SumizAI .com Twój drugi mózg, napędzany inteligencją.

https://sumizai.com/pl/blog/jak-dziala-spis-tresci.html

POD MASKĄ

Jak spis treści odkłada Twoje notatki?

Każdy osobisty system wiedzy umiera tak samo: przestaje działać odkładanie. Nie zbieranie — zbieranie jest łatwe i nawet przyjemne — tylko **odkładanie na półki**. Odpowiedź SumizAI polega na tym, żeby odkładanie było decyzją, którą model podejmuje przy każdej notatce, i żeby ta decyzja była ograniczona na tyle mocno, by pozostała spójna przez tysiące.

Aktualizacja: 2026-08-14 **8** pytań 3 min czytania

Q. Jaki kształt ma spis treści?

Drzewo rozdziałów, do sześciu poziomów, numerowane jednym ciągiem: `1.` → `1.1` → `a)` → `a.1)` → `a.1.1`. Notatki są liśćmi i biorą numer z tego samego ciągu, więc notatka ma adres, a nie tylko położenie.

Rozdziały mają nazwę, opis i pozycję, a etykiety są wyliczane z pozycji w drzewie, nie przechowywane. Przesuniesz rozdział — wszystko pod nim numeruje się od nowa.

To prawdziwa struktura utrzymywana przez aplikację, a nie widok generowany w locie z tagów. Właśnie dlatego można poprosić model, żeby umieścił w niej notatkę.

Q. Jak nowa notatka znajduje swoją półkę?

Model dostaje ponumerowane rozdziały i ma **wskazać numer**. Może też poprosić o nowy podrozdział pod istniejącym. Nowa gałąź najwyższego poziomu pojawia się wyłącznie, gdy nic nie pasuje nigdzie.

Wskazywanie zamiast nazywania to decyzja projektowa, która sprawia, że całość działa. Wcześniejsze podejście prosiło model o nazwanie właściwego rozdziału dla każdej notatki i dawało dokładnie to, czego można się spodziewać: *Warzywa*, *uprawa warzyw* i *Uprawa roślin warzywnych* jako trzy siostrzane gałęzie znaczące jedno.

Model czyta skrót drzewa — około 4 000 znaków — a nie pełny konspekt, więc koszt odkładania nie rośnie wraz z wielkością vaulta.

Q. Co, jeśli model odłoży coś w złe miejsce?

Przesuwasz to i nic w tym nie jest destrukcyjne. Drzewo edytuje się ręcznie: zmień nazwę rozdziału, kolejność, zagnieźdź głębiej, podnieś poziom wyżej, dodaj, usuń.

Usunięcie rozdziału nigdy nie osierocia notatek. Jego zawartość przechodzi do rodzica, bo jedynym niezmiennikiem, który aplikacja utrzymuje, jest to, że **notatka nigdy nie jest poza spisem treści**.

Jeśli jakaś notatka mimo wszystko zostanie nieodłożona — na przykład po przerwanej operacji — zostanie wciągnięta przy następnym odczycie spisu treści, zamiast siedzieć niewidocznie, dopóki jej nie poszukasz.

Q. Co przebudowa robi takiego, czego nie robi zwykłe odkładanie?

Widzi wszystko naraz. Odkładanie przy tworzeniu patrzy na jedną notatkę wobec istniejącego drzewa, co jest właściwą decyzją przy dostępnych informacjach — i nigdy nie naprawi struktury, która narosła krzywo, zanim ta notatka powstała.

Przebudowa czyta wszystkie notatki partiami po sześćdziesiąt i układa rozdziały od nowa modelem głównym. Notatka pominięta zostaje na swojej półce, a nieudana odpowiedź nie rusza drzewa. Nie może wydarzyć się w połowie.

Na wynik nałożone jest celowe ograniczenie: drzewo ma pozostać **płaskie — maksymalnie cztery poziomy i żadnego rozdziału-parasola**, który zawiera wszystko. Strażnik zdejmuje parasol, gdy model go zostawi, bo korzeń nad korzeniem dokłada kliknięcie i zero informacji.

Q. Kiedy uruchamiać przebudowę?

Po intensywnym okresie pracy, który zmienił kształt tego, co wiesz — a nie według harmonogramu.

Sygnałem jest spis treści, który przewijasz zamiast czytać: rozdziały, które miały sens przy dwudziestu notatkach, a teraz mieszczą sześćdziesiąt, albo trzy gałęzie, które ciągle Ci się mylą.

Można ją uruchamiać wielokrotnie bez ryzyka, a drzewo da się potem edytować, więc nie ma powodu traktować tego jak wielkiej decyzji.

Q. Dlaczego przebudowa ogranicza się do czterech poziomów, skoro drzewo dopuszcza sześć?

Bo głębokość to koszt płacony przy każdym czytaniu i korzyść płacona raz, przy zapisie.

Sześć poziomów istnieje po to, żeby struktura, która naprawdę ich potrzebuje, nie była zablokowana — niektóre tematy faktycznie zagnieżdżają się tak głęboko. Ale zostawiony sam sobie model chętnie wyprodukuje piękną pięciopoziomową taksonomię czterdziestu notatek, a Ty nigdy nie użyjesz poziomu czwartego i piątego inaczej niż przeklikując się przez nie.

Przebudowa optymalizuje więc coś konkretnego: drzewo, które przejrzysz na jednym ekranie i przejdziesz w dwóch kliknięciach.

Q. Jak spis treści wygląda na dysku?

Jako `base.md` w katalogu vaulta: konspekt ze względnymi odnośnikami do plików w `notes/`.

To właśnie sprawia, że wyeksportowany vault jest używalnym dokumentem, a nie workiem plików. Otwórz `base.md` w Obsidianie, VS Code albo na GitHubie i masz działający indeks — odnośniki się rozwiązują, bo są względnymi ścieżkami między prawdziwymi plikami.

Jest generowany, a nie pisany ręcznie, więc pozostaje w zgodzie z drzewem w aplikacji, zamiast być drugą rzeczą do utrzymywania.

Q. Czy przydaje się do czegoś poza przeglądaniem?

To mapa, którą dostaje model. Każde Twoje pytanie idzie razem ze spisem treści vaulta i to na tej podstawie aplikacja decyduje, które notatki warto wciągnąć w całości.

Struktura nie jest więc dekoracją dla ludzi — dobrze uformowane drzewo wyostrza wyszukiwanie kontekstu, a rozjechane je rozmywa. To jest praktyczny argument za okazjonalną przebudową.

To też kolejność, w której wszystko się eksportuje. PDF albo prezentacja idą za spisem treści, więc poprawienie drzewa poprawia dokument, który komuś wręczasz.