Aplikacja wizualizująca działanie klastra MongoDB

Autorzy:

Kacper Cieślak
Daniel Kuźmierkiewicz
Michał Petrykowski

Wstęp

We współczesnych bazach danych nadzorowanie i monitorowanie systemów rozproszonych takich jak klaster MongoDB jest niezbędne. Prezentowana aplikacja ułatwia to zadanie zapewniając:

- Wizualizację węzłów.
- Śledzenie baz danych i kolekcji.
- Aktualizacje zmian danych w czasie rzeczywistym.

Aplikacja wykorzystuje biblioteki MongoDB, komponenty .NET i Windows Forms. Węzły MongoDB wewnątrz klastra są hostowane przy pomocy Docker Compose.

Funkcje aplikacji

Inicjalizacja i wizualizacja węzła

Program dynamicznie inicjuje i wyświetla węzły klastra za pomocą przycisków ułożonych w układzie kołowym. Każdy przycisk odpowiada węzłowi i dostarcza istotnych informacji o stanie po kliknięciu na niego.

Główne funkcje:

- InitializeNodesFromDatabase(): łączy się z zestawem replik MongoDB i pobiera szczegóły węzła.
-ArrangeNodes(): dynamicznie układa węzły w okręgu na podstawie bieżącego rozmiaru okna.

Monitorowanie w czasie rzeczywistym

Program sprawdza zmiany w kolekcjach w różnych węzłach i wizualnie podświetla te zmiany, sprawdzanie stanu bazy danych odbywa się co 5 sekund:

- CheckNodeForUpdates(): Pobiera bieżący stan kolekcji i porównuje go z poprzednim stanem.
- BlinkNodeAdded() i BlinkNodeRemove(): wizualne oznaczenie dodawania i usuwania elementów w bazie.

Panel informacyjny

Aplikacja zawiera panel boczny wyświetlający metryki klastra, takie jak:

- Całkowita liczba dokumentów.
- Całkowity rozmiar pamięci masowej.
- Liczba baz danych.

Panel jest aktualizowany dynamicznie za pomocą UpdateInfoPanel().

Przegląd techniczny

Główne komponenty

Łączność MongoDB:

- Używa klasy MongoClient z biblioteki MongoDB.Driver do nawiązywania połączenia i pobierania danych.
- Dzieli plik YAML w celu utworzenia stringa konfigurującego połączenie.

Interfejs graficzny:

- Zbudowany przy użyciu formularzy Windows Forms.
- Implementuje kontrolki przycisków dla węzłów i panel do wyświetlania informacji.

Aktualizacje w czasie rzeczywistym:

- Timer wywołuje aktualizację co 5 sekund w celu sprawdzenia czy interfejs odzwierciedla najnowsze dane.

Integracja Docker Compose:

- Węzły MongoDB w klastrze są wdrażane przy użyciu Docker Compose.
- Mapowania portów są wyodrębniane z pliku Docker Compose YAML w celu dynamicznego skonstruowania ciągu połączenia.

Wyzwania i rozwiązania

Wyzwanie: Dynamiczne rozmieszczanie węzłów

Dynamiczne rozmieszczanie węzłów wymagało obliczeń w celu pozycjonowania przycisków w układzie kołowym.

Rozwiązanie: Wdrożenie wzorów trygonometrycznych w ArrangeNodes() w celu obliczenia pozycji przycisków względem środka formularza.

Wyzwanie: Responsywny interfejs użytkownika

Utrzymanie responsywności podczas zapytań do bazy danych.

Rozwiązanie: Zastosowanie asynchronicznych wzorców programowania, co zapewnia, że wątek interfejsu użytkownika nie jest dotknięty zadaniami pobierania danych.

Wyzwanie: Analiza konfiguracji Dockera

Wyodrębnienie informacji konfiguracyjnych z plików YAML.

Rozwiązanie: Wykorzystanie wyrażeń regularnych do analizy mapowań portów i dynamicznego konstruowania ciągu połączenia.

Odniesienia

- Dokumentacja MongoDB: https://www.mongodb.com/docs
- Dokumentacja Microsoft .NET Framework: https://learn.microsoft.com/en-us/dotnet
- Windows Forms Overview: https://learn.microsoft.com/en-us/dotnet/desktop/winforms
- Dokumentacja Docker Compose: https://docs.docker.com/compose