POLSKO-JAPOŃSKA AKADEMIA
TECHNIK KOMPUTEROWYCH

Wydział Informatyki

Katedra Inżynierii Oprogramowania

Inżynieria Oprogramowania i Baz Danych

Grzegorz Warchoł
Nr albumu S29233

Uniwersalny generator danych biznesowych działający w Chmurze

Praca magisterska napisana
pod kierunkiem:

Dr inż. Mariusz Trzaska

Warszawa, miesiąc, rok obrony

Streszczenie

Praca dotyczy problemu generowania randomizowanych danych tymczasowych przy projektowaniu nowych lub modyfikacji istniejących aplikacji opartych na bazach danych. Programiści tracą dużo czasu na generowanie danych tymczasowych w różnych aplikacjach, eksporcie ich do różnych formatów danych, następnie odpowiednim imporcie za pomocą innych aplikacji. Istniejące rozwiązania pozwalają na generowanie pewnych danych tymczasowych, ale nie zapewniają bezpośredniego połączenie z bazami danych. Często są to płatne rozwiązanie komercyjne.

Proponowane rozwiązanie pozwoli na dowolne tworzenie takich danych, przygotowane poprzez system importu z różnych formatów (np. json, arkusze kalkulacyjne) oraz odpowiednie ich kategoryzowanie. Generator będzie miał bogatą startową bazę gotowych danych podzielonych na wiele kategorii (np. dane osobowe, adresy, muzyka, natura i wiele innych). Poprzez implementacje połączeń z różnymi typami baz danych (np. relacyjne: *Microsoft SQL Server, MySQL / MariaDB, PostgresSql, Oracle DB,* , nierelacyjne(nosql): *MongoDB*), generator pozwoli na bezpośredni eksport danych do wybranej bazy danych, wybranych tabel, formatów plików (np. *sql, xlsx, csv, xml*). Rozwiązanie może działać jako lokalna instancja aplikacji www (intranet) lub dostępna publicznie Internecie.

Słowa kluczowe:

Dane losowe, projektowanie baz danych

Spis treści

1. WSTĘP ... 5

1.1. Motywacja ... 5
1.2. Cel pracy ... 5
1.3. Zakres pracy ... 5
1.4. Rezultat pracy ... 6

2. ANALIZA ISTNIEJĄCYCH ROZWIĄZAŃ ... 7

2.1. Macaroo – Random Data Generator and API Mocking Tool ... 7
2.2. Data Generator ... 9
2.3. Random data generator ... 10
2.4. Randat – Free random data generator ... 11
2.5. Podsumowanie ... 12

3. PROPOZYCJA ROZWIĄZANIA ... 13

3.1. Wnioski z analizy istniejących narzędzi z którymi musi się zmierzyć nowe rozwiązanie .. 13
3.2. Podstawowe wymagania dotyczące proponowanego rozwiązania ... 13
3.3. Proponowane rozwiązanie ... 14

4. WYKORZYSTANE TECHNOLOGIE ... 16

4.1. PHP ... 16
4.2. Symfony Framework ... 17
4.3. Hotwire Turbo / Stimulus ... 19
4.4. CSS / JS: Bootstrap Framework ... 23
4.5. MySQL / MariaDB ... 23
4.6. Oracle Database ... 24
4.7. Microsoft SQL Server ... 24
4.8. PostgreSQL ... 25
4.9. MongoDB ... 25

5. PROTOTYP UNIWERSALNEGO GENERATORA DANYCH BIZNESOWYCH ... 27

5.1. Architektura rozwiązania ... 27
5.2. Baza danych ... 27
5.3. Struktura katalogów ... 32
5.4. Łączenie z systemami baz danych ... 34
5.4.1. MySQL / MariaDB ... 37
5.4.2. Microsoft SQL Server ... 38
5.4.3. PostgresSql ... 39
5.4.4. Oracle Database ... 40
5.4.5. MongoDB ... 42
5.5. Generowanie danych tymczasowych ... 42
5.6. Instalacja ... 45
5.6.1. PHP ... 46
5.6.2. Środowisko JavaScript ... 46
5.6.3. Baza danych MariaDB ... 47
5.7. Interfejs użytkownika ... 48
5.7.1. Strona główna ... 48
5.7.2. Menadżer danych – Przeglądaj / edytuj ... 49
5.7.3. Menadżer danych – Stwórz dane ... 50
5.7.4. Menadżer danych – Kategorie / Tagi ... 51
5.7.5. Menadżer danych – Importuj dane ... 52

5.7.6. Połączenia ... 52
5.7.7. Eksport danych – Eksport do pliku ... 53
5.7.8. Eksport danych – Eksport do bazy danych ... 56

6. PODSUMOWANIE ... 59

6.1. Wady i zalety ... 59
6.2. Perspektywy rozwoju prototypu ... 59
6.3. Wnioski i podsumowanie ... 60

BIBLIOGRAFIA ... 61

LISTINGI ... 64

SPIS RYSUNKÓW ... 65

1. Wstęp

W dzisiejszych czasach, gdzie aplikacje bazodanowe stanowią jedną z głównych elementów działania procesów informatycznych wielu przedsiębiorstw i usług cyfrowych, istotnym aspektem projektowania i testowania tych aplikacji jest efektywne generowanie danych tymczasowych. Ta praca skupia się na problemach i wyzwaniach związanych z generowaniem takich danych, biorąc pod uwagę zarówno potrzeby programistów, jak i ograniczenia istniejących rozwiązań.

1.1. Motywacja

Podstawową motywacją stojącą za tą pracą jest rozpoznanie i rozwiązanie problemu, z którym borykają się programiści i inżynierowie danych podczas projektowania, testowania i wdrażania aplikacji bazodanowych. Obecne metody generowania danych tymczasowych często okazują się czasochłonne, niewystarczająco elastyczne oraz oddzielone od bezpośredniego środowiska pracy z bazami danych. Specyfika pracy z aplikacjami bazodanowymi wymaga od programistów nie tylko umiejętności pisania skryptów i obsługi baz danych, ale także zdolności do szybkiego tworzenia reprezentatywnych, randomizowanych danych, które mogą być użyte do testowania i weryfikacji różnych scenariuszy. Brak efektywnych narzędzi do generowania takich danych lub utrudniony do nich dostęp stwarza istotne przeszkody w procesie rozwoju oprogramowania.

Dodatkową motywacją jest chęć ułatwienia współpracy między zespołami programistów a działami analizy danych. Umożliwienie szybkiego generowania i udostępniania danych testowych może znacząco przyspieszyć procesy weryfikacji i testowania, co z kolei przekłada się na zwiększenie efektywności całego procesu projektowego.

Wreszcie, głównym celem tej pracy jest demonstracja wykorzystania nowoczesnych technologii i metod programowania w tworzeniu rozwiązań, które usprawniają proces tworzenia danych tymczasowych. Dodatkowo, omawiane rozwiązanie ma na celu zapewnienie lepszej integracji z różnymi systemami baz danych i oferowanie elastyczności w dostosowywaniu do indywidualnych potrzeb użytkownika.

1.2. Cel pracy

Głównym celem pracy jest opracowanie i przedstawienie nowego rozwiązania, które usprawni i zautomatyzuje proces tworzenia randomizowanych danych tymczasowych, co ma bezpośredni wpływ na efektywność pracy programistów oraz architektów baz danych. Praca ta proponuje kompleksowe podejście, które umożliwi tworzenie danych tymczasowych w sposób bardziej efektywny i elastyczny. Dzięki obsługi kilku popularnych systemów baz danych prezentowane rozwiązanie może być niezwykle pomocne dla specjalistów IT którzy w ramach swoich obowiązków zajmują się równoległą obsługą różnych środowisk, takich jak *Microsoft SQL Server, MySQL / MariaDB, PostgresSql, Oracle DB, MongoDB.*

1.3. Zakres pracy

Zakres pracy obejmuje wstępny opis problematyki generowania danych tymczasowych, motywację pracy oraz wskazując na jego znaczenie w kontekście współczesnego projektowania aplikacji bazodanowych. Następnie przedstawia istniejące rozwiązania z krótkim opisem oraz analizą

wad i zalet tych generatorów. W kolejnym etapie praca przedstawia ogólny zarys proponowanego rozwiązania oraz jego potencjalne korzyści dla programistów, architektów oraz organizacji. W czwartym rozdziale opisany został zbiór technologii wykorzystanych do proponowanego rozwiązania. W piątym rozdziale został objęty szczegółowy opis architektury oraz implementacji proponowanego rozwiązania. Ostatni rozdział obejmuje podsumowanie pracy wraz z wnioskami i propozycjami rozwoju zaimplementowanego rozwiązania.

1.4. Rezultat pracy

Rezultatem tej pracy magisterskiej jest analiza problematyki danych randomizowanych oraz opracowanie kompleksowego rozwiązania do generowania tych danych tymczasowych, które bezpośrednio odpowiada na zidentyfikowane potrzeby i wyzwania stojące przed programistami i inżynierami danych. Rozwiązanie to łączy w sobie zaawansowane techniki programowania z elastycznością w obsłudze różnych formatów danych i baz danych.

Kluczowym rezultatem jest stworzenie aplikacji, która umożliwia szybkie generowanie danych tymczasowych, które można bezpośrednio eksportować do różnych systemów baz danych. Aplikacja ta oferuje bogaty zestaw funkcjonalności, w tym:

- moduł tworzenia danych randomizowanych, wraz z możliwości eksportu do plików w różnych formatach (np. csv, xlsx, sql).
- moduł podstawowego eksportu współpracujący z różnymi typami baz danych (np. *MySQL, PostgreSQL, MongoDB*).
- wbudowane bogata biblioteka szablonów danych, które mogą być dostosowywane do konkretnych potrzeb użytkownika.
- intuicyjny interfejs użytkownika umożliwiający łatwą konfigurację parametrów generowania danych.

Dodatkowo, praca ta dostarcza analizę stworzonego rozwiązania, porównując je z istniejącymi narzędziami dostępnymi na rynku.

Wreszcie, praca zawiera zestaw zaleceń i kierunków dalszego rozwoju narzędzia, wskazując na potencjalne obszary ulepszeń i rozszerzeń funkcjonalności, co może przyczynić się do dalszego zwiększenia jego użyteczności w przyszłości.

2. Analiza istniejących rozwiązań

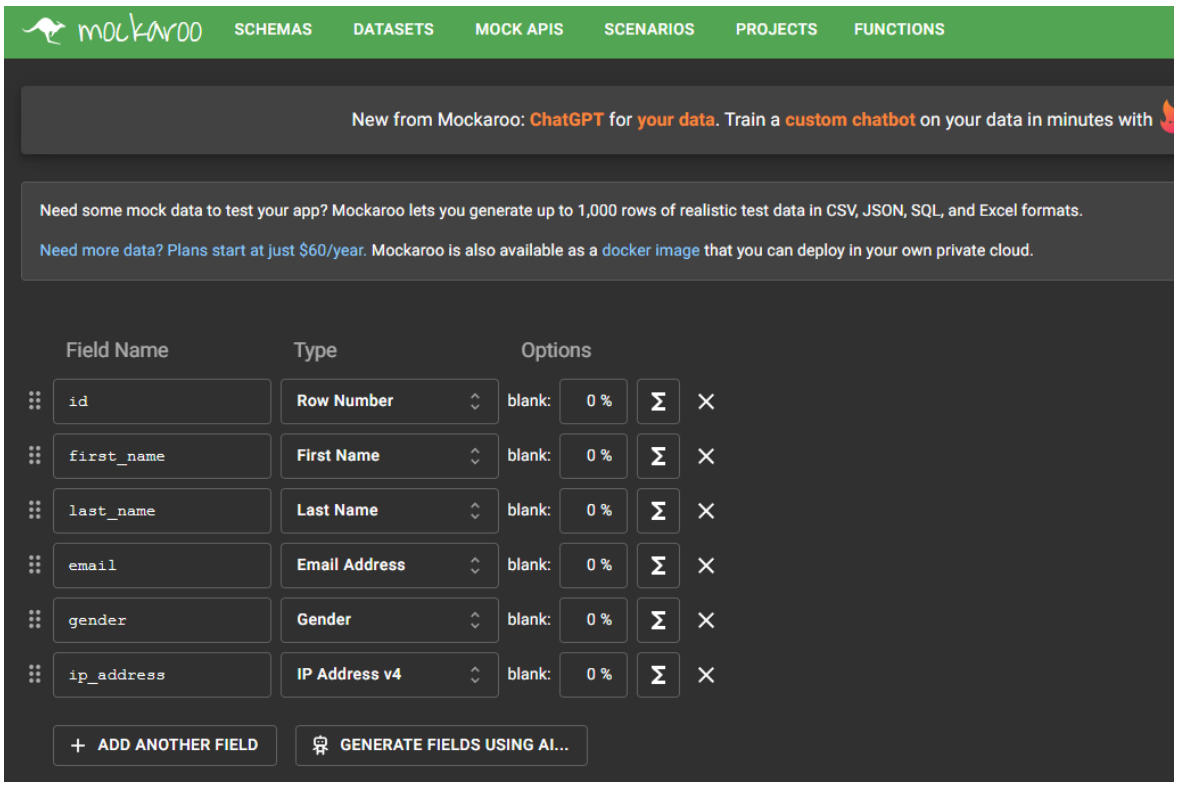
Generowanie randomizowanych danych biznesowych jest ważnym i niezbędnym elementem procesu projektowania każdej aplikacji bazodanowej. Dla przeprowadzenia testów wydajnościowych, funkcjonalnych czy podstawowej prezentacji we wczesnych fazach np. pre–alpha, programiści i architekci systemów muszą posiadać dostęp do odpowiednio przygotowanego zestawu danych testowych. Istnieje wiele rozwiązań służących do generowania tego typu danych, które przybierają różne formy, takie jak aplikacje internetowe (www) czy biblioteki programistyczne dostępne dla wielu języków programowania.

Niniejsza analiza koncentruje się na rozwiązaniach w formie aplikacji internetowych, dokładnie omawiając ich zalety i wady. W ramach tej analizy, zostaną rozpatrzone różne dostępne na rynku narzędzia, zwracając uwagę na aspekty takie jak łatwość użytkowania, funkcjonalność, wsparcie dla różnych typów baz danych, a także elastyczność w dostosowywaniu do konkretnych wymagań projektowych.

Celem tej części pracy jest nie tylko zaprezentowanie istniejących rozwiązań, ale także zidentyfikowanie ich ograniczeń, co pozwoli lepiej zrozumieć przestrzeń do innowacji i ulepszeń w obszarze generowania danych testowych. Ta analiza stanowi kluczowe tło dla dalszych rozważań i opracowania proponowanego nowego rozwiązania, które zostanie szczegółowo opisane w kolejnych rozdziałach pracy.

Wybór omawianych rozwiązań został dokonany poprzez popularność w wyszukiwarce Google [1] oraz poprzez analizę artykułów [2] oraz zapytań na popularnych forach internetowych [3].

2.1. Macaroo – Random Data Generator and API Mocking Tool



Rysunek 1 – Macaroo, źródło: [4]

Mockaroo [4] to internetowe narzędzie do generowania danych testowych. Jest to najbardziej rozbudowane z rozwiązań omawianych w tym rozdziale. Pozwala na tworzenie danych o różnych typach, takich jak tekst, liczby, daty, adresy e–mail, numery telefonów itp. Dane można generować losowo lub według określonych reguł. Rysunek 1 przedstawia stronę główną narzędzia Mockaroo.

Zalety:

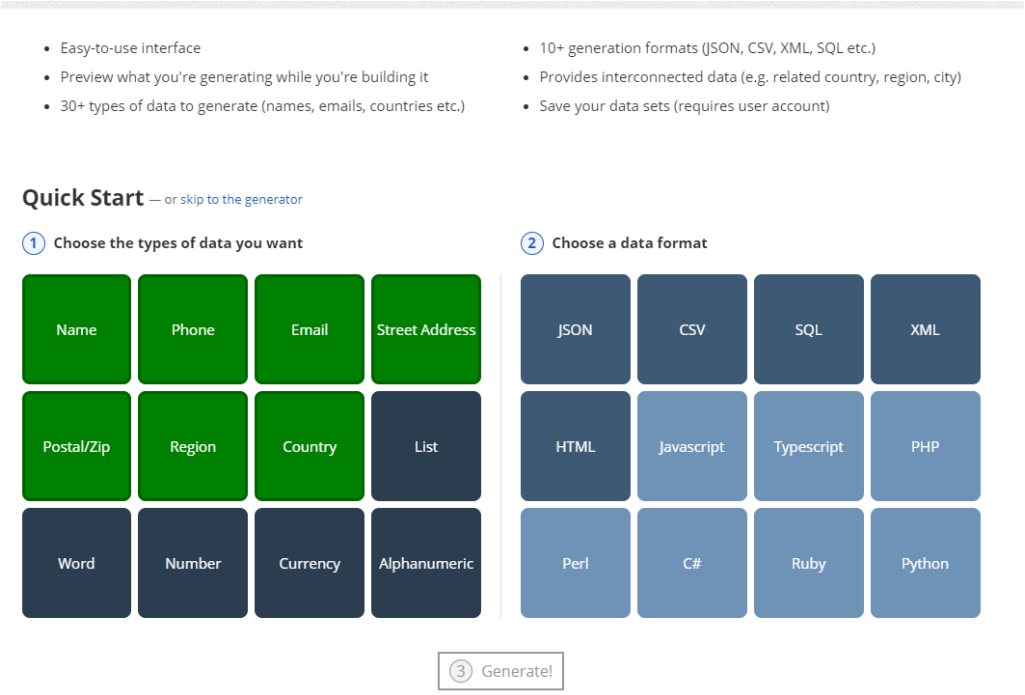
- prosta obsługa i intuicyjna obsługa GUI;
- szeroki zakres możliwości;
- możliwość generowania danych zgodnie z określonymi regułami, np. zakres wartości, które mogą być generowane lub określenie reguł pozwalających na wskazanie rodzaju tworzonych danych;
- możliwość generowania danych poprzez modele językowe;
- dostęp do API;
- możliwość eksportu danych do podstawowych formatów jak np. JSON, CSV, SQL;

Wady:

- wymaga tworzenia konta aby skorzystać z niektórych darmowych funkcji;
- darmowa wersja pozwala na generowanie tylko ograniczonej liczby danych w postaci 1000 wierszy na plik oraz maksymalnie 200 zapytań API dziennie;
- większość zaawanasowanych funkcji są dostępne tylko w wersji płatnej;
- opisana ogólnie i niejasno polityka prywatności;

Mockaroo to przydatne narzędzie do generowania danych testowych dla użytkowników mało wymagających którzy nie chcą ponosić dodatkowych kosztów na takie rozwiązania. Wersja premium rozszerza zakres możliwości, oferując dostęp do zaawansowanych funkcji, które mogą zaspokoić potrzeby bardziej wymagających użytkowników. Darmowa wersja pozwala na generowanie ograniczonej liczby danych jednakże pomimo pewnych ograniczeń może być dobrym rozwiązaniem do podstawowych zastosowań.

2.2. Data Generator



Rysunek 2 – Data generator, źródło: [5]

Generatedata.com [5] to przydatne narzędzie do generowania danych testowych. Jest prosty w obsłudze i oferuje szeroki zakres możliwości. Darmowa wersja pozwala na generowanie ograniczonej liczby danych. Rysunek 2 przedstawia stronę główną projektu Generatedata.com.

Zalety:

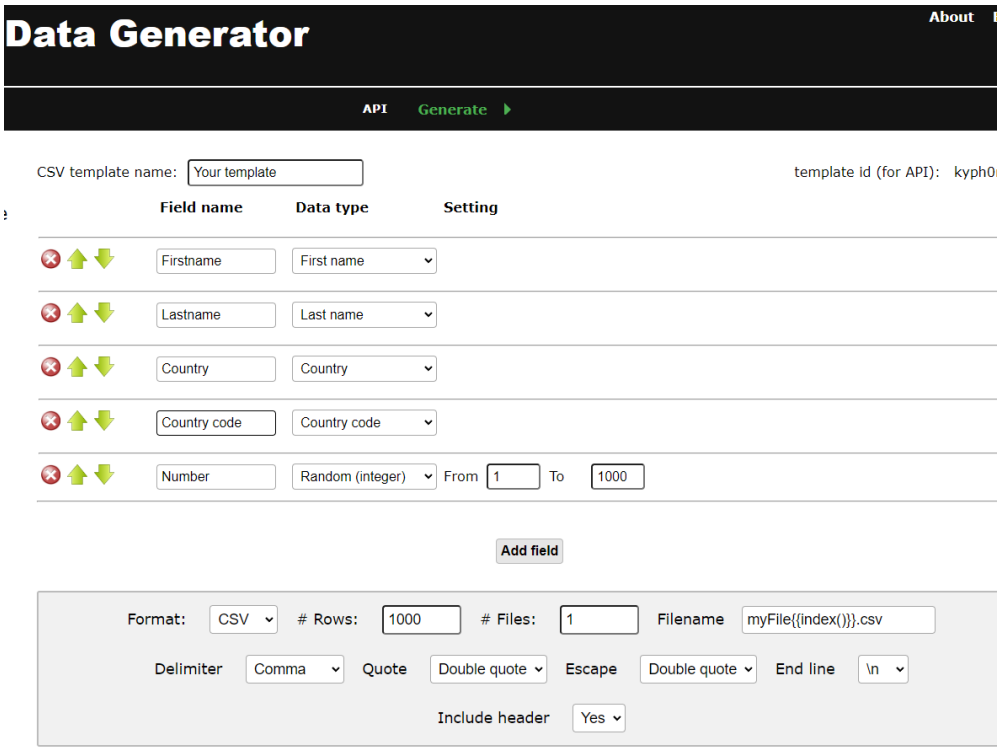
- prosty w obsłudze interfejs użytkownika;
- możliwość szybkiego podglądu generowanych danych;
- szeroki wybór typów danych do generowania;
- możliwość generowania danych w różnych formatach;
- możliwość generowania danych połączonych;
- możliwość zapisywania zestawów danych;

Wady:

- darmowa wersja pozwala na generowanie tylko ograniczonej liczby danych;
- możliwość zapisywania zestawów danych tylko po ówczesnej rejestracji;
- niektóre funkcje są dostępne tylko w wersji płatnej;
- brak zaawansowanych funkcji;

Generatedata.com nie wyróżnia się znacznie na tle innych rozwiązań. Ciekawy i intuicyjny w obsłudze interfejs użytkownika. Narzędzie może być przydatne do podstawowych zastosowań.

2.3. Random data generator



Rysunek 3 – Random data generator, źródło: [6]

Narzędzie do generowania danych testowych na stronie ExtendsClass [6] pozwala na generowanie losowych danych testowych zbliżonych do rzeczywistych danych. Składa się z trzech modułów:

- podstawowy generator danych (CSV/JSON/XML/SQL): Pozwala na generowanie danych w różnych formatach, z pojedynczym poziomem (bez podobiektów, tablic, atrybutów XML itp.);
- złożony generator danych JSON: Umożliwia generowanie danych JSON ze złożoną strukturą drzewa (z podobiektami, tablicami itp.);
- złożony generator danych XML: Umożliwia generowanie danych XML ze złożoną strukturą drzewa (z pod–tagami, atrybutami tagów itp.);

Zalety:

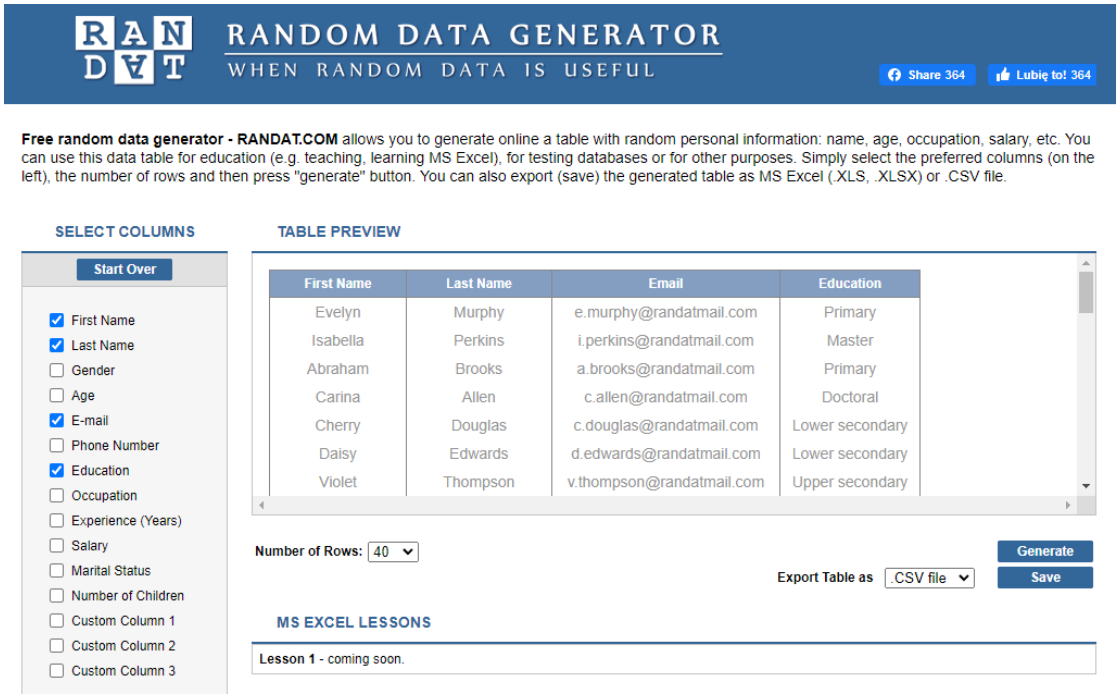
- darmowy;
- szeroki wybór typów danych do generowania;
- możliwość generowania danych w różnych formatach;
- możliwość dostosowania generowanych danych za pomocą wyrażeń regularnych;
- dostęp do api;

Wady:

- niezbyt przyjezdny interfejs;
- api pozwala tylko na 200 żądań na dzień oraz 5000 wierszy na żądanie;
- zapis szablonów tylko po rejestracji;

Opisywane narzędzie jest ciekawą propozycją dla programistów i architektów baz danych. Duża możliwość konfiguracji danych oraz dostęp do API jest na pewno dużym plusem. Toporny i nieprzyjazny interfejs oraz ograniczenia API mogą jednak być niewystarczające dla wymagających użytkowników. Rysunek 3 ukazuje ekran tworzenia danych wraz z możliwością zapisu szablonu.

2.4. Randat – Free random data generator



Rysunek 4 – Randat – Free random data generator, źródło: [7]

Randat [7] to bardzo proste internetowe narzędzie do generowania danych testowych. Pozwala na tworzenie danych o różnych typach, takich jak tekst, liczby, daty, adresy e–mail, numery telefonów itp. Dane można generować losowo lub według prostych określonych reguł. Rysunek 4 przedstawia stronę główną narzędzia.

Zalety:

- prosty w interfejs użytkownika;
- darmowy
- kilka możliwości parametryzacji generowanych danych

Wady:

- pomimo prostoty, przestarzały interfejs;
- ograniczone możliwości eksportu do trzech rodzajów plików (xls, xlsx, csv);
- prawdopodobnie narzędzie przestało być rozwijane;

Randat jest najmniej zaawansowanym narzędziem analizowanym w tym rozdziale. Pozwala na proste generowanie danych tymczasowych, jednak z tak ograniczonymi funkcjami może być niewystarczający nawet dla podstawowych zastosowań.

2.5. Podsumowanie

W poprzednich podpunktach rozdziału 2 skupiono się na analizie istniejących rozwiązaniach do generowania danych tymczasowych. Omówione zostały główne cechy, zalety i ograniczenia narzędzi takich jak Mockaroo, Generatedata.com, narzędzie ExtendsClass i Randat, z uwzględnieniem ich zdolności do generowania różnorodnych formatów danych, stopnia złożoności, możliwości integracji z bazami danych, kwestii bezpieczeństwa i zaufania, dostępności oraz realistyczności generowanych danych.

Poniżej przedstawiono główne cechy analizowanych rozwiązań:

- większość analizowanych narzędzi oferuje podstawowy zakres formatów danych, takich jak CSV, JSON, SQL i XML, jednakże . To podkreśla potrzebę elastyczności w generowaniu danych, aby odpowiadać na różnorodne wymagania projektowe.
- narzędzia różnią się stopniem złożoności, od prostych generatorów, takich jak Generatedata.com, do bardziej złożonych, jak narzędzia ExtendsClass. Istnieje zatem luka w rozwiązaniach, które mogą jednocześnie oferować prostotę obsługi przy zachowaniu możliwości generowania złożonych struktur danych.
- brak bezpośredniej integracji z systemami baz danych jest widoczny u wszystkich narzędzi. Zintegrowane rozwiązanie, które bezpośrednio łączy się z różnymi bazami danych, mogłoby znacznie zwiększyć efektywność pracy programistów.
- ważne jest, aby nowe rozwiązanie było transparentne i bezpieczne, z jasno określoną polityką prywatności i zabezpieczeniami danych.
- dostępność darmowych narzędzi kontra płatnych subskrypcji, jak w przypadku Mockaroo, wskazuje na potrzebę bardziej dostępnych rozwiązań, zwłaszcza dla mniejszych firm i niezależnych deweloperów, aby zniwelować jak najbardziej koszty.
- oferowanie realistycznych danych testowych jest kluczowe w symulowaniu rzeczywistych scenariuszy użytkowania. Istnieje potrzeba ulepszenia algorytmów generowania danych, aby lepiej odwzorowywały rzeczywiste warunki oraz możliwość bezpiecznego składowania danych które zostały utworzone na potrzeby konkretnych projektów.

Podkreślona zostaje potrzeba rozwiązania, które zintegruje te cechy w sposób bardziej holistyczny, adresując luki i wyzwania zidentyfikowane w analizowanych narzędziach.

3. Propozycja rozwiązania

Analiza dostępnych na rynku rozwiązań ujawniła znaczące braki w ich funkcjonalności, co zainspirowało do stworzenia bardziej kompleksowego narzędzia. To narzędzie, generator danych biznesowych działający w chmurze, zostało zaprojektowane tak, aby odpowiadać na kluczowe potrzeby szerokiego grona użytkowników i minimalizować istniejące niedogodności.

W rozdziale tym zostanie opisane podstawowe wnioski z analizy narzędzi istniejących na rynku, wymagania narzucone proponowane rozwiązaniu oraz opis propozycji rozwiązania.

3.1. Wnioski z analizy istniejących narzędzi z którymi musi się zmierzyć nowe rozwiązanie

Po analizie narzędzi dostępnych na rynku stwierdzono, że proponowane narzędzie powinno skupić się na działaniu w następujących obszarach:

- wszechstronność eksportu formatów danych: narzędzie powinno oferować elastyczność w generowaniu danych w różnych formatach, takich jak CSV, JSON, SQL i XML, aby sprostać wymaganiom różnorodnych projektów.
- złożoność vs prostota: rozwiązanie musi znaleźć równowagę między oferowaniem zaawansowanych funkcji a łatwością użytkowania, tak aby korzystanie z narzędzia było jak najbardziej wydajne.
- integracja z systemami baz danych: ważne jest, aby narzędzie mogło bezpośrednio współpracować z różnymi systemami baz danych, ułatwiając eksport danych.
- bezpieczeństwo: musi zapewnić wysoki poziom bezpieczeństwa danych oraz być transparentne w kwestiach ich użytkowania.
- dostępność i koszty: rozwiązanie powinno być dostępne przy poniesieniu.
- realistyczność danych: generator danych powinien oferować możliwość tworzenia realistycznych, wiarygodnych danych testowych, które odzwierciedlają rzeczywiste scenariusze użytkowania.

3.2. Podstawowe wymagania dotyczące proponowanego rozwiązania

W odpowiedzi na wnioski wynikające z analizy narzędzi rynkowych opisanych w punkcie 2, niezbędne jest zdefiniowanie podstawowych wymagań dla proponowanego narzędzia. Te wymagania powinny obejmować szeroki zakres aspektów, począwszy od kwestii infrastrukturalnych, przez funkcjonalności techniczne, aż po cele biznesowe. Wymagania te będą fundamentem dla stworzenia narzędzia, które nie tylko wypełni luki istniejące w obecnych rozwiązaniach, ale także zaoferuje innowacyjne podejście odpowiadające potrzeby użytkowników środowiskach i działach programistycznych pracujących nad wszelkim wielkościami skomplikowanie aplikacji bazodanowych.

Podstawowe wymagania:

- wysoki poziom skalowalności i dostępność: dzięki oparciu na infrastrukturze chmurowej, generator będzie mógł obsługiwać tworzyć duże wolumeny danych oraz obsługiwać liczne żądania, co jest kluczowe dla dużych organizacji i złożonych projektów. Równocześnie będzie mógł działać na mniej wydajnych systemach w zależności od potrzeb użytkownika.

- elastyczność formatów danych: umożliwienie generowania danych w różnych formatach, w tym CSV, JSON, SQL i XML, z możliwością łatwej konfiguracji struktury danych;
- zaawansowane funkcje generowania danych: wprowadzenie funkcji umożliwiających generowanie bardziej złożonych struktur danych, poprzez przygotowane szablony generatorów, wyrażania regularne itp.;
- integracja z bazami danych: bezpośrednie połączenie z różnymi systemami baz danych, umożliwiające łatwy eksport wygenerowanych danych;
- bezpieczeństwo i prywatność: zastosowanie odpowiednich standardów bezpieczeństwa, aby zapewnić ochronę generowanych danych;
- łatwość użytkowania: intuicyjny interfejs użytkownika, który pozwala użytkownikom na szybkie tworzenie i modyfikowanie schematów danych skupiający się na oszczędzaniu czasu przy tych procesach. Dwujęzyczna lokalizacja narzędzia w zwiększa dostępność aplikacji, ale także ułatwienie jej użytkowania przez osoby mówiące w różnych językach, co może być szczególnie istotne w środowisku międzynarodowym.

3.3. Proponowane rozwiązanie

Proponowane rozwiązanie zostało zaprojektowane, aby spełniać wymagania określone w punkcie 3.2. Zostało ono opracowane z myślą o wypełnieniu luk i eliminacji wad, które zostały zidentyfikowane podczas analizy istniejących generatorów dostępnych na rynku.

Narzędzie to będzie aplikacją internetową, dostępną za pośrednictwem przeglądarek internetowych. Charakteryzować się będzie przede wszystkim intuicyjnym, responsywnym interfejsem użytkownika, który umożliwi łatwe i szybkie generowanie danych tymczasowych, a także ich eksport i import. Dzięki temu narzędzie to będzie nie tylko funkcjonalne, ale również wygodne i efektywne w użytkowaniu.

Kluczowymi elementami aplikacji będą następujące moduły:

- moduł eksportu:
  - eksport danych do plików: pozwoli użytkownikom eksportowanie wygenerowanych danych tymczasowych do różnych formatów plików, takich jak CSV, JSON, XML, itp.. Ułatwi integrację z innymi systemami i narzędziami, zapewniając elastyczność w wyborze formatu danych. Implementacja nastąpi poprzez zastosowanie standardowych bibliotek do obsługi plików, z możliwością dostosowania struktury eksportowanych danych.
  - eksport danych bezpośrednio do bazy danych: pozwali na bezpośrednie wprowadzanie wygenerowanych danych do wybranej bazy danych. Wyeliminuje potrzebę pośrednich kroków w procesie migracji danych, co zaoszczędzi czas programistom i zwiększa efektywność pracy. Integracja z różnymi systemami baz danych, zostanie przeprowadzone poprzez użycie odpowiednich sterowników baz danych.
- moduł zarządzania danymi:
  - przeglądania biblioteki danych: umożliwia przeglądanie i wybieranie z różnych kategorii danych, dostarczonych wraz z aplikacją oraz tych utworzonych przez użytkownika.
  - tworzenie nowych danych / edycja i parametryzowanie utworzonych danych: umożliwia tworzenie nowych zbiorów danych oraz ich edycję, kategoryzowanie oraz oznaczenia według potrzeb użytkownika.

    - import danych: mechanizmy importu danych z takich formatów jak CSV, XLSX.
- moduł zarządzania połączeniami
    - tworzenie i edycja połączń z bazami danych: umożliwi konfigurację i zarządzanie połączeniami z różnymi typami baz danych poprzez interfejs użytkownika do konfiguracji połączeń. Zapewni wsparcie podstawowych protokołów i parametrów połączeń.

Te moduły zapewnią kompleksowe wsparcie dla procesu generowania, zarządzania i eksportowania danych tymczasowych. Implementacja intuicyjnego i responsywnego interfejsu użytkownika w każdym z tych modułów zwiększy użyteczność aplikacji, czyniąc ją narzędziem przyjaznym dla użytkowników o różnym stopniu zaawansowania.

4. Wykorzystane technologie

Rozdział ten koncentruje się na technologiach użytych w realizacji rozwiązania omawianego w tej pracy magisterskiej. Wykorzystanie odpowiednich narzędzi technologicznych jest kluczowe dla efektywnego tworzenia i zarządzania tymczasowymi danymi. Niniejszy rozdział dokładnie omawia wybrane technologie, wyjaśniając ich rolę i znaczenie w kontekście całego projektu począwszy od podstaw działania i prezentacji po założenia funkcjonalne prototypu dotyczące eksportu danych tymczasowych.

4.1. PHP

PHP [8] nazwa która jest akronimem od *"PHP: Hypertext Preprocessor",* to popularny język skryptowy ogólnego przeznaczenia, który szczególnie nadaje się do tworzenia stron internetowych. Szybki, elastyczny i pragmatyczny, PHP obsługuje wszystko, od prostych stron np. w postaci bloga po najpopularniejsze strony internetowe na świecie. PHP może współpracować z różnymi systemami baz danych, najczęściej z MySQL / MariaDB, ale również z innymi takimi jak PostgreSQL, Oracle, czy Microsoft SQL Server.

PHP może działać na praktycznie każdym popularnym na serwerze WWW, takim jak Apache lub Nginx. Skrypty PHP są przetwarzane na serwerze, a wynik jest wysyłany do klienta w formie zwykłego HTML lub innego formatu (np. JSON, XML) w zależności od rodzaju żądania. PHP może być osadzony bezpośrednio w kodzie HTML, co pozwala na dynamiczne generowanie treści stron. Jest częścią popularnych stosów technologicznych, takich jak: LAMP (Linux, Apache, MySQL, PHP), WAMP (Windows, Apache, MySQL, PHP) oraz MAMP (Mac, Apache, MySQL, PHP). Świadczy to o jego niezależności względem używanych systemów.

Jako język skryptowy, PHP jest interpretowane, nie kompilowane, co ułatwia szybkie tworzenie i testowanie kodu bez konieczności. Charakteryzuje się prostą składnią, podobną do języka C, co sprawia, że jest łatwy do nauki, szczególnie dla początkujących. PHP ma wbudowane wiele funkcji, min. do interakcji z bazami danych, co jest kluczowe dla wielu aplikacji internetowych. Jako oprogramowanie open–source, PHP jest dostępne bezpłatnie i ma jedną z największych społeczności wspierającą jego rozwój.

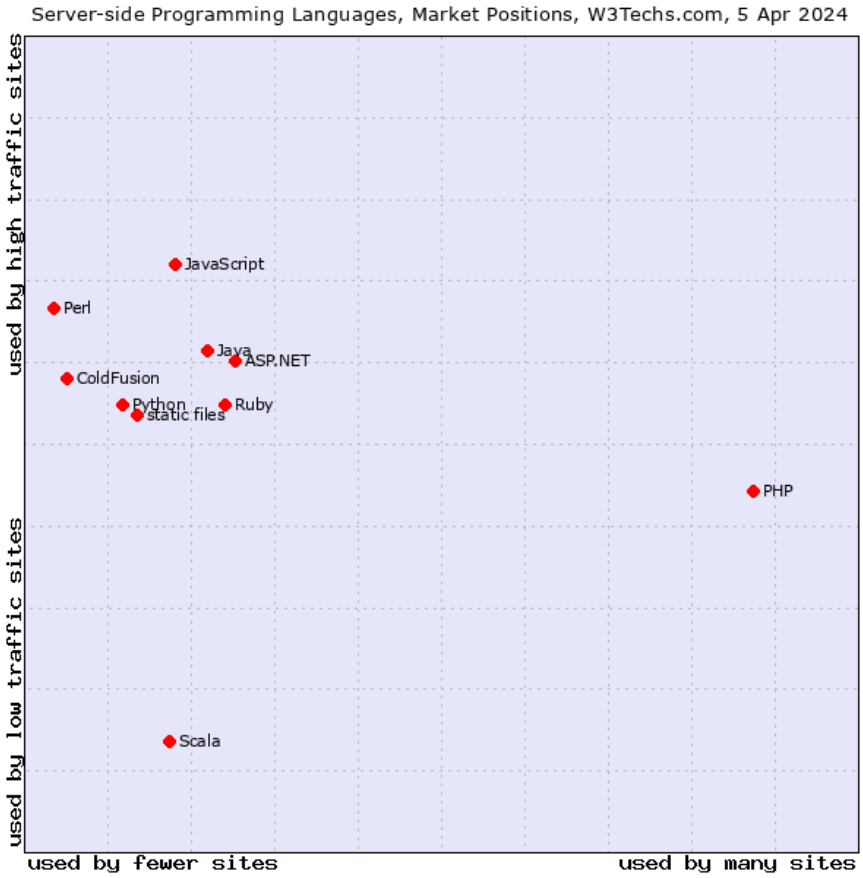
PHP jest szeroko stosowany do tworzenia dynamicznych stron internetowych i systemów zarządzania treścią (CMS), takich jak WordPress, Wikipedia, Yahoo [9]. Używany jest także do tworzenia skomplikowanych aplikacji internetowych, w tym e–commerce, portali społecznościowych, i aplikacji biznesowych. PHP może być również używane do tworzenia API i usług webowych, obsługujących różne formaty danych, takie jak JSON i XML.

Bardzo duża ilość narzędzi, bibliotek oraz frameworków związanych z PHP, powoduje iż można użyć tego języka do każdego rodzaju projektów, począwszy od małych stron, po aplikacje wykorzystujące komponenty typu microservice oraz duże modułowe aplikacje dla każdego rodzaju biznesu. Jednym z najbardziej popularnych i znanych jest Symfony Framweork, znany z wszechstronności, elastyczności oraz praktycznie nieograniczonych możliwości, zostanie omówiony w kolejnym podpunkcie. Na osobne wzmiankowanie zasługuje narzędzie Composer – menedżer zależności dla PHP, który potrafi zintegrować naszą aplikację z setkami tysięcy narzędzi dostępnych w repozytoriach.

Przez swoją prostotę oraz popularność która sięga ponad 76% [10] w internacie co jest druzgocącą przewagą względem innych języków, PHP często był krytykowany w kontekście bezpieczeństwa. Przy takiej popularności trudno z minimalizować wystąpienia błędów programistycznych, jednak ewolucja tego języka spowodowała iż pod tym względem pisania bezpiecznego kodu nie ustępuje choćby językowi JAVA [11]. Głównym priorytetem w pianiu

bezpiecznego kodu PHP dla aplikacji WWW jest zapobieganie wstrzykiwaniu SQL poprzez używanie przygotowanych zapytań, poprawna obsługa błędów i logowanie, aby uniknąć ujawnienia wrażliwych informacji, oraz implementacja środków ochrony przed atakami XSS i CSRF, takich jak walidacja danych wejściowych i tokeny sesji.

Rysunek 5 przedstawia raport pozycji rynkowej dla języków programowania wykonywalnych po stronie serwera na dzień 05 kwietnia 2024 roku.



Rysunek 5 – Raport pozycji rynkowej dla języków programowania wykonywalnych po stronie serwera, źródło: [12]

PHP wyróżnia się pod względem szybkości w porównaniu do innych języków szczególności w środowiskach w które występują mała ilość konkurencyjnych zapytań [13]. Od wersji PHP 8 wprowadzono technikę JIT *(just–in–time compilation)* pozwalającą na kompilowanie programu do kodu maszynowego bezpośrednio przed jego wykonaniem, usprawniając przy tym szybkość wykonywania skryptów języka.

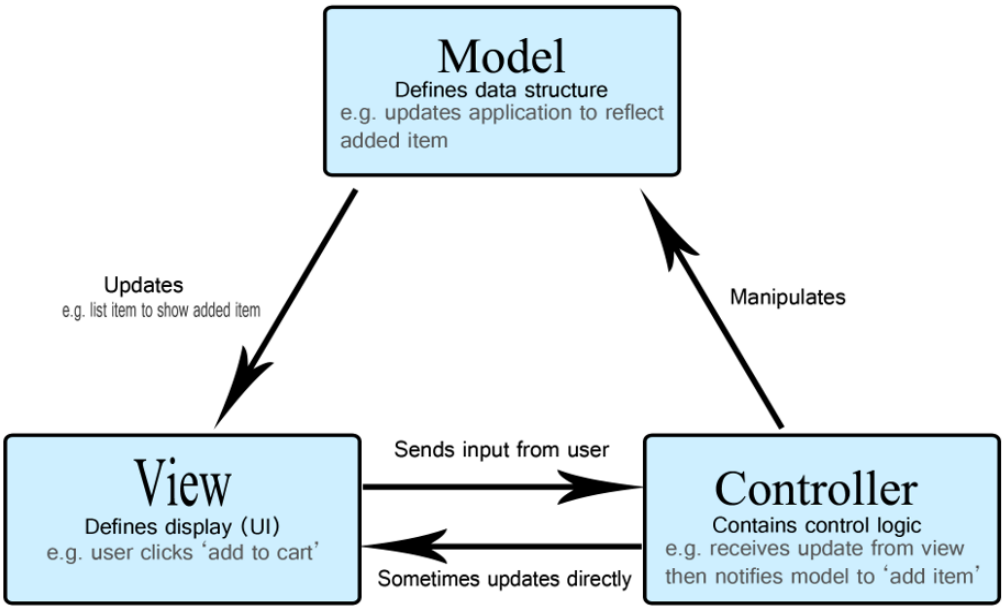
Podsumowując, PHP jest dojrzałym i wszechstronnym językiem, który odgrywa kluczową rolę w rozwoju aplikacji internetowych. Jego łatwość użycia i wsparcie dla różnorodnych baz danych czynią go ważnym narzędziem dla programistów i inżynierów oprogramowania. PHP zostanie użyty jako język typu *server–side* w opisanym w tej pracy generatorze danych biznesowych opartego na chmurze.

4.2. Symfony Framework

W kontekście opracowywanego narzędzia, wybrano język skryptowy PHP opisany w punkcie 4.1 i zdecydowano się na zastosowanie frameworka Symfony, napisanego w tym języku. Symfony,

będąc zaawansowanym i popularnym narzędziem tego typu, wyróżnia się swoją modularnością, wydajnością oraz elastycznością. Jest to szczególnie korzystny wybór dla tworzenia złożonych aplikacji webowych, gdyż oferuje on solidne fundamenty i bogaty zestaw komponentów oraz narzędzi. Ten framework umożliwia budowę dobrze zorganizowanych, wydajnych aplikacji, przystosowanych do dynamicznie zmieniających się wymagań nowoczesnego środowiska webowego.

Framework ten charakteryzuje się architekturą opartą na wzorcu Model–View–Controller (MVC) [14], co ułatwia separację logiki aplikacji od interfejsu użytkownika, zapewniając tym samym lepszą organizację kodu i łatwiejszą współpracę w zespołach programistycznych. Jednym z kluczowych aspektów Symfony jest jego modułowa budowa, która umożliwia łatwe dodawanie i usuwanie funkcjonalności, dzięki czemu można elastycznie dostosowywać aplikację do zmieniających się wymagań biznesowych. Na Rysunek 6 pokazany został schemat Modal View Controller (MVC).



Rysunek 6 – Schemat Modal View Controller (MVC) – źródło: [15]

Symfony odznacza się również zaawansowanym systemem tzw. bundli [16], który ułatwia zarządzanie kodem i jego modularność. To sprawia, że rozwój i utrzymanie dużych aplikacji staje się znacznie prostsze i bardziej efektywne. W kontekście wydajności, Symfony jest zoptymalizowany do obsługi dużych obciążeń, co jest kluczowe w dużych aplikacjach i systemach. Framework ten oferuje także rozbudowane opcje używania cache, które znacząco poprawiają czas odpowiedzi aplikacji, co jest niezwykle ważne w dzisiejszym szybkim świecie cyfrowym.

Aspekt bezpieczeństwa jest bardzo ważnym elementem frameworka Symfony. Zawiera on liczne wbudowane mechanizmy zabezpieczające, które pomagają chronić aplikację przed różnymi zagrożeniami, takimi jak ataki SQL Injection, Cross–Site Scripting (XSS) czy Cross–Site Request Forgery (CSRF). Dzięki temu programiści mogą skupić się na budowaniu funkcjonalności, mając pewność, że podstawowe kwestie bezpieczeństwa są już adekwatnie adresowane.

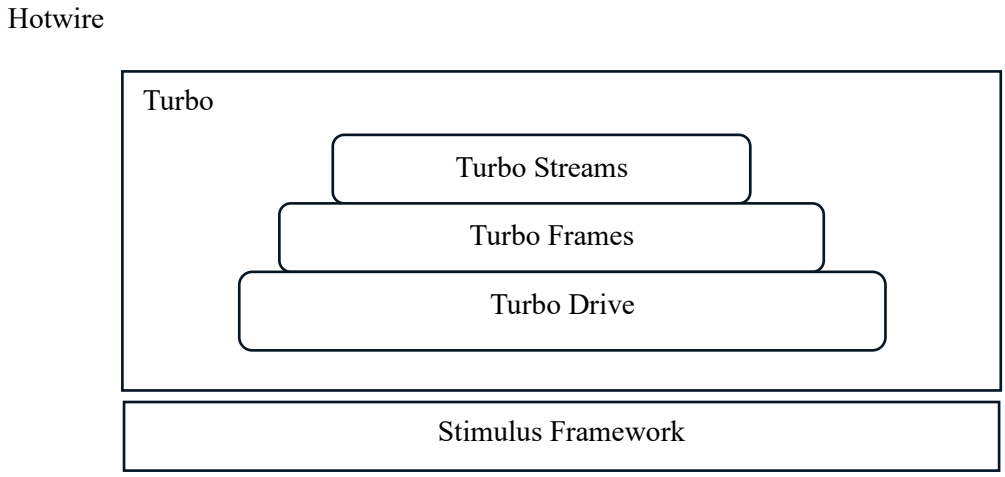
Kolejnym ważną częścią opisywanego frameowrka jest system szablonów Twig, który jest domyślnym tego typu komponentem Symfony. System szablonów to silnik generujący widoki w różnych narzędziach typu framework. Jego główne cechy to czysta i zrozumiała składnia, która ułatwia tworzenie czytelnych i łatwych do utrzymania szablonów. Jest bezpieczny, automatycznie zarządzając ucieczką danych *(escaping)*, co pomaga zapobiegać atakom typu XSS. Ponadto, Twig charakteryzuje się dużą elastycznością, pozwalając na definiowanie własnych funkcji, filtrów i rozszerzeń, oraz oferuje

wysoką wydajność dzięki kompilacji szablonów do natywnego kodu PHP oraz przechowywania w pamięci cache wygenerowanych widoków.

Podsumowując, wykorzystanie Symfony Framework do stworzenie proponowanego narzędzia zapewnia mocne i elastyczne środowisko do budowy zaawansowanej aplikacji www. Modułowość tego framowerka, wydajność i skoncentrowanie na bezpieczeństwie czynią go idealnym wyborem dla tego projektu, który wymaga nie tylko odpowiedniej funkcjonalności, ale także skalowalności, wydajności i bezpieczeństwa.

4.3. Hotwire Turbo / Stimulus

Kolejnym ważnym elementem technologii wykorzystanych w proponowanym rozwiązaniu jest Hotwire Turbo i Stimulus. Te nowoczesne narzędzia skupione na obszarze front–end aplikacji www opracowane przez Basecamp. Firma ta znana jest min. z prowadzenia na rynek bardzo popularny i chwalony, otwrtoźródłowy framework Ruby on Rails. Turbo oraz Stimulus [17] reprezentują innowacyjne podejście do budowania dynamicznych interfejsów użytkownika bez konieczności polegania na ciężkich frameworkach JavaScript. Rysunek 7 przedstawia w prosty sposób układ stosu Turbo oraz Stimulus w pakiecie Hotwire.



Rysunek 7 – Stos bibliotek Hotwire, źródło: Opracowanie własne

Turbo to biblioteka JavaScript, która umożliwia ładowanie stron internetowych bez konieczności przeładowywania całej strony. Dodatkowo Trubo może być bezpośrednio zintegrowane z Symfony Framework poprzez bundle Symfony UX Turbo. Zawiera takie moduły jak:

- Turbo Drive – pozwala na zastąpienie zmienionego fragmentu HTML bez konieczności pełnego przeładowania strony. Biblioteka automatycznie przechwytuje operacje kliknięcia na elementach typu *<a href>* wysyłając żądania HTTP poprzez Fetch API [18] następnie aktualizuje tylko zmieniony kod HTML. Potrafi modyfikować adres URL, co zapewnia normalne zachowanie przeglądarki, takie jak żądania operacji historii, przejść w przód i w tył.
  W kontekście formularzy, Turbo Drive również wykorzystuje mechanizmy, który umożliwiają wysyłanie danych formularza na serwer bez pełnego przeładowania strony, przyspieszając tym samym proces przesyłania danych.

- Turbo Frames – umożliwia podzielenie strony internetowej na niezależne obszary, które mogą być ładowane w sposób leniwy *(lazy loading)*. Dzięki temu, tylko te fragmenty strony, które są aktualnie potrzebne, są pobierane z serwera. To podejście znacznie poprawia wydajność aplikacji internetowej, redukując czas ładowania i zasoby sieciowe potrzebne do wyświetlenia strony.
- Turbo Streams – oferuje mechanizm dostarczania zmian na stronie internetowej przy użyciu tylko HTML i zestawu akcji podobnych do CRUD *(Create, Read, Update, Delete)*. Eliminuje to konieczność pisania skryptów JavaScript do aktualizowania stanu aplikacji, co upraszcza rozwój i utrzymanie dynamicznych elementów strony.

Na Listing 1 pokazano przykład instalacji za pomocą menadżera pakietów *npm*. Listing 2 prezentuje bezpośrednie osadzenia w kodzie HTML, oraz użycia Turbo Drive. Dwie strony *index.html* z kodem HTML widocznym na Listing 3 oraz *data.html* oraz na Listing 4 różnią się tylko zawartością elementu * *. Podczas operacji kliknięcia na link kierujący do podstrony *data.html* żądanie jest obsługiwane poprzez typ *fetch*. Następnie zostaje zmieniony tylko kod HTML różniący się od strony z której zostało wywołane żądanie, bez przeładowania strony. Adres URL zostaje zmieniony na docelowy. Działania te zaprezentowane zostały na Rysunek 8 oraz Rysunek 9.

Listing 1 – Instalacja za pomocą menadżera pakietów npm, źródło: Opracowanie własne

```
npm install @hotwired/turbo
```

Listing 2 – Instalacja za pomocą bezpośredniego osadzenie w kodzie HTML, źródło: Opracowanie własne

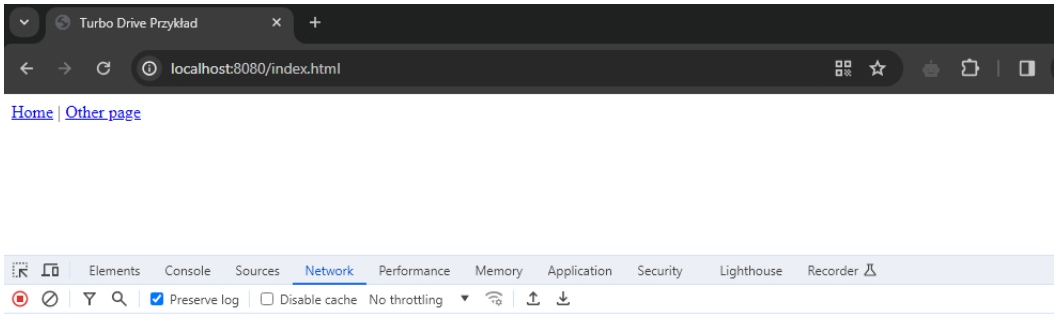
```
<script src=https://cdn.skypack.dev/@hotwired/turbo type="module"></script>
```

Listing 3 – Przykładowe użycie Turbo Drive – index.html, źródło: Opracowanie własne

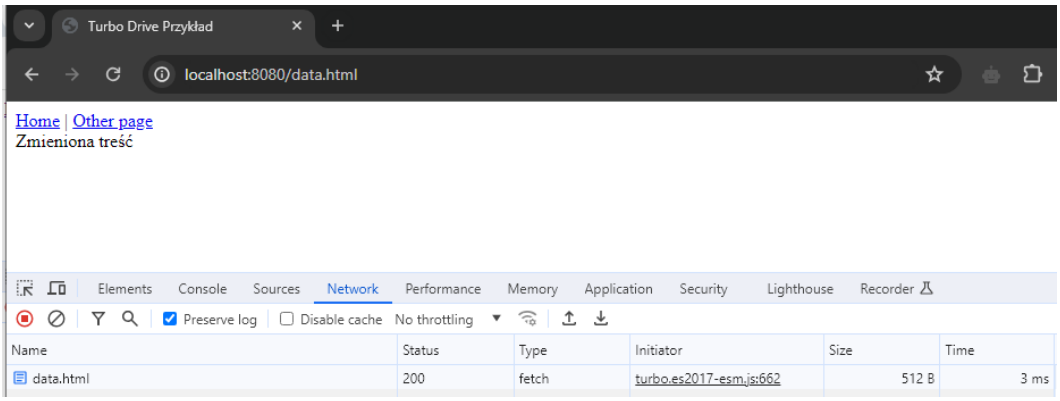
```
<!DOCTYPE html>
<html>
<head>
    <title>Turbo Drive Przykład</title>
    <script src=https://cdn.skypack.dev/@hotwired/turbo type="module"></script>
</head>
<body>
<nav>
    <a href="/index.html">Home</a> |
    <a href="/data.html">Other page</a>
</nav>

<div id="content">
    <!-- Tutaj Turbo Drive załaduje treść -->
</div>
</body>
</html>
```

Listing 4 - Przykładowe użycie Turbo Drive – data.html, źródło: Opracowanie własne

```
<!DOCTYPE html>
<html>
<head>
    <title>Turbo Drive Przykład</title>
    <script src="https://cdn.skypack.dev/@hotwired/turbo" type="module"></script>
</head>
<body>
<nav>
    <a href="/index.html">Home</a> |
    <a href="/data.html">Other page</a>
</nav>

<div id="content">
    Zmieniona treść
</div>
</body>
</html>
```



Rysunek 8 – Przykładowe użycie Turbo Drive – index.html, źródło: Opracowanie własne



Rysunek 9 – Przykładowe użycie Turbo Drive – data.html, źródło: Opracowanie własne

Stimulus to framework JavaScript, który umożliwia tworzenie aplikacji internetowych o wydajności tzw. *Single Page Applications* bez konieczności pisania dodatkowych skryptów JavaScript. Działa to poprzez skojarzenie komponentów HTML z obiektami JavaScript, zwanymi kontrolerami. Kontrolery mogą być używane do obsługi zdarzeń, takich jak kliknięcie przycisku lub zmiana wartości pola tekstowego. Dodatkowo Stimulus może być bezpośrednio zintegrowany z Symfony Framework przez bundle Stimulus & Symfony UX.

Podstawowe cechy Stimulus framework:

- łatwy w użyciu, wystarczy dodać atrybut data–controller do elementu HTML, aby przypisać go do kontrolera javascript.
- efektywny: nie wymaga pisania skryptów JavaScript do obsługi zdarzeń.
- rozszerzalny: można go dostosować do własnych potrzeb, tworząc własne kontrolery.

Na Listing 5 przedstawiono przykład instalacji Stimulus framework za pomocą menadżera pakietów npm oraz przykład działania na prostym przykładzie. Po instalacji został stworzony skrypt widoczny na Listing 6 z kontrolerem *hello_controller.js* który będzie przetwarzał żądania funkcyjne. W pliku *application.js* którego kod widoczny jest na Listing 7 został zainicjalizowany Stimulus oraz uprzednio stworzony kontroler. W pliku *index.html* widocznym na Listing 8 element HTML z atrybutem *data–controller="hello"* został przetworzony przez kontroler i zawartość tego elementu została zmieniona.

Listing 5 – Instalacja za pomocą menadżera pakietów npm, źródło: Opracowanie własne

```
npm install @hotwired/stimulus
```

Listing 6 – Kontroler hello_controller.js, źródło: Opracowanie własne

```
import { Controller } from "@hotwired/stimulus";

export default class extends Controller {
    connect() {
        this.element.textContent = "Hello, Stimulus!";
    }
}
```

Listing 7 – Inicjalizacja Stimulus oraz kontrolera – application.js, źródło: Opracowanie własne

```
import { Application } from "@hotwired/stimulus";
import HelloController from "./hello_controller";

const application = Application.start();
application.register("hello", HelloController);
```

Listing 8 – Zmiana treści elementu z atrybutem data–controller – index.html, źródło: Opracowanie własne

```
<!DOCTYPE html>
<html>
<head>
    <title>Stimulus Przykład</title>
    <script src="application.js" type="module"></script>
</head>
<body>
<div data-controller="hello">
    <!-- Tekst zostanie zastąpiony przez Stimulus, connect() -->
</div>
</body>
</html>
```

Podsumowując, wykorzystanie Turbo i Stimulus w generatorze danych biznesowych będzie stanowiło znaczące ulepszenie w sposobie tworzenia interfejsu użytkownika, zapewniając jednocześnie prostotę i efektywność w rozwoju oprogramowania. Bezpośrednia integracja z Symfony pozwoli na szybszy i prostszy rozwój narzędzia przy zachowaniu wysokich standardów pisania kodu.

4.4. CSS / JS: Bootstrap Framework

W rozwiązaniu proponowanym w niniejszej pracy, zdecydowano się na wybranie tzw. frameworka front–endowego, który zapewni gotowe biblioteki CSS oraz JavaScript do wygodnej budowy interfejsu użytkownika. Wybór padł na Bootstrap Framework [19]. Jest to jeden z najpopularniejszych frameworków front–endowych, z łatwą składnią, możliwością rozbudowy oraz liczną społecznością użytkowników. Oferowany jest na licencji MIT, tak więc może być używany w dowolnym projekcie.

Charakteryzuje się systemem siatki (grid system), który umożliwia łatwe tworzenie layoutów dostosowujących się do różnych rozmiarów ekranów, pozwoli generatorowi być używanym na ekranach o dowolnej wielkości.

Bootstrap wyróżnia się również bogatym zestawem gotowych do użycia komponentów, takich jak przyciski, formularze, karty, czy nawigacje, co znacząco przyspiesza proces tworzenia interfejsu użytkownika. Możliwość korzystania z gotowych lub dostosowania stylów i wyboru konkretnych komponentów pozwala na stworzenie unikalnego wyglądu aplikacji, zachowując przy tym spójność oraz estetyczność tworzonego interfejsu.

Dodatkowo, Bootstrap oferuje wiele pluginów JavaScript, które wprowadzają interaktywne elementy, takie jak modal, nawigacje poprzez rozwijane menu, systemy zakładek , bez konieczności pisania skomplikowanego kodu CSS oraz JavaScript. To sprawia, że interfejs użytkownika staje się bardziej dynamiczny i angażujący, poprawiając ogólne doświadczenie użytkownika.

Podsumowując, Bootstrap Framework pozwoli znacząco uatrakcyjnić wygląd interfejsu w opisywanym generatorze. Jego elastyczność, łatwość użycia oraz bogaty zestaw narzędzi czynią go idealnym wyborem dla szybkiego i efektywnego tworzenia responsywnych oraz estetycznie atrakcyjnych interfejsów użytkownika.

4.5. MySQL / MariaDB

MySQL to popularny relacyjny system zarządzania bazami danych, który opiera się na języku SQL, właścicielem tego rozwiązania jest firma Oracle. Jest to otwarte oprogramowanie, szeroko stosowane w różnorodnych aplikacjach internetowych i korporacyjnych ze względu na jego niezawodność, wydajność i elastyczność. System ten charakteryzuje się prostotą w użyciu, co czyni go atrakcyjnym rozwiązaniem dla wielu programistów i administratorów baz danych [20]. Opisywany prototyp pozwala na eksport danych tymczasowych do tego oprogramowania, a sposób łączenia i pobrania schematu bazy został przedstawiony w podpunkcie 5.4.1.

Baza MariaDB, odgrywa podstawową rolę w kontekście przechowywania uprzednio stworzonych danych tymczasowych, tworzenia nowych danych oraz kompleksowego zarządzania nimi. Jest to otwartoźródłowy system zarządzania relacyjnymi bazami danych, który powstał jako fork jednej z najpopularniejszych relacyjnych systemów baz danych MySQL. Został stworzony przez oryginalnych twórców MySQL, co zapewnia wysoką kompatybilność i ciągłość rozwoju [21]. Jego cechą charakterystyczną jest wysoka wydajność, niezawodność i łatwość w skalowaniu, co czyni go idealnym wyborem dla aplikacji wymagających efektywnego przetwarzania dużych wolumenów danych.

MariaDB oferuje zaawansowane funkcje, takie jak replikacja, partycjonowanie, i wsparcie dla różnorodnych silników przechowywania, w tym InnoDB i MyRocks. Jest również zgodna z SQL, co ułatwia integrację z istniejącymi aplikacjami i narzędziami. Dodatkowo, MariaDB zapewnia silne wsparcie dla procedur przechowywanych, wyzwalaczy (*trigger*), widoków (*view*), a także zapewnia zaawansowane mechanizmy bezpieczeństwa, w tym szyfrowanie danych w locie i w spoczynku.

Jednym z kluczowych aspektów MariaDB jest jej społeczność i model rozwoju oparty na otwartym źródle. Dzięki temu, użytkownicy i programiści mają możliwość aktywnego uczestnictwa w rozwoju i udoskonalaniu bazy danych, co przekłada się na jej ciągłą adaptację do nowych wyzwań i potrzeb technologicznych.

Baza znajduje szerokie zastosowanie zarówno w małych i średnich przedsiębiorstwach, jak i w dużych korporacjach, gdzie jest wykorzystywana do zarządzania ważnymi dla działalności bazami danych. Jej elastyczność i wydajność sprawiają, że jest ona dobrze przyjęta w środowiskach wymagających niezawodności, wysokiej dostępności oraz łatwości zarządzania dużymi zbiorami danych.

Wybór opisywanego powyżej systemu bazy danych pozwoli na wydajne składowanie stworzony danych tymczasowych oraz potencjalny łatwy rozwój aplikacji. Skalowalności, prostota obsługi, duże możliwości oraz format projektu open-source dodatkowo wskazują na poprawność wyboru tego rozwiązania.

4.6. Oracle Database

Oracle Database to potężny system zarządzania relacyjnymi bazami danych (RDBMS) opracowany przez firmę Oracle Corporation. Implementuje model relacyjny, w którym dane są przechowywane w powiązanych tabelach, zgodnie ściśle ze standardami SQL. Oferuje bogaty zestaw wbudowanych typów danych, w tym numerycznych, znakowych, daty/czasu, binarnych, obiektowych oraz specjalistycznych typów takich jak przestrzenne, multimedialne czy dokumentowe.

Jedną z kluczowych cech Oracle Database jest jego architektura wielowarstwowa [22], która umożliwia separację warstwy aplikacyjnej, serwerowej oraz magazynowej. Taka architektura zapewnia wysoką skalowalność, wydajność oraz elastyczność w zarządzaniu bazą danych. Ponadto, Oracle Database posiada architekturę wielomodelową, co pozwala na przechowywanie i zarządzanie różnorodnymi typami danych w obrębie jednej bazy danych w tym relacyjnych, obiektowych, dokumentowych czy nawet wielowymiarowych struktur. Dzięki temu istniej możliwość budowania złożonych aplikacje które korzystają z różnorodnych struktur danych bez potrzeby integracji z zewnętrznymi systemami bazodanowymi.

Generator pozwala na pełny eksport wygenerowanych danych tymczasowych do Oracle DB, proces pobrania schematu bazy danych został opisany w podpunkcie 5.4.4.

4.7. Microsoft SQL Server

Microsoft SQL Server to kompleksowy system zarządzania bazami danych opracowany przez firmę Microsoft. Jest to oprogramowanie przeznaczone do zarządzania oraz przetwarzania dużych ilości danych w strukturze relacyjnej. SQL Server oferuje szeroki zakres funkcjonalności, w tym zaawansowane możliwości w zakresie analizy danych, przetwarzania transakcji, inteligencji biznesowej (BI) oraz zarządzania danymi i aplikacjami. Charakteryzuje się wysoką wydajnością, skalowalnością, niezawodnością oraz zabezpieczeniami, co sprawia, że jest popularnym wyborem wśród przedsiębiorstw na całym świecie.

Oprogramowanie to wspiera różne języki zapytań i skryptów, w tym przede wszystkim T–SQL (Transact–SQL), który jest rozbudowaną wersją języka SQL specyficzną dla SQL Server. T-SQL umożliwia pisanie zaawansowanych zapytań, tworzenie procedur składowanych, funkcji definiowanych przez użytkownika, wyzwalaczy, a także zarządzanie transakcjami i analizę danych przy użyciu funkcji analitycznych [23].

Oprócz tradycyjnej relacyjnej bazy danych, SQL Server oferuje również możliwość przechowywania danych niestrukturalnych, takich jak pliki, dokumenty XML czy obiekty binarnych dużych obiektów (BLOB). Pozwala to na tworzenie aplikacji, które wymagają integracji różnych typów danych w ramach jednego zintegrowanego rozwiązania.

Kolejną istotną cechą SQL Server jest ścisła integracja z innymi produktami Microsoft, takimi jak .NET Framework, Visual Studio, SharePoint i usługami chmury Microsoft Azure. Ta integracja ułatwia tworzenie kompleksowych aplikacji biznesowych przy wykorzystaniu znanych programistom narzędzi i technologii.

Opisywany prototyp pozwala na pełny eksport wygenerowanych danych tymczasowych do MS Sql Server. Pobranie schematu bazy danych do którego ma zostać wykonany eksport został opisany w podpunkcie 5.4.2.

4.8. PostgreSQL

PostgreSQL to owartoźródłowy, zaawansowany system zarządzania relacyjnymi bazami danych (RDBMS), który cieszy się dużą popularnością na świecie od wielu lat. W porównaniu do innych otwartych systemów baz danych jedną z kluczowych cech PostgreSQL jest jego rozszerzalność. Użytkownicy mogą dodawać nowe typy danych, funkcje, operatory czy nawet języki skryptowe, co pozwala na dostosowanie bazy do specyficznych potrzeb aplikacji. Ponadto, system ten zawiera większość zaawansowanych funkcje, wsparcie dla kluczy obcych, widoków, wyzwalaczy oraz procedur składowanych [24].

Szczególnym wyróżnianie zasługuje zastosowaniu zaawansowanych technik MVCC (*Multi–Version Concurrency Control*), które stosowane są do obsługi transakcji, zapewniają wysoką przepustowość i szybkość działania przy jednoczesnym zachowaniu spójności danych. System ten posiada również silne wsparcie dla pracy z dużymi zbiorami danych, co obejmuje zaawansowane indeksowanie, partycjonowanie tabel oraz narzędzia do replikacji danych.

PostgreSQL oferuje również bogate wsparcie dla pracy z dużymi zbiorami danych. Posiada zaawansowane techniki indeksowania, w tym indeksy GiST, GIN i SP-GiST, umożliwiające efektywne wyszukiwanie danych. Ponadto, obsługuje partycjonowanie tabel, które pozwala na podzielenie dużych zestawów danych na mniejsze, łatwiejsze do zarządzania części. W celu zapewnienia wysokiej dostępności danych, PostgreSQL obsługuje replikację danych, zarówno w trybie asynchronicznym, jak i synchronicznym.

Dla systemu PostgreSQL wdrożono w generatorze wsparcie eksportu danych tymczasowych. Sposób pobrania schematu do tego oprogramowania został opisany w podpunkcie 5.4.3.

4.9. MongoDB

MongoDB jest popularnym, otwartym systemem bazy danych NoSQL, który wykorzystuje model danych oparty na dokumentach. W przeciwieństwie do relacyjnych baz danych, MongoDB nie przechowuje danych w tabelach ze sztywno zdefiniowaną strukturą, ale w zbiorach (kolekcjach) dokumentów w formacie BSON (binarny odpowiednik JSON).

Elastyczność schematu dokumentów jest jedną z ważnych zalet MongoDB, ponieważ pozwala na łatwe dostosowanie i ewolucję struktury danych bez potrzeby przeprowadzania kosztownych operacji modyfikacji schematu, co jest często wymagane w tradycyjnych relacyjnych bazach danych. Dokumenty w ramach tej samej kolekcji mogą mieć różne struktury i zestawy pól. W zakresie skalowalności, MongoDB oferuje wbudowane mechanizmy replikacji i partycjonowania danych (*sharding*). Replikacja zapewnia redundancję i wysoką dostępność, podczas gdy *sharding* [25] pozwala na rozłożenie danych na wiele maszyn, zapewniając horyzontalną skalowalność i możliwość obsługi ogromnych ilości danych oraz bardzo wysokiego obciążenia.

Opisywany prototyp wspiera eksport danych randomizowanych do MongoDB, a sposób pobrania schematu dla danej kolekcji został opisany 5.4.5.

5. Prototyp uniwersalnego generatora danych biznesowych

Rozdział ten rozpoczyna się od przedstawienia podstawowej architektury proponowanego rozwiązania oraz opisu struktury przechowywania danych wewnątrz generatora. Następnie opisane zostały system baz danych, z którymi generator współpracuje oraz metody łączenia i pobierania schematów strukturalnych baz danych do których ma nastąpić eksport. Kolejnym etapem jest opis instalacji generatora. W dalszej części rozdziału został przedstawiony interfejs użytkownika, różne metody oraz scenariusze generowania danych tymczasowych, zarówno podstawowe, jak i bardziej zaawansowane. Na koniec zaprezentowane zostały przykłady importowania danych do generatora oraz ich eksportu do plików oraz poszczególnych systemów baz danych.

5.1. Architektura rozwiązania

Architektura uniwersalnego generatora danych biznesowych została zaprojektowana z naciskiem na elastyczność i wydajność. Centralnym elementem tej architektury jest aplikacja www stworzona z wykorzystaniem zaawansowanego frameworka Symfony, który jest oparty na języku skryptowym PHP. Symfony, znane ze swojej modularności i rozszerzalności, zapewnia solidną podstawę do efektywnego tworzenia i zarządzania złożonymi procesami generowania danych.

Interfejs użytkownika generatora jest kreowany przy użyciu systemu szablonów Twig, który oferuje intuicyjny i elastyczny sposób na projektowanie responsywnych i estetycznie przyjemnych interfejsów. Dzięki swojej zdolności do oddzielenia logiki aplikacji od prezentacji interfejsu użytkownika, Twig umożliwia szybkie i efektywne wprowadzanie zmian w wyglądzie aplikacji bez ingerencji w jej główną funkcjonalność.

Pod względem przechowywania danych, generator korzysta z bazy danych MariaDB, która jest wydajnym i niezawodnym rozwiązaniem typu open–source.

W ogólnym ujęciu, ta zintegrowana architektura zapewnia solidną, elastyczną i skalowalną platformę do generowania różnorodnych danych biznesowych, odpowiadając tym samym na szeroki zakres potrzeb użytkowników końcowych i scenariuszy biznesowych.

5.2. Baza danych

W prezentowanym rozwiązaniu przechowywanie danych jest ważnym elementem działania całej aplikacji. Struktura bazy została zaprojektowany w taki sposób, aby zapewnić jak największą wydajność w przetwarzaniu danych oraz zapewnić prostotę w ewentualnej rozbudowie systemu. Strukturę bazę danych możemy podzielić na dwie części. Pierwszą która przechowuje informacje o połączeniach z systemami baz danych (*MySQL / MariaDB, Microsoft SQL Server, Postgres SQL, Oracle, Mongo DB*). Druga natomiast przechowuje wygenerowane dane tymczasowe stałe, które są dostarczone wraz z pierwszą instalacją aplikacji oraz dane stworzone i zachowane przez użytkownika.

Poniżej przedstawiono listę tabel bazy danych oraz wykaz zawierający szczegółowe informacje o wszystkich kolumnach wraz z opisem ich zastosowania. Każda z tabel z bazy danych ma odwzorowanie w pliku encji względem użycia *ORM Doctrine.* Pliki encji znajdują się w katalogu *Entity* opisanym w podpunkcie 5.3. Na Listing 9 oraz Listing 10 zostały pokazane fragmenty listing kodu z plików encji tych obiektów. Następnie na Rysunek 10 przedstawiona została graficzna reprezentacja schematu bazy danych, na którym oparty jest generator danych tymczasowych.

Lista tabel w analizowanym rozwiązaniu obejmuje:

- *connection* – przechowywanie połączeń do baz danych, wykaz pól wyszczególnia Tabela 1;
- *connection_type* – rodzaj bazy danych, strukturę przedstawia Tabela 2;
- *data_category* – kategorie danych tymczasowych przechowywane w bazie danych, lista pól ukazuje Tabela 6;
- *data_type* – rodzaj danych, strukturę przedstawia Tabela 4;

- Tabela 4*data_name* – dane tymczasowe, kolumny wyszczególnia Tabela 7;
- *data_tag* – lista tagów do oznaczeń danych tymczasowych, lista kolumn pokazuje Tabela 3;
- *data_tag_data* – łącznik danych tymczasowych z tagami, struktura kolumn przedstawiona w Tabela 5;
- *column_type_memory* – przechowywanie przypisani kategorii danych do kolumn baz danych w celach eksportu, lista pól pokazana w Tabela 8.

Tabela 1 – Lista kolumn tabeli connection, źródło: Opracowanie własne

Pole w bazie danych	**Rodzaj / typ**	**Opis**
id	Integer	Klucz główny tabeli
type_id	Integer	Klucz obcy do tabeli *connection_type*
description	Varchar	Opis / nazwa połączenia
port	Integer	Numer portu na którym ma być zestawione połączenie
host	Varchar	Host na którym znajduje się silnik baz danych
username	Varchar	Nazwa użytkownika łączącego się bazą danych
password	Varchar	Hasło użytkownika łączącego się z bazą danych
dsn	Varchar	*Data Source Name*, nazwa źródła danych, jeżeli puste używa do połączenia innych wartości innych pól
db_name	Varchar	Nazwa bazy danych do której ma nastąpić połączenie
collaction	Varchar	Nazwa zbioru, używane w zależności od typu bazy danych

Tabela 2 – Lista kolumn tabeli connection_type, źródło: Opracowanie własne

Pole w bazie danych	**Rodzaj / typ**	**Opis**
id	Integer	Klucz główny tabeli
name	Varchar	Nazwa połączenia rodzaju bazy danych
kind	Varchar	Rodzaj bazy danych, np. *relational* lub *document*

Tabela 3 – Lista kolumn tabeli data_tag, źródło: Opracowanie własne

Pole w bazie danych	**Rodzaj / typ**	**Opis**
id	Integer	Klucz główny tabeli
name	Varchar	Nazwa tagu

Tabela 4 – Lista kolumn tabeli data_type, źródło: Opracowanie własne

Pole w bazie danych	**Rodzaj / typ**	**Opis**
id	Integer	Klucz główny tabeli
name	Varchar	Nazwa typu danych

Tabela 5 – Lista kolumn tabeli data_tag_data, źródło: Opracowanie własne

Pole w bazie danych	**Rodzaj / typ**	**Opis**
id	Integer	Klucz główny tabeli
data_name_id	Integer	Klucz obcy do tabeli *data_name*
data_tag_id	Integer	Klucz obcy do tabeli *data_tag*

Tabela 6 – Lista kolumn tabeli data_category, źródło: Opracowanie własne

Pole w bazie danych	**Rodzaj / typ**	**Opis**
id	Integer	Klucz główny tabeli
name	Varchar	Nazwa kategorii
parent_id	Integer	Klucz obcy do tabeli *data_category*
type_id	Integer	Klucz obcy do tabeli *data_type*
is_default	Boolean	Oznaczenie czy kategoria jest rodzajem domyślnym (brak możliwości edycji) czy stworzonym przez użytkownika
is_automatic	Boolean	Oznaczenie czy kategoria zawiera dane generowane i przechowywane z bazy danych czy dane generują odpowiednie metody w aplikacji

Tabela 7 – Lista kolumn tabeli data_name, źródło: Opracowanie własne

Pole w bazie danych	**Rodzaj / typ**	**Opis**
id	Integer	Klucz główny tabeli
name	Varchar	Nazwa / wartość danych randomizowanych
category_id	Integer	Klucz obcy do tabeli *data_category*
is_constant	Boolean	Oznaczenie czy dane jest rodzajem stałym (brak możliwości edycji) czy stworzonym przez użytkownika

Tabela 8 – Lista kolumn tabeli column_name_category, źródło: Opracowanie własne

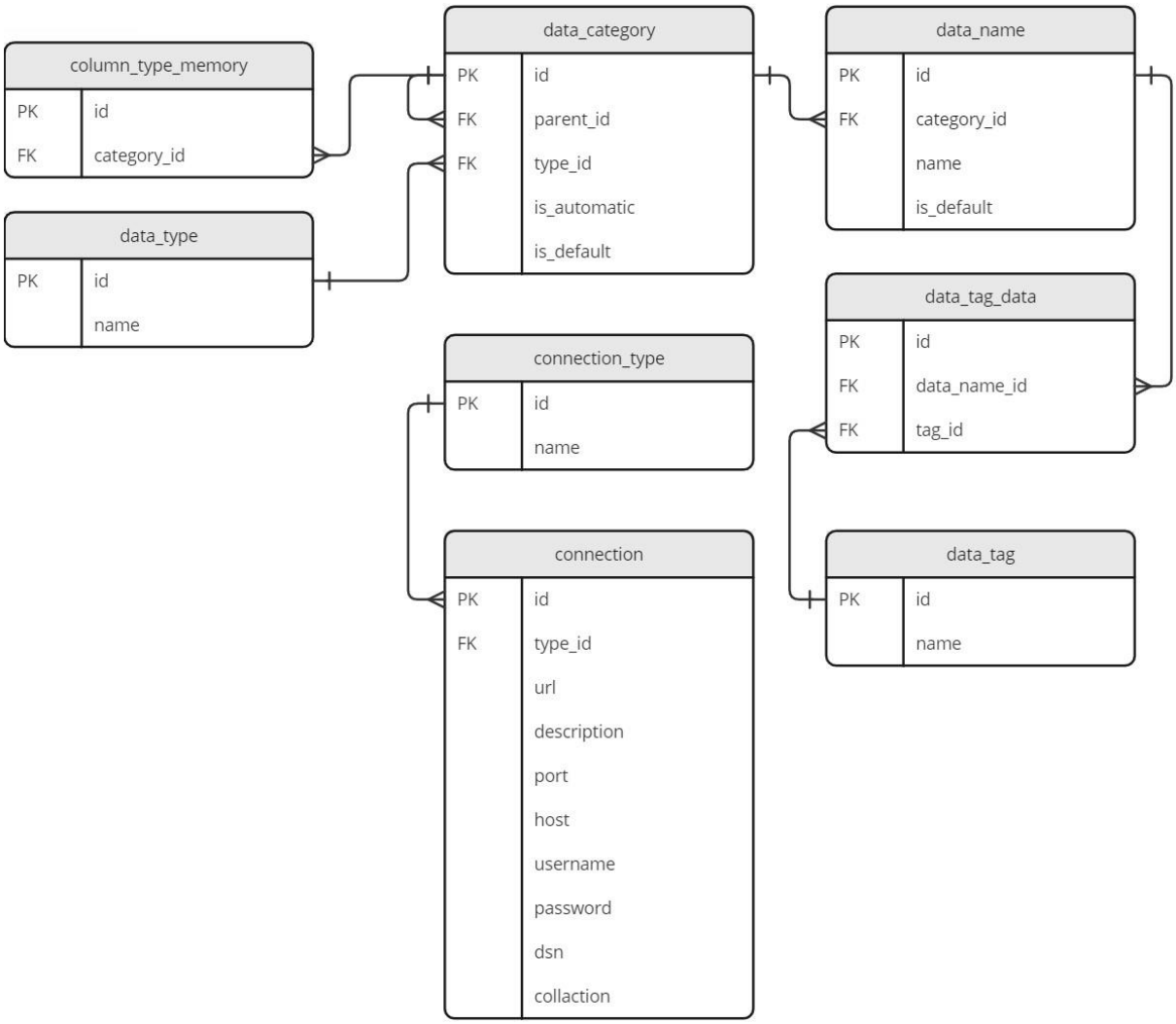
Pole w bazie danych	**Rodzaj / typ**	**Opis**
id	Integer	Klucz główny tabeli
category_id	Integer	Klucz obcy do tabeli *data_category*
column_id	Varchar	Nazwa kolumny / pola z tabeli bazy danych do której ma docelowy eksport

Listing 9 – Encja tabeli data_name, źródło: Opracowanie własne

```php
<?php

namespace App\Entity;

use App\Repository\DataNameRepository;
use Doctrine\Common\Collections\ArrayCollection;
use Doctrine\Common\Collections\Collection;
use Doctrine\ORM\Mapping as ORM;
/**
 * DataName Entity
 *
 * Encja dla tabeli data_name
 */
#[ORM\Entity(repositoryClass: DataNameRepository::class)]
//#[Broadcast]
class DataName
{
    /**
     * @var int|null Unikalny identifikator
     */
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;

    /**
     * @var string|null Nazwa danych tymczasowych
     */
    #[ORM\Column(length: 255)]
    private ?string $name = null;

    /**
     * @var bool|null Czy dane stałe czy stworzne przez użytkownika
     */
    #[ORM\Column(nullable: false, options: ['default' => 0])]
    private ?bool $isConstant = null;

    /**
     * @var DataCategory|null Katagoria danych
     */
    #[ORM\ManyToOne(inversedBy: 'dataNames')]
    #[ORM\JoinColumn(nullable: false)]
    private ?DataCategory $category = null;

    /**
     * @var Collection<int, DataTagData> Kolekcja tagów dla danych tymczasowych
     */
    #[ORM\OneToMany(mappedBy: 'dataName', targetEntity: DataTagData::class)]
    private Collection $dataTagData;

    /**
     * @var Collection<int, DataTagData> Kolekcja danych do tagów
     */
    #[ORM\OneToMany(mappedBy: 'dataName', targetEntity: DataTagData::class)]
    private Collection $q;
```

Listing 10 – Encja tabeli data_category, źródło: Opracowanie własne

```php
<?php

namespace App\Entity;

use App\Repository\DataCategoryRepository;
use Doctrine\Common\Collections\ArrayCollection;
use Doctrine\Common\Collections\Collection;
use Doctrine\ORM\Mapping as ORM;

/**
 *  DataCategory Entity
 *
 *  Encja dla tabeli data_category
 */
#[ORM\Entity(repositoryClass: DataCategoryRepository::class)]
class DataCategory
{
    /**
     * @var int|null Unikalny identifikator
     */
    #[ORM\Id]
    #[ORM\GeneratedValue]
    #[ORM\Column]
    private ?int $id = null;
    /**
     * @var string|null Nazwa kategorii
     */
    #[ORM\Column(length: 255)]
    private ?string $name = null;

    /**
     * @var self|null Rodzic kategorii
     */
    #[ORM\ManyToOne(targetEntity: self::class, inversedBy: 'dataCategories')]
    private ?self $parent = null;

    /**
     * @var Collection<int, self> Kolekcja kategorii dziedzicznych
     */
    #[ORM\OneToMany(mappedBy: 'parent', targetEntity: self::class)]
    private Collection $dataCategories;

    /**
     * @var Collection<int, DataName> Kolecja Danych tymczasowych przypisanych do kategorii
     */
    #[ORM\OneToMany(mappedBy: 'category', targetEntity: DataName::class)]
    private Collection $dataNames;
    /**
     * @var DataType|null Rodzaj kategorii
     */
    #[ORM\ManyToOne(inversedBy: 'dataType')]
    #[ORM\JoinColumn(nullable: true)]
    private ?DataType $type = null;
    /**
     * @var bool|null Czy kategoria domyślna, czy stworzona przez użytkownika
     */
    #[ORM\Column(options: ['default' => 0])]
    private ?bool $isDefault = null;
    /**
     * @var @var bool|null Czy kategoria danych automatycznych
     */
    #[ORM\Column]
    private ?bool $isAutomatic = null;
```



Rysunek 10 – Schemat bazy danych uniwersalnego generatora danych biznesowych działającego w Chmurze, źródło: Opracowanie własne

5.3. Struktura katalogów

Struktura katalogów kluczowym elementem zarządzania i organizacji projektu. W ramach realizowanego rozwiązania na przykładzie zastosowanego frameworka Symfony opisanego w rozdziale 4.2. struktura katalogów została zaprojektowana tak, aby ułatwić deweloperom prace, zapewniając jednocześnie elastyczność i skalowalność aplikacji.

Poniżej został przedstawiony szczegółowy opis głównych katalogów w strukturze generatora:

- assets/: katalog zawiera zasoby jak pliki JavaScript, arkusze stylów CSS (oraz ich źródła SCSS przed kompilacją) oraz wszelkie obrazy. Pliki umieszczone w tym katalogu są przetwarzane przez narzędzie Webpack, aby zoptymalizować je do użycia w środowisku produkcyjnym, co obejmuje kompilację SCSS do CSS oraz plików JavaScript

- bin/: zawiera wykonywalne pliki, takie jak konsolę Symfony, która jest narzędziem wiersza poleceń służącym do interakcji z aplikacją, np. do uruchamiania zadań w ramach frameworka, takich jak czyszczenie cache czy aktualizacja schematu bazy danych.
- config/: przechowuje wszystkie pliki konfiguracyjne aplikacji. Struktura konfiguracji w Symfony jest bardzo elastyczna i pozwala na definiowanie konfiguracji w różnych formatach, takich jak YAML, XML lub PHP. W tym katalogu znajdują się także definicje routingu, pakietów, usług oraz środowisk (np. dev, test, prod).
- node_modules/: katalog bibliotek node
- public/: jest to katalog publiczny aplikacji, który zawiera punkt wejścia do aplikacji (index.php) oraz zasoby takie jak obrazy, skompilowane lub sparsowane pliki skryptowe JavaScript, arkuszy stylów, obrazki, które są bezpośrednio dostępne dla użytkownika końcowego.
- src/: jeden z najważniejszych katalogów, tzw. serce aplikacji znajduje w którym się większość kodu źródłowego generatora. Podstawowymi katalogami zawartymi w nim są:
    - Controller/: zawiera klasy, tzw. kontrolery odpowiadające za przetwarzanie żądań HTTP, wykonywanie logiki biznesowej i zwracanie odpowiedzi, często w formie wyrenderowanego widoku html lub danych np. formacie json.
    - DQL/: katalog do przetrzymywania rozszerzeń języka DQL. Jest on obiektowym językiem zapytań używanym przez ORM Doctrine do interakcji z bazą danych w sposób abstrakcyjny, niezależnie od konkretnego silnika bazy danych.
    - Entity/: zawiera klasy encji, które są mapowaniem obiektów PHP na rekordy bazy danych poprzez użycie ORM Doctrine. Encje reprezentują dane, są powiązane z tabelami bazy danych użytych w generatorze.
    - EventListener/: przetrzymuje klasy które są wykorzystywane do obsługi zdarzeń w aplikacji odpowiedzi na wywołanie zdarzenia, takiego jak żądanie HTTP, bez konieczności ingerencji w główny przepływ aplikacji oraz w konkretnych kontrolerach.
    - Form/: zawiera pliki klas formularzy, które są wykorzystywane do budowania i obsługi formularzy w rozwiązaniu poprzez odpowiednie funkcje frameworka Symfony.
    - Model/: klasy modeli, które nie są bezpośrednio mapowane na bazy danych przez używany ORM. Pomagają na interakcję z różnymi systemami baz danych
    - Repository/: zawiera repozytoria dla encji, które są klasami w Doctrine ORM zawierające zapytania do bazy danych aplikacji jak i tych do których ma nastąpić eksport danych tymczasowych. Zwracają encje, kolekcje encji, dane w postaci tabelarycznej lub wykonują operacje bezpośrednio na danym silniku baz danych.
    - Response/: klasy odpowiedzi, które rozszerzające domyślne odpowiedzi HTTP w zastosowanym frameworku
    - Twig/: przechowuje pliki klas które tworzą dodatkowe rozszerzenia funkcjonalności szablonów Twig, takie jak metody czy filtry danych
- templates/: w katalogu znajdują się wszystkie szablony Twig, które są używane do generowania widoków w rozwiązaniu. Zorganizowane są w podkatalogach odpowiadając organizacji routingu aplikacji.
- translations/: zawiera pliki tłumaczeń generatora zorganizowanego
- var/: przechowuje zmienne pliki generowane przez aplikację, takie jak pliki cache widoków oraz danych, logi zdarzeń.
- vendor/: zawiera zależności aplikacji, w tym komponenty Symfony oraz inne pakiety zainstalowane za pomocą narzędzia do zarządzania zależnościami Composer.

W katalogu głównym projektu generatora zawarte są ważne pliki w kwestii konfiguracji aplikacji, min:

- webpack.config.js – a zawiera konfigurację dla Webpacka, narzędzia służącego do kompilacji, tworzenia paczek i bundli oraz optymalizacji zasobów takich jak JavaScript, CSS i obrazy. Konfiguracja obejmuje ustawienia dotyczące punktów wejścia aplikacji, reguł przetwarzania różnych typów plików, pluginów ułatwiających minifikację i optymalizację zasobów.
- package.json – plik wykorzystywany przez menedżera pakietów npm (Node Package Manager) i używany jest do zarządzania zależnościami projektu JavaScript, skryptami, wersjami i metadanymi projektu
- composer.json – analogicznie do package.json, plik composer.json jest wykorzystywany przez system zarządzania zależnościami Composer dla PHP. W generatorze definiuje on zależności projektu PHP, w tym biblioteki i pakiety używane w aplikacji
- .env – przechowuje zmienne środowiskowe używane w projekcie, np. do konfiguracji połączenia z główną bazą danych, informacje o typie środowiska (dev, prod, test) oraz wszelkie inne podstawowe parametry konfiguracyjne.

Zrozumienie powyżej opisanej struktury i umiejętne wykorzystanie jej możliwości było kluczowej dla efektywnej pracy nad prototypem generatora. Struktura ta zapewnia przyjazne wytyczne dotyczące organizacji projektu, co przekłada się na lepszą czytelność kodu oraz wyższą jakość i łatwość w utrzymaniu i dalszego rozwoju projektowanego generatora.

5.4. Łączenie z systemami baz danych

Prototyp oferuje możliwość eksportu danych tymczasowych w dwóch formatach, zgodnie z określoną strukturą: bezpośrednio do pliku (w formatach SQL, JSON, XML, CSV) oraz do wybranych, wiodących systemów baz danych, takich jak: MySQL / MariaDB, Microsoft SQL Server, PostgreSQL, Oracle, a także do systemów nierelacyjnych (NoSQL), takich jak MongoDB.

W kontekście połączenia z wymienionymi systemami baz danych, w prototypie wykorzystywany jest do tego celu rozszerzenie PHP pod nazwą PDO. PDO (*PHP Data Objects*), jest lekkim, spójną interfejsem do dostępu do baz danych w PHP. Pozwala ono na tworzenie kodu dostępu do bazy danych, który jest przenośny między różnymi systemami. Użycie PDO w projekcie zapewnia abstrakcję dostępu do bazy danych, co znaczy, że można łatwo zmieniać typ używanego systemu bazodanowego bez konieczności przepisywania kodu. PDO oferuje również rozszerzone możliwości bezpieczeństwa, w tym automatyczne zabezpieczanie przed atakami, np. *SQL injection*, poprzez odpowiednie parsowanie przygotowanych zapytań oraz filtrację parametrów. To sprawia, że PDO doskonale sprawdził się w opisywanym projekcie, przyśpieszając czas implementacji przy zachowaniu wysokiego stopnia bezpieczeństwa oraz wydajności. Warto wspomnieć iż niezbędnym czynnikiem do prawidłowego działania mechanizmu eksportu do wybranego systemu baz danych są właściwe uprawnienie użytkownika inicjującego połącznie. Użytkownik mus posiadać kompleksowe uprawnienia do zarządzania obiektami bazy nie tylko w kontekście dodawania i usuwania ale również np. możliwości zdejmowania sprawdzania kluczy obcych na kolumnach (np. ustawienia opcji *SET FOREIGN_KEY_CHECKS = 0* w systemie MySQL / MariaDB lub możliwości użycia polecenie *TRUNCATE*).

Listing 11 prezentuje kod nawiązania połączenia w generatorze dla baz danych MySQL / MariaDB.

Listing 11 – Przykład połączenia za pomocą PDO z system MySQL / MariaDB, źródło: Opracowanie własne

```
/**
 * Łączenie z bazą danych z użyciem PDO
 */
public function connect(): void
{
    $this->connection = new \PDO(
            "MySQL:host=" . $this->conn->getHost() . ";
         dbname=" . $this->conn->getDbName(),
         $this->conn->getUsername(), $this->conn->getPassword(),
        [\PDO::ATTR_TIMEOUT => 5]
    );
}
```

W dalszej części przedstawione oraz opisane zostały poszczególne systemy baz danych, z którymi współpracuje prototyp, wraz ze szczegółami implementacji mechanizmu pobrania oraz przetwarzania ich schematów, poprzez generowanie odpowiednich zapytań. Analiza oraz przetworzenie schematów ułatwia następnie eksport danych tymczasowych do wspomnianych środowisk.

W celu pobrania schematu bazy tworzone jest specyficzne zapytanie które kreuje tablicę złożoną ze wszystkich ich tabel. Struktura zawiera listę kolumn uwzględniając przy tym ich indywidualne właściwości, które różnią się w zależności od typu bazy danych. W kontekście baz relacyjnych każda kolumna może mieć przypisane parametry takie jak: typ pola, oznaczenie klucza głównego, oznaczenie klucz obcego, wskazanie na tabelę i kolumnę klucza obcego. Bazy nierelacyjne ze względu na swoją specyfikę w której mniej koncentrują się na ścisłej strukturze i relacjach ograniczają się do oznaczenie typu pola. W celu efektywnego interakcji pomiędzy różnymi systemami baz każda posiada własne mechanizmy połączenie, eksportu danych oraz generowania schematu, zgodny z określonym modelem. Te mechanizmy opisane są ogólnie i dziedziczone jako metody z interfejsu zdefiniowanego w pliku IConnection.php. Klasy modeli połączeń zawierają się w katalogu Model opisanym strukturze katalogów w podrozdziale 5.3.

Rysunek 11 prezentuje diagram obrazujący poszczególne modele z listą metod oraz dziedziczeniem z interfejsu IConnection.php.

Rysunek 11 – Diagram modeli baz danych , źródło: Opracowanie własne

5.4.1. MySQL / MariaDB

System MariaDB jest forkiem MySQL więc zapewnia pełną z nim kompatybilność. W opisywanym rozwiązaniu oba systemy są uważane za tożsame z taką samymi możliwościami łączenia, generowania oraz eksportu danych.

W celu pobrania schematu wybranej bazy danych, konstruowane jest specjalne zapytanie do systemowej bazy danych *information_schema* [26], która pełni rolę meta–bazy w systemach MySQL / MariaDB i zawierają szczegółowe informacje o wszystkich obiektach bazodanowych. Informacje dotyczące tabel przechowywane są w tabeli *TABLES*, zawierając tam informacje jak nazwy tabel, typy silników przechowywania danych czy też daty ich utworzenia. Szczegółowe dane o kolumnach każdej z tabel znajdują się w tabeli *COLUMNS*. Ta sekcja bazy *information_schema* dostarcza informacji na temat typów danych poszczególnych kolumn, ich różnych właściwości jak możliwości występowania null, wartości domyślnych oraz innych ograniczeń i konfiguracji. Informacje dotyczące kluczy, w tym kluczy głównych oraz kluczy obcych, są przechowywane w tabeli *KEY_COLUMN_USAGE*. Ta tabela umożliwia identyfikację relacji między tabelami, wskazując na powiązania i zależności kluczowe dla struktury relacyjnej bazy danych.

Baza systemowa *information_schema* zawiera wiele więcej tabel, które dostarczają dodatkowych informacji o bazach danych, min. takie jak:

- *SCHEMATA* – zawiera min. informacje o dostępnych schematach bazy danych, domyślnym kodowaniu znaków
- *STATISTICS* – dostarcza informacje o indeksach utworzonych w tabelach
- *VIEWS* – zawiera informacje o zdefiniowanych widokach, umożliwiając analizę ich struktury i zapytań SQL, na których są oparte
- *ROUTINES* – przechowuje informacje o procedurach składowanych i funkcjach w bazie danych, w tym nazwy, typ (procedura lub funkcja), parametry, typ danych zwracanego przez funkcje oraz ciało procedury / funkcji
- *TRIGGERS* – informacje o wyzwalaczach (triggerach) zdefiniowanych w bazie danych
- *PARAMETERS* – dostarcza wiedzę o parametrach dla procedur składowanych i funkcji.
- *TABLE_PRIVILEGES / COLUMN_PRIVILEGES* – odnosi się do uprawnień przyznanych na poziomie poszczególnych tabel / kolumn w tabelach.

Listing 12 przedstawia metodę zwracającą string z instrukcją SQL która jest wykorzystywana do pobrania schematu bazy danych dla MySQL / MaraDB.

Listing 12 – Metoda zwracająca zapytanie pobrania schematu bazy danych dla MySQL / MariaDB, źródło: Opracowanie własne

```
/**
 * Zapytanie SQL do pobrania schematu bazy danych.
 *
 * @return string
 */
public function getDbSchemaQuery(): string
{
    $dbName = $this->connection->quote($this->conn->getDbName());
    return "SELECT
                    t.TABLE_SCHEMA, t.TABLE_NAME,c.COLUMN_NAME, c.COLUMN_TYPE,
                    c.IS_NULLABLE,
        CASE
                        WHEN c.COLUMN_KEY = 'PRI' THEN 'PRI'
                        WHEN c.COLUMN_KEY = 'MUL' THEN 'FOR'
                        ELSE NULL
                    END AS 'COLUMN_KEY',
                    k.REFERENCED_TABLE_NAME, k.REFERENCED_COLUMN_NAME
                FROM  information_schema.TABLES t
                    JOIN information_schema.COLUMNS c ON t.TABLE_SCHEMA = c.TABLE_SCHEMA AND
t.TABLE_NAME = c.TABLE_NAME
                    LEFT JOIN information_schema.KEY_COLUMN_USAGE k
                        ON t.TABLE_SCHEMA = k.TABLE_SCHEMA AND t.TABLE_NAME = k.TABLE_NAME AND
c.COLUMN_NAME = k.COLUMN_NAME
                WHERE  t.TABLE_SCHEMA = {$dbName};";
}
```

5.4.2. Microsoft SQL Server

W celu pobrania struktury bazy w MSSQL używany jest obiekt *information_schema* oraz inne widoki katalogowe. W odróżnieniu od baz MySQL / MariaDB gdzie uważany jest za specjalną bazę danych, *information_schema* jest zbiorem widoków standardowych ANSI SQL, które dostarczają informacji o metadanych bazy danych [27]. Widoki katalogowe to tzw. widoki systemowe które prezentują informacje o obiektach bazy danych, użytkownikach, uprawnieniach i innych ważnych aspektach bazy danych. Niektóre widoki katalogowe używane do przeglądani oraz pobrania struktury bazy w systemach MS SQL Server:

- *sys.tables* – przechowuje informacje o tabelach w bazie danych;
- *sys.columns* – zawiera informacje o kolumnach w tabelach;
- *sys.types* – dostarcza informacje o typach danych zdefiniowanych w systemie;
- *sys.indexes / sys.index_columns* – opisują indeksy i ich kolumny;
- *sys.foreign_keys / sys.foreign_key_columns* – przechowują informacje o kluczach obcych i ich relacjach między tabelami;
- *sys.views* – zawiera informacje o widokach zdefiniowanych w bazie danych;

Na Listing 13 zaprezentowana jest metoda zwracająca string z instrukcją SQL która służy do pobrania schematu bazy danych dla MS SQL Server. Zwraca uwagę większy poziom skomplikowania w budowie zapytania w porównaniu do systemu MySQL / MariaDB.

Listing 13 – Metoda zwracająca zapytanie pobrania schematu bazy danych dla Microsoft SQL Server, źródło: Opracowanie własne

```
<?php
/**
 * Zapytanie SQL do pobrania schematu bazy danych.
 *
 * @return string
 */
public function getDbSchemaQuery(): string
{
    return "SELECT t.TABLE_SCHEMA, t.TABLE_NAME, c.COLUMN_NAME,  c.DATA_TYPE +
        CASE
            WHEN c.CHARACTER_MAXIMUM_LENGTH IS NOT NULL THEN '(' +
CAST(c.CHARACTER_MAXIMUM_LENGTH AS NVARCHAR) + ')'  ELSE ''  END AS COLUMN_TYPE,
    CASE WHEN c.IS_NULLABLE = 'YES' THEN 'YES' ELSE 'NO' END AS IS_NULLABLE,
    CASE WHEN EXISTS (
            SELECT 1 FROM sys.index_columns ic
            INNER JOIN sys.indexes i ON ic.index_id = i.index_id AND ic.object_id =
i.object_id             WHERE ic.object_id = OBJECT_ID(t.TABLE_SCHEMA + '.' + t.TABLE_NAME)
            AND ic.column_id = COLUMNPROPERTY(ic.object_id, c.COLUMN_NAME, 'ColumnId')
            AND i.is_primary_key = 1
        ) AND EXISTS ( SELECT 1 FROM sys.foreign_keys AS fk
            INNER JOIN sys.foreign_key_columns AS fkc ON fk.object_id =
fkc.constraint_object_id WHERE fk.parent_object_id = OBJECT_ID(t.TABLE_SCHEMA + '.' +
t.TABLE_NAME)AND fkc.parent_column_id = COLUMNPROPERTY(OBJECT_ID(t.TABLE_SCHEMA + '.' +
t.TABLE_NAME), c.COLUMN_NAME, 'ColumnId') )
        THEN 'FOR' WHEN EXISTS (SELECT 1 FROM sys.index_columns ic
            INNER JOIN sys.indexes i ON ic.index_id = i.index_id AND ic.object_id =
i.object_id
        WHERE ic.object_id = OBJECT_ID(t.TABLE_SCHEMA + '.' + t.TABLE_NAME)
            AND ic.column_id = COLUMNPROPERTY(ic.object_id, c.COLUMN_NAME, 'ColumnId')
            AND i.is_primary_key = 1)
        THEN 'PRI' WHEN EXISTS (
            SELECT 1 FROM sys.foreign_keys AS fk
            INNER JOIN sys.foreign_key_columns AS fkc ON fk.object_id =
fkc.constraint_object_id
            WHERE fk.parent_object_id = OBJECT_ID(t.TABLE_SCHEMA + '.' + t.TABLE_NAME)
            AND fkc.parent_column_id = COLUMNPROPERTY(OBJECT_ID(t.TABLE_SCHEMA + '.' +
t.TABLE_NAME), c.COLUMN_NAME, 'ColumnId') ) THEN 'FOR'
        ELSE '' END AS COLUMN_KEY,
    OBJECT_NAME(fk.referenced_object_id) AS REFERENCED_TABLE_NAME,
    COL_NAME(fk.referenced_object_id, fk.referenced_column_id) AS REFERENCED_COLUMN_NAME
FROM
    information_schema.TABLES t
    JOIN information_schema.COLUMNS c ON t.TABLE_SCHEMA = c.TABLE_SCHEMA AND t.TABLE_NAME =
c.TABLE_NAME
    LEFT JOIN sys.foreign_key_columns AS fk
        ON OBJECT_ID(t.TABLE_SCHEMA + '.' + t.TABLE_NAME) = fk.parent_object_id
        AND COLUMNPROPERTY(OBJECT_ID(t.TABLE_SCHEMA + '.' + t.TABLE_NAME), c.COLUMN_NAME,
'ColumnId') = fk.parent_column_id
       WHERE t.TABLE_TYPE = 'BASE TABLE'
       ORDER BY t.TABLE_SCHEMA, t.TABLE_NAME, c.ORDINAL POSITION;";
}
```

5.4.3. PostgresSql

W PosgresSql aby pobrać schemat bazy danych do użycia w generatorze, podobnie jak w systemach MySQL / MariaDB wykorzystywana jest baza *information_schema* [28], którego kod instrukcji został przedstawiony na Listing 14. W kontekście tego schematu różnice jakie zachodzą

pomiędzy tymi systemami to min.: PostgreSQL ściśle przestrzega standardu SQL w zakresie schematu, MySQL/MariaDB zawiera rozszerzenia niestandardowe, które nie są zgodne ze standardem SQL. PostgreSQL udostępnia szerszy zakres informacji w schemacie *information_schema*, obejmujący dodatkowo np. role, funkcje, ograniczenia, reguły natomiast MySQL / MariaDB ma węższy zakres informacji, skupiając się głównie na tabelach, kolumnach, widokach i procedurach.

Listing 14 – Metoda zwracająca zapytanie pobrania schematu bazy danych dla PostgreSQL , źródło: Opracowanie własne

```
/**
 * Zapytanie SQL do pobrania schematu bazy danych.
 *
 * @return string
 */
public function getDbSchemaQuery(): string
{
    return ' SELECT
            t.table_schema AS "TABLE_SCHEMA", t.table_name AS "TABLE_NAME",
            c.column_name AS "COLUMN_NAME", UPPER(c.data_type) AS "COLUMN_TYPE",
            c.is_nullable AS "IS_NULLABLE", CASE
                WHEN tc.constraint_type = \'PRIMARY KEY\' THEN \'PRI\'
                WHEN tc.constraint_type = \'FOREIGN KEY\' THEN \'FOR\'
                ELSE NULL
            END AS "COLUMN_KEY",
            ccu.table_name AS "REFERENCED_TABLE_NAME",
            ccu.column_name AS "REFERENCED_COLUMN_NAME"
        FROM information_schema.tables t
            JOIN information_schema.columns c ON t.table_schema = c.table_schema AND
t.table_name = c.table_name
            LEFT JOIN information_schema.key_column_usage kcu ON t.table_schema =
kcu.table_schema AND t.table_name = kcu.table_name AND c.column_name = kcu.column_name
            LEFT JOIN information_schema.table_constraints tc ON kcu.table_schema =
tc.table_schema AND kcu.table_name = tc.table_name AND kcu.constraint_name =
tc.constraint_name
            LEFT JOIN information_schema.constraint_column_usage ccu ON tc.constraint_schema =
ccu.constraint_schema AND tc.constraint_name = ccu.constraint_name
        WHERE
            t.table_schema NOT IN (\'pg_catalog\', \'information_schema\')
            AND t.table_type = \'BASE TABLE\'
        ORDER BY t.table_schema, t.table_name, c.ordinal_position';
}
```

5.4.4. Oracle Database

W celu pobrania schematu bazy danych Oracle DB w prototypie wykorzystano widoki systemowe zawarte w oprogramowaniu [29], które zwracają potrzebne informację do przetworzenia struktury obiektu. Poniżej lista użytych widoków które zwracają potrzebne dane:

- *ALL_TAB_COLUMNS* – zawiera informacje o kolumnach we wszystkich tabelach dostępnych dla użytkownika, takie jak nazwy kolumn, typy danych, precyzja, długość itd. Jest to kluczowe narzędzie do analizy struktury tabel;
- ALL_CONS_COLUMNS – dostarcza informacji o kolumnach zaangażowanych w ograniczenia (constraints) takie jak klucze główne, klucze obce itp. Pozwala zrozumieć relacje między tabelami oparte na kluczach;
- ALL_CONSTRAINTS – zawiera szczegółowe informacje o wszystkich ograniczeniach zdefiniowanych dla tabel dostępnych użytkownikowi, w tym typ ograniczenia. Umożliwia zrozumienie reguł integralności danych;
- ALL_TABLES – zapewnia ogólne informacje o tabelach, takie jak nazwy, przestrzenie tabel, atrybuty. Daje wysokopoziomowy przegląd struktury bazy;

Metoda zwracający zapytanie w celu pobrania schemat bazy dla systemu Oracle DB pokazana została na Listing 15.

Listing 15 – Metoda zwracająca zapytanie pobrania schematu bazy danych dla OracleDB, źródło: Opracowanie własne

```
/**
 * Zapytanie SQL do pobrania schematu bazy danych.
 * @return string
 */
public function getDbSchemaQuery(): string
{
    $dbName = $this->connection->quote($this->conn->getDbName());
    return "SELECT
            utc.TABLE_NAME, utc.COLUMN_NAME,
            utc.DATA_TYPE ||
            CASE  WHEN utc.DATA_TYPE = 'NUMBER' THEN '(' ||
COALESCE(TO_CHAR(utc.DATA_PRECISION), '*') || ',' || COALESCE(TO_CHAR(utc.DATA_SCALE), '0')
|| ')' WHEN utc.DATA_TYPE LIKE '%CHAR%' THEN '(' || TO_CHAR(utc.CHAR_LENGTH) || ')'
                ELSE ''
            END AS COLUMN_TYPE, utc.NULLABLE as IS_NULLABLE,
            (SELECT CASE WHEN cons.CONSTRAINT_TYPE = 'P' THEN 'PRI' ELSE 'FOR' END
             FROM ALL_CONS_COLUMNS col
             INNER JOIN ALL_CONSTRAINTS cons ON col.CONSTRAINT_NAME = cons.CONSTRAINT_NAME
             WHERE col.TABLE_NAME = utc.TABLE_NAME
             AND col.COLUMN_NAME = utc.COLUMN_NAME
             AND col.OWNER = '{$dbName}'
             AND (cons.CONSTRAINT_TYPE = 'P' OR cons.CONSTRAINT_TYPE = 'R')) AS COLUMN_KEY,
            (SELECT rcc.TABLE_NAME
             FROM ALL_CONS_COLUMNS acc
             INNER JOIN ALL_CONSTRAINTS ac ON acc.CONSTRAINT_NAME = ac.CONSTRAINT_NAME AND
ac.CONSTRAINT_TYPE = 'R'
             INNER JOIN ALL_CONS_COLUMNS rcc ON ac.R_OWNER = rcc.OWNER AND
ac.R_CONSTRAINT_NAME = rcc.CONSTRAINT_NAME
             WHERE acc.TABLE_NAME = utc.TABLE_NAME
             AND acc.OWNER = '{$dbName}'
             AND acc.COLUMN_NAME = utc.COLUMN_NAME) AS REFERENCED_TABLE_NAME,
            (SELECT rcc.COLUMN_NAME
             FROM ALL_CONS_COLUMNS acc
             INNER JOIN ALL_CONSTRAINTS ac ON acc.CONSTRAINT_NAME = ac.CONSTRAINT_NAME AND
ac.CONSTRAINT_TYPE = 'R'
             INNER JOIN ALL_CONS_COLUMNS rcc ON ac.R_OWNER = rcc.OWNER AND
ac.R_CONSTRAINT_NAME = rcc.CONSTRAINT_NAME
             WHERE acc.TABLE_NAME = utc.TABLE_NAME AND acc.OWNER = '{$dbName}'
             AND acc.COLUMN_NAME = utc.COLUMN_NAME) AS REFERENCED_COLUMN_NAME
        FROM ALL_TAB_COLUMNS utc
        WHERE utc.OWNER = '{$dbName}'
        ORDER BY utc.TABLE_NAME,utc.COLUMN_ID
    ";
}
```

5.4.5. MongoDB

W opisywanym prototypie jako schemat dla kolekcji w bazie MongoDB przyjęto pobranie pierwszego rekordu danej kolekcji [30]. Rekord ten następnie jest przetworzony rekurencyjne w celu zbudowania tablicy podobnej jak w przypadku relacyjnych baz danych. Na Listing 16 przedstawione zostały metody pobrania rekordu oraz rekurencyjne zbudowanie struktury w celu wykorzystania do przypisania i eksportu danych tymczasowych.

Listing 16 – Metody przetwarzające rekurencyjnie dokument w MongoDB,, źródło: Opracowanie własne

```
/**
 * Pobranie rekordu z bazy danych oraz przetworzenie go na strukturę dokumentu
 * @return array
 **/
public function getDbSchema(): array
{
    $namespace = $this->conn->getDbName() . '.' . $this->conn->getCollaction();
    $query = new \MongoDB\Driver\Query([], ['limit' => 1]);
    $cursor = $this->getConnection()->executeQuery($namespace, $query);
    $document = current($cursor->toArray());
    $structure = [];
    if ($document) {
        $structure = $this->documentStructure($document, '');
    }
    return [$this->conn->getCollaction() => ['column' => $structure]];
}
/**
 * Rekurencyjne generowanie struktury dokumentu
 * @param array|object $doc The document to generate the structure for MongoDB Document
 * @param string $indent
 * @return array
 */
private function documentStructure($doc): array
{
    $structure = [];
    foreach ($doc as $key => $value) {
        if ($key != '_id') {
            if ((is_array($value) || is_object($value))) {
                $structure[$key] = $this->documentStructure($value);  // rekursywnie
            } else {
                $structure[$key] = [
                    'TABLE_SCHEMA' => $this->conn->getDbName(),
                    'TABLE_NAME' => $this->conn->getCollaction(),
                    'COLUMN_NAME' => $key,
                    'COLUMN_TYPE' => $this->getColumnType($value),
                    'IS_NULLABLE' => 1,
                    'COLUMN_KEY' => $key == '_id' ? 'PRI' : '',
                    'REFERENCED_TABLE_NAME' => '',
                    'REFERENCED_COLUMN_NAME' => $value,
                ];
            }
        }
    }
    return $structure;
}
```

5.5. Generowanie danych tymczasowych

W opisywanym prototypie generowanie struktury danych tymczasowych realizowane jest na różnorodne sposoby. Po dokonaniu przez użytkownika wyboru kategorii danych, aplikacja sprawdza czy wybrana kategoria ma w bazie ustawioną wartość pola *is_automatic = false*. W takim przypadku system pobiera losowe wartości odpowiadające wybranej kategorii. W ramach instalacji prototypu tworzona jest również baza danych zawierająca kilkadziesiąt kategorii wraz z przypisanymi danymi.

Kategorie i dane mają odpowiednie oznaczenie w bazie w polach *is_default = true* oraz *is_constant = true* co oznacza iż nie mogą być edytowane lub usuwane przez użytkownika. W kategoriach z takim oznaczeniem istniej możliwość tworzenie i przypisywania własnych danych, którymi użytkownik będzie mógł dowolnie zarządzać.

Poniżej częściowa lista kategorii domyślnych wraz z danymi i informacją co te dane przedstawiają, które zostają dostarczona wraz z instalacją prototypu. W nawiasach podana została ilość wierszy danych:

Adres

- Gmina (2221) – lista gmin w Polsce;
- Kod pocztowy (4004) – list polskich kodów pocztowych;
- Miasto (185) – lista polskich miast;
- Państwa (326) – lista państw w wersji, różne wersje językowe;
- Powiaty (363) – lista powiatów w Polsce;
- Stany USA (50) – lista stanów w USA;
- Ulica (238) – lista ulic polskich ;
- Województwo (16) – lista województw w Polsce;

Dane osobowe

- Dokument ID (816) – lista przykładowych numerów dowodów osobistych;
- Email (38668) – adresy email z losowo wygenerowanych domen utworzone poprzez połączenie kategorii imion i nazwiska;
- Imię (38668) – lista imiona z bazy PESEL Głównego Urzędu Statystycznego (stan na 23.10.2023);
- Nazwa użytkownika(38668) – nawa użytkownika utworzone poprzez połączenie kategorii imion i nazwiska;
- Nazwisko (33395) – lista nazwisk z bazy PESEL Głównego Urzędu Statystycznego (stan na 23.10.2023);
- Płeć (4) – płcie w wersji polskiej i angielskiej;

Pojazdy

- Motory (122) – modele motocykli;
- Samochód (168) – modele samochodów;
- Samoloty (43) – modele samolotów;
- Statki (57) – modele statków;

Praca

- Nazwy firm(74) – nazwy popularnych firm;
- Przemysł rodzaje (114) – rodzaje przemysłu, różne wersje językowe;
- Stanowiska (79) – rodzaje stanowisk, różne wersje językowe;
- Umiejętności (64) – umiejętności pracownika, różne wersje językowe;

Sport

- Drużyny(105) – drużyny z kilku krajów;
- Rodzaje (68) – rodzaje sportów, różne wersje językowe;
- Role (85) – lista ról / pozycji sportowych, różne wersje językowe;
- Wyposażenia (80) – list akcesoriów sportowych, różne wersje językowe;

Powyższa lista stanowi jedynie fragment początkowego zestawu kategorii danych. Użytkownicy mają możliwość dodawania do każdej z nich własnej listy danych, a także etykietowania tych danych, aby ułatwić ich wyszukiwanie. Ponadto, mogą tworzyć własne grupy kategorii i zasilać je spersonalizowanymi danymi, aby zwiększyć elastyczność i dostosowanie systemu do indywidualnych potrzeb.

W przypadku kiedy wybrana kategoria ma wartość pola *is_automatic = false* system bierze wartości parametrów zadanych przez użytkownika, wykonuje konkretną funkcję przypisaną do identyfikatora kategorii, następnie generuje odpowiednie dane. Przykładowo wybierając kategorię *Integer random* generator pobiera od użytkownika przedział wartości początkowej i końcowej (np. 0 – 100) i generuje liczby losowe z tego przedziału. Poniżej lista kategorii generowania danych automatycznych wraz z parametrami które przyjmują funkcje generujące te dane:

- *Daty (data_od – data_do)* – generuje losowe daty z podanego zakresu w formacie YYY–MM–DD;
- *Float (zakres_od – zakres_do)* – generuje liczby zmienno–przecinkowe z podanego zakresu;
- *Integer autoincrement (od)* – generuje liczby calkowite z incrementacją o jeden, począwszy od parametru początkowego;
- *Integer random (od – do)* – tworzy całkowite liczby losowe z podanego zakresu
- *Pattern regex (wzór)* – tworzy losowy string według zadanego wyrażenia regularnego, przykładowe użycia funkcji wraz z wynikiem:
    - generowanie losowego znaku literowo / liczbowego
        - wzór: *[a–zA–Z0–9_]*
        - wynik: K
    - generowanie losowego znaku nie literowego
        - wzór: *[^a–zA–Z0–9_]*
        - wynik: !
    - generowanie 10 liczb całkowitych z zakresu 0–9:
        - wzór: *\d{10}*
        - wynik: 4176935124
    - wygenerowanie z zastosowanie grupowania losowych wartość, np. wygeneruj znak ‘a’ lub losowy 5–cio znakowy ciąg składający się z liter ‘d’ lub y:
        - wzór: *a|(y|d){5}*
        - wynik a *lub* yddyy
    - wygenerowanie losowego adresu http lub https w domenie .com lub .pl:
        - wzór:*(https|http)://www\.[a–z0–0–9]{4,12}\.(com|pl)*
        - wynik: *https://www.o3kigdto8s.com*
    - wygenerowanie losowego adresu email w domenie .com lub .pl:
        - wzór: *[a–z0–0–9]{3,10}@[a–z0–0–9]{3,10}\.(com|pl)*
        - wynik: *syg4@orr.pl*

- *Pattern string (wzór)* – uproszczona funkcja względem funkcji z wyrażeniami regularnymi. Tworzy string według zadanego wzoru lub szablonu. Pozwala na wygenerowania w prosty sposób często używanych danych. Poniżej lista parametrów które może przyjąć funkcja oraz przykładowe wyniki działania:
    - *\ip* – generuje losowy adres IP, np.: *\ip* – 184.145.23.4
    - *\pesel* – generuje losowy nr PESEL, np.: – 68040545188
    - *\regon* – generuje losowy nr REGON, np.: – 648757487
    - *\nip* – generuje losowy nr NIP, np.: – 9380897641
    - *\w{x}* – generuje napis o długości x (duże i małe litery), np.: *\w{3}* – xYs
    - *\W{x}* – generuje napis o długości x (duże litery), np.: *\W{3}* – XYS
    - *\d{x}* – generuje liczby całkowite o długości x, np.: *\d{4}* – 8765

Kolejnym sposobem generowania danych randomizowanych jest wbudowany tryb edytora JavaScript który umożliwia dowolną manipulację danymi z wybranych kategorii. W widoku wyboru kategorii (więcej o interfejsie użytkownika zostało opisane w rozdziale 0) po włączeniu opcji w zakładce „Stwórz dane używając JavaScript" oraz wybraniu przynajmniej jednej kategorii, istnieje możliwość dowolnej manipulacji danymi za pomocą edytora JavaScript.

Mechanizm działania tej funkcjonalności polega na tym iż po wybraniu kategorii, zostaje ona umieszczona w specjalnej stałej tablicy JavaScript `categoryTab`, którego kod przedstawiono na Listing 17. Po wprowadzeniu kodu w edytorze JavaScript generator sprawdza składnie i zwraca wynikowy rekord przypisany do zmiennej `createdText`. W przypadku błędu składni użytkownikowi zostaje przedstawione okno z błędem. Wprowadzony kod JavaScript jest zapisany w zmiennej sesyjnej bezpośrednio przy polu przepisania, czyli kolumnie eksportu. Podczas procesu eksportu danych do pliku lub systemu baz danych, dla wybranej kategorii dane zostają wygenerowane w sposób opisany wcześniej, biorąc pod uwagę wartość atrybutu *is_automatic*. Ostatnim krokiem jest wykonanie po stronie użytkownika funkcji JavaScript `eval()`. Działanie `eval()` polega na interpretacji i wykonaniu przekazanego jako argument ciągu znaków jako kodu JavaScript. W opisywanym przypadku generowania danych, za każdym razem, gdy potrzebny jest nowy rekord, `eval()` wykonuje wcześniej przygotowany kod JavaScript, który manipuluje lub tworzy dane w oparciu o zdefiniowane instrukcje.

Fragment przykładowego kodu demonstrującego metodę przetwarzania danych z dwóch wybranych kategorii, takich jak „Imię" i „Nazwisko" zawarty jest w Listing 18. Generator iteruje przez tablicę `categoryTab`, która zawiera elementy z tych kategorii. Dla każdego elementu *v* w tablicy, jego wartość (`v.value`) jest przekształcana poprzez usunięcie wszelkich znaków niebędących literami ani cyframi (za pomocą wyrażenia regularnego `/[^a-zA-Z0-9 ]/g`), a następnie konwertowanie wszystkich liter na małe. Wynikowe ciągi są łączone w jeden ciąg `createdText`, z dodatkowym separatorem w postaci kropki (`'.'`) między każdym elementem. Na końcu usuwana jest nadmiarowa kropka. Wynikiem jest postać danych które możemy użyć jako np. pole nazwa użytkownika.

Listing 17 - Tablica JavaScript z wybranymi kategoriami danych, źródło: Opracowanie własne

```
const categoryTab = [{"id": 1,"name": "Imię","value": "Grzegorz","tag": []},{"id":
2,"name": "Nazwisko","value": "Swierkot","tag": []}];
```

Listing 18 – Przykładowy kod JavaScript manipulujący danymi, źródło: Opracowanie własne

```
categoryTab.forEach(function(v){
    createdText += v.value.replace(/[^a-zA-Z0-9 ]/g, '').toLowerCase() + '.'
})
createdText = createdText.slice(0,-1);
```

Wynik: `grzegorz.swierkot`

5.6. Instalacja

Proces instalacji prototypu obejmuje skonfigurowanie środowiska PHP, JavaScript oraz bazy danych MariaDB. Jako serwer www zostanie użyty moduł zawarty w PHP. Niezbędne będzie również instalacja odpowiednich narzędzi do zarządzania zależnościami kodu dla PHP oraz JavaScript. W tym podrozdziale przedstawiono poszczególne etapy instalacji wraz z niezbędnymi składnikami konfiguracyjnymi dla lokalnego środowiska Windows.

5.6.1. PHP

Pierwszym etapem jest pobranie interpretatora PHP w minimalnej wersji 8.1 oraz zainstalowanie i poprawne skonfigurowanie wtyczek niezbędnych do funkcjonowania generatora. Zmiany należy nanieść w głównym pliku konfiguracyjnym php.ini:

- extension=php_fileinfo.dll
- extension=php_gd.dll
- extension=php_apcu.dll
- extension=php_openssl.dll
- extension=php_curl.dll
- extension=php_pdo_sqlsrv_81_nts_x64.dll
- extension=php_oci8_19.dll
- extension=php_pdo_oci.dll
- extension=php_pdo_pgsql.dll
- extension=php_mongodb.dll
- extension=php_gettext.dll
- extension=php_gmp.dll
- extension=php_intl.dll
- extension=php_pdo_MySQL.dll
- extension=php_imap.dll
- extension=php_mbstring.dll
- extension=php_enchant.dll
- extension=php_soap.dll

Następnie należy dodać katalog PHP do zmiennej środowiskowej PATH, aby móc uruchamiać PHP z każdego miejsca w wierszu poleceń. Można to zrobić, otwierając *Panel sterowania > System i zabezpieczenia > System > Zaawansowane ustawienia systemu > Zmienne środowiskowe,* a następnie edytując zmienną Path, dodając ścieżkę do katalogu PHP.

W kolejnym kroku niezbędna jest instalacja narzędzi do zarządzania kodu Composer, który pobierze i skonfiguruje składniki frameworka Symfony. W celu instalacji Composer należy odwiedzić stronę *getcomposer.org* i postępować zgodnie z instrukcjami instalacji dla systemu Windows. Po instalacji, w katalogu głównym projektu należy użyć komendy widocznej na Listing 19:

Listing 19 – Instalacja pakietów za pomocą Composer, źródło: Opracowanie własne

```
composer install
```

Po wykonaniu powyższej komendy, Composer pobierze wszystkie wymagane zależności projektowe dla środowiska PHP, które są umieszczone w pliku *composer.json,* następnie umieści je w katalogu *vendor.*

5.6.2. Środowisko JavaScript

W celu prawidłowego skonfigurowania JavaScript należy zainstalować środowisko uruchomieniowa Node.js, które można pobrać ze strony *nodejs.org.* Po ukończeniu instalacji, należy sprawdzić czy poprawnie zainstalował się również menadżer pakietów *npm* używając komendy pokazanej na Listing 20.

Listing 20 – Sprawdzenie wersji menadżera pakietów npm, źródło: Opracowanie własne

```
npm –v
```

Następnie w celu instalacji wszystkich zależności projektowych oraz skompilowanie plików JavaScript należy użyć komend widocznych na Listing 21.

Listing 21 – Instalacja zależności oraz kompilacja plików JavaScript za pomocą npm, źródło: Opracowanie własne

```
npm install

npm run watch
```

Po wykonaniu powyższych kroków środowisko JavaScript zostanie automatycznie skonfigurowane.

5.6.3. Baza danych MariaDB

Instalacja i konfiguracja bazy danych MariaDB do użycia w opisywanym prototypie obejmuje dwuetapowy proces. Rozpoczyna się od pobrania odpowiedniej wersji dla systemu Windows oprogramowania z oficjalnej strony *mariadb.org* . Na tej stronie znajdują się szczegółowe instrukcje, które krok po kroku przeprowadzą przez proces instalacji, upewniając się, że wszystkie komponenty są prawidłowo skonfigurowane. Po zakończeniu instalacji, następnym ważnym krokiem jest przygotowanie samej bazy danych, co obejmuje utworzenie nowej bazy oraz załadowanie do niej początkowej struktury i danych. Struktura ta, wraz z zestawem początkowych kategorii i danych tymczasowych, jest dostarczana w formie pliku SQL znajduje się w katalogu głównym projektu pod nazwą *baza_generator_danych_tymczasowych.sql*. Plik ten zawiera wszystkie niezbędne polecenia SQL do utworzenia tabel, zdefiniowania relacji między nimi oraz wstępne wypełnienie bazy danych początkowymi kategoriami i danymi. Wczytanie tego pliku do MariaDB może być wykonane przy użyciu narzędzia wiersza poleceń lub interfejsu graficznego, takiego jak phpMyAdmin,

Po instalacji silnika bazy oraz utworzeniu struktury należy odpowiednio zmodyfikować znajdujący się w katalogu głównym plik *.env*. Przechowuje on zmienne środowiskowe używane w projekcie, np. do konfiguracji połączenia z główną bazą danych. Na Listing 22 została przedstawiona zawartość wiersza z tego pliku z przykładowym wypełnieniem wartości konfiguracyjnych połączenia bazy danych.

Listing 22 – Przykładowy wpisy konfiguracyjny pliku .env do połączenia z bazą danych, źródło: Opracowanie własne

```
DATABASE_URL=MySQL://generator:passwd@localhost:3306/generator?serverVersion=10.4.8-
MariaDB
```

Generator danych tymczasowych, będąc aplikacją webową, do swojego działania wymaga serwera www. Do uruchomienia opisywanego rozwiązania można użyć dowolnego popularnego oprogramowania takiego jak Apache czy Nginx lub również można skorzystać z prostego, ale efektywnego serwera wbudowanego w PHP. Jest on szczególnie przydatny w fazie rozwoju i testowania aplikacji oraz gdy zachodzi potrzeba szybko uruchomić aplikację bez konieczności skomplikowanej konfiguracji zewnętrznego serwera www. W środowisku developerskim, nastawionym na dużą wydajność przetwarzanie setki tysięcy żądań zalecane jest skorzystanie z bardziej wydajnego rozwiązania.

Aby uruchomić wbudowany serwer PHP, wystarczy wykonać w wierszu poleceń komendę pokazaną na Listing 23.

Listing 23 – Uruchomienie serwera www, źródło: Opracowanie własne

```
php –S localhost:8080 –t public
```

Polecenie to składa się ona z kilku elementów, które mają kluczowe znaczenie dla prawidłowego działania serwera:

- *php* – jest to wywołanie interpretera PHP, który uruchamia serwer;
- *–S* – ta flaga, która inicjuje serwer wbudowany w PHP;
- *localhost:8080* – określa adres i port, na którym serwer będzie nasłuchiwał połączeń. W tym przypadku, localhost oznacza, że serwer będzie dostępny lokalnie na komputerze, na którym jest uruchomiony, a 8080 to numer portu;
- *–t public* – ten parametr określa publiczny katalog główny aplikacji, czyli miejsce, z którego serwer będzie serwował pliki;

Uruchomienie serwera w ten sposób pozwala na dostęp do aplikacji poprzez przeglądarkę internetową, dostępne adresu *http://localhost:8080*.

5.7. Interfejs użytkownika

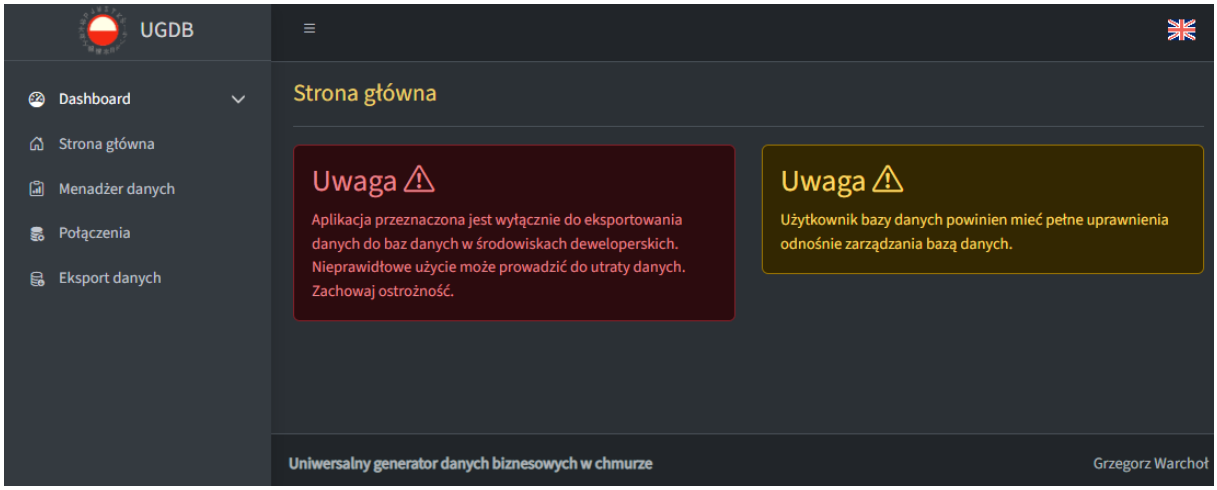
Opisywany prototyp został zaprojektowany jako aplikacja internetowa, z naciskiem na estetykę oraz funkcjonalność interfejsu użytkownika. Wykorzystanie frameworka CSS Bootstrap oraz technologii Hotwire Turbo pozwoliło na stworzenie aplikacji nie tylko estetycznej wizualnie, ale także responsywnej, nowoczesnej, zapewniającą wrażenia tzw. *single–page application.* Aplikacja posiada polską oraz angielską wersję językową, dzięki temu wzrasta jest przystępność w środowiskach międzynarodowych.

Poniżej przedstawiono listę menu generatora wraz z głównymi podstronami. W dalszej części dokładnie omówione zostały poszczególne elementy menu wraz z funkcjami dostępnymi na każdej z podstron.

- Strona główna
- Menadżer danych
  - Przeglądaj / edytuj
  - Stwórz dane
  - Kategorie / Tagi
  - Importuj dane
- Połączenia
- Eksport danych
  - Eksport do pliku
  - Eksport do bazy danych

5.7.1. Strona główna

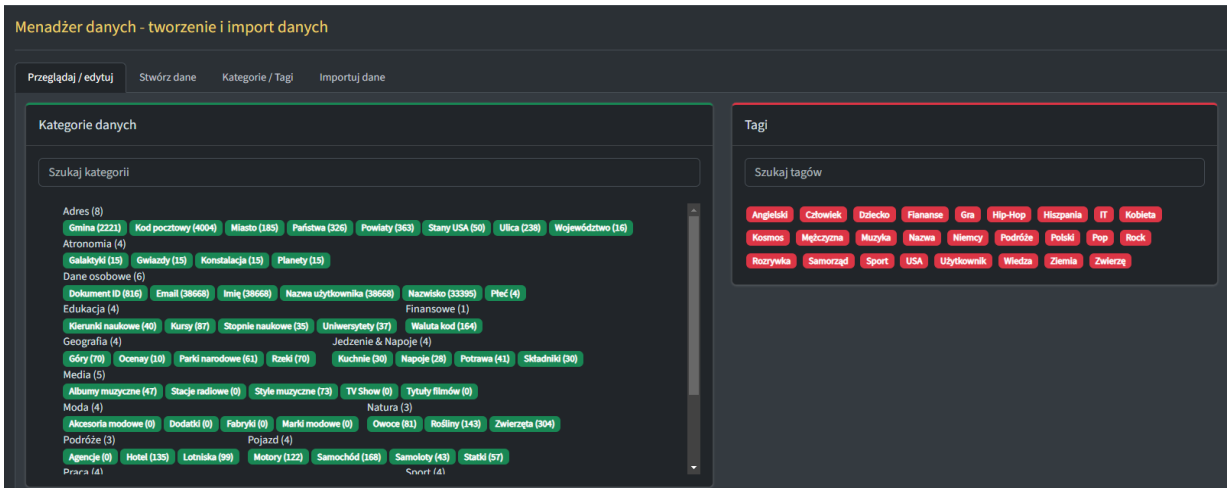
Punkt startowy aplikacji, zapewniający informacje i uwagi odnośnie generatora. Zastosowanie się do tych uwag jest niezwykle istotne w kwestii zabezpieczenie spójności danych w bazach do których przeprowadzany jest eksport. Stronę główną aplikacji przedstawiono na Rysunek 13.



Rysunek 12 – Strona główna, źródło: Opracowanie własne

5.7.2. Menadżer danych – Przeglądaj / edytuj

Na tej stronie możemy przeglądać dane według kategorii oraz tagów. Istnieje możliwość wyszukiwania tych obiektów. Po kliknięci kategorii lub tagu wyświetli się lista danym tymczasowych przypisanych do danej kategorii lub zawierająca oznaczenia danego tagu. Widok główny został ukazany na Rysunek 13.

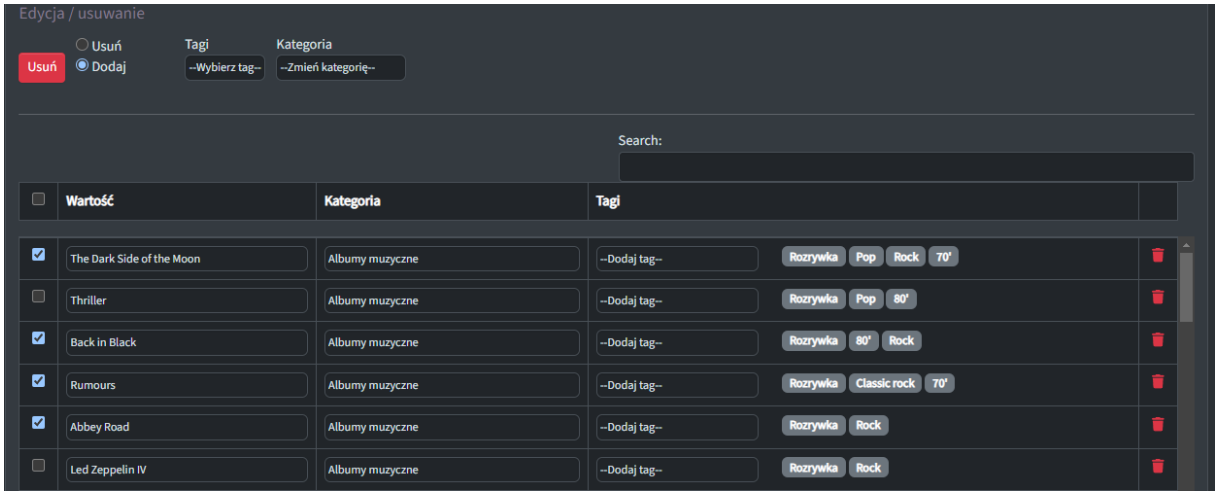


Rysunek 13 – Menadżer danych – lista kategorii i tagów, źródło: Opracowanie własne

Po kliknięciu na daną kategorię lub tag, zostanie wyświetlona lista danych tymczasowych według zadanego filtra. Widok listy został przedstawiony na Rysunek 14. W pierwszej kolumnie listy znajdują się formatki checkbox, którymi możemy zaznaczać / odznaczać dane w celu masowej edycji. Po zaznaczeniu minimum jednej wartości pojawi się panel akcji do zarządzania wszystkimi zaznaczonymi obiektami. Widniejące w nim opcje to:

- usunięcia z bazy
- oznaczenie usunięcia lub dodania
  - kategorii
  - tagu

W kolejnych kolumnach listy danych tymczasowych możemy, indywidualnie dla każdego wiersza zmienić kategorię danych, oznaczyć tagiem, po kliknięciu w istniejący tag usunąć przypisanie. W ostatniej kolumnie znajduje się opcja trwałego usunięcia danych. Edycja danych tymczasowych jest możliwa tylko dla wartości stworzonych przez użytkownika (wartość pola w bazie *is_constant = false)*. W danych początkowe, dostarczanych wraz z instalacją systemu (wartość pola w bazie *is_constant = true)* użytkownik może jedynie edytować tagowanie, pozostałe są niezmienne.



Rysunek 14 – Menadżer danych – lista danych z możliwością edycji, źródło: Opracowanie własne

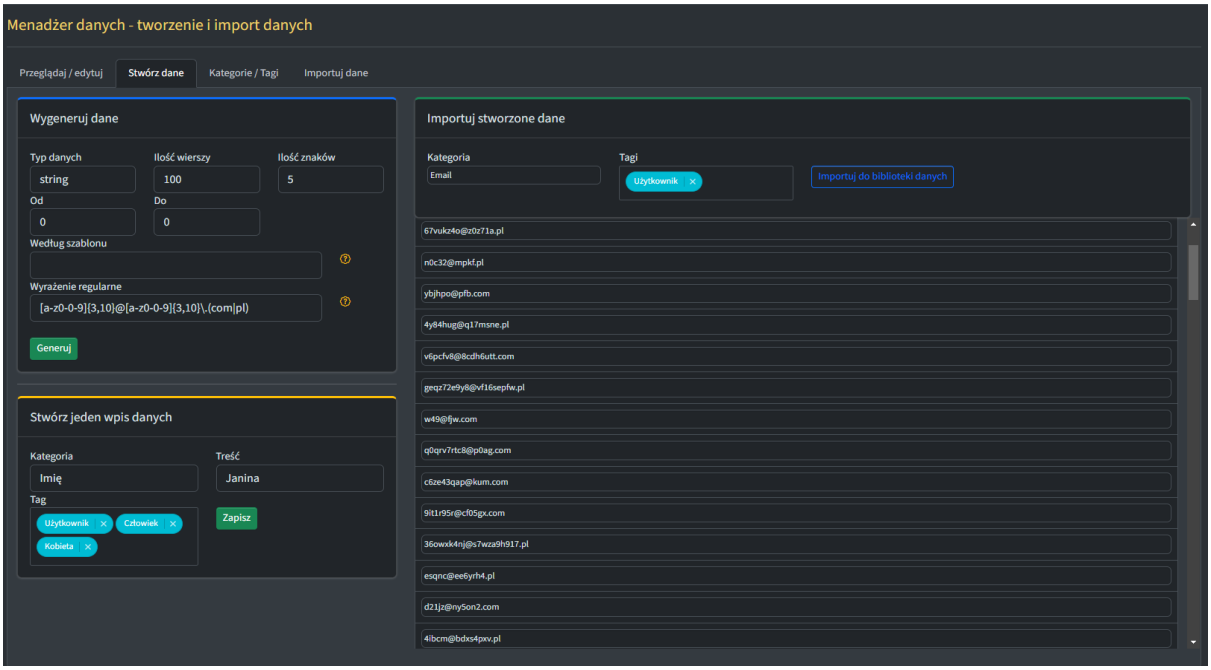
5.7.3. Menadżer danych – Stwórz dane

W tym widoku, pokazanym na Rysunek 15 istnieje możliwość stworzenie w prosty sposób własnych danych tymczasowych oraz zaimportowania i zachowania ich w bazie głównej prototypu. Po lewej stronie znajdują się dwa panele. W pierwszym o nazwie *Wygeneruj dane* wykonywana jest generacje według zadanych ilości wierszy oraz innych parametrów taki jak:

- Typ danych – oznaczanie czy typ to *string* czy *numer;*
- Ilość znaków – dla typu *string* ile generować znaków, dla typu *number;*
- Od / Do – długość napisu dla typu *string* lub liczby dla *numer;*
- Według szablonu – tworzy napis według stworzonych wcześniej szablonów np. \ip – generowanie numerów IP, \nip – generowanie numerów NIP. Więcej opisane zostało w podrozdziale 5.5. Jeżeli pole nie jest puste, wszystkie powyższe parametry są ignorowane;
- Wyrażenie regularne – tworzy ciąg według wyrażenie regularnego. Więcej opisane zostało w podrozdziale 5.5 W przypadku gdy to pole nie jest puste, wszystkie powyższe parametry są ignorowane;

W drugim panelu *Stwórz jeden wpis danych* jak nazwa wskazuje istnieje możliwość dodania bezpośrednio do biblioteki danych jednej wartości wraz z kategorią oraz oznaczeniem tagów.

Po prawej stronie opisywanego widoku znajduje się obszar wynikowy generowania poprzez panel *Wygeneruj dane.* Po weryfikacji poprawności stworzonych danych użytkownik może wybrać odpowiednią kategorię danych oraz oznaczyć odpowiednimi tagami. Przycisk *Importuj do biblioteki danych* pozwala na zapis bezpośrednio do bazy generatora. Podczas importu system sprawdza czy istnieje wartość danych z kategorią w celu zabezpieczenia się przed tworzeniem duplikatów i nadmiarowością danych. Dane stworzone i zaimportowane w ten sposób są w pełni zarządzalne przez użytkownika, wpisy w bazie mają wartość pola *is_constant = false.*

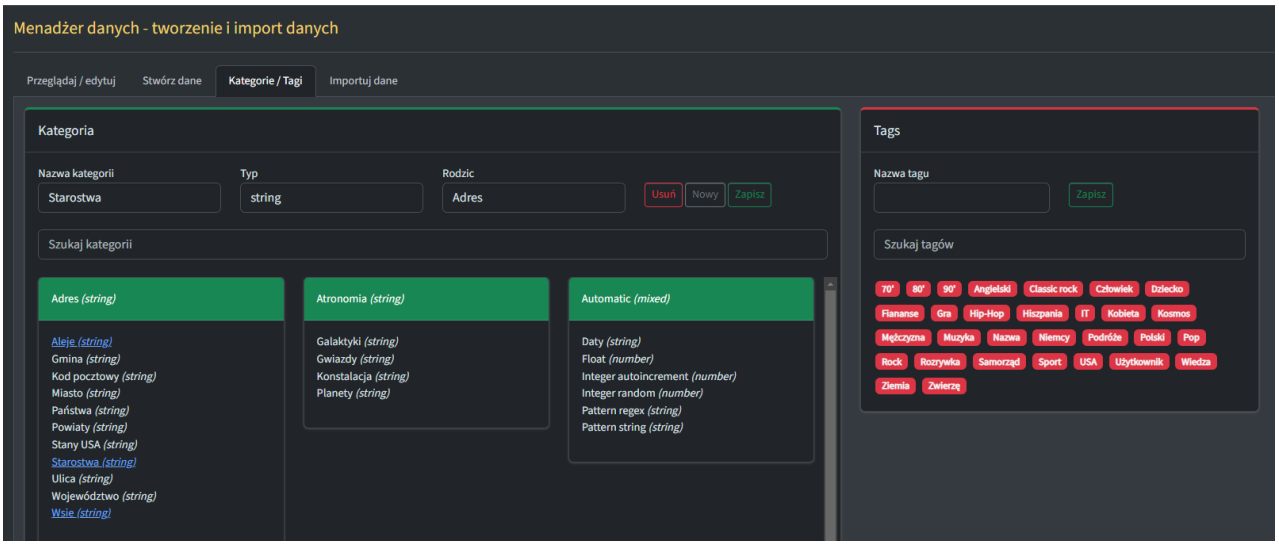


Rysunek 15 – Menadżer danych tworzenie i import danych do bazy generatora, źródło: Opracowanie własne

5.7.4. Menadżer danych – Kategorie / Tagi

Moduł ten odpowiada za dodawanie, edycję, usuwanie kategorii danych oraz tagów. Kategorii powstałych z pierwszą instalacją użytkownik nie może edytować. W kontekście tagów nie ma takich ograniczeń i może swobodnie nimi zarządzać.

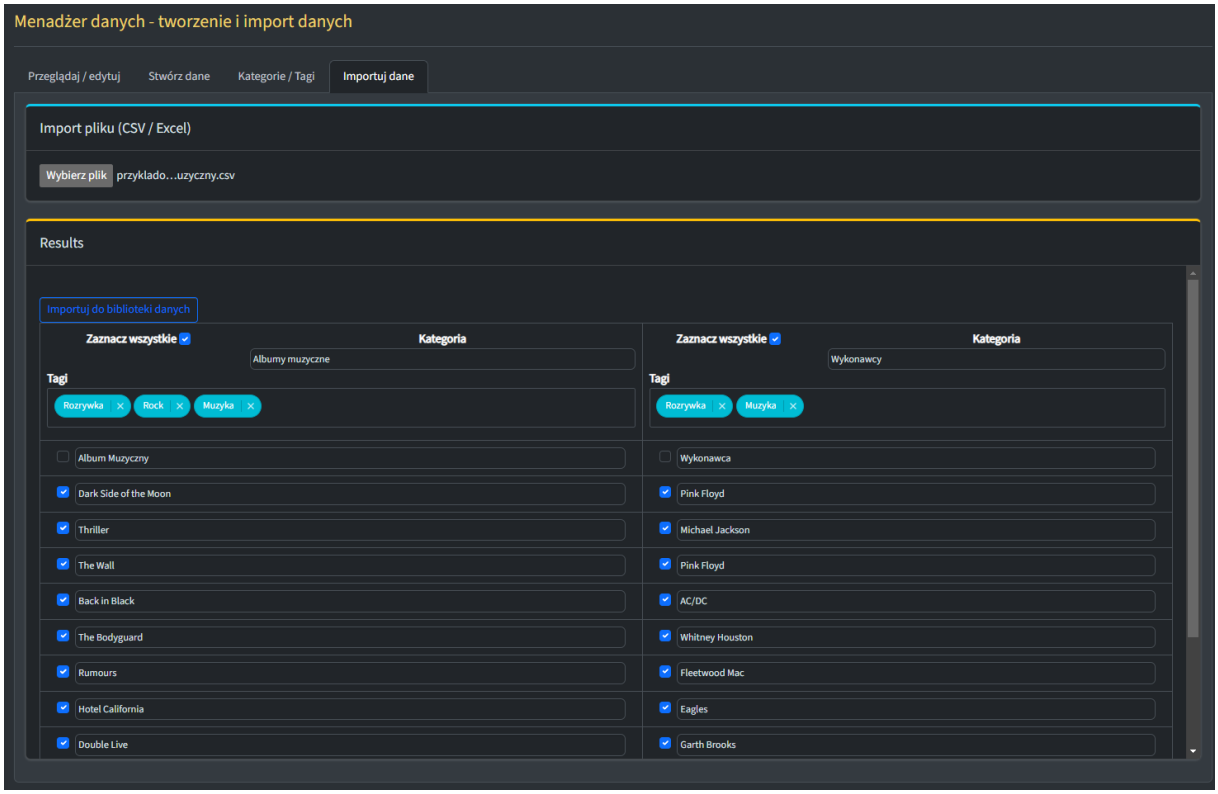
Podczas wypełnienia formularza z nową kategorią, możemy oznaczyć rodzica istniejącego lub wpisać tę kategorie jak rodzica (pole *Rodzic* jako puste), następnie dodać nowe podkategorie. Kategorie stworzone przez użytkownika są widoczne w liście w formie hiperłączy z możliwością interakcji, w bazie generatora mają ustawioną wartość pola *is_default = false*. Na Rysunek 16 pokazano widok zakładki Kategorie / Tagi. Widoczne jest kilka kategorii stworzonych przez użytkownika.



Rysunek 16 – Menadżer danych – Kategorie / Tagi, źródło: Opracowanie własne

5.7.5. Menadżer danych – Importuj dane

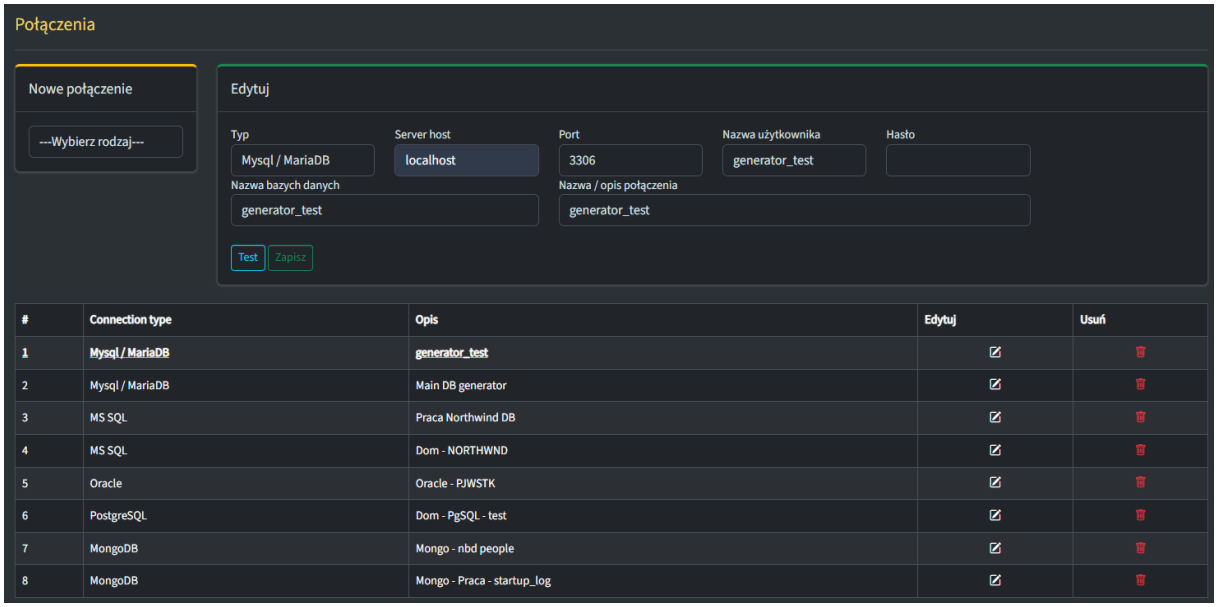
W tym interfejsie generator umożliwia import do biblioteki danych z dwóch typów plików: CSV oraz Microsoft Excel. W przypadku plików CSV, system automatycznie wykrywa, jakim znakiem rozdzielone są kolumny. Dla obu rodzajów plików dostępny jest podgląd importowanych danych, umożliwiający odznaczenie niepotrzebnych wierszy oraz edycję wartości w każdym wierszu. Przed finalizacją importu, użytkownik powinien przypisać dane do odpowiednich kategorii oraz, w razie potrzeby, otagować je dla ułatwienia organizacji i wyszukiwania. Przycisk *Importuj do biblioteki danych* wykonuje proces importu do bazy generatora, następnie system informuje o ilości rekordów zaimportowanych z pominięciem duplikatów. Widok tego interfejsu został przedstawiony na Rysunek 17.



Rysunek 17 – Menadżer danych – Importuj dane, źródło: Opracowanie własne

5.7.6. Połączenia

Strona ta, przedstawiona na Rysunek 18, służy do konfiguracji połączeń z bazami danych do których ma nastąpić eksport danych tymczasowych. Użytkownik może wybrać typ połączenia (MySQL / MariaDB, MS SQL, PostgreSQL, Oracle DB, MongoDB) oraz wpisać odpowiednie parametry. Po zdefiniowaniu połączenia dostępna jest opcja wykonania próbnego połączenia, co pozwala na weryfikację poprawności podanych danych konfiguracyjnych. W przypadku powodzenia próby, użytkownik otrzymuje potwierdzenie, że połączenie zostało pomyślnie skonfigurowane. W przypadku błędu, wyświetlany jest komunikat z informacją, co może pomóc w diagnozowaniu i rozwiązywaniu problemów z połączeniem. Na stronie wyświetlana jest lista wszystkich skonfigurowanych połączeń, dając łatwy dostęp do ich edycji lub usunięcia.



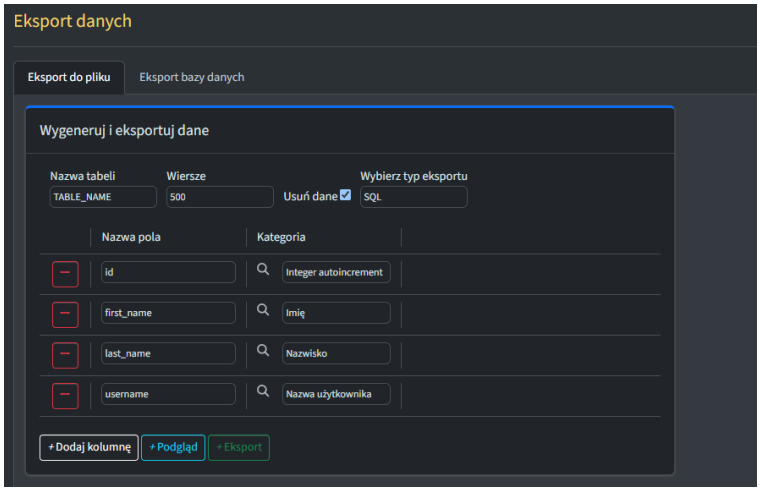
Rysunek 18 - Połączenia, źródło: Opracowanie własne

5.7.7. Eksport danych – Eksport do pliku

W tym widoku pokazanym na Rysunek 19 użytkownik ma możliwość wygenerowania danych tymczasowych oraz eksport do plików w formatach SQL, CSV, XML, JSON, XLSX. W panelu *Wygeneruj i eksportuj dane* znajduje się formularz z polami, w celu konfiguracji parametrów eksportu:

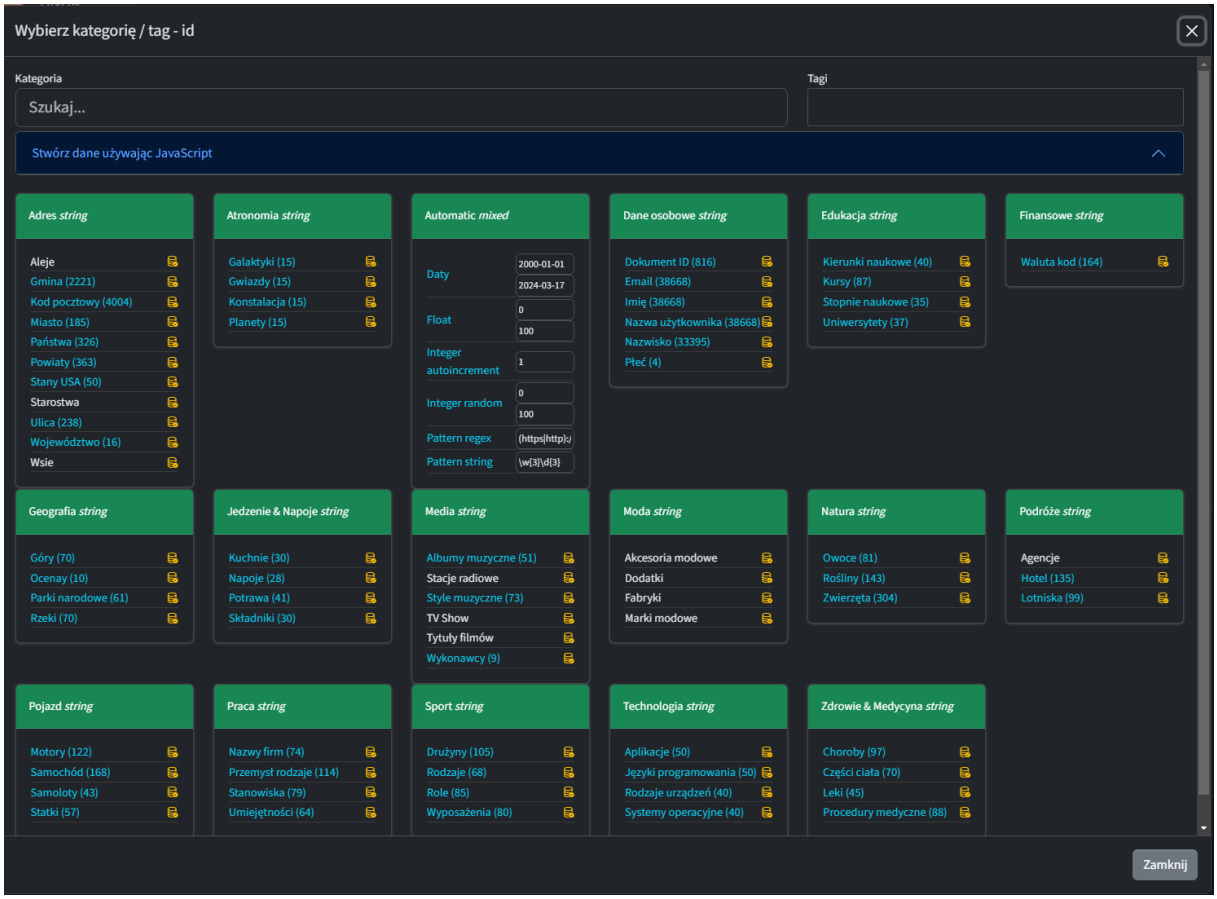
- Nazwa tabeli – dla pliku w typu SQL dodaje podaną nazwę tabeli do instrukcji *INSERT INTO*
- Wiersze – ilość danych do wygenerowania
- Usuń dane – zaznaczenie dla pliku typu SQL dodaje instrukcję DELETE FROM
- Wybierz typ eksportu – rodzaj eksportowanego pliku

Poniżej formularza znajduje się lista kolumn, które mają zostać dołączone do eksportowanego pliku. Użytkownicy mogą dowolnie dodawać nowe kolumny, usuwać istniejące lub zmieniać ich nazwy, co zapewnia dużą elastyczność w kształtowaniu struktury eksportowanych danych.

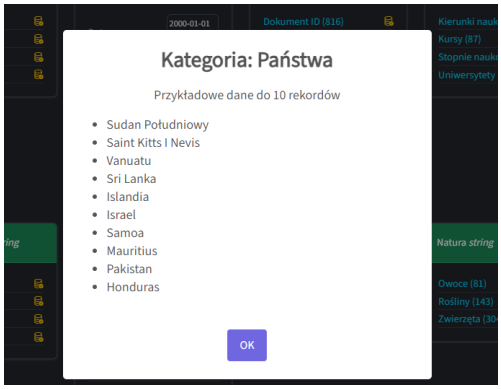


Rysunek 19 – Eksport danych – Eksport do pliku, źródło: Opracowanie własne

Ikona lupy znajdująca się obok nazwy kolumny eksportowanej prowadzi do zmiany przypisanej kategorii. Po naciśnięciu pojawi się okno wraz wyborem kategorii danych które zostało pokazane na Rysunek 20. Na górze tego okna znajduje się pole wyszukiwania, które umożliwia filtrowanie kategorii danych według ich nazwy lub tagu, ułatwiając znalezienie pożądanej kategorii. Poniżej pól wyszukiwania znajduje się opcja *„Stwórz dane używając JavaScript”*. Lista kategorii danych jest zaprezentowana poniżej, zgrupowana tematycznie, aby ułatwić nawigację. Dla każdej kategorii użytkownik może zobaczyć czy dane do niej są przypisane. W przypadku kategorii, które nie mają przypisanych danych, opcja przypisania kolumny do tej kategorii jest nieaktywna, co wskazuje na konieczność wyboru innej kategorii lub dodania danych do obecnej. Kliknięcie na żółtą ikonkę bazy danych, znajdującą się obok nazwy kategorii, wyzwoli wyświetlenie 10 przykładowych rekordów danych tymczasowych przypisanych do tej kategorii, przykład widoczny na Rysunek 21.

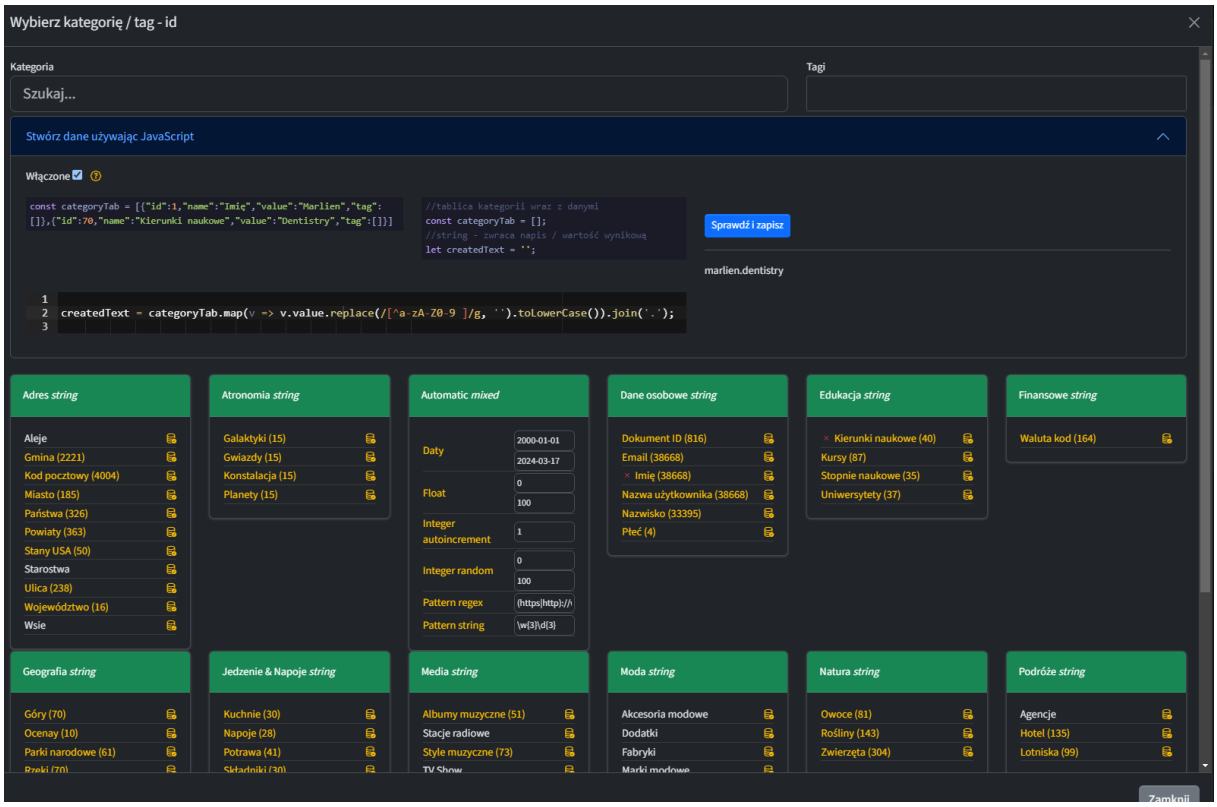


Rysunek 20 – Eksport danych – okno wyboru kategorii, źródło: Opracowanie własne



Rysunek 21 – Eksport danych – okno przykładowych danych kategorii, źródło: Opracowanie własne

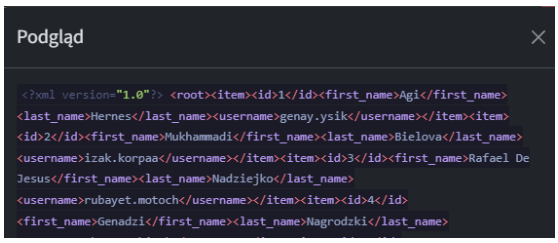
Aktywacja opcji *„Stwórz dane używając JavaScript”* uruchamia panel, w którym użytkownicy mogą zastosować język JavaScript wbudowanym edytorze do modyfikacji, generowania skomplikowanych wartości danych tymczasowych lub nawet tworzenia nowych wartości z niczego. Po włączeniu tej funkcji, poprzez zaznaczenie checkboxa *„Włączone”,* wszystkie hiperłącza nazw kategorii przybierają żółty kolor. Jest to informacja dla użytkownika, że obecnie kliknięcie w nazwę kategorii nie spowoduje jej bezpośredniego przypisania do kolumny eksportu, ale zamiast tego doda ją do tablicy w JavaScript. To zmienia sposób interakcji z listą kategorii, umożliwiając bardziej złożone operacje na danych. Naciśniecie przycisku *Sprawdź i zapisz* spowoduje sprawdzenie poprawności skryptu oraz przypisanie go do kolumny. Na Rysunek 22 pokazano widok użycia opisanej wyżej opcji, więcej na temat na temat generowania danych za pomocą JavasScript zostało opisane w podrozdziale 5.5.



Rysunek 22 – Eksport danych – tworzenie danych tymczasowych za pomocą JavaScript, źródło: Opracowanie własne

Po wybraniu kategorii danych i zamknięcia okna wyboru użytkownik może wygenerować podgląd według rodzaju pliku eksportu jaki został wybrany. Przykładowe ekrany wynikowe tej opcji zostały przedstawione na Rysunek 23 oraz Rysunek 24 odpowiednio dla eksportu SQL oraz XML.

Rysunek 23 – Eksport danych – podgląd SQL, źródło: Opracowanie własne



Rysunek 24 – Eksport danych – podgląd XML, źródło: Opracowanie własne

Naciśniecie przycisku *Eksport* generuje plik wynikowy.

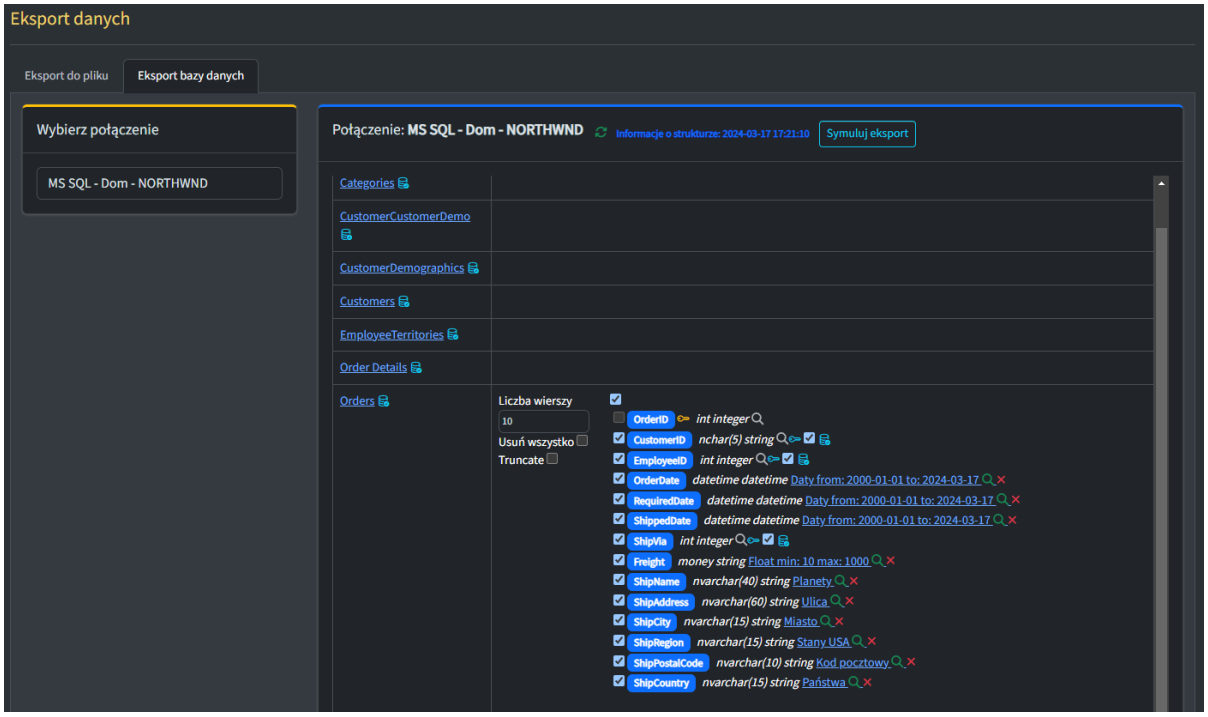
5.7.8. Eksport danych – Eksport do bazy danych

W tym module użytkownik ma możliwość bezpośredniego eksportu wygenerowanych danych tymczasowych do wybranej bazy danych. Proces ten umożliwia łatwe i szybkie przekazywanie dużych ilości danych bezpośrednio do systemów bazodanowych, co jest szczególnie przydatne w testowaniu, rozwoju aplikacji czy analizie danych.

Po lewej stronie widoku znajduje się wybór wcześniej zdefiniowanych połączeń. Po wybraniu odpowiedniego połączenia pojawia się lista tabel (w przypadku bazy nierelacyjnej pojawi się jedna tabela o nazwie kolekcji). Obok nazwy każdej tabeli znajduje się niebieska ikona bazy danych, po naciśnięciu której system pokaże 10 pierwszych rekordów z wybranej tabeli. Naciśnięcie nazwy tabeli mapuje wszystkie kolumny w niej zawarte prezentując ich listę wraz z informacjami oraz opcje eksportu dla tabeli:

- Liczba wierszy – ilość rekordów które będą wyeksportowana;
- Usuń wszystko – dodanie dla baz relacyjnych instrukcji SQL *DELETE FROM;*
- Truncate – dodanie instrukcji SQL, odpowiedniej dla danego typu bazy relacyjnej usuwającej całą zawartość tabeli oraz instrukcje zwalniające sprawdzanie zależności dla kluczy obcych, co umożliwia szybkie czyszczenie tabeli bez naruszania integralności referencyjnej;

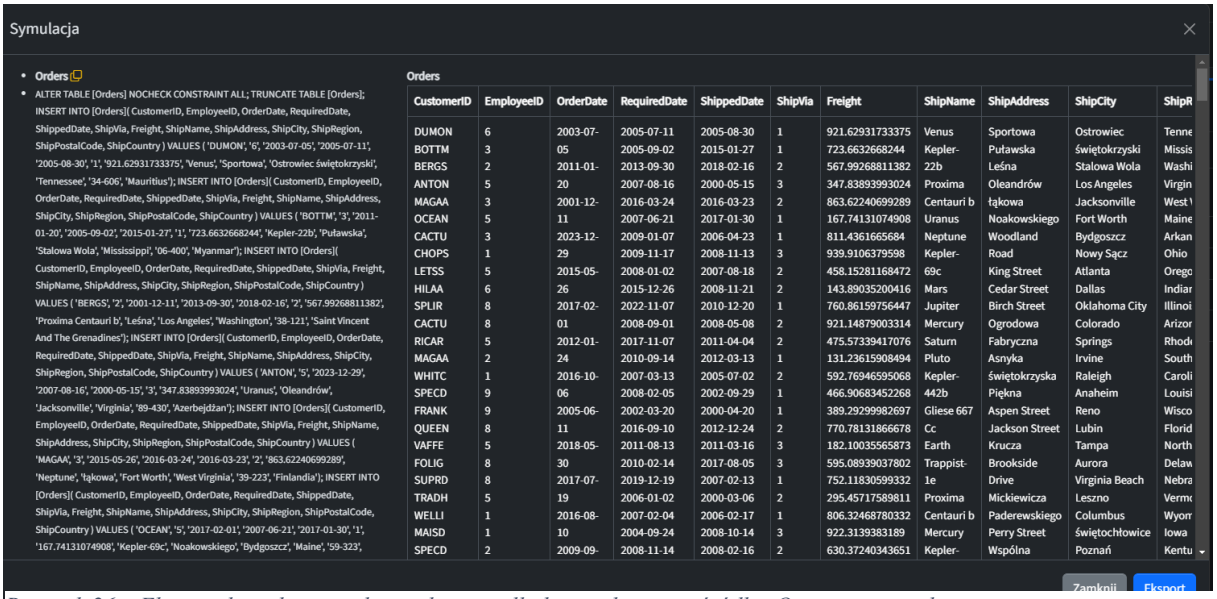
Przy wygenerowanej liście kolumn znajduje się checkbox którego zaznaczenie spowoduje iż system będzie brał pud uwagę eksport danych do tej kolumny. Następnie prezentowana jest nazwa kolumny. Jeśli obok nazwy kolumny znajduje się ikona żółtego klucza, oznacza to, że kolumna ta pełni rolę klucza głównego w tabeli. Klucz główny jest unikatowym identyfikatorem rekordów w tabeli, co ma kluczowe znaczenie dla integralności i organizacji danych. Następnie prezentowany jest typ danych kolumny, ikona lupy która pozwala na przypisanie kategorii danych analogicznie jak to miało miejsce w module eksportu do pliku. Następnie przy kolumnie może znaleźć się ikona niebieskiego klucza, które informuje iż kolumna posiada klucz obcy. W takim przypadku pojawia się również checkbox którego zaznaczenie powoduje iż dane do tej kolumny będą brane losowo z kolumny klucza obcego. Dodatkowo przy checkboxie znajduje niebieska ikonka bazy danych po naciśnięciu której system pokaże 10 pierwszych rekordów z tabeli właściwej kluczowi obcemu. Rysunek 25 przedstawia widok po wybraniu połączenia oraz kolumny wraz z już przypisanymi kategoriami danych.



Rysunek 25 – Eksport danych – Eksport do bazy danych, źródło: Opracowanie własne

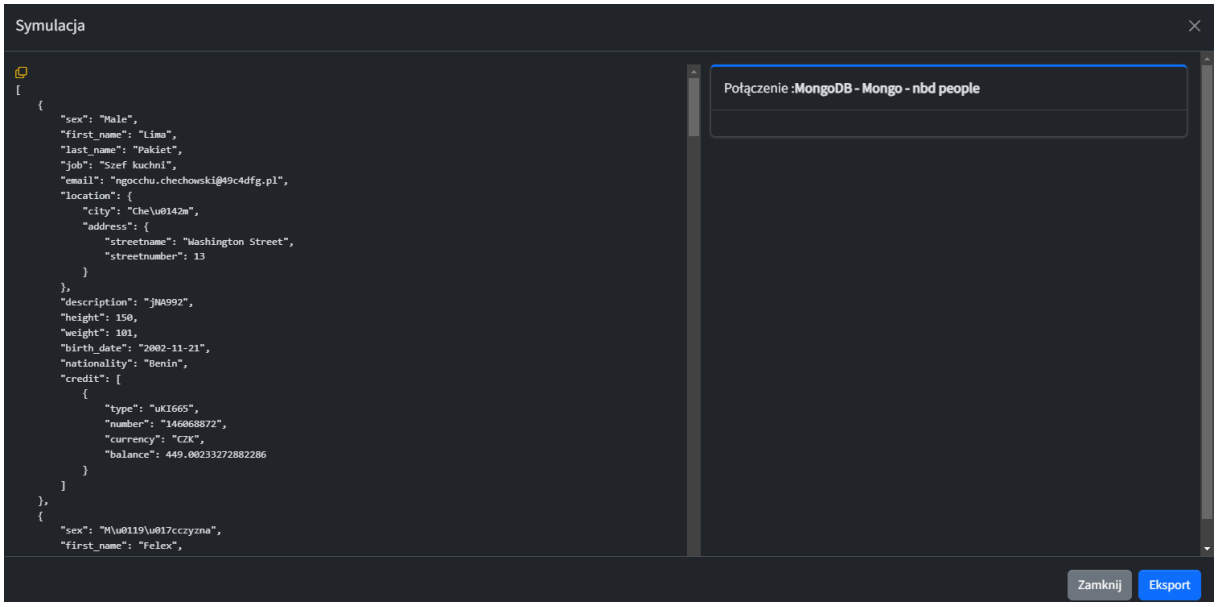
Przycisk *Symuluj Eksport* uruchamia okno podglądu, które przedstawia, jakie dane zostaną przekazane do bazy danych w ramach eksportu. Widoczna jest pełna instrukcję SQL w formie tekstowej oraz tabelaryczny podgląd danych, które mają zostać wyeksportowane.. Obok nazwy tabeli znajduje się żółta ikona – kliknięcie jej pozwala na szybkie skopiowanie przygotowanej instrukcji SQL do schowka. Daje to możliwość wykorzystania wygenerowanej instrukcji SQL w innym narzędziu lub zachowanie dla późniejszego użycia. W przypadku eksportu do bazy nierelacyjnej, dane prezentowane są w formacie JSON. Rysunek 26 oraz Rysunek 27 przedstawiają przykładowe ekrany z symulacją eksportu danych odpowiednio dla bazy relacyjnej i nierelacyjnej.

Naciśnięciu przycisku *Eksport* wywołuje procedurę eksportu do wybranej bazy danych.



CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	ShipVia	Freight	ShipName	ShipAddress	ShipCity	ShipR
DUMON	6	2003-07-	2005-07-11	2005-08-30	1	921.62931733375	Venus	Sportowa	Ostrowiec	Tenne
BOTTM	3	05	2005-09-02	2015-01-27	1	723.6632668244	Kepler-	Puławska	świętokrzyski	Missis
BERGS	2	2011-01-	2013-09-30	2018-02-16	2	567.99268811382	22b	Leśna	Stalowa Wola	Washi
ANTON	5	20	2007-08-16	2000-05-15	3	347.83893993024	Proxima	Oleandrów	Los Angeles	Virgin
MAGAA	3	2001-12-	2016-03-24	2016-03-23	2	863.62240699289	Centauri b	łąkowa	Jacksonville	West
OCEAN	5	11	2007-06-21	2017-01-30	1	167.74131074908	Uranus	Noakowskiego	Fort Worth	Maine
CACTU	3	2023-12-	2009-01-07	2006-04-23	1	811.4361665684	Neptune	Woodland	Bydgoszcz	Arkan
CHOPS	1	29	2009-11-17	2008-11-13	3	939.9106379598	Kepler-	Road	Nowy Sącz	Ohio
LETSS	5	2015-05-	2008-01-02	2007-08-18	2	458.15281168472	69c	King Street	Atlanta	Orego
HILAA	6	26	2015-12-26	2008-11-21	2	143.89035200416	Mars	Cedar Street	Dallas	Indiar
SPLIR	8	2017-02-	2022-11-07	2010-12-20	1	760.86159756447	Jupiter	Birch Street	Oklahoma City	Illinoi
CACTU	8	01	2008-09-01	2008-05-08	2	921.14879003314	Mercury	Ogrodowa	Colorado	Arizor
RICAR	5	2012-01-	2017-11-07	2011-04-04	2	475.57339417076	Saturn	Fabryczna	Springs	Rhode
MAGAA	2	24	2010-09-14	2012-03-13	1	131.23615908494	Pluto	Asnyka	Irvine	South
WHITC	1	2016-10-	2007-03-13	2005-07-02	2	592.76946595068	Kepler-	świętokrzyska	Raleigh	Caroli
SPECD	9	06	2008-02-05	2002-09-29	1	466.90683452268	442b	Piękna	Anaheim	Louisi
FRANK	9	2005-06-	2002-03-20	2000-04-20	1	389.29299982697	Gliese 667	Aspen Street	Reno	Wisco
QUEEN	8	11	2016-09-10	2012-12-24	2	770.78131866678	Cc	Jackson Street	Lubin	Florid
VAFFE	5	2018-05-	2011-08-13	2011-03-16	3	182.10035565873	Earth	Krucza	Tampa	North
FOLIG	8	30	2010-02-14	2017-08-05	3	595.08939037802	Trappist-	Brookside	Aurora	Delaw
SUPRD	8	2017-07-	2019-12-19	2007-02-13	1	752.11830599332	1e	Drive	Virginia Beach	Nebra
TRADH	5	19	2006-01-02	2000-03-06	2	295.45717589811	Proxima	Mickiewicza	Leszno	Verm
WELLI	1	2016-08-	2007-02-04	2006-02-17	1	806.32468780332	Centauri b	Paderewskiego	Columbus	Wyon
MAISD	1	10	2004-09-24	2008-10-14	3	922.3139383189	Mercury	Perry Street	świętochłowice	Iowa
SPECD	2	2009-09-	2008-11-14	2008-02-16	2	630.37240343651	Kepler-	Wspólna	Poznań	Kentu

Rysunek 26 – Eksport danych – symulacja eksportu dla bazy relacyjnej, źródło: Opracowanie własne



Rysunek 27 – Eksport danych – symulacja eksportu dla bazy nierelacyjnej, źródło: Opracowanie własne

6. Podsumowanie

W ramach niniejszej pracy przedstawiono kompleksowe rozwiązanie w postaci generatora danych tymczasowych, zaprojektowanego z myślą o wspieraniu programistycznych działów deweloperskich, architektów baz, w prowadzeniu procesów testowania obciążenia, analizy danych oraz rozwoju aplikacji. W tym rozdziale zostały opisane wady i zalety prototypu oraz możliwy dalszy rozwój funkcjonalności narzędzia.

6.1. Wady i zalety

Prototyp niweluje braki innych rozwiązań na rynku, jego skalowalna budowa powoduje iż może być używana w każdym środowisku. Dzięki implementacji szerokiego zakresu funkcjonalności, takich jak wsparcie dla różnorodnych formatów eksportu danych (SQL, CSV, XML, JSON, XLSX), elastyczne konfiguracje połączeń z bazami danych, dużej biblioteki danych początkowych oraz możliwość zaawansowanej manipulacji danymi za pomocą skryptów JavaScript, narzędzie to stanowi wszechstronne wsparcie dla deweloperów, analityków i testerów oprogramowania.

Podstawowe zalety osiągnięcia opisanego w tej pracy projektu obejmują:

- elastyczność w generowaniu danych – łatwość definiowania struktur danych zgodnie z potrzebami, korzystając z gotowych szablonów lub tworząc własne konfiguracje;
- zapewnienie wsparcia dla różnych systemów baz danych – integracja z popularnymi systemami bazodanowymi, takimi jak MySQL / MariaDB, PostgreSQL, MS SQL, Oracle DB, Mongo DB umożliwia bezpośredni eksport danych do docelowych środowisk;
- dostosowanie do potrzeb użytkownika – możliwość personalizacji danych poprzez skrypty JavaScript daje potężne narzędzie do tworzenia danych, które dokładnie odpowiadają wymaganiom testowym;
- intuicyjny interfejs użytkownika – zastosowanie nowoczesnych rozwiązań UI/UX, narzędzie jest łatwe w obsłudze, co przyczynia się do jego większej przystępności i efektywności użytkowania.

Z podstawowych wad prototypu należy wymienić:

- brak możliwości personalizacji dla użytkownika oraz w ramach projektu. Jeżeli do jednej instancji prototypu posiada dostęp wielu użytkowników pracują nad jednym zestawem danych;
- ograniczenia z łącznością do różnych baz danych są uzależnione od możliwości połączeń wspieranych przez PHP;
- brak wsparcia dla mniej popularnych systemów baz danych jak Firebird czy IBM DB2.

6.2. Perspektywy rozwoju prototypu

Perspektywy dalszego rozwoju generatora danych tymczasowych mogą obejmować:

- rozszerzenie zakresu wspieranych formatów danych;
- integrację z dodatkowymi mniej popularnymi systemami baza danych;
- dodanie oprogramowania w formie wtyczki do popularnych systemów zarzadzania bazami danych jak np. DataGrip firmy JetBrains;
- rozwój dodatkowych modułów, np.:
    - symulacji obciążeń;

    - automatycznego testowania zgodności i spójności danych;
- wprowadzenie funkcji personalizacji środowiska generatora, np.:
    - indywidualnie dla każdego użytkownika osobno;
    - współpracy zespołowej i zarządzania projektami;
    - udostępnia scenariuszy tworzenia danych pomiędzy projektami.

6.3. Wnioski i podsumowanie

Wnioski wyciągnięte z pracy nad projektem podkreślają znaczenie posiadania niezawodnych, elastycznych narzędzi do generowania danych tymczasowych, które mogą znacząco przyspieszyć programistyczne procesy deweloperskie i testowe, jednocześnie zapewniając wysoką jakość i spójność danych testowanych aplikacji. Odpowiednio dobrane i skonfigurowane dane tymczasowe umożliwiają budowę lepszych scenariuszy użytkowania aplikacji co może skutkować o wiele lepszym procesem tworzenia oprogramowania. Dzięki temu możliwe jest dokładniejsze przetestowanie różnorodnych przypadków użycia, obciążeń i zachowań aplikacji, co prowadzi do wykrycia i wyeliminowania potencjalnych błędów, nieoczekiwanych i nieplanowanych zdarzeń we wczesnych etapach cyklu rozwoju programowania. Skutkuje to oszczędnością czasu i kosztów, ponieważ problemy są identyfikowane i rozwiązywane zanim aplikacja trafi do środowiska produkcyjnego.

W zakończeniu, niniejszy projekt pokazuje, że skuteczne narzędzia do generowania danych tymczasowych są kluczowym elementem w ekosystemie rozwoju oprogramowania. Umożliwiają one oszczędność czasu, zwiększenie efektywności procesów, poprawę jakości produktu końcowego oraz takie korzyści jak zapewnienie zgodności z przepisami i wymogami prawnymi. Inwestycja w tego typu narzędzia jak część procesu produkcji i wytwarzani oprogramowania może przynieść znaczące korzyści dla firm i organizacji, pozwalając im na dostarczanie wysokiej jakości produktu w sposób bardziej wydajny i bezpieczny.

Bibliografia

[1] Google, „Google," 21 11 2023. [Online]. Available: https://www.google.com/search?q=random+data+generator. [Data uzyskania dostępu: 21 11 2023].

[2] functionize, „functionize.com," 25 01 2023. [Online]. Available: https://www.functionize.com/blog/online-test-data-generators. [Data uzyskania dostępu: 01 10 2023].

[3] Reddit, „Reddit," 05 06 2023. [Online]. Available: https://www.reddit.com/r/BusinessIntelligence/comments/xr4kcv/mock_data_generator_suggestions/. [Data uzyskania dostępu: 15 10 2023].

[4] Mockaroo, „Mockaroo," 01 01 2024. [Online]. Available: https://www.mockaroo.com/. [Data uzyskania dostępu: 02 02 2024].

[5] „Data Generator," 01 01 2024. [Online]. Available: https://generatedata.com/. [Data uzyskania dostępu: 25 03 2024].

[6] extendsclass, „extendsclass," 01 01 2024. [Online]. Available: https://extendsclass.com/random-data-generator.html. [Data uzyskania dostępu: 20 03 2024].

[7] Randat, „Randat," 01 01 2024. [Online]. Available: http://www.randat.com/. [Data uzyskania dostępu: 01 03 2024].

[8] PHP, „PHP," 01 12 2023. [Online]. Available: https://www.php.net/. [Data uzyskania dostępu: 01 03 2024].

[9] Wikipedia, „Programming languages used in most popular websites," 01 01 2024. [Online]. Available: https://en.wikipedia.org/wiki/Programming_languages_used_in_most_popular_websites. [Data uzyskania dostępu: 25 03 2024].

[10] W3Tech, „Usage statistics of server-side programming languages for websites," 01 01 2024. [Online]. Available: https://w3techs.com/technologies/overview/programming_language. [Data uzyskania dostępu: 25 03 2024].

[11] bacancytechnology, „PHP vs Java: The Ideal Programming Language in 2024," 01 01 2024. [Online]. Available: https://www.bacancytechnology.com/blog/php-vs-java#php-vs-java-comparison-table. [Data uzyskania dostępu: 01 02 2024].

[12] w3techs, „Market Position Report of Top 10 Server-side Programming Languages," 05 04 2024. [Online]. Available: https://w3techs.com/technologies/market/programming_language/10. [Data uzyskania dostępu: 05 04 2024].

[13] B. Peabody, „Server-side I/O Performance: Node vs. PHP vs. Java vs. Go," 01 01 2024. [Online]. Available: https://www.toptal.com/back-end/server-side-io-performance-node-php-java-go. [Data uzyskania dostępu: 01 03 2024].

[14] Mozilla, „MVC - MDN Web Doc," 01 01 2024. [Online]. Available: https://developer.mozilla.org/en-US/docs/Glossary/MVC. [Data uzyskania dostępu: 15 03 2024].

[15] Mozilla, „MVC - MDN Web Docs Glossary: Definitions of Web-related terms," 01 01 2024. [Online]. Available: https://developer.mozilla.org/en-US/docs/Glossary/MVC. [Data uzyskania dostępu: 05 04 2024].

[16] Symfony, „The Bundle System," 01 01 2024. [Online]. Available: https://symfony.com/doc/6.3/bundles.html. [Data uzyskania dostępu: 25 03 2024].

[17] hotwired, „HotWired," 01 01 2024. [Online]. Available: https://hotwired.dev/. [Data uzyskania dostępu: 25 03 2024].

[18] Mozilla, „MDN - Web APIs," 01 03 2024. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch. [Data uzyskania dostępu: 25 03 2024].

[19] Bootstrap, „Bootstrap," 01 01 2024. [Online]. Available: https://getbootstrap.com/. [Data uzyskania dostępu: 25 03 2024].

[20] S. K. Vinicius M. Grippa, MySQL. Jak zaprojektować i wdrożyć wydajną bazę danych. Wydanie II, Helion, 2022.

[21] MariaDB, „MariaDB versus MySQL: Compatibility," 01 01 2024. [Online]. Available: https://mariadb.com/kb/en/mariadb-vs-mysql-compatibility/. [Data uzyskania dostępu: 05 04 2024].

[22] Oracle, „Introduction to the Multitenant Architecture," 01 01 2024. [Online]. Available: https://docs.oracle.com/en/database/oracle/oracle-database/23/cncpt/multitenant-architecture.html#GUID-BBAD3448-8A5A-4397-BD3A-F05D945DB7F8. [Data uzyskania dostępu: 05 04 2024].

[23] M. Corporation, „Transact-SQL reference (Database Engine)," 13 07 2023. [Online]. Available: https://learn.microsoft.com/en-us/sql/t-sql/language-reference?view=sql-server-ver16. [Data uzyskania dostępu: 05 04 2024].

[24] Z. Dybikowski, PostgreSQL. Wydanie II, Helion, 2012.

[25] MongoDB, „Sharding - MongoDB Manual," 01 01 2024. [Online]. Available: https://www.mongodb.com/docs/manual/sharding/. [Data uzyskania dostępu: 05 04 2024].

[26] MariaDB, „Information Schema Tables - MariaDB Knowledge Base," 01 02 2024. [Online]. Available: https://mariadb.com/kb/en/information-schema-tables/. [Data uzyskania dostępu: 15 03 2024].

[27] M. Corporation, „System Information Schema Views (Transact-SQL) - SQL Server," Microsoft, 23 08 2023. [Online]. Available: https://learn.microsoft.com/en-us/sql/relational-databases/system-information-schema-views/system-information-schema-views-transact-sql?view=sql-server-ver16. [Data uzyskania dostępu: 15 03 2024].

[28] T. P. G. D. Group, „PostgreSQL Documentation," 08 02 2024. [Online]. Available: https://www.postgresql.org/docs/current/infoschema-columns.html. [Data uzyskania dostępu: 20 03 2024].

[29] Oracle, „System Tables and Views," Oracle, 01 01 2024. [Online]. Available: https://docs.oracle.com/database/timesten-18.1/TTSYS/systemtables.htm#TTSYS720. [Data uzyskania dostępu: 18 03 2024].

[30] S. Francia, MongoDB and PHP. Document-Oriented Data for Web Developers, O'Reilly Media, 2012.

Listingi

Listing 1 – Instalacja za pomocą menadżera pakietów npm, źródło: Opracowanie własne 20
Listing 2 – Instalacja za pomocą bezpośredniego osadzenie w kodzie HTML, źródło: Opracowanie własne 20
Listing 3 – Przykładowe użycie Turbo Drive – index.html, źródło: Opracowanie własne 20
Listing 4 - Przykładowe użycie Turbo Drive – data.html, źródło: Opracowanie własne 21
Listing 5 – Instalacja za pomocą menadżera pakietów npm, źródło: Opracowanie własne 22
Listing 6 – Kontroler hello_controller.js, źródło: Opracowanie własne 22
Listing 7 – Inicjalizacja Stimulus oraz kontrolera – application.js, źródło: Opracowanie własne 22
Listing 8 – Zmiana treści elementu z atrybutem data–controller – index.html, źródło: Opracowanie własne..... 22
Listing 9 – Encja tabeli data_name, źródło: Opracowanie własne 30
Listing 10 – Encja tabeli data_category, źródło: Opracowanie własne 31
Listing 11 – Przykład połączenia za pomocą PDO z system MySQL / MariaDB, źródło: Opracowanie własne . 35
Listing 12 – Metoda zwracająca zapytanie pobrania schematu bazy danych dla MySQL / MariaDB, źródło: Opracowanie własne 38
Listing 13 – Metoda zwracająca zapytanie pobrania schematu bazy danych dla Microsoft SQL Server, źródło: Opracowanie własne 39
Listing 14 – Metoda zwracająca zapytanie pobrania schematu bazy danych dla PostgreSQL , źródło: Opracowanie własne 40
Listing 15 – Metoda zwracająca zapytanie pobrania schematu bazy danych dla OracleDB, źródło: Opracowanie własne 41
Listing 16 – Metody przetwarzające rekurencyjnie dokument w MongoDB,, źródło: Opracowanie własne 42
Listing 17 - Tablica JavaScript z wybranymi kategoriami danych, źródło: Opracowanie własne 45
Listing 18 – Przykładowy kod JavaScript manipulujący danymi, źródło: Opracowanie własne 45
Listing 19 – Instalacja pakietów za pomocą Composer, źródło: Opracowanie własne 46
Listing 20 – Sprawdzenie wersji menadżera pakietów npm, źródło: Opracowanie własne 47
Listing 21 – Instalacja zależności oraz kompilacja plików JavaScript za pomocą npm, źródło: Opracowanie własne 47
Listing 22 – Przykładowy wpisy konfiguracyjny pliku .env do połączenia z bazą danych, źródło: Opracowanie własne 47
Listing 23 – Uruchomienie serwera www, źródło: Opracowanie własne 47

Spis rysunków

Rysunek 1 – Macaroo – źródło: [4] ... 7
Rysunek 2 – Data generator – źródło: [5] ... 9
Rysunek 3 – Random data generator – źródło: [6] ... 10
Rysunek 4 – Randat – Free random data generator – źródło: [7] ... 11
Rysunek 5 – Raport pozycji rynkowej dla języków programowania wykonywalnych po stronie serwera, źródło: [12] ... 17
Rysunek 6 – Schemat Modal View Controller (MVC) – źródło: [15] ... 18
Rysunek 7 – Stos bibliotek Hotwire, źródło: Opracowanie własne ... 19
Rysunek 8 – Przykładowe użycie Turbo Drive – index.html, źródło: Opracowanie własne ... 21
Rysunek 9 – Przykładowe użycie Turbo Drive – data.html, źródło: Opracowanie własne ... 21
Rysunek 10 – Schemat bazy danych uniwersalnego generatora danych biznesowych działającego w Chmurze, źródło: Opracowanie własne ... 32
Rysunek 11 – Diagram modeli baz danych , źródło: Opracowanie własne ... 36
Rysunek 12 – Strona główna, źródło: Opracowanie własne ... 49
Rysunek 13 – Menadżer danych – lista kategorii i tagów, źródło: Opracowanie własne ... 49
Rysunek 14 – Menadżer danych – lista danych z możliwością edycji, źródło: Opracowanie własne ... 50
Rysunek 15 – Menadżer danych tworzenie i import danych do bazy generatora, źródło: Opracowanie własne .. 51
Rysunek 16 – Menadżer danych – Kategorie / Tagi, źródło: Opracowanie własne ... 51
Rysunek 17 – Menadżer danych – Importuj dane, źródło: Opracowanie własne ... 52
Rysunek 18 - Połączenia, źródło: Opracowanie własne ... 53
Rysunek 19 – Eksport danych – Eksport do pliku, źródło: Opracowanie własne ... 53
Rysunek 20 – Eksport danych – okno wyboru kategorii, źródło: Opracowanie własne ... 54
Rysunek 21 – Eksport danych – okno przykładowych danych kategorii, źródło: Opracowanie własne ... 54
Rysunek 22 – Eksport danych – tworzenie danych tymczasowych za pomocą JavaScript, źródło: Opracowanie własne ... 55
Rysunek 23 – Eksport danych – podgląd SQL, źródło: Opracowanie własne ... 55
Rysunek 24 – Eksport danych – podgląd XML, źródło: Opracowanie własne ... 55
Rysunek 25 – Eksport danych – Eksport do bazy danych, źródło: Opracowanie własne ... 57
Rysunek 26 – Eksport danych – symulacja eksportu dla bazy relacyjnej, źródło: Opracowanie własne ... 57
Rysunek 27 – Eksport danych – symulacja eksportu dla bazy nierelacyjnej, źródło: Opracowanie własne ... 58

Spis tabel

Tabela 1 – Lista kolumn tabeli connection, źródło: Opracowanie własne 28
Tabela 2 – Lista kolumn tabeli connection_type, źródło: Opracowanie własne 28
Tabela 3 – Lista kolumn tabeli data_tag, źródło: Opracowanie własne 29
Tabela 4 – Lista kolumn tabeli data_type, źródło: Opracowanie własne 29
Tabela 5 – Lista kolumn tabeli data_tag_data, źródło: Opracowanie własne..... 29
Tabela 6 – Lista kolumn tabeli data_category, źródło: Opracowanie własne..... 29
Tabela 7 – Lista kolumn tabeli data_name, źródło: Opracowanie własne 29
Tabela 8 – Lista kolumn tabeli column_name_category, źródło: Opracowanie własne..... 30