PRAKTYCZNA LISTA KONTROLNA · GEORGE M. J. ZAK

Zanim wdrożysz aplikację zbudowaną z AI

AI sprawiło, że niemal każdy może dziś zbudować i wypuścić w świat aplikację, stronę, automatyzację albo agenta. To naprawdę wspaniałe. Ale „udało mi się to zbudować" nie znaczy „można to bezpiecznie uruchomić w produkcji". Większość aplikacji zbudowanych z AI nie przechodzi tych samych podstawowych kontroli bezpieczeństwa. Ten poradnik to praktyczny przewodnik — bez teorii, bez gadania o compliance — co należy sprawdzić, zanim Twój projekt zbudowany z pomocą AI wejdzie do internetu. Wraz z konkretnym promptem, który możesz wkleić do swojego asystenta AI, żeby zrobił za Ciebie większość roboty.

Cyberbezpieczeństwo | Era AI | ~12 min czytania | v1 · 2026-06-06

George M. J. Zak · Strategia AI, cyberbezpieczeństwo i ludzka strona technologii

CO ZNAJDZIESZ W ŚRODKU

SEKCJA · 01 · PUNKT WYJŚCIA

01. Dlaczego to ma teraz znaczenie

Przez lata pracowałem w cyberbezpieczeństwie, zarządzaniu ryzykiem, governance i technologii — wszystko to zanim narzędzia AI w obecnej formie w ogóle istniały. To, co widzę w 2026 roku, jest zarazem niezwykłe i odrobinę przerażające: ludzie, którzy nigdy w życiu nie napisali linijki kodu, wypuszczają teraz aplikacje. Prawdziwe. Łączące się z prawdziwymi bazami danych. Przechowujące dane prawdziwych użytkowników. Pobierające prawdziwe pieniądze.

To w sumie wspaniała sprawa. Tworzenie oprogramowania wraca wreszcie do rąk ludzi, którzy chcą rozwiązywać własne problemy. Przed-AI-owy świat IT traktował programowanie jak kapłaństwo; AI to zdemokratyzowało. I dobrze.

Ale część poświęcona bezpieczeństwu nigdy nie była tą łatwą częścią pracy. Była ukryta za tym kapłaństwem. A teraz tysiące aplikacji zbudowanych z AI ląduje co tydzień w produkcji, tworzonych przez ludzi, którzy nigdy nie musieli zgłębiać warstwy bezpieczeństwa, bo zawsze ktoś inny robił to za nich.

> **JEDNA NIEWYGODNA PRAWDA**
>
> To, że *udało Ci się* coś zbudować, nie znaczy, że można to bezpiecznie wdrożyć. To, że AI pomogło Ci to zbudować, nie znaczy, że AI sprawdziło, czy jest to bezpieczne. To są dwie różne umiejętności, a AI jest obecnie znacznie lepsze w tej pierwszej niż w drugiej.

Ta lista kontrolna to dokładnie ta sama, którą stosuję u klientów i u siebie. Przejdź przez nią przed wdrożeniem. Potem pokaż ją swojemu ulubionemu asystentowi AI i poproś, żeby wykonał faktyczną pracę. Połączenie — Twój osąd plus wykonanie AI — to sposób, żeby wypuścić coś, czego nie będziesz musiał się wstydzić za trzy miesiące.

SEKCJA · 02 · TOŻSAMOŚĆ

02. Uwierzytelnianie

Większość incydentów bezpieczeństwa zaczyna się przy drzwiach wejściowych. Jeśli źle zaprojektowałeś wejście, reszta budynku przestaje mieć znaczenie.

SPRAWDŹ:

- ☐ **Strony administracyjne są chronione od początku do końca.**

  Nie tylko strona, która wyświetla panel admina — także endpointy API, do których ten panel się odwołuje. Brama w interfejsie to wygoda dla uczciwych użytkowników; brama w API to prawdziwa ochrona. Jeśli istnieje URL, który zwraca wrażliwe dane bez sprawdzania uprawnień, ten URL prędzej czy później zostanie znaleziony.

- ☐ **Konta użytkowników są odpowiednio zabezpieczone.**

  Zasady haseł, które nie walczą z menedżerami haseł. Weryfikacja adresu email zanim użytkownik dostanie dostęp do czegokolwiek wrażliwego. Blokada konta po wielokrotnych nieudanych próbach logowania — żeby brute-force kosztował atakującego więcej niż Ciebie.

- ☐ **Hasła są przechowywane poprawnie.**

  Hashowane przez bcrypt, argon2 albo scrypt — nigdy w postaci jawnej, nigdy szyfrowaniem odwracalnym. Sole są indywidualne dla każdego użytkownika. Jeśli Twój asystent AI wygenerował przechowywanie haseł w postaci jawnej, popraw to przed wdrożeniem. To błąd, który w produkcji potrafi zakończyć karierę.

- ☐ **MFA jest dostępne dla wrażliwych funkcji.**

  Nawet jeśli go nie wymusisz, daj możliwość. TOTP przez aplikację (Authenticator i podobne) to tania, sprawdzona opcja. SMS jest lepszy niż nic, ale jest podatny na SIM swap.

- ☐ **Tokeny sesyjne są poprawnie obsługiwane.**

  HttpOnly, Secure, SameSite=Lax (albo Strict przy szczególnie wrażliwych przepływach). Rozsądny czas ważności. Unieważnione po wylogowaniu. Powiązane z urządzeniem, jeśli to możliwe.

WZÓR, KTÓRY NAJCZĘŚCIEJ WIDZĘ

Kod uwierzytelniania wygenerowany przez AI wygląda poprawnie, bo trzyma się znanych wzorców. Ale AI często pomija te szczegóły implementacyjne, które dopiero te wzorce naprawdę zabezpieczają — rotację tokenów, unieważnianie sesji po zmianie hasła, ochronę przed atakami powtórzeniowymi. Zapytaj wprost: *„W mojej implementacji uwierzytelniania pokaż mi, co się dzieje, gdy: (a) użytkownik zmienia hasło, (b) wycieka token sesji, (c) użytkownik się wylogowuje, (d) ktoś próbuje to samo hasło 50 razy. Pokaż każdą ścieżkę w kodzie."*

SEKCJA · 03 · DANE

03. Bezpieczeństwo bazy danych

Jeśli użytkownicy mogą czytać nawzajem swoje dane, to nie masz aplikacji. Masz wyciek danych z interfejsem na wierzchu.

SPRAWDŹ:

- ☐ **Row Level Security jest włączone w każdej tabeli zawierającej dane użytkowników.**

  Szczególnie jeśli korzystasz z Supabase, Firebase albo dowolnej usługi, w której klucz anon/client jest dostępny w przeglądarce. RLS to ściana między „co kod ma robić" a „co przeglądarka faktycznie może zrobić". Bez tego ściany po prostu nie ma.

- ☐ **Użytkownicy nie mogą sięgnąć po dane innych użytkowników.**

  Sprawdź to konkretnym testem. Zaloguj się jako użytkownik A. Spróbuj odczytać rekord, który należy do użytkownika B. Jeśli API go zwróci, masz błąd eskalacji poziomej — jeden z najczęstszych błędów w kodzie generowanym przez AI, bo AI dodaje endpoint odczytu bez ograniczenia go do `auth.uid()`.

- ☐ **Uprawnienia są zgodne z zasadą najmniejszych przywilejów.**

  Rola anon powinna mieć jak najmniej uprawnień. Klucz service-role nigdy nie powinien trafiać do przeglądarki. Tabele dostępne tylko dla administratora powinny mieć całkowicie odebrany dostęp dla anon.

- ☐ **Wrażliwe pola nie są nadmiernie wystawione.**

  Email, telefon, adres, numer dokumentu — jeśli użytkownik nie musi tego widzieć, API nie powinno tego zwracać. Lista użytkowników zwracająca wszystkie kolumny to lista, która przy złym zapytaniu wyciągnie wszystko.

- ☐ **Kopie zapasowe istnieją i były testowane.**

  „Mamy backupy" to nie to samo co „sprawdziliśmy, że potrafimy z nich odtworzyć dane". Ćwiczenia odtwarzania są nudne; są też tym, co odróżnia firmy, które się podnoszą, od tych, które nie.

PUŁAPKA SUPABASE / FIREBASE / NO-CODE

Hostowane backendy zazwyczaj mają „anon" albo „client" klucz wbudowany w Twój kod frontendowy. To jest celowe i w porządku — *pod warunkiem że masz skonfigurowane RLS albo odpowiedniki*. Wiele aplikacji zbudowanych z AI ląduje w produkcji z całkowicie wyłączonym RLS na każdej tabeli. To znaczy, że klucz anon — który każdy odwiedzający może wyciągnąć z Twojego JavaScriptu — ma pełen odczyt i zapis na całej Twojej bazie. Sprawdź to przed startem. Najlepiej dziś. To darmowa naprawa, która zmienia Twoje bezpieczeństwo z gruntu.

SEKCJA · 04 · SEKRETY

04. Sekrety i klucze API

Każdy sekret, który trafia do Twojego repozytorium albo do bundle'a frontendowego, jest sekretem na otwartym internecie. Traktuj go odpowiednio.

SPRAWDŹ:

- ☐ **Klucze API nie znajdują się w kodzie frontendowym.**

  Wszystko, co jest w katalogu `src/` albo `public/`, staje się częścią JavaScriptu, który przeglądarka pobiera. Przeszukaj repozytorium pod kątem prefiksów wszystkich usług, których używasz — Stripe (`sk_`), Resend (`re_`), OpenAI (`sk-`) itd. Jeśli któryś z nich pojawia się poza odwołaniem do zmiennej środowiskowej, od razu obróć klucz.

- ☐ **Sekrety są przechowywane w zmiennych środowiskowych.**

  Nie w kodzie. Nie w plikach konfiguracyjnych wrzuconych do gita. Zawsze upewnij się, że `.gitignore` zawiera `.env*` i że Twój plik z sekretami jest po stronie ignorowanej.

- ☐ **Dane testowe nie trafiły do produkcji.**

  Jeśli w fazie developmentu używałeś testowego klucza Stripe, sprawdź, że produkcja ma prawdziwy klucz live. Jeśli używałeś testowej audience w Resend, upewnij się, że produkcja nie pisze już do niej. Boilerplate generowany przez AI często zostawia testowe poświadczenia w środku.

- ☐ **Klucze kont serwisowych mają minimalny zakres uprawnień.**

  Klucz konta serwisowego, który może czytać wszystko w Twoim projekcie Google Cloud, to klucz, którego nie powinieneś mieć. Ogranicz uprawnienia do konkretnych bucketów, konkretnych projektów, konkretnych ról.

- ☐ **Jeśli klucz kiedykolwiek wyciekł, został rotowany.**

  Bez wyjątków. Jeśli wrzuciłeś go do publicznego repozytorium na dziesięć sekund — klucz jest skompromitowany. GitHub indeksuje commity w ciągu minut; boty skanują nowe repozytoria w ciągu godzin.

SEKCJA · 05 · WEJŚCIE

05. Walidacja danych wejściowych

Każdy formularz, każdy endpoint API, każdy parametr w URL-u to okazja dla kogoś, żeby wysłać coś, czego się nie spodziewałeś. Domyślnie zakładaj, że dane wejściowe są złośliwe.

SPRAWDŹ:

- ☐ **Walidacja po stronie serwera w każdym formularzu.**

  Walidacja po stronie klienta służy UX-owi. Walidacja po stronie serwera to bezpieczeństwo. Jeśli AI dało Ci tylko walidację po stronie klienta, to znaczy, że nie masz żadnej walidacji. `curl` omija każdą kontrolę w JavaScripcie.

- ☐ **SQL injection nie jest możliwy.**

  Używaj zapytań parametryzowanych / prepared statements. Nigdy nie sklejaj danych wejściowych z zapytaniem SQL. Jeśli używasz ORM-a albo query buildera (Supabase, Prisma itd.), zazwyczaj jest to obsłużone — ale sprawdź wszystkie surowe zapytania, które napisałeś sam.

- ☐ **XSS (cross-site scripting) jest zablokowany.**

  Jeśli Twoja aplikacja wyświetla tekst dostarczony przez użytkownika, ten tekst musi być escapowany albo sanitizowany. React, Vue i Svelte robią to domyślnie — dopóki nie użyjesz `dangerouslySetInnerHTML` albo `v-html`. Skontroluj każde użycie.

- ☐ **Uploadowane pliki są walidowane.**

  Sprawdzaj typ MIME po stronie serwera (nie tylko nagłówek od klienta). Limituj rozmiar pliku. Usuwaj metadane. Jeśli przyjmujesz obrazy, rozważ przepuszczenie ich przez bibliotekę, która zostawi tylko dane pikselowe. Nigdy nie ufaj nazwie pliku.

- ☐ **Pułapki honeypot są zaimplementowane na publicznych formularzach.**

  Ukryte pole input o nazwie `website` albo `hp`, którego prawdziwi użytkownicy nie wypełnią, ale boty tak. Jeśli jest wypełnione, po cichu odrzuć zgłoszenie. Najtańszy filtr antyspamowy, jaki istnieje.

SEKCJA · 06 · ENDPOINTY

06. Publiczne API

Każdy endpoint API to publiczna powierzchnia, chyba że jawnie ją zabezpieczyłeś. Traktuj każdy z nich jak prawdopodobny cel ataku.

SPRAWDŹ:

- ☐ **Rate limiting jest na miejscu.**

  Na IP, na użytkownika, na endpoint. Większość platform (Cloudflare, Vercel, AWS API Gateway) oferuje to kilkoma kliknięciami. Skorzystaj. Endpoint logowania bez limitu to nieograniczony cel ataku brute-force.

- ☐ **Wrażliwe dane nie są wystawione.**

  Endpoint zwracający profil użytkownika nie powinien zwracać hasha hasła, wewnętrznych ID ani flag administratora. Zdefiniuj schemat publiczny jawnie i trzymaj się go.

- ☐ **Atakujący nie mogą enumerować użytkowników ani rekordów.**

  Endpoint rejestracji zwracający „ten email już istnieje" pozwala na enumerację — atakujący może po kolei sprawdzać adresy i odkrywać, kto jest zarejestrowany. Endpoint logowania, który rozróżnia „złe hasło" od „nie ma takiego użytkownika", też pozwala na enumerację. Ujednolić odpowiedzi.

- ☐ **CORS jest skonfigurowany ściśle.**

  Jeśli nie potrzebujesz dostępu cross-origin, nie pozwalaj na niego. Jeśli potrzebujesz, dopuść tylko konkretne origin-y, którym ufasz — nie ∗.

- ☐ **Metody HTTP są ograniczone do tego, czego endpoint potrzebuje.**

  Endpoint tylko do odczytu nie powinien przyjmować POST. Endpoint modyfikujący dane nie powinien przyjmować GET. Doprecyzuj każdą trasę.

SEKCJA · 07 · WIDOCZNOŚĆ

07. Logowanie i monitoring

Nie da się bronić tego, czego nie widzisz. Pierwsze, co odbiera Ci atakujący, to zdolność zauważenia, że tam jest.

SPRAWDŹ:

- ☐ **Zdarzenia istotne dla bezpieczeństwa są logowane.**

  Próby logowania (udane i nieudane), zmiany hasła, podniesienia uprawnień, akcje administratora, błędy API, wyzwolenia rate-limitu. Trzymaj je na osobnej powierzchni logowania niż zwykłe logi aplikacyjne, jeśli to możliwe.

- ☐ **Logi są przechowywane na tyle długo, żeby do czegoś się przydały.**

  90 dni minimum dla większości aplikacji. Dłużej dla wszystkiego, co podlega regulacjom. Logi usuwane po 24 godzinach nie wyłapią niczego sensownego.

- ☐ **Potrafisz wykryć nadużycie w trakcie.**

  Nagły skok nieudanych logowań? Nietypowa geografia? Nowe IP wykonujące wywołania do API administracyjnego? Skonfiguruj choć jeden, dwa alerty. Wykrywanie anomalii jest trudne; alertowanie na oczywiste rzeczy jest łatwe.

- ☐ **Wiesz, kiedy coś zawodzi po cichu.**

  Usługa wysyłania maili zwracająca błąd, którego Twój kod nie sprawdza, będzie zawodzić niewidocznie aż klienci zaczną dzwonić. Podłącz narzędzie do śledzenia błędów aplikacji (Sentry, Rollbar albo coś podobnego). Darmowe plany pokrywają potrzeby większości małych aplikacji.

- ☐ **Dane osobowe nie znajdują się w logach.**

  Hasła, pełne adresy email, dane płatnicze, dokumenty tożsamości — nic z tego nie powinno trafiać do logów. Logi też wyciekają. Jeśli musisz zalogować identyfikator użytkownika, zhashuj go.

SEKCJA · 08 · SPECYFIKA AI

08. Ryzyka specyficzne dla AI

Jeśli Twoja aplikacja używa AI — jakiegokolwiek AI, LLM-ów, embeddingów, agentów, wyszukiwania wektorowego — masz nową warstwę ryzyk, których pięć lat temu jeszcze nie było. Żadne z nich nie jest egzotyczne. Wszystkie są dzisiaj możliwe do wykorzystania w ataku.

RYZYKO	POWAGA	CO ZWERYFIKOWAĆ
Prompt injection	**KRYTYCZNE**	Tekst dostarczony przez użytkownika trafia do Twojego system promptu. Atakujący pisze „zignoruj poprzednie instrukcje i..." — i Twój agent słucha. Traktuj wszystkie dane wejściowe od użytkownika jako niezaufane; nigdy nie sklejaj ich bezpośrednio z promptami. Używaj wyraźnych separatorów; waliduj wyjście, zanim coś na jego podstawie zrobisz.
Nadmierne uprawnienia agenta / narzędzia	**KRYTYCZNE**	Jeśli Twój agent potrafi wykonywać kod, wysyłać emaile, przelewać pieniądze albo modyfikować dane — jaki jest zasięg szkód, gdy prompt injection się powiedzie? Ogranicz uprawnienia narzędzi do minimum. Przy akcjach o dużym wpływie wymagaj potwierdzenia człowieka.
Eksfiltracja danych przez wyjście	WYSOKIE	AI mające dostęp do bazy danych może zostać podstępem zmuszone do wyjawienia danych, których użytkownik nie powinien widzieć. Jeśli model widzi więcej, niż użytkownik ma prawo widzieć, ta luka zostanie wykorzystana. Ogranicz dostęp AI do danych, do których uprawniony jest właśnie pytający użytkownik.
Wybuch kosztów	WYSOKIE	Pojedynczy nieuprawniony użytkownik bez limitu może w jedno popołudnie nabić Ci rachunek za tokeny na tysiące dolarów. Limituj endpointy AI. Ustaw maksymalny miesięczny wydatek tokenów na użytkownika. Monitoruj nietypowe użycie.
Niebezpieczna obsługa wyjścia	WYSOKIE	Jeśli renderujesz wyjście z AI jako HTML, atakujący może wstrzyknąć HTML, który uruchomi się w przeglądarkach Twoich użytkowników. Jeśli wykonujesz kod wygenerowany przez AI, atakujący może wstrzyknąć złośliwy kod. Traktuj wyjście z AI jak niezaufane wejście do kolejnego systemu.
Wyciek danych treningowych	ŚREDNIE	Jeśli fine-tunujesz model na danych użytkowników, te dane mogą pojawić się w odpowiedziach skierowanych do innych użytkowników. Jeśli musisz trenować na danych użytkowników, sanityzuj je i zdobądź wyraźną zgodę. Większość aplikacji nie potrzebuje fine-tuningu; rozważ retrieval-augmented generation zamiast tego.
Po cichu podmieniony	ŚREDNIE	Twój dostawca po cichu aktualizuje model. Nowa wersja zachowuje się inaczej. Zachowania, na których polegałeś,

RYZYKO	POWAGA	CO ZWERYFIKOWAĆ
model		przestają działać — albo, co gorsza, zaczynają działać subtelnie źle. Tam, gdzie dostawca na to pozwala, przypinaj konkretne wersje modelu; testuj przed odpięciem.
Wyciek logów konwersacji	ŚREDNIE	Jeśli logujesz konwersacje z AI, te logi mogą zawierać dane osobowe, sekrety albo treści firmowe, które użytkownik podzielił z modelem. Stosuj do tych logów co najmniej ten sam poziom ochrony, co do danych użytkowników. Często wyższy.

MODEL MENTALNY

Agent LLM w Twojej aplikacji jest mniej więcej tak godny zaufania, jak wykonawca, którego spotkałeś na budowie pięć minut wcześniej. Mądry, zdolny, chętny do pomocy — i absolutnie nie powinien dostać kluczy generalnych, dostępu do bankowości ani niepilnowanego wstępu do tych części budynku, których nie pokazałbyś obcemu. Zasada minimalnych uprawnień ma w komponentach AI większe znaczenie niż gdziekolwiek indziej w stosie, bo komponenty AI są łatwiejsze do zmanipulowania.

SEKCJA · 09 · WDROŻENIE

09. Wzmacnianie wdrożenia

Ostatnia mila. Jeśli wdrożenie zrobisz źle, możesz wszystko inne zrobić dobrze i mimo to mieć publiczny incydent bezpieczeństwa.

SPRAWDŹ:

- ☐ **HTTPS jest wymuszony wszędzie.**

  Bez HTTP. Bez mixed content. Nagłówek HSTS ustawiony z rozsądnym max-age. Zapytania po zwykłym HTTP powinny być przekierowane, a nie po prostu dostępne.

- ☐ **Nagłówki bezpieczeństwa są skonfigurowane.**

  Content-Security-Policy (nawet permisywny start jest lepszy niż żaden). X-Content-Type-Options: nosniff. X-Frame-Options albo frame-ancestors. Referrer-Policy. To pięć linii konfiguracji, które blokują połowę popularnych ataków.

- ☐ **Cloudflare / WAF / podobne — aktywne.**

  Jeśli nie stoisz za CDN-em z WAF-em, jesteś wystawiony bezpośrednio. Darmowa wersja Cloudflare radzi sobie z większością tego, czego potrzebuje mała aplikacja. Skorzystaj.

- ☐ **Mitygacja DDoS jest włączona.**

  Cloudflare daje to za darmo w niewielkiej skali. Vercel i podobne platformy domyślnie zawierają rate limiting. Upewnij się, że odpowiednie funkcje są włączone.

- ☐ **Podatności w zależnościach są sprawdzane.**

  Uruchamiaj `npm audit` / `pip-audit` / odpowiednik przy każdym wdrożeniu. Naprawiaj critical i high przed startem. Włącz Dependabota albo Renovate, żeby dowiadywać się o nowych CVE bez sprawdzania ręcznie.

- ☐ **Domena jest wzmocniona.**

  DNSSEC włączony. Blokada w rejestrze (registrar lock) aktywna. Prywatność WHOIS włączona. Silne hasło + MFA na koncie u dostawcy DNS. Twoja domena to Twoja tożsamość w internecie — utrata jej kontroli rujnuje cały tydzień.

SEKCJA · 10 · SKRÓT

10. Prompt do Twojego AI

Tu przychodzi praktyczny ruch. Skopiuj poniższy prompt do swojego asystenta AI. Większość pracy z listy powyżej wykona za Ciebie. Twoje zadanie to przeczytać, co wyciągnie, i zdecydować, co jest warte naprawy.

SKOPIUJ TEN PROMPT:

Zachowuj się jak senior architekt cyberbezpieczeństwa wykonujący przegląd przed wdrożeniem tej aplikacji.

Twoim zadaniem jest zidentyfikowanie ryzyk bezpieczeństwa na czterech poziomach powagi: CRITICAL (blokujące wdrożenie), HIGH (do naprawy w tym tygodniu), MEDIUM (do naprawy w tym miesiącu), LOW (do śledzenia i powrotu).

Przejdź systematycznie przez te kategorie:

1. Uwierzytelnianie — strony admina, konta użytkowników, obsługa haseł, MFA, bezpieczeństwo sesji
2. Bezpieczeństwo bazy danych — RLS albo odpowiednik, kontrola dostępu poziomego, ujawnianie wrażliwych pól
3. Sekrety — klucze API w frontendzie, higiena zmiennych środowiskowych, testowe poświadczenia w produkcji
4. Walidacja danych wejściowych — kontrole po stronie serwera, SQL injection, XSS, uploady plików, honeypoty
5. Publiczne API — rate limiting, ujawnianie wrażliwych danych, enumeracja, CORS, ograniczenia metod
6. Logowanie — co jest logowane, retencja, wykrywanie nadużyć, wykrywanie cichych awarii, dane osobowe w logach
7. Specyfika AI — prompt injection, uprawnienia narzędzi, obsługa wyjścia, wybuch kosztów
8. Wdrożenie — HTTPS, nagłówki bezpieczeństwa, WAF, podatności w zależnościach, wzmocnienie domeny

Dla każdego znaleziska podaj:
— Kategorię i powagę
— Jednozdaniowy opis ryzyka
— Konkretny plik albo miejsce w moim kodzie
— Konkretną poprawkę, którą mogę zastosować

Zakończ priorytetową listą zadań do wykonania przed wdrożeniem, posortowaną po powadze.

Bądź skrupulatny. Bądź konkretny. Nie pocieszaj mnie — wyciągnij prawdziwe problemy. Bądź uczciwy co do niepewności, gdy czegoś nie widzisz z kodu, który Ci pokazałem.

JAK UŻYWAĆ DOBRZE

Daj AI dostęp do swojego repozytorium, jeśli możesz — przez asystenta świadomego kodu, integrację z IDE albo wklejając reprezentatywne pliki. Prompt jest najbardziej użyteczny, gdy AI widzi prawdziwy kod, a nie tylko Twój opis aplikacji. Iteruj: po pierwszym przejściu dopytuj o konkretne znaleziska („pokaż mi dokładną zmianę kodu dla znaleziska #3"). Potem sprawdź, czy każda poprawka faktycznie weszła. AI czasem twierdzi, że coś naprawiło, podczas gdy w rzeczywistości tego nie zrobiło.

11. EU AI Act i kwestie governance

Jeśli Twoja aplikacja dotyka europejskich użytkowników, EU AI Act obowiązuje niezależnie od tego, gdzie zarejestrowana jest Twoja firma. Akt jest egzekwowany falami przez cały 2026 rok; większość obowiązków dla AI ogólnego przeznaczenia wchodzi w sierpniu 2026. Kilka wymagań tego aktu mapuje się bezpośrednio na listę kontrolną powyżej.

CO SPRAWDZIĆ, JEŚLI WYPUSZCZASZ DO UE ALBO DLA UE:

- ☐ **Artykuł 12 — prowadzenie ewidencji.**
  Systemy AI wysokiego ryzyka muszą prowadzić logi. Twój monitoring AI i logi audytowe nie są opcjonalne, jeśli jesteś w zakresie regulacji — są ustawowe.
- ☐ **Artykuł 14 — nadzór człowieka.**
  Systemy wysokiego ryzyka wymagają nadzoru człowieka nad wyjściem modelu. Nadmiernie uprawniony agent / narzędzie to nie tylko ryzyko bezpieczeństwa — to też problem zgodności, jeśli Twój agent działa bez nadzoru w obszarze wysokiego ryzyka.
- ☐ **Artykuł 10 — zarządzanie danymi.**
  Jakość danych treningowych, ograniczanie biasu, lineage danych. Jeśli fine-tunujesz, musisz mieć historię tego, jak dane treningowe zostały zebrane, zweryfikowane i sprawdzone pod kątem biasu.
- ☐ **Artykuł 15 — dokładność, odporność, cyberbezpieczeństwo.**
  Ten artykuł mapuje się bezpośrednio na całą listę kontrolną powyżej. Akt wyraźnie wymaga od dostawców AI zajęcia się cyberbezpieczeństwem. Twoja postawa bezpieczeństwa jest teraz artefaktem regulacyjnym.
- ☐ **Nakładka RODO.**
  Dane osobowe przechodzące przez komponenty AI nadal są danymi osobowymi w rozumieniu RODO. Prawo do usunięcia, minimalizacja danych, ograniczenie celu — wszystko nadal obowiązuje. AI nie zwalnia nikogo z ochrony danych.

JEŚLI NIE JESTEŚ PEWIEN, CZY TWOJA APLIKACJA JEST „WYSOKIEGO RYZYKA" WG EU AI ACT

Lista wysokiego ryzyka z Załącznika III obejmuje scoring kredytowy, screening rekrutacyjny, decyzje o dostępie do edukacji, zastosowania w ściganiu przestępstw, infrastrukturę krytyczną i kilka innych obszarów. Większość aplikacji konsumenckich nie jest wysokiego ryzyka. Ale obowiązki dla dostawców modeli AI ogólnego przeznaczenia stosują się do *każdego*, kto wypuszcza produkt AI na rynek UE, a duża część dobrych praktyk bezpieczeństwa mapuje się bezpośrednio na oczekiwania regulacyjne. Bezpieczne działanie jest teraz podłogą, nie sufitem.

12. Szczere zakończenie

Budowanie nigdy nie było łatwiejsze niż teraz. Zabezpieczanie tego, co zbudujesz, nadal wymaga uwagi. Połączenie — AI buduje, Ty i AI razem weryfikujecie — to przepływ, który daje na końcu oprogramowanie, którego nie trzeba się wstydzić.

Żadna z pozycji z tej listy nie jest trudna sama w sobie. Wyzwanie polega na tym, że łatwo je przeoczyć. Każda jest jednym z tych zadań „zrobię to później", które budowniczowie omijają, bo wdrożenie wydaje się pilniejsze niż zabezpieczenie. A potem, trzy miesiące po starcie, coś idzie nie tak. Zwykle po cichu. Zwykle widoczne dopiero z perspektywy czasu.

Przejdź przez listę. Użyj promptu. Przeczytaj, co AI wyciągnie. Napraw to, co krytyczne, przed wdrożeniem. Resztę śledź na liście zadań z terminami. Wdrażaj.

A potem, trzy miesiące później, zrób to znowu. Bezpieczeństwo żywej aplikacji nie jest stanem — jest praktyką.

DLACZEGO PUBLIKUJĘ TO ZA DARMO

Jeśli wypuszczasz w świat oprogramowanie zbudowane z AI, cały ekosystem zyskuje, gdy robisz to bezpiecznie. Złe incydenty — wycieki baz danych, skandale z prompt injection, agenci AI podejmujący decyzje, których nie powinni — spowalniają adopcję wszystkim. Sposób na to, żeby era twórców z AI mogła się dalej rozwijać, to zadbanie o to, żeby ludzie, którzy w niej budują, wypuszczali rozsądnie zabezpieczony software. Ta lista to jeden mały wkład w tym kierunku.

Jeśli to pomogło, możesz to przekazać dalej.

Podziel się z jedną osobą, która wypuszcza w świat aplikacje zbudowane z pomocą AI. Wyślij to zespołowi inżynierskiemu w firmie, w której pracujesz. Uruchom na własnej aplikacji i napisz mi, co Cię zaskoczyło.

Jeśli Twój zespół potrzebuje bardziej bezpośredniej pomocy — przegląd architektury, gotowość na EU AI Act, projektowanie governance AI albo praktyczna praca nad bezpieczeństwem — to praca, którą robię. Odezwij się.

— George M. J. Zak · jorgemjzak.com · 2026-06-06