL I ZAKRES TESTU

Na rynku pojawiają się modele językowe, których główną zaletą jest szybkość odpowiedzi. Celeris-1 jest jednym z nich: producent opisuje go jako model ogólnego przeznaczenia o inteligencji zbliżonej do GPT-5 i czasie odpowiedzi do 15 razy krótszym[1]. Dla firmy, która rozważa użycie takiego modelu, istotne jest pytanie praktyczne: do jakich zadań można go użyć bez ryzyka błędów, a do jakich nie.

Sprawdziłem model w pięciu rodzajach zadań, od najdłuższych do najkrótszych:

- napisanie artykułu eksperckiego po polsku o zadanej długości, z przypisami i bibliografią,
- napisanie tego samego artykułu po angielsku i przetłumaczenie go na polski,
- zbudowanie kompletnej strony internetowej w jednym pliku,
- poprawienie błędów w tej stronie na podstawie listy z testów,
- krótkie zadania: klasyfikacja wiadomości, wyciąganie danych i proste obliczenia.

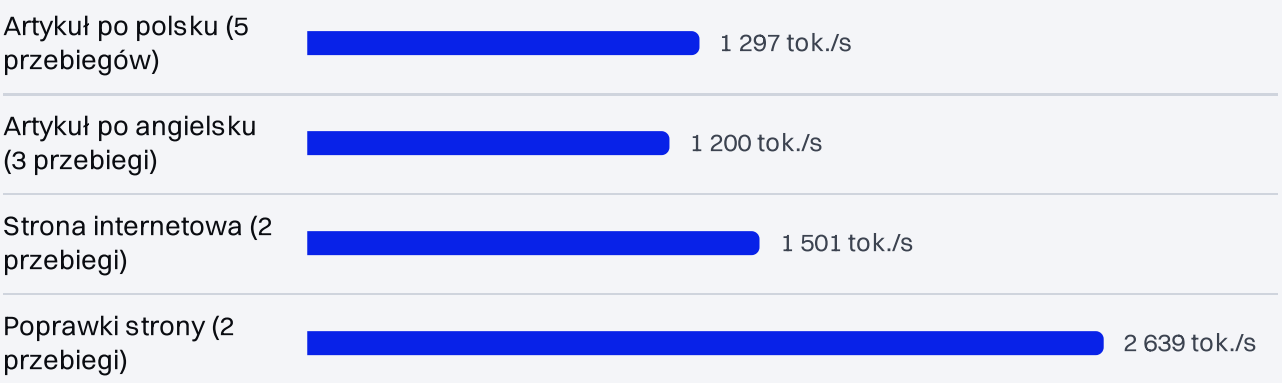
Nie porównywałem modelu z innymi modelami we własnych testach. W części 3 przytaczam dane producenta o szybkości innych modeli. Raport nie rozstrzyga więc, czy deklaracja producenta o poziomie inteligencji jest trafna. Opisuje, jak model zachował się w konkretnych zadaniach, i podaje liczby, które każdy może odtworzyć. Szczegóły i ograniczenia opisuję w części 10.

3. SZYBKOŚĆ I KOSZT

3.1. Pomiar własny

Czas mierzyłem po stronie klienta, od wysłania zapytania do odebrania całej odpowiedzi, bez przesyłania strumieniowego. Zawiera on więc także czas przesyłania danych przez sieć.

Wykres 1. Szybkość generowania odpowiedzi według rodzaju zadania (mediana, tokeny odpowiedzi na sekundę)



ZADANIE	WARTOŚĆ
Artykuł po polsku (5 przebiegów)	1 297 tok./s
Artykuł po angielsku (3 przebiegi)	1 200 tok./s
Strona internetowa (2 przebiegi)	1 501 tok./s
Poprawki strony (2 przebiegi)	2 639 tok./s

Źródło: test własny, 16-17 listopada 2026 r.

W dwunastu długich odpowiedziach szybkość wynosiła od około 800 do 2800 tokenów na sekundę, z medianą około 1300. Najszybciej model oddawał poprawioną wersję strony, czyli odpowiedź, która w większości powtarza tekst otrzymany na wejściu.

W praktyce oznacza to:

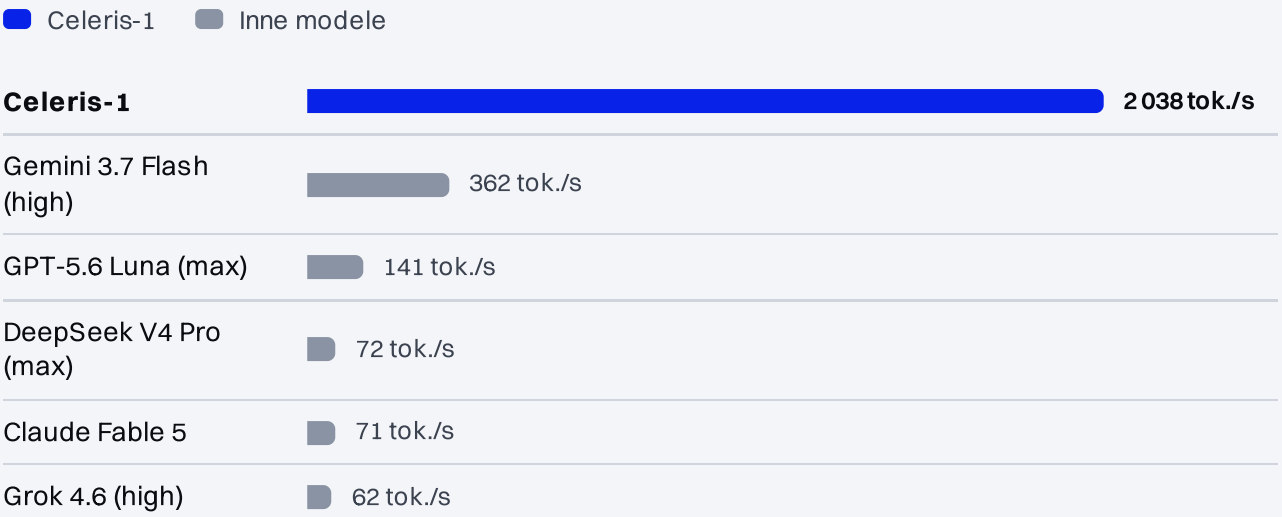
- artykuł o długości 16-19 tysięcy znaków: od 2,6 do 5,4 sekundy,
- strona internetowa licząca ponad 900 linii kodu: 5,6 i 5,9 sekundy,
- klasyfikacja jednej wiadomości: mediana 0,43 sekundy.

Według cennika producenta milion tokenów wejściowych kosztuje 0,20 dolara, a milion tokenów odpowiedzi 0,70 dolara[2]. Artykuł z testu (około 4500 tokenów odpowiedzi) kosztuje więc około 0,3 centa, a strona internetowa (około 8800 tokenów) około 0,6 centa. Wszystkie zapytania wykonane w ramach testu, łącznie z próbami wstępnymi, kosztowały razem około 9 centów.

3.2. Porównanie z innymi modelami według danych producenta

Sam nie mierzyłem szybkości innych modeli. Poniższe dwa zestawienia pochodzą z materiałów producenta i przytaczam je jako jego deklaracje, których nie weryfikowałem.

Wykres 2. Szybkość generowania odpowiedzi w wybranych modelach według producenta Celeris-1 (tokeny odpowiedzi na sekundę)



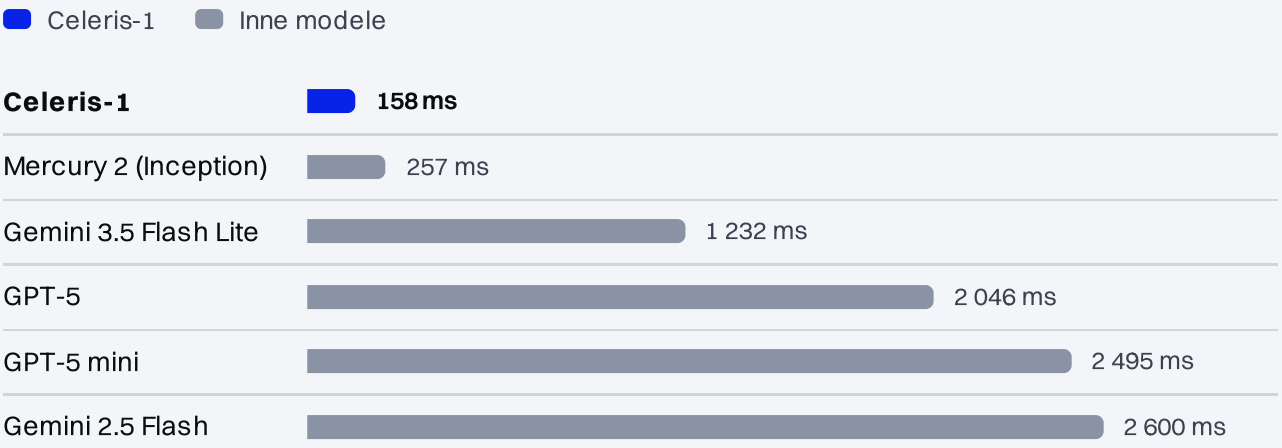
MODEL	CELERIS-1	INNE MODELE
Celeris-1	2 038 tok./s	
Gemini 3.7 Flash (high)		362 tok./s
GPT-5.6 Luna (max)		141 tok./s
DeepSeek V4 Pro (max)		72 tok./s
Claude Fable 5		71 tok./s
Grok 4.6 (high)		62 tok./s

Źródło: dane producenta, celeris.ai, dostęp 17 listopada 2026 r. Producent nie podaje warunków pomiaru

Według tego zestawienia Celeris-1 generuje 2038 tokenów na sekundę, czyli 5,6 razy więcej niż drugi model na liście[3]. Strona nie podaje daty pomiaru, użytych poleceń ani długości odpowiedzi. Nazwy modeli i ustawień przytaczam w brzmieniu podanym przez producenta.

W moim teście mediana wyniosła około 1300 tokenów na sekundę, a najwyższy wynik około 2800. Wartości te nie są wprost porównywalne z deklaracją producenta, bo mój pomiar obejmuje przesyłanie danych przez sieć i czas oczekiwania na początek odpowiedzi. Potwierdzają jednak rząd wielkości: ponad tysiąc tokenów na sekundę.

Wykres 3. Czas odpowiedzi na jedno pytanie testu MMLU-Pro według producenta Celeris-1 (mediana, milisekundy; mniej znaczy szybciej)



MODEL	CELERIS-1	INNE MODELE
Celeris-1	158 ms	
Mercury 2 (Inception)		257 ms
Gemini 3.5 Flash Lite		1 232 ms
GPT-5		2 046 ms
GPT-5 mini		2 495 ms
Gemini 2.5 Flash		2 600 ms

Źródło: dane producenta, celeris.ai, lipiec 2026 r.

Drugie zestawienie ma opisaną metodę [4]. Producent zmierzył medianę czasu odpowiedzi na pojedyncze pytania testu wiedzy MMLU-Pro, z wyłączonym rozumowaniem, tym samym zestawem poleceń dla wszystkich modeli. Dla większości modeli podał czas raportowany przez serwer dostawcy, a dla dwóch modeli Gemini czas mierzony po stronie klienta, więc podstawa pomiaru nie jest jednakowa.

Przy czytaniu obu wykresów trzeba pamiętać o trzech rzeczach. Po pierwsze, pomiary wykonał producent ocenianego modelu, a nie niezależny podmiot. Po drugie, zestawy modeli na obu wykresach są różne i pochodzą z różnych okresów. Po trzecie, szybkość nie mówi nic o jakości odpowiedzi, którą opisują dalsze części raportu.

4. DŁUGI TEKST PO POLSKU

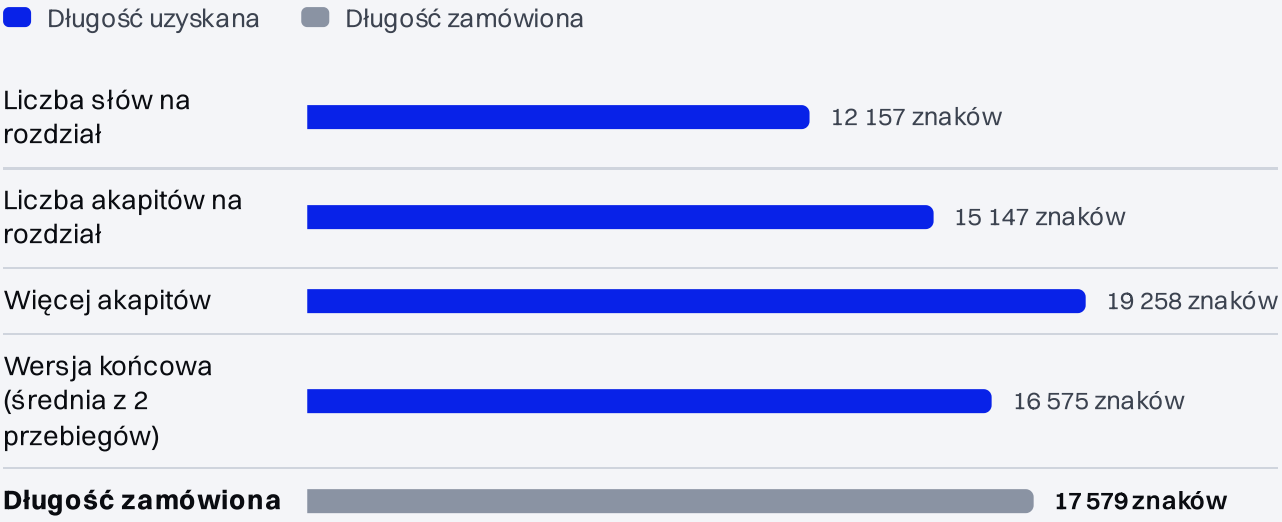
Zadaniem było napisanie artykułu „Ocena zgodności usług cyfrowych z Europejskim Aktem o Dostępności” o długości około 17 579 znaków ze spacjami, z nagłówkami, przypisami i bibliografią.

4.1. Sterowanie długością

Na krótkie polecenie z samą liczbą znaków model odpowiedział szkicem o długości około jednej trzeciej zamówionego tekstu. Dodał też komentarz, że pełnej długości nie da się uzyskać w jednej odpowiedzi, oraz wskazówki, jak tekst samodzielnie rozbudować. Lista źródeł zawierała pozycje, których nie byłem w stanie zidentyfikować, i błędny numer dyrektywy.

Kolejne wersje polecenia zawierały gotową strukturę, zamkniętą listę faktów i źródeł oraz zakaz komentarzy. Różniły się sposobem określenia długości.

Wykres 4. Długość artykułu w zależności od sposobu określenia długości w poleceniu (znaki ze spacjami)



WERSJA POLECENIA	DŁUGOŚĆ UZYSKANA	DŁUGOŚĆ ZAMÓWIONA
Liczba słów na rozdział	12 157 znaków	
Liczba akapitów na rozdział	15 147 znaków	
Więcej akapitów	19 258 znaków	
Wersja końcowa (średnia z 2 przebiegów)	16 575 znaków	
Długość zamówiona		17 579 znaków

Źródło: test własny, 16-17 listopada 2026 r.

Model nie trzyma się liczby słów: gdy polecenie podawało liczbę słów dla każdego rozdziału, pisał około 60% zamówionej objętości. Trzyma się natomiast liczby akapitów. Polecenie w postaci „rozdział 4: siedem akapitów, każdy na podany temat" dało tekst o długości 92-96% zamówionej. Dwa przebiegi tego

samego polecenia różniły się o około 700 znaków, mimo ustawienia, które powinno dawać powtarzalne odpowiedzi.

4.2. Poprawność językowa

Wszystkie polskie teksty zawierały wyrazy, które nie istnieją. Nie są to pojedyncze literówki, tylko słowa z brakującymi, powtórzonymi lub przestawionymi fragmentami. Przykłady z jednego artykułu: „usługawcy", „ujednacający", „sposóbierzalny", „optymalizacjiizacji", „dzięciokrotności", „Zrozumumiałość". Błąd „Europeskiego" powtarzał się w każdym przebiegu, także w tytule.

Pisownię sprawdziłem programem Hunspell z polskim słownikiem[5]. W pięciu tekstach liczących łącznie około 10 200 słów znalazłem około 110 takich wyrazów, czyli około 11 na 1000 słów. Najdłuższy tekst miał najwyższy odsetek: około 14 na 1000.

Narzędzie wykrywa tylko wyrazy, których nie ma w słowniku. Nie widzi błędów, które przypadkiem dają istniejące słowo, na przykład „łat do obsługi" zamiast „łatwe w obsłudze", ani błędów gramatycznych. Rzeczywista liczba usterek jest więc wyższa.

4.3. Treść i przypisy

Fakty podane w poleceniu model przenosił do tekstu poprawnie: daty, numery aktów prawnych i nazwy organów zgadzały się we wszystkich przebiegach. Po podaniu zamkniętej listy źródeł nie dopisywał własnych.

Słabsza była treść, którą model dodawał od siebie. Dominowały ogólne stwierdzenia, a część była błędna. Przykład: zasadę solidności z WCAG model opisał jako gwarancję, że „technologia będzie działać poprawnie mimo błędów w kodzie", podczas gdy chodzi o zgodność treści z programami użytkownika, w tym z technologiami wspomagającymi[6].

Odwołania do źródeł model wstawiał mechanicznie. Po wymaganiu co najmniej trzech odwołań na rozdział w tekście pojawiło się ich około 80, często przy zdaniach, których wskazane źródło nie potwierdza.

5. TEKST PO ANGIELSKU I TŁUMACZENIE

To samo polecenie przetłumaczyłem na angielski i uruchomiłem trzy razy. Teksty miały od 17 511 do 20 802 znaków, czyli trafiały w zamówioną długość bez dodatkowych poprawek polecenia. Sprawdzenie pisowni słownikiem angielskim[7] nie wykazało żadnego błędnie zapisanego wyrazu w trzech tekstach liczących łącznie ponad 7,5 tysiąca słów. Opis zasad WCAG był poprawny.

Następnie dwa angielskie artykuły przetłumaczyłem tym samym modelem na polski, na dwa sposoby: cały tekst w jednym zapytaniu oraz rozdział po rozdziale. Polecenie zawierało słowniczek terminów.

Wykres 5. Nieistniejące słowa w tekstach wygenerowanych przez model (liczba na 1000 słów)



RODZAJ TEKSTU	WARTOŚĆ
Tekst pisany po angielsku	0,0 na 1000
Tekst pisany po polsku	10,7 na 1000
Tłumaczenie z angielskiego na polski	16,7 na 1000

Źródło: test własny, 16-17 listopada 2026 r. Teksty polskie: Hunspell ze słownikiem pl_PL, po ręcznym odjęciu skrótów i nazw. Teksty angielskie: pyspellchecker

Tłumaczenie nie rozwiązało problemu. Cztery przetłumaczone teksty zawierały łącznie około 158 nieistniejących słów na około 9500, czyli około 17 na 1000. Dzielenie na rozdziały nie zmieniło wyniku, a mimo słowniczka model pisał „usługawca” zamiast „usługodawca”. Zachowane zostały natomiast struktura, długość i niemal wszystkie odwołania do źródeł.

Wynik wskazuje, że źródłem błędów nie jest długość odpowiedzi ani trudność zadania, tylko samo generowanie polskich wyrazów. Model dobrze rozumie tekst polski, co pokazują wyniki w części 8, ale pisze po polsku z błędami.

6. GENEROWANIE STRONY INTERNETOWEJ

Model otrzymał specyfikację strony typu one-page dla fikcyjnej firmy wykonującej audyty dostępności: jeden plik, osiem sekcji, motyw ciemny i jasny, menu mobilne, lista pytań i odpowiedzi rozwijana przyciskami, formularz z walidacją oraz zgodność z WCAG 2.1 na poziomie AA. To samo zadanie wykonał w wersji polskiej i angielskiej.

Obie strony powstały w niecałe 6 sekund i liczą ponad 900 linii kodu. Sprawdziłem je w przeglądarce w szerokości 1440 i 360 pikseli oraz narzędziem axe-core[8].

Tabela 1. Wynik testu wygenerowanych stron

SPRAWDZANY ELEMENT	WERSJA POLSKA	WERSJA ANGIELSKA
Kompletny plik	tak, bez zewnętrznych bibliotek	tak, bez zewnętrznych bibliotek
Sekcje i nagłówki	komplet sekcji, poprawna hierarchia nagłówków	komplet sekcji, poprawna hierarchia nagłówków
Układ na telefonie	bez przewijania w poziomie	bez przewijania w poziomie
Menu mobilne	działa	działa
Lista pytań	działa	działa
Przełącznik motywu	działa	nie działa
Formularz	walidacja działa, bez oznaczenia błędnych pól dla czytnika ekranu	walidacja działa
Zbyt niski kontrast	11 elementów w motywie ciemnym, 4 w jasnym	3 elementy
Błędy w konsoli	niepoprawna ikona SVG	niepoprawna ikona SVG
Małe elementy klikalne	8 na telefonie	brak

Kod jest uporządkowany: używa zmiennych CSS, znaczników semantycznych, odnośnika „Przejdź do treści” i powiązania komunikatów błędów z polami formularza. Układ jest schludny, choć szablonowy.

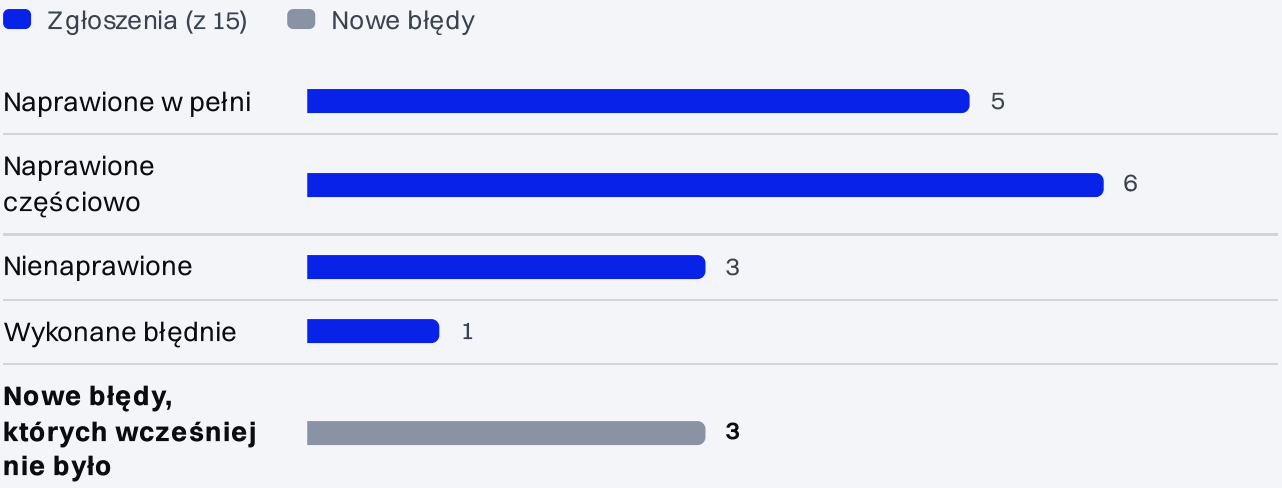
Błędy mają wspólną cechę: nie widać ich bez testów. Przełącznik motywu w wersji angielskiej wygląda poprawnie i zapisuje wybór, ale kolory się nie zmieniają, bo reguła CSS została zapisana w sposób, który nigdy nie ma zastosowania. Polecenie wymagało kontrastu co najmniej 4,5:1 [9], a główne przyciski w wersji angielskiej mają 2,14:1. Ikony w kartach usług to w większości przypadkowe kształty.

W wersji polskiej doszły błędy wyglądu, w tym przycisk bez stylu i numery kroków nachodzące na tytuły, zdublowana karta usługi oraz nieistniejące słowa w treści i komunikatach formularza, na przykład „Proszy podać imię” i „poprawnny adres e-mail”.

7. POPRAWIANIE BŁĘDÓW

W drugim kroku model otrzymał własny plik oraz listę błędów znalezionych w testach, z nazwami elementów i zmierzonymi wartościami: 10 zgłoszeń dla wersji polskiej i 5 dla angielskiej. Miał zwrócić cały poprawiony plik. Wynik sprawdziłem tymi samymi testami.

Wykres 6. Wynik jednej rundy poprawek (liczba zgłoszeń, obie wersje strony łącznie)



WYNIK	ZGŁOSZENIA (Z 15)	NOWE BŁĘDY
Naprawione w pełni	5	
Naprawione częściowo	6	
Nienaprawione	3	
Wykonane błędnie	1	
Nowe błędy, których wcześniej nie było		3

Źródło: test własny, 16-17 listopada 2026 r.

Model dobrze wykonał poprawki proste i punktowe: zamianę tekstu, powiększenie obszaru klikalnego, poprawę reguły CSS odpowiedzialnej za motyw jasny, wyśrodkowanie nagłówków. Przepisał przy tym cały plik bez utraty sekcji i bez zepsucia menu, listy pytań i formularza.

Gorzej wypadły poprawki, których skutek trzeba sprawdzić:

Kontrast. W wersji polskiej model poprawił przyciski, ale numery kroków nadal miały zbyt niski kontrast, a logo straciło kontrast, który wcześniej miało. W wersji angielskiej nie zmienił kolorów przycisków, dopisał natomiast w kodzie komentarz o wysokim kontraście.

Zależność między motywami. Przycisk poprawiony w motywie ciemnym stał się w motywie jasnym biały na białym tle.

Ikony SVG. Model dwukrotnie otrzymał dokładny komunikat błędu i dwukrotnie zostawił ten sam kod.

Fokus w formularzu. W wersji angielskiej fokus po nieudanej walidacji trafia na ostatnie błędne pole zamiast na pierwsze.

Literówki. Z ośmiu wskazanych wyrazów model poprawił sześć, a w dopisanym tekście wprowadził nową usterkę.

Trzeci z nowych błędów to strzałki dopisane przy odnośnikach w menu wersji angielskiej, o które polecenie nie prosiło.

8. KRÓTKIE ZADANIA

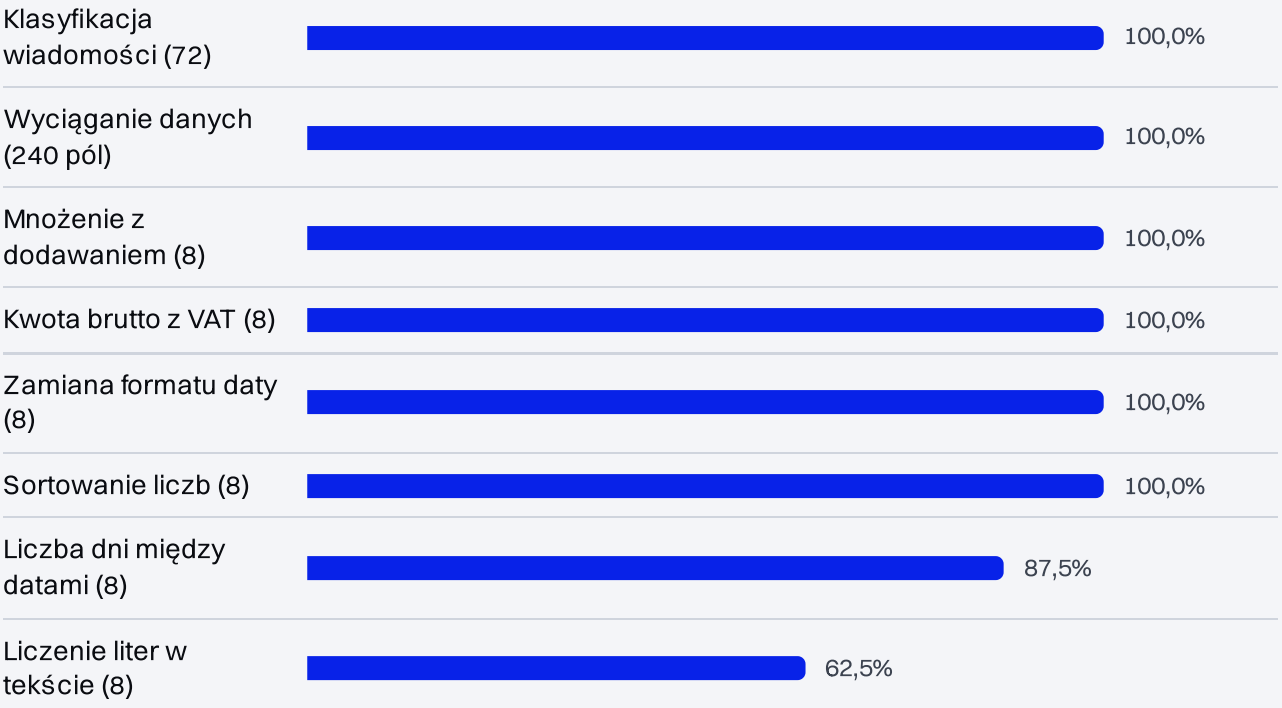
Ostatnia część testu obejmowała 180 zadań, w których odpowiedź jest krótka i ma ustaloną formę.

Klasyfikacja. 72 wiadomości klientów sklepu internetowego, po polsku i po angielsku, do jednej z pięciu kategorii: dostawa, zwrot, faktura, płatność, problem techniczny. Część wiadomości była pisana potocznie, bez polskich znaków i z literówkami, a część łączyła dwa wątki.

Wyciąganie danych. 60 wiadomości, z których należało wyodrębnić numer zamówienia, datę, kwotę i telefon. Wiadomości zawierały dane mylące: drugą datę, liczbę sztuk i kod pocztowy. Daty, kwoty i numery telefonów miały różne formaty.

Obliczenia i przekształcenia. 48 zadań w sześciu grupach po osiem.

Wykres 7. Trafność w krótkich zadaniach (odsetek poprawnych odpowiedzi)



ZADANIE (LICZBA PRZYKŁADÓW)	WARTOŚĆ
Klasyfikacja wiadomości (72)	100,0%
Wyciąganie danych (240 pól)	100,0%
Mnożenie z dodawaniem (8)	100,0%
Kwota brutto z VAT (8)	100,0%
Zamiana formatu daty (8)	100,0%
Sortowanie liczb (8)	100,0%
Liczba dni między datami (8)	87,5%
Liczenie liter w tekście (8)	62,5%

Źródło: test własny, 16-17 listopada 2026 r., n = 180

Klasyfikację i wyciąganie danych model wykonał bezbłędnie, a każda ze 132 odpowiedzi, które miały być obiektem JSON, była poprawnym JSON-em. Polski język wiadomości nie miał wpływu na wynik.

Cztery błędy dotyczą zadań, w których trzeba liczyć na dłuższym dystansie. Model pomylił się o jeden dzień przy odstępie około pół roku i trzy razy źle policzył wystąpienia litery, na przykład w słowie „Mississippi” nie znalazł żadnej litery „p”.

9. WNIOSKI

Model nadaje się do zadań krótkich, o ustalonej formie i sprawdzalnym wyniku. Klasyfikacja zgłoszeń, kierowanie wiadomości, wyciąganie danych do JSON i normalizacja formatów to zastosowania, w których szybkość i niska cena przekładają się na realną korzyść, a ryzyko błędu w teście było znikome.

Polski tekst do publikacji wymaga innego modelu albo pełnej korekty. Dotyczy to także tłumaczeń i krótkich komunikatów w interfejsie. Model czyta po polsku dobrze, ale pisze z błędami.

Długość i strukturę tekstu najlepiej narzucać liczbą akapitów i tematem każdego z nich. Limity podane w słowach lub znakach model ignoruje.

Fakty i źródła trzeba podać w poleceniu. Model poprawnie je przenosi, ale to, co dopisuje sam, bywa błędne, a odwołań do źródeł nie można traktować jako wiarygodnych.

Wygenerowany kod trzeba testować automatycznie po każdej zmianie. Model nie sprawdza własnej pracy: oddaje funkcje, które wyglądają na gotowe, a nie działają, deklaruje spełnienie wymagań, których nie spełnia, i przy poprawkach wprowadza nowe błędy. Przy odpowiedzi w kilka sekund opłacalna jest pętla: generowanie, test, lista błędów, kolejna runda.

Obliczeń nie należy zlecać modelowi. Daty, kwoty i liczby lepiej wyciągnąć modelem, a działania na nich wykonać w kodzie.

10. METODYKA I OGRANICZENIA

10.1. Przebieg

Testy przeprowadziłem 16 i 17 listopada 2026 r. przez interfejs programistyczny producenta, zgodny z formatem zapytań OpenAI[1], na koncie z bezpłatnym kredytem startowym. Wszystkie zapytania miały temperaturę 0 i były wysyłane bez polecenia systemowego. Zadania krótkie wysyłano równolegle, po sześć naraz.

Strony internetowe testowałem w przeglądarce Chromium w trybie bez interfejsu, w oknach o szerokości 1440 i 360 pikseli. Kontrast i pozostałe reguły dostępności sprawdzałem narzędziem axe-core, a działanie przełącznika motywu, listy pytań, formularza i menu mobilnego skryptem, który wykonuje kliknięcia i odczytuje stan strony.

Pisownię tekstów polskich sprawdzałem programem Hunspell 1.7.2 ze słownikiem pl_PL, a angielskich biblioteką pyspellchecker 0.9.1. Z listy wyrazów nieznanych słownikowi ręcznie usunąłem skróty, nazwy własne, tytuły źródeł i poprawne wyrazy nieobecne w słowniku. Liczby nieistniejących słów w tekstach polskich są więc przybliżone.

10.2. Dane

Wiadomości do klasyfikacji napisałem ręcznie. Wiadomości do wyciągania danych i zadania obliczeniowe zostały wygenerowane losowo z ustalonym ziarnem, więc poprawne odpowiedzi wynikają z generatora, a test można powtórzyć na tych samych danych.

10.3. Ograniczenia

Brak własnego porównania z innymi modelami. Raport opisuje testy jednego modelu, a dane o szybkości innych modeli pochodzą od producenta. Nie wynika z niego, czy inne modele o podobnej szybkości lub cenie wypadają lepiej, czy gorzej.

Mała liczba przebiegów w zadaniach długich. Artykuły generowałem od dwóch do pięciu razy, a każdą wersję strony raz. Inne uruchomienie może dać inne błędy.

Jeden temat i jeden rodzaj strony. Wyniki dla innych dziedzin, dłuższych dokumentów i bardziej złożonych aplikacji mogą się różnić.

Dane syntetyczne w zadaniach krótkich. Kategorie były wyraźnie opisane w poleceniu, a wiadomości krótkie. Na prawdziwych zgłoszeniach, zwłaszcza długich i niejednoznacznych, trafność będzie niższa.

Ocena treści jest częściowo jakościowa. Poprawność merytoryczną i trafność odwołań oceniałem, czytając teksty, bez formalnej skali.

Pomiar czasu po stronie klienta. Wyniki zależą od połączenia sieciowego i obciążenia usługi w chwili testu.

Stan z dwóch dni. Model i usługa mogą być aktualizowane, więc późniejsze wyniki mogą być inne.

PRZYPISY

Celeris, dokumentacja usługi, strona główna: opis modelu Celeris-1 i formatu zapytań, dostęp 17 listopada 2026 r.

Celeris, dokumentacja usługi, cennik: stawki za milion tokenów dla modelu Celeris-1, dostęp 17 listopada 2026 r.

Celeris, strona główna usługi, rysunek 4: liczba tokenów odpowiedzi na sekundę w wybranych modelach, dostęp 17 listopada 2026 r.

Celeris, How we test Celeris-1: speed and quality on the same footing, lipiec 2026 r.

Hunspell, program do sprawdzania pisowni, wersja 1.7.2, ze słownikiem języka polskiego pl_PL.

W3C, Web Content Accessibility Guidelines (WCAG) 2.1, W3C Recommendation, zasada 4 i wytyczna 4.1.

pyspellchecker, biblioteka języka Python do sprawdzania pisowni, wersja 0.9.1.

Deque Systems, axe-core, dokumentacja reguł.

W3C, Web Content Accessibility Guidelines (WCAG) 2.1, W3C Recommendation, kryterium sukcesu 1.4.3.

BIBLIOGRAFIA

Celeris, dokumentacja usługi. docs.celeris.ai

Celeris, How we test Celeris-1: speed and quality on the same footing, lipiec 2026 r. celeris.ai/blog-benchmarks

Celeris, strona główna usługi. celeris.ai

Celeris, cennik. docs.celeris.ai/pricing

Deque Systems, axe-core, dokumentacja reguł. github.com/dequelabs/axe-core

Hunspell, program do sprawdzania pisowni. hunspell.github.io

pyspellchecker, biblioteka do sprawdzania pisowni. pypi.org/project/pyspellchecker

W3C, Web Content Accessibility Guidelines (WCAG) 2.1, W3C Recommendation. w3.org/TR/WCAG21

SŁOWNIK

Duży model językowy (LLM)

Model sztucznej inteligencji wytrenowany na bardzo dużych zbiorach tekstu, który generuje tekst lub kod w odpowiedzi na polecenie. Skrót od angielskiego large language model.

Token

Jednostka tekstu, na której operuje model językowy: wyraz, jego fragment lub znak interpunkcyjny. W tokenach podaje się długość odpowiedzi, szybkość i cenę.

Polecenie (prompt)

Tekst przekazywany modelowi językowemu, który opisuje zadanie do wykonania, jego warunki i oczekiwaną formę odpowiedzi.

Temperatura

Ustawienie modelu językowego określające losowość odpowiedzi. Wartość 0 powinna dawać odpowiedzi najbardziej powtarzalne.

MMLU-Pro

Publiczny test wiedzy dla modeli językowych złożony z pytań wielokrotnego wyboru z wielu dziedzin. Służy do porównywania modeli na tym samym zestawie pytań.

JSON

Tekstowy format zapisu danych w postaci par nazwa i wartość, łatwy do odczytania przez programy. Skrót od JavaScript Object Notation.

Klasyfikacja

Przypisanie tekstu do jednej z ustalonych z góry kategorii, na przykład zgłoszenia klienta do działu, który ma się nim zająć.

Ekstrakcja danych

Wyodrębnianie z tekstu konkretnych informacji, na przykład numeru zamówienia, daty lub kwoty, i zapisanie ich w ustalonej strukturze.

Hunspell

Otwarty program do sprawdzania pisowni, używany między innymi w przeglądarkach i pakietach biurowych. Wskazuje wyrazy, których nie ma w słowniku danego języka.

axe-core

Otwarte narzędzie firmy Deque Systems do automatycznego testowania dostępności stron internetowych.

WCAG

Wytyczne dla dostępności treści internetowych opracowane przez W3C. Skrót od Web Content Accessibility Guidelines. Aktualna wersja to 2.2.

Kontrast

Stosunek jasności tekstu do jasności tła. WCAG na poziomie AA wymaga co najmniej 4,5:1 dla zwykłego tekstu.

Fokus

Wskazanie elementu strony, który aktualnie przyjmuje polecenia z klawiatury. Zwykle widoczny jako obramowanie wokół odnośnika, przycisku lub pola.

Mediana

Wartość środkowa: połowa wyników jest od niej niższa, a połowa wyższa. W przeciwieństwie do średniej nie zależy od pojedynczych skrajnych wyników.

JAK CYTOWAĆ

Załucki T., *Szybki model językowy w praktyce: test Celeris-1. Teksty, kod strony internetowej i krótkie zadania*, 2026, dostęp online.

Tekst ma charakter przeglądowy i informacyjny. Nie stanowi porady prawnej. Stan prawny i stan badań na dzień publikacji.