# Sisukord

**Rozwiązanie AI Ask Directo: po prostu zapytaj, a AI odpowie na podstawie Twoich danych w Directo** ..... 3
***Konfiguracja serwera MCP Directo – Claude Desktop i Gemini CLI*** ..... 3
***1. Wprowadzenie – dlaczego MCP?*** ..... 3
Czym jest MCP? ..... 3
Jakie są korzyści? ..... 3
Architektura ..... 4
***2. Sprawdzenie wymagań wstępnych*** ..... 5
2.1 Wymagane dla wszystkich ..... 5
2.2 Dodatkowo dla użytkowników Claude Desktop ..... 5
2.3 Dodatkowo dla użytkowników Gemini CLI ..... 5
2.4 Uprawnienia administratora — kiedy są potrzebne? ..... 6
***3. Wspólne kroki (wykonaj raz)*** ..... 7
Krok 3.1 — Instalacja Node.js (wymagana) ..... 7
Wybierz ścieżkę ..... 7
ŚCIEŻKA A — Szybka instalacja za pomocą skryptów (zalecana) ..... 8
ŚCIEŻKA B — Ręcznie, krok po kroku (alternatywa dla ścieżki A) ..... 10
Krok B1 — Ustawienie zasad wykonywania PowerShell ..... 10
Krok B2 — Moduł CredentialManager ..... 10
Krok B3 — Utworzenie folderu serwera MCP ..... 11
Krok B4 — Umieszczenie directo-mcp.js ..... 11
Krok B5 — Utworzenie package.json ..... 12
Krok B6 — Instalacja zależności ..... 12
Krok B7 — Bezpieczne zapisanie klucza API ..... 13
***4. Konfiguracja Claude Desktop*** ..... 13
Krok 4.1 — Lokalizacja pliku konfiguracyjnego ..... 14
Krok 4.2 — Plik konfiguracyjny ..... 15
Krok 4.3 — Ponowne uruchomienie Claude Desktop ..... 16
Krok 4.4 — Testowanie ..... 16
***5. Konfiguracja Gemini CLI*** ..... 17
Krok 5.1 — Instalacja Gemini CLI ..... 17
Krok 5.2 — Ustawienie klucza API Google ..... 18
Krok 5.3 — Lokalizacja pliku konfiguracyjnego ..... 18
Krok 5.4 — Dodanie serwera MCP ..... 19
Krok 5.5 — Zaufanie do folderu (Trusted Folders) ..... 19
Krok 5.6 — Testowanie ..... 20
***6. Rozwiązywanie problemów*** ..... 21
Typowe błędy (dowolny klient) ..... 21
Błędy specyficzne dla Gemini CLI ..... 22
***7. Narzędzia MCP*** ..... 23
Warto wiedzieć ..... 23
***8. Bezpieczeństwo*** ..... 24
***9. Aktualizacja i wersje*** ..... 24
Której wersji używam? ..... 24
Jak zaktualizować ..... 24
***10. Źródła*** ..... 25

# Rozwiązanie AI Ask Directo: po prostu zapytaj, a AI odpowie na podstawie Twoich danych w Directo

Rozwiązanie Ask Directo opiera się na serwerze MCP (Model Context Protocol), czyli – w uproszczeniu – małym lokalnym programie pomocniczym działającym na Twoim komputerze. Instalując MCP na swoim komputerze, pozwalasz asystentowi AI (Claude, Gemini) bezpiecznie odczytywać dane bezpośrednio z Twojego systemu Directo. Wystarczy, że zadasz pytanie zwykłym językiem.

Więcej o możliwościach Ask Directo: https://directo.ee/ask-directo.

## Konfiguracja serwera MCP Directo – Claude Desktop i Gemini CLI

Ten przewodnik wyjaśnia, jak połączyć system ERP Directo z asystentem AI (Claude lub Gemini) za pośrednictwem lokalnego serwera MCP (Model Context Protocol). Przewodnik jest przeznaczony zarówno dla użytkowników Claude Desktop, jak i Gemini CLI – wspólne kroki wykonuje się tylko raz, a następnie wybiera się sekcję dla swojego klienta AI.

## 1. Wprowadzenie – dlaczego MCP?

### Czym jest MCP?

MCP (Model Context Protocol) to otwarty standard, który pozwala asystentom AI komunikować się bezpośrednio z systemami zewnętrznymi — bazami danych, API (interfejsami programistycznymi aplikacji) i aplikacjami.

**Bez MCP** praca z asystentem AI wygląda tak:

1. Kopiujesz dane z Directo
2. Wklejasz je do rozmowy z AI
3. AI odpowiada
4. Kopiujesz odpowiedź z powrotem do Directo

**Z MCP** wygląda to tak:

1. Pytasz zwykłym językiem: „Które towary są poniżej minimalnego stanu magazynowego?”
2. AI samo odpytuje API Directo (interfejs programistyczny aplikacji)
3. AI analizuje dane i od razu odpowiada

### Jakie są korzyści?

| Sytuacja | Bez MCP | Z MCP |
|---|---|---|
| Sprawdzenie stanu magazynowego | Otwórz Directo → wyszukaj → skopiuj → wklej do AI | „Które towary mają mniej niż 10 sztuk na stanie?” |
| Analiza faktur | Eksportuj CSV → prześlij do AI → zapytaj | „Którzy klienci nie zapłacili w ciągu ostatnich 90 dni?” |
| Raport sprzedaży | Przygotuj ręcznie → skopiuj → przeanalizuj | „Porównaj sprzedaż z tego miesiąca z ubiegłym rokiem” |
| Śledzenie zamówień | Otwórz kilka widoków → skopiuj informacje | „Które zamówienia są opóźnione o więcej niż 5 dni?” |

## Architektura

```
Ty (zwykły język)
        ↕
Klient AI (Claude Desktop lub Gemini CLI)
        ↕  Protokół MCP (lokalnie)
Serwer MCP (na Twoim komputerze, Node.js)
        ↕  HTTPS
API Directo (chmura)
```

Serwer MCP działa **na Twoim komputerze**. Klucz API Directo nigdy nie opuszcza Twojego komputera. Dane pobrane z Directo są jednak przesyłane dalej do chmury dostawcy AI, aby model mógł je odczytać i sformułować odpowiedź w zwykłym języku – tak samo jak każda inna treść wiadomości w rozmowie.



**Instalacja lokalna — ważne, aby to zrozumieć:**

- Serwer MCP jest instalowany **tylko na Twoim własnym komputerze**
- **Nie jest współdzielony** — współpracownicy nie mają dostępu do Twojego serwera MCP
- Każdy użytkownik musi wykonać wszystkie kroki **samodzielnie**, na własnym komputerze
- Serwer MCP działa tylko wtedy, gdy **Twój komputer jest włączony**, a klient AI uruchomiony
- Jeśli chcesz udostępnić serwer MCP wielu użytkownikom, potrzebujesz osobnego rozwiązania z serwerem hostowanym (wykracza to poza zakres tego przewodnika)
- Jeden klucz API należy do **jednej bazy danych Directo**. Ten serwer MCP jest powiązany dokładnie z bazą, w której utworzono klucz, i nie może odpytywać innej bazy (np. przełączać się między bazą produkcyjną a testową). Samo AI nie wie, z którą bazą połączony jest serwer: jeśli twierdzi, że odpytuje inną bazę, to nieprawda. Pasek adresu Directo pokazuje nazwę bazy (`login.directo.ee/BAZA/…`): upewnij się, że klucz API został utworzony we właściwej bazie.

# 2. Sprawdzenie wymagań wstępnych

Przed instalacją sprawdź wszystkie wymagania. Brak któregoś z nich to najczęstsza przyczyna niepowodzenia.

## 2.1 Wymagane dla wszystkich

| Wymaganie | Jak sprawdzić | Rozwiązanie |
|---|---|---|
| Windows 10 lub 11 (64-bit) | Start → Ustawienia → System → Informacje | — |
| Node.js 22 lub nowszy (LTS) | `node –version` | nodejs.org → LTS |
| npm | `npm –version` (działa dopiero po ustawieniu zasad wykonywania, patrz niżej) | Instalowany razem z Node.js |
| Zasady wykonywania PowerShell | `Get-ExecutionPolicy` → powinno zwrócić RemoteSigned | Patrz niżej |
| Moduł CredentialManager | `Get-Module -ListAvailable CredentialManager` | Patrz niżej |
| Klucz API Directo | Directo → Admin → Settings → API | Zwróć się do administratora Directo |

## 2.2 Dodatkowo dla użytkowników Claude Desktop

| Wymaganie | Jak sprawdzić | Link |
|---|---|---|
| Zainstalowany Claude Desktop | Claude jest widoczny w menu Start | https://claude.ai/download |
| Konto Anthropic | Logowanie działa | https://claude.ai |

Oficjalna dokumentacja: https://docs.claude.ai

## 2.3 Dodatkowo dla użytkowników Gemini CLI



⚠ **Gemini CLI nie obsługuje już użytkowników zalogowanych bezpłatnym kontem Google.** 18 czerwca 2026 r. Gemini CLI przestało obsługiwać subskrybentów Google AI Pro i Ultra oraz użytkowników bezpłatnych (Gemini Code Assist dla osób indywidualnych), którzy zostali przeniesieni do Antigravity CLI. Gemini CLI można nadal używać z:

- **płatnym kluczem Gemini API**
- **licencją Gemini Code Assist Standard / Enterprise**

Według strony limitów Google bezpłatny (nieopłacony) klucz Gemini API jest ograniczony do 250 zapytań dziennie i wyłącznie modeli Flash. Komunikaty Google na ten temat są jednak sprzeczne (strona limitów wymienia tę opcję, ale baner informuje, że użytkownicy „unpaid tier” zostali przeniesieni do Antigravity CLI), dlatego zanim zaczniesz na nim polegać, sprawdź, czy Twój klucz działa z Gemini CLI. Ten przewodnik nie obejmuje konfiguracji MCP w Antigravity CLI.

| Wymaganie | Jak sprawdzić | Link |
|---|---|---|
| Konto Google | Dostęp do gmail.com działa | https://accounts.google.com |
| Klucz API Google AI Studio | https://aistudio.google.com/apikey | Utwórz klucz API |
| Zainstalowane Gemini CLI | `gemini –version` | `npm install -g @google/gemini-cli` |

Oficjalna dokumentacja: https://geminicli.com/docs

## 2.4 Uprawnienia administratora — kiedy są potrzebne?



**Dobra wiadomość:** prawie każdy krok można wykonać **bez** uprawnień administratora. Uprawnienia administratora są naprawdę potrzebne tylko w jednym momencie — przy instalacji Node.js.

| Krok | Uprawnienia administratora? | Powód |
|---|---|---|
| Instalacja Node.js | **Tak** — wymaga tego instalator | Instalacja na poziomie systemu (Program Files) |
| Zasady wykonywania PowerShell | Nie | `-Scope CurrentUser` zapisuje w rejestrze HKCU |
| Moduł CredentialManager | Nie | `-Scope CurrentUser` instaluje w profilu użytkownika |
| npm install (serwer MCP) | Nie | Instalacja na poziomie użytkownika |
| Konfiguracja Claude Desktop | Nie | Foldery użytkownika |
| Konfiguracja Gemini CLI | Nie | Foldery użytkownika |
| Zapisanie klucza API | Nie | Menedżer poświadczeń, wpis na poziomie użytkownika |

**Ryzyka i ważne ostrzeżenie:**

- Instalator Node.js — niskie ryzyko, oficjalne oprogramowanie
- **Żadnego z pozostałych kroków nie uruchamiaj jako administrator**, nawet jeśli sugeruje to wcześniejsza wersja tego przewodnika lub przykłady znalezione w internecie — `-Scope CurrentUser` istnieje właśnie po to, aby uprawnienia administratora nie były potrzebne
- Jeśli Twoje codzienne konto i uprawnienia administratora pochodzą z **osobnych kont Windows** (częste w środowiskach firmowych/domenowych), uruchomienie czegoś jako administrator wykonuje operację w profilu *innego* konta — moduł, ustawienie lub klucz „znikają” wtedy z perspektywy Twojego zwykłego użytkownika. Szczegóły w rozdziale o rozwiązywaniu problemów.
- Nie uruchamiaj `npm install` jako administrator — powoduje to konflikty uprawnień
- `npm install -g` (np. Gemini CLI, krok 5.1) wymaga uprawnień administratora tylko wtedy, gdy globalny prefiks npm wskazuje na folder `Program Files` — krok 5.1 zawiera rozwiązanie, które tego unika

# 3. Wspólne kroki (wykonaj raz)



Te kroki wykonujesz **na własnym komputerze**. Jeśli współpracownik chce korzystać z tej samej konfiguracji, musi wykonać te kroki samodzielnie **na swoim komputerze**.

## Krok 3.1 — Instalacja Node.js (wymagana)



To jedyny krok, który jest **wymagany na obu ścieżkach** opisanych poniżej i którego nie można zautomatyzować (wymaga uprawnień administratora). Wykonaj go przed przejściem dalej, niezależnie od wybranej ścieżki.

1. Wejdź na https://nodejs.org i pobierz wersję **LTS** (22 lub nowszą)
2. Uruchom pobrany plik `.msi` jako administrator
3. Zainstaluj z ustawieniami domyślnymi — nie zmieniaj lokalizacji instalacji

Sprawdź instalację (zwykły PowerShell, nie jako administrator):

```
node --version
```

Oczekiwany wynik:

```
v22.11.0 lub nowsza
```



Celowo nie sprawdzamy tu jeszcze `npm –version` — npm to skrypt PowerShell (`npm.ps1`) i przed ustawieniem zasad wykonywania (skrypt ścieżki A lub krok B1) zwróci błąd `running scripts is disabled`, nawet jeśli Node.js i npm są poprawnie zainstalowane. To nie jest błąd, lecz oczekiwane zachowanie na tym etapie. `npm –version` zadziała po ukończeniu ścieżki A/B.



Jeśli PowerShell zgłasza „not recognized” — zamknij okno i otwórz je ponownie. Node.js dodaje się do zmiennej PATH tylko w nowych sesjach PowerShell.

## Wybierz ścieżkę



Poniżej opisano **dwie alternatywne ścieżki** prowadzące do tego samego wyniku — działającego serwera MCP z ustawionym kluczem API i klientem AI. **Wykonaj tylko JEDNĄ z nich, nie obie.**

| | Ścieżka A — Skrypty | Ścieżka B — Ręcznie |
|---|---|---|
| Czas | ~5 minut | ~20 minut |

| | Ścieżka A — Skrypty | Ścieżka B — Ręcznie |
|---|---|---|
| Najlepsza dla | Większości użytkowników | Gdy chcesz samodzielnie zrozumieć każdy krok lub ścieżka A nie działa w Twoim środowisku |
| Obejmuje kroki | A1—A2 | B1—B7 |

Jeśli wybierzesz ścieżkę A i zakończy się ona powodzeniem, **całkowicie pomiń ścieżkę B (B1—B7)** — te kroki opisują ręcznie dokładnie to, co skrypty już za Ciebie zrobiły. Ścieżka B to materiał pomocniczy do rozwiązywania problemów, a nie dodatkowy obowiązkowy krok.

## ŚCIEŻKA A — Szybka instalacja za pomocą skryptów (zalecana)

### Krok A1 — Pobranie plików

Pobierz `directo-mcp-files.zip` i rozpakuj go:

directo-mcp-files.zip

Pliki są dołączone do wiki jako **archiwum ZIP**, a nie jako osobne linki — DokuWiki domyślnie blokuje przesyłanie plików `.js` i `.ps1` (ze względów bezpieczeństwa, patrz `conf/mime.conf`), natomiast `.zip` jest domyślnie dozwolony. Nie zmieniaj rozszerzenia plików (np. na `.txt`) jako „poprawki” — złe rozszerzenie jest właśnie przyczyną późniejszych błędów typu `MODULE_NOT_FOUND`.

Skrypty wyświetlają wszystkie komunikaty po angielsku: `[OK]` = powodzenie, `[!]` = wymaga Twojej uwagi, `[ERROR]` = niepowodzenie.

Skrypty są od czasu do czasu aktualizowane. Jeśli już wcześniej pobrałeś ZIP (np. starsza kopia leży w folderze Pobrane), **zawsze pobierz świeżą wersję**. Po uruchomieniu skryptu jego pierwszy wiersz pokazuje numer wersji (np. `v1.5`). Jeśli nie widzisz numeru wersji, masz starą kopię.

Rozpakuj archiwum (prawy przycisk myszy na pliku ZIP → **Wyodrębnij wszystkie** lub w PowerShell):

```
Expand-Archive "$env:USERPROFILE\Downloads\directo-mcp-files.zip" -
DestinationPath "$env:USERPROFILE\Downloads\directo-mcp-files"
cd "$env:USERPROFILE\Downloads\directo-mcp-files"
```

Przed uruchomieniem skryptów trzeba zdjąć dwa domyślne ograniczenia bezpieczeństwa Windows — oba są wymagane, nie opcjonalne:

**1) Zezwól w ogóle na uruchamianie skryptów w PowerShell:**

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
```



Bez tego każdy kolejny skrypt zakończy się błędem `running scripts is disabled on this system` / `UnauthorizedAccess`. `-Scope CurrentUser` nie wymaga uprawnień administratora.

**2) Usuń znacznik Windows „pobrano z internetu” (Mark of the Web):**

```
Get-ChildItem "$env:USERPROFILE\Downloads\directo-mcp-files" -Recurse |
Unblock-File
```



To również jest wymagane, nie opcjonalne — pliki pobrane przez przeglądarkę są domyślnie blokowane **nawet jeśli zasady wykonywania są już ustawione poprawnie**. Windows oznacza pliki pobrane z internetu osobnym znacznikiem bezpieczeństwa, który przenosi się także na pliki rozpakowane z archiwum ZIP — `RemoteSigned` wymaga dla takich plików podpisu cyfrowego, którego nasze skrypty nie mają. Bez tego kroku zobaczysz `is not digitally signed` / `UnauthorizedAccess`, co wygląda myląco podobnie do błędu z punktu 1, ale ma inną przyczynę i inne rozwiązanie.

Archiwum zawiera cztery pliki: `directo-mcp.js` (sam serwer MCP), `set-directo-apikey.ps1`, `install-claude-mcp.ps1`, `install-gemini-mcp.ps1`.

**Krok A2 — Uruchomienie skryptów**

W tym samym oknie PowerShell (nie jako administrator), w właśnie rozpakowanym folderze:
A2.1 Klucz API (wspólny dla obu klientów, wykonaj najpierw)

```
.\set-directo-apikey.ps1
```

A2.2 Wybierz swojego klienta AI (lub obu):
Claude

```
.\install-claude-mcp.ps1
```

Gemini

```
.\install-gemini-mcp.ps1
```

Na końcu każdy skrypt wyświetla podsumowanie: `[OK]` = gotowe, `[!]` = nadal wymaga Twojego działania (np. pełnego ponownego uruchomienia Claude Desktop), `[ERROR]` = niepowodzenie, z komunikatem, co zrobić.



**Jeśli skrypty zakończyły się powodzeniem, rozdział 3 jest zakończony.** Przejdź od razu do rozdziału 4.3 (Claude Desktop) i/lub rozdziału 5 (Gemini CLI) — ścieżka B (poniżej) nie jest potrzebna.

## ŚCIEŻKA B — Ręcznie, krok po kroku (alternatywa dla ścieżki A)



To **alternatywa dla ścieżki A**, a nie jej uzupełnienie. Wykonaj ją tylko wtedy, gdy skrypty ścieżki A nie zadziałały lub chcesz samodzielnie przejść przez każdy krok. Jeśli ścieżka A zakończyła się powodzeniem, pomiń tę sekcję.

### Krok B1 — Ustawienie zasad wykonywania PowerShell

Sprawdź bieżące ustawienie:

```
Get-ExecutionPolicy
```

Jeśli odpowiedź jest inna niż `RemoteSigned`, popraw to w **zwykłym** oknie PowerShell (nie jako administrator):

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
```



`-Scope CurrentUser` zapisuje ustawienie tylko w gałęzi rejestru `HKEY_CURRENT_USER`, a nie w lokalizacji dla całego komputera — dlatego nie wymaga uprawnień administratora. Uprawnienia administratora byłyby potrzebne tylko dla `-Scope LocalMachine`, którego ten przewodnik nie używa.

Pojawi się prośba o potwierdzenie → naciśnij Y → Enter.

### Krok B2 — Moduł CredentialManager

Otwórz **zwykłe** okno PowerShell (nie jako administrator) — na dokładnie tym samym koncie użytkownika, na którym później będzie działał serwer MCP:

```
Install-Module -Name CredentialManager -Force -Scope CurrentUser
```



`-Scope CurrentUser` istnieje właśnie po to, aby nie były potrzebne uprawnienia administratora — moduł instaluje się w Twoim własnym profilu (`Documents\WindowsPowerShell\Modules`). Jeśli polecenie poprosi o potwierdzenie dostawcy NuGet lub „niezaufanego repozytorium”, odpowiedz Y — te kroki również nie wymagają uprawnień administratora.

Sprawdź, czy instalacja się powiodła (w tym samym zwykłym oknie PowerShell):

```
Get-Module -ListAvailable CredentialManager
```

Powinna zostać wyświetlona nazwa i wersja modułu.



⚠ Jeśli Twoje codzienne konto służbowe i uprawnienia administratora pochodzą z **osobnych kont Windows** (częste w konfiguracjach firmowych/domenowych), upewnij się, że ten krok i późniejszy krok B7 są wykonywane na **tym samym koncie**, na którym później będzie uruchamiany `node directo-mcp.js`. Okno PowerShell otwarte z podwyższonymi uprawnieniami może w rzeczywistości działać z tożsamością innego konta, przez co moduł lub klucz API trafi do niewłaściwego profilu, a późniejszy krok niezauważalnie się nie powiedzie.

## Krok B3 — Utworzenie folderu serwera MCP

Otwórz zwykłe okno PowerShell (nie jako administrator) i uruchom:

```
New-Item -ItemType Directory -Force -Path "$env:USERPROFILE\directo-mcp"
```



Zawsze używaj zmiennej `$env:USERPROFILE` zamiast ręcznie wpisywać `C:\Users\TWOJANAZWA`. Na komputerach przyłączonych do domeny (środowiska firmowe/AD) rzeczywisty folder profilu może różnić się od nazwy logowania — może np. zawierać sufiks domeny (`TWOJANAZWA.DOMENA`). `$env:USERPROFILE` zawsze podaje rzeczywistą, poprawną ścieżkę.

## Krok B4 — Umieszczenie directo-mcp.js



Ten plik zawiera wszystkie 41 narzędzi (40 interfejsów API Directo oraz narzędzie przeglądu dostępu `get_api_access`), a parametry każdego narzędzia zostały sprawdzone z rzeczywistymi specyfikacjami Directo. Nie kopiuj tego pliku ręcznie z Notatnika ani z żadnego innego źródła — zawsze używaj świeżej, oficjalnej kopii, aby nie skończyć z nieaktualną lub błędnie sparametryzowaną wersją.

Pobierz plik `directo-mcp.js` — znajduje się w archiwum `directo-mcp-files.zip` podlinkowanym w ścieżce A (rozpakuj je, patrz krok A1) — i umieść go w tym folderze:

```
$env:USERPROFILE\directo-mcp\directo-mcp.js
```

Najprostszy sposób: jeśli plik jest już w rozpakowanym podfolderze Pobrane, skopiuj go w PowerShell:

```
Copy-Item "$env:USERPROFILE\Downloads\directo-mcp-files\directo-mcp.js"
"$env:USERPROFILE\directo-mcp\directo-mcp.js"
```



Upewnij się, że nazwa pliku to dokładnie `directo-mcp.js` — a nie `directo-`

mcp.js.txt. Sprawdź:

```
Test-Path "$env:USERPROFILE\directo-mcp\directo-mcp.js"
```

Powinno zwrócić True. Jeśli False, sprawdź dokładną nazwę pliku:

```
Get-ChildItem "$env:USERPROFILE\directo-mcp"
```

## Krok B5 — Utworzenie package.json

Otwórz nowe okno Notatnika i wklej:

```
{
  "name": "directo-mcp",
  "version": "1.0.0",
  "type": "module",
  "description": "Directo ERP MCP Server",
  "main": "directo-mcp.js",
  "scripts": {
    "start": "node directo-mcp.js"
  },
  "dependencies": {
    "@modelcontextprotocol/sdk": "^1.0.0",
    "zod": "^3.22.0"
  }
}
```

Zapisz go w tym samym folderze pod nazwą package.json (typ pliku: Wszystkie pliki).

## Krok B6 — Instalacja zależności

Zwykły PowerShell (nie jako administrator):

```
cd "$env:USERPROFILE\directo-mcp"
npm install
```

Pomyślna instalacja tworzy w tym folderze folder node_modules oraz plik package-lock.json.

To **lokalna** instalacja npm (trafia do własnego folderu node_modules w folderze directo-mcp), a nie globalna — nigdy nie wymaga uprawnień administratora ani specjalnej konfiguracji prefiksu npm, niezależnie od tego, jak pierwotnie zainstalowano Node.js.

### Krok B7 — Bezpieczne zapisanie klucza API

Klucz API Directo jest zapisywany w Menedżerze poświadczeń Windows — zaszyfrowany, wyłącznie w ramach Twojego konta użytkownika. Nikt inny na tym komputerze nie widzi tego klucza.

```
New-StoredCredential -Target DIRECTO_API_KEY -UserName directo -Password
YOUR_REAL_KEY_HERE -Persist LocalMachine
```

Sprawdź, czy zapis się powiódł:

```
(Get-StoredCredential -Target
DIRECTO_API_KEY).GetNetworkCredential().Password
```

Powinien zostać wyświetlony Twój klucz API.

**Zmiana (rotacja) klucza API w przyszłości:**

```
Remove-StoredCredential -Target DIRECTO_API_KEY
New-StoredCredential -Target DIRECTO_API_KEY -UserName directo -Password
NEW_KEY -Persist LocalMachine
```



Wykonaj ten krok na **dokładnie tym samym koncie użytkownika**, na którym później będzie uruchamiany `node directo-mcp.js`. Patrz ostrzeżenie w kroku B2 dotyczące ryzyka osobnego konta administratora.



Skrypt ścieżki A (`set-directo-apikey.ps1`) robi to samo w bezpieczniejszy sposób (pyta o klucz bez wyświetlania go na ekranie) i działa również przy rotacji — używaj go, jeśli możesz.

## 4. Konfiguracja Claude Desktop



Najpierw ukończ rozdział 3 (krok 3.1 + ścieżka A albo ścieżka B). Konfiguracja Claude Desktop zakłada, że serwer MCP już istnieje.



Serwer MCP działa tylko **na Twoim własnym komputerze** i tylko w **aplikacji Claude Desktop** — nie w wersji webowej claude.ai i nie na komputerach innych użytkowników.



**Zalecane:** jeśli w ramach ścieżki A (rozdział 3) uruchomiłeś skrypt `install-claude-mcp.ps1`, kroki 4.1—4.2 są już wykonane — skrypt automatycznie wyszukuje, modyfikuje i tworzy kopię zapasową pliku konfiguracyjnego. Przejdź od

razu do kroku 4.3.

## Krok 4.1 — Lokalizacja pliku konfiguracyjnego

Plik konfiguracyjny Claude Desktop `claude_desktop_config.json` znajduje się w systemie Windows w jednym z czterech miejsc, **w zależności od wersji i sposobu instalacji**. Skrypt ścieżki A sam znajduje właściwy plik. W przypadku ścieżki ręcznej (ścieżka B) znajdź go, wykonując poniższe kroki.

| Wariant | Lokalizacja |
|---|---|
| MSIX (claude.ai/download), dane przeniesione (Claude Desktop 3.1623 i nowsze) | `%LOCALAPPDATA%\Claude-Data\claude_pzs8sxrjxfjjc\` |
| Ten sam MSIX przed przeniesieniem danych | `%LOCALAPPDATA%\Packages\Claude_pzs8sxrjxfjjc\LocalCache\Roaming\Claude\` |
| Wersja z Microsoft Store (niezweryfikowane) | `%LOCALAPPDATA%\Claude-Data\anthropicpbc.claude_fnn82j28hfe8t\` |
| Wcześniejszy instalator `.exe` | `%APPDATA%\Claude\` |

Nowsze wersje Claude Desktop przeniosły swoje dane ze starego folderu pakietu do folderu `Claude-Data`. W starym folderze pozostaje tylko znacznik `.user-data-left…` (na niektórych komputerach także nieaktualny plik `claude_desktop_config.json`). **Plik, którego Claude już nie odczytuje, można edytować bez żadnego błędu, ale zmiana nigdy nie dociera do aplikacji**, a serwer MCP po cichu się nie ładuje. Zawsze używaj nowego folderu, jeśli istnieje.

Wyświetl, które z tych plików istnieją na Twoim komputerze (zwykły PowerShell):

```
"$env:LOCALAPPDATA\Claude-
Data\claude_pzs8sxrjxfjjc\claude_desktop_config.json",
"$env:LOCALAPPDATA\Packages\Claude_pzs8sxrjxfjjc\LocalCache\Roaming\Claude\c
laude_desktop_config.json",
"$env:LOCALAPPDATA\Claude-
Data\anthropicpbc.claude_fnn82j28hfe8t\claude_desktop_config.json",
"$env:APPDATA\Claude\claude_desktop_config.json" | Where-Object { Test-Path
$_ }
```

Użyj **pierwszego** pliku z listy (kolejność jest taka sama jak w tabeli: nowsze przed starszymi). Wyjątki:

- Jeśli lista zawiera plik `Packages\…`, a ten sam folder zawiera też plik `.user-data-left…`, ten plik jest nieaktualny — nie używaj go.
- Jeśli folder `%LOCALAPPDATA%\Claude-Data\claude_pzs8sxrjxfjjc` istnieje, ale nie ma w

nim pliku `claude_desktop_config.json`: otwórz Claude Desktop → Settings → Developer → Edit Config (to tworzy plik) i uruchom polecenie ponownie.
- Jeśli lista jest pusta: uruchom Claude Desktop co najmniej raz i ponownie uruchom polecenie.

## Krok 4.2 — Plik konfiguracyjny

Utwórz plik o nazwie `claude_desktop_config.json` we (właściwym, ustalonym w kroku 4.1) folderze.



Przy tworzeniu pliku w Notatniku: Plik → Zapisz jako → typ pliku **Wszystkie pliki (*.*)** → nazwa: `claude_desktop_config.json`

Zawartość pliku:

```
{
  "mcpServers": {
    "directo": {
      "command": "node",
      "args": [
        "C:\\Users\\YOURNAME\\directo-mcp\\directo-mcp.js"
      ]
    }
  }
}
```



Zastąp YOURNAME **rzeczywistą** nazwą folderu swojego profilu — a nie domniemaną nazwą użytkownika. Na komputerach przyłączonych do domeny (AD) rzeczywisty folder profilu może różnić się od nazwy logowania (np. zawierać sufiks domeny). Sprawdź dokładną wartość:

```
$env:USERPROFILE
```

W pliku JSON użyj dokładnie tego, co zwraca polecenie (z podwójnymi ukośnikami odwrotnymi). Przykład poglądowy — jeśli `$env:USERPROFILE` zwraca `C:\Users\john.user.COMPANY`, ścieżka w JSON powinna mieć postać `C:\\Users\\john.user.COMPANY\\directo-mcp\\directo-mcp.js`. To, czy sufiks domeny (`.COMPANY`) występuje, zależy od konfiguracji Twojego komputera — zawsze używaj rzeczywistego wyniku własnego polecenia, a nie tego przykładu.



Jeśli plik zawiera już inną treść (własne ustawienia Claude Desktop, np. blok `preferences`) — **nie nadpisuj jej**. `mcpServers` musi znajdować się na **najwyższym poziomie** pliku, na tym samym poziomie co `preferences`, a nie w jego wnętrzu. Jeśli nie masz pewności co do poprawnego rozmieszczenia nawiasów klamrowych, użyj skryptu ścieżki A, który edytuje plik jako strukturę (`ConvertFrom-Json`/`ConvertTo-Json`), a nie jako surowy tekst — to całkowicie eliminuje błędy w

rozmieszczeniu nawiasów.

Klucz API **nie** jest dodawany do pliku konfiguracyjnego — serwer automatycznie odczytuje go z Menedżera poświadczeń (skrypt ścieżki A lub krok B7).

## Krok 4.3 — Ponowne uruchomienie Claude Desktop

1. Kliknij prawym przyciskiem myszy ikonę Claude w **zasobniku systemowym** (przy zegarze, na pasku zadań) → **Quit** / **Exit**
2. Samo zamknięcie okna nie wystarczy — Claude Desktop nadal działa w tle
3. Jeśli opcja Quit nie jest dostępna — otwórz Menedżer zadań → znajdź proces Claude → Zakończ zadanie
4. Ponownie otwórz Claude Desktop i odczekaj kilka sekund

## Krok 4.4 — Testowanie

W rozmowie w Claude Desktop wpisz najpierw:

> Które interfejsy Directo są włączone dla mojego klucza API?

Claude użyje narzędzia `get_api_access` i wyświetli listę interfejsów. To zapytanie nie zwraca żadnych danych biznesowych z Directo, więc jest najbezpieczniejszym pierwszym testem. Następnie wypróbuj zapytanie o dane, np.:

> Jakie magazyny mam w Directo?

Claude użyje odpowiedniego narzędzia (tutaj `get_stocks`) i zwróci dane z Directo. Nad odpowiedzią zobaczysz szarą etykietę „Used directo integration”. Jeśli interfejs nie jest włączony dla Twojego klucza, Claude poinformuje, który interfejs należy włączyć w Directo.

**Za pierwszym razem Claude Desktop prosi o zgodę.** Przy pierwszym użyciu nowego narzędzia w oknie Claude Desktop pojawia się pytanie o potwierdzenie (np. Allow once / Allow always). Wybierz `Allow always`, jeśli ufasz temu narzędziu. Jeśli nie ma odpowiedzi, a po kilku minutach pojawia się komunikat „No result received”, sprawdź, czy gdzieś nie jest otwarte okno potwierdzenia.

Upewnij się, że odpowiedź zawiera dane z **Twojej bazy**: zapytaj o coś, co znasz z własnego Directo (np. znanego klienta lub magazyn), i porównaj.

**Sprawdzanie logów, jeśli coś nie działa:**

W aktualnych wersjach Claude Desktop logi znajdują się w `%LOCALAPPDATA%\Claude\logs`:

```
# Aktualne wersje Claude Desktop:
Get-Content "$env:LOCALAPPDATA\Claude\logs\mcp-server-directo.log" -Tail 30
```

```
# Wcześniejszy instalator .exe:
Get-Content "$env:APPDATA\Claude\logs\mcp-server-directo.log" -Tail 30
```

Oficjalna dokumentacja MCP dla Claude Desktop: modelcontextprotocol.io - Connect to local MCP servers



Oficjalna dokumentacja MCP nadal podaje dla Windows stare ścieżki (`%APPDATA%\Claude\…`). Krok 4.1 tego przewodnika opisuje, gdzie Claude Desktop obecnie przechowuje swoje pliki.

# 5. Konfiguracja Gemini CLI



Najpierw ukończ rozdział 3 (krok 3.1 + ścieżka A albo ścieżka B). Konfiguracja Gemini CLI zakłada, że serwer MCP już istnieje.



Serwer MCP działa tylko **na Twoim własnym komputerze** i tylko w terminalowej aplikacji Gemini CLI — nie w wersji webowej Gemini i nie na komputerach innych użytkowników.



⚠ Zanim przejdziesz dalej, upewnij się, że masz prawidłowe uwierzytelnienie: płatny klucz Gemini API, licencję Gemini Code Assist Standard / Enterprise lub bezpłatny klucz API działający z Gemini CLI. Gemini CLI nie działa już z bezpłatnym kontem Google (patrz krok 2.3).



**Zalecane:** jeśli w ramach ścieżki A (rozdział 3) uruchomiłeś skrypt `install-gemini-mcp.ps1`, kroki 5.1 oraz 5.3—5.4 są już wykonane. Przejdź tylko krok 5.2 (klucz API Google — skrypt go nie ustawia, ponieważ jest to osobne uwierzytelnienie Google, a nie Directo) oraz kroki 5.5—5.6.

### Krok 5.1 — Instalacja Gemini CLI



Ten krok nie wymaga uprawnień administratora — pod warunkiem, że globalna lokalizacja instalacji npm wskazuje na Twój własny profil. Jeśli nie jest to jeszcze skonfigurowane, zrób to jednorazowo (zwykły PowerShell, nie jako administrator):

```
mkdir "$env:APPDATA\npm-global" -Force
npm config set prefix "$env:APPDATA\npm-global"
[Environment]::SetEnvironmentVariable("Path", $env:Path +
```

```
";$env:APPDATA\npm-global", "User")
```

Całkowicie zamknij PowerShell i otwórz **nowe** okno (PATH odświeża się tylko w nowej sesji), a następnie zainstaluj Gemini CLI:

```
npm install -g @google/gemini-cli
```

Sprawdź:

```
gemini --version
```



⚠ Jeśli polecenie nadal wymaga uprawnień administratora, prawdopodobną przyczyną jest to, że prefiks npm został już wcześniej ustawiony (np. przez inne oprogramowanie) na `C:\Program Files\…`. Sprawdź bieżące ustawienie poleceniem `npm config get prefix` — po powyższym kroku powinno pokazywać `%APPDATA%\npm-global`, a nie folder Program Files.

## Krok 5.2 — Ustawienie klucza API Google

Wejdź na https://aistudio.google.com/apikey i utwórz klucz API.

Zapisz klucz jako trwałą zmienną środowiskową:

```
[System.Environment]::SetEnvironmentVariable("GEMINI_API_KEY",
"YOUR_GOOGLE_API_KEY", "User")
```

Zamknij PowerShell i otwórz go ponownie. Sprawdź:

```
$env:GEMINI_API_KEY
```

## Krok 5.3 — Lokalizacja pliku konfiguracyjnego

Konfiguracja Gemini CLI znajduje się w systemie Windows w:

```
$env:USERPROFILE\.gemini\settings.json
```

Sprawdź, czy folder istnieje:

```
Test-Path "$env:USERPROFILE\.gemini"
```

Jeśli odpowiedź to `False`, utwórz folder:

```
New-Item -ItemType Directory -Force -Path "$env:USERPROFILE\.gemini"
```

## Krok 5.4 — Dodanie serwera MCP

Utwórz lub edytuj plik `$env:USERPROFILE\.gemini\settings.json`:

```
{
  "mcpServers": {
    "directo": {
      "command": "node",
      "args": [
        "C:\\Users\\YOURNAME\\directo-mcp\\directo-mcp.js"
      ],
      "env": {
        "DIRECTO_API_KEY": "$DIRECTO_API_KEY"
      }
    }
  }
}
```



Zastąp `YOURNAME` **rzeczywistą** nazwą folderu swojego profilu — sprawdź dokładną wartość poleceniem `$env:USERPROFILE` (patrz ostrzeżenie w kroku 4.2 na ten sam temat).



**Uwaga dotycząca logiki klucza API:** `directo-mcp.js` najpierw odczytuje klucz z Menedżera poświadczeń (skrypt ścieżki A lub krok B7) — działa to **dokładnie tak samo dla Gemini jak dla Claude Desktop**, ponieważ oba uruchamiają ten sam plik serwera. Blok env jest tu **rozwiązaniem zapasowym**, a nie jedyną drogą — jeśli Menedżer poświadczeń jest już skonfigurowany, ten blok nie jest konieczny, choć też nie szkodzi. Jeśli chcesz go użyć, zapisz wartość jako zmienną środowiskową:

```
[System.Environment]::SetEnvironmentVariable("DIRECTO_API_KEY", "YOUR_DIRECTO_API_KEY", "User")
```

Gemini CLI obsługuje rozwijanie zmiennych środowiskowych w wartościach env za pomocą składni `$VAR_NAME` — dlatego JSON zawiera „`$DIRECTO_API_KEY`“, a nie sam klucz.

## Krok 5.5 — Zaufanie do folderu (Trusted Folders)

Gemini CLI ma funkcję bezpieczeństwa o nazwie **Trusted Folders**, która domyślnie nie pozwala łączyć serwerów MCP z niezaufanego folderu. Jeśli ta funkcja jest włączona, folder musi zostać oznaczony jako zaufany.

**Najprostszy sposób** — oznacz serwer jako zaufany bezpośrednio w `settings.json`, dodając „`trust“: true` do bloku `directo`:

```
{
  "mcpServers": {
    "directo": {
      "command": "node",
      "args": [
        "C:\\Users\\YOURNAME\\directo-mcp\\directo-mcp.js"
      ],
      "env": {
        "DIRECTO_API_KEY": "$DIRECTO_API_KEY"
      },
      "trust": true
    }
  }
}
```



„trust“: true pomija okna potwierdzania wywołań narzędzi. Używaj tego tylko dla serwerów, którymi w pełni zarządzasz samodzielnie — ten serwer Directo jest takim przypadkiem.

**Alternatywa** — jeśli funkcja Trusted Folders jest włączona, Gemini CLI przy pierwszym otwarciu tego folderu wyświetla okno zaufania. Wybierz tam **Trust folder**. Wybór jest zapisywany w `~/.gemini/trustedFolders.json` i pytanie pojawia się tylko raz dla każdego folderu. Aby później zmienić poziom zaufania, użyj w CLI polecenia `/permissions`.

### Krok 5.6 — Testowanie

Uruchom Gemini CLI:

```
gemini
```

Sprawdź połączenie MCP:

```
/mcp list
```

Powinno pokazać:

```
 directo - Ready (41 tools)
```

Zapytanie testowe:

> Które interfejsy Directo są włączone dla mojego klucza API?

Oficjalna dokumentacja MCP dla Gemini CLI: geminicli.com/docs/tools/mcp-server

# 6. Rozwiązywanie problemów

## Typowe błędy (dowolny klient)

| Komunikat błędu | Przyczyna | Rozwiązanie |
|---|---|---|
| `DIRECTO_API_KEY is missing` | Zapis w Menedżerze poświadczeń nie powiódł się lub serwer działa na innym koncie Windows | Powtórz krok B7 (lub ścieżka A: `.\set-directo-apikey.ps1`) na tym samym koncie, na którym działa Claude |
| `Unable to convert Credential object…` / `null-valued expression` | Klucz zapisano poleceniem cmdkey, co nie zawsze jest zgodne ze sposobem odczytu przez moduł CredentialManager | Usuń go poleceniem `Remove-StoredCredential -Target DIRECTO_API_KEY` i zapisz ponownie poleceniem `New-StoredCredential` (krok B7) lub skryptem ścieżki A |
| `Cannot find module` | npm install nie powiódł się LUB directo-mcp.js jest w złym folderze/ma złą nazwę | Powtórz krok B6; sprawdź `Test-Path „$env:USERPROFILE\directo-mcp\directo-mcp.js“` |
| `SyntaxError` | Plik directo-mcp.js jest uszkodzony lub niekompletny | Pobierz plik ponownie, patrz krok B4 (lub ścieżka A: A1) |
| `The „…“ interface is not enabled for this API key` | Ten interfejs nie jest włączony dla Twojego klucza API | Poproś administratora Directo o włączenie interfejsu dla klucza (API Direct). Zapytaj „Które interfejsy Directo są włączone dla mojego klucza API?”, aby zobaczyć, na co pozwala klucz |
| `Filter „…“ cannot be used … not selected for output` | To pole nie jest wybrane jako pole wyjściowe dla klucza | Użyj innego filtra lub poproś o dodanie pola do pól wyjściowych klucza |
| `Unknown filter name` | Błędna nazwa filtra (w nazwach rozróżniana jest wielkość liter) | Zapytaj, jakie filtry dopuszcza interfejs: „Jakie filtry dopuszcza interfejs …?” |
| `Filter „…“ has an invalid value syntax` | Wartość filtra ma zły format (data musi być w formacie ISO 8601, np. 2026-01-01T00:00:00) | Claude zwykle sam to poprawia; w razie potrzeby poproś go o użycie właściwego formatu |
| `The API key is missing or invalid (401)` | Klucz API Directo jest błędny, wygasł lub został usunięty | Sprawdź klucz w panelu administracyjnym Directo i zapisz go ponownie (`.\set-directo-apikey.ps1`) |
| `Access denied: the IP address of this computer is not allowed` | Klucz ma ograniczenie IP, a adres Twojego komputera nie jest dozwolony | Poproś administratora Directo o dopuszczenie adresu IP |
| `Rate limit exceeded (429)` | Zbyt wiele zapytań | Odczekaj chwilę i spróbuj ponownie (zalecany limit: do 60 zapytań na minutę) |
| `Directo reported a server error (5xx)` | Tymczasowy problem po stronie Directo | Spróbuj ponownie później; jeśli problem się utrzymuje, skontaktuj się z pomocą techniczną Directo |
| `running scripts is disabled` | Zasady wykonywania PowerShell | `Set-ExecutionPolicy RemoteSigned -Scope CurrentUser` (krok A1 lub B1) |

| Komunikat błędu | Przyczyna | Rozwiązanie |
|---|---|---|
| `is not digitally signed` / `UnauthorizedAccess` | Windows Mark-of-the-Web (plik pobrany przez przeglądarkę) | `Get-ChildItem <folder> -Recurse \| Unblock-File` (patrz krok A1) |
| `npm not recognized` | Node.js nie jest w zmiennej PATH | Zamknij i ponownie otwórz PowerShell |




Komunikaty błędów serwera są po angielsku. Claude wyjaśni je w Twoim języku.

| Problem | Rozwiązanie |
|---|---|
| Nie widać ikony młotka | Sprawdź plik logu (patrz niżej) |
| Serwer MCP się nie uruchamia | Sprawdź, czy nazwa pliku konfiguracyjnego nie ma rozszerzenia `.txt` |
| Brak folderu konfiguracyjnego | Uruchom raz Claude Desktop — foldery zostaną utworzone automatycznie |
| Konfiguracja zmieniona, ale nic się nie zmienia | Upewnij się, że edytujesz WŁAŚCIWY plik — patrz krok 4.1 (cztery możliwe lokalizacje; w aktualnym Claude Desktop plik znajduje się w `Claude-Data`, stary folder pakietu jest pusty lub nieaktualny) |
| Brak odpowiedzi (po kilku minutach: „No result received”) | Claude Desktop czeka na potwierdzenie użycia narzędzia: sprawdź, czy w Claude Desktop nie jest otwarte okno potwierdzenia, i wybierz Allow once / Allow always |
| MCP zniknął po aktualizacji Claude Desktop | Claude przeniósł swoje dane do nowego folderu (krok 4.1). Sprawdź, czy `claude_desktop_config.json` w nowym folderze zawiera wiersz z `directo-mcp.js` (`Select-String -Path „<plik>“ -Pattern „directo-mcp.js“`). Jeśli nie, uruchom ponownie skrypt ścieżki A. |

Plik logu (aktualne wersje Claude Desktop; drugi wiersz dotyczy wcześniejszego instalatora .exe):

```
Get-Content "$env:LOCALAPPDATA\Claude\logs\mcp-server-directo.log" -Tail 30
Get-Content "$env:APPDATA\Claude\logs\mcp-server-directo.log" -Tail 30
```

## Błędy specyficzne dla Gemini CLI

| Problem | Rozwiązanie |
|---|---|
| `/mcp list` pokazuje Disconnected | Dodaj `„trust“: true` do konfiguracji serwera (krok 5.4) lub oznacz folder jako zaufany w oknie dialogowym / poleceniem `/permissions` |
| Błąd `GEMINI_API_KEY` | Powtórz krok 5.2, otwórz nowe okno PowerShell |
| Limit zapytań / błąd 429 | Przekroczono limit zapytań bezpłatnego klucza API — poczekaj na odnowienie limitu lub użyj płatnego klucza API (patrz krok 2.3) |
| Gemini CLI nie loguje się kontem Google | Obsługa bezpłatnych kont Google zakończyła się 18 czerwca 2026 r.: użyj płatnego klucza Gemini API lub licencji Gemini Code Assist (patrz krok 2.3) |

Diagnostyka Gemini CLI:

```
/mcp list
```

# 7. Narzędzia MCP

Oba klienty (Claude Desktop i Gemini CLI) korzystają z tych samych 41 narzędzi (40 interfejsów API Directo oraz narzędzia przeglądu dostępu `get_api_access`):

| **Grupa** | **Narzędzia** |
|---|---|
| Sprzedaż | get_invoices, get_orders, get_sales_quotations, get_sales_contracts, get_receipts |
| Klienci | get_customers, get_suppliers, get_contacts |
| Towary / magazyn | get_items, get_itemclasses, get_stocklevels_sn, get_stocklevels, get_allocations, get_priceformulas, get_stocks |
| Zakupy | get_purchase_orders, get_purchase_invoices, get_purchase_payments, get_stock_receipts |
| Księgowość | get_transactions, get_accounts, get_financial_budgets, get_nettings |
| Logistyka | get_deliveries, get_movements |
| Pracownicy / kadry | get_users, get_absences, get_wage_calculations, get_expenses, get_approvals |
| Środki trwałe | get_assets, get_customer_assets |
| Inne | get_projects, get_objects, get_resources, get_events, get_recipes, get_replacementlogs, get_usagelogs |
| Synchronizacja | get_deleted |
| Przegląd dostępu | get_api_access |

## Warto wiedzieć

- **Narzędzie działa tylko wtedy, gdy jego interfejs jest włączony dla Twojego klucza API.** Administrator Directo wybiera, na które interfejsy i pola pozwala klucz. Jeśli brakuje interfejsu, komunikat błędu informuje, który interfejs należy włączyć w Directo.
- **Przegląd dostępu:** zapytaj „Które interfejsy Directo są włączone dla mojego klucza API?”. Wynik jest zapamiętywany na 1 dzień (plik `discovery-cache.json` w folderze `directo-mcp`). Jeśli zmienisz konfigurację klucza w Directo, powiedz Claude „Odśwież informacje o dostępie do Directo” (lub usuń ten plik i uruchom ponownie Claude Desktop). Przegląd Directo może pomijać niektóre włączone interfejsy (np. `stocklevels`), więc wypróbuj narzędzie: rzeczywisty komunikat błędu powie, czy brakuje dostępu.
- **Stany magazynowe:** preferuj narzędzie `get_stocklevels_sn`. Jeśli ten interfejs nie jest włączony dla klucza, użyj `get_stocklevels`.
- **Filtry:** każde narzędzie danych ma typowe filtry (np. data, klient, status) oraz filtr dowolny, który pozwala AI użyć dowolnego pola włączonego dla klucza (w tym pól wierszy). Filtrować można tylko pola wybrane jako wyjściowe dla klucza.
- **Rozmiar wyniku:** na początku każdej odpowiedzi podawana jest liczba trafień i rozmiar. Jeśli widnieje „LARGE” lub „TOO LARGE”, zawęź zapytanie (najlepiej do 1 miesiąca naraz). Bardzo duża odpowiedź może nie zmieścić się w pamięci AI, a sumy mogą okazać się niepełne.
- **Dane wrażliwe:** narzędzia dotyczące pracowników, wynagrodzeń, wydatków i dzienników użycia (`get_wage_calculations`, `get_absences`, `get_expenses`, `get_approvals`, `get_usagelogs`) działają, jeśli odpowiednie interfejsy są włączone dla Twojego klucza. O tym, czy zostaną one udostępnione AI, decyduje i odpowiada za to właściciel klucza.

# 8. Bezpieczeństwo

| Środek | Wyjaśnienie |
|---|---|
| Menedżer poświadczeń (oba klienty) | Klucz API jest zaszyfrowany i widoczny tylko dla Twojego konta Windows — Claude i Gemini odczytują ten sam wpis |
| Zmienna środowiskowa (rozwiązanie zapasowe dla Gemini) | Jeśli Menedżer poświadczeń jest skonfigurowany, blok `env` nie jest konieczny — patrz uwaga w kroku 5.4 |
| Serwer lokalny | Serwer MCP działa tylko na Twoim komputerze i nie jest publicznie dostępny |
| Dane przekazywane dostawcy AI | Wyniki zapytań są wysyłane do chmury dostawcy AI, aby model mógł na nie odpowiedzieć. Klucz API nigdy nie opuszcza Twojego komputera. |
| Minimalne uprawnienia | Ogranicz uprawnienia klucza API Directo tylko do potrzebnych interfejsów i pól (w tym danych pracowników i wynagrodzeń) |
| Pamięć podręczna przeglądu dostępu | Plik `discovery-cache.json` (w folderze `directo-mcp`) zawiera tylko nazwy interfejsów i pól, a nie dane, i jest powiązany z Twoim kluczem |
| OneDrive/Dropbox | Wyłącz folder `directo-mcp` z synchronizacji w chmurze |
| Uprawnienia administratora | Tylko do instalacji Node.js (krok 3.1) — wspólne wymaganie dla obu ścieżek — wszystko inne, łącznie z modułem CredentialManager, działa z `-Scope CurrentUser` bez uprawnień administratora (patrz 2.4) |

# 9. Aktualizacja i wersje

## Której wersji używam?

Wersja serwera MCP jest widoczna w wierszu startowym serwera w logu (`Directo MCP server v… started (N tools)`):

```
Select-String -Path "$env:LOCALAPPDATA\Claude\logs\mcp-server-directo.log" -Pattern "Directo MCP server v" | Select-Object -Last 1
```

Wersja skryptów jest widoczna w ich pierwszym wierszu podczas uruchamiania (np. `v1.5`).

## Jak zaktualizować

1. Pobierz **świeży** plik ZIP (patrz krok A1), rozpakuj go i otwórz PowerShell w rozpakowanym folderze.
2. Skopiuj nowy `directo-mcp.js` w miejsce starego (polecenie poniżej).
3. **Całkowicie** zamknij Claude Desktop (krok 4.3) lub Gemini CLI i otwórz ponownie.
4. Sprawdź wersję w logu (patrz wyżej).

Polecenie kopiowania:

```
Copy-Item .\directo-mcp.js "$env:USERPROFILE\directo-mcp\directo-mcp.js" -Force
```



Nowy `directo-mcp.js` nie wymaga nowych zależności, więc `npm install` nie jest potrzebne. Ponowne uruchomienie skryptów instalacyjnych **nie** aktualizuje pliku serwera: skrypt kopiuje `directo-mcp.js` tylko wtedy, gdy nie ma go jeszcze w folderze.

# 10. Źródła

**Claude Desktop:**

- Pobierz Claude Desktop
- Konfiguracja MCP w Claude Desktop
- Przewodnik: lokalne serwery MCP w Claude Desktop
- Foldery danych Claude Desktop w Windows (Anthropic)
- Oficjalna dokumentacja MCP

**Gemini CLI:**

- Dokumentacja Gemini CLI
- Konfiguracja MCP w Gemini CLI
- Limity i cennik Gemini CLI
- Przejście z Gemini CLI na Antigravity CLI (ogłoszenie)
- Klucze API Google AI Studio
- Dokumentacja Gemini Code Assist

**Directo:**

- Dokumentacja API Directo

**Ogólne:**

- Pobierz Node.js
- Moduł PowerShell CredentialManager

**Skrypty szybkiej instalacji** (patrz ścieżka A, rozdział 3):

- `directo-mcp.js` — serwer MCP (41 narzędzi)
- `set-directo-apikey.ps1` — zarządzanie kluczem API (konfiguracja początkowa + rotacja)
- `install-claude-mcp.ps1` — automatyczna instalacja dla Claude Desktop
- `install-gemini-mcp.ps1` — automatyczna instalacja dla Gemini CLI

From:
https://wiki.directo.ee/ - **Directo Help**

Permanent link:
**https://wiki.directo.ee/pl/ask_directo**

Last update: **2026/11/13 14:19**